ATM Cash Stock Prediction Using Different Machine Learning Approaches

by
D. ECE GÖKÇAY

Submitted to the Graduate School of Computer Science
in partial fulfilment of
the requirements for the degree of Master of Science

Sabancı University
December 2020

ATM CASH STOCK PREDICTION USING DIFFERENT MACHINE LEARNING APPROACHES

Approved by:

Date of Approval: December 25, 2020

Abstract

ATM CASH STOCK PREDICTION USING DIFFERENT MACHINE LEARNING APPROACHES

D. ECE GÖKÇAY

Computer Science M.S. THESIS, DECEMBER 2021

Thesis Supervisor: Prof. Berrin A. Yanıkoğlu
Thesis Co-Supervisor: Assoc. Prof. Abdullah Daşçı

Keywords: ATM stock prediction, Regression, Linear Regression, Support Vector Machines, Artificial Neural Networks, LSTM, ARIMA, Machine Learning

One of the most common problems related to banking systems is the Automated Teller Machine (ATM) cash demand forecasting. Cash shortage adversely affects customer satisfaction, while too much cash reduces bank's profitability. We have developed an ATM cash prediction system using different traditional statistical and machine learning approaches, including linear regression, support vector machines, artificial neural networks, LSTMs and traditional statistical analysis (ARIMA) on the same ATM data. We compared the results of these methods and showed that machine learning methods in comparison with ARIMA have higher accuracy. Also it was shown that among the machine learning models, LSTM gives the most accurate predictions and use less features compared to other models.

ÖZET

FARKLI MAKINE ÖĞRENMESI YÖNTEMLERI ILE ATM PARA ÇEKIM MIKTARI TAHMINI

D. ECE GÖKÇAY

Bilgisayar Bilimi ve Mühendisliği, Yüksek Lisans Tezi, ARALIK 2021

Tez Danışmanı: Prof. Dr. Berrin A. Yanıkoğlu
Tez Eş Danışmanı: Assoc. Prof. Abdullah Daşçı

Anahtar Kelimeler: ATM Nakit Tahmini, Regresyon, Doğrusal Regresyon, Destek Vektör Makinesi, Yapay Sinir Ağları, Uzun-Kısa Vadeli Hafıza Ağları, ARIMA, Makine Öğrenmesi

ATM nakit tahmini, banka sistemlerindeki en yaygın problemlerden biridir. ATM'de yeterli nakit bulunmaması, müşteri memnuniyetini azaltırken, gereğinden fazla para olması ise bankanın kar payını negatif etkiler. Bu çalışmada ATM'lerden çekilen para miktarını tahmin eden bir sistem geliştirilmiştir. Tahmin aşamasında doğrusal regresyon, destek vektör makinesi, yapay sinir ağları, derin öğrenme tekniklerinden olan Uzun Kısa Vadeli Hafıza Ağları ve istatiksel analiz (ARIMA) methodları kullanılarak modeller oluşturulmuş ve aynı ATM verisi üzerinde deneyler yapılmıştır.
Bu deneyler sonucunda makine öğrenmesi metotlarının, kullanılan istatiksel methoda göre çok daha iyi performans sergilediği gösterilmiştir. Ayrıca, makine öğrenmesi metotları içerisinde de LSTM modelinin çok daha az öznitelik kullanarak daha başarılı tahminler yaptığı belirlenmiştir.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATONS

ANN Artificial Neural Network

AR Auto Regressive

ARIMA Auto Regressive Integrated Moving Average

ATM Automated Teller Machine

CNN Convolutional Neural Network

GMDH General Group Method of Data Handling

GRNN General Regression Neural Network

LR Linear Regression

LSTM Long- Short Term Memory

MA Moving Average

MAE Mean Absolute Error

MAPE Mean Absolute Percentage Error

MSE Mean Squared Error

MSE Adaptive Moment Estimation

RBF Radial Basis Function

ReLu Rectified Linear Unit

RNN Recurrent Neural Network

SGD Stochastic Gradient Descent

SMAPE Symmetric mean absolute percentage

SVM Support Vector Machine

SVR Support Vector Regression

1. INTRODUCTION

ATMs are powerful interaction and communication points between banks and customers. By storing cash in ATMs, banks provide services to customers 24 hours a day, 7 days a week, regardless of whether they have an account in this bank. Almost every People can withdraw or deposit money from ATMs around the world, regardless of which bank their account is in, through the fact that ATMs are connected to universal bank networks. According to research conducted by The Interbank Card Center, while the number of ATMs was 51,941 across Turkey in 2018, it is 53,075 in 2020.(BKM, 2020)

The lack of sufficient cash in the ATM reduces the customer satisfaction by damaging the popularity of the bank, and the excessive amount of money will negatively affect the profit share of the bank as it decreases the availability of money. Effective currency management and control enables banks to predict demand and proactively manage the currency across their networks, with advanced algorithms to accurately predict money supply and demand.

Before the optimization of ATM cash management, banks used to manage ATM cash manually based on personal experiences and operating policies. Banks had deposited too much cash in ATMs to guarantee access to cash by preventing out of service. Even after the optimization, many banks hold 40% more cash in their ATMs than usually required, experts have advised between 15% and 20% of this excess cash is sufficient (Bilir & Doseyen, 2018).

1.1 Structure of the Thesis

The remaining of this thesis is organized as follows.

Chapter 2 provides a literature survey of the studies focused on the forecasting of withdrawal amount via machine learning regression models. Chapter 3 describes the datasets and basic knowledge about regression models that we used in the thesis. Chapter 4 goes over our experiment step by step, then shows and compares the result of the experiment. Chapter 5 provides summary and conclusions.

2. Review

Forecasting and optimization are two well-researched areas, however research on forecasting ATM withdrawal amounts is quite limited in number.

We found that there are two categories of publication about ATM cash optimization in Turkey and internationally. These are;

- optimization the daily amount of cash withdrawn from ATMs
- optimization the cash of recycled ATM that can withdraw and deposit money

Our study has been based on estimating the amount of cash withdrawn on a daily basis and unfortunately, there are only few similar studies on this subject. Publications similar to our study were examined.

Simutis et al. created a flexible neural network model with 3 years' data of an ATM from a bank in Lithuania for forecasting. (Simutis et al., 2007) For training, they used a three-layer feed-forward neural network trained with the Levenberg-Marquardt algorithm and historical data. Then the model was retuned every week with the last week's observations from ATM. On testing, a simulation dataset for ATM was produced. The dataset was simulated using weekly and monthly seasonality pattern and emulating the cash withdrawals from an ATM in Kaunas. Performed simulation and experimental tests have shown that the model achieved 1,5 - 2% MAPE for daily cash demand prediction on various simulation runs. On the other hand, when the model tested on the real ATM dataset, reached 15-20% MAPE.

Andrawis et al. used the concept of forecast combination and weekly seasonality, day of the month seasonality, and month of the year seasonality in their models for creating models. (Andrawis et al., 2011) In their approach, they chose the best 9 out of 140 models / preprocessing combinations which they produced for new combinations. These selections were based on comparison studies in the literature and time-series competitions. These new combinations were measured on the NN5 competition dataset. The best model among the 9 models with 18.95% SMAPE,

was created by combining linear models, neural networks, and Gaussian process regression with a simple mean.

In the study of Venkatesh et al., they advocate cash demand forecasting for groups of ATMs that share certain characteristics within the group, rather than forecasting cash for a single ATM. (Venkatesh et al., 2014) They divide ATMs into groups to have similar day-of-week withdrawal patterns. After this preparation, they estimated the amount of withdrawing with weekly seasonality parameters. They built 4 models via GMDH, GRNN, multi-layer feed-forward neural network (MLFF), and wavelet neural network (WNN) for predictions, and they measured these models on the NN5 competition dataset. They recognized that GRNN achieved the best result of 18.44% SMAPE among all their models.

In the study of Ekinci et al., they suggested clustering ATMs based on location information and optimizing the summation of daily cash withdraws in the forecasting process. (Ekinci et al., 2015) They grouped the ATMs using a nearest-neighbourhood approach based on distances and calculated the daily average cash withdrawals for each group. They develop linear regression models for cash demand forecasting for each group. When creating the model, they used information related to location and date (such as a categorical variable which denotes that the week is the week before holidays, a numeric variable which denotes the number of work-related buildings in the street). Lastly, they transformed the cluster forecast to individual one and predicted individual forecast amount to each ATM in the group. In their work, they used 152 ATMs located in Istanbul from a private bank, and they separate them into 27 groups based on latitudes and longitude. In this work, 22.69% Mean Absolute Error was obtained.

Serengil and Ozpınar also worked on prediction of daily withdrawal amount with Neural Network. (Serengil & Ozpinar, 2019) Their neural network model included 3 layers, and they fed the model with date's information. Their model served on 6500 individual ATMs all over Turkey. They reported that they reduced their cash expenses by 30% with ATM cash estimation and optimization on a data of 41 ATMs.

In Sarveswararao and Ravi's study, they generated forecasting models and measured them on 100 ATMs in India. (Vangala & Vadlamani, 2020) They formed the chaos in their datasets by inserting a dimension found with auto-correlation and Cao's method, and rebuilding the state space of the dataset with the delay. Also, they add information of day on the dataset for forecasting. They used ARIMA, Random Forest, SVR, ANN, GMDH, GRNN, LSTM and CNN methods for forecasting models. Their experiments showed that after creating chaos and adding day features to the dataset, all models achievement improved in the forecasting. Although

the RF gives a better SMAPE value (21.48%), LSTM, CNN and RF gave similar performance based on t-test respectively 0.32, 0.34, 0.34.

In this study, a system that makes ATM cash estimation using data from a private bank has been implemented. The most important difference from the above studies is that the error entries caused by physical problems in ATMs have not been ignored. The reason for this is to measure the responses of the system in real situations as much as possible.

3. Material & Methods

In our study, different regression models have been developed to estimate the daily cash amount withdrawn from ATM. Before explaining these models in detail, we will describe the dataset we use in our system.

