****

**Machine Learning**

**MAJOR PROJECT**

**Task: To compare the accuracy of different models on tweet dataset for identifying the gender from the tweet.**

**Submitted By:**

**Shreyas Bhakta**

**K Sai Sumanth Reddy**

**Gulabshanaaz Shaik Mohammad**

**Shaik Badulla**

**Sathvika Katikaneni**

**Aayush Surana**

**Gunjan Behera**

**Shaik Haneesa**

**Chavali Krishna Deepika**

**Tirumani Someswari Sai Rushitha**

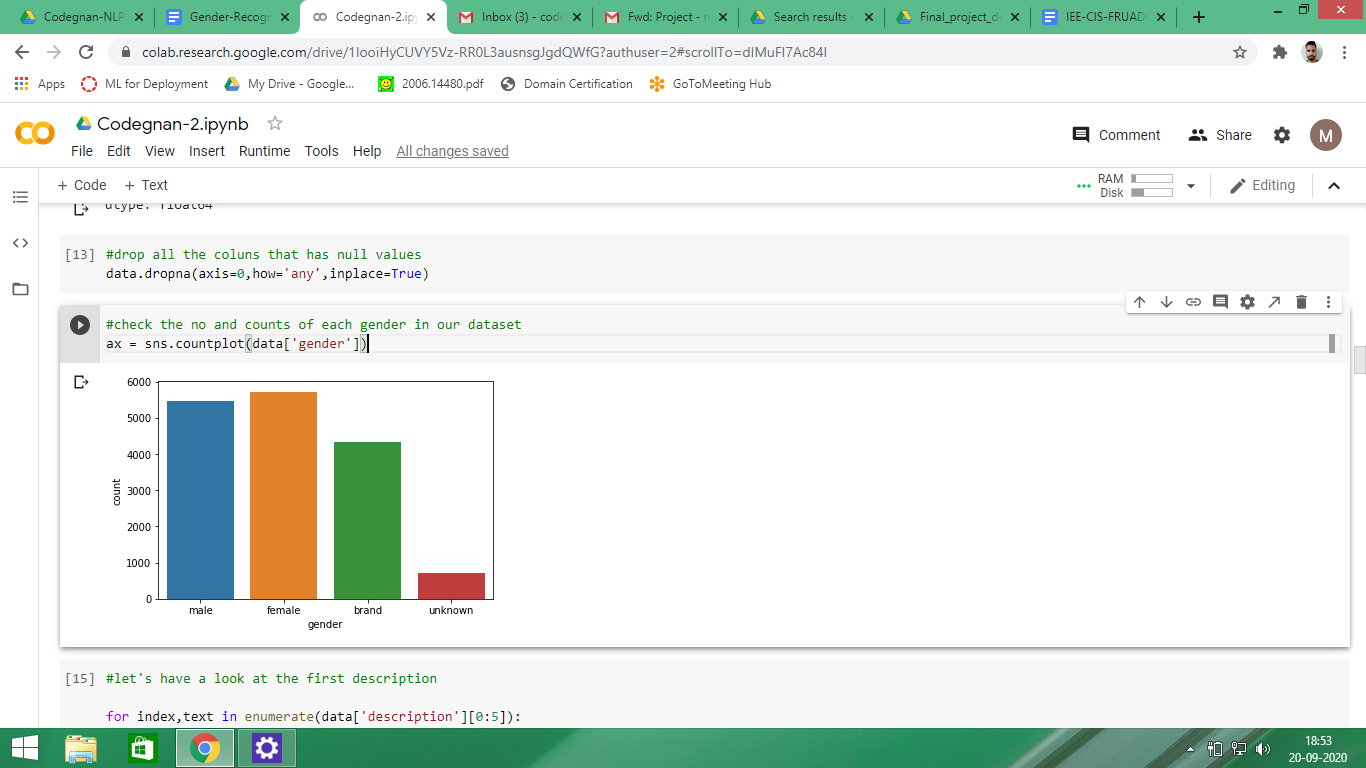
**Compare Accuracy of different Machine Learning Models tweets**

**DATASET DESCRIPTION -**  The data has 20050 number of rows and 26 features.

We have treated it as a NLP Problem so we removed all the columns apart from Description, texts and gender.

