Georgia State University

CSC6780 – Fundamentals of Data Science

Fall 2021

Final Project Report

SOLAR FLARE PREDICTION (MVTS)

Explode Legacy

Nigar Khasayeva

Shraddha Sharma

Akhil Arya

Table of Contents

1 Business Understanding

1.1 Business Problem

Solar flares and coronal mass ejections (CMEs) are events occurring in the solar corona and heliosphere. A flare is characterized by a sudden brightening by orders of magnitude in Extreme Ultra-Violet (EUV) and X-ray and, for large events, gamma-ray emissions, from a small area on the Sun, lasting from minutes to a few hours.

If intense, these sudden bursts of energy can affect the Earth's atmosphere and damage satellites and the electrical grid. And most importantly, we want to study space weather because of hazardous energetic protons thrown out of strong solar blasts. They can cause serious injuries to astronauts resulting in skin cancer or radiation poisoning. Hence, we want to propose a Machine Learning solution to forecast the Solar Flare events.

1.2 Dataset

For the solar flare prediction, we have restricted the classes to binary classifiers. If a region produces one or more flares within a given time interval, then it comes into the positive class and vice versa. In order to train our prediction models, we are going to use CLUS (Climate based Under Sampling) [*dmlab.cs.gsu.edu/data/SHARPS/BERKAY/v0.7/CLUS*] Data. The CLUS data set is derived from the SHARP series of Data. SHARP Data series, is a test bed dataset for machine learning practitioners, which is a cleaned and validated dataset. The advantage of this dataset is that we can train our several machine learning models using this data and can make comparisons among the performance of several models, irrespective of bias of the data.

We have used a multivariate time series (MVTS) dataset which is openly accessible to all. For our analysis, we have taken CLUS (Climatic under-sampled data) i.e. approx. 1:6 ratios are maintained of flare to non-flare values during sampling, which is also multivariate time series data. It has four partitions from which we have taken one partition for the training purpose, the second partition for the testing purpose, and the third one for validation purposes. Since we are doing binary classification, we have divided our target class into two i.e. Flare class (X class, M class) represented by 1 and Non-Flare class (B class, C class, FQ class) represented by 0.

Each Instance of our CLUS dataset has 51 features, which consists of magneto grams' physical parameters, spatial parameters integrated. The observation period of our time series data is 12 hours, and the predictions will be made for the next 24 hours. Each instance in our time has 60 observations with 12 minutes' cadence. We will be selecting a few features among all for the better performance of the classifier.

1.3. Proposed Analytics Solution

As an analytics solution for this problem, we suggest the Time Series Classification. We are going to build predictive models for our target variable based on our real-valued attributes. For the observation window, a 12 hours period will be embedded and a 24 hours period will be chosen for the prediction period. Before starting the progression, we are presenting pre-processing phases. These are interpolation for the missing values and the Standard Scaler for the normalization of our dataset.

We are going to implement three modelling approaches :

1. SK learn classifiers, which will be our baseline classifier. It will take input for each time series data as median, mean and standard deviation. We will predict our result based on prediction from decision tree classifier, random forest classifier and SVC
2. Through column concatenation, where all the time series variables were aggregated, we will consider it as a univariate time series. After which we will implement a Time Series forest classifier.
3. Here we will use a Random time series classifier for each feature separately and will ensemble them together.

We will use the SK learn time classifier on time series to capture the temporal components for our evaluation purpose. Our basic idea behind choosing SK learn classifiers was to compare the results with SK time models which are tree based classifiers.

2. Data Exploration and Preprocessing

Our dataset consists of high dimensionality which may result in lower performance for a classifier. It is beneficial to reduce the dimensionality by getting rid of features that are not very helpful at the classification task. Here, we are using highly ranked features from the Research Paper for selecting the features for our Analysis

FEATURE NAME	DESCRIPTION
TOTUSJH	Total unsigned current helicity
TOTBSQ	Total magnitude of Lorentz force
TOTPOT	Total photospheric magnetic free energy density
TOTUSJZ	Total unsigned vertical current
ABSNJZH	Absolute value of the net current helicity
SAVNCPP	Sum of the modulus of the net current per polarity
USFLUX	Total unsigned flux

Our final data frame consists of 30216 instances which we splitted into three sections i.e 0 to 9289 will be part of training , 9289 to 19667 will be part of testing and will be part of validation 19667 to 30216.

Our Analytics-based table will be a nested pandas dataframe, which will store a time series of 60 time instances in each cell. We have used the capability of the nested pandas dataframe, which is different from the conventional data frame structure. Our final data is presented in the image below:

	USFLUX	TOTUSJH	ABSNJZH	SAVNCPP	TOTBSQ	TOTPOT	TOTUSJZ	Target
0	0 2.325309 1 2.257020 2 2.164150 3...	0 1.809717 1 1.808893 2 2.140524 3...	0 -2.317018 1 -1.986666 2 -1.328308 3...	0 -2.104703 1 -1.660123 2 -1.128717 3...	0 2.557658 1 2.536412 2 2.404457 3...	0 2.034211 1 2.430360 2 2.195896 3...	0 1.437631 1 1.495962 2 1.886092 3...	1
1	0 -0.327391 1 -0.522681 2 -0.561586 3...	0 -0.695613 1 -0.831659 2 -0.812206 3...	0 -0.089806 1 0.407367 2 -0.153797 3...	0 0.926177 1 0.436984 2 -0.180986 3...	0 -1.472919 1 -1.626146 2 -1.713289 3...	0 -0.957924 1 -1.022498 2 -0.831049 3...	0 0.663321 1 0.386805 2 0.189967 3...	1
2	0 1.279723 1 1.195499 2 1.263750 3...	0 2.098368 1 1.454472 2 1.636940 3...	0 0.482595 1 0.873346 2 0.420986 3...	0 1.794760 1 1.172155 2 2.290417 3...	0 1.015588 1 0.927033 2 0.890572 3...	0 0.316325 1 -0.059308 2 -0.203118 3...	0 2.729711 1 2.027610 2 2.223843 3...	1
3	0 -2.229260 1 -2.138408 2 -2.089686 3...	0 -1.474491 1 -0.783034 2 -0.843511 3...	0 -1.187571 1 -0.846622 2 -0.302719 3...	0 -0.801030 1 -0.766697 2 -0.143573 3...	0 -1.580123 1 -1.493743 2 -1.506993 3...	0 -0.797485 1 -0.671026 2 -0.609521 3...	0 -1.666826 1 -0.820853 2 -0.733178 3...	1
4	0 1.123380 1 1.197733 2 1.186214 3...	0 1.096648 1 1.227370 2 0.943277 3...	0 2.428347 1 0.921478 2 0.370888 3...	0 -0.806662 1 -0.641252 2 -0.060963 3...	0 1.624315 1 1.631743 2 1.596815 3...	0 1.786427 1 1.725552 2 1.809404 3...	0 0.642283 1 0.906780 2 0.657050 3...	1
...	...	...	...	...	...	...	...	...
30211	0 1.696444 1 1.917201 2 1.191068 3...	0 1.857143 1 2.322095 2 1.351653 3...	0 0.328172 1 -0.302259 2 -0.541185 3...	0 -0.093336 1 1.748660 2 0.125931 3...	0 2.416756 1 2.381196 2 1.377450 3...	0 2.179502 1 2.328447 2 1.561117 3...	0 1.482202 1 2.024310 2 1.244903 3...	0
30212	0 -2.876099 1 -2.034231 2 -1.790196 3...	0 -1.823599 1 -1.134019 2 -1.305890 3...	0 -1.815695 1 -1.030319 2 -0.663786 3...	0 -1.898401 1 -0.160733 2 0.186851 3...	0 -0.949728 1 -0.446239 2 -0.340347 3...	0 0.393962 1 0.812046 2 0.485696 3...	0 -0.335460 1 0.332421 2 -0.103801 3...	0
30213	0 2.155847 1 1.853882 2 1.793171 3...	0 1.237016 1 1.212662 2 1.111261 3...	0 -1.701702 1 -0.212497 2 0.014375 3...	0 0.663951 1 1.307760 2 0.408522 3...	0 1.870364 1 1.735806 2 1.649660 3...	0 1.952022 1 1.648637 2 1.392252 3...	0 2.120259 1 1.613975 2 1.459005 3...	0
30214	0 -0.915769 1 -0.912835 2 -1.075617 3...	0 -0.945187 1 -0.162984 2 -0.718940 3...	0 0.701469 1 -0.103340 2 -0.376843 3...	0 0.868531 1 0.054291 2 -0.015583 3...	0 -0.997788 1 -1.045726 2 -1.171345 3...	0 0.822467 1 0.844878 2 0.403535 3...	0 -0.687771 1 -0.290261 2 -0.450709 3...	0

	No Flare count	**Flare Count**
Target training set	8035	1254
Test training set	8977	1401
Validate training set	9125	1424
TOTAL	26137	4079

