ECON3203: Project Report

ATM Cash Management using Predictive Modelling

By *Fantastic Four*

Ashleigh Feng (z5257997)
Ayra Islam (z5255744)
Charmaine Leow (z5255005)
Tiarne Lo (z5254839)

ECON3203 REPORT

Ashleigh Feng (z5257997) *Ayra Islam (z5255744)* *Charmaine Leow (z5255005)* *Tiarne Lo (z5254839)*

1. Short Summary: Background, Objective, Methods, Results, Conclusion

1.1 Background

ATM cash availability is key to Banks because the inability to meet the actual demand leads to customer dissatisfaction, while replenishing more than required leads to high costs. To avoid both cash under and over stacking, it is necessary to accurately forecast the cash amount in demand for each ATM and then optimise the replenishment activities accordingly. Our objective is to forecast with accuracy the daily cash demand per ATM, taking into account ATM historic withdrawals, ATM location and accessibility and calendar dates.

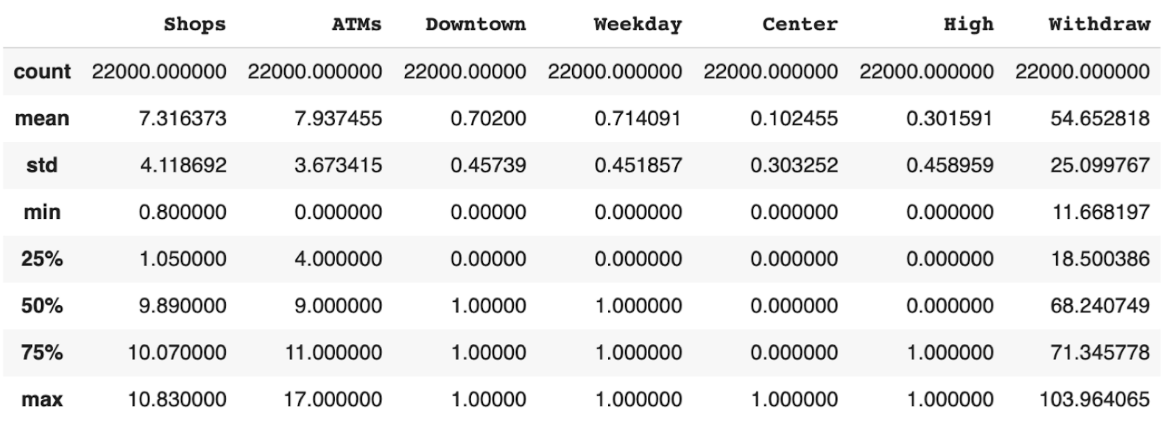
By understanding the bias variance tradeoff, we will build models which are generalisable and can perform well on test data, We will mitigate overfitting on the training data using regularised regression models, and will train and evaluate models built using lasso regression, ridge regression, elastic net regression and neural networks. Consequently, we will identify the model of best fit on the training data and then use these patterns to make predictions on unseen instances.

2. Exploratory Data Analysis

2.1 General Structure of the Dataset and Variables

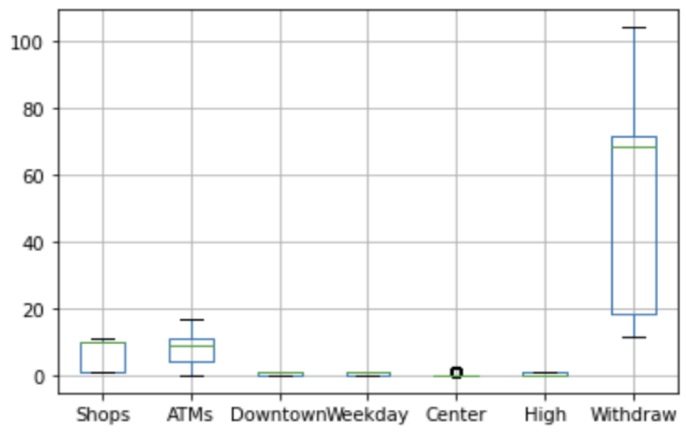
The dataset ATM_training.csv consists of 22000 observations and 7 variables: the response variable 'Withdraw' and independent variables 'Shops', 'ATMs', 'Downtown', 'Weekday', 'Center' and 'High'. Fortunately, there were no missing values in the dataset.

Furthermore, our team observed how each variable was distributed:

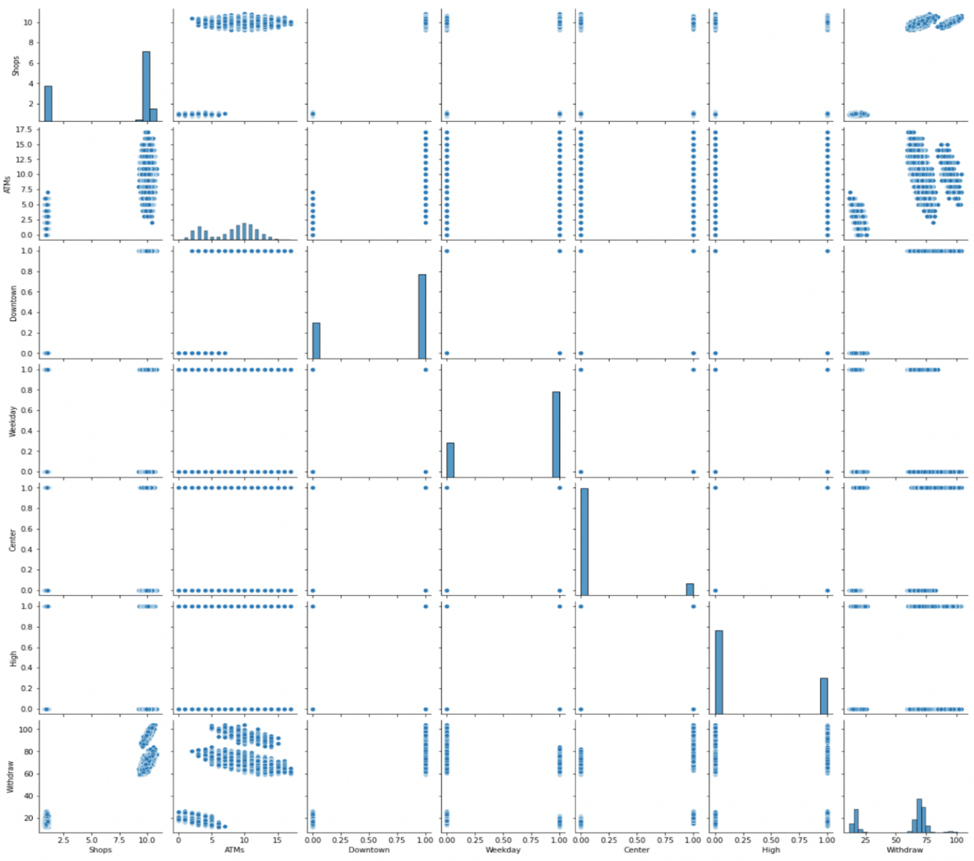


	Shops	ATMs	Downtown	Weekday	Center	High	Withdraw
count	22000.000000	22000.000000	22000.00000	22000.000000	22000.000000	22000.000000	22000.000000
mean	7.316373	7.937455	0.70200	0.714091	0.102455	0.301591	54.652818
std	4.118692	3.673415	0.45739	0.451857	0.303252	0.458959	25.099767
min	0.800000	0.000000	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

