Predictive Analytics Group Project
Predicting Cash Withdrawal Demand
Group 5

Ha Phuong Thao
Tran Tuan Minh
Hoang Thi Hoai Thanh
Hoang Minh Ngoc

1. Executive summary

Developing a sufficient predictive model for ATM cash demand is a crucial task for every bank to optimize cash management system.

In this report, we aim at identifying the best model to maximize the efficiency of the bank's cash management from utilizing numerous predictive models we tested in total, which are: Forward stepwise OLS, OLS with all variables and interaction terms, Lasso with original variables, Lasso with original variables with all interaction terms, and Deep Learning - Neural Network. In conclusion, the last model, Neural Network, after testing and comparing, is determined to be our final model to predict cash withdrawal with the relatively smallest train MSE and test MSE compared to the other predictive models.

Even though the aforementioned model is the most sufficient one among the ones we tested, there are several factors that prevent our team from further advancing the system for the bank's cash management. Firstly, the amount of business context provided is insufficient; second of all, the given dataset only contains 22,000 data points, which are likely to not be fully representative; and last but not least, the knowledge and expertise we have been equipped to carry out this project are not adequate for us to apply more sophisticated models. With such limitations, we suggest that the bank may create a model for segmenting its customers in the first place, and then examine their various traits and habits to select the optimal solution regarding predicting the withdrawal demand from different target customers. To increase the predictability of the model, more parameters should be included in addition to the withdrawal demand's type.

2. Business background

The bank needs to build a model to forecast the cash withdrawal in every ATM to optimize cash management system. With that in mind, this business has identified several factors that may affect the withdrawal demand. The response variable and covariate variables are described in the following table.

Variable	Description
Withdraw	The total cash withdrawn a day (in 1000 local currency)
Shops	Number of shops/restaurants within a walkable distance (in 100)
ATMs	Number of other ATMs within a walkable distance
Downtown	=1 if the ATM is in downtown, 0 if not
Weekday	=1 if the day is weekday, 0 if not
Center	=1 if the ATM is located in a center (shopping, airport, etc), 0 if not

High	=1 if the ATM has a high cash demand in the last month, 0 if not

3. Process overview

We started by cleaning the data - checking for any null values or abnormalities that may affect the quality of our analysis. We then conducted our EDA - using the following main tools: a heat matrix of correlations, residual plotting, and histogram (for dummy/ categorical variables). The aim of the EDA process is to detect potential multicollinearity as well as getting an overview of relationships among variables. This will aid the feature engineering process - i.e., choosing which variables to drop or add.

For our model building, we're trying several methods applicable for different combinations of independent variables and terms: classic OLS, Lasso, and deep learning. In the end, we will compare the MSE of each model and take into account the model complexity to choose the most suitable models.

4. Data preparation

The first step for data preparation is to import all necessary library and load both the training and the real test dataset using NumPy. After loading the dataset from the given csv file, our team moved forward to check the information and the description of the dataset. After that, we clean the data by checking whether the following criterias are satisfied. First, we checked if it was necessary to remove any irrelevant or duplicated observations from the given dataset. The next step was to fix any present structural errors, which can result in mislabeled categories or classes. Moving forward, our team examined if the dataset needed to be run through a cleansing program to deal with missing data or blank spaces, since many algorithms will not accept missing values. From there, we checked if there were any unwanted data outliers that should be filtered out in order to improve the overall performance of the dataset. And last but not least, the final step that we performed regarding cleaning data was to validate and authenticate the given dataset, to examine if it was of high quality, consistent, and properly formatted for the upcoming processes. After multiple procedures of data cleaning, our team concluded that the dataset was ready and of adequate condition to be performed on.

5. EDA

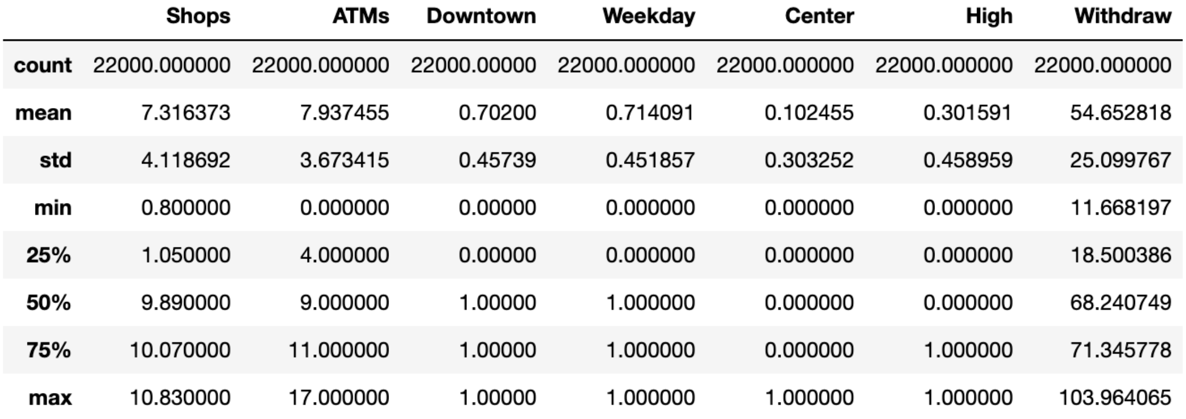
a. Data information

i. Training data

The training data given has 22,000 data points with no null values, and "Withdraw" (the amount of money taken from the ATM) is the response/ dependent variable. Among the variables, 4 are dummies ("Downtown", "Weekday", "Center", "High") and 2 are numerical ("Shops", "ATMs").

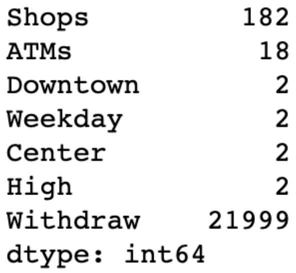
On average, there are around 7 shops and 8 ATMs around a given ATM. The mean amount withdrawn in each observation is $54.6 and the standard deviation is about 25.1.

In this dataset, the number of unique values are as in figure 1.



