Learning from Data - Week 5

Assignment 5

Mart Busger op Vollenbroek
S2174634

5.1.1

Using the default settings and five-fold cross validation, the following scores were achieved:
Accuracy: 0.828668238196
Precision: 0.828838286695
Recall: 0.828668238196
F1-score: 0.828629592989

5.1.2

C	Precision	Recall	F1-score
1.0	0.8288	0.8286	0.8286
0.5	0.8238	0.8236	0.8236
0.1	0.7740	0.7739	0.7739
1.5	0.8267	0.8266	0.8266
5.0	0.8117	0.8116	0.8116

The C value tells the SVM how much you want to avoid misclassifying each training sample. If you choose a larger value for C, the hyperplane chosen for the classification so that there is a smaller chance of misclassifying training examples. For smaller values of C however, the penalty is for misclassifying is lower, which implies a bigger hyperplane and a higher chance of misclassifying some training points.

5.1.3

The results from using the radial basis function (rbf) kernel are found in the table below:

C	gamma	Precision	Recall	F1-score
1.0	**0.7**	0.8259	0.8258	0.8258
1.0	**1.0**	0.8257	0.8256	0.8256
1.0	**0.9**	0.8262	0.8261	0.8261
1.0	**1.5**	0.8215	0.8215	0.8214

Based on these scores, the assumption that linear SVM's outperform rbf SVM's seems correct. The margins however, are small and changing the parameters of both SVM's did not do much. Maybe adding different features can help with achieving higher scores.

5.1.4

For the final version of the SVM, the linear kernel is used with a C = 1.0 value.

	Precision	Recall	F1-Score
Default	0.8288	0.8286	0.8286
Stopwords	0.8089	0.8088	0.8088
Bigrams	0.8334	0.8333	0.8333
Stopwords + Bigrams	0.7977	0.79750	0.7974
Trigrams	0.7990	0.7988	0.7987
Stopwords + Trigrams	0.6877	0.6839	0.6821
Stemming (Snowball)	0.8280	0.8280	0.8279
Bigrams + Stemming	0.8390	0.8389	0.8389

Using several different settings, the highest F1-score my system achieved was 0.8389, which is 1,2 % higher than the default settings. What I found a bit odd, was the fact that using stopwords seemed to harm the system's performance. Perhaps these stop words have some semantic value after all? Also using bigrams instead of trigrams worked better, especially after adding Snowball Stemming.

5.2.1

Using the K-Means algorithm, the following confusion matrix was obtained:

['health', 'books', 'camera', 'software', 'music', 'dvd']
[[894 76 0 886 41 103]
 [659 979 1 158 152 51]
 [530 56 959 349 40 65]
 [569 146 1 1057 66 76]
 [836 511 0 152 393 108]
 [670 841 0 158 222 109]]
Rand-Index: 0.11313016337995606
(0.18617047260842334, 0.21190319394203908, 0.19820511165826063)

1.) The instances are not evenly distributed among the clusters. There were only two classifications of camera where it was not actually a camera. There were however more cases were something else was predicted while it was actually a camera. Another thing that stands out is that many instances were put in the health cluster, especially compared to the other classes.
2.) Seeing that data points are often clustered to 'health', that cluster is most often confused.
3.) Not very good, seeing as the Rand-Index only scores 11,3%. This Rand-Index is adjusted for chance. Using the homogeneity, completeness and v-measure, one can see that these also score very low.

5.2.2

Table for sentiment analysis:

Settings	Homogeneity	Completeness	V-measure	Rand-Index
Default (n_iter = 10)	0.00127	0.00131	0.00129	0.00173
Snowball Stemming, n_iter = 10	0.00100	0.00103	0.00101	0.00135
Stemming, stop words filtering, n_iter = 10	0.00131	0.00135	0.00133	0.00179
Stemming, stop words filtering, n_iter = 100	0.00133	0.00137	0.00135	0.00182

Table for topic classification classes ‘health’ and ‘software’

Settings	Homogeneity	Completeness	V-measure	Rand-Index
Default (n_iter = 10)	0.05115	0.05198	0.05156	0.06991
Default (n_iter = 10), sublinear_tf = True	0.18958	0.19491	0.19221	0.24408
sublinear_tf = True, Snowball Stemming, n_iter = 10	0.34988	0.3797	0.36419	0.38476
sublinear_tf = True, Stemming, stop words filtering, n_iter = 10	0.34916	0.36874	0.35868	0.40532
sublinear_tf = True, Stemming, stop words filtering, n_iter = 100	0.28930	0.30431	0.29662	0.34706

Using the default settings, the scores for topic classification are higher than the scores for sentiment analysis. After adding the sublinear_tf paramater in the vectorizer, the scores became even better for the topic classification. This had little effect on the sentiment analysis however and is therefor not taken into account in the corresponding table.