1. Project Summary

This is a Machine Learning project which is meant to predict whether an applicant is a good or a bad client. Models will be built using the personal information of customers and their historical tendencies to pay back loans. Predictions will be made using labels that are not found in any of the datasets.

There is flexibility there as we are able to use any formula to generate corresponding binary credit status. Credit scores of either 0 or 1 will be generated and used as target labels.

# File system

.gitignore

decision\_tree.ipynb

logistic\_regression.ipynb

neural\_network.ipynb

pre\_processing.ipynb

random\_forest.ipynb

README.md

+---Output

credit\_decision\_tree\_optimized.pdf

credit\_decision\_tree\_optimized.png

credit\_decision\_tree\_preoptimized.pdf

credit\_decision\_tree\_preoptimized.png

full\_data.csv

pre\_encoded.csv

+---Resources

application\_record.csv

credit\_record.csv

# Preprocessing

## Datasource

Two .csv datasets (application\_record.csv and credit\_record.csv) were used for the project. They retrieved from Kaggle via the url:

<https://www.kaggle.com/datasets/rikdifos/credit-card-approval->prediction/download?datasetVersionNumber=3

### Application Record

Contains application records of customers and their personal information to be considered as the features to enable prediction. There is an ID column that enables merging with credit\_record.csv.

### Credit record

It contains previous credit records of customers in “application\_record.csv”. It contains previous monthly loan repayment records for customers. There is an ID column that enables merging with application\_record.csv.

The two datasets will be subsequently merged enabling us to extract features and target labels for model building.

## Limitations of Datasets

* One major challenge of these two data sets is that there is no distinctive target column for either “Good Credit” or “Bad Credit”. To solve this issue, Binary values will be deduced using the credit record dataset.
* Also, the data is heavily unbalanced potentially adding to the need to apply optimization if the desired accuracy is not achieved.

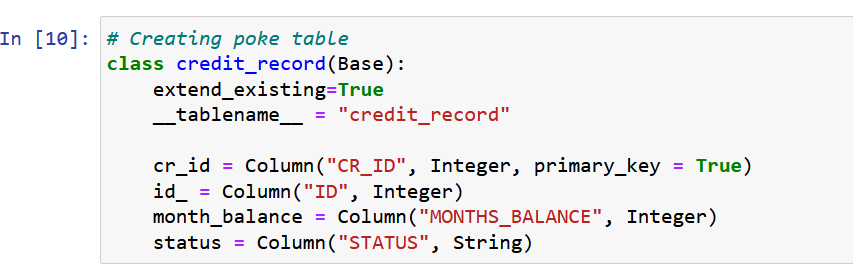
## Data Base Engineering

Data is extracted and loaded in their raw form into the database for scalability purposes which will allow for prospective applications and credit records to be added in the future.

Tools:

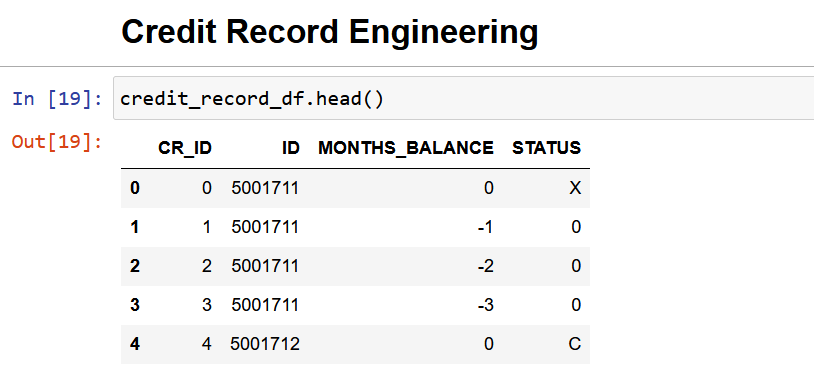
* PostgreSQL





## Credit Record Engineering

The relevant columns in the credit record Table are the “ID”, “MONTHS\_BALANCE”, and “STATUS” columns.



The dataset has a “STATUS” column which contains values with these explanations.

0: 1-29 days past due

1: 30-59 days past due

2: 60-89 days overdue

3: 90-119 days overdue

4: 120-149 days overdue

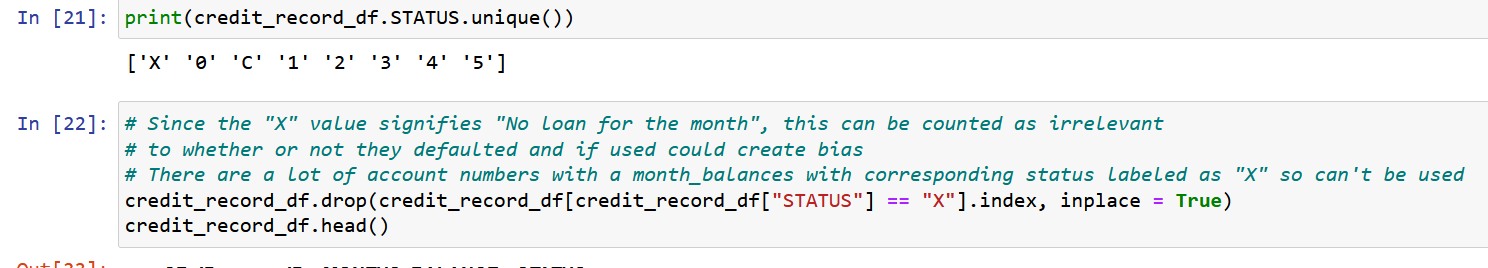
5: Overdue or bad debts, write-offs for more than 150 days

C: paid off that month

X: No loan for the month

### Converting STATUS values To Target Labels

We dropped all STATUS columns with “X” values as we couldn’t conclude whether a customer with no loan for a particular month should be considered as a customer with good credit or bad credit for that month.



