

Using the Varuse Classification model and implementing GA for the Feature Selection, Imbalance Multiclass Data

MPLS Flap route cause data was collected from Telecom various service raw data and used to develop a route that alternately promotes a destination network via one route to another. Pathological condition causes route flapping.

In this study, we examined and collected data from around 54,081 occurrences with seven features, the majority of which were continues numerical features and a few nominal categorical features. This data set contained three class labels: **Flap**, **Short Cut**, and **Long Cut**. I used the **OneHotEncoding** feature engineering method for all category features and the **StandardScaler** feature engineering method for the remaining numerical features.

The learning method in a classification model shows the underlying link between the features and labeled variables and identifies the model that best fits the training data. Classifiers do not function well on skewed datasets. In the end, the majority class or classes will categorize correctly than the minority class or classes.

Imbalanced data is defined as having a significantly greater number of observations in favor of one class than the other (s). This throws the model off balance. This is known as the **Class imbalance problem**. **Synthetic Minority Oversampling (SMOTE)**, which we are using in this project, is one method for dealing with imbalanced datasets. **SMOTE** is an algorithm that generates fresh sample data by combining the closest minority cases into synthetic examples.

With SMOTE, we have performing classifying techniques such as Logistic Regression, Random Forest, and GXBoost.

Logistic regression, is restricted to two-class classification issues by default. Some extensions, such as one-vs-rest, can allow logistic regression to be utilized for multi-class classification problems by first transforming the classification problem into several binary classification problems.

Random Forest can be applied to classification as well as regression difficulties. Random forest is a type of tree-based ensemble learning. It is a technique for supervised machine learning that is used for classification and regression applications. The random forest algorithm, as the name implies, builds a forest with many decision trees and averages predictions across many individual trees. It functions as a black box program. Random forest performs well with numerical and categorical data.

The **GXBoost algorithm** is a gradient boosting framework-based decision-tree-based ensemble Machine Learning technique. Artificial neural networks surpass all other algorithms or frameworks in prediction issues involving unstructured data (pictures, text, etc.). However, decision tree-based algorithms are now considered best-in-class for small-to-medium structured/tabular data.

1. Clean Data (Data preprocessing)

Data was cleansed in this project to remove duplicates and missing data. We had 43,239 occurrences with 7 attributes after cleaning the data.


1.1 Categorical Data Encoding Techniques

Data Encoding is an important pre-processing step in Machine Learning. It refers to the process of converting categorical or textual data into numerical format, so that it can be used as input for algorithms to process. The reason for encoding is that most machine learning algorithms work with numbers and not with text or categorical variables.

There are several methods for encoding categorical variables, including

- One-Hot Encoding
- Dummy Encoding
- Ordinal Encoding
- Binary Encoding
- Count Encoding
- Target Encoding

In this projects we used Target Encoding and One-Hot Encoding techniques. Example for Target Encoding



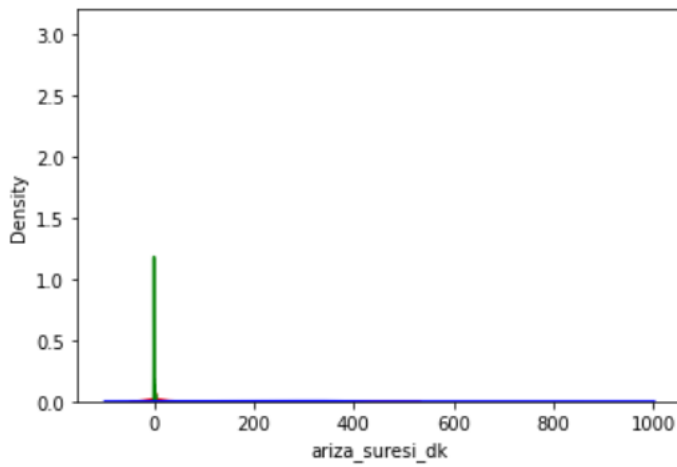
	olasi_neden	ticket_tipi_label		olasi_neden	mean_encoded
0		0		0	0.518610
1		2		1	0.775078
2		0		2	0.831917
3		1		3	0.338073
4		0		4	0.838323

As most of the machine learning algorithms recognise only numbers. Therefore, all non-numeric ordinal variables needs to be transformed into numeric. This can be achieved using a dictionary using Python's map function.

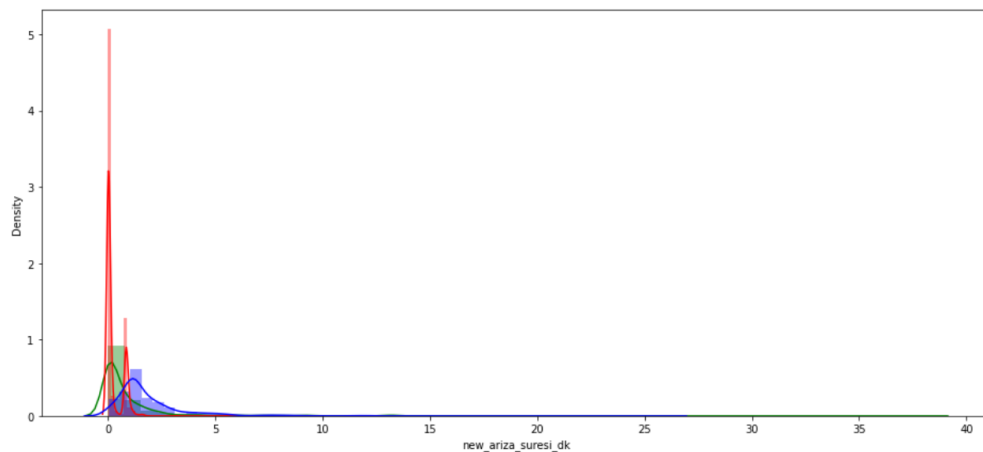
Before Starting with any modelling, one must first check problem context and try to get as much information possible. For the given dataset we are provided with average training score. But, before we make any comments on this feature's importance we should take a step back and think. How a ticket_tipi would be effect in a multistate and multi olasi_neden in our flap correlation relationship with envanter data and alarm data.

Here some analysis result before work on feature extraction on dataset.

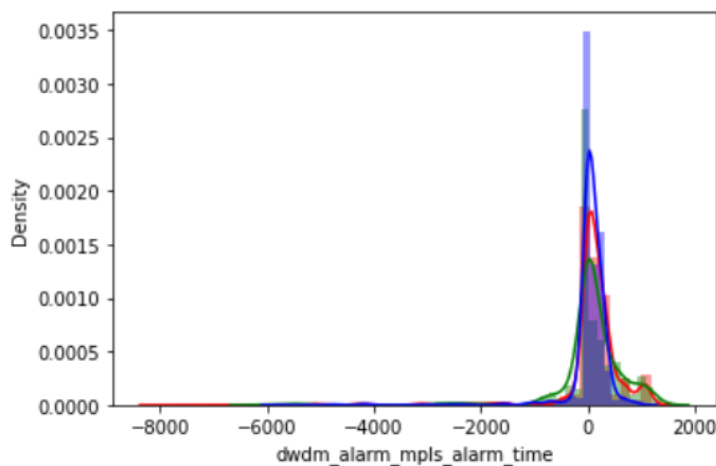
- I. Between Ticket_tipi with ariza_suresi_dk (target with olasi_neden and bord_type categorical value)



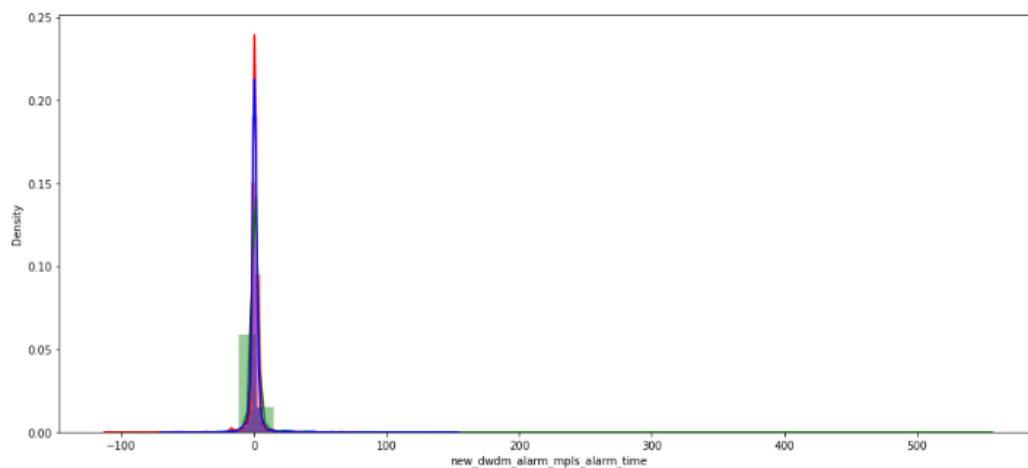
After impilment target encoding This is a more advanced encoding technique used for dealing with high cardinality categorical features, i.e., features with many unique categories. The average target value for each category is calculated and this average value is used to replace the categorical feature.



- II. Between Ticket_tipi with dwdm_alarm_mpls_alarm_time (target with olasi_neden and bord_type categorical value)



After impiliment target encoding



2. Data Exploration

```
<class 'pandas.core.frame.DataFrame'>
```

```
Int64Index: 43239 entries, 0 to 46008
```

```
Data columns (total 8 columns):
```

#	Column	Non-Null Count	Dtype
0		43239 non-null	object
1		43239 non-null	float64
2		43239 non-null	float64
3		43239 non-null	float64
4		43239 non-null	float64
5		43239 non-null	object
6		43239 non-null	object
7		43239 non-null	object

```
dtypes: float64(4), object(4)
```

```
memory usage: 3.0+ MB
```

Figure 1. analysis data information

	port_speed_value	ariza_suresi_dk	dwdm_alarm_mpls_alarm_time	toplaml_fibercizimuzunlugu
count	43239.000000	43239.000000	43239.000000	4.323900e+04
mean	35.518471	70.520966	-1610.443615	7.524669e+05
std	40.704043	131.967669	4828.858080	8.028658e+05
min	0.000000	0.050000	-21012.000000	-1.000000e+00
25%	10.000000	0.266667	-284.641667	2.399615e+05
50%	10.000000	2.050000	8.633333	6.024999e+05
75%	100.000000	83.150000	203.425000	8.240175e+05
max	100.000000	1411.950000	1338.400000	4.812263e+06

