SEGMENTING BANK CUSTOMERS USING SIMILARITIES IN CURRENT ACCOUNT DYNAMICS TO IMPROVE DAILY BANK BALANCE FORECASTING

by

Ajith Punnen

Submitted in partial fulfilment of the requirements for the degree
Master of Engineering (Computer Engineering)

in the

Department of Electrical, Electronic and Computer Engineering
Faculty of Engineering, Built Environment and Information Technology

UNIVERSITY OF PRETORIA

August 2018

SUMMARY

SEGMENTING BANK CUSTOMERS USING SIMILARITIES IN CURRENT ACCOUNT DYNAMICS TO IMPROVE DAILY BANK BALANCE FORECASTING

by

Ajith Punnen

Supervisor:	Prof. J.P. de Villiers
Co-supervisor:	Dr Conrad Beyers
Department:	Electrical, Electronic and Computer Engineering
University:	University of Pretoria
Degree:	Master of Engineering (Computer Engineering)
Keywords:	Time series forecasting, clustering, customer segmentation, bank balance forecasting

Liquidity risk is one of the key risks faced by banks in their daily operations. Following the recent financial crisis, more stringent measures have been put in place to ensure that banks adequately cater for sufficient liquidity and stable funding. In liquidity planning a difficult component is the modelling of indeterminate maturity products, which from a liabilities point of view includes transactional and savings accounts (demand deposit accounts). Banks utilise the balances in these products to also partly supply the funds necessary for loans and other forms of credit, which generate most of their profits. The purpose of this study was to find a way to accurately forecast the daily bank balance of a demand deposit account portfolio across the period of a year. This would help the banks to more efficiently handle liquidity planning and also generate more profit by utilising their funds more effectively. In accomplishing this the study also presented the hypothesis that using a hybrid model which combined segmentation with a popular forecasting method such as autoregressive integrated moving average (ARIMA) models would do better than a single

time series forecasting model. The purposes of the segmentation was to identify customers with similar current account dynamics e.g. salaried individuals in comparison to a small business owner.

Segmentation was facilitated by extracting features from the time series that identified patterns of salaried individuals in comparison to other account holders. These features were used by the k-means algorithm to form the segments. ARIMA models were then implemented for each of the segments and forecasts obtained per segment. These segment level forecasts were then aggregated to obtain the portfolio level forecasts. The results were then compared to building a single model to forecast the portfolio daily balance. Results from the study suggest that the hybrid model statistically performs significantly better than the single model over shorter forecast horizons. This study also attempted to find a way to score customers into one of the identified segments using information available on enrolment. However, results suggested that there is not enough discriminative power available in the information collected at enrolment but rather it is better to include information regarding a customer's first month's bank balance which significantly improved the classification accuracy.

LIST OF ABBREVIATIONS

AIC	Akaike's information criterion
AMI	Advanced metering infrastructure
ANN	Artificial neural networks
ARIMA	Autoregressive integrated moving average
ATM	Automatic teller machine
BCBS	Basel committee on bank supervision
BIC	Bayesian information criterion
CLV	Customer lifetime value
CNFS	Complex neural fuzzy system
CRM	Customer relationship management
DBSCAN	Density based spatial clustering on applications with noise
EM	Expectation-maximization
FBSA	Fuzzy c-mean based splitting algorithm
FCM	Fuzzy c-mean
GA	Genetic algorithm
GARCH	Generalised autoregressive conditional heteroscedasticity
GHSOM	Growing hierarchical self-organising map
GMDH	Group method of data handling
GP	Genetic programming
GRNN	General regression neural network
ICA	Independent component analysis
FITX	Taiwan index futures
KPLSR	Kernel partial least square regression
LDA	Linear discriminant analysis
MAPE	Mean Absolute Percentage Error
ML	Maximum likelihood
MLFF	Multi-layer feed forward neural network
OAS	Option-adjusted spread
PAM	Partitioning around medoids
PCA	Principal component analysis
POS	Point of sale

PSO	Particle swarm optimisation
PVE	Proportion of variance explained
RFM	Recency frequency and monetary
RLSE	Recursive least squares estimator
RMSE	Root mean square error
RSOM	Recurrent self-organising map
SMAPE	Symmetric mean absolute percentage error
SOM	Self-organising map
SVM	Support vector machine
SVR	Support vector regression
TAIEX	Taiwan stock exchange capitalization weighted stock index
TMSC	Taiwan semiconductor manufacturing
WNN	Wavelet neural network

TABLE OF CONTENTS

CHAPTER 1 INTRODUCTION

1.1 PROBLEM STATEMENT

1.1.1 Context of the problem

One of the major risks that banks have to deal with in their daily operation is liquidity risk. Liquidity risk is the current and potential risk that a bank might be unable to meet payments or clear obligations in a timely and cost-effective manner. It measures the bank's ability to meet net cumulative cash outflows within a certain time period (Matz & Neu, 2007). Failing to adequately cater for liquidity risk can have dire consequences for banks, this was apparent during the 2008 financial crisis where an inability to meet near-time commitments led to the demise of large investment banks such as Lehman Brothers and Bear Stearns.

Following the 2008 financial crisis, regulators have placed stricter requirements on banks in terms of liquidity and maintaining stable funding. Furthermore, the Basel Committee on Bank Supervision (BCBS), which is a committee that is mandated to help shape banking regulatory frameworks across the globe, has set out minimum liquidity standards in their recent publication (Basel Accord III) (Musakwa, 2013).

In order to ensure that banks meet the aforementioned liquidity standards and to also obtain a quantitative assessment of liquidity risk, banks perform liquidity gap analysis which involves cash flow modelling. This type of analysis is done based on various scenarios e.g. normal operating conditions, general market disruption, national macroeconomic disruption, a downgrade etc.

A difficult component in this analysis is the modelling of indeterminate maturity products (these are the banks' assets and liabilities that do not have a defined contractual period i.e. their maturity is unknown). From an asset side, these types of products include overdrafts and credit cards while on the liabilities side it is savings and current/transactional or cheque accounts. Current and savings accounts are also commonly referred to as demand deposits (Dzmuranova & Teply, 2015).

This study focuses on the liabilities side of indeterminate maturity products i.e. demand deposits. The difficulty in modelling these deposits is that customers can deposit or withdraw money from these accounts without prior notice. This task becomes challenging when taking into account the fact that the ways in which individuals in a certain deposit account portfolio utilise and obtain their funds can be very different. Some individuals might be salaried and use the bank's deposit account as their main account for transacting; others might run a small business through their deposit account and thus will receive funds into their accounts at irregular intervals etc. Their behaviour might also be influenced by factors such as deposit rates and other economic factors.

Furthermore, when the forecasts are on an aggregated level, which is over all the accounts, even the smallest error will be quite significant as the amounts are very large e.g. billions compared to tens of millions. This could result in significant profit wasted, as a result of not provisioning enough of the excess funds available for loans or it could result in an over utilisation of the funds which could in turn adversely affect the customers of the bank as well as the liquidity of the bank.

1.1.2 Research gap

In the banking sector there is no generally accepted framework in place to handle the modelling of demand deposits. Most banks use their own custom method for handling these products. A common strategy that is employed is to split the total deposit volume in these accounts into a stable part or core balance and the remainder into a volatile part. In

literature, this method is referred to as non-maturation theory. This method makes assumptions regarding the maturity of these deposits. The stable part is assigned a longer maturity when conducting the liquidity gap analysis while the volatile part is assigned shorter maturity horizons e.g. one month (von Feilitzen, 2011).

The aforementioned strategy is a viable solution since although these demand deposits are difficult to model, they are a very stable source of funding. This is because of the large number of customers that have demand deposit accounts and the fact that each customer only contributes a very small portion of the total volume. Therefore, most of the volume will remain with the bank as not all customers will behave in a similar manner and withdraw large amounts at the same time. However, profit margins can still be increased by using more advanced methods (von Feilitzen, 2011).

Most studies in literature that are associated with modelling demand deposits look at a technique called the replicating portfolio approach. This approach looks at assigning maturities and re-pricing dates to the demand deposits by creating a portfolio of fixed income instruments, with known maturities, that mirrors the cash flows of the demand deposit accounts (von Feilitzen, 2011). This approach will be further detailed in the literature study.

The replicating portfolio approach described above is a deterministic approach. Other techniques employed in literature include stochastic modelling approaches such as dynamic replicating portfolios and option-adjusted spread (OAS) models. The standard replicating approach relies purely on historical data to determine the optimal portfolio whereas the dynamic replicating portfolio approach makes use of simulations of future interest scenarios. In the case of OAS models, a stochastic process is used to account for the change in interest rates and it is used along with complex relationships between volumes, market rates and deposit rates to obtain the expected future cash flows (von Feilitzen, 2011).

Recently a Masters thesis has been published (Ahmadi-Djam & Belfrage Nordstrom, 2017) which looks at assessing the viability of using time series models to forecast deposit volumes in non-maturing liabilities i.e. demand deposits. However, all the aforementioned studies do not look at accounting for customer behaviour when modelling the volumes of the demand deposits. The abovementioned approaches have looked at forecasting or modelling demand deposits at an aggregated level. The viability of grouping customers that transact in a similar way in order to forecast or model their deposit volumes and combining these volumes to obtain an aggregated deposit volume, has not been explored in literature. Such an approach, which has been applied in other disciplines to improve forecasts and projections, could improve cash flow volume projections from demand deposit accounts.

1.2 RESEARCH OBJECTIVE AND QUESTIONS

The purpose of this study is to segment customers in a specific demand deposit account of a bank based on the similarity of their balance patterns in order to improve daily bank balance forecasts of this specific demand deposit account. By segmenting the customers and forecasting within the identified segments, the overall error in the daily portfolio level forecast should decrease in comparison to the results of a single forecasting model. Furthermore by identifying the characteristics of these segments, new customers can be assigned to one of these segments at enrolment thus helping to ascertain their balance patterns sooner.

The following research questions will be addressed in this study:

- What features of a bank balance time series can be used to identify customers with similar balance movement patterns?
- How should customers be segmented based on the similarity of their balances using clustering?
- Can segmentation help improve the results of a daily bank balance forecast for a demand deposit portfolio?

- What characteristics can be used to classify (based on a score) a future customer into one of the segments that have already been identified?

1.3 APPROACH

The proposed approach is to use the variance and mean of normalized balances over various time periods e.g. various dates in a month, weekly, quarterly etc. and clustering them together to form groups of customers with similar balance patterns. To illustrate this concept, think of a normal salaried person, who may have a high balance at the end of the month when the salary comes in but their balance will depleted at the beginning of the next month when various debit orders are completed. Finally somewhere towards the end of the month, before the period in which their salary comes in, their balances will be at the lowest point. These types of patterns can be identified by looking at the mean balances over certain periods of time and the variation in these balances. After identifying these segments or clusters, balances within these segments can be forecasted using an autoregressive integrated moving average (ARIMA) model. These results can be compared to the case where an ARIMA model was built for the entire group of customers i.e. without segmenting.

Lastly, for the different segments, significant variables/features can be identified from a secondary dataset, with customer information variables captured at enrolment, that best explain these segments. These features can be used in future to assign customers into their respective segments using a classifier at enrolment, which provides a means to forecast their balance using parameters associated with their segment. This proposed approach will be applied to a subset of the customers in a specific demand deposit account at a bank.

1.4 RESEARCH GOALS

The goals of this study are to develop a model to forecast daily bank balances for a set of customers in a demand deposit account over the course of a year. The next step is to see if these forecasts can be improved by segmenting the customers according to similarities in

their balances. Finally it would be determined if customers can be assigned to these identified segments using information available about them at enrolment.

1.5 RESEARCH CONTRIBUTION

Although various studies have explored the concepts of customer segmentation in the banking sector, none to the author's knowledge have explored customer segmentation in the banking sector for the purpose of improving daily bank balance forecasts. Furthermore, none of these studies have attempted to segment customers based on the similarities of their balances. Lastly, clustering or unsupervised learning has been used to improve forecasting results for various applications from stock prices to electricity load demand forecasting. However, the author has not come across any studies that have used it for daily bank balance forecasting.

Furthermore, previous studies that model demand deposits do so on an aggregated level. Although error margins can be quite significant when modelling demand deposit volumes at an aggregated level since the volumes are of large amounts e.g. billions instead of millions of rands. This study hopes to improve upon forecasts or projections at an aggregated level by modelling demand deposit volumes on a customer level and trying to capture the inherent variability at this level. Although this study is being conducted over a subset of customers, findings could be applied to an entire demand deposit account portfolio or to several different types of deposit account portfolios. Thus, helping banks to better ascertain the exact amount of funds they have available to provision for loans and other forms of credit etc.

In terms of use cases, apart from the bank that will use the outputs of this research, other users can include the South African Reserve Bank (SARB). In a liquidity crisis scenario where banks cannot secure additional funding through the interbank market, the lender of last resort will be the SARB. Thus the outputs of this study might be useful for the SARB to see if this approach can help banks to more effectively plan around liquidity risk etc.

1.6 RESEARCH OUTPUTS

A journal article was submitted to the Journal of the Operational Research Society (Taylor & Francis Online, 2018) in February 2019.

1.7 OVERVIEW OF STUDY

In Chapter 2 a literature study was performed to find out previous methodologies that have been followed with similar studies. Chapter 3 focuses on the theoretical background of the techniques used in this study. Chapter 4 details the methodology followed as well as providing a summary of the data used for the study as well as discussing the systems used for the study. Chapter 5 presents the results of the study as well as a discussion thereof. Finally, Chapter 6 provides some concluding remarks as well as providing recommendations for future studies.

CHAPTER 2 LITERATURE STUDY

2.1 CHAPTER OBJECTIVES

This chapter contains some of the strategies that have been applied in literature to tackle modelling of demand deposits and customer segmentation in Banking. It also contains examples of using clustering as a precursor to time series forecasting and the results thereof. The objective of the chapter is to explore previous studies in this field as well as to find strategies that might lend itself to this study.

2.2 MODELLING OF DEMAND DEPOSITS

Future cash flow projections of demand deposits are required to adequately cater for liquidity risk as mentioned in Section 1.1. A very popular approach to tackle this problem was mentioned in Section 1.2. However, methods described in literature have looked at handling this problem using more advanced techniques. Modelling of demand deposits have been mainly approached in literature in terms of the evolution of cash flows and secondly from a valuation perspective (Musakwa, 2013). The replicating portfolio and stochastic models, such as option-adjusted spread models, fall under the valuation category. Recently studies (Ahmadi-Djam & Belfrage Nordstrom, 2017; Wang, et al., 2015; Bielak, et al., 2015) have looked at using time series models and machine learning models to deal with this problem.

2.2.1 Models for the evolution of demand deposit cash flows

(Neu, 2007) and (Vento & La Ganga, 2009) model the future demand deposits account balance using a log-linear time series regression where the dependant variable is the log-transformed demand deposits balance. The intercept in this model is the current demand deposits balance, time accounts for the trend and there is also a normally distributed error term (Musakwa, 2013).

(Bardenhewer, 2007) follows an approach whereby the demand deposit volume is split into a deterministic trend component and a random component, which is interpreted as the deviation from the trend. The deterministic trend component is modelled using a time series regression where the dependant variables are the current demand deposit volume, time and deviations of customer interest rate on the demand deposit from its historical average. The deterministic trend component of the total volume is assumed to be invested using a replicating portfolio. The process followed in constructing a replicating portfolio has been briefly described in Section 1.2 and will be detailed in the next subsection. The remainder, i.e. the deviation from the projected volume from the trend is considered to be the demand deposit portfolio's cash-flow realised within one month (Musakwa, 2013).

2.2.2 Models based on valuation

As mentioned in the beginning of this section, studies that model demand deposits based on a valuation perspective make use of the replicating portfolio approach or a stochastic model. In some cases, the studies compare both a replicating portfolio approach and a stochastic model.

2.2.2.1 Replicating portfolio valuation methods

A replicating portfolio is made up of standard traded financial instruments such as fixed income securities, money market instruments and standard swaps. After identifying the set of financial instruments to be used to construct the replicating portfolio, the next step is to determine the portfolio weights for these instruments. The weights are usually obtained

using an optimisation procedure with the optimisation criterion being based on finding the optimal difference between the interest rate obtained on the replicating portfolio and the deposit rate on the demand deposit. This difference is also referred to as the spread. Different approaches have been followed in terms of finding the optimal spread. (Frauendorfer & Schurle, 2007) obtained the portfolio weights by minimising the expected downside deviation of the spread. Meanwhile (Maes & Timmermans, 2005) used the standard deviation of the spread. There are a few constraints that are considered in the optimisation problem used to solve for the portfolio weights. These include the fact that the weights of the individual financial instruments that make up the replicating portfolio must sum up to one and that these weights cannot be negative i.e. no short positions are allowed (Musakwa, 2013).

2.2.2.2 Stochastic Models

The modelling methodology followed in Section 2.2.1 is referred to as the static replication portfolio model. This means that the portfolio weights are computed once based on historical data and then used repeatedly. In the case of maturing investments, these are re-invested at the same maturity. The weights are kept constant. Unlike dynamic replicating portfolio models and OAS models, they don't account for future interest rate scenarios (von Feilitzen, 2011).

(Frauendorfer & Schurle, 2007) used a dynamic replicating portfolio approach and compared it with a static replication portfolio approach. (Maes & Timmermans, 2005) investigated the dynamics of Belgian saving deposit volumes and rates and compared static replication portfolio models, Monte Carlo valuation models and dynamic replication portfolio models (von Feilitzen, 2011).

2.2.3 Forecasting of demand deposits using time series and machine learning techniques

Forecasting models based on time series methods and machine learning techniques have recently been employed to tackle the problem of modelling demand deposit volumes. In a

study that is related to the topic, (Bielak, et al., 2015) investigated the use of statistical and machine learning methods to determine the optimal forecasting model to estimate the amount of cash withdrawn daily by customers of a Polish bank. It was suggested that calendar effects are important features that needed to be added to these models, since most economic time series are directly or indirectly linked to a daily activity which is recorded either daily, weekly, monthly or quarterly. The authors utilised time series models such as ARIMA and ARIMAX (ARIMA with additional explanatory variables), and compared it to an Artifical Neural Network (ANN) model. The conclusion from the study was that the variables associated with calendar effects improved the forecasting accuracy when using both the time series models and the ANN model.

(Cui, et al., 2014) compared the effectiveness of two time series prediction methods on forecasting bank cash flow, the two methods in question being the moving average and exponential smoothing methods. The authors concluded that the best method was an exponential smoothing method of order two. (Wang, et al., 2015) used a method that combines a neural network based on back propagation and grey prediction to improve the time series prediction of bank cash flow. This study focused more on the novel algorithm than the methodology behind modelling the bank cash flow.

(Ahmadi-Djam & Belfrage Nordstrom, 2017) recently published a Masters thesis that compares the effectiveness of various time series models, including Holt-Winters, Stochastic Factor, ARIMA and ARIMAX models, for forecasting deposit volumes. The study also included stock market volatility, market rate and deposit interest rate or deposit rate as explanatory variables. The stock market volatility was simulated using Monte Carlo simulations and market rate movements were simulated using a Vasicek model. The Holt-Winters and ARIMA models could not incorporate these explanatory variables, however the stochastic factor and ARIMAX models were able to do so. Forecasts were done for 3 and 6 month periods to allow for a sufficiently long period for the purposes of liquidity planning. The authors concluded that the ARIMAX model with seasonality (calendar effects) provided the best out of time performance.

2.3 CUSTOMER SEGMENTATION IN BANKING

Customer relationship management (CRM) is an important part of the success of any company in today's highly competitive market. It allows companies to improve customer retention, increase customer profitability, and generate value for the customer by tailoring products and services to cater for their needs. It also helps to reduce the costs of overall operation by stream lining processes *etc*. Customer segmentation is a pivotal part of CRM (Namvar, et al., 2010). An example of this would be a customer segmentation strategy that segments customers in terms of value. After identifying customers that are highly profitable the company could create strategies that can help to retain and attract these type of customers e.g. by providing incentive offers that are tailored to their individual needs or purchasing behaviour. (Hsieh, 2004). By the same token, the company can reduce the resources assigned to unprofitable customers who create more losses than profits (Namvar, et al., 2010).

Most customer segmentation studies in literature segment customers using a concept called customer lifetime value (CLV). According to (Kim, et al., 2006), CLV is defined as "the sum of the revenues gained from a company's customers over the lifetime of transactions after the deduction of the total cost of attracting, selling, and servicing customers, taking into account the time value of money". CLV can be decomposed into three components, namely current value, potential value and customer loyalty. Studies that utilise CLV to segment customers follow one of three approaches. They either segment customers using only CLV values, by using the components of CLV or by taking into account both CLV and other information such as socio-demographic information or transaction history etc. (Kim, et al., 2006). In the context of banking related studies, the last approach is widely followed.

A common practice used along with CLV in a customer segmentation study is recency, frequency and monetary (RFM) analysis (Namvar, et al., 2010) which helps to establish how recently a customer transacted or purchased, how often they transact or purchase and

the monetary value of their transactions or purchases. The exact definition of the R, F and M values do however change based on the nature of the study.

(Namvar, et al., 2010) used RFM, demographic and CLV data to segment customers from an Iranian bank. These researchers used k-means clustering to segment customers initially based on their RFM values, afterwards demographic variables were used to partition the already identified segments into more segments. CLV, which was obtained using a neural network trained on customer profiles and transactions, was then used to compare each of the identified segments based on customer value. Lastly, the profile of each segment was obtained by examining the characteristics of the segments based on all of the features that have already been mentioned. These researches claim that marketers can use these profiles to improve marketing strategies and tailor strategies to cater for each group (Namvar, et al., 2010).

(Khajvand & Tarokh, 2011) used RFM data to segment retail banking customers in order to estimate the customer future lifetime value. In the context of their study, recency referred to the time between the last transaction and first day of each season, frequency referred to the number of days between transactions in each season and the monetary term referred to the daily average balance in all of the customers' deposit accounts over the course of each season. Three different clustering algorithms were attempted by the authors of this study to build the segmentation model, namely the k-means algorithm, two-step algorithm and x-means algorithm. The best clustering algorithm and number of clusters to use were determined using the Dunn index. It was found that the k-means algorithm with four clusters produced the optimal Dunn index value. After obtaining the segments, a seasonal ARIMA model was used to predict the future CLV of each segment based on the CLV values of the past six seasons. The CLV values used to build the ARIMA model were obtained by computing the CLV score based on a weighted RFM model (Khajvand & Tarokh, 2011).

(Hsieh, 2004) used customer segmentation as part of a behaviour scoring model to analyse a bank's credit card customers in order to help manage them. Customers were segmented

based on their credit repayment ability and RFM attributes using an unsupervised learning algorithm known as self-organizing map (SOM). In the context of this study, the recency term referred to average difference in time between repayment and the day of a purchase, frequency referred to the average number of credit card purchases and monetary value referred to the yearly amount spent via the credit card. Segmenting the customers based on the above mentioned variables resulted in the identification of three major groups, namely revolver, transactional and convenience users. The transactional user pays off his/her account in full before the end of the interest free period while the revolver users does not pay off their balance in full each month. The convenience users on the other hand make large purchases using their credit cards and pays off their account over a period of months incurring interest on their balance. After identifying these segments, profiling was performed using their geographic and demographic characteristics using an Apriori association rule inducer. The authors claim that the insights from this study would help to make decisions on which groups should be encouraged to spend more and also help to manage debt recovery for those groups whose repayment ability is not good (Hsieh, 2004).

Other studies that looked at customer segmentation in the banking sector include (Xie, et al., 2014) and (Zakrzewska & Murlewski, 2005). (Xie, et al., 2014) looked at segmenting customers based on their consumption through the point of sale (POS) channel with the focus on identifying customers that would help increase fee income from card usage through POS machines. Meanwhile (Zakrzewska & Murlewski, 2005) was a comparative study that looked at the application of three clustering algorithms on data from the banking sector. The algorithms in question were the k-means algorithm, a two phase clustering algorithm and the Density Based Spatial Clustering on Applications with Noise (DBSCAN) algorithm. The dataset used for clustering consisted out of five variables, namely age, income, deposit, credit and a variable indicating profit or loss. The effectiveness of the clustering algorithms were determined based on the ability to handle high dimensionality, scalability and ability to detect outliers. The results of this study showed that the k-means algorithm was best suited for multidimensional and large datasets but was susceptible to outliers. The two-phase algorithm deals with noisy data quite well but struggled with high dimensionality and large sample sizes. Lastly the DBSCAN

algorithm created challenges in terms of finding optimal input parameters (Zakrzewska & Murlewski, 2005).

2.4 CLUSTERING TO IMPROVE TIME SERIES FORECASTING RESULTS

A time series is defined as a series of data points recorded sequentially in time. Time series forecasting is the process of predicting future values of a time series based on past values and sometimes other variables. One of the difficulties in time series forecasting is that in many domains the time series are non-stationary i.e. these series do not exhibit identical statistical properties at each point of time. This also implies that the relationship between the independent and dependent variables can undergo dynamic changes. This poses a challenge for most learning algorithms as these algorithms rely on a constant relationship between the independent and dependent variables. These algorithms rely on the presumption that the data being fitted is created by some form of a constant function (Hsu, et al., 2009).

A solution to the aforementioned problem that has been employed in literature is to hybridise several artificial techniques. This strategy works on the divide-and-conquer principle which takes a complex problem and breaks it up into several simple problems in order to help solve the original problem more easily. One of the common ways of implementing this strategy of hybrid models to the problem of time series forecasting is by employing a two-stage architecture. In this formulation, a clustering algorithm or an unsupervised learning algorithm is used to partition the original data into smaller regions where the data points share similar characteristics or distributions (Hsu, et al., 2009). After dividing the heterogeneous data into several homogeneous regions (Hsu, et al., 2009), each region or partition is modelled using a simple local model to help better forecast the non-stationary time series (Huang & Wu, 2010). This strategy helps in capturing the non-stationary attributes of the time series in question (Hsu, et al., 2009). This strategy has been applied to time series forecasting problems in various domains as can be seen from the remainder of this section.

2.4.1 The use of hybrid models to improve financial time series forecasting

One of the domains in which the hybrid model has been extensively examined is in the field of financial time series forecasting. Financial time series forecasting in this case alludes to stock price forecasting as well as forecasting the price of futures, bonds, exchange rates etc. Financial time series are usually non-stationary and are influenced by various factors such as economic conditions, political and environmental events, traders' expectations etc. (Hsu, et al., 2009). Furthermore, accuracy is paramount in this domain as accurately predicting stock prices, for example, can lead to financial gains if combined with a good trading strategy.

The most popular way to implement the two stage architecture for the hybrid model in this domain has been to combine an unsupervised learning algorithm known as the Self-Organizing Map (SOM) with support vector regression (SVR). SOM is an unsupervised learning algorithm that clusters objects having multi-dimensional attributes into a lower-dimensional space using competitive learning (Hsu, et al., 2009). SVR is a regression technique that is closely related to the Support Vector Machine (SVM) classifier in terms of theory and implementation. This regression technique has become popular over the last two decades due to its ability to generalise better in comparison to other artificial techniques such as the ANN (Hsu, et al., 2009).

In the context of the divide and conquer principle, SOM is used to partition the whole input space into several homogeneous regions. After forming these disjoint regions, different SVR models are constructed to model these different regions. In order to accurately capture the different characteristics of the partitioned regions, the most appropriate kernel function and optimal learning parameters that best fits the partitioned regions will used to build each SVR model. By doing this, it will ensure that each SVR model will be the most adequate one for a particular region. This is in contrast to a single SVR model which may not be able to efficiently learn each local input region but rather learns the entire input space globally (Tay & Cao, 2001).

(Tay & Cao, 2001) were the first to implement the aforementioned SOM-SVR two stage architecture to a financial time series problem. Their study compared the performance of the SOM-SVR hybrid model with that of a single SVR model on forecasting the Santa Fe exchange rate and the daily closing prices of five real future contracts. In order to create the partitioned regions, four lagged relative difference in percentage of price (RDP) values based on 5-data points and a transformed price obtained by subtracting a 15-day exponential moving average from the price were used. The output variable was RDP+5. As the number of partitions or clusters was not previously known a tree-structured architecture was utilised for the partitioning phase. This tree-structured architecture partitioned the input space repeatedly into two regions using SOM as long as the partition condition, which in this case was a predetermined limit to the number of training data points that could be in a partition, was not satisfied. The experiments from this study showed that this hybrid model managed to achieve both a higher prediction performance and faster convergence speeds when compared to a single SVR model (Tay & Cao, 2001).

While (Tay & Cao, 2001) tested the efficiency of the two stage architecture on the exchange rate, futures and bonds, (Hsu, et al., 2009) empirically tested the efficiency of this architecture in forecasting the closing prices of seven major stock market indices. In doing so, (Hsu, et al., 2009) used a growing hierarchical self-organizing map (GHSOM), which is a SOM technique that automatically grows the map size both in a hierarchical and horizontal way, instead of the tree-architecture proposed in (Tay & Cao, 2001) to obtain the optimal number of partitions. The same features and output variable described in (Tay & Cao, 2001) were used in (Hsu, et al., 2009). (Hsu, et al., 2009) also assessed the results of the two-stage hybrid model and the single SVR model in terms of directional symmetry, which gives a measure of the correctness of the predicted direction. Results showed that the two-stage architecture offered better predictive performance in comparison to the single SVR model (Hsu, et al., 2009).

(Huang & Tsai, 2009) applied the SOM-SVR hybrid model to a Taiwan index futures (FITX) dataset in order to predict the next day's price index. In doing so, they combined the SOM-SVR model with filter-based feature selection. Initially thirteen technical

indicators were considered as input variables, some of these included the relative strength index, moving average convergence and divergence, directional indicator up, psychological line etc. The feature selection method was used to identify the most important input attributes for the hybrid model. This would help to obtain higher accuracy and alleviate data complexity. The data set used in the study spanned a six year period, however for testing purposes the data set was split into five subsets. Each of the subsets contained a training period of five years while the test period comprised of two months. The prediction model was built for each of the five subsets and the average performance across all five subsets was used to evaluate the performance of the proposed approach. The results of the study suggested that the SOM-SVR model with feature selection outperforms the approach that uses just one SVR model in terms of average prediction accuracy and training time. It was also noted that the SOM-SVR model with feature selection is an improvement on the SOM-SVR model without feature selection in terms of prediction accuracy (Huang & Tsai, 2009).

Other studies in this domain have replaced SVR, as the predictive algorithm, in the two-stage architecture of the hybrid model with other machine learning algorithms. (Hsu, 2011) combined SOM with genetic programming (GP) to create a hybrid model to predict the next day's closing price for the finance and insurance sub-index of the Taiwan stock exchange capitalization weighted stock index (TAIEX). As with (Huang & Tsai, 2009), the historical stock trading data was first converted into the appropriate technical indicators after which SOM was used to produce the appropriate number of clusters. (Hsu, 2011) developed a clustering efficiency measure in order to determine the optimal number of clusters. After clustering the sample data, the closing price of the next day and the technical indicators were normalised on a scale of [-1,1]. Subsequently the entire dataset was split into 10 subsets and these subsets were partitioned into training, test and validation sets using a proportion of 4:1:1. The training and test sets in each case were used to obtain the optimal parameters for the GP models for each cluster. The accuracy of the hybrid model was assessed on the validation set, however unlike the previous studies the performance of the SOM-GP hybrid model was not compared to a single model (e.g. a

single GP model). (Hsu, 2011) however notes that the SOM-GP hybrid model can be used as a feasible tool for stock price prediction based on the results of the study.

(Huang & Wu, 2010) highlighted that one of the flaws of the hybrid model as mentioned up till now in this section is that the time related context between successive vectors is not accounted for. In order to address this issue, (Huang & Wu, 2010) suggested that the model needed a mechanism that can store the contextual information that is present between the successive input vectors. This aspect was included by replacing the SOM in the hybrid model with a Recurrent Self-Organizing Map (RSOM). A recurrent SOM explicitly includes recurrent connectivity into the neural output. This feature allows the recurrent SOM to explicitly capture temporal patterns in the original input. In this study RSOM was used to partition and store temporal context of the feature space, which was obtained by extracting features from the time series using wavelet analysis. Wavelet analysis was used because it was deemed to be an efficient way to capture the inherent time-scale features of non-stationary time series. After the partitioning, multiple kernel partial least square regression (KPLSR) models that best fit the partitioned regions were constructed for final forecasting. Two different data sets were used in this study, the first comprising of the major Asian stock indices and the second comprising of the G7 stock indices. All index data ranged from the period January 2004 to December 2005. The training of the KPLSR models was performed in a batch manner with the window of the training data set sliding with the current prediction day such that 300 days before the day of prediction was used as the training data set. The results of the proposed approach was compared with ANNs, SVMs and generalised autoregressive conditional heteroscedasticity (GARCH) models, with the proposed approach outperforming all the other models (Huang & Wu, 2010)

(Choudhury, et al., 2014) proposed a novel hybrid model which uses a two layer abstraction to cluster stock series data using SOM followed by k-means clustering of the SOM. In this case SOM serves as a dimensionality reduction tool to map the high dimensional data to a two dimensional space. After which k-means is used to cluster the results of the SOM for interpretation. The optimal number of clusters for the k-means algorithm was determined from various cluster validity indices. After this process, the

cluster having the best underlying stocks is selected for the regression function, which in this case was SVR. The SVR models used the time series of these stocks to predict future values. Following this a trading strategy is adopted based on the price and volatility (Choudhury, et al., 2014).

Other approaches to utilise hybrid models in this domain include the works of (Li, et al., 2013), who proposed a self-organizing complex neuro-fuzzy intelligent approach using complex fuzzy sets (CFSs) and a clustering method to address the problem of time series forecasting. A fuzzy system comprises of a set of fuzzy If-Then rules. In this approach the number of fuzzy rules for the complex neural fuzzy system (CNFS) is determined using a clustering algorithm called fuzzy c-mean (FCM) Based Splitting Algorithm (FBSA). After which a hybrid learning method is used to adapt the free parameters of the CNFS predictor, consisting of particle swarm optimisation (PSO) for the If-part and recursive least squares estimator (RLSE) for the Then-part parameters respectively. The proposed approach is applied to four time series forecasting problems, three of which are in the financial domain including the problems of forecasting the Taiwan Semiconductor Manufacturing (TMSC) stock price, the weekly exchange rate between the US dollar and Taiwan dollar and forecasting the daily IBM stock price. In the case of each time series problem, the proposed approach was compared with other approaches that have been attempted for the particular problem and therefore the methods that were compared cannot be detailed in this report for the sake of brevity. The proposed approach performed the best in all four experiments (Li, et al., 2013).

2.4.2 The use of hybrid models to improve forecasting results in other domains

Various other domains have utilised the hybrid model methodology for time series forecasting, these include product demand forecasting, electricity load forecasting and cash withdrawal forecasting. (Lu & Wang, 2010) applied a hybrid model, combining independent component analysis (ICA), GHSOM and SVR, for the purposes of product demand forecasting. Accurately being able to forecast the demand for a product allows a business to more effectively drive production, inventory, distribution, and buying plans

across their operations (Lu & Wang, 2010). This study used monthly sales data of 38 companies over a period of 96 months as its input data. This data was split in such a way that the first 68 data points were used for training while the rest was used for testing.

ICA was first used on the dataset to detect and remove noise; the output of this stage is something called a mixing matrix which describes the relationship between the independent components and the input data. GHSOM was then used to cluster this data to create many disjoint clusters based on the mixing matrix. Each of these clusters are then learned by a SVR model that best fits the cluster, these SVR models are built by finding the optimal parameters of the SVR models for each cluster. The final forecasting results for each company is determined by first determining the cluster to which it belongs to and then using the SVR model for the associated cluster. The proposed approach was compared to a single SVR model and a GHSOM-SVR hybrid model using different ratios of training and testing sample sizes at three different forecast horizons (1 month, 6-month and 12-month ahead forecasts). The results obtained suggest that the ICA-GHSOM-SVR model provides better forecasting accuracy than the single SVR and GHSOM-SVR models (Lu & Wang, 2010).

The hybrid model strategy of using clustering to improve forecasting results has also been applied for the purposes of electricity load forecasting by (Quilumba, et al., 2015). Load forecasting is a critical part of power systems planning and operations. Generally, load forecasting is conducted at a system level with high-voltage level data, little or no consideration is given to information at lower levels such as regional level, substation level or consumer/household level. Today, the advent of smart meters and the adoption of advanced metering infrastructure (AMI) have presented electric utilities with a large amount of energy usage information at household level. One of the potential applications of this smart meter data is to help enhance the utilities' ability to forecast electricity load demands. This would help the utilities better manage their power grid (Quilumba, et al., 2015).

(Quilumba, et al., 2015) used smart meter data and clustering to group customers by load consumption similarities in order to improve system level load forecasting. The authors decided to group customers instead of forecasting load at the household level because household level data is too volatile, not only does each household's daily load curve vary due to the fact that each household has different appliances and individuals with different usage patterns. Furthermore, load consumption can be erratic if looked at a customer level. Therefore, this exercise is not trivial and requires extensive knowledge of the external factors that affect a customer's consumption behaviour (Quilumba, et al., 2015).

(Quilumba, et al., 2015) used data obtained from two different electric utility companies, one from the United States and the other from Ireland. The US data set contained 21 months of 15 min load data and the Irish data set contained 17 months of half-hourly load data. In both cases, the first 12 months were used for fitting the model and the remainder was used for model evaluation. In order to group the customers, each day in the datasets was divided into five segments corresponding to main intraday consumption behaviour patterns. After which, an average consumption at each day of the week was obtained. Lastly, the load was normalised in the range of 0-1 to make sure that the customers were grouped according to who contributes to the total consumption at different times of the day. This generated the dataset utilised by the k-means algorithm, for the above mentioned two cases (Quilumba, et al., 2015).