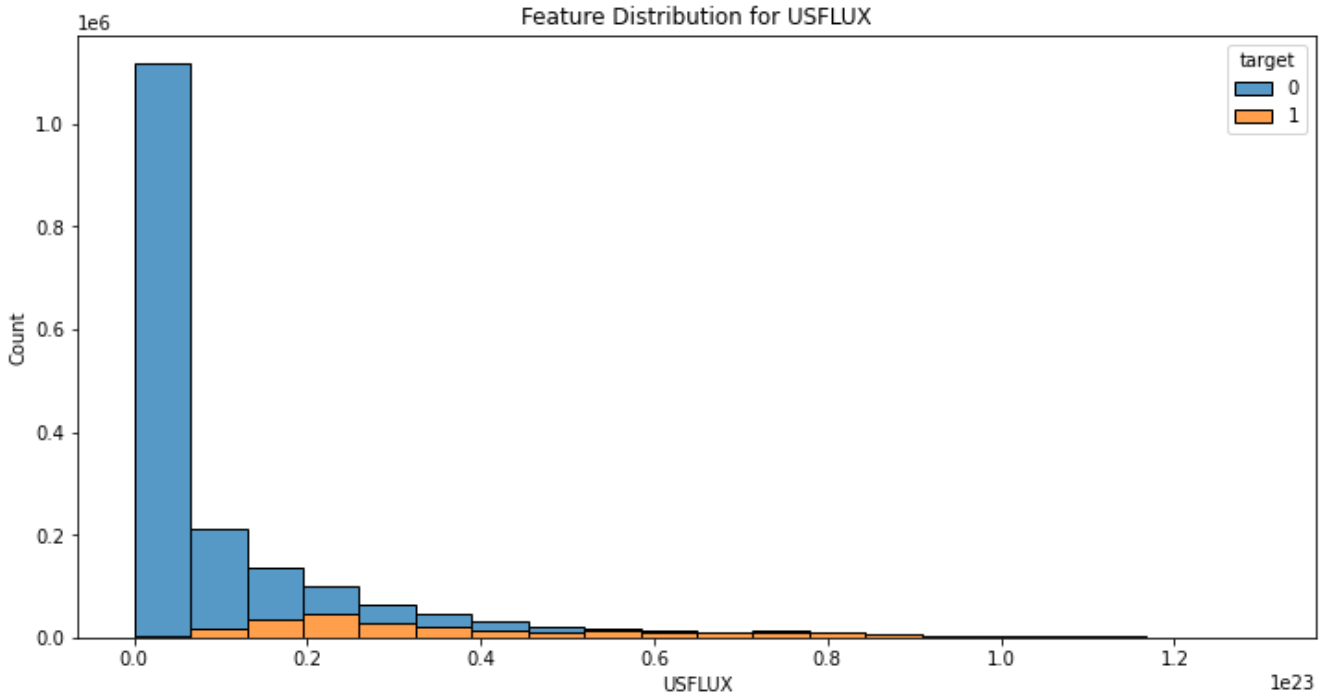
2.1 Data Quality Report

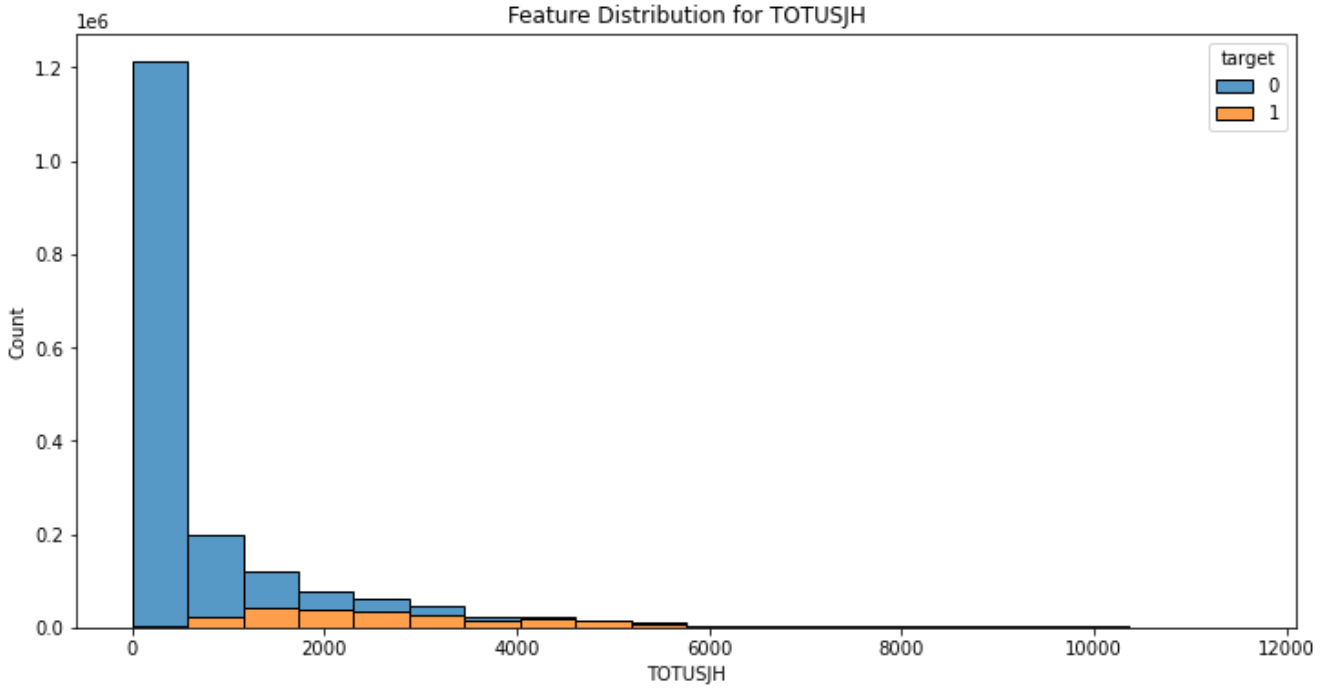
	Feature	Nulls Count	Min	Max	Avg Mean	Avg Std Dev	Std of Std Dev	Std Dev of Distrib	Q1	Median	Q3
0	USFLUX	4712.0	0.0	1.298067e+23	1.102793e+22	4.812524e+20	6.586893e+20	1.744321e+22	6.222610e+20	1.433425e+22	4.297629e+21
1	TOTUSJH	4712.0	0.0	1.152325e+04	8.093433e+02	3.825365e+01	5.643826e+01	1.382917e+03	4.160085e+01	9.529028e+02	2.710965e+02
2	ABSNJZH	4712.0	0.0	2.980218e+03	1.107824e+02	1.667655e+01	2.804408e+01	2.730961e+02	3.141913e+00	7.824690e+01	1.811220e+01
3	SAVNCPP	4712.0	0.0	1.085462e+14	4.648434e+12	8.838382e+11	1.243316e+12	1.007075e+13	2.772390e+11	3.970191e+12	1.312702e+12
4	TOTBSQ	4712.0	0.0	1.431918e+11	1.108083e+10	3.882663e+08	6.557893e+08	2.045840e+10	3.200768e+08	1.274459e+10	2.772911e+09
5	TOTPOT	4712.0	0.0	8.684370e+26	1.976124e+23	2.774457e+22	1.096261e+24	1.320571e+24	2.246169e+21	1.813834e+23	2.290849e+22
6	TOTUSJZ	4712.0	0.0	2.833974e+14	1.584594e+13	8.623452e+11	1.178739e+12	2.518471e+13	9.312461e+11	1.983504e+13	5.838776e+12

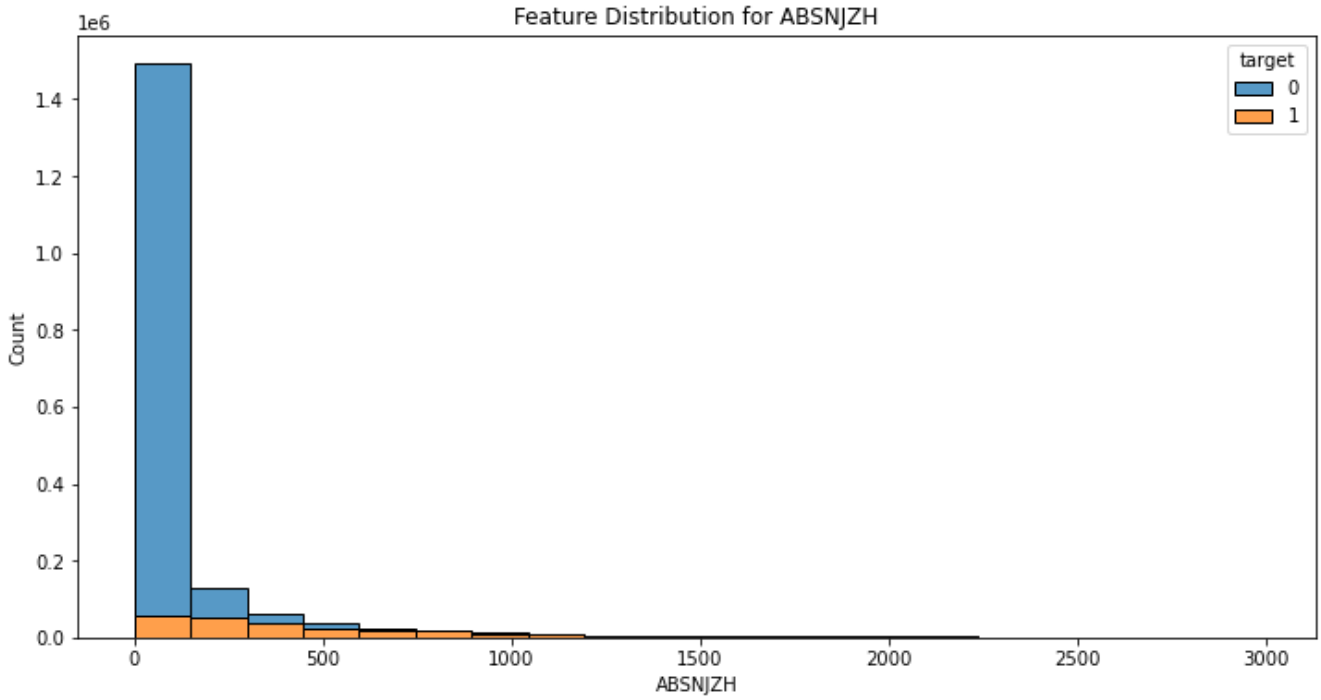
DESCRIPTION OF THE FEATURES IN ABOVE TABLE ARE AS FOLLOWS :

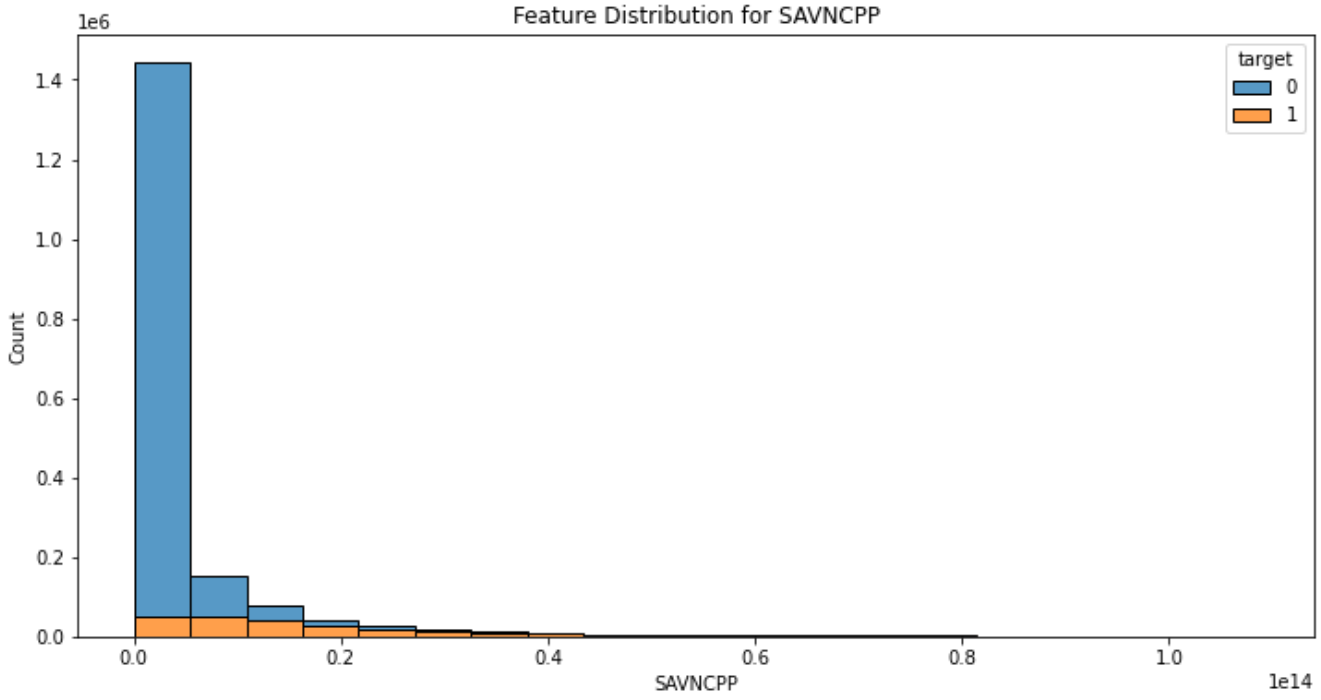
NULL COUNT: The count of null values for all the instances
MIN: The minimum value for all the instances
MAX: The maximum value for all the instances
AVG MEAN: The mean value for all the instances
AVG STD DEV: The average standard deviation value for all the instances
STD OF STD DEV: The standard deviation of the distribution of standard deviations.
STD DEV OF DISTRIB: The standard deviation of the distribution.
Q1: First quartile value
MEDIAN: Median value
Q3: Third quartile value

