# 16EE10063
# Swagatam Haldar
# Assignment- 5

PERFORMANCE OF SVM ALGORITHM FOR SPAM EMAIL CLASSIFICATION

1. Methodology- Details of the SVM Package used:

In this assignment I have used the SVM package from the Scikit-Learn library. There are three classes in the library SVM namely- SVC, NuSVC and LinearSVC capable of separating multi-class data. The LinearSVC, by default uses the linear kernel whereas in the SVC class we can supply gaussian, polynomial (of given degree), or linear kernels as per the requirement.

The dataset supplied consisted of 57 attributes with 4601 instances. The class of each instance was either 0 (NON-SPAM) or 1 (SPAM)

Some of the features took values over the real line whereas some were boolean. While pre-processing, I have scaled the feature values using the StandardScaler class of Scikit-Learn which standardizes features by *subtracting the mean and scaling to unit variance for each column.* This has provably reduced the training time for the SVM to train.

2. Results:

a) Here is the classification accuracy table as obtained for the **Linear Kernel** for different values of C:

Sl. No.	C =	Test Set Accuracy (%)	Training Set Accuracy (%)
1.	0.001	87.83	88.26
2.	0.01	90.80	92.45
3.	0.1	91.45	93.73
4.	1	91.31	93.82
5.	10	91.82	94.22
6.	100	91.17	94.04
7.	1000	91.24	93.91

b) Here is the classification accuracy table as obtained for the **Quadratic Kernel** for different values of C:

Sl. No.	C =	Test Set Accuracy (%)	Training Set Accuracy (%)
1.	0.001	59.59	62.21
2.	0.01	61.19	62.79
3.	0.1	71.25	72.98
4.	1	82.55	85.46
5.	10	91.89	95.12
6.	100	91.45	97.48
7.	1000	89.5	99.07

c) Here is the classification accuracy table as obtained for the **Gaussian(rbf) Kernel** for different values of C:

Sl. No.	C =	Test Set Accuracy (%)	Training Set Accuracy (%)
1.	0.001	59.52	61.06
2.	0.01	69.51	71.89
3.	0.1	89.21	91.49
4.	1	92.25	95.09
5.	10	92.90	97.14
6.	100	92.54	98.79
7.	1000	91.60	99.47

3. Special Observations:

The parameter C specifies the extent of tolerance that is permitted while classifying the training instances into classes.
If **C is large**, it is expected that very less number of training examples will get misclassified as high C will try to place the hyperplane in such a way to minimise the error in training set.
However, this setting is very prone to overfitting in the training Set and the same has been observed in case of non-linear Gaussian and Quadratic kernels.
Also, the linear kernel got almost saturated for high C values proving that it is less likely to overfit than its non-linear counterparts.

For **small C**, quite intuitively, as it gets quite a large degree of freedom, it is expected to get low classification accuracies for both training and test set. The tabulations are consistent with this fact.