STA 4365 HW 2

due March 1 by 11:59PM on Webcourses

Submission Format: Problem 1 can be hand-written and submitted in any legible format. Problems 2 and 3 must be submitted in either an R Markdown file or a Jupyter Notebook.

Problem 1: (6 points) In this problem, you will solve constrained optimization problems.

(a) Solve the following problem:

$$\text{minimize } f(\theta_1, \theta_2) = (\theta_1 - 1)^2 + 2(\theta_2 - 2)^2$$
$$\text{subject to } \theta_1 + \theta_2 = 5.$$

(b) Solve the following problem:

$$\text{minimize } f(\theta_1, \theta_2) = (\theta_1 - 1)^2 + 2(\theta_2 - 2)^2$$
$$\text{subject to } \theta_1 \geq 3 \quad \text{and} \quad \theta_1 + \theta_2 \geq 5.$$

Problem 2: (7 points) In this problem you will compare the performance of a variety of classifiers that you have learned about throughout this course and the previous course. The data is in the file `magic04.data` and the column names are in the file `magic04.names`. The last column is a categorical response with values `g` or `h`, and the rest of the columns are numerical features. You can read more about the dataset here.

(a) Load the data (can use the pandas function `read_table` with the arguments `sep=','` and `header=None`). Split the data into a training and test set. Scale and center the columns using the mean and standard deviation of each column from the *training set* (make sure you use the same scaling on the test set that is used on the training set).

(b) Learn the following models to classify the training data:

- **Logistic Regression**: Can import `LogisticRegression` from `sklearn.linear_model`.
- **LDA**: Can import `LinearDiscriminantAnalysis` from `sklearn.discriminant_analysis`.
- **KNN Classifier**: Need to choose the number of neighbors k.
- **Linear SVM**: Need to choose the margin penalty C as a hyperparameter.
- **Gaussian SVM**: Need to choose the margin penalty C and the radius width γ.
- **Random Forest**: Need to choose the number of trees.
- **Gradient Boosted Decision Tree**: Need to choose the number of trees and learning rate. If you want, you can also experiment with randomly selecting rows and columns when growing each tree.

To tune hyperparameters for each model, you can either use cross-validation or hand-tune by examining the model performance for reasonable values of the hyper-parameters.

(c) Apply your models to the test set. Report the accuracy, visualize an ROC curve, and report the AUC for each model. For Logistic Regression, Random Forests, and Gradient Boosted Decision Trees, report the most meaningful predictors.

Problem 3: (7 points) In this problem, you will compare the performance of several different regression models you have learned throughout this course and the previous course. The data is in the file `qsar_aquatic_toxicity.csv`. All columns are numerical features. The last column in the response variable, and the other columns are predictors. You can read more about the dataset here.

(a) Load the data (can use the pandas function `read_csv` with the arguments `sep=';'` and `header=None`). Split the data into a training and test set. Scale and center the columns using the mean and standard deviation of each column from the *training set* (make sure you use the same scaling on the test set that is used on the training set).

(b) Learn the following models to classify the training data:

- **Linear Regression**: Can import `LinearRegression` from `sklearn.linear_model`.
- **KNN Regression**: Need to tune the number of neighbors k.
- **Random Forest**: Need to choose the number of trees.
- **Gradient Boosted Decision Tree**: Need to choose the number of trees and learning rate. If you want, you can also experiment with randomly selecting rows and columns when growing each tree.

To tune hyperparameters for each model, you can either use cross-validation or hand-tune by examining the model performance for reasonable values of the hyper-parameters.

(c) Apply your models to the test set. Report the MSE on the test data along with the R^2 value for each model. For Linear Regression, Random Forests, and Gradient Boosted Decision Trees, report the most meaningful predictors.