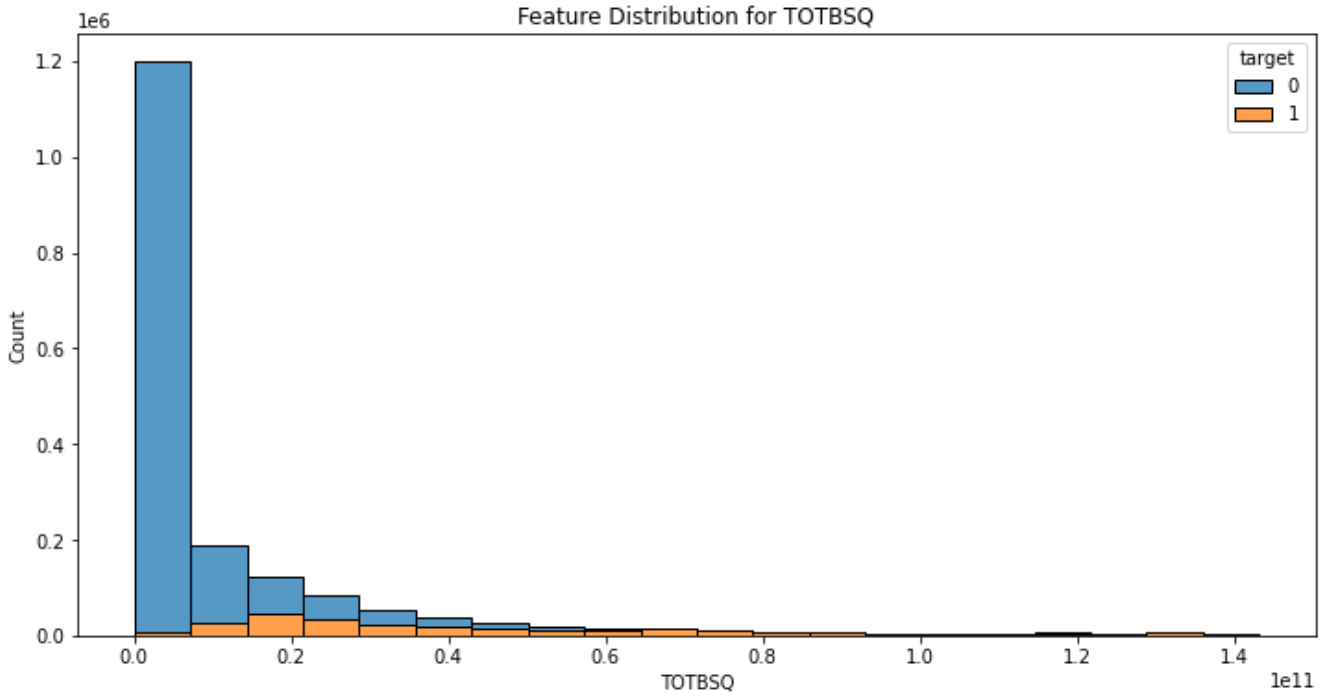
DISTRIBUTIONS PLOTS

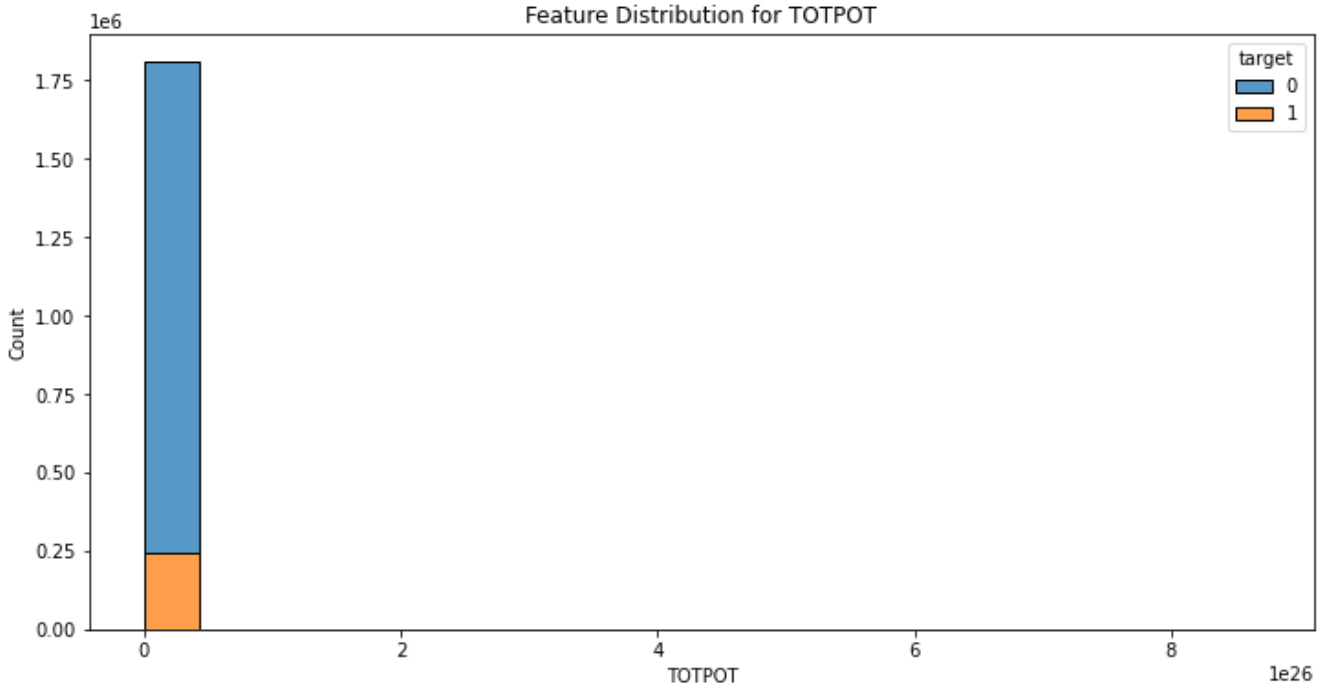





Feature Distribution for ABSNJZH
1e6
target
0
1
Count
ABSNJZH
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
0
500
1000
1500
2000
2500
3000


Feature Distribution for SAVNCPP
1e6
target
0
1
Count
SAVNCPP
1e14
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
0.0
0.2
0.4
0.6
0.8
1.0


Feature Distribution for TOTBSQ
1e6
1.2
1.0
0.8
0.6
0.4
0.2
0.0
Count
target
0
1
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
TOTBSQ
1e11


Feature Distribution for TOTPOT
1e6
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
Count
target
0
1
0
2
4
6
8
TOTPOT
1e26

