**Project Proposal**

**Title of the Project - Predictive Potential Spammer**

## Brief on the project :

Recently attackers are using freelance job sites such as Fiverr to distribute malware disguised as job offers. These job offers contain attachments that pretend to be the job brief but are actually installers for keyloggers such as Agent Tesla or Remote Access Trojan (RATs). Due to this many users lost their earnings, bidding fees and fake client projects, also some users lost their accounts too. Many of my LinkedIn connections faced it and some of them lost their professional growth, side income and stability.

Just after a month of this attack Fiverr come-up with a Kaggle competition on this to understand how data science people will solve this problem by using their differ methods and techniques.

**Objective : -** Find a good fit algorithm which will have Potential to Predict Spammers.

**Deliverables of the project :**

1. A high-level description of the general approach you will use to address the problem. This should include how you will evaluate and what evidence you are planning to gather (e.g. how you can solve the problem through experiments on data)

* List of questions your model/problem are designed to answer
* Details of the model , important findings, expecting observations and outcome
* Firstly assigining the variable by name “PPS\_df” to fetch the data from the csv file of fiverr\_data.
* This data contain 458798 rows and 53 columns.
* How the data looks like ?

By using head ,tail,sample method we can see that how the data

looks like.

Target variables - So , there is a label column which is dependent variable.

Independent variables - 'user\_id', 'X1', 'X2', 'X3', 'X4', 'X5', 'X6', 'X7', 'X8',

'X9', 'X10', 'X11', 'X12', 'X13', 'X14', 'X15', 'X16', 'X17', 'X18',

'X19', 'X20', 'X21', 'X22', 'X23', 'X24', 'X25', 'X26', 'X27', 'X28',

'X29', 'X30', 'X31', 'X32', 'X33', 'X34', 'X35', 'X36', 'X37', 'X38',

'X39', 'X40', 'X41', 'X42', 'X43', 'X44', 'X45', 'X46', 'X47', 'X48',

'X49', 'X50', 'X51'

* **checking the data types-**

By using the dtype of and also by using info method

So, there we have integer data and X13 is of float may be because it contain null values .

* **checking Nulls values of the data –:**

Using isnull method to check the null values of the data.

So,there we have some null values which is contained by the column X13.

**🡪 What if data containing specified characters?**

Assigining the variable name specified\_char which and creat one list which contain the specified characters.

And applying isin function and then apply the conditional loop(if else) to check wheather data is contain specified characters or not.

**🡪checking the correlation of the data-:**

A correlation coefficient close to 1 indicates a strong positive correlation, while a value close to -1 indicates a strong negative correlation.

**1. Correlation Direction:**

Positive correlation means that as one variable increases, the other tends to increase.

Negative correlation means that as one variable increases, the other tends to decrease.

For instance, in the label row, the correlation with X1 is -0.032693, indicating a slight negative correlation.

2. Missing Values:

The presence of NaN (Not a Number) values in the correlation matrix indicates missing or undefined values for some pairs of variables.

3. Highly Correlated Variables:

Identifying pairs of variables with high correlation (close to 1 or -1) may indicate redundancy.

For example, X6 and user\_id have a correlation coefficient of 0.723465, suggesting a strong positive correlation.

4. Correlation Magnitude:

The magnitude of the correlation coefficient indicates the strength of the relationship.

A correlation coefficient close to 0 suggests a weak or no linear relationship.

In the X42 row, the correlation between X3 and X9 is 0.045473 , suggesting a relatively weak positive correlation.

* Using mode to handle the null values in X13.
* Using Heat map to visualize the data and clearly we see that { 'X27','X29', 'X30', 'X33', 'X46', 'X47', 'X48' } contains lots of null values.

Drop those null values.

* Checking the outliers in the data. By using iqr method, and visualize it by using box plot.
* Giving range from 0 to 1 to the data ,By using MinMax scalar method.

In using minmax scalling because naïve bayse dose not perform on the standard scaling method (gives negative values also).

🡪 Splitting independent and dependent data.

Independent variable – X

Dependent variable – y (label)

* Handling the null values by replacing them by there mean , by using simpleimputer.
* check whether the dependent variables is labeled or not, by using .count method

0 446477

1. 12321

Label this data by using SMOTE method.

**Model creation**

* I apply Logistic model , decision tree , Random forest , Gausian naïve bayse , KNeighbour , SVM , XGBOOST algorithms to creat the model.