	Shops	ATMs	Downtown	Weekday	Center	High	Withdraw
count	22000.000000	22000.000000	22000.00000	22000.000000	22000.000000	22000.000000	22000.000000
mean	7.316373	7.937455	0.70200	0.714091	0.102455	0.301591	54.652818
std	4.118692	3.673415	0.45739	0.451857	0.303252	0.458959	25.099767
min	0.800000	0.000000	0.00000	0.000000	0.000000	0.000000	11.668197
25%	1.050000	4.000000	0.00000	0.000000	0.000000	0.000000	18.500386
50%	9.890000	9.000000	1.00000	1.000000	0.000000	0.000000	68.240749
75%	10.070000	11.000000	1.00000	1.000000	0.000000	1.000000	71.345778
max	10.830000	17.000000	1.00000	1.000000	1.000000	1.000000	103.964065

Figure 1: Training data description



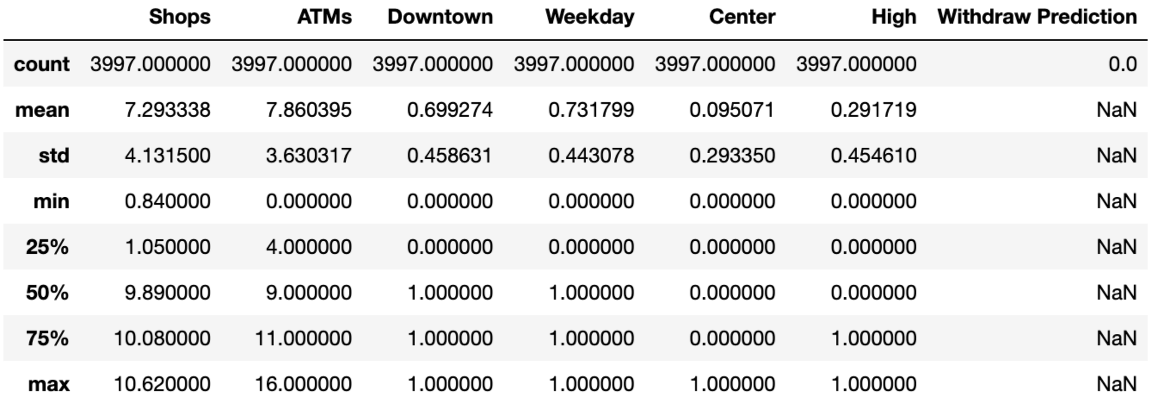
```
Shops          182
ATMs            18
Downtown         2
Weekday          2
Center           2
High             2
Withdraw     21999
dtype: int64
```

Figure 2: Number of training data unique values

ii. Test data

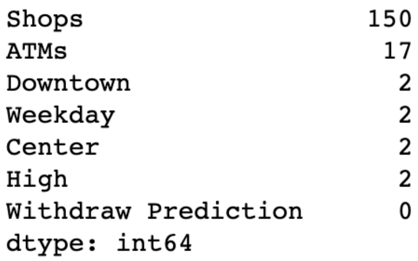
We also look into the test data to see if there is any significant difference. In general, the mean and the standard deviation of each variable are not much far from those of the training dataset.

In terms of unique values, the number of Shops unique values is slightly lower, but this can be attributed to the smaller pool of data.



	Shops	ATMs	Downtown	Weekday	Center	High	Withdraw Prediction
count	3997.000000	3997.000000	3997.000000	3997.000000	3997.000000	3997.000000	0.0
mean	7.293338	7.860395	0.699274	0.731799	0.095071	0.291719	NaN
std	4.131500	3.630317	0.458631	0.443078	0.293350	0.454610	NaN
min	0.840000	0.000000	0.000000	0.000000	0.000000	0.000000	NaN
25%	1.050000	4.000000	0.000000	0.000000	0.000000	0.000000	NaN
50%	9.890000	9.000000	1.000000	1.000000	0.000000	0.000000	NaN
75%	10.080000	11.000000	1.000000	1.000000	0.000000	1.000000	NaN
max	10.620000	16.000000	1.000000	1.000000	1.000000	1.000000	NaN

Figure 3: Test data description



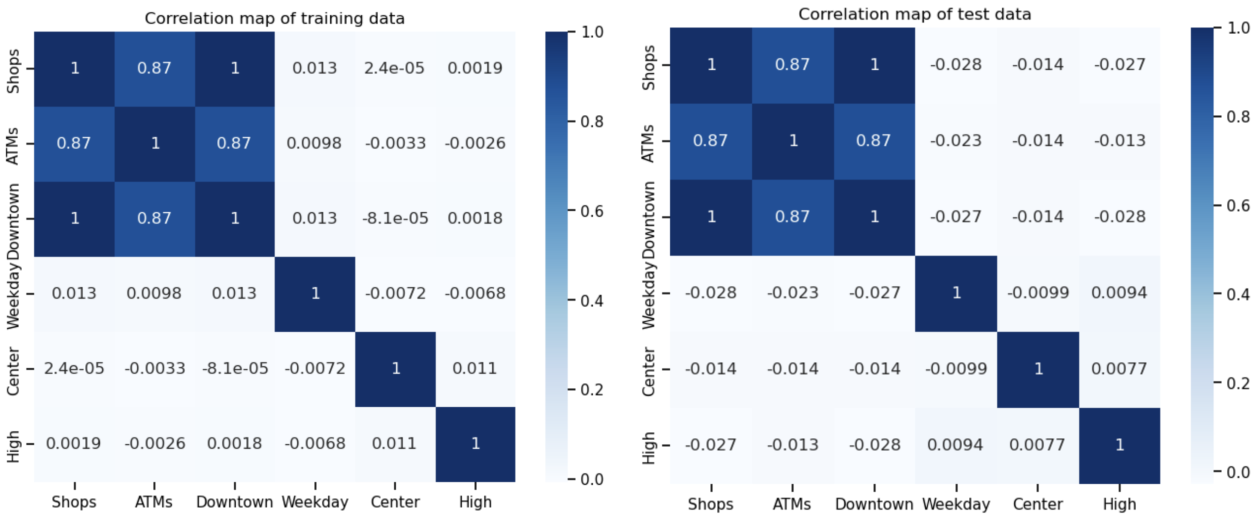
```
Shops                   150
ATMs                     17
Downtown                  2
Weekday                   2
Center                    2
High                      2
Withdraw Prediction       0
dtype: int64
```

Figure 4: Number of test data unique values

b. Correlation matrix (heat matrix)

We then checked the independent variables' correlation to find out potential multicollinearity.

