Social Media Sentiment Analysis

Using Pyspark

CISC

Team 2

Team members:	ID:
Omar, Naglaa	20399134
Elmongi, Salma	20398567
Mohamed, Shahd	20401743
Ragab, Salma	20399135
Hassan, Yara	20398563

Submitted to:

Dr. Anwar Hussein

This project is all about social media analytics and how analytics can help you understand what works for your competitors and their audience.
Social media provides a platform for people to express their opinions and thoughts about a topic, event, or product. By analyzing their sentiments, we can find out what interests them and how people feel about these issues.

The project is focused on understanding people's opinions on the 2019 House of Representatives candidate for Prime Minister.

We will have to use machine learning algorithms, such as logistic regression, naïve bayes, etc. this will involve many tasks including preprocessing, feature extraction, model fitting, and validation phases.

And in the end, we will visualize the results of our data analysis.

Highlights of this project:

1. Data Collection .
2. Analyze the sentiments of the texts from the data collected.
3. Gain useful knowledge after processing the collected data.
4. Compare contemporary text classification machine learning algorithms and justification.

1. Data Collection:

We search In Kaggle for a Social Media Sentiment data to reprocess it and use it in our project, we found Twitter and Reddit data sets we merged them together did reprocessing on them to use it in the project.

START OUR WORK:

In the beginning we Install Pyspark on Google Collab to start work on it.

Then we install the necessary libraries and move on to reprocess the data.

2- Data Preparation:

Our data schema:

- First, we have two social media datasets
    1. Twitter dataset
    2. Reddit dataset

- We Merge twitter dataset and reddit dataset in one data frame with **230436** data entries.

- Our data consists of two columns: column of the **clean_text**: String Type. and column of the **category**: Integer Type.
  The categories are:
  positive -> 1 negative -> -1 neutral -> 0

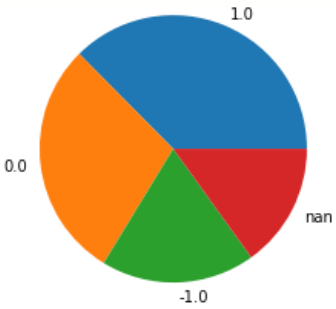
- We detect the outliers in the merged dataset, and it hasn't no outliers.

```
#detect outliers in our dataset
new_df = find_outliers(df)
new_df.show()
```

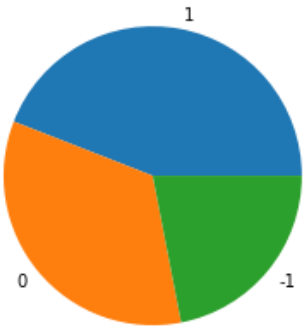
- We search for null values in our dataset and found that it has null values.

```
# Find Count of Null, None, NaN of All DataFrame Columns
from pyspark.sql.functions import col,isnan, when, count
df.select([count(when(isnan(c) | col(c).isNull(), c)).alias(c) for c in df.columns]
  ).show()
```

- +----------+--------+
- |clean_text|category|
- +----------+--------+
- | 104| 34755|
- +----------+--------+



- Pie chart after we drop the null values from the data, so the data entries remaining is **195578**.



Then we used Spark Machine Learning Pipelines API which is like Scikit-Learn to preprocess our data. Our pipeline includes three steps:

1- RegexTokenizer
2- StopwordsRemover
3-CountVectors
to filter our data and remove stop words from it such as: [https, http, amp, rt, t, c, the] because they will not affect our data.

- We split the data using random Split to **0.70 training** and **0.30 testing**.

- We describe the training data with showing the data statistical information by describe method:

```
+-------+----------------------+------------------+------------------+
|summary|            clean_text|          category|             label|
+-------+----------------------+------------------+------------------+
|  count|                136811|            136811|            136811|
|   mean|                   NaN|0.2217877217475203|0.7782122782524797|
| stddev|                   NaN|0.7823116472855645|0.7823116472855647|
|    min|                      |                -1|               0.0|
|    max| ( ͡° ͜ʖ ͡° ╾☆·。 ⊂ ...|                 1|               2.0|
+-------+----------------------+------------------+------------------+
```

