Loan granting analysis using ML methods   
Ignacio González

ig2348z@greenwich.ac.uk

University of Greenwich, CMS Department

Old Royal Naval College, United Kingdom

*Abstract*—The aim of this paper is the study of different machine learning approaches used to predict whether a loan will be granted or not. The paper will describe the whole process, from data pre-processing until the results of the classification procedure and why we used the classification approach. Some insights about the data were provided in the folder for the reader to check the people who are most likely to receive the loan.

Key words: machine learning, loan, machine learning methodologies, supervised learning and classification.

# Introduction

Machine learning (ML) is one of the branches of Artificial Intelligence (AI) in which different processes are studied with the main objective of emulating the human learning process. The range of application of the tools provided by the ML field is widening as new discoveries emerge. That is relevant nowadays because it could be used either in finances, medicine, artificial intelligence with different purposes, etc. In this paper, basic concepts of ML will be analysed with the purpose of showing our chosen approach for classifying whether a loan will be granted or not. The reader must note that this is a Machine Learning project (Applied Machine Learning view), thus it focuses on the development, understanding and results of the code rather than explaining the attributes of the people who will receive the loan (Business Intelligence view). Therefore, the results section will cover the obtained accuracies of different deployed models and their features. Nevertheless, Excel was used as an extra visualization tool for analyzing the people who are most likely to receive the loan.

# Related work

There are different papers that were critical for the student to decide to start researching more about this topic. The first one related to loan granting processes and the remaining ones related to credit approvals.

In the first one, Pazzani, M. J., & Brunk, C. A. (1991) tried different methods for error detecting in rule-based expert systems using a similar dataset that the one in this project.

Tumer, K., & Ghosh, J. (2002) developed a system that receives information from different classifiers based on older statistics for pattern recognition using a dataset about credit approval procedures.

# Dataset, Initial Questions and Pre-Processing

The question that was approached was *predicting whether the loan will be granted or not*. Thus, we began with the pre-processing procedure and then, it was divided into two different parts (regarding the dataset 1 or training dataset), the training and testing part, so as to fit the models and find their accuracy. After that, we were provided with an additional dataset (dataset 2) which has records for all the attributes but not the labels. We used the last dataset to predict some results. However, we could not get the real accuracy since we had nothing to compare the predictions with.

The first dataset (training dataset) has 614 records while the dataset 2 has 367 records. The number of records will change when pre-processing the data as shown in **Figure 4**. This is because we got rid of the records with ‘NaN’ values.

Also, a new question appeared: ‘*is there any correlation between the attributes?*’