With an initial correlation matrix (attached below), it is clear that 3 pairs of variables have a strong correlation with one another: "Shops" and "ATMs" (correlation coefficient 0.87), "Shops" and "Downtown" (correlation coefficient 1), and "ATMs" and "Downtown" (correlation coefficient 0.87). Looking at the test data also yields similar results. Therefore, we will prioritize detecting and mitigating multicollinearity in our methodology.



Correlation map of training data

	Shops	ATMs	Downtown	Weekday	Center	High
Shops	1	0.87	1	0.013	2.4e-05	0.0019
ATMs	0.87	1	0.87	0.0098	-0.0033	-0.0026
Downtown	1	0.87	1	0.013	-8.1e-05	0.0018
Weekday	0.013	0.0098	0.013	1	-0.0072	-0.0068
Center	2.4e-05	-0.0033	-8.1e-05	-0.0072	1	0.011
High	0.0019	-0.0026	0.0018	-0.0068	0.011	1

Correlation map of test data

	Shops	ATMs	Downtown	Weekday	Center	High
Shops	1	0.87	1	-0.028	-0.014	-0.027
ATMs	0.87	1	0.87	-0.023	-0.014	-0.013
Downtown	1	0.87	1	-0.027	-0.014	-0.028
Weekday	-0.028	-0.023	-0.027	1	-0.0099	0.0094
Center	-0.014	-0.014	-0.014	-0.0099	1	0.0077
High	-0.027	-0.013	-0.028	0.0094	0.0077	1

Figure 5: Correlation map of training and test data

c. Data visualization

i. Numerical variables

We first pair plotted the data to look for patterns among the independent variables, specifically the numerical ones ("Shops" and "ATMs"). There appears to be a clear segmentation - for observations with only 1 shop, there are usually a maximum of 7 ATMs nearby. Meanwhile, if there are 9-10 shops, there can be over 15 ATMs nearby.

This clustering supports our hypothesis of multicollinearity. These pairplots also show a tendency of data clusters. Hence, there may exist clusters with difference in characterisics and behaviour. In the context of this dataset, the dataset maybe a representation of different customer segmentations.

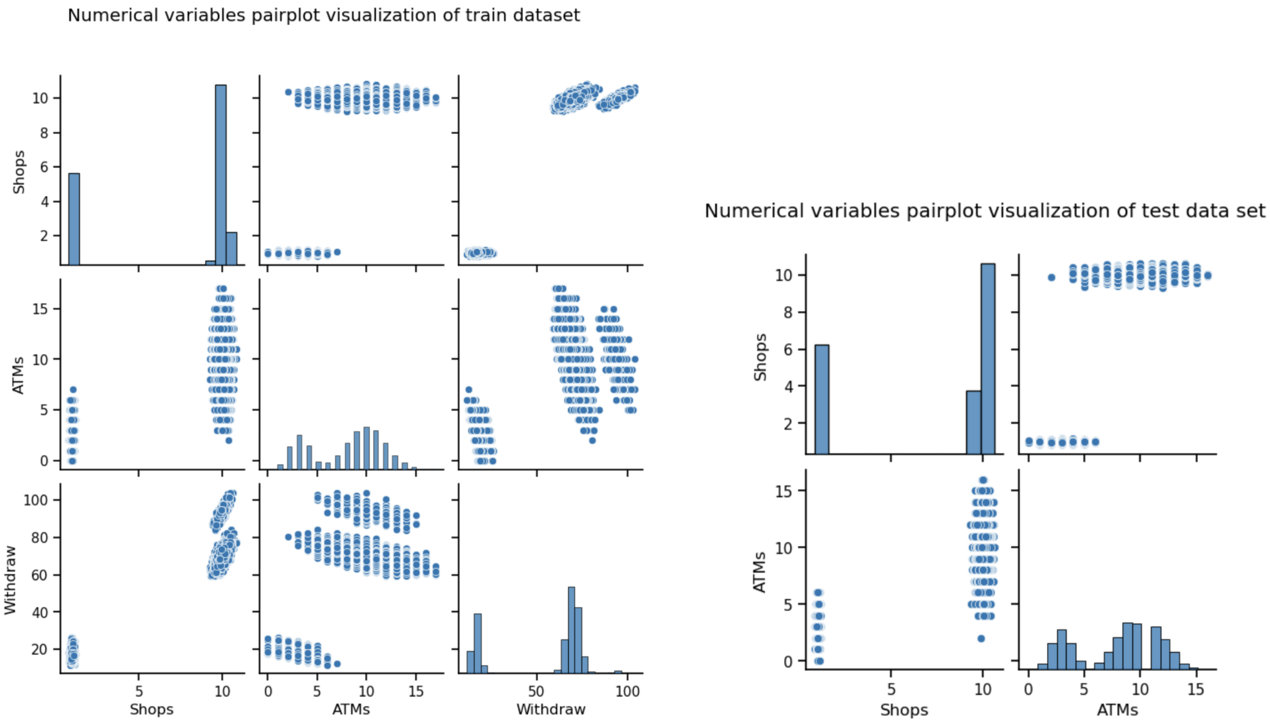


Figure 6: Numerical variables pairplot visualization of train and test dataset

ii. Categorical variables

Next, we analyzed categorical variables with histograms. Almost 2/3 of observations are made downtown, but only 1/10 are made in centers. Close to 2/3 are made on weekdays. Finally, 1/3 have witnessed high cash demands in the last month. In Center and High of the train dataset, class 0 outnumbered class 1, while in Downtown and Weekday, the situation was the opposite. Hence, the distribution of data was unequal, and the result may be biased.