The k-means algorithm was run on the dataset with number of clusters ranging from 1 to 12. Instead of making use of a clustering validity index to select the appropriate number of clusters, the optimal number of clusters was obtained by choosing the number of clusters that minimises the aggregated load forecasting error for the test set. The forecasting error generally decreases until the number of optimal clusters is reached. After which, the error usually increases as the number of clusters increases. The Mean Absolute Percentage Error (MAPE) was used as the error metric (Quilumba, et al., 2015).

After assigning each smart meter to a specific cluster, the load data in the group was summed up. The load for each group was then forecasted before being aggregated to

generate the system load forecast. The aggregated load forecasting accuracy was evaluated at this stage. (Quilumba, et al., 2015) used ANN models for forecasting as it is the technique that utilities usually use in practice. The forecasts were sub-hourly with different time horizons up to one day ahead. The results of their study showed that the proposed approach outperformed a single model for both datasets, with 3 clusters giving the best performance on the US utility data and 4 clusters performing the best on the Irish utility dataset (Quilumba, et al., 2015).

Lastly, (Venkatesh, et al., 2014) proposed a hybrid model combining clustering and ANNs for the purposes of automatic teller machine (ATM) cash demand forecasting. Cash demand forecasting for ATMs is important for banks as they need to stock up cash supplies at ATMs for a priority set period of time. This requires that cash be ordered well in advance. If these forecasts are inaccurate, banks can incur unnecessary costs e.g. if the forecasts are too high, an excessive amount of unused cash might be stored in the ATMs which will lead to costs for the bank. Meanwhile if the ATM runs out of cash, this can lead to a loss in profits and dissatisfied customers. In order to improve ATMs' cash demand forecasts, (Venkatesh, et al., 2014) proposed an approach that forecasts cash demand for groups of ATMs with similar day-of-the week cash demand patterns (Venkatesh, et al., 2014).

(Venkatesh, et al., 2014) used data from the NN5 time series competition for their study, this dataset contained daily cash withdrawal amounts over 2 years from 111 ATM centres across the UK. The first step in their proposed approach was to build a multiplicative time series model for each ATM centre in order to determine if there was any seasonality associated with the cash withdrawal amounts for each day of the week. The result of this was seven continuous seasonality parameters, each showing the effect of the particular day of the week e.g. Monday etc. on the withdrawal amount. These continuous seasonality parameters are then discretized. Following this, a comparison between the ATM centres' discretised withdrawal seasonality parameter sequence is obtained by calculating the Levenshtein distance using the Sequence Alignment Method (SAM). These distances were then provided to the Taylor-Butina algorithm for clustering. Finally, a model is fitted for

each ATM cluster to obtain forecasts. In this case four different types of ANNs, namely general regression neural network (GRNN), multi-layer feed forward neural network (MLFF), group method of data handling (GMDH) and wavelet neural network (WNN), were used to build the forecasting models. The results of this study showed that the proposed approach performed better than a single model fitted on the whole sample without clustering based on the symmetric mean absolute percentage error (SMAPE) (Venkatesh, et al., 2014).

2.5 CHAPTER SUMMARY

This chapter provided an overview of the various approaches that have been applied in terms of modelling of demand deposits, customer segmentation in the Banking sector and it also introduced the hybrid modelling approach to tackling time series forecasting problems. Modelling of demand deposits have been mainly tackled in literature using valuation methods such as the replicating portfolio approach and stochastic models. However, recent studies have attempted to tackle the problem from a forecasting perspective using time series models and machine learning techniques. From these studies it is interesting to note that both (Ahmadi-Djam & Belfrage Nordstrom, 2017) and (Bielak, et al., 2015) both found calendar effects to be an important aspect in their time series models. An improvement to standard time series models or machine learning models for forecasting has been the introduction of hybrid modelling approach using a two-stage architecture which combines clustering and a forecasting or regression technique. It has been utilized in studies across various domains as can be noted in Section 2.4. In the financial domain, the most common application of the strategy has been to apply it to the problem of stock price time series forecasting. A common strategy that has been followed has been to combine an unsupervised algorithm called SOM for clustering followed by the use of SVR for forecasting. This approach has found success in the financial time series forecasting domain. However, there are limitations to this approach in that both SOM and SVR are very computationally expensive and do not lend themselves to large datasets. The approach followed by (Quilumba, et al., 2015) is more aligned to the theme of this study. (Quilumba, et al., 2015) used segmentation at a lower level, in this case smart meters, to

improve forecasting at a higher level i.e. at the system load level in this case. A similar type of approach will be followed in this study.

.

CHAPTER 3 BACKGROUND THEORY

3.1 CHAPTER OBJECTIVE

This chapter describes the underlying theory surrounding the techniques utilised in this study. The chapter introduces the concepts of cluster analysis, time series forecasting and the details surrounding the classifiers used in the study. The techniques discussed in this section include the k-means clustering algorithm, ARIMA models, Linear Discriminant Analysis (LDA) and the random forest classifier.

3.2 CLUSTER ANALYSIS

Cluster analysis or clustering is an unsupervised learning task. Unsupervised learning pertains to a set of statistical methods designed to tackle problems where the observations do not have a target Y but instead just have a set of features $x_{i1}, x_{i2}, \cdots, x_{ip}$, where $i = 1, \cdots, n$ refers to the number of samples in the data set, and p refers to the number of features. Clustering partitions these observations in the data set into subgroups called clusters, based on the features. This results in a situation where each cluster contains observations that are similar to each other while being different to observations in other clusters (James, et al., 2013).

There are two main types of clustering, namely hard clustering and soft or fuzzy clustering. Hard clustering partitions the observations in the data set into K partitions, $C = \{C_1, \cdots, C_k\}$ $(K \leq N)$, such that the following three properties hold (Xu & Wunsch, 2005):

$$1)\ C_i \neq \emptyset, \quad i = 1, \cdots, K;$$
$$2)\ \bigcup_{i=1}^{K} C_i = \boldsymbol{X} \quad , \boldsymbol{X} = \{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n\}; \tag{3.1}$$
$$3)\ C_i \cap C_j = \phi, \qquad i, j = 1, \cdots, K \text{ and } i \neq j.$$

This implies that each observation only belongs to one cluster.

The remainder of this section provides a detailed explanation of clustering.

3.2.1 Distance and similarity measures

A primary requirement of a clustering algorithm is the ability to assess the closeness associated with a pair of observations, an observation and a cluster, or a pair of clusters. This is achieved through distance and similarity measures. The similarity or dissimilarity between different observations is stored in a symmetric matrix called the proximity matrix, whose size for a data set with n observations will be $n \times n$. The (i, j)th element in the proximity matrix refers to the similarity or dissimilarity measure for the ith and jth observations $(i, j = 1, \cdots, n)$. It must be noted that dissimilarity measures can be obtained from similarity measures as follows (Xu & Wunsch, 2005):

$$D_{ij} = 1 - S_{ij}, \tag{3.2}$$

where S_{ij} represents the similarity between the ith and jth observations, and $D_{ij}, S_{ij} \in [0,1]$.

The type of measure used is dependent on the type of features associated with the dataset. Features can be quantitative or qualitative, continuous or binary, nominal or ordinal, each type requiring a different type of measuring function. In dealing with continuous features, distance functions are used whereas qualitative features require the use of similarity measures (Xu & Wunsch, 2005). There are a variety of distance functions to choose from, these will not be explored in this report for the sake of brevity.

Binary features use similarity measures, the two most common being the following (Xu & Wunsch, 2005):

$$S_{ij} = \frac{n_{11} + n_{00}}{n_{11} + n_{00} + w(n_{10} + n_{01})}$$
$$w = 1, \text{ simple matching coefficient}$$
$$w = 2, \text{Rogers and Tanimoto measure} \tag{3.3}$$
$$w = \frac{1}{2}, \text{Gower and Legendre measure,}$$

$$S_{ij} = \frac{n_{11}}{n_{11} + w(n_{10} + n_{01})}$$
$$w = 1, \text{ Jaccard coefficient}$$
$$w = 2, \text{Sokal and Sneath measure} \tag{3.4}$$
$$w = \frac{1}{2}, \text{Gower and Legendre measure,}$$

where n_{00} and n_{11} represent the number of concurrent absences or presence of features in the two observations, and n_{01} and n_{10} take into account the features present only in one object. The measures defined by (3.3) compute the match between two observations directly while the measures defined by (3.4) focus on the co-occurrence features but ignore the effect of co-absence (Xu & Wunsch, 2005).

Nominal features are usually mapped into binary features. Ordinal features on the other hand can be compared using continuous dissimilarity measures as they order multiple states according to some reference. A common approach to handling observations consisting of mixed variables or features involves mapping all these variables into the interval [0,1] and using a distance measure like Euclidean distance. However, this approach is prone to the drawback of information loss. A better approach is to use a method proposed by Gower which is of the form (Xu & Wunsch, 2005):

$$S_{ij} = \frac{\sum_{l=1}^{d} \eta_{ijl} S_{ijl}}{\sum_{l=1}^{d} \eta_{ijl}}, \tag{3.5}$$

where S_{ijl} refers to the similarity for the lth feature which is calculated depending on the feature's variable type e.g. if it is a nominal or binary variable, the similarity is 1 if values of the two observations are equal and 0 otherwise. In the case of continuous variables their similarity is the absolute difference of the two values, normalised to the range of the feature. The indicator η_{ijl} is a 0 or 1 coefficient, which takes the value of 0 if the feature is missing in either or both the observations (i and j) and 1 otherwise (Xu & Wunsch, 2005).

3.2.2 An overview of different types of clustering algorithms

There are a variety of clustering algorithms in literature. This makes the task of categorising clustering algorithms difficult as in some cases the categories that these algorithms fall under may overlap (Han, et al., 2011). Clustering algorithms can be mainly categorised into hierarchical, partitioning, density-based and grid-based methods (Han, et al., 2011). This categorisation does not cover all the different clustering algorithms. This subsection provides the basic methodology behind some of the most popular types of clustering algorithms in literature.

3.2.2.1 Hierarchical clustering

Hierarchical clustering algorithms makes use of the proximity matrix to re-arrange data into a vertical or ranked structure (Xu & Wunsch, 2005). There are two types of hierarchical clustering algorithms, namely agglomerative and divisive hierarchical clustering. The difference between the two being how they generate the hierarchical structure. The agglomerative approach is a bottom-up approach, which begins with each observation creating its own group or cluster. It then successively combines the neighbouring clusters, until all the clusters are combined into one or until a predefined termination criterion. The divisive approach is a top-down approach and works in the opposite way. It begins with all the observations in one cluster. After which this one cluster is split into smaller clusters, over successive iterations, until each observation is in a cluster on its own, or until a predefined termination criterion takes effect (Han, et al., 2011).

The results of a hierarchical clustering algorithm can be presented in the form of a binary tree or dendrogram. This representation allows for easy to interpret descriptions and visualisations of the potential data clustering structures. Hierarchical clustering algorithms do however suffer from many drawbacks. One such drawback being that they lack robustness and as a result of this they are sensitive to noise and outliers. This can be associated with that the fact that once an observation has been assigned to a cluster, it cannot be assigned to another cluster in the next iteration, which means that this type of algorithm cannot rectify any possible misclassifications once it has happened. Furthermore the computational complexity for most hierarchical clustering algorithms is in the order of $O(n^2)$. This makes them unsuitable for clustering large data sets due to the quadratic computational complexity. This can lead to excessive execution times and can create storage problems (Xu & Wunsch, 2005).

3.2.2.2 Partitioning based clustering methods

Unlike hierarchical clustering algorithms, which yield successive levels of clusters by iterative combinations or separations, partitioning methods assign a set of observations into k clusters without any vertical or ranked structure (Xu & Wunsch, 2005). They are generally distance-based methods. Suppose the number of partitions to construct is given by k, the partitioning method starts out by creating an initial partition. It then uses an iterative relocation technique in order to improve the partitioning by moving observations from one cluster to another. In this way clustering can be thought of as an optimisation problem as the objective is to organise a set of observations into k subsets based on some criterion function. Obtaining globally optimal solutions with partitioning based clustering methods is often computationally prohibitive as this requires computing a list of all possible solutions. As an alternative, it is more common to follow a pragmatic approach such as adopting iterative greedy descent approaches like the k-means and k-mediods algorithms (Han, et al., 2011). These algorithms iteratively improve the clustering solution in such a way that at each iterative step, the value of the criterion is improved from its previous value (Hastie, et al., 2009). This is repeated until a local optimum is reached. These heuristic clustering algorithms find spherical-based shaped clusters in the data (Han, et al., 2011). Further details of the k-means algorithm a type of partitioning based

clustering algorithm, which happens to also be one of the most popular clustering algorithms, will be provided in the next section.

3.2.2.3 Density based clustering methods

One of the drawbacks of most partitioning algorithms is that they can only find spherical-shaped clusters in the data and therefore find it difficult to find clusters of arbitrary shapes. A reason for this drawback is the fact that most partitioning methods use distance as the criterion for clustering. Density based clustering methods overcome this problem by using density as the criterion instead of distance. The basic idea is to continue growing a given cluster as long as the density i.e. the number of observations in the neighbourhood of an observation exceeds some user defined limit. For example, for each observation with a cluster, the neighbourhood, defined by a specified radius has to hold at least a minimum number of points. These types of methods are good at filtering out noise or outliers and at finding clusters of arbitrary shape. A popular density-based clustering method is the density based spatial clustering of applications with noise (DBSCAN) algorithm. Apart from the aforementioned advantages, another advantage of this algorithm is that the user does not need to specify the number of clusters in the data a priori. This technique however requires selecting appropriate parameter values for two input parameters, namely ϵ and $MinPts$ which refer to the maximum radius of a neighbourhood and the minimum number of points required in the neighbourhood of a core observation respectively. Like many other type of clustering algorithms, this algorithm is also susceptible to the choice of parameter values. These parameter values are empirically set and difficult to determine, especially for real-world, high-dimensional data sets (Han, et al., 2011).

3.2.2.4 Grid-based clustering methods

Grid-based methods discretize the space of observations into a finite number of cells in order to form a grid structure. All clustering operations are then performed on this grid structure. These algorithms offer fast processing time irrespective of the number of observations and are only dependent on the number of cells in each dimension in the grid structure. They can also be combined with other clustering methods such as density-based methods and hierarchical methods (Han, et al., 2011).

3.2.2.5 Other types of clustering algorithms

Apart from the four main categories of clustering algorithms discussed above, other important clustering methods worth discussing include mixture densities-based clustering and neural networks-based clustering methods. A very popular neural networks-based clustering algorithm is the SOM algorithm that has been discussed in Section 2.4.1. This algorithm works on the principle of competitive learning. In competitive learning based neural networks, active neurons reinforce their neighbourhood within certain regions, while suppressing the activities of other neurons (Xu & Wunsch, 2005).

The purpose of SOM is to project input patterns represented in a high dimensional space onto a two-dimensional grid map, in the context of neural networks this forms the output layer. This layer consists of output units which will form the derived clusters (Tsiptsis & Chorianopoulos, 2009). Input patterns are fully connected to the output unit via adaptable weights (Xu & Wunsch, 2005), which are initially set to random values and tuned as the model training process proceeds. When input patterns are introduced to the output layer, the output unit compete to win them. Input patterns or observations are assigned according to the Euclidean distance measure, with each record's input values being compared to the centres of the output unit and the most similar output unit winning the observation. This assignment also results in the corresponding weights being adjusted so that when an observation with similar traits is introduced to the output unit in the future, it has a better chance of obtaining it (Tsiptsis & Chorianopoulos, 2009).

Furthermore, when an output neuron wins a record the weights of neighbouring neurons, meaning the output neurons that are symmetrically around the winning neuron, are also adjusted. In this way, similar clusters appear closer together on the output map as neighbouring units. Output units which do not win any observations are taken out of the final solution (Tsiptsis & Chorianopoulos, 2009). As with other clustering algorithms, the choice of a number of user-dependent parameters causes problems when applying SOM on real world problems (Xu & Wunsch, 2005). Users are required to choose the topology of the solution i.e. the number of rows and columns of the output map which in turn can be

thought of as the number of clusters. SOM, once trained, can suffer from a problem whereby areas of low pattern density become over-represented meanwhile areas of high-density become under-represented, this problem is referred to as input space density misrepresentation (Xu & Wunsch, 2005). Lastly, SOM also requires many iterations and weight adjustments, subsequently making it considerably slower than other clustering algorithms like k-means (Tsiptsis & Chorianopoulos, 2009). However, SOM can be integrated with other clustering algorithms such as k-means to provide more effective and faster clustering (Xu & Wunsch, 2005).

The last type of clustering method to be discussed in this section is mixture density-based clustering methods. This approach uses a probabilistic view where data observations in different clusters are assumed to be generated by different probability distributions. These distributions can arise from different density functions or from similar density functions with different parameter values. The most popular density function being the Gaussian density function as a result of its analytical tractability and its complete theory. Once the distributions are known, obtaining the clusters of a given data set is comparable to estimating the parameters of many different models. Maximum likelihood (ML) estimation is a popular approach for parameter estimation, in most cases the solutions to the likelihood equations cannot be obtained analytically and must be obtained by using iterative approaches which are suboptimal. Expectation-maximization (EM) being the most popular among these approaches. However, the EM algorithm has some notable frailties including the sensitivity to the selection of initial parameters, the effect of a singular covariance matrix, the possibility of convergence to a local optimum and slow convergence rates. Lastly, it is also worth noting that the EM algorithm and k-means algorithm are related under the assumption of a spherical Gaussian mixture (Xu & Wunsch, 2005).

3.2.3 The k-means clustering algorithm

As mentioned in Section 3.2.2, k-means is a partitioning-based clustering algorithm. It is also one of the most popular iterative descent clustering methods (Hastie, et al., 2009) as it

is very simple and can be implemented to solve many practical problems (Xu & Wunsch, 2005). In order to fully explain the k-means algorithm, it is first necessary to explain some of the underlying concepts starting with the concept of within-cluster variation or with-in cluster point scatter. As k-means is an iterative descent clustering method, it needs a loss or objective function to minimize, which in this case refers to how good the clustering solution is, as with any optimisation problem. Since the goal is to assign observations that are near each other to the same cluster, an ideal loss function is the within-cluster variation defined as follows (Hastie, et al., 2009):

$$W(C) = \frac{1}{2}\sum_{k=1}^{K}\sum_{C(i)=k}\sum_{C(i')=k} D_{ii'}, \tag{3.6}$$

where i refers to an observation within the cluster associated with k and i' refers to another observation within that cluster which is not i. This criterion gives a measure of the closeness of the sample belonging to the same cluster (Hastie, et al., 2009).

It is also worth noting that the total variation or total point scatter, which is a constant given the data and is independent of cluster assignment, is given by (Hastie, et al., 2009):

$$T = \frac{1}{2}\sum_{i=1}^{n}\sum_{i'=1}^{n} D_{ii'} = \frac{1}{2}\sum_{k=1}^{K}\sum_{C(i)=k}\left(\sum_{C(i')=k} D_{ii'} + \sum_{C(i')\neq k} D_{ii'}\right) \tag{3.7}$$

or

$$T = W(C) + B(C). \tag{3.8}$$

The term $B(C)$ in (3.8) refers to between-cluster variation, which is defined as follows (Hastie, et al., 2009):

$$B(C) = \frac{1}{2}\sum_{k=1}^{K}\sum_{C(i)=k}\sum_{C(i')\neq k} D_{ii'}. \tag{3.9}$$

It acquires a large value when observations in different clusters are far apart. It can also be noted that (Hastie, et al., 2009)

$$W(C) = T - B(C). \tag{3.10}$$

Therefore minimizing $W(C)$ is equivalent to maximizing $B(C)$.

The k-means algorithm is intended for the situation where all variables are numerical variables or quantitative variables, and squared Euclidean distance (Hastie, et al., 2009)

$$D_{ii'} = \sum_{j=1}^{p} \left(x_{ij} - x_{i'j}\right)^2 = \|\boldsymbol{x}_i - \boldsymbol{x}_{i'}\|^2 , \tag{3.11}$$

is chosen as the dissimilarity measure in (3.6). The within-cluster variation then takes on the following form (Hastie, et al., 2009):

$$\begin{aligned} W(C) &= \frac{1}{2}\sum_{k=1}^{K} \sum_{C(i)=k} \sum_{C(i')=k} \|\boldsymbol{x}_i - \boldsymbol{x}_{i'}\|^2 \\ &= \sum_{k=1}^{K} N_k \sum_{C(i)=k} \|\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k\|^2, \end{aligned} \tag{3.12}$$

where $\bar{\boldsymbol{x}}_k = (\bar{x}_{1k}, \dots, \bar{x}_{pk})$ is the mean vector associated with the kth cluster, and $N_k = \sum_{i=1}^{n} I(C(i) = k)$. Thus, the objective function for the clustering optimisation problem is minimized by assigning the n observations to the K clusters in a manner that minimizes the average dissimilarity between the points in a particular cluster and the cluster mean for each and every cluster. The optimisation problem can be written as follows (Hastie, et al., 2009):

$$C^* = \min_{C} \sum_{k=1}^{K} N_k \sum_{C(i)=k} \|\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k\|^2. \tag{3.13}$$

An iterative descent algorithm for solving (3.13) can be obtained by noting that for any set of observations S

$$\bar{\boldsymbol{x}}_S = \operatorname*{argmin}_{m} \sum_{i \in S} \|\boldsymbol{x}_i - \boldsymbol{m}\|^2 \ . \tag{3.14}$$

The optimisation problem in (3.13) then becomes

$$\min_{C, \{\boldsymbol{m}_k\}_1^K} \sum_{k=1}^{K} N_k \sum_{C(i)=k} \|\boldsymbol{x}_i - \boldsymbol{m}_k\|^2 . \tag{3.15}$$

This optimisation problem can be solved using an iterative optimisation procedure described in Algorithm 3.1 below.

Algorithm 3.1 k-means Clustering (Hastie, et al., 2009)

1. For a given cluster assignment C, the total cluster variance, given in (3.15), is minimized with respect to $\{m_1, \dots, m_K\}$ to obtain the cluster centroids as per (3.14).
2. Once a set of cluster centroids $\{m_1, \dots, m_K\}$ are obtained, (3.15) is minimized by assigning each observation to the closest (current) cluster centroid. This can be represented as follows:

$$C(i) = \underset{1 \le k \le K}{\operatorname{argmin}} \|\boldsymbol{x}_i - \boldsymbol{m}_k\|^2. \tag{3.16}$$

3. Repeat steps 1 and 2 until the assignments do not change.

The k-means clustering algorithm starts with the random assignment of each of the observations to one of the K clusters, this occurs before step 1 in Algorithm 3.1. Furthermore, the k-means algorithm reaches a final solution, which is a local optimum, when there are no more changes to the result as per step 3 in Algorithm 3.1. Since the k-means algorithm finds a local optimum instead of a global optimum, the initial random assignment of each observation to a cluster, mentioned earlier, has a huge bearing on the final result. Therefore, it is important to run the algorithm multiple times from different random initial configurations. The final solutions will then be the one for which the objective function in (3.15) is smallest (James, et al., 2013). Furthermore it must be noted that the number of clusters in the data set is an input required from the user with the k-means algorithm. Unfortunately, there is no efficient and universally accepted method for identifying the initial partitions and the number of clusters K (Xu & Wunsch, 2005).

One of the biggest advantages of the k-means clustering algorithm is that the algorithm has a time complexity of $O(NKd)$ and space complexity of $O(N + K)$. Since N is usually much larger than both $K$and d, the complexity becomes near linear to the number of samples in the data set. This makes the k-means algorithm very efficient at clustering large data sets. Although, it is worth noting that the k-means algorithm does have some drawbacks. However, these drawbacks are also well studied and in some cases various variants of k-means have been proposed to overcome some of these misgivings (Xu & Wunsch, 2005).

A drawback that the k-means algorithm has, which has already been highlighted in this section is that the user is required to stipulate the number of clusters K in the dataset a priori. A proposed solution to this problem is a technique called ISODATA which deals with the estimation of K. This technique dynamically changes the number of clusters by merging and splitting clusters based on a predefined threshold. This changes the problem of identifying the initial number of clusters into one of fine tuning the threshold parameter. The second drawback that has already been highlighted is that the k-means algorithm cannot offer convergence to a global optimum. A possible solution to this is to use stochastic optimisation techniques like simulated annealing, genetic algorithm (GA) or particle swarm optimisation (PSO) to find a global optimum; the downside to this approach being that these optimisation algorithms are very computationally expensive (Xu & Wunsch, 2005).

An additional disadvantage that the k-means algorithm has, is that it is sensitive to outliers and noise (Xu & Wunsch, 2005). The reason for this is that outliers tend to warp the position of the cluster centroid as they are much further away from a greater part of the data. This can lead to observations being assigned to the wrong clusters. This problem is compounded by using squared Euclidean distance as the dissimilarity measure (Han, et al., 2011). Squared Euclidean distance places greater importance on greater distances and outliers produce very large distances (Hastie, et al., 2009). As a solution to this problem both ISODATA and the Partitioning Around Medoids (PAM) algorithms account for the effect of outliers in the clustering procedure. ISODATA removes clusters with a small number of observations. The splitting operation of ISODATA eradicates the presence of elongated clusters which are typical of k-means. PAM on the other hand uses real data points (medoids) as the observation that is representative of other observations in the cluster, subsequently avoiding the effect of outliers (Xu & Wunsch, 2005). Unfortunately, solving this challenge comes out at the expense of extra computation (Hastie, et al., 2009). It is also worth noting that the k-medoids algorithm follows similar principles as the PAM algorithm and uses the discrete 1-medians as the cluster centroids (Xu & Wunsch, 2005).

The k-medoids algorithm also helps to eliminate another flaw in the k-means formulation which is that the definition of "means" limits the application of the k-means algorithm only to numerical variables. The k-medoids is applicable to problems where the calculation of means is not possible, as the medoids require no computation and are unfailingly available (Xu & Wunsch, 2005). An enhancement of the k-means algorithm which caters for categorical variables comes in the form of the k-modes algorithm, which replaces the means of clusters with modes. Furthermore, it makes use of different dissimilarity measures to extend k-means to categorical variables and uses a frequency-based method to update modes of clusters (Han, et al., 2011). Otherwise, it operates in the same way as the k-means algorithm (Xu & Wunsch, 2005).

3.2.4 Assessing clustering quality

A majority of clustering algorithms rely on some assumptions in order to partition the data set into clusters. Therefore, it is necessary to establish some way to evaluate the validity of the resulting clusters. An effective evaluation metric would need to assess the quality of the clusters, the degree to which a particular clustering scheme fits a specific data set and determine the most effective number of clusters in the data set (Charrad, et al., 2014). In the case of most clustering problems, the ground truth is unavailable and one cannot rely on extrinsic methods which would have compared the clustering against the group truth. Instead most clustering validity measures can be thought of as intrinsic methods, which evaluate the effectiveness of a clustering by considering how well the clusters are separated and how compact the clusters are (Han, et al., 2011).

Another matter that complicates the process of evaluating cluster validity is that a wide variety of validation indices have been proposed in literature over the years with no consensus on the most effective one. Furthermore, most software packages do not implement all the indices and programs are unavailable to test these indices and compare them. To that extent, Charrad et al (Charrad, et al., 2014) provides a comprehensive overview of over 30 validation indices, any reader requiring further information with regards to this particular topic should refer to this paper. As it is quite exhaustive to go

through all these indices, this study looks at a cluster validity index or measure known as the silhouette coefficient. This measure assesses both internal cohesion and the external separation of a clustering solution (Tsiptsis & Chorianopoulos, 2009). It is a popular clustering validity measure and has been highlighted in (Han, et al., 2011), (Tsiptsis & Chorianopoulos, 2009) and (Charrad, et al., 2014).

The silhouette coefficient is obtained as follows: firstly, in the case of each observation i in a cluster the average Euclidean distance to all other observations in the same cluster is obtained as $a(i)$. This value reflects the compactness of the cluster to which this observation currently belongs to. Next, for every observation i and for every cluster which does not have i as a member, compute the average Euclidean distance of the observation to all the members of the neighbouring cluster. After doing this for all clusters where i is not a member, compute $b(i)$ as the minimum such distance in terms of all clusters. This value reflects how far away this particular observation is from other clusters. The silhouette coefficient for the observation i can be obtained as follows (Tsiptsis & Chorianopoulos, 2009):

$$Si = \frac{[b(i) - a(i)]}{\max\{a(i), b(i)\}}. \tag{3.17}$$

The values of the silhouette coefficient range from -1 to 1, with values closer to 1 indicating a good clustering. This is because a value of 1 would mean that the $a(i)$ value is close to zero and therefore points to perfect homogeneity (Tsiptsis & Chorianopoulos, 2009). A silhouette coefficient value close to 1 for an observation indicates that the cluster containing this observation is compact and that this observation is far away from other clusters. A negative value for the silhouette coefficient indicates that the observation is closer to observations in another cluster than to the observations in its current cluster, which is undesirable (Han, et al., 2011).

The overall silhouette coefficient offers a quantifiable degree to the effectiveness of the entire clustering solution and is obtained by averaging the silhouette coefficients for all the observations. In general, it is suggested that an average silhouette coefficient greater than 0.5 indicates a reasonable good clustering solution, while a coefficient less than 0.2

indicates a poor clustering solution. Although the above mentioned validation measure makes intuitive sense and can help in identifying the most desirable solution, it is recommended that an analyst should not base their decision purely based on these validation measures. According to (Tsiptsis & Chorianopoulos, 2009) "a clustering solution is justified only if it makes business sense. Potential business value, interpretability and ease of use are factors that are the best benchmarks for determining the optimal clustering solution". In the case of this study, the approach followed by Quilumba *et al.* (Quilumba, et al., 2015) will be utilised. This means that the most effective number of clusters will be ascertained by finding the number of clusters that minimizes the overall forecasting error for the test set rather than relying on a clustering validity index. In the context of this study, the potential business value of minimizing the overall forecasting error is that it allows the bank to provision more funds for giving out loans and other forms of credit, resulting in higher revenue.

3.2.5 Clustering high dimensional data

Most clustering algorithms are not sufficient for handling high-dimensional data and work best when the number of features is small i.e. approximately less than 10 attributes (Han, et al., 2011). This is as a result of the "curse of dimensionality", which refers to the exponential growth of complexity in the case of multivariate function estimation under a high dimensionality feature space. According to (Xu & Wunsch, 2005), clustering algorithms based on distance measures may not be very effective in a high-dimensional space. This is as a result of the fact that the distance between two points that are near each other becomes no different from that of two points that are not close together when the dimensionality of the space is high enough (Xu & Wunsch, 2005).

In the case of most high-dimensional data problems, the data occupy a manifold with an intrinsic dimensionality that is much lower than the feature space dimensionality. Therefore, an obvious and commonly applied solution to the high-dimensionality problem is to make use of dimension reduction techniques. Dimensionality reduction not only helps to deal with the problem of clustering high-dimensional data but also helps to reduce

computational cost and makes it easier to visually inspect the clustering solution (Xu & Wunsch, 2005).

A common approach to using dimensionality reduction along with clustering is to use Principal Component Analysis (PCA) to extract important components from the original data, which are then used to do the clustering. PCA is an unsupervised learning technique that accounts for a significant portion of the variation in the data set while projecting the data onto a low-dimensional feature space (James, et al., 2013). It achieves this by finding the principal components of the data which are a sequence of projections of the data, mutually uncorrelated and ordered in variance. The principal components of a set of data in $\mathbb{R}^p$ provide a sequence of best linear approximations to that data, of all ranks $q \leq p$ (Hastie, et al., 2009).

The first principal component of a set of features $x_1, x_2, \dots, x_p$ is the normalised linear combination of the features (James, et al., 2013):

$$Z_1 = \phi_{11}x_1 + \phi_{21}x_2 + \cdots + \phi_{p1}x_p, \tag{3.18}$$

that has the largest variance. Normalized meaning that $\sum_{j=1}^{p} \phi_{j1}^2 = 1$. The elements $\phi_{11}, \dots, \phi_{p1}$ are the loadings of the first principal component; together they make up the principal component loading vector $\boldsymbol{\phi}_1 = \left(\phi_{11}, \phi_{21}, \dots, \phi_{p1}\right)^T$. For a data set $\boldsymbol{X}$ with dimensions $n \times p$, the process of obtaining the first principal component works as follows: first, each of the variables in $\boldsymbol{X}$ are centred to have a mean of zero. Then the next step is to obtain the linear combination of the sample feature values of the form:

$$z_{i1} = \phi_{11}x_{i1} + \phi_{21}x_{i2} + \cdots + \phi_{p1}x_{ip}, \tag{3.19}$$

that has the largest sample variance, subject to the constraint that $\sum_{j=1}^{p} \phi_{j1}^2 = 1$. The loading vector $\boldsymbol{\phi}_1$ which is utilised in (3.19) is obtained by solving the following optimisation problem (James, et al., 2013):

$$\max_{\phi_{11},\dots,\phi_{p1}} \left\{ \frac{1}{n} \sum_{i=1}^{n} \left(\sum_{j=1}^{p} \phi_{j1}x_{ij} \right)^2 \right\} \text{ subject to } \sum_{j=1}^{p} \phi_{j1}^2 = 1. \tag{3.20}$$

The loading vector $\boldsymbol{\phi}_1$ defines the direction in feature space along which the data vary the most. The projected values of the n data points $x_1, \ldots, x_n$ onto this direction are given by the terms $z_{11}, \ldots, z_{n1}$, which are referred to as the scores of the first principal component (James, et al., 2013).

Once the first principal component Z_1 has been obtained, the second principal component is obtained as the linear combination of $x_1, x_2, \ldots, x_p$ that has maximal variance out of all linear combinations that are uncorrelated with Z_1. In constraining Z_2 to be uncorrelated with Z_1, the result is equivalent to constraining the direction $\phi_2$to be orthogonal to the direction of $\boldsymbol{\phi}_1$. In order to solve for $\boldsymbol{\phi}_2$, $\boldsymbol{\phi}_1$ in (3.20) is replaced with $\boldsymbol{\phi}_2$ and an additional constraint is introduced which is that $\boldsymbol{\phi}_2$ has to be orthogonal to $\boldsymbol{\phi}_1$. The remaining principal components are obtained in a similar way. More formally, the principal component directions $\phi_1, \phi_2, \phi_3, \ldots$ are the ordered sequence of eigenvectors of the matrix $\boldsymbol{X}^T\boldsymbol{X}$, and the variances of the components are the eigenvalues. There are at most $\min(n-1, p)$ principal components (James, et al., 2013).

In order to determine how many of the principle components to keep, one needs to determine the proportion of variance explained (PVE) by each principal component. The total variance present in the data set, based on the assumption that the variables have been centred to have mean of zero, is given as follows (James, et al., 2013):

$$\sum_{j=1}^{p} Var(X_j) = \sum_{j=1}^{p} \frac{1}{n} \sum_{i=1}^{n} x_{ij}^2, \tag{3.21}$$

and the variance explained by the mth principal component is

$$\frac{1}{n} \sum_{i=1}^{n} z_{im}^2 = \frac{1}{n} \sum_{i=1}^{n} \left(\sum_{j=1}^{p} \phi_{jm} x_{ij} \right)^2 . \tag{3.22}$$

The PVE of the mth principal component is computed as follows (James, et al., 2013):

$$\frac{\sum_{i=1}^{n} \left(\sum_{j=1}^{p} \phi_{jm} x_{ij} \right)^2}{\sum_{j=1}^{p} \sum_{i=1}^{n} x_{ij}^2}. \tag{3.23}$$

The PVE of each principal component is a positive number and PVEs of all the principal components sum to one (James, et al., 2013).

Since the goal of using PCA was to reduce the dimensionality of the original data set, the analyst tries to identify the least number of principal components that account for a sizeable portion of variance in the data. A way to decide on this number is to examine a plot called the scree plot, which plots the proportion of variance explained against the corresponding principal component. One then looks for a point on the plot at which the proportion of variance explained by each subsequent principal component decreases significantly. The point at which this happens is usually called the elbow in the scree plot. Unfortunately, this type of graphical analysis is essentially ad hoc but is the most preferred option, as there is no analytical computation available to determine the optimal number of principal components. In practice, the first few principal components should be able to explain the data quite well and should lead to some interesting patterns. If this is not the case, looking at more principal components is unlikely to yield a better result (James, et al., 2013).

3.3 ARIMA MODELS

The problem of predicting the future daily bank balance of a portfolio of accounts is a time series forecasting problem. This section introduces the concept of time series forecasting as well as the type of time series forecasting model that will be used in this study, known as the ARIMA model.

According to (Deb, et al., 2017) “a time series is an ordered sequence of values recorded over equal intervals of time”. It can be either univariate or multivariate. In the univariate case, the time series contains a single variable recorded chronologically over time. Meanwhile, a multivariate time series refers to the case where there is a group of time series variables and one has to also consider their interactions (Deb, et al., 2017). This study deals only with univariate time series.

