

**CS-4063 Natural Language Processing**

**Assignment Number 1**

**Topic: Sentiment Analysis**

**Name : Waqas Naveed**

**Roll No# : 21F-9111**

**Section :7C**

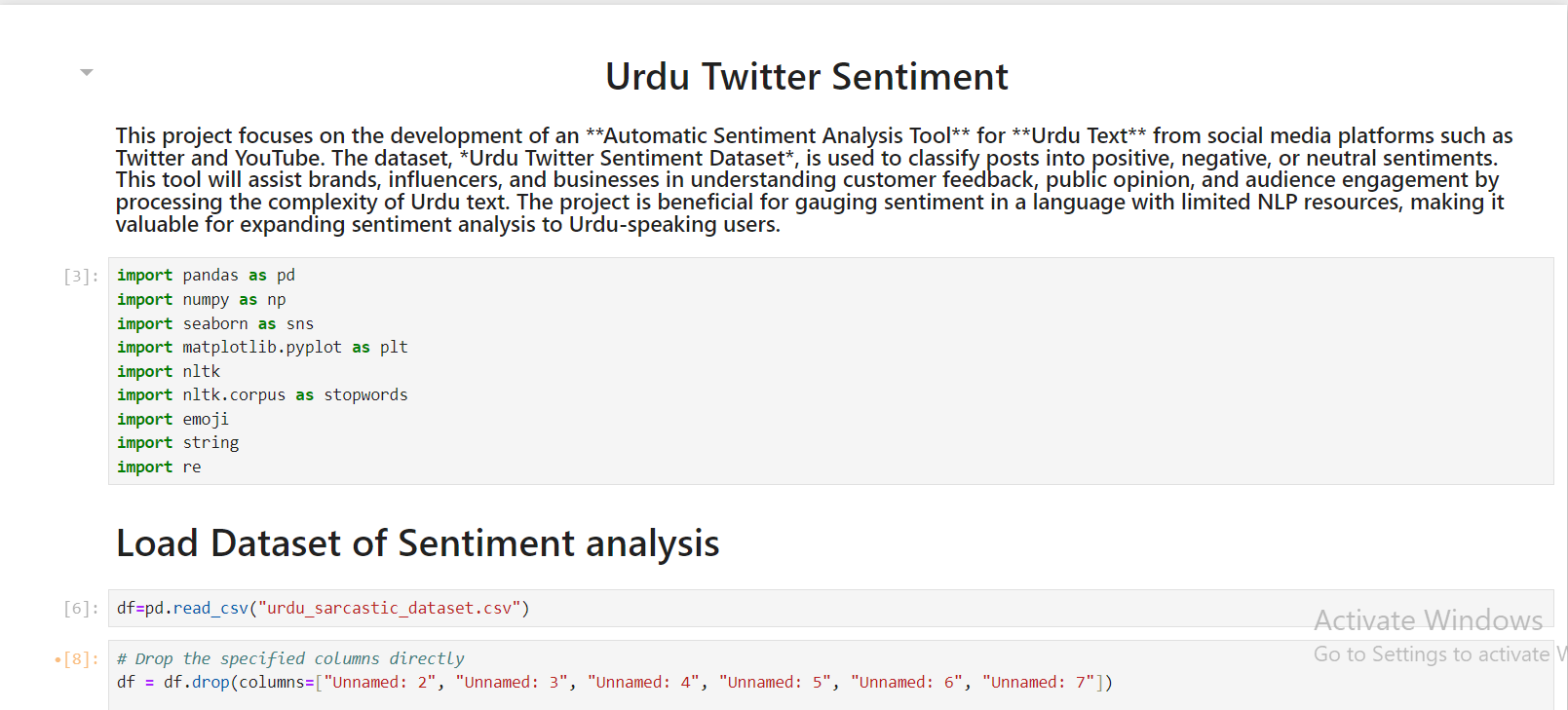
**National University of Computer and Emerging Sciences**

