ECON3203 Group Report

This report was produced by ECON3203 Group 31:
Andrew William (z5331372)
Dan Visitchaichan (z5331849)
Daniel Dinh (z5312318)
James Lu (z5317269)
Dylan Upton (z5308844)

Problem and Objective

In this project, we face the important problem of optimising the bank's cash management by making data-driven decisions on when to reload its ATM network.

Our objective is to develop a model that can predict the cash demand, specifically the amount of money withdrawn in a day using a series of factors.

Scope

The main deliverables of this project are a report that will outline the team's analysis of what is the best model in predicting withdrawal amount. We will also supply a separate document containing the code used to train our models and for predicting on the test set.

It will be our goal to present our findings concisely so they can be understood by both technical and non-technical people. The report and models will be developed based on the following data provided:

- ATM_training.csv
- ATM_test.csv

Both ATM_training and ATM_test contain features of ['Shops', 'Weekday', 'Center', 'High', 'ATMs', 'Downtown'], and the response variable is 'Withdraw'.

In this report, we will detail the exploratory data analysis we executed on these data, the modelling process we underwent with relevant justifications, and will present our findings regarding the performances of different models. We will then conclude our report by summarising what our chosen best model is for test set predictions.

Exploratory Data Analysis (EDA):

The training data we received has 22000 rows and 7 columns and no NULL data was present in either train or test sets. We are given that 'Withdraw' is the response variable (dependent variable) and the predictors (independent variables) are 'Shops', 'ATMs', 'Downtown', 'Weekday, 'Center' and 'High'. Of these predictor variables, we observe that 'Downtown', 'Weekday', 'Center' and 'High' are all binary variables. From this we also observe that the scales for each variable varied quite significantly, this means our models may benefit significantly from standardisation.

	Shops	ATMs	Downtown	Weekday	Center	High	Withdraw
Shops	1.000000	0.872903	0.999131	0.013014	0.000004	0.001820	0.985797
ATMs	0.872903	1.000000	0.873726	0.009766	-0.003306	-0.002616	0.824030
Downtown	0.999131	0.873726	1.000000	0.012664	-0.000101	0.001782	0.983574
Weekday	0.013014	0.009766	0.012664	1.000000	-0.007153	-0.006793	-0.050470
Center	0.000004	-0.003306	-0.000101	-0.007153	1.000000	0.010521	0.088103
High	0.001820	-0.002616	0.001782	-0.006793	0.010521	1.000000	0.021275
Withdraw	0.985797	0.824030	0.983574	-0.050470	0.088103	0.021275	1.000000

Figure 1: correlation matrix of train set variables

Referring to the correlation matrix above, it is clear that the variables 'Shops', 'Downtown', 'ATMs' and 'Withdraw' have a high positive correlation to each other, relative to the other variables (>0.85 correlation). This high correlation implies that 'Shops', 'Downtown' and 'ATMs' each individually are good predictors of 'Withdraw'. However, since 'Shops', 'Downtown' and 'ATMs' are covariates, the high correlation between these 3 variables poses the problem of multicollinearity. This may lead to issues such as high variance of estimated coefficients in linear regression. Next we look at the scatterplots of 'Downtown' against the highly correlated predictors.

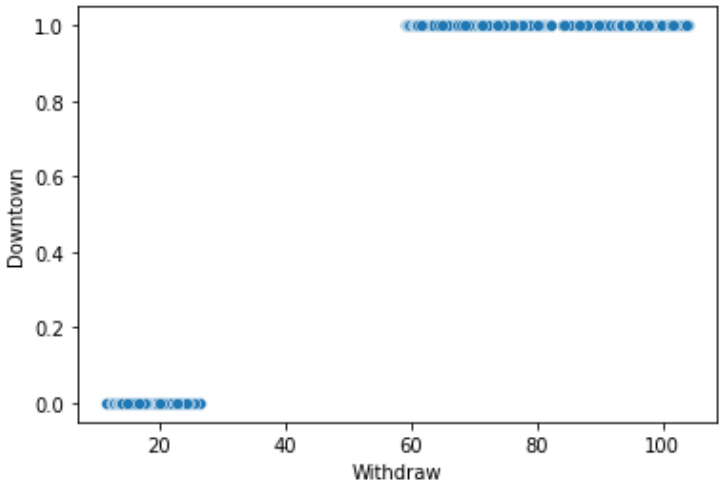


Figure 2: Scatterplot of 'Withdraw' against 'Downtown'

Looking closer at the scatterplot between ‘Downtown’ and ‘Withdraw’, shows a positive correlation, where when Downtown = 1, we observe that the values of ‘Withdraw’ are higher and when Downtown = 0, we observe lower values of ‘Withdraw’.

```
----- Downtown = 0 -----
count    6556.000000
mean       16.762582
std         1.976004
min        11.668197
25%        15.406924
50%        16.490919
75%        17.870472
max        26.383424
Name: Withdraw, dtype: float64
```

```
----- Downtown = 1 -----
count    15444.000000
mean        70.737277
std          5.251994
min         59.118507
25%         67.870035
50%         70.120582
75%         72.454415
max        103.964065
Name: Withdraw, dtype: float64
```

Figure 3: description of ‘Withdraw’ when ‘Downtown’ = 0 and ‘Downtown’ = 1

The above description shows that the values of ‘Withdraw’ given ‘Downtown’ = 0 do not overlap with the values of ‘Withdraw’ when ‘Downtown’ = 1. From this we infer that ‘Downtown’ may be a significant predictor for ‘Withdraw’ since the data is linearly separable.