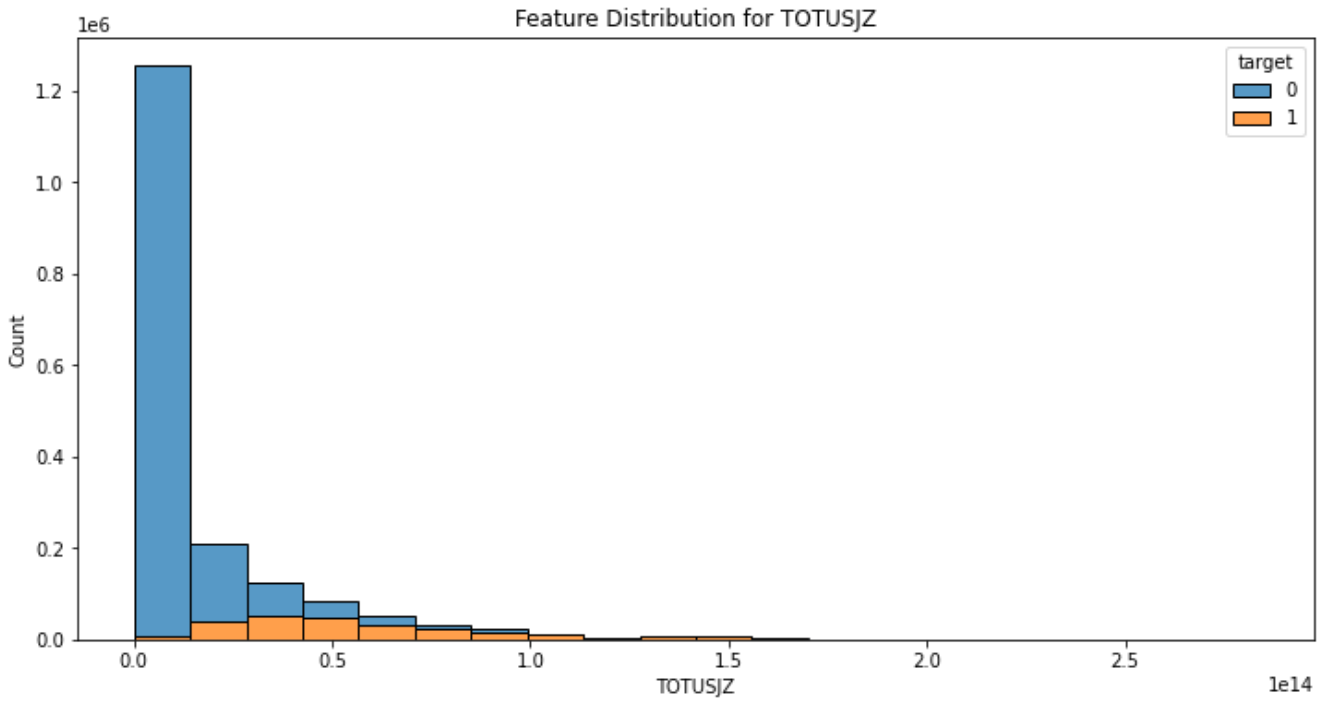
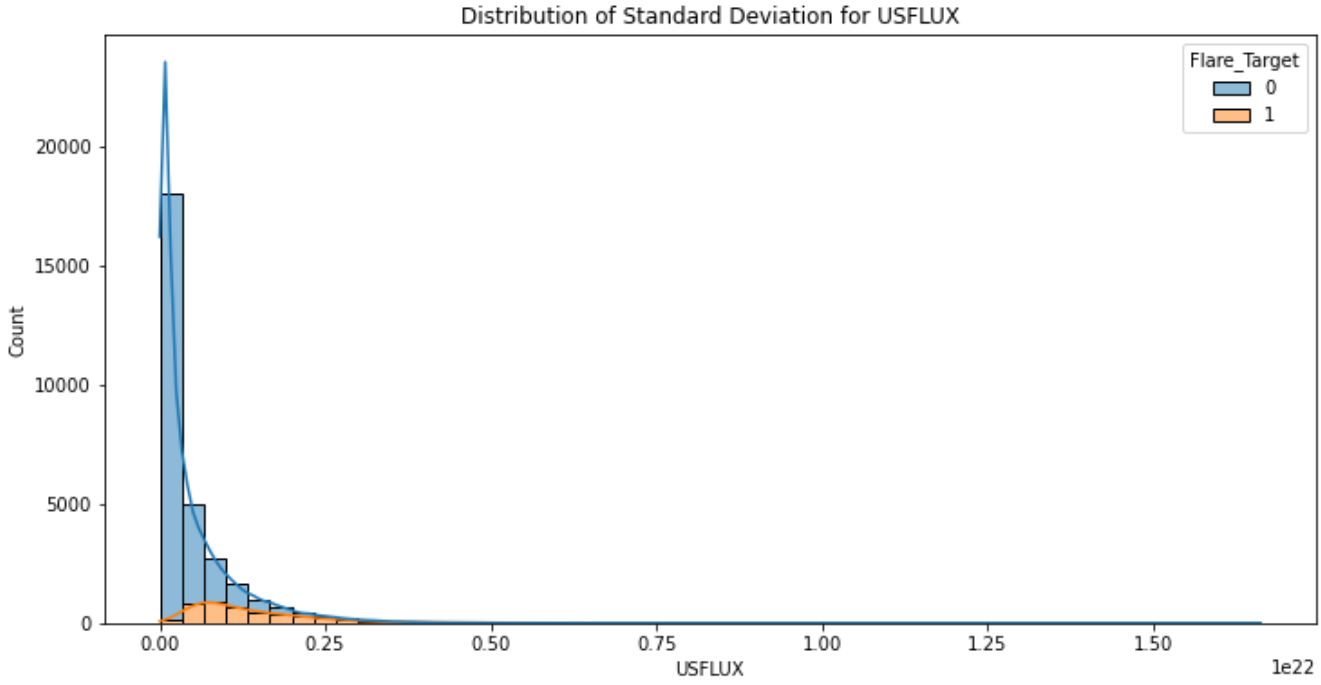
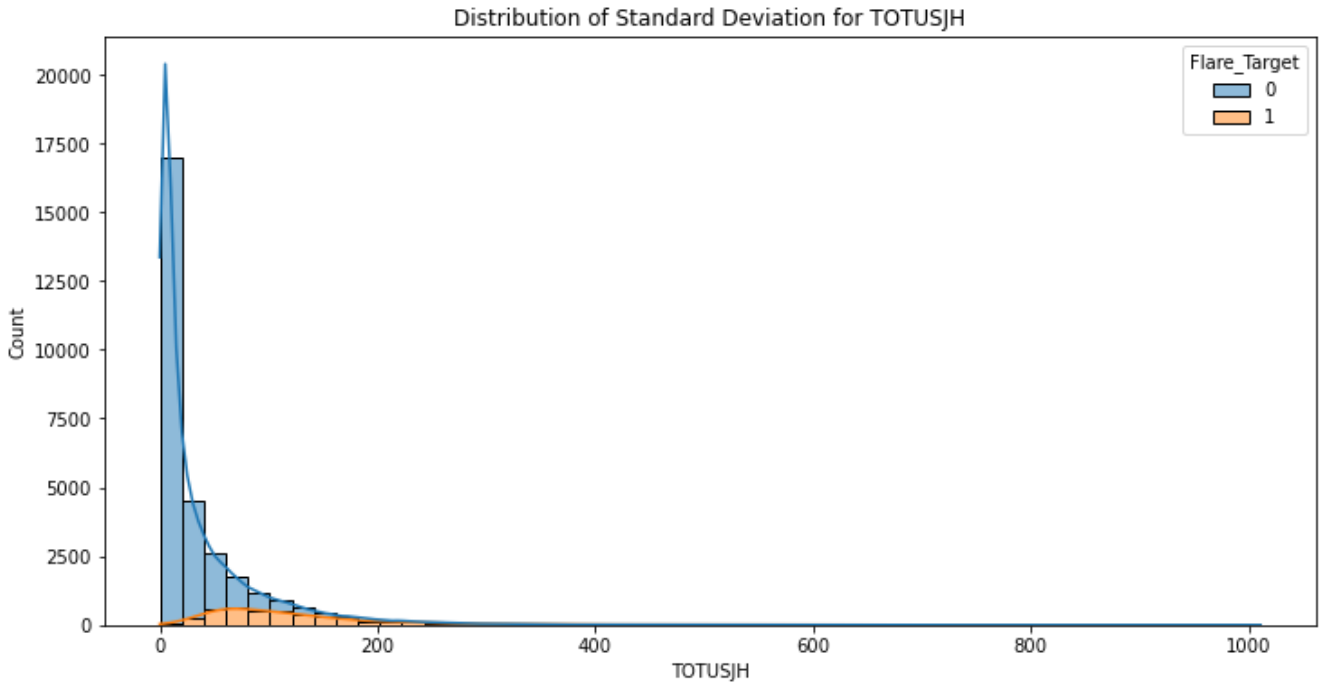
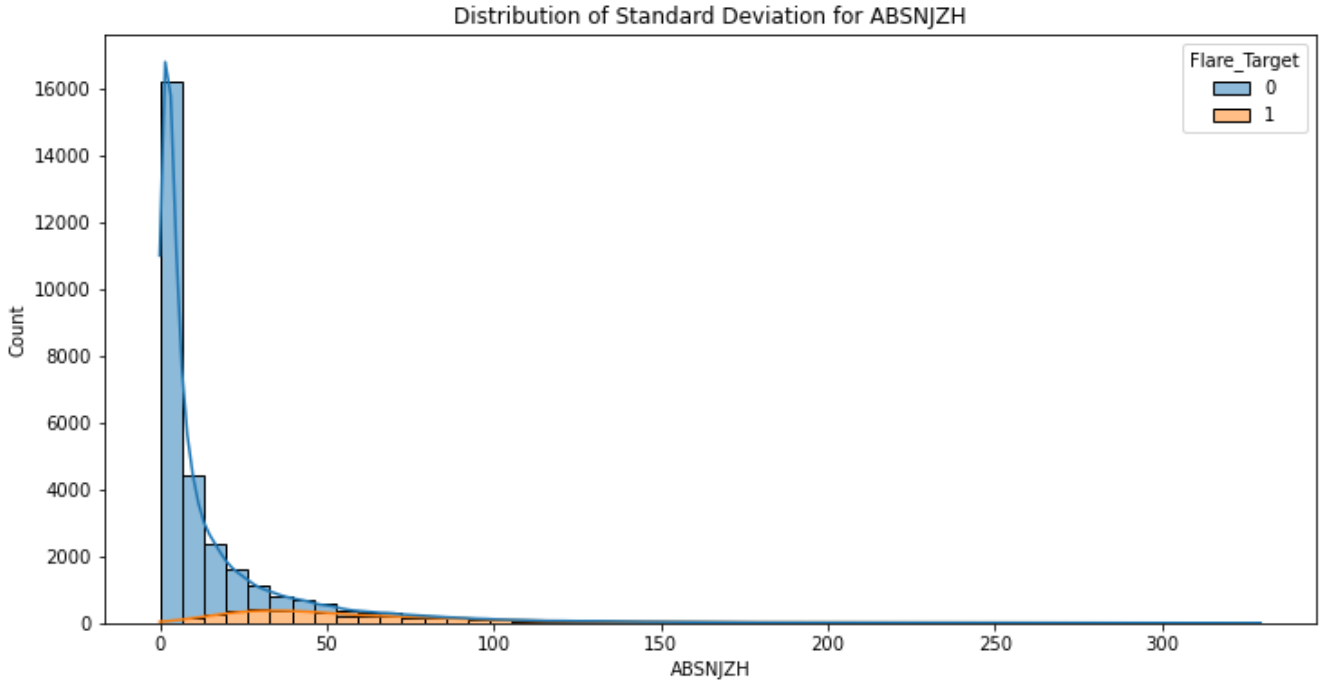


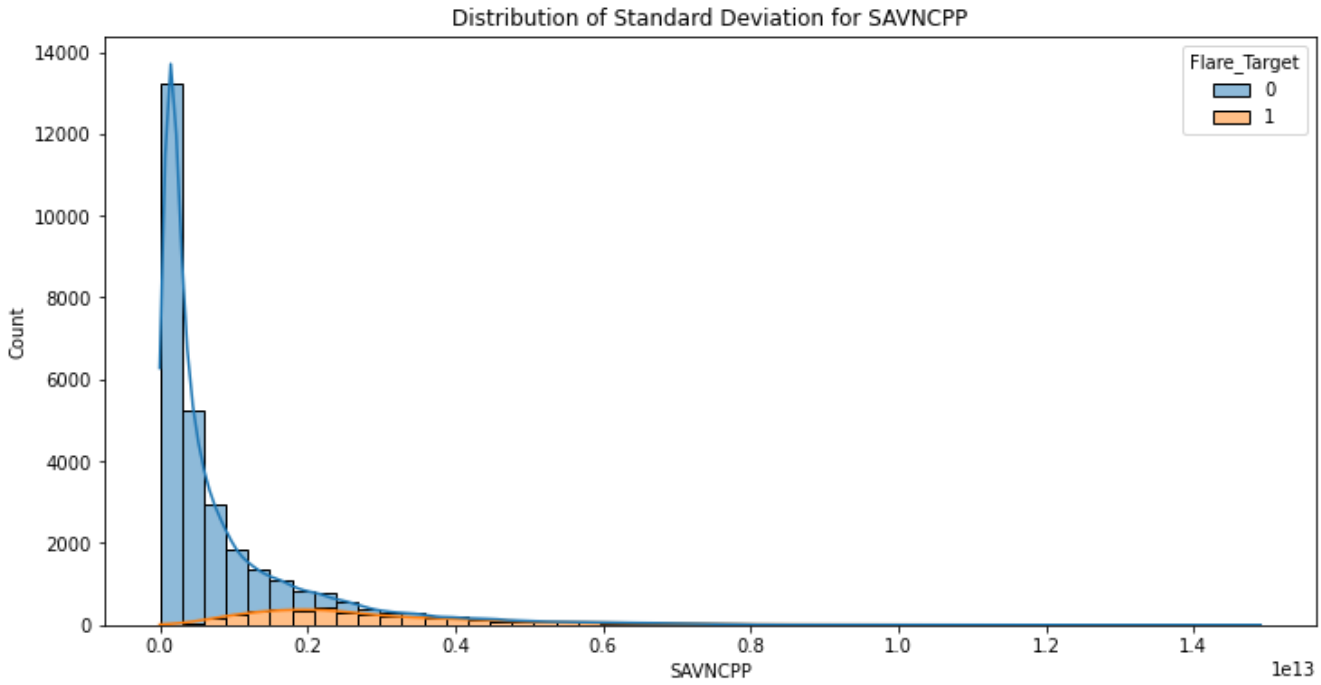
Figure 1. We have plotted the above histograms to show the feature distribution in relation to our target values (Flaring and Non-Flaring). The blue bars are representing the Non-Flaring instances and correspondingly the red bars are representing the Flaring instances.

