LOGISTIC REGRESSION AND KERNEL PCA

ABSTRACT

PURPOSE: The purpose of this experiment is to explore kernel principal components analysis and hyperplane separation by logistic regression and the perceptron rule of artificial neurons to observe their effectiveness.

METHODS: Methods included two main functions. a2q1 produced reduced dimensionality data which was augmented to find the artificial neuron and logistic regression model in separate functions. Kernel PCA was implemented in a2q2 which found clustering of data and a k-means algorithm found clustering by another method. 6 plots were produced in MATLAB code.

RESULTS: Results show two plots with the ROC curves for both the artificial neuron and the logistic regression model. Two graphs show the reduced data grouped by label with the hyperplane generated from the two different methods. Two graphs show the clustered Fisher Iris data to show the clustering from kernel PCA and kmeans clustering

CONCLUSIONS: Results showed that the logistic regression model had a hyperplane that was able to better classify data than the perceptron rule. Kernel PCA using a gaussian Gram matrix accurately clustered data.

INTRODUCTION

The objective of this assignment is to determine how well an artificial neuron as well as logistic regression can learn how to classify if a private institution is private or public from a set of high dimensional data. Further, we want to determine how well we can classify Fisher's Iris data using kernel PCA.

The perceptron rule for machine learning is one method of generating an artificial neuron that can create a separating hyperplane. In this algorithm, the hyperplane is adjusted if classes do not match their labels. Iteratively, the weight vector of the hyperplane is adjusted if there are false positives or false negatives produced by the weight vector, which are found from computing an error vector from a label y_i and a quantization q_i as $e_i = y_i - q_i$. The weight vector $\hat{w}$ is updated as $\hat{w} = \hat{w} + e_i\hat{x}_i$. Once the training data vectors $\hat{x}_i$ have been correctly separated by the hyperplane, then the perceptron rule has converged. Data can also be augmented to use in batch learning along with a label vector to find a quantization vector. This works on all observations simultaneously in iterations to produce the weight vector which the hyperplane represents.

The logistic function is another method of classifying data into labels by using probability and odds. The logistic function shows the probability that a vector has a classification of 1 by using the hyperplane along with a bias scalar, which performs a linear summation to a probability representing the activation function. This sigmoid activation function for artificial neurons is written for observations and a weight vector as $\phi_i = \frac{1}{1+e^{-\hat{x}_i^T\hat{w}}}$. Logistic regression uses this sigmoid function for classifications of observations by optimizing the accumulation of the residual errors and finds the likelihood that the activation function results match the labels. The steepest descent optimization method is a modification of the perceptron rule that uses a learning rate and the derivative of the logistic function to find an optimal weight vector incorporating uncertainty in errors.

Previously, principal components analysis (PCA) uses a scatter matrix to dimensionally reduce the data so it can be easily interpreted and described. A centering matrix G_m can be used as another method of rewriting the scatter matrix, which is also symmetric positive semidefinite. This is an observation style scatter matrix S_U which is written from the original data matrix A and the centering matrix G_m as $S_U = G_m[AA^T]G_M^T$. To then perform kernel PCA, we use the observational style scatter matrix along with a Gram matrix, defined using a kernel function as $\widehat{W}_{ij} = \kappa(a_i, a_j)$, and use the centering matrix to find the observational style scatter matrix $\hat{S}_U = G_m W G_m^T$. The gaussian kernel function is used in this assignment. Using this scatter matrix we can then compute the significant eigenvectors and the kernel PCA score vectors to classify the data.

A testable hypothesis for this study would be to implement both the perceptron rule and logistic regression in MATLAB, finding the ROC curves in which we can then observe the accuracies and area under the curve to analyze the performance of the hyperplane. Kernel PCA can be implemented on Fisher's Iris data and can be compared to a kmeans clustering algorithm to see how well it was able to perform.

METHODS

In this study, we implement the perceptron rule for machine learning to generate an artificial neuron as well as kernel PCA and k-means clustering. To implement the an evaluation process for separating hyperplanes of labelled data, first we need to load the data from the csv file, standardize and extract the labels. The data labels correspond to whether a college is a private institution or public. From there we can then call the functions to complete the main goals for the assignment.

