Credit Card Prediction Of A Customer
By Alexandru Iordache, Ioan Cosma, Sameer Ali

Introduction

The dataset that we decided to investigate and use for our report is based on data collected by a Bank on its customers, called 'Bank Loan Modelling' (Jacob, 2018) [1]. We found this to be an interesting dataset to study and work with as it provides a vast range of data based on a real world application for the majority of the population, resulting in having a complex domain that can be manipulated using different techniques and models to produce a variety of outputs. The data source that we used to find the dataset is 'Kaggle' (Kaggle. 2010) [2], which is a website that provides various published datasets that can be explored and manipulated for personal use and educational purposes.

The dataset consists of 14 columns and 5000 rows of data, with all of it being numerical as well as discrete. Each column has a unique feature as follows: 'ID', 'Age', 'Experience', 'Income', 'ZIP Code', 'Family', 'CCAvg', 'Education', 'Mortgage', 'Personal Loan', 'Securities Account', 'CD Account', 'Online', 'CreditCard'. This would imply that the dataset is suitable for modelling to be applied in order to train a model with a good amount of samples from the dataset to help solve a problem.

Some of the analytical questions asked in order to find and study a problem domain using this dataset are the following:
-What problem do we seek to address with this dataset?
-What methods are we going to use in order to solve our problem?
-What do we expect our output to be?
-What are some plausible explanations for the results given?

The publisher of the dataset recommends to "Use a classification model to predict the likelihood of a liability customer buying personal loans" (Jacob, 2018) [1] as the problem domain however, we found that this had already been explored and published by several other members of the Kaggle community (Kamil, 2018) [3]. Therefore, we decided to go about finding a problem domain by using the pre-processing methodology of preparing the data in order to analyze it further.

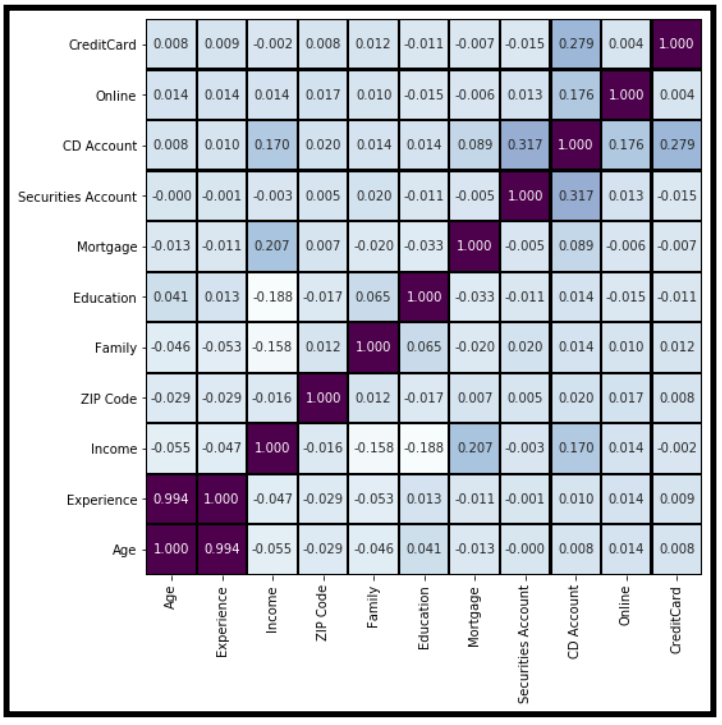
Data Preparation

To begin we imported our dataset into a 'Jupyter' notebook through the 'pandas' library in order to look at the type and range of values that each feature contained to help get an idea of what could be investigated further. By using the '.value_counts()' function provided by pandas we were able to see that there was a wide range of samples within the first 8 features, and that the last 5 features contained two separate values of either a '0' or '1'. This immediately helped us in being able to determine what our problem domain would be as it would be a sensible choice to use one of these 5 features as the 'target' feature for predictions during the modelling process as their values binary, which can be classified.

In order to then decide on which of these we would base on the problem domain, we looked to create a 'Heat map' (Galarnyk, 2016) [4] as this provides a visualization technique to view the correlations between features. Before doing so, we reviewed the purpose of these features and decided that we would remove the 'Personal Loan' feature as this had already been explored, 'ID' as this feature is used only by the Bank

for monitoring purposes, ‘CCAvg’ as this was a dependent feature of the ‘CreditCard’ feature. This left us with 11 features and 4 target features to decide from. We also made sure to check the data for missing values as this would affect the outputs from visualizations and especially models when being trained for prediction. Fortunately our dataset did not have any missing data, which allowed us to work with the most accurate representation of the data provided.

