Department of Computer Engineering
University of Peradeniya

CO 544 Machine Learning and Data Mining

Lab 02

5th of July 2021

1. Objective

Getting hands-on experience on linear models using Python.

2. Linear Regression

(a) Simple Linear regression

Simple linear regression involves single independent variable. The line which fits best to the given data points is called the 'Regression Line'. The equation of the regression line as follows:

$$y = \beta_0 + \beta_1 x$$

Where, y - predictive variable (dependent) , x - independent variable
β_0, β_1 - regression coefficients

In order to determine the regression line first determine the regression coefficients. Here 'Least Square Method' can be used to determine the regression coefficients.

$$\hat{\beta}_1 = \frac{\sum(x_i – y_i) - (n\bar{x}\bar{y})}{\sum(x_i)^2 – n\bar{x}} \tag{1}$$

$$\hat{\beta}_0 = \bar{Y} – \hat{\beta}_1\bar{X} \tag{2}$$

Where $\bar{x}$ and $\bar{y}$- mean values of y and x variables & n - number of observations

Now we can implement this in python using Numpy.

```
import numpy as np     # importing numpy module
import matplotlib.pyplot as plt     # importing matplotlib modules to plot

x= np.array([1,2,3,4,5,6,7,8,9,10])    # array of independent variable values
y= np.array([2,5,7,8,9,11,14,15,17,19])    # array of dependent variable values

n= np.size(x)     # get the number of observations

m_x= np.mean(x)     # determining the mean values of variables
m_y= np.mean(y)

SS_xy = np.sum(yx) - n m_y m_x      # to find b_1 estimator value
SS_xx = np.sum(xx) - n m_x m_x
```

```
b_1 = SS_xy/SS_xx      # determining the parameter values
b_0 = m_y - b_1 m_x

plt.scatter(x, y,color = "b", marker = "*",s = 60)    # plotting a scatter plot
plt.title('Simple Linear Regression')    # adding a title to the graph
plt.xlabel('Independent Variable') # adding axis labels
plt.ylabel('Dependent Variable')

y pred = b_0 + b_1 x      # predicting response variable values

plt.plot(x, y pred, color = "r")    # plotting the predicted line
plt.show()     # displaying the plot
```

(b) TODO 01

i. Import the 'Boston Housing.csv' data set. (This is a data set from Kaggle open datasets)

ii. Split 80% of data as the training set and rest as the test set.

iii. Divide the data into feature matrix (x) and response vector (y).
Features: RM, LSTAT, PTRATIO Response: MEDV.

iv. Use following expressions to estimate the parameter values:

$$\hat{\beta} = (X^{\intercal}X)^{-1}X^{\intercal}y \tag{3}$$

where,$y=\begin{pmatrix} y_1 \\ . \\ . \\ y_n \end{pmatrix}$, $\hat{\beta}=\begin{pmatrix} \beta_0 \\ . \\ . \\ \beta_p \end{pmatrix}$, X=$\begin{pmatrix} 1 & x_{11} & . & x_{1p} \\ . & . & . & . \\ 1 & x_{n1} & . & x_{np} \end{pmatrix}$

v. Predict the response values for the training and test set using the derived regression model.

$$\hat{y} = X\hat{\beta}$$

vi. Visualize the residual errors for both train and test sets in one graph.
(x axis- predicted value, y-axis- actual value)

Residual error=$y_i - \hat{y}$

3. Logistic Regression

Logistic regression is a classification algorithm. In a classification problem, the response variable y, can take only discrete values for given set of features x.

scikit-learn is a python module with simple and efficient tools for predictive analysis. In sklearn, all machine learning models are implemented as Python classes.

It also includes some standard data sets for use. The wine data set is one such standard data set of 13 features(key name-'data') and a target variable(key name- 'target') with 3 classes.

```
#Import standard data sets
from sklearn import datasets

#Import the Logistic regression model
from sklearn.linear_model import LogisticRegression
```

```
#Split data set into a train and test set
from sklearn.model_selection import train_test_split

wine_dataset =datasets.load_wine() #loading a data set from scikitlearn
x=wine_dataset["data"]#defining variables
y=wine_dataset["target"]#defining target variable values

#Splitting data set into a train and test set with 70% and 30%
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.3, random_state = 1)

log_reg = LogisticRegression() #calling logistic regression model
log_reg.fit(x_train,y_train) #fitting the model to train set

# predicting y values of test set
predictions = log_reg.predict(x_test)

print(accuracy_score(y_test, predictions))
```

(a) TODO 02

i. Load the 'digits' data set from the scikit-learn standard data sets.

ii. Split 80% of the data set to train and rest for the test set

iii. Train a Logistic regression model and predict values for the test set.

iv. Visualize the residual errors for both train and test sets in one graph. (x axis- predicted value, y-axis- actual value)

4. Submission

(a) In Lab Submission
Please submit whatever you have done within the lab. Rename your file as 16xxxlab02np.txt and 16xxxlab02pd.txt where xxx is your registration number.

(b) Complete Submission
Submit two text files for two sections. Your files should contain all your commands you have tried out in the lab with the answers for TODO sections (as comments in the text) and Try Outs .Rename it as 16xxxlab02np.txt and 16xxxlab02pd.txt where xxx is your registration number.

5. Deadline

The deadline: June 12, 2021, by 11.59 p.m
If you do not understand any concepts, make sure you get some help from instructors.