Figure 2. analysis data description

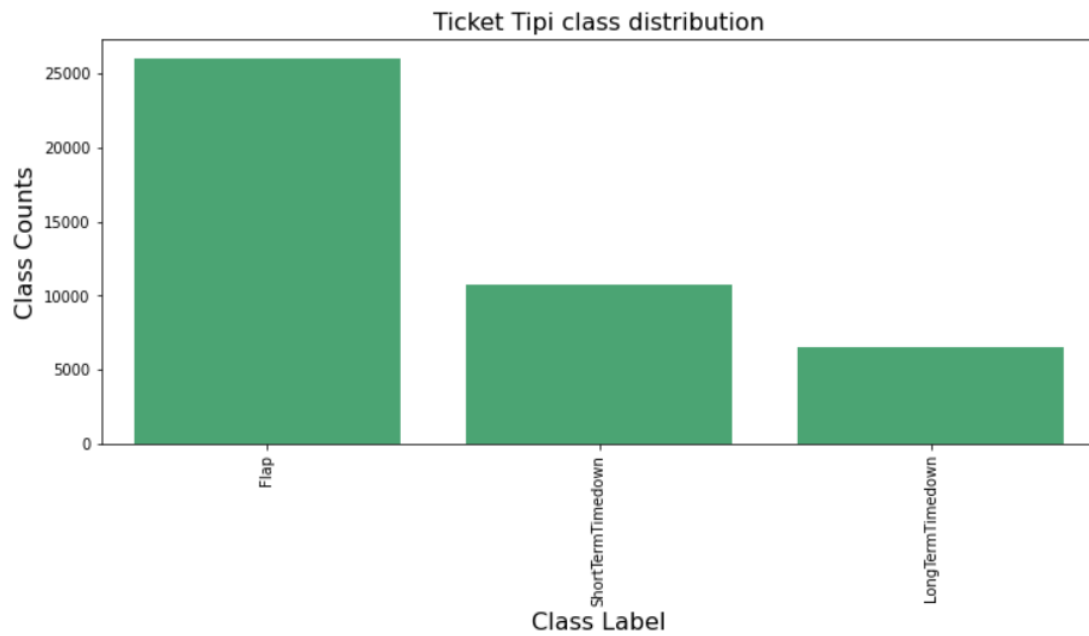


Figure 3. Class Labels count

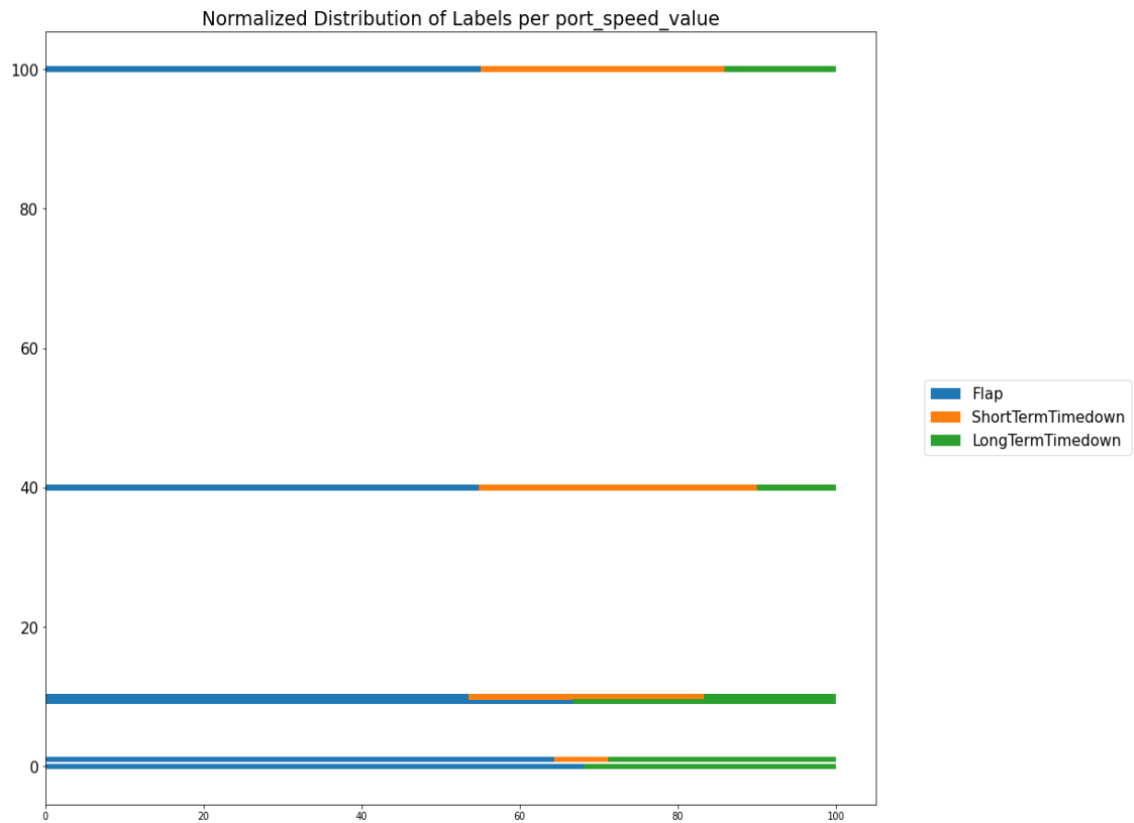


Figure 4. Normalization Distribution of labels per port speed value

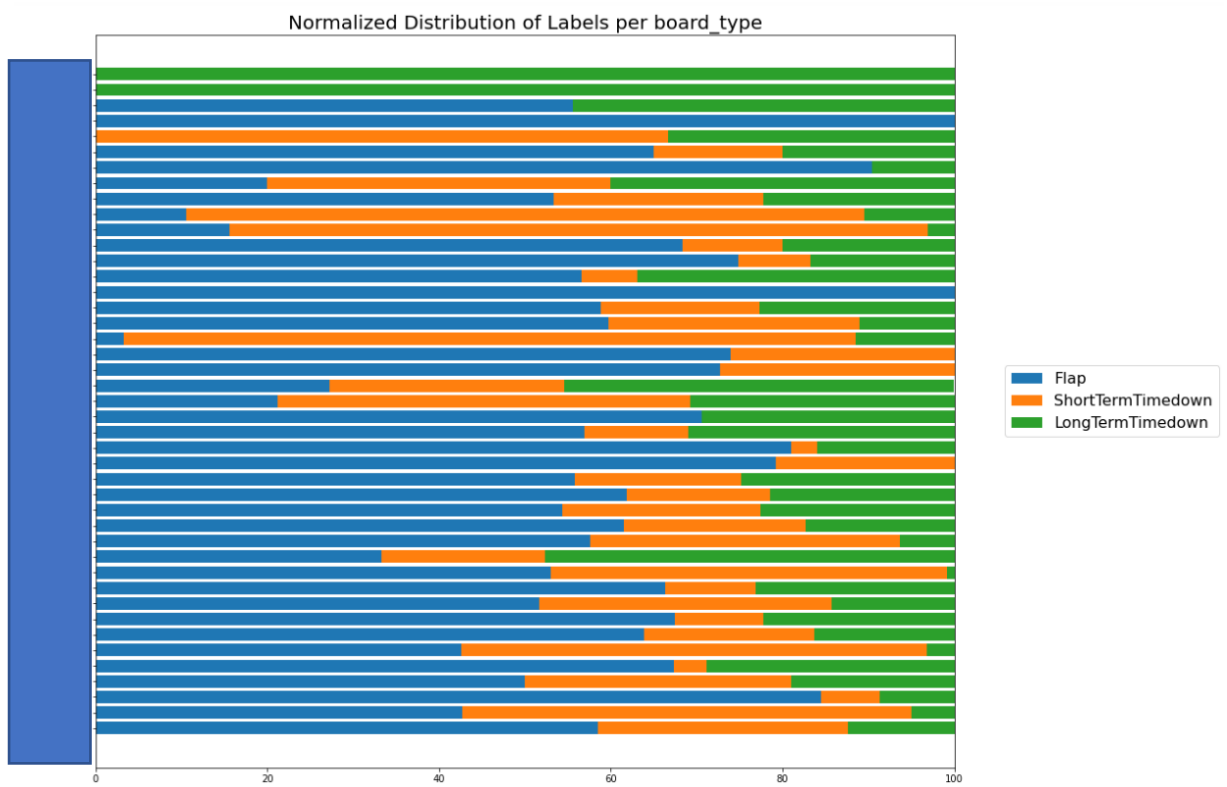


Figure 5. Normalization Distribution of labels per board type

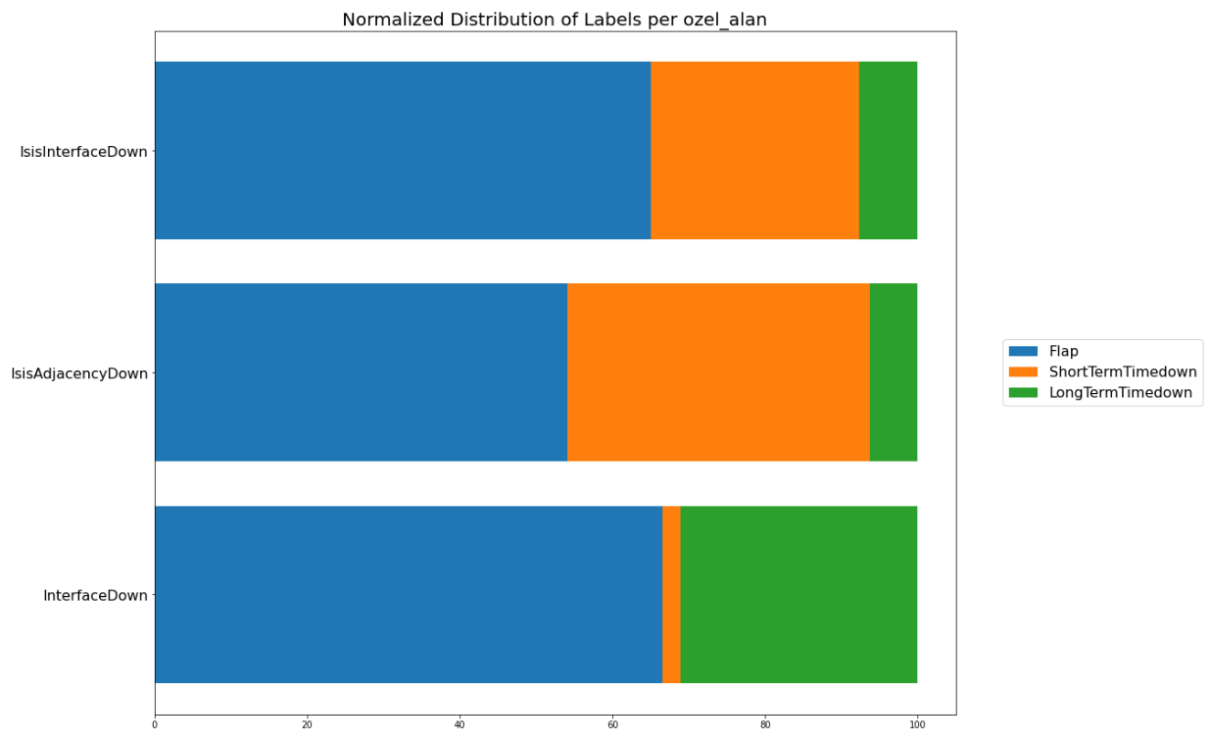


Figure 6. Normalization Distribution of labels per oze| alan