It became also clear that our target label values could be deduced using the following formula.

Good Credit = 0

Bad Credit = 1

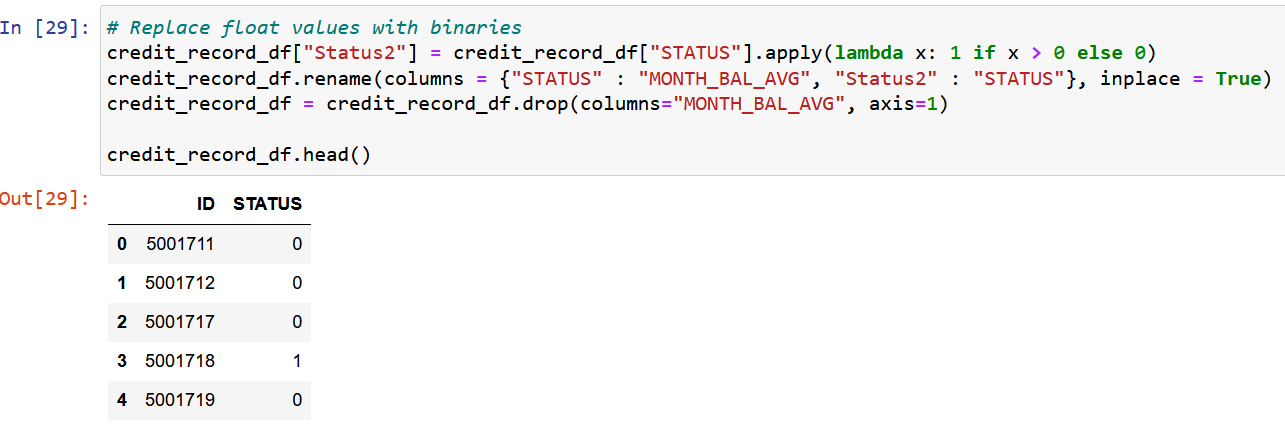


C: paid off that month = Good Credit = 0

0: 1-29 days past due = Good Credit = 0

All other STATUS values will be replaced with Bad Credit = 1.

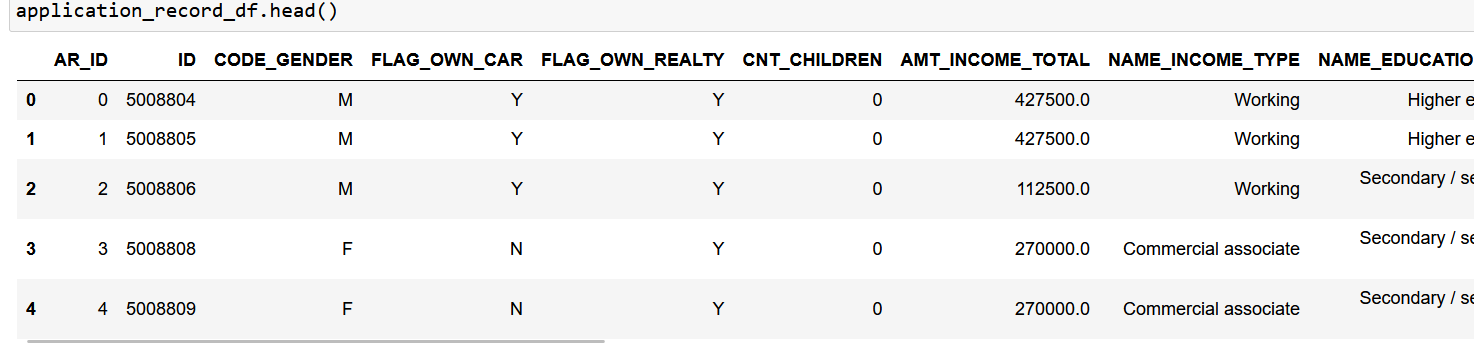
The averages of all the monthly statuses of a customer were calculated and any status value of more than 0 was classed as 1 (Bad credit), whiles all 0 values were classed as 0 (Good credit). This way all float values will also be replaced with binary values.



