Credit Card approval prediction and algorithm bias assessment in the Credit Card Dataset (Seanny, 2020)

Erblin Marku, MSc Computer Science, Queen Mary University of London
Data Analytics, March 2022

Abstract

I will perform a Credit Card Approval prediction on the Credit Card Dataset. The database is not labelled so for the first part of the analysis I will label the data myself and perform 2 supervised learning models and discuss the results. In the second part I will perform an unsupervised clustering method. *I will show how good are this model for future predictions by analysing result metrics*. My second objective is to assess algorithm bias. Bias is an ambiguous term and we may have heard a lot the saying – all data is biased. This is of course practically true because the data is collected by humans and also is collected for a certain purpose, which means certain data is gathered (which a human may have seen relevant) but also a large amount of data (which a human may have seen as not relevant) is left behind. We will take in consideration all kind of biases, but we will be more determined on bias which effects our prediction more.

1 Introduction

The public and private sectors are turning to Artificial Intelligence systems and Machine learning more and more to automate their workflow and decision-making techniques. Digitalization of data is disrupting most economic sectors, and this has called for regularization and call for attention when using machine learning algorithms which are not safe against bias. The availability of massive datasets has made it easy for humans to use Ai in their decision-making.

Despite the strong start of these algorithms in the latest years research has shown some troubling examples in which these decisions replicated or even amplified human bias. (CHODOSH, 2018) These examples and others showed that bias is present and cannot be easily reduced, although the best propositions come for a trade-off.

This trade-off has to be made between 'fairness' and 'accuracy' (Nicol Turner Lee, 2019)which practically means for businesses or corporation to take more risk in order to be fair and reduce bias. This cost is not always taken into consideration, but overall is a matter of trust. Other recommendations include the involvement of humans in maintaining the algorithms, government policies, employee algorithm literacy etc. All these come with cost and risk which is part of the trade-off.

2 Dataset Exploration

Our dataset has 2 '.csv' files. The first one has all client data labels and the second one has the client's financial data. We will use the second file to label our clients as credit card approved or not. The first database has a lot of information, starting with personal data for the client and going on with employment, family and real estate possessions. First, we will deal with some pre-processing on the main dataset and then get the labelling in place.

Our target value will be a column which I will call status and will have a binary value of 0 for denied or 1 for approved.

First let's import our dataset as a *pandas dataframe* and see how many entries we have.

```
4]: crd_df.info()
    #info says we have 18 columns and 438557 entries(r
    #we see that in the rows we have some data missing
    #the others seem ok for now

    <class 'pandas.core.frame.DataFrame'>
    RangeIndex: 438557 entries, 0 to 438556
    Data columns (total 18 columns):
     #   Column               Non-Null Count   Dtype
    ---  ------               --------------   -----
     0   ID                   438557 non-null  int64
     1   CODE_GENDER          438557 non-null  object
     2   FLAG_OWN_CAR         438557 non-null  object
     3   FLAG_OWN_REALTY      438557 non-null  object
     4   CNT_CHILDREN         438557 non-null  int64
     5   AMT_INCOME_TOTAL     438557 non-null  float64
     6   NAME_INCOME_TYPE     438557 non-null  object
     7   NAME_EDUCATION_TYPE  438557 non-null  object
     8   NAME_FAMILY_STATUS   438557 non-null  object
     9   NAME_HOUSING_TYPE    438557 non-null  object
     10  DAYS_BIRTH           438557 non-null  int64
     11  DAYS_EMPLOYED        438557 non-null  int64
     12  FLAG_MOBIL           438557 non-null  int64
     13  FLAG_WORK_PHONE      438557 non-null  int64
     14  FLAG_PHONE           438557 non-null  int64
     15  FLAG_EMAIL           438557 non-null  int64
     16  OCCUPATION_TYPE      304354 non-null  object
     17  CNT_FAM_MEMBERS      438557 non-null  float64
    dtypes: float64(2), int64(8), object(8)
    memory usage: 60.2+ MB
```

Figure 1 Structure of our main dataset

We can see we have 18 columns and 438557 entries in most of the. The OCCUPATION_TYPE columns seems to have less data; in this case we will have to fill these entries or drop the whole column. We will check later and decide what to do.