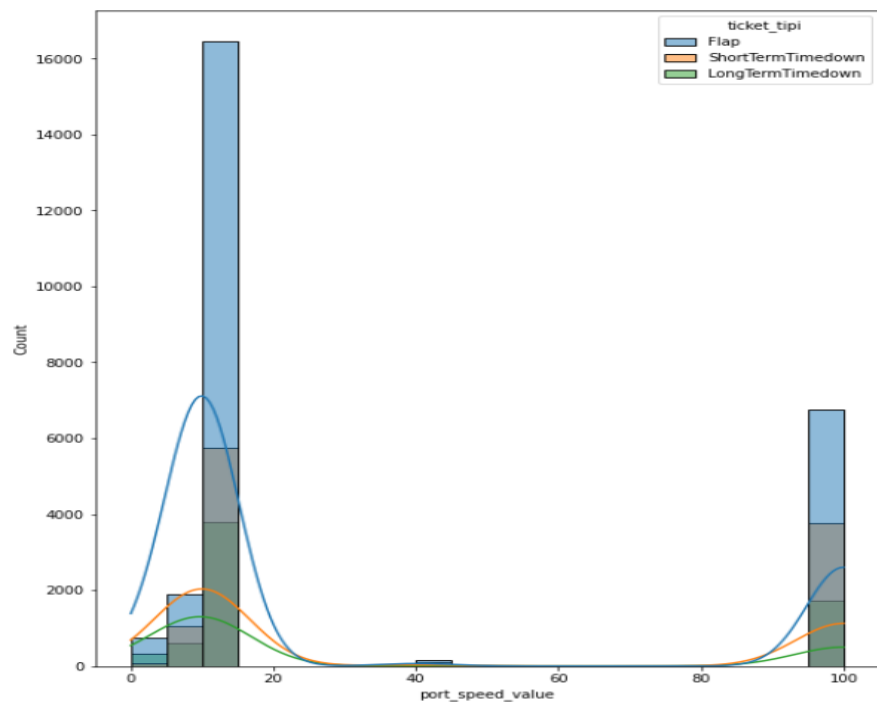


Figure 7. Histogram plot for port speed value feature

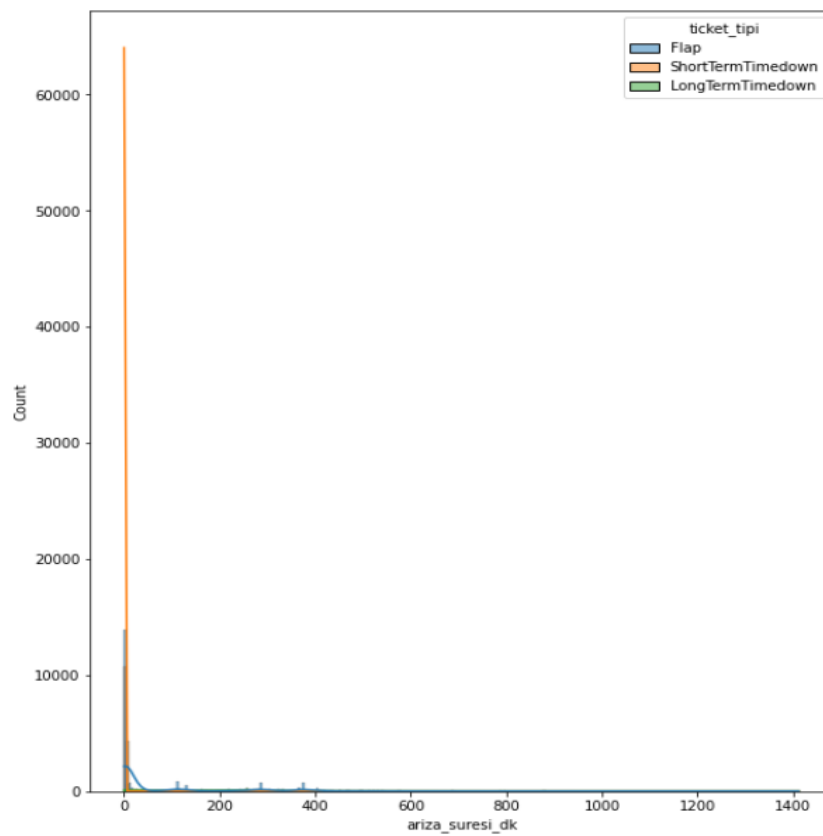


Figure 8. Histogram plot for ariza suresi dk features

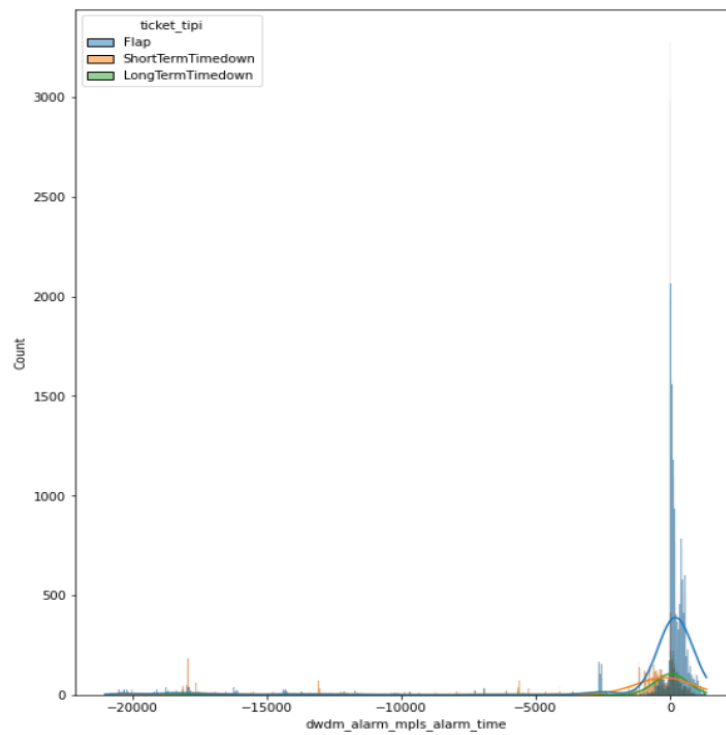


Figure 9. Histogram plot for dwdm alarm mpls alarm time features

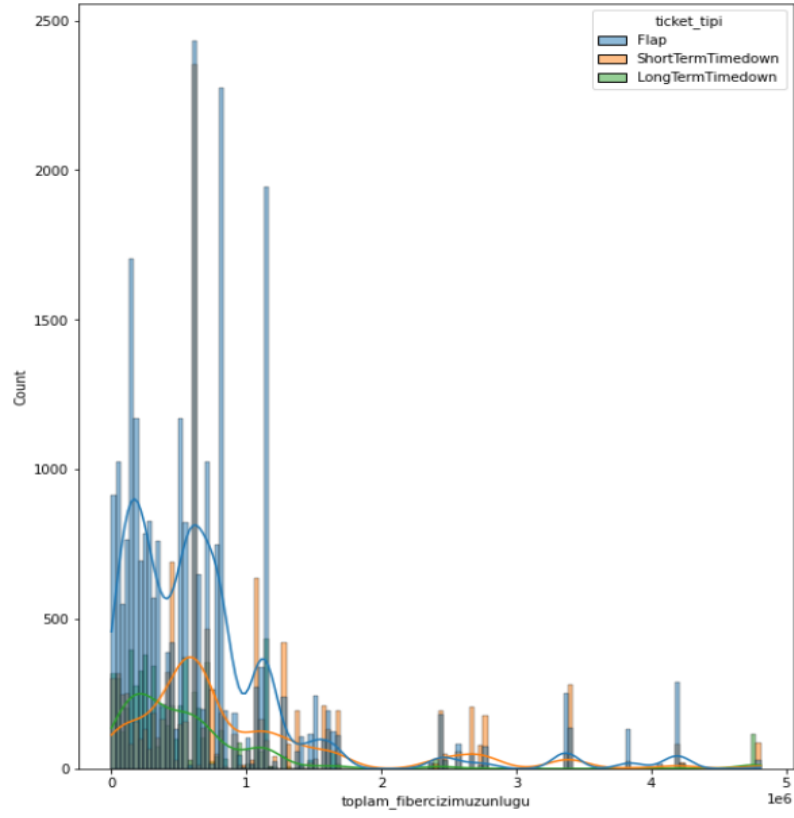
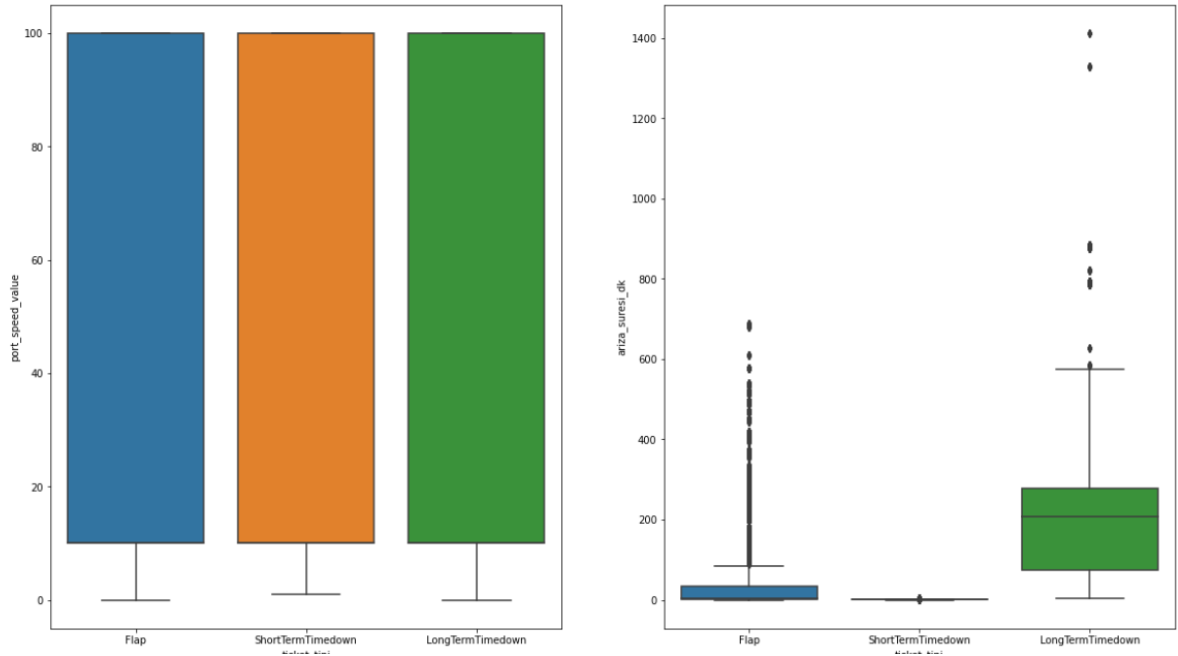


