Machine Learning & Data Mining
T3 2019

COMP9417 – Homework-1

Chirag Panikkasseril Unni – z5241855

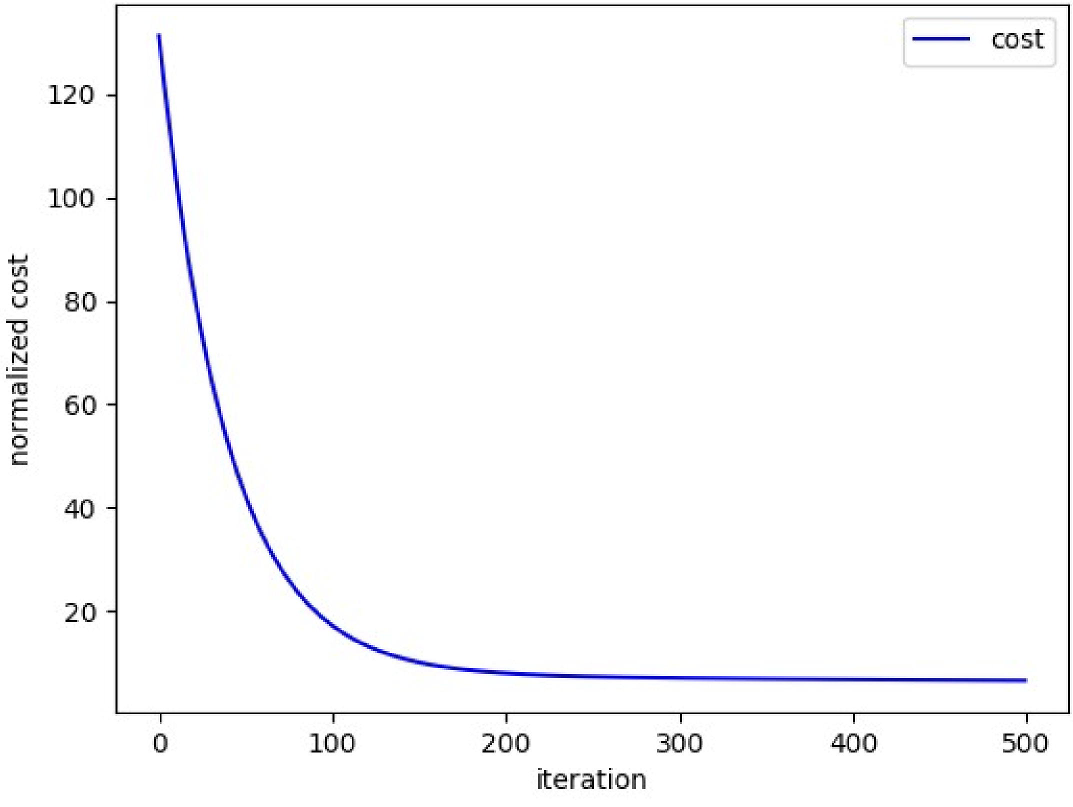
11/10/2019

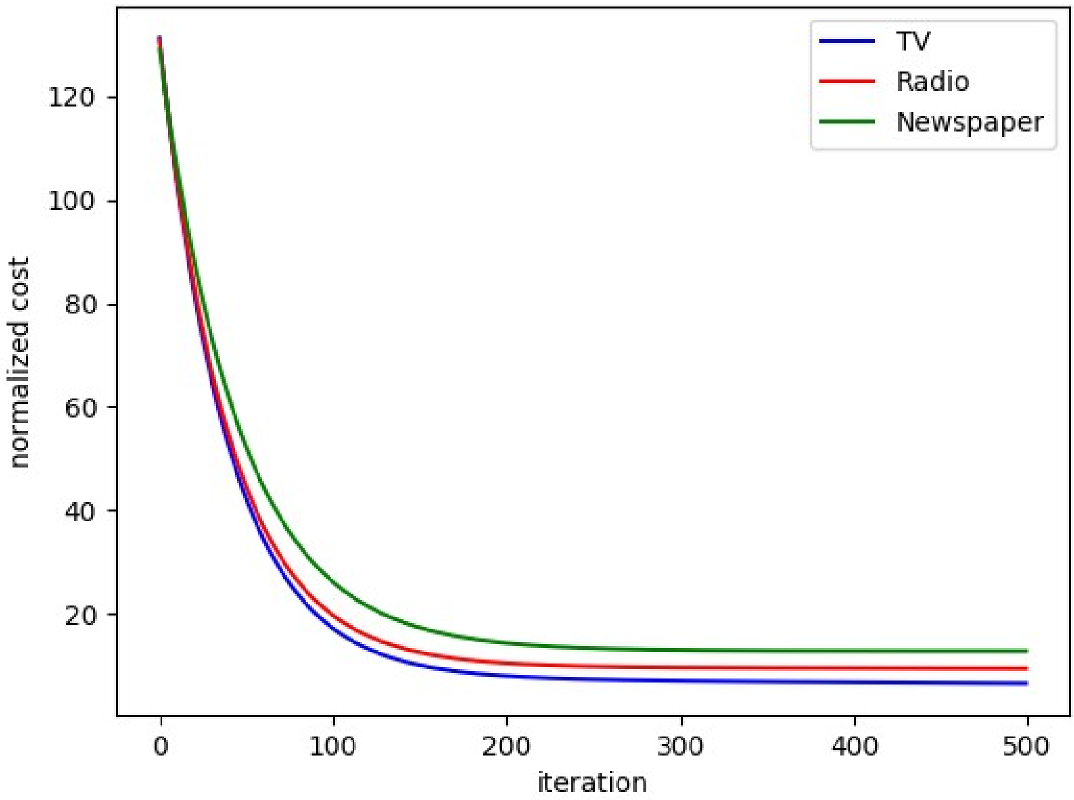
1. Estimated θ parameters for TV feature for the below parameters is

$\theta_0 = -1$, $\theta_1 = -0.5$
$\alpha = 0.01$
maximum iteration = 500

estimated Theta values for iteration:500:
$\theta_0 = 10.11455567$, $\theta_1 = 8.26795499$

2. A plot, which visualizes the change in cost function J θ at each iteration.





Cost curve for TV feature

Combined Cost Curve for all features

3. RMSE for TV **train** data: **3.6408394507948225**

4. RMSE for TV **test** data: **3.9092605131614144**

5. RMSE for Radio **test** data: **4.200750008993373**

6. RMSE for newspaper **test** data: **5.427934797044488**

7. Using the RMSE data we can see that estimated model can be ranked as:

TV > Radio > newspaper

Python 3 code:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt


def func_rmse(X, Y, theta):
    X1 = np.append(np.ones(shape=(X.shape[0], 1)), X, axis=1)
    y_hat = np.dot(X1, theta.T)
    m = X.shape[0]  # no.of training data items
