CS7052-Machine Learning

Workshop 5: Linear Regression and Gradient Descent

You will learn:

- To practice Linear models for Regression and Gradient Descent with several datasets

Open your Jupyter notebook and follow the instructions for completing tasks 1-3.

Make sure you understand the meaning of each line of code, make some changes to improve your understanding.

Task 1- Linear Regression for Diabetes dataset

The first task is to load the diabetes dataset, select one feature (bmi, the 2nd indexed feature), and split it into a training set (90%) and a test set (10%). Plot the data as a scatter plot as well as the linear Regression using *linear_model.LinearRegression()* function.

```
import numpy as np

import matplotlib.pyplot as plt

from sklearn import datasets, linear_model

from sklearn.metrics import mean_squared_error, r2_score

# Load the diabetes dataset

diabetes = datasets.load_diabetes()

# Select 10% for testing, 90% for training

data_len = len(diabetes.target)

nTestSamples = np.int32(0.1*data_len)

idx_test = np.arange(1, nTestSamples)

idx_train = np.arange(idx_test[-1]+1, data_len - idx_test[-1])


# extract the bmi feature

print(diabetes.feature_names[2])
```

```
X_diabetes = diabetes.data[:,np.newaxis,2]

X_test = X_diabetes[idx_test]

X_train = X_diabetes[idx_train]

y_test = diabetes.target[idx_test]

y_train = diabetes.target[idx_train]

# (For comparison purposes, create linear regression object)

regr = linear_model.LinearRegression()

# Train the model using the training sets

regr.fit(X_train, y_train)

# Make predictions using the testing set

y_pred = regr.predict(X_test)

# The coefficients

print('Intercept (b): \n', regr.intercept_)

print('Coefficients (w0): \n', regr.coef_)

# The mean squared error

print("Mean squared error: %.2f" % mean_squared_error(y_test, y_pred))

# Plot outputs

plt.scatter(X_test, y_test,  color='black')

plt.plot(X_test, y_pred, color='blue', linewidth=3)

plt.show()
```

Task 2- Writing a simple loss function

Run this code in your Jupyter Notebook, what is the purpose of this piece of code[1]?

```
import numpy as np
X = np.array ([4, 5, 6])
Y = np.array ([4, 5, 6 ])
def loss_func (W, X , y):
    l = 0
    for x, y in zip (X, Y):
        l += (W*x - y) ** 2
    return l / len (X)
loss_hist = []
for feed_W in np.linspace (-3,5 , num =15):
    curr_loss = loss_func (feed_W , X , Y)
    loss_hist.append (curr_loss)
    print ("{:6.3f} | {:10.5f}". format (feed_W, curr_loss))
```

[1] This is an adaptation of the code here: https://goodboychan.github.io/python/tensorflow/machine_learning/2020/09/07/02-Cost-Minimization-using-Gradient-Descent.html#Hypothesis-and-cost-function

Task 3 – Calculating gradient descent for linear regression

Write a piece of code to calculate gradient descent of a loss function

Remember, in order to calculate gradient descent, you start with initial values of [0,0] for slope and intercept (b) and then in a loop calculate loss function and calculate the gradient descent update, stop the loop when loss falls stop changing. Initialise the learning rate to 0.1

There are many online resources for inspirations. I list a few of them:

1. https://towardsdatascience.com/gradient-descent-in-python-a0d07285742f
2. https://towardsdatascience.com/implement-gradient-descent-in-python-9b93ed7108d1
3. https://medium.com/analytics-vidhya/gradient-descent-in-3-cells-f308a7c0f0bf
4. https://realpython.com/gradient-descent-algorithm-python

Show the output to your tutor when you are done.