Figure 10. Histogram plot for toplam fiber cizim uzunlugu



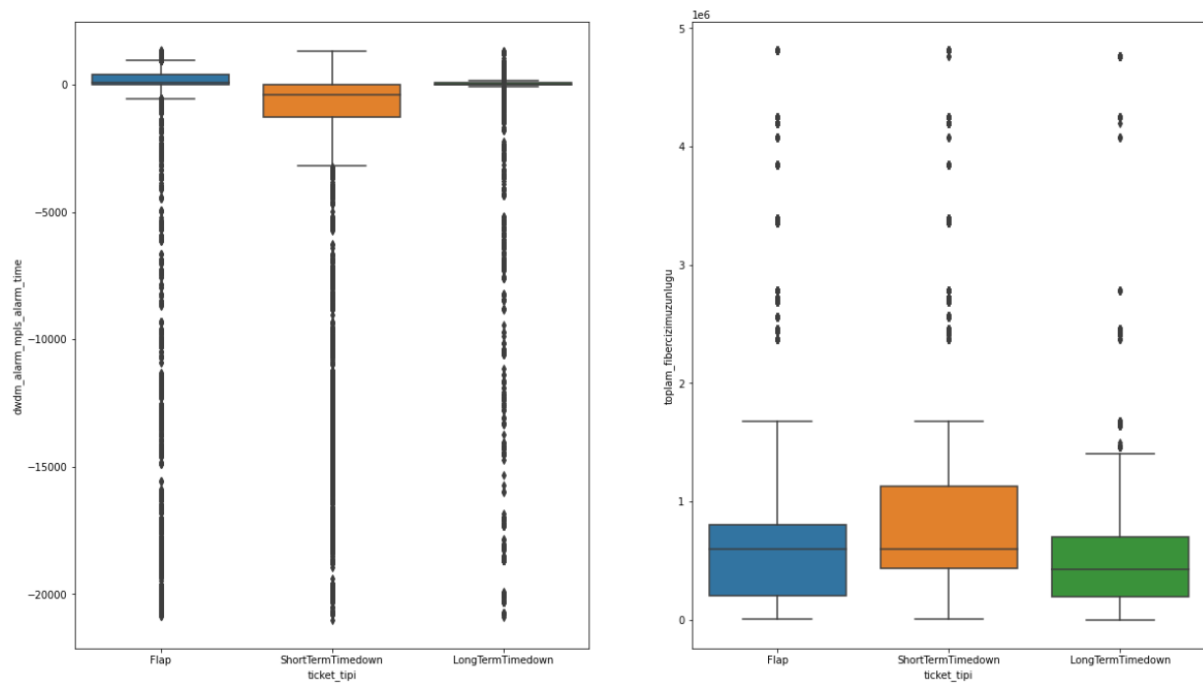


Figure 11. features boxplot

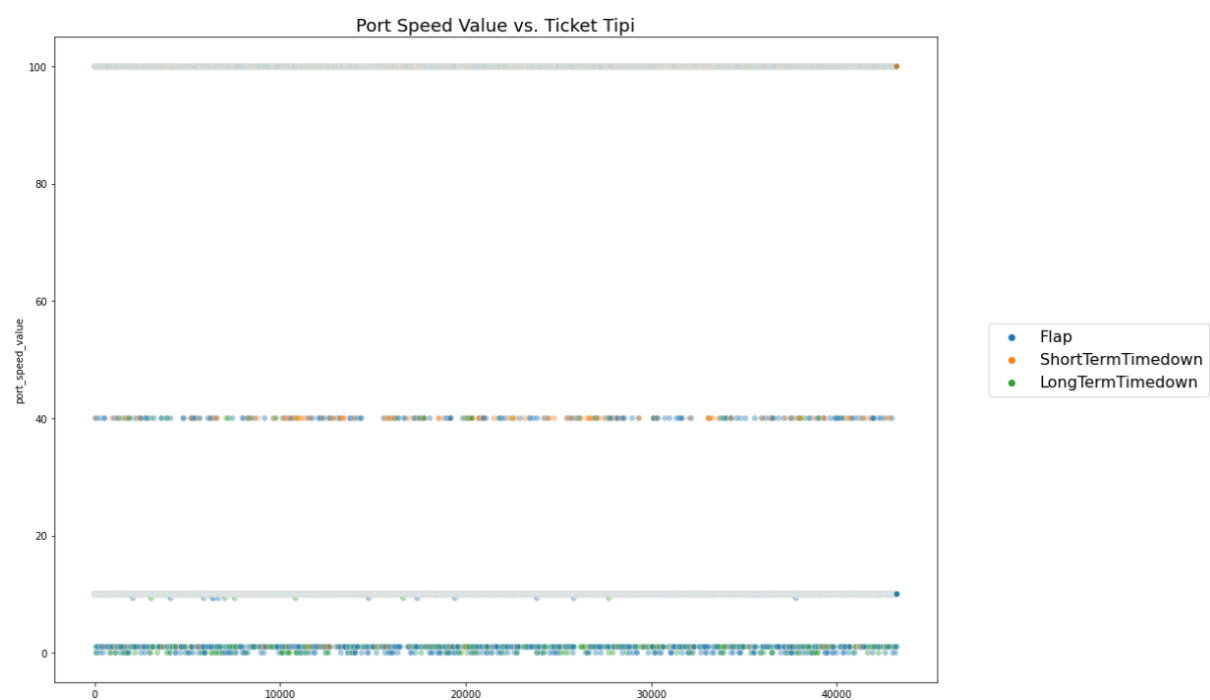


Figure 12. port speed value vs. ticket tipi

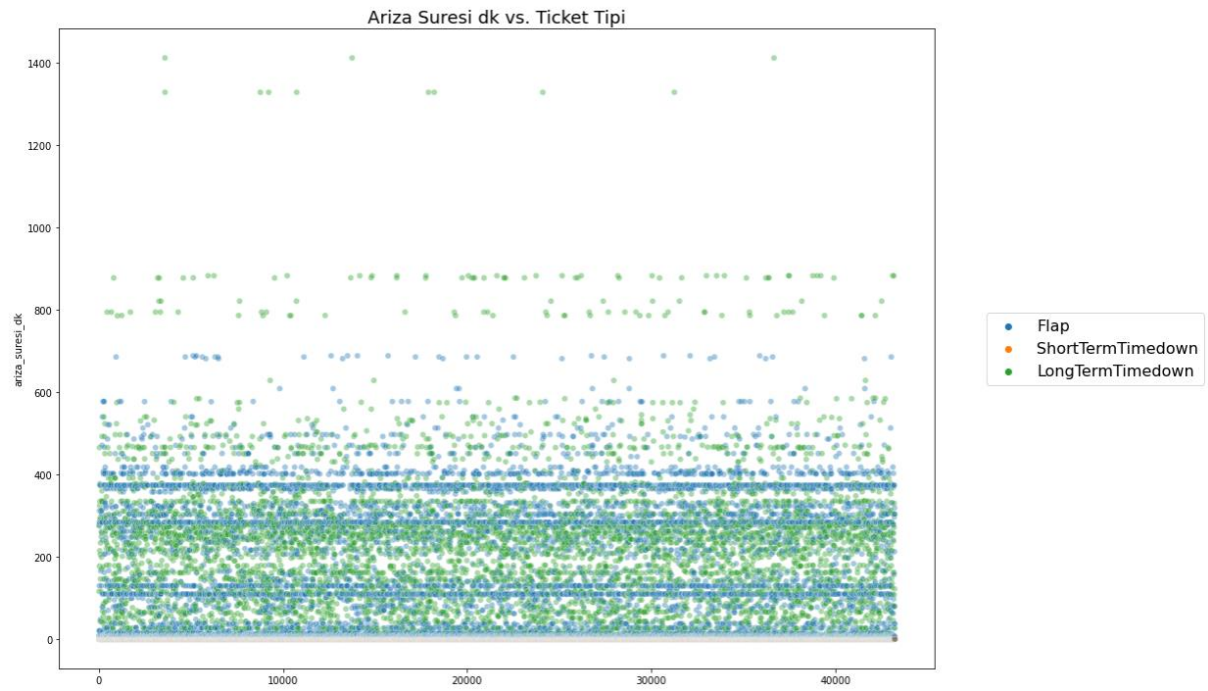


Figure 13. ariza suresi dk value vs. ticket tipi

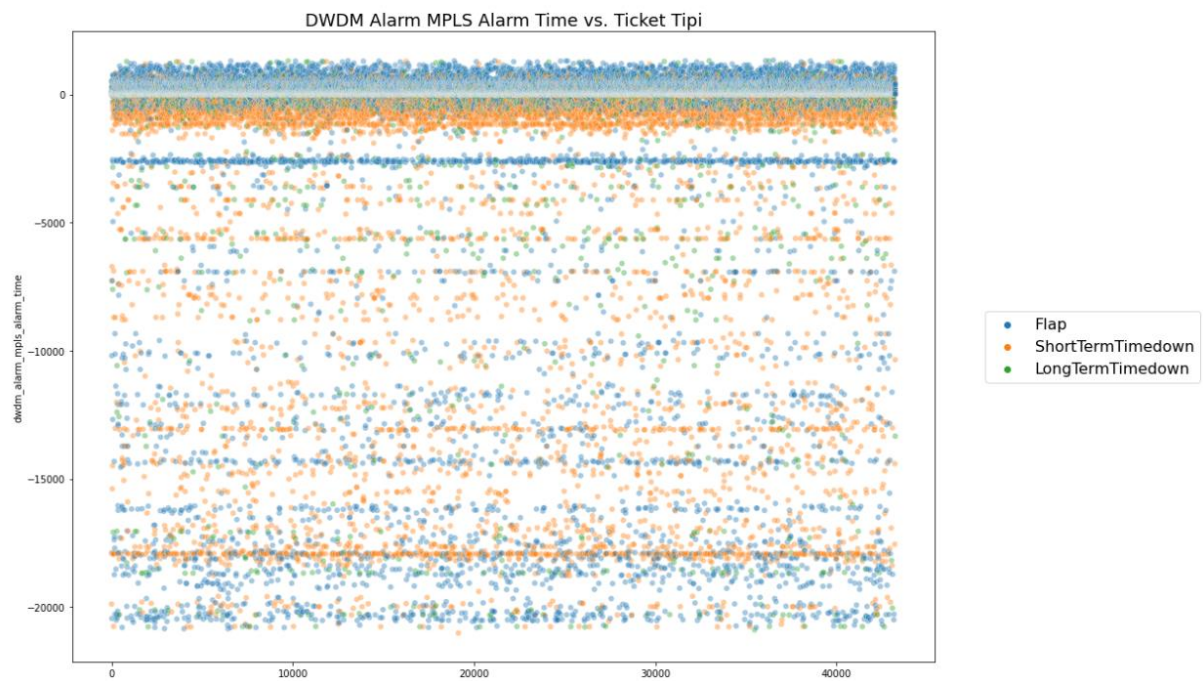


Figure 14. DWDM Alarm MPLS Alarm time value vs. ticket tipi

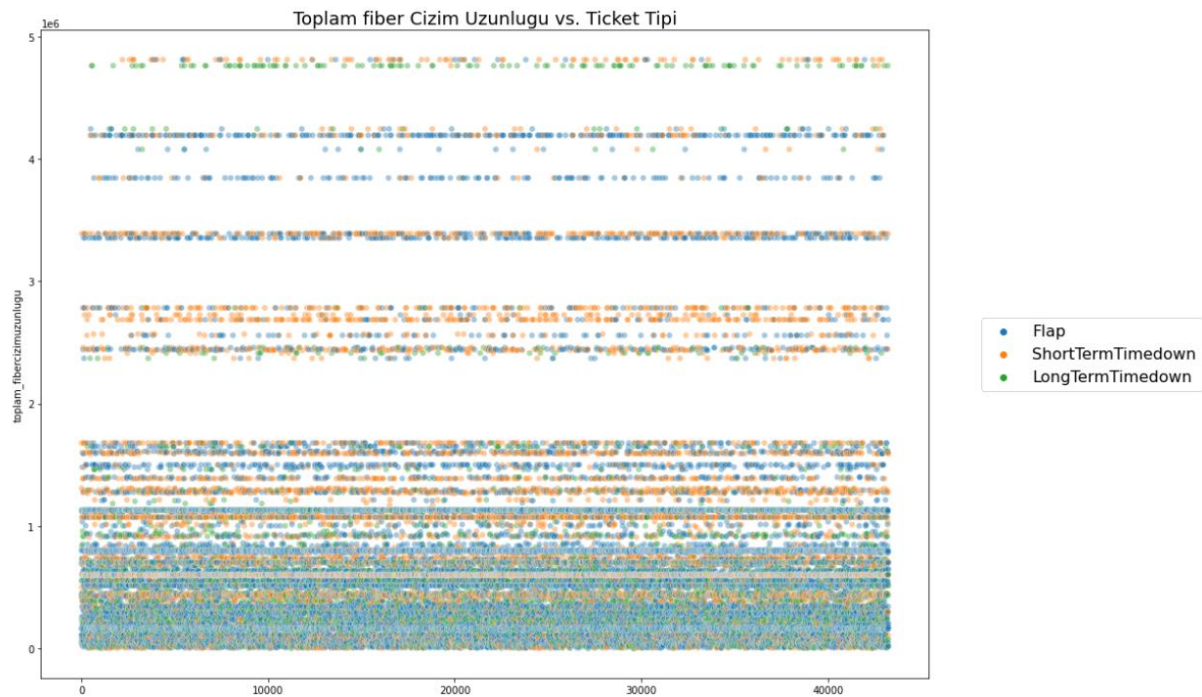


Figure 15. Toplam Fiber Cizim Uzunlugu value vs. ticket tipi

3. Feature Selection and Feature Extraction

Both feature selection and feature extraction are used for **dimensionality reduction** which is key to **reducing model complexity** and **overfitting**.

