TWEET CLASSIFICATION
Team: Zenith

How did we approach the problem?

- After brainstorming and extensive EDA, we approached this classification problem by combining multiple ML algorithms to predict the target variable(Antisemitism/Not antisemitism).
- First, we build Model 1 using the text data to predict the target. This was done using the Naive Bayes classifier.
- Once we get the probability from Model 1, we use this as a variable in the second Model.
- We explored multiple models for the second layer. Random forest Classifier gave us the accuracy of 100% on training data, which is why we finalised this model.
- So all in all we used a two-pronged approach to predict the target variable

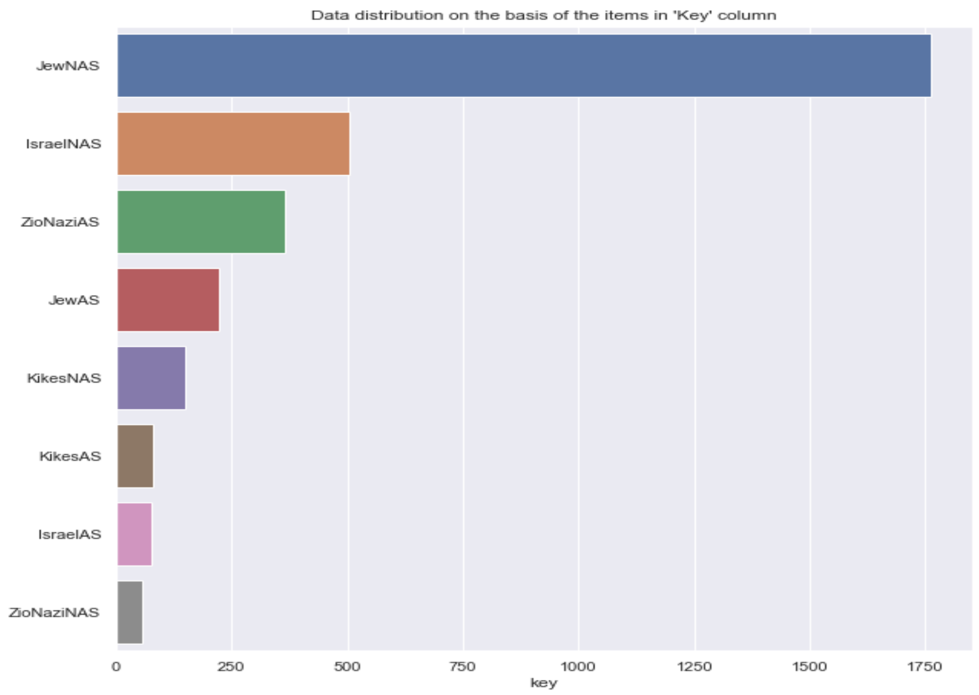
What steps did we take to predict?

STEP 1: DATA PRE-PROCESSING BEFORE NAIVE-BAYES

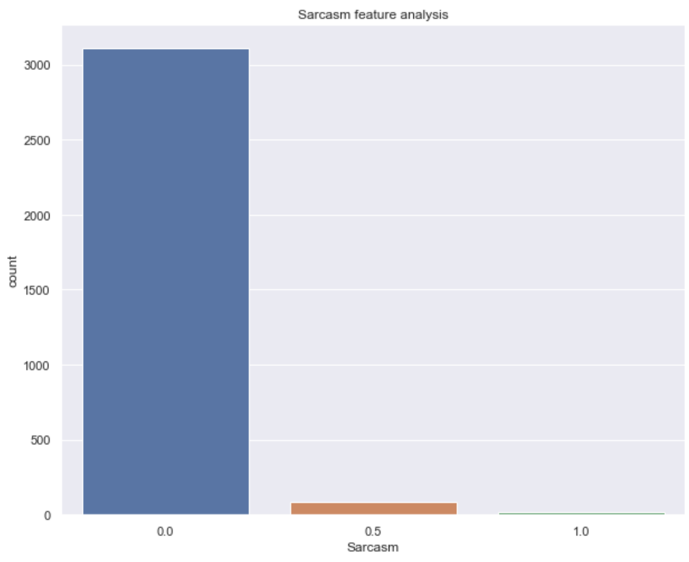
- In this step, stop words are removed from the tweets. For this, a list of 1628 stop words is downloaded from a GitHub repository on the basis of which these words are removed from tweets.
- Bags of words are also created on the basis of target values. If the target value is 1: A bag of truthful words is created and for the target value 0: a bag of deceptive words is created.
- The probability of truthful and deceptive sentences is also calculated in this step.
- Using the likelihood and prior probabilities, we computed posterior probabilities and classified with a threshold of 0.5