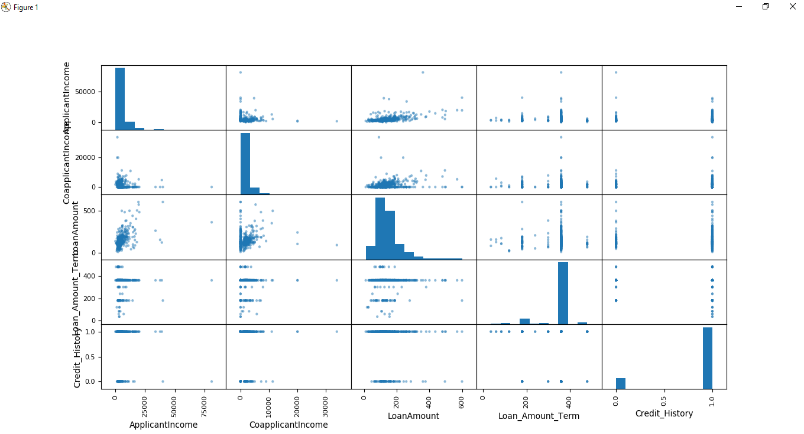


Figure Scatter-plot

At the first stages (and talking about the dataset 1), we can state that there is a correlation between ‘LoanAmount’ and ‘CoapplicationIncome’ and again ‘LoanAmount’ and ‘ApplicationIncome’ as shown in **Figure 1**. In these cases, it means, the bigger ‘Loan Amount’ gets, the bigger ‘CoapplicationIncome’ and ‘LoanAmount’ become. We can state that these variables have a positive lineal correlation. **Figures 2 and 3** show information about the numerical data available in the training dataset that could be useful to build our dashboard. It could be interesting to talk about the number of outliers of the ‘ApplicationIncome’ as it can be seen in the **Figure 2** knowing that the vast majority of records are between 0 and 10000 pounds. The same could be stated in the ‘CoapplicationIncome’ but in this particular case, the majority of values go from 0 to 5000 pounds.

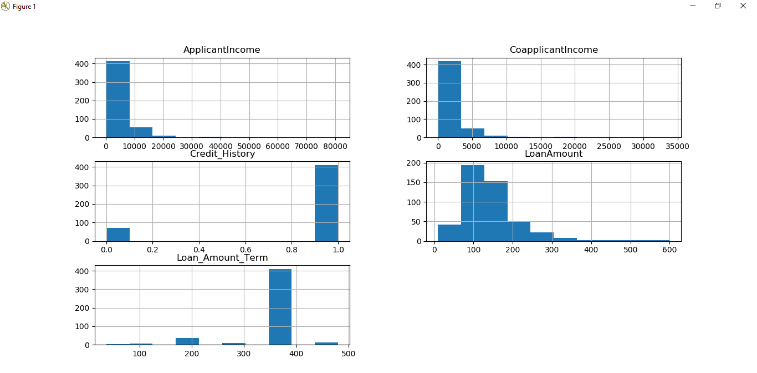


Figure Histograms

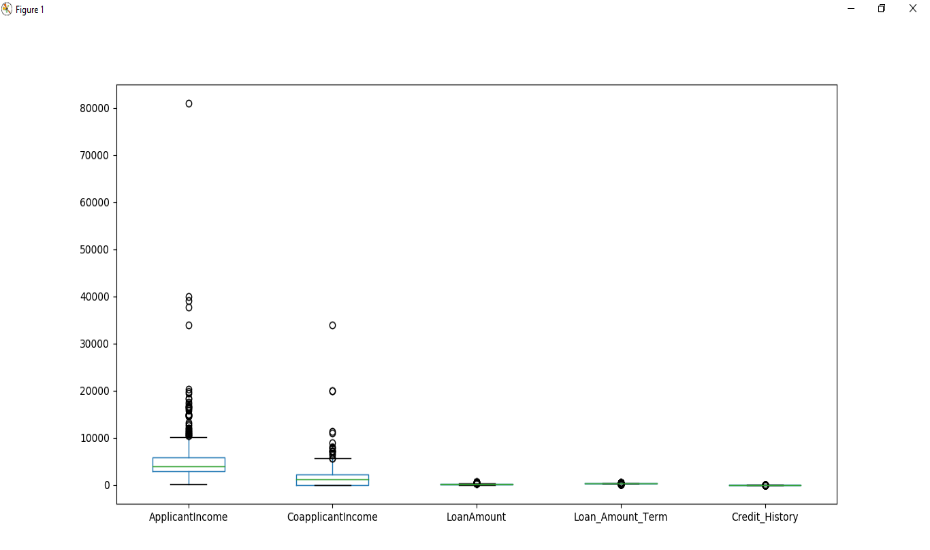


Figure Whisker-plots, from letf to the right side: ‘ApplicationIncome’, ‘CoapplicationIncome’, ‘LoanAmount’,’Loan\_Amount\_Term’,’Credit\_History’



Figure Datasets´ shape

A useful way of making sure the data is correctly uploaded is loading the first records to check them, **Figure 5**.