We identify the best feature subset during feature selection that contributes the most to our forecasted variable. The feature set has a significant impact on the computational power of machine learning models. Retaining the important elements reduces learning time and increases accuracy.

Here have two major data

Advantages of feature selection:

1. Reduce overfitting of the data to improve the generalization of models.
2. Remove unnecessary/redundant data.
3. Curtail the Curse of Dimensionality
4. Optimize training time

- **Genetic Algorithm for Feature selection**

A search and optimization method based on the theory of natural evolution is known as a genetic algorithm. By altering a group of people known as a population and then randomly choosing parents from this population to carry out reproduction in the form of mutation and crossover, the algorithm attempts to "mimic" the concept of human evolution. This

procedure keeps on till the halting criterion is satisfied. In the end, it provides the ideal person or solution.

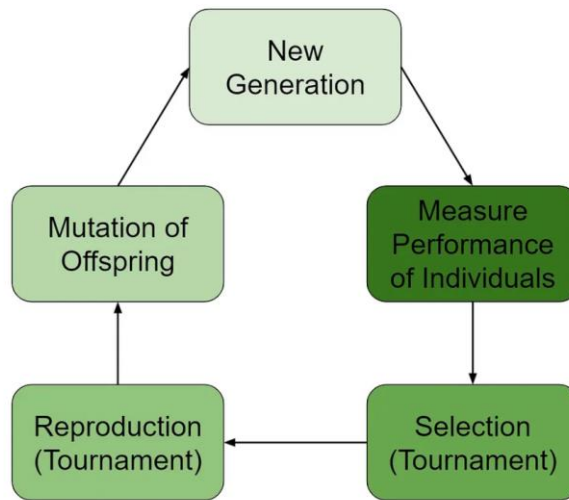


Figure 16. Genetic Algorithm workflow

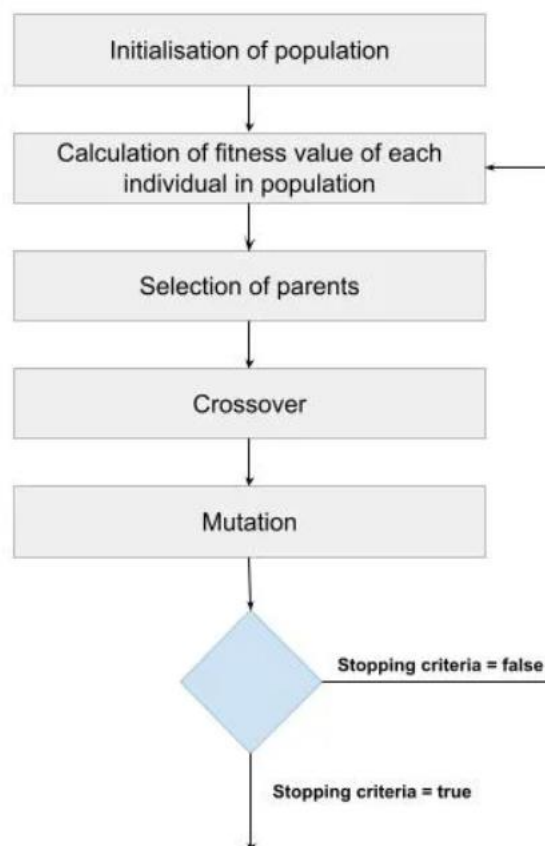


Figure 17. Genetic Algorithm Workflow

- **Function description in genetic algorithm:**
 1. **initilization_of_population()**: To initialize a random population.
 2. **fitness_score()**: Returns the best parents along with their score.
 3. **selection()**: Selection of the best parents.
 4. **crossover()**: Picks half of the first parent and half of the second parent.
 5. **mutation()**: Randomly flips selected bits from the crossover child.
 6. **generations()**: Executes all the above functions for the specified number of generations

- **GA work steps in Flap Correlation research:**
 1. run a function to initialize a random population.
 2. The randomized population is now run through the fitness function, which returns the best parents (highest accuracy).
 3. Selection from these best parents will occur depending on the n-parent parameter.
 4. After doing the same, it will be put through the crossover and mutation functions respectively.
 5. Cross over is created by combining genes from the two fittest parents by randomly picking a part of the first parent and a part of the second parent.
 6. The mutation is achieved by randomly flipping selected bits for the crossover child.
 7. A new generation is created by selecting the fittest parents from the previous generation and applying cross-over and mutation.¶
 8. This process is repeated for n number of generations.

4. Classification Models Analysis

Multiclass classification is a popular problem in supervised machine learning.

Problem – Given a dataset of m training examples, each of which contains information in the form of various features and a label. Each label corresponds to a class, to which the training example belongs. In multiclass classification, we have a finite set of classes. Each training example also has n features.

Aim of this section – We will use different multiclass classification methods such as, Random Forest Classification, Decision Trees Classification, XGBoost Classification, Naïve Bayes Classification, and K-Nearest Neighbors Classification KNN, etc. We will compare their accuracy on test data.

Approach – Load dataset from the source. Split the dataset into “training” and “test” data. Train RF, Decision tree, XGBoost, NB, and KNN classifiers on the training data. Use the above classifiers to predict labels for the test data. Measure accuracy and visualize classification.

4.1 Random Forest Classifier for Multi-class Imbalance Data

Here are some benefits of Random Forest that we can list.:

- Random Forest can be used for both classification and regression problems.
- Random Forest is a transparent machine learning methodology that we can see and interpret what's going on inside of the algorithm. Not just use it as a black box application.
- Random Forest works well with both categorical and numerical (continuous) features. You don't need to categorize (bucketize) numerical features before you use it. It supports these types of features at the same time.

Since Random Forest is a rule-based algorithm, you also don't need to normalize numerical features like we do for machine learning algorithms with linear separator.

- **Classification Report RF**

- Train part

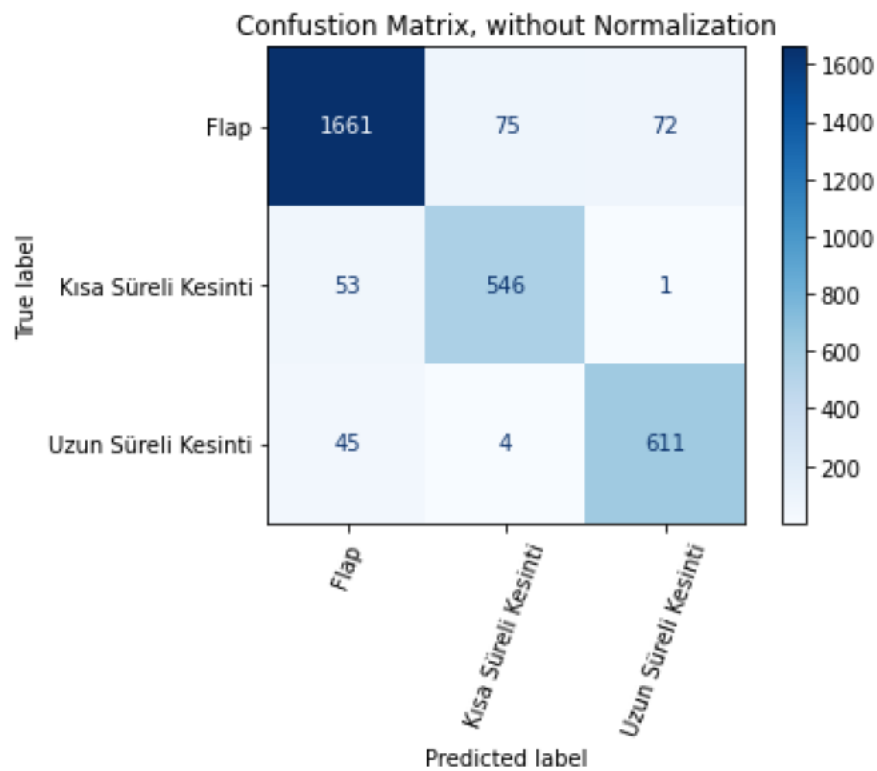
	precision	recall	f1-score	support
Flap	1.00	1.00	1.00	5238
Kısa Süreli Kesinti	1.00	1.00	1.00	1752
Uzun Süreli Kesinti	1.00	1.00	1.00	2213
accuracy			1.00	9203
macro avg	1.00	1.00	1.00	9203
weighted avg	1.00	1.00	1.00	9203

- Test part

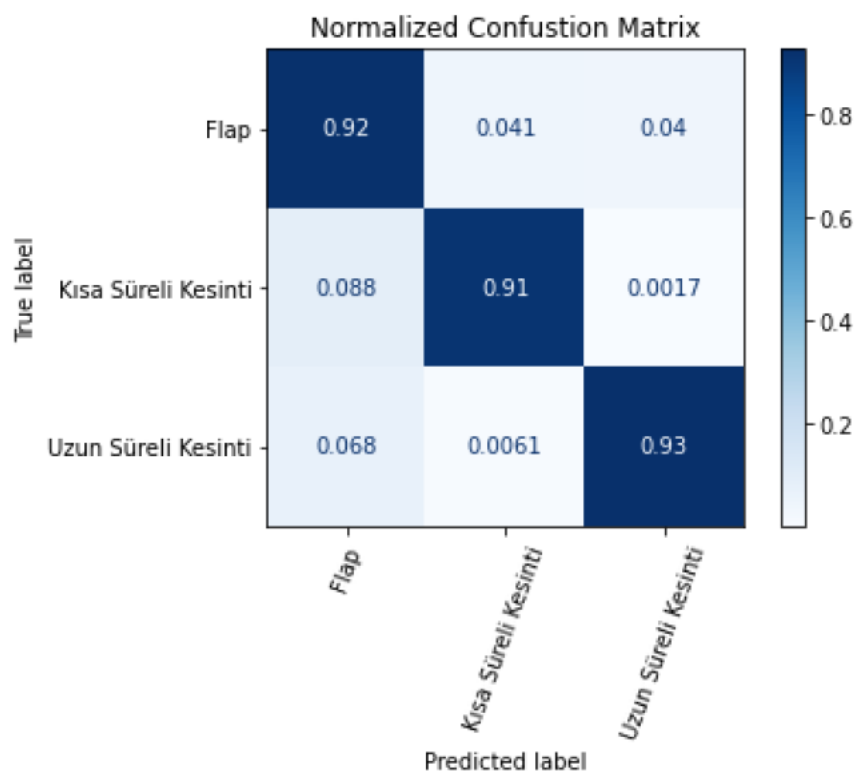
	precision	recall	f1-score	support
Flap	0.94	0.92	0.93	1808
Kısa Süreli Kesinti	0.87	0.91	0.89	600
Uzun Süreli Kesinti	0.89	0.93	0.91	660
accuracy			0.92	3068
macro avg	0.90	0.92	0.91	3068
weighted avg	0.92	0.92	0.92	3068

