RENEWIND

MODEL TUNING PROJECT

01/07/2023

Sania

Contents / Agenda

Executive Summary

- A machine learning model has been built to minimize the total maintenance cost of machinery/processes used for wind energy production.

- The final tuned model (XGBoost) was chosen after building ~6 different machine learning algorithms & further optimizing for target class imbalance (having few "failures" and many "no failures" in dataset) as well as finetuning the algorithm performance (hyperparameter and cross validation techniques)

- 6 different machine learning algorithms were fit on original training dataset - Logistic Regression - Random Forest - Bagging Classifier - Boosting Classifiers (AdaBoost, Gradient Boost, XGBoost)

- * 5 fold cross validation was performed

Class imbalance was handled by

- - Synthetic Minority Oversampling Technique (SMOTE)
- Random Under sampler & the 6 machine learning algorithms were fit again on OverSampling datasets.

Contd........

- The main attributes of importance for predicting failures vs. no failures were found to be "V18", "V39", "V26", "V3" & "V10“ in order of decreasing importance.

- Model performance was evaluated on validation data set to assess if model generalizes well or is prone to overfitting/underfitting

Business Problem Overview and Solution Approach

BUSINESS PROBLEM:

RENEWIND” is a company working on improving the machinery/processes involved in the production of wind energy using machine learning and has collected data of generator failure of wind turbines using sensors.

SOLUTION APPROACH:

The solution is to build various classification models, tune them and find the best one that will help identify failures so that the generator could be repaired before failing/breaking and the overall maintenance cost of the generators can be brought down.

The nature of predictions made by the classification model will translate as follows: True positives (TP) are failures correctly predicted by the model. False negatives (FN) are real failures in a wind turbine where there is no detection by model. False positives (FP) are detections in a wind turbine where there is no failure.

Here the objective is to reduce the maintenance cost so, we want a metric that could reduce the maintenance cost.

The minimum possible maintenance cost = Actual failures*(Repair cost) = (TP + FN)*(Repair cost)

The maintenance cost associated with model = TP*(Repair cost) + FN*(Replacement cost) + FP*(Inspection cost)

So, we will try to maximize the ratio of minimum possible maintenance cost and the maintenance cost associated with the model.

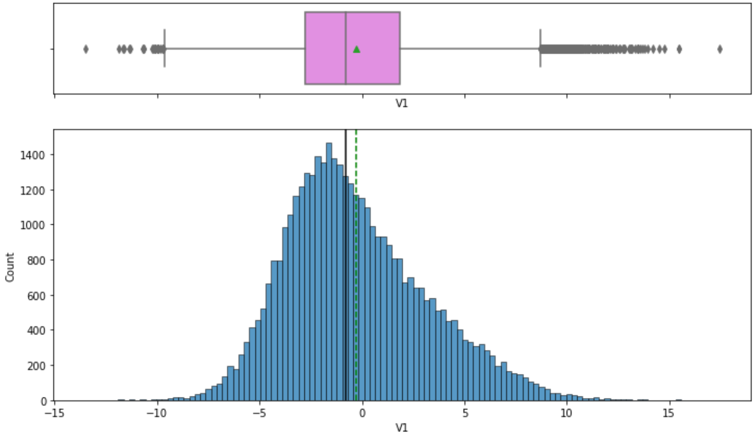
DATA OVERVIEW

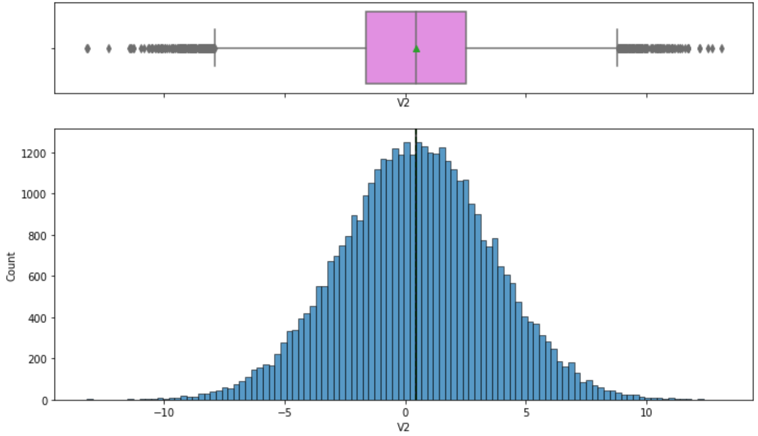
- There are 40,000 rows and 41 attributes (including the predictor) in the dataset.
- Of the 40 columns with sensor data, all of the columns are of datatype 'float64'. This is as expected, since the information from the sensors should be continuous data, both positive or negative.
- The dependent variable is of the datatype 'int64’. The variable is a binary variable i.e.

