Dataset : Default of Credit Card Clients Dataset

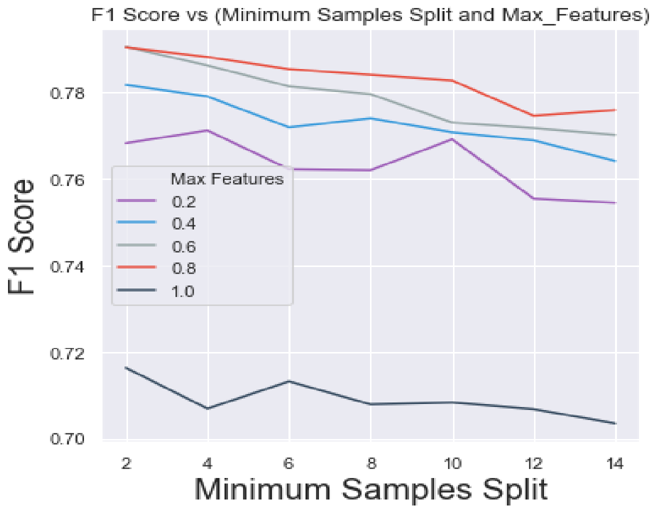
Classification Problem: This dataset contains information on default payments, demographic factors, credit data, history of payment, and bill statements of credit card. Problem to predict whether the client will default the payment next month. Class 0 (No Default): 23365 observations, Class 1 (Default): 6637 observations

Exploratory Data Analysis:

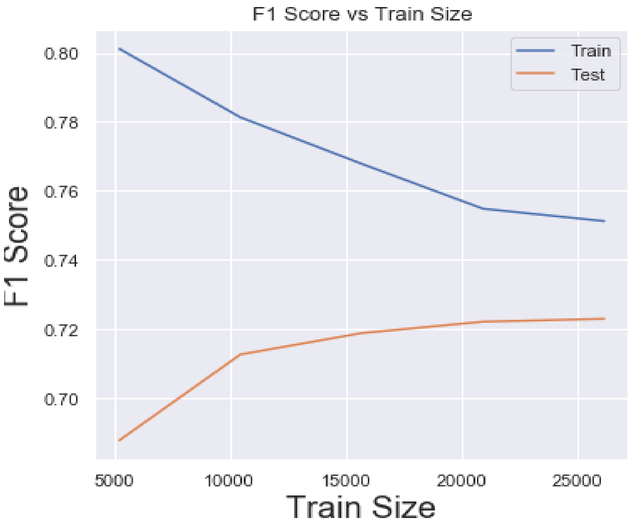
- It is a highly imbalanced dataset with Class 0 contributing to 78% of the observations and Class 1 with 22%
- Grouping the less frequently occuring class under Other for the features Education and Marriage.
- Creating Dummy variables for the categorical features Sex, Education and Marriage.
- Oversampling the minority class for DecisionTree and Boosted DecisionTree Algorithms and undersampling the majority class for Support Vector Machine Algorithm
- Due to imbalance of the classes in the dataset, F1 score is being considered as the metric for evaluation.

Decision Trees

Hyperparameter Tuning to find the Max_features and Minimum samples to be considered



Learning Curve using Cross Validation: Train Size Vs F1 score using the best parameters



Model implementation with cross validation to find the optimal hyperparameters:

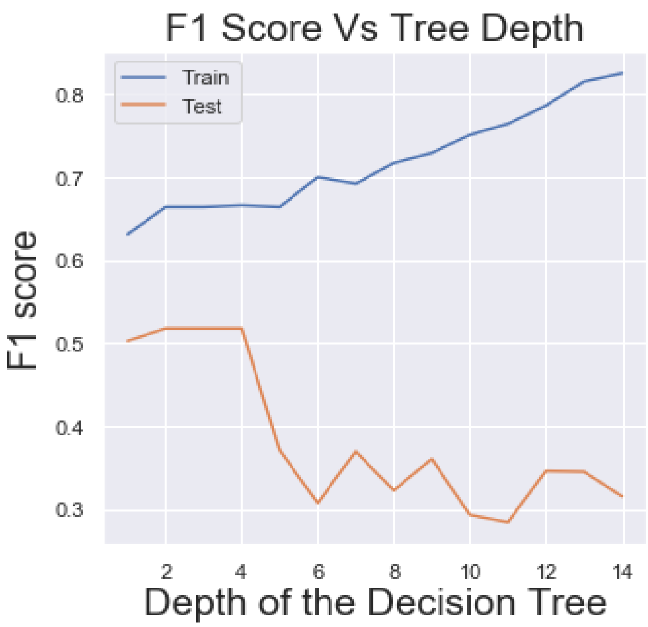
The plot helps us find the optimum values for the maximum features to be considered and minimum samples to be split.