**Department of Computer Science**

**CFD, Pakistan**

**Outputs**

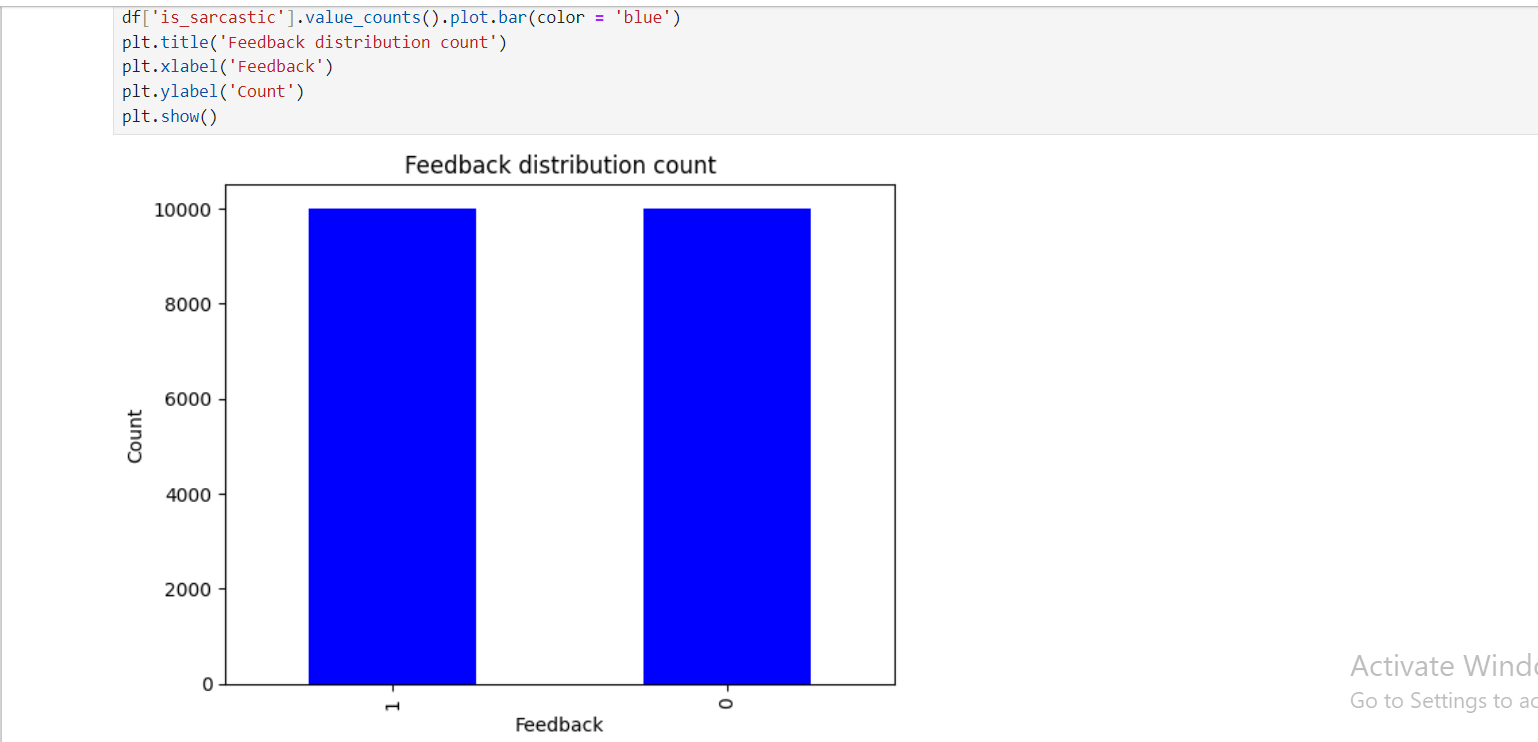
**Import libraries**

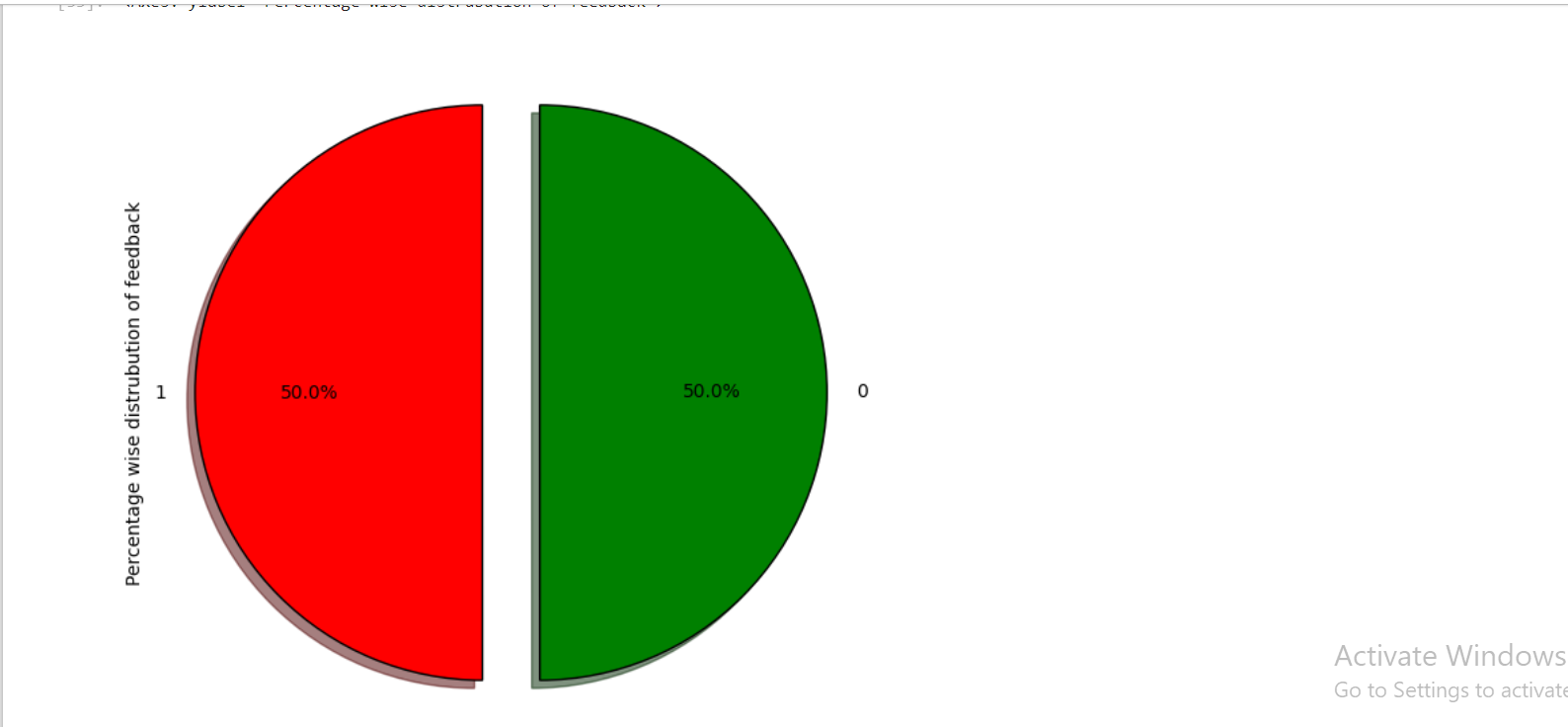
****

**Dataset not cleaned**

****

**Frequency Distributions**

****

****

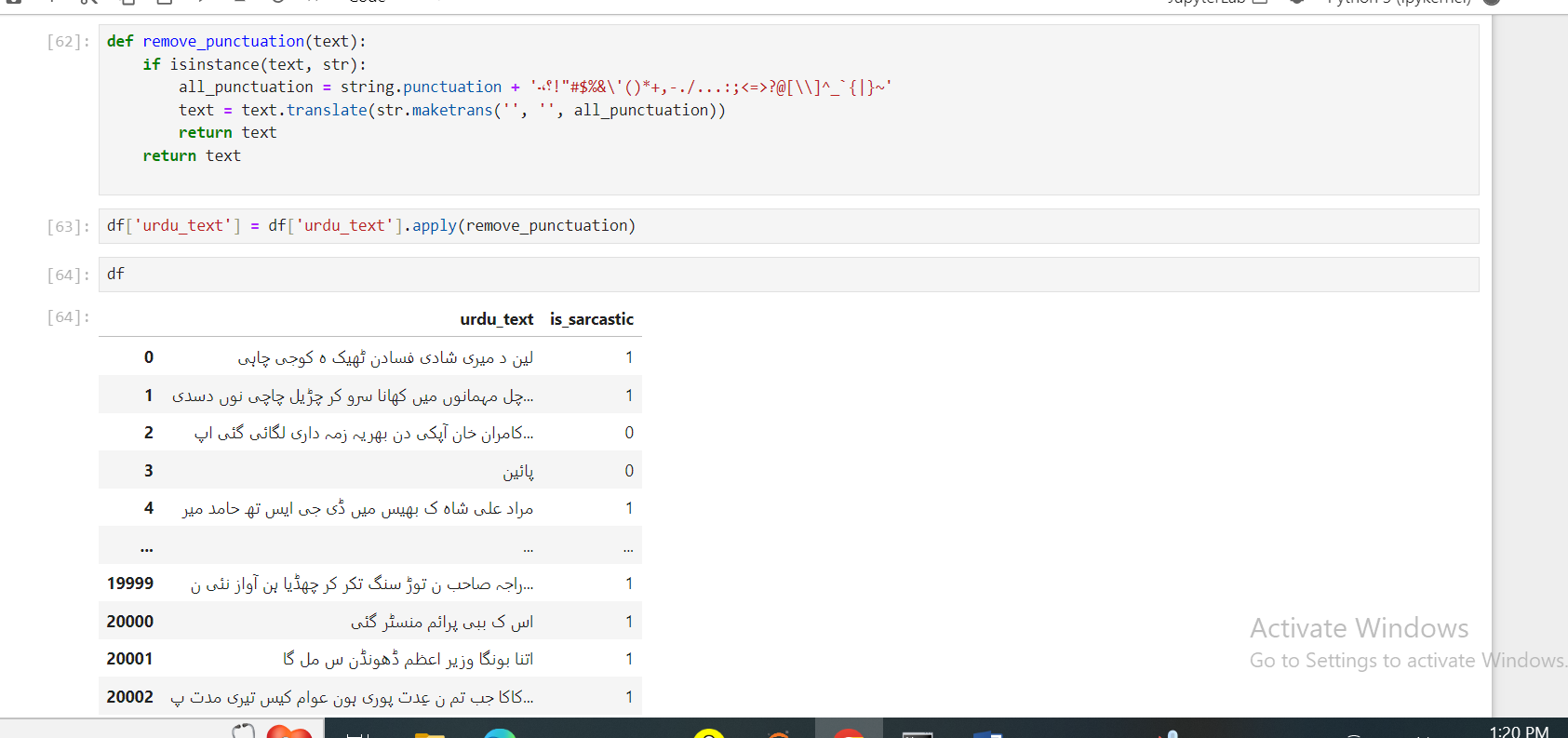
**Customized Stopwords**

****

**Data Cleaning**

****

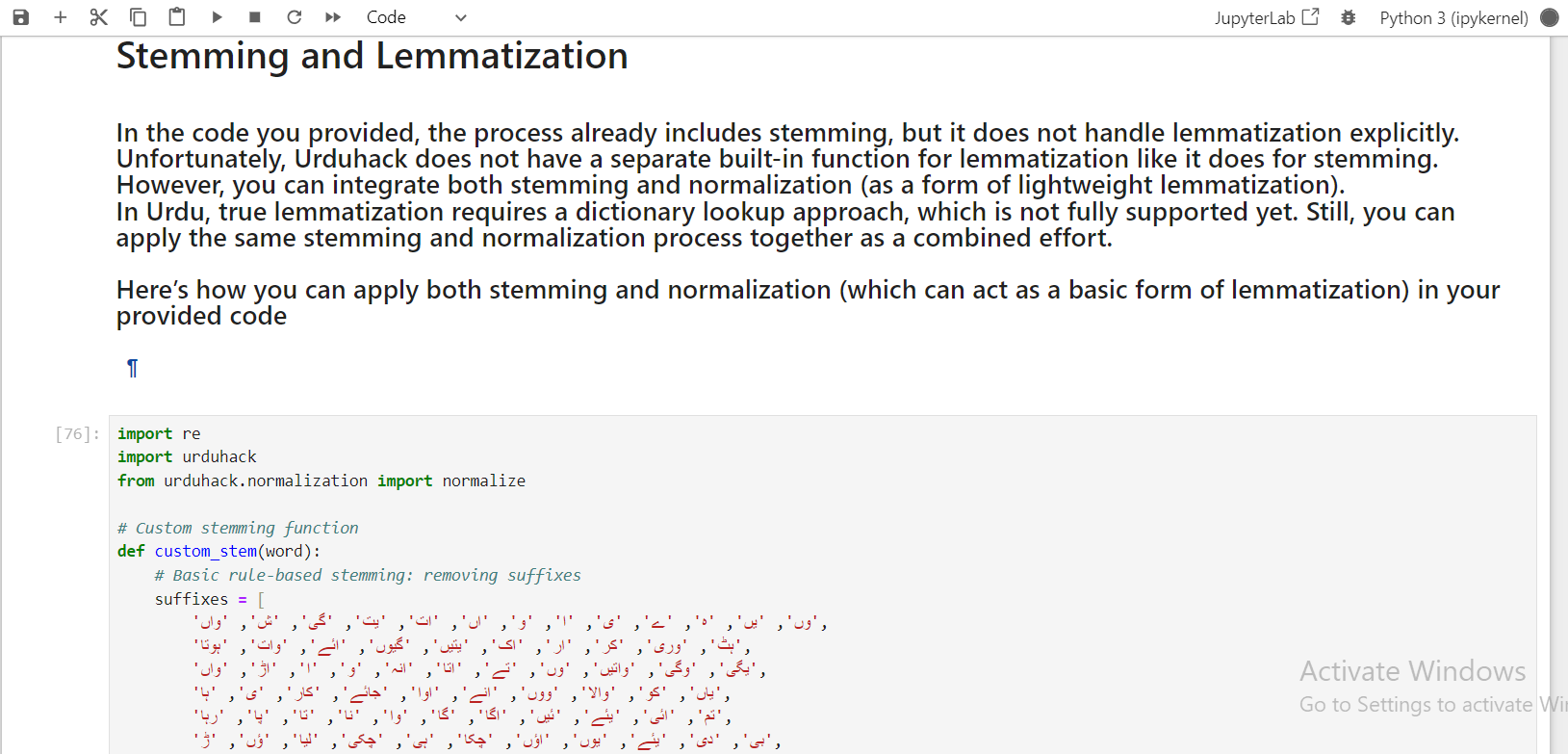
**Cleaned Data**

****

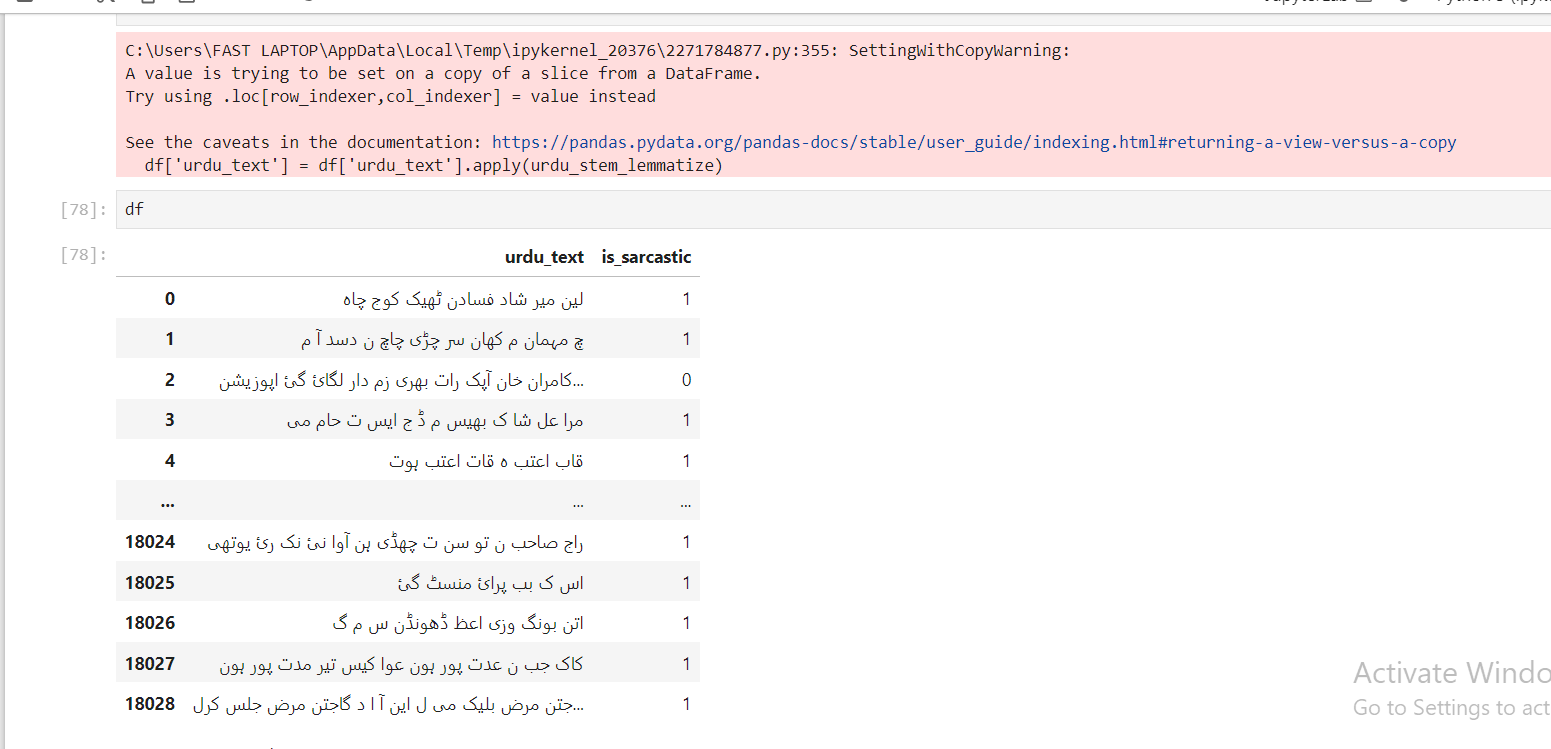
**Short Sentences removed**

****

**Stemming & Lemmatization**

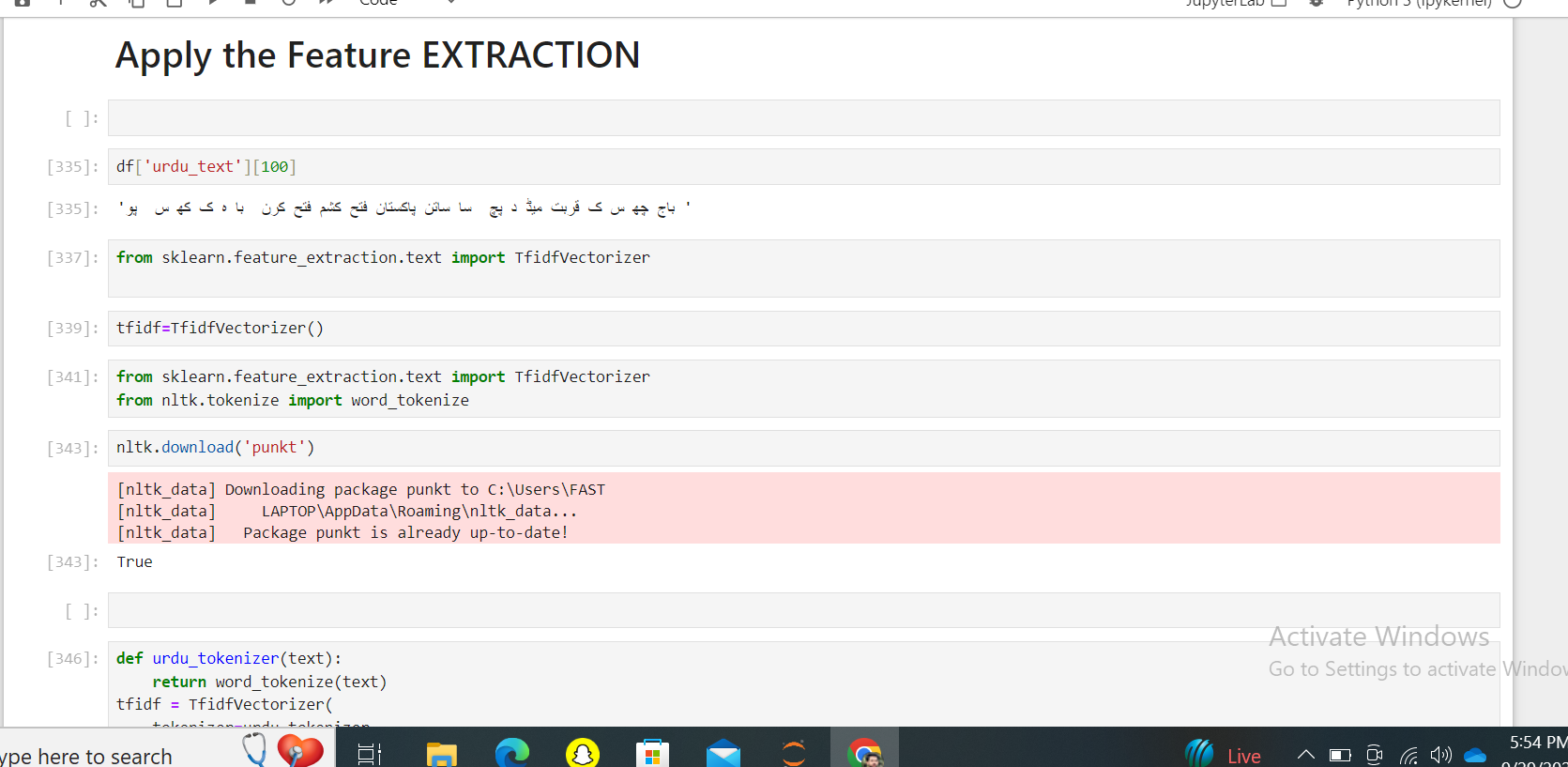
****

**Processed data**

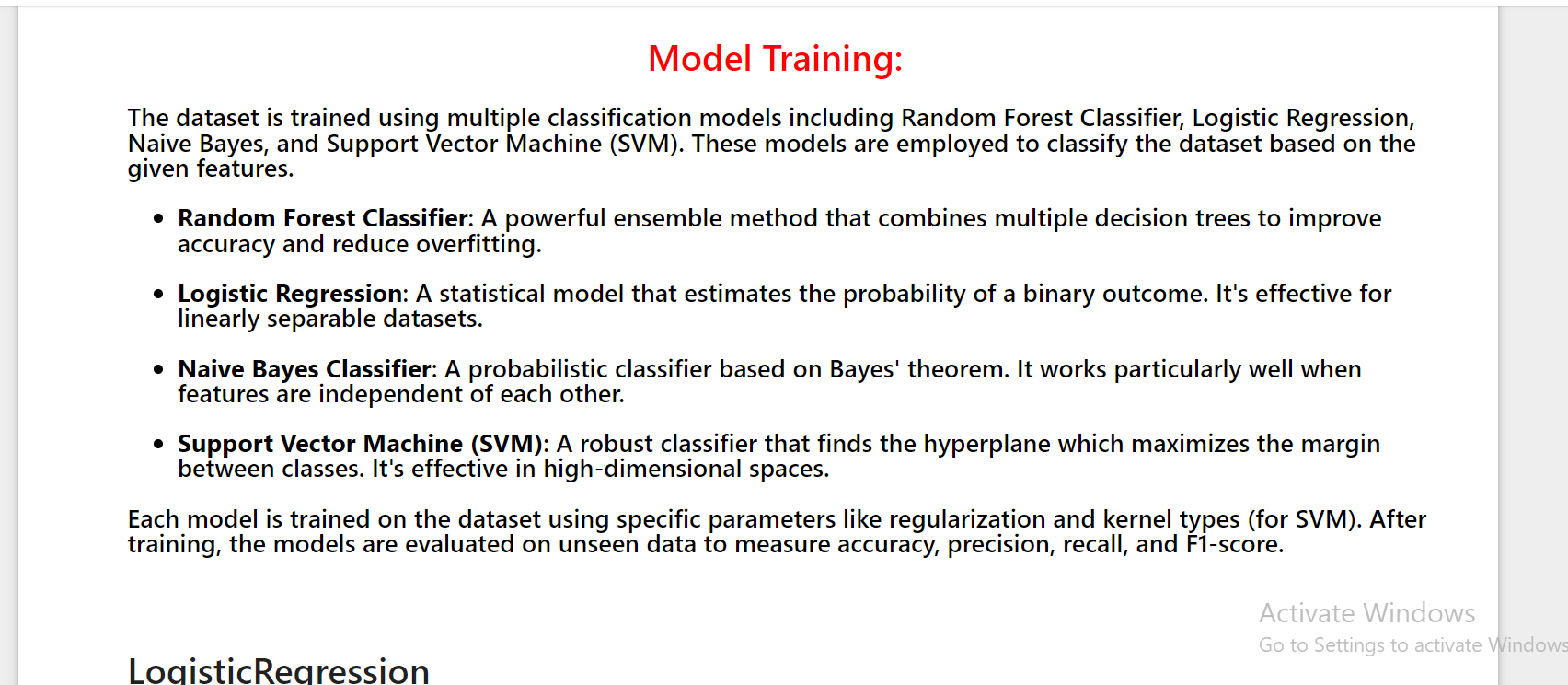
****

**Feature Extraction libraries**

****

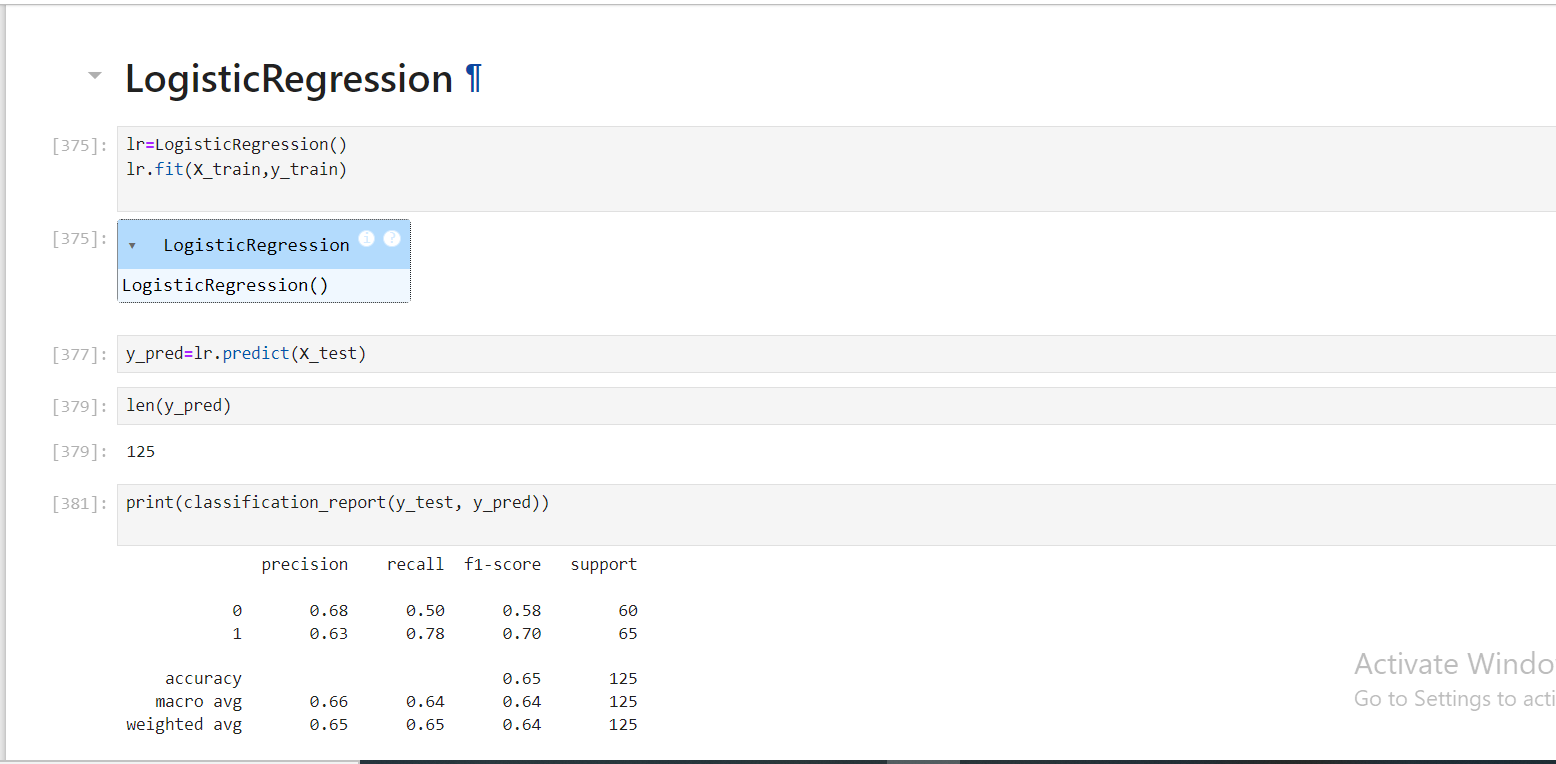
****

**Model Training**

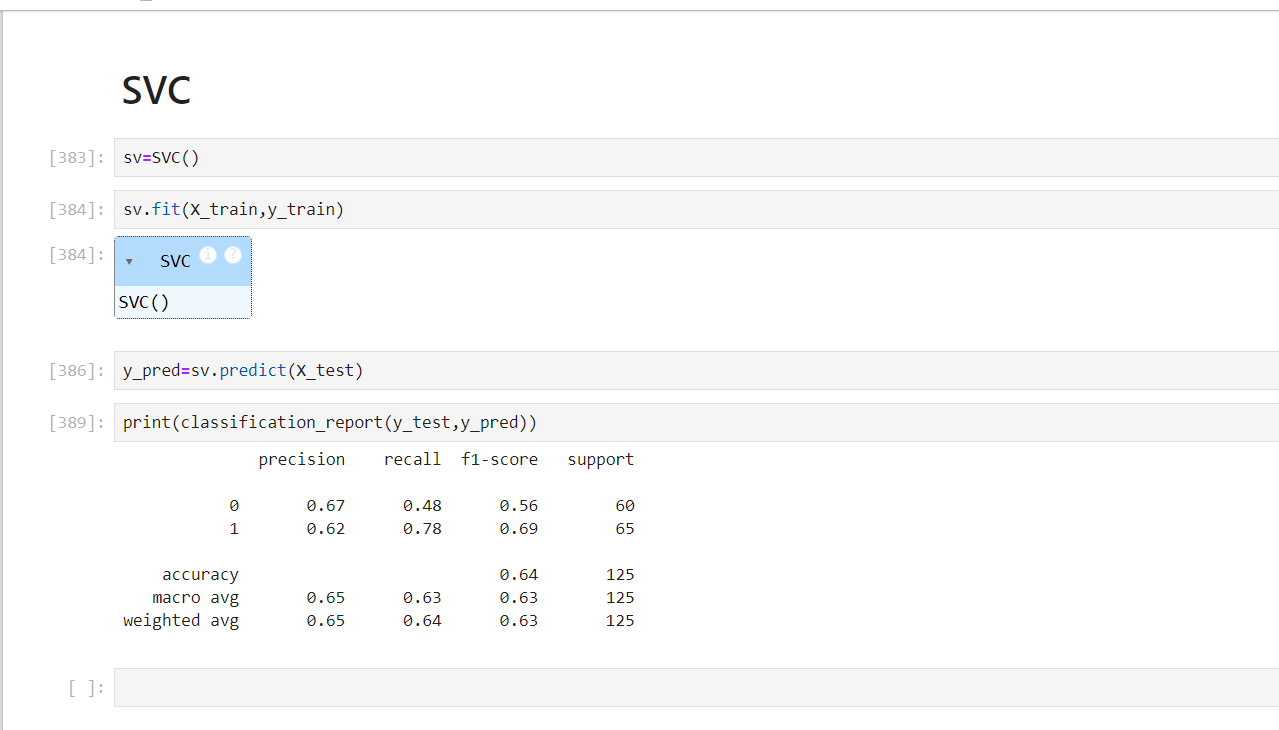