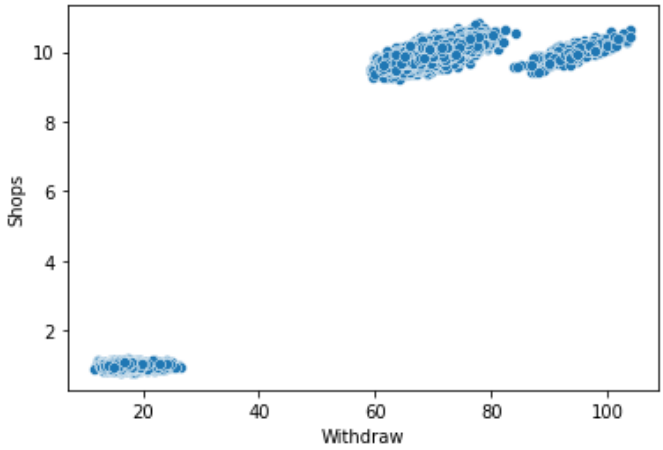


Figure 4: Scatterplot of ‘Withdraw’ against ‘Shops’

Similarly, we observe in the scatterplot between ‘Shops’ and ‘Withdraw’ that for greater values of ‘Shops’, we also see greater values of ‘Withdraw’, so there is a positive correlation between ‘Shops’ and ‘Withdraw’.

```
----- Shops < 5 -----
count    6556.000000
mean       16.762582
std         1.976004
min        11.668197
25%        15.406924
50%        16.490919
75%        17.870472
max        26.383424
Name: Withdraw, dtype: float64
```

```
----- Shops >= 5 -----
count    15444.000000
mean        70.737277
std          5.251994
min         59.118507
25%         67.870035
50%         70.120582
75%         72.454415
max        103.964065
Name: Withdraw, dtype: float64
```

Figure 5: description of ‘Withdraw’ when ‘Shops’ < 5 and ‘Shops’ >= 5

Again, we try to see if there is an overlap between when ‘Shops’ is less than 5 and greater than or equal to 5. From the above information, there is no overlap which indicates that ‘Shops’ is a significant predictor for ‘Withdraw’.

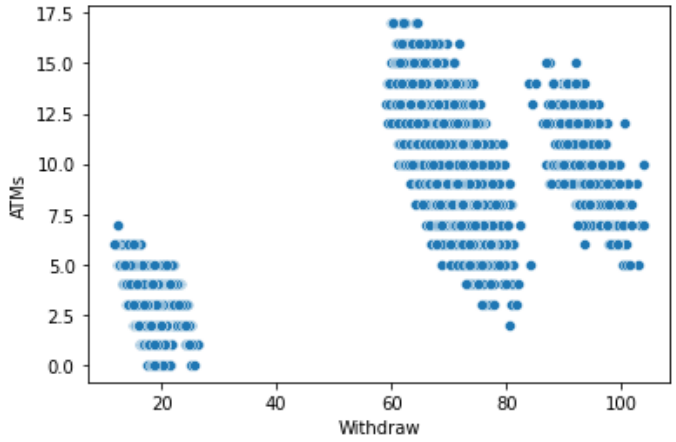


Figure 6: Scatterplot of ‘Withdraw’ against ‘ATMs’

Finally, looking at the scatterplot between ‘ATMs’ and ‘Withdraw’ our final highly correlated predictor variable, we can see a positive correlation between ‘ATMs’ and ‘Withdraw’. We also notice that the values seem to be clustered into 3 clusters.