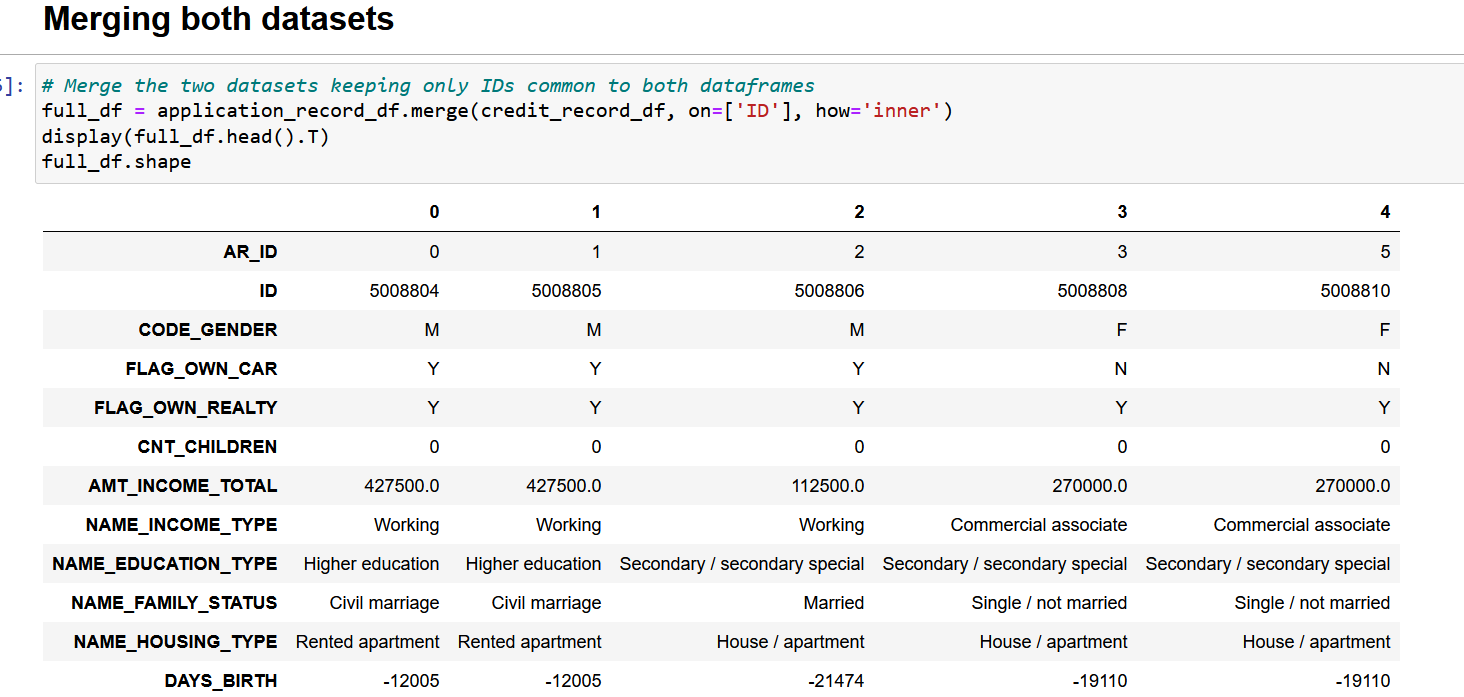
## Application record data Engineering

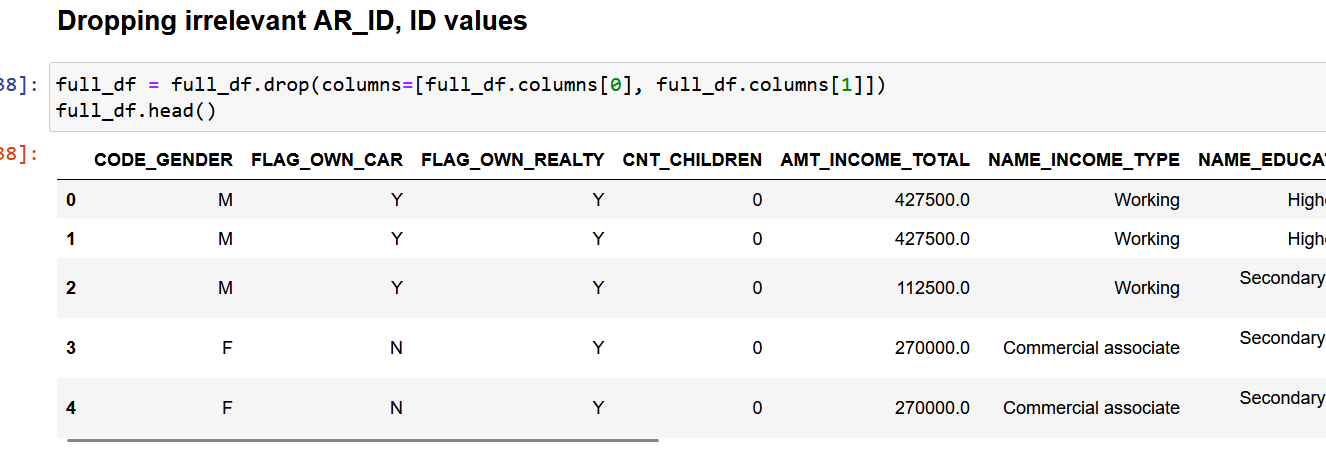
The relevant columns in the credit record Table are:

'CODE\_GENDER', 'FLAG\_OWN\_CAR', 'FLAG\_OWN\_REALTY', 'CNT\_CHILDREN', 'AMT\_INCOME\_TOTAL', 'NAME\_INCOME\_TYPE', 'NAME\_EDUCATION\_TYPE', 'NAME\_FAMILY\_STATUS', 'NAME\_HOUSING\_TYPE', 'DAYS\_BIRTH', 'DAYS\_EMPLOYED', 'FLAG\_MOBIL', 'FLAG\_WORK\_PHONE', 'FLAG\_PHONE', 'FLAG\_EMAIL', 'OCCUPATION\_TYPE' and 'CNT\_FAM\_MEMBERS' columns.



