The ‘spam’ data set contains 58 variables. The first 48 variables contain the frequency of the variable names (e.g. business) in emails. If the variable name starts with a number (e.g. num650), it indicates the frequency of the corresponding number (650 in this example). The variables 49 to 54 indicate the frequencies of characters ';', ‘(’, ‘[’, ‘!’, ‘\$’, and ‘\#’, whereas the variables 55-57 contain the average, longest and total run-length of capital letters. The variable 58 indicates the type of the mail and it´s classification to either 'nonspam' or 'spam', i.e. unsolicited commercial e-mail. The scales (variances) of these variables are quite different, hence the reason why we decided to scale the data set to improve the classification task:

Variable (Nr.)	Min	Average	Max
order (9)	0	0.09	5.26
capitalTotal (57)	1.0	283.3	15,841

Table 1: Example for different scale of variables - information of summary command for ‘spam’ data set

Interpretation of Results

We interpret our results by interpreting the proportion of votes for each class, which shows us the probability, if the prediction is correct in percentage, and the prediction accuracy in percentage, by comparing our predicted labels with the true labels.

With K=1, we train the training set in a relatively small range, the approximation error caused by ‘study’ will be reduced. However, the estimated error of learning increased because the predicted results are very sensitive to the adjacent points.

The probability that the prediction is correct is clearly 100% ‘spam’ or 100% ‘nonspam’, as we only consider one nearest neighbour this result is not a surprise. We might need to consider more nearest neighbours to solidify our prediction, in case the first nearest neighbour happens to be a noisy point, then our prediction will be wrongly classified. The KNN model with K = 1 is too complicated and will lead to a problem of overfitting in this case, which means we fit too closely to a limited set of data, which reduces the predictive ability and maintain a high flexibility.

With K = 9, our predictions are more stable and therefore there is less variance. By averaging over more data points for each prediction, this model reduces the variance by averaging together more noisy terms. The probability that the prediction is correct varies between both supplements, ‘spam’ and ‘nonspam’. In case the prediction is 75% correct for ‘nonspam’, there is a 25% chance that ‘spam’ might be the correct classification for this observation. With K=9 we still predict a good number of labels with 100% exactness towards one class.

With K = 25, the estimated error has been reduced by enlarging the training neighbourhood. However, the approximate error will increase i.e. the area predicting each class is more “smoothed”. The probability that the prediction is correct have decreased significantly. The results show that almost all predictions show a certain percentage of exactness for both ‘spam’ and ‘nonspam’. Prediction, where the probability is correct for 100% are infrequent.
The variables which are far away from our prediction now also have an impact on our results. When K increases, the KNN model will become too generalized and lead to an underfitting problem, which means we consider too many nearest neighbours, which makes our prediction inflexible.

According to our classification results, three different K values were used in the KNN model, which are 1, 9 and 25. The corresponding accuracies calculated were 0.81, 0.83 and 0.75, respectively. As we can see from the results, the highest accuracy appears when k=9, which is also in line with our expectations. Thus, from Figure 2, we can see that there was an upward trend in accuracy when K increased from 1 to 9, however the accuracy declined when K was further increased to 25.

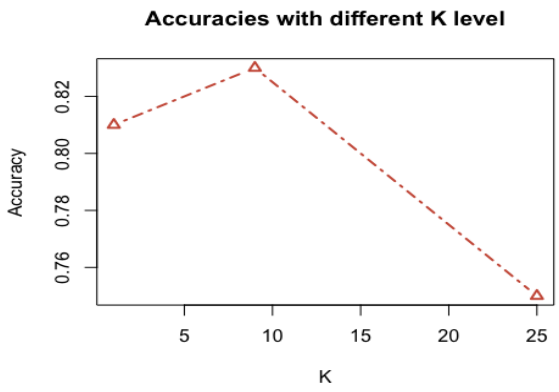

Figure 2: Accuracy of KNN-Classification, when K = 1, 9, 25

In conclusion, we can say that K=9 seems to be the optimal K-value out of K=1, 9, 25. As we consider a number of nearest neighbours high enough to avoid the underfitting problem, but small enough to avoid an overfitting problem, which is directly linked to the flexibility of the model: with increase number of k the flexibility decreases.

However, we have not tested more K values. In reality, we may test more K values by using the k-fold cross-validation method in order to find the optimal K level to avoid an over- and underfitting of our data and further improve the accuracy of our model.