Machine Failure Classifier

Data Mining and Machine Learning

Francesco De Vita

Brief Introduction to the Work

This project aims to apply data analysis and machine learning techniques for fault prediction in industrial production, using a real dataset from a metal component manufacturing process. The dataset contains information on various physical and chemical parameters of the process.

Dataset

The dataset we are using is called ‘ai4i2020’ and comes from an industrial production process of metal components.

- **The dataset contains 10 000 rows and 14 columns, each of which represents a feature of the process.**
- **Our goal is to use these features to predict failures and classify them into different types, such as tool wear, overheating, overvoltage, overload and random failures.**

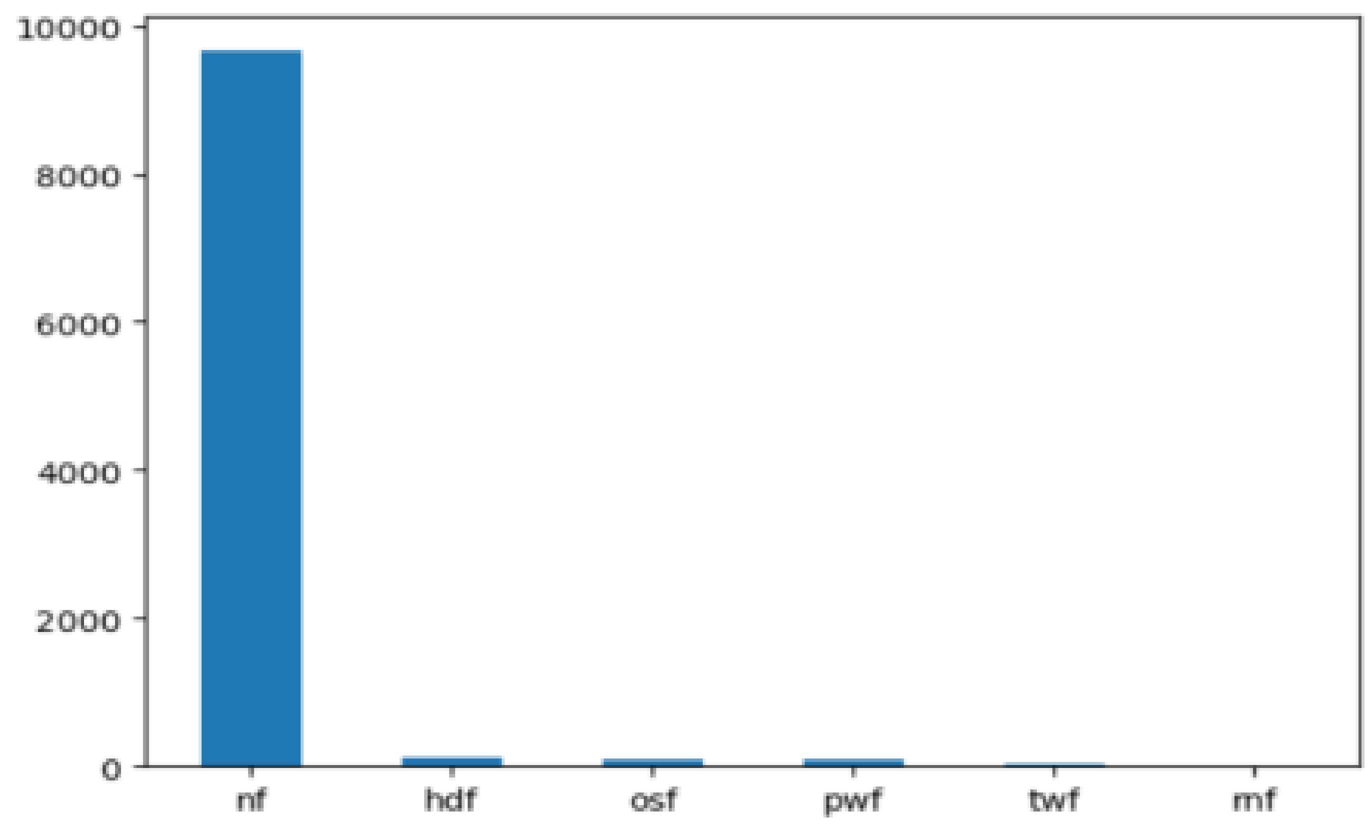
	UDI	Product ID	Type	Air temperature [K]	Process temperature [K]	Rotational speed [rpm]	Torque [Nm]	Tool wear [min]	Machine failure	TWF	HDF	PWF	OSF	RNF
0	1	M14860	M	298.1	308.6	1551	42.8	0	0	0	0	0	0	0
1	2	L47181	L	298.2	308.7	1408	46.3	3	0	0	0	0	0	0
2	3	L47182	L	298.1	308.5	1498	49.4	5	0	0	0	0	0	0
3	4	L47183	L	298.2	308.6	1433	39.5	7	0	0	0	0	0	0
4	5	L47184	L	298.2	308.7	1408	40.0	9	0	0	0	0	0	0

1 Dataset Sample

Dataset

The dataset is really unbalanced because the ‘no failure’ class has too much samples respect to the other classes, that’s why we are going to perform an OverSampling

nf	9652
[hdf]	106
[pwf]	80
[osf]	78
[twf]	42
[rnf]	18
[pwf, osf]	11
[hdf, osf]	6
[hdf, pwf]	3
[twf, osf]	2
[twf, rnf]	1
[twf, pwf, osf]	1



3 Class Distribution

Data Pre-Processing

- Attribute Creation
- Train-Test Split
- Outlier Detection
- Feature Selection
- Oversampling
- Normalization

Attribute Creation

Some of those feature were summarized into one to help the prediction of a specific type of Failure.

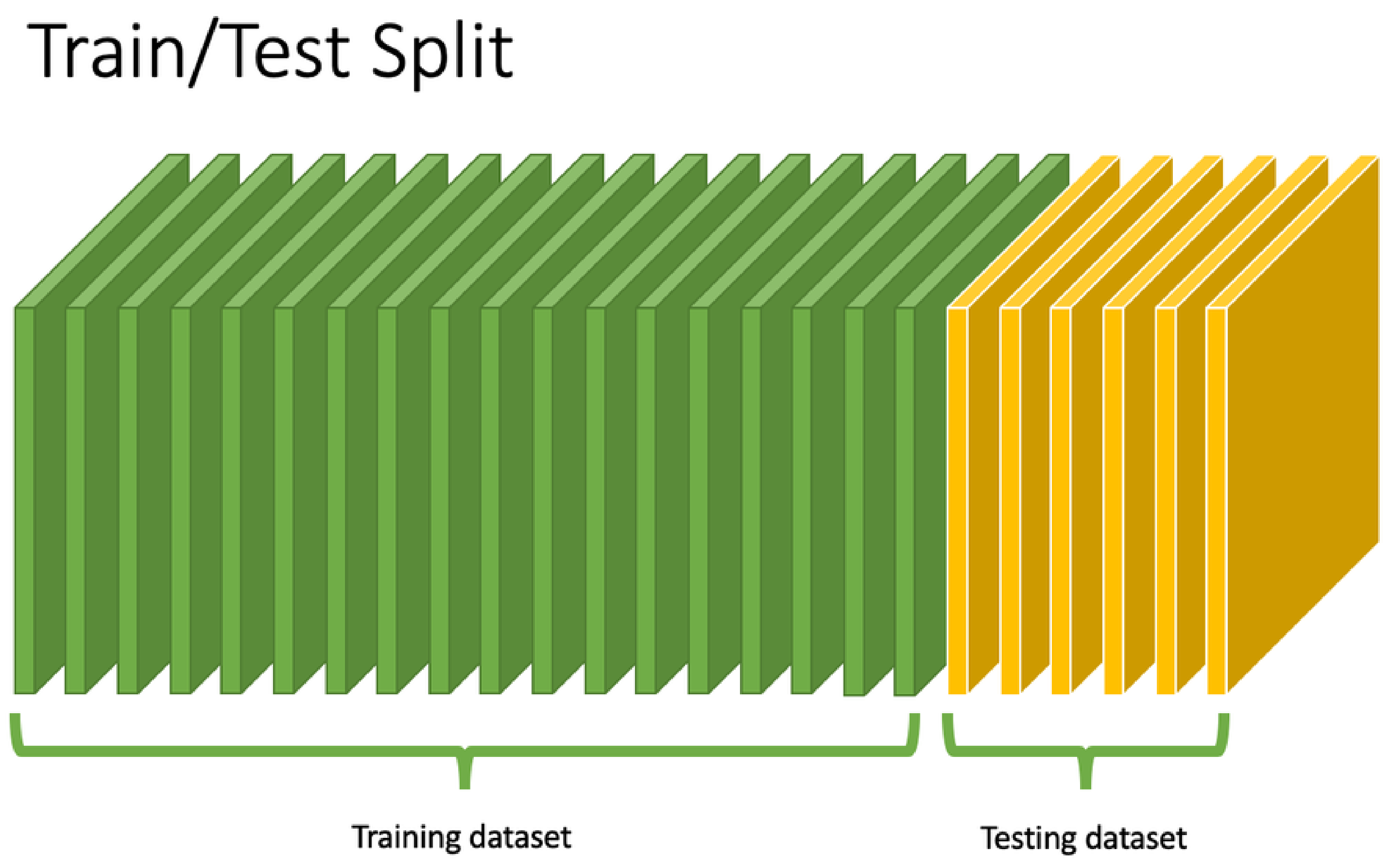
- **power: is the power absorbed by the machine during the process, calculated as the product of the rotational speed and the torque. Usefull to predict power failure (PWF).**
- **tool_torque: Is the product between the input torque and the output torque. Usefull to predict overstrain failure (OSF).**
- **delta_temperature: is the temperature difference between the air and the metal component, calculated as the air temperature minus the process temperature. Usefull to predict heat dissipation failure (HDF)**

```
df['power']=(df['rotational_speed']/60)*(6.28)*df['torque']
df['tool_torque']= df['tool_wear']*df['torque']
df['delta_temperature']= df['air_temperature']-df['process_temperature']
```