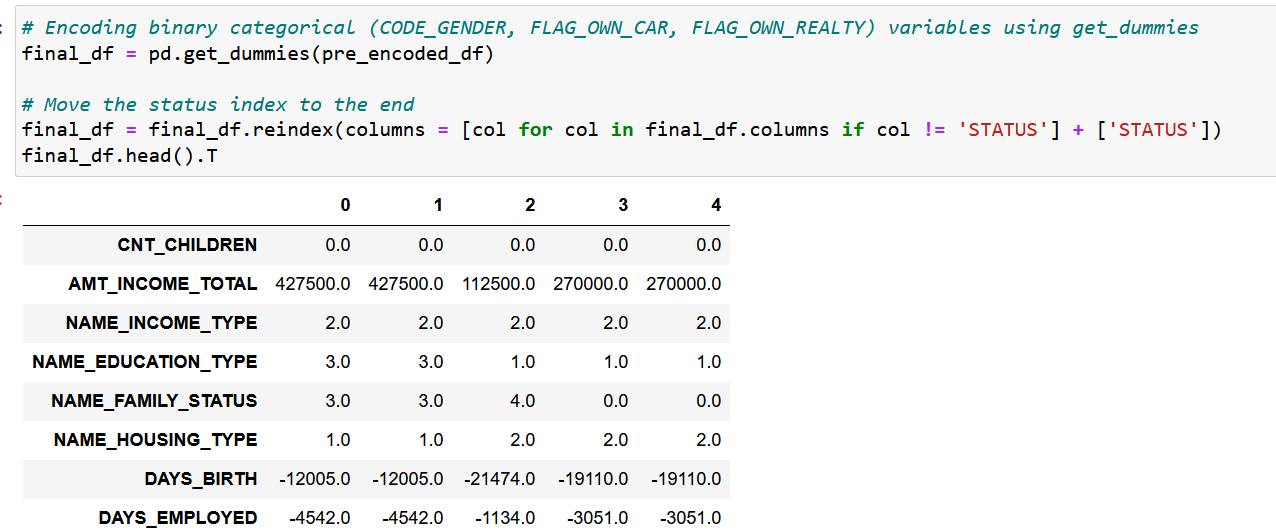
### Merging both datasets and dropping irrelevant columns

Both data sets were merged and the irrelevant columns were dropped finally.

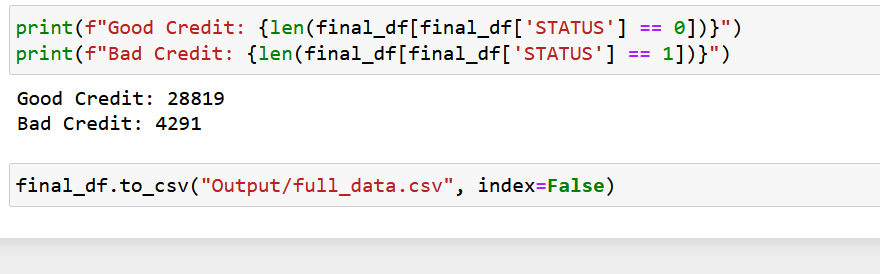


### Encoding Categorical columns





### Exporting final dataframe to a .csv file for use in model engineering



# Model Engineering

## Supervised Learning

Since we now have target labels, we will be building supervised learning labels.

### Classification Method

### Binary classification

There are two class labels (Good Credit “0” and Bad Credit “1”) for our classification.

