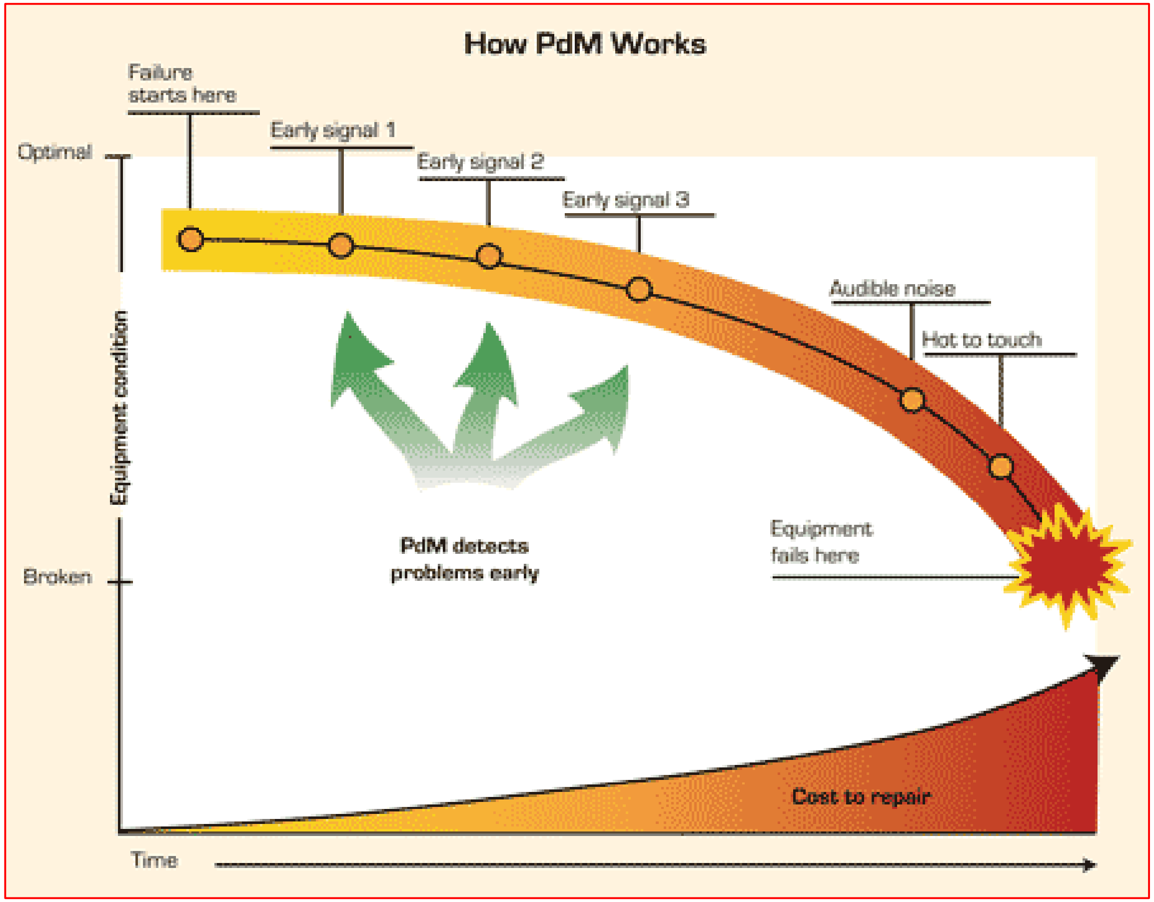
Predictive Maintenance : Classifier for Machine Failure

Machine Learning Foundation

By Chiranjit Pathak

Predictive Maintenance and its importance

As Industry 4.0 continues to generate media attention, many companies are struggling with the realities of AI implementation. Indeed, the benefits of predictive maintenance such as helping determine the condition of equipment and predicting when maintenance should be performed, are extremely strategic. Needless to say that the implementation of ML-based solutions can lead to major cost savings, higher predictability, and the increased availability of the systems.



In predictive maintenance scenarios, data is collected over time to monitor the state of equipment. The goal is to find patterns that can help predict and ultimately prevent failures.

Problem Statement

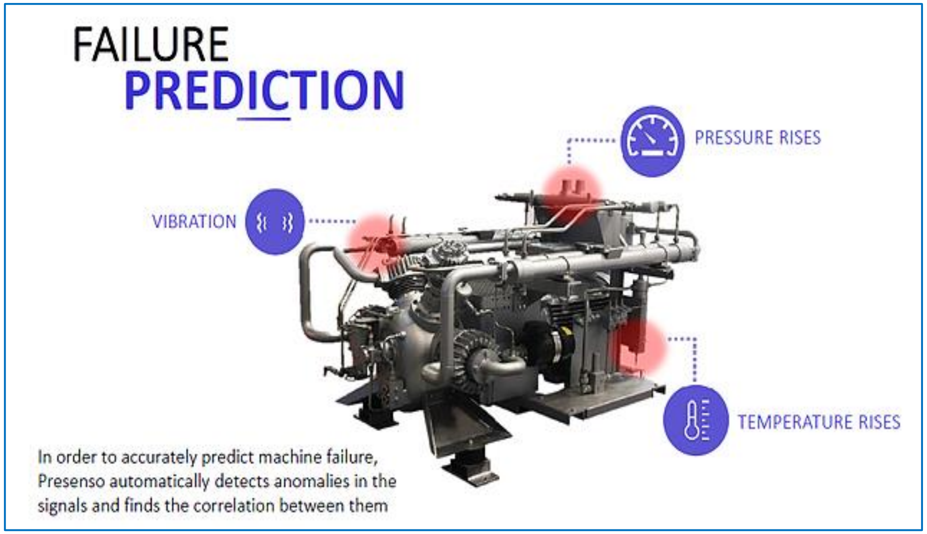
The goal is to predict failure of machine using different classification models.