2 Attribute Creation

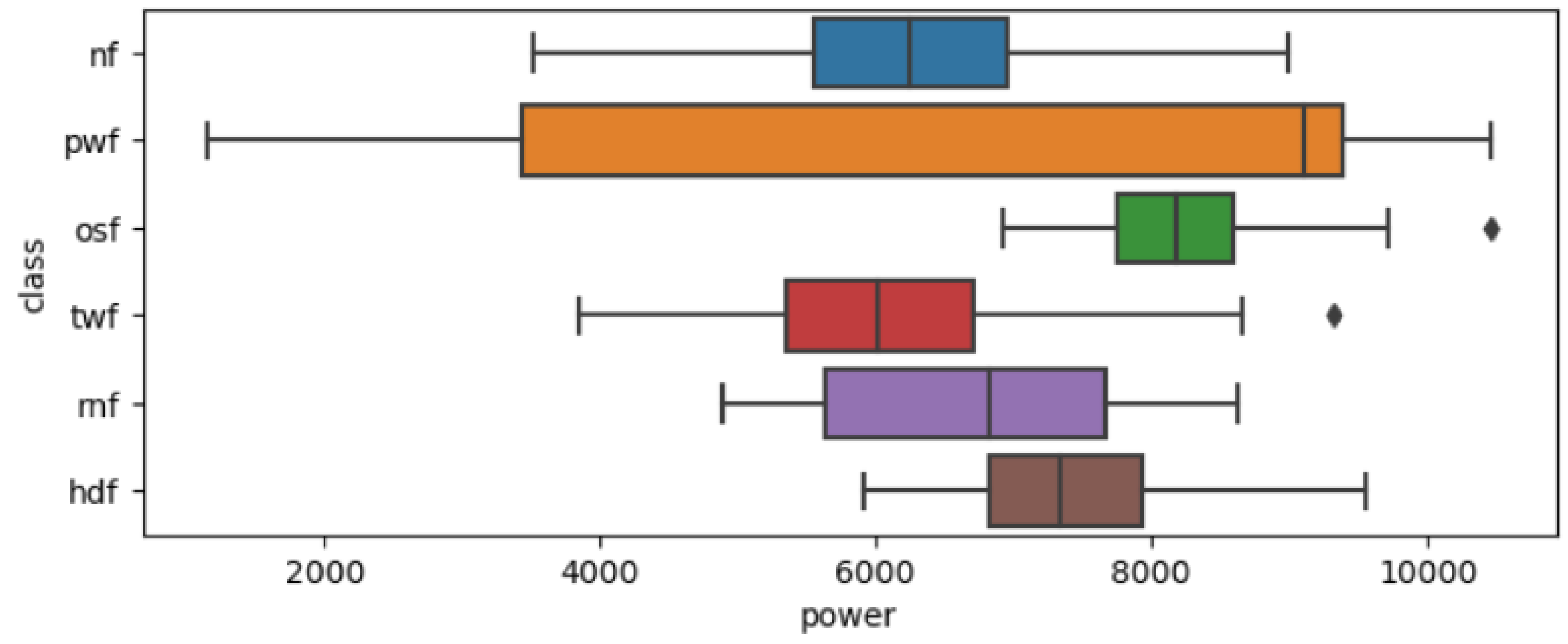
Train-Test Split

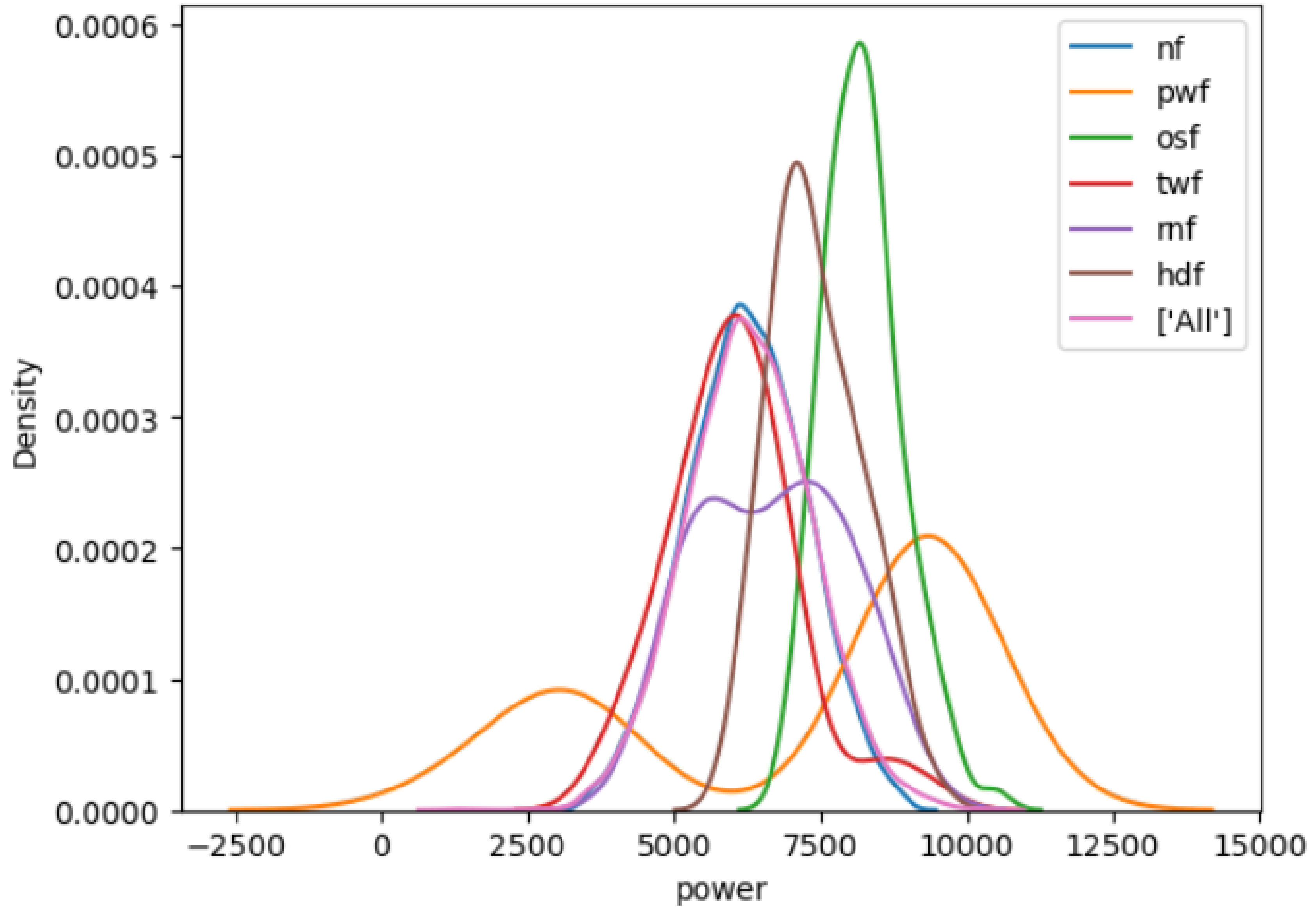
The first step after removing the null values and encode the features the dataset was divided into training and test sets.
We used 30% of the samples as test set with the Stratify option to maintain the same proportion of classes in both train and test set



Outlier Detection

We performed an analysis of the boxplots and the distribution plots of each variable for each Class and no outliers were found.





