***MODEL DESIGN:***

* **Importing datasets and handle data to be ready to make preprocessing :**

-We started by importing our libraries like: nltk, numpy, pandas, etc. we loaded our datasets (true news and fake news) as (tn and fn).

-Adding a column named "class" in each of the datasets and gave it values 1 for tn and 0 for fn.

-using pandas function “ concat “ to concatenate both tn and fn in rows and we settled the " ignore\_index = True " so that the new combined dataset will not preserve the original index values, and we added this in new dataframe called "df".

-maxing our dataframe in-place and reset the index using pandas function "sample" so that the "class" column values are mixed.

Then handle columns by collecting subject ,title and text in new col. News

* **Preprocessing:**

**-**After concatenating our datasets, we started the preprocessing step using:

* Removing punctuations and special characters.
* Making all letters lowercase.
* Tokenizing sentences.
* Lemmatizer
* Lastly, removing stopwords.

-Adding this cleaned data in “corpus”.

* **Feature extraction :**
* we used "CountVectorizer()" to extract our features as it converts our corpus into tokens and their counts. Then we transformed our corpus to array using "toarray()" function.
* Now we need the colume of class to identify the feature its +ve or –ve (true , false news)
* *Splitting data* by using sklearn built-in function "train\_test\_split" to split our data into train, test data , to be 80% train and 20% test.
* **Building the model:**

- Naive Bayes methods are a set of supervised learning algorithms based on applying Bayes’ theorem with the “naive” assumption of conditional independence between every pair of features given the value of the class variable.

-We use the Gaussian Naive Bayes so implements [**GaussianNB**](https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.GaussianNB.html#sklearn.naive_bayes.GaussianNB)

- passing our training set (X\_train, y\_train) to the classifier and its score was approximately 93%.

-then, make predicted the test data "X\_test" and stored in "svm\_predicted" to be used in the confusion matrix.

-The confusion matrix takes both predicted data and labels.