In our first task we are trying to compare the accuracy of the perceptron rule separating hyperplane with the logistic regression hyperplane. The data is augmented as Xmat. The perceptron rule hyperplane is calculated in the sepbinary function, in which the algorithm is used to linearly separate training vectors. A specified maximum number of iterations is used to create an optimal hyperplane using the augmented data vector. An error vector rvec and an estimate vector v_est are initialized and updated through each iteration of the loop. The estimate of the hyperplane is calculated using the heaviside function with Xmat and v_est, the errors are then generated in rvec. v_est is then updated and the loop continues to iterate if there is still data that is mis-classified using a variable called missed. After completion of the algorithm the v_final is set to equal v_est and returned from the function. The logreg function provided in the starter code is used to calculate the logistic regression estimate of the hyperplane.

Data is scored by projection of the hyperplane to the augmented data. Using these scores, an ROC curve with the area under the curve is generated using the perfcurve function along with the labels. The accuracies of the hyperplanes are stored in the variables acc_ann and acc_log. Positive and negative occurences of the scores are found and stored into logical arrays q_ann and q_log that can then be used to compute an error vector that finds which labels are mis-classified by subtracting yvec from q_ann and q_log. True positives and true negatives are then found by summation of all occurrences of zeros which can then be used to find the accuracies. Areas under the ROC curves and accuracies of the separating hyperplanes are then displayed along with ROC curve and hyperplane plots using plot and gscatter functions.

Kernel principal components analysis on Fisher's Iris data is implemented in the a5q2 function. Data is reduced down to two dimensions by pca with spectral decomposition by first using the gaussian centered Gram matrix which is computed in the gramgauss function. Within the gramgauss function, the matrix Kmat is computed using a nested for loop which iterates through each entry of Xmat and updates each entry of Kmat with the gaussian exponential function applied to each data entry X(i, j) in Xmat as exp(-1/sigma2*norm(Xmat(i, :) - Xmat(j, :))^2) , with sigma2 as a scalar passed through the function. Spectral decomposition is applied to Kmat, which is then used to sort the eigenvectors in descending order to find the most significant eigenvectors. These two eigenvectors are then used as the projection of the gram matrix to two dimensions, and the data is grouped using the gscatter function by the actual labels in one figure along with the labels generated from the kmeans clustering algorithm. This produces a visual representation of the grouping of data from kmeans

RESULTS

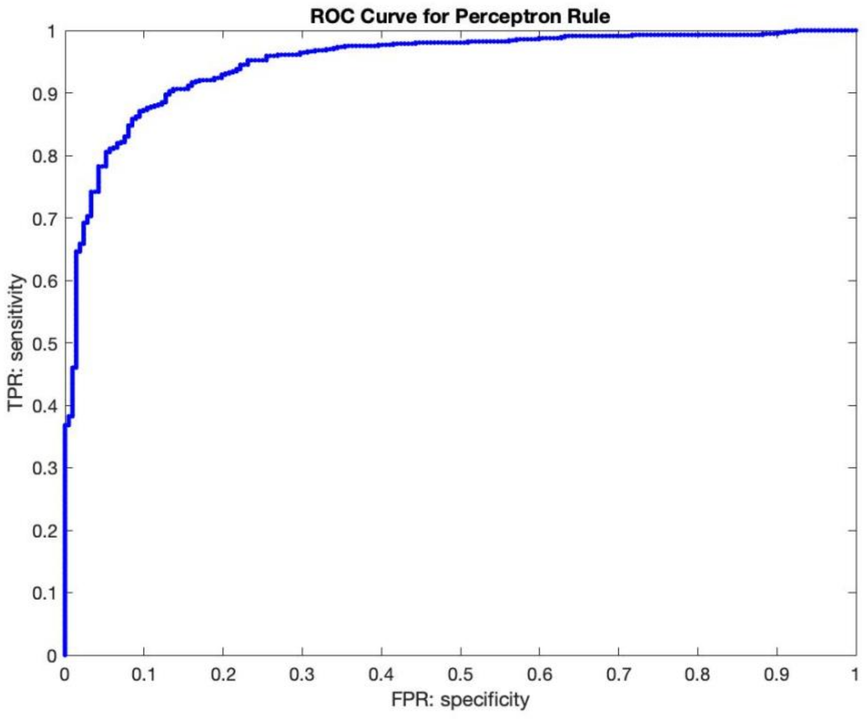

Figure 1.
Shows the ROC Curve for the perceptron Rule. Area under the curve for this ROC curve is calculated to be AUC = 0.9495.