Gender is our target variable and we have 4 labels in our gender column -

1. Male
2. Female
3. Brand
4. Unknow



**Challenges with textual data and Methods to overcome the challenges**

Textual data possesses lots of challenges and ambiguities . For example consider a tweet-

* It will have smiles
* It will have punctuations
* Social-media abbreviations
* Urls
* Unwanted whitespaces
* It will have hashtags etc.

To overcome these challenges we will have to do some pre-processing on our textual data.

To do the preprocessing we have made a function tweet\_to\_words .

We have removed the punctuations and numbers from the tweets

Remove hyperlinks

Remove old style retweet ‘RT’

Remove the # sign from hashtags and keep the word

Lower the case of each word

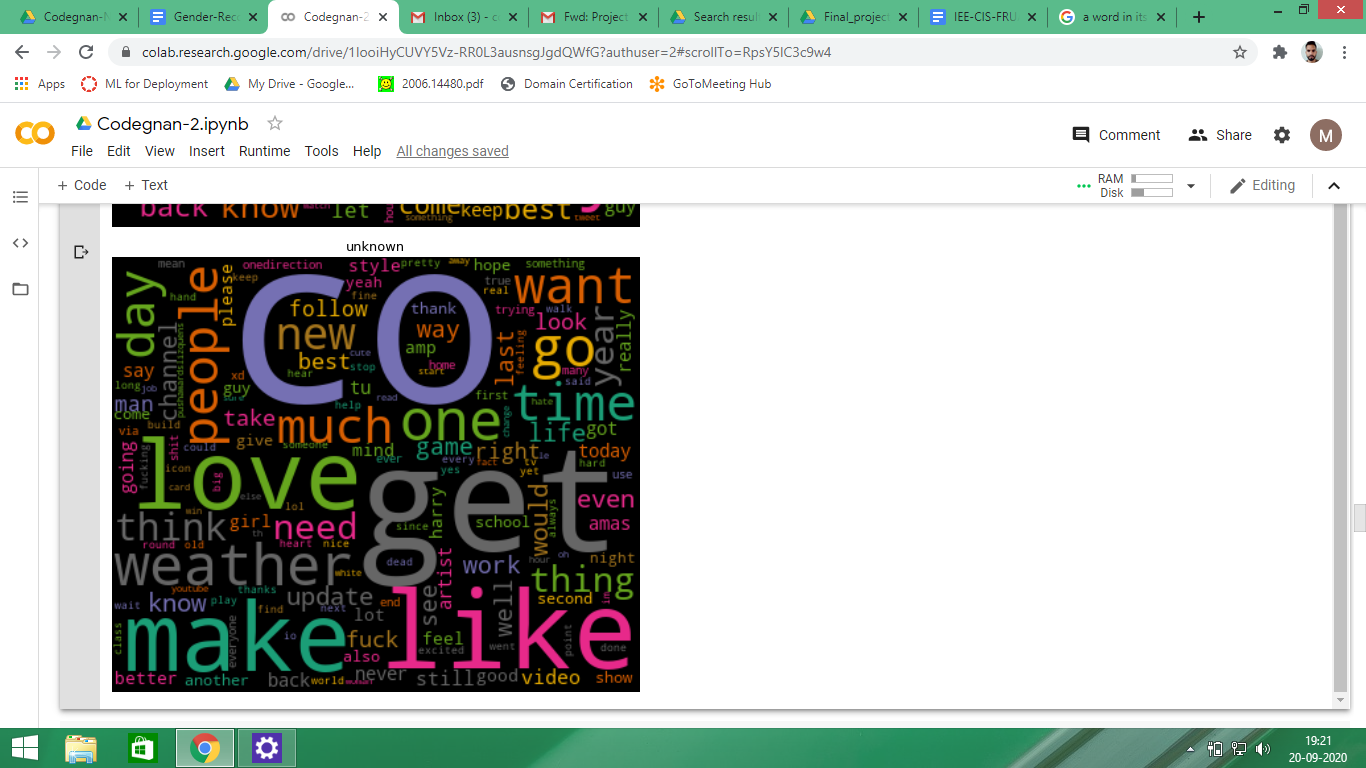
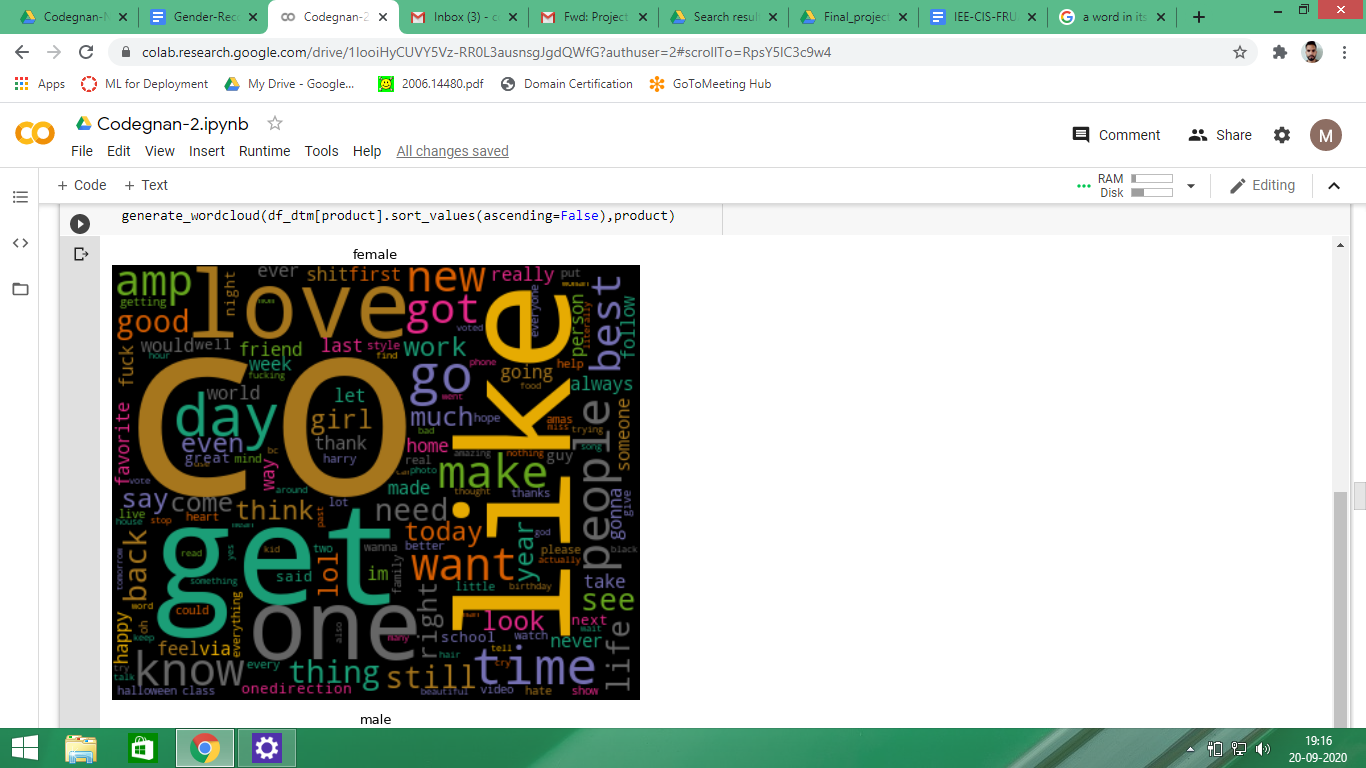
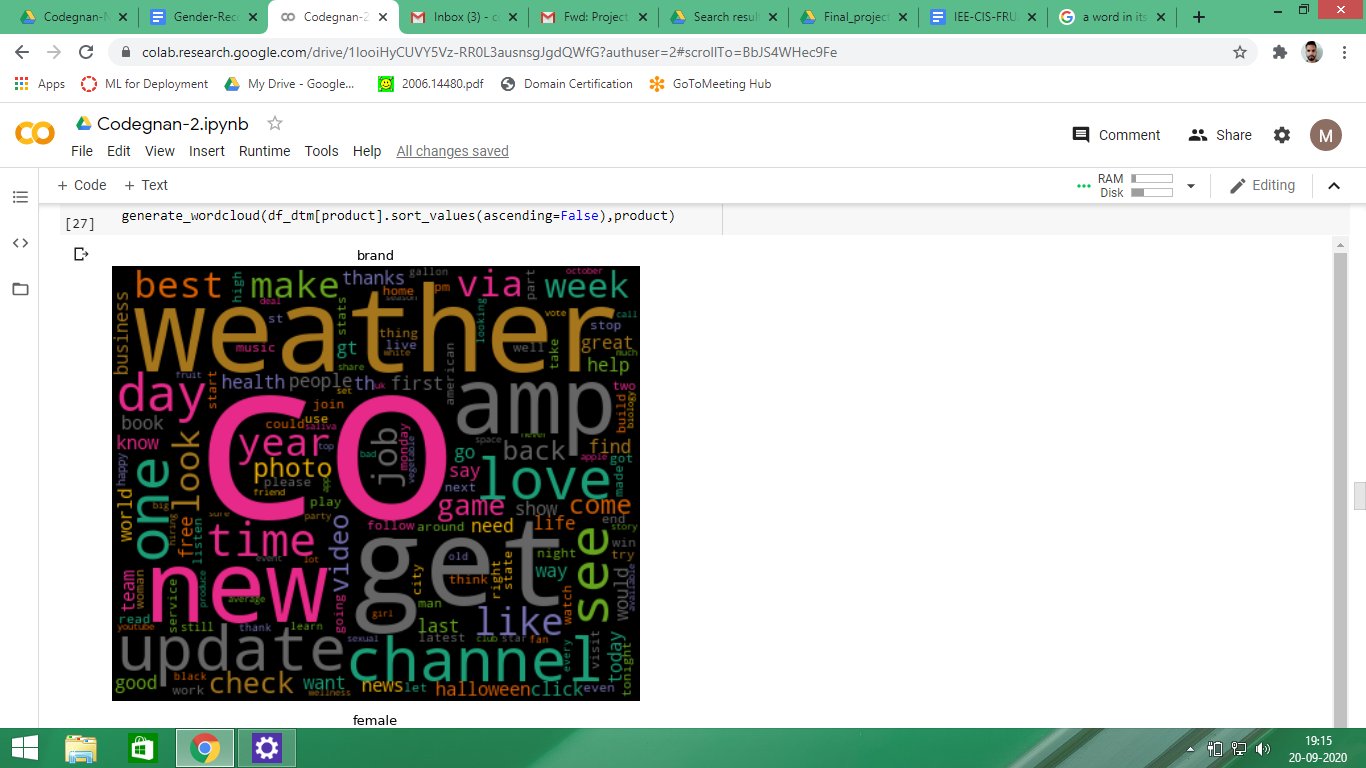
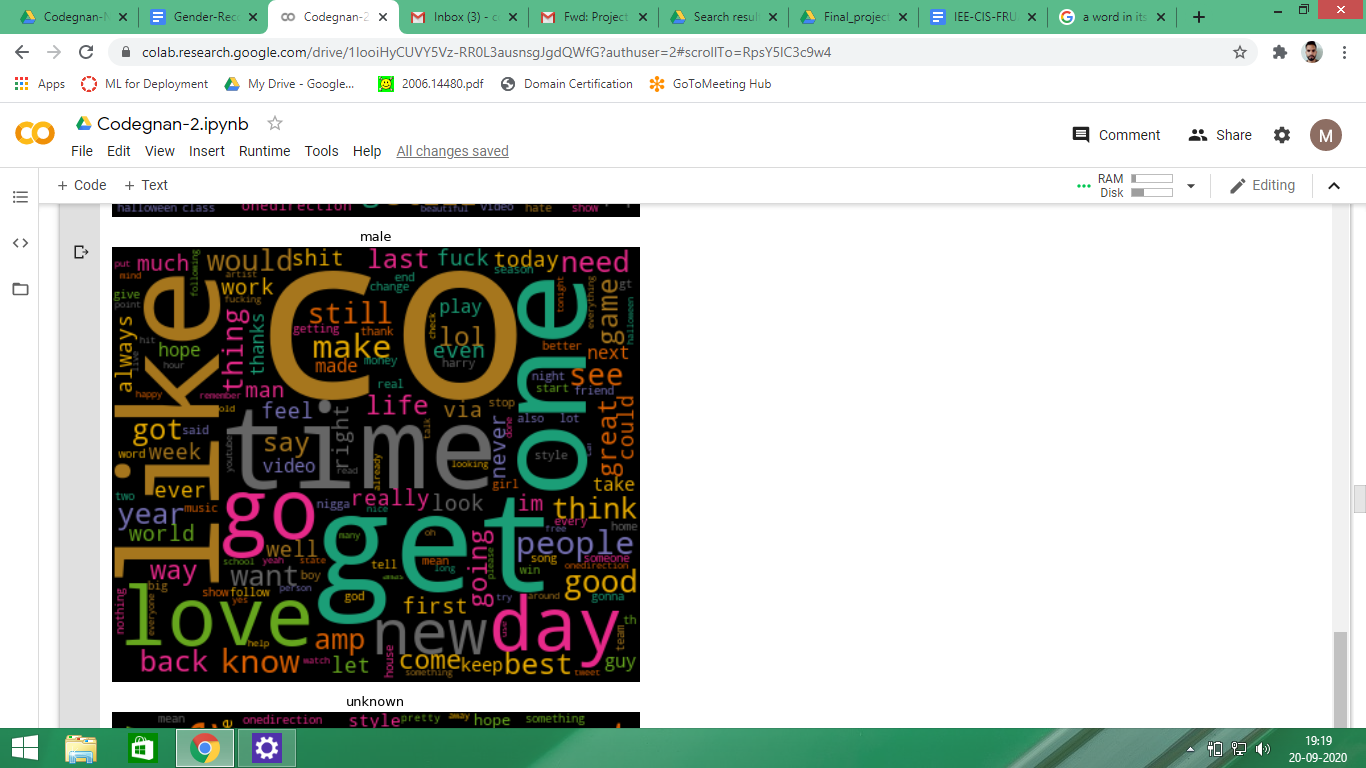
Tokenize each word

