Jefferson Parker

CS677

April 25, 2022

Homework 6, Written Answers

**Question 1.** For question 1, my class labels were (8 % 3 = 2) Classes 1 (negative) and 3 (positive).

* 1. – See code for implementation of linear kernel SVM in file japarker\_hw6\_1.py. The accuracy is 0.943 and the confusion matrix is plotted below.

Chart

Description automatically generated

* 1. - See code for implementation of Gaussian kernel SVM in file japarker\_hw6\_1.py. The accuracy is 0.957 and the confusion matrix is plotted below.

Chart

Description automatically generated

* 1. - See code for implementation of polynomial kernel SVM (degree = 3) in file japarker\_hw6\_1.py. The accuracy is 0.929 and the confusion matrix is plotted below.

Chart

Description automatically generated

**Question 2.** For question 2, I chose to use Logistic Regression as the classifier.

2.1 – See code for implementation of logistic regression in file japarker\_hw6\_2.py. The accuracy is 0.929 and the confusion matrix is plotted below.

Chart

Description automatically generated

2.2 – The table of predictor performance for SVM and logistic regression is shown below. In general, all the models worked relatively well with accuracy greater than 90%, but the Gassian SVM was better overall. Interestingly both the Polynomial SVM and Logistic Regression had the same accuracy but slightly different TPR and TNR.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Model | TP | FP | TN | FN | Accuracy | TPR | TNR |
| Linear SVM | 35 | 3 | 31 | 1 | 0.943 | 0.972 | 0.912 |
| Gaussian SVM | 35 | 2 | 32 | 1 | 0.957 | 0.972 | 0.941 |
| polynomial SVM | 32 | 1 | 33 | 4 | 0.929 | 0.889 | 0.971 |
| Logistic Regression | 35 | 4 | 30 | 1 | 0.929 | 0.972 | 0.882 |

**Question 3.**

3.1 – See code for implementation of kMeans with k = 1 – 8 in file japarker\_hw6\_3.py. The distortion / inertia vs k is plotted below:

Chart, line chart

Description automatically generated

Using the ‘knee’ method I chose k = 4 as my best value for k.

3.2 – See code in file japarker\_hw6\_3.py for random selection and plotting of clusters with k\* = 4 for two features (compactness and kernel\_length). In the plot below, centroids are marked with red X’s for visibility.

Chart, scatter chart

Description automatically generated

The data looks like it *could be* following an exponential curve, but that would be a stretch of the imagination. There is no obvious relationship between these variables on visual inspection.

3.3 – Each cluster was assigned a class label based on the majority of data points mapped to the cluster with centroids:

|  |  |  |
| --- | --- | --- |
| **compactness** | **kernel\_length** | **label** |
| 0.851077 | 5.222493 | 3 |
| 0.88709 | 6.268854 | 2 |
| 0.878872 | 5.483828 | 1 |
| 0.878265 | 5.863968 | 2 |

3.4 – Using Euclidean distance to map all data points in the file to the three largest clusters, the accuracy of this predictor was only 0.057 with the confusion matrix shown below:

Chart

Description automatically generated

3.5 – Applying the classifier of question 3.4, to observations with actual or predicted classes of 1 or 3 gave an accuracy of **zero** and a confusion matrix as shown below:

Chart

Description automatically generated

The new classifier is not comparable to any of the previous classifiers, which had at least some correct predictions. That said, when filtering the data set to only those data points with the included class labels the majority of the data was removed, not leaving much for evaluation.

My suspicion is that the chosen features were poorly suited for prediction with as the centroids for three of the clusters were nearly vertically aligned (see scatterplot in question 3.2), this effectively pushed the classification problem onto the other feature.