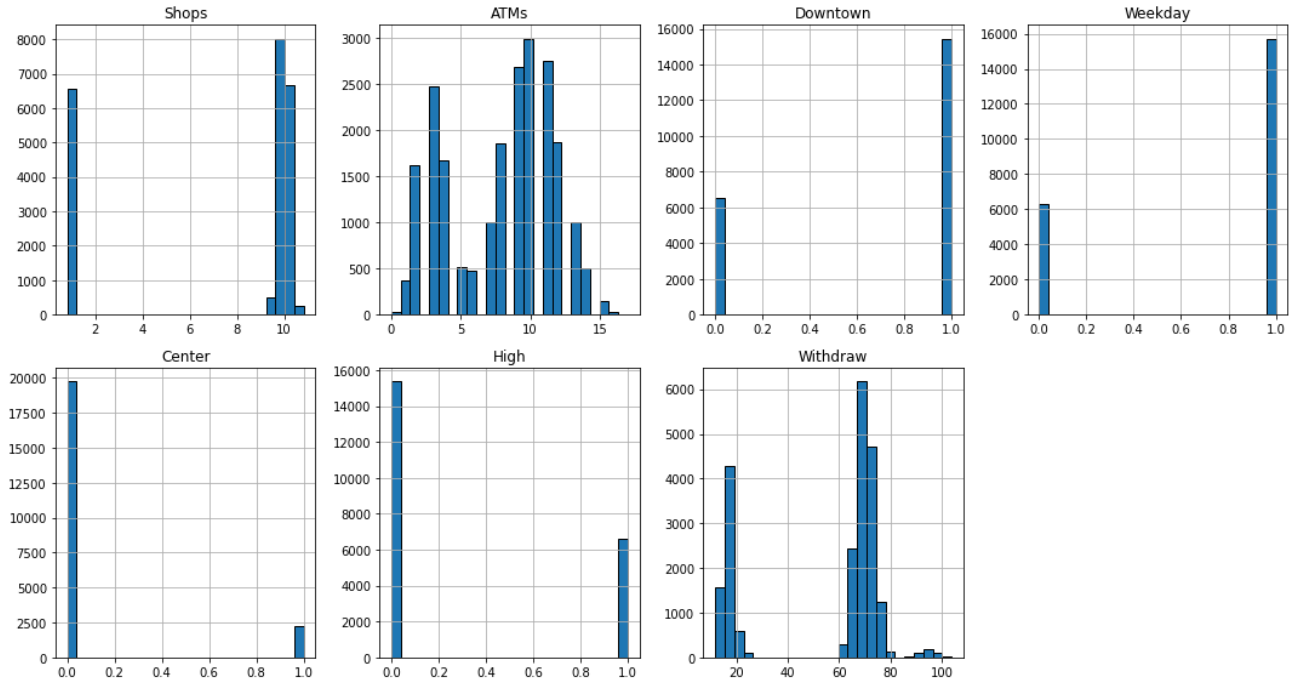
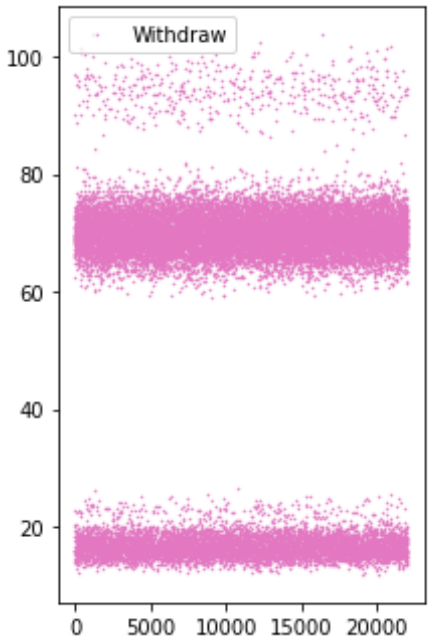


Figure 7: Histogram of each variable in dataset



Interestingly, looking at the histogram for ‘Withdraw’ in *Figure 7* we see 3 distinct regions within the distribution where there are more data points. This corresponds with the 3 clusters in *Figure 6*. Additionally, from the *Figure 8* scatterplot we observe regions of the ‘Withdraw’ distribution with no data points. We hypothesise that this may be a result of the lack of overlap in ‘Downtown’ and ‘Shops’ and high correlation as well between ‘Withdraw’ and ‘ATMs’. This further supports the idea that the highly correlated features are very significant in predicting the value of ‘Withdraw’.

We also note that there is a small negative correlation between ‘Withdraw’ and ‘Weekday’, a small positive correlation between ‘Withdraw’ and ‘Center’, and almost no correlation between ‘Withdraw’ and ‘High’.

Figure 8: Scatterplot of ‘Withdraw’ values in the train dataset

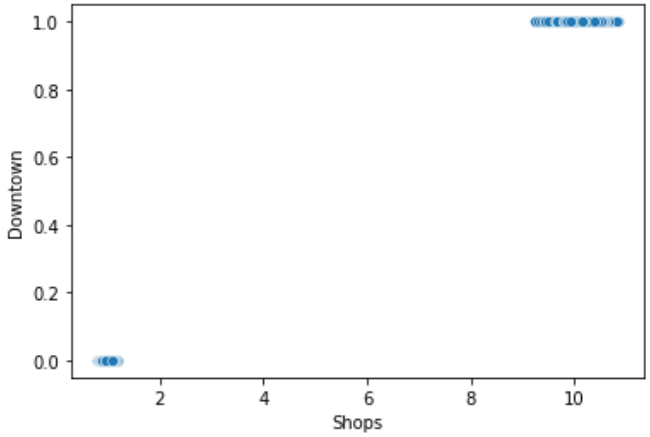


Figure 9: Scatterplot of ‘Shops’ against ‘Downtown’

Returning to the highly correlated predictors ‘Shops’ and ‘Downtown’, as we see from the above scatterplot of the two variables there is a very strong positive correlation (of 0.9991) between these features. To solve the issue of multicollinearity, we can remove the two highly correlated variables and create a new interaction variable (e.g. by adding/multiplying them together).

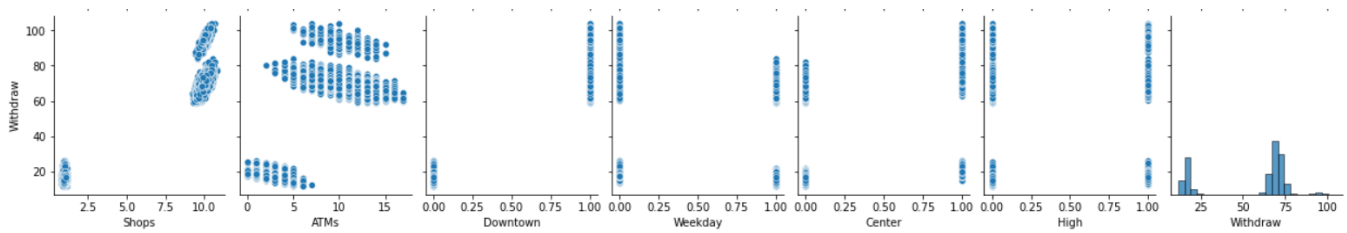


Figure 10: Scatterplot of 'Withdraw' against the predictors

Another thing to note, the scatterplots with 'Withdraw' always seem to be clustered as seen with the multiple clusters in each plot. This may potentially indicate that using a non-linear method may produce better results than if a linear method was to be used.