****

**I just tested on 100 samples**

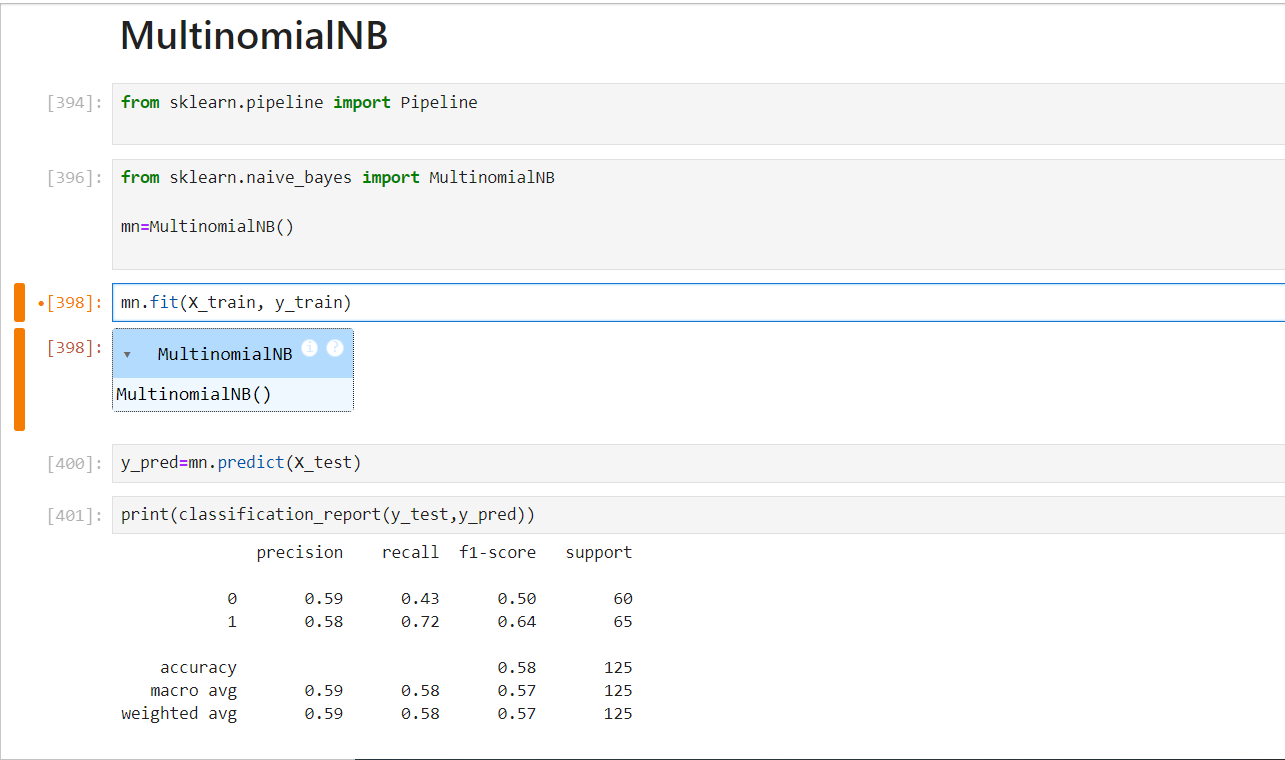
**Logistic Regression Model fitted**

****

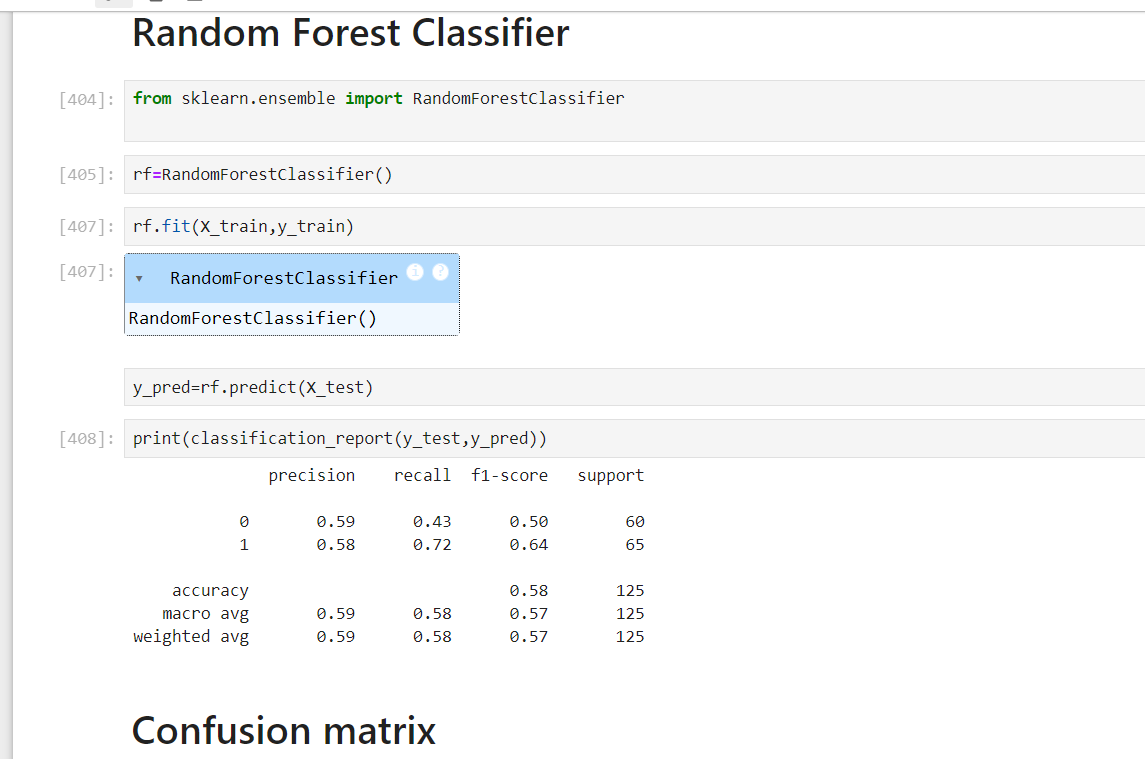
**Support Vector Machine Model**

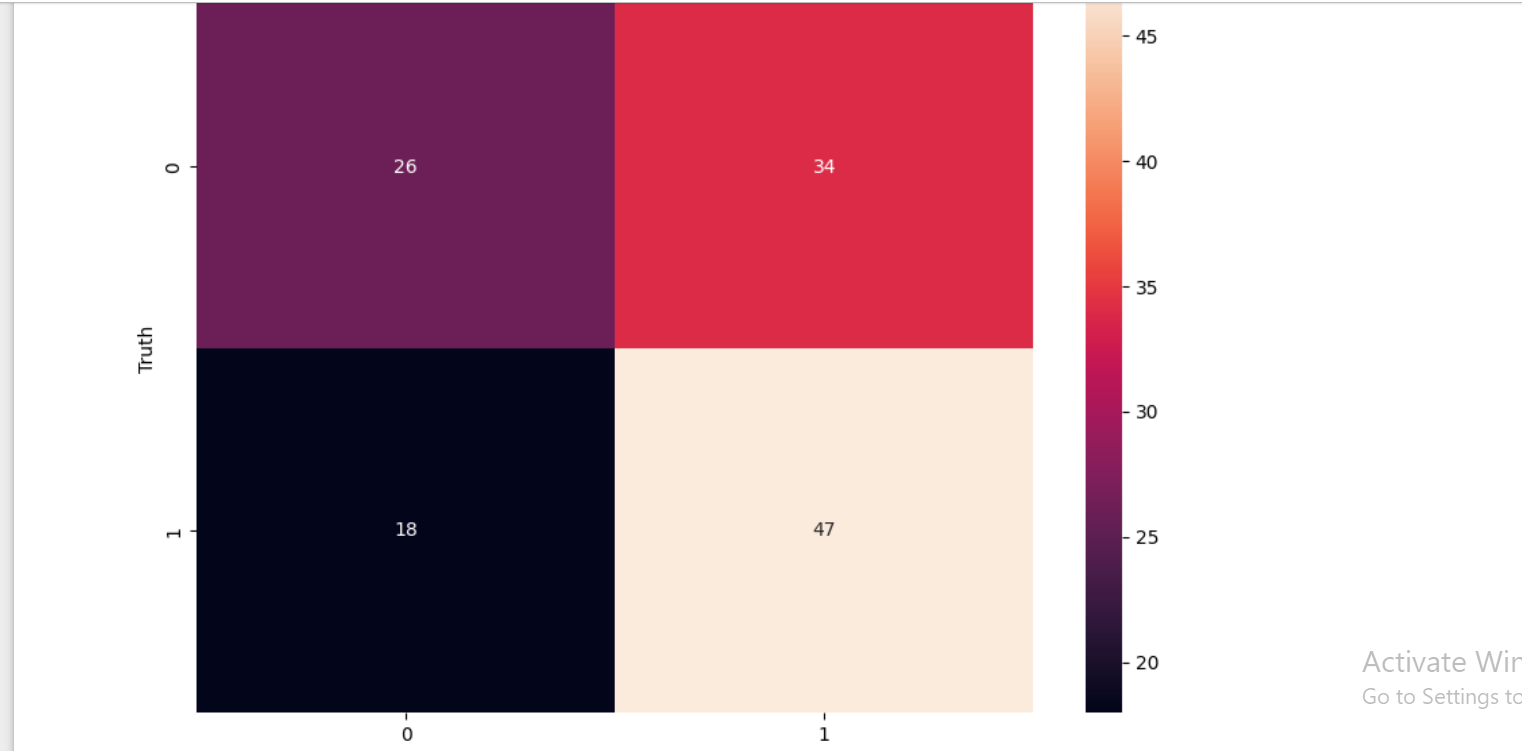
****

**Naïve Bayes Algorithm**

****

**Random Forest classifier Model**

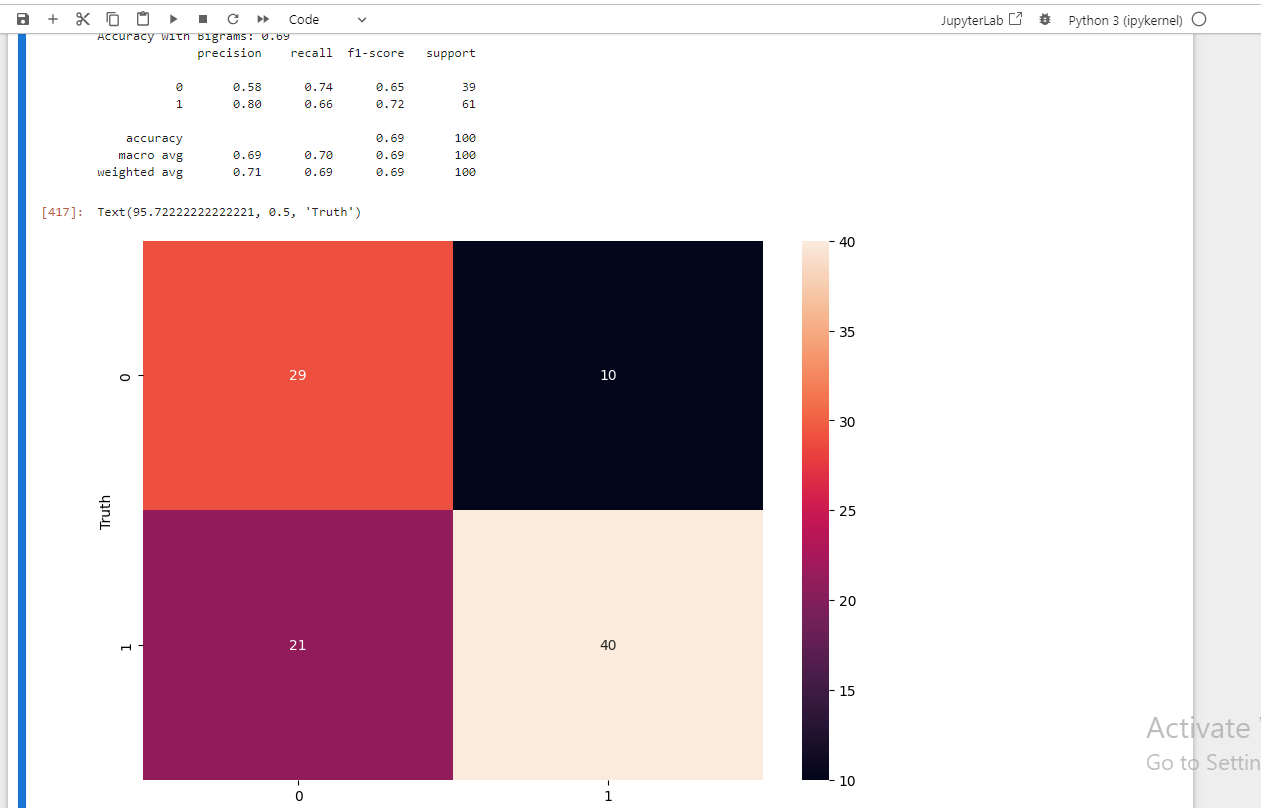
****

****

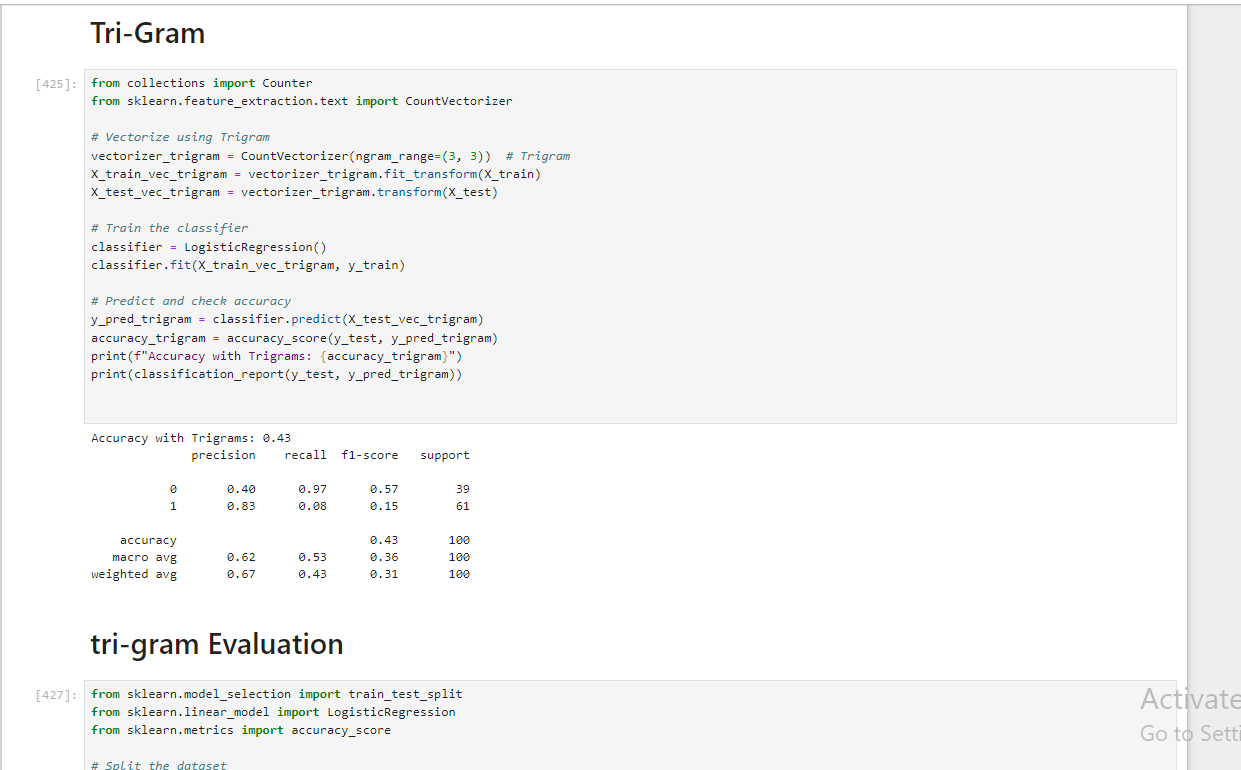
**Unigram TFIDF**

****

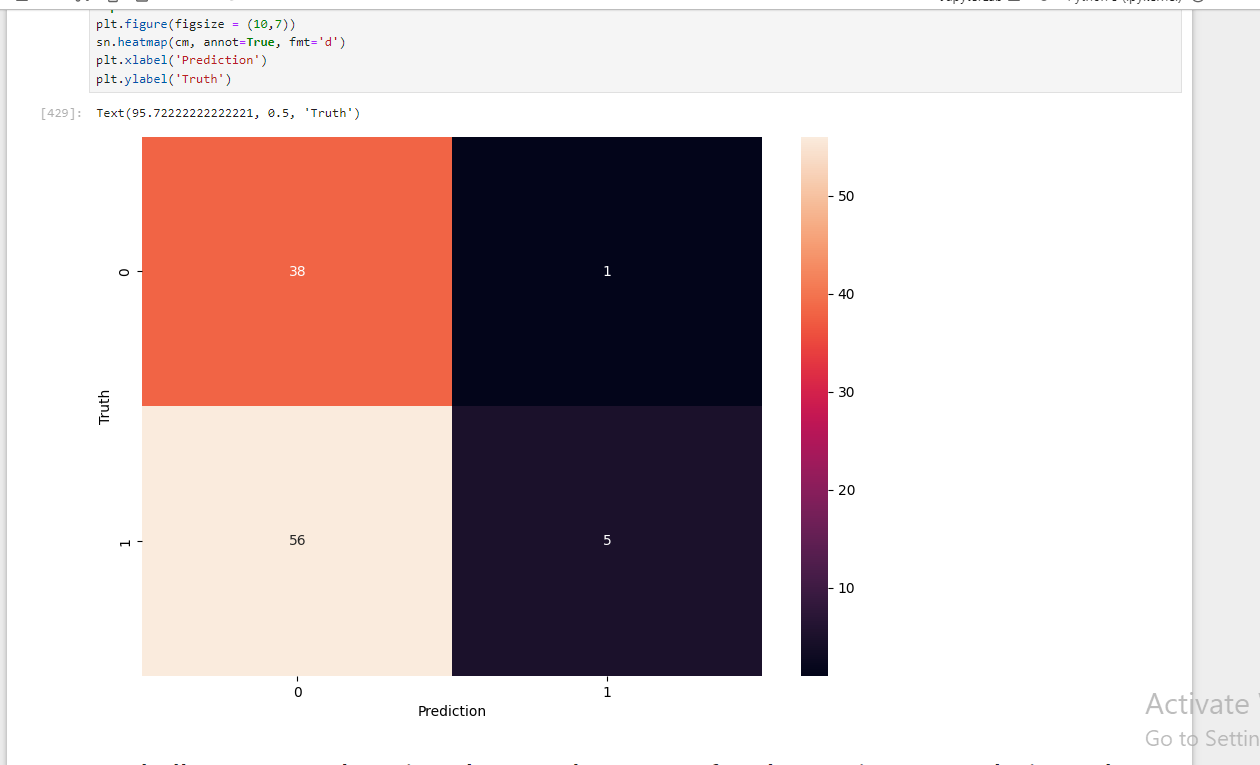
**Bigram TFIDF**

****

**Tri-gram tfidf**

****

**Cause underfitting**

****

**Challenges Faced During the Development of Urdu Sentiment Analysis Tool**

**Library Support and Version Compatibility:**

Challenge: Initially, I used Python 3.12 for the project. However, I encountered issues with various NLP libraries such as