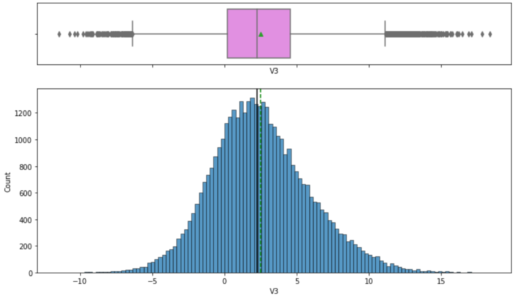
1 for ‘Failure’ and 0 for ‘No failure’.

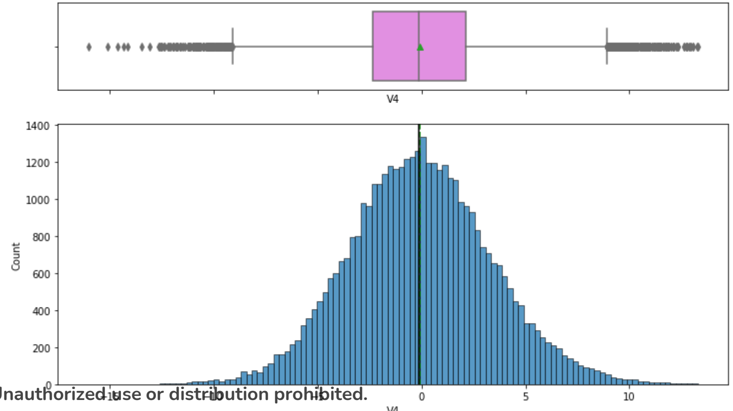
- "Target" class is imbalanced with 37813 or 94.53% "No failures (i.e., 0)" and 2187 or 5.47% "Failures (i.e., 1)"

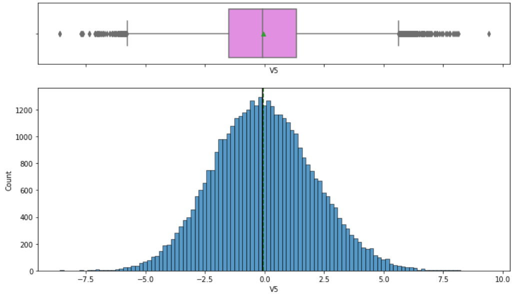
HISTOGRAMS AND BOXPLOTS FOR ALL VARIABLES