3 Statistical overview

All data is organized by a client ID, after removing duplicate values I will look at the *dataframe* description which gives me information about

mean and standard deviation among others. I need to look at the standard deviation values, if the values are large it means there is data in those columns positioned far from the mean, which means we may have outliners to remove so we build a more general model.

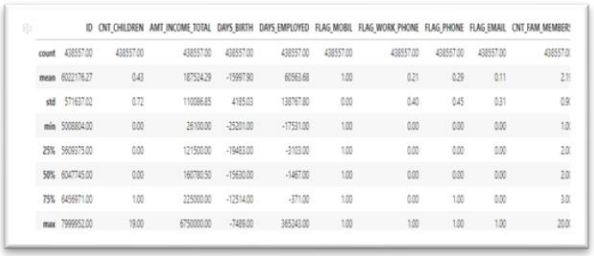
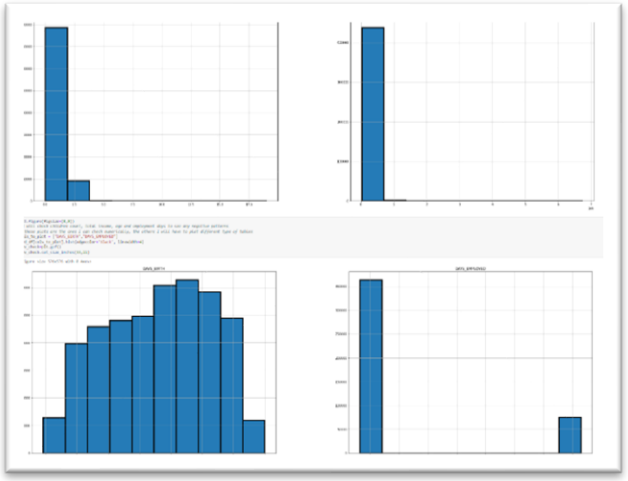


Figure 2 Describe function output.

As I can see from the describe function output, I may have problems with the AMT_INCOME_TOTAL, DAYS_BIRTH and DAYS_EMPLOYED. I need to check the graphs for these and perform some more pre-processing. I will convert the days for working and birth in years, so I have a normal distribution. For the total income I must see the graph.

4 Visualization

First, I will plot the numerical data, for this I will use 2 types of charts. We have 4 numerical labels and the first graphs will be histograms, then I will show distribution plots and plot box views for each of the to spot distribution and outliners. We can spot outliners from 3 of them, only age seems good in all graphs. For the others I will perform a cut for outliners.



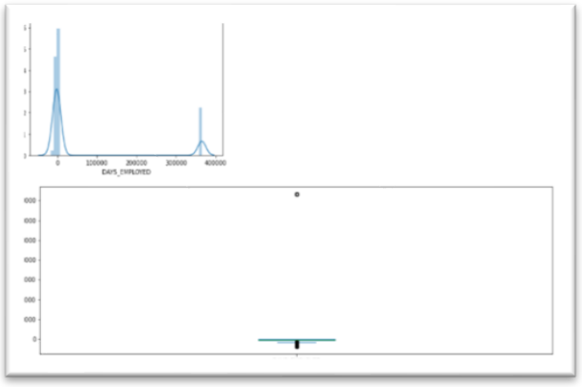


Figure 3 Histograms of numerical labels on the left. Distribution plot and box plot of days employed label on the right.

Next, I will plot the categorical values, for this I will only use the Histograms plot. I see no major problems in their distributions so I will not plot anymore graphs for these labels.

