Model Tuning – Julie Kistler

02/09/2024

Contents / Agenda

RenewWind

Executive Summary

Overview: Predictive maintenance is a strategic initiative designed to proactively anticipate component degradation and forecast future component capabilities. This approach involves identifying predictable failure patterns and replacing components before they fail, leveraging sensor data and advanced data analysis techniques. The ultimate aim is to significantly mitigate operational and maintenance costs.

Project Objectives: The primary focus of this project was to develop and fine-tune various classification models geared towards accurately identifying wind turbine failures. The overarching goal is to predict failures in advance, enabling timely repairs and ultimately reducing the overall maintenance expenditure.

Key Insights:

- **Best Performing Model:** The XGB oversampling model emerged as the top-performing model in the project.
- **Recall Metrics:** The XGB final model demonstrated exceptional performance with a 100% recall on the training set, 89% on the validation set, and 86% on the test set.
- **Production-Ready Pipeline:** The XGB pipeline, designed for production deployment, exhibited high accuracy on the test set.
- **Critical Features:** V36, V16, and V26 were identified as having the most significant impact on predicting wind turbine failures.

Executive Summary cont...

Recommendations:

- ReneWind is advised to prioritize improvement efforts on features V36, V16, and V26 to effectively reduce the number of failures.
- Focusing on enhancing these critical features is anticipated to lead to substantial savings in repair and replacement costs, contributing to increased overall operational efficiency and cost-effectiveness.

The successful implementation of predictive maintenance, with a specific emphasis on the identified critical features, stands to position the company for increased reliability and reduced operational expenditures in wind turbine maintenance.

Business Problem Overview and Solution Approach

ReneWind is a company in the renewable energy sector, dedicated to improving the machinery and processes involved in wind energy production through the implementation of machine learning. The company has a collection of data capturing instances of generator failure in wind turbines, obtained through sensors. To safeguard the sensitivity of this information, a ciphered version has been shared, with the type of data collected varying among different companies. The dataset is comprised of 40 predictors, with 20,000 observations in the training set and 5,000 in the test set.

The solution is to the develop and fine-tune a diverse array classification models. These models are designed to identify potential failures in wind turbine generators striving to proactively address failure issues before they lead to operational breakdowns, minimizing overall maintenance costs. The goal is to reduce overall maintenance costs through early detection and intervention.

Given the cost dynamics, with repair being substantially more economical than replacement, and inspection costs being lower than repair expenses, the selected model aims to optimize maintenance strategies for cost-effectiveness and enhance operational efficiency.

Data Overview

The predictions of the classification model are categorized as follows

- True positives (TP): Failures accurately predicted by the model, contributing to repair costs.
- False negatives (FN): Actual failures undetected by the model, resulting in higher replacement costs.
- False positives (FP): Detections where no actual failure occurs, leading to inspection costs.

- “1” in the target variables should be considered as “failure” and “0” represents “No failure”. -

Data Description

- The data provided is a transformed version of original data which was collected using sensors.
- Train.csv - To be used for training and tuning of models.
- Test.csv - To be used only for testing the performance of the final best model.

- Both the datasets consist of 40 predictor variables and 1 target variable -

Data Overview Cont…

Column	Dtype
V1	float64
V2	float64
V3	float64
V4	float64
V5	float64
V6	float64
V7	float64
V8	float64
V9	float64
V10	float64
V11	float64
V12	float64
V13	float64
V14	float64
V15	float64
V16	float64
V17	float64
V18	float64
V19	float64
V20	float64

40 float data types: (V1 – V40)
1 Integer data type: Target

Column	Dtype
V21	float64
V22	float64
V23	float64
V24	float64
V25	float64
V26	float64
V27	float64
V28	float64
V29	float64
V30	float64
V31	float64
V32	float64
V33	float64
V34	float64
V35	float64
V36	float64
V37	float64
V38	float64
V39	float64
V40	float64
Target	int64

ReNewind

Data Overview Cont…

ReNewind

Training Set

Columns	Rows
20000	41

- V1 has 18 missing values
- V2 has 18 missing values

Test Set

Columns	Rows
5000	41

- V1 has 5 missing values
- V2 has 6 missing values

-- There are no duplicate values --

Statistical Summary of the Data

- The data has been encrypted – not much can be discovered through the data
- V1 and V2 are missing values
- The target variable is a 0 – 1 variable with a mean of .056 and standard deviation of .23

Link to Appendix slide on Statistical Summary

EDA _ Univariate Analysis

- All independent variables appear to have a normal distribution with many outliers
- The target variable appears to have an imbalanced distribution – most of the data represents "0" meaning no failures

Train Data Distribution		
0	18990	95%
1	1100	5%

Test Data Distribution		
0	4718	94%
1	282	6%

The values of 0 and 1 and almost equally split in the both the train and test data

Link to Appendix slides on distribution analysis

Data Preprocessing

- There are no duplicate values: No treatment necessary.
- Missing value treatment: Used median to impute missing values in V1 and V2.
- Outlier check: There are outliers – we will leave them as they may have important value.
- Feature engineering: Target variable was dropped from the features variables (x) and into the dependent variable (y) for model prediction.
- Data preparation for modeling: The data has been divided into Train, Validation, and Test sets:
    - Train Set: (15000, 40)
    - Validation Set: (5000, 40)
    - Test Set: (5000, 40)

Model Performance Summary

ReNewind

XGBoost appears to be the best model with an Recall score of 89% on the validation set and Recall score of 86% on the test data. XGBoost was used to build the pipeline for the final model.