3.1 Dataset

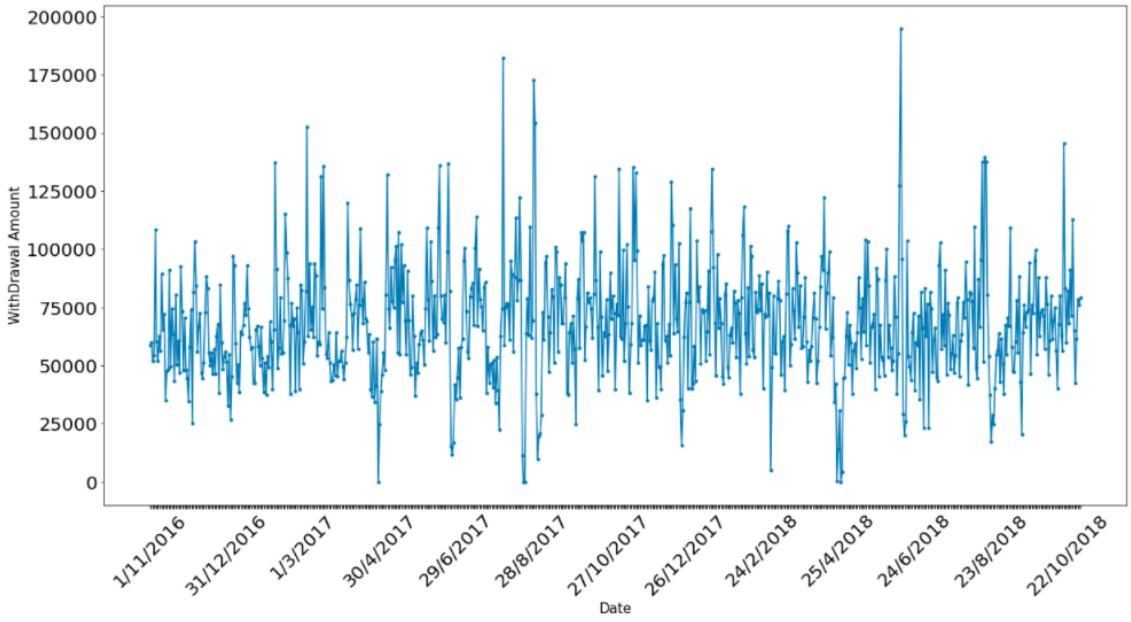
In the forecasting project, using real data is really important. The reason for this is that the forecasting models created are intended to be trained in the best way to respond to situations encountered in real life. The ATM data we used in our work is the ATM cash withdrawals from 2 different ATMs from the same private bank, in the periods of 1/11/2016 - 1/11/2018 (2-year period) and 7/12/2016 - 14/11/2018 (11-months period), respectively. This ATMs located in the same city but in different districts.

Before the preprocessing on datasets, we examined the distribution of the withdraw amount and some statistical data related to it, which can be seen in Table 3.1.

Table 3.1 Details about the datasets

	ATM_1	ATM_2
Number of record	731	708
Mean of Withdrawal Amount Daily (TL)	67,614	17,956
Standard Deviation of Withdraw Amount	25,001	10,053
Median of Withdrawal Amount Daily (TL)	64,950	16,505
Minimum Withdrawal Amount Daily (TL)	0	0
Maximum Withdrawal Amount Daily (TL)	194,980	60,170

Figure 3.1 Withdraw Amount Distribution from ATM_1



As can be seen in Table 3.1, our record of ATM_1 was obtained from an ATM that have much more withdrawal amounts on average compared to our ATM_2. While the average withdrawal amount in our ATM_1 is 67,614 TL, in our ATM_2 this amount is 17,956.

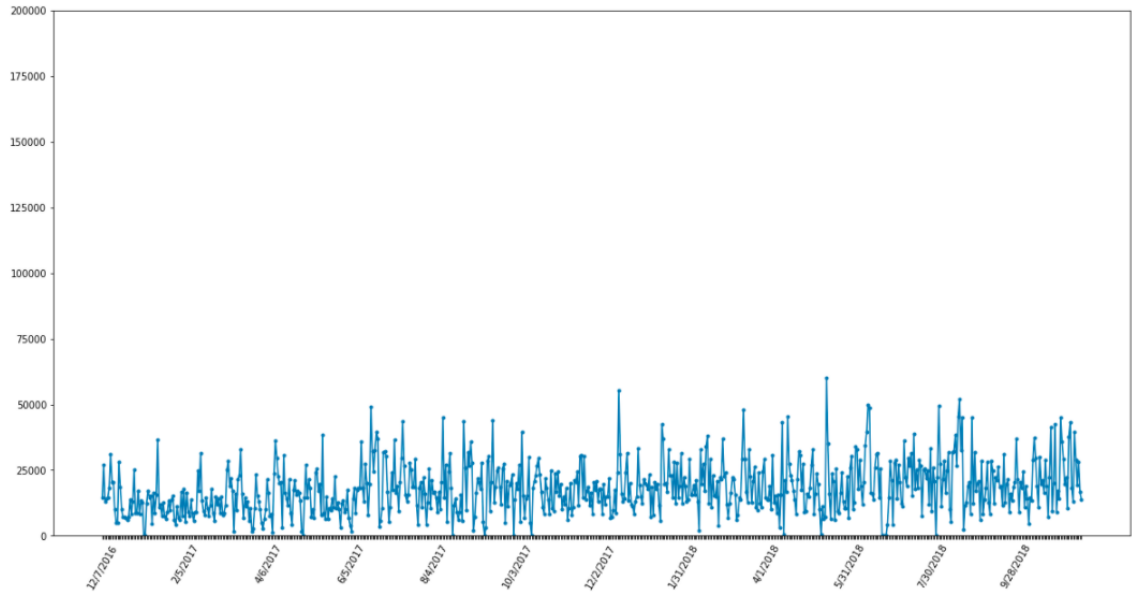
In Figures 3.1 and 3.2, the distribution of withdrawal amount by days is shown. As can be seen in these figures, it is thought that the amount of withdrawn cash in our ATM sets follows a weekly pattern.

In both raw ATM sets, besides the daily withdrawal amount, there are also 104 pieces of information about that day. For example; the dataset includes information on which day of the week it is, whether it is counted as a holiday, and which business day it corresponds to. Some of these attributes are shown as a categorical, some (eg. days of the week) as an numerical variable.

One of the most important steps in ATM prediction is exploratory data analysis. When we analyzed the withdrawal cash amounts in our dataset, we realized some extraordinarily low values. When we conveyed this situation to the project manager, they informed that ATMs did not work full time in those records due to some technical problems. When the dataset is used with these errors, the error rate of the model is calculated high because the withdrawal amount is lower than it should be. On the other hand, when models are created without these data, the system becomes far from real life.

In this study, we chose not to ignore the records because we wanted to keep it as

Figure 3.2 Withdraw Amount Distribution from ATM_2



close to real-life as we know that our system will decrease its success.

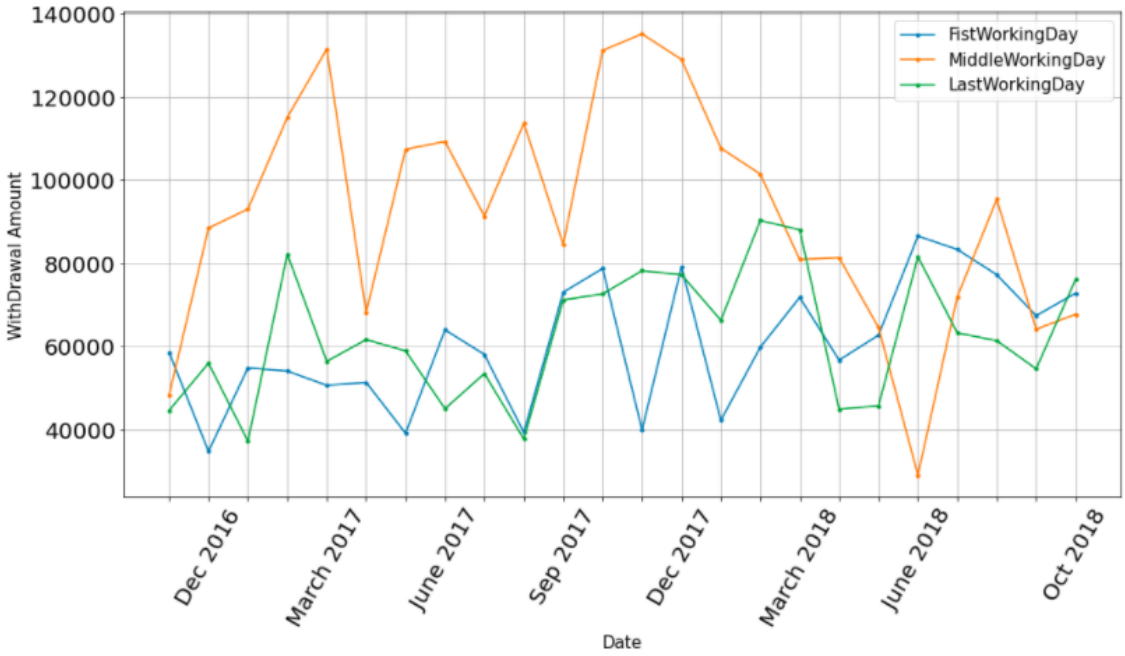
Following subsections, preprocessing on the dataset is shown. We explained adding new features to the dataset, selection of feature, scaling our dataset, creating feature sets.

3.1.1 New Features

In addition to the features in datasets, 7 new features that we thought and saw later can be effective in ATM forecasting have been created by using some features in the raw data. These are;

- The amount of cash withdrawn on the same day 1, 2, 3 and 4 weeks ago since we considered it is a pattern between weeks in the same month.
- The amount of cash withdrawn on the most similar day from the previous month since it is thought to be a close relationship between the months. The criteria for finding the most similar day is that there should be a maximum of 1 month between them and the number of working days of the record have to match. In this way, for example, if we chose a day that be the 2nd day and the first working day of that month; then, the first working day of the previous month was used as the most similar day, not the 2nd day.
- The average amount of money in the previous month as it is thought that the

Figure 3.3 Withdraw Amount distribution on set1



months in the dataset can affect each other.

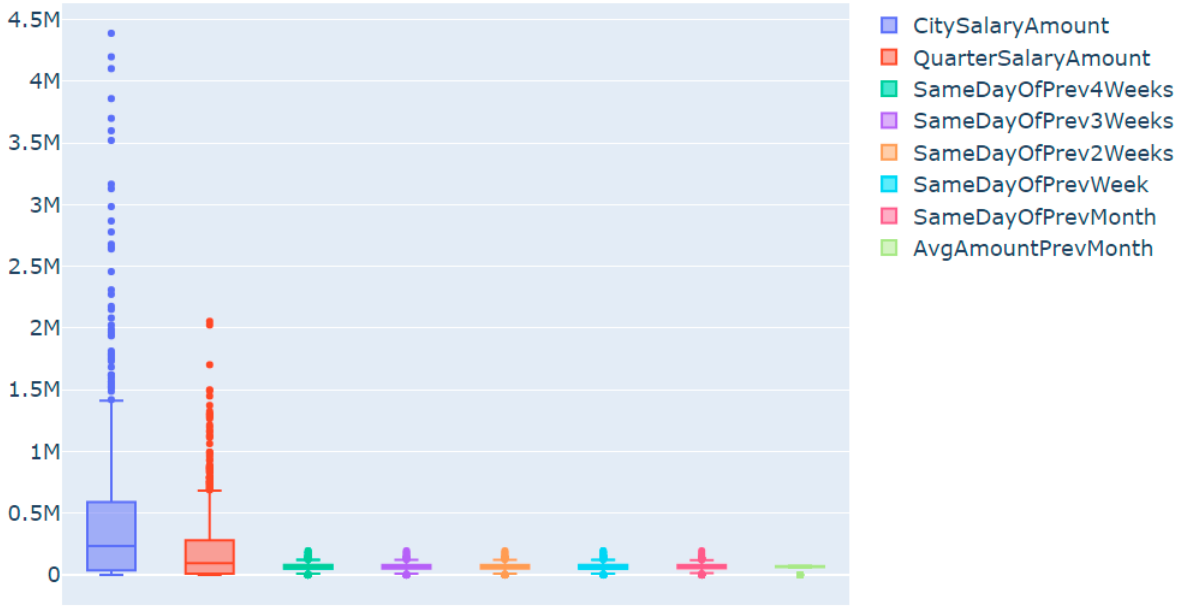
- Whether the data belongs to be the 1st or the 11th or the last working day of the month. In our country, employees' salary days vary from workplaces to workplaces, but the most commonly chosen payday is the first, 11th or last working day of the month. For this reason, we have added this information to our dataset as a new feature, since we think that it may be important in the ATM cash estimation.