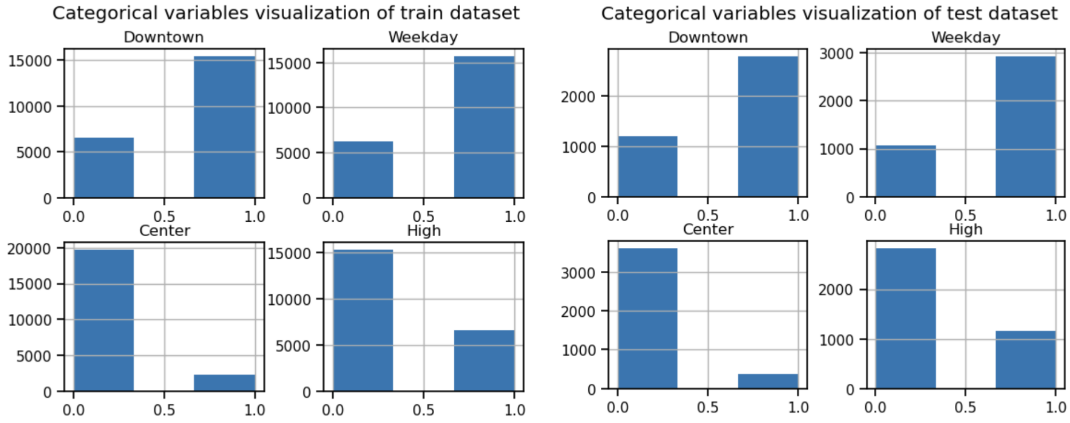


Figure 7: Categorical visualization of train and test dataset

iii. Quadratic terms analysis with 2 numerical variables: Shops and ATMs

We also residual plot the 2 numerical variables to see if we should add their quadratic terms as variables. However, as their residual plots are not U-shaped, this is likely not necessary.

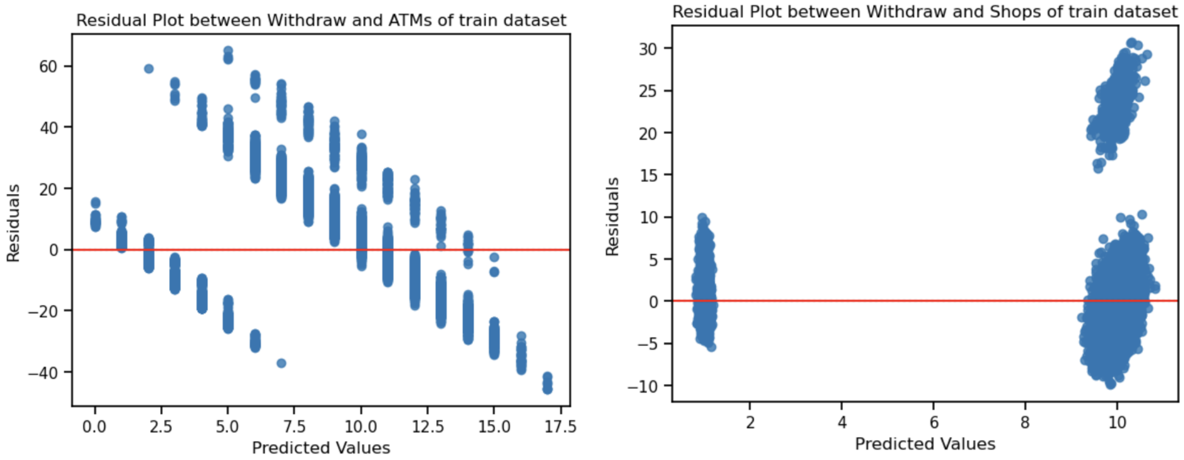


Figure 8: Residual plots of two numerical values with response variable

iv. Response variable

Finally, we plotted the response variable with a histogram to check its distribution. This have a special implication on the model-building. The result was not a normal bell-shaped distribution needed for the assumption of a linear regression model. However, it could indicate that another model may be a better fit for the model.

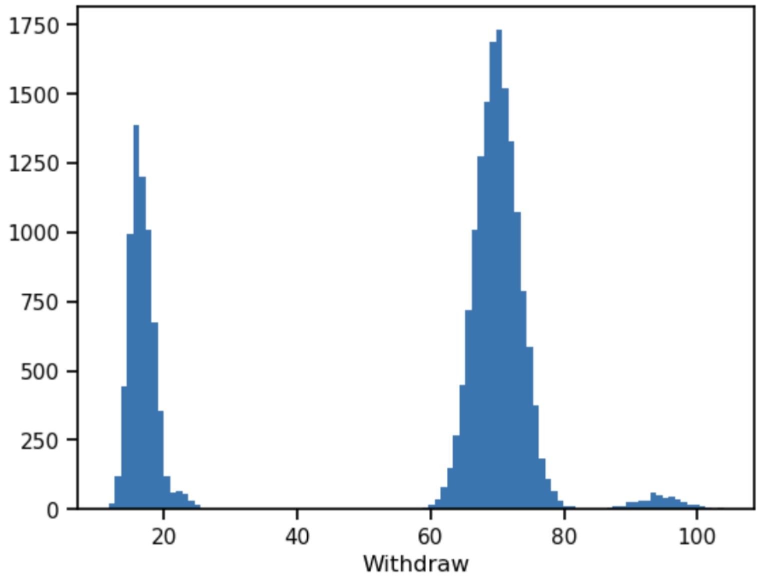
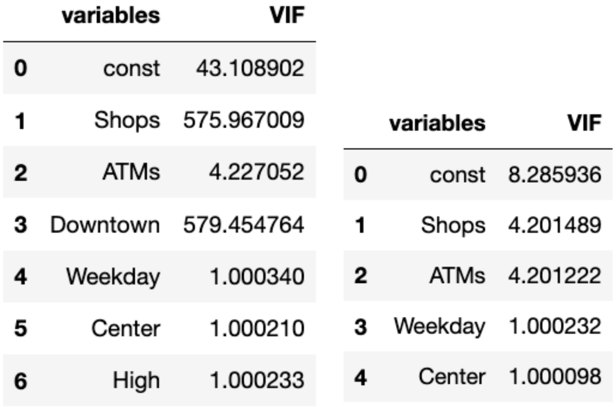


Figure 9: Response variable distribution

d. Feature engineering

Given the correlation heatmap above, we need to drop certain variables to reduce multicollinearity. We use VIF to analyze multicollinearity between independent variables and drop unnecessary variables. A high VIF score indicates a strong correlation of that variable with other variables. The results show that “Downtown” has the highest VIF, and by dropping it reduces the VIF scores of all other independent variables significantly to below the threshold 5.



	variables	VIF
0	const	43.108902
1	Shops	575.967009
2	ATMs	4.227052
3	Downtown	579.454764
4	Weekday	1.000340
5	Center	1.000210
6	High	1.000233

	variables	VIF
0	const	8.285936
1	Shops	4.201489
2	ATMs	4.201222
3	Weekday	1.000232
4	Center	1.000098

Figure 10: VIF score before and after dropping Downtown variable

We also add variables - specifically, the interaction terms, after dropping the Downtown variable. Since we do not have a business context to understand which variables are likely to interact with one another, we're adding all interaction terms possible to avoid biases.

e. Data processing

We begin by splitting the provided dataset into 2: one for training (70% of the data) and one for testing (internal validation for our own models- to be differentiated with the professor's testing dataset). We also standardized the data since there are large differences in the range of numerical variables. This will improve the data's uniformity and thus the model's accuracy.