Data loading and description

Data Source:
https://bigml.com/user/czuriaga/gallery/dataset/587d062d49c4a16936000810

The dataset comprises of **8,784 observations** of **28 columns**. Below is a table showing names of all the columns and their description.

Column Name	Description
Date	Date, time of recording data in 1 hr interval
Temperature	Temperature of atmosphere
Humidity	Humidity of atmosphere
Operator	Operator number
Measure 1 to 15	Parameters captured
Hours Since Previous Failure	hrs. from last failure
Failure	failure happened - Yes or No
Date.year,month,day,hrs,min,sec	date-time parameters in detail



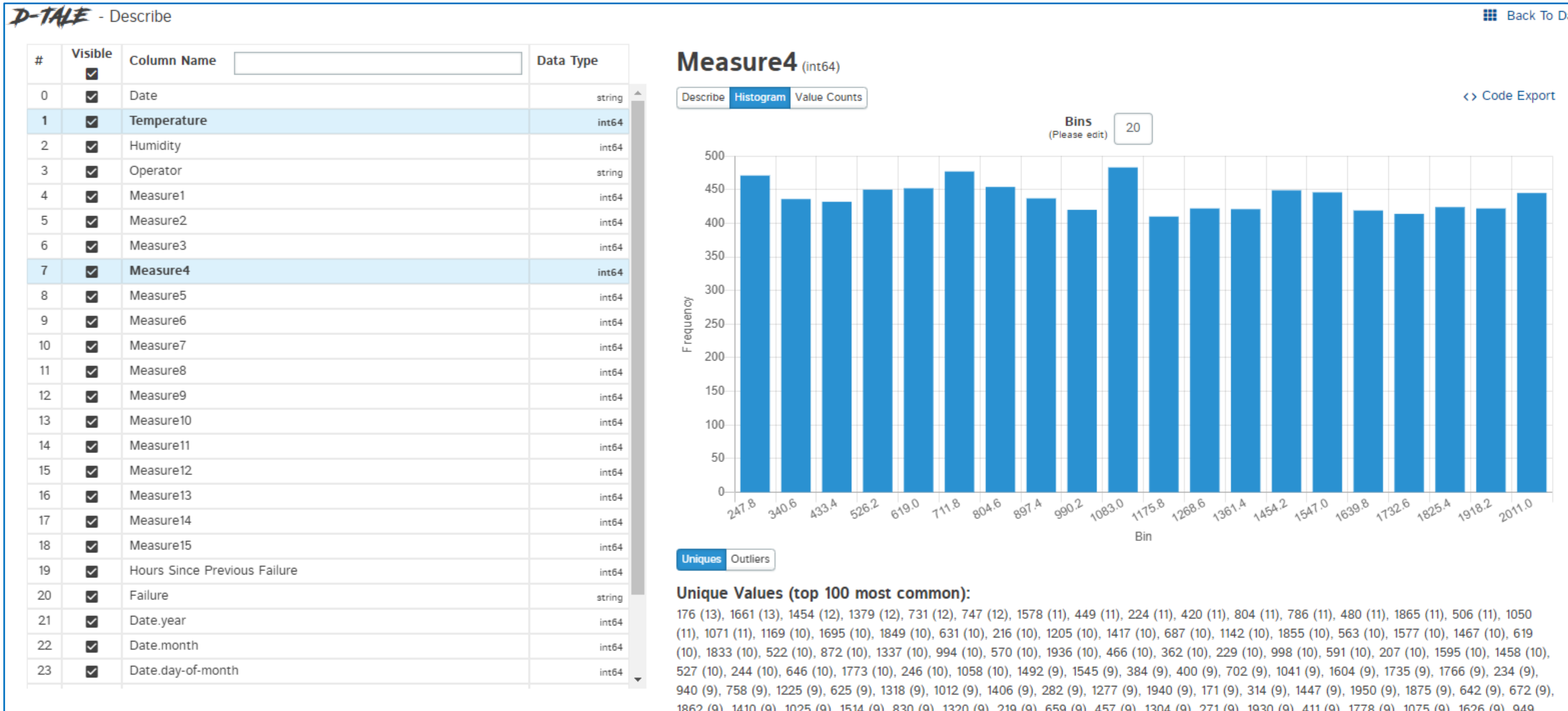
Based on the above collected data, EDA and ML model has been exercised in order to find an optimum Classification model which will be used for predicting the future machine failure. These has been accomplished using Numpy, Pandas, Seaborn, Matplotlib, D-Tale and Sklearn in Python