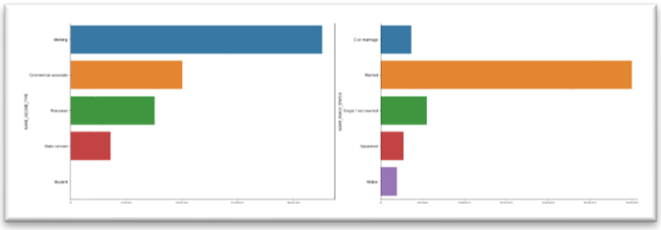


Figure 4 Histograms for Income type and Family status.

The last visualization will be for the yes/no labels, I will show these in pie charts to get an overall view of their distributions. We have 7 labels with yes/no or 1/0 values. The FLAG_MOBIL label seems to have only 1 value form the graph and count function so we will have to drop it since it is redundant for our model.

Figure 5 Pie charts for Gender, car and realty ownerships.

Here we may have a small problem about the balance, since is seems these labels tend to give more on one side, but we will explore the imbalance a little later since it is one of the main purposes of our analysis.

5 Pre-processing

This part will involve 6 steps:

- convert yes/no labels to 1/0 and other as categorical values

- calculate z-scores for problematic labels and remove outliners
- convert days employed and birth to years
- drop null value rows found in OCCUPATION TYPE from the dataframe
- remove FLAG_MOBIL label since it is redundant
- rename labels for ease of use

All the above are done manually by small functions of pandas explained in the code. For the outliner removals we could do it manually also, by looking at the values in the charts and remove all values above that. To be surer I will use the *z-score*. The *z-score* is the distance of a value from the mean in standard deviation units. So, it will be simple to build a function to spot that. After calculation the z-score I need to set a threshold for outliners, which in medical areas seems to get as low as 2 but in my analysis for credit card approval I want to keep all the data possible. So, by the empirical rule I will keep the threshold to 3, so I can keep 99.7% of my dataset intact. (HAYES, 2021)

After the pre-processing part this is the current form of my dataset.

```
[39]: crd_clean.count()
      #we can see that the rows are unified and we h

[39]: ID                   287950
      CODE_GENDER          287950
      FLAG_OWN_CAR         287950
      FLAG_OWN_REALTY      287950
      CNT_CHILDREN         287950
      AMT_INCOME_TOTAL     287950
      NAME_INCOME_TYPE     287950
      NAME_EDUCATION_TYPE  287950
      NAME_FAMILY_STATUS   287950
      NAME_HOUSING_TYPE    287950
      DAYS_BIRTH           287950
      DAYS_EMPLOYED        287950
      FLAG_MOBIL           287950
      FLAG_WORK_PHONE      287950
      FLAG_PHONE           287950
      FLAG_EMAIL           287950
      OCCUPATION_TYPE      287950
      CNT_FAM_MEMBERS      287950
      AGE                  287950
      YEARS_EMPLOYED       287950
      dtype: int64
```

Figure 6 Current size and types of our labels data.

6 Labelling

Now we will deal with the second file. While labelling I come across the other major problem. If we look at the counts of debit classification, we can see this.

```
[122]: crdr_df['STATUS'].value_counts()

[122]: C    442031
       0    383120
       X    209230
       1     11090
       5      1693
       2       868
       3       320
       4       223
       Name: STATUS, dtype: int64
```

Figure 7 Counts of labels in the dataset

```
[59]: res_df['Status'].value_counts()

[59]: 1    45847
      0      138
      Name: Status, dtype: int64
```

Figure 8 Counts of labelling data.

We have a problem because C, O and X values must be classified as good debts, but then we end up with an imbalanced dataset. So, if we want the prediction to work, we will have to deal with the imbalanced dataset, otherwise we will have to consider X values as bad debt, which is biased against the clients. We will go with the imbalanced dataset to check the results.