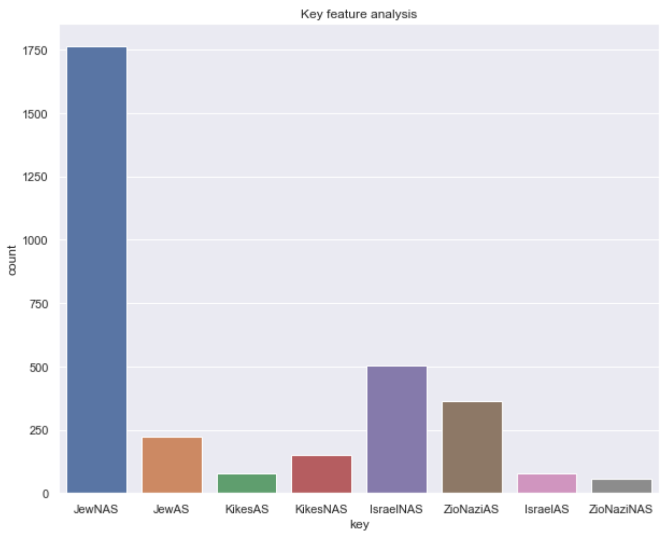
STEP 2: EXPLORATORY DATA ANALYSIS

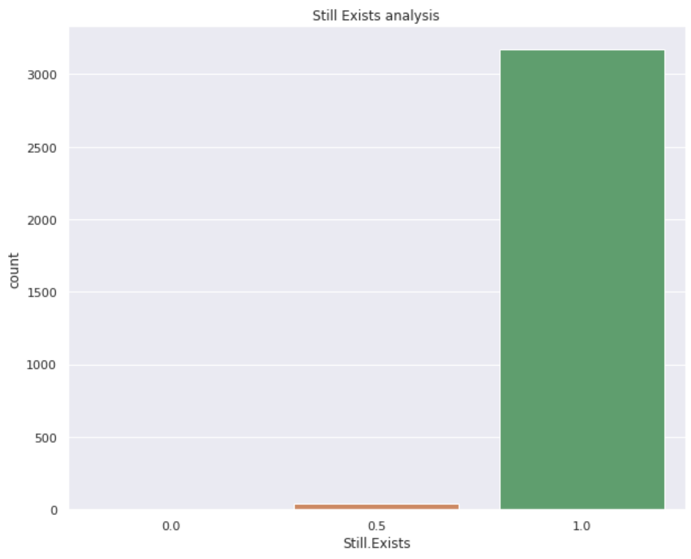
- Performed **univariate data analysis** on each column.