Figure First 20 records of the training dataset

As we can see in **Figure 6**, we have now access to information regarding mean and standard deviation of the numerical data. We can extract, using this information, values like outliers by just comparing the ‘max’ and ‘min’ with the ’mean’.

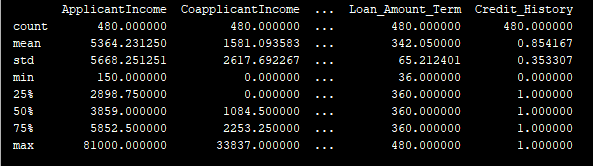


Figure General information of the shown attributes

Last but not least, relevant information about the extant categorical data is depicted in the **Figure 7**.

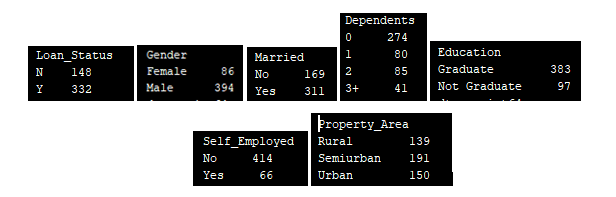


Figure Total records in dataset 1 grouped by attributes

After this stage, it was necessary to decide which methods we were going to use. Bearing in mind that our question was ‘Are we able to predict the results?’ what we needed to approach was a classification problem. Given certain number of attributes, predict the result (‘YES’ or ‘NO’) of the loan granting decision. Classification is a technique widely used in ML for different purposes like handwriting recognition, speech recognition, biometric identification, etc. There are different kinds of classifiers: Binary Classifiers (two possibilities) and Multi-Class Classifiers (a range of possibilities). We can affirm that classification is a supervised learning approach in which the system tries to learn from different inputs and each record has been labelled.

Before starting to talk more deeply about the methods, it would be interesting to say that all the data had to be pre-processed (in this particular case because it was necessary) with the purpose of eliminating ‘NaN’ (missing) values or substituting categorical data by numerical data, so we can compute an answer based on all the attributes. A scaling procedure was also performed with the purpose of eliminating differences between high values (‘ApplicationIncome’) and smaller ones (‘LoanAmount’) so we avoid the results to be biased. **Figures 8, 9 and 10**.

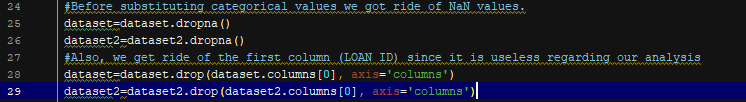


Figure ‘NaN’ values dropping process

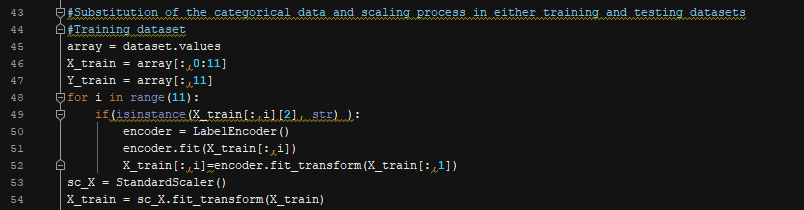


Figure Categorical substitution and scaling procedure of dataset 1

Therefore, we can conclude that what we need is a solution for classifying labels with the purpose of differentiating whether a person is going to get the loan or not.

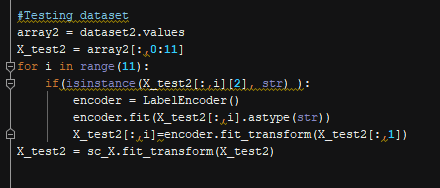


Figure Categorical substitution and scaling procedure of dataset 2

# Test and Validate Models

Once the datasets are already cleaned (either training dataset and testing dataset), we can start analyzing the models/methods. Notice in the code (accessible in the last section of the report) that stratified sampling was performed in order to avoid label distribution imbalance.