7 Dealing with imbalanced datasets

Most of the algorithms assume balanced datasets, so need to choose our model carefully to minimize the effect on the testing set. (Zhaohui Zheng, 2004). There are different ways to handle imbalanced datasets. Random Oversampling and Under sampling are an option, others being threshold method, one-class learning and cost-effective learning. The best option seems to be a hybrid method which combines classifiers. As resulted in (Sotiris Kotsiantis, 2006) the best result performers are hybrid methods which do better than cost-sensitive and random resampling.

The best performer is CatBoost which is an open source ml algorithm from Yandex. The main feature of CatBoost is that it works well with multiple categories of data. The other feature is that uses gradient boosting algorithm which is widely used in business. This model benefits us because it has robust results, which mean we don't need hyper-parameter tuning and the chances for overfitting are very low. This means we get a more generalized model but since it is not part of our analysis, we will try different methods to get near CatBoost. Since CatBoost works with gradient boosted decision trees, I will try in one of my models the DecisionTreeClassifier along with LogisticRegression. Then I will try the K-Means clustering with no labels so I can get an idea of how many clusters is this dataset possible to divide.

In order to check imbalance and biased information I will plot all the variables in composite graphs to see their counts and percentages in each of the '1' and '0' categories. Because sometimes the problem is not the

imbalanced dataset but rather the imbalanced classes. For this I wrote a function (Zsolt, 2021) to plot histograms for numerical labels and bar plots for categorical values.

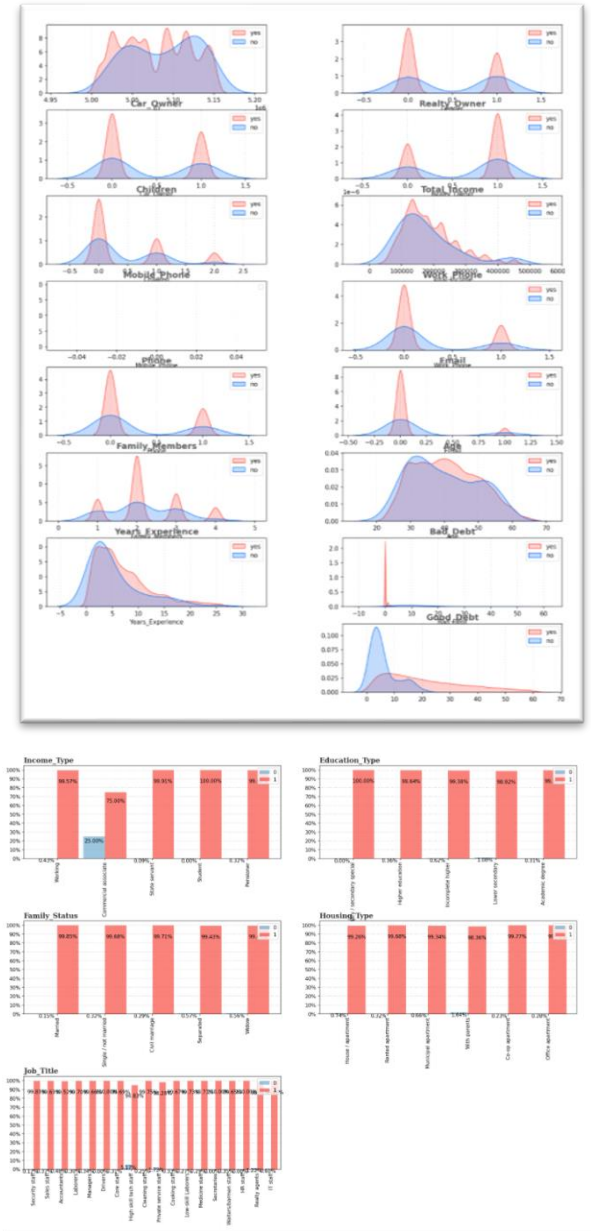


Figure 9 Histograms for numerical and %bars for categorical labels.