Table 3.2 New Features

New Features	
IsFirstWorkingDay	IsMiddWorkingDay
IsLastWorkingDay	AvgWithdrawAmountPrevMonth
AmountFromSameDayOfPrevMonth	AmountFromSameDayOfPrevWeek
AmountFromSameDayOfPrev2Weeks	AmountFromSameDayOfPrev3Weeks
AmountFromSameDayOfPrev4Weeks	

After adding new features, we realize that there are many blank values for first month of dataset because we cannot reach the create value for the many new features. For this reason, we did not use our first-month data on models in the training and testing phase. After adding new features, we look at distribution features on the withdrawal amount. It is quite hard to find a relationship between numerical features and withdrawal amount, but it can be easy for binary features. Figure 3.3 show us that there is a quite strong relationship between first/middle/last working day.

Figure 3.4 Box Plot on Numerical Features



3.1.2 Feature selection

Regression models try to find optimal parameters for every feature in the input dataset. Because of this, the existence of variables that have weak relation with the target can add uncertainty to the forecasts, increase the run-time and reduce the accuracy of the model. At the same time, using too many features while creating models negatively affects the interpretability of models and cause over-fitting in small datasets. Since our dataset is small and contains too many features (see Appendix A), we decided to make a feature selection to prevent the negative situations mentioned above.

In this selection process, we identified 31 features to be used in training and testing with the SelectKBest function provided by Scikit-Learn on the ATM_1. These features and their score are shown in Table 3.3. Among the selected features, 22 of them are from our raw data set, except for the 7 new features we generated. (New features can be seen in Table 3.2.) The features from our raw data set;

- What day of the week the data belongs to (these features are categorical),
- whether this day is a holiday or not,
- the amount of salary deposited in the city of that day
- the amount of salary deposited in the district of that day;
- information on which month the data belongs to (these features are categorical)

After feature selection, we looked distribution of numerical features via box graph, and we realized the distribution of each feature's range really different each other. We explained the solution to this problem next section.

Table 3.3 Selected Features and Their Scores by SelectKBest Feature Selection Algorithm

Selected Features	Scores
CitySalaryAmount	626881890.134
QuarterSalaryAmount	308663435.889
AmountFromSameDayOfPrev4Weeks	8215920.378
AmountFromSameDayOfPrev3Weeks	7701768.037
AmountFromSameDayOfPrev2Weeks	7346133.665
AmountFromSameDayOfPrevMonth	7159250.451
AmountFromSameDayOfPrevWeek	6870876.883
AvgWithdrawAmountPrevMonth	2131709.742
IsLastWorkingDay	707.000
IsMiddWorkingDay	707.000
DayOfWeek	698.630
IsFirstWorkingDay	676.760
Month Dummy Values	
IsInMay	657.210
IsInDecember	651.315
IsInJune	646.633
IsInOctober	645.419
IsInApril	644.603
IsInAugust	643.454
IsInNovember	640.041
IsInJuly	637.559
IsInFebruary	635.839
IsInJanuary	627.734
IsInMarch	625.769
IsInSeptember	620.236
Day Dummy Values	
IsFriday	615.285
IsWednesday	605.114
IsThursday	601.633
IsSunday	597.713
IsMonday	597.713
IsSaturday	595.370
IsTuesday	581.908

3.1.3 Scaling

Our datasets contain both binary and numeric properties. Due to the difference in the scales of the numerical feature; the step size in gradient descent is different in each feature for ANN. The difference in the scale of the numerical feature can cause some features to become dominant and some features passive in the SVR and LSTM models. To prevent this situation, we scale our data before submitting it to the model. For this purpose we use 4 different scaling techniques and measure their effect on our machine learning models' success.

The first technique we use is minimum - maximum scaler to rescale our data. This way our data has a distribution between 0 and 1. Minimum-Maximum scaling can be express this formula;

$$X_i' = \frac{X_i - X_{imin}}{X_{imax} - X_{imin}} \tag{3.1}$$

X_i = Value of ith feature
X_{imin} = Minimum value of the ith feature
X_{imax} = Maximum value of the ith feature

The second one is a standard scalar to rescale our data. This way our data have a distribution that centre is around the mean.Standard scaling can be express this formula;

$$X_i' = \frac{X_i{}^{-}\mu_i}{\sigma_i} \tag{3.2}$$

X_i = Value of ith feature
μ_i = Mean of ith feature
σ_i = Standard Deviation of ith feature

The third technique is the Max-Absolute scalar. The Max-Absolute scaler rescales our data in each attribute based on the attribute's maximum absolute value. This way our data has a distribution between 1 and -1. Max-Absolute scaling can be

express this formula;

$$X_i^{'} = \frac{X_i}{|X_{imax}|} \tag{3.3}$$

X_i = Value of ith feature
X_{imax} = Maximum value of the ith feature

The last technique is a logarithmic scalar to rescale our data. This way our data distributions are reduced. Logarithmic scaling can be express this formula;

$$X_i^{'} = logX_i \tag{3.4}$$

X_i = Value of ith feature

We apply these techniques to the dataset and measure each technique on our machine learning models. Since the models use different decision mechanisms, different scaling techniques in each model achieved the best results. LSTM models give the best result on the dataset rescaled by Standard scalar, while ANN and SVR give the best result on the dataset by rescaled by a min-max scalar.

3.1.4 Feature Sets

While evaluating the success of our regression models, 7 different feature sets were created with selected features and the effect of these feature sets on the success of the models was examined. As feature sets;

- F0: The amount of money withdrawn on the same day one week before the day belonging to the data;
- F1: What day of the week the data belongs to, whether this day is a holiday or not, the amount of money withdrawn 1 week before this day, the amount of money withdrawn on the most similar day in the previous month and the average amount of money in the previous month;
- F2: In addition to the F1 set, whether the data belongs to the 11th working day of the month or not, the amount of salary deposited in the city and district of that day;
- F3: In addition to the F2 set, information on which month the data belongs to and whether the month is the first or last business day of the month;
- F4: In addition to the F2 set, the amount of money withdrawn on the same day 2, 3 and 4 weeks before the date this data belongs;
- F5: a combination of F3 and F4 sets;
- F6: all features on the dataset after feature selection;
- F7: The 30-day withdrawal amount from the day that this data belongs to, whether this day is a holiday and which day is the selected attributes. This attribute set is used only on the LSTM model.

The regression models we created using these feature sets were tested. These feature sets can be seen in Table 3.4.

Table 3.4 Feature Sets Used in Regression Models

Feature Sets	Features
F0	AmountFromSameDayOfPrevWeek
F1	Day Dummy Values + IsHoliday + AmountFromSameDayOfPrevWeek + AmountFromSameDayOfPrevMonth + AvgWithdrawAmountPrevMonth
F2	F1 + IsMiddWorkingDay + CitySalaryAmount+ QuarterSalaryAmount
F3	F2 + Month Dummy Values + IsFirstWorkingDay+ IsLastWorkingDay
F4	F2 + AmountFromSameDayOfPrev2Weeks+ AmountFromSameDayOfPrev3Weeks+ AmountFromSameDayOfPrev4Weeks
F5	F3 + F4
F6	All features
F7	30DayWithdrawal + IsHoliday + Day Dummy Values

3.2 Regression Models

Regression is a general name that tries to find the intensity and quality of the relationship between a dependent variable and a series of other independent variables. The traditional method to regression on sequence data has been ARIMA, but recently machine learning models are used more and more.

The regression is addressed within the Supervised Machine Learning problems in machine learning. The Supervised Machine Learning problems can be divided into Regression and Classification problems. Both of them are aimed at the creation of a brief model that can estimate the value of the dependent attribute from attribute variables. However, while the dependent attribute is numerical in the regression problem, it is categorical for the classification problem. In short, the purpose of regression is to construct a mathematical equation that estimate the dependent target variable (y) as a function of independent feature variables(x). After finding

the most successful equation, this can be used to predict the "y" based on the new values of the "x".

Since our aim in our study is to estimate the amount of cash withdrawn from ATMs as close to its real amount, the problem is considered as a regression problem. In our project, we used 4 different machine learning methods and 1 statistical technique to create regression models. These models were created using Linear Regression, Supporter Vector Machine, Artificial Neural Networks, Long Short-Term Memory and ARIMA.This section includes a short explanation of the regression methods that LR, SVR, ANN, , LSTM and ARIMA.

3.2.1 Linear Regression

Linear regression, which is the most basic and common regression model, aims to predict the target value (dependent variable) from a set of independent variables. It can also help us understand the dependence of the target variable to the independent variables.

Our goal in this thesis is to find the variables (attributes) (X) that can be used to predict the target value (y), which the cash amount that will be withdrawn in the predicted day. The domain set X is a subset of $\mathbb{R}^n$, for some n, and the label set Y is the set of real numbers. In linear regression, We learn a linear function h: $\mathbb{R}^n \to \mathbb{R}$ that minimizes the residual error (Shalev-Shwartz & Ben-David, 2014).

The linear regression can be formulated as follows:

$$\begin{aligned} w &= [\beta_1, \beta_2, ..., \beta_n], \forall \beta_i \in \mathbb{R} \\ X_i &= [x_1, x_2, ..., x_n], \forall x_n \in \mathbb{R} \\ LR_{reg}(X_i) &= y_i = \langle w, X_i \rangle + \beta_0, \beta_0 \in \mathbb{R} \\ LR_{reg}(X) &= y_i = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_n x_n \end{aligned} \tag{3.5}$$

Then, we need to define a loss function that decides how much we should be tolerated for mistakes between $LR_{reg}(x)$ and y for regression. There are many different type of loss function.We used 3 different ones in our thesis. First one is to use the residual sum of squares between the real target values in the dataset, and the predicted values by our linear function, namely squared loss function,

$$l(LR_{reg}(x), y)) = (LR_{reg}(x) - y)^2) \tag{3.6}$$

The second loss function that we used is Ridge loss function(L2). Ridge loss function adds a penalty value to the update and reduce our all weights through the zero equally. The main reason for adding the penalty to update is minimizing the variance in the size of weights hence prevent over-fitting. Ridge loss function;

$$L2(LR_{reg}(x), y)) = (LR_{reg}(x) - y)^2) + \lambda(w)^2 \tag{3.7}$$

The last one is Lasso loss function(L1). The Lasso loss function has similar purpose with ridge regression and use penalty to minimize variance in the size of weights. The difference between Ridge and Lasso regression is that they use different penalty approach.The penalty method used by Lasso also plays an important role in feature selection in datasets with large number of features. Lasso loss function;

$$L1(LR_{reg}(x), y)) = (LR_{reg}(x) - y)^2) + \lambda|w| \tag{3.8}$$

3.2.2 Support Vector Regression

Support Vector Regression (SVR) is another powerful technique for regression. Before giving the formulation of SVR, we overview how SVM works.