The distributions for 'Center' and 'High' are positively skewed because their means are higher than their corresponding median values. All other variables are negatively skewed. Furthermore, there seems to be no outliers because for each column the difference between the 3rd quartile and the maximum value is not significant [2.1.1]. The box plots below verify that there are no visible outliers:



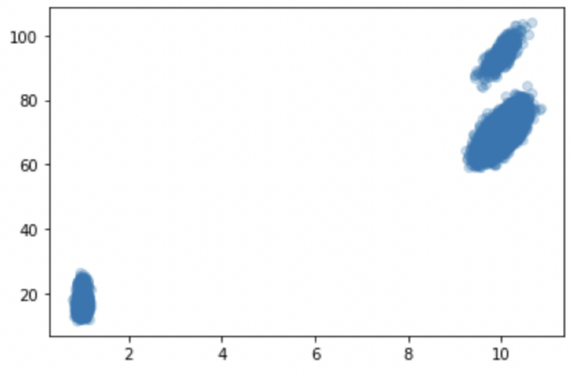
Our team also produced a pairwise plot to observe distributions of single variables and relationships between two variables [2.1.2].



Key Takeaways:

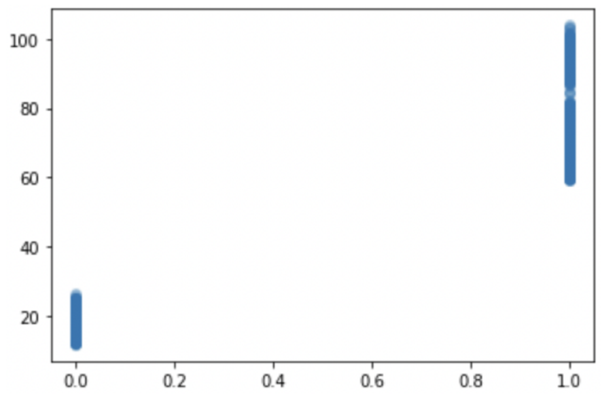
- ATMs and Withdraw were the only normally distributed variables.
- Linear relationships were observed between response variable Withdraw and dependent variables Shops, ATM and Downtown. Now, we will look at these relationships more closely.

1. **Withdraw vs Shops**



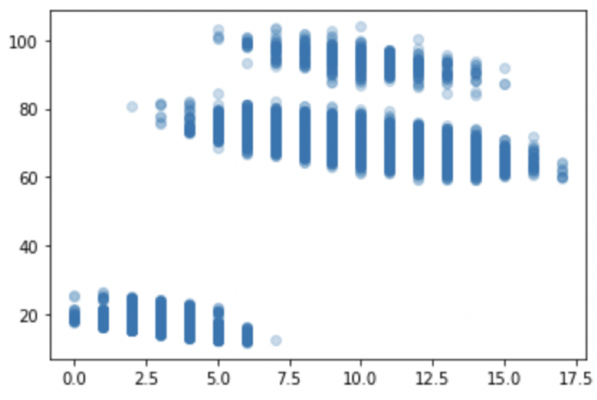
There exists a positive, linear relationship between Withdraw (y-axis) and Shops (x-axis).
When there are 10 shops, there appears to be two groups. The bottom cluster is larger than the smaller cluster above, meaning that there is a larger proportion of consumers who withdraw within the 60-80 range as opposed to consumers who withdraw within the 80-100.
Our group hypothesised that there could potentially be another factor involved in the relationship.

2. **Withdraw vs Downtown**



Positive linear relationship detected. Clearly, consumers located in Downtown are more likely to withdraw money.

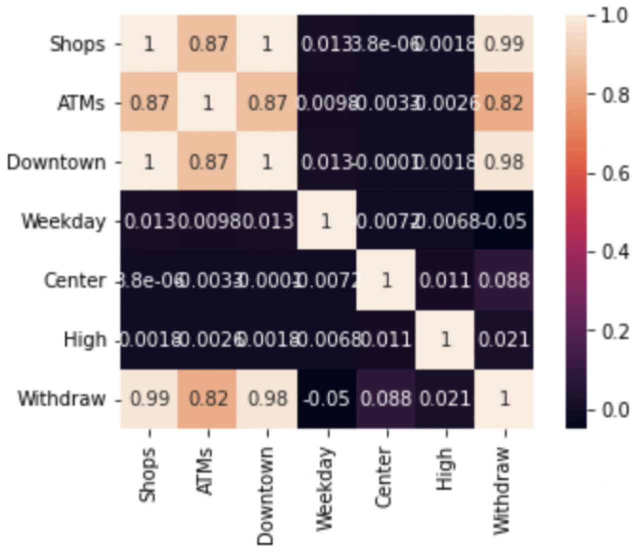
3. **Withdraw vs ATMs**



A positive linear relationship between ATMs and Withdraw is not as clear. ATMs within the 2.5 to 17.5 range experience higher customer withdrawals as opposed to ATMs within the 0.0 to 7.5 range. However, there are two different groups in the former group: a larger group of ATMs experience less withdrawal than the smaller group of ATMs within the 5.0 to 15.0 range. This result is similar to the result for the Withdraw vs Shops relationship and thus we concluded that there could be another factor interfering with this relationship.

2.2 Correlation matrix

As mentioned above, our group hypothesised that there could potentially be other factors interfering with the two variable relationships. We verified our results by constructing a correlation matrix:



This heatmap verified that Withdraw was positively and highly correlated with Shops (0.99), Downtown (0.98) and ATMs (0.82). Shops, ATMs and Downtown were highly correlated to each other, with Downtown and Shops achieving perfect positive correlation (1.00).

3. Methodologies

1. Linear Regression

Linear regression is a statistical modelling approach which maps a linear relationship of the response variable against one or more explanatory variables. The response variable (Y) must be continuous whereas the independent variables can either be continuous e.g. height or categorical e.g. sex.[3.1.1]

There are different types of linear regression models. One example is the univariate linear regression model, which demonstrates the relationship between the independent variable and a single dependent variable. In this study, we will construct a multivariate linear regression model whereby more than one independent variable is introduced into the model to explain the response/dependent variable. The model can be expressed in matrix notation $y = X\beta + \epsilon$.

Suppose there are n many observations in the dataset and p many predictors. Then,

- $y = \left(y_1, \dots, y_n\right)$ is a p dimensional vector of predicted values for the response variable.

- X is a $n \times (p + 1)$ matrix of all observed values of the independent variable including a constant vector corresponding to the intercept term.