Feature engineering and selection

The highly correlated covariates (Shops, DownTown and ATMs) result in a multicollinearity problem when we run linear regression type models such as multiple linear regression (MLR), ridge regression, lasso regression and elastic net regression. To prevent this, we investigated how to combine these variables.

Firstly, it is worth noting that the high correlation between 'Shops', 'Downtown' and 'ATMs' makes sense as "shops" are the number of shops in hundreds that are in walkable distance, "ATMs" are the number of atms in tens that are in walkable distance and "downtown" is if there are atms downtown. Clearly we can see that if an atm is downtown, this is more than likely going to be in walkable distance, hence ATMs and downtown are correlated. And if an ATM is in walkable distance, many shops will also be in walkable distance too. Hence why the three covariates are highly correlated.

To counter the highly correlated covariates we created several ways of combining the correlated covariates. We ran a linear regression with each combination method as a predictor, and included all the other predictors ('Weekday', 'Center', 'High'), with 'Withdraw' as the target variable. We then chose the best combination based on which combination method maximized the adjusted R^2, whilst also taking into account the AIC and BIC scores.

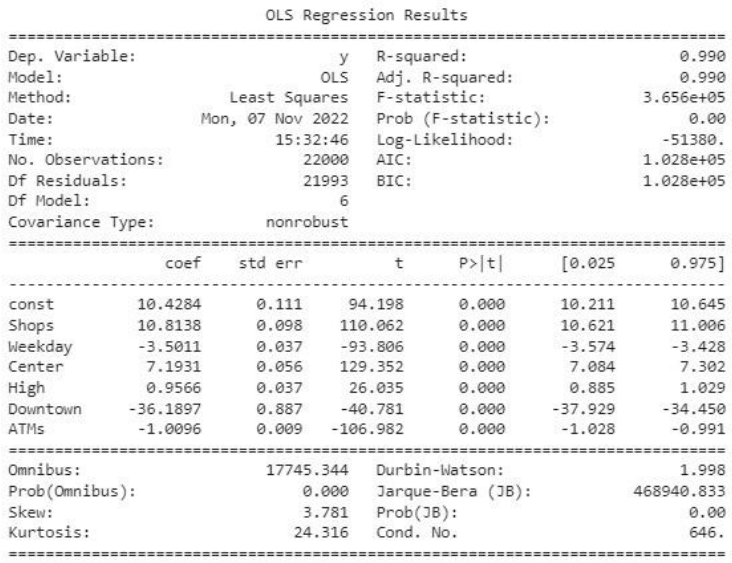
The results were as follows:

- Only include shops - predictors: ['Shops', 'Weekday', 'Center', 'High'].
    - 0.9840 adjusted R^2
    - AIC 1.135e+05
    - BIC 1.136e+05
- Only include ATMs - predictors: ['ATMs', 'Weekday', 'Center', 'High'].
    - 0.6910 adjusted R^2

    - AIC/BIC 1.784e+05
- Only include Downtown - predictors: ['Downtown', 'Weekday', 'Center', 'High'].
    - 0.9790 adjusted R^2
    - AIC 1.188e+05
    - BIC 1.189e+05
- Combining correlated features through addition - predictors: ['Shops' + ‘Downtown’ + ‘ATMs’, 'Weekday', 'Center', 'High'].
    - 0.9030 adjusted R^2
    - AIC/BIC 1.530e+05
- Combining correlated features through multiplication - predictors: ['Shops' * ‘Downtown’ * ‘ATMs’, 'Weekday', 'Center', 'High'].
    - 0.8230 adjusted R^2
    - AIC 1.661e+05
    - BIC 1.662e+05

Based on these results, we identified the best combination method to be keeping the variable ‘Shops’ and dropping the variables ‘Downtown’ and ‘ATMs’ since this results in the highest adjusted R^2 and lowest AIC and BIC. This means the best predictors after combining correlated covariates were ['Shops', 'Weekday', 'Center', 'High'].

We then did further feature engineering by introducing feature interactions to this set of predictors, combining them through multiplication. Specifically, we generated feature interactions of degrees up to 3 and ran a regression with all features as predictors. Finally, we selected the suitable feature interactions based on their p-values in the regression, removing feature interactions that had a p-value greater than 0.05. The results were as follows:



OLS Regression Results

Dep. Variable:	y	R-squared:	0.990
Model:	OLS	Adj. R-squared:	0.990
Method:	Least Squares	F-statistic:	3.656e+05
Date:	Mon, 07 Nov 2022	Prob (F-statistic):	0.00
Time:	15:32:46	Log-Likelihood:	-51380.
No. Observations:	22000	AIC:	1.028e+05
Df Residuals:	21993	BIC:	1.028e+05
Df Model:	6		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	10.4284	0.111	94.198	0.000	10.211	10.645
Shops	10.8138	0.098	110.062	0.000	10.621	11.006
Weekday	-3.5011	0.037	-93.806	0.000	-3.574	-3.428
Center	7.1931	0.056	129.352	0.000	7.084	7.302
High	0.9566	0.037	26.035	0.000	0.885	1.029
Downtown	-36.1897	0.887	-40.781	0.000	-37.929	-34.450
ATMs	-1.0096	0.009	-106.982	0.000	-1.028	-0.991

Omnibus:	17745.344	Durbin-Watson:	1.998
Prob(Omnibus):	0.000	Jarque-Bera (JB):	468940.833
Skew:	3.781	Prob(JB):	0.00
Kurtosis:	24.316	Cond. No.	646.