Time series forecasting is the process of using a model to predict future values of a time series based on previously observed values and can be defined as follows:

$$\hat{y}_{T+1|T} = f(y_T, y_{T-1}, \dots, y_1). \tag{3.24}$$

A popular time series model utilised for time series forecasting is the ARIMA model. It is primarily utilised when the time series in question is non-stationary. A time series is deemed to be stationary if its statistical properties (probability distribution) do not depend on the time at which the series is observed. In general, a stationary time series has no predictable patterns in the long-term. Therefore, a time series with trends or seasonality are referred to as non-stationary as these components affect the value of the time series at different times (Hyndman & Athanasopoulos, 2014).

A trend can be described as a long-term increase or decrease in the data. Meanwhile a seasonal pattern refers to the case where seasonal factors such as the time of the year, month or day of the week have an effect on the time series. Lastly time series can also exhibit cyclic patterns, this occurs when data exhibits rises and falls that are not of a fixed period i.e. these patterns have variable and unknown length unlike seasonal patterns that have a fixed and known length (Hyndman & Athanasopoulos, 2014). According to (Hyndman & Athanasopoulos, 2014) “a time series with cyclic behaviour, but no trend or seasonality, is stationary”.

An ARIMA model is based on the idea of transforming the time series to be stationary by making use of differencing (Deb, et al., 2017). Differencing computes the differences between consecutive observations. It helps to stabilise the mean of a time series by removing changes in the level of a time series, and therefore discarding trend and seasonality. First order differencing is the change between consecutive observations and can be written as follows (Hyndman & Athanasopoulos, 2014):

$$y'_t = y_t - y_{t-1}. \tag{3.25}$$

The differenced series will only have $T-1$ values as it is not possible to calculate a difference y'_1 for the first observation. If the data is not stationary after one order of

differencing, the data might have to be differenced a second time to obtain a time series that is stationary. This is referred to as second order differencing and can be obtained as follows (Hyndman & Athanasopoulos, 2014):

$$\begin{aligned} y_t'' &= y_t' - y_{t-1}' \\ &= (y_t - y_{t-1}) - (y_{t-1} - y_{t-2}) \\ &= y_t - 2y_{t-1} + y_{t-2}. \end{aligned} \tag{3.26}$$

It is almost never necessary to go beyond second-order differences, but if need be the idea above could be expanded (Hyndman & Athanasopoulos, 2014).

The seasonal difference of a time series is the series of changes from one season to the next. For example with monthly data there are 12 periods in a season and the seasonal difference of y at period t would then be $y_t' = y_t - y_{t-12}$ (The Pennsylvania State University, 2018). To put it more formally seasonal differencing can be represented as (Hyndman & Athanasopoulos, 2014):

$$y_t' = y_t - y_{t-m} \quad \text{where } m = \text{ number of periods per season.} \tag{3.27}$$

In certain cases, it will be necessary to have both seasonal differencing and ordinary differencing to obtain stationary data (Hyndman & Athanasopoulos, 2014).

An ARIMA model consists of three components, namely an autoregression part, differencing and a moving average part. Therefore, it is necessary to explain the different parts, with the differencing aspect having already been explained above. An autoregression model forecasts a variable of interest based on a linear combination of past values of the variable. Thus an autoregression model of order p can be written as follows (Hyndman & Athanasopoulos, 2014):

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + e_t, \tag{3.28}$$

where c is a constant and e_t is white noise. Equation (3.28) is usually referred to as an AR(p) model. Unlike the autoregression model, a moving average model uses past forecasting errors as follows (Hyndman & Athanasopoulos, 2014):

$$y_t = c + e_t + \theta_1 e_{t-1} + \theta_2 e_{t-2} + \cdots + \theta_q e_{t-q}, \tag{3.29}$$

where e_t is white noise. Equation (3.29) is usually referred to as an MA(q) model.

Combining differencing with autoregression and a moving average model produces a non-seasonal ARIMA model, which is given as follows (Hyndman & Athanasopoulos, 2014):

$$y'_t = c + \phi_1 y'_{t-1} + \cdots + \phi_p y'_{t-p} + \theta_1 e_{t-1} + \cdots + \theta_q e_{t-q} + e_t, \tag{3.30}$$

where y'_t is the differenced series, which could have been differenced more than once. Equation (3.30) is referred to as an ARIMA(p, d, q) model, where p is the order of the autoregressive part, d is the degree of order differencing involved and q is the order of the moving average part.

A seasonal ARIMA model is formed by including additional seasonal terms in the ARIMA models discussed thus far. It is written as follows (Hyndman & Athanasopoulos, 2014):

$$\text{ARIMA} \quad \underbrace{(p, d, q)}_{\substack{\uparrow \\ \text{Non-seasonal part of the model}}} \quad \underbrace{(P, D, Q)_m}_{\substack{\uparrow \\ \text{Seasonal part of the model}}}, \tag{3.31}$$

where m is the number of periods per season as defined earlier. In a seasonal ARIMA model, seasonal AR and MA terms predict y'_t using data values and errors at times with lags that are multiples of m (the span of the seasonality) (The Pennsylvania State University, 2018).

A variety of metrics are available to determine the order of an ARIMA model e.g. log likelihood, Akaike's Information Criterion (AIC), corrected AIC (AICc) and Bayesian Information Criterion (BIC). However, the best metric is to use the error on an out-of-sample set i.e. test set or validation set. This offers an advantage over the other metrics in that when models are evaluated using the aforementioned metrics, it is important that all models have the same orders of differencing. However, when comparing models using a test set, this is not a constraint. Therefore, we can compare various types of models, from those with only seasonal differencing to models with ordinary and seasonal differencing etc. Furthermore, in practice one would pick the best model regardless of whether or not it passes any residual tests (a test to see if there are any patterns that have still not been accounted for by the model) (Hyndman & Athanasopoulos, 2014).

Lastly the forecasts are obtained as follows: firstly, one expands the ARIMA equation so that y_t is on the left hand side and all other terms on the right. Then, the equation is rewritten by replacing t by $T + h$. Then on the right hand side of the equation, future observations are replaced by their forecasts, future errors by zero, and past errors by the corresponding residuals. This process begins at $h = 1$, they are then repeated for $h = 2,3,$ ... until all the forecasts are calculated (Hyndman & Athanasopoulos, 2014).

3.4 LINEAR DISCRIMINANT ANALYSIS

The k-means clustering algorithm discussed earlier forms the clusters or segments of customers that spend or accumulate their balance in similar ways. These segments are formed based on the customer base that is currently available for the study, however from an operational perspective it will be necessary to assign future customers to one of the identified segments. The problem of scoring or assigning a future customer to one of the identified segments is referred to as a classification problem. As there could be more than two segments this problem could well be a multi-class classification problem. This section introduces a linear or parametric method that is widely used for these type of problems, known as linear discriminant analysis (LDA).

A multi-class classification problem requires that an observation be classified into one of K classes, where $K \geq 2$. In this case, the target variable Y can take on K possible unique and unordered values. According to statistical decision theory, one needs to find the class posterior probabilities $\Pr(Y|X)$ for optimal classification. This is because the Bayes classifier, which assigns an observation to the class for which the class posterior probability $\Pr(Y = k|X)$ is largest, has the lowest possible error rate out of all classifiers (Hastie, et al., 2009).

Given that $f_k(X) \equiv \Pr(X = x|Y = k)$ is the class-conditional density of X in class $Y = k$, and π_k is the prior probability of class k, with $\sum_{k=1}^{K} \pi_k = 1$. The posterior probability can be computed using Bayes theorem as follows:

$$\Pr(Y = k|X = x) = \frac{\pi_k f_k(x)}{\sum_{l=1}^{K} \pi_l f_l(x)}. \tag{3.32}$$

The posterior probability $\Pr(Y = k|X = x)$ can be computed indirectly by plugging in estimates for π_k and $f_k(x)$. Estimates of π_k can be obtained by calculating the fraction of observations from the observations that belong to the kth class in the training set. However, estimates of $f_k(x)$ are difficult to obtain unless simple forms of these densities are assumed (James, et al., 2013). One possible assumption is to treat each class density as a multivariate Gaussian distribution given below (Hastie, et al., 2009):

$$f_k(\boldsymbol{x}) = \frac{1}{(2\pi)^{p/2}|\boldsymbol{\sigma}_k|^{\frac{1}{2}}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_k)^T \boldsymbol{\sigma}_k^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_k)}. \tag{3.33}$$

In the above equation $\boldsymbol{\mu}_k$ is the mean of class k. LDA arises in the case where the above assumption is made along with another assumption that the classes have a common covariance matrix $\boldsymbol{\sigma}_k = \boldsymbol{\sigma}\ \forall k$. In practice the parameters of the Gaussian distributions in equation (3.33) are obtained by using estimates from the training data as follows (Hastie, et al., 2009):

$$\begin{aligned} \hat{\boldsymbol{\mu}}_k &= \sum_{g_i=k} \frac{\boldsymbol{x}_i}{N_k}, \\ \hat{\boldsymbol{\sigma}} &= \sum_{k=1}^{K} \sum_{g_i=k} ((\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_k)(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_k)^T)/(N - K). \end{aligned} \tag{3.34}$$

LDA allocates an observation to the class for which the linear discriminant function given below:

$$\delta_k(\boldsymbol{x}) = \boldsymbol{x}^T \boldsymbol{\sigma}^{-1} \boldsymbol{\mu}_k - \frac{1}{2} \boldsymbol{\mu}_k \boldsymbol{\sigma}^{-1} \boldsymbol{\mu}_k + \log \pi_k \tag{3.35}$$

is largest (Hastie, et al., 2009).

3.5 RANDOM FOREST

Section 3.4 describes a parametric method that can be used to solve the classification problem of scoring a future customer into one of the identified segments. This section looks at a very popular non parametric classification algorithm known as the random forests algorithm that can be used for the same purpose.

The random forests algorithm is a significant improvement over a popular technique that makes use of a recursive binary partitioning algorithm known as a decision tree. A decision tree splits the feature space into a set of rectangles, and then fits a simple model such as a constant in each one. It uses recursive binary partitions, which works as follows: it starts with splitting the feature space into two regions by choosing a variable and split-point that achieves the best fit. The process of choosing a splitting variable and a split point is achieved through a greedy approach. Afterwards one or both of these regions are divided into two more regions, this continue until a chosen stopping criterion is met (Hastie, et al., 2009). In the representation of a decision tree a region R_m is denoted by a node m, with N_m observations. The observations in a node m are assigned to the majority class in that node, which can be defined as class $k(m) = \text{argmax}_k \hat{p}_{mk}$, where $\hat{p}_{mk}$ is the proportion of class k observations in node m, given as follows (Hastie, et al., 2009):

$$\hat{p}_{mk} = \frac{1}{N_m} \sum_{x_i \in R_m} y_i = k. \tag{3.36}$$

One of the drawbacks of decision trees is that they suffer from high variance. A solution for this is to use bootstrap aggregation or bagging, which is a method that helps to reduce the variance of an estimated prediction function. This technique works very well for high-variance, low-bias procedures, such as decision trees (Hastie, et al., 2009).

Bagging uses a statistical tool called the bootstrap, which does random sampling with replacement, to create B different bootstrap-sampled versions of the training data set. It then fits the same classification tree multiple times to these different training sets. A committee of these B different decision trees then cast a vote for the predicted class

(Hastie, et al., 2009). It must be noted that these trees are generally grown fully i.e. there is no pruning of the trees.

Random forests is an improvement on bagging in that it builds a large group of de-correlated trees, and then coalesces their results by taking the average. As in the case of bagging, random forests also builds the decision trees on bootstrapped training samples. Random forests is different from bagging in the sense that random forests only consider a random sample of m predictors as split candidates from the full set of p predictors, when fitting these decision trees. The split is allowed to use only one of the aforementioned m predictors. A new sample of m predictors is taken at each split, with a common choice for $m = \sqrt{p}$. By decorrelating the trees in this manner random forests make the average of the resulting trees less variable and hence more dependable. In order to avoid overfitting a sufficiently large number of trees have to be grown (James, et al., 2013). The random forest algorithm as described above is shown in Algorithm 3.2.

Algorithm 3.2 Random Forest for Classification (Hastie, et al., 2009)

1. For $b = 1$ to B:
    a) Make use of bootstrap sampling on the training set to obtain a sample of size N.
    b) Construct a random-forest tree T_b using aforementioned bootstrapped data. This is done by recursively repeating the steps listed below for each terminal node of the tree. The steps below are repeated up until the minimum node size n_{min} is reached.
        i. Choose m variables at random from the p variables.
        ii. Select the optimal variable/split-point out of the m.
        iii. Split the node into two daughter nodes.
2. Generate an collective of trees $\{T_b\}_1^B$.
3. If $\hat{C}_b(x)$ is the predicted class of the bth random-forest tree. Then $\hat{C}_{rf}^B(x) =$ majority vote $\{\hat{C}_b(x)\}_1^B$.

3.6 CHAPTER SUMMARY

This chapter covered the theory behind the techniques utilised in this study. In terms of cluster analysis, the k-means clustering algorithm and the various intricacies associated with this algorithm were introduced. ARIMA models for forecasting was also introduced, along with the theory of this model other practical aspects such as how to choose appropriate hyper-parameters were also discussed. Lastly two of the classifiers used in this study were also introduced in LDA and random forests.

The primary motivation behind using the k-means algorithm in this study was due to the computational constraints that were faced when doing this project. To provide some context around this, the data used for this study was provided by a financial institution. One of the conditions that had be to be adhered to in order to use this data for the study was that any experiments that had to be conducted on the data had to be done within the confines of the financial institution's computing environment. The computing platform that was used to conduct this study, as will be discussed in the following chapter, was a virtual machine with 32GB of RAM. This virtual machine was used by multiple users at any one moment (in fact it was being used by 10 or more analysts working at the financial institution, therefore the memory was shared amongst multiple users). Certain care had to be taken to ensure that the experiments in this study would not inconvenience the other users.

The memory available, along with the fact that it has to be shared, automatically rules out certain clustering algorithms like hierarchical clustering algorithms which requires obtaining a dissimilarity matrix of size n×n. Neural network based clustering algorithms like SOM are also computationally very intensive and not an option. The k-means algorithm is computationally very efficient even when compared to its variants like the k-mediods (Partitioning around mediods) algorithm which is less sensitive to outliers. The k-mediods algorithm requires computing pairwise distances which makes it more memory intensive. Lastly, in the dissertation the k-prototypes algorithm was not used mainly

because the data set to be used did not have any categorical features, it only had numerical features.

CHAPTER 4 METHOD AND DATA EXPLORATION

4.1 CHAPTER OBJECTIVE

This chapter provides an overview of the systems and data used for this study. It also provides the methodology that was followed and details how the concepts covered across the previous two chapters were utilised in the study.

4.2 SYSTEMS USED FOR THE STUDY

The analysis in this study was conducted on a virtual machine with specifications listed in Table 4.1 below.

Table 4.1 Hardware specifications of the systems used in the study

Processor	Intel(R) Xeon (R) CPU E5-2660 v3 @ 2.60 GHz 2.59 GHz (4 processors)
Installed memory (RAM)	32.0 GB
System type	64-bit Operating System

RStudio with R version 3.3.0 (R Core Team, 2013) was the programming language used to conduct the analysis. In doing so, several external R packages were also utilised, these are listed below:

- The `fpp` package (Hyndman, 2013) was used for the purposes of fitting the ARIMA models and for forecasting.

- The `MASS` package (Venables & Ripley, 2002) was used for the purposes of fitting the LDA model.
- The `randomForest` package was used for fitting the random forest model.
- The `cluster` package (Maechler, et al., 2013) was used to compute the silhouette coefficient.

4.3 DATA EXPLORATION

The datasets used for this study covers information about 51317 different accounts from a high end DDA account segment of a bank. There are two different datasets, one which consists out of the daily balances for these accounts for the period 2013-06-01 to 2017-06-30, and another which consists out of the customer information available for these customers in 2013 June e.g. demographics, number of properties, number of bank products held by the customer etc. The population used for this study only consisted of accounts which maintained a positive balance i.e. a balance greater than zero, for at least 95% of the 1491 day period covered by the time series data. This was considered to be a limitation in this study as the characteristics of these customers would be quite similar.

The goal of this study was to obtain more accurate forecasts of the total daily bank balance time series for the overall account portfolio, the time series in question which is the total daily bank balance over the period 2013-06-01 to 2017-06-30 can be seen in Figure 4.1. As can be seen from Figure 4.1, the period 2013-06-01 to 2016-06-30 was used as the training and validation period for forecasting. This part of the time series data was also used to extract features for segmentation. The period 2016-07-01 to 2017-06-30 was used as the test set or out of sample set for forecasting. This period was used to assess the effectiveness of the proposed approach by measuring the accuracy of the forecasting. The features that were used for segmentation were obtained for the period 2013-06-01 to 2016-06-30 rather than 2016-07-01 to 2017-06-30 in order to get a long enough historical period to capture the various behavioural dynamics in the population. Also when fitting the forecasting model, it was necessary to have sufficient historical time series data to fit the

model. Furthermore, it made logical sense to build the segments over the period in which the forecasting model was being fitted rather than the period that was used to test the forecasting model's performance.

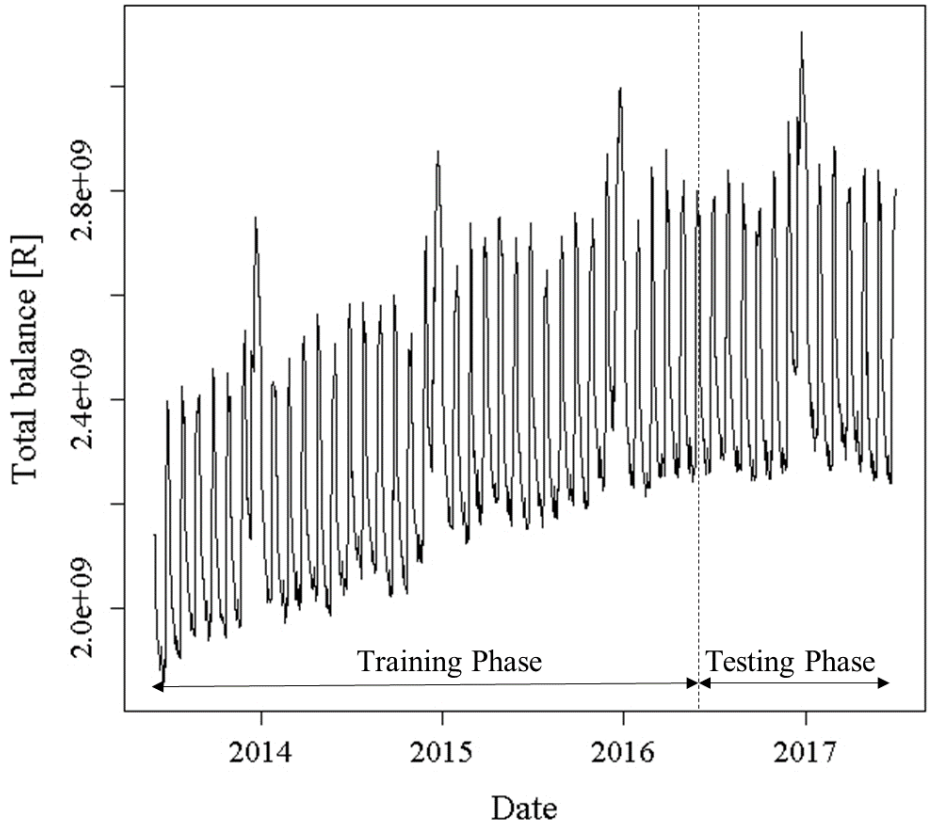


Figure 4.1. The time series data for the total daily bank balance for the population used for the study.

The variables included in the customer information dataset can be seen in Table 4.2. The categorical variables have been transformed into dummy variables (binary variables) for modelling purposes where necessary. This process involves creating a new binary variable for the different levels of a categorical variable. This variable takes a value of 1 when the value of the categorical variable is equal to the level for which the dummy variable was created and 0 otherwise. There is one fewer dummy variable than the number of levels in a categorical variable as one of the levels is kept as a baseline (James, et al., 2013).

Table 4.2 The variables in the customer information dataset.

Variable	Description of variable
Continuous variables	
CUST_AGE	Customer's Age
CUST_NO_CHILD	The number of children the customer has
CUST_TOT_NO_PROD	Total number of products with the bank
TOT_NO_SUBPROD	Total number of sub-products with the bank
NO_DDA_ACCT	Number of cheque or savings accounts with the bank
NO_ILP_ACCT	Number of loan accounts with the bank
NO_TDA_ACCT	Number of investment accounts with the bank
NO_ZFN_ACCT	Number of credit card accounts with the bank
NO_BANK_SERV	Number of bank services
NO_POST_ADDR	Number of postal addresses
NO_RES_ADDR	Number of residential addresses
INCOME_AMOUNT	Income amount given by customer
INCOME_ESTIMATE	Bank's estimate of customer's income
Categorical variables	
ACCT_LINK_Y	Indicator showing whether or not account is linked
CNTRY_NATNLITY_ZA	Whether or not customer is South African national
CUST_OCPTN_CDE	Classifies a customer according to a specific occupation category (27 levels)
CUST_SEX_CDE	Customer's gender (2 levels)
DEBT_COUNSEL_IND	Indicator showing whether or not customer is in debt counselling
HIGH_EDU_LVL	Customer's highest level of education (9 levels)
JNT_ACCT_IND	Joint account indicator
MRTL_STAT_CDE	Customer's relationship status (5 levels)
PROP_OWNR_IND	Whether or not customer owns, rents or leases property
SAL_IND	Whether or not customer's salary is deposited with this bank

Table 4.3 provides a summary of the continuous variables in the customer information dataset. From this table, it can be noted that in most cases the variables have a right-skewed distribution. The population used for this study seem to be predominantly middle aged and earns income that is characteristic of the middle class in South Africa.

Table 4.3 A summary of the continous variables in the customer information dataset

Attribute	Min	Q1	Median	Mean	Q3	Max
Age	0	33	41	42.55	51	95
Number of children	0	0	0	1.001	2	99
Total number of products	0	2	3	2.617	3	7
Total number of sub products	0	1	2	2.273	3	18
Number of DDA accounts	0	1	2	1.745	2	26
Number of ILP accounts	0	0	0	0.2509	0	8
Number of TDA accounts	0	0	0	0.659	1	59
Number of ZFN accounts	0	1	1	1.532	2	15
Number of bank services	0	0	1	1.15	2	11
Number of postal addresses	0	1	1	1.298	1	14
Number of residential addresses	1	1	1	1.891	2	20
Income amount	0	231800	360000	24860000	492000	8.89E+11
Income estimate	0	274109	399348	533844	570764	586251684

Further information about the population used in this study is provided in Figure 4.2, Figure 4.3 and Figure 4.4. From these figures it can be noted that the majority of the population are male, South African and own residential property. Furthermore, it can be noted from Figure 4.2 (d) that in majority of the cases the customers' salaries do not come into this particular bank as per the bank's data. Unfortunately, the variable used to record the customer's occupation does not provide many insights as the majority of the customers are assigned to the category "other", as can be seen from Figure 4.3. Surprisingly the highest qualification held by the majority of the customers is Grade 12, which means that they have completed high school education and have not pursued tertiary education.

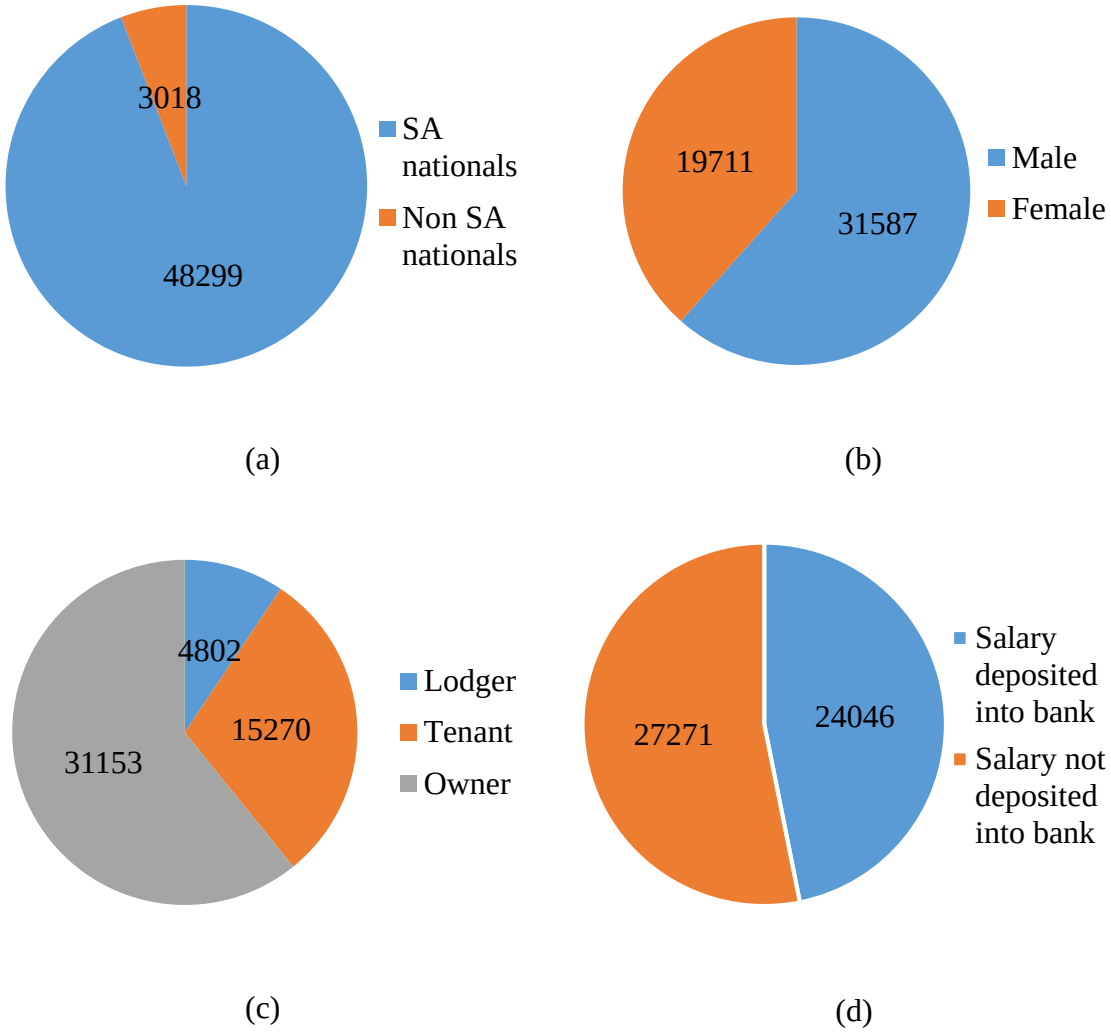


Figure 4.2. Various information about the population used in this study.

(a) Nationality of the population. (b) Gender. (c) Residential status in terms of housing. (d) Whether or not salary is deposited into the bank.

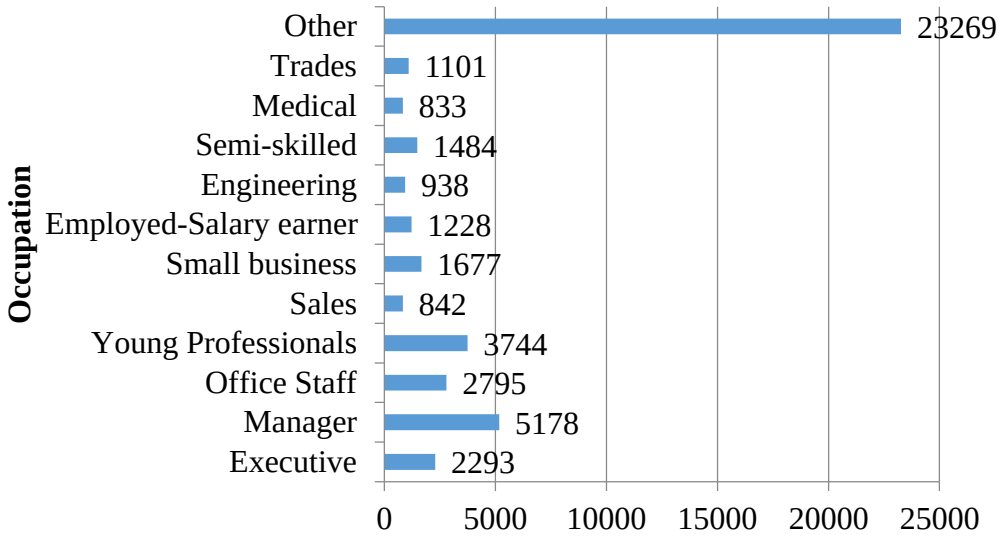


Figure 4.3. The types of occupation categories predominantly found in the population.

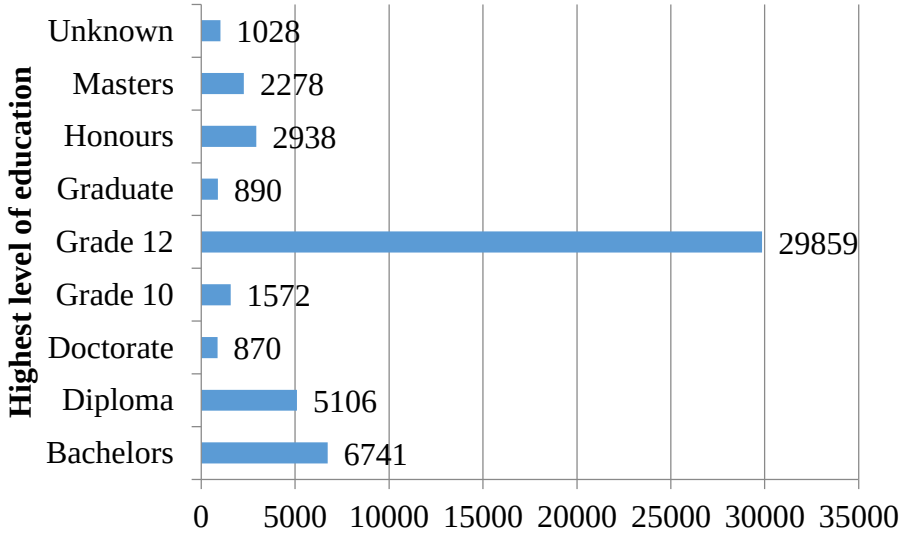


Figure 4.4. Highest level of qualification attained by customers in the population.

4.4 OVERVIEW OF METHODOLOGY

This study consists out of two parts, the first part being that of finding a way to accurately forecast the total daily bank balance of the portfolio being investigated. In doing so, the study looks at whether a hybrid modelling strategy combining customer segmentation with ARIMA models can outperform a single ARIMA model. The second part focuses on

finding a way to see if customers can be scored or classified into one of the segments identified from the segmentation process using the available customer information. This subsection provides an overview of the methodology used to carry out these two parts of the study.

The first part of the study was carried out as follows:

1. The initial step in the process was to extract features for carrying out the segmentation. That is to identify features for identifying similar balance behaviour in the accounts to form the segments. These features were obtained from the time series data from 2013-06-01 to 2016-06-30, thus not using the out of sample period (2016-07-01 to 2017-06-30).
2. The features obtained in step 1 were then used by the k-means algorithm to form the clusters or segments. After which ARIMA models were built for each of the clusters.
3. After fitting the ARIMA models, forecasts were obtained for each of the clusters. The results were then added together to obtain the forecasts for the total daily bank balance at portfolio level.
4. As k-means requires the number of clusters as an input, steps 2-3 was repeated with different number of clusters to obtain solutions for the hybrid modelling approach for different clustering solutions.
5. The solutions obtained in step 4 was then compared with that of the single model to determine whether the hybrid modelling approach outperforms a single model. In doing so, in order to ensure the results are not by chance different validation periods were used to fit the ARIMA models i.e. to determine the hyper parameters of the ARIMA models. These validation periods were chosen to be one year before the out of sample period (2015-07-01 to 2016-06-30), 180 days before the out of sample period, 545 days before the out of sample period and two years (730 days) before the out of sample period.
6. The difference between the hybrid solutions and the single model were compared across four different test periods within the out of sample period. These test periods were 30, 90, 180 and 365 day forecasting periods.

7. Two-sample t-tests were used to determine whether the results are statistically significant.

The second part of the study, which is the classification part was carried out using the clustering solution that performed the best for the one year validation period and the one year test period. In this case the clusters from the clustering solution become the classes in the classification problem. The second part was carried out as follows:

1. In order to determine which of the features from the customer information dataset best distinguishes customers within the different clusters, stepwise feature selection was carried out using `PROC STEPDISC` in SAS.
2. Thereafter two different classifiers were used to see which classifier would perform best for this classification problem. The one being a linear, parametric method in the form of LDA and the other being a non-parametric classifier in the form of random forest.
3. As there is no test set or out of sample set, the performance was assessed using 10-fold cross validation.

The two parts of the study detailed above are presented in the form of two flow diagrams in Figure 4.5 and Figure 4.6 respectively.

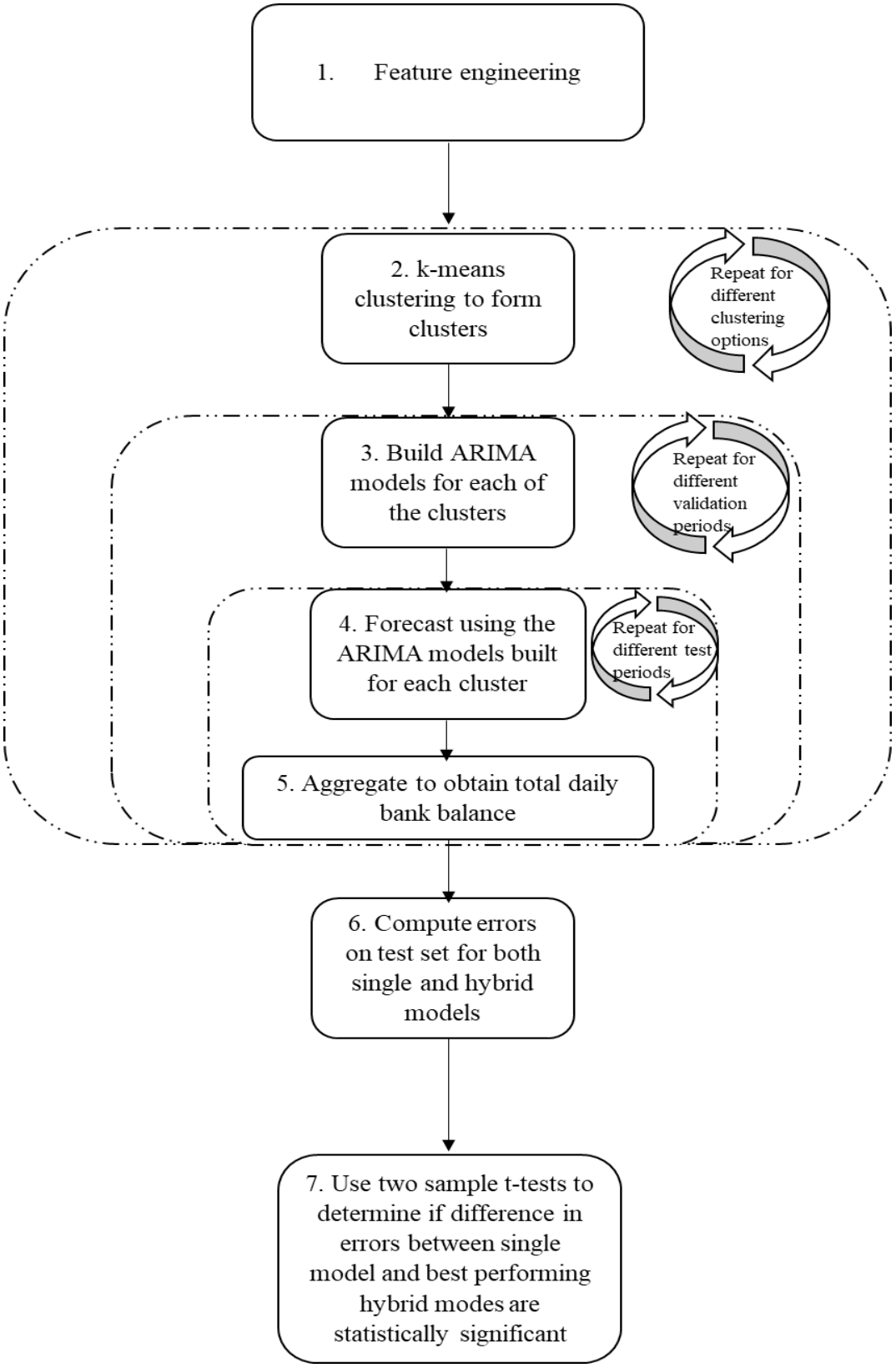


Figure 4.5. A flow diagram explaining the first part of this study.

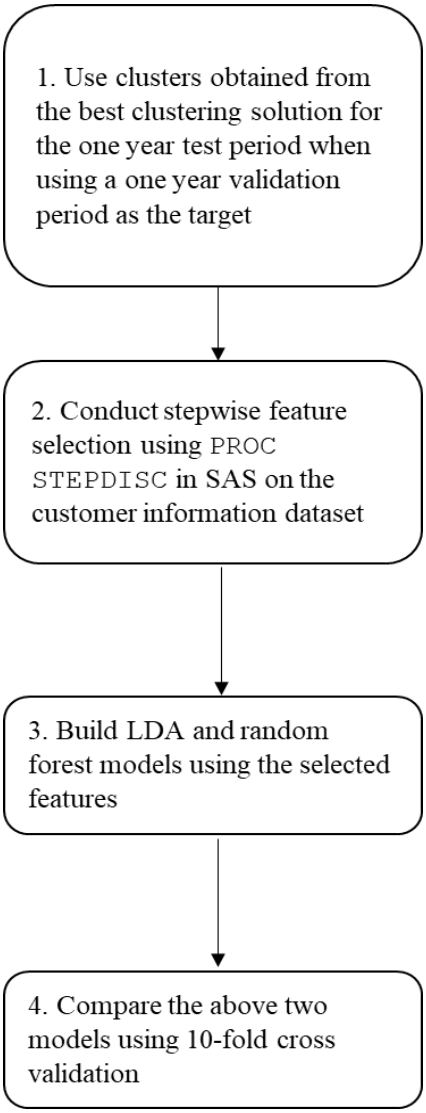


Figure 4.6. A flow diagram explaining the second part of this study.