- $\beta = \left(\beta_0, \beta_1, \dots, \beta_p\right)^T$ is a $p + 1$ dimensional vector of unknown regression coefficients including the intercept term. Each coefficient estimate provides a measure of how much the corresponding variable contributes to the response variable. For example, the withdrawal amount for a single customer will increase by β_2 for every unit increase in ATMs with all other independent variables held constant.

- Finally, $\epsilon = \left(\epsilon_1, \dots, \epsilon_n\right)^T$ is known as the error/noise term for all observations. It captures all other factors which explain the response variable. Assume that the expected value of the error term is 0 and its variance is σ^2.

Linear regression models are simple, easy to interpret and have been scientifically proven to make reliable predictions [3.1.2]. These models are versatile because they can be used more than just to map linear relationships. For example, polynomial terms can be added to a model to imitate non-linear relationships between two or more variables. Despite it being one of the basic forms of statistical modelling, it's flexibility and predictive power is the reason why it is still applicable in multiple areas of study including biology, behaviour and business [3.1.2].

2. Lasso Regression

LASSO regression, more formally known as Least Absolute Shrinkage and Selection Operator regression, is a "shrinkage and variable selection method for linear regression models" [3.2.1]. LASSO regression's main objective is to essentially find the most optimal subset of predictors from a model, 'optimal' meaning the subset that minimises the prediction error for the predetermined response variable, by imposing a penalty on the model equal to the absolute values of the coefficients' magnitudes, otherwise known as *L1 regularisation* [3.2.2] . In some cases, coefficients can be minimised all the way to zero, which would lead to its predictor variable being excluded from the model. Those predictor variables with regression coefficients of a non-zero value are "most strongly associated with the response variable" [3.2.1]. In the case of fairly large constraints imposed by the LASSO regression method, values for coefficients will shrink closer to zero; this is what leads to the more desirable 'sparse, simple' model with fewer predictors.

This, in fact, is one of the main reasons the LASSO regression method is so desirable - a less complex model with fewer parameters leads to a neater and simpler model, which in turn accounts for better interpretability than a complex model [3.2.3]. Additionally, LASSO regression is favoured over other regression methods for models that may show signs of (possibly high) levels of multicollinearity, as well as for models where one may hope to automate complicated, time consuming processes during model selection (eg. variable selection or elimination of unnecessary parameters) [3.2.3].

3. Ridge and Elastic Net Regression

Ridge regression still tries to find the best fit line by minimising the sum of squared errors. However, there is an additional penalty and is considered as the penalty for complex coefficients. This penalty is also referred to as the shrinkage penalty because it shrinks the estimators of our model to the true values. The penalty used by ridge regression is a type of shrinkage estimator called the ridge estimator. The ridge estimator is especially good at improving the least squares estimate when multicollinearity is present. [3.3.1] Mathematically, the shrinkage penalty is a tuning parameter lambda multiplied by the L2 regularization of the model coefficients. L2 regularization is the sum of the square of the coefficients of the ridge model. [3.3.2]

While Ridge regression uses L2 regularization as the (shrinkage) penalty and Lasso regression uses L1 regularization, ElasticNet regression can be thought of as a combination of lasso and ridge regression. Thus, the penalty used by ElasticNet is some combination of L1 regularization and L2 regularization. We still have the tuning parameter lambda that we need to specify, but lambda is then multiplied by some combination of L1 and L2 regularization. ElasticNet regularization incorporates penalties from both. When we build an ElasticNet model, there are two hyper parameters or design parameters that we have to tune. [3.3.3]

The first is a lambda which controls the magnitude of the shrinkage penalty applied. ElasticNet regression has one more tuning parameter - alpha or the ElasticNet mixing parameter. This mixing parameter actually determines what kind of regression model is built and we can use the value of alpha to build a ridge regression model, a lasso regression model or an ElasticNet model. Alpha controls the mix of penalties of L1 and L2 regularization, and lambda controls the magnitude of the penalty. [3.3.4]

Alpha values in ElasticNet regression are always between 0 and 1 while alpha = 0 corresponds to ridge regression. If alpha is set to 0, only L2 regularization is incorporated. An alpha value of between 0 and 1 can control this mix. [3.3.3]

OLS regression tries to fit the best fit line on our data by minimizing the sum of squared errors. This model may not always be a good one, especially if our dataset isn't large, and there is multicollinearity (when many attributes are related and have a linear relationship with one another) in our data, which is why we may need to turn to regularized regression models such as ridge and elastic net regression. [3.3.4]

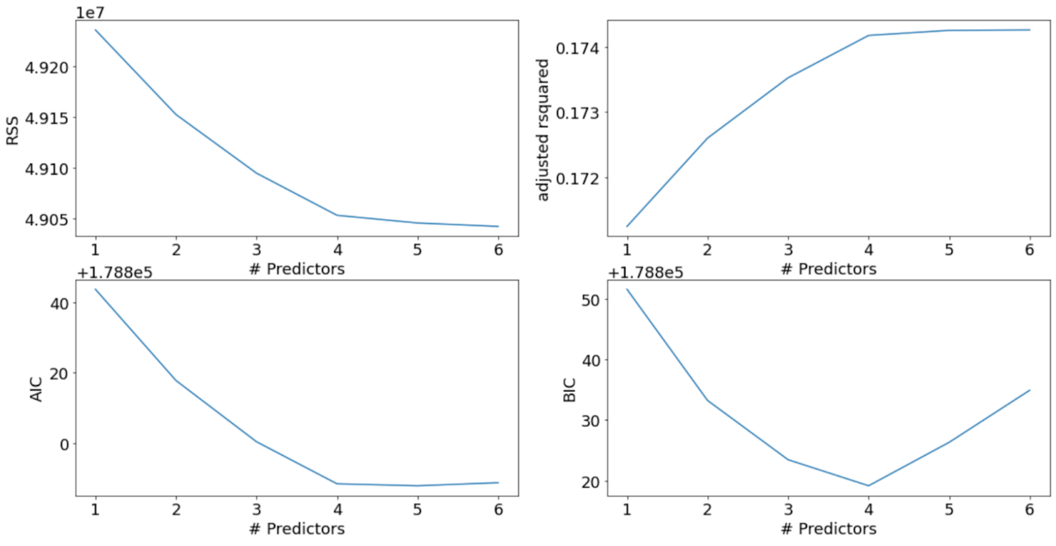
4. Criterion Methods: AIC AND BIC

AIC, otherwise known as Akaike Information Criterion, and BIC, otherwise known as Bayesian Information Criterion, are both common and effective ways of performing model selection, despite not being the same. AIC is used to determine the most appropriate and effective model for fitting the data by comparing different available and possible models. AIC is calculated from two vital components: a) the number of predictor variables used in the construction of the model, and b) the maximum likelihood estimate of the model (which provides information on how capable the chosen model is in reproducing the data). According to AIC analysis, the optimal model is one that is able to account for and explain the most variation using the least number of predictors/features [3.4.1]. BIC, in comparison with AIC, penalises parameters more strongly (penalty terms tend to be larger for BIC). It can be defined as "a criterion for model selection among a finite set of models" [3.4.2].