Training Performance				
	Gradient Boosting tuned with oversampled data	AdaBoost classifier tuned with oversampled data	Random forest tuned with undersampled data	XGBoost tuned with oversampled data
Accuracy	0.993	0.948	0.961	0.994
Recall	0.992	0.897	0.933	1
Precision	0.994	1	0.989	0.988
F1	0.993	0.945	0.96	0.994

Validation Performance				
	Gradient Boosting tuned with oversampled data	AdaBoost classifier tuned with oversampled data	Random forest tuned with undersampled data	XGBoost tuned with oversampled data
Accuracy	0.969	0.985	0.938	0.967
Recall	0.856	0.755	0.885	0.892
Precision	0.678	0.977	0.468	0.648
F1	0.757	0.852	0.612	0.75

Test Performance (XGBClassifier - (XGBoost - xgb2)		
Accuracy	[redacted]	0.966
Recall		0.858
Precision		0.649
F1		0.739

- Gradient Boosting performs well in the training set. Recall scores decrease in the validation set. Recall Score: 86%
- AdaBoost performs well in precision and show a possibility of overfitting in the accuracy scores. FI and Recall suffer a reduction in the validation set. Recall Score reduced to 76%
- Random Forest performs well in the training set; however Recall, Precision, and F1 scores are significantly reduced in the validation set. Recall Score: 89%
- XGBoost consistently performs well in the training set. There is reduction in recall in the validation set. Recall Score: 89%

[Link to Appendix slides on model assumptions]

Model building with pipeline

- Steps taken to build the final model:
  - Defined the pipeline (impute missing values, scale the features, and training the XGBoost classifier)
  - Separated the target variable and features
  - Treated missing values in the training set
  - Oversampled the data using SMOTE
  - Prepared test data (no imputation is performed – handled in pipeline)
  - Trained the model – fit the pipeline on the oversampled training data

- The test set preformed well in making predictions.
  - The recall score was 86% with an accuracy of 96% demonstrating good performance against the previous model performance summary (89% validation set and 86% on the test set).

- The three most important feature importance the model used for prediction are:
  - V36 (Most predominant feature)
  - V16
  - V26

[Link to Appendix slides on Feature Importance and Final Model Assumptions]