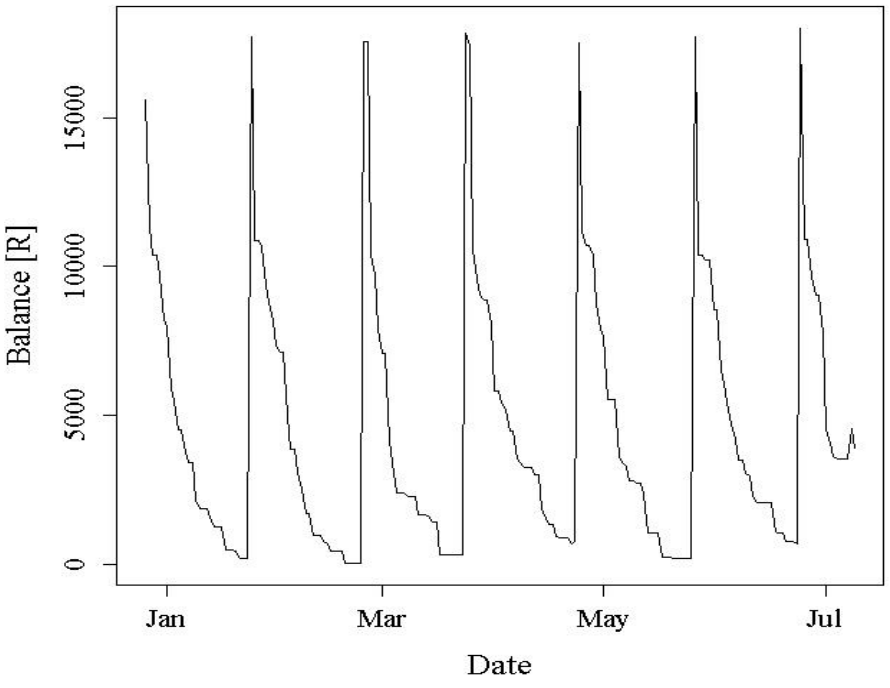
4.5 FEATURE ENGINEERING FROM TIME-SERIES DATA AND CLUSTERING

The goal in using a hybrid model which combines clustering with ARIMA models was to group together customers with similar balance utilisation and accumulation patterns. Having this in mind, the features used for segmentation had to be able to capture common balance patterns. The main idea behind this approach was to find groups of customers who show patterns which would be associated with salaried individuals and to separate these

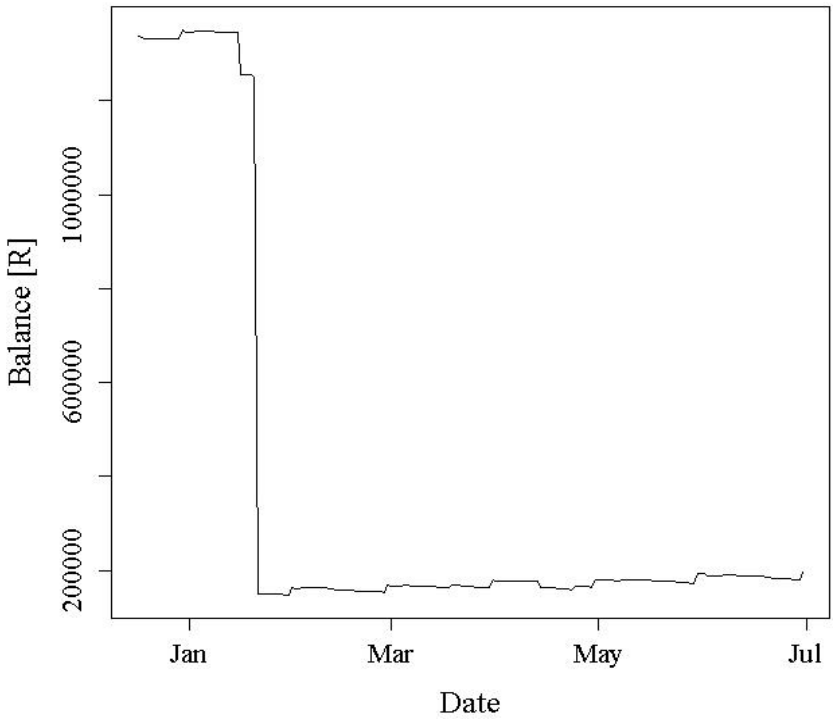
customers from ones who earn their income through other means e.g. small business owners, farmers etc. The latter types of customers would exhibit variable balance patterns with no clear consistent intra month patterns which could introduce some noise into the total daily bank balance forecasts on portfolio level. Examples of a salaried individuals balance pattern and a non-salaried individual's balance pattern can be found in Figure 4.7.

A salaried individual should exhibit a balance pattern where his/her balance reaches close to its maximum value on the day that the salary gets paid into his/her account. Subsequently most people will opt to pay various debit orders for usual expenses such as rent, credit card payments, telephone/utility bills etc. in the days close to when the salary comes in. This will deplete the balance to at least half its maximum value or less, until it reaches close to the minimum a few days before the salary is to be paid in again. This is apparent from Figure 4.7. Furthermore, if someone has consistent spending patterns the variability between the balances on certain dates over various periods e.g. months, should remain low. As is the case, the mean value of the balance and variance of the balance are good measures to capture these patterns.

In South Africa, people who earn monthly salaries are paid either on the 25th, 15th or end of the month. The balance of these individuals should reach close to a minimum balance on the 20th, 10th and 26th respectively. Although with dates of the salaries, the salary could come in one or two days late due to the weekend etc. Furthermore the end of the month changes according to the month, thus the 1st is a better representative of this date. To be clear, the dates chosen were the 1st, 6th, 10th, 16th, 20th and 26th, these dates should cover all afore-mentioned salaried patterns. Therefore, these dates were used as points to compute the mean and variance of the balance across the period 2013-06-01 to 2016-06-30. Furthermore, means and variances of the balance for different days of the week, for each of the months and for each of the years were also computed to pick up any another form of balance utilisation patterns.



(a)



(b)

Figure 4.7. Different types of balance patterns exhbitied by customers.

(a) Balance pattern of a salaried person. (b) Balance pattern of a non-salaried person.

As the goal of this exercise was to pick up balance patterns and since this should not be influenced by the actual amounts in the balances, the balances for each account were normalised between 0 and 1. This was done to ensure that the actual amounts in the balances would not influence the means and variances of the balances. For example, a person earning R 100000 a month and a person earning R 10000 a month might have similar balance utilisation patterns but because the amount is different they will have different means and variances and will not be grouped together. The one earning R 100000 a month will have a higher mean and variance amount in comparison to the one earning R 10000. Figure 4.8 provides an illustration of the way in which normalisation was used to tackle this issue, it provides a normalised version of the balance pattern found in Figure 4.7 (a).

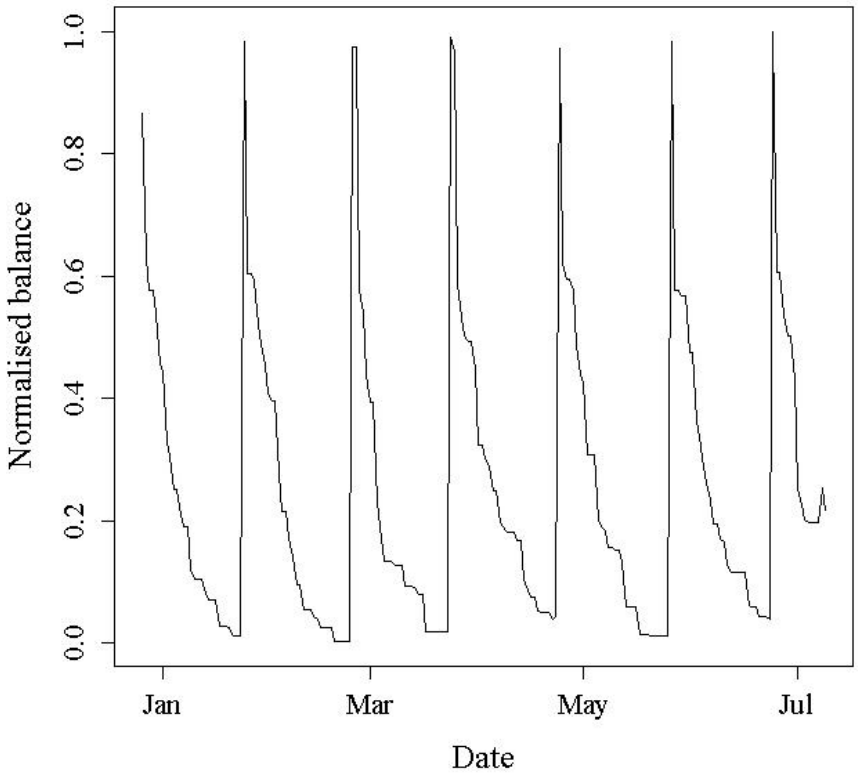


Figure 4.8. Normalised version of a salaried person's bank balance time series.

A total of 54 features were generated in this manner from the time series data i.e. by using the means and variances of different time periods of the normalised bank balance time series. However as noted in Section 3.2.5, clustering high dimensional data creates

problems due to the curse of dimensionality. As this is the case, PCA was used to extract the components of this feature set that contained the largest amount of variation. The `prcomp` function in R was used to apply PCA to the data set.

A scree plot was used to decide how many principal components were required, this plot is shown in Figure 4.9. From Figure 4.9 it can be seen that the first two principal components capture a significant amount of the variance, together they capture 80% of the variance in the data set as can be seen in Figure 4.10. Furthermore it can be seen from Figure 4.9 that after the second principal component the remaining components do not capture a noticeable amount of variation. Therefore the first two principal components were used for the clustering.

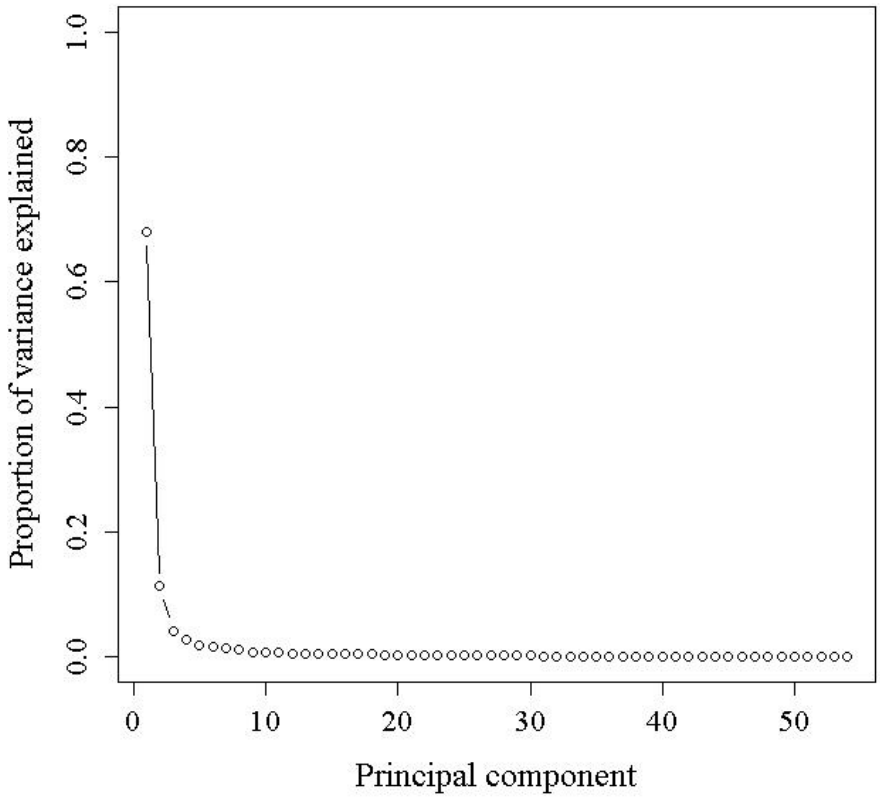


Figure 4.9. A scree plot depicting the proportion of variance explained by each principal component.

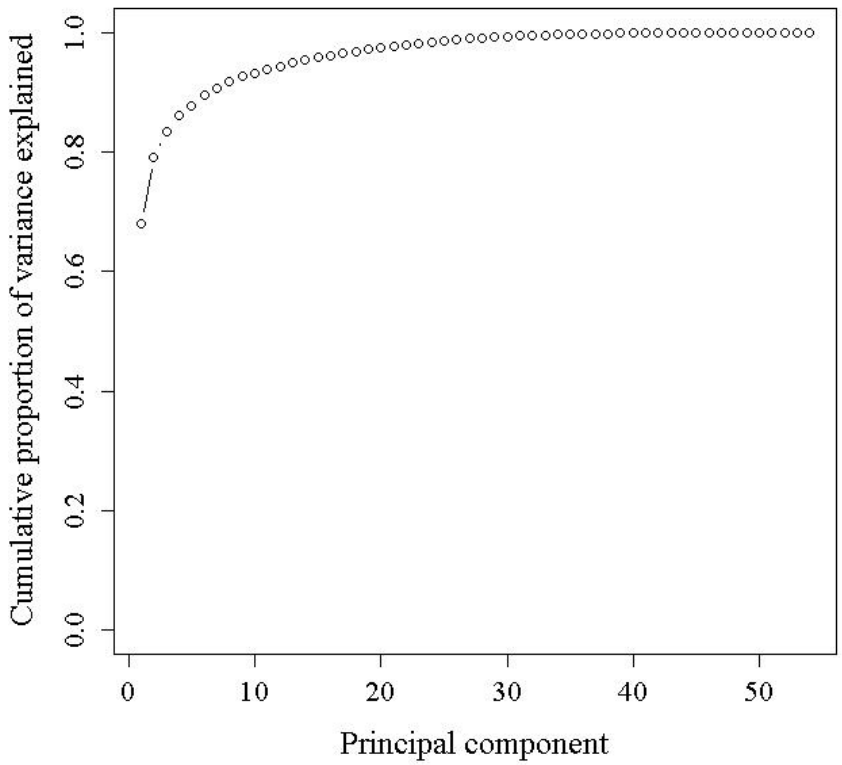


Figure 4.10. The cumulative proportion of variance explained by the principal components.

After obtaining the first two principal components, the `kmeans` function in R was used to obtain the clusters by means of the k-means algorithm. As mentioned in Section 3.2.3, because k-means is an iterative clustering algorithm and finds local optimums, it is necessary to have multiple random assignments for the initial cluster assignments. In this case 30 random initial assignments were used. The optimal number of clusters were determined by using the forecasting performance as a measure as mentioned in Section 3.2.4.

However, one can note that using the first two principal components forms better clusters than using the entire extracted feature set by looking at the silhouette coefficient for different number of clusters as shown in Figure 4.11. The silhouette coefficient ranges from -1 to 1 with values closer to 1 indicating a good clustering solution. In either case of with or without PCA, the silhouette coefficient suggests that two clusters represents the optimal number of clusters in the dataset.

The main reason for applying PCA was to reduce the dimensionality of the clustering set, as clustering algorithms are affected by the curse of dimensionality. In the case of the clustering set, as this was made of means and variances of various dates, months, days of the week etc. there were a large number of features (54 in total). As can be seen from Figure 4.11, the results of the silhouette coefficient after using PCA was much better than without PCA. Other advantages include the fact that since it reduces the dimensions of the set, it becomes more computationally efficient. Lastly, reducing the dimensions makes it easier to visually represent the clusters, even though this particular reason was not used in this study.

It must also be noted that even with its many drawbacks, it can be seen from the results in Figure 4.11 that the k-means algorithm still managed to consistently get silhouette coefficient values greater than 0.4 when combined with PCA, which means that it did fairly well in this particular study.

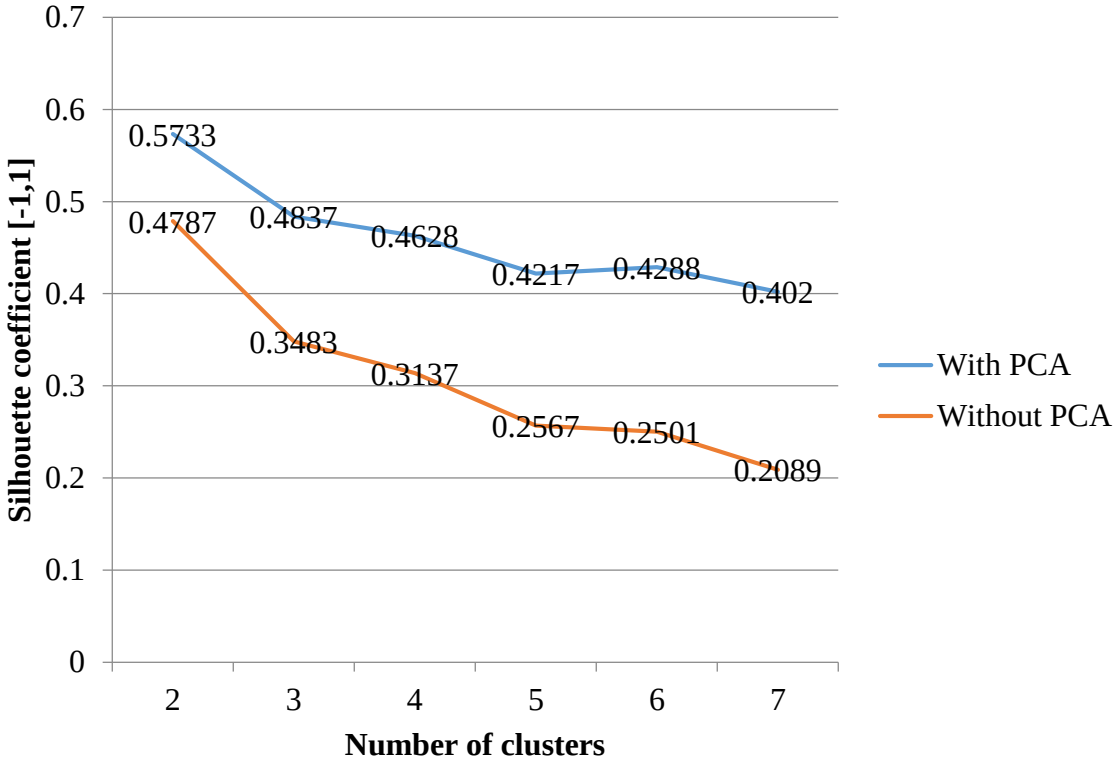


Figure 4.11. Silhouette coefficient for different number of clusters.

4.6 FEATURES SELECTION FOR CLASSIFICATION

The features from the customer information dataset that were used for the classification part were chosen using `PROC STEPDISC` in SAS, as was mentioned in Section 4.4. The `STEPDISC` procedure makes use of stepwise discriminant analysis to choose a subset of the quantitative variables in the modelling set that offers the best discrimination among the classes, which in this case is the cluster or segment to which the customer should belong to. The procedure relies on the assumption that the set of variables/features that make up each class is multivariate normal with a common covariance matrix (SAS Institute Inc., 2008). This is a drawback in that the dummy variables used in the classification dataset will not hold up to these assumptions. In terms of the numerical variables, the variables have been scaled to have a mean of 0 and standard deviation of 1 (mean normalisation), which will help to satisfy the aforementioned assumptions in terms of these variables.

According to (SAS Institute Inc., 2008), the procedure choses variables "to enter or leave the model according to the significance of an F test from an analysis of covariance, where the variables already chosen act as covariates and the variable under consideration is the dependent variable". This F test investigates the hypothesis that adding or removing a variable may improve the discriminatory ability of the model (SAS Institute Inc., 2008).

In the case of this study the `STEPDISC` procedure was used with backward elimination. Backward elimination starts with all the variables in the model except for ones that are linearly dependent on other variables. At each step, the variable that contributes least to the discriminatory power of the model is removed. This is determined by looking at the F-value and the significance level that is set for the F test. The backward elimination continues until none of the variables meet the removal criteria i.e. until all the variables have statistically significant P-values. In using the `STEPDISC` procedure SAS recommends the use of a moderate significance level, in the range of 10 to 25 percent (SAS Institute Inc., 2008). This study uses a significance level of 10 percent.

4.7 ASSESSING MODEL ACCURACY

This section looks at the processes followed to determine whether the proposed methodology is effective. The section covers aspects concerning how the hyper parameters for the ARIMA models were chosen, the type of error metrics used to assess forecasting accuracy as well as how classification accuracy was handled for the second part of the study.

4.7.1 Time-series forecasting

As mentioned in Section 4.4 the best way to determine the model order for an ARIMA model is to use the accuracy on an out of sample set or validation set. This approach was followed in this study. In doing so, the training phase in Figure 4.1 was further divided into a training and validation period. The hyper parameters of the ARIMA model that performed the best on the validation period were then used to fit the model on the whole training phase and then used to forecast over the testing phase. As mentioned in Section 4.4, different validation periods were used to ensure that the conclusions provided regarding the proposed approach were robust and were not dependent on some inherent property of the time series.

It is clear to see that the daily time series data used in this study, found in Figure 4.1, exhibits an annual pattern in the daily data i.e. what happened today has some similarities with what happened exactly a year ago. Although this is not exact, it is understandable that this effect exists as the days on which people get paid, the days of certain holidays and days on which debit orders etc. go off accounts remains the same over each year. Therefore the seasonal period of differencing is $m = 365.25$ days (Hyndman, 2017), in assuming this the effect of leap years is ignored as this cannot be accounted for practically. The seasonal ARIMA model allows one to seamlessly incorporate this aspect (the seasonality present in the time series) into the model. Certain time series models like exponential smoothing, weighted average models would not be able to replicate or forecast these patterns efficiently. Meanwhile to produce similar results with more complex machine

learning models like RFs, NN and SVR would require much more advanced feature engineering. The ARIMA model was selected for this study due to its ability to generate good results with regards to this problem without much complexity.

The `arima` function from the fpp package that was used for this study has a limitation in that it only allows a seasonal period up to $m = 350$ (Hyndman, 2017). This manifests in some issues in that one can not specify different options for the seasonal AR and MA parts of the ARIMA model as the package does not allow for it when using long seasonal periods. In practice it was found that the function did support seasonal differencing of order $D = 1$ with number of periods per season m set to 365.25. All the hyper parameters that were used in the study had seasonal differencing of order 1 with $m = 365.25$ as this pattern is prevalent in the daily time series data and it did not make sense to ignore it. The hyper parameters for the ARIMA model that were tested in this study are shown in Table 4.4. These hyper parameters were tested across the different validation periods

Table 4.4 The different hyper-parameters that were utilised in fitting the ARIMA models.

Model order	**p**	**d**	**q**	**P**	**D**	**Q**
1	0	0	0	0	1	0
2	0	1	0	0	1	0
3	1	1	1	0	1	0
4	0	1	1	0	1	0
5	1	1	0	0	1	0
6	2	1	0	0	1	0
7	3	1	0	0	1	0
8	2	1	1	0	1	0
9	1	1	2	0	1	0
10	2	1	2	0	1	0

11	2	2	2	0	1	0
12	1	2	1	0	1	0

Two different error metrics were used to assess the accuracy of the forecasts, namely the mean absolute percentage error (MAPE) and the root mean squared error (RMSE), and these are given below (Hyndman & Athanasopoulos, 2014):

$$MAPE = mean\left(\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100\right), \tag{4.1}$$

$$RMSE = \sqrt{mean((y_i - \hat{y}_i)^2)}, \tag{4.2}$$

where y_i is the ith observation and $\hat{y}_i$ denotes a forecast of y_i. The MAPE in (4.1) provides a scale-independent error metric and allows for easier comparisons of the errors within the different clusters, while the RMSE in (4.2) provides a measure of the error in the same scale as the data (Hyndman & Athanasopoulos, 2014).In the case of this study, the units of the RMSE are Rands (R).

4.7.2 Classification

The absence of an out of sample set for the classification part of the study meant that 10-fold cross validation was used to assess classification accuracy as it provides an unbiased estimate of the test set error. In terms of the classifiers, the `lda` function from the `MASS` package was used to fit the LDA model and the `randomforest` function from the `randomForest` package was used to fit the random forest model. In both cases the default settings for both classifiers were used. In terms of the random forest classifier, the default number of trees to grow was 500, the number of variables randomly sampled at each split was set to be $\sqrt{p}$ with p being the number of variables, and each tree was fully grown.

4.8 CHAPTER SUMMARY

This chapter discussed the methodology followed in this study and gave a summary of the data used in the study as well as providing details of the systems utilised to conduct the study. The chapter covered aspects such as how the features were engineered from the time series data for the purposes of clustering or segmentation, how PCA was used to aid in this process, the feature selection for classification and the matter of assessing the performance of the various approaches that were suggested in the study.

CHAPTER 5 RESULTS AND DISCUSSION

5.1 CHAPTER OBJECTIVE

This chapters presents the results of the study. It follows the proposed methodology from the previous chapter. The chapter starts with the results of the forecasting and provides a comparison between the results of the hybrid model and a single ARIMA model. After which the results of the feature selection and classification results are discussed. Finally the chapter concludes with a discussion of the results.

5.2 FORECASTING RESULTS OF THE SINGLE MODEL VS. HYRBID MODEL

Following the methodology in Section 4.4, after obtaining the clusters the next step in the proposed approach was to fit the ARIMA models for each of the clustering solutions as well as the single model. In doing so, various validation periods were also used. The RMSE on the validation sets were used to determine which set of hyper parameters to choose, the one with the lowest RMSE being chosen. Tables 5.1 to 5.4 provide the hyper parameters obtained for the ARIMA models for the 180, 365, 545 and 730 day validation periods respectively.

Table 5.1 The optimal hyper-parameters obtained for single and hybrid models for 180 day validation period.

Cluster number/Type of model	**p**	**d**	**q**	**P**	**D**	**Q**
Single model						
Single model	1	1	1	0	1	0
2 cluster hybrid model						
1	0	0	0	0	1	0
2	2	2	2	0	1	0
3 cluster hybrid model						
1	2	2	2	0	1	0
2	0	0	0	0	1	0
3	2	1	0	0	1	0
4 cluster hybrid model						
1	1	2	1	0	1	0
2	0	0	0	0	1	0
3	0	0	0	0	1	0
4	1	1	1	0	1	0
5 cluster hybrid model						
1	0	1	0	0	1	0
2	1	1	2	0	1	0
3	0	0	0	0	1	0
4	0	0	0	0	1	0
5	1	1	2	0	1	0
6 cluster hybrid model						
1	0	0	0	0	1	0
2	2	2	2	0	1	0
3	1	1	2	0	1	0

4	0	1	0	0	1	0
5	0	1	0	0	1	0
6	2	1	1	0	1	0
7 cluster hybrid model						
1	0	1	0	0	1	0
2	0	0	0	0	1	0
3	0	1	0	0	1	0
4	2	1	2	0	1	0
5	1	1	2	0	1	0
6	1	1	1	0	1	0
7	0	0	0	0	1	0

Table 5.2 The optimal hyper-parameters obtained for single and hybrid models for 365 day validation period.

Cluster number/Type of model	**p**	**d**	**q**	**P**	**D**	**Q**
Single model						
Single model	0	1	0	0	1	0
2 cluster hybrid model						
1	0	0	0	0	1	0
2	1	1	1	0	1	0
3 cluster hybrid model						
1	1	1	1	0	1	0
2	1	1	2	0	1	0
3	2	1	2	0	1	0
4 cluster hybrid model						
1	1	1	1	0	1	0
2	2	1	2	0	1	0

3	0	0	0	0	1	0
4	1	2	1	0	1	0
5 cluster hybrid model						
1	2	1	2	0	1	0
2	1	1	1	0	1	0
3	2	1	0	0	1	0
4	0	0	0	0	1	0
5	1	2	1	0	1	0
6 cluster hybrid model						
1	0	0	0	0	1	0
2	2	1	2	0	1	0
3	2	1	2	0	1	0
4	0	1	0	0	1	0
5	2	1	0	0	1	0
6	1	1	1	0	1	0
7 cluster hybrid model						
1	0	1	1	0	1	0
2	1	1	0	0	1	0
3	3	1	0	0	1	0
4	2	1	0	0	1	0
5	0	0	0	0	1	0
6	0	1	1	0	1	0
7	0	0	0	0	1	0

Table 5.3 The optimal hyper-parameters obtained for single and hybrid models for 545 day validation period.

Cluster number/Type of model	**p**	**d**	**q**	**P**	**D**	**Q**
Single model						
Single model	2	1	0	0	1	0
2 cluster hybrid model						
1	2	1	2	0	1	0
2	3	1	0	0	1	0
3 cluster hybrid model						
1	1	2	1	0	1	0
2	1	2	1	0	1	0
3	1	1	0	0	1	0
4 cluster hybrid model						
1	2	1	2	0	1	0
2	2	1	2	0	1	0
3	0	0	0	0	1	0
4	2	2	2	0	1	0
5 cluster hybrid model						
1	0	1	0	0	1	0
2	2	1	2	0	1	0
3	0	1	1	0	1	0
4	1	2	1	0	1	0
5	3	1	0	0	1	0
6 cluster hybrid model						
1	0	0	0	0	1	0
2	1	2	1	0	1	0
3	2	2	2	0	1	0

4	0	1	0	0	1	0
5	2	1	0	0	1	0
6	1	2	1	0	1	0
7 cluster hybrid model						
1	0	1	0	0	1	0
2	0	1	0	0	1	0
3	0	1	0	0	1	0
4	0	1	1	0	1	0
5	1	1	1	0	1	0
6	3	1	0	0	1	0
7	0	1	0	0	1	0

Table 5.4 The optimal hyper-parameters obtained for single and hybrid models for 730 day validation period.

Cluster number/Type of model	**p**	**d**	**q**	**P**	**D**	**Q**
Single model						
Single model	1	1	1	0	1	0
2 cluster hybrid model						
1	0	0	0	0	1	0
2	0	1	1	0	1	0
3 cluster hybrid model						
1	0	1	1	0	1	0
2	0	0	0	0	1	0
3	0	1	1	0	1	0
4 cluster hybrid model						
1	0	1	1	0	1	0
2	1	2	1	0	1	0

3	0	0	0	0	1	0
4	0	1	1	0	1	0
5 cluster hybrid model						
1	0	0	0	0	1	0
2	0	1	0	0	1	0
3	0	0	0	0	1	0
4	0	0	0	0	1	0
5	0	1	1	0	1	0
6 cluster hybrid model						
1	0	0	0	0	1	0
2	3	1	0	0	1	0
3	0	0	0	0	1	0
4	1	1	2	0	1	0
5	0	0	0	0	1	0
6	0	1	1	0	1	0
7 cluster hybrid model						
1	0	1	1	0	1	0
2	0	0	0	0	1	0
3	1	1	0	0	1	0
4	0	1	1	0	1	0
5	0	0	0	0	1	0
6	0	1	1	0	1	0
7	0	0	0	0	1	0

The errors obtained from the forecasts of the testing period using the single model and various hybrid models for the different validation periods over the 30 day test period are provided in Table 5.5. For each of the different validation periods the best performing hybrid model solution is highlighted in yellow for referral purposes for figures to follow.

Table 5.5 Forecasting errors on the test set over 30 day test period.

Type of model	180 day validation period		365 day validation period		545 day validation period		730 day validation period	
	Error metric							
	RMSE [R]	MAPE	RMSE [R]	MAPE	RMSE [R]	MAPE	RMSE [R]	MAPE
Single model	6.43E+07	1.48%	2.60E+08	10.19 %	1.47E+08	5.41%	6.43E+07	1.48%
Hybrid model with 2 clusters	7.53E+07	1.86%	7.21E+07	1.71%	1.02E+08	3.22%	7.13E+07	1.67%
Hybrid model with 3 clusters	1.19E+08	4.09%	6.84E+07	1.53%	2.99E+08	11.16 %	8.26E+07	2.33%
Hybrid model with 4 clusters	1.75E+08	6.27%	2.06E+08	7.42%	9.92E+07	2.92%	9.96E+07	2.91%
Hybrid model with 5 clusters	1.49E+08	5.36%	1.65E+08	5.93%	1.75E+08	6.36%	7.20E+07	1.80%
Hybrid model with 6 clusters	1.56E+08	5.63%	1.56E+08	5.63%	2.44E+08	9.19%	6.94E+07	1.64%
Hybrid model with 7 clusters	1.29E+08	4.54%	1.73E+08	6.34%	1.38E+08	4.90%	1.00E+08	3.25%

Plots of the forecasts for the various validation periods are provided in Figures 5.1 to 5.4 for the 30 day test period. In each case the forecasts obtained from the single model and the ones obtained from the best performing hybrid models, highlighted in Table 5.5, are plotted along with the actual total daily bank balance for the test period.

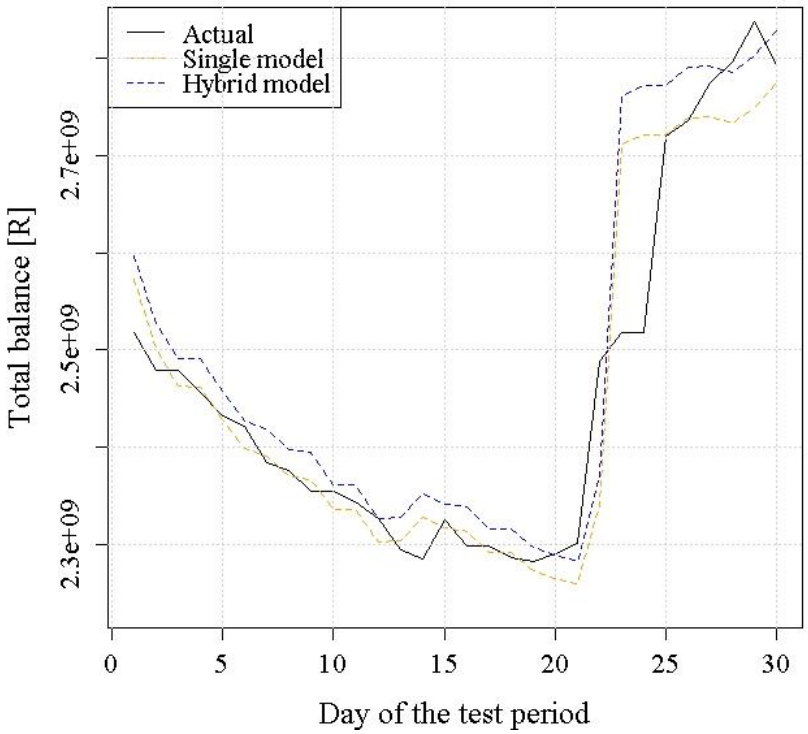


Figure 5.1. Forecasts obtained using the single model and the best performing hybrid model for the 180 day validation period over the 30 day test period.

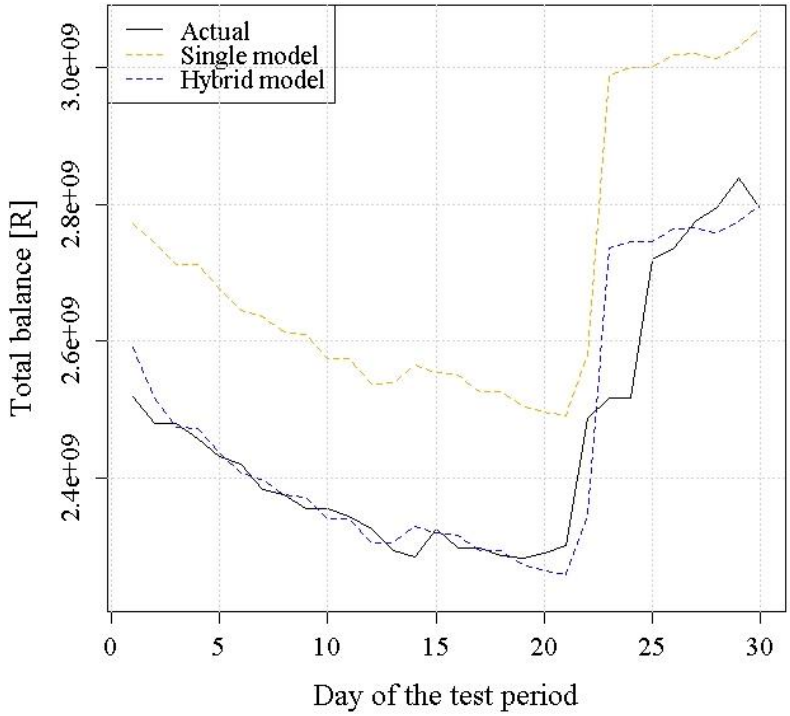


Figure 5.2. Forecasts obtained using the single model and the best performing hybrid model for the 365 day validation period over the 30 day test period.