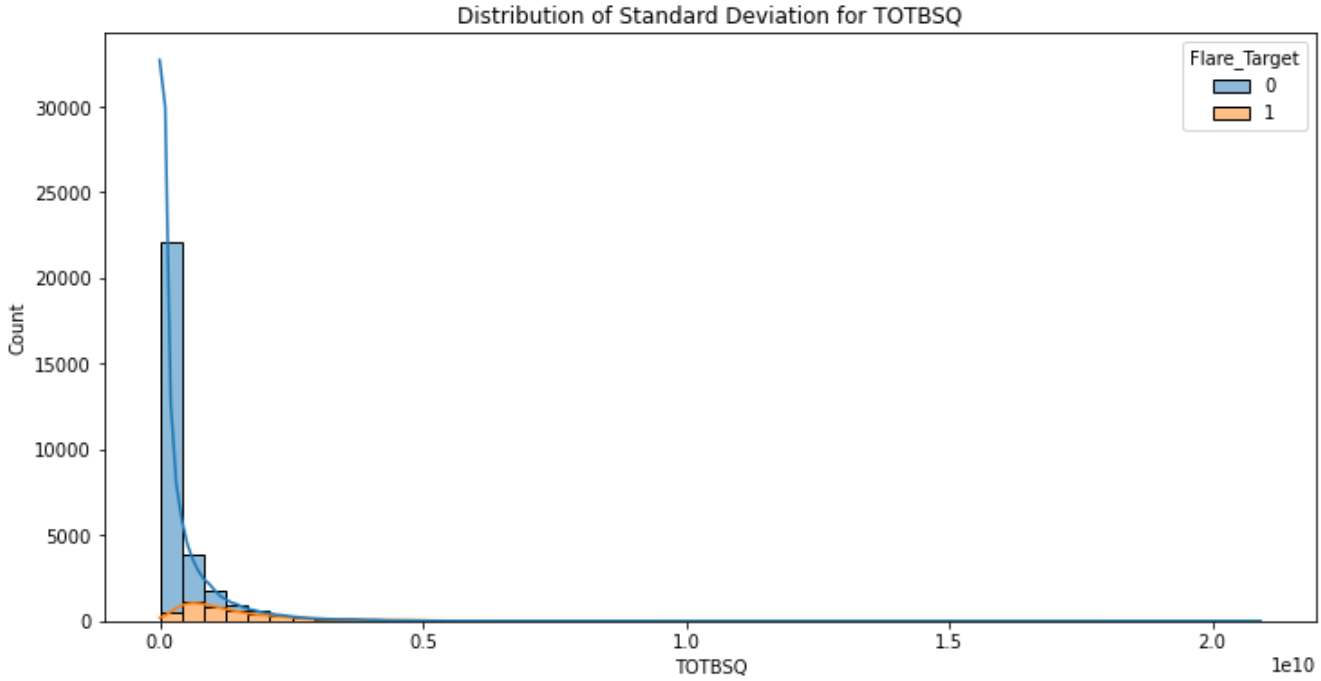
DISTRIBUTION OF STANDARD DEVIATION FOR FEATURES

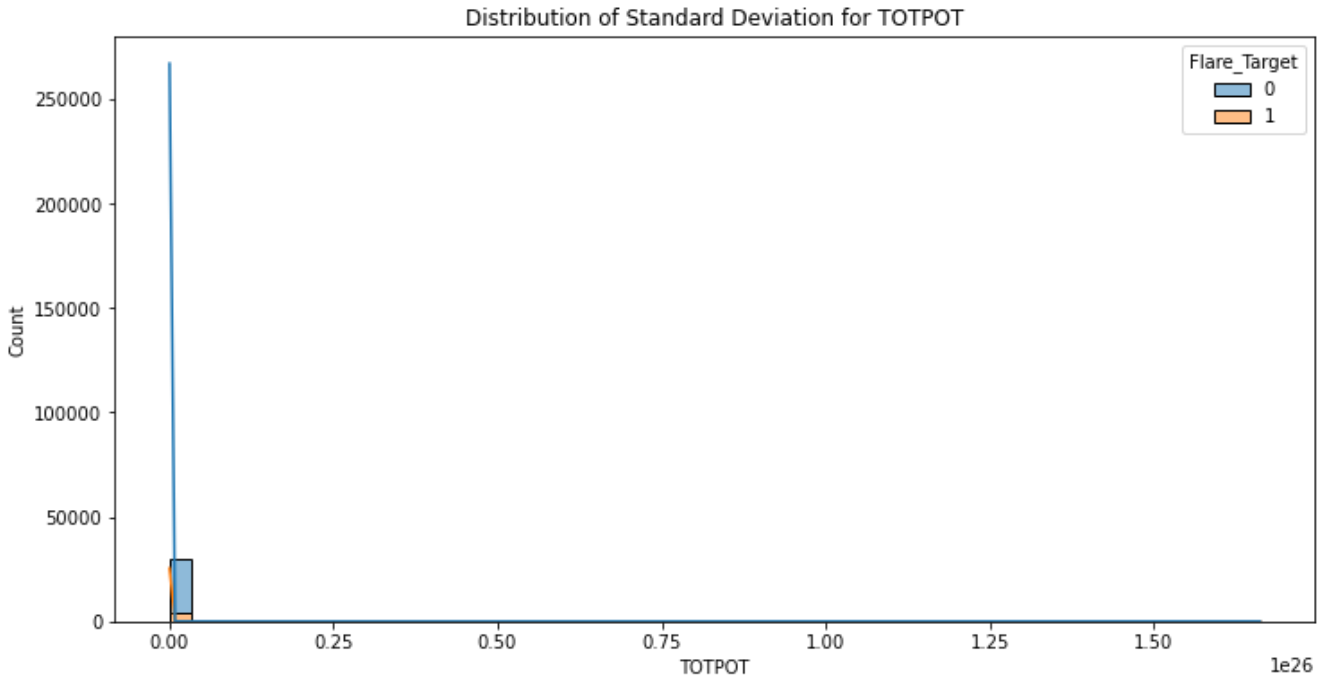


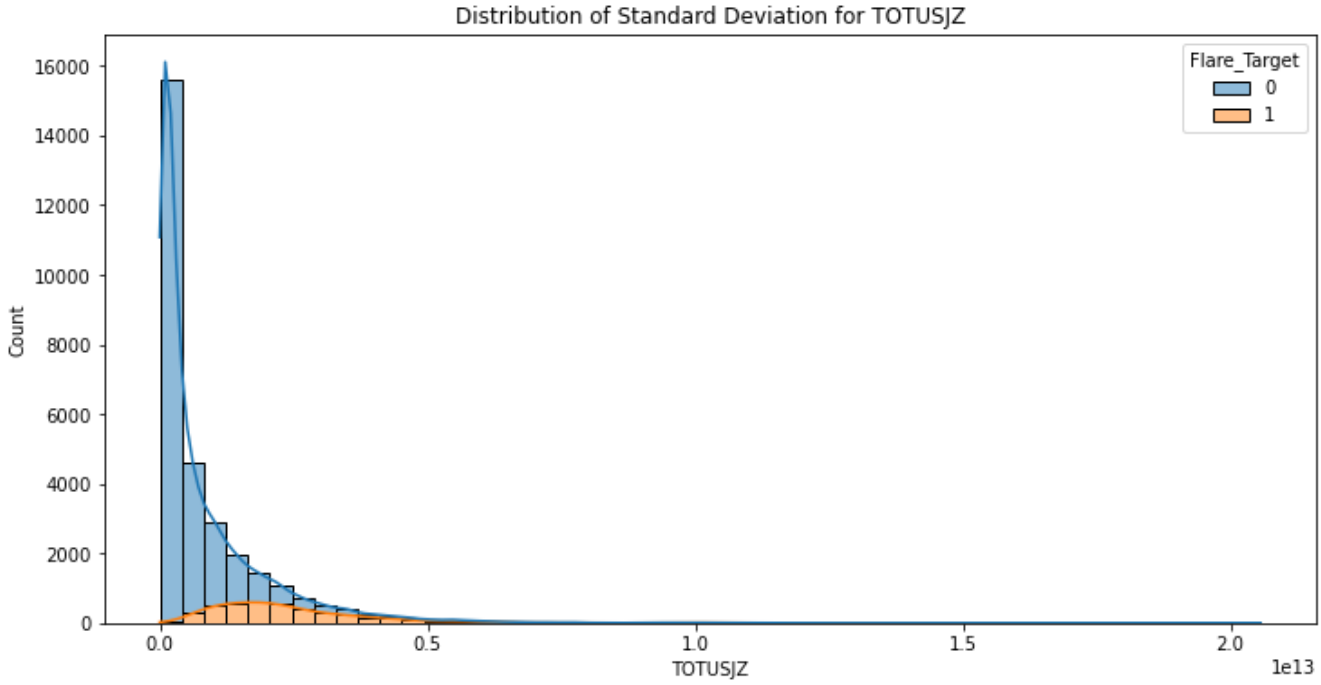

Distribution of Standard Deviation for TOTUSJH
Flare_Target
0
1
Count
TOTUSJH
0
2500
5000
7500
10000
12500
15000
17500
20000
0
200
400
600
800
1000


Distribution of Standard Deviation for ABSNJZH
Flare_Target
0
1
Count
ABSNJZH
0
2000
4000
6000
8000
10000
12000
14000
16000
0
50
100
150
200
250
300

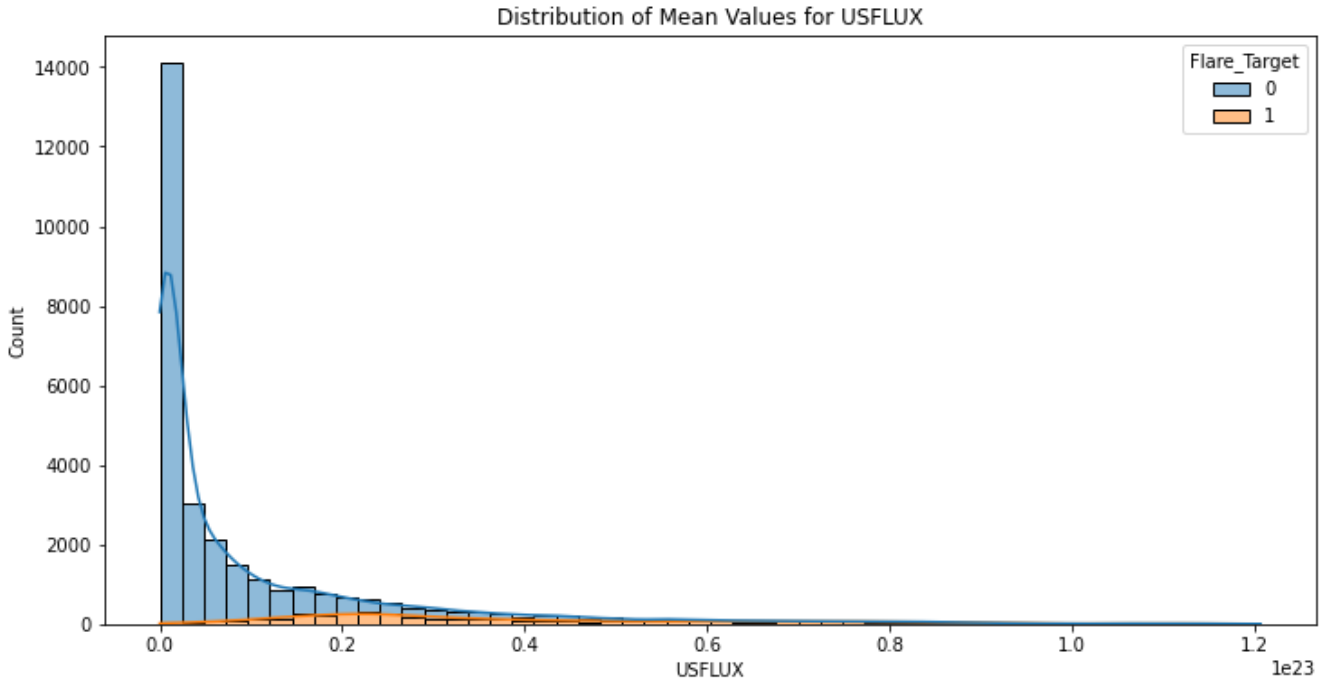

Distribution of Standard Deviation for SAVNCPP
Flare_Target
0
1
Count
14000
12000
10000
8000
6000
4000
2000
0
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
SAVNCPP
1e13