Similar to AIC, BIC is also partly based on the likelihood function. In the model-fitting process, overfitting may occur as a result of an increase in likelihood via the addition of parameters, however, BIC attempts to rectify this issue through the use of a penalty term (which tends to be larger than the penalty term of the AIC method) [3.4.2].

It is interesting to note that while AIC is particularly useful for making predictions as it is “asymptotically equivalent to cross-validation” [3.4.3], BIC is particularly good for reasoning and explanation, as consistent estimation of the “underlying data generating process” can be undertaken [3.4.3]. Additionally, the dimension of BIC is finite, and tends to be lower than that of AIC [3.4.4]. Interestingly, while BIC is often used in time series and linear regression for model identification selection, it can also be applied to a broader array of uses, such as to “any set of maximum likelihood-based models.” [3.4.2]

AIC and BIC were used in our project to perform best subset selection. Essentially, we constructed four different plots; one of ‘Number of Predictors’ against ‘RSS’, one of ‘Number of Predictors’ against ‘Adjusted R Squared’, one of ‘Number of Predictors’ again AIC, and one of ‘Number of Predictors’ against BIC. These plots essentially demonstrate how the values for these respective y-axis variables differ as the number of predictors included in our model change (anywhere from 1 predictor to 6 predictors).



Now, according to the RSS and Adjusted R Squared plots, the model with six predictors is the most optimal (as RSS is the lowest and Adjusted R Squared is the highest when the number of predictors is equal to 6). However, knowing that the model with the lowest AIC/BIC value should be selected, the AIC and BIC plots show that the model with four predictors is most optimal. Such models tend to favour simpler models, which offer the best compromise between bias and variance.

5. Deep Learning and Neural Networks

Feed Forward Neural Network

Feed Forward Neural Network also known as multi-layered networks of neurons (MLN) is a variant of neural network. It is composed of many neurons arranged in layers and they communicate through sending signals to each other over weighted connections [3.5.1]. It is called feedforward because it travels in the forward direction through the input nodes, then through the hidden layers and then through the output nodes.

There are no feedback connections, meaning that the output is not fed back into itself [3.5.2]. The sum of products of inputs and their weights are calculated which will then be fed to the output [3.5.3]. The layers in the network work by the input layer receiving data from outside of the network, the hidden layers then receive and process data within the network and finally the output layer sends data out of the network [3.5.1].

Feed Forward Neural Network is useful in learning the linear relationship between the predictors and the target variable. The predictors serve as inputs to the network and the target variable serves as the output of the network [3.5.4]. Since the neural network is biologically inspired by the human brains, comparable to the human brains, through processing data to predict future values, it is able to discover complex correlations hidden in the data. It is also useful in detecting patterns in the input data and producing an output that is free of noise [3.5.5]. Feed forward neural network is sometimes favoured over other regression algorithms because it also has the ability to learn non-linear and complex relationships between the inputs and outputs. Moreover, it doesn't have any restrictions on how input variables should be distributed [3.5.6].

6. K-Nearest Neighbours Regression

K-Nearest Neighbours is a non-parametric method because it doesn't require the data to have any specific distributions. It approximates the relationships between the predictors and the target variable by averaging the observations in the same neighbourhood [3.6.1]. The size of the neighbourhood (also known as K) is important in approximating a data point. If the value of K equals to 2, then 2 nearest neighbours to the data point will be averaged to determine the value of the data point. Popular distance metrics such as Euclidean distance, Manhattan distance, Minkowski distance and Hamming distance are methods to calculate the distance between a data point and its neighbours [3.6.2]. Cross validation is a popular method used to find the optimal K that minimises the mean squared error [3.6.1].

One of the biggest advantages of using K-Nearest Neighbours Regression over other regression models is that it is not required to fit any polynomial or interaction terms, and it also does not have strong assumptions of the form of its inputs. Unlike other regression models, K-nearest neighbours regression is a universal function of approximators. They are able to model the inputs linearly and nonlinearly, capturing any interactions and polynomial effects [3.6.3].

4. Results

Models - Modelling Results

7. Linear Regression Model

Our team used Python to construct a multivariate linear regression model. Regression coefficients were estimated using the Ordinary Least Squares (OLS) Method. The Residual Sum of Squares (RSS) is the squared sum difference between the observed and predicted values of the response variable. OLS estimates the unknown regression coefficients by minimising RSS. Our team removed the intercept term because the model without the intercept delivered more promising results than the model with the intercept.

Model 1: The full model with all predictors

$$Withdraw = \beta_1 Shops + \beta_2 ATMs + \beta_3 Downtown + \beta_4 Weekday + \beta_5 Center + \beta_6 High + \epsilon.$$

Measure:	**Result:**
R squared	0.998
Adjusted R squared	0.998
AIC	1.102e+05
BIC	1.103e+05
MSE	6.10186

R-squared (also known as the coefficient of determination) measures how much the variability of the dependent variable can be explained by all the independent variables. Adjusted R-squared is similar to R-squared but it takes into consideration the number of predictors in the model [4.1.1]. As shown above, the independent variables explain total variation of the response variable Withdraw quite well (99.8%). However, we would like to assess how well the model can predict new values. R-squared and adjusted R-squared does not account for this. Our team resorted to measures such as MSE, AIC and BIC to assess model predictability. The Mean Squared Error measures how close the regression line is to the observed data points. The sum of the distances from the regression line to data points are taken and squared to negate the effect of negative distances. The lower the MSE value, the better [4.1.2].

Information criterion methods such as the Akaike Information Criterion (AIC) and the Bayesian Information Criterion (BIC) were used to choose the best model which could explain the greatest amount of variation with the least number of predictors. AIC and BIC values for all chosen models were compared and the model with the lowest AIC/BIC value was chosen as the best model. Our team prioritised AIC and BIC over the R-squared measure because they are able to assess model fitness based on how well a model could fit a dataset and how complex it is [4.1.3]. We preferred a simple model that could explain total variation in the response variable.

Model 2: The model with all independent variables except High

$$Withdraw = \beta_1 Shops + \beta_2 ATMs + \beta_3 Downtown + \beta_4 Weekday + \beta_5 High + \epsilon.$$

The table below summarises the results obtained when each predictor was omitted from the full regression model:

	Shops	**ATMs**	**Downtown**	**Weekday**	**Center**	**High**
AIC	1.520e+05	1.149e+05	1.377e+05	1.138e+05	1.204e+05	1.111e+05
BIC	1.520e+05	1.149e+05	1.377e+05	1.138e+05	1.204e+05	1.112e+05
MSE	9.28896	9.28030	6.49317	8.59576	10.59522	6.30466

The model without the independent variable 'High' produced some promising results. It scored the lowest in terms of AIC, BIC and MSE. However, the model scores a slightly higher MSE than the full model (6.10186). Our group realised that omitting a predictor may not be the best option and thus investigated whether including an interaction term could improve our results.