- We describe the testing data with showing the data statistical information by describe method:

```
+-------+--------------------+------------------+------------------+
|summary|          clean_text|          category|             label|
+-------+--------------------+------------------+------------------+
|  count|               58767|             58767|             58767|
|   mean|              6949.8|0.2207531437711641|0.7792468562288359|
| stddev|  12787.608013229057|0.7806099070639413|0.7806099070639412|
|    min|                    |                -1|               0.0|
|    max|�modi mistake� ch...|                 1|               2.0|
+-------+--------------------+------------------+------------------+
```

3- Machine learning algorithms:

To Analysis This Project We Used 5 Different Algorithms

then we compare the performance of each model to each other

1) Logistic Regression
2) Naïve Bayes
3) Decision Trees
4) Random forest
5) One VS rest

In logistic Regression we used TF-IDF algorithm that can be broken down into two parts TF (term frequency) and IDF (inverse document frequency*).*
It is a handy algorithm that uses the frequency of words to determine how relevant those words are to a given document, so that was a try to increase the accuracy but unfortunately the accuracy decreased so we neglected it.

Also In logistic Regression we got `0.7635389430397557` for accuracy then we used Cross Validation and found that the accuracy increased to `0.8216800367152399` but we decided to keep the first accuracy because in the other models we didn't use Cross validation, to achieve equality between the models.

The performance of each Model:

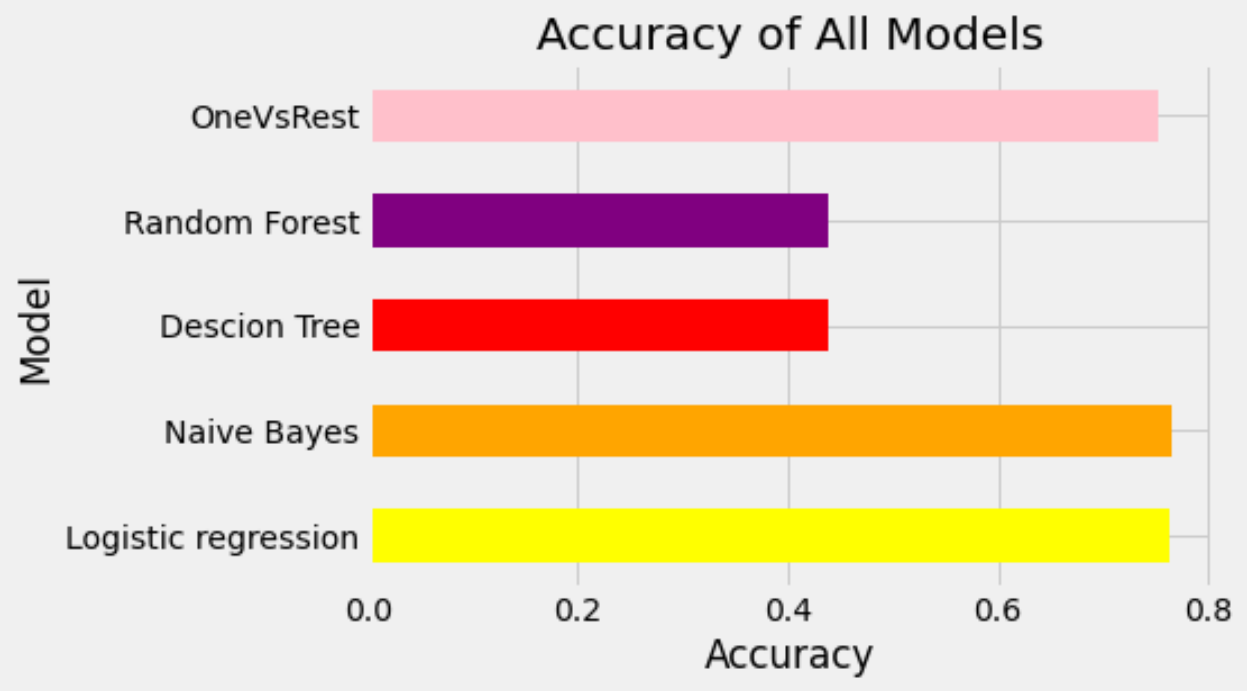
	Accuracy	f1	weightedPrecision	weightedRecall	weightedFalsePositiveRate
Logistic regression	0.763539	0.737919	0.756323	0.748464	0.154443
Naive Bayes	0.764851	0.763999	0.767959	0.764851	0.140228
Descion Tree	0.439464	0.269317	0.307442	0.439464	0.439107
Random Forest	0.439413	0.268282	0.193084	0.439413	0.439413
OneVsRest	0.751953	0.738523	0.766963	0.751953	0.156499

