## 1: 5-class Text classification

**Solution:**
Implemented various classification algorithms on the following features obtained through the training data set - email texts
FEATURES TRIED:

1. Simple Bag of words
2. Simple Bag of words - stop words
3. Simple Bag of words + Parts of speech (unigram model)
4. Simple Bag of words - stop words + Parts of speech(unigram model)
5. Mean Word2Vec
6. Mean Word2Vec + Parts of Speech

There were numerous other choice for features. TF-IDF(Term frequency inverse document frequency), bigram, trigram of Parts of speech. However it has been observed that they do not provide any substantial improvement over the bag of words or word2vec models. The feature description is as follows.

The **first one** is a standard bag of words model(checks if the word is present in the sentence or not - does not consider frequency of the word).
The **second one** is essentially the same with stop words removed.
The **third one** is the a bag of words vector appended to a vector of size 45 (unique Parts of speech) where we assign 1 to the index of the vector of parts of speech when the certain part of speech is present in the given sentence.
The **fourth one** is a combination of 2 and 3.
The **fifth one** is a word2vec model where each word is represented by a 300 dimensional vector learnt using model proposed by Mikolov et al(Link in references). Here, I have considered the mean of the word2vec representation. Since each word corresponds to a vector, take the sum of all those vectors and divide it by the number of words in that sentence. word2vec representations have been known to perform better in most cases, however after running these sets of experiments, it has been found that word2vec representations are perhaps more helpful when there is need for better inter-pretability of vectors and words. In this case we don't have such a scenario, hence these representations weren't as helpful as one might have assumed.
The **sixth feature** representation is similar to fifth with parts of speech vector appended as in the third feature representation.
Additionally, to reduce the feature vector length, PCA dimensionality reduction was also performed, however it resulted in reduction in performance on the held out validation data set

ALGORITHMS TESTED ON:

1. SVM-C
2. Logistic Regression

3. Random Forest
4. Decision Trees
5. Neural Networks
6. Naive Bayes (Multinomial)

Observations Accuracies of the models have been tested using held out validation data set from the given data set. (80% Training - 20% Validation).

1. word2vec features performed at mean accuracies of 58-64% on highly tuned hyperparameter settings
2. bag-of-word features performed at mean accuracies of 68-74% on tuned hyperparameter settings. (A substantial improvement over word2vec representation)
3. Using Parts of speech improved the accuracies by around 2-4%.
4. SVM-C and Neural networks performed best (70-74%). These are more robust models. Random forest(67-70%) performed second best followed by Naive Bayes models(67-69%), Logistic Regression(62-66%) and Decision Trees(58-64%) (Accuracies in brackets are on Bag-of-words + parts of speech representation)
5. Hyper parameter tuning was the biggest issue since slight change in initial values, plummeted the accuracies many-folds
6. Introducing more hidden layers resulted in diminished performance most likely due to over-fitting. The number of parameters increased however the training data size was only 3700 having dimensions of $\sim 4600$.

Hyperparameter Tuning: All hyperparameter tuning have been done through 5-fold cross validation using grid search on hyperparameters. The hyperparameters that have been tuned are :

1. SVM : C = $1 \times 10^3$
2. Logistic Regression: Optimization technique = LBFGS, C (Inverse of regularization strength) = $1 \times 10^3$
3. Random Forest: Number of trees: 150
4. Neural Network: Hidden Layers = 1(200), Learning Rate = $1 \times 10^3$, Maximum iterations = 15000. Double hidden layer with 400, 200 hidden units each, also gave nearly the same mean accuracies

Results
The final test results have been classified using bag-of-words + pos on a neural network architecture. Neural networks, on an average was giving the best results. The test labels have been generated 10 times and the one with max frequency on the 10 neural networks(same hyperparameters) have been taken. For eg., suppose the test labels generated on test data size = 3 by four neural network classifier were:

1. [1,2,3]
2. [1,2,2]
3. [2,1,2]
4. [1,2,1]

The final results based on max frequency would thus be : [1,2,2]

References:

1. Word2Vec Paper
2. Code written in python notebook format with the help of nltk and scikit-learn libraries. Available upon request