Model 3: The model with interaction term Weekday*Center

$$Withdraw = \beta_1 Shops + \beta_2 ATMs + \beta_3 Downtown + \beta_4 Weekday + \beta_5 Center + \beta_6 High + \beta_7 (Weekday * Center) + \epsilon.$$

Introducing the interaction term (Weekday * Center) significantly improved our results.

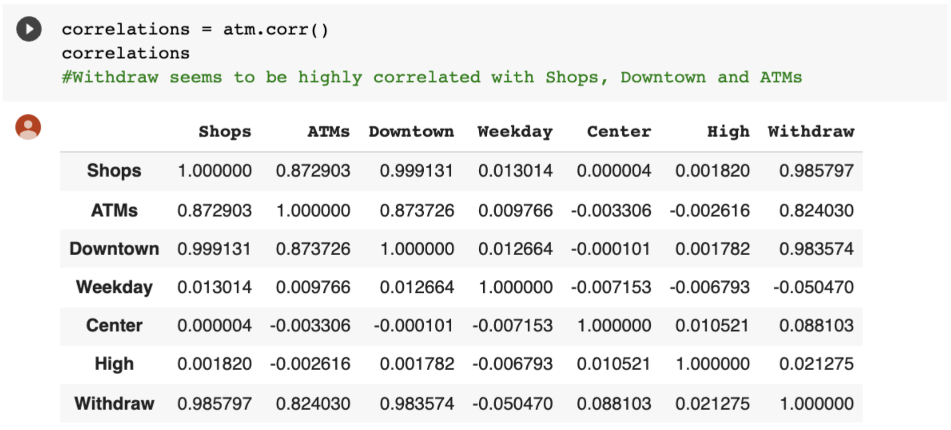
	Full Model	**Model without predictor 'High'**	**Model with interaction term 'Weekday*Center'**
AIC	1.102e+05	1.111e+05	9.479e+04
BIC	1.102e+05	1.112e+05	9.485e+04
MSE	6.10186	6.30466	2.32811

Our team experimented with all possible interaction terms. We found that the model with Weekday*Center scored the least AIC, BIC and MSE amongst all models with a single, unique interaction term (refer to Appendix). Furthermore, the AIC, BIC and MSE of the model are significantly lower than the values for the full model (as shown in the table above).

8. Lasso, Ridge and Elastic Net Regression model

LASSO

In the initial exploratory data analysis stage of our project, we constructed a correlation matrix for the provided data, which showed suspiciously high correlation values for Withdraw and Shops (0.986), Withdraw and Downtown (0.984), and Withdraw and ATMs (0.824). It became evident to us that using LASSO regression may be helpful in counteracting these potential multicollinearity issues we have faced.

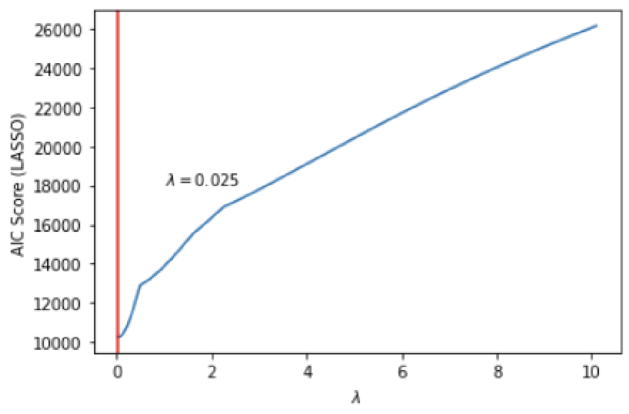


```
correlations = atm.corr()
correlations
#Withdraw seems to be highly correlated with Shops, Downtown and ATMs
```

	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

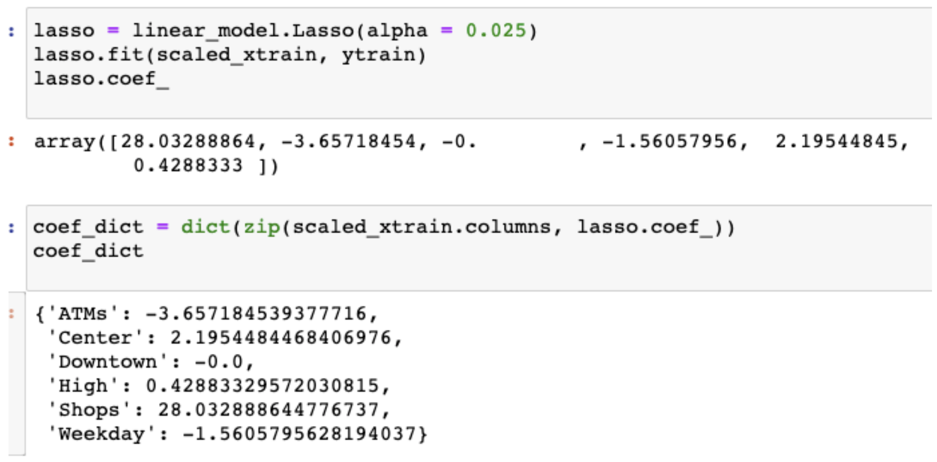
Note: atm_training.csv was split into test and training sets.

In terms of our LASSO-specific code, we essentially iterate through all lambda values from 0.015 to 10.1 by steps of 0.01, in an attempt to find the most optimal lambda (ie. the lambda value that gives us the lowest AIC/BIC values). We create a model using LASSO regression, that according to LASSO is considered the best, and fit this model on the training set. We then calculate Mean Squared Error (MSE) values as well as a value for the number of parameters, and use these inputs within our AIC and BIC functions. We repeat/iterate through this process in order to find the lambda that presents the lowest possible AIC and BIC values - essentially, we use LASSO regression here to perform variable selection.



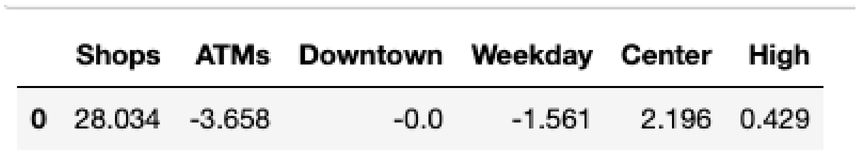
It is evident from the generated plot that the most optimal lambda is lambda = 0.025; at this value, AIC/BIC is minimised.

Using these AIC and BIC scores, we then go on to compute a final LASSO estimate which is chosen by these AIC/BIC values. We find that predictor ‘Downtown’ is removed, evidenced by its value of 0 below.