The SVM approach finds the optimal hyper-plane that separates two classes, maximizing the margin between classes. They can handle non-linear data using kernel functions that map implicitly to a higher dimensional plane, where the classes can be separable. Linear Kernel, polynomial kernel and Radial Basis Function (RBF) are most commonly used kernel functions, with RBF kernel having the highest capacity. The decision hyper-plane is represented by the support vectors which are the data points that are closest to the hyper-plane.

SVM can also be used in the regression problem, keeping all the main features that define the algorithm and called SVR. SVR uses the same policies as the SVM for classification (boundary lines, max-margin etc.), with just a few small differences.

Since the target value in regression is a real number, the formulation is changed slightly as follows (Shalev-Shwartz & Ben-David, 2014):

$$|y_i - w_i\, x_i| \leq \varepsilon + |\, \xi_i \,| \tag{3.9}$$

y_i = the target
w_i = the coefficient
x_i = the predictor (feature)
ε = maximum error
ξ_i = slack variable

Notice that as an important difference between SVR and simple regression is that, while the squared error is minimized in simple regression, SVR tries try to keep the error within a tolerance band (ε) (Shalev-Shwartz & Ben-David, 2014).

$$MIN \quad \frac{1}{2}\,||w||^2 + C \sum_{i=1}^{n} (|\xi_i|) \tag{3.10}$$

The regression line created in an example dataset with different kernel functions is illustrated in Figure 3.5 (Pedregosa et al., 2011).

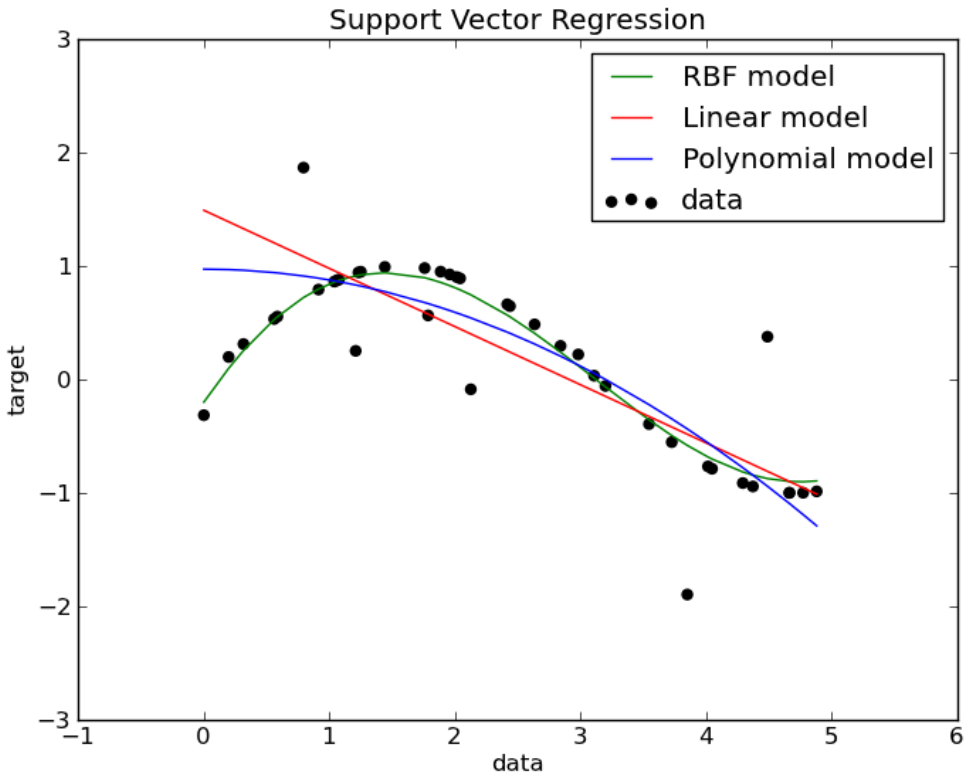


Figure 3.5 "SVR using linear and non-linear kernels",

3.2.3 Neural Network

Human nervous system can be modelled as a dense network containing numerous neurons and capable of extremely complex operations. Artificial neural network is a machine learning model created by imitating the human nervous system.

Learning with neural networks was proposed in 1944 by Warren McCullough and Walter Pitts (McCulloch & Pitts, 1943) and psychologist Frank Rosenblatt invent the first trainable neural network, the Perceptron (Rosenblatt, 1957). Before explaining how ANN working, we should look at the structure of a neuron.

A directed graph can express the ANN, with the vertex representing the neuron cell and the arc representing the connection between neurons. Each arc, which also known as edge or connection, in the network has weight value, the value represents the strength of the edge between neurons. Each neuron takes a certain number of inputs(X) and one extra value called bias(b). Each input is multiplied by the corresponding arc's weight(w_n) and the total weighted input form the net input (net) of the neuron. The output y of the neuron is then calculated using a non-linear activation function f_{net}. There are many activation functions. Most commons

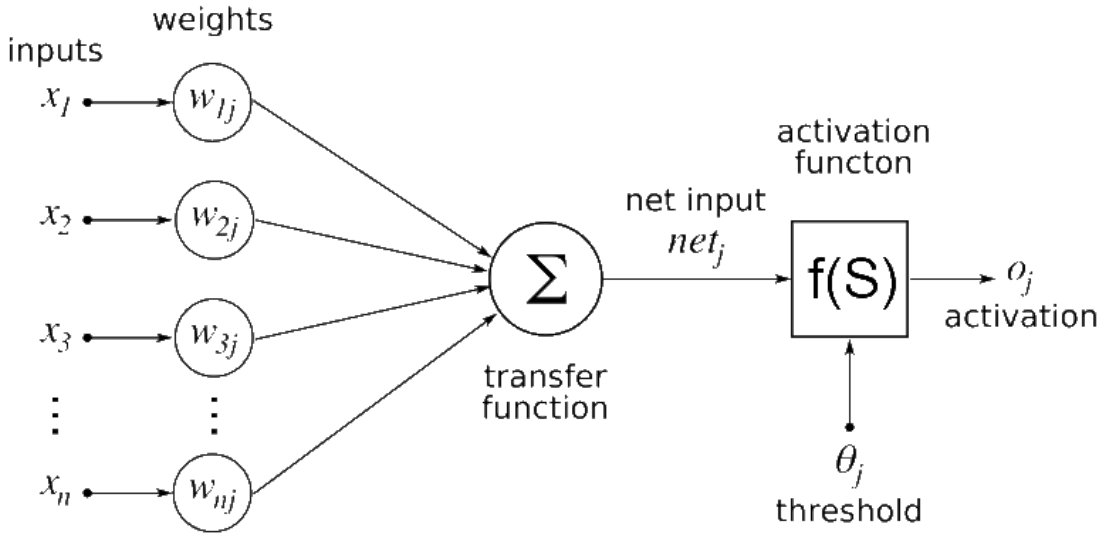


Figure 3.6 "Artificial Neural Network"

are sigmoid, tanh, logistic and ReLu. The computation done by a neuron can be summarized as:

$$
\begin{aligned}
X_i &= [x_1, x_2, ..., x_n], \forall x_n \in \mathbb{R} \\
W &= [w_1, w_2, ..., w_n], \forall w_n \in \mathbb{R} \\
f_{neuron} &= y_i = f_{activate}(X_i \dot{W}^T + b) \\
f_{neuron} &= y_i = b + w_1 x_1 + w_2 x_2 + ... + w_n x_n
\end{aligned}
\tag{3.11}
$$

The structure of the neuron can be seen in Figure 3.6 (Saini, 2017).

The architecture of an ANN consists of the layers and neurons in each layers and the connection pattern between them. The konowledge of the network is the value of the weights and biases. The network training consists of learning these weights and biases inside the network.

In what is called the *forward pass* or *forward propagation*, the first hidden layer receives the values from the input layer, processes these inputs in neurons, and transmits these values to the next layer. This process repeats until the output layer. Lastly, we take an output value from the output layer. (Shalev-Shwartz & Ben-David, 2014)

After forward propagation, we calculate an error between actual output and output generated by ANN via loss function, the most common lost functions are MSE and MAE for regression. The learning aims to minimize the loss, which is a function of the network‘s weights and biases, using gradient descent algorithm, called *back-propagation* in the context of neural networks. In this process, the error gradient is

passed from the output layer all the way to the input layer for each weight in ANN. the weights are updated from their current value by small amounts at each iteration, depending on the *learning rate*. There are many functions to update weights in the model. Most commons are SGD,MSE and RMSprop. This way help us to minimize error. In ANN, we repeat forward and back-propagation until reaching the minimum error values.

There are a lot of advantages of ANN. ANN is able to learn and model from non-linear data. Missing data does not affect ANN's working very much. ANNs can work on multiple tasks in parallel without disturbing each other task. (Awad & Khanna, 2015)

3.2.4 Long- Short Term Memory

Long-Short-Term Memory is a special Recurrent Neural Network that differs from other kinds in that it is able to learn long-term dependencies. LSTM were presented by Schmidhuber & Hochreiter (Hochreiter & Schmidhuber, 1997). They work amazingly well on many kinds of problems, therefore became more popular nowadays.

RNN consist of repeating modules as a form of a chain. In basic RNNs, this module has a simple construction and a single layer. Because of this, RNN can be able to connect few previous knowledge to the current task prediction. LSTM also have this structure, but the repeating modules are quite different. In the repeating modules have four neural network layers that interacts with each other, not a single layer. Because of this, holding information for a long time is their base behaviour and LSTM can be able to prevent the long-term dependency problem.(Olah, 2015)

Before the explaining LSTM diagram, we should talk about the notation we'll be using.

The LSTM cell take input from the dataset as a vector, output of previous cell and pipeline which carry much information. The most important part of LSTMs is the pipeline, also known as cell state. LSTM cell is able to add or delete information to the pipeline, by pointwise operations that structure are called gates. An LSTM cell has three gates and these serve a different purpose.

In LSTM, it is initially determined what knowledge to throw out from the cell state. Determination of this step is performed by a forget gate. The output of this gate goes to cell state and conveys that whether to keep the information or remove from

cell state.