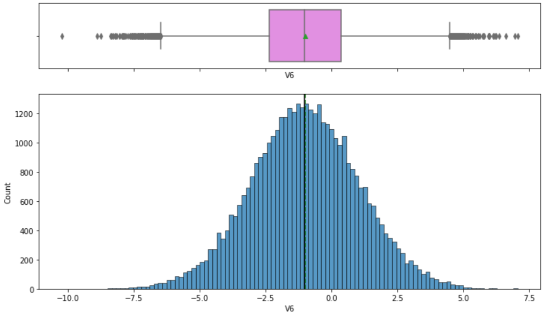


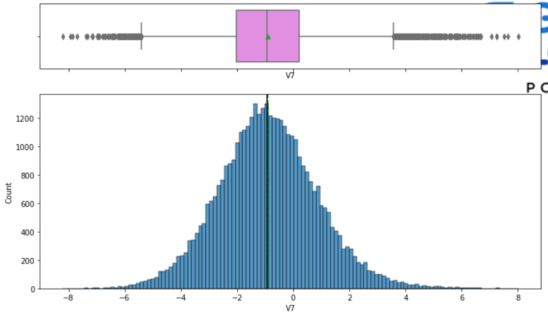


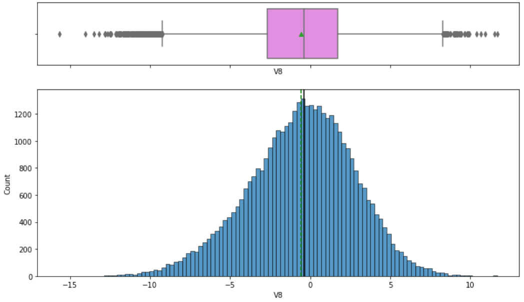


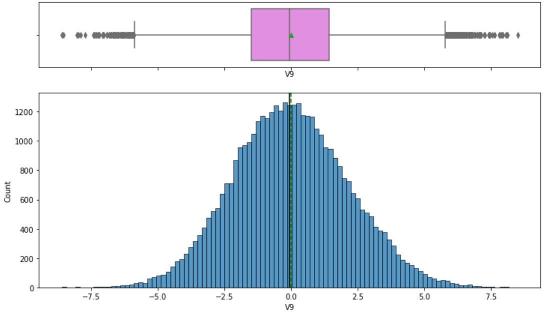


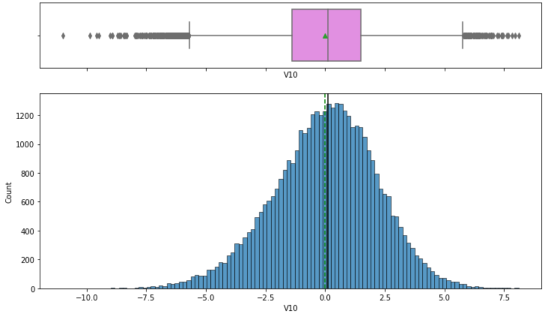

V5
Count

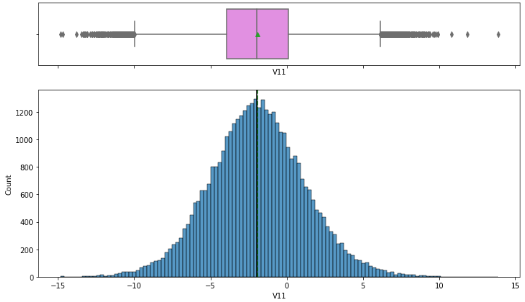

V6
Count

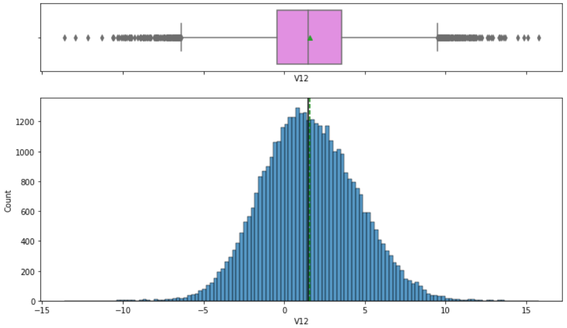

V7
Count

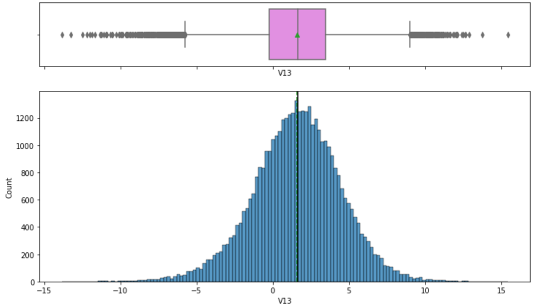

V8
Count

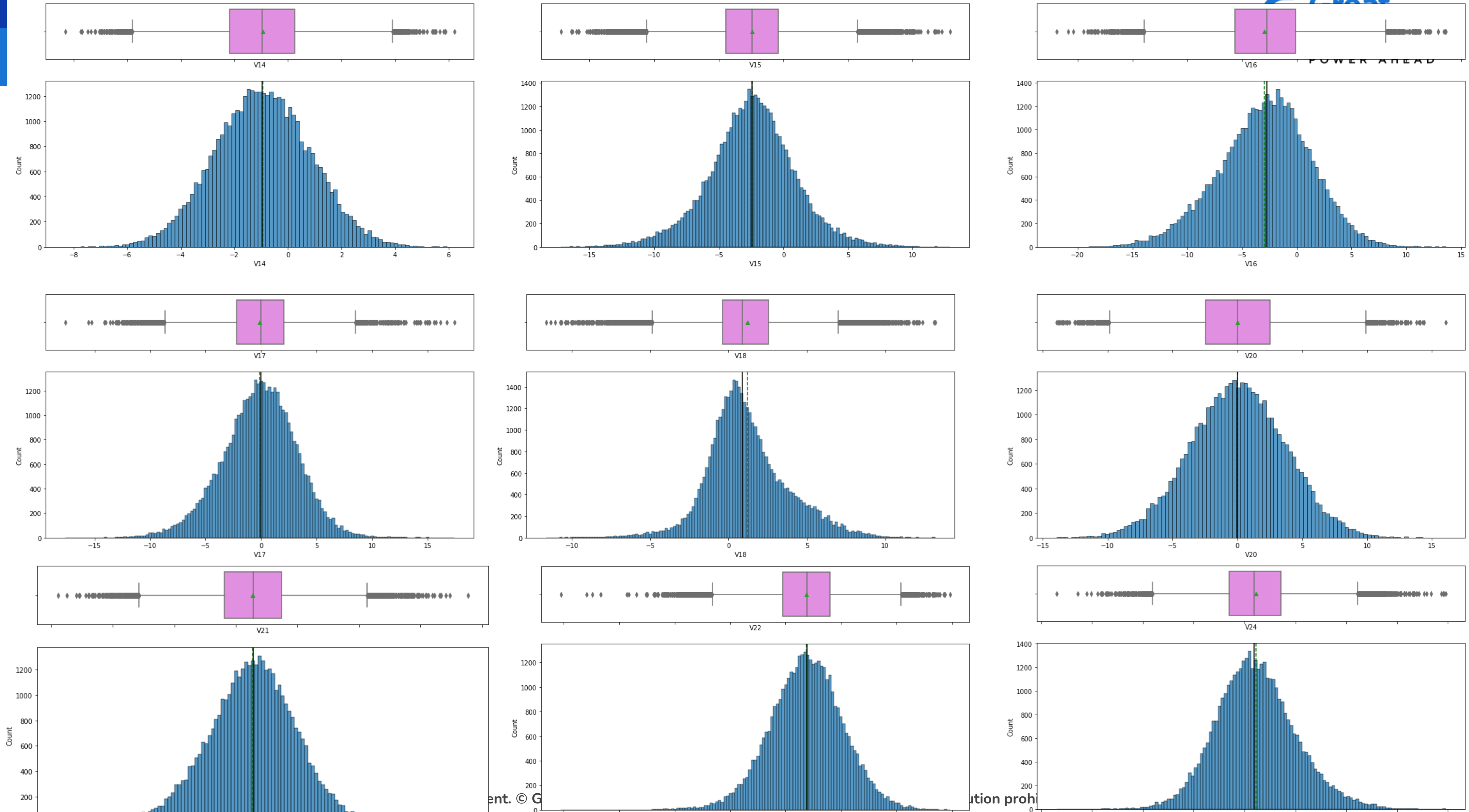

V9
Count