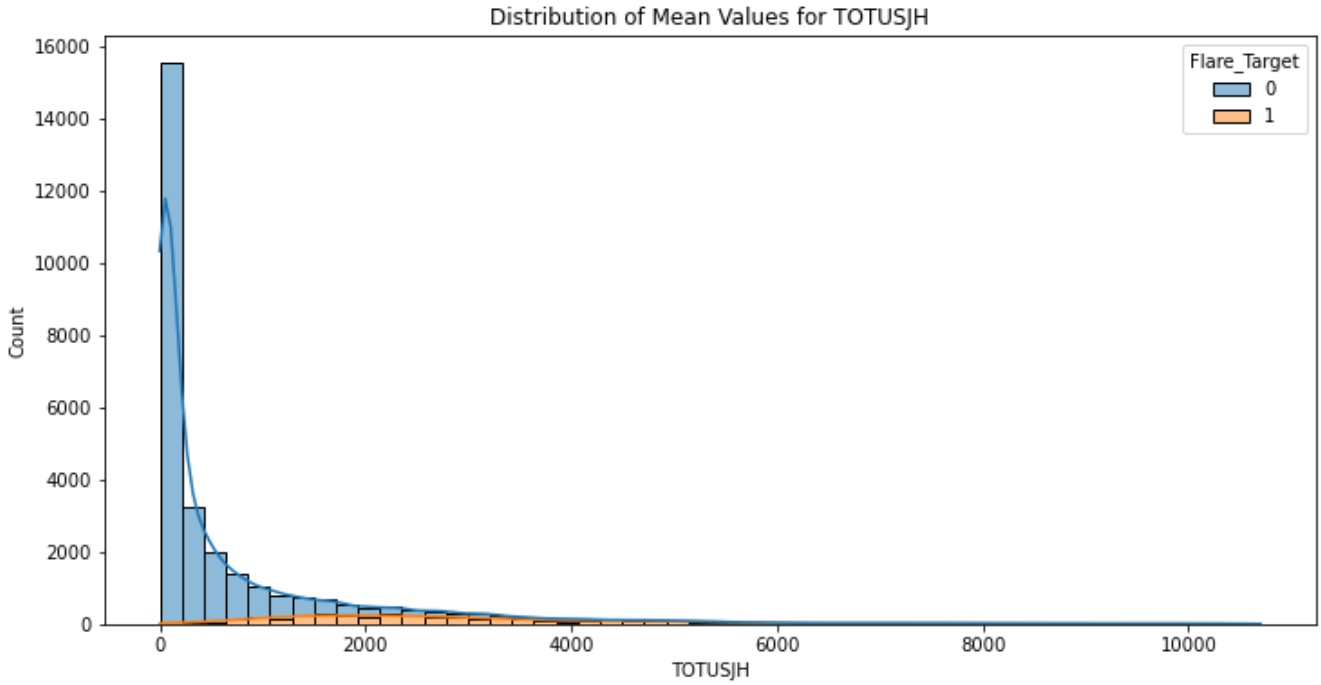

Distribution of Standard Deviation for TOTBSQ
Flare_Target
0
1
Count
30000
25000
20000
15000
10000
5000
0
0.0
0.5
1.0
1.5
2.0
TOTBSQ
1e10

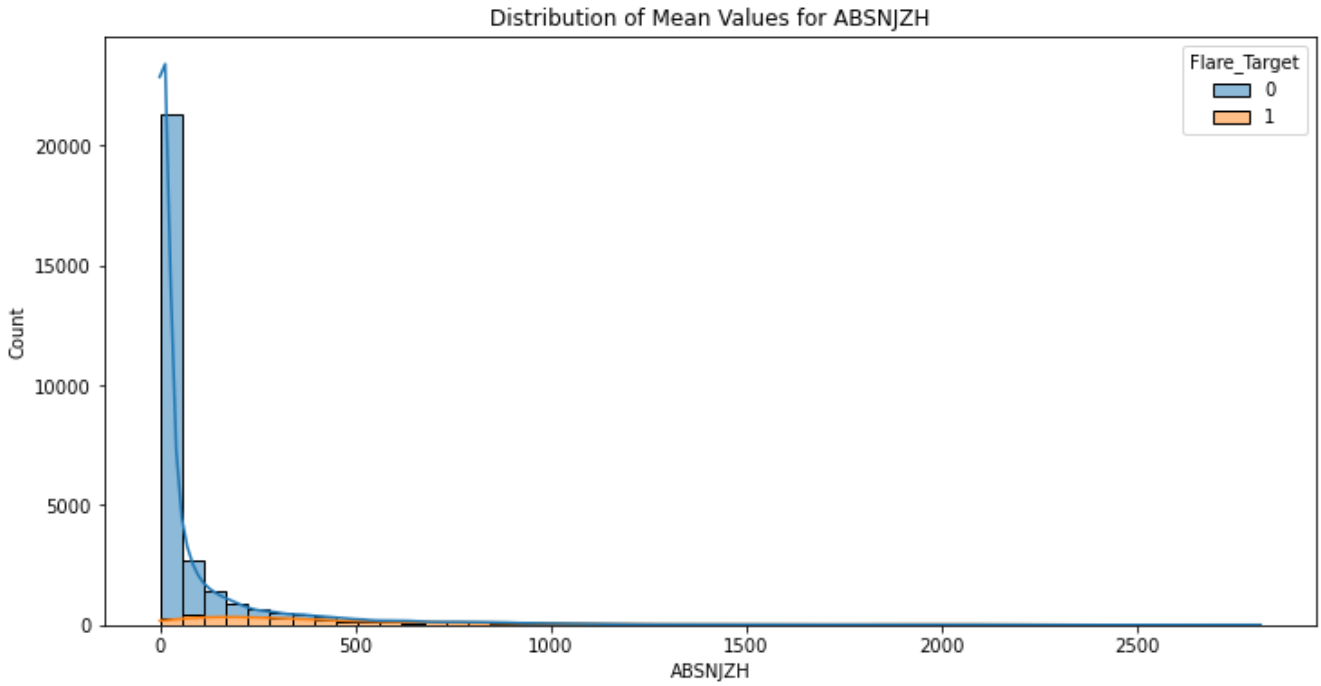

Distribution of Standard Deviation for TOTPOT
Flare_Target
0
1
Count
250000
200000
150000
100000
50000
0
0.00
0.25
0.50
0.75
1.00
1.25
1.50
TOTPOT
1e26

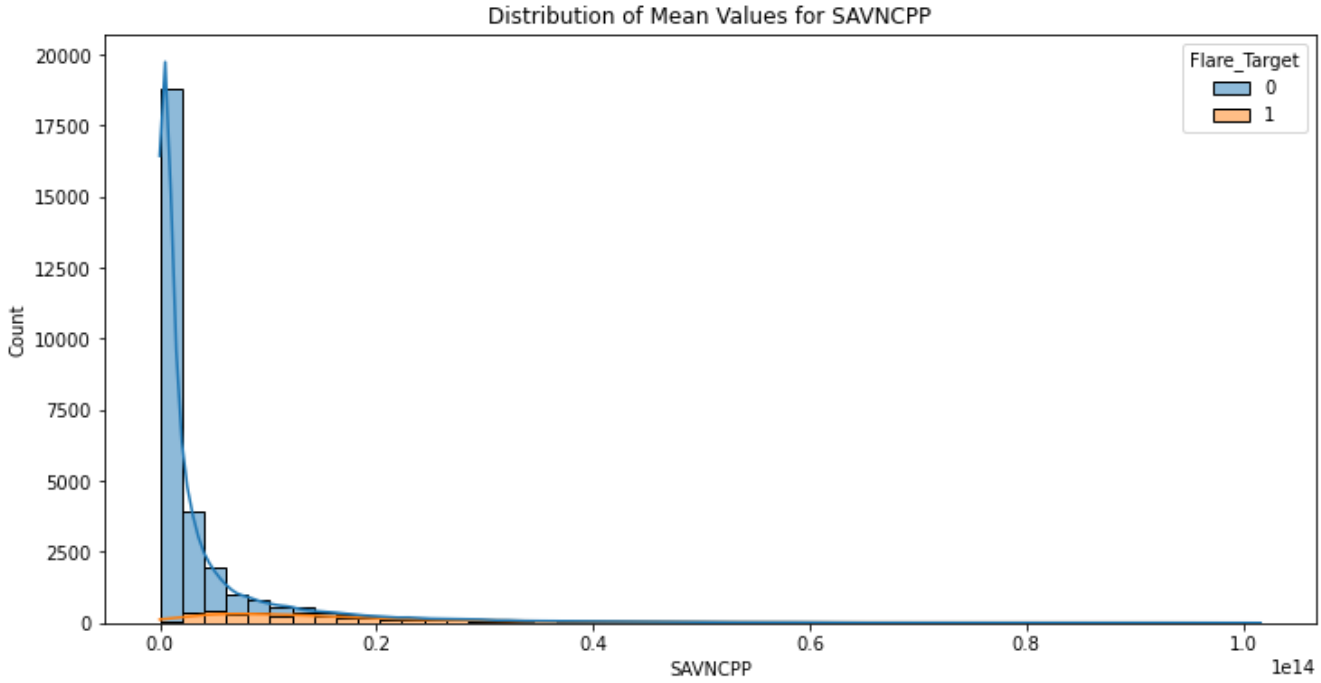

Distribution of Standard Deviation for TOTUSJZ
Flare_Target
0
1
Count
16000
14000
12000
10000
8000
6000
4000
2000
0
0.0
0.5
1.0
1.5
2.0
TOTUSJZ
1e13