APPENDIX

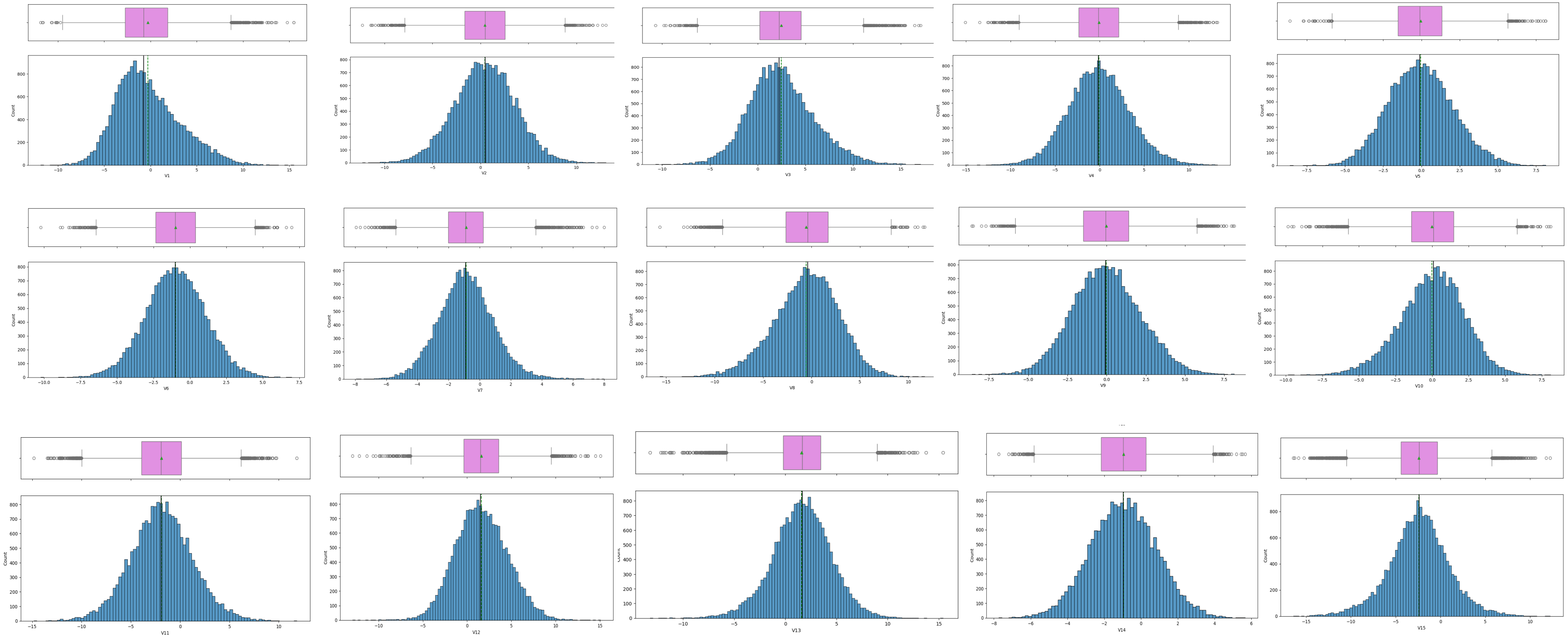
Statistical Summary of the Data

	count	mean	std	min	25%	50%	75%	max
V1	19982	-0.272	3.442	-11.876	-2.737	-0.748	1.84	15.493
V2	19982	0.44	3.151	-12.32	-1.641	0.472	2.544	13.089
V3	20000	2.485	3.389	-10.708	0.207	2.256	4.566	17.091
V4	20000	-0.083	3.432	-15.082	-2.348	-0.135	2.131	13.236
V5	20000	-0.054	2.105	-8.603	-1.536	-0.102	1.34	8.134
V6	20000	-0.995	2.041	-10.227	-2.347	-1.001	0.38	6.976
V7	20000	-0.879	1.762	-7.95	-2.031	-0.917	0.224	8.006
V8	20000	-0.548	3.296	-15.658	-2.643	-0.389	1.723	11.679
V9	20000	-0.017	2.161	-8.596	-1.495	-0.068	1.409	8.138
V10	20000	-0.013	2.193	-9.854	-1.411	0.101	1.477	8.108
V11	20000	-1.895	3.124	-14.832	-3.922	-1.921	0.119	11.826
V12	20000	1.605	2.93	-12.948	-0.397	1.508	3.571	15.081
V13	20000	1.58	2.875	-13.228	-0.224	1.637	3.46	15.42
V14	20000	-0.951	1.79	-7.739	-2.171	-0.957	0.271	5.671
V15	20000	-2.415	3.355	-16.417	-4.415	-2.383	-0.359	12.246
V16	20000	-2.925	4.222	-20.374	-5.634	-2.683	-0.095	13.583
V17	20000	-0.134	3.345	-14.091	-2.216	-0.015	2.069	16.756
V18	20000	1.189	2.592	-11.644	-0.404	0.883	2.572	13.18
V19	20000	1.182	3.397	-13.492	-1.05	1.279	3.493	13.238
V20	20000	0.024	3.669	-13.923	-2.433	0.033	2.512	16.052
V21	20000	-3.611	3.568	-17.956	-5.93	-3.533	-1.266	13.84
V22	20000	0.952	1.652	-10.122	-0.118	0.975	2.026	7.41
V23	20000	-0.366	4.032	-14.866	-3.099	-0.262	2.452	14.459
V24	20000	1.134	3.912	-16.387	-1.468	0.969	3.546	17.163
V25	20000	-0.002	2.017	-8.228	-1.365	0.025	1.397	8.223
V26	20000	1.874	3.435	-11.834	-0.338	1.951	4.13	16.836
V27	20000	-0.612	4.369	-14.905	-3.652	-0.885	2.189	17.56
V28	20000	-0.883	1.918	-9.269	-2.171	-0.891	0.376	6.528
V29	20000	-0.986	2.684	-12.579	-2.787	-1.176	0.63	10.722
V30	20000	-0.016	3.005	-14.796	-1.867	0.184	2.036	12.506
V31	20000	0.487	3.461	-13.723	-1.818	0.49	2.731	17.255
V32	20000	0.304	5.5	-19.877	-3.42	0.052	3.762	23.633
V33	20000	0.05	3.575	-16.898	-2.243	-0.066	2.255	16.692
V34	20000	-0.463	3.184	-17.985	-2.137	-0.255	1.437	14.358
V35	20000	2.23	2.937	-15.35	0.336	2.099	4.064	15.291
V36	20000	1.515	3.801	-14.833	-0.944	1.567	3.984	19.33
V37	20000	0.011	1.788	-5.478	-1.256	-0.128	1.176	7.467
V38	20000	-0.344	3.948	-17.375	-2.988	-0.317	2.279	15.29
V39	20000	0.891	1.753	-6.439	-0.272	0.919	2.058	7.76
V40	20000	-0.876	3.012	-11.024	-2.94	-0.921	1.12	10.654
Target	20000	0.056	0.229	0	0	0	0	1