Exploratory Data Analysis

Processing, Profiling and Encoding categorical features

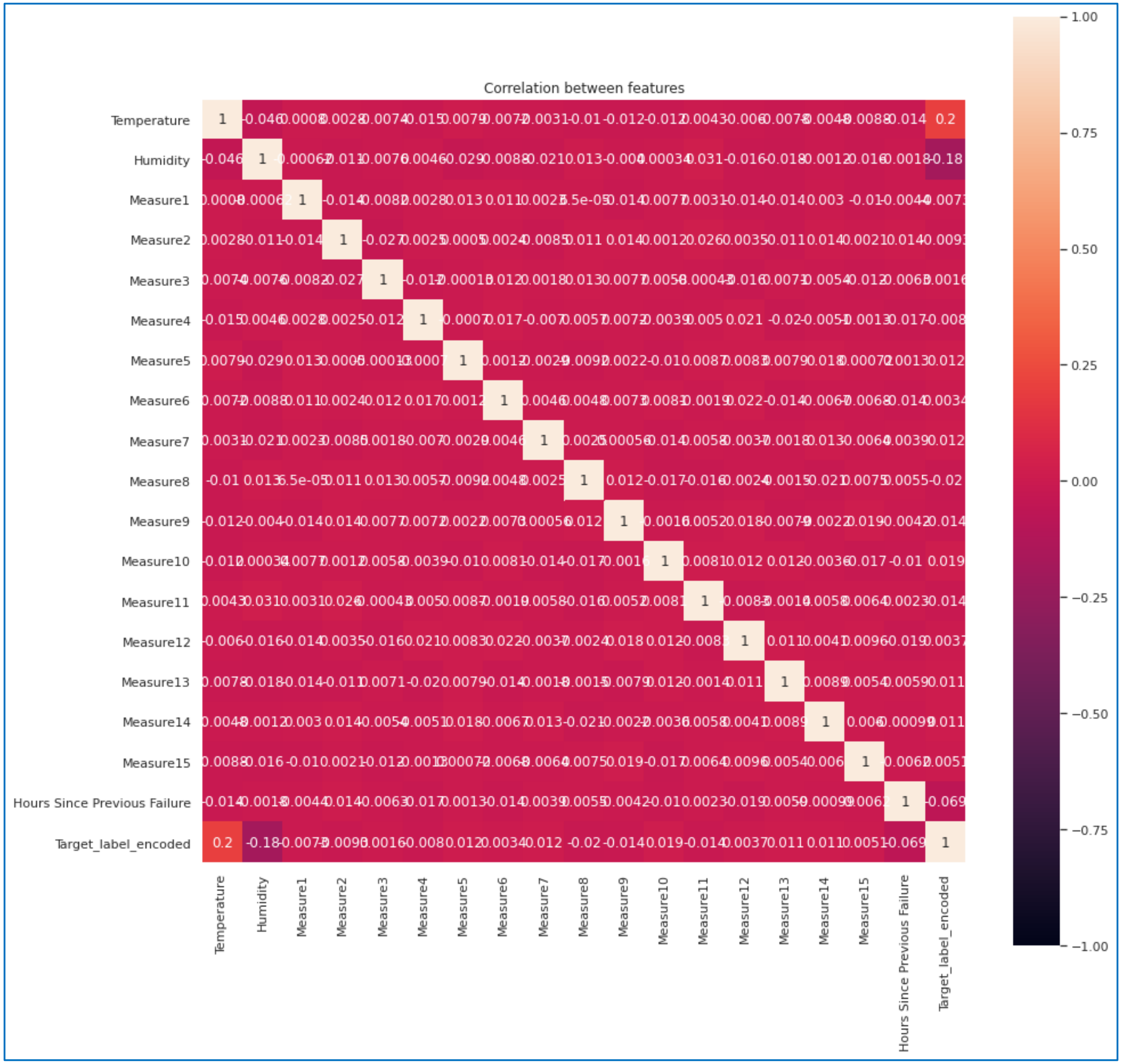
Exploratory Data Analysis : variable analysis

The D-Tale has been used to analyze the **Features** and **Target** as well.



There are no missing values, no outlier and no such skewness also observed among all the Measure 1 to 15, Temperature and Humidity. Operator column is categorical and Pandas dummy has been used to encode the same.

Exploratory Data Analysis : correlation matrix



The features are not correlated among each other and mildly correlated with the Target.

So directly can be proceeded to build and evaluate the optimum model.

