University of Roehampton London

Week-6

Lab Task Solution

Question:1

Now, using column stacking, create a Matrix that contains 3 columns. The first column is specifically a column with ones, the second column contains an array that hold values: 2, 3, 4 and the third column contains an array that holds values: 8, 9, 10.

Solution:

```
y_r = np.array([2, 3, 4], ndmin=2)
print("Vector y_r is:\n", y_r)
print("The shape of y_r is :", y_r.shape)
y_c = np.transpose(y_r)
print("\nVector y_c is:\n", y_c)

z_r = np.array([8, 9, 10], ndmin=2)
print("Vector z_r is:\n", z_r)
print("The shape of z_r is :", z_r.shape)
z_c = np.transpose(z_r)
print("\nVector z_c is:\n", z_c)

Y = np.column_stack([np.ones(y_c.shape), y_c, z_c])
print("The design matrix X is:\n", Y)
print("\nThe shape of the matrix X is", Y.shape)
```

Question:2

Let's go back to the first regression example you have seen today, where x was the predictor and y was the label with training data as:

x = [0.3000, -0.7700, 0.9000, -0.0400, 0.7400, -0.5800, -0.9200, -0.2100, -0.5400, 0.6800]

y=[1.1492, 0.3582, 1.9013, 0.9487, 1.3096, 0.9646, 0.1079, 1.1262, 0.6131, 1.0951]

You are now to find the optimal solution for this given dataset and display the linear regression line with proper plotting of the datapoints that are given below:
x_test=[0.2000, 0.990, 0.3450, -0.9910, -0.2480, 0.1000, 0.9120, 0.7450, 0.3450, -0.1000]

Solution:

```
x = np.array([0.3000, -0.7700, 0.9000, -0.0400, 0.7400, -0.5800, -
0.9200, -0.2100, -0.5400, 0.6800], ndmin=2).T
y = np.array([1.1492,  0.3582, 1.9013,  0.9487, 1.3096,  0.9646,  0.107
9,  1.1262,  0.6131, 1.0951], ndmin=2).T

plt.plot(x, y, 'o', label="Training Data")
plt.xlabel("x (attribute)", fontsize=18)
plt.ylabel("y (label)", fontsize=18)
plt.xlim(-1,1)
plt.ylim(-0.5,2.5)
plt.grid(alpha=0.2)
plt.legend(fontsize=12)
plt.show()


X = np.column_stack([np.ones(x.shape), x])
print("The design matrix is:\n", X)

XTX = np.dot(X.T, X) # Step 1
XTX_inv = np.linalg.inv(XTX) # Step 2
XTX_invXT = np.dot(XTX_inv, X.T) # Step 3

w = np.dot(XTX_invXT, y)
print("The 2 parameters of the least squares linear solution are\n", w)


x_LS = np.array([0.2000, 0.990, 0.3450, -0.9910, -
0.2480, 0.1000, 0.9120, 0.7450, 0.3450, -0.1000],ndmin=2).T
X_LS = np.column_stack([np.ones(x_LS.shape), x_LS])
y_LS = np.dot(X_LS, w)


plt.plot(x, y, 'o', label="Dataset")
plt.plot(x_LS, y_LS, label="Linear prediction")
plt.xlabel("x (attribute)", fontsize=18)
plt.ylabel("y (label)", fontsize=18)
plt.xlim(-1,1)
plt.ylim(-0.5,2.5)
plt.grid(alpha=0.2)
plt.legend(fontsize=12)
plt.show()
```

Question:3

Now, you are given with another dataset that contains the price (in thousand £) and size (in sq. feet) of the houses.

Find the prices for the houses of the following sizes: 2000, 1200, 1480, 2200, 1890, 1050, 2390

Also, draw the trend-line using the predicted values.

Solution:

```
df_house_price = pd.read_csv('./House Price Dataset.csv')
df_house_price.info()
```

```python
predictor= df_house_price.iloc[:,1]
x=np.array(predictor, ndmin=2).T
print (x)
label= df_house_price.iloc[:,0]
y=np.array(label, ndmin=2).T
print (y)

plt.plot(x, y, 'o', label="Training Data")
plt.xlabel("x (size)", fontsize=18)
plt.ylabel("y (price)", fontsize=18)
plt.xlim(1000,2500)
plt.ylim(200,500)
plt.grid(alpha=0.2)
plt.legend(fontsize=12)
plt.show()


X = np.column_stack([np.ones(x.shape), x])
print("The design matrix is:\n", X)
XTX = np.dot(X.T, X) # Step 1
XTX_inv = np.linalg.inv(XTX) # Step 2
XTX_invXT = np.dot(XTX_inv, X.T) # Step 3

w = np.dot(XTX_invXT, y)
print("The 2 parameters of the least squares linear solution are\n", w)


x_arr = np.array([2000, 1200, 1480, 2200, 1890, 1050, 2390]).T
X_arr = np.column_stack([np.ones(x_arr.shape), x_arr])
y_arr = np.dot(X_arr, w)

print(x_arr)
print(y_arr)

plt.plot(x, y, 'o', label="Dataset")
plt.plot(x_arr, y_arr, label="Linear prediction")
plt.xlabel("x (size)", fontsize=18)
plt.ylabel("y (price)", fontsize=18)
plt.xlim(1000,2500)
plt.ylim(150,500)
plt.grid(alpha=0.2)
plt.legend(fontsize=12)
plt.show()
```