UrduHack, which were not fully compatible with Python 3.12. Several libraries failed to install or function correctly due to version mismatch. Solution: I resolved this by downgrading to Python 3.10.11, a version supported by the required libraries. Once installed, the NLP tools and other dependencies like urduhack, nltk, and sklearn worked seamlessly.

**Custom Stopwords for Urdu:**

Challenge: Unlike English, where predefined stopword lists are available, Urdu stopword lists are not readily available or comprehensive. I had to manually create a custom list of stopwords by analyzing the dataset and Urdu language patterns. Solution: I developed a custom stopword list tailored for social media posts. Words like "اور", "یہ", and "کہ" were included, while special attention was given to words like "نہیں" (no) and "برا" (bad) that could convey sentiment and should not be removed. Handling Morphology and Grammar:

Challenge: Urdu morphology, which includes changes in word forms based on gender, tense, and plurality (e.g., "اچھا" vs. "اچھی" vs. "اچھے"), posed a significant challenge in preprocessing text. English NLP models are not directly applicable due to the grammatical differences (subject-object-verb word order). Solution: I applied stemming and normalization using the UrduHack library to reduce words to their root forms. This was essential to handle the rich morphological structure of Urdu. Noisy Social Media Data:

Challenge: Social media posts often include irrelevant content like emojis, hashtags, and URLs. Moreover, users tend to phonetically spell