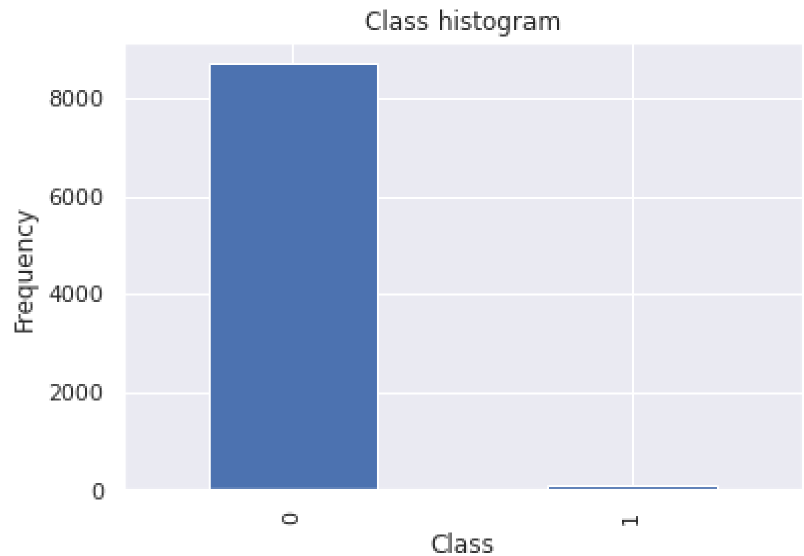
Handling Class Imbalance

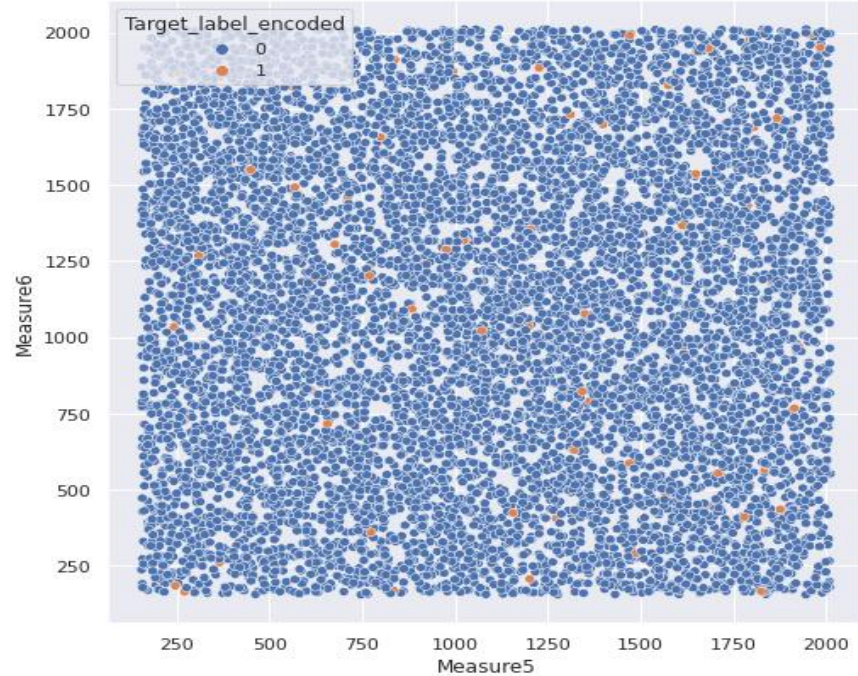
"Target" contains Binary Class imbalance

Class Imbalance handling

It has been observed that the **"Yes (1)"** presents **< 1%** against **"No (0)"**

```
0    8703
1    81
Name: Target_label_encoded, dtype: int64
```



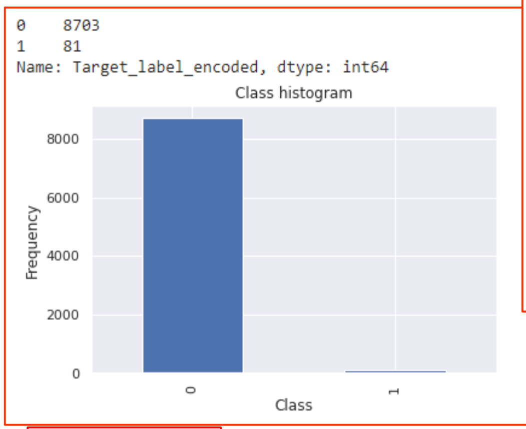


Under sampling may lead to loose some of the informative data so oversampling has been chosen as a next step. Now if Train-Test split is being carried out after oversampling done, some information from Train set may bleed to Test set thru' synthetic sample so it has been decided to split first and then oversample the Train set.

Train-Test Split

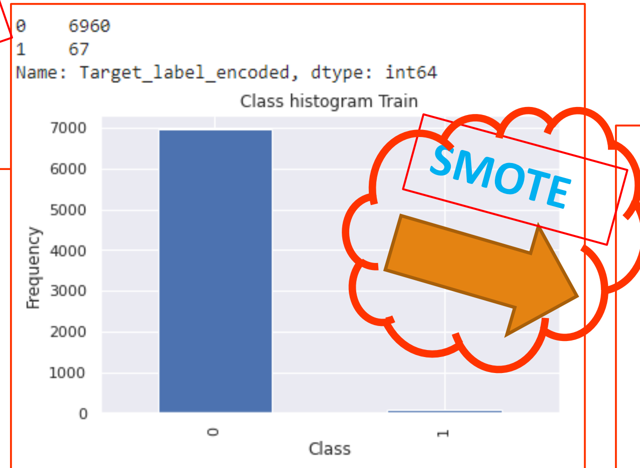
Dataset has been split first and then Oversampling adopted

