**PROJECT 2**

In this project, we are using Naïve Bayes classification for classifying text files.

Here is the step by step process of the approach used to implement Naïve Bayes Algorithm:

Step 1: Extract and read Data from the system for the analysis.

Step 2: Removing punctuations, stop words. The function used for this is “remove\_punctuation”. This function sorts through all the punctuation symbols. Vectorize data and use TF-IDF to score the words based on importance.

Step 3: Reading files and splitting the data. The Training to Testing ratio is 50-50.

Step 4: The word count is calculated. And the number of times the word has appeared is measured, which is added to the list.

Step 5: Final probabilities are calculated.

Step 6: Based on these probabilities the accuracy is calculated.

OUTPUT: Accuracy = 0.8263826382638264 or 82%

**How to run the code:**

The only variable that you need to give a value before running the code is ‘path’. You should give the whole path to the directory where the data is located.

**Algorithm:**

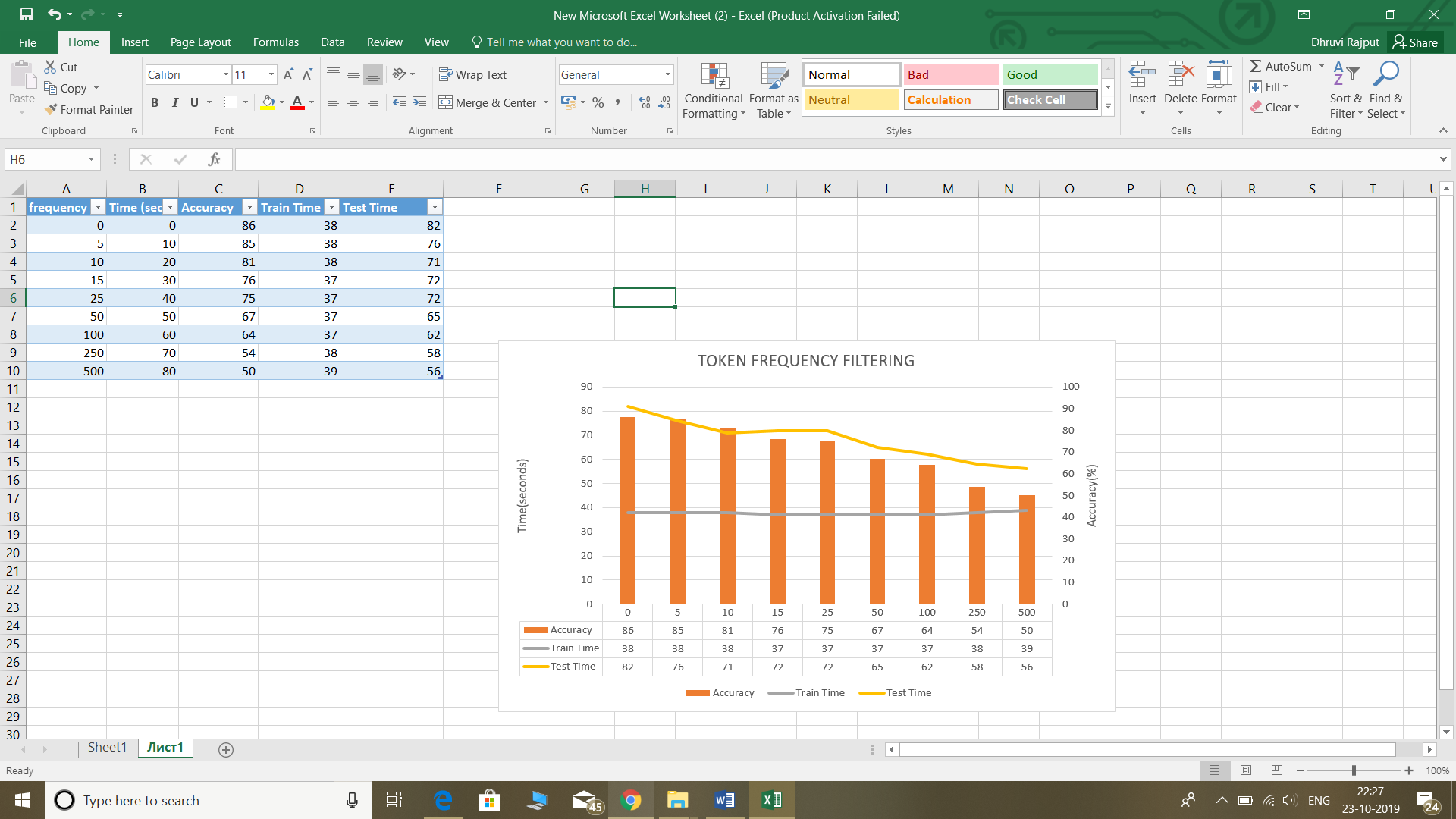
The project has three phases: Preprocessing, training, and testing. In the preprocessing phase, we clean the data. There are some characters that they need to either be removed or be replaced with space character.

We consider two list remove\_list and replace\_list, and remove all the characters inside the remove\_list from the data and replaced all character inside the replaced\_list with space.

