**Approach**

1. Machine Learning –

**Approach 1: 2 separate vectorizer for positive and negative classes**

1. Downloading the data and loading the data in pandas dataframe

2. Identification of missing values and replacing numerical values with median and the text values with NA

3. Adding the combined column by using the context, statement and justification column.

4. Splitting and Encoding the subject column as one-hot vectors

5. Pre-processing the combined columns to remove punctuations, unwanted space and newline characters.

6. Preparing the binary class label

7. Fitting the tfidf vector on the positive and negative train set classes on the combined column.

8. Transforming the train, valid and test data combined column from the tfidf vectorizer fitted on the train set.

9. Combining the tfidf vectorizer features and the relevant columns of original dataframe and saving the train, test and valid set in feather format

10. Hyper-Parameter tuning on Logistic Regression

Best Binary – ~63%

Best 6 way - ~26%

11. Hyper-Parameter tuning on Random Forest Regression

Best Binary - ~71%

Best 6 way - ~35%

**Approach 2: 1 vectorizer and learning tfidf values from positive and negative classes simultaneously**

Following steps are perforemd while preparing the data

1. Downloading the data and loading the data in pandas dataframe

2. Identification of missing values and replacing numerical values with median and the text values with NA

3. Adding the combined column by using the context, statement and justification column.

4. Splitting and Encoding the subject column as one-hot vectors

5. Pre-processing the combined columns to remove punctuations, unwanted space and newline characters.

6. Preparing the binary class label

7. Fitting the tfidf vector on the train set combined column.

8. Transforming the train, valid and test data combined column from the tfidf vectorizer fitted on the train set.

9. Combining the tfidf vectorizer features and the relevant columns of original dataframe and saving the train,test and valid set in feather format

10. Hyper-Parameter tuning on Logistic Regression

Best Binary – ~63%

Best 6 way - ~26%

11. Hyper-Parameter tuning on Random Forest Regression

Best Binary - 72.5%

Best 6 way - 41.5%

**Deep Learning**

Declaration: this approach is motivated by the FastAI course and I have used the FastAI library methods

Steps followed:

1. Downloading the data and loading the data in pandas dataframe

2. Identification of missing values and replacing numerical values with median and the text values with NA

3. Preparing the binary class label

4. Preparing a reputation label

df\_train['reputation'] = df\_train['false']\*-2 + df\_train['barely\_true']\*-1+df\_train['pants\_on\_fire']\*-3 + df\_train['half\_true'] + df\_train['mostly\_true']\*2

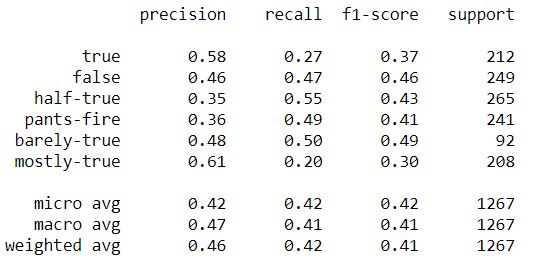
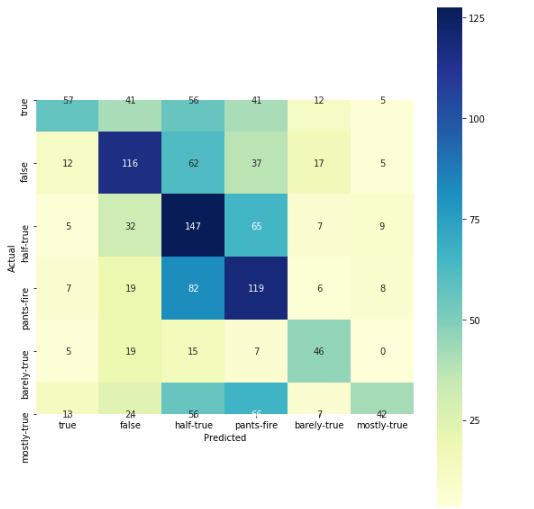
5. Combing all the columns and appending a field token before each field.

6. Using the FastAI approach first we will prepare a language model and saving the encoder weights. Then loading the encoder weights train a LSTM classifier model.

7. Accuracy Binary - ~64%

8. Accuracy 6 way - ~30%

**Best Results**



**Running the code**

**Upload the code in Google Colab and hit run-all**