Features Selection

To select the most relevant features for the classification model, eliminating those that are useless, we used the mutual information criterion. In particular, we applied the "mutual_info_classif" method, which calculates the mutual information between each feature and the class variable, returning a value between 0 and 1.
Than the column with Mutual Information really close to 0 has been dropped from the DataFrame

$$I(C,fi) = \sum_{c \in C} \sum_{f_i \in F} p(c,fi) \cdot \log \frac{p(c,fi)}{p(c)p(fi)}$$

$$NI(f_i,f_s) = \frac{I(f_i;f_s)}{min\{H(f_i),H(f_s)\}}$$

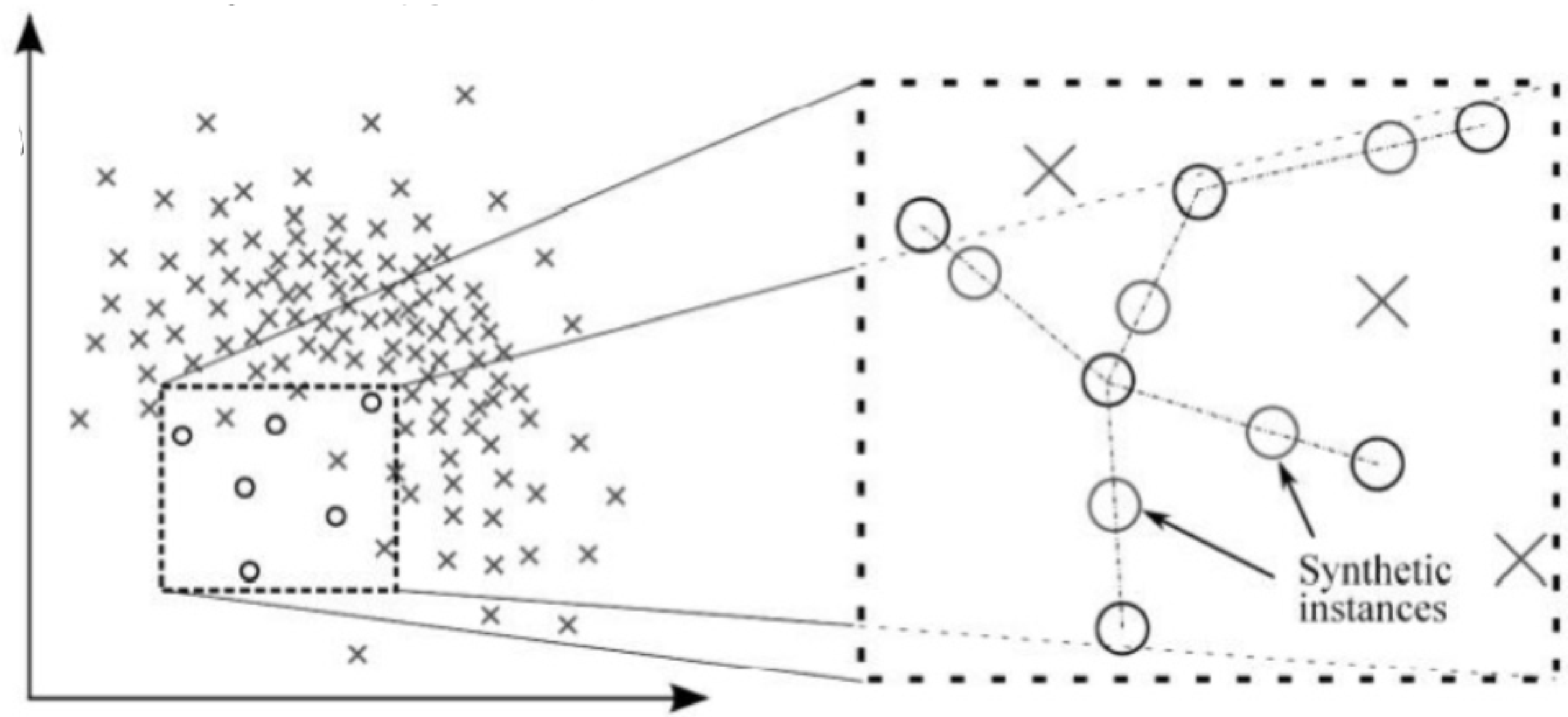
$$G = I(C,f_i) - \frac{1}{S} \sum_{f_s \in S} NI(f_i;f_s)$$

