STAT 639

DATA MINING & ANALYSIS

Project Report

Aarthi Vasudevan – 532007505

Akruthi Srikanth – 332008804

Sravya Gopireddy – 433003798

Task-1: Supervised Learning

For this task, 6 classifiers namely, Logistic Regression, Naive Bayes, K-Nearest Neighbours, Random Forest, SVM, and Extreme gradient boosting (XG Boost) are trained. XGBoost is an advanced implementation of the gradient boosting algorithm, it uses a combination of decision trees and gradient boosting to achieve high accuracy and reduce overfitting. We trained classifiers using scaled and normal data. We defined grids for possible tuning parameters for each classifier and used cross-validation with 5 folds to estimate the test error for each parameter. Then we chose the optimal classifier as the one that gave the least test error.

We implemented our own k-fold cross-validation code that does the following:

1. Divide the data from (x,y) into k folds (k=5 in this case)
2. Start a for loop from 1 to k
3. In each step, construct a test set by picking one-fold
4. Construct training sets using the rest of the k-1 folds
5. Train the classifier using k-1 folds data and Predict response using the left out kth fold
6. Estimate the test error
7. Follow a similar process till the for loop ends and compute the mean of all the test errors

The test error we choose is the mean misclassification rate. It is the mean of the number of samples that the classifier misclassified.

Error= $\sum$(Predicted !=Actual)/(Length(Actual))

We factorized the actual response variable y wherever required. The predicted values for some classifiers were in the [0,1] range, we used a threshold of 0.5 to convert them into classes.

For logistic regression, we used a regularization penalty with varying alphas from 0 to 1 in steps of 0.05. No tuning parameters were used for Naive Bayes. For KNN, we used a grid of neighbors from 1 to 50. For random forest, we used 1000 trees and used a grid of mtry (the number of variables at each split), starting from the best recommendation of $\sqrt{500}$ then $\sqrt{500} + 50$, and from 100 to 500 with gaps of 50, since 500 is the number of input features for this dataset. For SVM, we constructed a grid using linear, polynomial, and radial kernels with a cost of 10^{-1} to 10^{5} and gamma from 10^{-1} to 10^{2}. For polynomial, we included degrees from 1 to 10. For XG boost, we considered 500 trees, grid for a fraction of columns to be randomly sampled for each tree as $\sqrt{500}/400$,.25,.5,1 and minimum loss reduction as 0,0.1,0.5,1 and the maximum depth of each tree as 5,7,9 and learning rate as 10^{-4} to 10^{-1} and finally binary logistic loss as the objective.

Here are the results from all the classifiers we trained:

Model	**Error**	**Optimal Parameters**
Logistic Regression	0.37	alpha=0.05
<u>KNN</u>	**<u>0.2825</u>**	**<u>k = 6</u>**
Naive Bayes	0.3725	NA
Random Forest	0.3175	mtry=100
SVM	0.2975	cost=0.1,gamma=0.1,kernel=poly,degree=4
<u>XGBoost</u>	**<u>0.2825</u>**	**<u>colsamp=0.25, gamma=0, nrounds=500, max_depth=9, eta=0.01</u>**

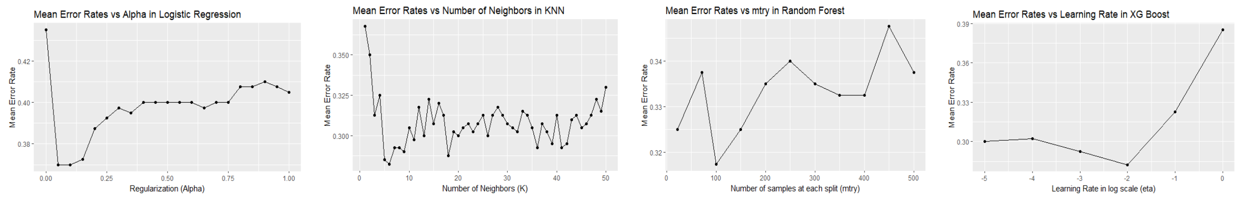

Figure: Error rates vs tuning parameters for supervised learning models

As we can see from the table here, KNN and XG Boost have resulted in the best performance out of all. The optimal parameters that resulted in this estimated error were also mentioned above.

From this, our best-estimated test error is 28.25% from KNN and XG Boost.

After looking at values from each iteration of cross-validation, we decided to go ahead with the XG Boost as our best classifier since it had similar error rates in each iteration. Also, XG Boost is an ensemble algorithm that combines weak decision trees to create a stronger model, so ideally this should perform better classification for our dataset since it has high dimensionality. We trained the new XG Boost model with the above optimal parameters and used it to predict values for the new test dataset.

Task-2: Unsupervised Learning

For the Unsupervised learning task, the methods namely Principal Component Analysis (PCA), K-Means clustering, t-Distributed Stochastic Neighbor Embedding (t-SNE), and Agglomerative Clustering.

Principal Component Analysis (PCA) is performed, and the Scree plot is obtained. From the scree plot, it is observed that the optimal number of principal components is **6**. Since the number of principal components is proportional to the number of clusters, the optimal number of clusters obtained using PCA is **6.**

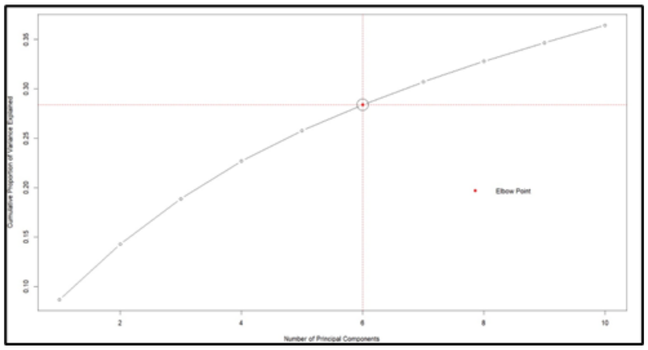

Figure: Scree Plot

Now, the other methods namely K-Means clustering, t-Distributed Stochastic Neighbor Embedding (t-SNE), and Agglomerative Clustering are evaluated using Silhouette score and Within Cluster sum of squares (WCSS) to determine the optimal number of clusters.