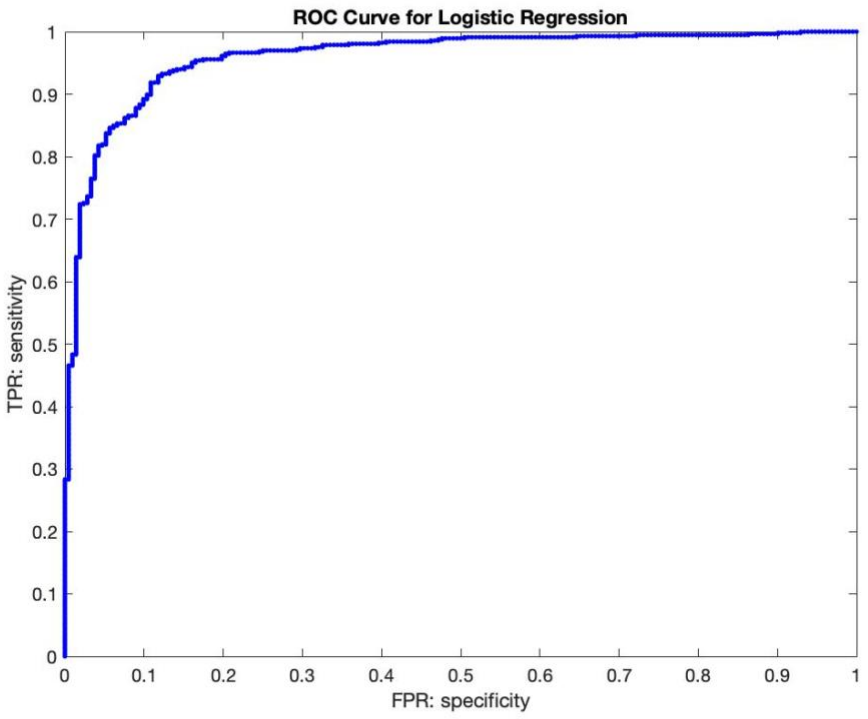

Figure 2. Shows the ROC curve for Logistic Regression. Area under the curve for this ROC curve is calculated to be AUC = 0.9600.

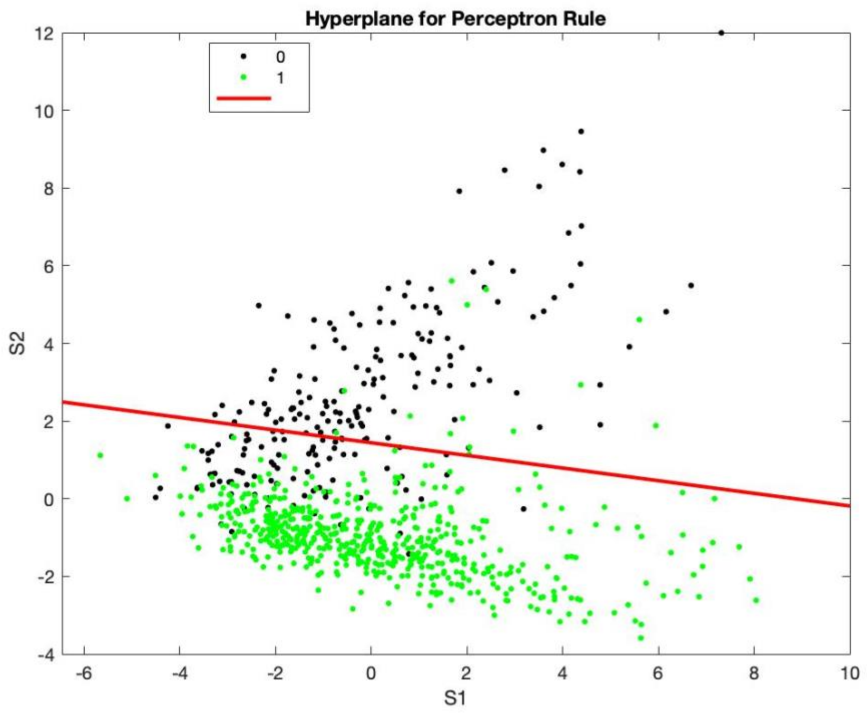

Figure 3. Shows the data reduced to two dimensions with the separating hyperplane generated by the perceptron rule. Label 0 represents a public college and label 1 represents a Private college. Accuracy = 0.8764