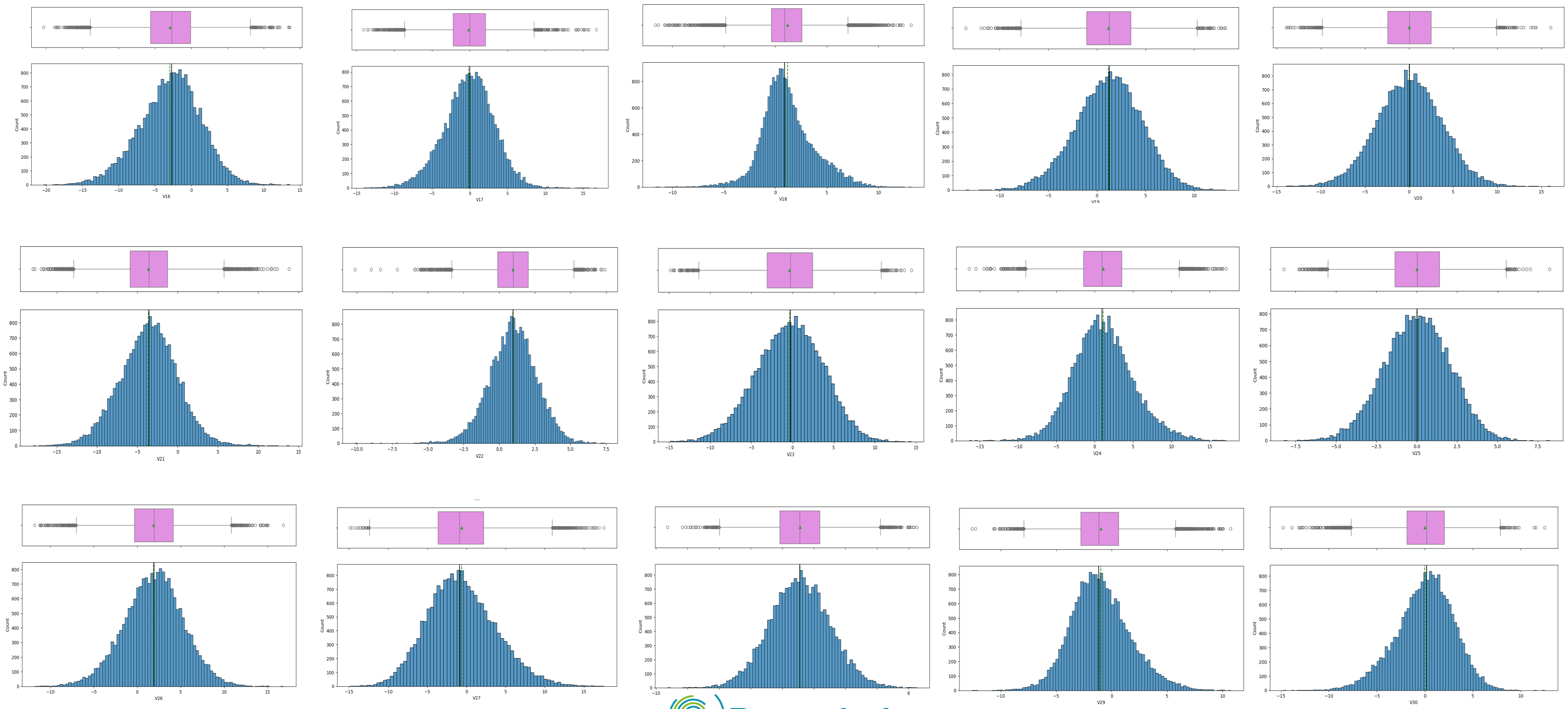
- It appears V1 and V2 are missing some values.
- Based upon the large variance between min and max data compared to the mean it appears the data set may have outliers.
- The target variable is demonstrating and integer type of data of 0 and 1.