* Replace\_list = ["'",'!','/','\','=',',',':', ‘\n’]
* Remove\_list = ['<','>','?','.','"',')','(','|','-','#','\*','+']

We convert all the uppercase letters to lowercase ones. clean\_text function does this part and cleans the data. In the training phase, we select first 500 files of each class, and split the file with character space. We count the number of occurrences of each word in class’s files and put them in a dictionary. After the training phase, we get totally around 170,000 different words in all classes (for 10,000 files).

In the testing phase: we pick files randomly from the remaining files from the training phase, function get\_file gives us the files. The same as before, we clean the file with clean\_text function, after that we comput the probability of each word in this file for all classes.

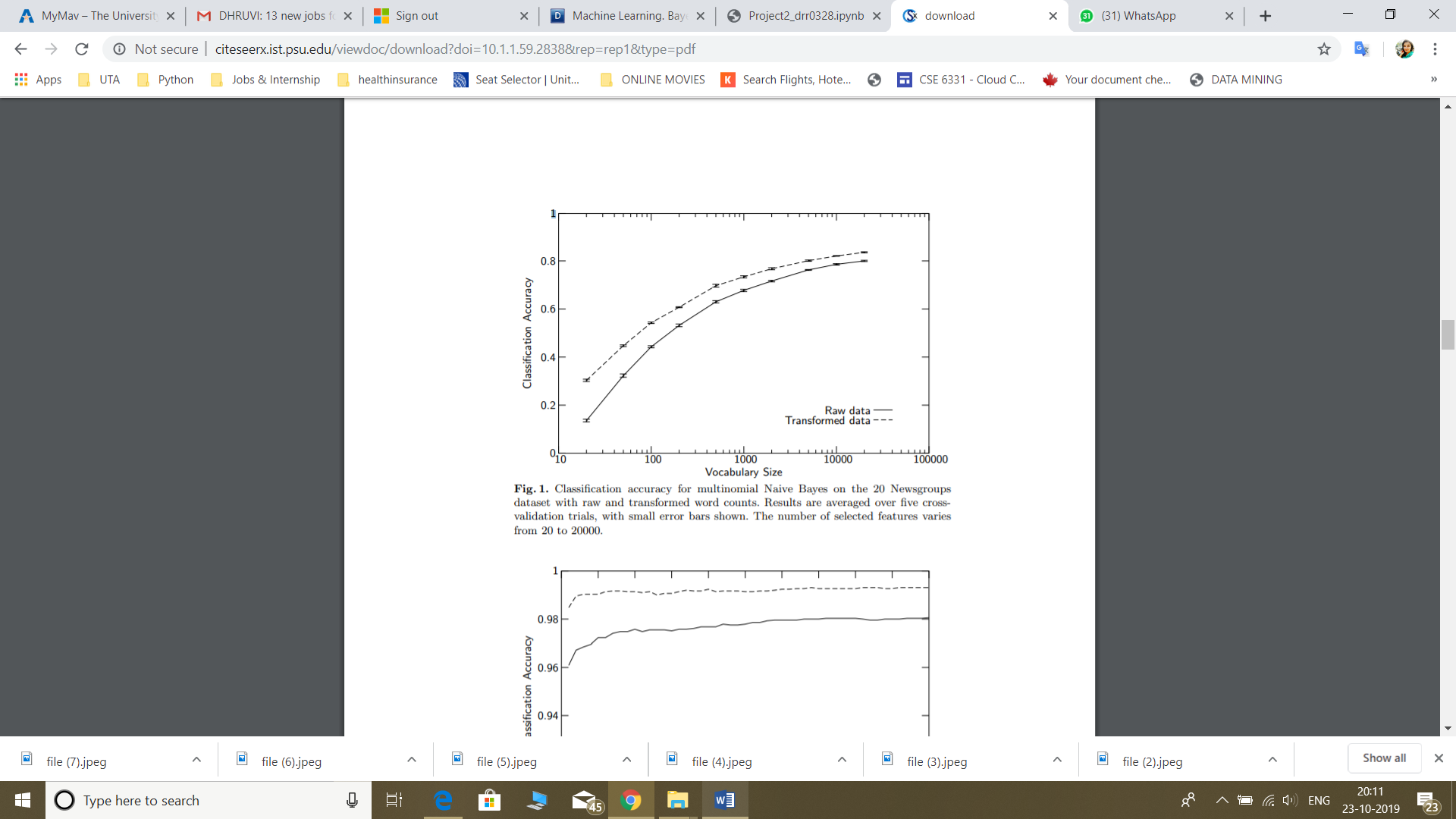


# Text-Classification-using-NaiveBayes-

**TF-IDF**

To avoid the issues of larger files getting more weights after “Bag of Words”, we introduce the “TF-IDF” (Term Frequency — Inverse Document Frequency). This will not change the dimension of our data.





Classification accuracy for multinomial Naïve Bayes on the 20 Newsgroups dataset with raw and transformed word counts. Results are averaged over five cross-validation trails, with small error bars shown. The number of selected features varies from 20 to 20000.