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{3.12}$$

t = indexes the time step
f_t = forget gate's activation vector
σ = sigmoid function.
W_f = the weights of the forget gate
h_t = hidden state vector, output vector of the LSTM unit , $h_0 = 0$
x_t = input vector to the LSTM unit
b_f = bias vector parameters of forget gate

In the second step, it is determined what new knowledge is kept in the cell state. Firstly, the input gate layer chooses which values will be updated. Then, a neural network layer produces a vector using new values which can be added to the cell state.(Olah, 2015)

$$\begin{aligned} i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \\ \tilde{C}_t &= tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \end{aligned} \tag{3.13}$$

i_t = input/update gate's activation vector
$\tilde{C}_t$ = cell input activation vector
W_c = weights of the cell state
W_i = weights of the input gate
b_c = bias vector parameters of cell state
b_i = bias vector parameters of input gate

$$C_t = f_t C_{t-1} + i_t \tilde{C}_t \tag{3.14}$$

C_t = cell state vector, $c_0 = 0$

The third and last step is to decide what is the output. This output is produced based on cell state and output gate. At first, which pieces of the cell state will be the output of the cell are decided by the output gate. Next, the cell state is filtered through the output of the output gate and this process give LSTM cell's output.(Olah, 2015)

$$\begin{aligned} o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \\ h_t &= o_t tanh(C_t) \end{aligned} \tag{3.15}$$

o_t = output gate's activation vector
W_o = weights of the output gate
b_o = bias vector parameters of the output gate

3.2.5 ARIMA

The ARIMA model is an important and common forecasting tool in statistic science. It is the basis of many approaches in time-series analysis.ARIMA was introduced by is Box and Jenkins (Box & Jenkins, 1976), therefore the model is also known as Box-Jenkins models. (Chatfield, 2000)

The ARIMA method analyses univariate time series data, transfer data, and intervention data and predicts based on their inferences using the ARIMA or ARMA model. To understanding ARIMA's workflow, we should know the AR and MA model's process.

In an AR model, the prediction(y_t) is calculated by p past values(Y_{t-1}) in the time-series. The predictions which are generated by AR can be expressed by an equation like:

$$
\begin{aligned}
A &= [\alpha_1, \alpha_2, ..., \alpha_{t-1}] \\
Y_{t-1} &= [y_1, y_2, ..., y_{t-1}] \\
AR(p) = y_t &= \sum_{i=1}^{p} (\alpha_i y_{t-i}) + \varepsilon + \alpha_0 \\
y_t &= \alpha_0 + \alpha_1 y_{t-p} + ... + \alpha_{p-1} y_{t-2} + \alpha_p y_{t-1} + \varepsilon
\end{aligned}
\tag{3.16}
$$

$\alpha_0 =$ constant variable
$\varepsilon \quad =$ random shocks

In an MA model, the prediction(y_t) is calculated by error terms of past values(ε_{t-1}) in the time-series. The predictions which are generated by AR can be expressed by an equation like:

$$
\begin{aligned}
B &= [\beta_1, \beta_2, ..., \beta_{t-1}] \\
\varepsilon_{t-1} &= [\varepsilon_1, \varepsilon_2, ..., \varepsilon_{t-1}] \\
MA(q) = y_t &= \sum_{i=1}^{q} (\beta_i \varepsilon_{t-i}) + \beta_0 \\
y_t &= \beta_0 + \beta_1 \varepsilon_{t-p} + ... + \beta_{p-1} \varepsilon_{t-2} + \beta_p \varepsilon_{t-1} + \varepsilon
\end{aligned}
\tag{3.17}
$$

$\beta_0 =$ constant variable

ARIMA model is combined to AR and MA model so estimate a value in a time-series with a linear combination of the time-series' past errors and past values of time series. ARIMA model takes 3 parameters for creating model, they are (p, d, q). These are;

- p is the number of previous periods used for estimation. It is the same in the AR model.
- d is the number of different transformations required to stabilize the time series.
- q is the number of errors that belongs to previous periods used for estimation. It is same in the MA model.

(SAS Institute Inc., 2014)

If you create a model with (p,0,q), We can say the formula of this model equals to AR(p)+MA(q). Analysing the performance of the ARIMA models is three stages: Identify, Estimate, Forecast (Box & Jenkins, 1976)

4. Experiments & Result Analysis

In this thesis, we worked on forecasting withdrawal amount of ATM by using dataset taken from a private Turkish bank. The following section explores details of the development environment, preparations for experiment and result analysis.

4.1 Experimental Environment

We use Scikit-Learn library (Pedregosa et al., 2011) to machine learning function and statmodels library (Seabold & Perktold, 2010) to ARIMA models in Python environment Python 3.6.9. Development IDE is Google Colab.

Test server operating system is Windows 10 Home 64-bit OS. The processor of the machine is Intel ® Core ® i5-7200U CPU @ 2.50 GHz and installed memory RAM is 4 GB.

4.2 Preparing Data

In this subsection, we explained which way we followed to split our dataset into train and test sets.f

4.2.1 Train & Test Set Split

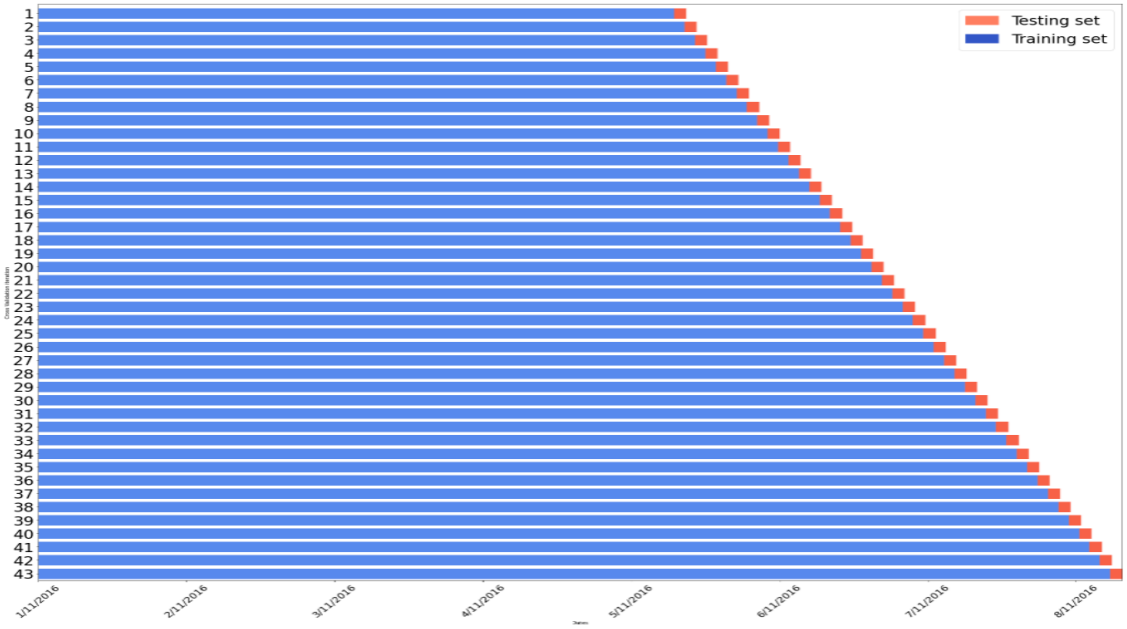
Cross-validation is a method that allows the performance of models to be measured without overfitting or underfitting on small datasets. In classical cross-validation technique; the dataset is split into several groups that have equal number of data randomly, models are trained on all groups except one and are tested on the remaining group. This process are repeated until models are tested on each group and the final score are calculated as the average of scores in each group.The classical cross validation cannot be used in time series estimation, because it is necessary to avoid leaking information from the future and to prevent make predictions for the past using future knowledge.

Despite the above-mentioned problems, it is inevitable to use cross-validation in studies where small datasets are used, as our study. Therefore, different cross-validation techniques have been created that provided the inter-dependencies is maintained in time series problems.There are 2 different cross validation techniques commonly used in time series. One of them adopts training models on expanding sets and testing them in the next fixed-length group. The other one creates the fixed-length train and test sets by sliding on the dataset. In our study, we followed first approach for cross validation.We use 2 different prediction period; weekly and monthly.

In weekly prediction; in the current data set, after separating the 13th and 14th months as the verification set and performing the parameter optimization, our models were trained with extended training sets and always predicted the next week. In this way, the first training set contains data from 2nd to 14th months and test set is next week; then second training is expanded to include the week that was the test set in the previous step, and the next week is the test set. This process repeats until the final week on dataset became test set (this is 43 times on ATM_1, 40 times ATM_2 because ATM_1 has 103 weeks, while ATM_2 has 100 weeks.)Each test set consists of a period of 1-week post-training set. We record the model's prediction for each test sets. Finally, the average of 43 test sets' error (60-103 weeks) evaluated as the model testing error on ATM_1. (The number of the test set is 40 for ATM_2). The train and test sets created with cross-validation for ATM_1 can be seen in Figure 4.1. Due to the heavy workload and taking very long time, there is no weekly predictions were made on the LSTM.

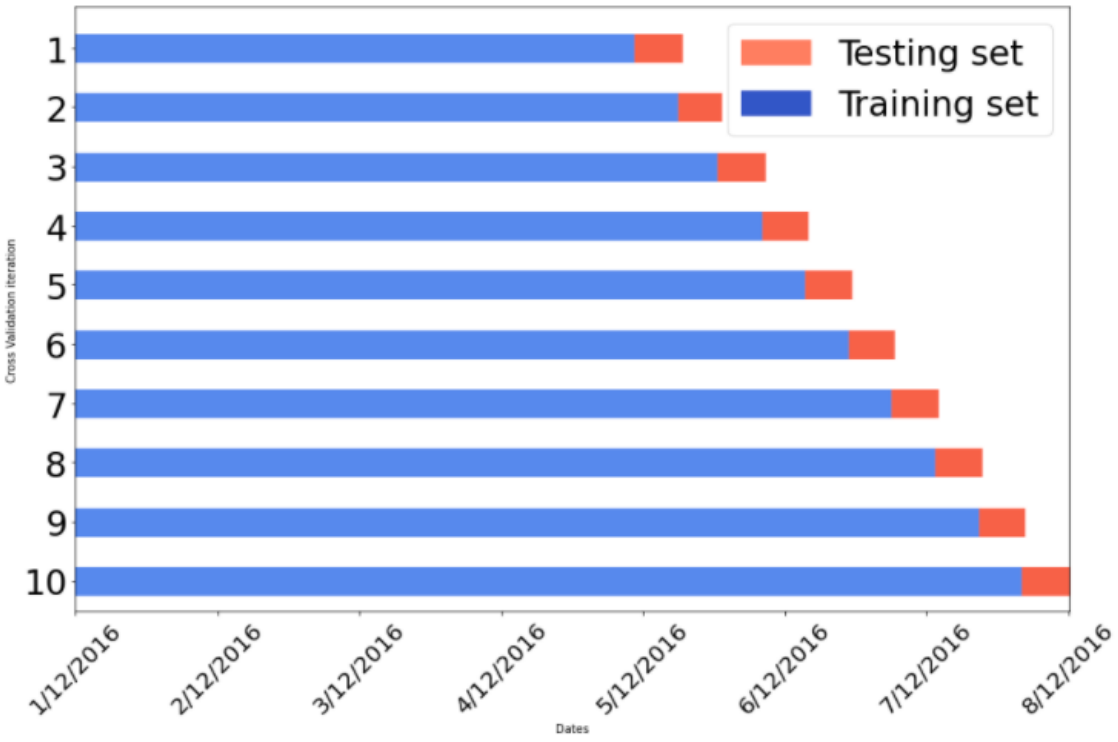
In monthly prediction; we predicted the next month and extend our training set 1 month. In this way, the first training set contains data from 2nd to 14th months and the test set is next month; then the second training is expanded to include the

Figure 4.1 Train and Test Set on Cross Validation for Weekly Prediction



month that was the test set in the previous step, and the next month is the test set. This process repeats until the final month on dataset became test set (this is 10 times on ATM_1, 9 times ATM_2 because ATM_1 has 24 months, while ATM_2 has 23 months.)Each test set consists of a period of 1-month post-training set. We record the model's prediction for each test sets. Finally, the average of 10 test sets' error (15-24 month) evaluated as the model testing error on ATM_1. (The number of the test set is 9 for ATM_2). The train and test sets created with cross-validation for ATM_1 can be seen in Figure 4.2

Figure 4.2 Train and Test Set on Cross Validation for Monthly Prediction



It should be noted that the training set used for each test evaluation it is getting bigger and bigger, but the size of the testing set is constant; e.g. training set contains data from 1st to 102nd weeks and the test set is 103rd weeks for the final run on the ATM_1.

