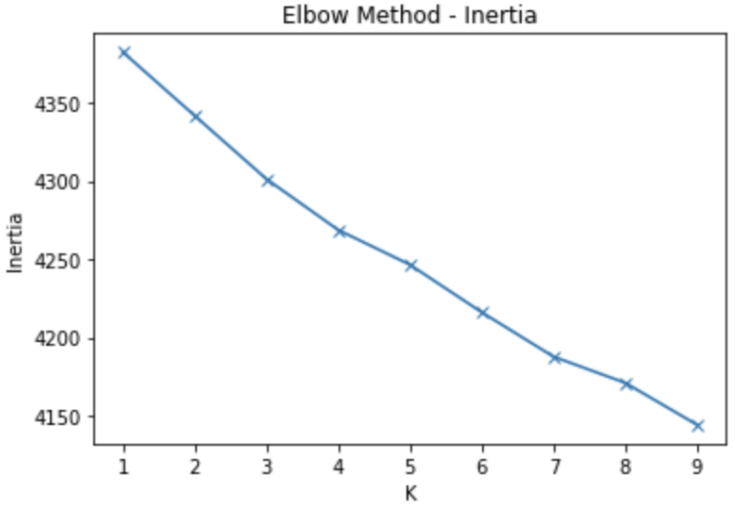
Practical Machine Learning Documentation

Iordachescu Anca Mihaela, 407

The assigned task was modelling two different unsupervised machine learning models. I have chosen K-means and DBSCAN models and the dataset is called Spam Text Message Classification which contains message texts with two types of labels: ham or spam.

1. K-Means model

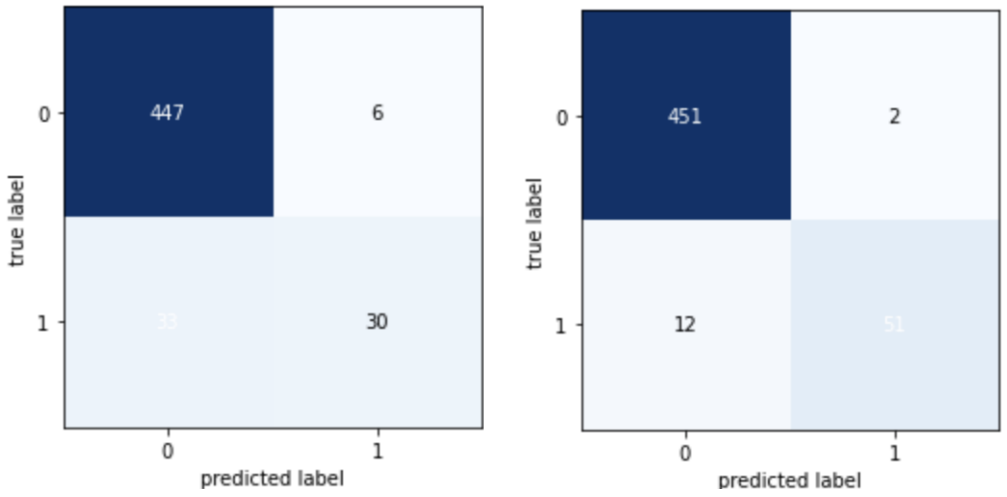
- First of all, because of the fact that this dataset contains only a CSV file, I had to **split the dataset into two smaller dataset**: one for training and one for testing. The percentages of splitting are: 90% training data and 10% testing data.
- The next step was **preprocessing the text data**. For each message text, I have converted the uppercase letter into lowercase and removed any character that is not alpha or space. After splitting the message into words, I have removed the English stop words that appear in the message text.I have extracted the **TF-IDF features** in order to get a matrix that shows how relevant words are to message texts, setting the *max_features* param equal to 500 in order to select the top 500 term frequency words.
- After that, I have declared and **hyper-tuning** the K-Means model. I have used the **Elbow method** and I have tried different values for the *n_cluster* param of the model using GridSearchCV, ***n_cluster*** $\in$ ***{1, 2, … 10}***. As the Figure below shows, the best value for parameter n_cluster is 10, the inertia score being 4125.



- Once I trained the model with the **param = 10** that obtained **the best performance**, due to the fact that the number of clusters is bigger than the number of actual classes, I had to assign each cluster to a single class (ham/spam) that has the most samples within the cluster. I calculated the confusion matrix on the train dataset in order to achieve the new labels for the clusters and then I assigned each test sample to the cluster which has the minimum square distance between it and the cluster ones. In the following table there can be seen: the cluster and the label that it was assigned to (*label 0 - ham / label 1 - spam*).