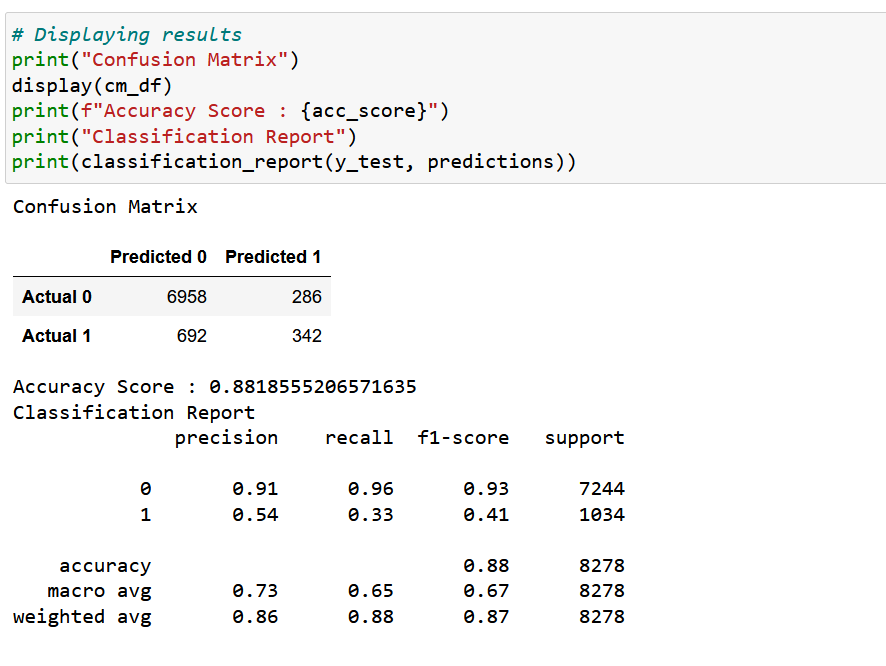
### Model selection

We are considering 4 Models for our project utilizing a mix of different optimization methods to enhance our efficiency.

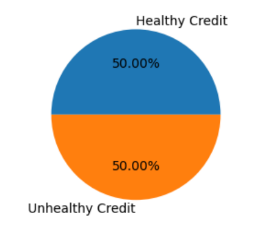
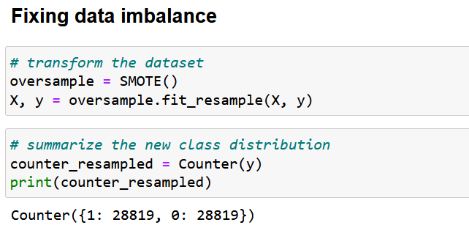
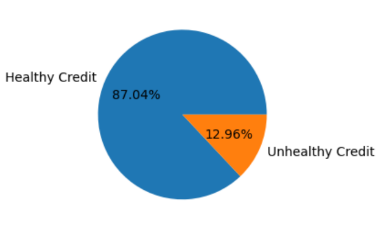
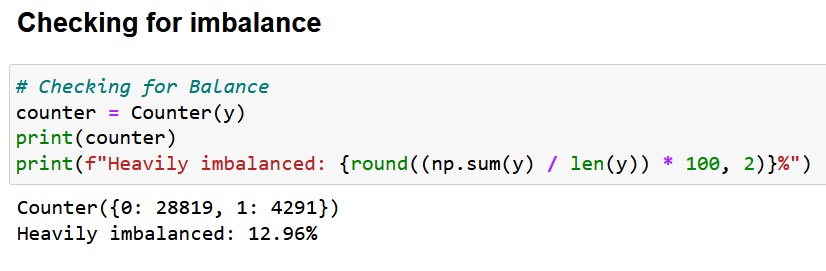
1. Random Forest
2. Decision Tree
3. Logistic Regression
4. Neural Network

#### Random Forest

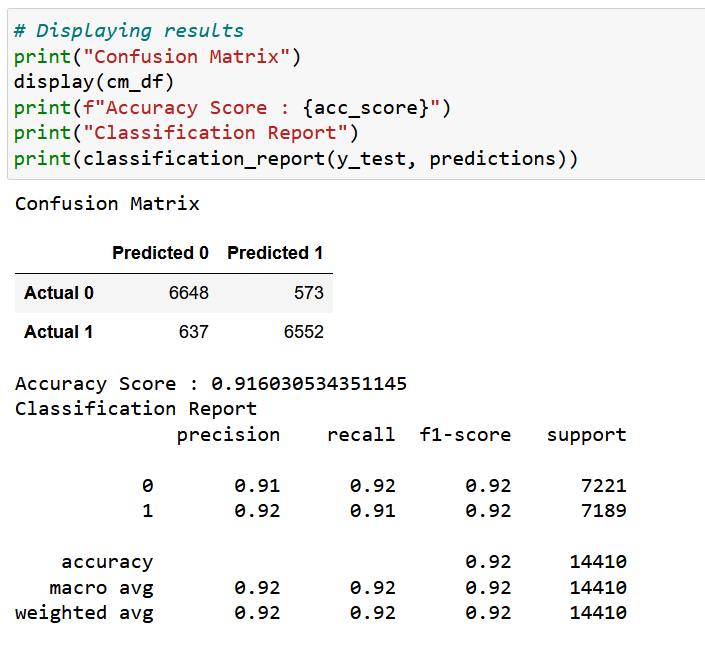
##### Training with the preoptimized or non-resampled dataset