Optimal max_features is 0.8. It is suprising to see that the accuracies are not good enough when max_features = 1 (i.e all the features are selected). 2 or 4 can be chosen for the minimum samples split.

Parameters: criteria – entropy, max_depth: 10, max_features: 0.8, min_samples_split: 2

As we see, the models perform well with both the training and the testing F1 scores converging with increasing training sizes. The model is overfitting when the training size is low. The model is not learning much from the data after 15000 data points.

F1 Score Vs Tree Depth

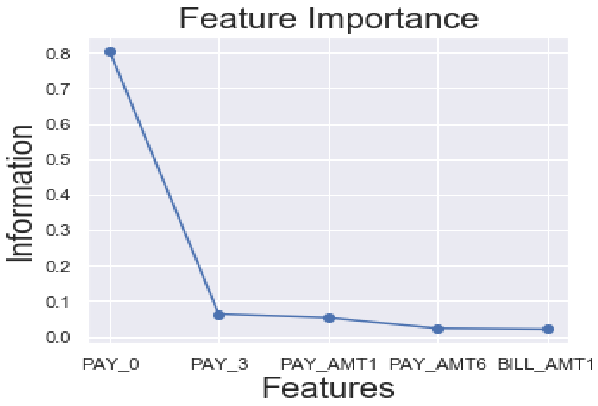


Parameters: criteria – entropy, max_features: 0.8, min_samples_split: 2

It is quite interesting to see that the test F1 score decreases after the tree reaches depth 4. We will have to prune the tree at depth 4. *This could be because the features are not contibuting any information towards classifying the target variable.*

Let’s plot the features importance plot to arrive at a conclusion.

Feature Importance

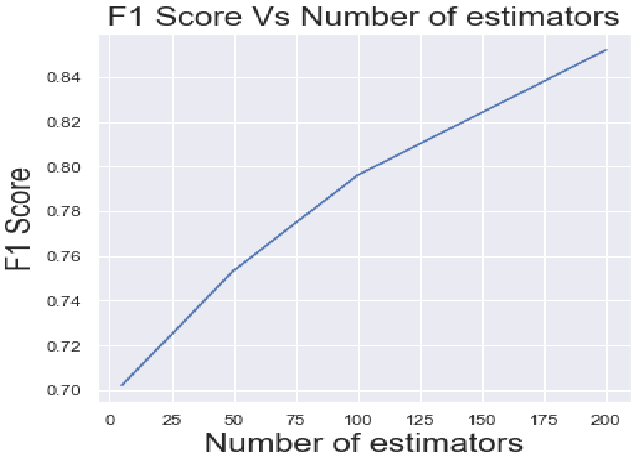


Feature **Pay_0** has the most information gain.

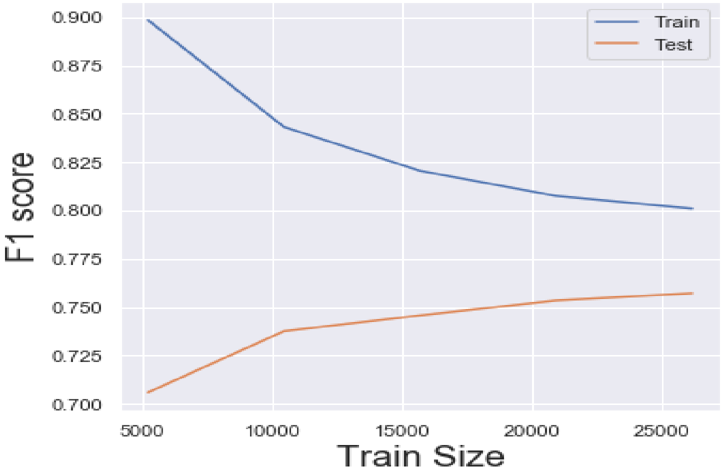
It is importance is close to 80% in the dataset. The other features have only negligible importance. This is the reason explaining why the F1 scores are decreasing after reaching the depth of 4.