The selected methods were (Pedregosa *et al.*,2011):

* **Logistic Regression**: Statistical approach in which there are a range of independent variables that decide the result.
* **K-Neighbors Classifier**: k-NN classifies results depending on neighbors. The result is the most common value among its k neighbors (the nearest ones).
* **Decision trees**: Classifies data based on rules. Once a differentiator is selected, the method considers the different answers and chooses the most accurate one.
* **Linear Discriminant Analysis**: It is a generalization of Fisher´s linear discriminant, in which using statistical analysis, it finds a linear combination of the attributes (it uses independent variables).
* **Gaussian Naïve Bayes**: It assumes the independence between variables (that is why it is naïve). The best part of this method is that a small number of variables is required to predict new values.
* **C-Support Vector Classification**: Different implementation of the same algorithm. Another popular method is SVM (Support Vector Machine). It analyzes different points in space and divide them by a gap as wide as possible. Results are stated depending on the side of the plane of the point we are analyzing.

According to giant *Microsoft (Microsoft Azure, 2019)*, it suggests us trying different approaches depending on the attributes and records available.

1. SVM (Support Vector Machine): For linear models with more than 100 features.
2. Logistic Regression: For linear models that need fast training.
3. Bayes: For linear models that need fast training.
4. Decision Forest: Provides fast training with high accuracy.

k-NN and Linear Discriminant Analysis were tried as well because their popularity. Cross-validation (**Figure 11**) was performed with the purpose of assessing the results of the statistical analysis. It created 10 different/independent groups (training and test data groups) with the purpose of validating the models and calculate the mean of the accuracy and the standard deviation (we will call them ‘t\_accuracy’ and ‘t\_std’ in this report).

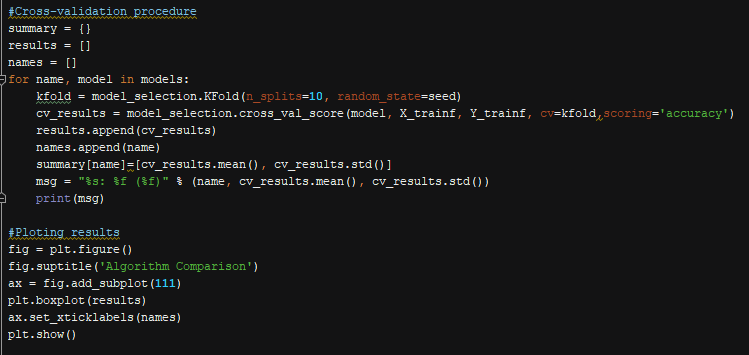


Figure Cross-validation

**Figure 12 and 13** show different ways of evaluating the t\_accuracy of the models. In both we can find out which one is the worst model regarding this experiment (decision tree) and using the whiskers plot we can compare which ones were closest of being the best (LR, SVC and Linear Discriminant Analysis), because their outliers.