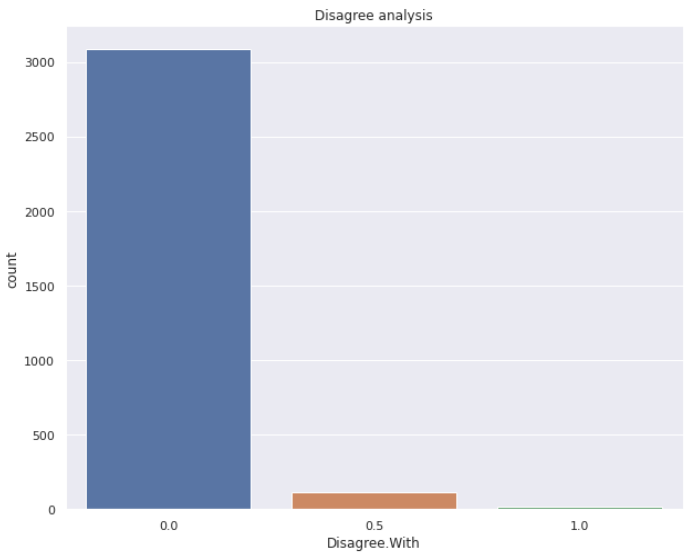


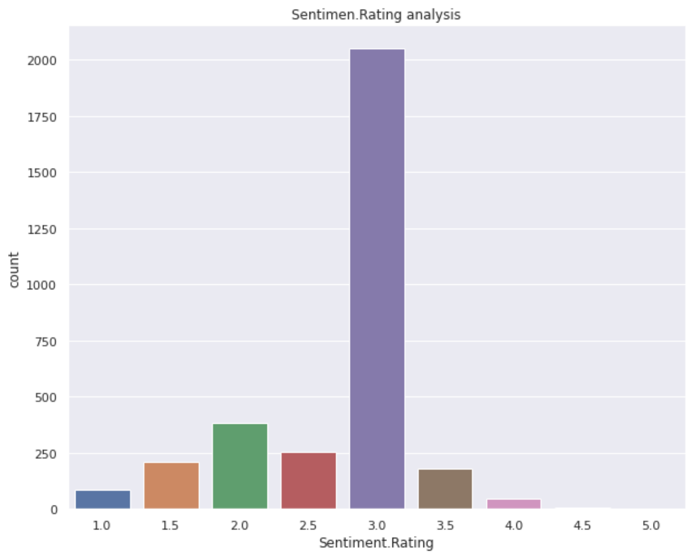
Visualise the count of each category of ‘key’ column and can conclude that JewNAS has the vast majority of entries compared to the rest of the others.

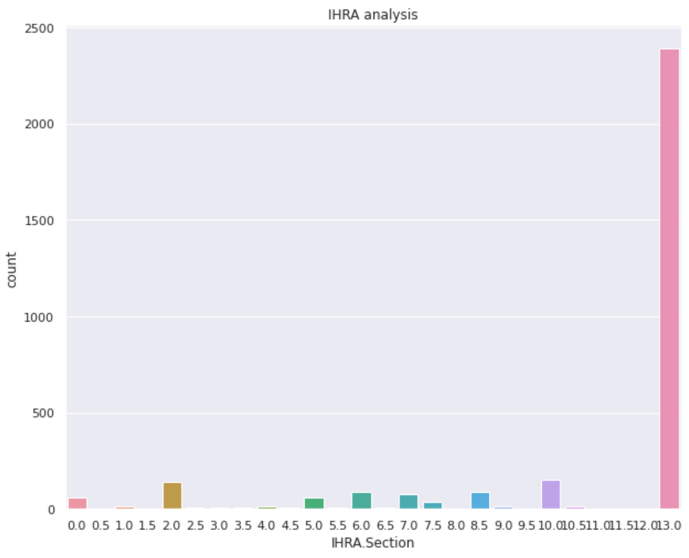

Sarcasm feature analysis
count
3000
2500
2000
1500
1000
500
0
0.0
0.5
1.0
Sarcasm


Key feature analysis
count
1750
1500
1250
1000
750
500
250
0
JewNAS
JewAS
KikesAS
KikesNAS
IsraelNAS
ZioNaziAS
IsraelAS
ZioNaziNAS
key

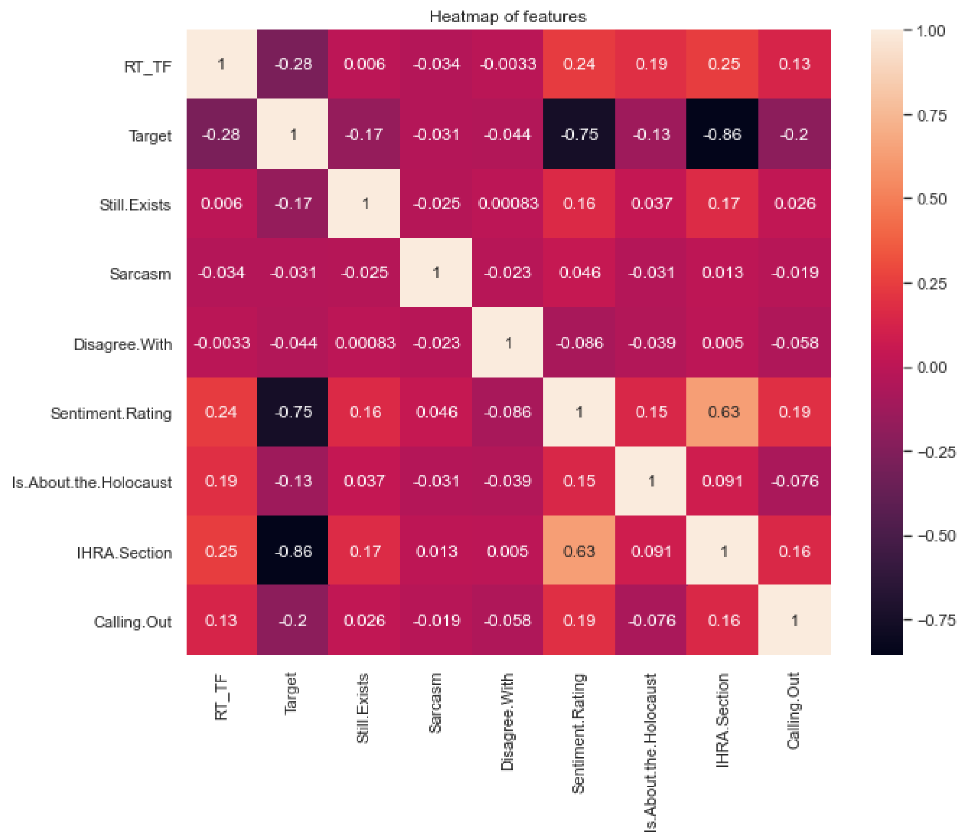

Still Exists analysis
count
3000
2500
2000
1500
1000
500
0
0.0
0.5
1.0
Still.Exists