4.3 Generating Models

In this thesis, we use a traditional statistical method and different machine learning algorithms to create the models because we want to compare them with each other at the end of the experiments. Therefore, We created 1 LR model, 1 SVR model, 1 ANN model, 1 ARIMA model and 2 LSTM models. LSTM models take data as (30 x 9) shape. The difference between LSTM models is the number of a hidden layer on models. One of them has only 1 hidden layer, another one has 2 hidden layers. Next subsection, parameters of the models explained.

4.4 Parameter Fine-Tuning

In machine learning, finding optimal parameters for the model is one of the most critical parts. Because it directly affects the model's predictions and accuracy.

Optimal parameters of our models are determined by Grid-Search Algorithm provided Scikit-Learn library (Pedregosa et al., 2011). This algorithm runs the model with all combination of given parameters and returns parameters that give the best accuracy. All given parameters to Grid Search Algorithm and parameters returned by Grid Search Algorithms can be seen in Table 4.1.

Table 4.1 Parameter Fine-Tuning on Regression Models

Regression Models	Possible Parameters	Best Parameters
LR	α_{L1}: {0.25, 0.5, 1, 1.5, 2, 3, 5,8,10,15,20,50,60,80,100,200,500,10000} α_{L2} : {0.25, 0.5, 1, 1.5, 2, 3, 5,8,10,15,20,50,100,200,500,1000,10000} lossfunction:{ Residual Sum, Lasso, Ridge}	$\alpha_{L1} = 80$ $\alpha_{L2} = 5$ lossfunction=Lasso
SVR	kernel: {linear, rbf, poly} ε:{0.01,0.1,0.2,0.3,0.4,0.5,0.8} γ : {0.01, 0.001, 0.0001} C : {1, 10, 20, 30, 40,50,4000,6000, 20000,8000,12000} degree : {2,3}	kernel = linear $\varepsilon = 0.8$ γ= 0.0001 C = 4000 degree = 2
ANN	hidden layer sizes :{(20,),(18,),(30,),(25,),(34,),(50,),(100,),(120,),(200,)} activation :{relu, tanh} learning rate init :{0.001,0.0001} tol:{10^7,10^10} α :{0.001,0.005, 0.01}	hidden layer sizes= (100,) activation = relu learning rate init = 0.001 tol = 10^7 $\alpha = 0.01$
LSTM	batch size : {4,5,6,7} number of neurons: [5,15] epoch :[1, 30] learning rate : {0.001,0.0001}	batch size = 5 number of neurons= 8 epoch = 10 learning rate= 0.0001
ARIMA	p :{0, 1, 2, 7, 8} d:{0, 1} q : [0,4]	p = 8 d = 0 q = 2

In the LR method, we used 3 different loss function. After comparing them, we reached best performance on Lasso loss function with 80 alpha. The biggest 10 coefficient on Linear Regression can be seen in Table 4.2. When we look at the table, we can say that importance of features does not change very much with different prediction period and whether the data belongs to the weekend is most effective on the prediction.

(Pedregosa et al., 2011)

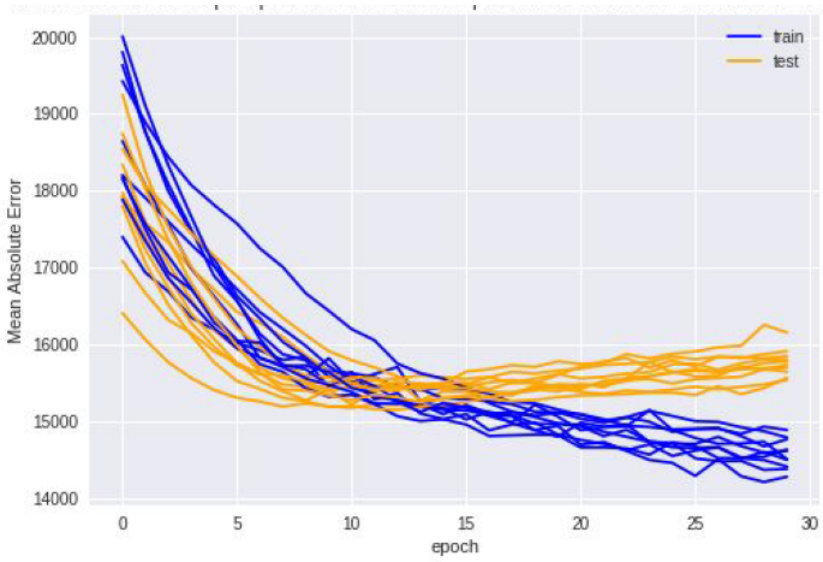
In the SVR method, we reached the best performance with linear core, 0.8 epsilon value, 0.0001 gamma value, second degree and 4000 regularization parameter.

In the ANN method, we found the best result was obtained with relu activation function, 0.01 alpha value, 1 hidden layer consisting of 100 cells, and 0.001 initial learning rate.

Table 4.2 Biggest 10 Coeeficient on Linear Regression

Monthly	Weekly
IsSaturday: 18,646	IsSaturday: 17,978
IsSunday: -13,733	IsSunday: -13,742
IsMiddWorkingDay: 10,948	IsMiddWorkingDay: 10,623
IsInSeptember: -7,909	IsInSeptember: -7,775
IsInOctober: 7,430	IsInOctober: 7,315
IsInApril: -6,511	IsInApril: -6,599
IsFirstWorkingDay: -5,696	IsInAugust: 5,335
IsInFebruary: 5,562	IsInMay: 5,183
IsInMay: 5,510	IsInDecember: -5,111
IsInAugust: 5,231	IsFirstWorkingDay: -4,995

Figure 4.3 Comparison of Epoch number via MSE



LSTM models were generated by 8 neurons, 5 batch size, 0.0001 learning rate and 10 epoch. Figure 4.1. shows the change in MSE of the LSTM model has 1 hidden layer by epoch number. This figure also shows that overfitting started after 10th epoch.

To test our hypothesis that machine learning models will be more successful than traditional statistical models, one of the most widely used statistical model, ARIMA model, was created with (8,0,3) order.

Feature Set	MEAN ABSOLUTE ERRORS Train Set/Test Set		
Weekly			
	LR	SVR	NN
F0	19,050 / 17,980	18,554 / 17,504	19,188 / 17,915
F1	16,613 / 16,002	16,462 / 15,432	17,126 / 15,965
F2	15,364 / 15,899	**15,584 / 15,384**	15,599 / 15,910
F3	14,825 / 15,935	15,138 / 15,804	**15,200 / 15,783**
F4	15,289 / 15,891	15,348 / 15,491	15,731 / 16,082
F5	14,703 / 16,270	15,051 / 15,836	15,234 / 15,835
F6	**14,491 / 15,550**	14,786 / 15,628	15,634 / 16,054
Monthly			
	LR	SVR	NN
F0	19,107 / 18,710	18,590 / 17,435	19,269 / 18,149
F1	16,683 / 16,518	16,520 / 15,447	**17,086 / 15,684**
F2	15,317 / 16,289	**15,621 / 15,367**	15,612 / 16,071
F3	14,753 / 16,313	15,181 / 15,838	15,753 / 15,706
F4	15,248 / 16,300	15,379 / 15,477	15,663 / 16,208
F5	14,703 / 16,270	15,051 / 15,836	15,234 / 15,835
F6	**14,445 / 15,949**	14,822 / 15,725	15,859 / 16,559
		LSTM	
	Single Layer		One Hidden Layer
F7	**15,063/13,755**		14,972/ 13,824

Table 4.3 MAE Values of Regression Models Created Using Different Attribute Sets on ATM_1. The best results for each model are shown in bold.

4.5 Results

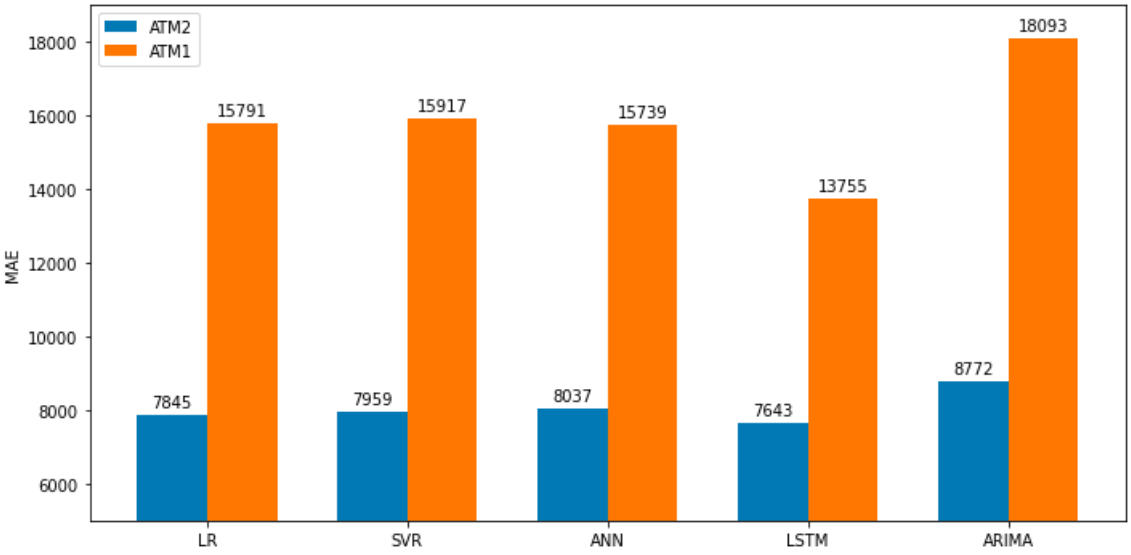
Mean Absolute Error (MSE) metric was used for evaluation.In this problem, the Mean Absolute Percentage Error (MAPE) metric is also used, but in our case, situations, where money withdrawals are zero due to physical malfunctions, create a problem.