Boosting

Hyperparameter tuning to find the Number of estimatores



Learning curve with cross validation: Training size Vs F1 Score using the best parameters

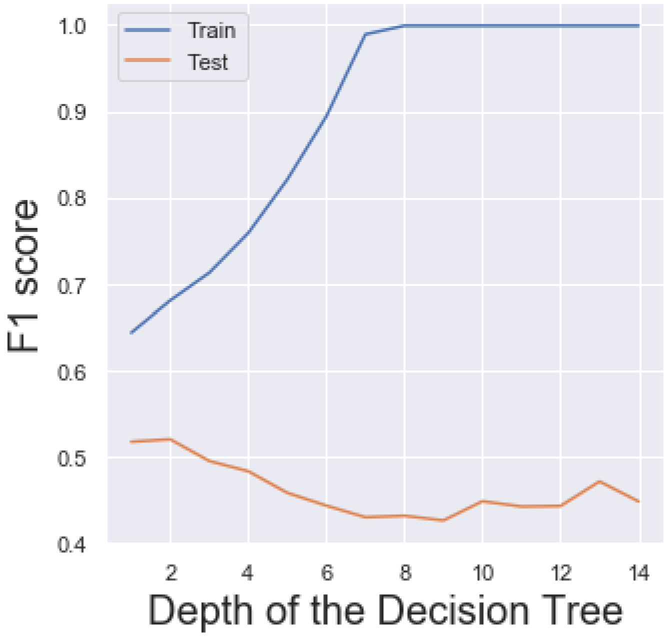


<u>Model implemented with Cross Validation on different number of estimators.</u>

The F1-score for the training data is increasing as the number of estimators or learners increases. We see that the saturation is not being reached even with 200 estimators. Due to this, we cannot decide on the optimal value for the number of estimators. Let’s use the default number of estimators, which is 50.

Parameters: Base- DecisionTree, Number of estimators is 50, max_depth: 4, max_features: 0.8

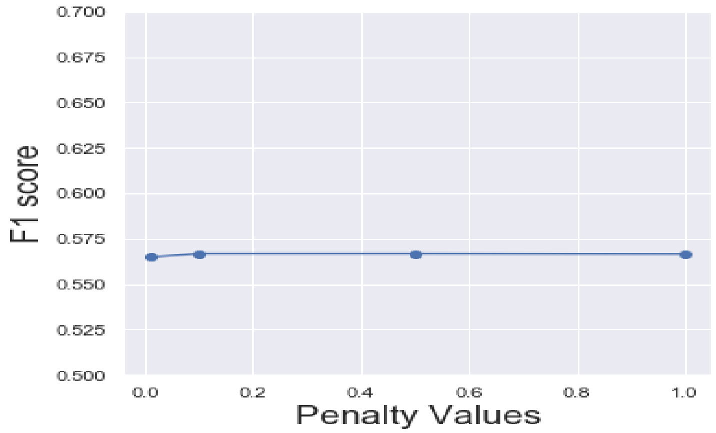
The oversampled data is generalising well as we the train and validation F1 scores converging with increasing train size. The model is neither overfitting nor underfitting.



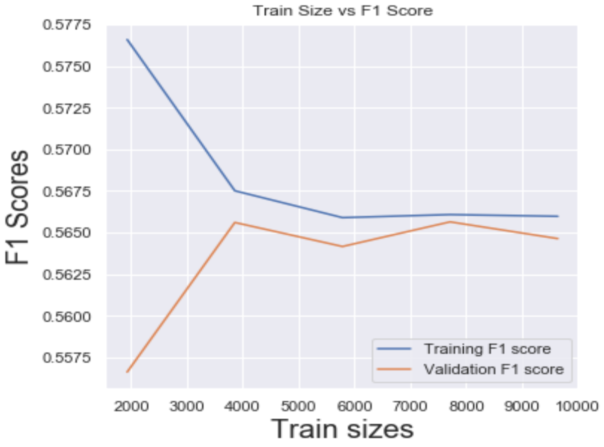
Although we see good performances with the training data, the model is not generalising well on the test as we see the F1 score for the test set is decreasing with increasing depths. Depth between 2 to 4 can be chosen. We should not get carried away by the training F1 scores as this is clear overfitting. Also, the training set has oversampled data of the minority class could be the reason for the model to perform well in the training data.