6. Methodology and Model estimation

We choose 3 main methods: OLS, Lasso and Deep learning. These are 3 different methods for forecasting, so that we could have an overview about how each method perform. From then, we apply these methods for different combinations of variables, with both original data and additional variables (interaction terms).

1. Model 1: Forward stepwise OLS with the available variables (without interaction terms)
2. Model 2: OLS with available variables and interaction terms
3. Model 3: Lasso regression with available variables (without interaction terms)
4. Model 4: Lasso regression with available variables and interaction terms
5. Model 5: Deep learning - Neural Network

We would explain in detail the kinds of models and our steps to come up with the final model in each section. After that, we would aggregate the results into one section to compare and choose the most optimal model.

Note of why we choose Lasso regression over other methods for restricting multicollinearity:

1. We have remove the potential of multi-colinearity ("Downtown") in the EDA section above.

2. As a result of (1), we could choose a method that leads to a more sparse solution. Lasso does fit that criteria and also has better intepretability.
3. Lasso can shrink unnecessary variables to 0. This is extremely helpful when the model has many variables.

a. **Model 1: Forward stepwise OLS with the available variables (without interaction terms)**

Ordinary Least Square regression (OLS) is the technique that find the coefficients that minimize the residual sum of square the most, using for simple and multiple linear regression.

For the variable selection process, we choose forward stepwise and then add each variable one by one. We would investigate all available models (without interaction terms) by adding different cases of variables and compare their train and test MSE to choose significant variables. We would train the model using the train data set, calculate the MSE from the training dataset using cross value score method with cv=10, and calculate the MSE of the test data set. For the model selection process in this case, we would choose the combination with the lowest test MSE to go on with another variable.

The result is in the table below this paragraph. We can see that adding “High” variable would not affect the result significantly based on the MSEs, so our final variable selections for OLS are “Shops”, “Center”, “ATMs”, “Weekday”.

	Shops	**ATMs**	**Weekday**	**Center**	**High**
Train MSE	18.1049	204.3476	631.5952	628.099	632.8213
Test MSE	*16.9889*	197.3542	621.2501	618.2906	622.6064
	Shops + ATMs	**Shops + Weekday**	**Shops + Center**	**Shops + High**	
Train MSE	14.4483	15.4792	13.1324	17.8672	
Test MSE	13.7953	14.7094	*12.3041*	16.7526	
	Shops + Center + ATMs	**Shops + Center + Weekday**	**Shops + Center + High**		
Train MSE	9.5608	10.5693	12.9127		
Test MSE	*9.0945*	10.035	12.0981		
	Shops + Center + ATMs + Weekday	**Shops + Center + ATMs + High**			
Train MSE	7.0077	9.358			
Test MSE	*6.7296*	8.8972			

	Shops + Center + ATMs + Weekday + High				
Train MSE	6.814				
Test MSE	6.5408				

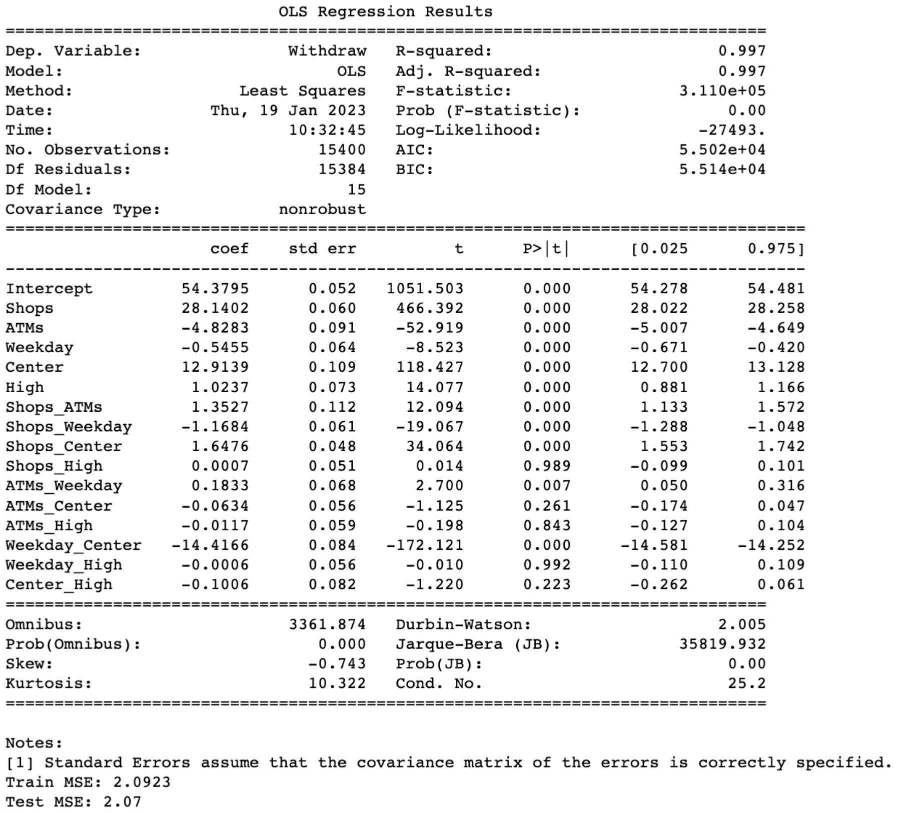
Figure 11: MSEs score for different selected variables

The final result of OLS will be discussed after this methodology section.

b. Model 2: OLS with available variables and interaction terms

As we do not have the business know-hows to accurately indicate which variables should involve in interaction terms, we choose the methodology of first adding all the interaction terms into the model (with all variables “High", “Shops”, “Center”, “ATMs”, “Weekday”) to train and test, then we would remove the interaction terms that have the P-values higher than 5%. The methodology of training and testing the model using OLS is the same as model 1 above. We do not use forward stepwise at this stage, since it would be time-consuming and inefficient.