3- Data Visualization:

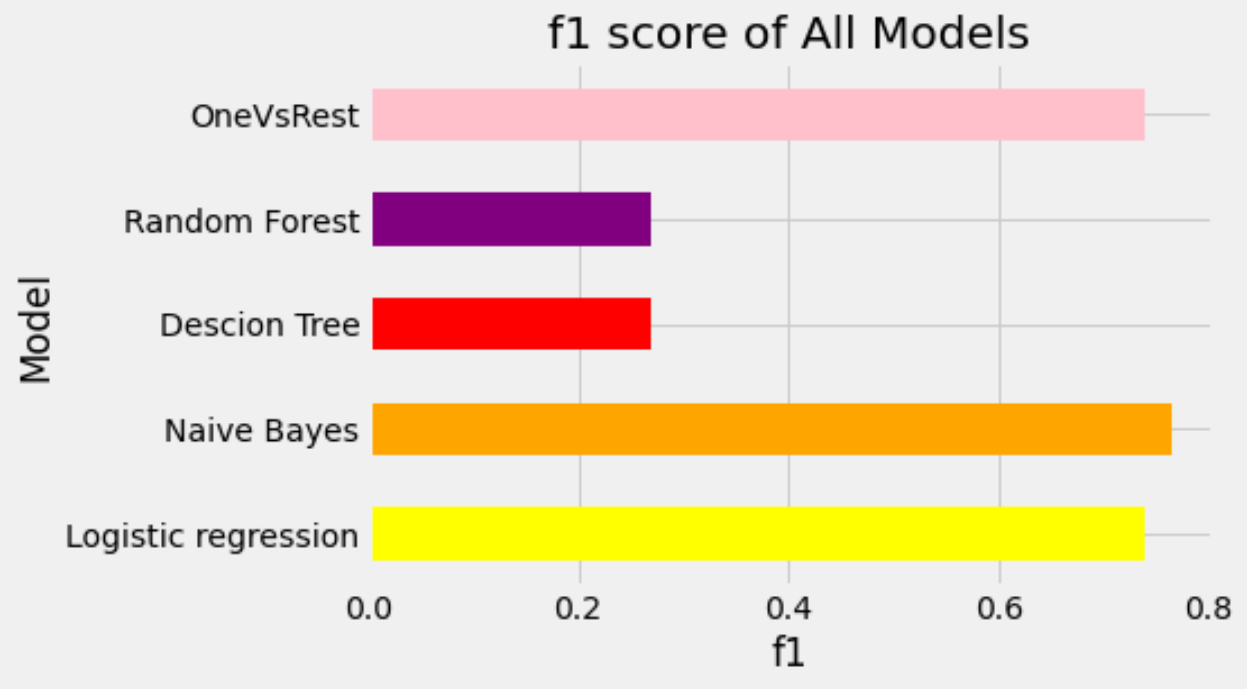
There are different techniques for data Visualization: they are diagrams, charts, and graphs. While Matplotlib, Seaborn, and Folium are three Python libraries that are used for data visualization.

We used bar plot to visualize each model performance and visualize the comparison between them. and used Histogram to visualize the distribution of the categories we have.