Cluster	0	1	2	3	4	5	6	7	8	9
Label	0	0	0	1	0	0	0	0	0	0

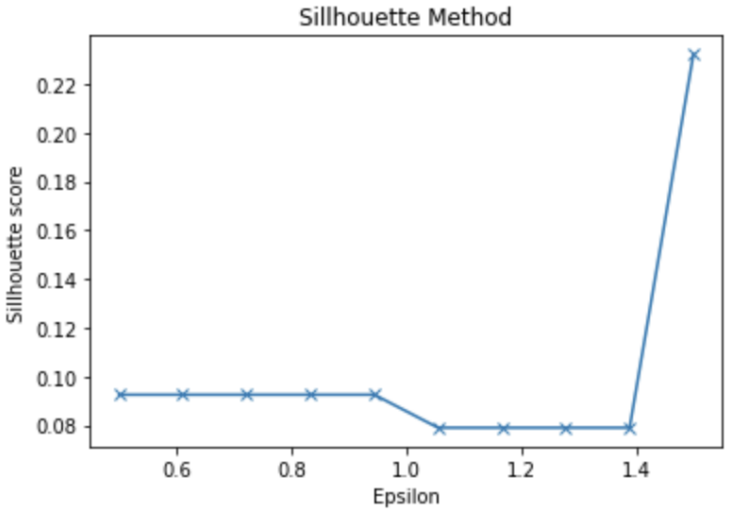
- **The accuracy of the model** tested on the testing set is equal to **0.92**. When it comes to comparing with random chance, because of the fact that this problem is a binary classification, **the random chance** is equal to 0.5 and it is obvious that my model performs better than random chance. For the supervised model, I have trained a **SVM** with the default params(C=1, kernel=rbf) on the same data and got an accuracy score equal to **0.97**. Looking at the both accuracy and confusion matrices, it is obvious that the **supervised model performs better** than the KMeans, having the number of false positive and false negative three times smaller than those of Kmeans. However, the K Means model performs very well, achieving an accuracy above 0.9.



Left: confusion matrix of K Means; Right: confusion matrix of SVM

2. DBSCAN model

- First of all, because of the fact that this dataset contains only a CSV file, I had to **split the dataset into two smaller dataset**: one for training and one for testing. The percentages of splitting are: 90% training data and 10% testing data.
- The next step was **preprocessing the text data**. For each message text, I have converted the uppercase letter into lowercase and removed any character that is not alpha or space. After splitting the message into words, I have removed the English stop words that appear in the message text and lemmatize the remaining words. I have extracted the bag of words features using **CountVectorizer**, extracting both unigrams and bigrams. I set the *max_features* param equal to 100 in order to select the top 100 term frequency words.
- After that, I have declared and **hyper-tuning** the DBScan model. I have used the **Silhouette method** in order to get the best performance and I have tried different values for the *eps* param of the model, ***eps*** $\in$ ***linspace(0.5, 1.5, 10)*** with linspace = evenly spaced numbers calculated over the interval (0.5, 1.5). As the Figure below shows, the best value for parameter eps is 1.5 which obtains 5 clusters, the silhouette score being 0.24.

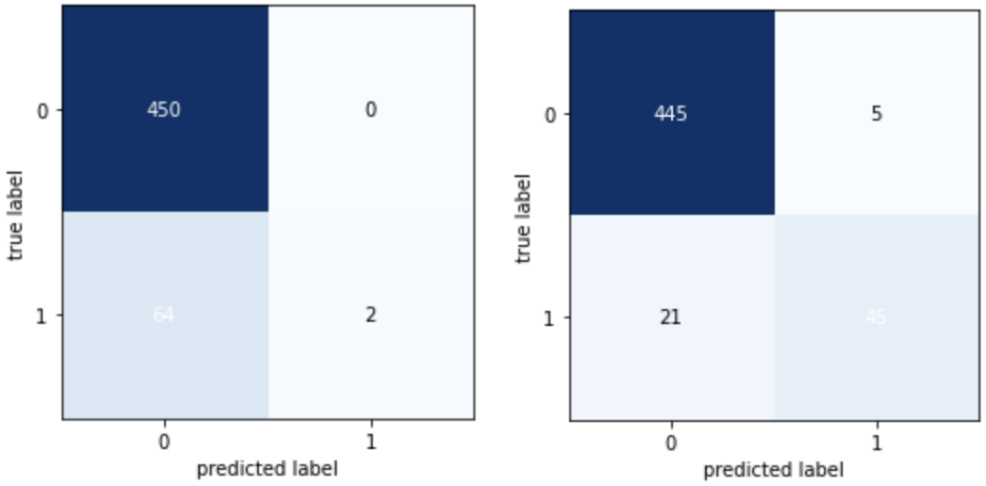


- Once I trained the model with the **eps = 1.5** that gives 5 clusters and that obtained **the best performance**, due to the fact that the number of clusters is bigger than the number of actual classes, I had to assign each cluster to a single class (ham/spam) that has the most samples within the cluster. I calculated the confusion matrix on the train dataset in order to achieve the new labels for the clusters and then I assigned each test

sample to the cluster which has the minimum square distance between it and the cluster ones. In the following table it can be seen the cluster and the label that it was assigned to (*label 0 - ham / label 1 - spam*).

Cluster	0	1	2	3	4
Label	0	1	1	1	0

- **The accuracy of the model** tested on the testing set is equal to **0.87**. When it comes to comparing with **random chance**, because of the fact that this problem is a binary classification, the random chance is equal to 0.5 and it is obvious that my model performs better than random chance. For the supervised model, I have trained a **SVM** with the default params (C=1, kernel=rbf) on the same data and got an accuracy score equal to **0.95**. Looking at the both accuracy and confusion matrices, it is obvious that the **supervised model performs better** than the DBScan, the DBScan number of false negatives being greater than the DBScan one and having no false positives. However, the K Means model performs very well, achieving an accuracy above 0.8.



Left: confusion matrix of DBScan; Right: confusion matrix of SVM