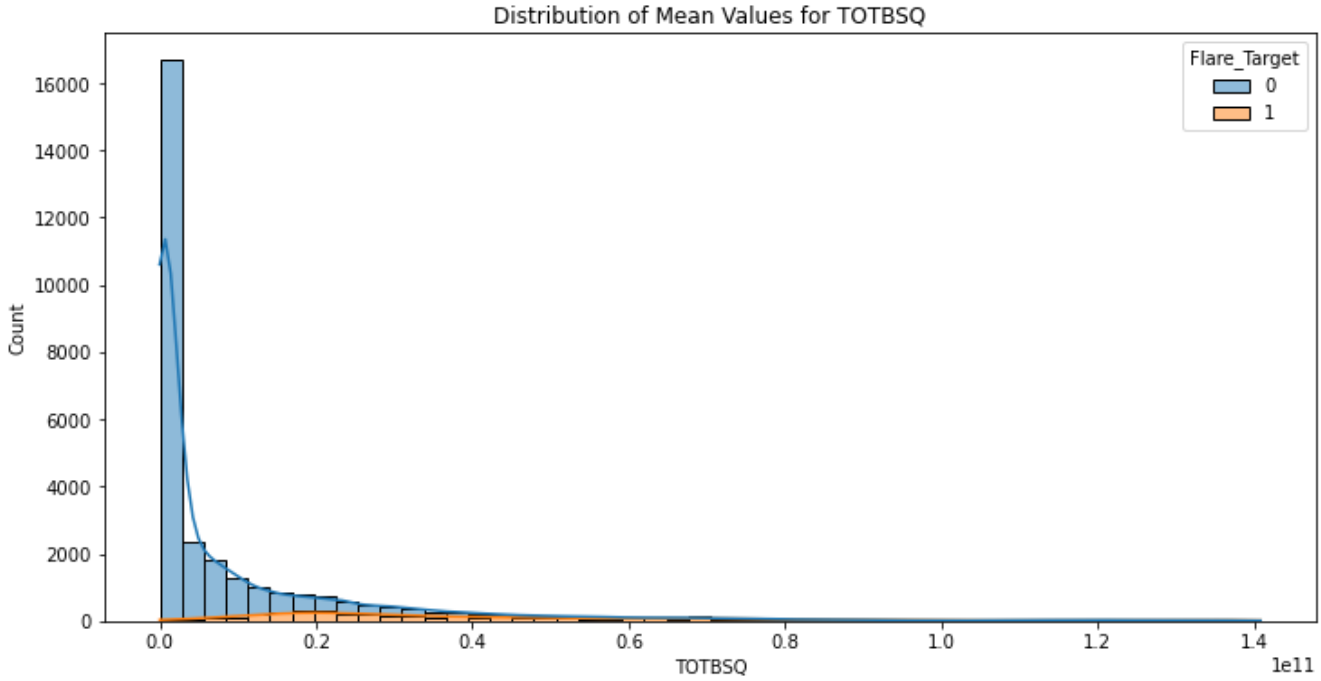
DISTRIBUTION OF THE MEAN VALUES FOR THE FEATURES

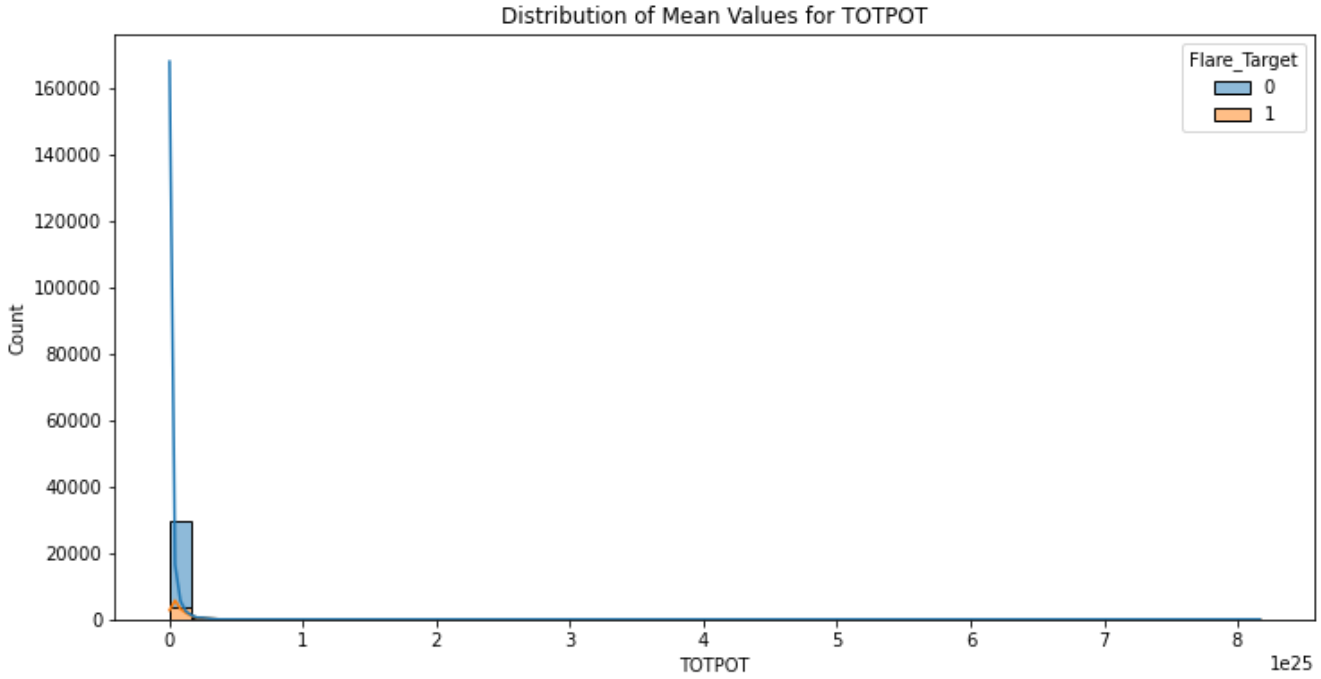


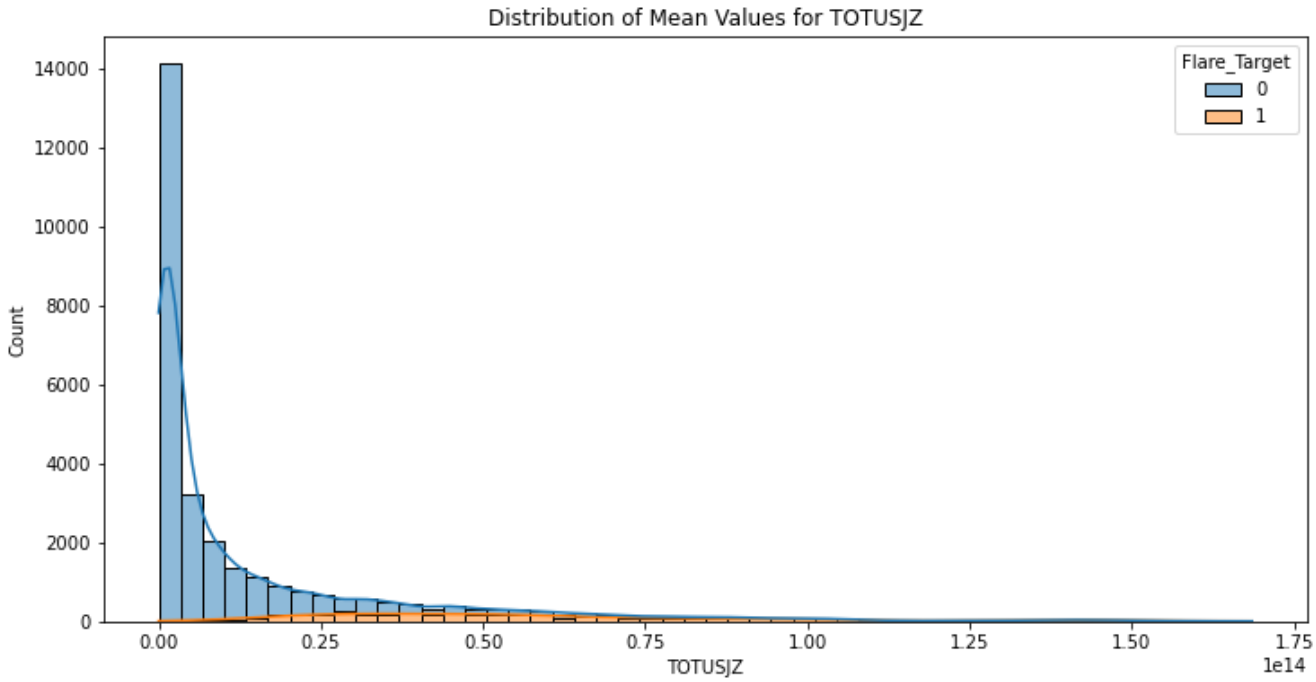



Distribution of Mean Values for ABSNJZH
Flare_Target
0
1
Count
20000
15000
10000
5000
0
0
500
1000
1500
2000
2500
ABSNJZH


Distribution of Mean Values for SAVNCPP
Flare_Target
0
1
Count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
0.2
0.4
0.6
0.8
1.0
1e14
SAVNCPP


Distribution of Mean Values for TOTBSQ
Flare_Target
0
1
Count
16000
14000
12000
10000
8000
6000
4000
2000
0
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
TOTBSQ
1e11


Distribution of Mean Values for TOTPOT
Flare_Target
0
1
Count
160000
140000
120000
100000
80000
60000
40000
20000
0
0
1
2
3
4
5
6
7
8
TOTPOT
1e25



2.2. Missing values and Outliers

To handle the missing values found in the table, we have used linear interpolation techniques for each instance. Here we are approximating a value of the feature using two known values in the proximity of the missing values using forward filling or backward filling process.

2.3 Normalization

We have done normalization in two ways :

1. **Global Normalization**

   Another step in the preprocessing stage was normalization. For our dataset, this step was completed using the Standard Scaler. We used a defined StandardScaler instance with default hyperparameters. Then we called the fit_transform() function and passed our dataset to create a normalized form of our dataset. This process was completed on the whole distribution of the time series dataset.

2. **Instance based Normalization**

   In our data set, we have one time series which consists of 60 time stamps. So for this process we did normalization in a batch of 60 instances. In which we used an iterative function which used to normalize batches of 60 instances one by one and

append them into a dataframe. We used Standard Scaler normalization in which the scaler standardizes a feature by subtracting the mean and then scaling to unit variance. Unit variance means dividing all the values by the standard deviation.

We tested our models on both the normalized data and we observed that our models' performances were reduced by a great margin. So we decided to go ahead with data which is not normalized.