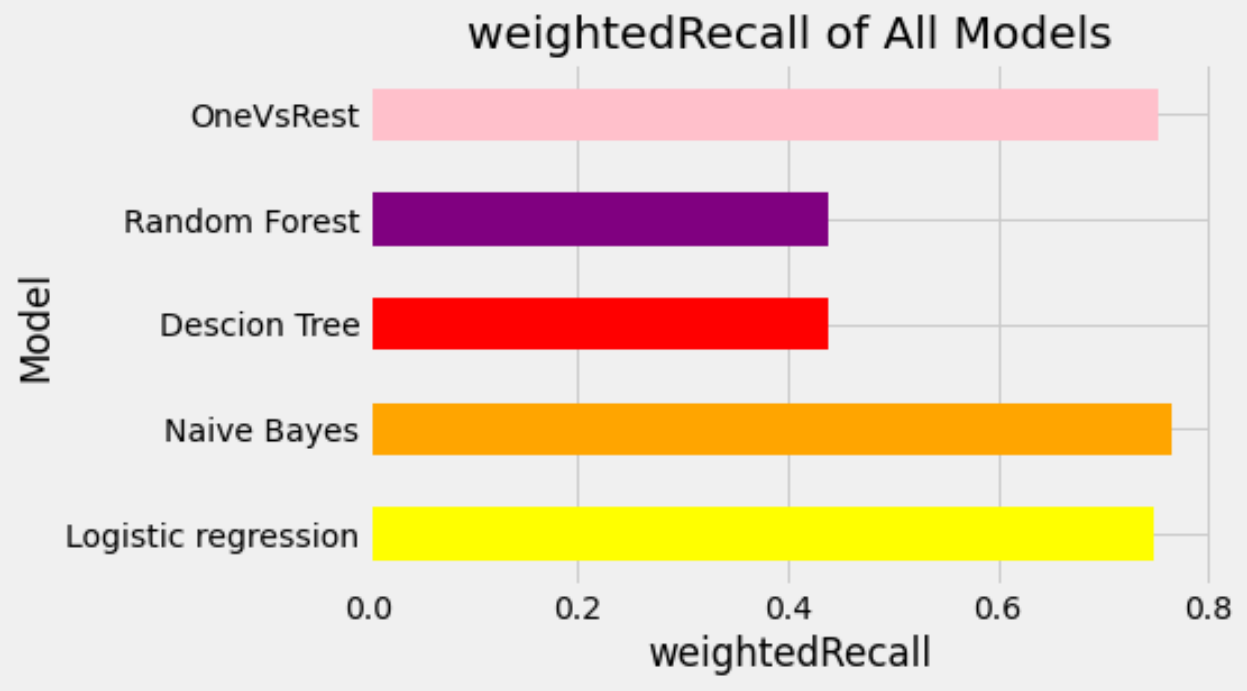
The first graph shows the accuracy of All Models.



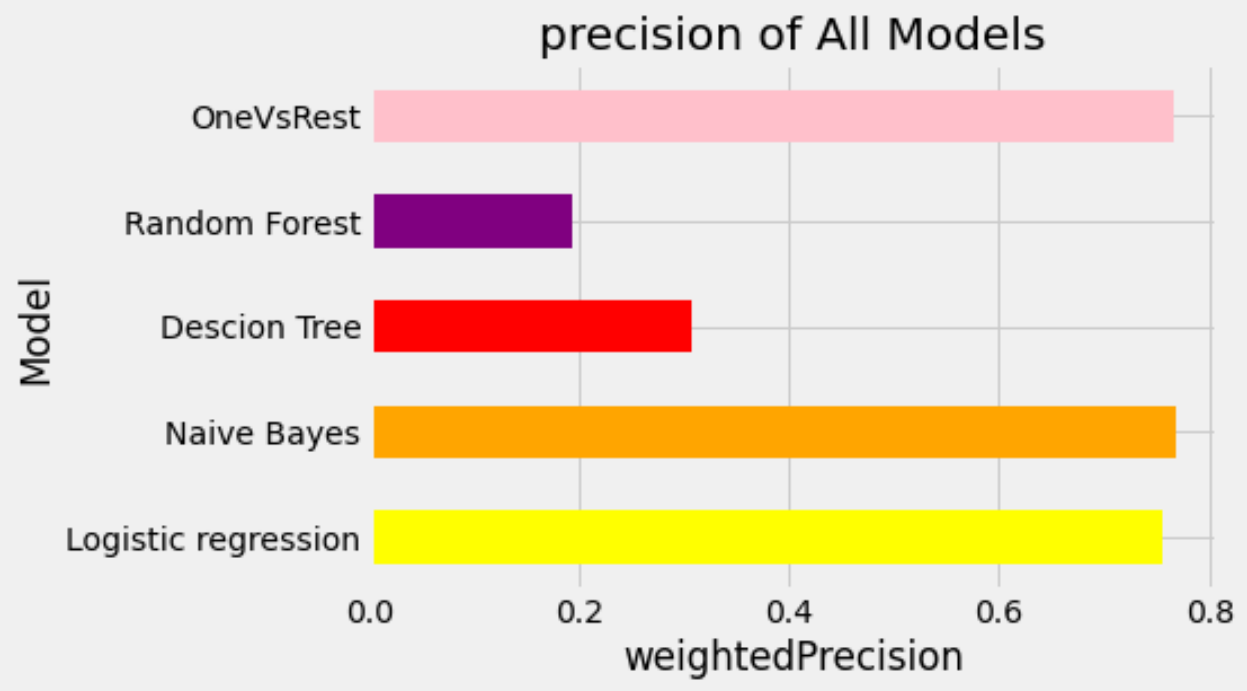
This graph shows the F1_Score of all models.