# **logistic model**

1. Insights interpret from Confusion matrix -:

True Positive (TP): 89219

True Negative (TN): 504

False Positive (FP): 134

False Negative (FN): 1903

Insights:

1. Accuracy:

The overall accuracy of the model is 98%, indicating that it correctly predicts the class label for approximately 98% of the samples.

2. Precision:

Precision for class 0 (Negative class) is high (98%), indicating that when the model predicts the class as 0, it is correct 98% of the time.

Precision for class 1 (Positive class) is lower (79%), indicating that when the model predicts the class as 1, it is correct only 79% of the time.

3. Recall (Sensitivity):

Recall for class 0 is very high (100%), indicating that the model correctly identifies nearly all instances of class 0.

Recall for class 1 is low (21%), indicating that the model fails to identify a significant portion of instances of class 1.

4. F1-Score:

The F1-score for class 0 is high (99%), reflecting a good balance between precision and recall for class 0.

The F1-score for class 1 is relatively low (33%), indicating a lower balance between precision and recall for class 1.

5. Support:

The support represents the number of actual occurrences of each class in the dataset. Class 0 has much higher support (89353) compared to class 1 (2407), indicating class imbalance.

**Random Forest Model**

Overall Accuracy: The random forest model achieves an accuracy of approximately 98.59% on the testing data, indicating that it correctly classifies the majority of instances.

Confusion Matrix: The confusion matrix reveals that the model predicts class 0 (negative class) with high accuracy, as evidenced by the large number of true negatives (89120) and the small number of false positives (177). However, it is less accurate in predicting class 1 (positive class), as indicated by the moderate number of false negatives (1120) and true positives (1343).

Precision and Recall: The precision for class 0 is very high (99%), indicating a low false positive rate. However, the precision for class 1 is slightly lower (88%), suggesting that there is a relatively higher false positive rate for this class. The recall for class 0 is excellent (100%), indicating that the model correctly identifies nearly all instances of class 0. In contrast, the recall for class 1 is moderate (55%), indicating that the model misses a significant portion of class 1 instances.

F1-score: The F1-score, which considers both precision and recall, is high for class 0 (99%), indicating a good balance between precision and recall. However, for class 1, the F1-score is lower (67%), indicating a less balanced performance between precision and recall.

Class Imbalance: The support values in the classification report indicate a significant class imbalance, with a much larger number of instances for class 0 (89297) compared to class 1 (2463). This class imbalance could impact the model's performance, particularly in terms of its ability to correctly classify instances of the minority class (class 1).

**XGBOOST Model**

Accuracy: The model achieves an accuracy of approximately 98.72%, indicating that it correctly classifies the majority of instances.

Confusion Matrix:

True negatives (TN): 1473

False positives (FP): 239

False negatives (FN): 934

True positives (TP): 89114

This matrix demonstrates that the model is particularly effective at correctly identifying instances of the majority class (0), but it does have some false negatives for the minority class (1).

Precision and Recall:

Precision for class 0 is 99%, indicating a low false positive rate.

Precision for class 1 is 86%, indicating that when the model predicts class 1, it is correct 86% of the time.

Recall for class 0 is 100%, suggesting the model effectively captures all instances of class 0.

Recall for class 1 is 61%, indicating that the model identifies 61% of actual instances of class 1.

F1-score: The F1-score, which balances precision and recall, is high for class 0 (99%) and moderate for class 1 (72%).

Support: The support values indicate a significant class imbalance, with a much larger number of instances for class 0 compared to class 1.

|  | **Model** | **Score** |
| --- | --- | --- |
| **0** | Logistic Regression | 97.78 |
| **1** | Random Forest | 98.59 |
| **2** | Decision Tree | 97.52 |
| **3** | XGBoost Classifier | 98.72 |
| **4** | KNeighbors model | 97.92 |
| **5** | Gaussian naive\_bayse | 67.12 |
| **6** | svm | 97.95 |

xgboost and random forest is the algorithms which gives good accuracies.

**Data set source:**

<https://www.kaggle.com/competitions/predict-potential-spammers-on-fiverr/data>

**Software: Google Coloab**

**Individual Details:**

**Name – Paras Jeevan**

**Email Id –** [**parasjeevan05@gmail.com**](mailto:parasjeevan05@gmail.com)

**Phone Number - 8533938407**