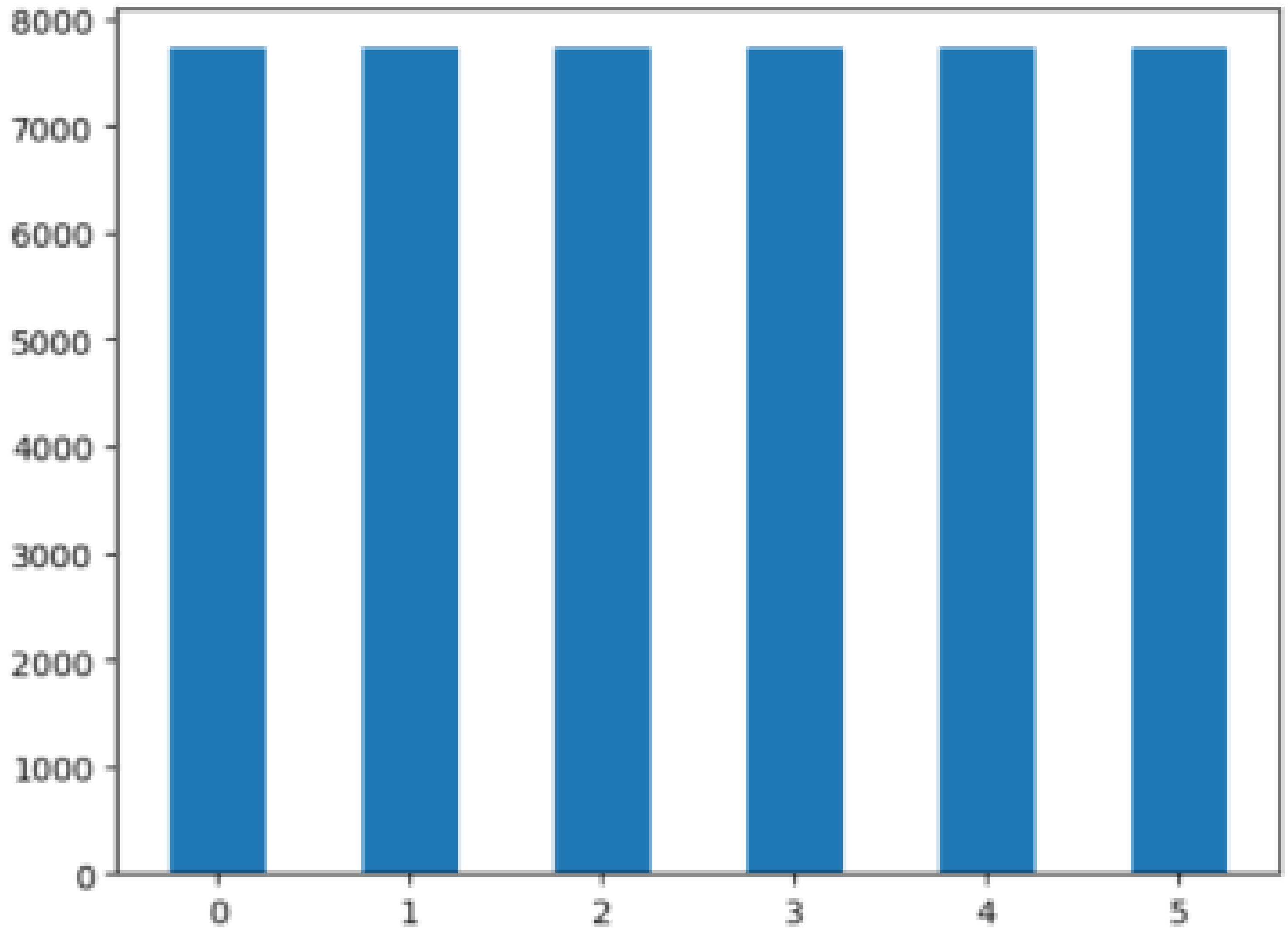
5 Mutual Information

	Score
power	0.076855
torque	0.075924
tool_torque	0.060307
rotational_speed	0.054328
tool_wear	0.028456
delta_temperature	0.028590
air_temperature	0.018997
process_temperature	0.010143
type	0.003298

Oversampling

We use SMOTE oversampling to address the class imbalance problem during the training. This could prevent the model from being biased towards the class with more samples.





4 SMOTE Oversampling

Normalization

Since there are no outliers, we can use MinMax Normalization to make the values of a variable homogeneous and to facilitate the use of the machine learning models.