The result’s model with all interaction terms is in the table below this paragraph. It indicates that *“Shops_High”, “ATMs_Center”, “ATMs_High”, “Weekday_High”*, and *“Center_High”* all have p-value higher than 5% and would be remove for the final result of model 2.



```
                            OLS Regression Results
==============================================================================
Dep. Variable:               Withdraw   R-squared:                       0.997
Model:                            OLS   Adj. R-squared:                  0.997
Method:                 Least Squares   F-statistic:                 3.110e+05
Date:                Thu, 19 Jan 2023   Prob (F-statistic):               0.00
Time:                        10:32:45   Log-Likelihood:                -27493.
No. Observations:               15400   AIC:                         5.502e+04
Df Residuals:                   15384   BIC:                         5.514e+04
Df Model:                          15
Covariance Type:            nonrobust
==================================================================================
                     coef    std err          t      P>|t|      [0.025      0.975]
----------------------------------------------------------------------------------
Intercept         54.3795      0.052   1051.503      0.000      54.278      54.481
Shops             28.1402      0.060    466.392      0.000      28.022      28.258
ATMs              -4.8283      0.091    -52.919      0.000      -5.007      -4.649
Weekday           -0.5455      0.064     -8.523      0.000      -0.671      -0.420
Center            12.9139      0.109    118.427      0.000      12.700      13.128
High               1.0237      0.073     14.077      0.000       0.881       1.166
Shops_ATMs         1.3527      0.112     12.094      0.000       1.133       1.572
Shops_Weekday     -1.1684      0.061    -19.067      0.000      -1.288      -1.048
Shops_Center       1.6476      0.048     34.064      0.000       1.553       1.742
Shops_High         0.0007      0.051      0.014      0.989      -0.099       0.101
ATMs_Weekday       0.1833      0.068      2.700      0.007       0.050       0.316
ATMs_Center       -0.0634      0.056     -1.125      0.261      -0.174       0.047
ATMs_High         -0.0117      0.059     -0.198      0.843      -0.127       0.104
Weekday_Center   -14.4166      0.084   -172.121      0.000     -14.581     -14.252
Weekday_High      -0.0006      0.056     -0.010      0.992      -0.110       0.109
Center_High       -0.1006      0.082     -1.220      0.223      -0.262       0.061
==============================================================================
Omnibus:                     3361.874   Durbin-Watson:                   2.005
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            35819.932
Skew:                          -0.743   Prob(JB):                         0.00
Kurtosis:                      10.322   Cond. No.                         25.2
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
Train MSE: 2.0923
Test MSE: 2.07
```

Figure 12: OLS Regression result of all variables (including interaction terms)

c. Model 3: Lasso regression with available variables (without interaction terms)

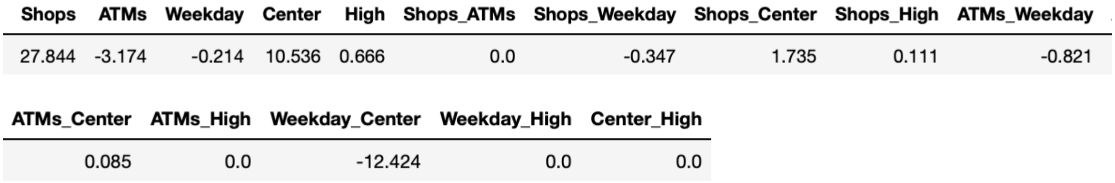
Lasso is one of methods to fix the multicollinearity. Lasso would have more spare solution compared to Ridge as it would restrict the estimation of coefficients towards zero in rhombus shape instead of square. The choice of Alpha here is important, as there is a trade-off in Alpha selection. If Alpha is too small, all variables are selected; if Alpha is too large, significant variables might be removed.

Using the models with all variables (excluding the interaction terms), we first choose Alpha using the cross validation method with cv = 5. The result of Alpha here is 0.0248, and it indicates that all variables selected is significant.

We would then compute the train MSE and test MSE similar to OLS to compare the results.

d. Model 4: Lasso regression with available variables and interaction terms

To take into account the interaction terms, we first still choose the optimal Alpha = 0.0248 by using the cross validation method with cv = 5 for all variables. There are some variables shrink to 0, so this model will include only the following variables: Shops, ATMs, Weekdays, Center, High, Shops_Weekday, Shops_Center, Shops_High, ATMs_Weekday, ATMs_Center, Weekday_Center. Table below indicates the results of coefficients:



Shops	ATMs	Weekday	Center	High	Shops_ATMs	Shops_Weekday	Shops_Center	Shops_High	ATMs_Weekday
27.844	-3.174	-0.214	10.536	0.666	0.0	-0.347	1.735	0.111	-0.821

ATMs_Center	ATMs_High	Weekday_Center	Weekday_High	Center_High
0.085	0.0	-12.424	0.0	0.0

Figure 13: Coefficients result of Lasso regression with interaction terms

Again, we would then compute the train MSE and test MSE similar to OLS to compare the results.

e. Model 5: Deep learning - Neural Network Model

For our final model, among other deep learning models, we opt for Neural Network, as it is a robust method for prediction and can solve highly complex tasks. Moreover, since Neural Network can capture non-linear relationship, we do not need to include any interaction terms or transform any variable (except for the standardization process).