    rmse = np.sqrt((1 / m) * np.square(Y - y_hat).sum())
    return rmse


def gradient_descent(X, Y, theta, alpha, max_iter):
    X1 = np.append(np.ones(shape=(X.shape[0], 1)), X, axis=1)
    cost = np.ones(shape=(max_iter, 1))
    theta_hist = np.ones(shape=(max_iter, 2))
    m = X.shape[0]  # no.of training data items
    for i in range(max_iter):
        theta_hist[i] = theta
        h = np.dot(X1, theta.T)
        cost[i] = (1 / (2 * m)) * (np.square(Y - h).sum())
        theta = theta + ((alpha / m) * np.sum(((Y - h) * X1),
axis=0))

    return cost, theta_hist


def normalize(x, min, max):
    new = ((x - min) / (max - min))
    return new


df = pd.read_csv("Advertising.csv")
data = np.array(df)

# 1. Pre--processing:

minValues = np.amin(data, axis=0)[1:5]
maxValues = np.amax(data, axis=0)[1:5]

min_TV, min_Radio, min_News, min_Sales = minValues
max_TV, max_Radio, max_News, max_Sales = maxValues

# min-max normalization

for i in range(data.shape[0]):
    data[i][1] = normalize(data[i][1], min_TV, max_TV)
    data[i][2] = normalize(data[i][2], min_Radio, max_Radio)
    data[i][3] = normalize(data[i][3], min_News, max_News)