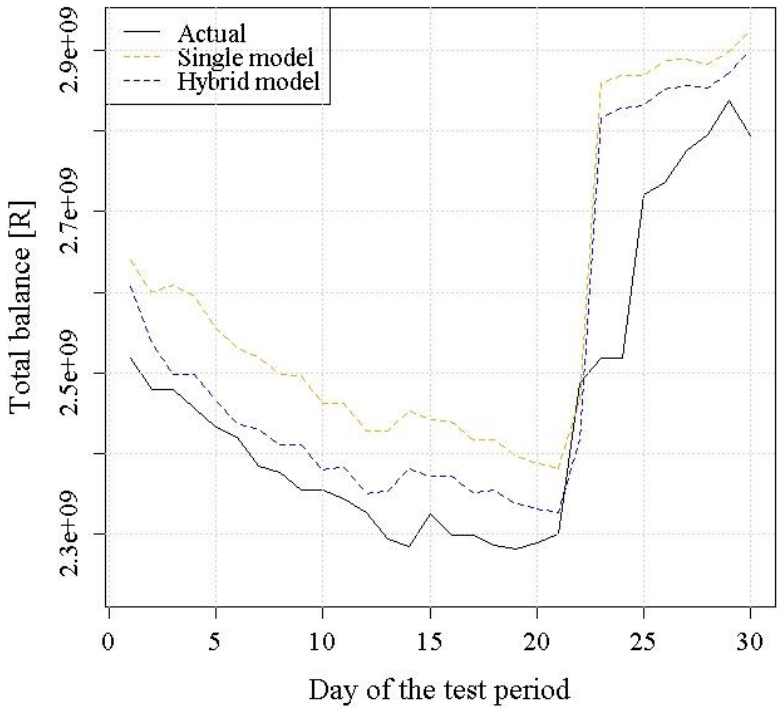


Figure 5.3. Forecasts obtained using the single model and the best performing hybrid model for the 545 day validation period over the 30 day test period.

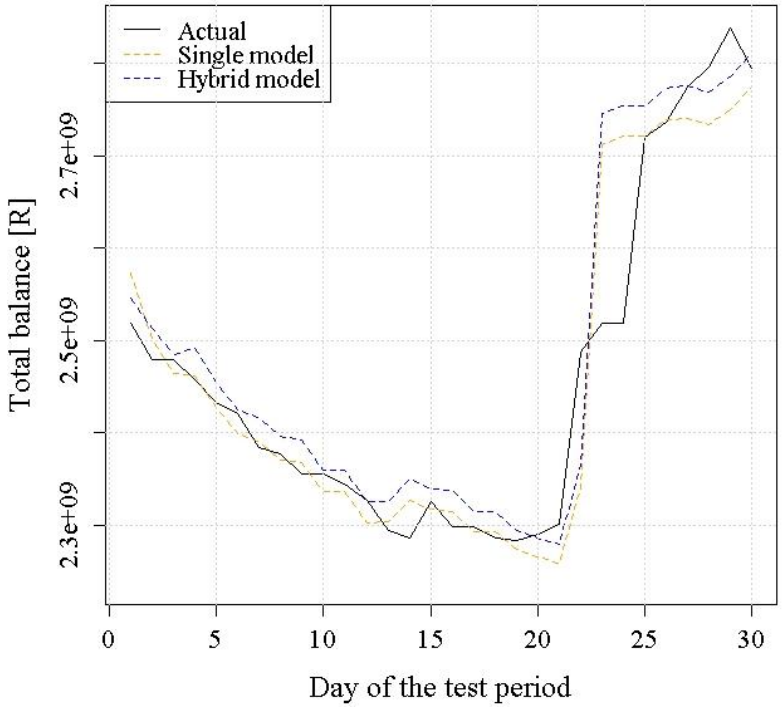


Figure 5.4. Forecasts obtained using the single and the best performing hybrid model for the 730 day validation period over the 30 day test period.

The absolute percentage errors obtained using the single model and the best performing hybrid models for the different validation periods are shown in Figures 5.5 to 5.8 in the case of the 30 day test period. In each case the errors are plotted along with a one standard deviation band around it. This is to visually inspect whether the difference between the errors obtained using the single model and the hybrid model are statistically significant. The x-axis in this case represents the number of days in the testing period.

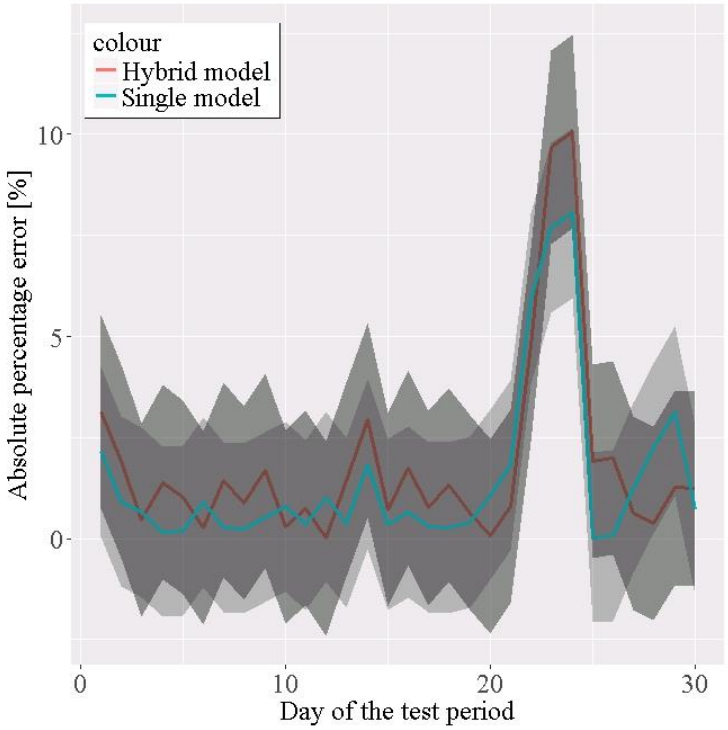


Figure 5.5. Comparison of the errors obtained using a single model and the best performing hybrid model for the 180 day validation period over the 30 day testing period.

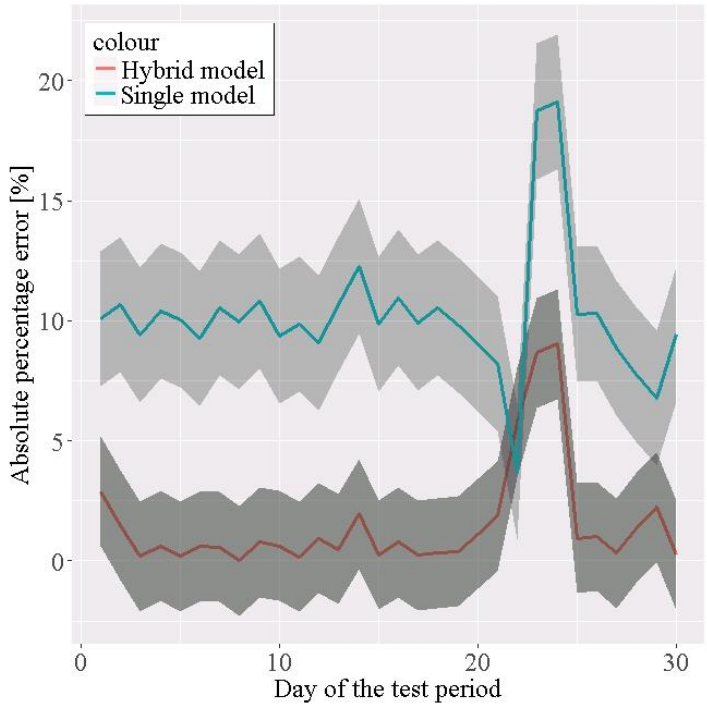


Figure 5.6. Comparison of the errors obtained using a single model and the best performing hybrid model for the 365 day validation period over the 30 day testing period.

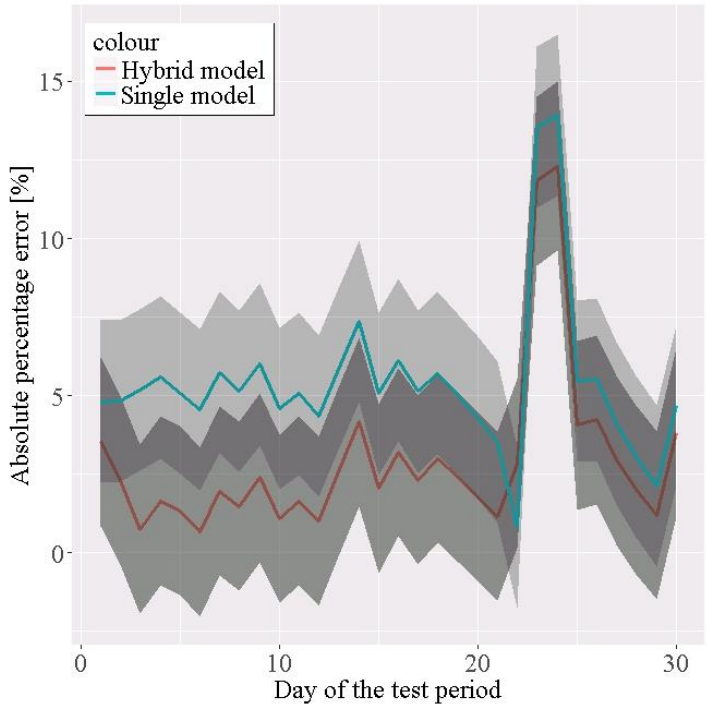


Figure 5.7. Comparison of the errors obtained using a single model and the best performing hybrid model for the 545 day validation period over the 30 day testing period.

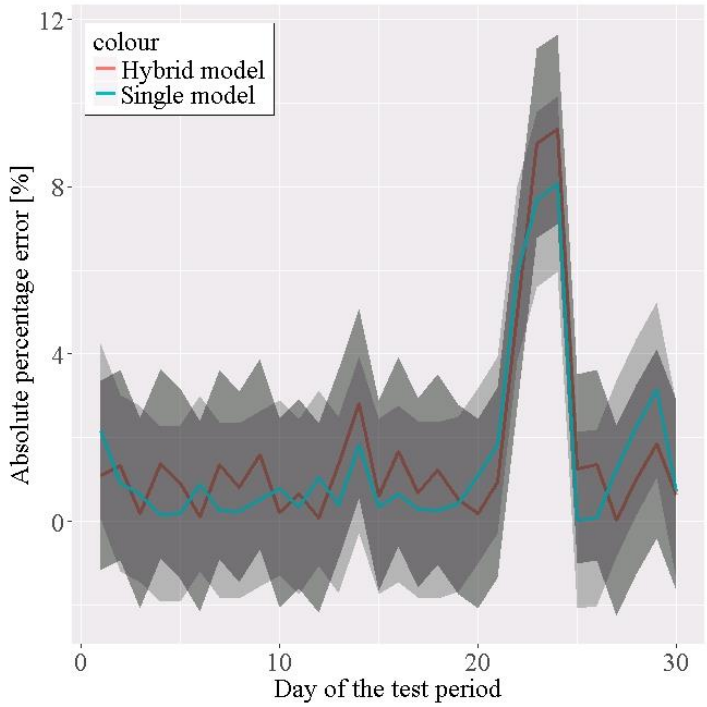


Figure 5.8. Comparison of the errors obtained using a single model and the best performing hybrid model for the 730 day validation period over the 30 day testing period.

Two sample t-tests were also conducted to determine if the difference between the errors obtained using the single model and the best performing hybrid model was statistically significant. The t-tests conducted were one-sided with the null hypothesis being that the true difference in means are equal. In all the t-tests $\mu_1$was the mean of the hybrid model and μ_2 was the mean of the single model. In the cases where the single model has a lower error than the best performing hybrid model, the t-tests had an alternative hypothesis of $\mu_1 - \mu_2 > 0$, and in the cases where the best performing hybrid had a lower error than the single model, the t-tests had an alternative hypothesis of $\mu_1 - \mu_2 < 0$. The results of these t-tests for the 30 day test period are shown in Table 5.6.

In interpreting the t-test P-values to follow, it must be noted that for t-tests in R there is a lower limit on the P-value calculation of 2.22×10^{-16}. The reason for this limit is probably due to double digit precision and to avoid numerical underflow. In terms of interpretation it is a value below which one can be quite confident that the value will be fairly numerically meaningless.

Table 5.6 Results of the two-sample t-tests for the 30 day testing period.

Validation period	Alternative hypothesis	P-value
180 day	$\mu_1 - \mu_2 > 0$	0.2613
365 day	$\mu_1 - \mu_2 < 0$	(<2.2e-16) - Numerically insignificant
545 day	$\mu_1 - \mu_2 < 0$	0.00028
730 day	$\mu_1 - \mu_2 > 0$	0.3885

A similar set of results as above are given for the 90, 180 and 365 day test periods. The forecasting errors for the 90 day test period are shown in Table 5.7. Plots of the forecasts for the various validation periods are provided in Figures 5.9 to 5.12 for the 90 day test period. The absolute percentage errors obtained using the single model and the best performing hybrid models for the different validation periods are shown in Figures 5.13 to 5.16 in the case of the 90 day test period. The results of the t-tests for the 90 day test period are shown in Table 5.8.

Table 5.7 Forecasting errors on the test set over 90 day test period.

Type of model	180 day validation period		365 day validation period		545 day validation period		730 day validation period	
	Error metric							
	RMSE [R]	MAPE	RMSE [R]	MAPE	RMSE [R]	MAPE	RMSE [R]	MAPE
Single model	8.63E+07	2.34%	2.94E+08	11.57 %	1.83E+08	6.87%	8.63E+07	2.34%
Hybrid model with 2 clusters	1.12E+08	3.40%	1.01E+08	2.95%	1.37E+08	4.66%	1.02E+08	2.97%
Hybrid model with 3 clusters	1.58E+08	5.67%	9.43E+07	2.59%	6.12E+08	22.46 %	1.17E+08	3.75%
Hybrid model with 4 clusters	3.30E+08	11.87 %	4.30E+08	15.46 %	2.35E+08	7.94%	2.43E+08	8.20%
Hybrid model with 5 clusters	1.82E+08	6.68%	2.34E+08	8.68%	2.63E+08	9.69%	1.04E+08	3.16%
Hybrid model with 6 clusters	2.02E+08	7.48%	1.60E+08	5.67%	4.15E+08	15.51 %	1.01E+08	2.99%
Hybrid model with 7 clusters	1.64E+08	5.93%	2.14E+08	8.00%	1.74E+08	6.33%	1.36E+08	4.71%

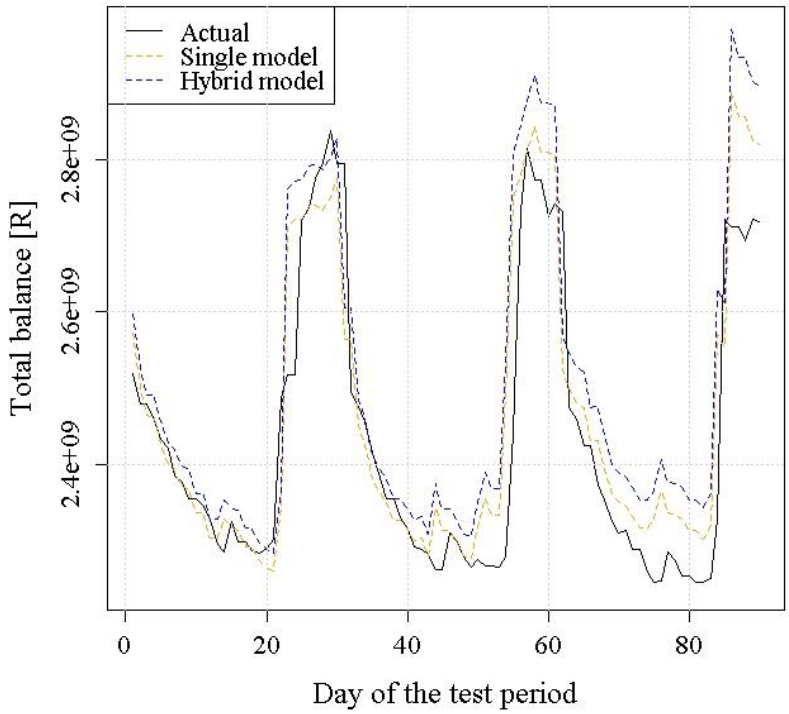


Figure 5.9. Forecasts obtained using the single model and the best performing hybrid model for the 180 day validation period over the 90 day test period.

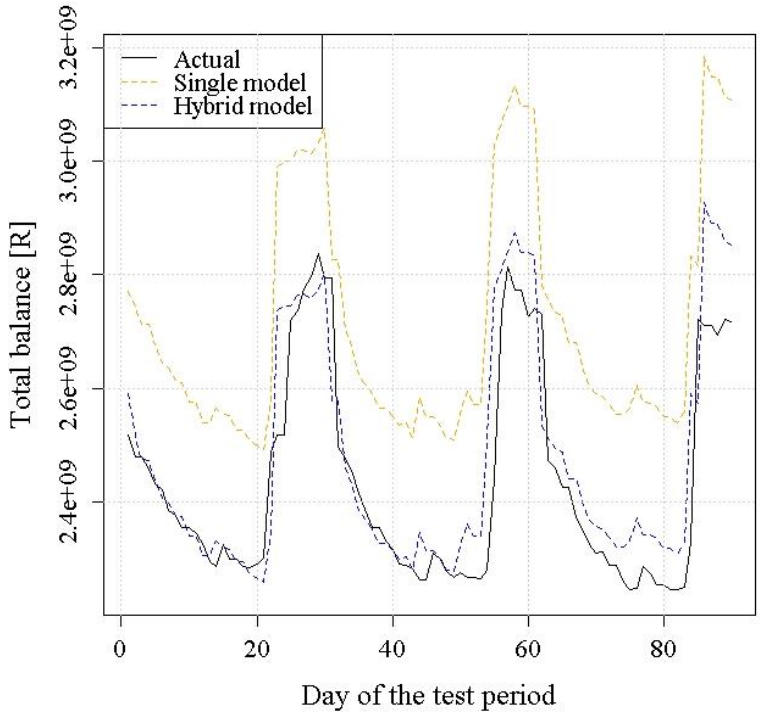


Figure 5.10. Forecasts obtained using the single model and the best performing hybrid model for the 365 day validation period over the 90 day test period.

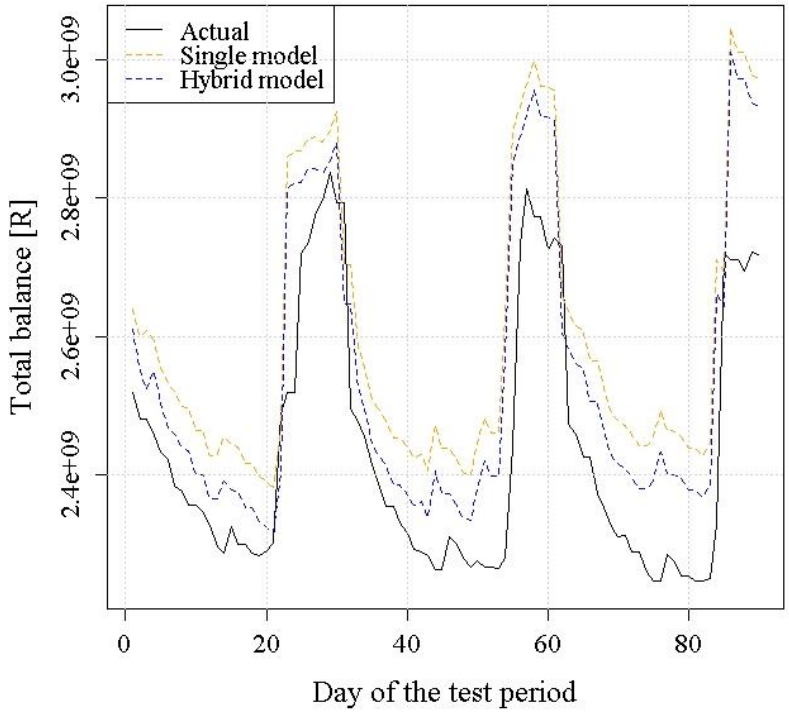


Figure 5.11. Forecasts obtained using the single model and the best performing hybrid model for the 545 day validation period over the 90 day test period.

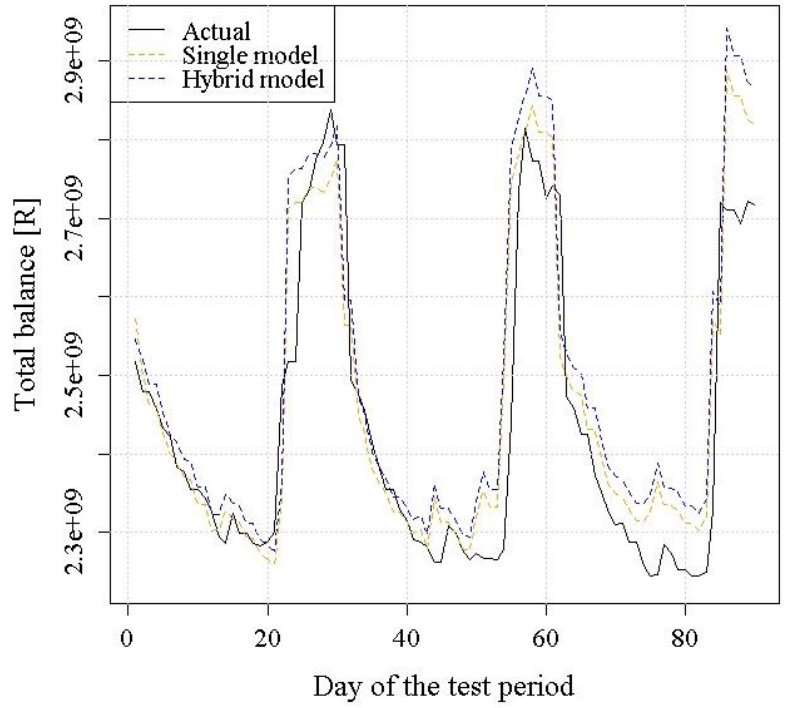


Figure 5.12. Forecasts obtained using the single model and the best performing hybrid model for the 730 day validation period over the 90 day test period.

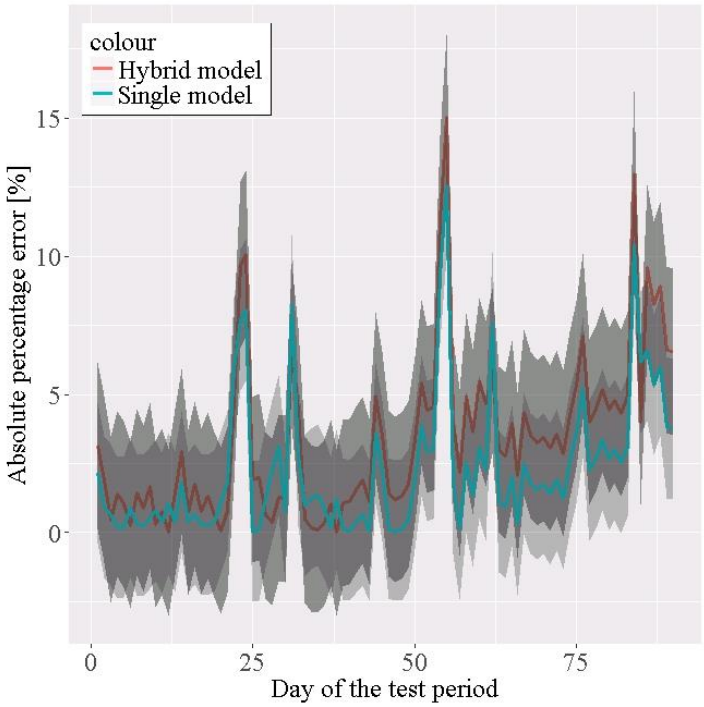


Figure 5.13. Comparison of the errors obtained using a single model and the best performing hybrid model for the 180 day validation period over the 90 day testing period.

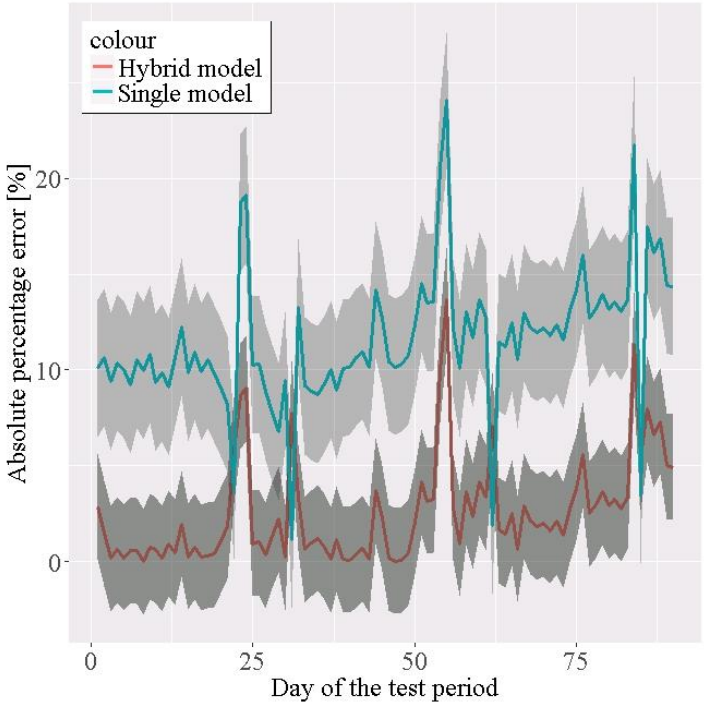


Figure 5.14. Comparison of the errors obtained using a single model and the best performing hybrid model for the 365 day validation period over the 90 day testing period.

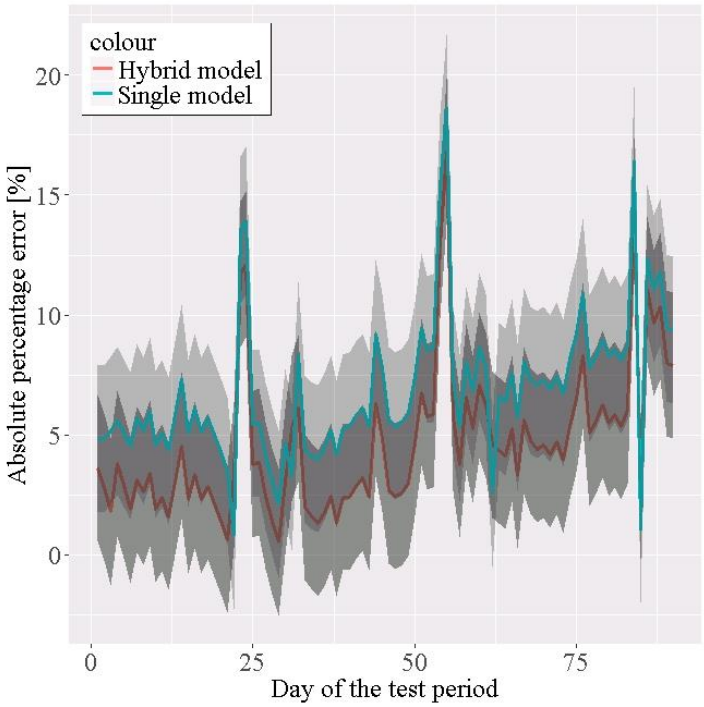


Figure 5.15. Comparison of the errors obtained using a single model and the best performing hybrid model for the 545 day validation period over the 90 day testing period.

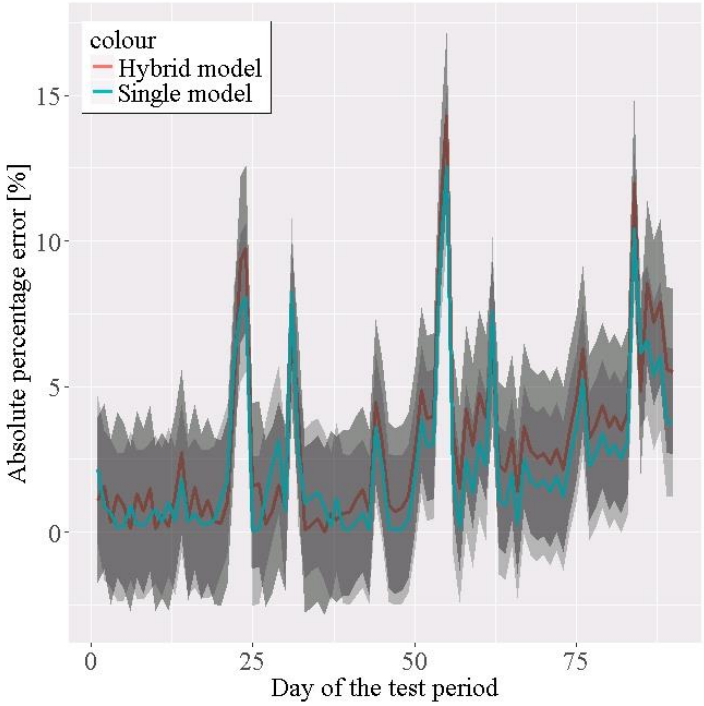


Figure 5.16. Comparison of the errors obtained using a single model and the best performing hybrid model for the 730 day validation period over the 90 day testing period.

Table 5.8 Results of the two-sample t-tests for the 90 day testing period.

Validation period	Alternative hypothesis	P-value
180 day	$\mu_1 - \mu_2 > 0$	0.005831
365 day	$\mu_1 - \mu_2 < 0$	(<2.2e-16) - Numerically insignificant
545 day	$\mu_1 - \mu_2 < 0$	1.34E-06
730 day	$\mu_1 - \mu_2 > 0$	0.06129

The forecasting errors for the 180 day test period are shown in Table 5.9. Plots of the forecasts for the various validation periods are provided in Figures 5.17 to 5.20 for the 180 day test period. The absolute percentage errors obtained using the single model and the best performing hybrid models for the different validation periods are shown in Figures 5.21 to 5.24 in the case of the 180 day test period. The results of the t-tests for the 180 day test period are shown in Table 5.10.

Table 5.9 Forecasting errors on the test set over 180 day test period.

Type of model	180 day validation period		365 day validation period		545 day validation period		730 day validation period	
	Error metric							
	RMSE [R]	MAPE	RMSE [R]	MAPE	RMSE [R]	MAPE	RMSE [R]	MAPE
Single model	1.04E+08	2.83%	3.22E+08	12.39 %	2.08E+08	7.64%	1.04E+08	2.83%
Hybrid model with 2 clusters	1.45E+08	4.56%	1.23E+08	3.65%	1.63E+08	5.51%	1.24E+08	3.69%
Hybrid model with 3 clusters	1.93E+08	6.79%	1.14E+08	3.21%	1.21E+09	41.29 %	1.40E+08	4.53%
Hybrid model with 4 clusters	6.21E+08	21.07 %	8.43E+08	28.62 %	4.73E+08	15.79 %	5.02E+08	16.69 %
Hybrid model with 5 clusters	2.10E+08	7.55%	2.88E+08	10.54 %	3.88E+08	13.97 %	1.26E+08	3.89%
Hybrid model with 6 clusters	2.36E+08	8.63%	1.93E+08	6.83%	7.33E+08	25.66 %	1.22E+08	3.69%
Hybrid model with 7 clusters	1.92E+08	6.80%	2.45E+08	9.01%	2.03E+08	7.24%	1.62E+08	5.52%

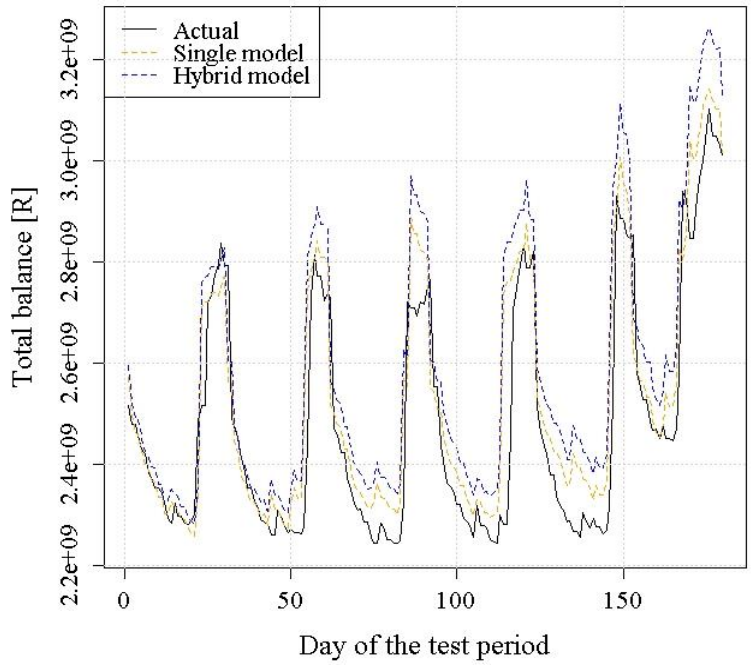


Figure 5.17. Forecasts obtained using the single model and the best performing hybrid model for the 180 day validation period over the 180 day test period.

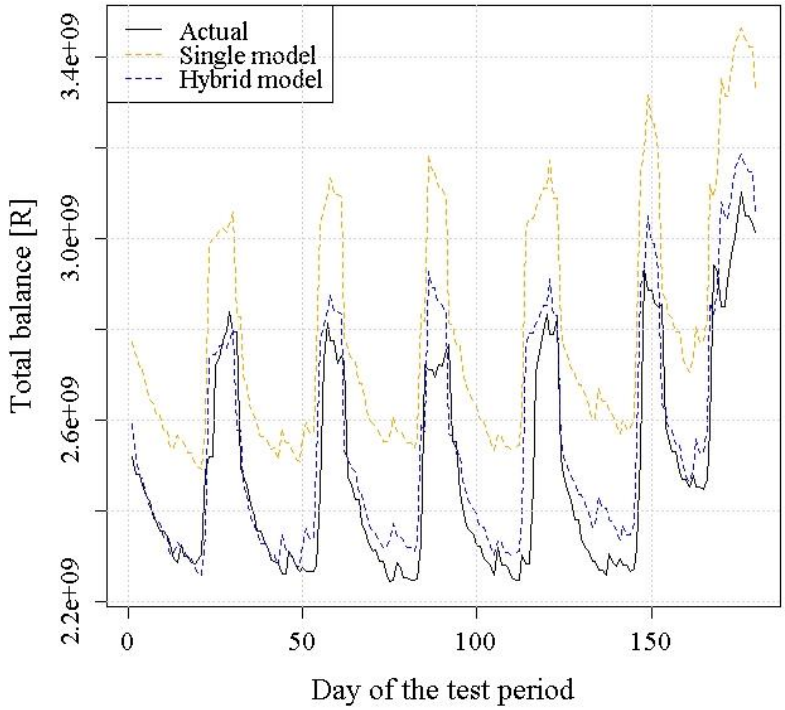


Figure 5.18. Forecasts obtained using the single model and the best performing hybrid model for the 365 day validation period over the 180 day test period.

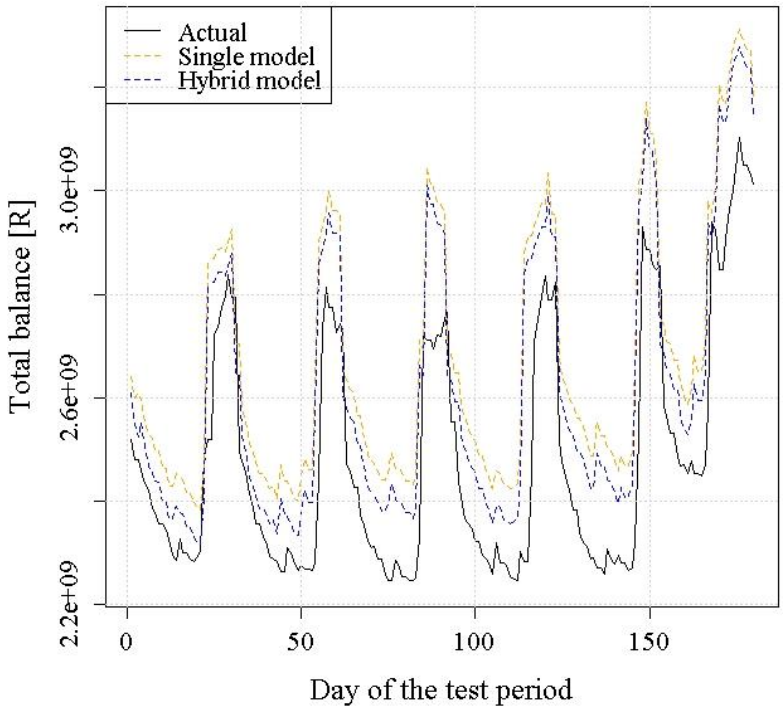


Figure 5.19. Forecasts obtained using the single model and the best performing hybrid model for the 545 day validation period over the 180 day test period.

