ASSIGNMENT-2 REPORT
DATA ANALYTICS FOR WELL-BEING

AKSHAY AGARWAL
(903000875)

Classifier Output:

CLASSIFIER		Accuracy	Precision	Recall	F1
Naïve Bayes	**LIWC**	0.897947328	0.899401198	0.891218354	0.895291079
(Multinomial)	**Threshold 50**	0.530880595	0.868131868	0.043098745	0.082120582
	Threshold 100	0.541506176	0.845410628	0.04867872	0.092056812
	Threshold 500	0.888431399	0.982155113	0.784109589	0.87202925
SVM	**LIWC**	0.913051898	0.911357341	0.910996835	0.911177052
	Threshold 50	0.513082747	0.5	0.000545554	0.001089918
	Threshold 100	0.52251295	0.5	0.000834492	0.001666204
	Threshold 500	0.987913402	0.97643909	0.999178082	0.987677725
knn - 1	**LIWC**	0.897172734	0.898602794	0.890427215	0.894496324
	Threshold 50	0.48877673	0.487573	0.979268958	0.651010971
	Threshold 100	0.48293266	0.479667031	0.977746871	0.643596082
	Threshold 500	0.954044362	0.976630121	0.92739726	0.951377178
knn - 10	**LIWC**	0.919635941	0.933169332	0.900316456	0.91644856
	Threshold 50	0.489972108	0.488198589	0.98172395	0.65211089
	Threshold 100	0.486120335	0.481212287	0.976077886	0.644622026
	Threshold 500	0.990569797	0.983517968	0.997260274	0.99034145
knn - 100	**LIWC**	0.916150271	0.917830076	0.910205696	0.914001986
	Threshold 50	0.535529287	0.796491228	0.061920349	0.114907618
	Threshold 100	0.543764112	0.80075188	0.059248957	0.11033411
	Threshold 500	0.98871032	0.980070024	0.996986301	0.988455792
knn - 1000	**LIWC**	0.909372579	0.906471981	0.908623418	0.907546424
	Threshold 50	0.532341612	0.811158798	0.051554828	0.096947935
	Threshold 100	0.541771816	0.808510638	0.052851182	0.09921671
	Threshold 500	0.979545756	0.960242233	0.999178082	0.979323308

Discussion:

a. The baseline chance model would have classified a tweet as sad tweet 48.5% of the times, and as happy tweet 51.5% of the times. Naïve Bayes i.e. multinomial naïve Bayes performed only slightly better than this for n-gram models with threshold values as 50 and 100. Multinomial naïve Bayes is the logical choice since the feature vector in neither binary nor along Gaussian distribution, and binary and Gaussian naïve Bayes performed worse or closer to the chance model. The reason for poor performance of Naïve Bayes in the above case is probably because of the feature vector that we have selected. For threshold 50 and threshold 100, the no.of n-grams are very high, leading to very few matches of n-grams for every tweet, and resulting in most values being close to zero as frequency count for an n-gram, thus resulting in the model performing poorly and close to chance model. We see that this performance drastically improves with threshold 500 and affect/liwc features. The reason for threshold 500 working is that this leads to a drastic reduction in the no.of n-grams being considered, and thus is more representative of the sentiment expressed in the tweet. In case of LIWC, since instead of looking at each n-gram we classify the n-grams or words into a sentiment and then consider their aggregated values, therefore the model is able to map smaller frequencies as well and provides a more accurate representation of the sentiment expressed in the tweet, leading to better accuracy, precision and recall.

   In case of SVM too we see that LIWC and threshold 500 perform much better and in fact classify the tweets with high accuracy as again the representative sentiment is captured in a stronger way, especially with threshold 500. We expected and observed similar trends for all values of k in knn classifier.

   Every classifier performs best with threshold 500, and the reason for this is that in LIWC we are considering words whose occurrence is low, and are not weighting the frequency of those occurrences. Threshold 500 represents to a large extent, the n-grams which have been used in this particular set of tweets and thus the frequency counts of each n-gram provide a more comprehensive and accurate classification.

b. Classifier performance depends greatly on the data distribution. In the case of the data considered, even knn performs extremely well, in-fact outperforming SVM in some cases, and especially in those cases where SVM performance is poor e.g. threshold 100 and threshold 50. This is because of the fact that knn looks at the nearest neighbors for classification, and thus is able to capture the minuscule relationships when it comes to looking at a larger feature vector set in case of threshold 50 and threshold 100. Consider the following case, in case of SVM, it tries to take the data to higher dimension in order to classify the same, however in case of knn, it considers the nearest neighbors based on Euclidean distance, and thus maps more closely to similar sentiment tweets. This is because n-grams for happy and sad tweets are fundamentally different, due to which knn looks more closely only at relevant tweets. Knn-1000 performs best since it is able to take into account more close and relevant tweets. We see that in case of threshold 50 and threshold 100, for knn1 and knn10, precision value is low, the reason for this being that it classifies more tweets as fp. This is because since it is dependent on Euclidean distance, and considers average of neighbors at same distance, thus it takes into account a much larger number of neighbors, as given the size of feature vector that is bound to happen.

Some classifiers perform better than the others because of the fundamental way that they look at data, and in what representation the data is present, their bias variance, if the data is linearly separable or not, how we train the classifiers etc. SVM in this case should and did perform well as expected, however knn also performed comparatively due to the fact that similar data points might be clustered together, especially for more representative feature vectors. We could have further optimized by using boosting etc.

c. Choice of feature vectors greatly defines the performance of the classification algorithms. This is because some feature vectors are able to capture the information separating the outputs more clearly than the rest. In the above case, we see that threshold 500 performs best, this can be thought of because of the reason that it captures the most frequently occurring n-grams and thus can act as defining features, also it focuses more on more frequently occurring labels as compared to LIWC, which considering even less frequently occurring terms for a particular affect results in loose classification. Threshold 500 includes bi-grams and tri-grams as well and is thus able to perform much better, since relationships and phrases are more effectively captured in tri-grams. Though bi-grams often don't give good results by themselves, however since they are accompanied with a list of restricted unigrams(which also occurs in liwc dictionaries), thus we see that the classifier performs really well. In case of threshold 50 and threshold 100 n-grams, since there are too many n-grams, thus the data becomes less linearly separable, resulting in loose and not so good classification, and poor accuracy. We can perform PCA in order to choose the most-effective set of feature vectors and improve our classification algorithm further.

   Threshold 50 and threshold 100 do not perform too well probably because of the fact that the bias is more as most values are zero due to large feature vector set. Another important thing is the implementation detail, the n-gram frequency in tweet had been calculated, rather than representing n-gram frequency in a tweet by its frequency in corpus. This is an important level of detail, and would have improved the performance on threshold 50 and threshold 100, however intuitively I felt that threshold 500 should perform better if considering frequency on n-gram in tweet. This is also a part of the reason that knn performs better since now the nearest neighbors are different.