From the visualization below, it can be seen that there is a surprising lack of strong positive correlations between most features. This is also the case regarding the potential target features shown in the top 4 rows, although they do have slightly higher correlations among each other. However, we decided not to use any of these potential target features together for modelling as it would create another question of deciding which one to use as an input and the other as a target. As we felt there was no clear target to choose from based on this visualization, we came to the conclusion that the CreditCard feature would an interesting target to base our problem domain on, due to this being a primary asset as well as an income stream for a Bank and so, being able to make predictions on whether a customer will use a credit card or not would be a useful indicator for its profitability:

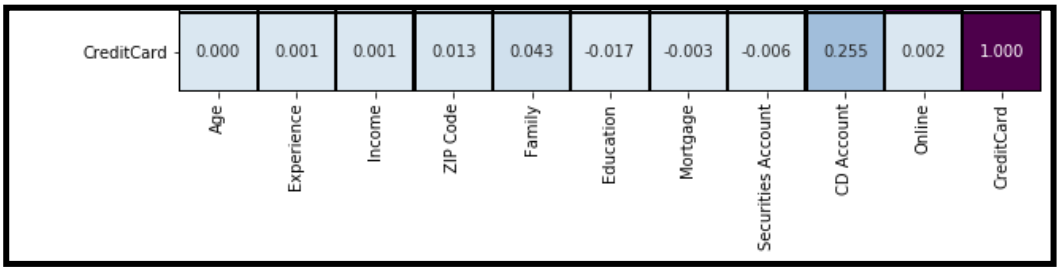


We then had to decide what we would be using as our ‘input’ features for modelling. After specifically reviewing the CreditCard row from the Heat map we selected the Family feature as an input as it had the highest positive correlation, and also consisted of only 4 different values. We also decided to use the Income feature as our second input as the most common purpose of a customer using a credit card, is for purchasing goods that they cannot afford to pay for at the time due to financial issues. It also has a negative correlation with both of the other selected features, which we presumed would lead to interesting results.

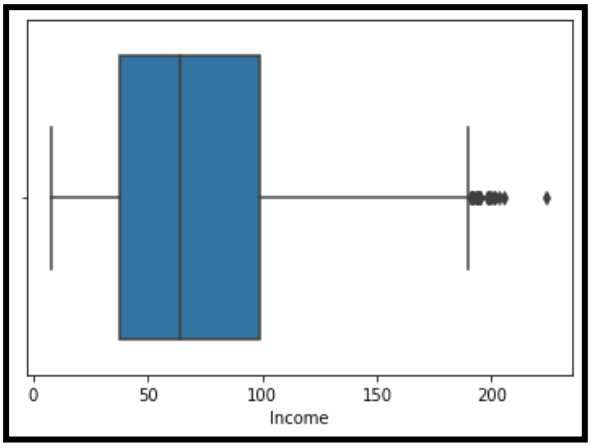
After coming to the decision to make the CreditCard feature the target, it was important to check what the split of its values are before moving further since it is a binary feature. The ‘.value_counts()’ function was used again on this feature, which showed the following split: ‘0 = 3530, 1 = 1470’

These results were a concern as there is a great disparity between the two, which would skew the accuracy of prediction for a model due to the bias that would be provided in the training samples towards the majority class. This also strengthens the reasoning behind the problem domain as a Bank would ideally want more of their customers using a credit card than less. In order to resolve this issue, we looked at the techniques of 'under/oversampling' to allow the values to equalize. We felt it made the most logical sense to use the 'Random Undersampling' technique (Alencar, 2017) [5] as there would still be close to 3000 total samples of data available after reducing the values of the majority class to meet the minority class, as well as preventing the issue that may occur with oversampling where the data that would be replicated from the minority class could misrepresent the class, as it would be a significant increase of more than double and then affect prediction accuracy.