Figure 11: Regression with all original covariates as predictors.

OLS Regression Results

Dep. Variable:	y	R-squared:	0.993
Model:	OLS	Adj. R-squared:	0.993
Method:	Least Squares	F-statistic:	2.341e+05
Date:	Mon, 07 Nov 2022	Prob (F-statistic):	0.00
Time:	15:33:01	Log-Likelihood:	-46997.
No. Observations:	22000	AIC:	9.402e+04
Df Residuals:	21985	BIC:	9.414e+04
Df Model:	14		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
1	11.5444	0.065	177.122	0.000	11.417	11.672
Shops	5.9427	0.008	762.107	0.000	5.927	5.958
Weekday	-2.0827	0.077	-26.991	0.000	-2.234	-1.931
Center	2.9656	0.186	15.959	0.000	2.601	3.330
High	1.0304	0.116	8.856	0.000	0.802	1.258
Shops Weekday	0.0119	0.009	1.297	0.195	-0.006	0.030
Shops Center	2.0119	0.022	93.526	0.000	1.970	2.054
Shops High	-0.0080	0.014	-0.581	0.561	-0.035	0.019
Weekday Center	0.0171	0.212	0.081	0.936	-0.399	0.434
Weekday High	0.0153	0.137	0.112	0.911	-0.253	0.283
Center High	-0.0871	0.248	-0.351	0.725	-0.573	0.399
Shops Weekday Center	-2.0081	0.024	-83.429	0.000	-2.055	-1.961
Shops Weekday High	0.0093	0.016	0.577	0.564	-0.022	0.041
Shops Center High	0.0163	0.024	0.684	0.494	-0.031	0.063
Weekday Center High	-0.0067	0.216	-0.031	0.975	-0.430	0.417

Omnibus:	168.132	Durbin-Watson:	2.004
Prob(Omnibus):	0.000	Jarque-Bera (JB):	264.427
Skew:	-0.013	Prob(JB):	3.81e-58
Kurtosis:	3.536	Cond. No.	272.

Figure 12. Regression with all feature interactions.

OLS Regression Results

Dep. Variable:	y	R-squared:	0.993
Model:	OLS	Adj. R-squared:	0.993
Method:	Least Squares	F-statistic:	5.463e+05
Date:	Mon, 07 Nov 2022	Prob (F-statistic):	0.00
Time:	15:33:01	Log-Likelihood:	-47000.
No. Observations:	22000	AIC:	9.401e+04
Df Residuals:	21993	BIC:	9.407e+04
Df Model:	6		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	11.4691	0.038	298.660	0.000	11.394	11.544
Shops	5.9507	0.004	1680.246	0.000	5.944	5.958
Weekday	-1.9733	0.032	-61.939	0.000	-2.036	-1.911
Center	2.9510	0.093	31.776	0.000	2.769	3.133
High	1.0340	0.030	34.340	0.000	0.975	1.093
Shops Center	2.0129	0.014	144.601	0.000	1.986	2.040
Shops Weekday Center	-2.0024	0.012	-169.230	0.000	-2.026	-1.979

Omnibus:	168.177	Durbin-Watson:	2.004
Prob(Omnibus):	0.000	Jarque-Bera (JB):	264.421
Skew:	-0.014	Prob(JB):	3.82e-58
Kurtosis:	3.536	Cond. No.	58.1

Figure 13: Regression with only significant feature interactions retained.

The result of our feature engineering left us with significant interaction terms of Shops*Center, Shops*Weekday*Center. We note from *Figure 13* that the regression including the new interaction terms resulted in an adjusted R-squared score of 0.993 and AIC of 94013.6683, which was an improvement over the regression using the original dataset covariates (*Figure 11*), which had an adjusted R-squared score of 0.990 and AIC of 102773.7533. With this, we conclude that the best set of features (with correlated covariates dropped and feature interactions) for linear regression type models was ['Shops', 'Weekday', 'Center', 'High', 'Shops Center', 'Shops Weekday Center'].

Modelling approach

In this section, we describe the models we tried, experiments we ran, and how we conducted the training and selection of the best model for our final predictions. Before we begin, it is worth noting the list of all predictors includes [‘Shops’, ‘ATMs’, ‘Downtown’, ‘Weekday’, ‘Center’, ‘High’]. The response variable is ‘Withdraw’.

Models used

The models we tried in our project include Multiple Linear Regression, Lasso Regression, Ridge Regression, Elastic Net Regression, K-Nearest Neighbours Regressor (KNN) and Neural Network. From here on we will periodically use the term ‘linear regression type models’ to refer to the first 4 regression models mentioned.

Experiments

Other than trying a variety of models, we also tried a variety of combinations of data preprocessing methods for our dataset. This led us to run our entire model training and testing process on 6 experiments, where each experiment had a different set of preprocessing techniques applied to the dataset.

