Thomas Koutsidis

In [1]:
```python
import pandas as pd
import numpy as np
```

In [2]:
```python
df = pd.read_csv('model.csv')
df_v = pd.read_csv('val.csv')
```

In [3]:
```python
# Cleaning and preparing our df for model.
# There are no NaN values, just 0s. I replaced the 0s with the mean of the column.
# I did not clean the column "default" because that column will be for predictions.

for column in df.columns:
    if column != "default":
        df[column].replace(0, df[column].mean(), inplace = True)
```

In [4]:
```python
# Cleaning and preparing our df for val.
# There are no NaN values, just 0s. I replaced the 0s with the mean of the column.
# I did not clean the column "default" because that column will be for predictions.

for column in df_v.columns:
    if column != "default":
        df_v[column].replace(0, df_v[column].mean(), inplace = True)
```

In [5]:
```python
# K Fold Cross Validation is a good way to avoid data leakage.
# I will perform K Fold on the datasets.

from sklearn.model_selection import KFold

kf = KFold(n_splits = 10, shuffle = True)

# Resources used:
# https://towardsdatascience.com/what-is-data-leakage-and-how-can-it-be-avoided-in-machine-learning-eb435a27c3e3
# https://www.statology.org/k-fold-cross-validation-in-python/
```

In [6]:
```python
# Below is a logistic regression.

from sklearn.linear_model import LogisticRegression
from sklearn.metrics import roc_auc_score

auc_scores = []

for train_index, test_index in kf.split(df):
    X_train, X_test = df.iloc[train_index], df.iloc[test_index]
    y_train, y_test = df["default"].iloc[train_index], df["default"].iloc[test_index]

    model = LogisticRegression(solver = 'liblinear')
    model.fit(X_train, y_train)

    predictions = model.predict_proba(X_test)
    auc_scores.append(roc_auc_score(y_test, predictions[:, 1]))

# Resources used:
# https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_auc_score.html
# https://www.statology.org/auc-in-python/
# https://stackoverflow.com/questions/61184906/difference-between-predict-vs-predict-proba-in-scikit-learn
# https://www.kaggle.com/discussions/questions-and-answers/155495
# https://towardsdatascience.com/dont-sweat-the-solver-stuff-aea7cddc3451
```

In [7]:
```python
print("Scores:", auc_scores)
mean_auc_score = np.mean(auc_scores)
print("Mean:", mean_auc_score)
```

```
Scores: [0.8271581995375672, 0.8093723263591819, 0.8155871044069801, 0.871156899620666, 0.8062554575287106, 0.8342679712513613, 0.8180831092883073, 0.8200026542913899, 0.8250598136204239, 0.8009913292266697]
Mean: 0.8227934865131257
```

In [8]:
```python
# Below is a decision tree regression.
# Decision tree regression is my second method, in addition to logistic regression.

from sklearn.tree import DecisionTreeClassifier

auc_scores2 = []

for train_index, test_index in kf.split(df):
    X_train, X_test = df.iloc[train_index], df.iloc[test_index]
    y_train, y_test = df["default"].iloc[train_index], df["default"].iloc[test_index]

    model_decision_tree = DecisionTreeClassifier()
    model_decision_tree.fit(X_train, y_train)

    predictions_decision_tree = model_decision_tree.predict_proba(X_test)
    auc_scores2.append(roc_auc_score(y_test, predictions_decision_tree[:, 1]))

# https://stackoverflow.com/questions/61184906/difference-between-predict-vs-predict-proba-in-scikit-learn
# https://scikit-learn.org/stable/auto_examples/tree/plot_tree_regression.html
```

In [9]:
```python
print("Scores:", auc_scores2)
mean_auc_score2 = np.mean(auc_scores2)
print("Mean:", mean_auc_score2)
```

```
Scores: [1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0]
Mean: 1.0
```

In [10]:
```python
predictions_logistic_regression = model.predict_proba(df_v)
print("Logistic Regression Predictions:", predictions_logistic_regression)
```

```
Logistic Regression Predictions: [[0.88935039 0.11064961]
 [0.9706202  0.0293798 ]
 [0.81951566 0.18048434]
 ...
 [0.9236294  0.0763706 ]
 [0.92568837 0.07431163]
 [0.93847746 0.06152254]]
```

In [11]:
```python
predictions_decision_regression = model_decision_tree.predict_proba(df_v)
print("Decision Tree Regression Predictions:", predictions_decision_regression)
```

```
Decision Tree Regression Predictions: [[0. 1.]
 [0. 1.]
 [0. 1.]
 ...
 [1. 0.]
 [1. 0.]
 [1. 0.]]
```

In [12]:
```python
# Saving results into a CSV file

predictions_logistic_regression = pd.DataFrame(predictions[:, 1])
predictions_logistic_regression.to_csv("results1.csv")
```

In [13]:
```python
# Saving results into a CSV file

predictions_decision_tree = pd.DataFrame(predictions_decision_tree[:, 1])
predictions_decision_tree.to_csv("results2.csv")
```