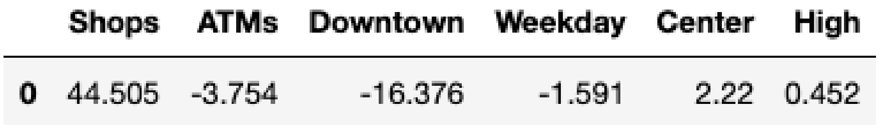
```
: lasso = linear_model.Lasso(alpha = 0.025)
  lasso.fit(scaled_xtrain, ytrain)
  lasso.coef_

: array([28.03288864, -3.65718454, -0.        , -1.56057956,  2.19544845,
          0.4288333 ])

: coef_dict = dict(zip(scaled_xtrain.columns, lasso.coef_))
  coef_dict

: {'ATMs': -3.657184539377716,
   'Center': 2.1954484468406976,
   'Downtown': -0.0,
   'High': 0.42883329572030815,
   'Shops': 28.032888644776737,
   'Weekday': -1.5605795628194037}
```

Following this, we decided to verify our results using sklearn.linear_model’s LassoCV, which is used as Lasso Regression Cross Validation implementation [4.8.1]. Taking ‘5’ as the input for ‘cv’ within the LassoCV function (representing the considered number of folds when cross validation is applied) followed by utilising the lasso.fit function, we are left with the same conclusion: ‘Downtown’ is removed.



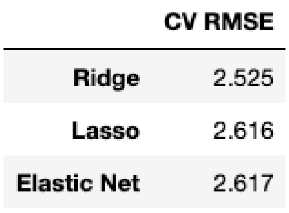
	Shops	ATMs	Downtown	Weekday	Center	High
0	28.034	-3.658	-0.0	-1.561	2.196	0.429

Interestingly, when we attempted to further verify this result using Ridge regression and RidgeCV with the same ‘cv’ value of 5, we found that ‘Downtown’ was *not* removed, evidenced by its non-zero value shown below. We believe this may be due to the fact that LASSO regression is more strict than Ridge regression in terms of penalisation.



	Shops	ATMs	Downtown	Weekday	Center	High
0	44.505	-3.754	-16.376	-1.591	2.22	0.452

Finally, looking at values for CV RMSE (Coefficient of Variation of Root-Mean Squared Error) for Ridge, Lasso, and Elastic Net, it is evident that Ridge appears to perform the best among the three methods (ie. the method that led to the ‘Downtown’ predictor *not* being removed) as it has the lowest CV RMSE value (2.525) of the three methods. This is a fairly interesting result (given LASSO tends to shrink variables, and in this case led to the removal of ‘Downtown’) - this predictor may need to undergo further investigation.



	CV RMSE
Ridge	2.525
Lasso	2.616
Elastic Net	2.617

Next, we compare our results further using Ridge and Elastic Net regularization to see if Lasso may have been too aggressive in dropping possible predictors.

RIDGE

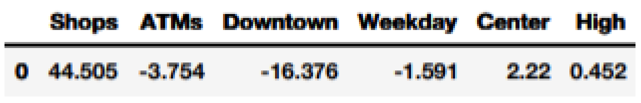
Given coefficients are part of the penalty function, this means coefficients that get very large are penalised in our model. When we increase the value of lambda, the bias is unchanged, but the variance of the model drops. Larger lambda means the penalty for complex coefficients is larger. We can now get down to building our ridge regression model.

Lambda is a design parameter which we will control. This is referred to as the hyper-parameter of the ridge regression model. To find what value of lambda will give us the best model, we use cross-validation by trying 500 evenly distributed lambda values selected between -10 and 20. We use cross-validation with 5 folds of our data to find the right lambda values for our dataset, for our model.

Cross-validation requires us to build multiple ridge regression models with different lambda values to find the optimal lambda. It turns out there is a function in the sklearn.linear_model package where RidgeCV will perform cross-validation on our ridge regression model to find an optimal lambda.

This mixing parameter determines what kind of regression model we’re building. We use the value of alpha to build a ridge regression model, a lasso regression model or an ElasticNet model.

Behind the scenes, this mixing parameter is used to tweak our penalty function. alpha = 0 signifies that we are performing ridge regression. From alphas = alphas, this is the sequence of alphas/lambdas that we want to test to find the optimal lambda using cross-validation. Here our optimal lambda is a little more than 0, specifically 0.01 and we use this to build our model. Thus knowing this, fitting the ridge regression model on our data is extremely straightforward.



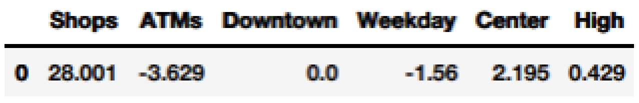
	Shops	ATMs	Downtown	Weekday	Center	High
0	44.505	-3.754	-16.376	-1.591	2.22	0.452

Now with this in mind, let's take a look at the coefficients of the ridge regression that we fit on our data using cross-validation. Observe that none of the coefficients are close to 0, and also that none of them have been set to 0. Ridge regression does not perform model selection which is a drawback as it doesn't select the important variables in our regression analysis. It includes all the predictor variables that we've specified in the final model but doesn't check to see whether our variables are significant or not.

ELASTIC NET

Next we look into building our Elastic Net regression model, where we'll perform some hyper-parameter tuning to find the best values of both alpha as well as lambda. We will tune these parameters using cross-validation, using the enet_cv function. Observed in the above code, we have incorporated a ElasticNetCV function that will allow us to tune our training process. We perform cross-validation with 5 folds of our data.

We train our Elastic Net model to predict the cash demand Withdraw using all other predictors. We use the training data to automatically train our Elastic Net model using cross-validation to find the best alpha and lambda for our model. The best alpha value for our dataset is 0.025 which controls the mix of the penalties of L1 and L2 regularization. An alpha of 0.025, which is what we've used makes our model very close to the ridge regression model with alpha nearly equivalent to 0. Now that we have our best model, let's use it for evaluation.



	Shops	ATMs	Downtown	Weekday	Center	High
0	28.001	-3.629	0.0	-1.56	2.195	0.429

Let's see what the coefficients of the final model are. By specifying the best lambda, this gives us the coefficients of our final Elastic Net model. Some of the coefficients are close to 0, but have not been set to 0. Notably, 'Downtown' variable was removed here.

Finally, by looking at the evaluation RMSE statistics of our elastic net model (2.617) it is not much better than our LASSO (2.616) or Ridge (2.525). It is clear that for our particular dataset, using regularized regression models, Ridge performs the best out of the 3 regularised methods. This likely also means that the improvement that we saw in the lasso regression model, was quite possibly purely due to chance.

9. Deep Learning model

It is widely known that feed forward neural networks often overfit the data, even after standardizing the variables. This is due to the flexibility neural network models provide in capturing underlying data structure. In order to reduce overfitting, we need to carefully choose the number of units and hidden layers to use in our feed forward neural network model. The selection of numbers of batch size and epochs are also important because the model should stop updating once validation loss is not decreasing. In our feed forward neural network model, we will be using equal size layers (known as rectangular-shaped neural nets). The training set provided had been split so that we can evaluate our model on the testing set. Standardization of predictors had been applied to both the training and testing sets.

Our feedforward neural network is fitted with all the predictors provided in the dataset. To reduce training time, we selected batch sizes of 10 and 10 epochs to find the best number of hidden units using 1 hidden layer. Based on the following results, 18 hidden units perform the best in terms of the mean squared errors.