The experiments are as follows:

1. Experiment 1 - Raw data
2. Experiment 2 - Standardised data
3. Experiment 3 - Regular data with correlated covariates combined
4. Experiment 4 - Standardised data with correlated covariates combined
5. Experiment 5 - Regular data with correlated covariates combined and feature interactions
6. Experiment 6 -Standardised data with correlated covariates combined and feature interactions

Note that the term ‘standardised data’ refers to standardising the predictors by subtracting mean and dividing by standard deviation.

In experiments 1 and 2, we used all predictors from the raw data : ['Shops', 'Weekday', 'Center', 'High', 'ATMs', 'Downtown'].

In experiments 3 and 4, we combined the correlated covariates using the method mentioned in our Feature Engineering and Selection section, so the predictors used were : ['Shops', 'Weekday', 'Center', 'High'].

In experiments 5 and 6, we extended the processing in experiments 3 and 4 by introducing feature interactions as mentioned in the Feature Engineering and Selection section. The final list

of predictors was: ['Shops', 'Weekday', 'Center', 'High', 'Shops Center', 'Shops Weekday Center'].

Model Training & Testing

In each experiment, before we started the modelling process, the first thing we did was split the 'ATM_training.csv' into a train set and validation set, using a random 70-30 split (70% train, 30% validation) seeded by a random state to ensure consistent split across experiments. We conducted model parameter selection and fitted all the models **only** using the train set data. We chose to use this split because we did not have a full test set, hence we made an assumption that a randomly split validation set would reflect the test set and could hence indicate how well each model might perform on the full test set. This helped us ensure that we were not overfitting on the train set, reducing variance.

The best parameters were selected during training via several methods, then the models were applied to the validation set. The validation set mean squared error (MSE) was computed for each model, and the model with the lowest validation MSE was picked as the winner model for that experiment. This would be repeated for each of the 6 experiments.

The experiment and model combination that led to the lowest overall validation MSE was then chosen to be used for test set predictions. We used the winning experiment preprocessing techniques, and re-trained the winning model for that experiment on the entire 'ATM_training.csv', before predicting on the test set, 'ATM_test.csv'.

Model Parameter Selection Method

The first two techniques for parameter selection were applied to all models **except neural networks.** The third technique discusses hyperparameter selection for neural networks.

#1 - Cross-Validation (CV)

We applied cross-validation with 5 folds on the train set, and used the train set CV metric to determine the best hyperparameters for each model. Note that the metric to optimise in the cross-validations were limited to what option was available in the package we used for this part (*sklearn)*.

For lasso and ridge regression, we optimised the lambdas (penalty term) based on minimising the CV R^2 - we were able to also try CV MSE for ridge regression (ridgeCV allowed this).

For elastic net regression, we optimised the lambdas (penalty term) and l1_ratio term based on minimising the CV R^2.

The grid of lambda parameters we searched through for ridge regression and elastic net regression include 200 lambdas between e^-10 and e^20, defined by '`np.exp(np.linspace(-10,20,200))`', which was the range we used in tutorials. Note that

lasso regression in *sklearn* created their own set of lambdas to search through based on the dataset, so we did not use our search space for lasso regression.

In elastic net regression, the l1_ratio parameter we searched through included 13 numbers range between 0.01 and 0.99, specifically: '`[0.01,0.05,0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,0.95,0.99]`'. This range followed the material taught in tutorials.

For KNN regressor, we optimised the number of neighbours K using the optimization metric of CV MSE. We searched through 20 values of K in a linear space between 1 and 61 with steps of 3, implemented by the code '`np.arange(1, 61,3)`'. This was similar to the range in tutorials.

Note that no parameter selection was necessary for multiple linear regression because its closed form solution from using ordinary least squares does not require external parameters unlike in the other regressions.

#2 - AIC and BIC

For lasso and ridge regression, we also implemented the selection of the lambda parameter based on what yielded the lowest AIC and BIC on the train set, through the *statsmodels* package. We searched through 200 lambdas (for both models) over an exponential range, defined by '`np.exp(np.linspace(-10,20,200))`' (same as in CV method). This range mimicked what was taught in tutorials.

#3 - Neural Network Parameter Selection

Unlike the non deep-learning based methods, hyperparameters for neural networks weren't selected using 5-fold cross validation and AIC/BIC. Cross validation was not performed for neural networks due to time constraints.

The hyperparameter tuning for neural networks was done in two passes. In the first pass, we defined a grid of hyperparameters to search through, trained the model with each set of parameters on the training set, and predicted on the validation set. The validation set MSE was used to determine the best hyperparameters from the first pass.

The hyperparameters tuned included the kernel_initializer: `['uniform', 'lecun_uniform', 'normal', 'zero', 'glorot_normal', 'glorot_uniform', 'he_normal', 'he_uniform']`,and hidden layer sizes: `[8,16,32,64]`. The number of hidden layers from 1-3 hidden layers were also tested for each experiment.

In the second pass, further hyperparameter tuning was done by hand starting from the best hyperparameters from the first pass. The main hyperparameter tuned at this stage was the number of nodes in each layer. Similar to the first pass, the hyperparameters that optimised the validation set MSE were chosen as the best neural network hyperparameters for an experiment.

Findings

The table below showcases our validation set MSE for all models in each experiment. The cell highlighted in green indicates the best result from each experiment, the cell highlighted in red indicates the worst result from each experiment.