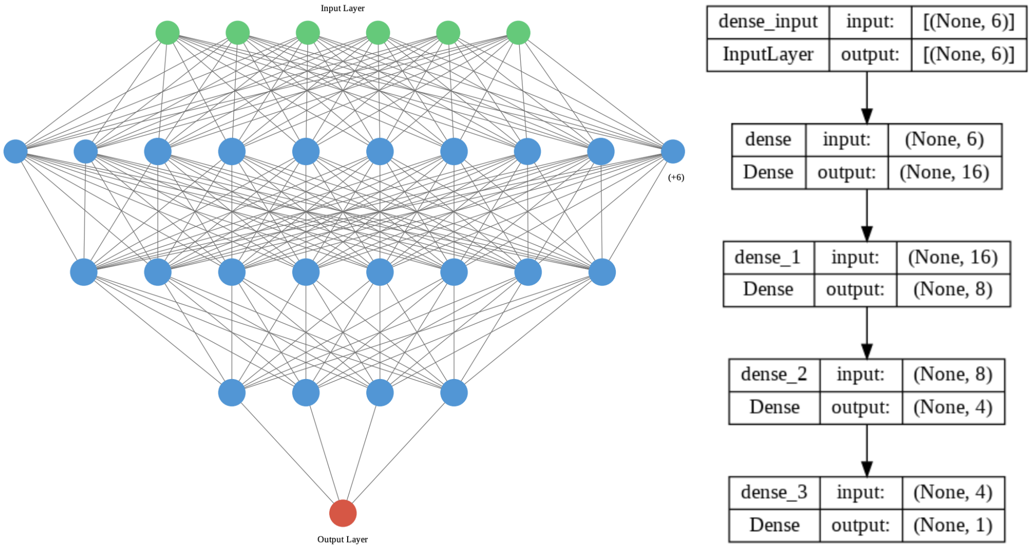
In this model, we assume that multicollinearity is not a big issue with Neural Network, since Neural Network can capture nonlinearity. Therefore, we still include Downtown in our model.

We build a simple neural network model with 5 layers in total, and the number of nodes are 6,16,8,4,1, respectively (left-base pyramid shape). As for the number of layers, we follow the conventional use of “rule of thumb”:

1. Start with 2-3 hidden layers (does not include the first and the last layers)
2. Number of nodes of intermediate layers should be a number from the geometric progression of 2, e.g., 4, 8, 16, 32,...
3. Activation function: use Relu for intermediate layers

For the activation functions, we made trials and error to choose the suitable activation functions: sigmoid, tanh and relu. Eventually, we all choose Relu, as we want to capture non-linear relationships with a very robust method. Both sigmoid and tanh has a problem of gradient vanishing, and it does not work well when there are variables with very different scales as in this dataset (for example: numerical variables with categorica variables). Meanwhile, tanh is somewhat simlar to Sigmond.

For the optimizer, we choose Adam with its default learning rate.



dense_input	input:	[(None, 6)]
InputLayer	output:	[(None, 6)]

dense	input:	(None, 6)
Dense	output:	(None, 16)

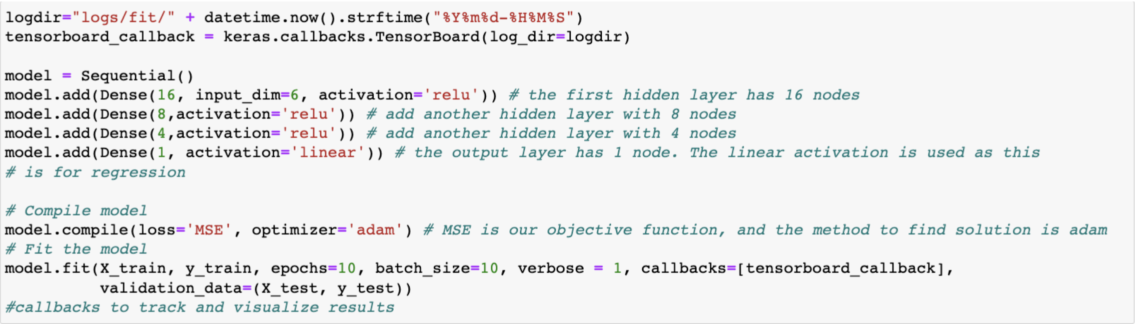
dense_1	input:	(None, 16)
Dense	output:	(None, 8)

dense_2	input:	(None, 8)
Dense	output:	(None, 4)

dense_3	input:	(None, 4)
Dense	output:	(None, 1)

Figure 14: Visualization of neural network model

The code for fitting NN model is as below with notes:



```
logdir="logs/fit/" + datetime.now().strftime("%Y%m%d-%H%M%S")
tensorboard_callback = keras.callbacks.TensorBoard(log_dir=logdir)

model = Sequential()
model.add(Dense(16, input_dim=6, activation='relu')) # the first hidden layer has 16 nodes
model.add(Dense(8,activation='relu')) # add another hidden layer with 8 nodes
model.add(Dense(4,activation='relu')) # add another hidden layer with 4 nodes
model.add(Dense(1, activation='linear')) # the output layer has 1 node. The linear activation is used as this
# is for regression

# Compile model
model.compile(loss='MSE', optimizer='adam') # MSE is our objective function, and the method to find solution is adam
# Fit the model
model.fit(X_train, y_train, epochs=10, batch_size=10, verbose = 1, callbacks=[tensorboard_callback],
          validation_data=(X_test, y_test))
#callbacks to track and visualize results
```

Figure 15: The code for fitting NN model

After fitting the model, the results seem to outperform any other aboved mention model of OLS and Lasso in terms of predicting. The MSE of both train and test are near to each other and as low as 0.26, indicating that there are no overfitting.

However, it may be hard to interpret the relations of those variables, since NN is not good for interpretation and implication.

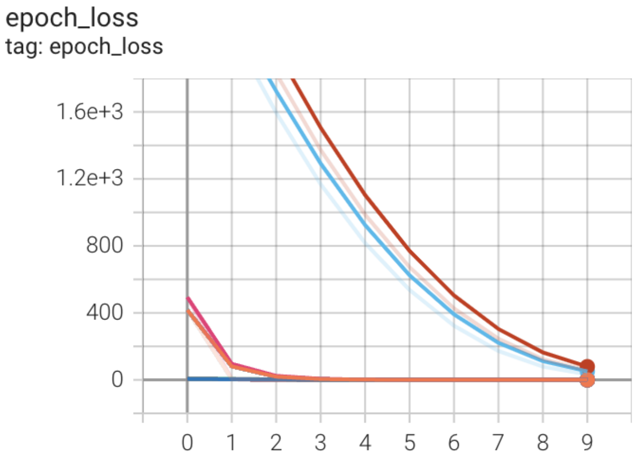
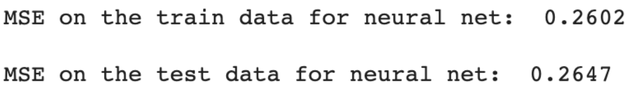


Figure 16: MSE loss after each epoch iteration



```
MSE on the train data for neural net:  0.2602

MSE on the test data for neural net:  0.2647
```

Figure 17: MSE output for both train and test data

Note that since deep learning is not always deterministic, so that the MSE result could slightly change, depends on the environment it is implemented on. In this report, the MSE evaluation for Neural Network has been conducted on Google Colaboratory, so it is slightly different compared to that of Jupyter Notebook.