Results from finding the best number of units:

Number of hidden units	**Training MSE**	**Testing MSE**
2	16.310	15.721
3	16.178	15.593
4	8.654	8.086
5	5.993	5.934
6	0.768	0.790
7	0.510	0.526
8	0.398	0.407
9	0.522	0.538
10	0.479	0.487
11	0.442	0.448
12	0.404	0.415
13	0.345	0.357
14	0.385	0.389
15	0.311	0.317
16	0.353	0.358
17	0.323	0.321
18	0.293	0.300
19	0.315	0.326
20	0.331	0.329

Now, using 18 units, we would like to find the optimal number of hidden layers. Based on the results below, 1 hidden layer performs the best. Mean squared error increases as the number of hidden layers increases.

Results from finding the best number of hidden layers with 10 units in each layer:

Hidden Layers	Training MSE	Testing MSE
1	0.293	0.300
2	0.328	0.325
3	0.334	0.328
4	0.389	0.380

We can see that from the summary of the network structure that there are 145 trainable parameters in our defined network:

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 18)                126
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 19
=================================================================
Total params: 145
Trainable params: 145
Non-trainable params: 0
_________________________________________________________________
```

Finally, we would now like to find the optimal batch size and epochs that will produce the lowest mean squared error. Keras' early stopping call back function was used to help monitor the validation loss. Training procedure will then be stopped when validation loss is not decreased after a certain iterations (patience = 5), and hence producing the optimal epochs as shown in the table below with respect to the batch size. Notice that in each row in the table, the model is neither overfitting nor underfitting. We found that a batch size of 12 with 26 epochs performed the best as they gave the lowest MSE score.

Results from finding the optimal batch size and epochs using 1 hidden layer

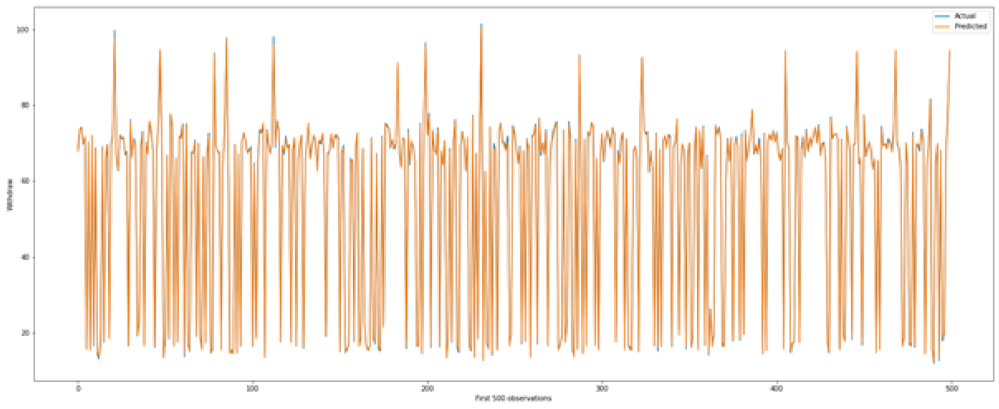
and 18 units in the hidden layer:

Batch size	Optimal epochs	Training MSE	Testing MSE
5	14	0.345	0.351
6	14	0.322	0.325
7	14	0.338	0.346
8	26	0.315	0.324
9	26	0.300	0.308
10	26	0.283	0.285

11	26	0.281	0.286
12	26	0.278	0.280
13	26	0.280	0.281
14	34	0.301	0.300
15	34	0.298	0.299

By using the optimal hyperparameters found (1 hidden layer, 18 units, batch size of 12 and 26 epochs), our feed forward neural network model can then learn the optimal weights of the network edges. From the plot below, we can see that the model fits the data very well with a testing MSE of 0.280.

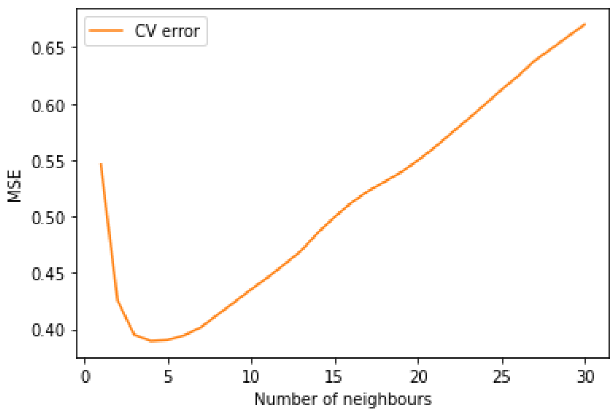
Plot of predicted values vs actual values for the first 500 observations:



10. K-Nearest Neighbours Regression model

Finding the optimal nearest neighbours, K, is not only important in improving the accuracy and approximations of a data point but it also affects whether the model generalizes well. A low value of K could lead to overfitting and a high value of K could lead to underfitting. Python's implementation of cross validation score is used to help us with selecting the optimal K. The advantage of using this is that a validation set is not required. The training set provided had been split so that we can evaluate on the testing set. Standardization of predictors had been applied to both the training and testing sets. Using a cross-validation fold of 10 and setting the scoring metric as negative mean squared error, we were able to find that the optimal K for our model is 4. Using this chosen K, we were able to achieve a mean squared error of 0.373 in our own created test set. However, this model is still overfitting because it has a training error of 0.266.

Cross-validation results to find the optimal K:



Conclusion

The purpose of our analysis was to develop a predictive model for cash demand based on the given ATM dataset and find out which features of the data contributed the most significantly to optimise a bank's cash management. We started off by understanding the various model use cases and kept this in mind when building and training our models to find the best fit.

In conclusion, it is clear that our feedforward Neural Networks model provided the highest accuracy, with the lowest MSE of 0.280. From our analysis, we have identified that all predictor variables contribute significantly to our data and have successfully identified a predictive model based on our own test data split.

Appendix

	Shops, ATMs	**Shops, Downtown**	**Shops, Weekday**	**Shops, Center**	**Shops, High**	**ATMs, Downtown**
AIC	1.065e+05	1.035e+05	1.077e+05	1.094e+05	1.100e+05	1.077e+05
BIC	1.066e+05	1.036e+05	1.078e+05	1.095e+05	1.101e+05	1.077e+05
MSE	6.10290	6.04375	5.96502	5.65020	6.10245	6.10293

	ATMs, Weekday	**ATMs, Center**	**ATMs, High**	**Downtown, Weekday**	**Downtown, Center**	**Downtown, High**
AIC	1.075e+05	1.098e+05	1.099e+05	1.081e+05	1.093e+05	1.101e+05
BIC	1.076e+05	1.098e+05	1.100e+05	1.081e+05	1.094e+05	1.101e+05
MSE	6.00930	5.70066	6.10306	5.96291	5.63863	6.10256

	Weekday, Center	Center, High
AIC	9.479e+04	1.102e+05
BIC	9.485e+04	1.103e+05
MSE	2.32811	6.10686

References

[2.1.1] Patil, P., 2021. *What is Exploratory Data Analysis?*. [online] Medium. Available at: <https://towardsdatascience.com/exploratory-data-analysis-8fc1cb20fd15> [Accessed 2 November 2021].

[2.1.2] Koehrsen, W., 2021. *Visualizing Data with Pairs Plots in Python*. [online] Medium. Available at: <https://towardsdatascience.com/visualizing-data-with-pair-plots-in-python-f228cf529166> [Accessed 5 November 2021].