ReNewind

Univariate Analysis_histograms and boxplots for all the variables

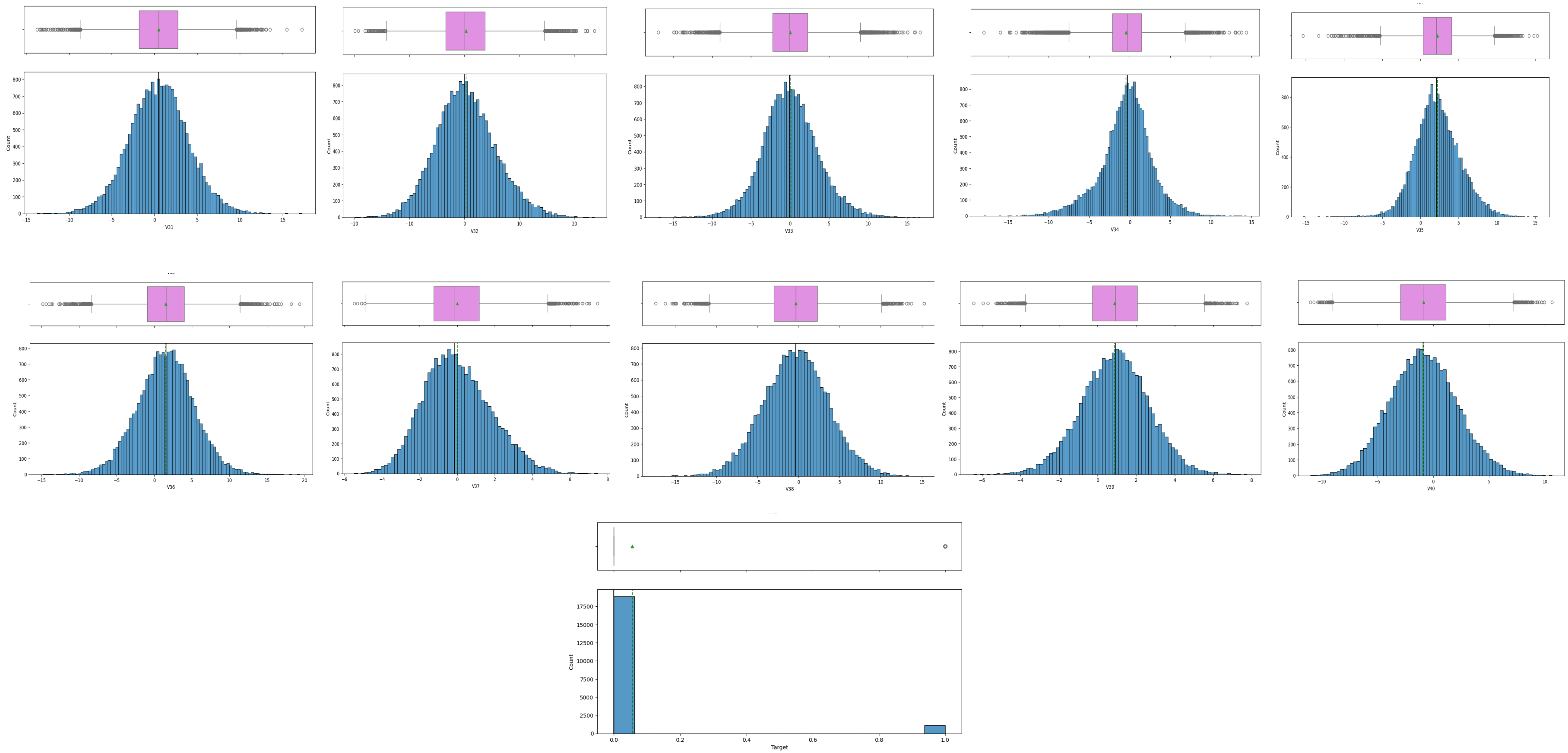


Univariate Analysis_histograms and boxplots for all the variables cont…



ReNewind

Univariate Analysis_histograms and boxplots for all the variables cont…



Model Performance Summary (original data)

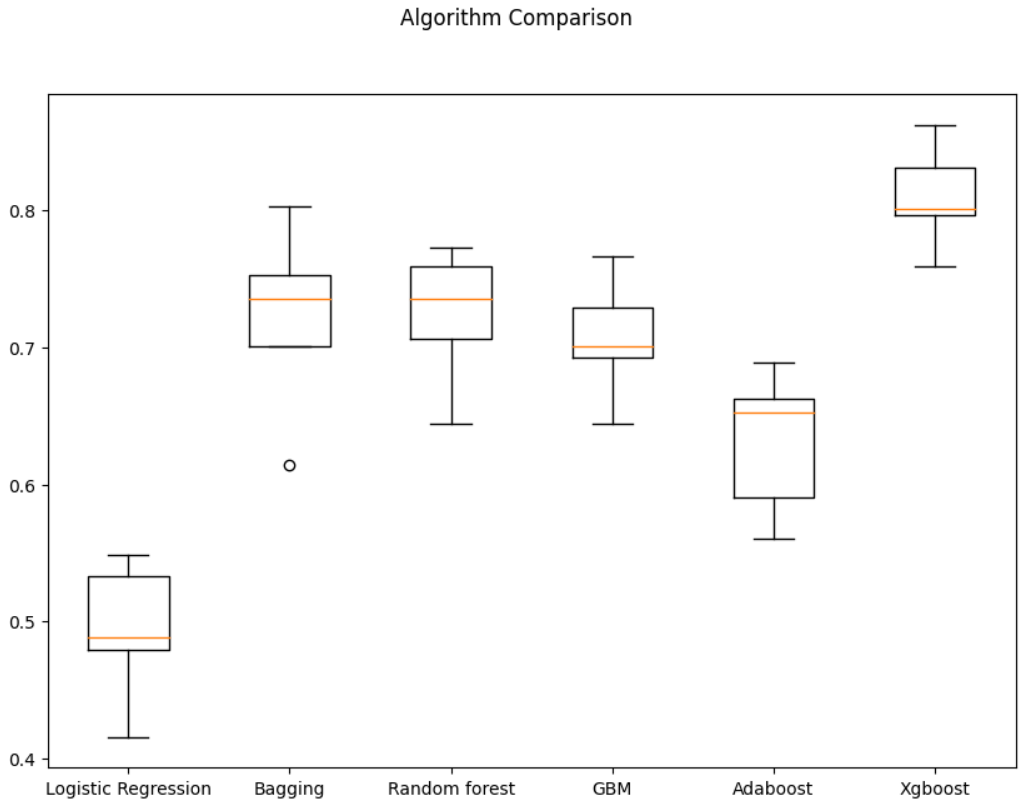
-- Used the recall score type to compare parameter combinations –

Cross-Validation Performance on Training Set	
Logistic Regression	0.4927566553639709
Bagging	0.7210807301060529
Random forest	0.7235192266070268
GBM	0.7066661857008874
Adaboost	0.6309140754635308
Xgboost	0.8100497799581561

Validation Performance	
Logistic Regression	0.48201438848920863
Bagging	0.7302158273381295
Random forest	0.7266187050359713
GBM	0.7230215827338129
Adaboost	0.6762589928057554
Xgboost	0.8309352517985612

- The cross validation performance scores are similar to the validation performance scores
- All the models appear to be suffering from overfitting except Logistic Regression
- XGBoost is giving the highest cross-validated recall followed by Random Forest and Bagging

Model Performance Summary (original data) cont...



XGBoost is demonstrating the best performance.

XGBoost, Random Forest, Bagging and GBM appear to have the highest recall scores.

