MSCI:7000 – Homework 1

Due: February 16th, 2020, 11:59 PM

Guidelines for Submission

Writeup

Please include your name and your email on the first page of your writeup. Try to be as concise as possible when answering questions. Whenever you use ideas or code from other resources (papers/websites), acknowledge them. Whenever you make any assumptions or design decisions, please mention those explicitly in your writeup. Please remember - **your write-up must be your own**.

Collaboration Policy

It is perfectly fine to discuss the homework with other students. However, it is important that your code has to be done independently. Your write-up must be your own. Any violations will be taken very seriously.

Code Submission

All the code must be implemented using Python, sklearn, numpy, matplotlib, pandas. If you are not familiar with some of these, there are several online tutorials available. Please consult those before reaching out to the instructor for guidance. All the code files (.py or .ipynb) should be uploaded to ICON along with your writeup. Any submissions without the complete code will not be graded. Please include detailed comments in your code. Make sure your code runs without errors.

Notes for Implementation

If and when you are using built-in functions, please stick to our suggestions (if any) as much as possible (since results may vary with different implementations).

Please use one-hot encoding for categorical variables. The datasets you will be working with may have missing values. Python, sklearn, and related libraries have some capability to handle missing values. However, if you are running into errors and need to handle missing values explicitly, do the following: a) In case of continuous variables, impute using the mean of the corresponding attribute, and also create a new variable for the corresponding attribute - set it to 1 when the attribute value is missing for a particular data point and a 0 otherwise. b) In case of categorical variables, treat missing values as a category in itself and encode it as part of the one-hot encoding.

Problem Statement

A logistic regression model has the form $p(y = 1|x) = \frac{1}{1+e^{-w_0-wx}}$, where w_0 is an intercept, w is a vector of parameters and x is an input vector. w_0 and w need to be learned from data. $p(y = 1|x)$ represents the probability of input x to be in the positive class and $1- p(y = 1|x)$ represents the probability that x is in the negative class. To train a logistic regression model, one often maximizes the log likelihood of data, which is

$$\begin{aligned}\log \prod_{i=1}^{n} p(y_i|x_i) &= \log \prod_{i=1}^{n} p(y = 1|x)^{y_i} (1 - p(y = 1|x))^{1-y_i} \\ &= \sum_{i=1}^{n} \log p(y = 1|x)^{y_i} (1 - p(y = 1|x))^{1-y_i},\end{aligned}$$

equivalent to minimizing

$$-\sum_{i=1}^{n} y_i \cdot \log p(y = 1|x_i) - (1 - y_i) \cdot \log(1 - p(y = 1|x_i))$$

1. [40 points] Implementing Classifiers
   Please upload two separate code files to question 1.a and 1.b. Each file should contain code that read the train and test data (I will change the directory to the adult_train.csv and adult_test.csv on my computer when I test your code), process the data and train the model. The file should then print out the parameters w.

   a. [20 points] Implement a gradient descent algorithm for learning a logistic regression classifier. Show a plot of convergence (the x-axis represents the number of iterations and the y-axis represents the learning objective).

   b. [20 points] Implement a simulated annealing algorithm for learning a logistic regression classifier. You need to set the appropriate temperature and number of iterations in the algorithm. Show a plot of convergence.

2. [15 points] Now apply the above two versions of logistic regression to a census income dataset where the goal is to predict whether income exceeds $50K/yr based on census data. The dataset contains both categorical as well as continuous attributes.

   Train the model you developed above using the dataset adult_train.csv. To evaluate your model, report precision-recall curve, F1-score, ROC and AUC on the test set: adult_test.csv. Include the plots of the curves mentioned above in the report.

3. [10 points] Call the Lasso model from sklearn.linear_model (set alpha = 1) and train it using adult_train.csv.
   a. Report the ROC curve and AUC.
   b. Compare the number of non-zero coefficients in the three models you have built so far. What do you observe and why?

4. [35 points] Approximating a black-box:

a. [5 pionts] Call the RandomForestClassifier from sklearn.tree, use the default parameter setting and train it using the adult_train.csv data above.
b. [5 points] Generate the feature importance plot for the random forest model you built (built in function)
c. [15 points] Now you "explain" the random forest using one of the logistic regression model you built. This time the target variable is the prediction from the random forest, instead of the true value in the dataset.
d. [10 points] Compare the "w" in the logistic regression model and the feature importance plot from 4.b. Are the insights consistent?