Model approach and Model

5. Model approach

4 different models have been tested for this analysis plus a final stacking ensemble method which will include the four of them:

- Logistic regression.
- Decision tree classifier.
- Random forest.
- XGBBoost.

Every model will be tested for:

- Normalized data.
- Normalized-balanced data.
- Non normalized data.
- Non normalized-balanced data.

5.1 Dataset Normalization

- Apply list comprenhension.
- This is column-wise and min-max normalization.
- Could also be used mean & std normalization method with Pandas.
- Print the results to make sure it is correct.

```
non normalized: (395937, 31)
normalized: (395937, 31)
non normalized 1: (7005, 31)
non normalized 0: (388932, 31)
normalized 1: (7005, 31)
normalized 0: (388932, 31)
```

Insights:

- Since only 1.8% of the clients contracted a loan ,checking the performance of the model using also balanced data is required.

Balancing Dataset

Unbalanced dataset

- Sklearn used for 'train' and 'test' split for the 90% of non normalized and normalized dataset.
- The function `**train_test_split**` made random partitions for the two subsets below. A random state and a 0.25 test_size are also specified.
- Printed results fro checking purposed.

Non normalized

```
X: (356343, 32)
X_train: (267257, 32)
X_test: (89086, 32)
y_train: (267257, 1)
y_test: (89086, 1)
n° of 1  (1540, 1)
n° of 0  (87546, 1)
```

Non normalized

```
X_train_n: (267257, 32)
X_test_n: (89086, 32)
y_train_n: (267257, 1)
y_test_n: (89086, 1)
0.0    87546
1.0     1540
Name: got_loan, dtype: int64
```

Balanced dataset

- Repited the same for normalized and non normalized dataset but now balancing the dataset.
- Defined the features and the target.
- Synthetic Minority Oversampling Technique(SMOTENC) used for Over-sampling, under-sampling and the combination between both.
- Tested with the models, the one with best results is under-sampling.
- Defined 30% under- sampling strategy.
- Printed resutls for checking purposes.

Non normalized

```
Dataset after resampling:
Xb_resampled shape (27265, 30)
yb_resampled shape (27265, 1)
number of 0 and 1: [20973  6292]
```

Train,test

```
X_train_b: (20448, 30)
X_test_b: (6817, 30)
y_train_b: (20448, 1)
y_test_b: (6817, 1)
number of ceros yb_resampled: 20973
number of ones yb_resampled: 6292
```

Validation

```
0    38881
1      713
Name: got_loan, dtype: int64
```

Normalized

```
Dataset after resampling:
Xb_resampled shape (27265, 30)
yb_resampled shape (27265, 1)
number of 0 and 1: [20973  6292]
```

Train,test

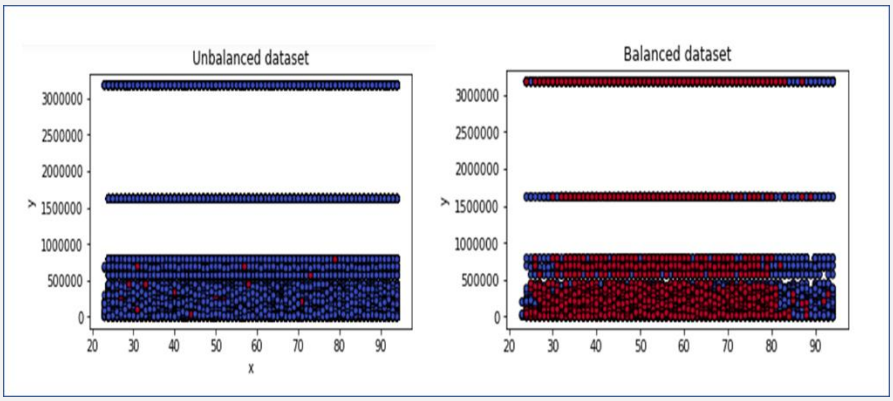
```
X_train_nb: (20448, 30)
X_test_nb: (6817, 30)
y_train_nb: (20448, 1)
y_test_nb: (6817, 1)
number of ceros ynb_resampled: 20973
number of ones ynb_resampled: 6292
```