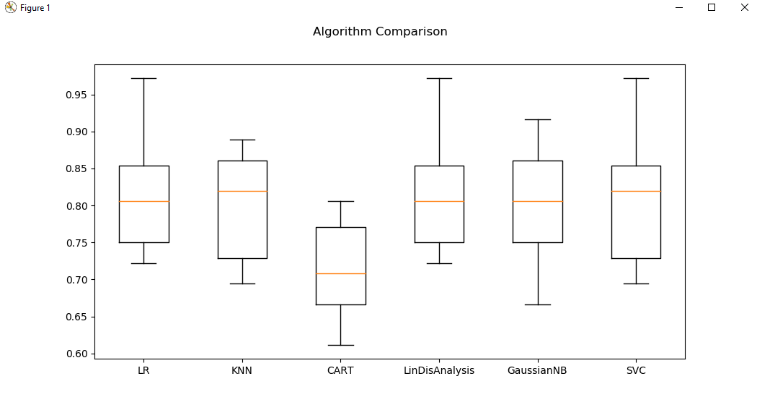


Figure Whisker-plot of the accuracy means of the models

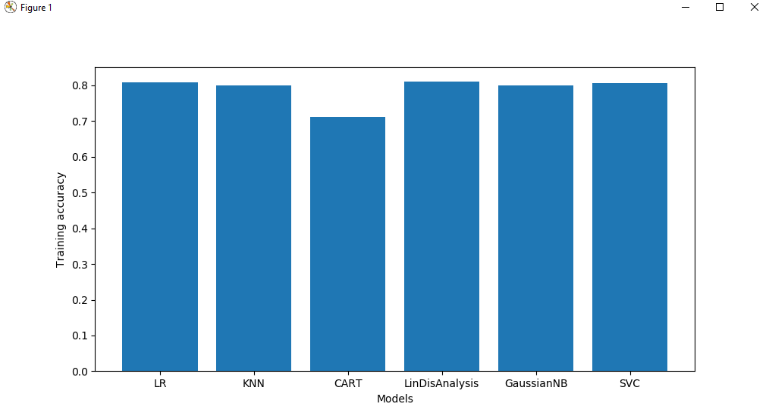


Figure Histogram of the accuracy means of the models

**Figure 14** shows a graph in which ‘t\_std’ is presented, and we can state that ‘CART’ has the lowest std and ‘SVC’ the highest one. That means, during the process of cross-validation, ‘CART’ model was the one with smallest standard deviation, the results of the accuracies were similar for the same model (low std) even though it was the worst one for the case under study.

Tested accuracy represents the accuracy obtained when testing the models once they were fitted (still dataset 1). The graph in the **Figure 15** concludes that the best accuracy comes from the Logistic Regression although in the ‘t\_accuracy’ graph it was not clear.

