**4. Apply the LIME explainer on the classifier you implemented in the previous step for the Tubespam dataset and write a 2 page summary with your findings (40 points).**

Random Forest Classifier was implemented in task #3. The algorithm has three fundamental hyperparameters, which were set prior to preparing. These incorporate hub size, the quantity of trees, and the quantity of highlights examined. From that point, the random forest classifier was utilized to take care of classification. The reason for choosing random forest is because of its key benefits which include the decreased danger of overfitting, simple to decide highlight significance, and the adaptability it offers. Decision trees risk overfitting as they will in general firmly fit every one of the examples inside preparing information. Random forest makes it simple to assess variable significance, or commitment, to the model. One can figure the relative component importance. It can give great exactness regardless of whether the higher volume of information is absent. There are a couple of approaches to assess significance. Highlight packing additionally makes the random forest classifier a powerful instrument for assessing missing qualities as it keeps up exactness when a segment of the information is absent. These reasons stated are why we considered random forest to be a good option particularly for this assignment.

In order to complete task #3, we started off the process by connecting to google drive since colab was used for the previous tasks. Then installed the required libraries including lime and sklearn in order to deal with regression classification, clustering algorithms, and random forests. Next in sequence, we read the data which printed 350 rows and 5 columns. In terms of fetching the data and training a classifier, the row for is attained for content (description) and class (0 or 1). The data is split with a size of 0.2, 20%. The content is then vectorized as a result of the random forest classifier inability to accept strings, so int is taken instead. Once that’s done, we run the random forest classifier, following with fitting the data. At that point it's time to predict the data, which ended up with a chance of 97% of getting it correct. Make pipeline is imported to construct from the given estimators. The reason for class names being regular or spam is due to the fact that the data is predicting if the content is regular meaning if it's real or spam. Next, an explanation is generated for 6 figures, with a probability of 93% when the index was 2. Conclusively, the difference between the original prediction and our prediction ended up being 0.0. Lastly, the graphs visualize the data obtained throughout the previous steps which showed local explanation for class Spam in horizontal bar graph format and the prediction probabilities in terms of Regular and Spam.

Random forests turned out to be great for specific arrangements of issue types given particular sorts of information. From conducting task #3 it has shown to be a remarkable calculation, due to the solid outcomes and high accuracy. Reason being they required practically no information readiness. They dealt with paired highlights, downright highlights, and mathematical highlights with no requirement for scaling. Also because of its versatility and simplicity. Random Forests was applicable to a wide assortment of displaying assignments, it functioned admirably for regression errands, and for classification tasks. In the event that not of the subsequent model, of the learning calculation itself. The essential random forest learning calculation was written in a couple of lines of code. In spite of the fact that obviously the random forest strategy is less sensitive when it comes to varieties in the preparation set. Nevertheless, we guaranteed that the classifier ensured better drawdown control and higher strength. These benefits were sufficiently significant to make the additional intricacy worth it.