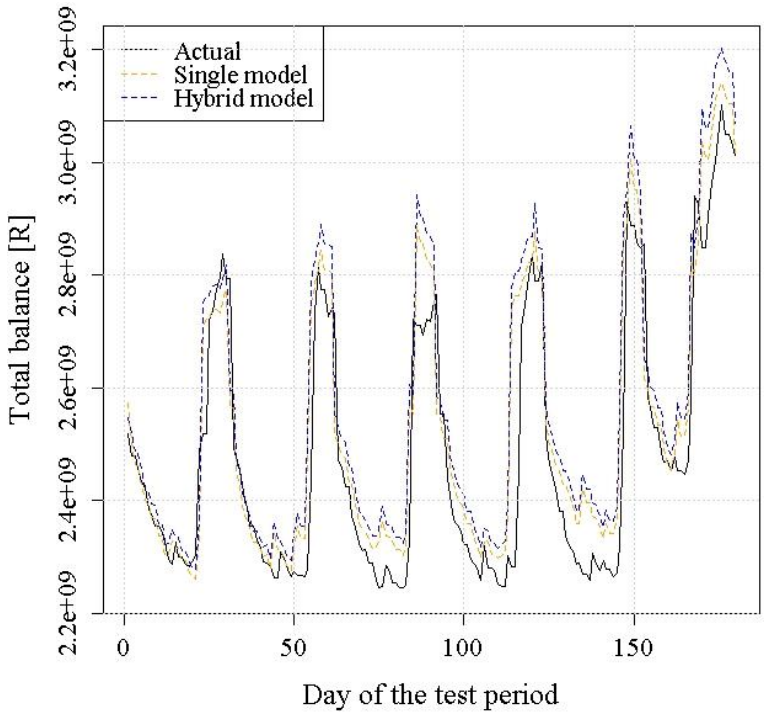


Figure 5.20. Forecasts obtained using the single model and the best performing hybrid model for the 730 day validation period over the 180 day test period.

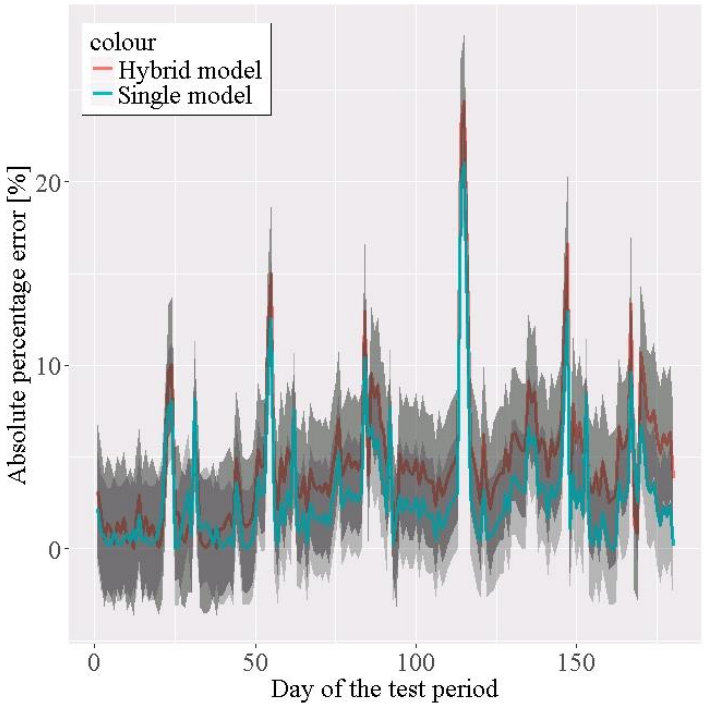


Figure 5.21 . Comparison of the errors obtained using a single model and the best performing hybrid model for the 180 day validation period over the 180 day testing period.

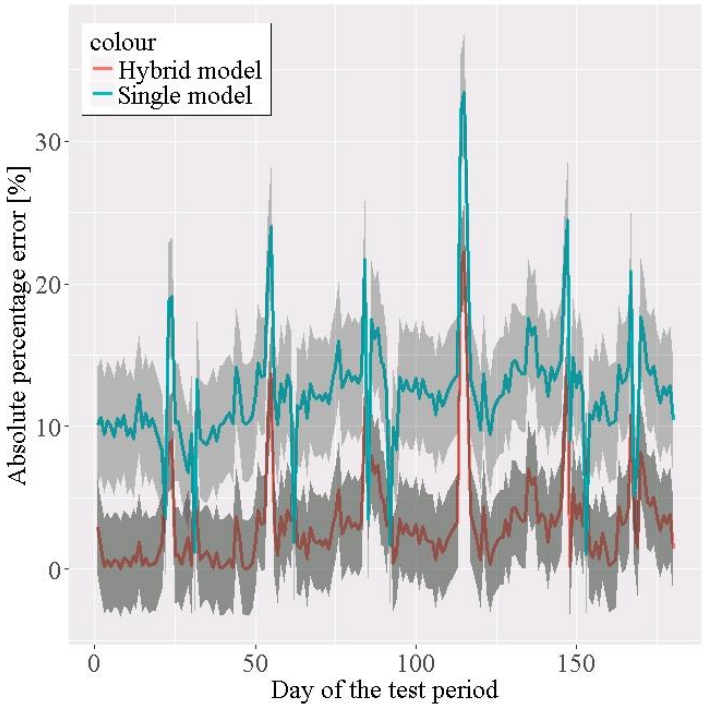


Figure 5.22. Comparison of the errors obtained using a single model and the best performing hybrid model for the 365 day validation period over the 180 day testing period.

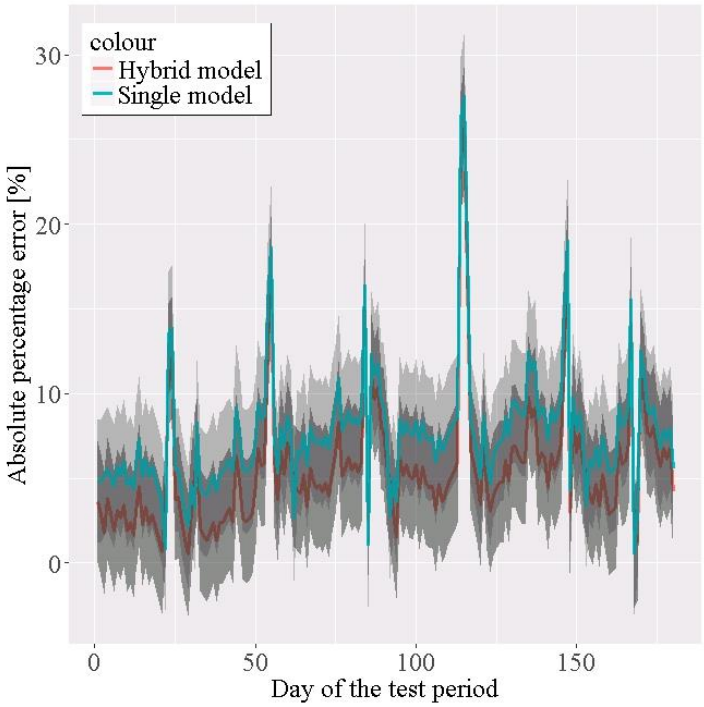


Figure 5.23. Comparison of the errors obtained using a single model and the best performing hybrid model for the 545 day validation period over the 180 day testing period.

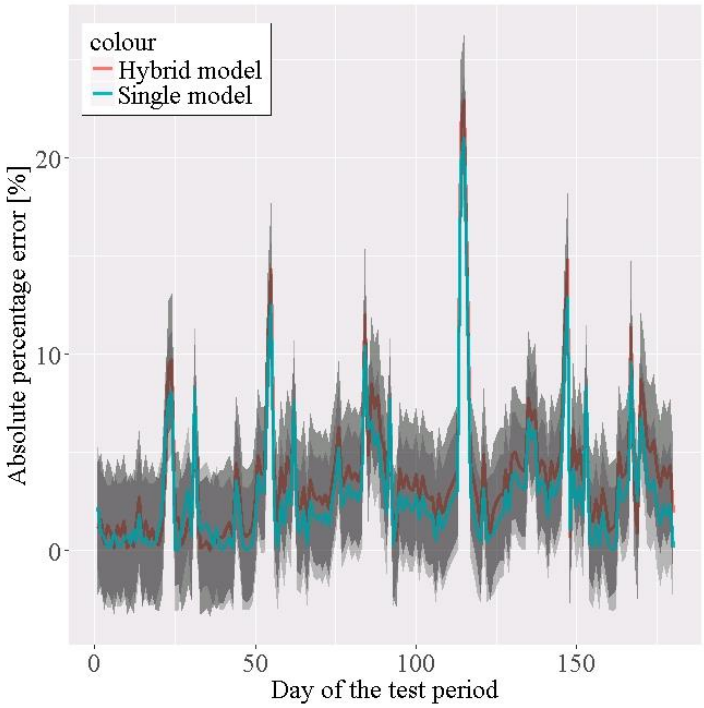


Figure 5.24. Comparison of the errors obtained using a single model and the best performing hybrid model for the 730 day validation period over the 180 day testing period.

Table 5.10 Results of the two-sample t-tests for the 180 day testing period.

Validation period	Alternative hypothesis	P-value
180 day	$\mu_1 - \mu_2 > 0$	9.86E-07
365 day	$\mu_1 - \mu_2 < 0$	(<2.2e-16) - Numerically insignificant
545 day	$\mu_1 - \mu_2 < 0$	2.45E-08
730 day	$\mu_1 - \mu_2 > 0$	0.005702

The forecasting errors for the 365 day test period are shown in Table 5.11. Plots of the forecasts for the various validation periods are provided in Figures 5.25 to 5.28 for the 365 day test period. The absolute percentage errors obtained using the single model and the best performing hybrid models for the different validation periods are shown in Figures 5.29 to 5.32 in the case of the 365 day test period. The results of the t-tests for the 365 day test period are shown in Table 5.12.

Table 5.11 Forecasting errors on the test set over 365 day test period.

Type of model	180 day validation period		365 day validation period		545 day validation period		730 day validation period	
	Error metric							
	RMSE [R]	MAPE	RMSE [R]	MAPE	RMSE [R]	MAPE	RMSE [R]	MAPE
Single model	1.03E+08	3.13 %	3.32E+08	13.41 %	2.23E+08	8.58 %	1.03E+08	3.13 %
Hybrid model with 2 clusters	1.59E+08	5.66 %	1.21E+08	4.06 %	1.80E+08	6.77 %	1.30E+08	4.49 %
Hybrid model with 3 clusters	2.15E+08	8.16 %	1.08E+08	3.39 %	2.53E+08	9.88 %	1.51E+08	5.50 %
Hybrid model with 4 clusters	2.04E+08	7.74 %	2.46E+08	9.41 %	1.44E+08	5.03 %	1.11E+08	3.64 %
Hybrid model with 5 clusters	2.14E+08	8.21 %	2.24E+08	8.51 %	1.98E+08	7.60 %	1.32E+08	4.61 %
Hybrid model with 6 clusters	2.00E+08	7.62 %	2.26E+08	8.71 %	2.77E+08	10.84 %	1.27E+08	4.36 %
Hybrid model with 7 clusters	1.91E+08	7.22 %	1.80E+08	6.75 %	2.35E+08	9.09 %	1.75E+08	6.53 %

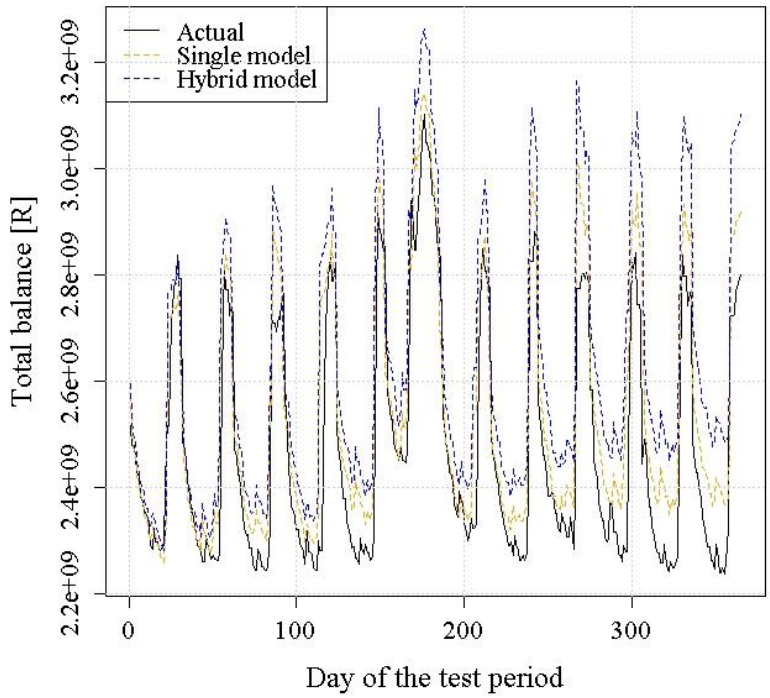


Figure 5.25. Forecasts obtained using the single and the best performing hybrid model for the 180 day validation period over the 365 day testing period.

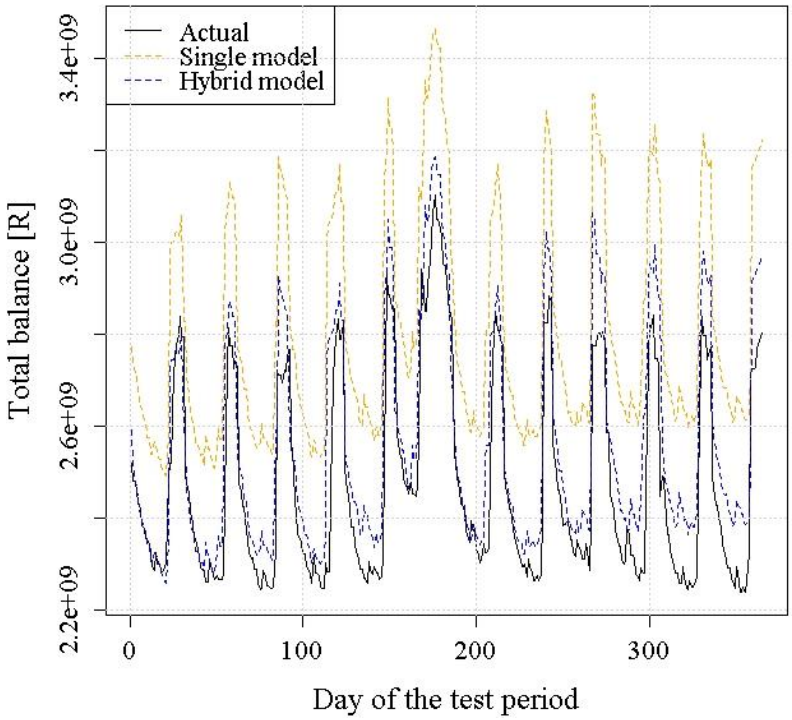


Figure 5.26. Forecasts obtained using the single and the best performing hybrid model for the 365 day validation period over the 365 day testing period.

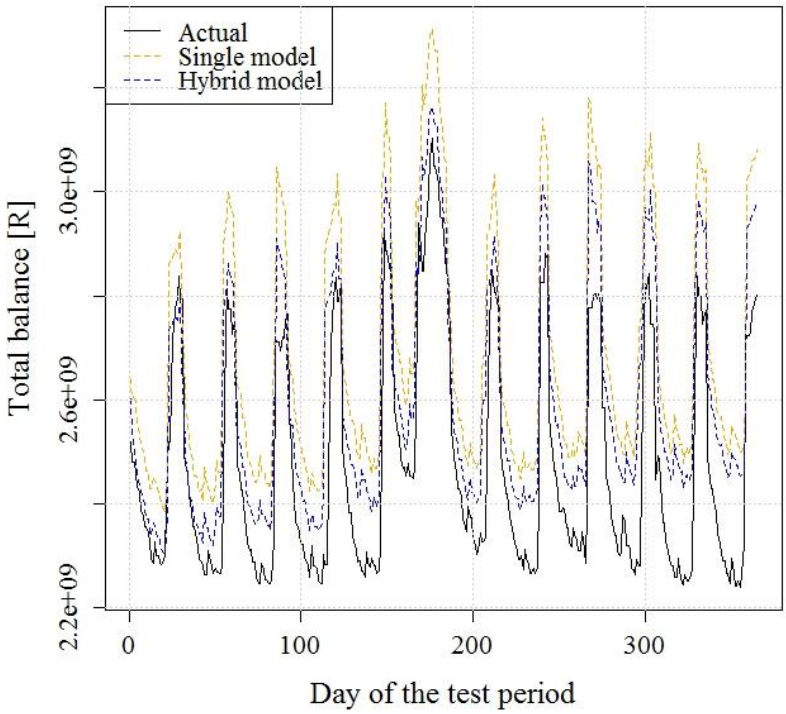


Figure 5.27. Forecasts obtained using the single and the best performing hybrid model for the 545 day validation period over the 365 day testing period.

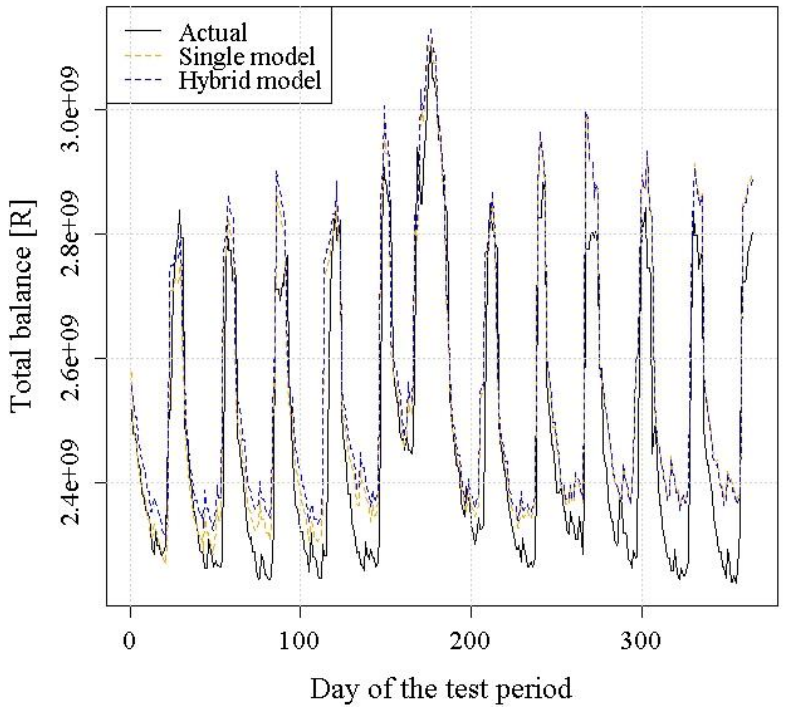


Figure 5.28. Forecasts obtained using the single and the best performing hybrid model for the 730 day validation period over the 365 day testing period.

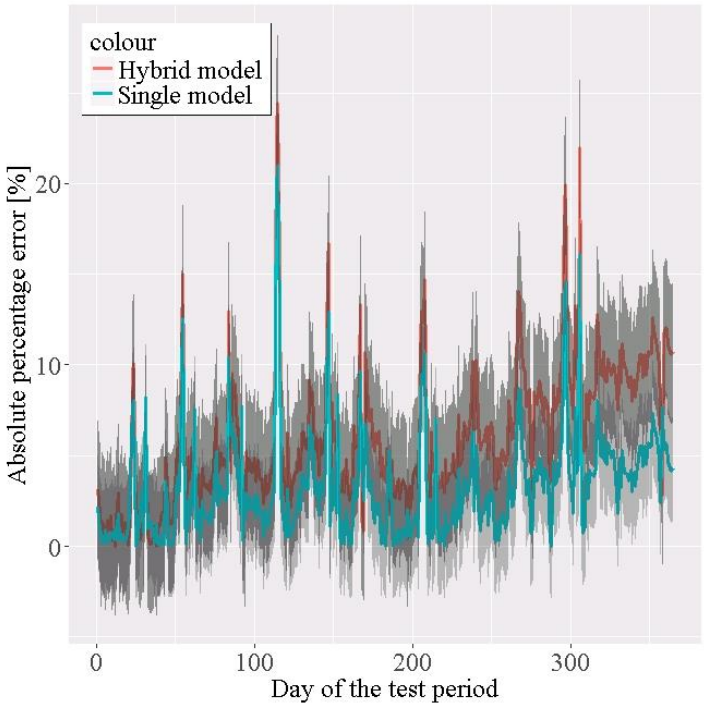


Figure 5.29. Comparison of the errors obtained using a single model and the best performing hybrid model for the 180 day validation period over the 365 day testing period.

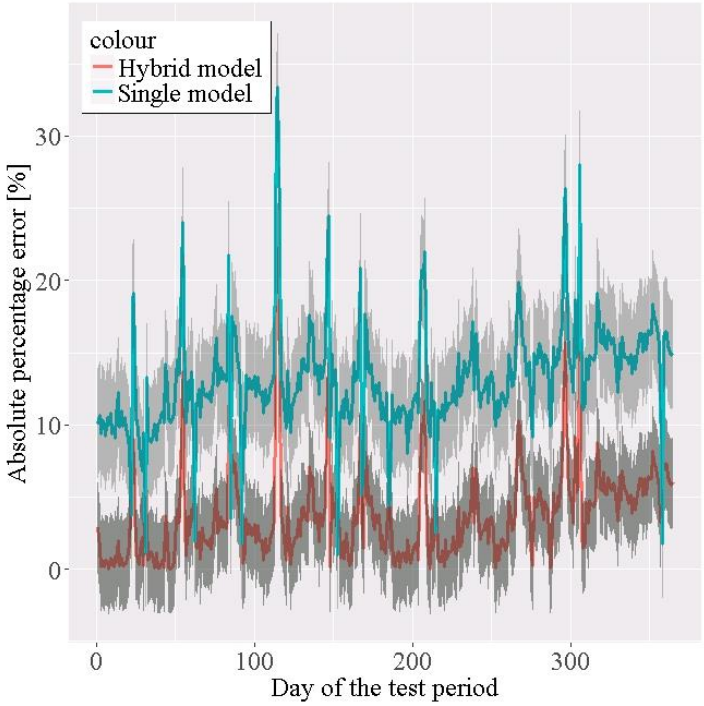


Figure 5.30. Comparison of the errors obtained using a single model and the best performing hybrid model for the 365 day validation period over the 365 day testing period.

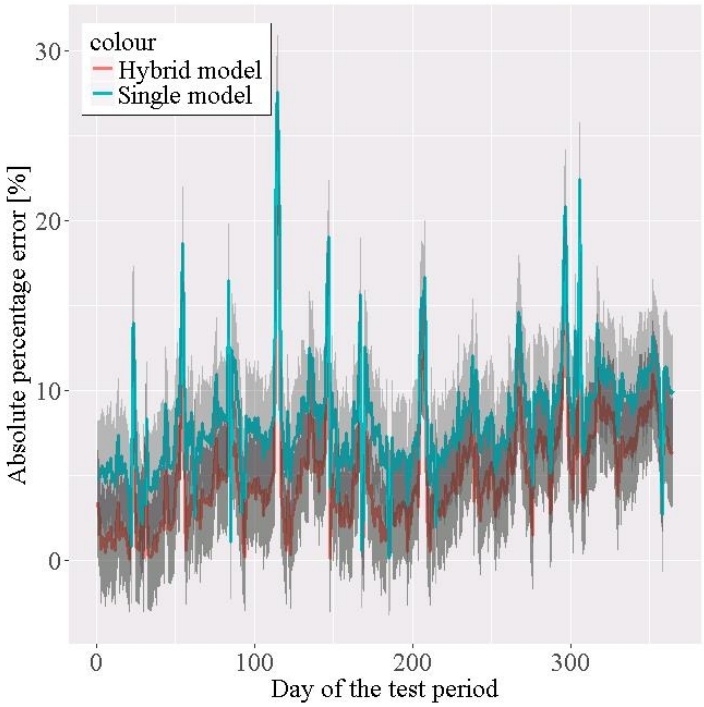


Figure 5.31. Comparison of the errors obtained using a single model and the best performing hybrid model for the 545 day validation period over the 365 day testing period.

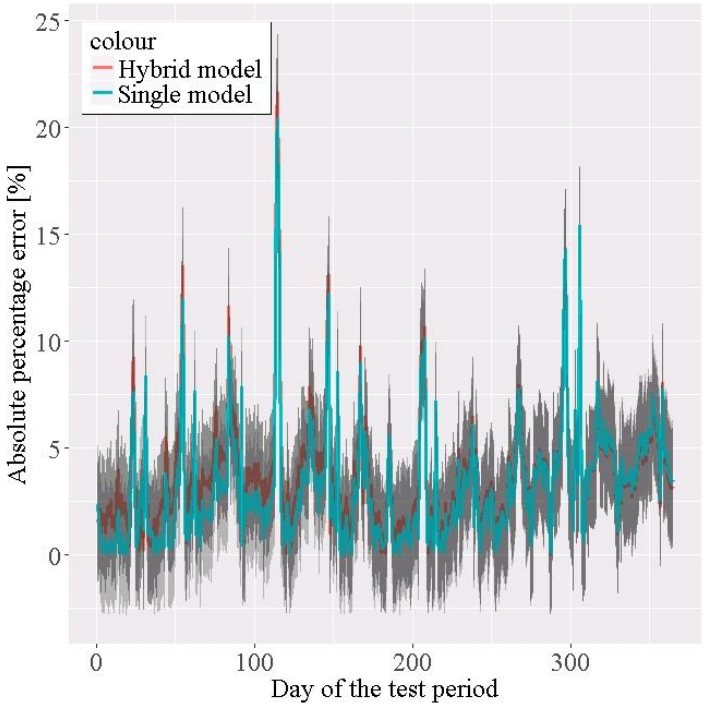


Figure 5.32. Comparison of the errors obtained using a single model and the best performing hybrid model for the 730 day validation period over the 365 day testing period.

Table 5.12 Results of the two-sample t-tests for the 365 day testing period.

Validation period	Alternative hypothesis	P-value
180 day	$\mu_1 - \mu_2 > 0$	(<2.2e-16) - Numerically insignificant
365 day	$\mu_1 - \mu_2 < 0$	(<2.2e-16) - Numerically insignificant
545 day	$\mu_1 - \mu_2 < 0$	(<2.2e-16) - Numerically insignificant
730 day	$\mu_1 - \mu_2 > 0$	0.006695

5.3 FEATURES SELECTED FOR CLASSIFICATION

The second part of this study focused on finding a way to score or classify a customer into one of the segments or clusters identified from the segmentation process using the available customer information. As mentioned in Section 4.4 and as illustrated in Figure 4.6, the classification part of the study was conducted by using the optimal clustering solution for the combination of the 365 day validation period and 365 day test period as the target. From Table 5.11, it can be noted that the best performing clustering solution for this particular combination of periods is the three cluster solution. Therefore, the classification part of the study looked at solving a three class classification problem.

In order to select the features that were to be used for the classification, `PROC STEPDISC` in SAS was used with options set to backward selection and significance level to stay set to 0.1. The features that were selected by this procedure, from the customer information dataset variables in Table 4.2, are shown in Table 5.13. The F value and probabilities are from the F-test which is a part of the analysis of covariance, which is conducted using `PROC STEPDISC` as mentioned in Section 4.6.

Table 5.13 Features selected for classification from the customer information dataset.

Variable	F Value	Pr>F
CUST_AGE	228.32	<.0001
CUST_TOT_NO_PROD	25.95	<.0001
TOT_NO_SUBPROD	38.77	<.0001
NO_DDA_ACCT	21.81	<.0001
NO_ILP_ACCT	11.06	<.0001
NO_TDA_ACCT	11.16	<.0001
NO_ZFN_ACCT	9.46	<.0001
NO_BANK_SERV	8.5	0.0002
NO_POST_ADDR	21.59	<.0001
INCOME_AMOUNT	3.09	0.0456
CNTRY_NATNLITY_ZA	4.04	0.0176
CUST_OCPTN_CDE_PM	10.58	<.0001
CUST_OCPTN_CDE_RT	14.51	<.0001
CUST_OCPTN_CDE_SM	16.09	<.0001
CUST_OCPTN_CDE_TR	3.7	0.0247
CUST_OCPTN_CDE_ZZ	49.62	<.0001
CUST_SEX_F	42.37	<.0001
DEBT_COUNSEL_IND_N	27.63	<.0001
HIGH_EDU_LVL_BCH	12.01	<.0001
HIGH_EDU_LVL_DIP	15.82	<.0001
HIGH_EDU_LVL_DOC	3.88	0.0207
HIGH_EDU_LVL_G10	8.53	0.0002
HIGH_EDU_LVL_G12	12.19	<.0001
HIGH_EDU_LVL_GRD	10.04	<.0001
HIGH_EDU_LVL_HNR	11.62	<.0001
HIGH_EDU_LVL_MST	10.14	<.0001
JNT_ACCT_IND_Y	22.67	<.0001
MRTL_STAT_CDE_S	3.33	0.0359
MRTL_STAT_CDE_U	61.47	<.0001
PROP_OWNR_IND_L	27.48	<.0001
PROP_OWNR_IND_T	22.67	<.0001
SAL_IND_N	86.35	<.0001

5.4 CLASSIFICATION RESULTS

The proportion of the three classes or in this case clusters in the number of samples or observations are given in Table 5.14. The proportion values in Table 5.14 suggest an imbalance in the dataset, with cluster 2 having a very small proportion of the samples.

Table 5.14 Proportion of the samples in each class.

Class	Number of samples	Proportion of the samples
Cluster 1	27243	53%
Cluster 2	6746	13%
Cluster 3	17328	34%

As mentioned in Section 4.7.2, the lack of a test or out of sample set meant that 10-fold cross validation was used to assess the accuracy. Furthermore, as this is a multiclass classification problem measures such as precision, recall, F-score and AUC are not truly applicable. The measure for determining the accuracy of the classification in this study is therefore purely based on classification accuracy. The results of the classification part based on the customer information variables chosen in Table 5.13 are shown in Table 5.15 and Table 5.16 for LDA and random forest classifier respectively.

Table 5.15 Results of the 10-fold cross validation on the customer information variables using LDA.

Average classification accuracy	53.81%
Average true class 1 accuracy	95.11%
Average true class 2 accuracy	5.53%
Average true class 3 accuracy	7.70%

Table 5.16 Results of the 10-fold cross validation on the customer information variables using random forest classifier.

Average classification accuracy	53.24%
Average true class 1 accuracy	89.21%
Average true class 2 accuracy	4.83%
Average true class 3 accuracy	15.54%

Unfortunately the results in the above two tables suggest that the customer information dataset does not have enough distinguishing information to correctly classify the customers into the different clusters. In order to improve the classification accuracy, the normalised balances on the earlier chosen dates (1st, 6th, 10th, 16th, 20th and 26th) for the first month of the time series i.e. 2013 June, was included into the dataset. These balances were normalised according to that month's balances for each of the accounts. This process in practice would resemble waiting one month to see the balances of a customer before scoring him/her into the obtained cluster. After including these balances the total number of variables in the dataset was 80. Feature selection was once again done using `PROC STEPDISC` with the options set to the ones described in Section 5.3. The features that were selected are shown in Table 5.17.

Table 5.17 Features selected for classification from new derived dataset with customer information and first month's balance information.

Variable	F Value	Pr>F
CUST_AGE	145.84	<.0001
CUST_TOT_NO_PROD	16.11	<.0001
TOT_NO_SUBPROD	26.05	<.0001
NO_DDA_ACCT	16.78	<.0001
NO_ILP_ACCT	4.58	0.0103
NO_TDA_ACCT	6.98	0.0009
NO_ZFN_ACCT	6.25	0.0019
NO_BANK_SERV	4.24	0.0145
NO_POST_ADDR	17.27	<.0001
INCOME_AMOUNT	4.83	0.008
CUST_OCPTN_CDE_PL	2.6	0.0742
CUST_OCPTN_CDE_PM	6.07	0.0023
CUST_OCPTN_CDE_RT	12.84	<.0001
CUST_OCPTN_CDE_SM	6.51	0.0015
CUST_OCPTN_CDE_ZZ	38.05	<.0001
CUST_SEX_F	28.68	<.0001
DEBT_COUNSEL_IND_N	20.11	<.0001
HIGH_EDU_LVL_BCH	10.56	<.0001
HIGH_EDU_LVL_DIP	11.82	<.0001
HIGH_EDU_LVL_DOC	4.36	0.0128

HIGH_EDU_LVL_G10	8.27	0.0003
HIGH_EDU_LVL_G12	11.31	<.0001
HIGH_EDU_LVL_GRD	10.15	<.0001
HIGH_EDU_LVL_HNR	11.21	<.0001
HIGH_EDU_LVL_MST	10.53	<.0001
JNT_ACCT_IND_Y	16.76	<.0001
MRTL_STAT_CDE_D	16.63	<.0001
MRTL_STAT_CDE_M	43.97	<.0001
PROP_OWNR_IND_L	11.5	<.0001
PROP_OWNR_IND_T	12.86	<.0001
SAL_IND_N	59.36	<.0001
BAL_1	61.59	<.0001
BAL_6	13.6	<.0001
BAL_10	22.74	<.0001
BAL_16	10.04	<.0001
BAL_20	138.15	<.0001
BAL_26	521.41	<.0001

The results of the classification part based on the variables chosen in Table 5.17 are shown in Table 5.18 and Table 5.19 for LDA and random forest classifier respectively.

Table 5.18 Results of the 10-fold cross validation on the newly derived variables using LDA.

Average classification accuracy	62.30%
Average true class 1 accuracy	92.08%
Average true class 2 accuracy	31.94%
Average true class 3 accuracy	27.31%

Table 5.19 Results of the 10-fold cross validation on the newly derived variables using random forest classifier.

Average classification accuracy	65.36%
Average true class 1 accuracy	86.12%
Average true class 2 accuracy	25.36%
Average true class 3 accuracy	48.31%

5.5 DISCUSSION OF RESULTS

This section discusses the results that have been presented in the earlier sections, starting with the forecasting performances of both the single and hybrid models, followed by the feature selection and finally the classification results.

5.5.1 Forecasting performance

In terms of the forecasts it can be noted that in both the case of the single model and the best performing hybrid models, the forecasts were deviating further away from the actual values in the testing period as the forecasting horizon increased. The actual value starts to decline from the upward trend seen in the training period in Figure 4.1 and maintains this downward trend throughout the testing period. The results obtained using the 730 day validation period across the various test periods seem to come closest to replicating the actual values as can be seen in Figure 5.4, Figure 5.12, Figure 5.20 and Figure 5.28. The lowest error values were obtained using the single model with the 180 day and 730 day validation periods. This was across all the test periods as can be seen from Table 5.5, Table 5.7, Table 5.9 and Table 5.11.

From Table 5.5 and Figures 5.1-5.4, one can see that the single model does very well when using either the 180 day or 730 day validation periods. Meanwhile the hybrid models do best over the 365 day validation period. This has to do with the trends of the time series that is being forecasted. In the case of the single model, for both instances it has hyper parameters of $p = 1, d = 1, q = 1$ and seasonal differencing of $D = 1$. These ARIMA model parameter settings create an effect whereby it extrapolates the local trend and adjusts for seasonality. Considering that in the time series in question, during the testing phase the time series retains a similar trend to that in the 180 day validation period, the single model with those parameters does extremely well. The hybrid model is very close, the differences between the models is not statistically significant. However, certain dynamics that it captures is more difficult to forecast and thus the forecast is slightly worse off.

However, it can be seen that using the 365 and 545 day validation periods where there is more variation in the time series in terms of trend, the single models that do best are ones which create an increasing trend pattern. This does very poorly over the test period. Meanwhile, the hybrid model captures a bit more of the dynamics of the time series and is able to create a slightly damped forecast which corresponds to the testing phase.

The hybrid model outperforms the single model over the shorter test periods. In the case of the 30 day test period, the best performing hybrid model has lower error rates than the single model across the 365 and 545 day validation periods. The differences in this case are statistically significant as shown in Table 5.6. In the case of the 180 and 730 day validation periods the differences between the hybrid and single models are not statistically significant. This can be noted from the high P-values of 0.2613 and 0.3885 from the two sample t-tests for the 180 and 730 day validation periods for the 30 day test period, as shown in Table 5.6.

Over the 90 day test period, the error rates are once again significantly lower across the 365 and 545 day validation periods as can be seen from Table 5.8. However, in the case of the 180 day validation period the single model outperforms the hybrid model and the differences in results are statistically significant as suggested by the P-value of 0.005831 from the two-sample t-test. In the case of 730 day validation period, the single model has a lower error rate but the differences in results are not statistically significant as the P-value from the two-sample t-test is 0.06129 (if you use an α of 0.05, then this value is not statistically significant).

In the case of the longer test periods, which is the 180 and 365 day test periods, the difference in errors between the hybrid model and the single model are not conclusive. This is because the hybrid model outperforms the single model across two of the validation periods while the single model does the same across the other two validation periods.

In the case of the 180 day validation period, the single model outperforms the hybrid model, as can be seen in Figure 5.21 and Figure 5.29, and the differences are statistically significant as shown in Table 5.10 and Table 5.12. It is a similar story in the case of the 730 day validation period, as can be seen in Figure 5.24 and Figure 5.32 as well as the aforementioned tables. Meanwhile, in the case of the 365 and 545 day validation periods, the hybrid model outperforms the single model as shown in Figure 5.22, Figure 5.23, Figure 5.30 and Figure 5.31. The differences are statistically significant as shown in the aforementioned tables.

As mentioned in Section 4.4, the optimal number of clusters for the hybrid model was chosen based on the forecasting performance. For the 30 day test period, the results in Table 5.5 show that the optimal number of clusters were 2, 3, 4 and 6 for the 180, 365, 545 and 730 day validation periods respectively. In the case of the 90 and 180 day test periods, the optimal number of clusters were 2, 3, 2 and 2 for the 180, 365, 545 and 730 day validation periods respectively as shown in Table 5.7 and Table 5.9. Whilst for the 365 day test period, the results in Table 5.11 show that the optimal number of clusters were 2, 3, 4 and 4 for the 180, 365, 545 and 730 day validation periods respectively. In 8 out of the 16 cases i.e. half of the time, the optimal number of clusters has been greater than 2 which was the optimal number of clusters in the dataset as given by the silhouette coefficient in Figure 4.11.