As we can see from the graphs all the labels are imbalanced and they have large values and counts in the *yes (*'1') categories. This will affect massively our result which I predict our models to do very poorly on the 0 class and perform very good on the 1 class.

I will check for correlations so I don't get the model to train on similar labels which will affect my results.

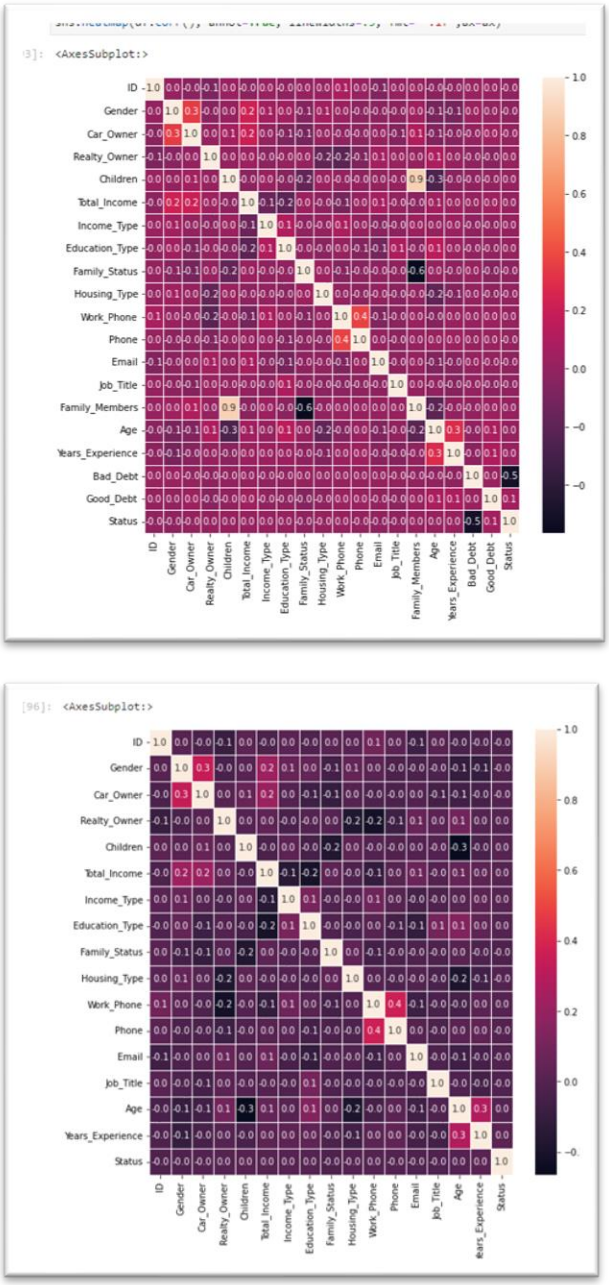
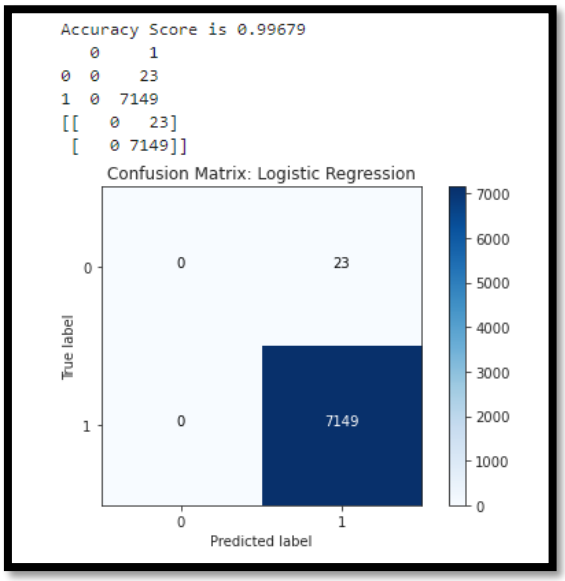


Figure 10 sns Heatmap with correlation, on the right the heatmap after removing correlating label.