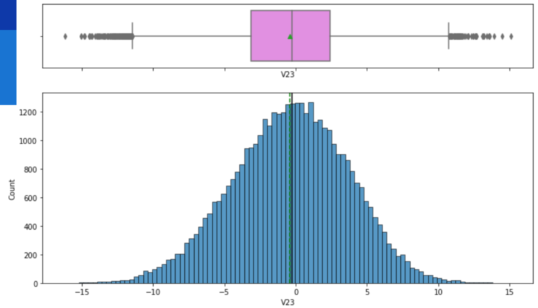

V10
Count

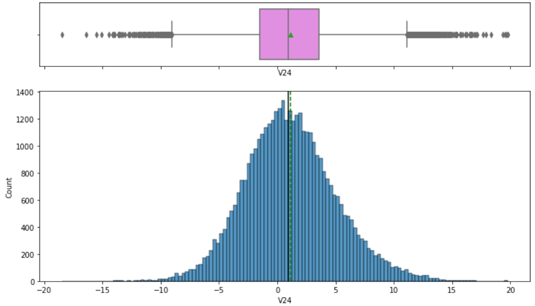

V11
Count

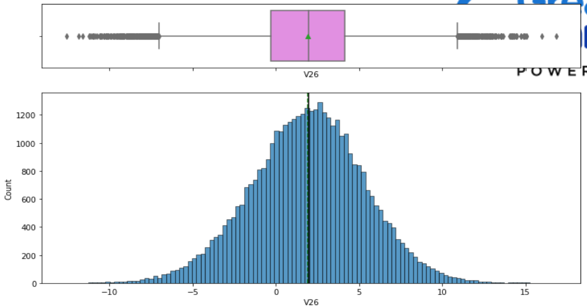

V12
Count

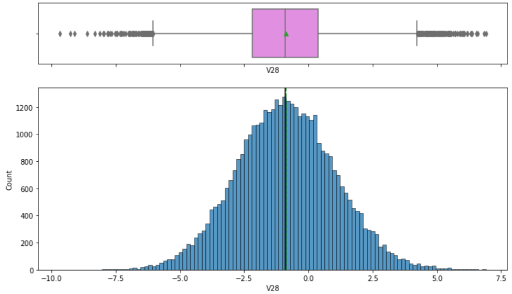

V13
Count

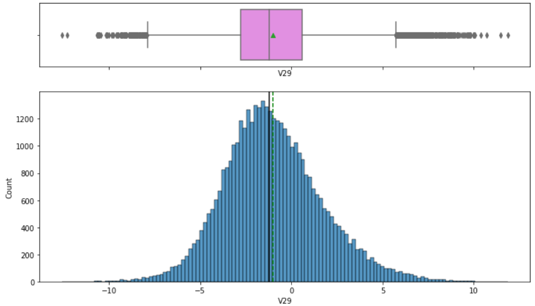

V14
V15
V16
Count
V17
V18
V20
V21
V22
V24

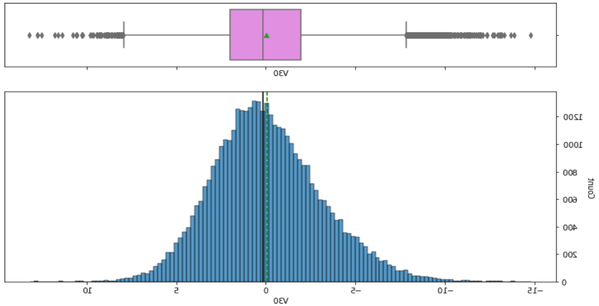

V23
Count

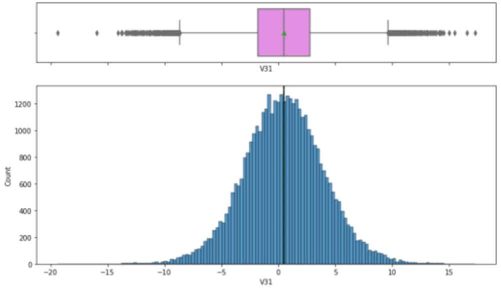

V24
Count

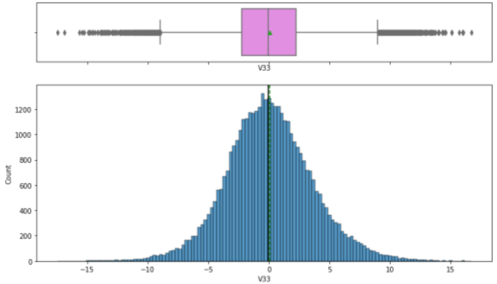