- **Confusion Matrices RF**

- Confusion Matrix, without Normalization



- Confusion Matrix Normalization

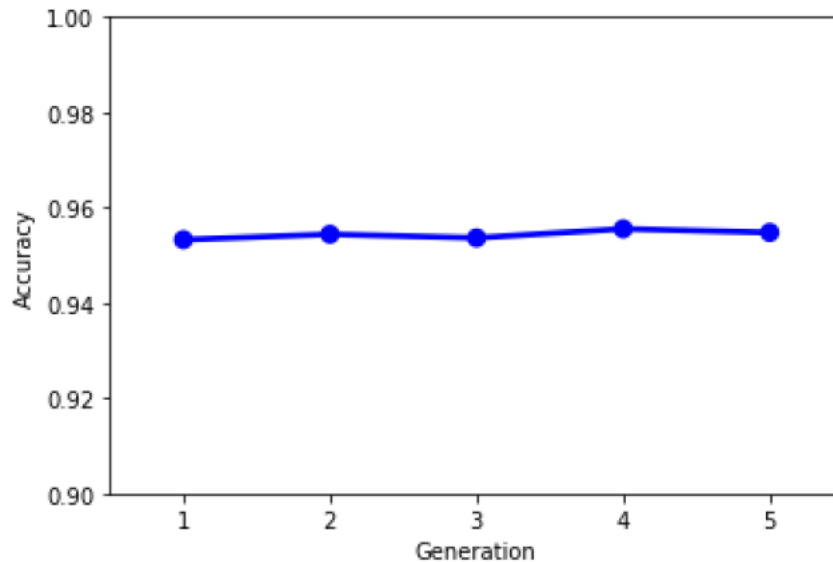


- Compute Multiclass AUC Score for RF

0.9866129843414235

- **SMOTE+GA RF Classifier**

Best score in generation 1 : [0.9532639545884579]
 Best score in generation 2 : [0.954399243140965]
 Best score in generation 3 : [0.9536423841059603]
 Best score in generation 4 : [0.9555345316934721]
 Best score in generation 5 : [0.9547776726584674]



4.2 Decision Tree Classifier for Multi-class Imbalance Data

Nodes and branches make up decision trees. The nodes can also be divided into a root node (the tree's first node), decision nodes (sub-nodes that branch out based on circumstances), and leaf nodes (nodes that stay put). Every node utilizes one and only one independent variable to split into two or more branches since the decision tree has an if-then structure. The independent variable may be continuous or categorical. To decide whether to split a node for categorical variables, the categories are used; for continuous variables, the algorithm generates numerous threshold values that serve as the decision-maker.

- **Classification Report DT**

- Train part

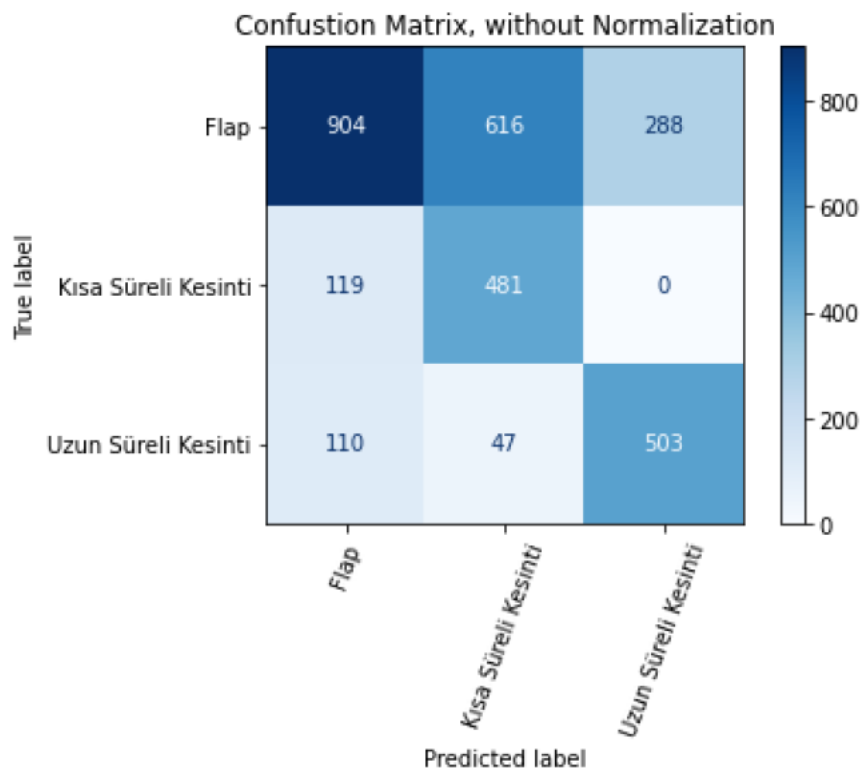
	precision	recall	f1-score	support
Flap	0.79	0.49	0.61	5238
Kısa Süreli Kesinti	0.41	0.79	0.54	1752
Uzun Süreli Kesinti	0.67	0.79	0.73	2213
accuracy			0.62	9203
macro avg	0.63	0.69	0.63	9203
weighted avg	0.69	0.62	0.63	9203

- Test part

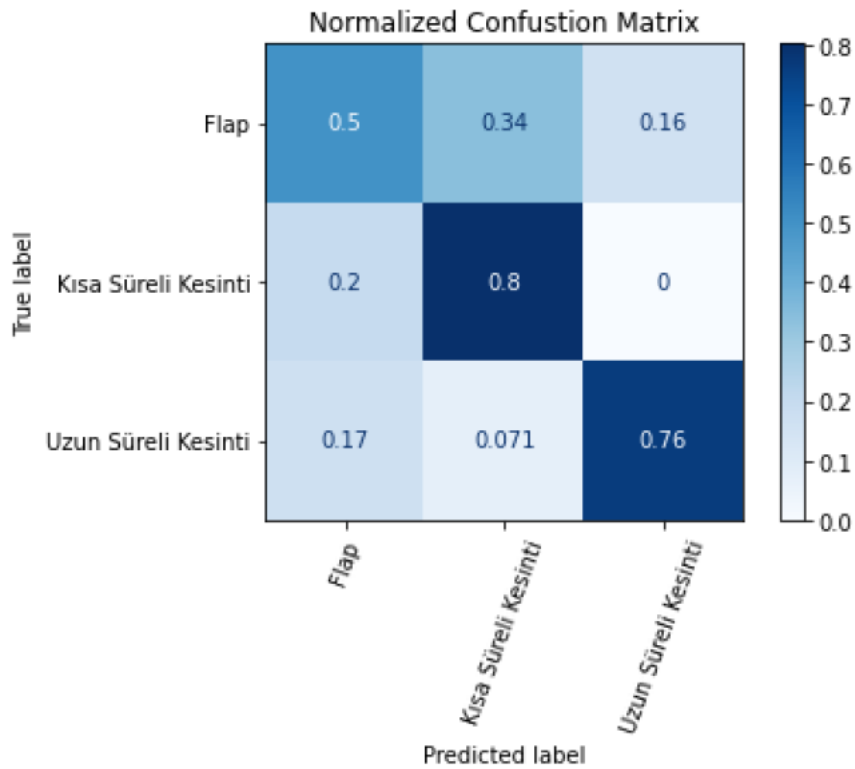
	precision	recall	f1-score	support
Flap	0.80	0.50	0.61	1808
Kısa Süreli Kesinti	0.42	0.80	0.55	600
Uzun Süreli Kesinti	0.64	0.76	0.69	660
accuracy			0.62	3068
macro avg	0.62	0.69	0.62	3068
weighted avg	0.69	0.62	0.62	3068

- **Confusion Matrices DT**

- Confusion Matrix, without Normalization



- Confusion Matrix Normalization

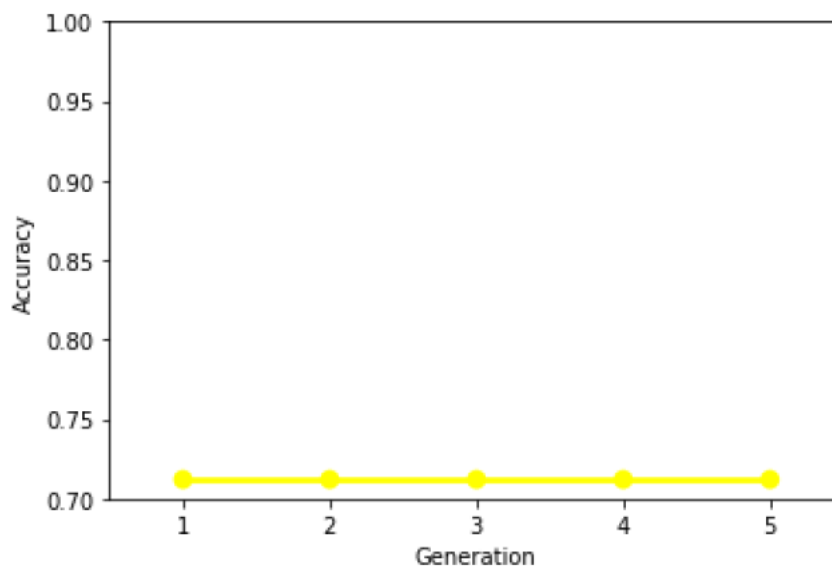


- Compute Multiclass AUC Score for DT

0.8113506127829071

- SMOTE+GA DT Classifier

Best score in generation 1 : [0.7120151371807001]
 Best score in generation 2 : [0.7120151371807001]
 Best score in generation 3 : [0.7120151371807001]
 Best score in generation 4 : [0.7120151371807001]
 Best score in generation 5 : [0.7120151371807001]



4.3 XGBoost Classifier for Multi-class Imbalance Data

A gradient boosting framework is used by the ensemble machine learning method XGBoost, which is decision-tree based. Artificial neural networks frequently outperform all other algorithms or frameworks in prediction issues involving unstructured data (pictures, text, etc.). However, decision tree-based algorithms are currently thought to be best-in-class for small- to medium-sized structured/tabular data.

A number of cutting-edge features for model tuning, computing environments, and algorithm improvement are provided by the XGBoost implementation. Gradient Boosting (GB), Stochastic Gradient Boosting (SGB), and Regularized Gradient Boosting (RGB) are the three primary types of gradient boosting that it can execute. It is also strong enough to handle fine tweaking and the inclusion of regularization parameters.