# 2. Creating test and training set
train_data = np.copy(data[:-10])
test_data = np.copy(data[-10:])

# 3. Gradient descent
theta0 = -1
theta1 = -0.5
alpha = 0.01
max_iter = 500
theta = np.array([[theta0, theta1]])

# train TV x Sales
print(f'train TV x Sales')
X = train_data[:, [1]]
Y = train_data[:, [4]]

cost_TV, theta_hist = gradient_descent(X, Y, theta, alpha,
max_iter)
print(f'min of J(0):{np.amin(cost_TV)}')
est_theta = theta_hist[-1:]
print(f'estimated Theta values for iteration:{max_iter}:
{est_theta}')

# RMSE TV train data
rmse = func_rmse(X, Y, est_theta)
print(f'RMSE for TV train data:{rmse}')
print()

# A plot, which visualises the change in cost function J θ at
each iteration.
plt.figure()
plt.plot(np.arange(max_iter), cost_TV, 'b', label='TV')
plt.xlabel('iteration')
plt.ylabel('normalized cost')
plt.legend()
plt.show()

# test TV x Sales
print(f'test TV x Sales')

X = test_data[:, [1]]
Y = test_data[:, [4]]

# RMSE TV test data
```

```python
rmse = func_rmse(X, Y, est_theta)
print(f'RMSE for TV test data:{rmse}')
print()

#train Radio x Sales
print(f'train Radio x Sales')

X = train_data[:, [2]]
Y = train_data[:, [4]]

cost_Radio, theta_hist = gradient_descent(X, Y, theta, alpha,
max_iter)
print(f'min of J(O):{np.amin(cost_Radio)}')

est_theta = theta_hist[-1:]
print(f'estimated Theta values for iteration:{max_iter}:
{est_theta}')

# RMSE Radio train data
rmse = func_rmse(X, Y, est_theta)
print(f'RMSE for Radio train data:{rmse}')
print()


# test Radio x Sales
print(f'test Radio x Sales')

X = test_data[:, [2]]
Y = test_data[:, [4]]

rmse = func_rmse(X, Y, est_theta)
print(f'RMSE for  Radio test data:{rmse}')
print()


#train newspaper x Sales
print(f'train newspaper x Sales')

X = train_data[:, [3]]
Y = train_data[:, [4]]

cost_news, theta_hist = gradient_descent(X, Y, theta, alpha,
max_iter)
print(f'min of J(O):{np.amin(cost_news)}')

est_theta = theta_hist[-1:]
print(f'estimated Theta values for iteration:{max_iter}:
{est_theta}')
```

```python
# RMSE newspaper train data
rmse = func_rmse(X, Y, est_theta)
print(f'RMSE for newspaper train data:{rmse}')
print()


# test newspaper x Sales
print(f'test newspaper x Sales')

X = test_data[:, [3]]
Y = test_data[:, [4]]

rmse = func_rmse(X, Y, est_theta)
print(f'RMSE for  newspaper test data:{rmse}')
print()

# A plot, which visualises the change in cost function J θ at
each iteration.
plt.figure()
plt.plot(np.arange(max_iter), cost_TV, 'b', label='TV')
plt.plot(np.arange(max_iter), cost_Radio, 'r', label='Radio')
plt.plot(np.arange(max_iter), cost_news, 'g',
label='Newspaper')
plt.xlabel('iteration')
plt.ylabel('normalized cost')
plt.legend()
plt.show()
```

Console Output:

```
/usr/bin/python3.7 /home/chirag/PycharmProjects/9417homework1/h.py
train TV x Sales
min of J(0):6.627855953231973
estimated Theta values for iteration:500:[[10.11455567
8.26795499]]
RMSE for TV train data:3.6408394507948225

test TV x Sales
RMSE for TV test data:3.9092605131614144

train Radio x Sales
min of J(O):9.493644157608461
estimated Theta values for iteration:500:[[10.82283051
7.01105184]]
RMSE for Radio train data:4.357440569327013

test Radio x Sales
RMSE for  Radio test data:4.200750008993373
```

```
train newspaper x Sales
min of J(O):12.846999312254798
estimated Theta values for iteration:500:[[12.96616311
3.83655296]]
RMSE for newspaper train data:5.068924799650277

test newspaper x Sales
RMSE for  newspaper test data:5.427934797044488


Process finished with exit code 0
```