Table 4.3 and 4.4 show the MAE values on the test data of machine learning models trained using different attribute sets. As the values in the table show, different models were able to make the best estimates with different feature sets and the best feature set can be change testing on another ATM data or different prediction period. While the artificial neural networks model gave the most successful performance when trained using the F1 feature set on monthly prediction, Supporting

Feature Set	MEAN ABSOLUTE ERRORS Train Set/Test Set		
Weekly			
	LR	SVR	NN
F0	7292 / 8319	7084 / 8544	7253 / 8326
F1	6490 / 7670	6231 / 7663	6572 / 7712
F2	6231 / 7548	**5949 / 7531**	**6337 / 7498**
F3	6143 / 7511	5465 / 7837	6339 / 7504
F4	6209 / 7565	5807 / 7565	6405 / 7504
F5	6131 / 7536	5378 / 7833	6328 / 7405
F6	**5842 / 7498**	4592 / 7562	6386 / 7731
Monthly			
	LR	SVR	NN
F0	7243 / 8302	7032 / 8651	7172 / 8498
F1	6450 / 7650	6190 / 7760	6561 / 7542
F2	6191 / 7523	**5911 / 7589**	6459 / 7624
F3	6106 / 7501	5413 / 7930	6414 / 7618
F4	6170 / 7551	5769 / 7634	**6360 / 7500**
F5	6094 / 7534	5329 / 7938	6348 / 7539
F6	**5816 / 7417**	4557 / 7714	6176 / 7352
		LSTM	
	Single Layer		One Hidden Layer
F7	**4,126/ 7,643**		4,275/ 7,836

Table 4.4 MAE Values of Regression Models Created Using Different Attribute Sets on ATM_2. The best results for each model are shown in bold.

Figure 4.4 Best MAE values of Models for Monthly Predictions on ATM_1,ATM_2



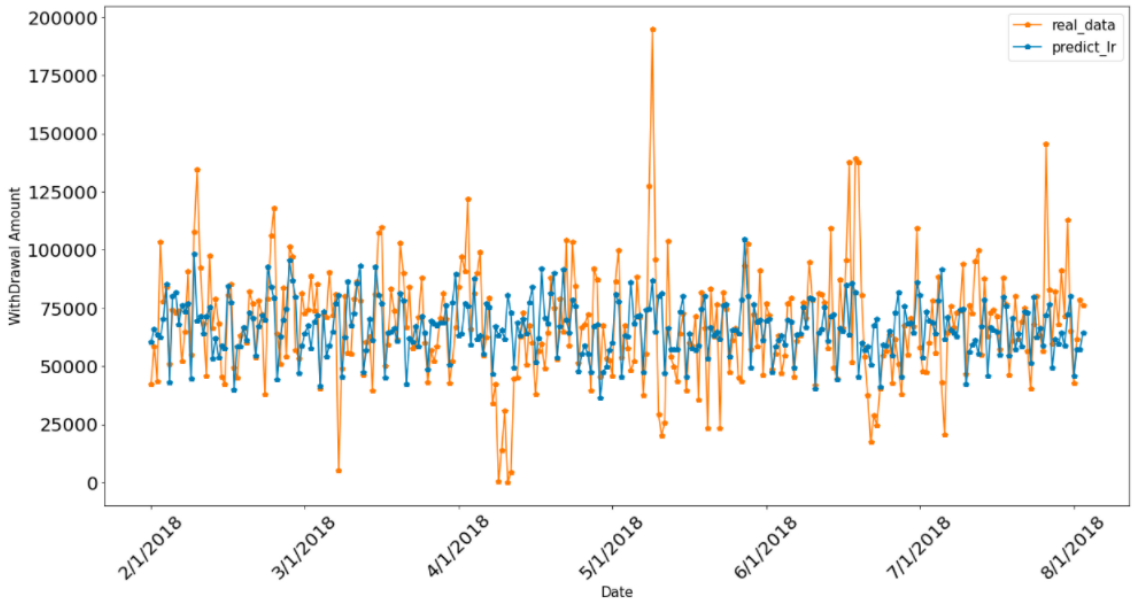
Vector Machine models were given when trained with the F4 feature set on ATM_1 on same task. On the other hand, when using the ATM_2, ANN gave the most successful performance when trained using the F4 feature set, not F1 on monthly prediction.Table 4.3 and Table 4.4 show us there is no need to create different feature sets for Linear Regression, the model gives the best result on F6 that have all 31 features. As another important inference, it was seen that the LSTM model made more successful predictions compared to other models, although much fewer features were used. We can say that LSTM and ANN work best on less feature compared to LR and SVR. Also these results show that there is not much differences between weekly and monthly prediction for each model on both ATM dataset.

As can be seen from Figure 4.4, 1 hidden layer LSTM network gave the best result with 13,755 errors. This result corresponds to an average absolute percentage error of % 20.2 for ATM_1, which is close enough to the rate that experts consider adequate. On the other hand, LSTM is not succesfull on ATM_2 dataset LSTM is not as successful on the ATM_2 dataset as on the ATM_1 dataset.

The ARIMA model that we generated for comparison of machine learning regression techniques and traditional statistical method's achievement gave MAE as 18,093 on ATM_1 and 8,365 on ATM_2 in monthly prediction.

The best MAE values for monthly forecasting each regression model can be seen in Figure 4.4. On both of the ATMs, while ARIMA performed worst among all models, LSTM performed best. On the other hand, ANN and LR works better on ATM_2 while SVR work better on ATM_1. In Figure 4.5, we can see predictions made by LR model on monthly forecasting and real values for ATM_1 in the same graph.

Figure 4.5 Predictions of LR vs Real Values



This figure shows that the LR model predicts successfully on withdrawal amount that closes to the average withdrawal amount, but can not handle the min and max peaks. In Figure 4.6, we can see predictions made by the SVR model on monthly forecasting and real values for ATM_1 in the same graph. This figure also shows that the SVR model predicts successfully on withdrawal amount that closes to the average withdrawal amount, but can not handle the min and max peaks. Although both SVR and LR can not handle extreme values, SVR is more succesful to predict withdrawal amount closes to the average than LR.

In Figure 4.7 and Figure 4.8, we can see predictions made by ANN and LSTM with 1 hidden layer models on monthly prediction and real values for ATM_1 in the same graphs. This figures also show that LSTM and ANN model predicts more successfully on withdrawal amount that closes to the average withdrawal amount than SVR and LR model, at the same time they handle a few minimum and maximum peaks. Both LSTM and ANN tend to catch minimum peak compared to the maximum.

Figure 4.6 Predictions of SVR vs Real Values

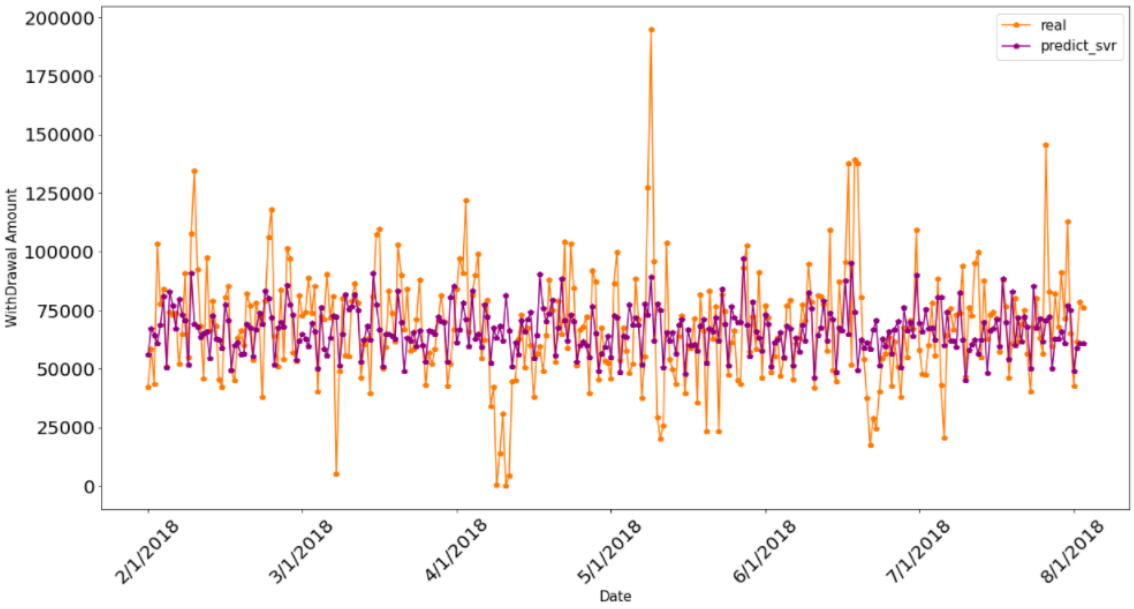


Figure 4.7 Predictions of ANN vs Real Values

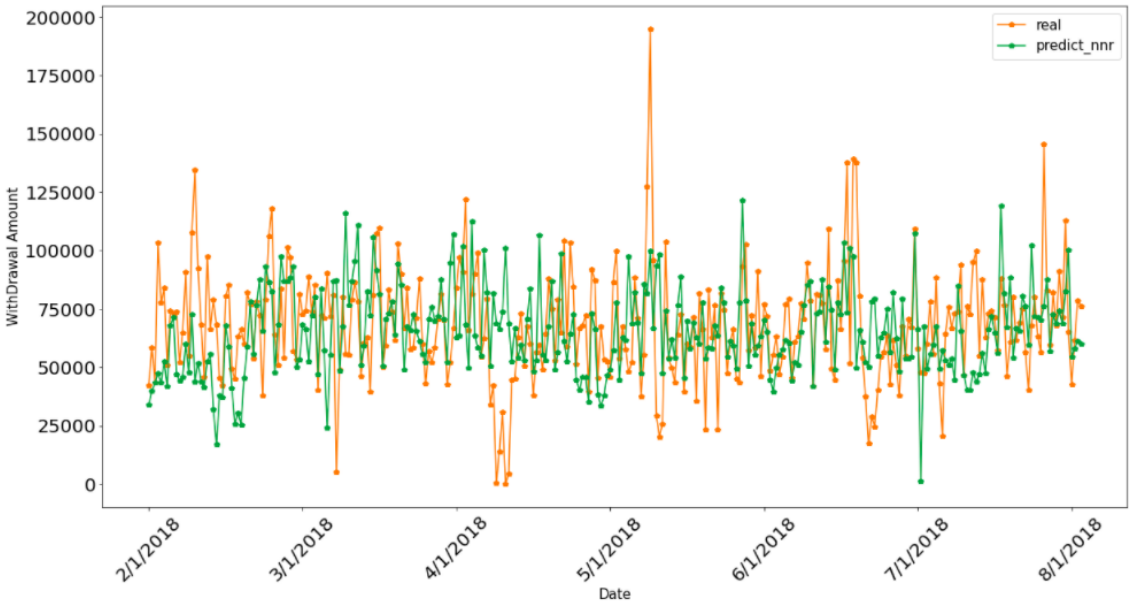
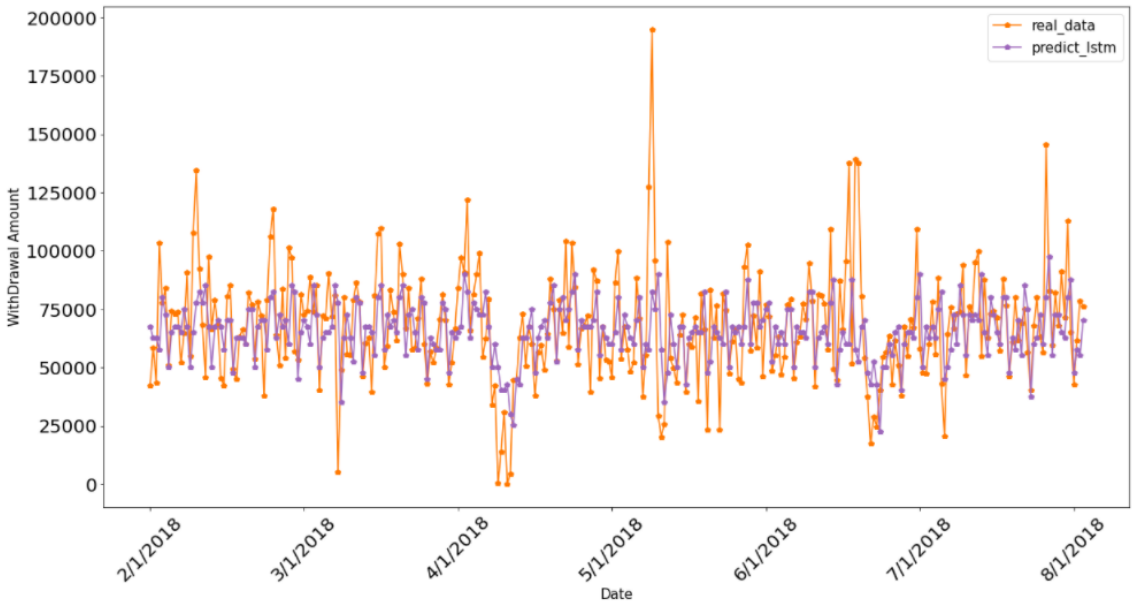