- **Classification Report XGB**

- Train part

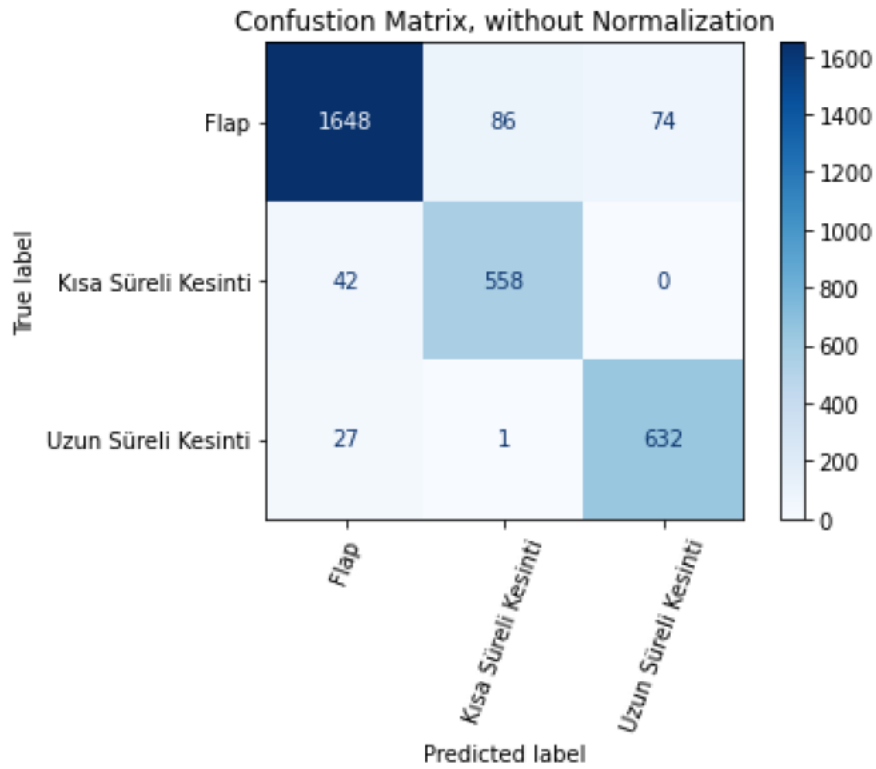
	precision	recall	f1-score	support
Flap	0.99	0.97	0.98	5238
Kısa Süreli Kesinti	0.94	0.99	0.96	1752
Uzun Süreli Kesinti	0.97	1.00	0.98	2213
accuracy			0.98	9203
macro avg	0.97	0.98	0.98	9203
weighted avg	0.98	0.98	0.98	9203

- Test part

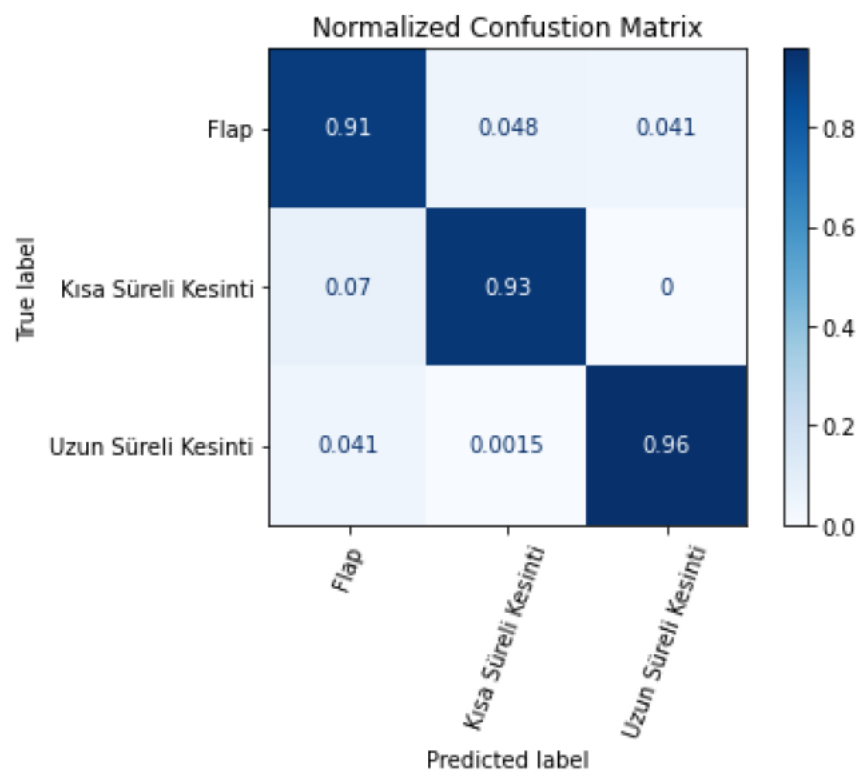
	precision	recall	f1-score	support
Flap	0.96	0.91	0.94	1808
Kısa Süreli Kesinti	0.87	0.93	0.90	600
Uzun Süreli Kesinti	0.90	0.96	0.93	660
accuracy			0.93	3068
macro avg	0.91	0.93	0.92	3068
weighted avg	0.93	0.93	0.93	3068

- **Confusion Matrices XGB**

- Confusion Matrix, without Normalization



- Confusion Matrix Normalization



- Compute Multiclass AUC Score for XGB

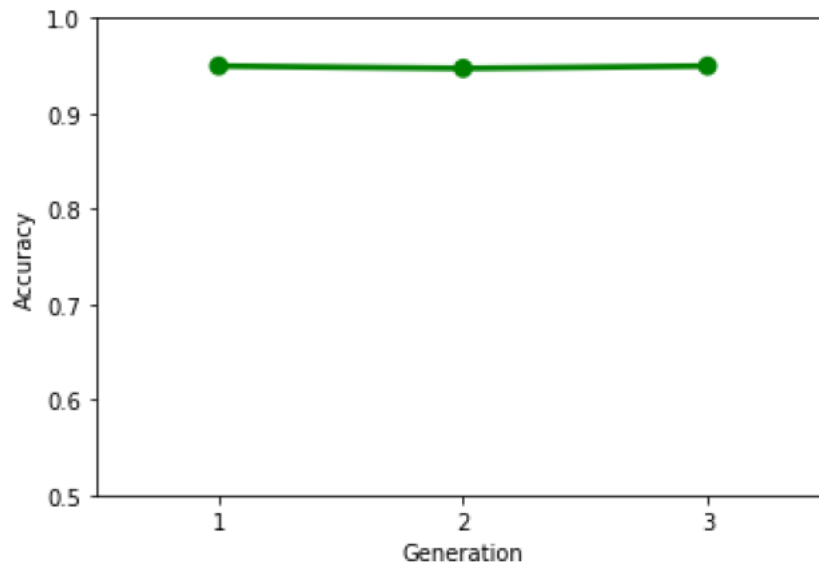
0.9881381986614

- **SMOTE+GA XGB Classifier**

Best score in generation 1 : [0.9496688741721855]

Best score in generation 2 : [0.94720908230842]

Best score in generation 3 : [0.9496688741721855]



4.4 Naïve Bayes Classifier for Multi-class Imbalance Data

A naive classifier is a classification algorithm with no logic that provides a baseline of performance on a classification dataset.

It is important to establish a baseline in performance for a classification dataset. It provides a line in the sand by which all other algorithms can be compared. An algorithm that achieves a score below a naive classification model has no skill on the dataset, whereas an algorithm that achieves a score above that of a naive classification model has some skill on the dataset.

Sklearn has GaussianNB, MultinomialNB, CategoricalNB, BernoulliNB → Given data that has categories, numerical, binary features. We calculate Mean and variance for each feature based on class and apply GaussianNB for project analysis.

- **Classification Report NB**

- Train part

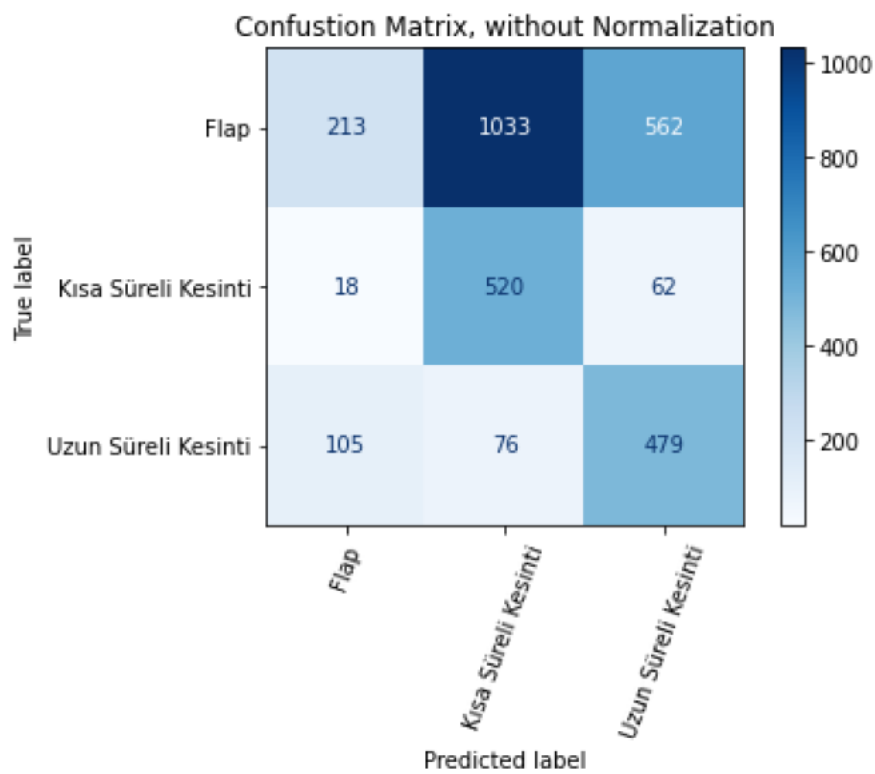
	precision	recall	f1-score	support
Flap	0.60	0.12	0.19	5238
Kısa Süreli Kesinti	0.32	0.88	0.47	1752
Uzun Süreli Kesinti	0.47	0.74	0.58	2213
accuracy			0.41	9203
macro avg	0.47	0.58	0.42	9203
weighted avg	0.52	0.41	0.34	9203

- Test part

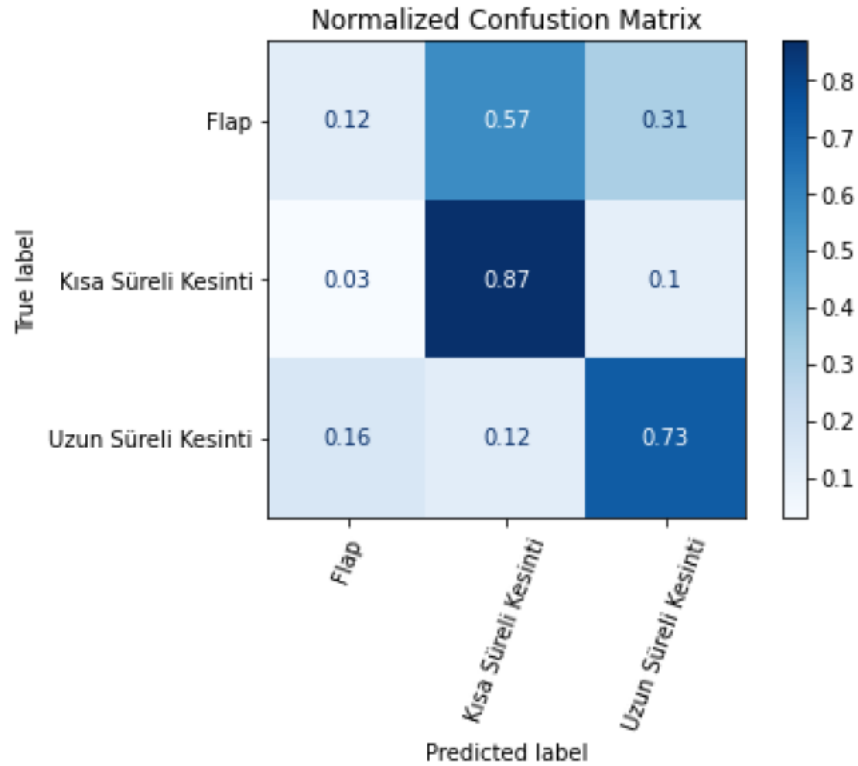
	precision	recall	f1-score	support
Flap	0.63	0.12	0.20	1808
Kısa Süreli Kesinti	0.32	0.87	0.47	600
Uzun Süreli Kesinti	0.43	0.73	0.54	660
accuracy			0.40	3068
macro avg	0.46	0.57	0.40	3068
weighted avg	0.53	0.40	0.33	3068