##### Training with the optimized or non-resampled dataset

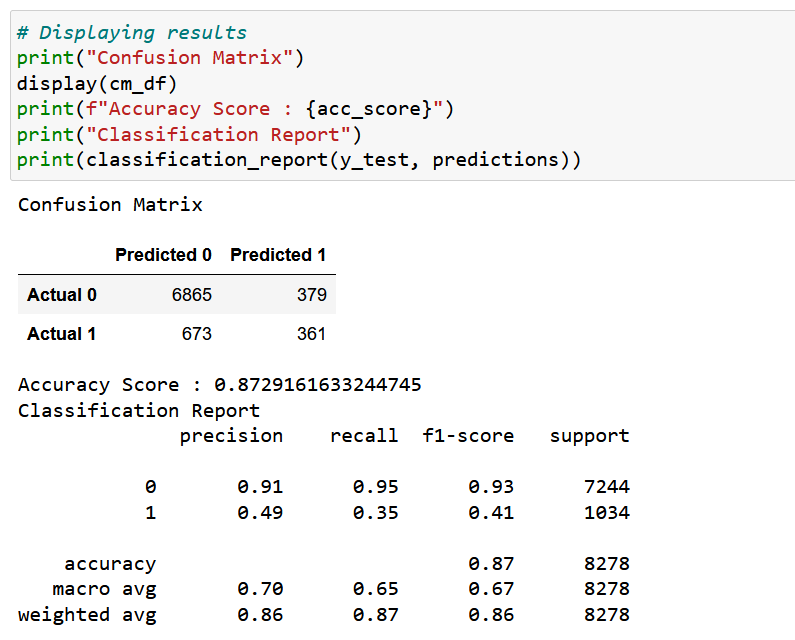


After Resampling, the accuracy score is **0.92** which is an improvement over the **0.88** for preoptimization

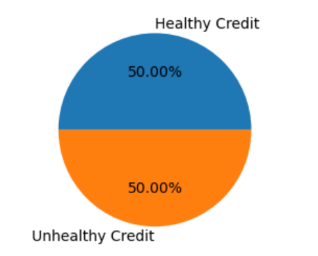
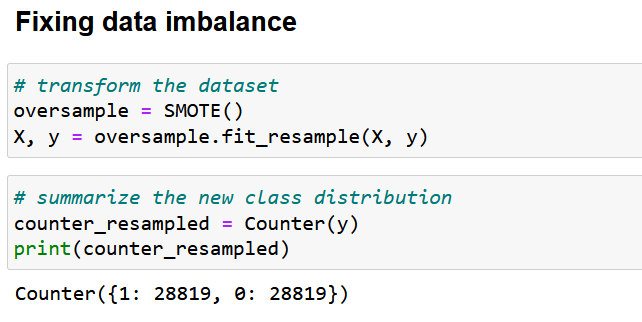
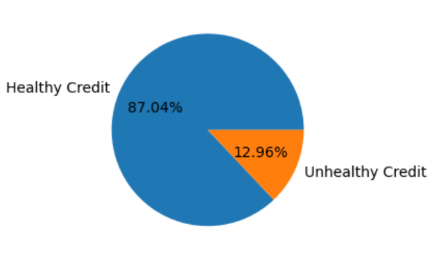
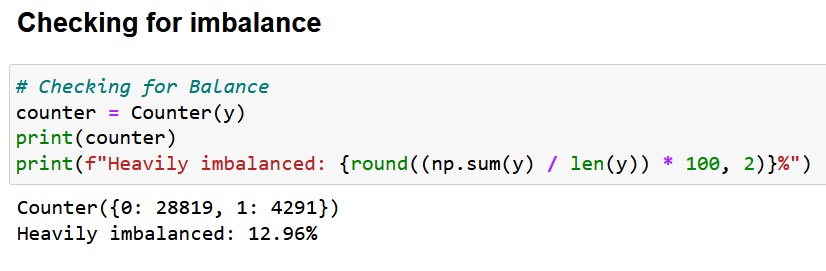


#### Decision Tree

##### Training with the preoptimized or non-resampled dataset



##### Training with the optimized or non-resampled dataset



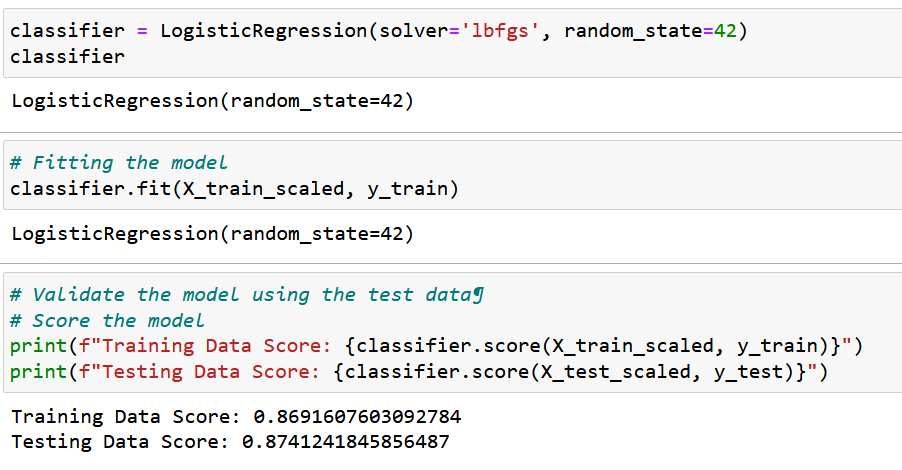
After Resampling, the accuracy score is **0.90** which is an improvement over the **0.87** for preoptimization