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

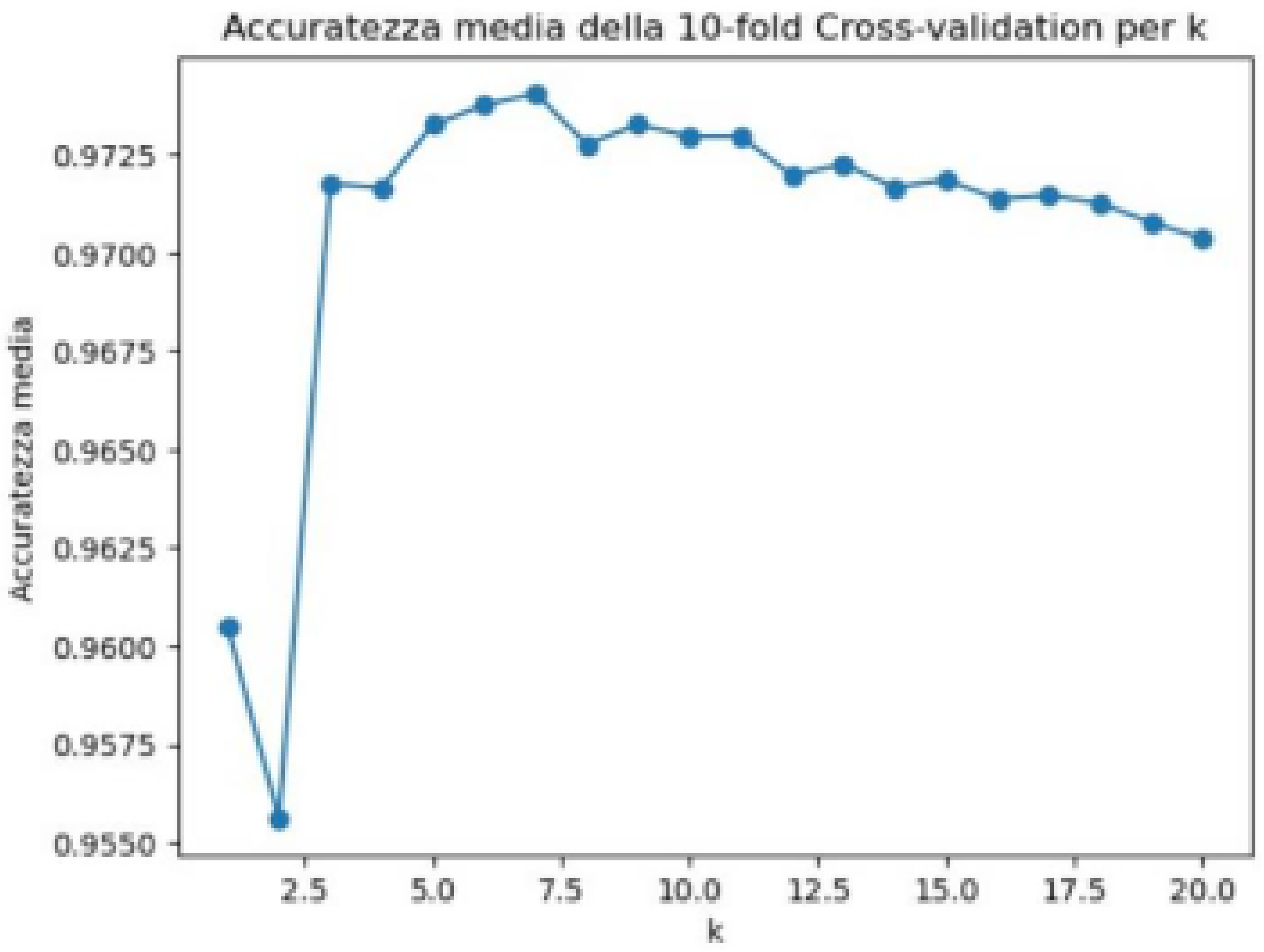
```
from sklearn.preprocessing import MinMaxScaler
sc = MinMaxScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

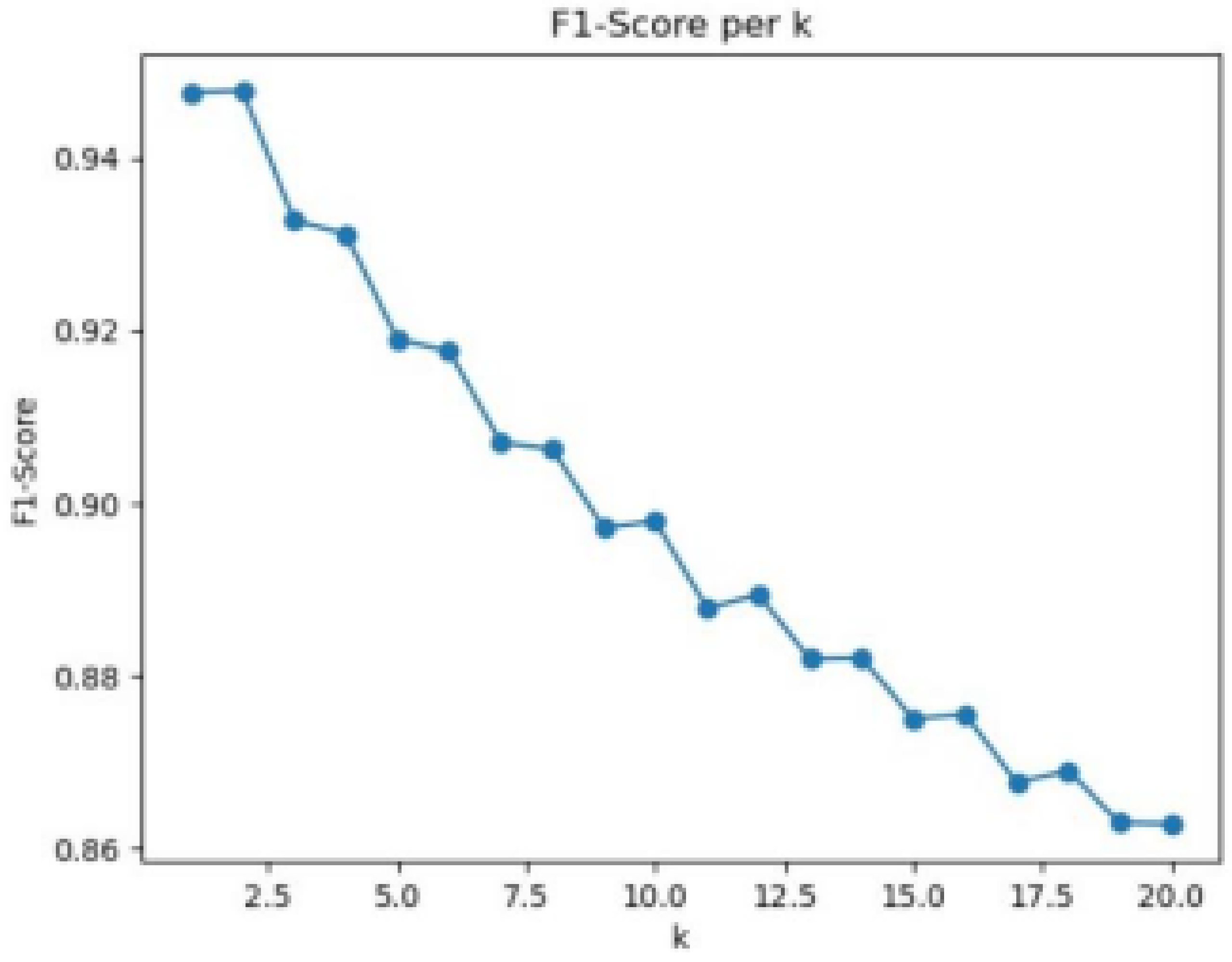
Classification

- KNN
- Decision Tree
- Random Forest
- Naive Bayes

KNN

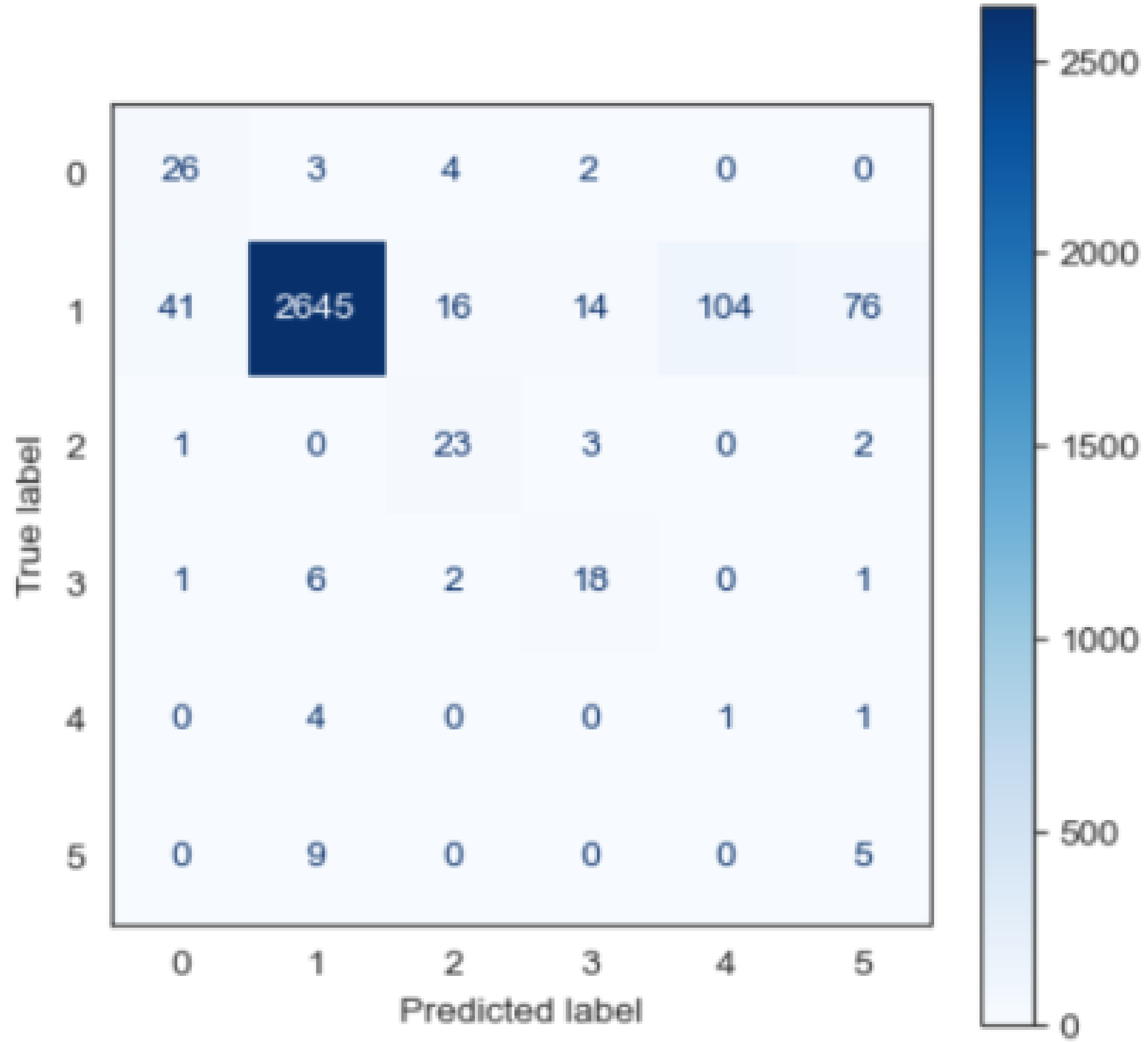
We tried to perform the KNN with different k (from 1 to 20) to understand which was the best value