- **Confusion Matrices NB**

- Confusion Matrix, without Normalization



- Confusion Matrix Normalization

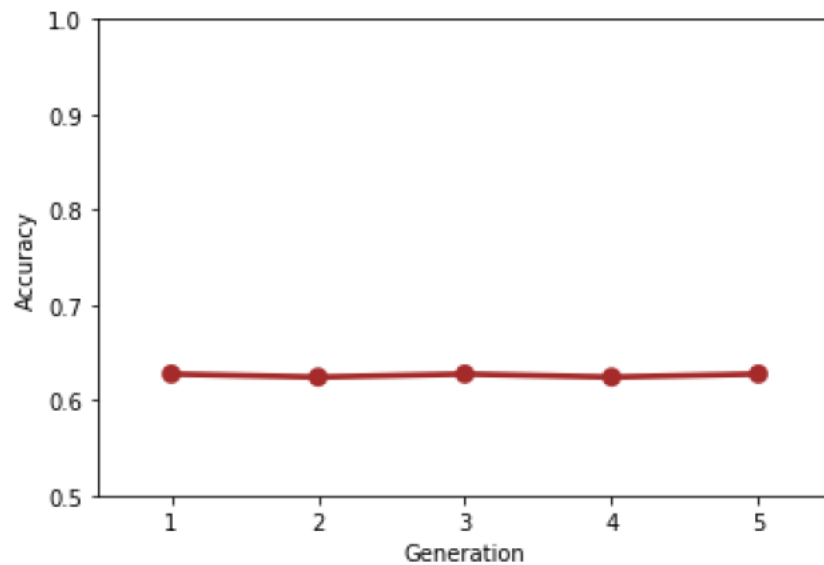


- Compute Multiclass AUC Score for NB

0.7451126927347526

- SMOTE+GA NB Classifier

Best score in generation 1 : [0.6274361400189215]
 Best score in generation 2 : [0.6242194891201513]
 Best score in generation 3 : [0.6274361400189215]
 Best score in generation 4 : [0.6242194891201513]
 Best score in generation 5 : [0.6274361400189215]



4.5 K-Nearest Neighbors Classifier for Multi-class Imbalance Data

In general, the k-nearest neighbor algorithm has no issues with unbalanced classes.

The algorithm will not favor anyone based on class size because it is unaffected by it in any manner. The outlier will typically receive its own class if you try to run k-means with an obvious outlier and k+1.

It is a non-parametric classification algorithm that does not require training. As the name suggests, it finds the “k” nearest data points for a given unknown data point, and the class which is dominant within those “k” neighbors is predicted as the output class. To find the neighbors, it uses distance metrics like Euclidean distance and Manhattan distance. You can find the optimal value for k using hyperparameter tuning. This algorithm is not very practical for cases where class distribution is skewed, or the selection of the k parameter is incorrect.

- **Classification Report KNN**

- Train part

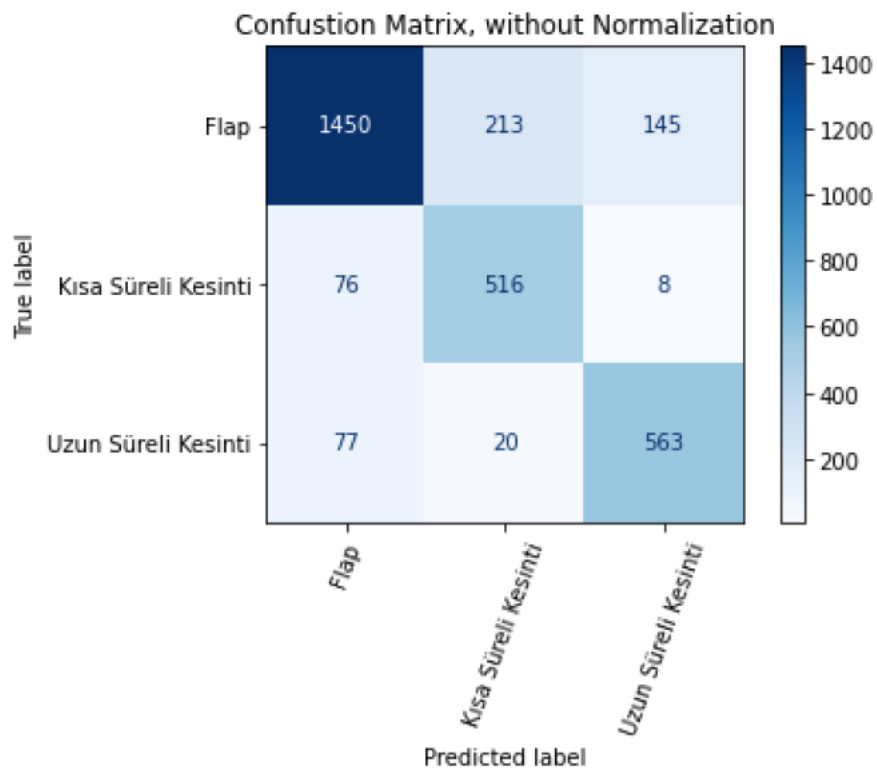
	precision	recall	f1-score	support
Flap	0.96	0.87	0.91	5238
Kısa Süreli Kesinti	0.80	0.95	0.87	1752
Uzun Süreli Kesinti	0.88	0.95	0.91	2213
accuracy			0.90	9203
macro avg	0.88	0.92	0.90	9203
weighted avg	0.91	0.90	0.91	9203

- Test part

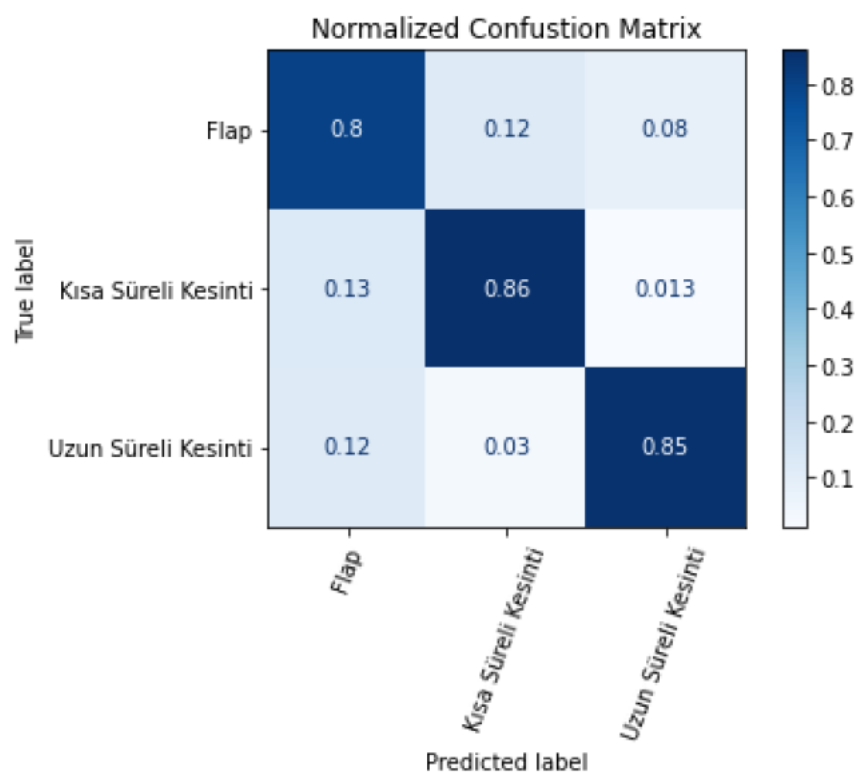
	precision	recall	f1-score	support
Flap	0.90	0.80	0.85	1808
Kısa Süreli Kesinti	0.69	0.86	0.77	600
Uzun Süreli Kesinti	0.79	0.85	0.82	660
accuracy			0.82	3068
macro avg	0.79	0.84	0.81	3068
weighted avg	0.84	0.82	0.83	3068

- **Confusion Matrices KNN**

- Confusion Matrix, without Normalization



- Confusion Matrix Normalization

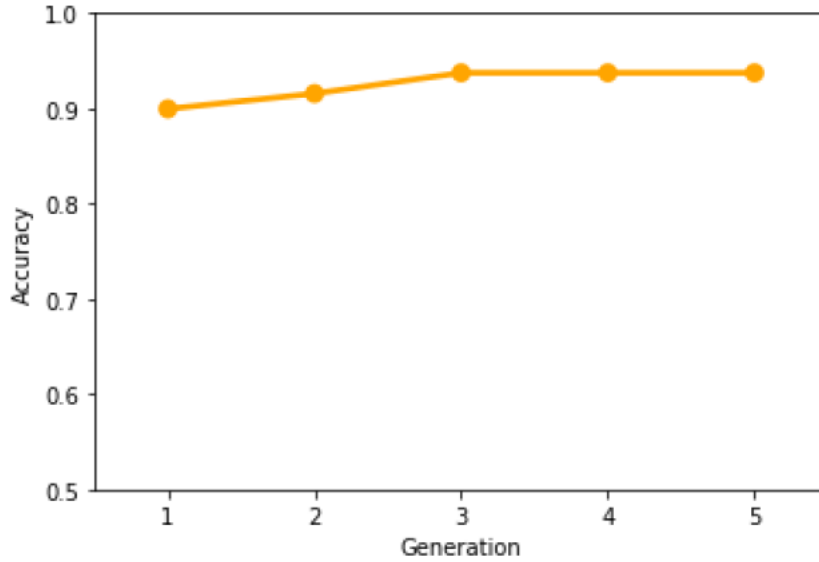


- Compute Multiclass AUC Score for KNN

0.9399177612952463

- **SMOTE+GA KNN Classifier**

Best score in generation 1 : [0.8995269631031221]
 Best score in generation 2 : [0.9156102175969726]
 Best score in generation 3 : [0.9373699148533585]
 Best score in generation 4 : [0.9373699148533585]
 Best score in generation 5 : [0.9373699148533585]



5. Result Outputs

	Recall			F1-Score			
Model	Flap	Kısa Süreli Kesinti	Uzun Süreli Kesinti	Flap	Kısa Süreli Kesinti	Uzun Süreli Kesinti	Overall Accuracy
Random forest Classifier	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Decision Tree Classifier	0.49	0.79	0.79	0.61	0.54	0.73	0.62
XGBoost Classifier	0.97	0.99	1.00	0.98	0.96	0.98	0.98
Naive Bayes Classifier	0.12	0.88	0.74	0.19	0.47	0.58	0.41
KNN	0.87	0.95	0.95	0.91	0.87	0.91	0.90

Model	Compute Multiclass AUC Score
Random forest Classifier	0.98
Decision Tree Classifier	0.81
XGBoost Classifier	0.98
Naive Bayes Classifier	0.74
KNN	0.93

	Feature engineering with the Genetic Algorithm and SMOTE				
Model	Gen_1	Gen_2	Gen_3	Gen_4	Gen_5
Random forest Classifier	0.953	0.954	0.953	0.955	0.954
Decision Tree Classifier	0.712	0.712	0.712	0.712	0.712
XGBoost Classifier	0.949	0.947	0.949	-	-
Naive Bayes Classifier	0.627	0.624	0.627	0.624	0.627
KNN	0.899	0.915	0.937	0.937	0.937