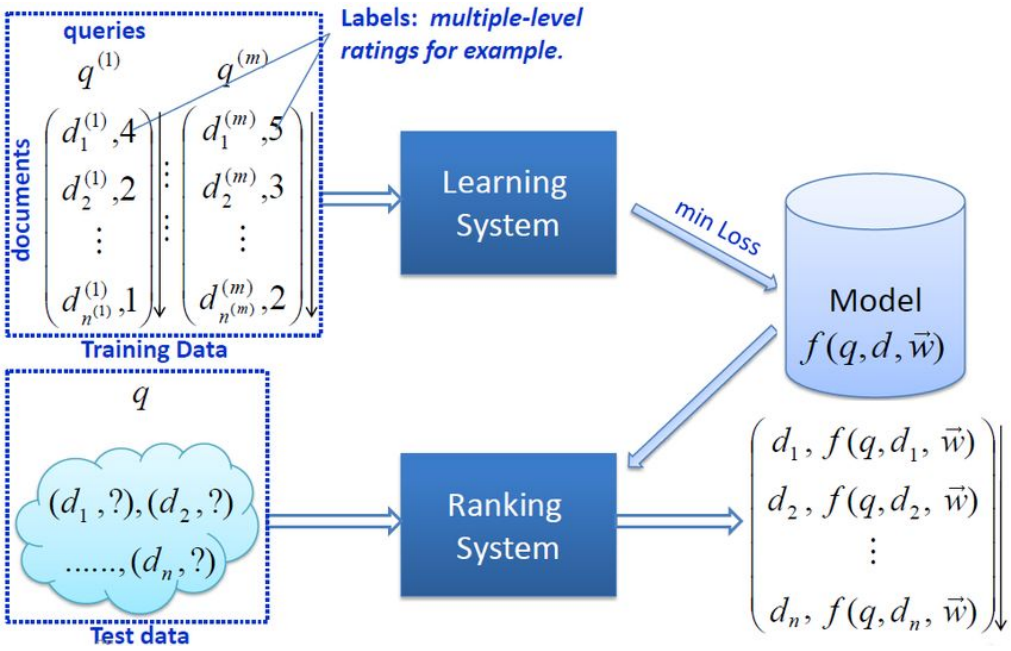
CSE 574 LEC001

Project 1.2

Learning to rank using linear regression



Malini Anbazhagan

Person# 50289383

University at Buffalo

Buffalo, NY 14260

Contents

Introduction

The goal of this project is to use machine learning to solve a problem that arises in Information Retrieval, one known as the Learning to Rank (LeToR) problem.

There are two methods used to train a linear regression model. Closed form solutions and Stochastic gradient descent.

Linear Regression Model:

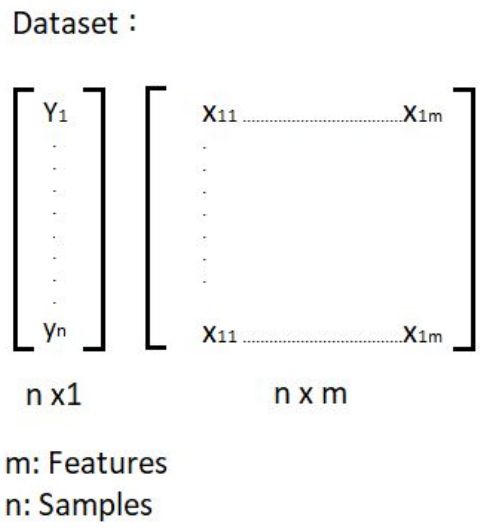


Figure 1.1

Linear regression attempts to model the relationship between two variables by fitting a linear equation to observed data.

Our linear regression function y(x,w) has the form: $Y(x,w) = w^t \Phi(x)$ where w is weight, Φ is vector M basis function

An equation is said to be a *closed-form solution* if it solves a given problem in terms of functions and mathematical operations from a given generally accepted set.

Gradient descent is an optimization algorithm used to find the values of parameters (coefficients) of a function (f) that minimizes a cost function (cost).

Code Explanation

```
#import the packages.
from sklearn.cluster import KMeans
import numpy as np
import csv
import math
import matplotlib.pyplot
from matplotlib import pyplot as plt
```

```
#Initialize the variables
maxAcc = 0.0
maxIter = 0
C_Lambda = 0.9
TrainingPercent = 80
ValidationPercent = 10
TestPercent = 10
M = 15
PHI = []
IsSynthetic = False
```

Figure 2.1

The code imports all the necessary libraries and packages. Initialize the variables.

Pyplot, math, matplotlib.pyplot, csv, numpy, KMeans are the needed libraries which is imported.

From the data that had been given, 80% of the data is used as training data, 10% of the data is used as testing data and remaining 10% of the data is used as validation data.

M is the Mu or the center of the cluster. It defines number of clusters to have.

Then various functions are defined. The list of functions are given below.