After balancing the data of the target set we created an updated Heat map visualization to see if the correlation values would change. The correlation between the inputs and the target increased, with the Income value changing from negative to positive:



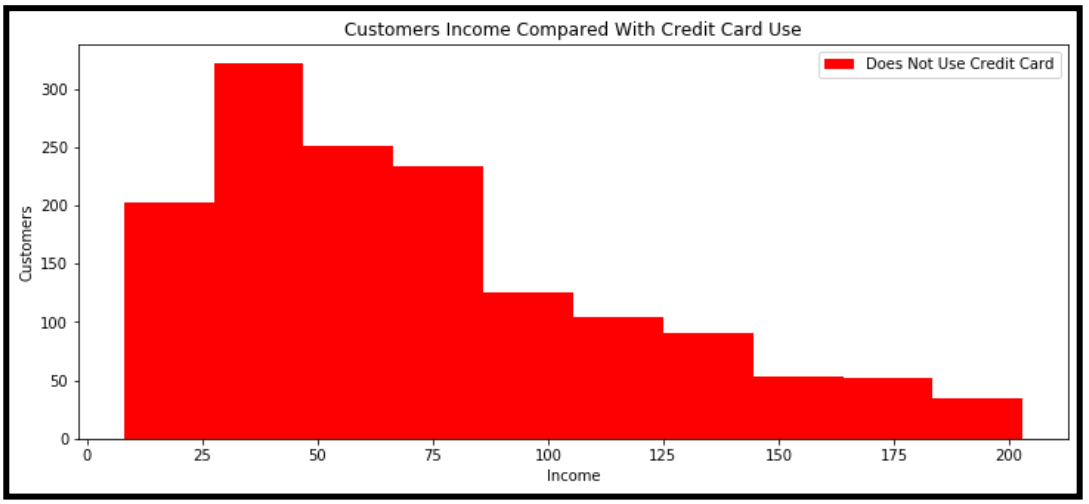
Next we created further visualizations to investigate the values for the inputs. After counting the values of the Income feature, we found that there was a wide range of values and so, we had to consider if there were any outlier values that would affect model prediction if unaccounted for. This was firstly checked by creating a 'Box plot' to show any values that were outside of the quartile range. The results showed that there were several values outside of the range around the area of 200K, which was a concern:

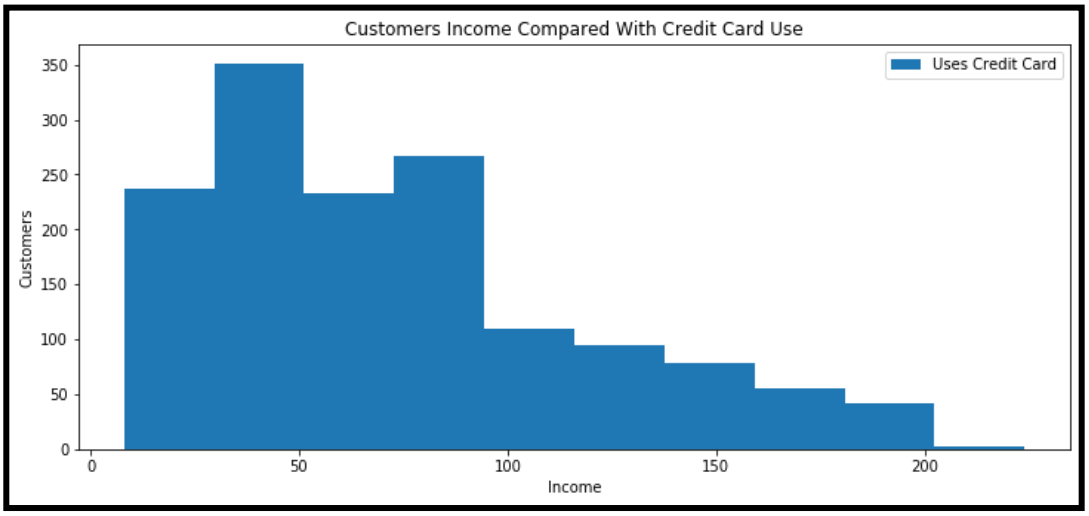


The diagram however, was not as easy to interpret the exact values that were shown to be outliers, so we used Tukey's rule (Chen, 2016) [5] to cross check this and show the values:

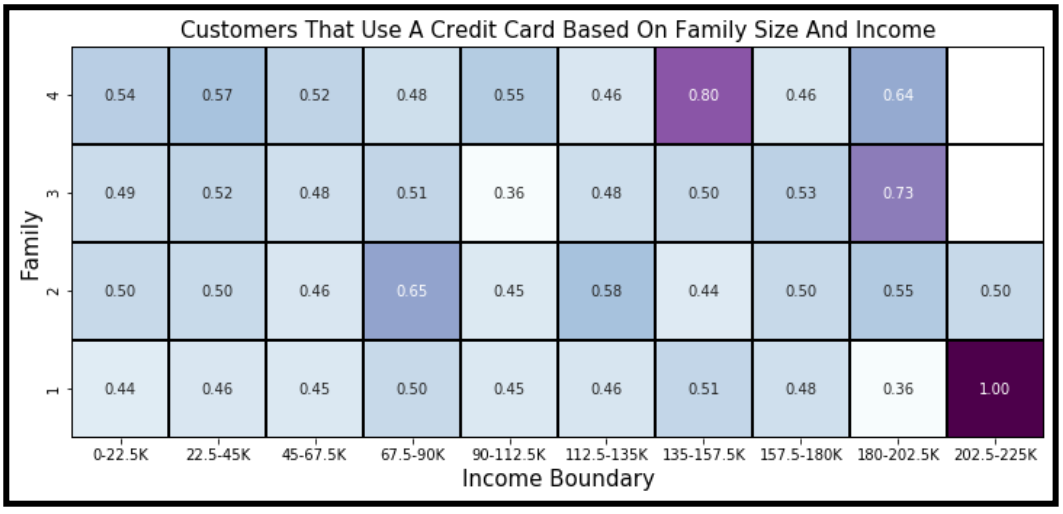
'print(np.sort(tukey_values))
[191 192 193 194 195 198 199 200 201 203 205 224]'

Before deciding to simply remove these outliers from the dataset we wanted to see how they had an affect on the variety of values for this feature. To do this, a 'Histogram' on the amount of customers based on their Income was created (Chen, 2016) [6]. The histograms below follow an almost identical pattern, which implies that the outlier variables don't skew the results so we decided to keep them in for sampling:





Following this we looked to see if the Family feature once combined with the Income feature would follow this consistency or point to bias on a specific family size. Another Heat map was created based on these three features however, as there was such a large amount of values for the Income, the values had to be grouped into ‘Bins’ (Galarnyk, 2016) [4] so that it could properly display on a Heat map. The Heat map shows a consistency with the percentage of customers using a credit card between the different criteria:



A breakdown showing the sum of the values for the Family and CreditCard features for each Income Bin. Shows that there are more customers earning up to 100K than those earning more than 100K, which also leads to significantly more families, and more credit cards being used by them:

Income	Family	CreditCard
(0.0, 22.5]	776	153
(22.5, 45.0]	1793	378
(45.0, 67.5]	1259	231
(67.5, 90.0]	1391	292
(90.0, 112.5]	533	111
(112.5, 135.0]	503	128
(135.0, 157.5]	243	66
(157.5, 180.0]	293	67
(180.0, 202.5]	167	42
(202.5, 225.0]	5	2

Modelling

Now that we had completed the data preparation, we were satisfied to be able to create a model that could be trained to predict whether a customer would want to use a Credit Card by checking the values of their Family and Income. This meant the models would be trained through supervised learning as they would be using well labelled data. This would also produce a binary result as the CreditCard feature as a target class and so, logically we experimented with a variety of classification models to work on the problem domain.

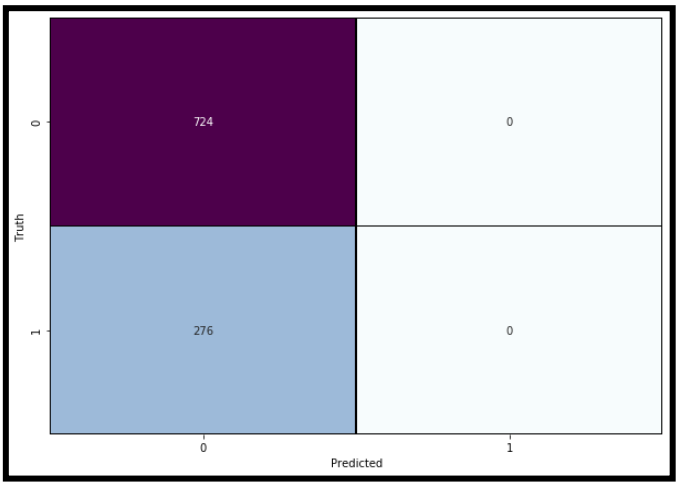
Before creating a model we established a training and testing split of 80/20 on the sample values in order for the model to be trained as best as possible, especially since there is a variety of possible outcomes without an obvious trend or correlation between the inputs and target. We then looked to implement the following classification models:

Decision Tree, Random Forest, Logistic Regression, Support Vector Machine, Neural Network

We used accuracy, precision, recall and f1-score as our evaluation tools although, there was a greater emphasis to have the highest recall possible when predicting a class of 1. This is due to the nature of the problem domain, where the model should be able to predict ‘True Positives’ as best as possible so that the Bank can look to target customers that fall in this category through advertising and incentives without having to worry about as much as this cannot lead to severe repercussions if a ‘False Positive’ is predicted.

Initially the models were trained without the use of undersampling to see what they would output. The outputs between each model were very consistent showing an average total accuracy on both the training and test set of 70%, which would imply that the models were doing fairly well at making a prediction. However, after using the ‘Confusion Matrix’ technique on each model it was clear to see that the models were strong at predicting ‘True Negatives’ but very poor at predicting ‘True Positives’ and in some cases making no predictions at all for the latter.

Logistic Regression Confusion Matrix:



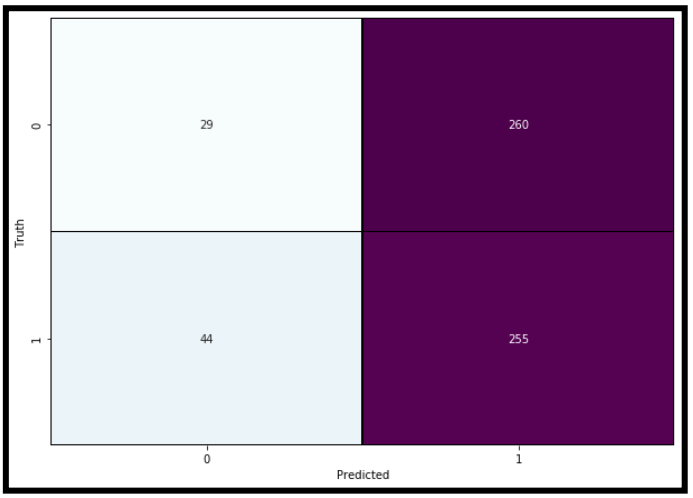
A further breakdown using the ‘Classification Report’ technique showed similar results among each model, with the average precision being 72% and the recall being 95% for class 0. The average precision of class 1 was 10% and the recall was 2%. This shows that although the models have a generally high accuracy, they perform extremely poorly in regards to solving the problem domain. This is likely due to the fact that the model is working with a majority of samples towards class 0, causing it to be weak at predicting the correct output for class 1.

SVM Classification Report:

	‘precision	recall	f1-score	support
0	0.70	0.99	0.82	697
1	0.27	0.01	0.02	303’

We then conducted the training again using the undersampled version of the dataset. The output of each model maintained the consistency between each other but showed a significant decrease in the average training accuracy of 55% as well as the average testing accuracy with 50%. The confusion matrix criteria outputs for each model provided values that were fairly close to each other, which would imply that they would be unreliable for predicting either class value.

Decision Tree Classifier Confusion Matrix:

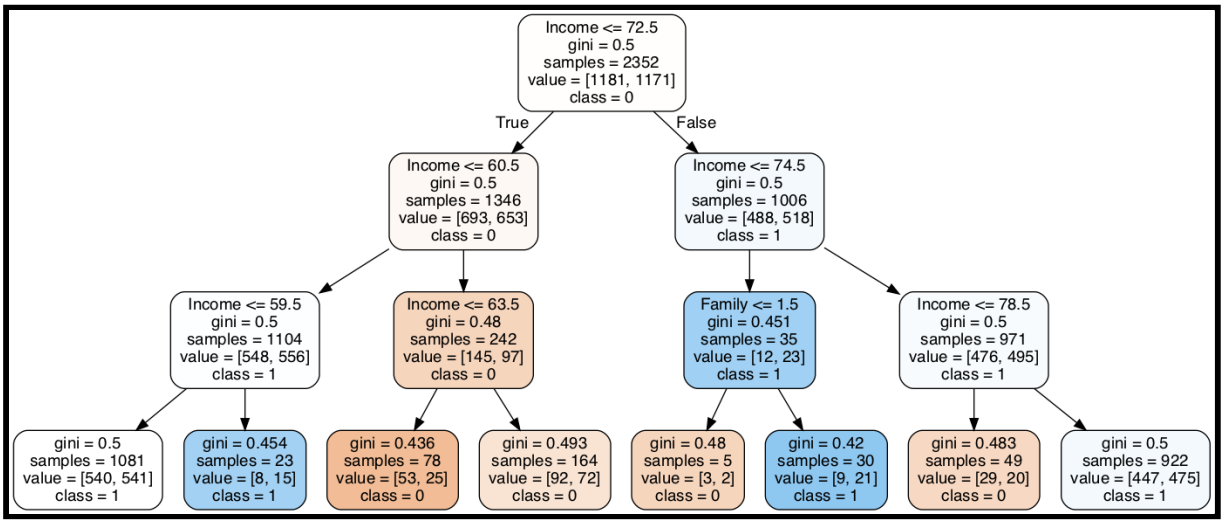


The classification report showed an average precision of 50% and recall was 55% on class 0. The average precision of class 1 was 45% and recall was 50%. Although this shows weak values in terms of accuracy, the recall for class 1 greatly improved from this change to the data sampling, which is beneficial for solving the problem domain.

Logistic Regression Classification Report:

	‘precision	recall	f1-score	support
0	0.48	0.28	0.35	307
1	0.46	0.67	0.55	281’

From the outputs of these models we decided that the Decision Tree Classifier was the most suitable model to use for this prediction as it provided the highest f1-score for class 1 with 63% and a recall of 85%. We used the criteria ‘max_depth = 3, min_samples_leaf = 5’ within the parameters of the model, which also helped to balance the prediction class as we noticed from a visualization using ‘Graphviz’ (Navlani, 2018) [7] that there were multiple cases where a leaf node that was class 1 would have less than 5 samples in it. The output including these parameters shown below also helped us to better understand how this model operated and made predictions than the others.



Conclusion

To conclude, we found this dataset interesting to study whilst also being able to investigate it deeply in order to find a problem and attempt to solve it using a machine learning model. Managing to make sense of the vast amounts of data through visualizations greatly encourages the process of exploration, and enabled us to discover things that we might not have expected to see prior. Although we did find this dataset and problem domain to be quite engaging due to it being based on a real world application, the model performance was not as strong as we would have liked it to be, presumably due to the low correlation between the features that were used when training the models. We learned that the process of selecting the best possible features and samples is paramount to a model being successful in regards to prediction.

References

[1] Sunil Jacob. (2018). Bank Loan Modelling.
Available at: <https://www.kaggle.com/itsmesunil/bank-loan-modelling>
[Accessed 12 Nov. 2019]

[2] Kaggle. (2010). Kaggle: Your Home for Data Science.
Available at: <https://www.kaggle.com>
[Accessed 12 Nov. 2019]

[3] Syahiid Nur Kamil. (2018). Personal Loan Dataset (Binary Classification)
Available at: <https://www.kaggle.com/iconoclash/personal-loan-dataset-binary-classification>
[Accessed 12 Nov 2019]

[4] Michael Galarnyk. (2016). Heat Maps using Matplotlib and Seaborn.
Available at:
<https://github.com/mGalarnyk/Python_Tutorials/blob/master/Request/Heat%20Maps%20using%20Matplotlib%20and%20Seaborn.ipynb>
[Accessed 23 Nov. 2019]

[5] Rafael Alencar. (2017). Resampling strategies for imbalanced datasets.
Available at: <https://www.kaggle.com/rafjaa/resampling-strategies-for-imbalanced-datasets>
[Accessed 23 Nov. 2019]

[6] April Chen. (2016). Pre-Modeling: Data Preprocessing and Feature Exploration in Python.
Available at: <:https://github.com/aprilypchen/depy2016/blob/master/DePy_Talk.ipynb>
[Accessed 24 Nov. 2019]

[7] Avinash Navlani. (2018). Decision Tree Classification in Python.
Available at:
<https://www.datacamp.com/community/tutorials/decision-tree-classification-python>
[Accessed 05 Dec. 2019]

John Fisher. (2018). diabetesNeuralNetwork.
Available at:
<https://github.com/jg-fisher/diabetesNeuralNetwork/blob/master/keras_diabetes_classification.py>
[Accessed 09 Dec. 2019] Not referred to in report, used for parameter tuning neural network model