Model Performance Summary (oversampled data)

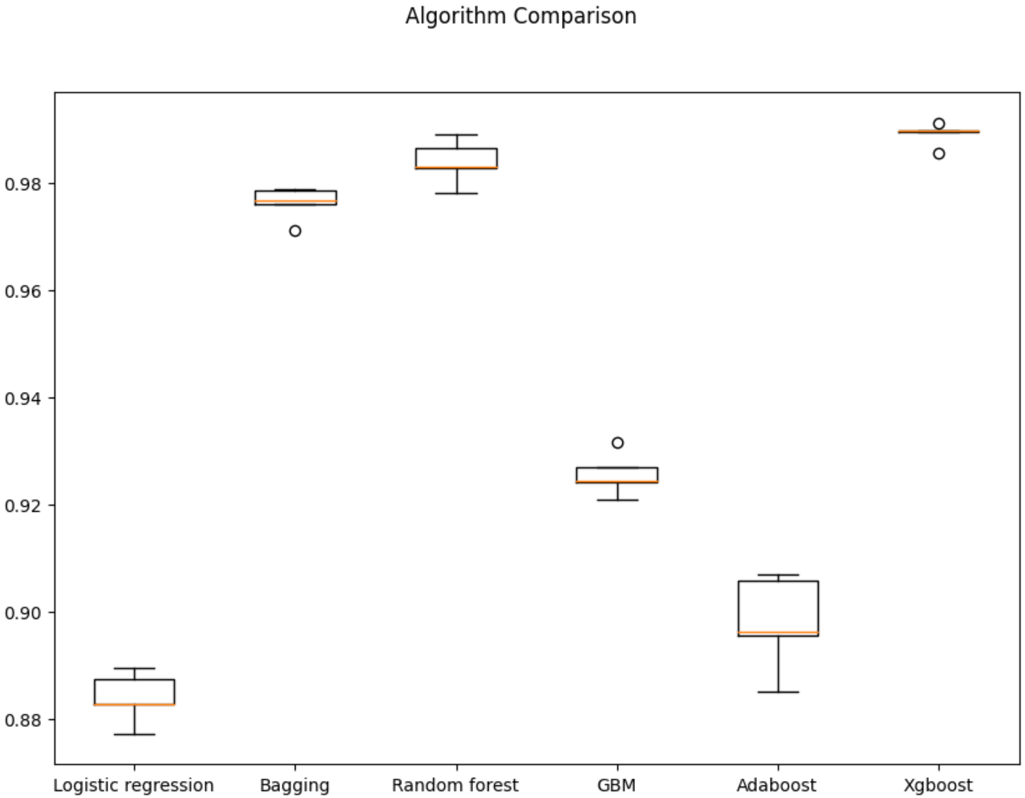
- Oversampling method chosen was the Synthetic Minority Over Sampling Technique (SMOTE)

Cross-Validation Performance on Training Set	
Logistic Regression	0.883963699328486
Bagging	0.9762141471581656
Random forest	0.9839075260047615
GBM	0.9256068151319724
Adaboost	0.8978689011775473
Xgboost	0.9891305241357218

Validation Performance	
Logistic Regression	0.8489208633093526
Bagging	0.8345323741007195
Random forest	0.8489208633093526
GBM	0.8776978417266187
Adaboost	0.8561151079136691
Xgboost	0.8669064748201439

- The cross validation performance scores are much higher than the validation performance scores
- XGBoost is giving the highest cross-validated recall followed by Random Forest and Bagging

Model Performance Summary (oversampled data) cont…



XGBoost is demonstrating the best performance.

XGBoost, Random Forest, Bagging and GBM appear to have the highest recall scores.

All scores have increased from the original data

Model Performance Summary (undersampled data)

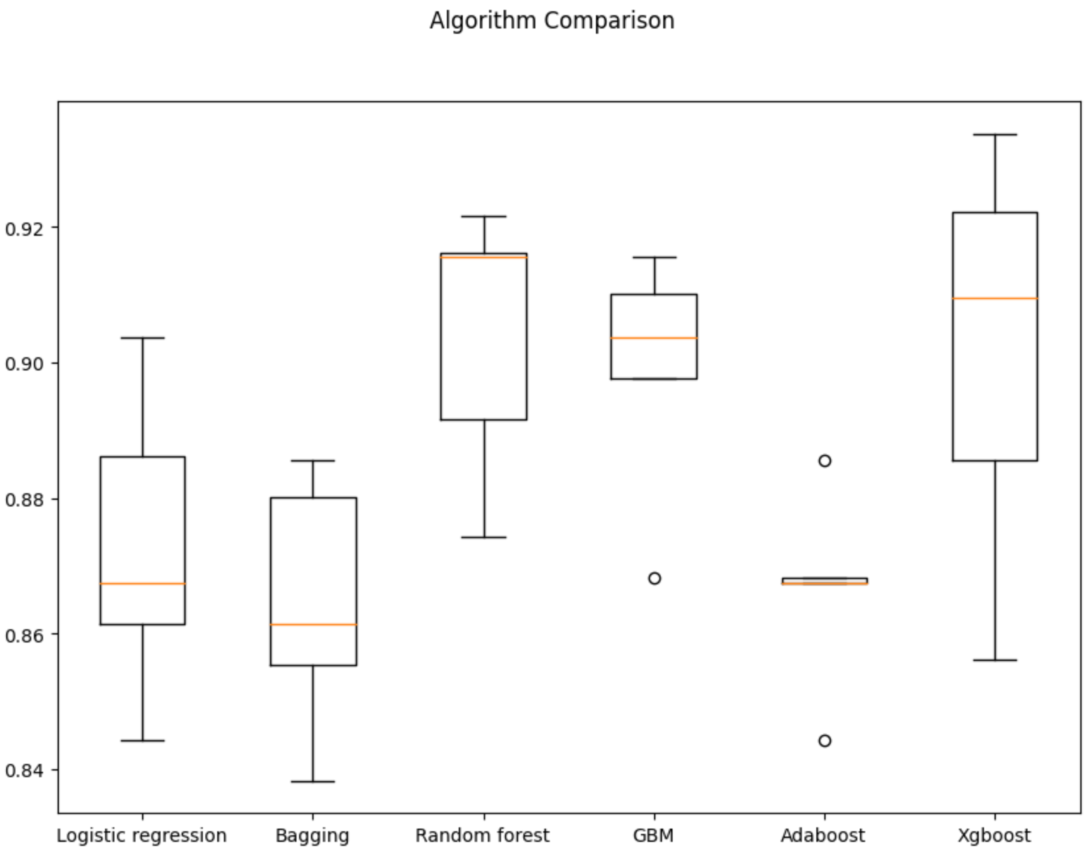
- Under sampling method chosen was Random Undersampler