As we can see from the Image the best k is 3 because it gives us the best F1-Score and accuracy





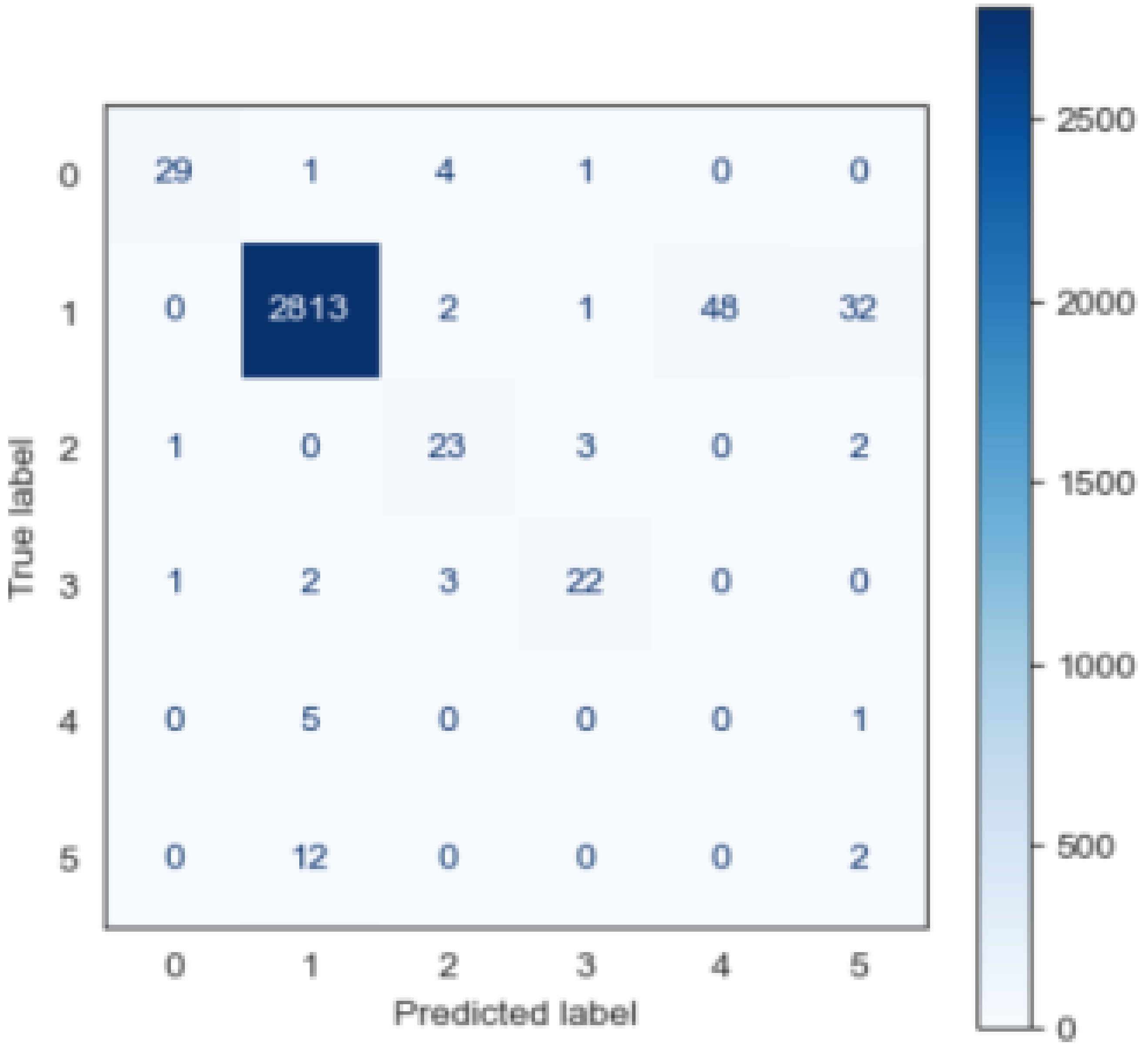
KNN

	precision	recall	f1-score	support
0	0.38	0.74	0.50	35
1	0.99	0.91	0.95	2896
2	0.51	0.79	0.62	29
3	0.49	0.64	0.55	28
4	0.01	0.17	0.02	6
5	0.06	0.36	0.10	14
accuracy			0.90	3008
macro avg	0.41	0.60	0.46	3008
weighted avg	0.97	0.90	0.93	3008



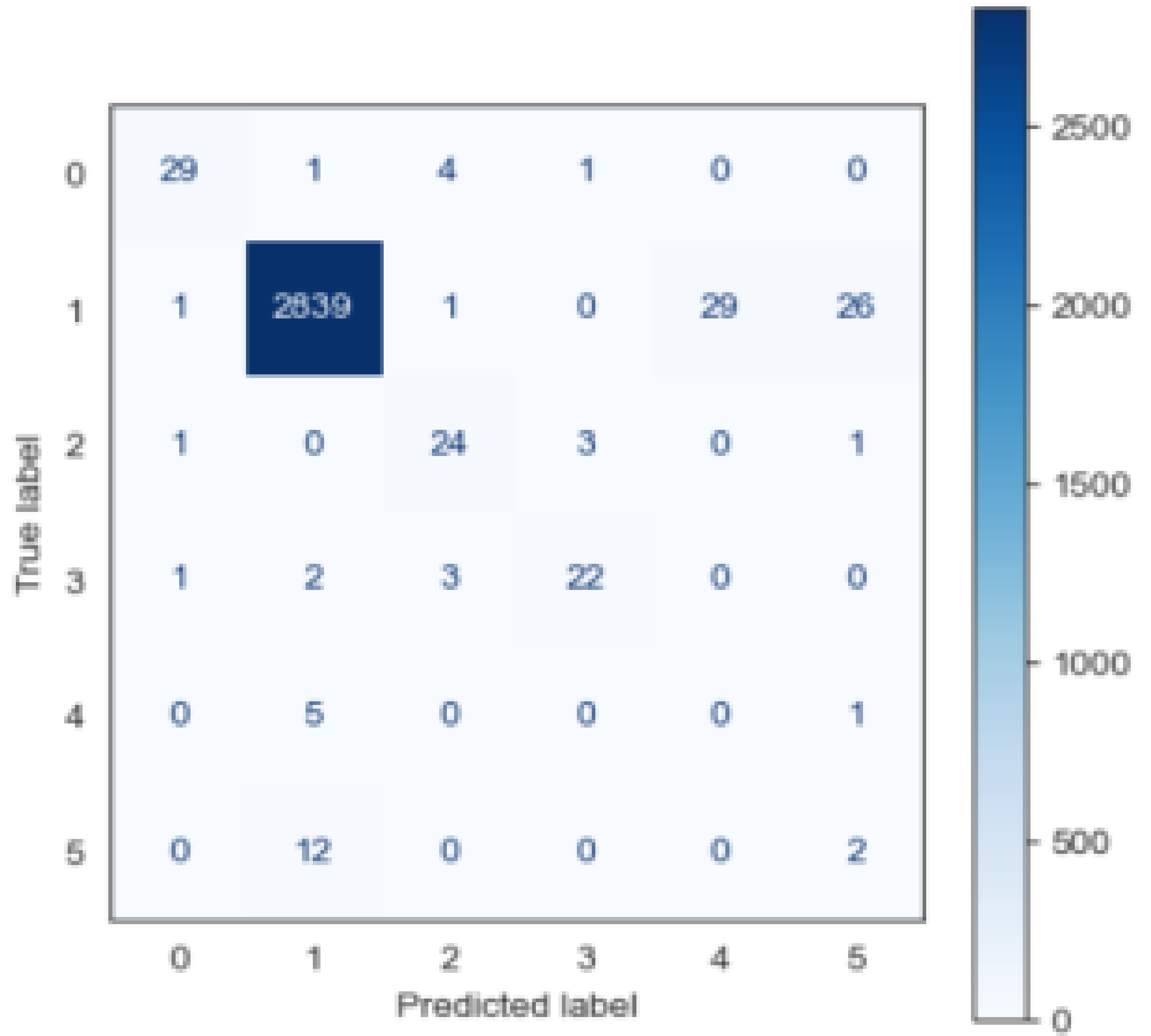
Decision Tree