#### Logistic Regression

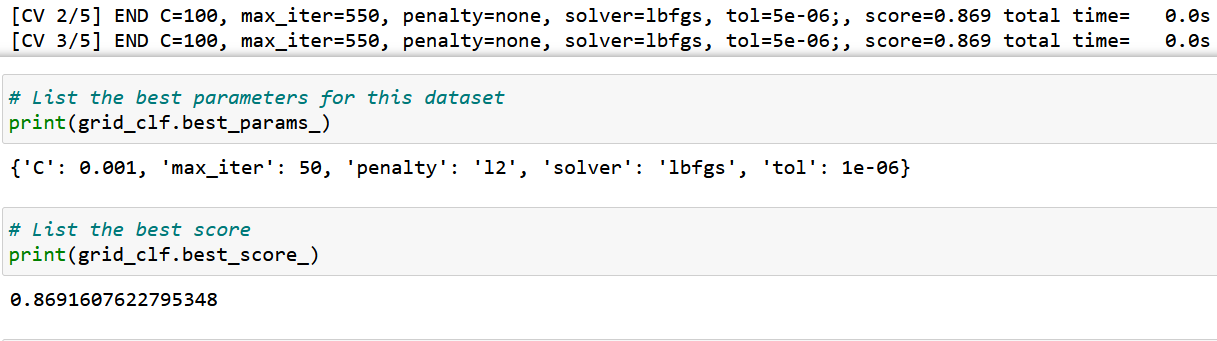
##### Training with no Optimization

Preoptimization parameters



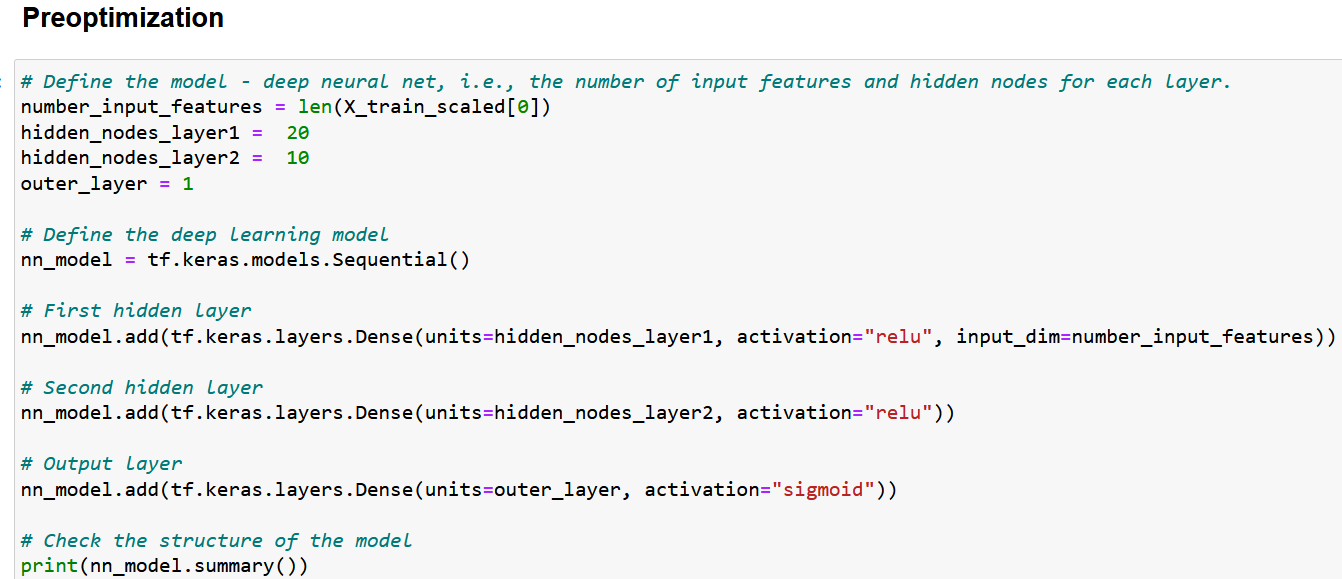
##### Auto Optimization using Hyperparameter tuning[¶](http://localhost:8888/notebooks/neural_network.ipynb#Auto-Optimization-using-Hyperparameter-tuning)