Train-Test Split and Oversampling

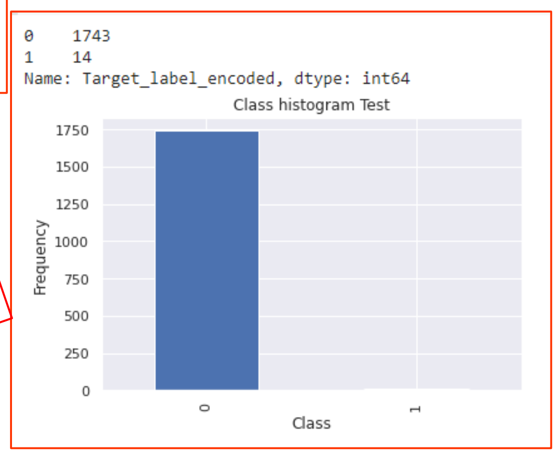


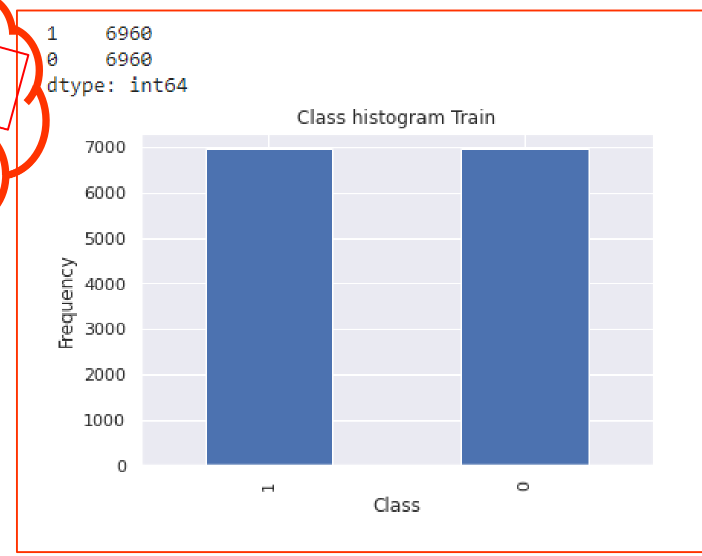
Dataset

Train = 80%



Test = 20%





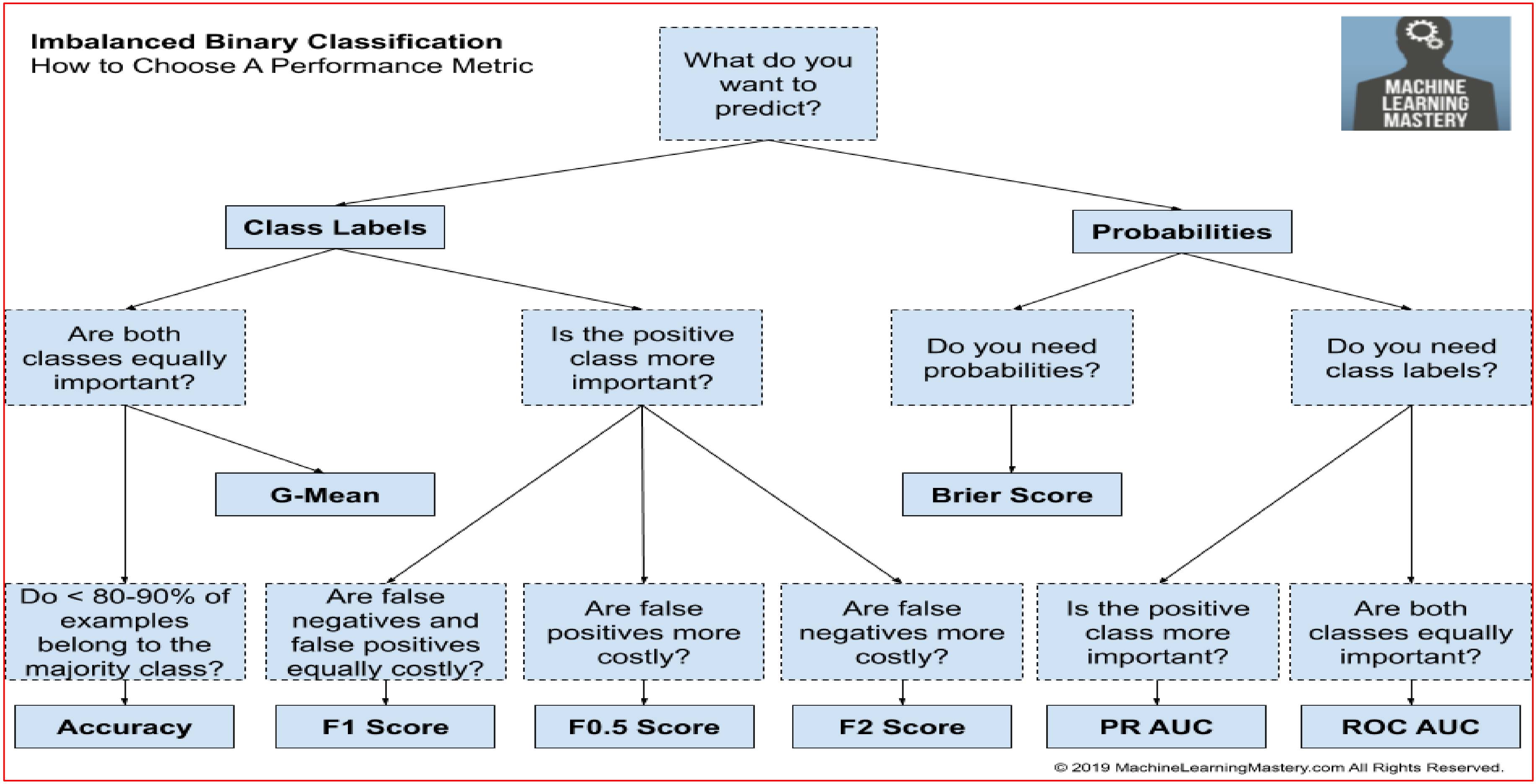
Train set after Oversampling

"Sklearn" is being used to split the train-test and **"imblearn"** for oversampling the train set.