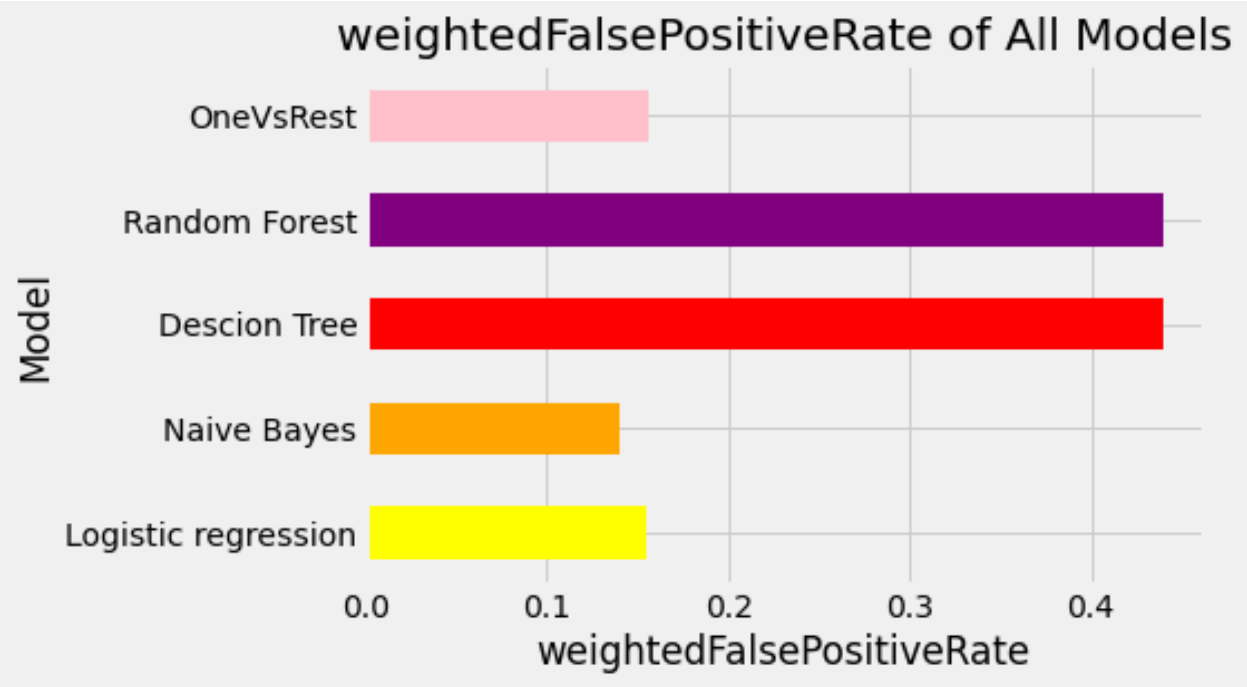
This graph shows the Weighted Recall of all models.



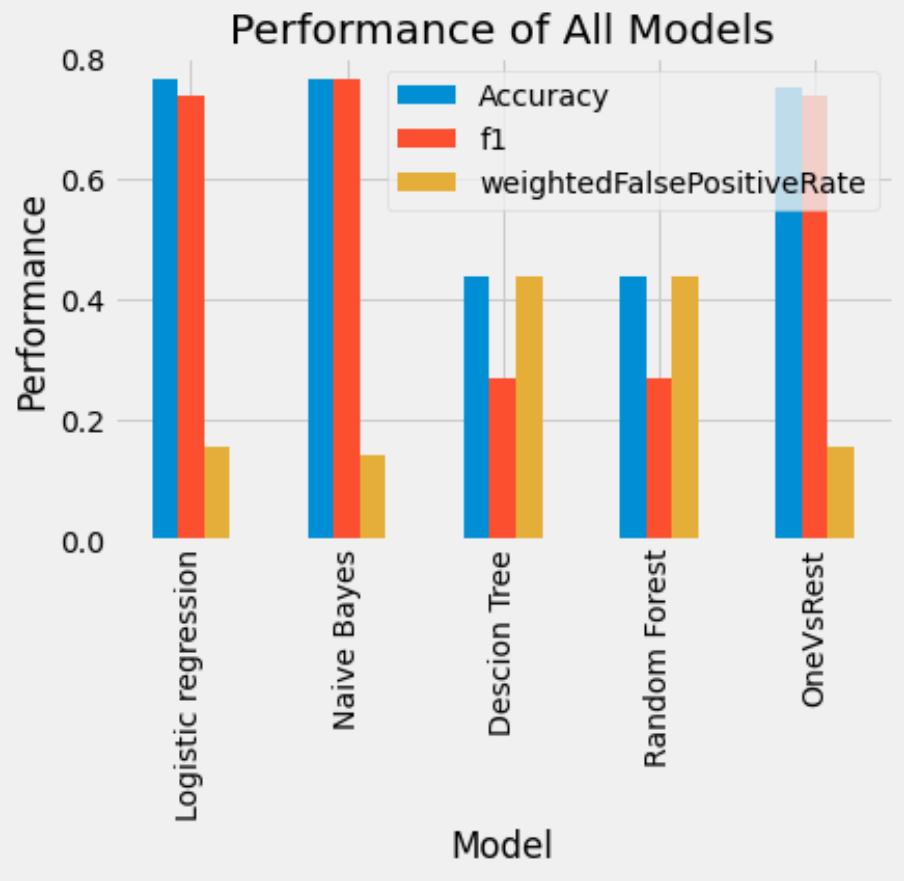
This graph shows the Recall of all models**.**