Support Vector Machine - Linear SVM

Hyperparameter Tuning to find the optimal penalty



Learning Curve with Cross Validation
Training sizes Vs F1 Score using best parametes



Model implemented with Cross Validation on different penalities.

C represents the severity of the penalty that is added to the support vectors.

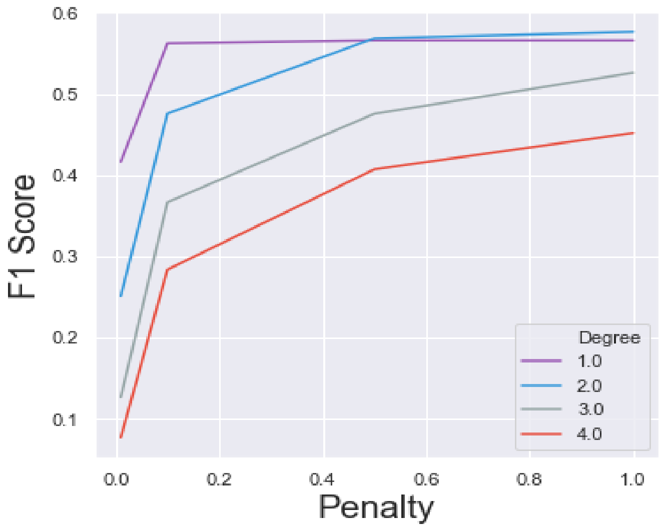
F1 score is the same with varying penalities from 0.1 to 1. Adding penalty does not have any affect on the accuracy with this dataset. So, the number of the points chosen as support vectors for varying penalities are the same.

Parameters: Penalty: 0.1, kernel: Linear

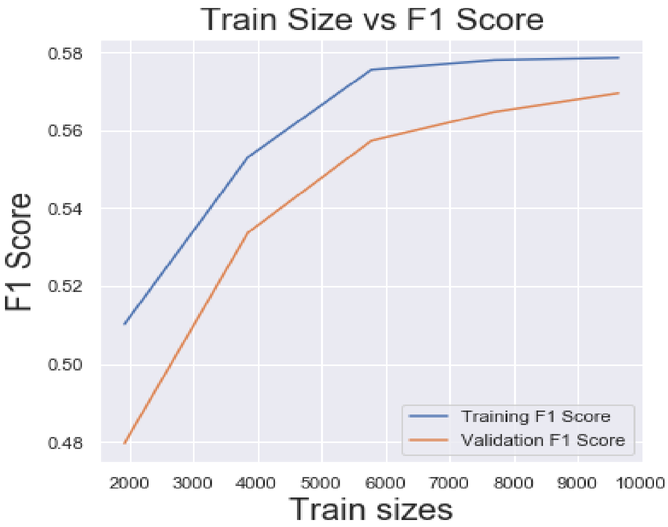
We see very small changes in F1 scores with varying training sizes. So, the train and validation scores are converging, the model is not learning much from the data points.

Support Vector Machine – Polynomial kernel

Hyperparameter tuning to find the best Degree and Penalty



Learning Curve with Cross Validation: Train size Vs F1 score using the Best parameters