Model	Validation Set MSE for Experiment #					
	1	2	3	4	5	6
KNN regression (K chosen from CV MSE)	0.4457	0.3681	3.4888	3.5087	3.5035	3.4907
Linear regression	6.2636	6.2636	10.4077	10.4077	4.1712	4.1712
Elastic net regression (lambda & l1_ratio chosen from CV R^2)	6.8404	6.7457	10.5222	10.4026	4.5162	4.1919
Ridge regression (lambda chosen from AIC)	6.2636	6.2636	10.4077	10.4077	4.1712	4.1712
Ridge regression (lambda chosen from BIC)	6.2636	6.2636	10.4077	10.4077	4.1712	4.1712
Ridge regression (lambda chosen from CV R^2)	6.2630	6.2635	10.4072	10.4077	4.1712	4.1712
Ridge regression (lambda chosen from CV MSE)	6.2630	6.2633	10.4077	10.4077	4.1712	4.1712
Lasso regression (lambda chosen from AIC)	6.7004	6.4798	10.4076	10.4077	4.1712	4.1712
Lasso regression (lambda chosen from BIC)	6.7004	6.4798	10.4076	10.4077	4.1712	4.1712
Lasso regression (lambda chosen from CV R^2)	6.8267	6.7391	10.5087	10.4019	4.5133	4.1904

** Results exclude neural networks, these will be discussed in the last part of this section.

Experiment 1 and 2

For both experiments 1 and 2, we note that KNN regression had the lowest validation set MSE of 0.4457 and 0.3681. The parameters K (number of neighbours) chosen during training that corresponds to these results were both K = 4. The worst result came from elastic net regression

with a validation MSE of 6.8404 and 6.7457. KNN's MSE was much lower than the other regression models, which had MSEs of around 6.

Experiment 3 and 4

For both experiments 3 and 4, we note that KNN regression had the lowest validation set MSE of 3.4888 and 3.5087. The parameters K (number of neighbours) chosen during training that corresponds to these results were K = 46 and 37 respectively. The worst result for experiment 3 came from elastic net regression with a validation MSE of 10.5222. For experiment 4, most of the ridge, lasso and linear regression models shared the worst results of 10.4077. Like before, all models other than KNN had a large MSEs of around 10, which were much more than KNN's MSE.

We note that there was a large increase in validation MSE for all models as we transitioned from experiment 1 & 2, to 3 & 4.

Experiment 5 and 6

For both experiments 5 and 6, we note that KNN regression had the lowest validation set MSE of 3.5035 and 3.4907. The parameters K (number of neighbours) chosen during training that corresponds to these results were K = 43 and 49 respectively. The worst result for experiment 3 came from elastic net regression with a validation MSE of 10.5222. For experiment 4, most of the ridge, lasso and linear regression models shared the worst results of 10.4077. Like before, all models other than KNN had a large MSEs of around 10, which were much more than KNN's MSE.

We note that there was a large decrease in validation MSE for linear regression type models, and similar performances in validation MSE for KNN from experiments 3 & 4 to 5 & 6.

Overall Results

For the linear regression type models (MLR, lasso, ridge, elastic net), the best validation MSE across all experiments was found in experiment 6 - where correlated covariates were dropped, feature interactions were introduced to the subset, and standardisation was applied. Their validation MSEs range from 4.1712 - 4.1919.

This was within expectations since our feature selection and engineering, which was performed to optimise the performance of linear regression (maximising adjusted R^2), yielded the data preprocessing techniques in experiment 6. Hence, linear regression type models naturally benefited from the engineered and selected features.

For the KNN regression model, the best validation MSE across all experiments was found in experiment 2, where the predictors were all the covariates from the data, with standardisation applied. It achieved a validation MSE of 0.3681, which was the lowest of all validation MSEs.

One last interesting result was that the K corresponding to the best results in experiments 1

and 2 were small with K = 4, when compared to the K corresponding to best results in experiments 3,4,5,6 (all K > 35). In other words, only a small number of similar observations were needed to predict the outcome for experiments 1 and 2, whereas a large number of similar observations was required for predictions in the other experiments. This indicated that for KNN, the predictive power of using all covariates from the raw data as predictors might be stronger than when we drop correlated covariates and introduce feature interactions to the smaller subset of covariates. Hence, the best features engineered for linear regression type models were not necessarily good features for other models such as KNN regression.

Neural Network

We chose to log the neural network results separately as its hyperparameter tuning was done differently to the other models. The following table documents the results of the first pass of hyperparameter tuning.

Experiment Number	Validation MSE	Model Parameters Number of hidden layers, kernel_initilizer, hidden layer (L1,L2,L3) sizes
1	1.8067	2 hidden layers, lecun_uniform, L1 = 64, L2 = 32
2	0.8192	2 hidden layers, he_normal, L1 = 64, L2 = 32
3	4.4390	2 hidden layers, lecun_uniform, L1 = 64, L2 = 32
4	3.5584	2 hidden layers, glorot_normal, L1 = 64, L2 = 64
5	4.0802	2 hidden layers, lecun_uniform, L1 = 64, L2 = 8
6	3.6133	2 hidden layers, lecun_uniform, L1 = 64, L2 = 32

We observe that 2 hidden layers consistently outperforms 1 or 3 hidden layers for each experiment. Also, the best number of nodes in the first hidden layer is 64 which may indicate that a larger hidden layer 1 size leads to better results. The best result from the first pass of hyperparameter tuning was in experiment 2, with a validation MSE of 0.8192 (2 hidden layers, he_normal, L1 = 64, L2 = 32).