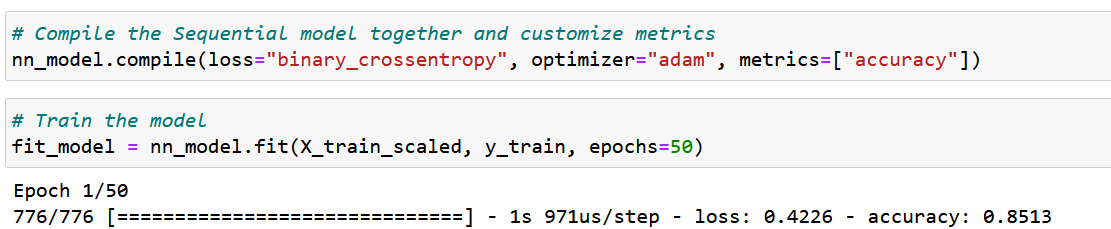


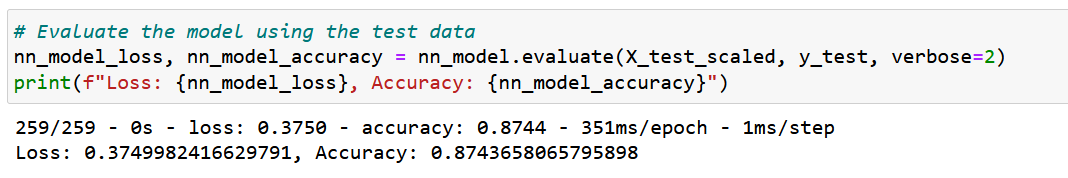


#### Neural Network

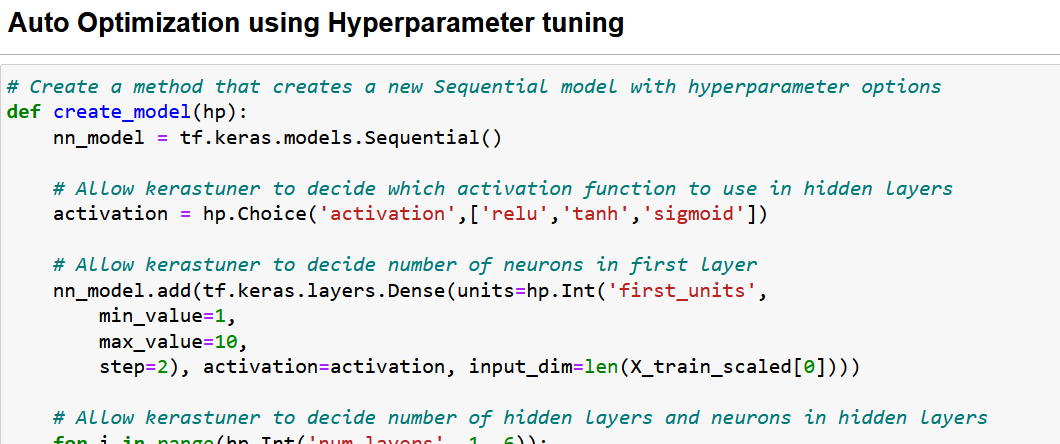
##### Training with no Optimization

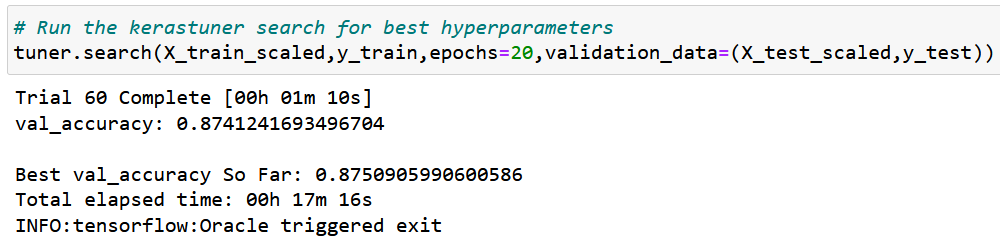
Preoptimization parameters

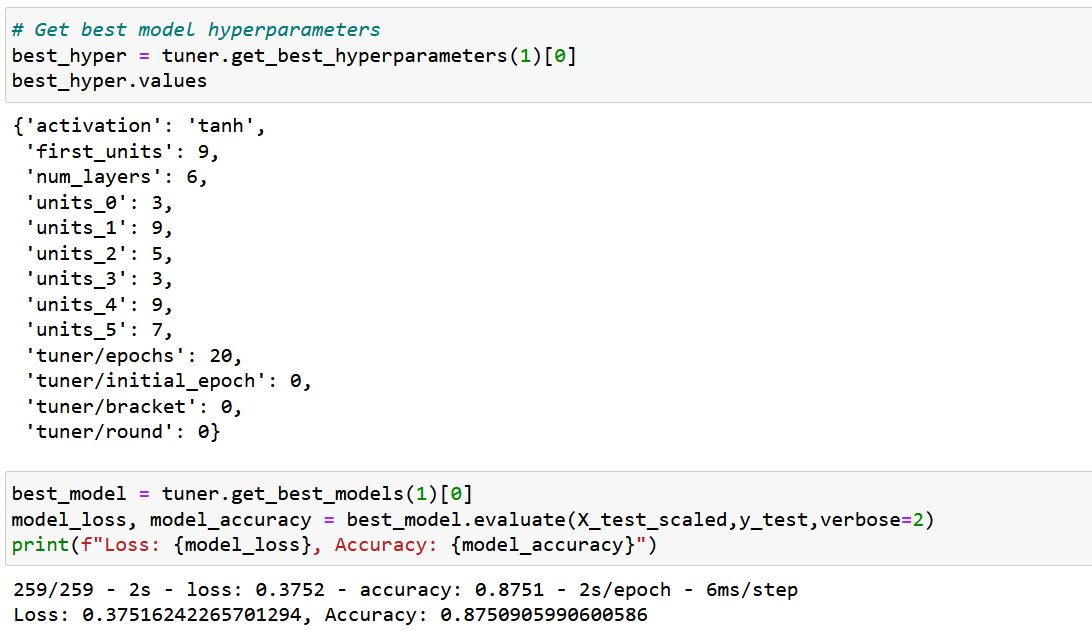




##### Auto Optimization using Hyperparameter tuning[¶](http://localhost:8888/notebooks/neural_network.ipynb#Auto-Optimization-using-Hyperparameter-tuning)







# Summary