EGR 598: Machine Learning and Artificial Intelligence (Fall 2023)

Instructor: Shenghan Guo

Assignment 3

This assignment is worth 200 points. It will be due on Thursday, 11/30/2022, and submitted through Canvas. Code should be written in Python. Other programming language can be used upon instructor's approval. A written report should be submitted as a separate document along with the code through Canvas. If you use Jupyter Notebook, then you may submit a single ".ipynb" file through Canvas.

Data: A dataset "transfusion.data" is provided to you. Data description for "transfusion.data" can be found in "transfusion.names". Please read through the document and then use the data to do the following tasks.

Note: In the dataset, columns 1-4 are attributes, and column 5 is the ground-truth class. You may consider the samples as independent and identically distributed (iid).

Part 1 (55 pts)

1. Write a code to implement *online k-means algorithm* on the given dataset using $k = 2$. Report the number of iterations used for the algorithm to converge. Follow the pseudo code in Figure 12.2 of the textbook. You may use or not use stochastic gradient descent. If not using it, you can update the cluster centers as the mean of instances up to $\boldsymbol{x}_t$ that belong to that cluster. Use a threshold of 0.001 to evaluate the convergence of cluster centers. (Hint: Upon algorithm convergence, all cluster centers should change no more than the threshold.) (35 pts)

2. In your clustering result from 1, you may have "cluster 1", "cluster 2", etc. Does the cluster index necessarily represent the class (i.e., "whether donated blood" for this dataset)? Why or why not? (5 pts)

3. Compare the clustering results and the ground-truth glass types. Use a confusion matrix to summarize the accuracy of clustering. (10 pts)

4. What are the connection and difference between *online k-means* and *k-means* algorithm? (5 pts)

Part 2 (50 pts)

For this part, randomly shuffle the instances in "transfusion.data", and then take the first 400 (after shuffling) as training data and the rest as testing data.

1. Train a Naïve Bayes classifier using the training data. Validate the trained classifier with testing data. Summarize the classification performance with a confusion matrix and the prediction accuracy. (Hint: use "GaussianNB" in Python library "sklearn.naive_bayes".) (20 pts)

2. Train a support vector machine (SVM) with *radial basis kernel* using the training data. Validate the trained SVM with testing data. Summarize the classification performance with a confusion matrix and the prediction accuracy. (Hint: use "svm" in Python library "sklearn".) (20 pts)

3. Comment and compare the performance of Naïve Bayes and SVM. (5 pts)

4. Both SVM and *logistic regression* build a separating hyperplane to do the classification. What is the difference between their way of formulating the hyperplane? (5 pts)

Part 3 (50 pts)

For this part, randomly shuffle the instances in “transfusion.data”, and then take the first 400 (after shuffling) as training data and the rest as testing data.

1. Train *Bagged Classification Trees* to do classification analysis for the data set. Use $L = 500$ learners. Summarize the classification performance on testing set with a confusion matrix. (Hint: use Python library “sklearn.ensemble”.) (20 pts)

2. Train *Classification Trees with AdaBoost* to do classification analysis for the data set. Use $L = 500$ learners. Summarize the classification performance on testing set with a confusion matrix. (Hint: use Python library “sklearn.ensemble”.) (20 pts)

3. Bagged classification trees in 1 has an alternative, widely adopted method name. What is that? (5 pts)

4. Briefly explain the difference between *Bagging* and *Boosting*. (5 pts)

Part 4 (45 pts)

For this part, randomly shuffle the instances in “transfusion.data”, and then take the first 400 (after shuffling) as training data and the rest as testing data. Assume column 2 (“frequency”) in “transfusion.data” is the dependent variable and columns 1, 3, 4 are independent variables.

1. Use the training data, fit the mean vector and the covariance matrix. (5 pts)

2. Assume that the data follow a *Gaussian process*. Predict the dependent variable values for the testing data using *Gaussian process regression*. Calculate the mean squared error for the prediction. (Hint: Use Python library “sklearn.gaussian_process”. You will get a scalar for mean squared error.) (20 pts)

3. Repeat the model fitting and prediction problem in 2 using *multivariate linear regression*. Calculate the mean squared error for the prediction. (15 pts)

4. Comment on the difference between *Gaussian Process regression* and *multivariate linear regression*. (5 pts)