V26
Count

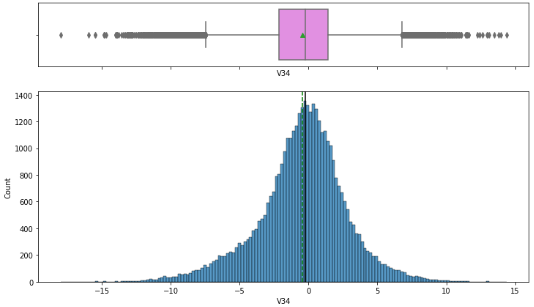

V28
Count


V29
Count



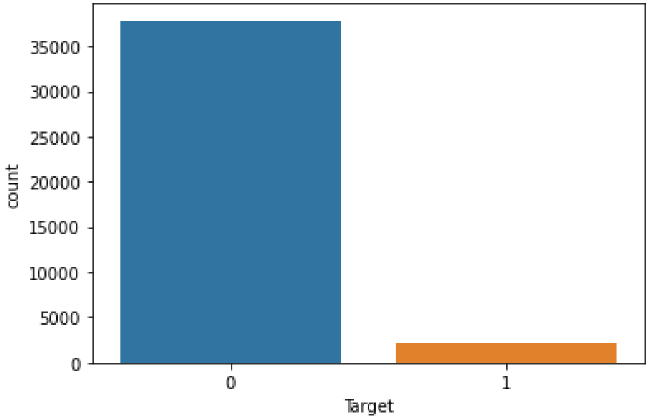

V31
Count


V33
Count


V34
Count

EDA INSIGHTS

- The distribution of the above histogram and boxplot for all variables is uniform
- There are positive and negative outliers for all the attributes.
- "Target" class is imbalanced with 37813 or 94.53% "No failures (i.e., 0)" and 2187 or 5.47% "Failures (i.e., 1)"