Cross-Validation Performance on Training Set	
Logistic Regression	0.8726138085275232
Bagging	0.8641945025611427
Random forest	0.9038669648654498
GBM	0.8990621167303946
Adaboost	0.8666113556020489
Xgboost	0.9014717552846114

Validation Performance	
Logistic Regression	0.8525179856115108
Bagging	0.8705035971223022
Random forest	0.8920863309352518
GBM	0.8884892086330936
Adaboost	0.8489208633093526
Xgboost	0.89568345323741

- The cross validation performance scores are similar to the validation performance scores
  - Bagging shows signs of overfitting in the validation performance scores
- Random Forest is giving the highest cross-validated recall followed by XGBoost and GBM
- Undersampling improves performance

Model Performance Summary (undersampled data)



Random Forest and XGBoost are demonstrating the best performance.

XGBoost, Random Forest, Adaboost and GBM appear to have the highest recall scores.

All models demonstrate a good general performance.

Hyperparameter Tuning

ReNewind

- Four (4) models were chosen for hyperparameter tuning: AdaBoost, Random Forest, Gradient Boosting (GBM), and XGBoost – these models were choses based on test runs.

    - Random Forest was tuned using hyperparameter tuned undersampled data. It performed well on the undersampled data. Random Forest can handle imbalanced data and can perform well with little to no tuning. We used this hyperparameter tuning with undersampled data to optimize the performance.

    - AdaBoost, GDM, and XGBoost were tuned using hyperparameter tuning oversampled data as they are boosting algorithms. They perform well with a variety of datasets and several hyperparameters the can optimize their performance.

Hyperparameter Tuning _ AdaBoost (oversampled data)

Best Parameters	
n_estimators	200
learning_rate	.2
base_estimator	DecisionTreeClassifier(max_depth=3, random_state=1)} with CV score=0.9714853746337214

```
AdaBoostClassifier
AdaBoostClassifier(base_estimator=DecisionTreeClassifier(max_depth=3,
                                                         random_state=1),
                   learning_rate=0.2, n_estimators=200)
      base_estimator: DecisionTreeClassifier
      DecisionTreeClassifier(max_depth=3, random_state=1)
            DecisionTreeClassifier
      DecisionTreeClassifier(max_depth=3, random_state=1)
```

Training Performance

	Accuracy	Recall	Precision	F1
0	0.948	0.897	1.000	0.945

Validation Performance

	Accuracy	Recall	Precision	F1
0	0.985	0.755	0.977	0.852

AdaBoost performs well in precision and show a possibility of overfitting in the accuracy scores. FI and Recall scores suffer a reduction in the validation set. Recall Score reduced to 76%

Hyperparameter Tuning _ Random Forest (undersampled data)

Best Parameters	
n_estimators	300
min_samples_leaf	2
max_samples	.5
max_features	sqrt
CV score	0.8990116153235697

```
RandomForestClassifier
RandomForestClassifier(max_samples=0.5, min_samples_leaf=2, n_estimators=300,
                       random_state=1)
```

Training Performance

	Accuracy	Recall	Precision	F1
0	0.961	0.933	0.989	0.960

Validation Performance

	Accuracy	Recall	Precision	F1
0	0.938	0.885	0.468	0.612

Random Forest performs well in the training set; however Recall, Precision, and F1 scores are suffering a reduction in the validation set. Recall Score: 89%

Hyperparameter Tuning _ GBM (oversampled data)

Best Parameters	
subsample	0.7
n_estimators	125
max_features	0.5
learning_rate	1
CV score	0.9723322092856124

```
GradientBoostingClassifier
GradientBoostingClassifier(learning_rate=1, max_features=0.5, n_estimators=125,
                           random_state=1, subsample=0.7)
```

Training Performance

	Accuracy	Recall	Precision	F1
0	0.993	0.992	0.994	0.993

Validation Performance

	Accuracy	Recall	Precision	F1
0	0.969	0.856	0.678	0.757

Gradient Boosting performs well in the training set. Recall, precision and F1 scores decrease in the validation set. Recall Score: 86%