3. Model Selection and Evaluation

3.1 Model Selection

We have done three modeling techniques, which are as follows :

1. **SK Learn Classifiers with the Feature Extraction**

   Our Data Set has a temporal component, so here we have used the TS Fresh library to use the TSFreshFeatureExtraction method with the help of which we calculated features like Mean, Median and Standard Deviation for our complete data set. Which looks like:

	ABSNJZH__median	ABSNJZH__mean	ABSNJZH__standard_deviation	SAVNCPP__median	SAVNCPP__mean	SAVNCPP__standard_deviation
0	283.356510	291.813104	47.945573	1.357856e+13	1.340867e+13	2.551524e+12
1	66.619653	60.367148	27.764524	8.933433e+12	8.835084e+12	8.927245e+11
2	127.237679	126.322234	13.837997	1.424802e+12	1.573723e+12	6.779822e+11
3	746.403673	737.612053	54.688547	4.978973e+13	4.852730e+13	4.374883e+12
4	23.520698	23.577261	14.918347	8.099298e+12	8.163373e+12	1.108614e+12

Then we splitted the data into three categories: Training, Test and Validation datasets. On the final data set which was made with the help of TS Fresh libraries we applied a few SK Learn classifiers like Random Forest, Support Vector Machine, Decision Tree classifier and K-Nearest Neighbors. After training our model we tested it on our test and validation datasets. Then, we calculated the confusion matrix and the values as TSS, HSS, GSS, Precision, and Recall for both of the datasets. After comparison of Recall values for each model, we decided to select the Random forest classifier among the all

SKLearn classifiers we have tried.

2. Column Ensemble Based Time Series Forest Classifier

Since we already have a normal baseline classifier, we also tried training the model which can capture the temporal properties of our time series data. We have used the sktime python library, which is a very useful library for time series analysis. For our Multivariate Time Series Data, we have used the column Ensemble approach. In this approach, one classifier is fitted for each time series column and their predictions aggregated:

- For each time series feature, we have fitted a Time Series Forest Classifier with 200 estimators. A time series forest is an ensemble of decision trees built on random intervals. The trees will be ensembled based on Average Probability estimates. TSF will compute the statistical features (mean, std, slope) for each random time interval and train the several ensembles based on those features.
- The Predictions made by each TSF on different time series columns will be then aggregated together. Soft Voting method is used for combining the predictions.

As we have already described above, our training data has some sort of imbalance present. In order to tackle the issue, we have also tried random undersampling, where we kept a ratio of 1:4 to the Flares and Non Flares classes respectively. We have trained our model on both original data as well as undersampled data. The results of both the approaches will be described below.

3. Column concatenation utilizing Time Series Forest Classifier

This approach to solving our MVTS problem offered by sklearn is applying a classifier after concatenating all columns of our time series into one. Utilizing ColumnConcatenator this way we are transforming the series into only one very long column. After this process we are using the Time Series Forest Classifier with estimators number equal to 250. Also in this approach, we again have trained our model both on original and undersampled datasets.

3.2 Model Evaluation

We are presenting our model evaluation based on the following evaluation metrics:

1. The *Heidke Skill Score* (HSS) measures the refinement in a prediction compared to a random prediction

$$\text{HSS} = \frac{2 * ((TP * TN) - (FN * FP))}{P * (FN + TN) + N * (TP + FP)}$$

2. The *True Skill Statistic* score (TSS) represents matches and differences between observation and forecast.

$$\text{TSS} = \frac{TP}{TP + FN} - \frac{FP}{FP + TN}$$

3. The *Gilbert Skill Score* (GSS) covers the number of hits due to chance events (event frequency multiplied by predicted events).

$$\text{GSS} = \frac{TP - CH}{TP + FP + FN - CH}$$

$$\text{Here, CH} = \frac{(TP + FP) * (TP + FN)}{TP + FP + FN + TN}$$

4. The *False Alarm Rate* is the ratio between false events and predicted positives.

$$\text{FAR} = \frac{FP}{FP + TP}$$

5. The *Recall Flare* is a ratio of true positives to the total number positives classes (TP and FN)

$$\text{Recall(Flare)} = \frac{TP}{TP + FN}$$

1. Model Evaluation for Random Forest Classifier

	TSS	HSS	GSS	Recall Flare	FAR
Test Data	0.56	0.63	0.46	0.59	0.22
Validate Data	0.60	0.54	0.37	0.67	0.46

Table 1. Random Forest Classifier on original dataset

2. Model Evaluations for column ensemble based TSF approach

	HSS	TSS	GSS	FAR	Recall Flare
Testing	0.61	0.54	0.44	0.22	0.56
Validation	0.58	0.60	0.41	0.38	0.66

Table 2. Column Ensemble using Time Series Forest on original dataset

	HSS	TSS	GSS	FAR	Recall Flare
Testing	0.67	0.66	0.51	0.27	0.70
Validation	0.60	0.68	0.43	0.41	0.77

Table 3. Column Ensemble using Time Series Forest on undersampled dataset

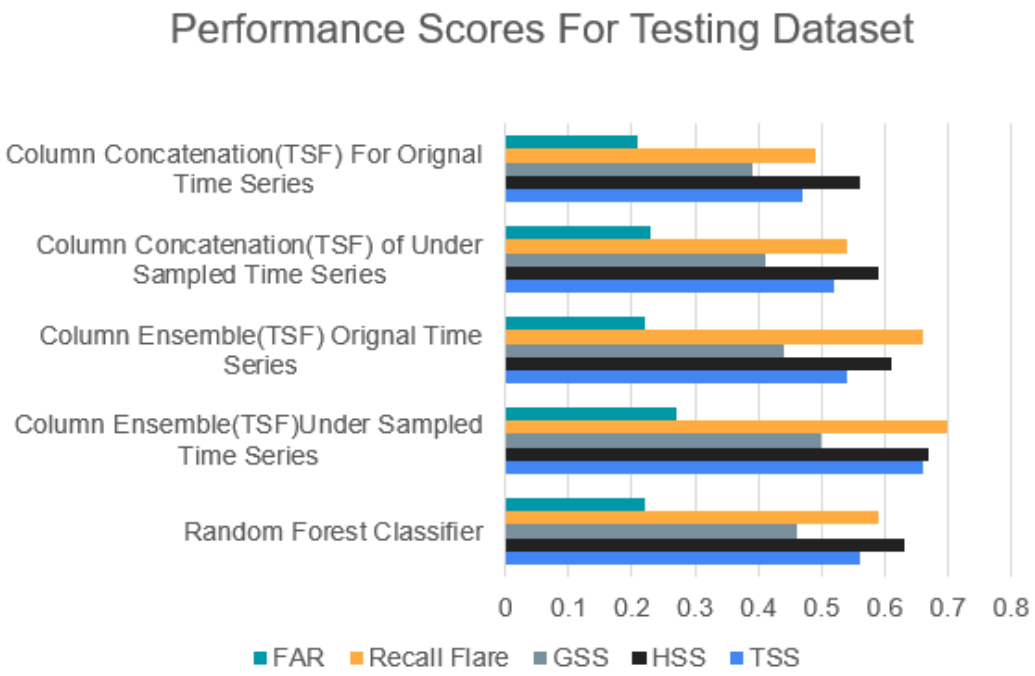
3. Model Evaluation for column concatenation based TSF approach

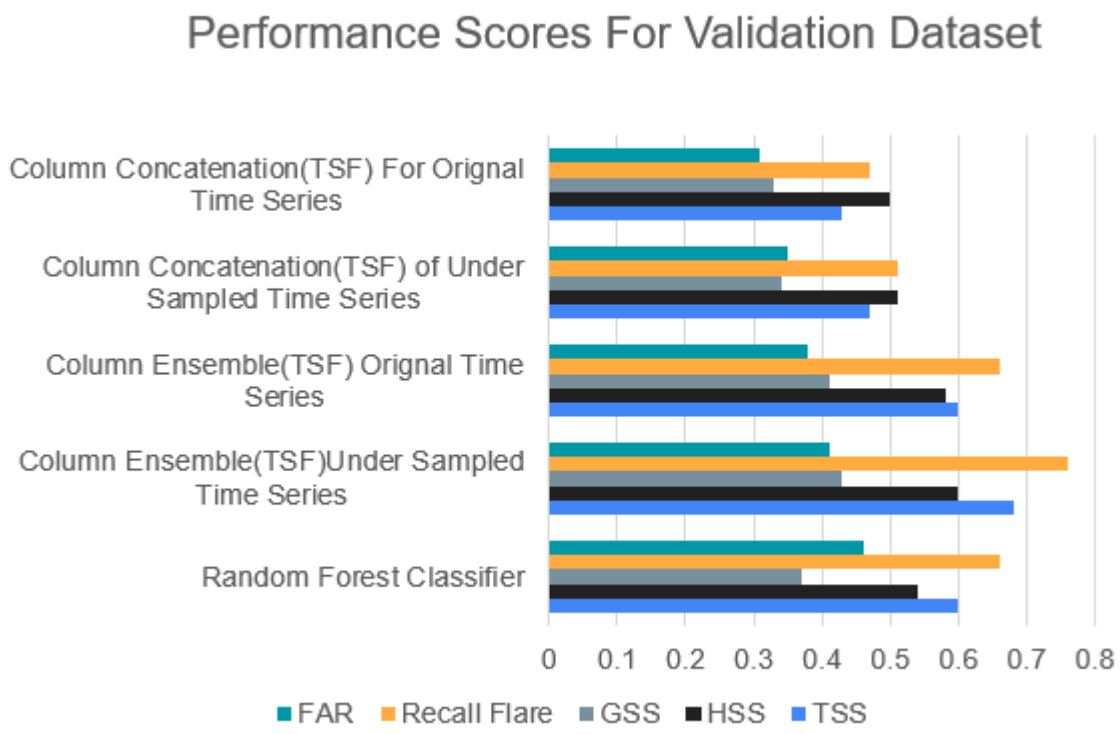
	HSS	TSS	GSS	FAR	Recall Flare
Testing	**0.56**	**0.48**	**0.39**	**0.21**	**0.50**
Validation	**0.50**	**0.43**	**0.33**	**0.31**	**0.47**

Table 4. Column Concatenation using Time Series Forest on original dataset

	HSS	TSS	GSS	FAR	Recall Flare
Testing	**0.59**	**0.52**	**0.42**	**0.24**	**0.55**
Validation	**0.51**	**0.47**	**0.35**	**0.36**	**0.52**

Table 5. Column Concatenation using Time Series Forest on undersampled dataset





4. Conclusion

In our project for classifications of solar flares using climatically under sampled data time series data, we are doing forecasts for solar events in 24 hours windows using machine learning. In our work, we have tried mainly three approaches. After validations of our trained models on unseen data, classification using a column ensemble approach gives us best results. Hence, we suggest using this model for deployment purposes. In the production environment, it is equally important to continuously monitor the performance of the model. We will have to take appropriate measures if for some reason, the model performance will start dropping. As a future work, we can perform various experiments on our model, which will give us a clearer picture about our model performance. We would suggest implementing K fold cross validation, Bootstrapping validation techniques as a part of future work.

References

[1] *Anli Ji, Berkay Aydin, Manolis K. Georgoulis, Rafal Angryk, "All-Clear Flare Prediction Using Interval-based Time Series Classifiers"*

[2] *R. A. Angryk, P. C. Martens, B. Aydin, D. Kempton, S. S. Mahajan, S. Basodi, A. Ahmadzadeh, X. Cai, S. F. Boubrahimi, S. M. Hamdi, M. A. Schuh, and M. K. Georgoulis,* **"***Multivariate time series dataset for space weather data analytics"*

[3] *Monica G. Bobra, Sebastien Couvidat,* **"***Solar Flare Prediction Using SDO/HMI Vector Magnetic Field Data with a Machine-Learning Algorithm"*