DATA PREPROCESSING

- There are 46 missing values for attribute "V1" and 39 missing values for attribute "V2“
- **Median of the attribute was used to impute any missing values in the attributes** (i.e., for “V1” and “V2” which had 46 & 39 missing values) - This was performed separately on training set and validation set to prevent any **data leak**!
- There are no duplicate values in the dataset.
- There are Outliers in all the attributes . The outliers are not treated and are assumed to be valuable data.
- The dataset is split into TRAINING , VALIDATION AND TESTING SET.
- The nature of predictions made by the classification model will translate as follows:

1. True positives (TP) are failures correctly predicted by the model.
2. False negatives (FN) are real failures in a generator where there is no detection by model.
3. False positives (FP) are failure detections in a generator where there is no failure.

We need to choose the metric which will ensure that the maximum number of generator failures are predicted correctly by the model.

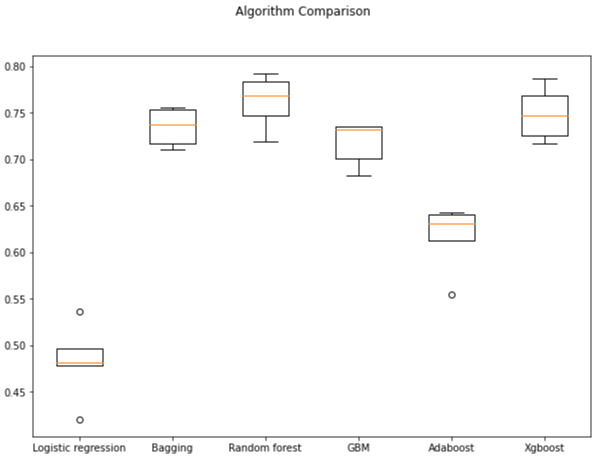
1. We would want Recall to be maximized as greater the Recall, the higher the chances of minimizing false negatives.
2. We want to minimize false negatives because if a model predicts that a machine will have no failure when there will be a failure, it will increase the maintenance cost.