In the second pass, we replicated experiment 2 settings, and further hand-tuned the hyperparameters starting from the first pass results. As a result, **the final best validation MSE of 0.2801** was achieved via the model parameters: {2 hidden layers, glorot_normal, L1 = 128, L2 = 64}.

Conclusion

Among the non deep-learning based methods, KNN had the best performance on the validation MSE of 0.3681. After extensive tuning of neural networks, we found that neural networks did

outperform KNN with a validation MSE of 0.2801. However, the difference between these two validation MSEs was extremely small. KNN is known to be a much simpler and interpretable model in comparison to neural networks, hence it is also more stable and reproducible than neural networks.

We believe that the simplicity and reproducibility of KNN outweighs the small increase in validation MSE performance provided by neural networks, hence, we pick KNN as our final model.

The excellent performance of KNN was not surprising since in the EDA section, we saw that the response variable ‘withdraw’ appears in clusters when plotted against the covariates in the pairplots, meaning that observations with similar values in their features would likely have similar values for ‘withdraw’. For a particular observation, KNN regression predicts the outcome by taking averages from the outcomes of other observations with similar features, making it very suitable in this application. **Hence, the best model we chose to predict on the test set was KNN with K = 4, with all covariates included as predictors and standardisation applied.**

Appendix

Experiment 1 Results

Parameters selected through CV on train set

```
Best LASSO Lambda: 0.10163842175930277
Best Ridge Lambda: 0.007639891638883671
Best Ridge Lambda based on MSE: 0.1
Best Elastic Net Lambda: 0.10266507248414422, Best L1_Ratio : 0.99
Best KNN Regressor K = 4
```

Parameters selected through AIC BIC on train set

```
Best Lasso Lambda based on BIC: 0.029670916964953026
Best Lasso Lambda based on AIC: 0.029670916964953026
Best Ridge Lambda based on BIC: 4.5399929762484854e-05
Best Ridge Lambda based on AIC: 4.5399929762484854e-05
```

Experiment 2 Results

Parameters selected through CV on train set

```
Best LASSO Lambda: 0.02471489457616553
Best Ridge Lambda: 0.004864064066783492
Best Ridge Lambda based on MSE: 0.1
Best Elastic Net Lambda: 0.02496453997592478, Best L1_Ratio : 0.99
Best KNN Regressor K = 4
```

Parameters selected through AIC BIC on train set

```
Best Lasso Lambda based on BIC: 0.008882980908164468
Best Lasso Lambda based on AIC: 0.008882980908164468
Best Ridge Lambda based on BIC: 4.5399929762484854e-05
Best Ridge Lambda based on AIC: 4.5399929762484854e-05
```

Experiment 3 Results

Parameters selected through CV on train set

```
Best LASSO Lambda: 0.10163842175930277
Best Ridge Lambda: 0.9509904521556567
Best Ridge Lambda based on MSE: 0.1
Best Elastic Net Lambda: 0.10266507248414422, Best L1_Ratio : 0.99
Best KNN Regressor K = 46
```

Parameters selected through AIC BIC on train set

```
Best Lasso Lambda based on BIC: 4.5399929762484854e-05
Best Lasso Lambda based on AIC: 4.5399929762484854e-05
Best Ridge Lambda based on BIC: 4.5399929762484854e-05
Best Ridge Lambda based on AIC: 4.5399929762484854e-05
```

Experiment 4 Results

Parameters selected through CV on train set

```
Best LASSO Lambda: 0.02471489457616553
Best Ridge Lambda: 0.33103616882803794
Best Ridge Lambda based on MSE: 0.1
Best Elastic Net Lambda: 0.02496453997592478, Best L1_Ratio : 0.99
Best KNN Regressor K = 37
```

Parameters selected through AIC BIC on train set

```
Best Lasso Lambda based on BIC: 4.5399929762484854e-05
Best Lasso Lambda based on AIC: 4.5399929762484854e-05
Best Ridge Lambda based on BIC: 4.5399929762484854e-05
Best Ridge Lambda based on AIC: 4.5399929762484854e-05
```

Experiment 5 Results

Parameters selected through CV on train set

```
Best LASSO Lambda: 0.10163842175930277
Best Ridge Lambda: 0.33103616882803794
Best Ridge Lambda based on MSE: 0.1
Best Elastic Net Lambda: 0.10266507248414422, Best L1_Ratio : 0.99
Best KNN Regressor K = 43
```

Parameters selected through AIC BIC on train set

```
Best Lasso Lambda based on BIC: 0.0001516447643109633
Best Lasso Lambda based on AIC: 0.0001516447643109633
Best Ridge Lambda based on BIC: 4.5399929762484854e-05
Best Ridge Lambda based on AIC: 4.5399929762484854e-05
```

Experiment 6 Results

Parameters selected through CV on train set

```
Best LASSO Lambda: 0.02471489457616553
Best Ridge Lambda: 0.08523769121745499
Best Ridge Lambda based on MSE: 0.1
Best Elastic Net Lambda: 0.02496453997592478, Best L1_Ratio : 0.99
Best KNN Regressor K = 49
```

Parameters selected through AIC BIC on train set

```
Best Lasso Lambda based on BIC: 4.5399929762484854e-05
Best Lasso Lambda based on AIC: 4.5399929762484854e-05
Best Ridge Lambda based on BIC: 4.5399929762484854e-05
Best Ridge Lambda based on AIC: 4.5399929762484854e-05
```