Hyperparameter Tuning _ XGBoost (oversampled data)

Best Parameters	
subsample	0.9
scale_pos_weight	10
n_estimators	150
learning_rate	0.1
gamma	0
CV score	0.9960475154078072:

```
XGBClassifier
XGBClassifier(base_score=None, booster=None, callbacks=None,
              colsample_bylevel=None, colsample_bynode=None,
              colsample_bytree=None, device=None, early_stopping_rounds=None,
              enable_categorical=False, eval_metric='logloss',
              feature_types=None, gamma=0, grow_policy=None,
              importance_type=None, interaction_constraints=None,
              learning_rate=0.1, max_bin=None, max_cat_threshold=None,
              max_cat_to_onehot=None, max_delta_step=None, max_depth=None,
              max_leaves=None, min_child_weight=None, missing=nan,
              monotone_constraints=None, multi_strategy=None, n_estimators=150,
              n_jobs=None, num_parallel_tree=None, random_state=1, ...)
```

Training Performance

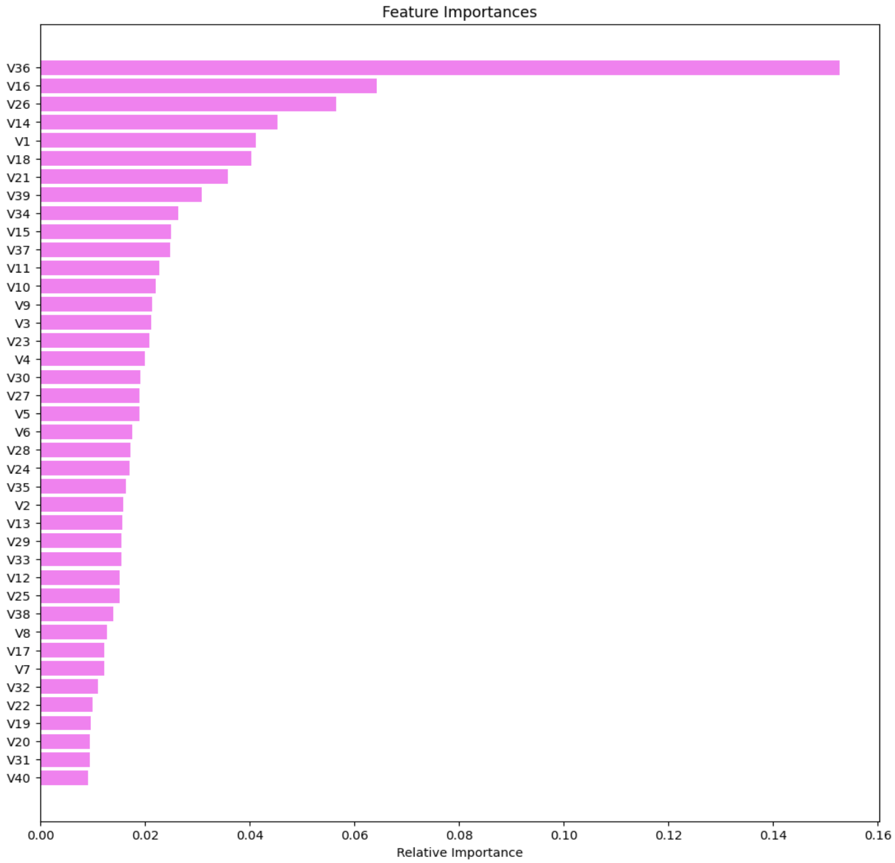
	Accuracy	Recall	Precision	F1
0	0.994	1.000	0.988	0.994

Validation Performance

	Accuracy	Recall	Precision	F1
0	0.967	0.892	0.648	0.750

XGBoost consistently performs well in the training set. There is reduction in recall, precision and F1 scores in the validation set. Recall Score: 89%

Feature Importance _ Building Best Model



V36 is the dominate feature that holds a very high level of importance in the model prediction.

V16 and V26 features follow in importance.

Pipeline _ Build Final Model

Best Parameters	
subsample	0.9
scale_pos_weight	10
n_estimators	150
learning_rate	0.1
gamma	0
CV score	0.9960475154078072:

```
Pipeline
SimpleImputer
SimpleImputer(strategy='median')
StandardScaler
StandardScaler()
XGBClassifier
XGBClassifier(base_score=None, booster=None, callbacks=None,
              colsample_bylevel=None, colsample_bynode=None,
              colsample_bytree=None, device=None, early_stopping_rounds=None,
              enable_categorical=False, eval_metric='logloss',
              feature_types=None, gamma=0, grow_policy=None,
              importance_type=None, interaction_constraints=None,
              learning_rate=0.1, max_bin=None, max_cat_threshold=None,
              max_cat_to_onehot=None, max_delta_step=None, max_depth=None,
              max_leaves=None, min_child_weight=None, missing=nan,
              monotone_constraints=None, multi_strategy=None, n_estimators=150,
              n_jobs=None, num_parallel_tree=None, random_state=1, ...)
```

Oversampling method chosen was the Synthetic Minority Over Sampling Technique (SMOTE)

Test Performance

Accuracy	Recall	Precision	F1
0.958	0.858	0.585	0.695

The final model appears to be performing well with the XGBoost Classifier. Recall Score: 86% - similar to the test recall score of 86% and validation recall score of 89%.