Model Building & Evaluation

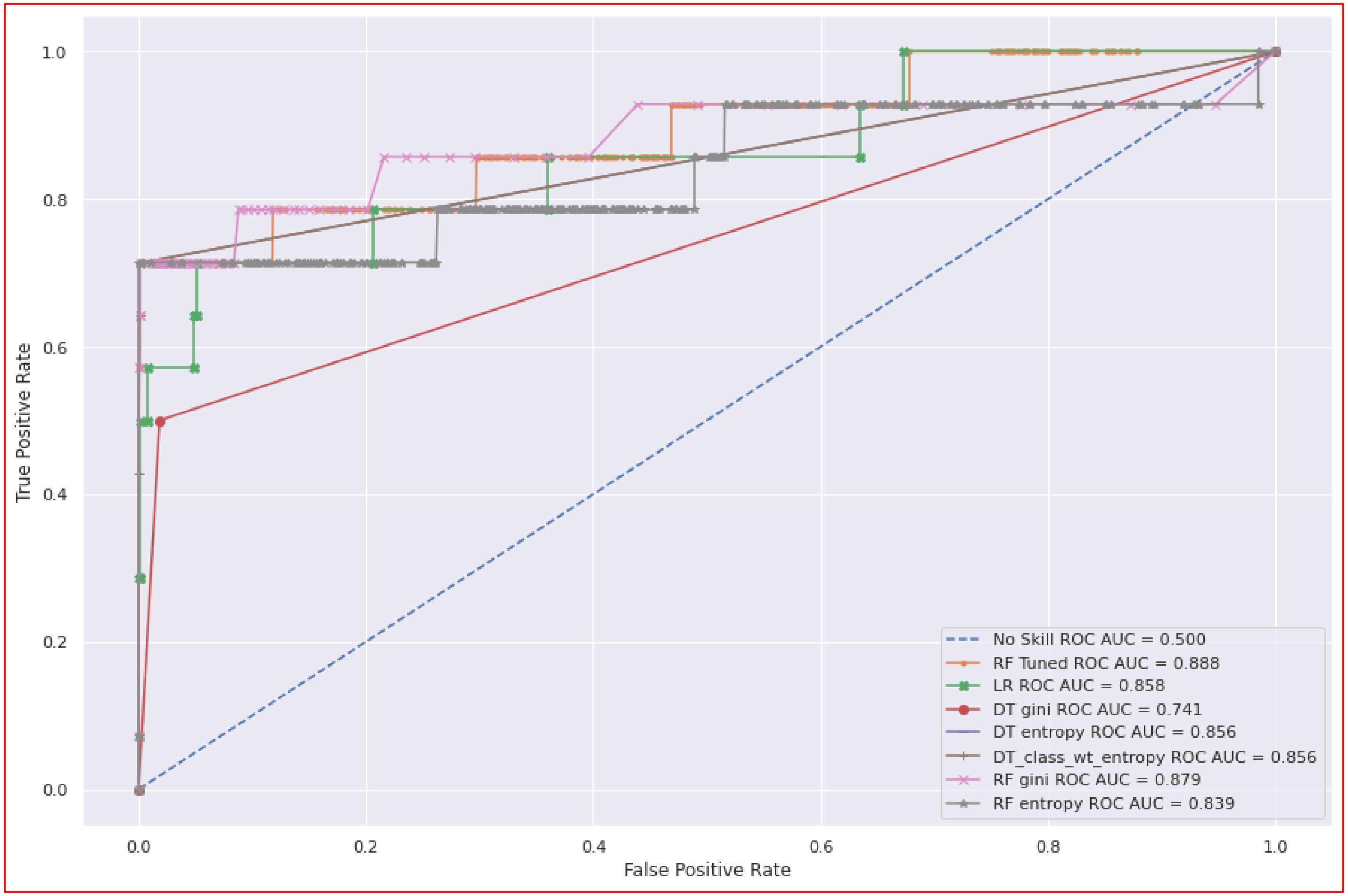
Binary Classification techniques are being adopted, tuned and Metric Evaluation done