Validation

```
0    38881
1      713
Name: got_loan, dtype: int64
```

5.2 Visualization unbalanced and balanced dataset

- Converted to an array with numpy.
- Converted to float/int to be able to use parameters like cmap when plotted.
- marker='o'.c=y_arr_list, to higlight the dependent variable, points size s=25,
- edgecolor='k', edge color of the marker,cmap=plt.cm.coolwarm



Train,Test,Validation

- Hold out 10% for validation
- ‘sample’ used to make sure is random
- No replacement
- Print the results

```
90% of clients
non normalized 1: (6292, 31)
non normalized 0: (350051, 31)
normalized 1: (6292, 31)
normalized 0: (350051, 31)
```

```
10% of clients
non normalized val 1 (713, 31)
non normalized val 0: (38881, 31)
normalized val 1: (713, 31)
normalized val 0: (38881, 31)
```

6. Model

- 5 models were used to decide the one which better adjusted to the binary classification analysis (Logistic Regression[4], Decision Tree Classifier[5], Random Forest Classifier[6], XGBoost[7], Stack ensemble model[8]).
- All models were run with the 4 type of pre-processed datasets.
- The approaches were the same for all 4:
  - Creating the model.
  - Feeding the training data into the model to fit.
  - Tuning its internal parameters to maximize predictions.
  - Testing.
  - Checking performance with the following evaluation matrix: confusion_matrix, acc score, ROC curve. For the ROC curve some transformations were required to avoid scalar issues.
  - Getting a classification report for lts for all the models and iterations. ROC results have been prioritized as well as hyperparameters built to reach maximum of true positives.
  - Using the model for the validation dataset(10%) with the same evaluation matrix for validation purposes.

Metrics:

Models		Logistic regression	Decision tree Classifier	Random Forest	XGBoost
Non normalized balanced	Test	F1-score(0): 0.76 F1-score(1): 0.39 AUC Score: 0.65 ROC: 0.63	F1-score(0): 0.90 F1-score(1): 0.73 AUC Score: 0.85 ROC: 0.89	F1-score(0): 0.94 F1-score(1): 0.79 AUC Score: 0.90 ROC: 0.96	F1-score(0): 0.94 F1-score(1): 0.79 AUC Score: 0.90 ROC: 0.96
	Validation	F1-score(0): 0.83 F1-score(1): 0.06 AUC Score: 0.71 ROC: 0.64	F1-score(0): 0.93 F1-score(1): 0.18 AUC Score: 0.86 ROC: 0.90	F1-score(0): 0.97 F1-score(1): 0.33 AUC Score: 0.94 ROC: 0.96	F1-score(0): 0.98 F1-score(1): 0.37 AUC Score: 0.96 ROC: 0.97
Normalized balanced	Test	F1-score(0): 0.85 F1-score(1): 0.64 AUC Score: 0.79 ROC: 0.89	F1-score(0): 0.91 F1-score(1): 0.75 AUC Score: 0.87 ROC: 0.90	F1-score(0): 0.94 F1-score(1): 0.78 AUC Score: 0.91 ROC: 0.88	F1-score(0): 0.94 F1-score(1): 0.79 AUC Score: 0.79 ROC: 0.97
	Validation	F1-score(0): 0.89 F1-score(1): 0.12 AUC Score: 0.80 ROC: 0.88	F1-score(0): 0.93 F1-score(1): 0.19 AUC Score: 0.87 ROC: 0.90	F1-score(0): 0.94 F1-score(1): 0.37 AUC Score: 0.95 ROC: 0.86	F1-score(0): 0.98 F1-score(1): 0.39 AUC Score: 0.96 ROC: 0.97

Stacking ensemble method

The Stacking ensemble technique was used to increase the predictive force of the classifier. The stability of the final model is increased. Even when they are individually give back bad results.