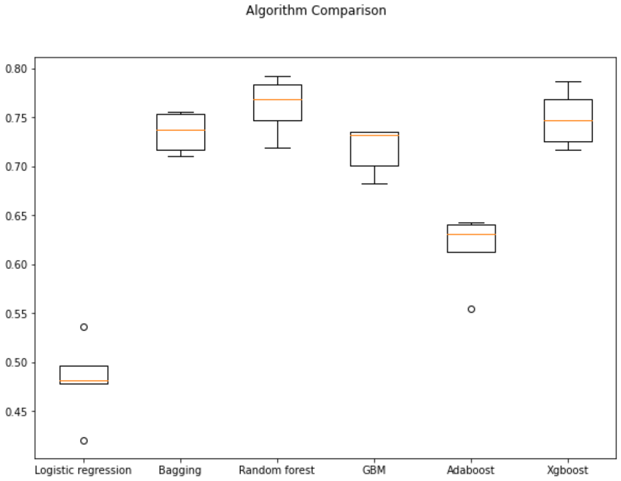
MODEL PERFORMANCE And SELECTION

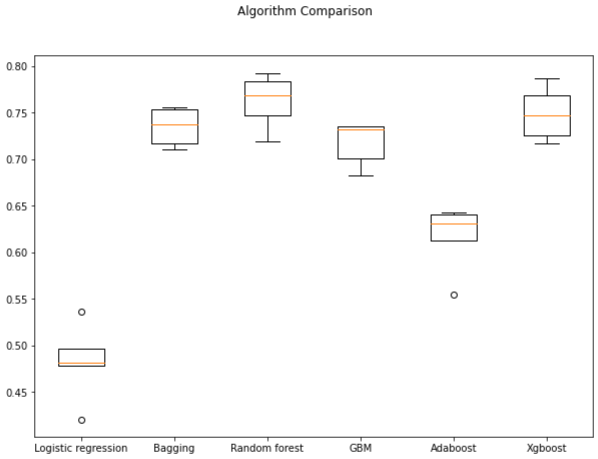
	CROSS VALIDATION COST	VALIDATION PERFORMANCE
ORIGINAL DATASET	Logistic regression: 0.48292682926829267 Bagging: 0.7347560975609755 Random forest: 0.7621951219512195 GBM: 0.7170731707317073 Adaboost: 0.6164634146341463 Xgboost: 0.748780487804878	Logistic regression: 0.4625228519195612 Bagging: 0.7349177330895795 Random forest: 0.7659963436928702 GBM: 0.714808043875685 Adaboost: 0.6142595978062158 Xgboost: 0.7385740402193784
SMOTE	Logistic regression: 0.8755997227156277 Bagging: 0.9706277576361164 Random forest: 0.9782793622519118 GBM: 0.914598180586688 Adaboost: 0.8970030978718991 Xgboost: 0.9100142881554313	Logistic regression: 0.41917808219178077 Bagging: 0.8502325581395348 Random forest: 0.9125840537944284 GBM: 0.7386973180076628 Adaboost: 0.4957356076759062 Xgboost: 0.7409126063418406
UNDER-SAMPLING	Logistic regression: 0.8560959657851797 Bagging: 0.8652322692542072 Random forest: 0.8981658195552162 GBM: 0.8884067384981461 Adaboost: 0.8670648871745764 Xgboost: 0.8853598158899804	Logistic regression: 0.4052516411378556 Bagging: 0.6560111188325226 Random forest: 0.7423312883435583 GBM: 0.6662097326936258 Adaboost: 0.44020474639367146 Xgboost: 0.6689798750867454

Summary on Model selection

- Models built on original dataset have given generalized performance on cross validation training and validation sets unlike models built on oversampled and undersampled sets.
- Mean cross validation scores on training sets are highest with XGBoost, Random Forest & Bagging Classifiers (~77, ~71 and ~68% respectively). These models will be tuned further to try to increase performance







Originaldatasets

SMOTE

UNDERSAMPLER

Performance comparison

Training performance comparison:

	Gradient Boosting tuned with oversampled data	XGBoost tuned with oversampled data	Bagging classifier tuned with oversampled data	Random forest tuned with oversampled data
Accuracy	0.976	0.948	0.500	1.000
Recall	0.969	0.998	1.000	0.999
Precision	0.982	0.908	0.500	1.000
F1	0.975	0.951	0.667	1.000

VALIDATION PERFORMANCE COMPARISON

	Gradient Boosting tuned with oversampled data	XGBoost tuned with oversampled data	Bagging classifier tuned with oversampled data	Random forest tuned with oversampled data
Accuracy	0.964	0.889	0.055	0.990
Recall	0.872	0.918	1.000	0.872
Precision	0.625	0.320	0.055	0.948
F1	0.728	0.475	0.104	0.909