Disagree analysis
count
3000
2500
2000
1500
1000
500
0
0.0
0.5
1.0
Disagree.With


Sentimen.Rating analysis
count
2000
1750
1500
1250
1000
750
500
250
0
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
Sentiment.Rating


IHRA analysis
count
2500
2000
1500
1000
500
0
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0
8.5
9.0
9.5
10.0
10.5
11.0
11.5
12.0
13.0
IHRA.Section

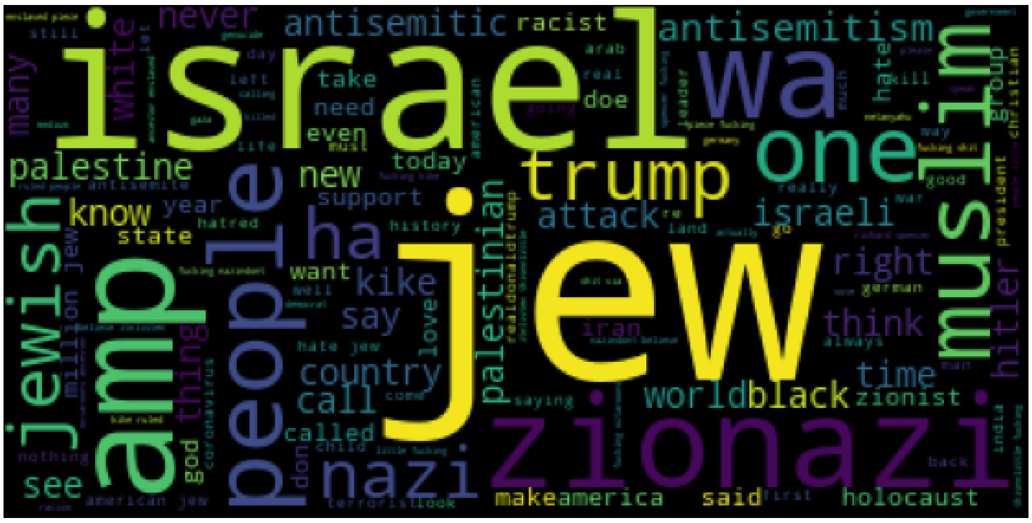
- **Correlation** between different columns is plotted as a correlation plot.



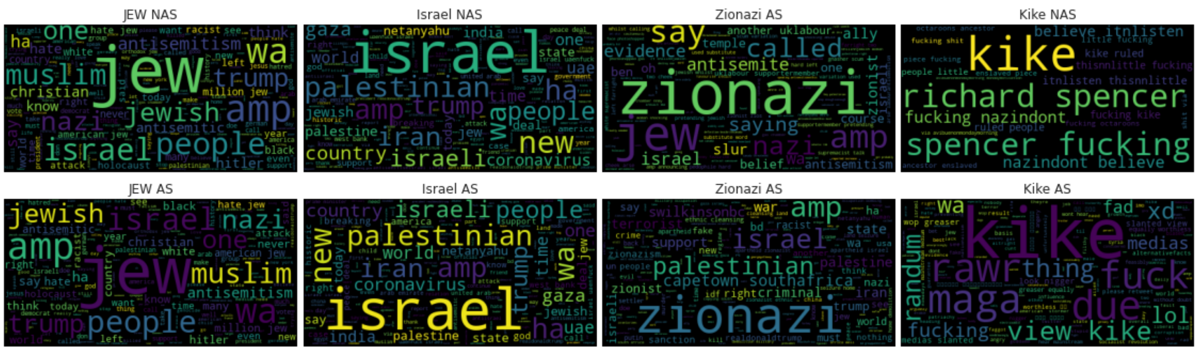
Correlation plot(Heatmap) for all dataset features for getting better insights of dependency of columns on one another.

- **Word Clouds**
  To get a collection of clusters of the words in different sizes with respect to their frequency.