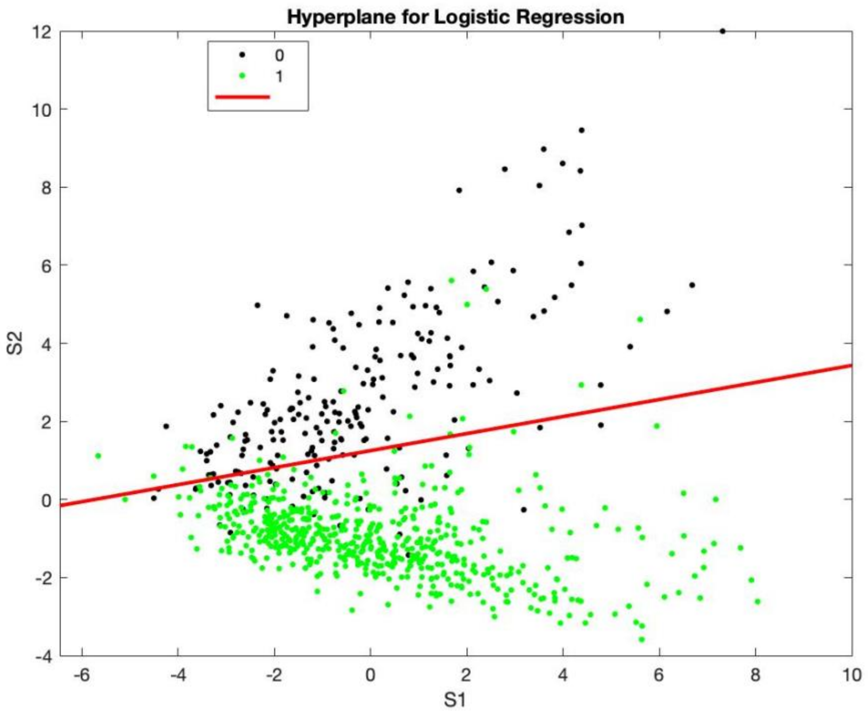

Figure 4. Shows the data reduced to two dimensions with the separating hyperplane generated by logistic regression. Label 0 represents a public college and label 1 represents a private college. Accuracy = 0.9112

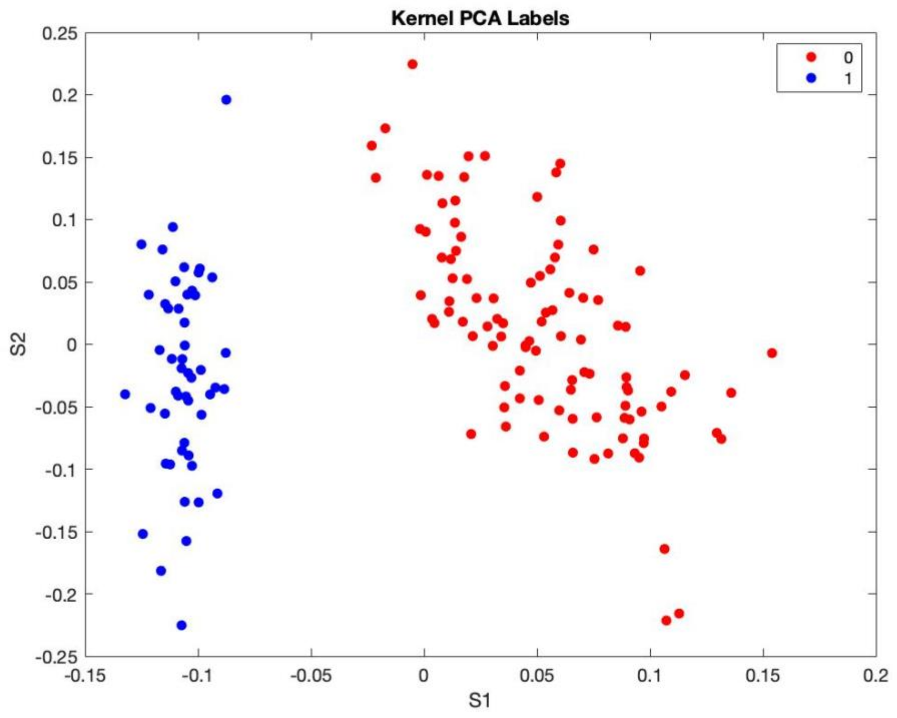

Figure 5. Kernel Principal component analysis using the spectral decomposition of the Gram matrix projected to two dimensions. Data is grouped by yvec labels.