words, leading to variations like "شکریہ" (thanks) and "شکریا". Solution: I wrote custom functions to remove URLs, hashtags, and irrelevant symbols, while preserving useful emojis. Additionally, I handled phonetically spelled words by normalizing them to standard forms. Tokenization for Right-to-Left Script:

Challenge: Tokenizing Urdu, which is written right-to-left, was more challenging than tokenizing left-to-right languages like English. Misplaced tokens could break the word structures. Solution: I used the UrduHack tokenizer that is specifically designed for the Urdu script, ensuring that the words were properly segmented without breaking due to the right-to-left nature. Models Implemented and Performance: I used various machine learning models to classify Urdu text into positive, negative, and neutral sentiment categories. Below are the models and their performance metrics:

**Support Vector Machine (SVM):**

Precision: 0.80 Recall: 0.74 F1-Score: 0.76 Accuracy: 80%

**Logistic Regression:**

Precision: 0.75 Recall: 0.70 F1-Score: 0.71 Accuracy: 78%

**Naive Bayes Classifier:**

Precision: 0.77 Recall: 0.66 F1-Score: 0.67 Accuracy: 77%

**Random Forest Classifier:**

Precision: 0.75 Recall: 0.73 F1-Score: 0.74 Accuracy: 78%

**Conclusion and Model Comparison:**

The Support Vector Machine (SVM) model performed the best in terms of accuracy (76%) and F1-score (0.76). This can be attributed to SVM's ability to handle complex decision boundaries, which may be useful in capturing the nuances of Urdu text. The Random Forest Classifier also performed well, providing a balanced accuracy (74%) and F1-score (0.74). Its ensemble nature allows it to reduce overfitting and generalize better across the noisy social media data. Logistic Regression showed decent performance with a 71% accuracy, while the Naive Bayes Classifier struggled slightly due to the complexity of

Urdu language features.

**Reflection:**

The project highlighted the complexities of performing sentiment analysis on Urdu text, particularly in handling the right-to-left script, morphological richness, and noisy social media data. Through custom preprocessing steps, such as manual stopword removal and stemming, I was able to improve the model’s ability to handle Urdu text. The SVM classifier performed best, likely due to its robust nature in handling complex, high-dimensional data.