```
Classification report:
              precision    recall  f1-score   support

           0       0.94      0.83      0.88        35
           1       0.99      0.97      0.98      2896
           2       0.72      0.79      0.75        29
           3       0.81      0.79      0.80        28
           4       0.00      0.00      0.00         6
           5       0.05      0.14      0.08        14

    accuracy                           0.96      3008
   macro avg       0.59      0.59      0.58      3008
weighted avg       0.98      0.96      0.97      3008
```



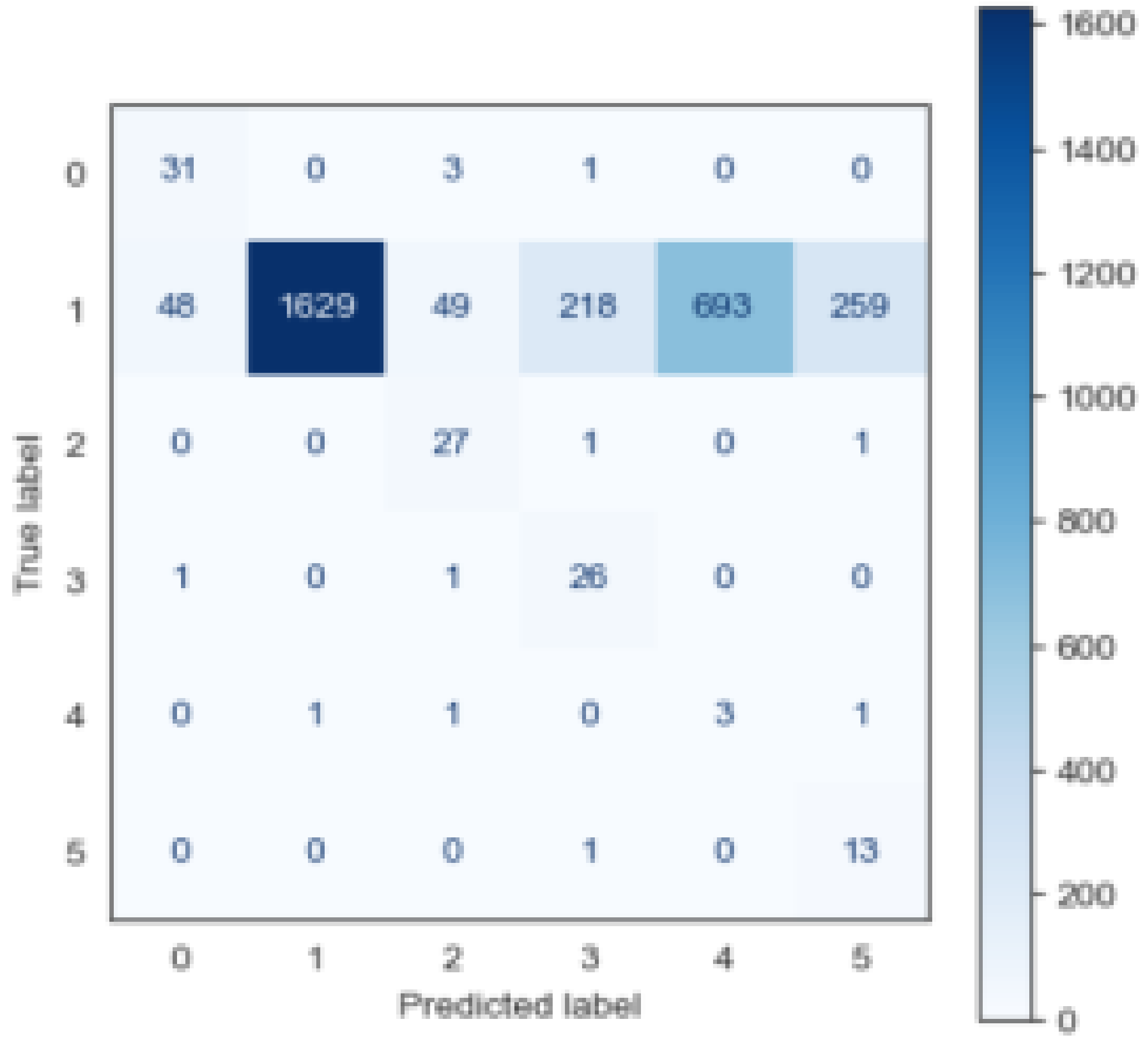
Random Forest

	precision	recall	f1-score	support
0	0.91	0.83	0.87	35
1	0.99	0.98	0.99	2896
2	0.75	0.83	0.79	29
3	0.85	0.79	0.81	28
4	0.00	0.00	0.00	6
5	0.07	0.14	0.09	14
accuracy			0.97	3008
macro avg	0.59	0.59	0.59	3008
weighted avg	0.98	0.97	0.98	3008



Bayesian Classifier

	precision	recall	f1-score	support
0	0.39	0.89	0.54	35
1	1.00	0.56	0.72	2896
2	0.33	0.93	0.49	29
3	0.11	0.93	0.19	28
4	0.00	0.50	0.01	6
5	0.05	0.93	0.09	14
accuracy			0.57	3008
macro avg	0.31	0.79	0.34	3008
weighted avg	0.97	0.57	0.71	3008



Results Comparison

From the results of the classification report and the confusion matrix, it emerges that the most performing classifier is the Random Forest, which achieved the highest values in all the metrics.

	hdf	nf	osf	pwf	rnf	twf
KNN	0.50	0.95	0.62	0.55	0.02	0.10
Decision Tree	0.88	0.98	0.75	0.80	0.00	0.08
Random Forest	0.87	0.99	0.79	0.81	0.00	0.09
Bayesian Classifier	0.54	0.72	0.49	0.19	0.01	0.09

F1-Score for each Class

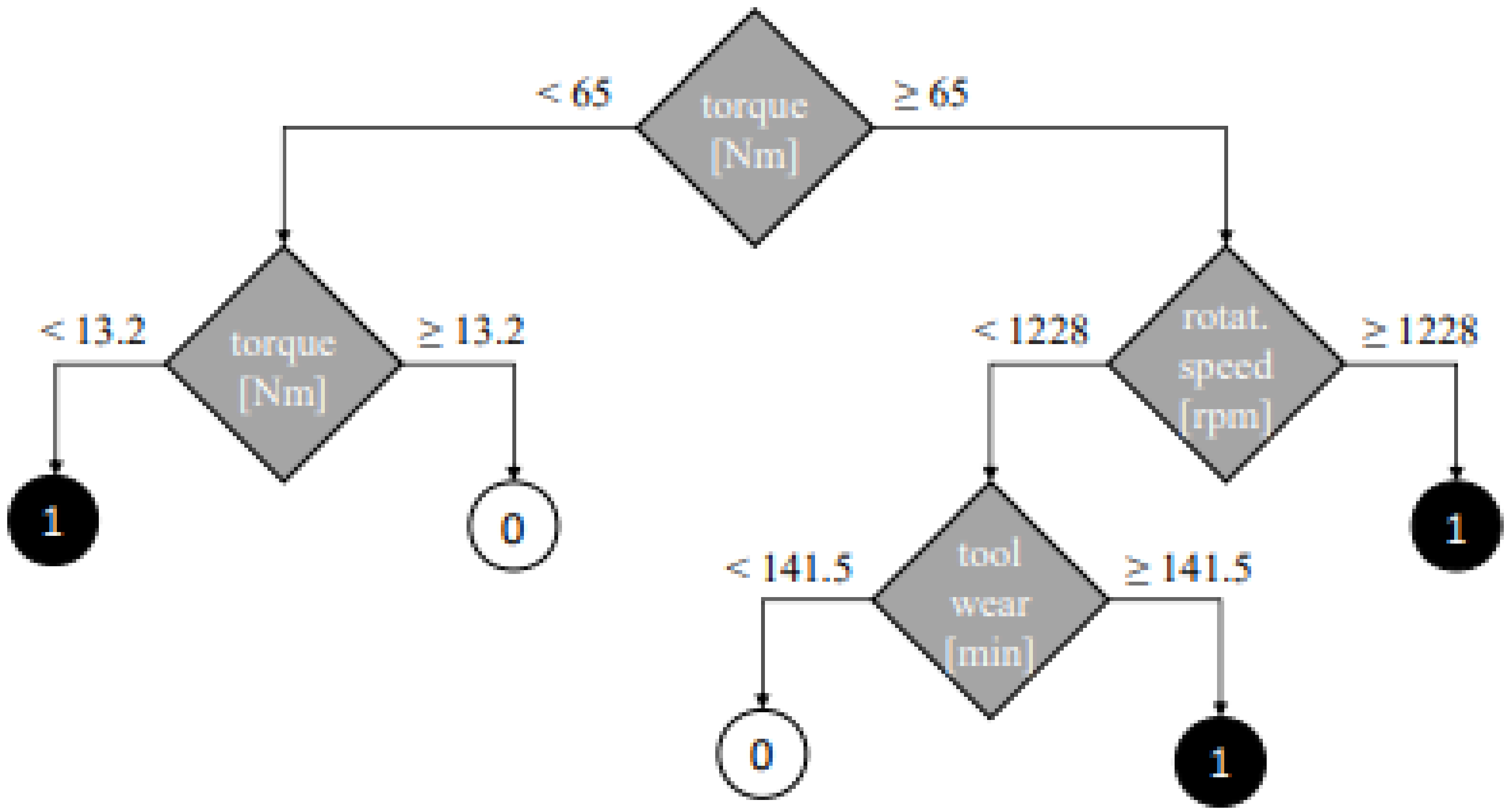
	KNN	Decision Tree	Random Forest	Bayesian Classifier
F1-Score	0.93	0.97	0.98	0.71

Average F1-Score

Comparison with other Works

“Explainable Artificial Intelligence for Predictive Maintenance Applications” by S. Matzka

In this paper they used a Bagged Tree Model composed by 15 Trees, each tree has built only on 4 different features to be more understandable from human



Comparison with other Works

We have quite the same result as this model but their has an advantage on the Understandability and the Simplicity of the Results

	precision	recall	f1-score	support
0	0.91	0.83	0.87	35
1	0.99	0.98	0.99	2896
2	0.75	0.83	0.79	29
3	0.85	0.79	0.81	28
4	0.00	0.00	0.00	6
5	0.07	0.14	0.09	14
accuracy			0.97	3008
macro avg	0.59	0.59	0.59	3008
weighted avg	0.98	0.97	0.98	3008

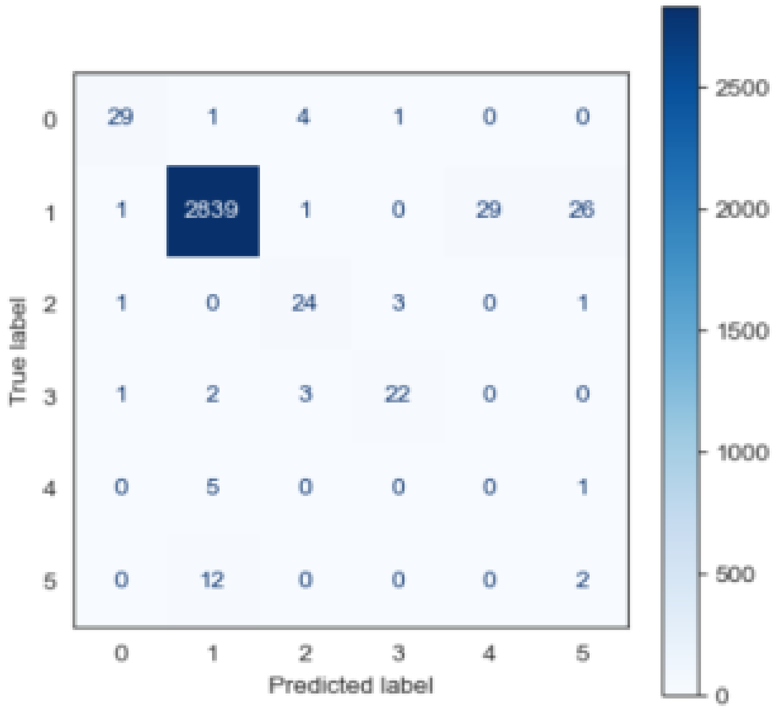


TABLE I. CONFUSION MATRIX OF THE BAGGED TREES ENSEMBLE CLASSIFIER USING 5-FOLD CROSS VALIDATION.

		true class	