<<The main principle behind ensemble modelling is to group weak learners together to form one strong learner>>Robert R.F. DeFilippi

After importing the libraries required, the list of classifier models previoully used were included.

Metrics:

Models		Stacked model
Non normalized balanced	Test	F1-score(0): 0.94 F1-score(1): 0.78 AUC Score: 0.90 ROC: 0.92
	Validation	F1-score(0):0.97 F1-score(1): 0.32 AUC Score: 0.94 ROC: 0.85
Normalized balanced	Test	F1-score(0): 0.94 F1-score(1): 0.79 AUC Score: 0.90 ROC: 0.89
	Validation	F1-score(0): 0.97 F1-score(1): 0.33 AUC Score: 0.94 ROC: 0.90

ie: Balanced test & validation results (Stacking ensemble model)

Stacking roc-auc

```
3-fold cross validation:

roc_auc: 0.89 (+/- 0.00) [lg]
roc_auc: 0.90 (+/- 0.00) [dt]
roc_auc: 0.96 (+/- 0.00) [rf]
roc_auc: 0.88 (+/- 0.03) [StackingClassifier]
```

Stacking testing

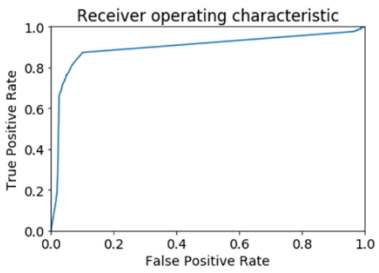
```
=== Confusion Matrix ===
[[4968  256]
 [ 381 1212]]
=== classification_report ===
              precision    recall  f1-score   support

         0.0       0.93      0.95      0.94      5224
         1.0       0.83      0.76      0.79      1593

    accuracy                           0.91      6817
   macro avg       0.88      0.86      0.87      6817
weighted avg       0.90      0.91      0.91      6817

=== accuracy_score ===
0.9065571365703389
```

```
=== roc_curve ===
0.8966439120616708
```



Stacking validation

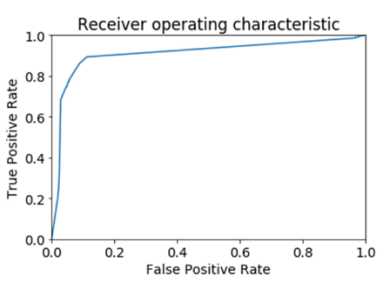
```
=== classification_report ===
              precision    recall  f1-score   support

         0.0       1.00      0.95      0.97     38881
         1.0       0.21      0.77      0.33       713

    accuracy                           0.94     39594
   macro avg       0.60      0.86      0.65     39594
weighted avg       0.98      0.94      0.96     39594

=== accuracy_score ===
0.9427438500782946
```

```
=== roc_curve ===
0.9090131094796281
```



Insights:

- The classification report for unbalanced dataset showed overfitting on the value 0. In this particular case, balancing data improved the results.
- For fine tuning the models, 'randint' was used. The 'randint' module would assist to initialize random values for the hyper parameters specified.
- For simplification purposes, only balanced results were shown. All the results can be found in previous iterations of the notebook.
- "The problem is that a single metric, such as classification accuracy, is an incomplete description of most real-world tasks." (Doshi-Velez and Kim 2017) *christophm.github.io*
- For this particular exercise, the ROC has been prioratized to reach the maximum amount of real potential clients (this illustrates how accurate the model is to detect true positives by ploting true positives againts false positives).

[4][5]Logistic regression: : https://christophm.github.io/interpretable-ml-book/storytime.html

[6]Random forest: https://jakevdp.github.io/PythonDataScienceHandbook/05.08-random-forests.html

[7]XGBoost: https://towardsdatascience.com/xgboost-python-example-42777d01001e

[8]Stacking ensemble: https://medium.com/@rrfd/boosting-bagging-and-stacking-ensemble-methods-with-sklearn-and-mlens-a455c0c982de

Although the initial information included in the dataset is real, in order to present this work and to comply with current regulations, all data has been anonymized.