[3.1.1] Schneider, A., Hommel, G. and Blettner, M., 2010. Linear Regression Analysis. *Deutsches Aerzteblatt Online*, DOI: 10.3238/arztebl.2010.0776

[3.1.2] Ibm.com. 2021. *About Linear Regression*. [online] Available at: <https://www.ibm.com/au-en/analytics/learn/linear-regression> [Accessed 9 November 2021].

[3.2.1] Ved, 2016. Introduction to lasso regression. *d4datascience.com*. Available at: https://d4datascience.in/2016/06/29/introduction-to-lasso-regression/ [Accessed November 3, 2021].

[3.2.2] Tyagi, N., 2021. L2 vs L1 regularization in machine learning: Ridge and Lasso regularization. *L2 vs L1 Regularization in Machine Learning | Ridge and Lasso Regularization*. Available at: https://www.analyticssteps.com/blogs/l2-and-l1-regularization-machine-learning [Accessed November 4, 2021].

[3.2.3] Glen, S., 2015. Lasso regression: Simple definition. *Statistics How To*. Available at: https://www.statisticshowto.com/lasso-regression/ [Accessed November 5, 2021].

[3.3.1] NCSS Statistical Software, 2021. *Ridge Regression* [online] Available at:

<https://ncss-wpengine.netdna-ssl.com/wp-content/themes/ncss/pdf/Procedures/NCSS/Ridge_Regression.pdf> [Accessed: 4 November, 2021].

[3.3.2] Nagpal A., 2017. *L1 and L2 Regularization Methods* [online] Available at:

<https://towardsdatascience.com/l1-and-l2-regularization-methods-ce25e7fc831c> [Accessed 6 November, 2021].

[3.3.3] Oleszak M., 2019. Regularization: Ridge, Lasso and Elastic Net *Datacamp.* [online] Available at: <https://www.datacamp.com/community/tutorials/tutorial-ridge-lasso-elastic-net>[Accessed 6 November, 2021].

[3.3.4] Wessel N. van Wieringen., 2021. *Lecture notes on Ridge Regression* [online] Available at: <https://arxiv.org/pdf/1509.09169.pdf> [Accessed 5 November, 2021].

[3.4.1] Bevans, R., 2021. An introduction to the akaike information criterion. *Scribbr*. Available at: https://www.scribbr.com/statistics/akaike-information-criterion/ [Accessed November 5, 2021].

[3.4.2] Datalab, A., 2019. What is Bayesian Information Criterion (BIC)? *Medium*. Available at: https://medium.com/@analyttica/what-is-bayesian-information-criterion-bic-b3396a894be6 [Accessed November 5, 2021].

[3.4.3] S, P., 2010. Difference between AIC and bic. *Difference Between.net*. Available at: http://www.differencebetween.net/miscellaneous/difference-between-aic-and-bic/ [Accessed November 6, 2021].

[3.4.4] Kandekar, S., Difference Between AIC and BIC (With Table). *Ask Any Difference*. Available at: https://askanydifference.com/difference-between-aic-and-bic/ [Accessed November 7, 2021].

[3.5.1] Tran, M., 2021. *ECON3203 Econometric Theory and Methods Neural network 1*.

[3.5.2] Kumar, N., 2019. *Deep Learning: Feedforward Neural Networks Explained*. [online] Medium. Available at:<https://medium.com/hackernoon/deep-learning-feedforward-neural-networks-explained-c34ae3f084f1> [Accessed 6 November 2021].

[3.5.3] Imran, M., 2020. *Advantages of Neural Networks - Benefits of AI and Deep Learning*. [online] Folio3AI Blog. Available at: <https://www.folio3.ai/blog/advantages-of-neural-networks/> [Accessed 6 November 2021].

[3.5.4] Science Direct, 2009. *Feedforward Neural Networks*. [online] Available at: <https://www.sciencedirect.com/topics/chemical-engineering/feedforward-neural-networks> [Accessed 6 November 2021].

[3.5.5] Bari, A., Chaouchi, M. and Jung, T., 2021. *How Predictive Analysis Neural Networks Work*. [online] dummies. Available at: <https://www.dummies.com/programming/big-data/data-science/how-predictive-analysis-neural-networks-work/> [Accessed 6 November 2021].

[3.5.6] Mahanta, J., 2017. *Introduction to Neural Networks, Advantages and Applications*. [online] Medium. Available at: <https://towardsdatascience.com/introduction-to-neural-networks-advantages-and-applications-96851bd1a207> [Accessed 6 November 2021].

[3.6.1] Teixeira-Pinto, A., 2021. *2 K-nearest Neighbours Regression | Machine Learning for Biostatistics*. [online] Bookdown.org. Available at: <https://bookdown.org/tpinto_home/Regression-and-Classification/k-nearest-neighbours-regression.html> [Accessed 7 November 2021].

[3.6.2] Joby, A., 2021. *What Is K-Nearest Neighbor? An ML Algorithm to Classify Data*. [online] Learn.g2.com. Available at: <https://learn.g2.com/k-nearest-neighbor> [Accessed 7 November 2021].

[3.6.3] Stack Exchange Cross Validated. 2018. *Are interactions only useful in the context of regression?*. [online] Available at: <https://stats.stackexchange.com/questions/181183/are-interactions-only-useful-in-the-context-of-regression> [Accessed 7 November 2021].

[4.1.1] Potters, C., 2021. *R-Squared vs. Adjusted R-Squared: What's the Difference?*. [online] Investopedia. Available at: <https://www.investopedia.com/ask/answers/012615/whats-difference-between-rsquared-and-adjusted-rsquared.asp> [Accessed 9 November 2021].

[4.1.2] Glen, S., 2021. *Mean Squared Error: Definition and Example*. [online] Statistics How To. Available at: <https://www.statisticshowto.com/probability-and-statistics/statistics-definitions/ mean-squared-error/> [Accessed 5 November 2021].

[4.1.3] Brownlee, J., 2021. *Probabilistic Model Selection with AIC, BIC, and MDL*. [online] Machine Learning Mastery. Available at: <https://machinelearningmastery.com/probabilistic-model- selection-measures/> [Accessed 5 November 2021].

[4.8.1] Kumar , A., 2020. Lasso regression explained with python example . *Data Analytics, Data, Data Science, Machine Learning, AI*. Available at: https://vitalflux.com/lasso-ridge-regression-explained-with-python-example/ [Accessed November 6, 2021].