Silhouette Score: It is a measure of how well-defined the clusters are in a clustering analysis, based on both how similar each point is to its own cluster and how dissimilar it is to other clusters. It takes on values between -1 and 1, where a value of 1 indicates that the clustering is highly consistent and a value of -1 indicates that the clustering is highly inconsistent.

Within Cluster sum of squares (WCSS): It is a measure of the compactness of the clusters in a clustering analysis. It measures the sum of the squared distances between each data point and the centroid of the cluster to which it belongs. The goal of clustering is to minimize the WSS. When the WSS is small, it indicates that the data points within each cluster are tightly clustered around the centroid, and hence the clusters are well-defined.

Depending on the minimum value of WCSS and maximum value of Silhouette Score, the optimal number of clusters is identified as **6** for the methods K-Means clustering and t-Distributed

Stochastic Neighbor Embedding (t-SNE), and **8** for Agglomerative Clustering. The results are as follows:

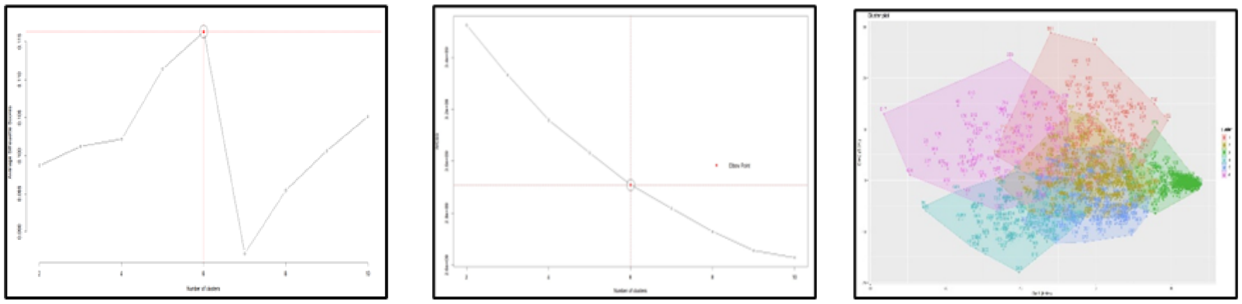

Figure: K-Means - Optimal number of clusters based on Silhouette Score, WCSS, Cluster Plot

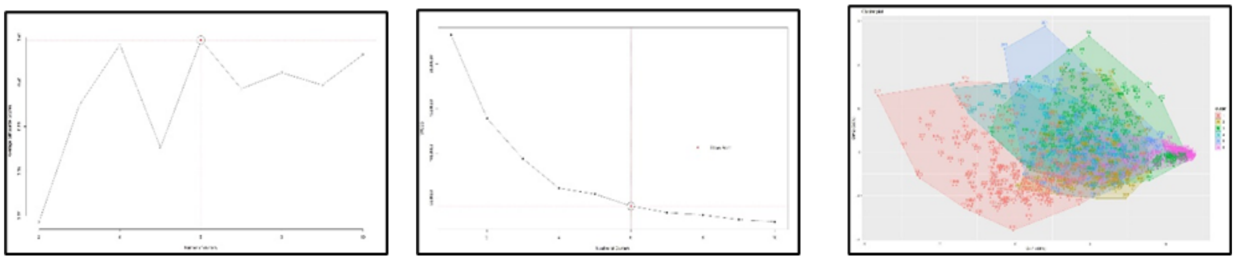

Figure: t-SNE - Optimal number of clusters based on Silhouette Score, WCSS, Cluster Plot

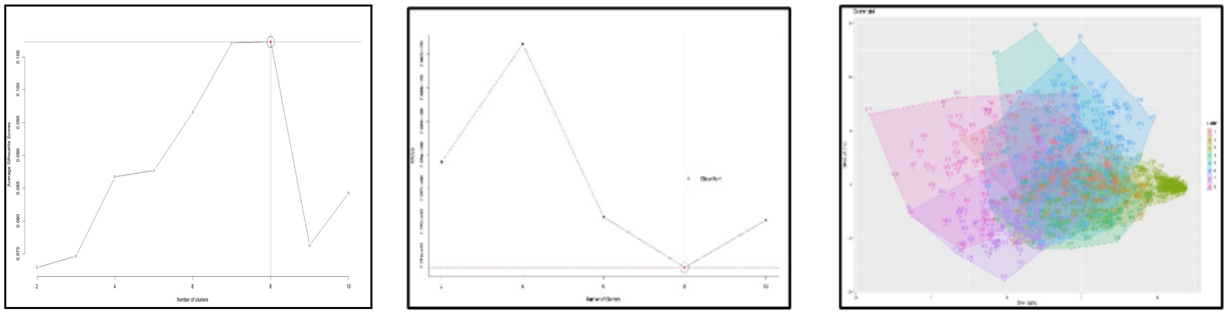

Figure: Agglomerative Clustering - Optimal number of clusters based on Silhouette Score, WCSS, Cluster Plot