5.5.2 Feature selection

In terms of the initial customer information dataset, 32 out of the 74 variables were selected by the `PROC STEPDISC` procedure in SAS. The top five variables in terms of significance i.e. F value were customer age (CUST_AGE), the dummy variables SAL_IND_N, MRTL_STAT_CDE_U, CUST_OCPTN_CDE_ZZ and CUST_SEX_F. In terms of the newly derived dataset combining the customer information dataset and the balance variables, 37 out of the 80 variables were selected by the PROC STEDISC procedure. The top five variables in terms of significance were BAL_26, CUST_AGE, BAL_20, BAL_1 and SAL_IND_N. The significance of the balance variables can be

noted. It is also interesting to note that the salary indicator came up as a very significant variable in both cases.

5.5.3 Classification performance

In terms of the classification performance, the LDA performed better than the random forest classifier with the features selected from the customer information dataset. Although the LDA just classified almost all the samples into the majority class which was class 1 or cluster 1. It had a 95.11% on class 1 but only 5.53% and 7.70% for class 2 and 3 respectively. Both the LDA and random forest performed very poorly on the dataset with features selected from the customer information dataset with only around 53% classification accuracy for both.

In order to improve the classification accuracy the first month's normalised balances on the dates used to build the segments were included to the customer information dataset. The classification performance increased for both the LDA and random forest classifier by utilising the features that were selected from this newly derived dataset. The classification accuracy for LDA and random forecast increased by 8.49% and 12.12% respectively. The random forest does better than LDA with this derived dataset. The performance on the minority class i.e. class 2 or cluster 2 improved by 26.41% and 20.53% for LDA and random forest respectively. Furthermore, the performance on class 3 improved by 19.61% and 32.77% for LDA and random forest respectively. The performance on class 1 was still very good at 92.08% and 86.12% for LDA and random forest respectively.

5.6 CHAPTER SUMMARY

This chapter presented and discussed the results of the study. The results of the forecasting showed that the hybrid model performs statistically significantly better than the single model over the shorter test periods. This can be noted from the results of the 30 and 90 day test periods, where the hybrid model outperformed the single model over majority of the different validation periods while in other cases performed in a statistically insignificant

manner in comparison to the single model. However, over the longer test periods there was not enough conclusive evidence to suggest that the hybrid modelling strategy presented in this study outperforms or does worse than a single model. This can be attributed to the fact that predicting further into the future introduces greater modelling uncertainty, which could benefit a single model that averages for all effects. The sample that was provided for this study seems to be dominated by salaried individuals who maintain positive balances in their accounts. Logically in a scenario where the portfolio level forecast is much more complicated and varying than the one found in this study it is possible that a segment level forecasting approach aggregated up to a portfolio level could potentially yield better results. The time series pattern produced when aggregating all of these individuals together does not seem to be a very complex one and a portfolio level forecast does seem to do better overall. The segment level forecasts perform worse because some of the segments that are found are very difficult to forecast, the accuracy on these complex segment forecasts are low and when aggregating the segment level forecasts together the results are slightly worse because of this. The portfolio level forecasts are able to average out the effects much more efficiently.

The results of the classification part showed that the customer information obtained in the initial registration of the customer did not provide enough distinguishing information to adequately score a customer into one of the identified segments. Including the first month's normalised balance to the available customer information improved the results of the classification for both classifiers. It significantly improved the results on the two smaller classes.

CHAPTER 6 CONCLUSION

The purpose of this study was to find a way to accurately forecast the daily bank balance of a demand deposit account portfolio across the period of a year. In accomplishing this the study also presented the hypothesis that using a hybrid model which combined segmentation with a popular forecasting method such as ARIMA models would do better than a single time series forecasting model. The purposes of the segmentation was to identify customers with similar balance utilisation and accumulation patterns e.g. salaried individuals in comparison to a small business owner.

Segmentation was facilitated by extracting features from the time series that identified patterns of salaried individuals in comparison to other account holders. These features were used by the k-means algorithm to form the segments or clusters. After which ARIMA models were built for each of the segments, following which forecasts were obtained per segment. These segments were aggregated to obtain the portfolio level forecasts. The results were then compared to building a single model to forecast the portfolio daily balance.

Results from the study suggest that the hybrid model does perform statistically significantly better than the single model over the shorter forecast horizons. This is evident from the results of the 30 and 90 day forecasting periods where the hybrid model outperformed the single model over two out of the four validation periods and in the case of the other two, the differences between the two modelling methodologies were not statistically significant. Across the longer test periods there was not enough conclusive evidence to suggest that either the single model or hybrid model did better. The modelling

uncertainty associated with predicting further into the future could bode well for a single model which averages for all effects.

The second part of the study involved finding a way to score customers into one of the identified segments using information available on enrolment. The results however have suggested that the features available from the customer information data set are not distinguishable enough to identify the segments with accuracy. However, including information regarding a customer's first month's bank balance significantly improved the classification accuracy.

It is recommended that in future studies a wider population be used when following a similar approach. This study was limited in that the population used in this study maintained a positive balance for a significant period of time. This only allowed to pick up behaviour apparent in a select few customers. It is also recommended that the approach be applied to other demand deposit account portfolios to see if it can improve the overall modelling of demand deposits (not just current or transactional accounts as in this case but also savings deposits etc.). In doing so, the study will require total demand deposit volumes across the bank, which is difficult to obtain but would help present a more holistic picture. Additionally simulations for market rates would also be an interesting variable to be included in future studies. Lastly, it is also recommended that other clustering techniques such as SOM be used if computational resources allow for it.

REFERENCES

Ahmadi-Djam, A. & Belfrage Nordstrom, S., 2017. *Forecasting Non-Maturing Liabilities,* s.l.: s.n.

Bardenhewer, M., 2007. Modelling non-maturing products. In: *Liquidity risk: Measurement and management.* Singapore: John Wiley and Sons, pp. 220-256.

Bielak, J., Burda, A., Kowerski, M. & Pancerz, K., 2015. Modelling and Forecasting Cash Withdrawals in the Bank. *Barometr Regionalny. Analizy i prognozy,* Issue 4, pp. 165-177.

Charrad, M., Ghazzali, N., Boiteau, V. & Niknafs, A., 2014. NbClust: An R Package for Determining the Relevant Number of Clusters in a Data Set. *Journal of Statistical Software,* 61(6).

Choudhury, S. et al., 2014. A real time clustering and SVM based price-volatility prediction for optimal trading strategy. *Neurocomputing,* Issue 131, pp. 419-426.

Cui, W.-H., Wang, J.-S. & Ning, C.-X., 2014. Time Series Prediction Method of Bank Cash Flow and Simulation Comparison. *Algorithms,* 7(4), pp. 650-662.

Deb, C. et al., 2017. A review on time series forecasting techniques for building energy consumption. *Renewable and Sustainable Energy Review,* Volume 74, pp. 902-924.

Dzmuranova, H. & Teply, P., 2015. Duration of Demand Deposits in Theory. *Procedia Economics and Finance,* Issue 25, pp. 278-284.

Frauendorfer, K. & Schurle, M., 2007. Liquidity risk: Measurement and management. In: *Dynamic modeling and optimisation of non maturing accounts.* Singapore: John Wiley and Sons, pp. 327-359.

Han, J., Kamber, M. & Pei, J., 2011. *Data Mining: Concepts and Techniques.* Waltham: Morgan Kaufmann.

Hastie, T., Tibshirani, R. & Friedman, J., 2009. *The Elements of Statistical Learning.* New York: Springer-Verlag New York.

Hsieh, N.-C., 2004. An integrated data mining and behavioral scoring model for analyzing bank customers. *Expert Systems with Applications,* Issue 27, pp. 623-633.

Hsu, C.-M., 2011. A hybrid procedure for stock price prediction by integrating self-organizing map and genetic programming. *Expert Systems with Applications,* Issue 38, pp. 14026-14036.

Hsu, S.-H., Hsieh, J. P.-A., Chih, T.-C. & Hsu, K.-C., 2009. A two-stage architecture for stock price forecasting by integrating self-orgranizing map and support vector regression. *Expert Systems with Applications,* Issue 36, pp. 7947-7951.

Huang, C.-L. & Tsai, C.-Y., 2009. A hybrid SOFM-SVR with a filter-based feature selection for stock market forecasting. *Expert Systems with Applications,* Issue 36, pp. 1529-1539.

Huang, S.-C. & Wu, T.-K., 2010. Integrating recurrent SOM with wavelet-based kernel partial least square regressions for financial forecasting. *Expert Systems with Applications,* Issue 37, pp. 5698-5705.

Huang, S.-C. & Wu, T.-K., 2010. Integrating recurrent SOM with wavelet-based kernel partial least square regressions for financial forecasting. *Expert Systems with Applications,* Issue 37, pp. 5698-5705.

Hyndman, R., 2017. *Forecasting with long seasonal periods.* [Online] Available at: https://robjhyndman.com/hyndsight/longseasonality/ [Accessed 20 December 2017].

Hyndman, R. J., 2013. *fpp: Data for "Forecasting: principles and practice". R package version 0.5.*. s.l.:s.n.

Hyndman, R. J., 2017. *Seasonal periods.* [Online] Available at: https://robjhyndman.com/hyndsight/seasonal-periods/ [Accessed 20 December 2017].

Hyndman, R. J. & Athanasopoulos, G., 2014. *Forecasting: Principles and Practice.* s.l.:otexts.com.

James, G., Witten, D., Hastie, T. & Tibshirani, R., 2013. *An introduction to statistical learning with applications in R.* New York: Springer-Verlag New York.

Khajvand, M. & Tarokh, M. J., 2011. Estimating customer future value of different customer segments based on adapted RFM model in retail banking context. *Procedia Computer Science,* Issue 3, pp. 1327-1332.

Kim, S.-Y., Jung, T.-S., Suh, E.-H. & Hwang, H.-S., 2006. Customer segmentation and strategy development based on customer lifetime value: A case study. *Expert Systems with Applications,* Issue 31, pp. 101-107.

Li, C., Chiang, T.-W. & Yeh, L.-C., 2013. A novel self-orgranizing complex neuro-fuzzy approach to the problem of time series forecasting. *Neurocomputing,* Issue 99, pp. 467-476.

Lu, C.-J. & Wang, Y.-W., 2010. Combining independent component analysis and growing hierarchical self-organizing maps with support vector regression in product demand forecasting. *International Journal of Production Economics,* Issue 128, pp. 603-613.

Maechler, M. et al., 2013. *cluster: Cluster Analysis Basics and Extensions.*. s.l.:s.n.

Maes, K. & Timmermans, T., 2005. Measuring the interest rate risk of Belgian regulated savings deposits. *National Bank of Belgium, Financial Stability Review,* pp. 137-157.

Matz, L. & Neu, P., 2007. *Liquidity risk measurement and management: a practitioner's guide to global best practices.* Singapore: Johan Wiley & Sons.

Musakwa, F. T., 2013. *Measuring bank funding liquidity risk.* s.l., Acturial Approach for Financial Risks Colloquium in Lyon.

Namvar, M., Gholamian, M. R. & Khakabi, S., 2010. *A Two Phase Clustering Method for Intelligent Customer Segmentation.* s.l., s.n.

Neu, P., 2007. Liquidity risk measurement. In: *Liquidity risk: Measurement and management.* Singapore: John Wiley and Sons, pp. 15-36.

Quilumba, F. L. et al., 2015. Using smart meter data to improve the accuracy of intraday load forecasting considering customer behavior similarities. *IEEE Transactions on Smart Grid,* 6(2), pp. 911-918.

R Core Team, 2013. *R: A language and environment for computing.* Vienna: R Foundation for Statistical Computing.

SAS Institute Inc., 2008. *SAS/STAT® 9.2 User's Guide.* Cary, NC: SAS Institute Inc..

Tay, F. E. H. & Cao, L. J., 2001. Improved financial time series forecasting by combining Support Vector Machines with self-organizing feature map. *Intelligent Data Analysis,* Issue 5, pp. 339-354.

Taylor & Francis Online, 2018. *Journal of the Operational Research Society.* [Online] Available at: https://www.tandfonline.com/toc/tjor20/current [Accessed 9 October 2018].

The Pennsylvania State University, 2018. *STAT 510 Applied Time Series Analysis.* [Online]
Available at: https://onlinecourses.science.psu.edu/stat510/node/67 [Accessed 14 December 2017].

Tsiptsis, K. & Chorianopoulos, A., 2009. *Data Mining Techniques in CRM.* West Sussex: John Wiley & Sons, Ltd.

Venables, W. & Ripley, B., 2002. *Modern Applied Statistics with S.* 4th ed. New York: Springer.

Venkatesh, K., Ravi, V., Prinzie, A. & Van den Poel, D., 2014. Cash demand forecasting in ATMs by clustering and neural networks. *European Journal of Operational Research,* Issue 232, pp. 383-392.

Vento, G. & La Ganga, P., 2009. Bank liquidity risk management and supervision: Which lessons from recent market turmoil?. *Journal of Money, Investment and Banking,* Issue 10, pp. 78-125.

von Feilitzen, H., 2011. *Modeling non-maturing liabilities,* s.l.: s.n.

Wang, J., Ning, C. & Cui, W., 2015. *Time series prediction of bank cash flow based on grey neural network algorithm.* s.l., Estimation, Detection and Information Fusion (ICEDIF), 2015 International Conference on (pp.272-277).IEEE.

Xie, Y. et al., 2014. *Applied Research on Customer's Consumption Behavior of Bank POS Machine Based on Data Mining.* s.l., s.n.

Xu, R. & Wunsch, D., 2005. Survey of Clustering Algorithms. *IEEE Transactions on Neural Networks,* 16(3).

Zakrzewska, D. & Murlewski, J., 2005. *Clustering Algorithms for Bank Customer Segmentation.* s.l., s.n.

ADDENDUM A ARIMA MODEL PARAMETER SELECTION

A.1 ADDENDUM OBJECTIVE

This chapter provides the RMSE on the validation sets for the single and hybrid models when using different combinations of the ARIMA hyper parameters. These results were used to choose the optimal hyper parameters for the ARIMA model as described in Section 5.2.

A.2 180 DAY VALIDATION RESULTS

This section shows the optimal hyper parameters obtained using the 180 day validation period for the various models.

Table A.1 Hyper parameter selection for single model using 180 day validation results

Single Model							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	109343347
2	0	1	0	0	1	0	113992910
3	1	1	1	0	1	0	79180945
4	0	1	1	0	1	0	79760909
5	1	1	0	0	1	0	102061946
6	2	1	0	0	1	0	95933505
7	3	1	0	0	1	0	91225469
8	2	1	1	0	1	0	79266345
9	1	1	2	0	1	0	79210469

10	2	1	2	0	1	0	84643341
11	2	2	2	0	1	0	522315451
12	1	2	1	0	1	0	478205136

Table A.2 Hyper parameter selection for two cluster hybrid model using 180 day validation results

Cluster 1							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	30650600
2	0	1	0	0	1	0	49761230
3	1	1	1	0	1	0	44531380
4	0	1	1	0	1	0	44638560
5	1	1	0	0	1	0	47731760
6	2	1	0	0	1	0	46737740
7	3	1	0	0	1	0	45977440
8	2	1	1	0	1	0	44529010
9	1	1	2	0	1	0	44530050
10	2	1	2	0	1	0	44532050
11	2	2	2	0	1	0	88168800
12	1	2	1	0	1	0	96445240
Cluster 2							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	1.06E+08
2	0	1	0	0	1	0	81883790
3	1	1	1	0	1	0	67572820
4	0	1	1	0	1	0	67118830
5	1	1	0	0	1	0	74147480
6	2	1	0	0	1	0	70701840
7	3	1	0	0	1	0	68631590
8	2	1	1	0	1	0	67484970
9	1	1	2	0	1	0	67526360
10	2	1	2	0	1	0	66559630
11	2	2	2	0	1	0	64262890
12	1	2	1	0	1	0	74123990

Table A.3 Hyper parameter selection for three cluster hybrid model using 180 day validation results

Cluster 1							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	74472780
2	0	1	0	0	1	0	67149270
3	1	1	1	0	1	0	49998580
4	0	1	1	0	1	0	50174850
5	1	1	0	0	1	0	59959900
6	2	1	0	0	1	0	56454970
7	3	1	0	0	1	0	54025340
8	2	1	1	0	1	0	49989990
9	1	1	2	0	1	0	49992040
10	2	1	2	0	1	0	50779840
11	2	2	2	0	1	0	49339660
12	1	2	1	0	1	0	64453080
Cluster 2							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	31055800
2	0	1	0	0	1	0	41750890
3	1	1	1	0	1	0	40342300
4	0	1	1	0	1	0	40462010
5	1	1	0	0	1	0	40968700
6	2	1	0	0	1	0	40734220
7	3	1	0	0	1	0	40445020
8	2	1	1	0	1	0	40340060
9	1	1	2	0	1	0	40343750
10	2	1	2	0	1	0	40329490
11	2	2	2	0	1	0	188064100
12	1	2	1	0	1	0	76492800
Cluster 3							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	52044870
2	0	1	0	0	1	0	28955100
3	1	1	1	0	1	0	28913030
4	0	1	1	0	1	0	28900680
5	1	1	0	0	1	0	28004140
6	2	1	0	0	1	0	27828820

7	3	1	0	0	1	0	27876110
8	2	1	1	0	1	0	28918400
9	1	1	2	0	1	0	28916170
10	2	1	2	0	1	0	28787310
11	2	2	2	0	1	0	89624210
12	1	2	1	0	1	0	41498310

Table A.4 Hyper parameter selection for four cluster hybrid model using 180 day validation results

Cluster 1							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	5.4E+07
2	0	1	0	0	1	0	3.1E+07
3	1	1	1	0	1	0	3.6E+07
4	0	1	1	0	1	0	3.6E+07
5	1	1	0	0	1	0	3.2E+07
6	2	1	0	0	1	0	3.3E+07
7	3	1	0	0	1	0	3.4E+07
8	2	1	1	0	1	0	3.6E+07
9	1	1	2	0	1	0	3.6E+07
10	2	1	2	0	1	0	3.6E+07
11	2	2	2	0	1	0	3.8E+07
12	1	2	1	0	1	0	3E+07
Cluster 2							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	2.3E+07
2	0	1	0	0	1	0	3.3E+07
3	1	1	1	0	1	0	3.3E+07
4	0	1	1	0	1	0	3.3E+07
5	1	1	0	0	1	0	3.3E+07
6	2	1	0	0	1	0	3.3E+07
7	3	1	0	0	1	0	3.3E+07
8	2	1	1	0	1	0	3.3E+07
9	1	1	2	0	1	0	3.3E+07
10	2	1	2	0	1	0	2.5E+07
11	2	2	2	0	1	0	4.1E+07
12	1	2	1	0	1	0	4E+07
Cluster 3							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**

1	0	0	0	0	1	0	1.6E+07
2	0	1	0	0	1	0	2.1E+07
3	1	1	1	0	1	0	1.7E+07
4	0	1	1	0	1	0	1.7E+07
5	1	1	0	0	1	0	2E+07
6	2	1	0	0	1	0	1.9E+07
7	3	1	0	0	1	0	1.8E+07
8	2	1	1	0	1	0	1.7E+07
9	1	1	2	0	1	0	1.7E+07
10	2	1	2	0	1	0	1.7E+07
11	2	2	2	0	1	0	2.8E+07
12	1	2	1	0	1	0	5.8E+07
Cluster 4							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	6.7E+07
2	0	1	0	0	1	0	6.8E+07
3	1	1	1	0	1	0	4.4E+07
4	0	1	1	0	1	0	4.5E+07
5	1	1	0	0	1	0	5.9E+07
6	2	1	0	0	1	0	5.5E+07
7	3	1	0	0	1	0	5.2E+07
8	2	1	1	0	1	0	4.4E+07
9	1	1	2	0	1	0	4.4E+07
10	2	1	2	0	1	0	4.5E+07
11	2	2	2	0	1	0	4.6E+07
12	1	2	1	0	1	0	5.5E+07

Table A.5 Hyper parameter selection for five cluster hybrid model using 180 day validation results

Cluster 1							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	47853120
2	0	1	0	0	1	0	32740350
3	1	1	1	0	1	0	35882140
4	0	1	1	0	1	0	35411930
5	1	1	0	0	1	0	32764560
6	2	1	0	0	1	0	33147270
7	3	1	0	0	1	0	33556030
8	2	1	1	0	1	0	35795210

9	1	1	2	0	1	0	35837230
10	2	1	2	0	1	0	34841960
11	2	2	2	0	1	0	34057790
12	1	2	1	0	1	0	37742140
Cluster 2							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	31236790
2	0	1	0	0	1	0	19322320
3	1	1	1	0	1	0	16338150
4	0	1	1	0	1	0	16371270
5	1	1	0	0	1	0	18065590
6	2	1	0	0	1	0	17504350
7	3	1	0	0	1	0	17120440
8	2	1	1	0	1	0	16338250
9	1	1	2	0	1	0	16338040
10	2	1	2	0	1	0	16367390
11	2	2	2	0	1	0	64237540
12	1	2	1	0	1	0	42805290
Cluster 3							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	12791100
2	0	1	0	0	1	0	17193530
3	1	1	1	0	1	0	17173550
4	0	1	1	0	1	0	17169960
5	1	1	0	0	1	0	17166240
6	2	1	0	0	1	0	20746460
7	3	1	0	0	1	0	19838640
8	2	1	1	0	1	0	19877280
9	1	1	2	0	1	0	17247930
10	2	1	2	0	1	0	20282710
11	2	2	2	0	1	0	28274900
12	1	2	1	0	1	0	27863450
Cluster 4							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	18320120
2	0	1	0	0	1	0	20893180
3	1	1	1	0	1	0	19644480
4	0	1	1	0	1	0	19771990
5	1	1	0	0	1	0	20381430

6	2	1	0	0	1	0	20102280
7	3	1	0	0	1	0	19846600
8	2	1	1	0	1	0	19645910
9	1	1	2	0	1	0	19645350
10	2	1	2	0	1	0	19645550
11	2	2	2	0	1	0	31245320
12	1	2	1	0	1	0	40453970
Cluster 5							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	49534640
2	0	1	0	0	1	0	55949230
3	1	1	1	0	1	0	34777190
4	0	1	1	0	1	0	36595880
5	1	1	0	0	1	0	49138400
6	2	1	0	0	1	0	45400930
7	3	1	0	0	1	0	42533350
8	2	1	1	0	1	0	34778760
9	1	1	2	0	1	0	34776630
10	2	1	2	0	1	0	34791710
11	2	2	2	0	1	0	37270230
12	1	2	1	0	1	0	41934780

Table A.6 Hyper parameter selection for six cluster hybrid model using 180 day validation results

Cluster 1							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	15742200
2	0	1	0	0	1	0	20295760
3	1	1	1	0	1	0	19049570
4	0	1	1	0	1	0	19071160
5	1	1	0	0	1	0	19750140
6	2	1	0	0	1	0	19445880
7	3	1	0	0	1	0	19183470
8	2	1	1	0	1	0	19040570
9	1	1	2	0	1	0	19045140
10	2	1	2	0	1	0	19001660
11	2	2	2	0	1	0	24174720
12	1	2	1	0	1	0	34907360
Cluster 2							

Model order	p	d	q	P	D	Q	RMSE [R]
1	0	0	0	0	1	0	14567990
2	0	1	0	0	1	0	18188910
3	1	1	1	0	1	0	14337220
4	0	1	1	0	1	0	14370660
5	1	1	0	0	1	0	16897120
6	2	1	0	0	1	0	16215950
7	3	1	0	0	1	0	15719610
8	2	1	1	0	1	0	14331930
9	1	1	2	0	1	0	14333610
10	2	1	2	0	1	0	14371850
11	2	2	2	0	1	0	14081290
12	1	2	1	0	1	0	30702150
Cluster 3							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	23277770
2	0	1	0	0	1	0	23821580
3	1	1	1	0	1	0	23822610
4	0	1	1	0	1	0	23815080
5	1	1	0	0	1	0	23811040
6	2	1	0	0	1	0	23959360
7	3	1	0	0	1	0	24116100
8	2	1	1	0	1	0	24927190
9	1	1	2	0	1	0	20610100
10	2	1	2	0	1	0	25877590
11	2	2	2	0	1	0	26056260
12	1	2	1	0	1	0	25557980
Cluster 4							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	43765770
2	0	1	0	0	1	0	14348380
3	1	1	1	0	1	0	15261800
4	0	1	1	0	1	0	15176240
5	1	1	0	0	1	0	14856010
6	2	1	0	0	1	0	15019120
7	3	1	0	0	1	0	15150080
8	2	1	1	0	1	0	15342470
9	1	1	2	0	1	0	15317370
10	2	1	2	0	1	0	15326890

11	2	2	2	0	1	0	68281180
12	1	2	1	0	1	0	22882130
Cluster 5							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	44985630
2	0	1	0	0	1	0	31312620
3	1	1	1	0	1	0	34192410
4	0	1	1	0	1	0	33711020
5	1	1	0	0	1	0	31347160
6	2	1	0	0	1	0	31709420
7	3	1	0	0	1	0	32088660
8	2	1	1	0	1	0	34107710
9	1	1	2	0	1	0	34147630
10	2	1	2	0	1	0	34167780
11	2	2	2	0	1	0	33009830
12	1	2	1	0	1	0	35022310
Cluster 6							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	45448860
2	0	1	0	0	1	0	54870450
3	1	1	1	0	1	0	33750210
4	0	1	1	0	1	0	36018110
5	1	1	0	0	1	0	48226500
6	2	1	0	0	1	0	44563900
7	3	1	0	0	1	0	41708290
8	2	1	1	0	1	0	33714760
9	1	1	2	0	1	0	33725590
10	2	1	2	0	1	0	33729370
11	2	2	2	0	1	0	36557020
12	1	2	1	0	1	0	40137790

Table A.7 Hyper parameter selection for seven cluster hybrid model using 180 day validation results

Cluster 1							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	44724470
2	0	1	0	0	1	0	29208280

3	1	1	1	0	1	0	34958350
4	0	1	1	0	1	0	34407770
5	1	1	0	0	1	0	30019000
6	2	1	0	0	1	0	30736780
7	3	1	0	0	1	0	31597520
8	2	1	1	0	1	0	34929810
9	1	1	2	0	1	0	34943370
10	2	1	2	0	1	0	34956760
11	2	2	2	0	1	0	34909370
12	1	2	1	0	1	0	50605010
Cluster 2							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	18256370
2	0	1	0	0	1	0	21530400
3	1	1	1	0	1	0	30977220
4	0	1	1	0	1	0	21528470
5	1	1	0	0	1	0	21528170
6	2	1	0	0	1	0	22461230
7	3	1	0	0	1	0	22221030
8	2	1	1	0	1	0	26527250
9	1	1	2	0	1	0	28752720
10	2	1	2	0	1	0	29131290
11	2	2	2	0	1	0	38181140
12	1	2	1	0	1	0	39063650
Cluster 3							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	41967320
2	0	1	0	0	1	0	16285800
3	1	1	1	0	1	0	16970870
4	0	1	1	0	1	0	16905630
5	1	1	0	0	1	0	16692250
6	2	1	0	0	1	0	16816400
7	3	1	0	0	1	0	16906600
8	2	1	1	0	1	0	17002060
9	1	1	2	0	1	0	16991460
10	2	1	2	0	1	0	16998650
11	2	2	2	0	1	0	27009310
12	1	2	1	0	1	0	17366420
Cluster 4							

Model order	p	d	q	P	D	Q	RMSE [R]
1	0	0	0	0	1	0	35891860
2	0	1	0	0	1	0	52855750
3	1	1	1	0	1	0	31962120
4	0	1	1	0	1	0	36216970
5	1	1	0	0	1	0	46919020
6	2	1	0	0	1	0	43458010
7	3	1	0	0	1	0	41048510
8	2	1	1	0	1	0	31755790
9	1	1	2	0	1	0	31706030
10	2	1	2	0	1	0	31635460
11	2	2	2	0	1	0	36598170
12	1	2	1	0	1	0	36681250
Cluster 5							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	18151820
2	0	1	0	0	1	0	18052880
3	1	1	1	0	1	0	17395620
4	0	1	1	0	1	0	17497200
5	1	1	0	0	1	0	17735150
6	2	1	0	0	1	0	17564570
7	3	1	0	0	1	0	17416770
8	2	1	1	0	1	0	17400640
9	1	1	2	0	1	0	17396490
10	2	1	2	0	1	0	26517380
11	2	2	2	0	1	0	58429360
12	1	2	1	0	1	0	36440110
Cluster 6							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	23456250
2	0	1	0	0	1	0	20718420
3	1	1	1	0	1	0	17599870
4	0	1	1	0	1	0	17670420
5	1	1	0	0	1	0	19273230
6	2	1	0	0	1	0	18615560
7	3	1	0	0	1	0	18258820
8	2	1	1	0	1	0	17618280
9	1	1	2	0	1	0	17608360
10	2	1	2	0	1	0	17775430

11	2	2	2	0	1	0	17639820
12	1	2	1	0	1	0	30768490
Cluster 7							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	9206325
2	0	1	0	0	1	0	15794870
3	1	1	1	0	1	0	11826000
4	0	1	1	0	1	0	12016880
5	1	1	0	0	1	0	14814310
6	2	1	0	0	1	0	14271660
7	3	1	0	0	1	0	13816290
8	2	1	1	0	1	0	11865460
9	1	1	2	0	1	0	11839810
10	2	1	2	0	1	0	12147970
11	2	2	2	0	1	0	44775580
12	1	2	1	0	1	0	36949180

A.3 365 DAY VALIDATION RESULTS

This section shows the optimal hyper parameters obtained using the 365 day validation period for the various models.

Table A.8 Hyper parameter selection for single model using 365 day validation results

Single Model							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	128678563
2	0	1	0	0	1	0	76286539
3	1	1	1	0	1	0	83644289
4	0	1	1	0	1	0	82757673
5	1	1	0	0	1	0	76734695
6	2	1	0	0	1	0	76708371
7	3	1	0	0	1	0	77004033
8	2	1	1	0	1	0	83309653
9	1	1	2	0	1	0	83523591
10	2	1	2	0	1	0	81015773
11	2	2	2	0	1	0	282610274
12	1	2	1	0	1	0	321908434

Table A.9 Hyper parameter selection for two cluster hybrid model using 365 day validation results

Cluster 1							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	50493480
2	0	1	0	0	1	0	55341006
3	1	1	1	0	1	0	62319910
4	0	1	1	0	1	0	61437566
5	1	1	0	0	1	0	56294263
6	2	1	0	0	1	0	57130249
7	3	1	0	0	1	0	58248343
8	2	1	1	0	1	0	62251271
9	1	1	2	0	1	0	62275801
10	2	1	2	0	1	0	62310122
11	2	2	2	0	1	0	471028931
12	1	2	1	0	1	0	351743924
Cluster 2							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	95621645
2	0	1	0	0	1	0	60558067
3	1	1	1	0	1	0	59247093
4	0	1	1	0	1	0	59794286
5	1	1	0	0	1	0	60471093
6	2	1	0	0	1	0	60914121
7	3	1	0	0	1	0	60984728
8	2	1	1	0	1	0	59376934
9	1	1	2	0	1	0	59323214
10	2	1	2	0	1	0	60079283
11	2	2	2	0	1	0	140468734
12	1	2	1	0	1	0	108368338

Table A.10 Hyper parameter selection for three cluster hybrid model using 365 day validation results

Cluster 1							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	71236263
2	0	1	0	0	1	0	44532868
3	1	1	1	0	1	0	44468082

4	0	1	1	0	1	0	44664223
5	1	1	0	0	1	0	44555814
6	2	1	0	0	1	0	44674668
7	3	1	0	0	1	0	44731756
8	2	1	1	0	1	0	44481098
9	1	1	2	0	1	0	44476699
10	2	1	2	0	1	0	44741995
11	2	2	2	0	1	0	58264067
12	1	2	1	0	1	0	71712832
Cluster 2							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	35380979
2	0	1	0	0	1	0	55362986
3	1	1	1	0	1	0	35096613
4	0	1	1	0	1	0	55678103
5	1	1	0	0	1	0	55658836
6	2	1	0	0	1	0	34122177
7	3	1	0	0	1	0	37518029
8	2	1	1	0	1	0	33047416
9	1	1	2	0	1	0	32938000
10	2	1	2	0	1	0	39500442
11	2	2	2	0	1	0	88804518
12	1	2	1	0	1	0	312061784
Cluster 3							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	48155496
2	0	1	0	0	1	0	32906412
3	1	1	1	0	1	0	27795296
4	0	1	1	0	1	0	27772559
5	1	1	0	0	1	0	32031094
6	2	1	0	0	1	0	31816117
7	3	1	0	0	1	0	31267341
8	2	1	1	0	1	0	27758067
9	1	1	2	0	1	0	27778869
10	2	1	2	0	1	0	27458020
11	2	2	2	0	1	0	219039816
12	1	2	1	0	1	0	131546160

Table A.11 Hyper parameter selection for four cluster hybrid model using 365 day validation results

Cluster 1							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	47808678
2	0	1	0	0	1	0	39474146
3	1	1	1	0	1	0	31872759
4	0	1	1	0	1	0	32281248
5	1	1	0	0	1	0	38550187
6	2	1	0	0	1	0	38227133
7	3	1	0	0	1	0	37550869
8	2	1	1	0	1	0	31911418
9	1	1	2	0	1	0	31894025
10	2	1	2	0	1	0	32097548
11	2	2	2	0	1	0	227672234
12	1	2	1	0	1	0	180315409
Cluster 2							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	36146327
2	0	1	0	0	1	0	41514151
3	1	1	1	0	1	0	34014492
4	0	1	1	0	1	0	41324134
5	1	1	0	0	1	0	31183761
6	2	1	0	0	1	0	29905519
7	3	1	0	0	1	0	29711255
8	2	1	1	0	1	0	30052155
9	1	1	2	0	1	0	29731106
10	2	1	2	0	1	0	29357486
11	2	2	2	0	1	0	180596908
12	1	2	1	0	1	0	176342282
Cluster 3							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	12835946
2	0	1	0	0	1	0	29411634
3	1	1	1	0	1	0	32001496
4	0	1	1	0	1	0	32179791
5	1	1	0	0	1	0	29897710
6	2	1	0	0	1	0	30326293

7	3	1	0	0	1	0	30848997
8	2	1	1	0	1	0	32073319
9	1	1	2	0	1	0	32025620
10	2	1	2	0	1	0	35660996
11	2	2	2	0	1	0	88043426
12	1	2	1	0	1	0	45130313
Cluster 4							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	64098348
2	0	1	0	0	1	0	39532931
3	1	1	1	0	1	0	40337865
4	0	1	1	0	1	0	40699361
5	1	1	0	0	1	0	39596797
6	2	1	0	0	1	0	39822866
7	3	1	0	0	1	0	39938088
8	2	1	1	0	1	0	40355135
9	1	1	2	0	1	0	40350079
10	2	1	2	0	1	0	40778347
11	2	2	2	0	1	0	83399458
12	1	2	1	0	1	0	38641788

Table A.12 Hyper parameter selection for five cluster hybrid model using 365 day validation results

Cluster 1							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	55552869
2	0	1	0	0	1	0	33357972
3	1	1	1	0	1	0	37414802
4	0	1	1	0	1	0	38489781
5	1	1	0	0	1	0	34140854
6	2	1	0	0	1	0	34996141
7	3	1	0	0	1	0	35747102
8	2	1	1	0	1	0	36896796
9	1	1	2	0	1	0	36879494
10	2	1	2	0	1	0	36872408
11	2	2	2	0	1	0	73718801
12	1	2	1	0	1	0	31990070

Cluster 2							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	30212638
2	0	1	0	0	1	0	56648763
3	1	1	1	0	1	0	40782723
4	0	1	1	0	1	0	56579911
5	1	1	0	0	1	0	50108659
6	2	1	0	0	1	0	40220558
7	3	1	0	0	1	0	28813386
8	2	1	1	0	1	0	29783098
9	1	1	2	0	1	0	40924222
10	2	1	2	0	1	0	32331505
11	2	2	2	0	1	0	99672100
12	1	2	1	0	1	0	102769249
Cluster 3							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	23954101
2	0	1	0	0	1	0	19258123
3	1	1	1	0	1	0	18494462
4	0	1	1	0	1	0	18569937
5	1	1	0	0	1	0	19046056
6	2	1	0	0	1	0	18951323
7	3	1	0	0	1	0	18823538
8	2	1	1	0	1	0	18493378
9	1	1	2	0	1	0	18494919
10	2	1	2	0	1	0	18492506
11	2	2	2	0	1	0	204398285
12	1	2	1	0	1	0	125785230
Cluster 4							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	45307240
2	0	1	0	0	1	0	31060491
3	1	1	1	0	1	0	28297266
4	0	1	1	0	1	0	28285090
5	1	1	0	0	1	0	30235878
6	2	1	0	0	1	0	29792376
7	3	1	0	0	1	0	29286177
8	2	1	1	0	1	0	28299074
9	1	1	2	0	1	0	28298525

10	2	1	2	0	1	0	28297963
11	2	2	2	0	1	0	128905383
12	1	2	1	0	1	0	123629279
Cluster 5							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	33644663
2	0	1	0	0	1	0	18239974
3	1	1	1	0	1	0	17961562
4	0	1	1	0	1	0	17920169
5	1	1	0	0	1	0	18114343
6	2	1	0	0	1	0	18119143
7	3	1	0	0	1	0	18057668
8	2	1	1	0	1	0	17955272
9	1	1	2	0	1	0	17958032
10	2	1	2	0	1	0	17914912
11	2	2	2	0	1	0	140900638
12	1	2	1	0	1	0	85393610