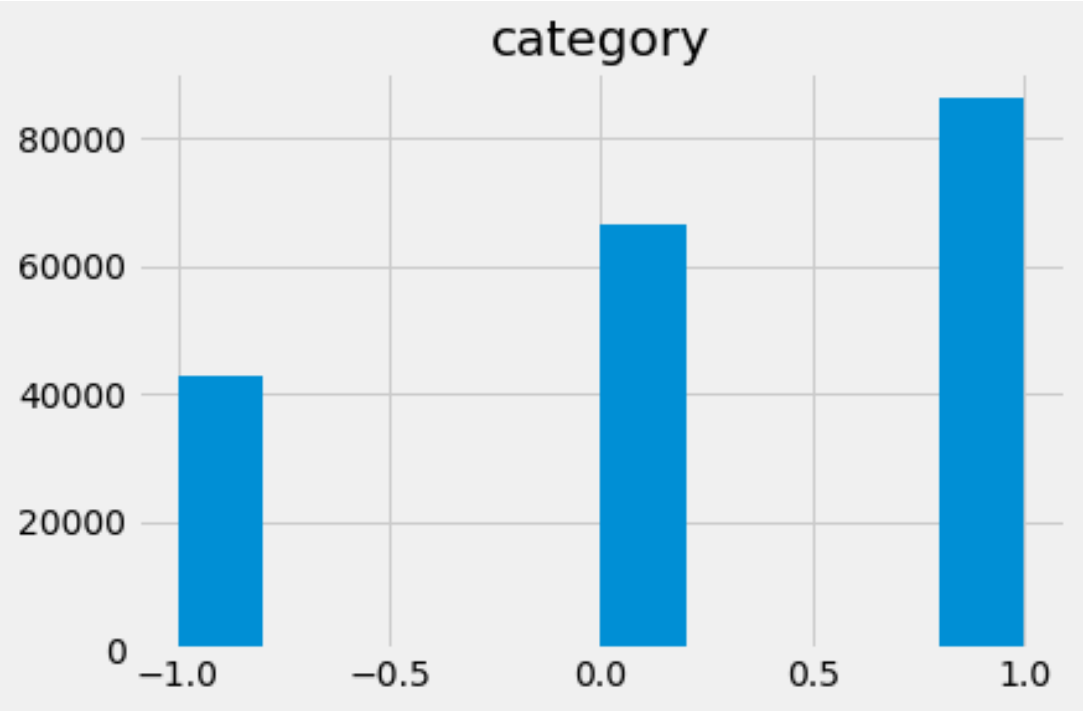
This graph shows the False Positive Rate of all models.



This graph shows the Performance of all models



This graph shows the distribution of the categories we have.



Project Work Load:

Names	Roles
Naglaa Omar	**Worked in Data Collection and participate in Data Analysis and writing the Report.**
Salma Elmongi	**Worked in Data Visualization and participate in writing the Report.**
Salma Yasser	**Worked in Data Visualization and participate in writing the Report.**
Shahd Mohammed	**Worked in Data Collection and participate in Data Analysis and writing the Report.**
Yara Mohammed	**Worked in Data Preprocessing and Participate in Data Analysis and writing the Report.**