Figure 4.8 Predictions of LSTM vs Real Values



5. Discussion

In this thesis, while the models were being trained and testing, we use given dataset without changing any value. The incorrect inputs caused by the physical problems of ATM in the dataset were not eliminated, and we believe that this incorrect data decrease our models' success. For ensure this claim, we made small experiment.

When we look the dataset of ATM_1, we find 3 records had 0 withdrawal amounts. To prove our claim, we impute this values with different techniques and train and weekly test SVR models on them. First technique (IM1) is filling these values with withdrawal amount from previous week. Second one (IM2) is filling these values with withdrawal amount from previous month. Third technique (IM3) is filling these values with average withdrawal amount from previous and following weeks. The last technique (IM4) is filling these values with average withdrawal amount from previous and following months.Mean absolute error of SVR can be seen in the Table 5.1.

Feature Set	NoImpute	IM1	IM2	IM3	IM4
F0	18,554 / 17,504	18343 / 17456	18377/ 17087	**18413 / 17078**	18274 / 17087
F1	16,462 / 15,432	16538 / 15700	16329 / 15051	**16365 / 15042**	16218 / 15050
F2	15,584 / 15,384	15631 / 15544	15461 / 15140	**15497 / 15132**	15350 / 15140
F3	15,138 / 15,804	15173 / 16118	14947 / 15421	**14983 / 15413**	14850 / 15397
F4	15,348 / 15,491	15434 / 15649	15230 / 15239	**15266 / 15230**	15119 / 15238
F5	14,703 / 16,270	15058 / 16082	14831 / 15435	14867 / 15427	**14727/ 15405**
F6	14,786 / 15,628	13356 / 16077	14592 / 15505	14628 / 15497	**14479/ 15492**

Table 5.1 MAE of SVR on different impute techniques

The Table 5.1. illustrates that imputing dataset generally increase model's success. SVR makes much better prediction for all feature sets on imputed dataset except first one. We achieved the best result in dataset imputed with third and fourth technique. The important part in this experiment, we only filled 3 record that had 0 withdrawal amounts and even this increase the model's success. We know that there are some incorrect withdrawal amounts caused by physical problem and not all of them is 0. In next studies, the problem can be addressed more comprehensively and imputing techniques that affect the success rate more can be investigated.

6. Conclusions

In this thesis study, it is aimed to estimate the amount of money to be withdrawn from ATM daily, with the least error rate, by considering the cash withdrawn from ATM in the past. For this purpose, 6 different models were created using linear regression, supporting vector machine, artificial neural networks, LSTM and ARIMA algorithms and the error rates of these models were compared with each other.

While the models are being trained, we use the training set that we expand with a specific cross validation method. In testing phase, we make predictions for 2 different period. First one is weekly prediction and other one is monthly prediction. The test results show us that there is not many differences between weekly and monthly prediction. Additionally, the effect of 8 different feature sets on the models was also examined, and it was seen that the models gave the best performance on different feature sets and the feature set can be change when using the different ATM or different test period. In addition, the incorrect inputs caused by the physical problems of ATM in the dataset were not eliminated, so our system was built on a more realistic dataset.

The experimental results showed that the LSTM network had the best results with 13.755 and 7643 MAE on the ATMs, since the LSTM units have the ability to capture long-term interactions and recover in Recurrent Neural Networks (RNNs). In addition, other machine learning models created gave much better results than ARIMA model that give 18093 and 8365 MAE on the ATMs, and it confirms our hypothesis that machine learning models will make much more successful predictions than statistical models.

In the thesis, we used only date information of the historical data to forecasting. In future studies, the success rate of such models can be increased by adding extra information such as ATM location information and applying different preprocessing processes. Further, the success rate can be perfectible with using different imputing techniques for incorrect record.

BIBLIOGRAPHY

Andrawis, R. R., Atiya, A. F., & El-Shishiny, H. (2011). Forecast combinations of computational intelligence and linear models for the NN5 time series forecasting competition. *International Journal of Forecasting*, *27*(3), 672–688.

Awad, M. & Khanna, R. (2015). *Efficient Learning Machines: Theories, Concepts,and Applications for Engineers and System Designer,*. APress Open.

Bilir, C. & Doseyen, A. (2018). Optimization of atm and branch cash operations using an integrated cash requirement forecasting and cash optimization model. *Business & Management Studies: An International Journal*, *6*(1), 237–255.

BKM (2020). Number of pos, atm, cards. Technical report, BKM.

Box, G. & Jenkins, G. (1976). *Time Series Analysis: Forecasting and Control, Revised Edition,*. Holden Day,.

Chatfield, C. (2000). *Time-Series Forecasting*. Chapman & Hall/ CRC.

Ekinci, Y., Lu, J., & Duman, E. (2015). Optimization of atm cash replenishment with group-demand forecasts. *Expert Systems with Applications*, *42 (7)*, 3480–3490.

Hochreiter, S. & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780.

McCulloch, W. S. & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, *5*, 115—-133.

Olah, C. (2015). Understanding lstm networks.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Rosenblatt, F. (1957). The perceptron—a perceiving and recognizing automaton. Technical report, 85-460-1,Cornell Aeronautical Laboratory.

Saini, G. (2017). Artificial neural network.

SAS Institute Inc. (2014). *SAS/ETS 13.2 User's Guide*. Cary: SAS Institute Inc.

Seabold, S. & Perktold, J. (2010). statsmodels: Econometric and statistical modeling with python. In *9th Python in Science Conference*.

Serengil, S. I. & Ozpinar, A. (2019). Atm cash flow prediction and replenishment optimization with ann,. *Uluslararası Mühendislik Araştırma ve Geliştirme Dergisi,*, *11 (1),*, 402–408,.

Shalev-Shwartz, S. & Ben-David, S. (2014,). *Understanding Machine Learning:From Theory to Algorithms,*. Cambridge,: Cambridge University Press,.

Simutis, R., Dilijonas, D., Bastina, L., & Friman, J. (2007). A flexible neural network for atm cash demand forecasting. (pp. 162–165).

Vangala, S. & Vadlamani, R. (2020). Atm cash demand forecasting in an indian bank with chaos and deep learning,. *CoRR*.

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*, *232*(2), 383–392.

APPENDIX A

Features of Raw ATM Dataset

AtmId: Unique Id of ATM that given from bank
CurrencyId: Unique Id of currency that can be withdrawn from the bank
CitySalaryCount: Count of the salary with payday in the city
CitySalaryAmount: Amount of the salary with payday that in the city
QuarterSalaryCount: Count of the salary with payday that in the city
QuarterSalaryAmount: Amount of the salary with payday that in the city

Date Dummy Values

- Is the instance on Day 1 of the month
- Is the instance on Day 2 of the month
- Is the instance on Day 3 of the month
- Is the instance on Day 4 of the month
- Is the instance on Day 5 of the month
- Is the instance on Day 6 of the month
- Is the instance on Day 7 of the month
- Is the instance on Day 8 of the month
- Is the instance on Day 9 of the month
- Is the instance on Day 10 of the month
- Is the instance on Day 11 of the month
- Is the instance on Day 12 of the month
- Is the instance on Day 13 of the month
- Is the instance on Day 14 of the month
- Is the instance on Day 15 of the month
- Is the instance on Day 16 of the month
- Is the instance on Day 17 of the month
- Is the instance on Day 18 of the month
- Is the instance on Day 19 of the month

- Is the instance on Day 20 of the month
- Is the instance on Day 21 of the month
- Is the instance on Day 22 of the month
- Is the instance on Day 23 of the month
- Is the instance on Day 24 of the month
- Is the instance on Day 25 of the month
- Is the instance on Day 26 of the month
- Is the instance on Day 27 of the month
- Is the instance on Day 28 of the month
- Is the instance on Day 29 of the month
- Is the instance on Day 30 of the month
- Is the instance on Day 31 of the month
- Is the instance last Monday of the month
- is the instance last Tuesday of the month
- is the instance last Wednesday of the month
- is the instance last Thursday of the month
- is the instance last Friday of the month
- reverse of all day dummy values

Day Dummy Values

- Is the instance on Monday
- Is the instance on Tuesday
- Is the instance on Wednesday
- Is the instance on Thursday
- Is the instance on Friday
- Is the instance on Saturday
- Is the instance on Sunday

Month Dummy Values

- Is the instance in January
- Is the instance in February
- Is the instance in March
- Is the instance in April
- Is the instance in May
- Is the instance in June
- Is the instance in July
- Is the instance in August
- Is the instance in September
- Is the instance in October
- Is the instance in November
- Is the instance in December

WithdrawalDate: Date of the record
DayOfWeek: Which day of the week (numeric)
DayOfMonth: Which day of the month (numeric)
WorkDayOfWeek: Which working day of the week (numeric)
IsHoliday: Is the instance at holiday?
WorkDayOfMonth: Which working day of the month (numeric)
WithdrawalAmount: Withdrawal Amount observed on the date
DepositAmount: Deposit Amount observed on the date
Holiday : Is the instance at national or religious holiday