PMDS505P Data Mining and Machine Learning
Experiment 2

January 2025

1 Work to do today

Note: Make a single pdf file of the work you are doing in jupyter notebook and upload with proper format.

Linear Regression

Today we will implement the linear regression technique to fit a simple model in connection with the dataset "Training_set_heights200" available for you to download in moodle.

- Download the dataset 'Training_set_heights200.csv'.
- import numpy, pandas and matplotlib to your program and load your dataset to a dataframe.
  import pandas as pd
  import numpy as np
  from sklearn.metrics import mean_squared_error
  import matplotlib.pyplot as plt
  data = pd.read_csv("Training_set_heights200.csv")
- The objective is to fit the given data to a straight line $y = mx + n$ where the variable x is the Height component and y is the Weight component from the dataset you have imported.
- Use MinMaxScaler() to scale the data to 0 to 1 range.
  from sklearn import preprocessing
  MM = preprocessing.MinMaxScaler()
  x = MM.fit_transform(data)
  print(x)
- The data should be loaded to the X and Y variables.
  X = x[:,0]
  Y = x[:,1]

- import Linearregression to your program from sklearn using the command

  from sklearn.linear_model import LinearRegression

- Create an object of the class LinearRegression as **reg** as given below
  reg = LinearRegression()

- Fit the data using reg.fit command as given below,
  reg.fit(X,Y)

- Now you can find your parameters m and n by printing **reg.intercept_** and **reg.coef_.**

- Next we will try to implement the same problem using a machine learning technique.

- Now using train_test_split function split the dataset into 70:30 split with 30 for testing.
  from sklearn.model_selection import train_test_split
  X_train,X_test,y_train,y_test = train_test_split(x[:,0],x[:,1],test_size=0.30, random_state=0)

- Write the code for gradient decent algorithm to find the values of m and n based on the data. As a part of that initialize the values of m and n as 0 and then define a function **gd()** which takes input as a data set, m and n values and learning rate L and returns new updated values of m and n.

  Do 900 iterations of the gradient_descent() in a for loop and print the latest value of m and n.

```
def gd(xt,yt,m_b,n_b,L):
    D_m=0
    D_n=0
    for i in range(len(xt)):
        D_m = D_m+ (2/len(xt))*((m_b*xt[i]+n_b -yt[i])*xt[i])
        D_n = D_n+ (2/len(xt))*(m_b*xt[i]+n_b -yt[i])
    m_b = m_b - L*D_m
    n_b = n_b - L*D_n
    return m_b,n_b
m = 0
n = 0
L = 0.5
epochs = 900
for i in range(epochs):
    m,n = gd(X_train,y_train,m,n,L)
    print(m,n)
```

Figure 1: Enter Caption

- Change the learning rate to be a bigger value and check whether your algorithm is converging to a valid values for the parameters m and n.

- predict the new *y_pred* value for the values of x and plot the scatter plot of the initial data and line which is being fitted to the data.
  plt.scatter(X_train,y_train)
  y_pred = m*X_train+n
  plt.plot(X_train,y_pred)

- Verify whether the value of m and n are near to actual values.

- Now use the testing data and predict the mean square error for the testing set using this model.
  from sklearn.metrics import mean_squared_error
  y_test_pred = m * X_test + n
  testing_error = mean_squared_error(y_test, y_test_pred)
  print("Testing Mean Squared Error:", testing_error)

• Now create a linear regression model using gradient decsent technique to predict height in terms of weight. Choose suitable value of learning rate and also modify your program so that your training terminates when the parameter values are the same for 3 decimal places in consecutive iterations in the gradient descent function you are writing. Then **Multiple Regression**

- Download the dataset 'Book1.csv' from moodle. This dataset has information regarding the house price data based on more than one feature. Open the CSV file and see the different features and the target variable Y (house price) also.

- Load the dataset to a dataframe.

- Drop the furnishing status column and then we will be left out with 5 features $(X_1, X_2, X_3, X_4, X_5)$ to predict the house price (Y)

- We are looking for a model $Y = a_0 + a_1(X_1) + a_2(X_2) + ... + a_5X_5$.

- USe MinMaxScaler() to scale the data to 0 to 1 range.

- Do the traintestsplit of your data.

- Write an appropriate gradient descent algorithm and determine the values of the parameters involved in the prediction function using the testing data and print them.

- Also use the inbuild LinearRegression class and create an object of this class and fit the model and check for the values of the parameters of your model.

- Also print the testing error for the model you have created using the test set you have.