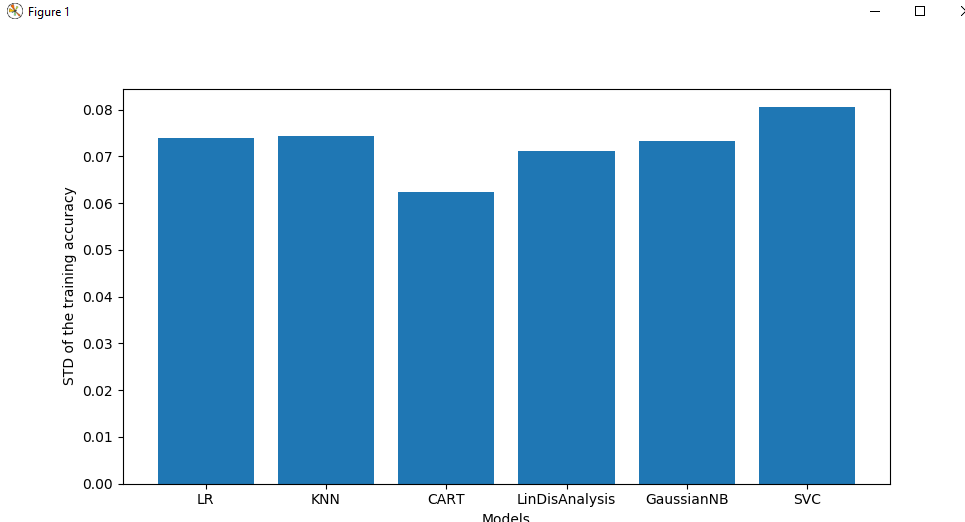


Figure STD means of the models

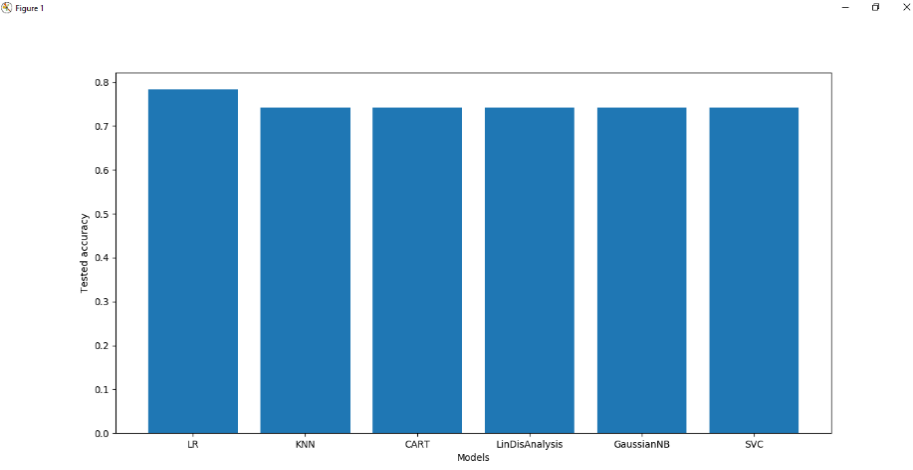


Figure Tested accuracyof the models

The reader will find a summary of the models´ features in **Table 1**.

|  |  |  |  |
| --- | --- | --- | --- |
| **Model** | **t\_accuracy** | **t\_std** | **Tested accuracy** |
| **LR** | 0.8083333333333333 | 0.0739640386427464 | 0.7833333333333333 |
| **KNN** | 0.7999999999999999 | 0.07432826755699808 | 0.7416666666666667 |
| **CART** | 0.7222222222222222 | 0.05957669608202007 | 0.7333333333333333 |
| **LDA** | 0.8111111111111111 | 0.071145824860365 | 0.7416666666666667 |
| **GaussianNB** | 0.8 | 0.07328281087929402 | 0.7416666666666667 |
| **SVC** | 0.8055555555555556 | 0.08050764858994135 | 0.7416666666666667 |

Table 1 Models´ features

# Final Analysis and Visualization

In this section we are going to focus on the results that we have obtained. However, it is necessary to talk about the fitting process of the models. Snips of code as an example of how ‘LR’ was fitted could be found in **Figure 16**. We are going to focus on this particular model since it was the best for the dataset according to its accuracy although all the models follow a similar way of fitting process. The code for the rest of the models could be found in the same folder.

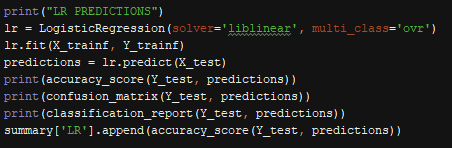


Figure Code screenshot that contains the fitting and predicting procedure

In **Figure 16** we can find how we defined the classifier of the model ‘LR’, how we fitted the classifier and how we obtained the predictions (the results) based on the ‘Y\_test’. ‘Y\_test’ is part of the main dataset (the training dataset) and we used it so as to measure the accuracy on a real test. Nevertheless, we were provided with another dataset (dataset 2) which does not have labels. Thus, we can predict the values from this dataset, but we cannot compare them in order to get the accuracy of these predictions.

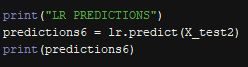


Figure ‘LR’ prediction for the dataset 2

**Figure 17** depicts the piece of code that deals with the predictions of the second dataset ,and the results are shown in **Figure 18**. Unfortunately, we were not provided with the labels of the records therefore, we cannot be sure of the accuracy of this procedure.