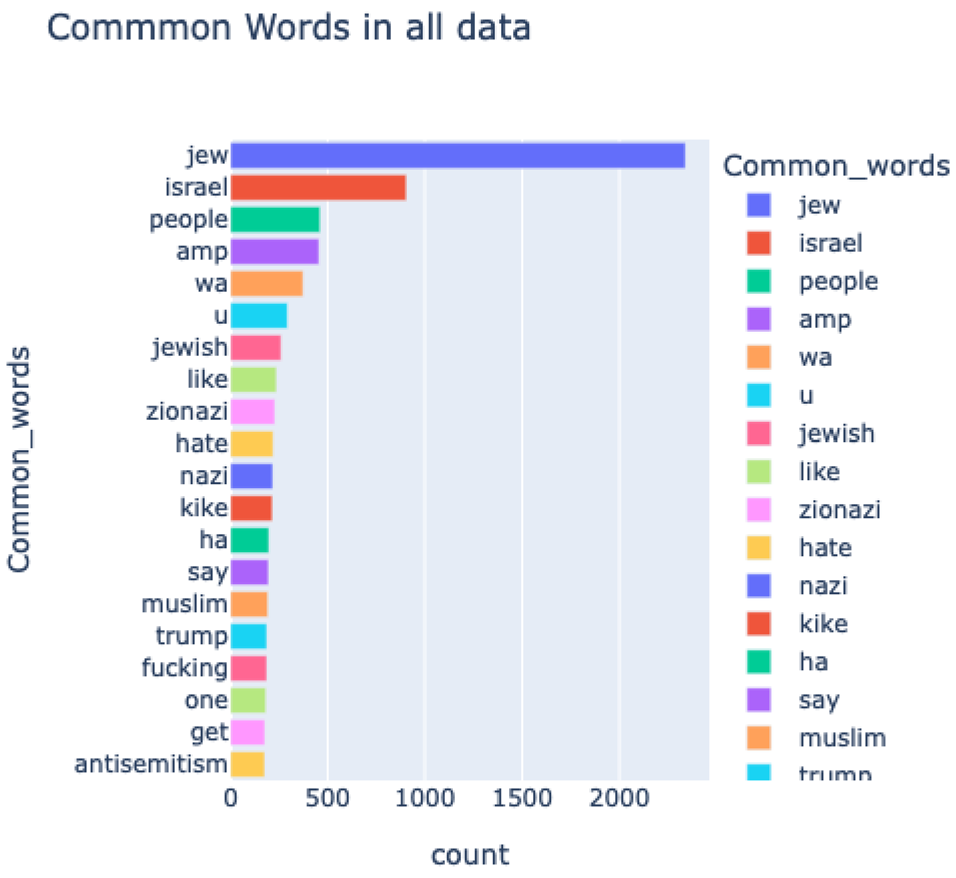
  First word cloud is build from the whole dataset and already can see few words with big sizes that might explain the higher frequency of few categories of 'key' feature.



Second word cloud is a collection of 8 different clouds visualised for each category.



- **Word count** of most frequent words in the whole dataset.Can



Can relate the size of a few words in word cloud with the above word count visualisation.

- For each category (Jew, Israel, ZioNazi and Kikes) and sub category (Antisemitism and Not antisemitism) calculated common words with respect to their count to get relation between different subgroups of tweets from a categ

STEP 3: NAIVE-BAYES CLASSIFIER

- In this step, a Naive-Bayes classifier is implemented where the conditional probability of a sentence is calculated given the word.
- Laplace smoothing is also applied in this step.

STEP 4: DATA PRE-PROCESSING AFTER NAIVE-BAYES

- All the unnecessary and less important columns like addtiional_comments, sample.name, create.date, etc. are removed from the dataset.
- Merged the same columns for different experts x and y by taking the mean of them. These columns are still.exists, disagree.with, sarcasm, is.about.the.holocaust, sentiment.rating, IHRA.section, and calling.out.
- Removed the above columns with x and y ratings separately.
- Merged 2 holocaust columns because some values are in one column and the remaining are in another column.

STEP 5: ONE HOT VECTOR ENCODING

- Performed one hot vector encoding on the key column.

STEP 6: SPLIT TRAIN DATA

- Split train data 80-20 as train-test.

STEP 7: RANDOM FOREST CLASSIFIER

- Predicted the results using a random forest classifier.

STEP 8: RESULTS

- Displayed results as accuracy score and classification report for the given data.
- Test file on which our model is being evaluated might not be best representative of the original population. One reason could be noise present in the test dataset is very large compared to the training dataset.
- Hyperparameter tuning is one of the options that may(or may not) increase the test accuracy but the size of the training dataset is not large enough to implement it effectively.