Table A.13 Hyper parameter selection for six cluster hybrid model using 365 day validation results

Cluster 1							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	45943924
2	0	1	0	0	1	0	27479840
3	1	1	1	0	1	0	27837313
4	0	1	1	0	1	0	27432024
5	1	1	0	0	1	0	26797320
6	2	1	0	0	1	0	26572003
7	3	1	0	0	1	0	26550906
8	2	1	1	0	1	0	27559495
9	1	1	2	0	1	0	27539045
10	2	1	2	0	1	0	27546575
11	2	2	2	0	1	0	26852346
12	1	2	1	0	1	0	108439942
Cluster 2							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	18140132
2	0	1	0	0	1	0	21334838
3	1	1	1	0	1	0	18469752

4	0	1	1	0	1	0	18749627
5	1	1	0	0	1	0	20463471
6	2	1	0	0	1	0	20038228
7	3	1	0	0	1	0	19581103
8	2	1	1	0	1	0	18469898
9	1	1	2	0	1	0	18470949
10	2	1	2	0	1	0	18462611
11	2	2	2	0	1	0	153527407
12	1	2	1	0	1	0	127242794
Cluster 3							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	42880657
2	0	1	0	0	1	0	36715287
3	1	1	1	0	1	0	27918810
4	0	1	1	0	1	0	28285518
5	1	1	0	0	1	0	34808505
6	2	1	0	0	1	0	33836419
7	3	1	0	0	1	0	32707836
8	2	1	1	0	1	0	27933882
9	1	1	2	0	1	0	27928476
10	2	1	2	0	1	0	27926826
11	2	2	2	0	1	0	230987905
12	1	2	1	0	1	0	154891177
Cluster 4							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	24045480
2	0	1	0	0	1	0	33167204
3	1	1	1	0	1	0	34232767
4	0	1	1	0	1	0	33798360
5	1	1	0	0	1	0	33165198
6	2	1	0	0	1	0	33276529
7	3	1	0	0	1	0	33437099
8	2	1	1	0	1	0	34238904
9	1	1	2	0	1	0	34232342
10	2	1	2	0	1	0	34227220
11	2	2	2	0	1	0	79733852
12	1	2	1	0	1	0	40268213
Cluster 5							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**

1	0	0	0	0	1	0	27244043
2	0	1	0	0	1	0	42431973
3	1	1	1	0	1	0	42314156
4	0	1	1	0	1	0	42413128
5	1	1	0	0	1	0	39758395
6	2	1	0	0	1	0	34443678
7	3	1	0	0	1	0	28806698
8	2	1	1	0	1	0	36239359
9	1	1	2	0	1	0	42301447
10	2	1	2	0	1	0	28823664
11	2	2	2	0	1	0	55788219
12	1	2	1	0	1	0	63416992
Cluster 6							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	37427302
2	0	1	0	0	1	0	24094685
3	1	1	1	0	1	0	20844665
4	0	1	1	0	1	0	21194558
5	1	1	0	0	1	0	23639040
6	2	1	0	0	1	0	23739742
7	3	1	0	0	1	0	23508780
8	2	1	1	0	1	0	20878714
9	1	1	2	0	1	0	20866130
10	2	1	2	0	1	0	21056013
11	2	2	2	0	1	0	119300409
12	1	2	1	0	1	0	92737548

Table A.14 Hyper parameter selection for seven cluster hybrid model using 365 day validation results

Cluster 1							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	39714408
2	0	1	0	0	1	0	29034531
3	1	1	1	0	1	0	28024091
4	0	1	1	0	1	0	27833715
5	1	1	0	0	1	0	28241613
6	2	1	0	0	1	0	28113998

7	3	1	0	0	1	0	27944591
8	2	1	1	0	1	0	28018694
9	1	1	2	0	1	0	28021671
10	2	1	2	0	1	0	28025987
11	2	2	2	0	1	0	40990132
12	1	2	1	0	1	0	32245697
Cluster 2							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	16416306
2	0	1	0	0	1	0	16695966
3	1	1	1	0	1	0	16978543
4	0	1	1	0	1	0	16690247
5	1	1	0	0	1	0	16350487
6	2	1	0	0	1	0	16536431
7	3	1	0	0	1	0	18850046
8	2	1	1	0	1	0	18067877
9	1	1	2	0	1	0	16507544
10	2	1	2	0	1	0	20113406
11	2	2	2	0	1	0	66273964
12	1	2	1	0	1	0	64846061
Cluster 3							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	35972709
2	0	1	0	0	1	0	13367708
3	1	1	1	0	1	0	12620952
4	0	1	1	0	1	0	12663768
5	1	1	0	0	1	0	12984270
6	2	1	0	0	1	0	12826559
7	3	1	0	0	1	0	12554170
8	2	1	1	0	1	0	12588756
9	1	1	2	0	1	0	12640615
10	2	1	2	0	1	0	12555048
11	2	2	2	0	1	0	259905181
12	1	2	1	0	1	0	175061966
Cluster 4							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	38265283
2	0	1	0	0	1	0	25788658
3	1	1	1	0	1	0	25596162

4	0	1	1	0	1	0	25856017
5	1	1	0	0	1	0	25552700
6	2	1	0	0	1	0	25536196
7	3	1	0	0	1	0	25581561
8	2	1	1	0	1	0	25551098
9	1	1	2	0	1	0	25546740
10	2	1	2	0	1	0	25543997
11	2	2	2	0	1	0	27495083
12	1	2	1	0	1	0	33990449
Cluster 5							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	16200081
2	0	1	0	0	1	0	33880467
3	1	1	1	0	1	0	21764466
4	0	1	1	0	1	0	33858252
5	1	1	0	0	1	0	28994253
6	2	1	0	0	1	0	22986135
7	3	1	0	0	1	0	17294214
8	2	1	1	0	1	0	17240685
9	1	1	2	0	1	0	23205577
10	2	1	2	0	1	0	20375650
11	2	2	2	0	1	0	266498393
12	1	2	1	0	1	0	68447743
Cluster 6							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	20490416
2	0	1	0	0	1	0	15876430
3	1	1	1	0	1	0	14712146
4	0	1	1	0	1	0	14706980
5	1	1	0	0	1	0	15607358
6	2	1	0	0	1	0	15630520
7	3	1	0	0	1	0	15502245
8	2	1	1	0	1	0	14712495
9	1	1	2	0	1	0	14712241
10	2	1	2	0	1	0	14712945
11	2	2	2	0	1	0	15570062
12	1	2	1	0	1	0	72636438
Cluster 7							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**

1	0	0	0	0	1	0	9497469
2	0	1	0	0	1	0	11926588
3	1	1	1	0	1	0	14642068
4	0	1	1	0	1	0	14499379
5	1	1	0	0	1	0	12232790
6	2	1	0	0	1	0	12424136
7	3	1	0	0	1	0	12671973
8	2	1	1	0	1	0	14615494
9	1	1	2	0	1	0	14630146
10	2	1	2	0	1	0	14450813
11	2	2	2	0	1	0	59485435
12	1	2	1	0	1	0	54965086

A.4 545 DAY VALIDATION RESULTS

This section shows the optimal hyper parameters obtained using the 545 day validation period for the various models.

Table A.15 Hyper parameter selection for single model using 545 day validation results

Model order	p	d	q	P	D	Q	RMSE [R]
1	0	0	0	0	1	0	202017945
2	0	1	0	0	1	0	73681789
3	1	1	1	0	1	0	74530256
4	0	1	1	0	1	0	74174688
5	1	1	0	0	1	0	73710742
6	2	1	0	0	1	0	73680775
7	3	1	0	0	1	0	73719278
8	2	1	1	0	1	0	74337604
9	1	1	2	0	1	0	74467629
10	2	1	2	0	1	0	115114608
11	2	2	2	0	1	0	2029244208
12	1	2	1	0	1	0	447594212

Table A.16 Hyper parameter selection for two cluster hybrid model using 545 day validation results

Cluster 1							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	91575240
2	0	1	0	0	1	0	54867290
3	1	1	1	0	1	0	54931840
4	0	1	1	0	1	0	55233630
5	1	1	0	0	1	0	55193770
6	2	1	0	0	1	0	55154140
7	3	1	0	0	1	0	55281000
8	2	1	1	0	1	0	55029810
9	1	1	2	0	1	0	54974060
10	2	1	2	0	1	0	47108970
11	2	2	2	0	1	0	192761000
12	1	2	1	0	1	0	174972400
Cluster 2							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	118237800
2	0	1	0	0	1	0	65591040
3	1	1	1	0	1	0	70345900
4	0	1	1	0	1	0	68610090
5	1	1	0	0	1	0	65197130
6	2	1	0	0	1	0	65353330
7	3	1	0	0	1	0	64840410
8	2	1	1	0	1	0	69745620
9	1	1	2	0	1	0	70025690
10	2	1	2	0	1	0	66558860
11	2	2	2	0	1	0	518496100
12	1	2	1	0	1	0	104545000

Table A.17 Hyper parameter selection for third cluster hybrid model using 545 day validation results

Cluster 1							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	90380030
2	0	1	0	0	1	0	52770430
3	1	1	1	0	1	0	53949460
4	0	1	1	0	1	0	53080090
5	1	1	0	0	1	0	52377790
6	2	1	0	0	1	0	52399720
7	3	1	0	0	1	0	51820930
8	2	1	1	0	1	0	53690420
9	1	1	2	0	1	0	53795830
10	2	1	2	0	1	0	53120210
11	2	2	2	0	1	0	147302300
12	1	2	1	0	1	0	42923130
Cluster 2							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	66484310
2	0	1	0	0	1	0	53806850
3	1	1	1	0	1	0	53816830
4	0	1	1	0	1	0	53852640
5	1	1	0	0	1	0	53865370
6	2	1	0	0	1	0	55181520
7	3	1	0	0	1	0	44712970
8	2	1	1	0	1	0	44511990
9	1	1	2	0	1	0	42989720
10	2	1	2	0	1	0	47364410
11	2	2	2	0	1	0	263352100
12	1	2	1	0	1	0	40273350
Cluster 3							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	56938590
2	0	1	0	0	1	0	25010790
3	1	1	1	0	1	0	27220840
4	0	1	1	0	1	0	26267360
5	1	1	0	0	1	0	24908520
6	2	1	0	0	1	0	24948500

7	3	1	0	0	1	0	24985880
8	2	1	1	0	1	0	26845580
9	1	1	2	0	1	0	27034400
10	2	1	2	0	1	0	25302740
11	2	2	2	0	1	0	655527600
12	1	2	1	0	1	0	243247400

Table A.18 Hyper parameter selection for four cluster hybrid model using 545 day validation results

Cluster 1							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	56115790
2	0	1	0	0	1	0	25999560
3	1	1	1	0	1	0	26317700
4	0	1	1	0	1	0	26079170
5	1	1	0	0	1	0	25999150
6	2	1	0	0	1	0	25961580
7	3	1	0	0	1	0	25938240
8	2	1	1	0	1	0	26231580
9	1	1	2	0	1	0	26270920
10	2	1	2	0	1	0	25929730
11	2	2	2	0	1	0	259912600
12	1	2	1	0	1	0	208274800
Cluster 2							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	53302810
2	0	1	0	0	1	0	24796350
3	1	1	1	0	1	0	24780950
4	0	1	1	0	1	0	24789570
5	1	1	0	0	1	0	115591000
6	2	1	0	0	1	0	23680470
7	3	1	0	0	1	0	26717090
8	2	1	1	0	1	0	24043720
9	1	1	2	0	1	0	23884410
10	2	1	2	0	1	0	23505550
11	2	2	2	0	1	0	25096270
12	1	2	1	0	1	0	28862540

Cluster 3							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	30050380
2	0	1	0	0	1	0	50011610
3	1	1	1	0	1	0	51154940
4	0	1	1	0	1	0	51288620
5	1	1	0	0	1	0	50338630
6	2	1	0	0	1	0	50361040
7	3	1	0	0	1	0	50518390
8	2	1	1	0	1	0	51262610
9	1	1	2	0	1	0	51202910
10	2	1	2	0	1	0	54451120
11	2	2	2	0	1	0	198537700
12	1	2	1	0	1	0	33872520
Cluster 4							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	77277610
2	0	1	0	0	1	0	52395280
3	1	1	1	0	1	0	53257880
4	0	1	1	0	1	0	51514260
5	1	1	0	0	1	0	51664780
6	2	1	0	0	1	0	51448370
7	3	1	0	0	1	0	50597920
8	2	1	1	0	1	0	52791330
9	1	1	2	0	1	0	52946530
10	2	1	2	0	1	0	51683900
11	2	2	2	0	1	0	36944160
12	1	2	1	0	1	0	38257950

Table A.19 Hyper parameter selection for five cluster hybrid model using 545 day validation results

Cluster 1							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	50088316
2	0	1	0	0	1	0	26369700
3	1	1	1	0	1	0	26651338
4	0	1	1	0	1	0	26604292

5	1	1	0	0	1	0	26379731
6	2	1	0	0	1	0	26423237
7	3	1	0	0	1	0	26440276
8	2	1	1	0	1	0	26642097
9	1	1	2	0	1	0	26646258
10	2	1	2	0	1	0	26588258
11	2	2	2	0	1	0	64632675
12	1	2	1	0	1	0	75836668
Cluster 2							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	37968925
2	0	1	0	0	1	0	14757904
3	1	1	1	0	1	0	15249746
4	0	1	1	0	1	0	14859714
5	1	1	0	0	1	0	14556614
6	2	1	0	0	1	0	14505891
7	3	1	0	0	1	0	14513209
8	2	1	1	0	1	0	15086574
9	1	1	2	0	1	0	15178617
10	2	1	2	0	1	0	14270897
11	2	2	2	0	1	0	230888548
12	1	2	1	0	1	0	197791555
Cluster 3							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	40896980
2	0	1	0	0	1	0	19029020
3	1	1	1	0	1	0	19024800
4	0	1	1	0	1	0	19023740
5	1	1	0	0	1	0	19160180
6	2	1	0	0	1	0	19280940
7	3	1	0	0	1	0	19053610
8	2	1	1	0	1	0	20271360
9	1	1	2	0	1	0	19066740
10	2	1	2	0	1	0	19192160
11	2	2	2	0	1	0	28205830
12	1	2	1	0	1	0	27922080
Cluster 4							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	25845520

2	0	1	0	0	1	0	50172240
3	1	1	1	0	1	0	50503240
4	0	1	1	0	1	0	50356900
5	1	1	0	0	1	0	50277860
6	2	1	0	0	1	0	50272640
7	3	1	0	0	1	0	50378690
8	2	1	1	0	1	0	50392500
9	1	1	2	0	1	0	50463370
10	2	1	2	0	1	0	50278260
11	2	2	2	0	1	0	142120800
12	1	2	1	0	1	0	15771940
Cluster 5							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	63020087
2	0	1	0	0	1	0	45203047
3	1	1	1	0	1	0	49245808
4	0	1	1	0	1	0	43789518
5	1	1	0	0	1	0	44210930
6	2	1	0	0	1	0	43700077
7	3	1	0	0	1	0	42902434
8	2	1	1	0	1	0	49191609
9	1	1	2	0	1	0	49182502
10	2	1	2	0	1	0	49202524
11	2	2	2	0	1	0	360889508
12	1	2	1	0	1	0	53678336

Table A.20 Hyper parameter selection for six cluster hybrid model using 545 day validation results

Cluster 1							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	25470180
2	0	1	0	0	1	0	43603290
3	1	1	1	0	1	0	44299110
4	0	1	1	0	1	0	44035350
5	1	1	0	0	1	0	43791710
6	2	1	0	0	1	0	43866280
7	3	1	0	0	1	0	44064090
8	2	1	1	0	1	0	44199690
9	1	1	2	0	1	0	44259270

10	2	1	2	0	1	0	44171850
11	2	2	2	0	1	0	357170500
12	1	2	1	0	1	0	38134790
Cluster 2							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	19780880
2	0	1	0	0	1	0	12611190
3	1	1	1	0	1	0	12964840
4	0	1	1	0	1	0	13063060
5	1	1	0	0	1	0	12936190
6	2	1	0	0	1	0	13104780
7	3	1	0	0	1	0	13294030
8	2	1	1	0	1	0	12978180
9	1	1	2	0	1	0	12963060
10	2	1	2	0	1	0	13295290
11	2	2	2	0	1	0	198596700
12	1	2	1	0	1	0	12267320
Cluster 3							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	21231280
2	0	1	0	0	1	0	26931790
3	1	1	1	0	1	0	30863070
4	0	1	1	0	1	0	27012200
5	1	1	0	0	1	0	31116980
6	2	1	0	0	1	0	30211830
7	3	1	0	0	1	0	27072940
8	2	1	1	0	1	0	26622700
9	1	1	2	0	1	0	25655770
10	2	1	2	0	1	0	24157890
11	2	2	2	0	1	0	13108990
12	1	2	1	0	1	0	24320280
Cluster 4							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R**
1	0	0	0	0	1	0	48927190
2	0	1	0	0	1	0	28900670
3	1	1	1	0	1	0	29211610
4	0	1	1	0	1	0	29180570
5	1	1	0	0	1	0	28992970
6	2	1	0	0	1	0	29040240

7	3	1	0	0	1	0	29317010
8	2	1	1	0	1	0	29242160
9	1	1	2	0	1	0	29177090
10	2	1	2	0	1	0	29308830
11	2	2	2	0	1	0	792547800
12	1	2	1	0	1	0	610641000
Cluster 5							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	46545340
2	0	1	0	0	1	0	25587920
3	1	1	1	0	1	0	25564480
4	0	1	1	0	1	0	25554770
5	1	1	0	0	1	0	25531280
6	2	1	0	0	1	0	25515080
7	3	1	0	0	1	0	25519350
8	2	1	1	0	1	0	25562170
9	1	1	2	0	1	0	25563550
10	2	1	2	0	1	0	25564320
11	2	2	2	0	1	0	141623800
12	1	2	1	0	1	0	46544960
Cluster 6							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	60549110
2	0	1	0	0	1	0	45496100
3	1	1	1	0	1	0	47674850
4	0	1	1	0	1	0	43114620
5	1	1	0	0	1	0	44448480
6	2	1	0	0	1	0	43903590
7	3	1	0	0	1	0	43045090
8	2	1	1	0	1	0	47673790
9	1	1	2	0	1	0	47673930
10	2	1	2	0	1	0	47906620
11	2	2	2	0	1	0	77889260

Table A.21 Hyper parameter selection for seven cluster hybrid model using 545 day validation results

Cluster 1							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	45181860
2	0	1	0	0	1	0	25942300
3	1	1	1	0	1	0	26444850
4	0	1	1	0	1	0	26738090
5	1	1	0	0	1	0	26205080
6	2	1	0	0	1	0	26460350
7	3	1	0	0	1	0	26403110
8	2	1	1	0	1	0	26572360
9	1	1	2	0	1	0	26517620
10	2	1	2	0	1	0	26711590
11	2	2	2	0	1	0	113957200
12	1	2	1	0	1	0	59693510
Cluster 2							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	45181860
2	0	1	0	0	1	0	25942300
3	1	1	1	0	1	0	26444850
4	0	1	1	0	1	0	26738090
5	1	1	0	0	1	0	26205080
6	2	1	0	0	1	0	26460350
7	3	1	0	0	1	0	26403110
8	2	1	1	0	1	0	26572360
9	1	1	2	0	1	0	26517620
10	2	1	2	0	1	0	26711590
11	2	2	2	0	1	0	113957200
12	1	2	1	0	1	0	59693510
Cluster 3							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	47918210
2	0	1	0	0	1	0	35163840
3	1	1	1	0	1	0	35406080
4	0	1	1	0	1	0	35423790
5	1	1	0	0	1	0	35320450
6	2	1	0	0	1	0	35343330

7	3	1	0	0	1	0	35570630
8	2	1	1	0	1	0	35416420
9	1	1	2	0	1	0	35377870
10	2	1	2	0	1	0	35473320
11	2	2	2	0	1	0	181548500
12	1	2	1	0	1	0	452864400
Cluster 4							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	49088180
2	0	1	0	0	1	0	29715080
3	1	1	1	0	1	0	31132010
4	0	1	1	0	1	0	27653480
5	1	1	0	0	1	0	29068470
6	2	1	0	0	1	0	28811650
7	3	1	0	0	1	0	28380280
8	2	1	1	0	1	0	30253530
9	1	1	2	0	1	0	35291730
10	2	1	2	0	1	0	37435860
11	2	2	2	0	1	0	41593070
12	1	2	1	0	1	0	57028030
Cluster 5							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	31295390
2	0	1	0	0	1	0	61098710
3	1	1	1	0	1	0	29145360
4	0	1	1	0	1	0	61090660
5	1	1	0	0	1	0	60333640
6	2	1	0	0	1	0	63534410
7	3	1	0	0	1	0	Error can not solve
8	2	1	1	0	1	0	Error can not solve
9	1	1	2	0	1	0	Error can not solve
10	2	1	2	0	1	0	Error can not solve
11	2	2	2	0	1	0	Error can not solve
12	1	2	1	0	1	0	Error can not solve
Cluster 6							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	27065200
2	0	1	0	0	1	0	14875790
3	1	1	1	0	1	0	15821440

4	0	1	1	0	1	0	15292420
5	1	1	0	0	1	0	14880340
6	2	1	0	0	1	0	14861690
7	3	1	0	0	1	0	14858750
8	2	1	1	0	1	0	15620620
9	1	1	2	0	1	0	15710170
10	2	1	2	0	1	0	15197050
11	2	2	2	0	1	0	479414900
12	1	2	1	0	1	0	115903500
Cluster 7							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	9206325
2	0	1	0	0	1	0	15794870
3	1	1	1	0	1	0	11826000
4	0	1	1	0	1	0	12016880
5	1	1	0	0	1	0	14814310
6	2	1	0	0	1	0	14271660
7	3	1	0	0	1	0	13816290
8	2	1	1	0	1	0	11865460
9	1	1	2	0	1	0	11839810
10	2	1	2	0	1	0	12147970
11	2	2	2	0	1	0	44775580
12	1	2	1	0	1	0	36949180

A.5 730 DAY VALIDATIION RESULTS

This section shows the optimal hyper parameters obtained using the 730 day validation period for the various models.

Table A.22 Hyper parameter selection for single model using 730 day validation results

Single Model							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	217341778
2	0	1	0	0	1	0	104945482
3	1	1	1	0	1	0	90622377
4	0	1	1	0	1	0	105865834
5	1	1	0	0	1	0	1.51E+12

6	2	1	0	0	1	0	147618516
7	3	1	0	0	1	0	467912599
8	2	1	1	0	1	0	148001884
9	1	1	2	0	1	0	95010122
10	2	1	2	0	1	0	1.159E+09
11	2	2	2	0	1	0	9.321E+10
12	1	2	1	0	1	0	1.138E+09

Table A.23 Hyper parameter selection for two cluster hybrid model using 730 day validation results

Cluster 1							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	115954000
2	0	1	0	0	1	0	175110300
3	1	1	1	0	1	0	172134200
4	0	1	1	0	1	0	171357400
5	1	1	0	0	1	0	171874900
6	2	1	0	0	1	0	383605400
7	3	1	0	0	1	0	575243300
8	2	1	1	0	1	0	375230000
9	1	1	2	0	1	0	222303900
10	2	1	2	0	1	0	492316600
11	2	2	2	0	1	0	71049740000
12	1	2	1	0	1	0	1489787000
Cluster 2							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	114498254
2	0	1	0	0	1	0	98950240
3	1	1	1	0	1	0	94535546
4	0	1	1	0	1	0	92066165
5	1	1	0	0	1	0	99945011
6	2	1	0	0	1	0	98599374
7	3	1	0	0	1	0	98657715
8	2	1	1	0	1	0	93432327
9	1	1	2	0	1	0	94132469
10	2	1	2	0	1	0	93446686
11	2	2	2	0	1	0	861449501

12	1	2	1	0	1	0	835535978

Table A.24 Hyper parameter selection for three cluster hybrid model using 730 day validation results

Cluster 1							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	88350770
2	0	1	0	0	1	0	70587390
3	1	1	1	0	1	0	66075300
4	0	1	1	0	1	0	65384360
5	1	1	0	0	1	0	71101740
6	2	1	0	0	1	0	69852740
7	3	1	0	0	1	0	69852360
8	2	1	1	0	1	0	66084790
9	1	1	2	0	1	0	66069980
10	2	1	2	0	1	0	66070740
11	2	2	2	0	1	0	330208800
12	1	2	1	0	1	0	233483800
Cluster 2							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	94636930
2	0	1	0	0	1	0	188762200
3	1	1	1	0	1	0	187792000
4	0	1	1	0	1	0	189700600
5	1	1	0	0	1	0	216615100
6	2	1	0	0	1	0	295453000
7	3	1	0	0	1	0	314897700
8	2	1	1	0	1	0	263518600
9	1	1	2	0	1	0	165887100
10	2	1	2	0	1	0	228817500
11	2	2	2	0	1	0	2.79874E+13
12	1	2	1	0	1	0	8376620000
Cluster 3							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	53703170
2	0	1	0	0	1	0	40190470
3	1	1	1	0	1	0	39836070

4	0	1	1	0	1	0	38665950
5	1	1	0	0	1	0	41706260
6	2	1	0	0	1	0	41433350
7	3	1	0	0	1	0	41540930
8	2	1	1	0	1	0	39791700
9	1	1	2	0	1	0	39736530
10	2	1	2	0	1	0	39985590
11	2	2	2	0	1	0	692133200
12	1	2	1	0	1	0	478684700

Table A.25 Hyper parameter selection for four cluster hybrid model using 730 day validation results

Cluster 1							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	53547890
2	0	1	0	0	1	0	48764420
3	1	1	1	0	1	0	47840140
4	0	1	1	0	1	0	46142180
5	1	1	0	0	1	0	50001860
6	2	1	0	0	1	0	49328870
7	3	1	0	0	1	0	49335220
8	2	1	1	0	1	0	47484070
9	1	1	2	0	1	0	47661300
10	2	1	2	0	1	0	47606190
11	2	2	2	0	1	0	613540000
12	1	2	1	0	1	0	415127700
Cluster 2							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	58066160
2	0	1	0	0	1	0	93897120
3	1	1	1	0	1	0	93934090
4	0	1	1	0	1	0	92604170
5	1	1	0	0	1	0	90013370
6	2	1	0	0	1	0	125362000
7	3	1	0	0	1	0	116267000
8	2	1	1	0	1	0	95033160
9	1	1	2	0	1	0	109780900

10	2	1	2	0	1	0	94832520
11	2	2	2	0	1	0	4.82426E+18
12	1	2	1	0	1	0	47745170
Cluster 3							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	51563730
2	0	1	0	0	1	0	91298770
3	1	1	1	0	1	0	88268690
4	0	1	1	0	1	0	93517280
5	1	1	0	0	1	0	151556500
6	2	1	0	0	1	0	182333900
7	3	1	0	0	1	0	238562400
8	2	1	1	0	1	0	174668200
9	1	1	2	0	1	0	1.21772E+11
10	2	1	2	0	1	0	164009500
11	2	2	2	0	1	0	2.40401E+16
12	1	2	1	0	1	0	11353585000
Cluster 4							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	75051980
2	0	1	0	0	1	0	56089110
3	1	1	1	0	1	0	52643760
4	0	1	1	0	1	0	51978080
5	1	1	0	0	1	0	56827540
6	2	1	0	0	1	0	55950230
7	3	1	0	0	1	0	55954720
8	2	1	1	0	1	0	52588410
9	1	1	2	0	1	0	52669830
10	2	1	2	0	1	0	52564200
11	2	2	2	0	1	0	219537600
12	1	2	1	0	1	0	231887500

Table A.26 Hyper parameter selection for five cluster hybrid model using 730 day validation results

Cluster 1							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	49244310
2	0	1	0	0	1	0	57130140
3	1	1	1	0	1	0	54375410
4	0	1	1	0	1	0	53330230
5	1	1	0	0	1	0	57587880
6	2	1	0	0	1	0	56591150
7	3	1	0	0	1	0	56523480
8	2	1	1	0	1	0	54114170
9	1	1	2	0	1	0	53974840
10	2	1	2	0	1	0	54161430
11	2	2	2	0	1	0	207188900
12	1	2	1	0	1	0	260169100
Cluster 2							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	34246970
2	0	1	0	0	1	0	21044170
3	1	1	1	0	1	0	21697860
4	0	1	1	0	1	0	21639510
5	1	1	0	0	1	0	21603320
6	2	1	0	0	1	0	21699500
7	3	1	0	0	1	0	96360440
8	2	1	1	0	1	0	30353950
9	1	1	2	0	1	0	36233410
10	2	1	2	0	1	0	25751950
11	2	2	2	0	1	0	472860900
12	1	2	1	0	1	0	130361400
Cluster 3							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	45580830
2	0	1	0	0	1	0	83280810
3	1	1	1	0	1	0	83266710
4	0	1	1	0	1	0	82191030
5	1	1	0	0	1	0	80297710
6	2	1	0	0	1	0	111047600

7	3	1	0	0	1	0	102740600
8	2	1	1	0	1	0	88229710
9	1	1	2	0	1	0	89630250
10	2	1	2	0	1	0	82777320
11	2	2	2	0	1	0	5.41377E+18
12	1	2	1	0	1	0	147225900
Cluster 4							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	55079250
2	0	1	0	0	1	0	117448900
3	1	1	1	0	1	0	124069400
4	0	1	1	0	1	0	120621000
5	1	1	0	0	1	0	151027100
6	2	1	0	0	1	0	185793800
7	3	1	0	0	1	0	192949400
8	2	1	1	0	1	0	148923400
9	1	1	2	0	1	0	174318000
10	2	1	2	0	1	0	162504100
11	2	2	2	0	1	0	2.62272E+14
12	1	2	1	0	1	0	12426804000
Cluster 5							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	59639380
2	0	1	0	0	1	0	41489690
3	1	1	1	0	1	0	39088830
4	0	1	1	0	1	0	38432260
5	1	1	0	0	1	0	41648750
6	2	1	0	0	1	0	40947460
7	3	1	0	0	1	0	40905690
8	2	1	1	0	1	0	107864100
9	1	1	2	0	1	0	39049330
10	2	1	2	0	1	0	38867550
11	2	2	2	0	1	0	142263600
12	1	2	1	0	1	0	130236700

Table A.27 Hyper parameter selection for six cluster hybrid model using 730 day validation results

Cluster 1							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	50944130
2	0	1	0	0	1	0	116564800
3	1	1	1	0	1	0	Can not solve
4	0	1	1	0	1	0	Can not solve
5	1	1	0	0	1	0	Can not solve
6	2	1	0	0	1	0	Can not solve
7	3	1	0	0	1	0	Can not solve
8	2	1	1	0	1	0	Can not solve
9	1	1	2	0	1	0	Can not solve
10	2	1	2	0	1	0	Can not solve
11	2	2	2	0	1	0	Can not solve
12	1	2	1	0	1	0	Can not solve
Cluster 2							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	24847390
2	0	1	0	0	1	0	16870790
3	1	1	1	0	1	0	17203460
4	0	1	1	0	1	0	17757150
5	1	1	0	0	1	0	15636080
6	2	1	0	0	1	0	15626450
7	3	1	0	0	1	0	15542660
8	2	1	1	0	1	0	17430700
9	1	1	2	0	1	0	17243170
10	2	1	2	0	1	0	17428280
11	2	2	2	0	1	0	332564000
12	1	2	1	0	1	0	215718600
Cluster 3							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	33711880
2	0	1	0	0	1	0	100927400
3	1	1	1	0	1	0	87506930
4	0	1	1	0	1	0	99475580
5	1	1	0	0	1	0	85705030
6	2	1	0	0	1	0	98018730
7	3	1	0	0	1	0	93723500

8	2	1	1	0	1	0	88421390
9	1	1	2	0	1	0	101692300
10	2	1	2	0	1	0	91342510
11	2	2	2	0	1	0	242937400
12	1	2	1	0	1	0	133560100
Cluster 4							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	42106700
2	0	1	0	0	1	0	44580680
3	1	1	1	0	1	0	34409930
4	0	1	1	0	1	0	44096010
5	1	1	0	0	1	0	32923470
6	2	1	0	0	1	0	49080900
7	3	1	0	0	1	0	83143620
8	2	1	1	0	1	0	37082840
9	1	1	2	0	1	0	30510860
10	2	1	2	0	1	0	33148500
11	2	2	2	0	1	0	1.97908E+14
12	1	2	1	0	1	0	33975785000
Cluster 5							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	46352070
2	0	1	0	0	1	0	52917650
3	1	1	1	0	1	0	52960050
4	0	1	1	0	1	0	51098990
5	1	1	0	0	1	0	53929060
6	2	1	0	0	1	0	53307130
7	3	1	0	0	1	0	53283180
8	2	1	1	0	1	0	52549810
9	1	1	2	0	1	0	52787730
10	2	1	2	0	1	0	52576410
11	2	2	2	0	1	0	175081000
12	1	2	1	0	1	0	233230100
Cluster 6							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	57368950
2	0	1	0	0	1	0	40083240
3	1	1	1	0	1	0	36965320
4	0	1	1	0	1	0	36525120

5	1	1	0	0	1	0	39692660
6	2	1	0	0	1	0	38965410
7	3	1	0	0	1	0	38933970
8	2	1	1	0	1	0	124527800
9	1	1	2	0	1	0	37035230
10	2	1	2	0	1	0	36771170
11	2	2	2	0	1	0	162561900
12	1	2	1	0	1	0	149123900

Table A.28 Hyper parameter selection for seven cluster hybrid model using 730 day validation results

Cluster 1							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	47191340
2	0	1	0	0	1	0	44505550
3	1	1	1	0	1	0	45004940
4	0	1	1	0	1	0	42635530
5	1	1	0	0	1	0	47002500
6	2	1	0	0	1	0	46572380
7	3	1	0	0	1	0	46594260
8	2	1	1	0	1	0	43251590
9	1	1	2	0	1	0	44393240
10	2	1	2	0	1	0	43257620
11	2	2	2	0	1	0	224719500
12	1	2	1	0	1	0	246408800
Cluster 2							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	17589370
2	0	1	0	0	1	0	78268000
3	1	1	1	0	1	0	61219230
4	0	1	1	0	1	0	76738020
5	1	1	0	0	1	0	59871300
6	2	1	0	0	1	0	63887850
7	3	1	0	0	1	0	65908110
8	2	1	1	0	1	0	60829360
9	1	1	2	0	1	0	65800120
10	2	1	2	0	1	0	60395240

11	2	2	2	0	1	0	283182900
12	1	2	1	0	1	0	134483500
Cluster 3							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	41443250
2	0	1	0	0	1	0	49557260
3	1	1	1	0	1	0	47659740
4	0	1	1	0	1	0	48705410
5	1	1	0	0	1	0	34120090
6	2	1	0	0	1	0	41277990
7	3	1	0	0	1	0	72715700
8	2	1	1	0	1	0	34712390
9	1	1	2	0	1	0	48377260
10	2	1	2	0	1	0	37271850
11	2	2	2	0	1	0	4.66529E+16
12	1	2	1	0	1	0	2.16966E+11
Cluster 4							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	46321580
2	0	1	0	0	1	0	30143150
3	1	1	1	0	1	0	28401000
4	0	1	1	0	1	0	28385370
5	1	1	0	0	1	0	29458650
6	2	1	0	0	1	0	29161070
7	3	1	0	0	1	0	29128710
8	2	1	1	0	1	0	28602460
9	1	1	2	0	1	0	28465490
10	2	1	2	0	1	0	28587650
11	2	2	2	0	1	0	201167300
12	1	2	1	0	1	0	298695700
Cluster 5							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	57622910
2	0	1	0	0	1	0	128286700
3	1	1	1	0	1	0	129217000
4	0	1	1	0	1	0	129326400
5	1	1	0	0	1	0	125279100
6	2	1	0	0	1	0	145621700
7	3	1	0	0	1	0	149089200

8	2	1	1	0	1	0	123998000
9	1	1	2	0	1	0	78231770
10	2	1	2	0	1	0	129701600
11	2	2	2	0	1	0	8.11549E+15
12	1	2	1	0	1	0	118505400
Cluster 6							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	27178340
2	0	1	0	0	1	0	21013120
3	1	1	1	0	1	0	17589460
4	0	1	1	0	1	0	17469680
5	1	1	0	0	1	0	21610500
6	2	1	0	0	1	0	21022990
7	3	1	0	0	1	0	20810380
8	2	1	1	0	1	0	17599170
9	1	1	2	0	1	0	17600720
10	2	1	2	0	1	0	17595010
11	2	2	2	0	1	0	308108600
12	1	2	1	0	1	0	276385200
Cluster 7							
Model order	**p**	**d**	**q**	**P**	**D**	**Q**	**RMSE [R]**
1	0	0	0	0	1	0	22783530
2	0	1	0	0	1	0	25375120
3	1	1	1	0	1	0	23686750
4	0	1	1	0	1	0	23534440
5	1	1	0	0	1	0	24167670
6	2	1	0	0	1	0	23732940
7	3	1	0	0	1	0	99123430
8	2	1	1	0	1	0	25436550
9	1	1	2	0	1	0	42452660
10	2	1	2	0	1	0	65414000
11	2	2	2	0	1	0	179564300
12	1	2	1	0	1	0	487011600