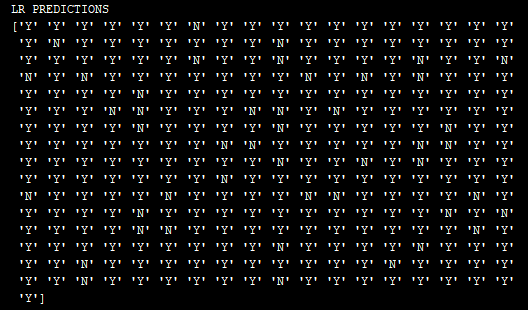


Figure Results of the predictions

It could be interesting to compare the confusion matrix and the final report of either ‘CART’ and ‘LR’ models since they were the worst and the best one:



Figure CART´s confusion matrix

A confusion matrix is used as a tool to visualize the performance of a supervised-learning model (**Figures 20, 21, 22 and 23).** Each column represents the number of prediction of each class (Yes/No) while each row represents the real instances. C0,0 represents true negatives, C1,0 represents false negatives, C1,1 deals with true positives and finally, C0,1 is the false positive group (Narkhede, 2018).

‘What can we extract from the classification report?’

* **Micro average**: averaging the total true positives, false negatives and false positives
* **Macro average**: averaging the unweighted mean per label
* **Weighted average**: averaging the support-weighted mean per label
* **Precision**= (True positives)/(True positives+ false positives)
* **Recall**= (True positives)/(True positives+ false negatives)
* **F1**= 2x ((precision\*recall)/(precision+recall))

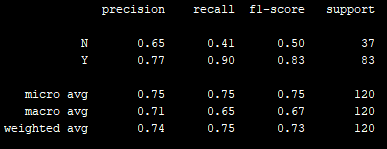


Figure CART´s report



Figure LR’s confusion matrix

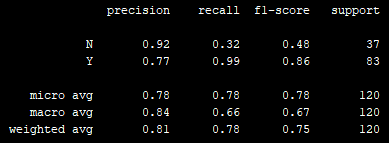


Figure LR´s report

LR has fewer true and false negatives and more true and false positives.

## Future work

The purpose of this sub-section is to critically assess the part of the project that can be improved. All the tasks of the project were achieved, and, in some cases, the student went beyond the requirements (assessing more than 3 models in this case).

As a future work, it could be interesting to improve the accuracy by code optimizations. Also, the testing part could have been better if we had access to the labels of the records since we could have measured the accuracy. For future work, richer (bigger) datasets would be really useful.

Also, the ‘NaN’ elimination procedure. The student deleted all the rows with ‘NaN’ values, but the optimal solution would be methods like mean/median substitution either for categorical or numerical data.

# Conclusions

To sum up, the final objective was to predict accurately the results, and for this purpose optimized models must be used. In our case, some models were analysed although the obtained accuracy was a little bit low in all of them. The whole process and code were presented and explained. Due to the limit of the pages for the project, some insights about the data were provided in the folder for the reader to check the people who are most likely to receive the loan.

# References

Tumer, K., & Ghosh, J. (2002). Robust combining of disparate classifiers through order statistics. Pattern Analysis & Applications, 5(2), 189-200. [Online] Available at: <https://link.springer.com/content/pdf/10.1007/s100440200017.pdf> [Accessed: 18/04/2019]

Pazzani, M. J., & Brunk, C. A. (1991). Detecting and correcting errors in rule-based expert systems: an integration of empirical and explanation-based learning. Knowledge acquisition, 3(2), 157-173. [Online] Available at: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.18.1750&rep=rep1&type=pdf> [Accessed: 18/04/2019]

[Scikit-learn: Machine Learning in Python](http://jmlr.csail.mit.edu/papers/v12/pedregosa11a.html), Pedregosa et al., JMLR 12, pp. 2825-2830, 2011. [Online] Available at: https://scikit-learn.org/stable/index.html [Accessed: 22/03/2019]

Microsoft Azure, 2019, ‘Machine learning algorithm cheat sheet for Azure Machine Learning Studio’ [Online] Available at: https://docs.microsoft.com/en-us/azure/machine-learning/studio/algorithm-cheat-sheet [Accessed: 22/03/2019]

Narkhede Sarang, 2018, ‘Understanding Confusion Matrix’, Medium. [Online] Available at: https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62 [Accessed: 22/03/2019]