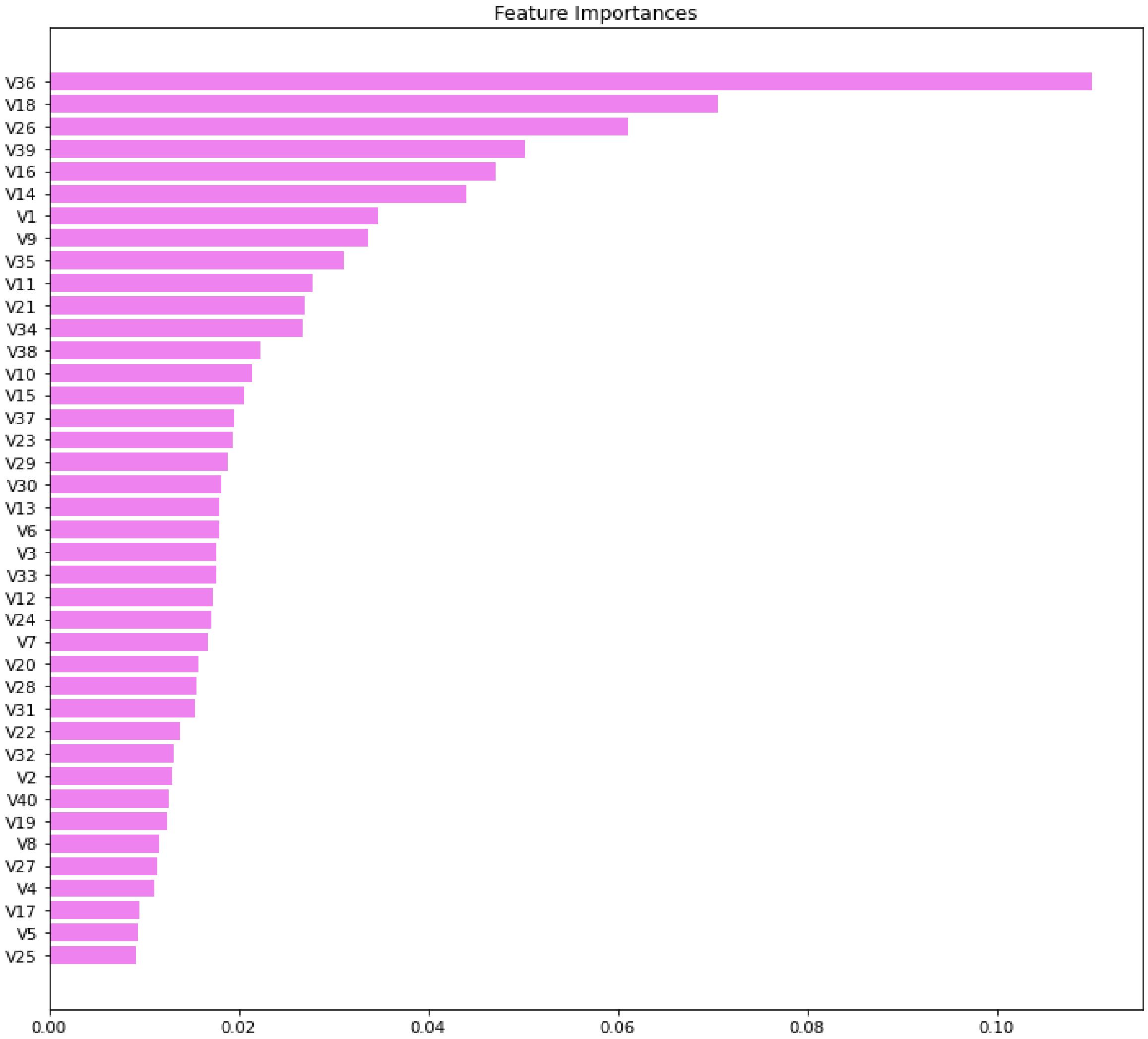
The XGBoost Tuned model with oversampled data is giving the highest Recall score of 0.918 on the Validation Set.
We will choose this tuned model to see if it can generalize well on the testing dataset

The XGBoost tuned model is generalizing well on the test data with Recall score of 0.887 and Acuuracy of 0.886. But a low precision and F1 score

- Test performance:

Accuracy	Recall	Precision	F1
0.886	0.887	0.311	0.460

- Furthermore, the data pre processing steps & XGBoost tuned learning model were automated by building pipelines (for productionized model)



Great Learning
POWER AHEAD

The top attributes which have the maximum importance for making accurate failure/ no-failure predictions are "V36", "V18", "V26", "V39" & "V16

Productionize and test the final model using pipelines

- Now that we have a final model, let's use pipelines to put the model into production. We know that we can use pipelines to standardize the model building, but the steps in a pipeline are applied to each and every variable.
- Since we have only one datatype in the data, we don't need to use column transformer here

<u>The OUTPUT of the Final model fitted using pipeline</u>

```
Pipeline(steps=[('imputer', SimpleImputer(strategy='median')),
               ('XGB',
                XGBClassifier(eval_metric='logloss', gamma=3, n_estimators=250,
                              random_state=1, scale_pos_weight=10,
                              subsample=0.9))])
```

BUSINESS INSIGHTS AND RECOMMENDATIONS

- The machine learning model has been built to minimize the total maintenance cost of machinery/processes used for wind energy production.
- The XGBOOST model built from oversampled data provided the highest RECALL score across both the validation and testing data and performed similarly well on both, leading us to the conclusion that this model should generalize well in production.
- The model is expected to generalize well in terms of predictions on TEST dataset.
- The top three most important sensors for predicting a generator failure are V36, V18, and V26
- This Data can be used to refine the process of collecting more frequent sensor information to be used in improving the machine learning model to further decrease maintenance costs.

Great Learning

Happy Learning !