From the heatmap we can see that family members and children are correlating. So, we will use only children since family members have also a bit of correlation with family status.

8 Cross-Validation and Results for Supervised methods

According to research the imbalanced dataset needs careful cross validation because if we use the simple one, we will get overfitting models. After trying the normal cross validation with both *LogisticRegression* and *DecisionTreeClassifier*, the results were that my model was overfitting.



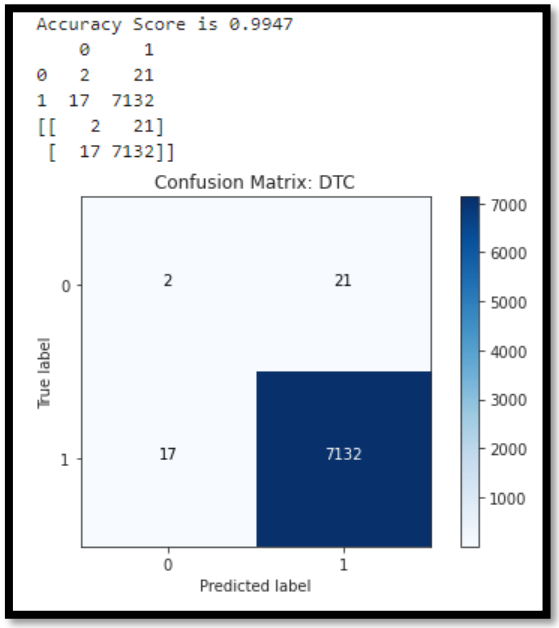


Figure 11 Results of LogisticRegression and DecisionTreeClassifier

According to (M. S. Santos, 2918) the overoptimistic model is not affected by the dataset sample size or the imbalance ration of the classes. Rather it is a matter of prediction complexity, which after their testing resulted that over the oversampling methods SMOTE+TL and MWMOTE were the best performers. Although I am going to try a SMOTE method in this analysis to check the differences.

As an extra on my analysis, I have performed a SMOTE oversampling on my dataset but still the Logistic regression model perform very poorly on the 0 label and we can see the failure by the shape of the ROC curve in the notebook.

9 Best Features

I have performed a *selectKbest* features for our dataset to see which features perform best and try the model only with those. As the SMOTE predicted the best feature is the income amount but still the results are poor with now significant improvement on the prediction results. Here are the 5 best features.

Figure 12 5 best performing features

10 Kmeans

First thing to do when performing Kmeans is feature scaling (Roy, 2020) . Since our model will se number it may give bigger weights to labels with higher values, for this we need to scale the data down to some number that will mean the features will be treated the same from the algorithm. K-Means and PCA both use Euclidian distance which makes feature scaling a must in our case. The scaling is performed using *MinMaxScaler*, more information can be found on the notebook.

Kmeans with 2 clusters is tried and the results seem not good.

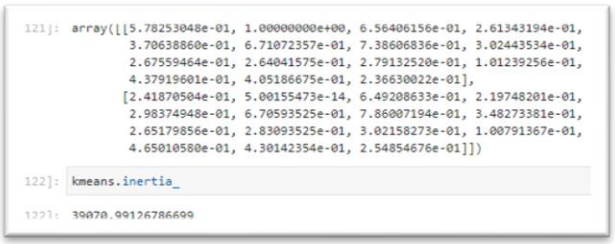


Figure 13 The centers of the clusters and inertia value.