1. GetTargetVector: It is used to fetch the raw data from csv file.
2. GenerateRawData: It is used to get the data in matrix format.
3. GenerateTrainingTarget: It is used to get training data. 80% of the raw data is used as training set.
4. GenerateTrainingDataMatrix: The function converts raw data into matrix obtained from function GenerateTrainingTarget.
5. GenerateValData: It is used to get validation set. 10% of the the raw data is used as validation set.
6. GenerateValTargetVector: It returns array of training data.
7. GenerateBigSigma: Generaetes the bias for gradient descent.
8. GetScalar: It gives scalar values in return.
9. GetRadialBasisOut: It returns Φ

10. GetPhiMatrix: The phi matrix is generated, where phi is the function needed to calculate regression
11. GetWeightsClosedForm: It returns the weights which is needed in closed form.
12. GetPhiMatrix: Phi matrix is generated.
13. GetValTest: It returns y. The y is shown in figure 1.1
14. GetErms: It calculates the root mean square error.

Then the data is processed. The training data set, testing data set and validation data set is created.

Prepare Training Data

```
#function calling and storing data, print the variable. It computes Training Data
TrainingTarget = np.array(GenerateTrainingTarget(RawTarget,TrainingPercent))
TrainingData   = GenerateTrainingDataMatrix(RawData,TrainingPercent)
print(TrainingTarget.shape)
print(TrainingData.shape)
```

```
80% Training Target Generated..
80% Training Data Generated..
(55699,)
(41, 55699)
```

Prepare Validation Data

```
#function calling and storing data, print the variable. It computes validation data
ValDataAct = np.array(GenerateValTargetVector(RawTarget,ValidationPercent, (len(TrainingTarget))))
ValData    = GenerateValData(RawData,ValidationPercent, (len(TrainingTarget)))
print(ValDataAct.shape)
print(ValData.shape)
```

```
10% Val Target Data Generated..
(6962,)
(41, 6962)
```

Prepare Test Data

```
#function calling and storing data, print the variable. It computes validation data
TestDataAct = np.array(GenerateValTargetVector(RawTarget,TestPercent, (len(TrainingTarget)+len(ValDataAct))))
TestData = GenerateValData(RawData,TestPercent, (len(TrainingTarget)+len(ValDataAct)))
print(ValDataAct.shape)
print(ValData.shape)
```

```
10% Val Target Data Generated..
(6962,)
(41, 6962)
```

Figure 2.2

Then the closed form solution and Gradient descent solution is done. Each gives different accuracy for different mu.

Closed Form with 0.9 Lambda		Gradient descent with 0.003 learning rate	
Mu	Accuracy	Mu	Accuracy
3	70.21	3	70.5
4	7023	4	70.17
5	70.11	5	71.04
6	70.16	6	70.78
7	69.94	7	70.69
8	69.98	8	70
9	69.86	9	71
10	69.35	10	70.69
11	69.94	11	70.78
12	69.36	12	70.65

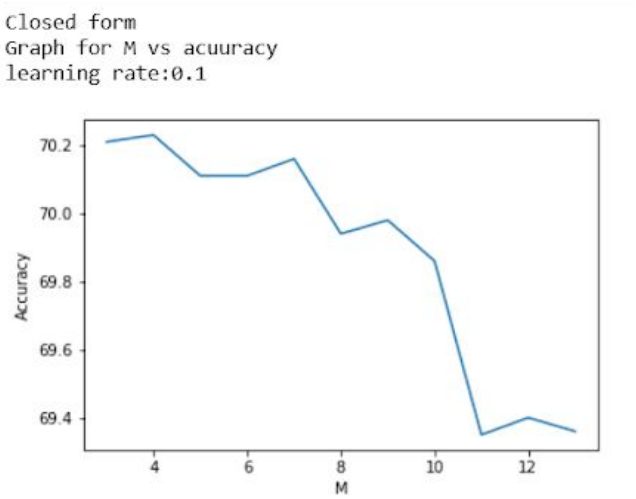


Figure 2.3

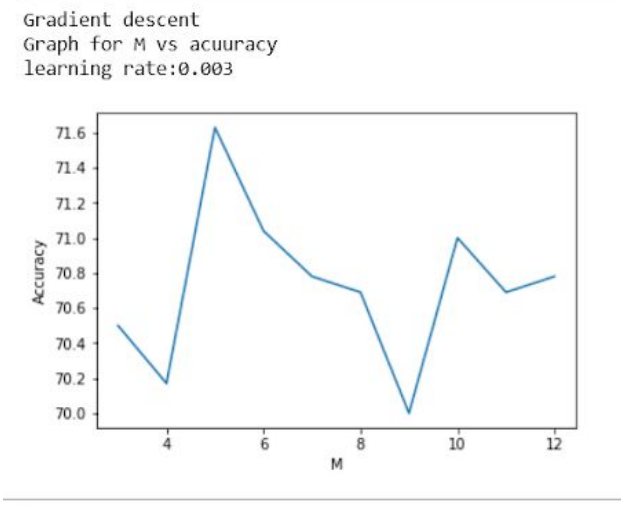


Figure 2.4

The graphs are displayed below:

Summary

From this project I learned various types of libraries, packages, functions, parameters and hyperparameters,API’s available for machine learning. How each of them interacts with the data and delivers the output. I also learned how a build a linear model is built and how to train a model.

Reference

1. Numpy Manual

   https://docs.scipy.org/doc/numpy-1.14.0/reference/routines.linalg.html

2. Stack exchange

   https://stats.stackexchange.com/questions/70848/what-does-a-closed-form-solution-mean

3. Gradient Descent

   https://machinelearningmastery.com/gradient-descent-for-machine-learning/

4. K- means clustering

   https://jakevdp.github.io/PythonDataScienceHandbook/05.11-k-means.html