Model Building & Evaluation metric framework



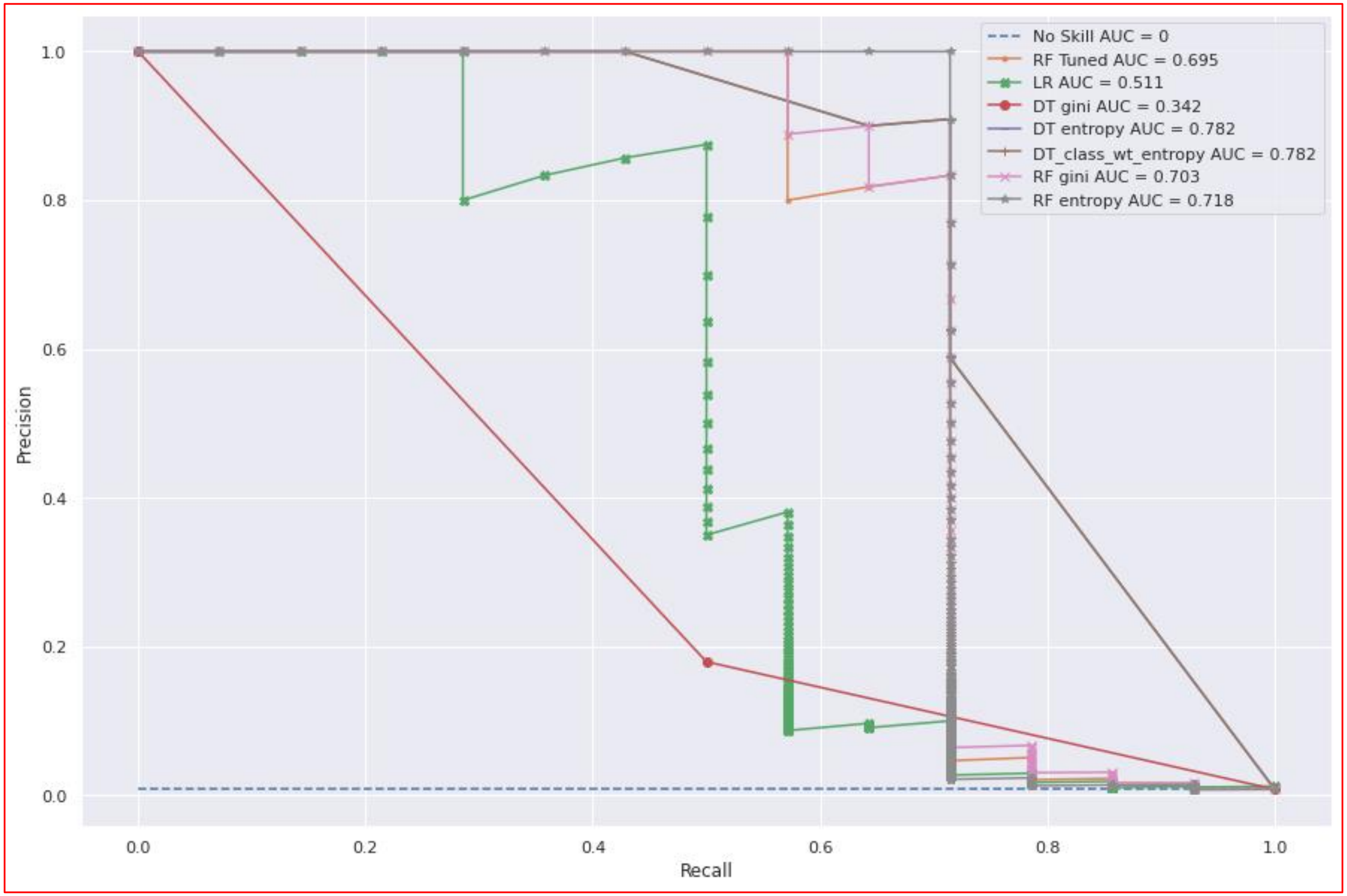
All Classification models are being adopted. **F2 score**, **ROC AUC** and **PR AUC** have been used as the evaluation metric for the models.