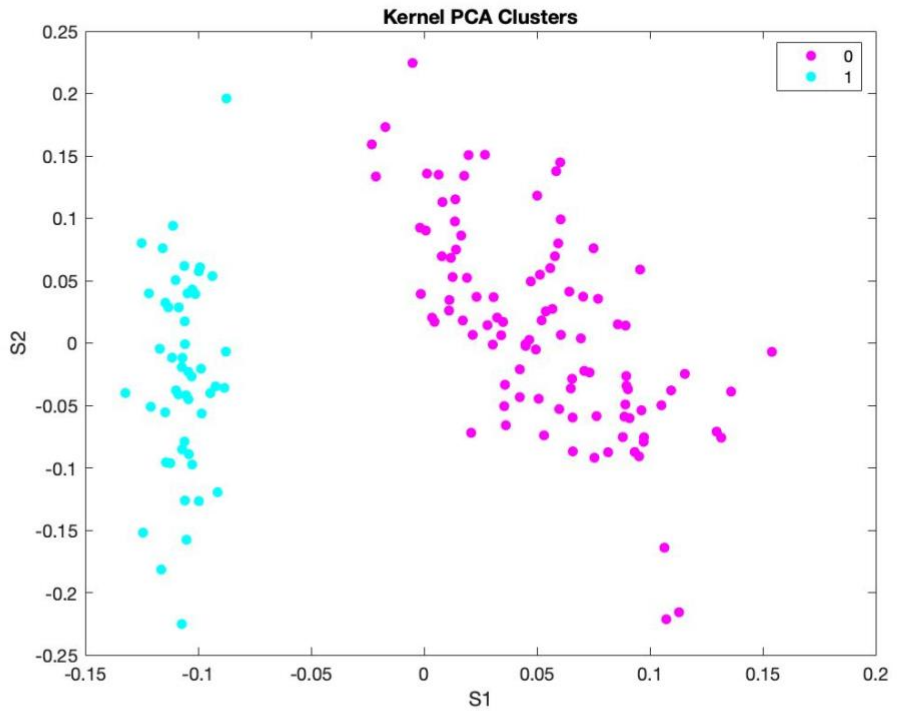

Figure 6. Kernel principal component analysis using the spectral decomposition of the Gram matrix projected to two dimensions. Data is grouped by the kmeans clustering algorithm.

Discussion

In our implementation of the hypothesis, we first produced two ROC curves in Figures 1 and 2. These curves represent the overall performance of the separating hyperplane at varying threshold values. The area under the curve for the two ROC curves of the perceptron rule and logistic regression are found to be 0.9495 and 0.9600, respectively. From these AUC's indicate that the hyperplane for logistic regression was able to perform better overall by a slight margin in comparison to the hyperplane for the artificial neuron.

Figures 3 and 4 represent the data reduced to two dimensions clustered by their respective labels. The hyperplane for the perceptron rule in figure 3 and the logistic regression model in figure 4 are shown to visually differentiate the hyperplanes. In the graph it is unclear to see how well each of the hyperplanes do individually because of the dense graph. In our methods we were able to find the accuracy of the hyperplanes by using the true positives and true negatives. The accuracy for the perceptron rule and logistic regression model are found to be 0.8764 and 0.9112, respectively. By comparing these values, this further shows that the logistic regression classifier performed better relative to the artificial neuron. A logistic regression model is more complex than the more basic perceptron rule. A logistic regression model takes into account more uncertainty in its error in comparison to the linear step function from the perceptron rule. This can explain the difference in performance from the two classifiers.

Our second question for this study required us to implement kerned PCA to determine how well it is able to cluster data. Kernel PCA is shown in figure 5. Visually, it is clear to see that there are two clusters of the reduced data. Comparing figure 5 with figure 6 shows that the kernel PCA with a Gaussian function was able to cluster the data, since the k-means algorithm is able to produce the same clustering as the gaussian function Gram matrix.

Logistic regression is extremely useful in machine learning to classify data and make further predictions. Logistic regression in conjunction with PCA is a useful tool used in the real world with large sets of data from experiments where models can accurately distinguish important variables in the data from PCA and can predict classifications.

To summarize, our implementation of an artificial neuron and logistic regression shows that logistic regression was more accurate in classifying data and has better overall performance over a range of threshold values. Moreover, kernel PCA was able to accurately cluster data to labels. Our testable hypothesis in this experiment was capable of producing the desired outcomes.

References

Ellis, R. Class 28 Classification – Perceptron Rule. Retrieved April 16, 2021, from https://onq.queensu.ca/content/enforced/505646-CISC271W21/notes/Class28.pdf

Ellis, R. Class 30 Classification – Logistic Regression. Retrieved April 16, 2021, from https://onq.queensu.ca/content/enforced/505646-CISC271W21/notes/Class30.pdf

Ellis, R. Class 32 Nonlinear Separation – Kernel PCA. Retrieved April 16, 2021, from https://onq.queensu.ca/content/enforced/505646-CISC271W21/notes/Class25.pdf