Assignment 2
Cogsci 188 Summer
Sentiment Classification

Gyuseung Hwang
A13528608

K Nearest Neighbors (KNN)

KNN algorithm calculates distance from a certain to every data points in the dataset. Then, the model classify a point by looking at the top K elements.

Following three parameters are the most important for KNN.

weight : By default weight is all uniform. Setting weight = ‘distance’ makes closer points have higher influence when classifying a point.

Algorithm: By default, the model automatically choose the most appropriate algorithm to find nearest point based on the input values. For example, using kd_tree will be more efficient(O(n logn)) than using brute force(O(n)).

p (Power parameter) : By default, the m is 2, which is Euclidean metric. p = 1, is for manhattan distance and user can use any function.

Here we used p = 5, and default parameters for rest. Because the greater p value don’t help improving accuracy and p less than five can increase bias. This model is useful because it doesn’t require any training. However, as the number of data increases, it takes more time to predict a single point.

Perceptron

Perceptron makes its predictions based on a linear predictor function combining a set of weights with the feature vector.

Following three parameters are the most important for Perceptron.

max_iter : The maximum number of passes over the training data. By default it is 1000.

shuffle : whether or not the data should be shuffled after each epoch. By default it is true.

penalty : Constant that multiplies the regularization term if regularization is used. Defaults to 0.0001.

Following table is how the changes in n affect the total time and accuracy. It shows that Increasing maximum iteration don’t help to optimize the perceptron model. Perceptron model is fast and accurate, so it is widely used in neural networks. However, it doesn’t not give best separating plane compares to other methods like regression and SVM.

	N = 1	N = 10	N = 100	N = 1000
Time	0.05ms	0.20 ms	1.60ms	13.30ms
Accuracy	0.82	0.80	0.82	0.79

Random Forest

Use multiple decision tree classifiers on datasets and uses averaging to prevent over-fitting.

Following three parameters are the most important for Random Forest.

min_samples_split : the minimum number of samples required to split an internal node:

max_depth : Maximum depth of each tree. By default it is None.

max_features : The number of features to consider when looking for the best split.

I picked this model because it is relatively faster than other regression algorithms (took <0ms). Here is the table why I chose n_estimater = 100, and max_depth = 10. In general, decision tree works well with both continuous and categorical inputs. However, the error rate is high when training set is small in comparison with the number of classes.

	N = 10, n_est = 10	N = 10, n_est = 100	N = 100, n_est = 10	N = 100, n_est = 100
Accuracy	0.78	0.75	0.81	0.82

Logistic Regression

Logistic regression is the appropriate regression analysis to conduct when the dependent variable is binary by using logistic function.

Following parameters are the most important for Logistic Regression.
penalty : Used to specify the norm used in the penalization
C : Inverse of regularization strength. Smaller values specify stronger regularization
class_weight : Weights associated with classes in the form {class_label: weight}. If not given, all classes are supposed to have weight one.

I picked this model because it is best known binary classifier algorithm. Since, default parameter gave me pretty good accuracy, I did not change it. Logistic regression is fast O(p) because it only has to calculate one equation for each point. However, we can't solve non-linear problems with logistic regression since it's decision surface is linear.

Support Vector Machine

Draw a line that maximize the sum of all distances from all data points and use that line to classify a data. It also is effective in high dimensional spaces.

Following parameters are the most important for SVM.
kernel : Specifies the kernel type to be used in the algorithm. It must be one of ‘linear’, ‘poly’, ‘rbf’, ‘sigmoids’, ‘precomputed’ or a callable function.
degree : Degree of the polynomial kernel function. It is 3 by default.
Gamma : Kernel coefficient for ‘rbf’, ‘poly’ and ‘sigmoid.

I picked this model because it gave me the best accuracy. Also, default parameter gave me pretty good accuracy, I did not change it. At the same time this model took the longest time train my data sets. Also, in general, SVM is not appropriate when datasets are noisier with overlapping classes

Complexity

	KNN	Perceptron	Random Forest	Logis. Regression	SVM
Training	X	O(knp)	$O(n^2\sqrt{p}\, n_{trees})$	$O(np^2 + p^3)$	$O(n^2p + n^3)$
Prediction	O(np)	O(p)	$O(pn_{trees})$	O(p)	$O(n_{vector} * p)$

“Computational Complexity of Machine Learning Algorithms.” The Kernel Trip, https://www.thekerneltrip.com/machine/learning/computational-complexity-learning-algorithms/.

Methodology

Our goal is to create a model which can distinguish positive and negative sentiments from movie reviews. Word2vec was used in order to create our datasets. It converts a group of words into a number where each dimension share common contextual meaning in close proximity to one another in the space. It has been implemented by a team of researchers at Google, and widely used for semantic analysis.

Given train data file, different binary classification models were implemented and tested. Data files consist of two files, ‘train_data.txt', and 'test_data.txt’. For each file, the first column contains the feature vector either 1 or 0. For 2nd to 101th column, feature number ranging about -10~+10 is given.

Result

	KNN	Perceptron	Random Forest	Logis. Regression	SVM
PCA 0	0.7455 (117.2 ms)	0.8218 (1.1 ms)	0.8141 (17.6 ms)	0.8676 (0.6 ms)	0.8719 (251.9 ms)
PCA 2 (2.9ms)	0.5205 (0.1 ms)	0.7125 (0.2 ms)	0.5241 (4.2 ms)	0.5247 (0.05 ms)	0.5261 (60.3 ms)
PCA 10 (3.0ms)	0.5712 (2.7 ms)	0.5747 (0.2 ms)	0.5709 (8.7 ms)	0.5852 (0.05 ms)	0.5748 (55.1 ms)

Conclusion

Using PCA compression enables us to process the data faster on every algorithm. It is especially useful for SVM where it takes a lot of time without PCA. However, as it reduce some degree of information of the data, the accuracy is compromised. SVM gave me the best accuracy overall, but some other hyperparameter tuning would give me better result.