Then we have done lemmatization

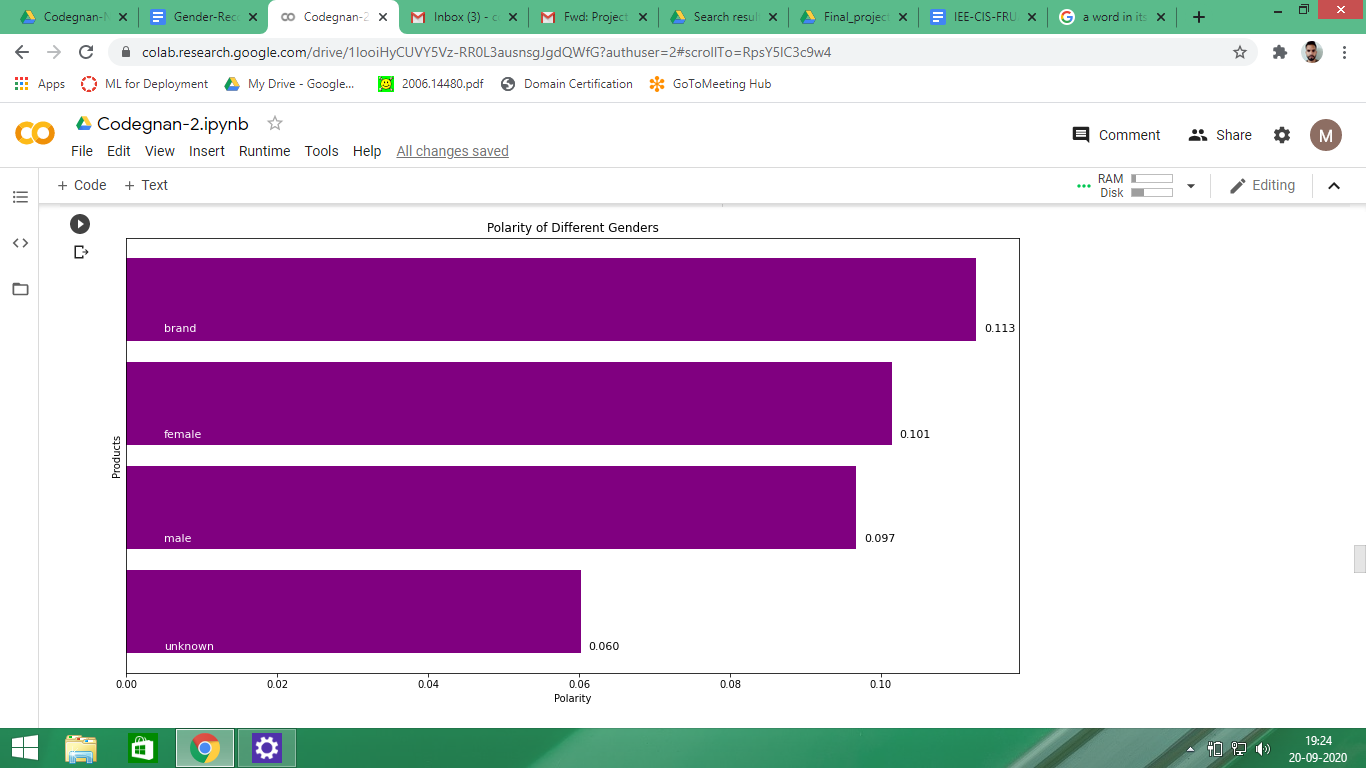
After doing the preprocessing task next we did Exploratory Data Analysis on our preprocessed data.

**Exploratory Data Analysis**

We wanted to know the most common words in each gender and for that we prepared a word cloud for each gender



Next we have checked the overall polarity of each label in the gender column.



**Representing words/textual data as Vectors**

To represent the words in vector form we have made use of 2 methods -

1. Bag of Words / CountVectorizer as it creates a sparse matrix we have just taken a maximum of 8000 features and have applied the Naive Bayes, Logistic regression and Random Forest Algorithm.

**Naïve Bayes : 0.706936**

**Logistic Regression: 0.728879**

**Random forest: 0.982742**

**KNN: 0.476071**

1. TF-IDF / TfidfVectorizer - Tf-idf conserves the context of the words somewhat. we have just taken maximum of 8000 features and have applied the Naive Bayes, Logistic regression and Random Forest Algorithm
2. **Naïve Bayes : 0.706936**
3. **Logistic Regression: 0.728879**
4. **Random forest: 0.982742**
5. **KNN: 0.476071**

**We can conclude that the Random Forest is out performing the two algorithms for both the methods.**