		failure	operation
predicted class	failure	294 (86.7 %)	45 (13.3 %)
	operation	121 (1.3 %)	9,540 (98.7 %)

$$Precision = \frac{9540}{9540 + 121} = 0.98$$

$$Recall = \frac{9540}{9540 + 45} = 0.99$$

$$F1 - Score = \frac{2 * 0.98 + 0.99}{0.98 + 0.99} = 0.98$$

Comparison with other Works

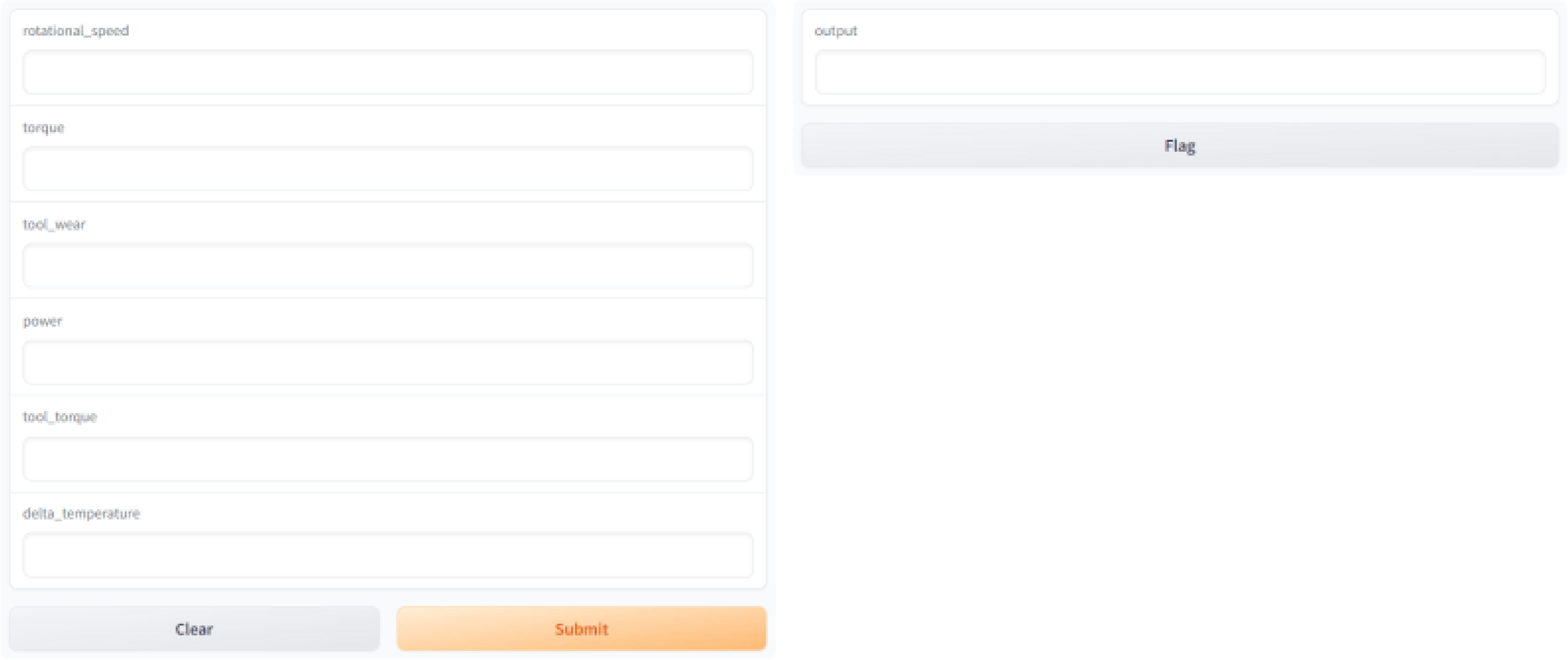
In this domain, the explainability of the results is essential. That is why the paper's evaluation focused on the model's explanation rather than the classification results.
To improve the explanation quality, they applied the Z-Score normalization and selected the best 2 Features in absolute value for explanation.

Comparison with other Works

UID	Mode	air temp.	proc. temp.	rot. speed	torq.	tool wear
2672	TWF	299,7	309,3	1399	41,9	221
		-0,15	-0,47	**-0,78**	0,19	**1,78**
3237	HDF	300,8	309,4	1342	62,4	113
		0,40	-0,41	**-1,10**	**2,25**	0,08
464	PWF	297,4	308,7	2874	4,2	118
		-1,30	-0,88	**7,45**	**-3,59**	0,16
250	OSF	298	308,3	1405	56,2	218
		-1,00	-1,15	-0,75	**1,63**	**1,73**

UID	Mode	BTC	Explanation	Scr
2672	TWF	0	high tool wear of 221 mins low rot. speed of 1399 rpm	+
3866	TWF	1	high tool wear of 228 mins high air temp. of 302.6 K	+
6341	TWF	0	high tool wear of 210 mins low rot. speed of 1397 rpm	+
8358	TWF	0	high tool wear of 210 mins low rot. speed of 1397 rpm	+
9019	TWF	0	high tool wear of 217 mins low air temp. of 297.3 K	+
3237	HDF	0	high torque of 62.8 Nm low rot. speed of 1342 rpm	+
4079	HDF	1	high torque of 62.8 Nm low rot. speed of 1294 rpm	+
4174	HDF	1	high air temp. of 302.2 K low rot. speed of 1346 rpm	+ +
4327	HDF	1	high torque of 55.8 Nm low rot. speed of 1362 rpm	+
4502	HDF	1	high torque of 54.0 Nm low rot. speed of 1307 rpm	+ +
464	PWF	1	high rot. speed of 2874 rpm low torque of 4.2 Nm	+
1493	PWF	1	high torque of 58.5 Nm low air temp. of 298 K	+
3001	PWF	1	high torque of 72.8 Nm low rot. speed of 1324 rpm	+
7537	PWF	1	high rot. speed of 2579 rpm low torque of 12.5 Nm	+
8583	PWF	1	high torque of 72.8 Nm low proc. temp. of 308.1 K	+
250	OSF	1	high tool wear of 218 mins high torque of 56.2 Nm	+ +
3020	OSF	1	high tool wear of 207 mins high torque of 54.2 Nm	+ +
5400	OSF	1	high tool wear of 218 mins high air temp. of 302.8 K	+
7592	OSF	1	high torque of 61.3 Nm high tool wear of 202 mins	+ +
9660	OSF	1	high torque of 61.9 Nm high tool wear of 216 mins	+ +

User Interface



Possible Improvements

- Adding to the user Interface an Explanation of the Decision
- Train the model on Dataset with better Data Quality
- Implementation with sensors for a real-time Prediction