Inertia value tells us how our model is doing. It is the squared distance of the samples from their cluster's centre. It has values from 0 and up. Ours is very high which means our model is not performing well. Our accuracy is 0.58 which is better than the overfitting models, because it means our model is getting nearly half of the labels right but we need to see how it performs on the False Negatives and True Negatives because

we are still dealing with an imbalanced dataset and so we need to get a general model for both approval and denial.

```
156]: print(classification_report(y, labels))
                  precision    recall  f1-score   support

               0       0.00      0.43      0.01        82
               1       1.00      0.58      0.73     23824

        accuracy                           0.58     23906
       macro avg       0.50      0.50      0.37     23906
    weighted avg       0.99      0.58      0.73     23906
```

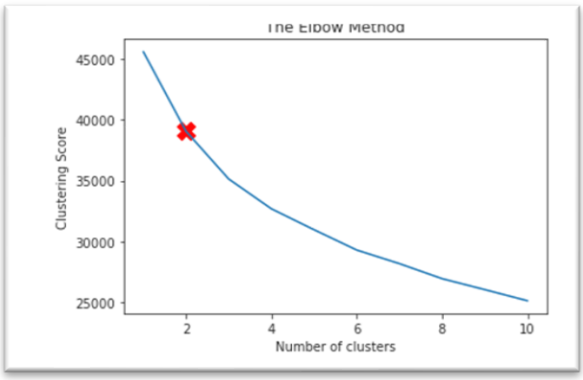


Figure 14 Classification report and Elbow method graph

First, we will measure the silhouette score from 2 to 10 clusters to see if we may have any better option and it seems like the best is 2 clusters with the largest distance of nearly 0.46. The other forms seem to have smaller distances to neighbour clusters.

```
    print ("For n_clusters = {}, silhouette score is {}".format
For n_clusters = 2, silhouette score is 0.4655578254479565
For n_clusters = 3, silhouette score is 0.37983935312155875
For n_clusters = 4, silhouette score is 0.385698979883062
For n_clusters = 5, silhouette score is 0.39590169268464354
For n_clusters = 6, silhouette score is 0.3825713649401952
For n_clusters = 7, silhouette score is 0.39012701517667386
For n_clusters = 8, silhouette score is 0.38859668350543514
For n_clusters = 9, silhouette score is 0.3740123368719553
For n_clusters = 10, silhouette score is 0.36756960989728626
```

Figure 15 Silouette score for clusters 2-10

Let's see the confusion matrix for our model.

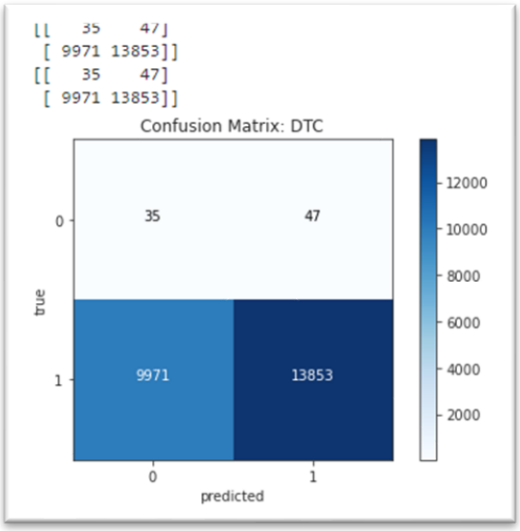


Figure 16 Confusion Matrix for K-means

To see the clusters in predicted and true graphic representation I will use PCA and from that we can see that the model does well on 1 label but misses or mis clusters the 0 labels.

