LING 572 – HW4

Q2

Table 1: Test accuracy using **real-valued** features

k	Euclidean Distance	Cosine function
1	0.62	0.72
5	0.636666666666667	0.666666666666667
10	0.646666666666667	0.633333333333333

Q3

Table 2: Test accuracy using **binary** features

k	Euclidean Distance	Cosine function
1	0.623333333333333	0.826666666666667
5	0.536666666666667	0.823333333333333
10	0.563333333333333	0.813333333333333

Q5

Table 4: Test accuracy using **real-valued** features, **k=1**, cosine function

***p*0**	**Number of related features**	**Test accuracy**
baseline	32846	0.72
0.001	2401	0.743333333333333
0.01	3895	0.743333333333333
0.025	5223	0.753333333333333
0.05	7484	0.736666666666667
0.1	8499	0.75

Q6

Table 5: Test accuracy using **binary** features, **k=10**, cosine function

***p*0**	**Test accuracy**
baseline	0.813333333333333
0.001	0.846666666666667
0.01	0.836666666666667
0.025	0.843333333333333
0.05	0.82
0.1	0.853333333333333

Q7

For real value features, the Cosine function produced better accuracy over the Euclidean distance when K=1 (Table 1). As we increased the value of K, the Cosine function accuracy declined while the Euclidean distance accuracy improved slightly. The accuracy difference between Cosine function and Euclidean distance for real-valued features are marginal when K > 1.

Binarizing the features did not improve the performance of the Euclidean distance model (Table 2). In fact the performances for Euclidean distance with Binary features, especially when K=5 and K=10 were generally lower. On the other hand, the performance for Cosine function using binary features improved significantly across all K values and in addition, the accuracy variances among the different K values were generally small.

Filtering off certain real-valued features improved the performance of the Cosine function at K=1 (Table 4). However, the performance improvement was not directly proportionate to the number of features excluded. Trial and error using different ***p*0** values was required and in our experiment, the optimal ***p*0** value that yielded the best accuracy was 0.025.

Binarizing and reducing the features with K=10 also improved the performance of the Cosine function (Table 5). Similar to real-value features experiment from Table 4, trial and error was needed to find the optimal ***p*0** value and in this experiment, the best accuracy was achieved when ***p*0** value was equal to 0.1.

Overall, kNN model that is based on the Cosine function produced consistently better result at all K values as compared to one that is based on Euclidean distance. Binarizing the features would improve the performance of a kNN model that is based on the Cosine function but this has no similar effect on one that is based on Euclidean distance. In fact at certain K values, binarizing the features could even degrade the performance of the kNN model that is based on Euclidean distance. Excluding unrelated features would also improve the performance of the kNN model that is based on the Cosine function.

Additional note:

Feature filtering program included in the submission. To run the program type:

filter_feature.sh *input_vector_filename feat_list p_value* > *output_vector_filename*

input_vector_filename is the input vector filename for example, train.vectors.txt

feat_list is the name of feature list

p_value is the ***p*0** value

output_vector_filename is the output file containing the related features after filtering

End of HW4 – Joint submission by

Scott Mantei

Wee Teck Tan

Course Name: LING 572