Model Evaluation : ROC AUC



ROC AUC wise RF Tuned (RandomSearch CV) is the best. **RF (Gini)** is also good

Model Evaluation : PR AUC

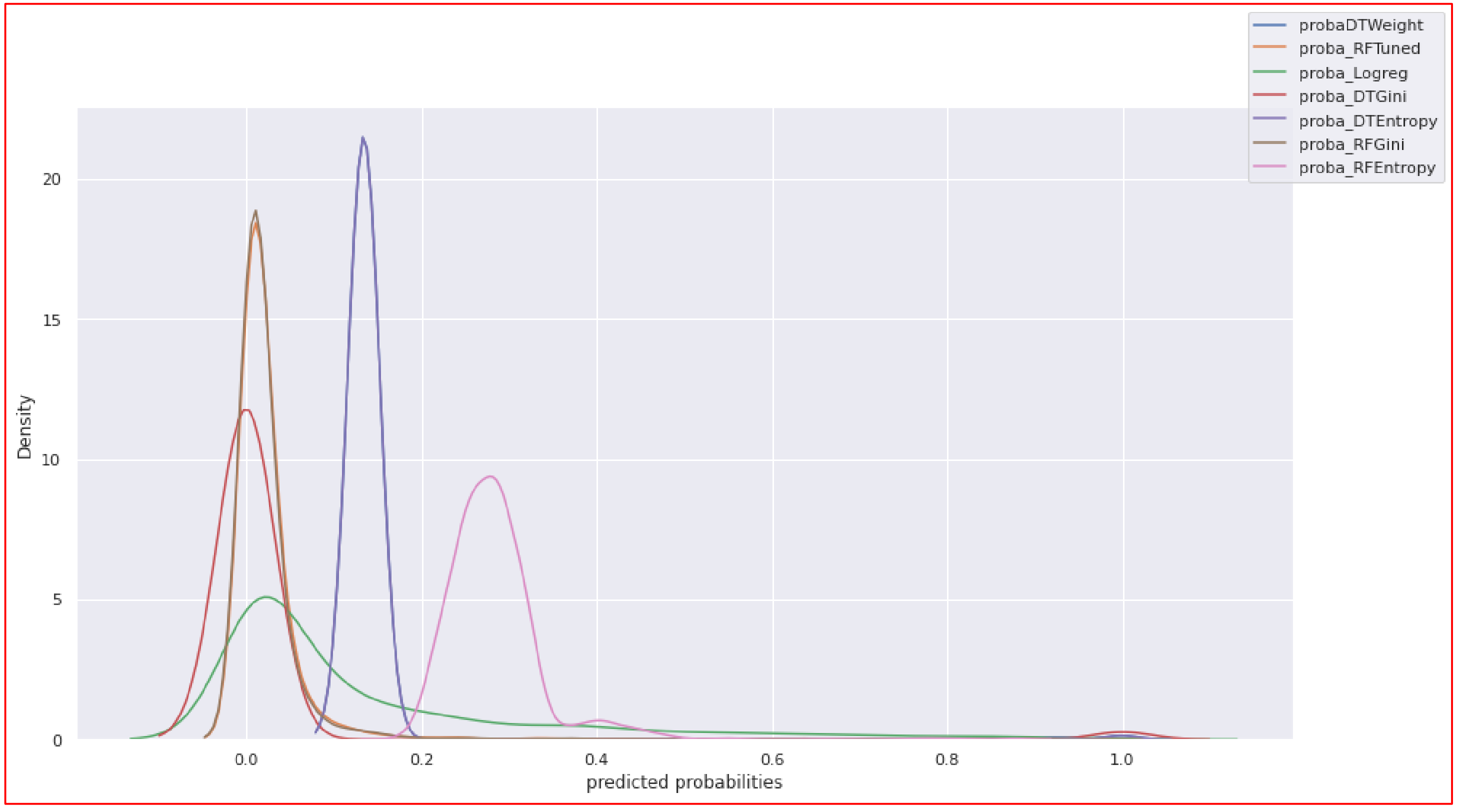


PR AUC wise **DT Entropy** & **DT (with Class weight)** both are best. **RF (Entropy)** and **RF (Gini)** also shows 70% and above AUC.

Model Evaluation : Parameters & Scores

Sl No	Model	Parameters	Accuracy	F2 Score	Precision	Recall	ROC AUC	PR AUC
1	**Logistic Regression**	Penalty = ‘l2’ ; c =1.0 ; solver = ‘lbfgs’	0.93	0.939	0.08	**0.71**	0.858	0.511
2	**DT – Gini**	min_samples-leaf =1 min_samples_split = 2	0.98	0.980	0.18	0.50	0.741	0.342
3	**DT - Entropy**	min_samples_leaf =4 min_samples_split = 5 max_leaf_node = 10	0.99	0.994	0.59	**0.71**	0.856	**0.782**
4	**RF – Gini**	min_samples_leaf =1 min_samples_split = 2 n_estimator = 600	1.00	**0.997**	**0.83**	**0.71**	**0.879**	**0.703**
5	**RF – Entropy**	min_samples_leaf =3 min_samples_split = 4 n_estimator = 100	1.00	0.996	0.77	**0.71**	0.839	0.718
6	**DT_Entropy (Tuned) (with Class weight)**	Class weight = {1:700} min_samples_leaf =3 min_samples_split = 4 n_estimator = 100	0.99	0.994	0.59	**0.71**	0.856	**0.782**
7	**RF Tuned (RandomSearch CV)**	criterion = ‘gini’ min_samples_leaf =1 min_samples_split = 7 n_estimator = 168 max_depth = 30 max_features = ‘log2’	1.00	**0.997**	**0.83**	**0.71**	**0.888**	0.695

Model Evaluation : Predicted Probability of Minority class



On average except **RF (Entropy)** predicted probabilities for minority class are below 0.2 (which resulted as outcome **"0" or "No Fault"**)→ means **"threshold"** has almost no significance further on the model selection.

Cross Validation & Model Finalization

Cross Validation are being adopted before finalization of model and Prediction made

Cross Validation on Train dataset and Final Prediction of Test dataset

```
[ ] rfc

    RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                           criterion='gini', max_depth=None, max_features='auto',
                           max_leaf_nodes=None, max_samples=None,
                           min_impurity_decrease=0.0, min_impurity_split=None,
                           min_samples_leaf=1, min_samples_split=2,
                           min_weight_fraction_leaf=0.0, n_estimators=600,
                           n_jobs=None, oob_score=False, random_state=None,
                           verbose=0, warm_start=False)

[ ] from sklearn.model_selection import cross_validate

[ ] cv_results_rf = cross_validate(rfc, df_Xosmpl,df_yosmpl.values.reshape(-1,), cv=5, scoring='recall',verbose = 2)
```

```
[ ] cv_results_rf['test_score']

    array([0.99928161, 1.        , 0.99928161, 0.99928161, 1.        ])

[ ] cv_results_rf['test_score'].mean()

    0.9995689655172415

[ ] cv_results_rf['test_score'].std()

    0.000351938181434354
```

```
[ ] y_pred_test = rfc.predict(X_testlr)

[ ] test_predictions = X_testlr.copy()

[ ] test_predictions['actual'] = y_testlr
    test_predictions['predicted'] = y_pred_test
    test_predictions
```

```
[ ] test_predictions.to_csv('test_predictions.csv')
```

RF (Gini) model cross validated **(cv=5)** with **"Recall" as scoring** and performs well with **0.7307 (mean)** and **0.079 (s.d.)** and final prediction made

Conclusion and Next Steps

Conclusion and Recommendation on next steps

- Model for binary classification with <1% minority class ("Yes" or "1") has been built well with the help of SMOTE oversampling technique.

- Random Forest (with Gini) performed well and has been selected for this predictive maintenance case study.

- "Recall" achieved maximum 0.71 and it is expected that this can further be improved with Neural network based models.

- This case study can also be referred and used for similar type binary classification problem.

Thanks for Reading

Lets collaborate and happy to receive any feedback/suggestion/comment at…….

pathak.chiranjit@gmail.com