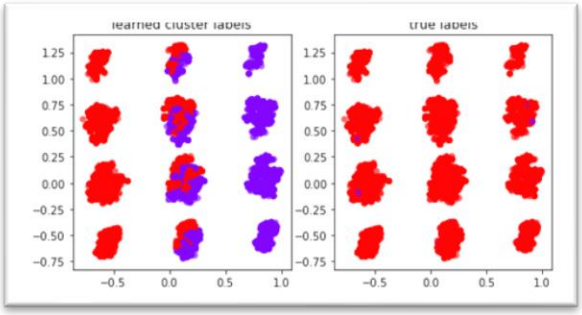


Figure 17 PCA decomposition of true clusters and K-means clusters

As a last resource I found I used a Gaussian Mixture model because I thought maybe the data is in way that we need to account for variance. Gaussian Mixture models are probabilistic models rather that hard clustering like K-means. So maybe doing a soft clustering is going to get our accuracy better but it did not work since the Accuracy score was 0.56 which is worst than Kmeans.

```
## import the GaussianMixture class
from sklearn.mixture import GaussianMixture
gm = GaussianMixture(n_components=2, random_state=123, n_i
preds = gm.fit_predict(X)
correct_labels = sum(y == preds)
print("Result: %d out of %d samples were correctly labeled
print('Accuracy score: {0:0.2f}'. format(correct_labels/fl

Result: 13362 out of 23906 samples were correctly labeled.
Accuracy score: 0.56
```

Figure 18 Gaussian Mixture Model accuracy

1 Further work

For further work and result there are some possibilities. First, we need an accurate labelling method since mine seems either to overfit or be not accurate. Another way is to populate the database with more negative debts so we can achieve some balance in the dataset and get a more general model.

Regarding the model we could use *CatBoost* which by the research is the most accurate for this type of imbalanced dataset. More elaborate work can be done with features, there is a way in *CatBoost* with *gridsearch* to evaluate the features and get the best ones to then use in the model. Due to the time needed for the assignment this was not possible (Liudmila Prokhorenkova, 2019).

Other propositions include hybrid methods like training each target class separately and then get samples of each trained set to get the overall model together. (Sotiris Kotsiantis, 2006)

2 Conclusion

We can conclude by the evaluation metrics of our model that there has trace of algorithm bias since it overfits on the class which has more data but poorly performs on the other class. This is mainly due to the imbalance of the dataset and means that our model will be inclined to say yes to the credit approval. The K-means without labels is better but when we check the results still algorithm bias is present. The model performs better on yes classes. These models are not general and are not good predictors so, further work must be done. The performance of the K-means model is still acceptable, but we must keep in mind that the 0.58 is a low accuracy and the algorithm will miss nearly half of the predictions.

References

CHODOSH, S. (2018). Courts use algorithms to help determine sentencing, but random people get the same results. *POPULAR SCIENCE*.

Dienes, E. (2011). *CTSPEDIA*. Retrieved from CTSPEDIA: https://www.ctspedia.org/do/view/CTSpedia/OutLier#:~:text=Any%20z%2Dscore%20greater%20than,standard%20deviations%20from%20the%20mean.

HAYES, A. (2021). Z-Score. *Investopedia*, 1.

Liudmila Prokhorenkova, G. G. (2019). *CatBoost: unbiased boosting with categorical features.* Yandex, Moscow, Russia: Moscow Institute of Physics and Technology, Dolgoprudny, Russia.

M. S. Santos, J. P. (2918). Cross-Validation for Imbalanced Datasets: Avoiding Overoptimistic and Overfitting Approaches. *IEEE Computational Intelligence Magazine*, 59-76.

Nicol Turner Lee, P. R. (2019). *Algorithmic bias detection and mitigation: Best practices and policies to reduce consumer harms.* Michigan, USA: BROOKINGS.

S. Moro, P. C. (2014). A Data-Driven Approach to Predict the Success of Bank Telemarketing. Decision Support Systems. *Elsevier*, 62:22-31.

Seanny. (2020). *kaggle* . Retrieved from https://www.kaggle.com/: https://www.kaggle.com/rikdifos/credit-card-approval-prediction

Sotiris Kotsiantis, D. k. (2006). Handling imbalanced dataset: A review. *GESTS International Transactions on Computer Science and Engineering* (p. Vol.30). ESDL, University of Patrs, Greece.

Zhaohui Zheng, X. W. (2004). Feature selection for text categorization on imbalanced data. *ACM SIGKDD Explorations Newsletter*, volume 6.

Zsolt, L. (2021, April). *Stackoverflow.* Retrieved from Stackoverflow: https://stackoverflow.com/a/67076347/4852724