7. Model selection

In the EDA part, it can be observed that the real test dataset has the similar general trend and pattern as the training dataset. Hence, we could make assumption that the model will work well and does not have much difference in test error compared to Train MSE and Test MSE. In this part, we would delve into the interpretation of each model discussed and compare them to find the optimal solution.

The summary of all models could be found in the table below. In the table, original variables mean all original variables without Downtown (except for the case of model 5), since we have removed Downtown out of our model due to multicollinearity. In the business context, we could clearly see that if the ATM is located in downtown, there is a high chance that the number of shops near ATM is also high, so the Downtown variable is highly correlated with other variables.

The process for model selection is based purely on the data analysis and interpretation of each model. The criteria for model selection is through the test MSE and the complexity of the model.

	Model 1	Model 2	Model 3	Model 4	Model 5
Method	Forward stepwise OLS	OLS with all variables	Lasso	Lasso	Neural network
Input variables	Original variables	Original variables with all interaction terms	Original variables	Original variables with all interaction terms	Original variables (with Downtown)
Train MSE	7.0077	2.0885	6.8366	2.2569	0.2602
Test MSE	6.7296	2.0703	6.54	2.2159	0.2647
Final variables included	Shops, Center, ATMs, Weekday	Shops, ATMs, Weekday, Center, High, Shops_ATMs, Shops_Weekday, Shops_Center, ATMs_Weekday, Weekday_Center	Shops, ATMs, Weekday, Center, High	Shops, ATMs, Weekdays, Center, High, Shops_Weekday, Shops_Center, Shops_High, ATMs_Weekday, ATMs_Center, Weekday_Center	Original variables
Variables with negative correlations	ATMs, Weekday	ATMs, Weekday, Shops_Weekday, Weekday_Center	ATMs, Weekday	ATMs, Weekday, Shops_Weekday, ATMs_Weekday, Weekday_Center	N/A
Interaction terms		x		x	(taken into account in the model itself)
Complexity (based on the number of variables and the method)	Low	Medium	Low	Medium	High

Figure 18: Overall results of all models

We could clearly see model 1 and 3 has the highest MSEs among groups, indicating that the model would be better if we include interaction terms. Hence, we would not choose model 1 and 3.

For model 2 and 4 that have the interaction terms, it indicates they are better models with lower MSEs around 2. Interaction terms accoun for better models also can be explained in the business context. For example, interaction term Shops_High indicates the increase in number of shops would increases the high cash demand.

However, if we look at the regression lines of these models 2 and 4 in the figure below, we could see some actual data is out of best fit lines, indicating that actual data might not best fit in the linear regression model for prediction.

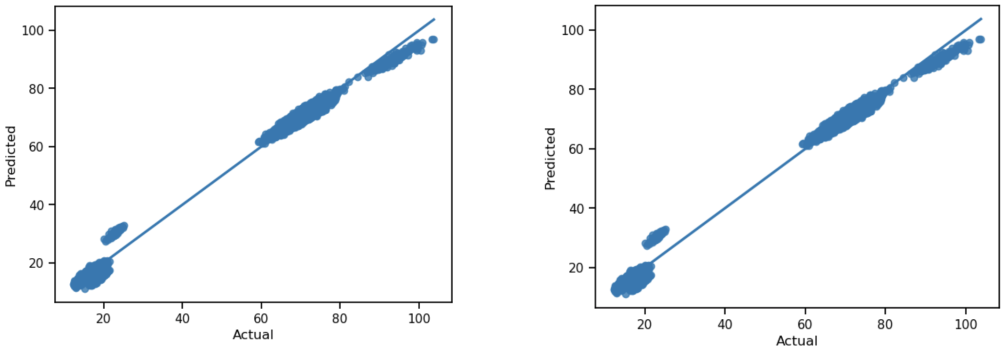


Figure 19: Regression lines of model 2 (left) and model 4 (right)

Comparing those models with the metric - Test MSE, the Neural Network is currently outperforming other models. This model has the lowest both train and test MSE, as the nature of neural network is to optimize the non-linear model (not only in linear fashion like others) and to include the interaction terms while modifying the variables. Therefore, model 5 is our final model for predicting cash withdrawal.

8. Limitations

a. Lack of business context

Since we're provided data without any additional context about the banks or the ATM networks, we cannot make several decisions that will enhance the accuracy of our models (e.g., which interaction terms to choose).

b. Lack of comprehensive data

Since we're only provided 22,000 data points (which we assume to not be fully representative of the ATMs' activity since its beginning), the training data may not provide the full picture. For instance, there may be cyclical trends not observed during the given time period.

c. Lack of ML techniques

Due to a lack of expertise, we're unable to apply more advanced ML techniques which might provide additional insights for the model.

9. Suggestions

Although the simple model built in this report could predict the demand, the business could take further actions to improve predictability of withdrawal demand and the comprehensiveness of the problem.

As the data visualization shows, there might exist different data clusters in the model. Hence, to enhance the accuracy when predicting demand for withdrawal, the business could first build a model for customer segmentation, then investigate into different characteristics and behaviors of the customer. After that, the decision maker could decide which way is the best for predicting the demand of different target customers.

The data analysis also shows that in most cases, ATMs and Weekday have a negative correlation with the final response variable, showing that these two factors may have different trends with other variables.

As for the nature of withdrawal demand, other factors should also be taken into consideration to improve the model predictability, such as time-series data. For example, the demand could be influenced by the time in the day, or the kind of date (example: normal day and holiday), etc.