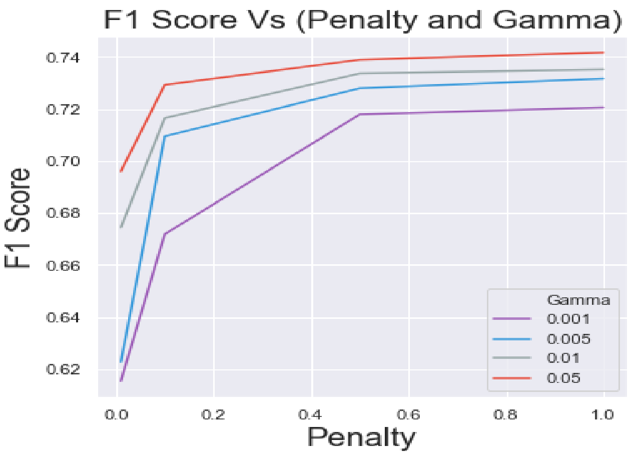
Model implemented with Cross Validation on different penalties and Degrees

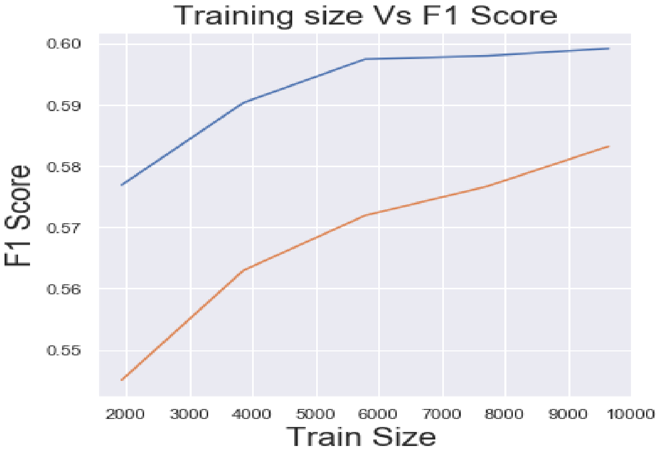
Degree allows for a more flexible decision boundary. **Optimal Penalty: 0.5, Optimal Degree: 1.** After this point, the model has attained saturation.

Model implemented with Cross Validation on different training sizes.

F1 scores of both the training and validation data are increasing and seem to be converging. Saturation is reached with 6000 training points. Although, the training F1 scores are increaing, it is not overfitting as we are performing cross validation.

Support Vector Machine – Radial Kernel





Model implemented with Cross Validation on different gamma's and penalities.

Optimal Gamma – 0.05, Optimal Penalty: 0.5

With the increase in Gamma, data points closer to the seperation line are used and vice versa.

Model implemented with Cross Validation on different training sizes.

F1 scores of both the training and validation data are increasing and seem to be converging. The training F1 score increases upto 6000 data points after which saturation is attained.

Test Result Predictions and evaluating the metrics of all algorithms

Kernel	Linear SVM	Polynomial SVM	Radial SVM	DecisionTree	Boosted DecisionTree
F1 Score	0.50	0.52	0.52	0.52	0.50

The F1 score against Default 1 is low (close to 0.50%) and is almost the same for all the algorithms.

As witnessed from the learning curves with training and validation F1 score, we had decent results on the data for each of the algorithms with the model seeming to be generalising well. But the results on the actual test dataset seem be totally different. This could be beacause of the reasons below

- The training data was oversampled and undersampled due to the high imbalance in the dataset. Due to this, the model was able to perform better on the training and validation data. But the model witnessed the actual test data, which was highly imbalance, it was not able to perform well. This clearly explains that the model was not able to generalise the data even after sampling.
- Insufficeint number of features as the features already present in the data did not help in determining the default class. (As seen from the decisiontree feature importance plot), only PAY_0 seemed to be contributing towards perdicting the target class).

How the Model Performance can be improved

- When more data with the minority class is available or more relatable features are present in the dataset.
- Performing some feature transformation may help improve the performance, athough it is not guaranteed. Feature transformation was tried on some of the features but it did not help in improving the model performance.

Other information

- I ran the SVM Radial alrogithms with only the repayment status variables (PAY_0 to Pay_6) and the test results were the same (i.e) no improvement in the test F1 score. This is clealy an indicator that the other features had no relation with the target variable.
- Implemented the algorithms after some data transformations like squaring the repayment status variables (PAY_0 to Pay_6) and the test F1 score resulted the same.
- The distrution of the default class for each feature was almost the same. For example, for each category in features marriage, sex, the default class was distributed in the ratio of 80:20 only.