# Lab3_Ex1

May 30, 2020

[1]:

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

[2]:

```
#import Boston_Housing.csv
dataset = pd.read_csv('E:/University Works/3rd Year/Semester 6/CO 544 - Machine␣
 ↪Learning and Data Mining/Lab/3/Boston_Housing.csv',sep=',')
print(dataset.head())
```

```
      RM  LSTAT  PTRATIO      MEDV
0  6.575   4.98     15.3  504000.0
1  6.421   9.14     17.8  453600.0
2  7.185   4.03     17.8  728700.0
3  6.998   2.94     18.7  701400.0
4  7.147   5.33     18.7  760200.0
```

[3]:

```
print ("Dataset Length: ", len(dataset))
print ("Dataset Shape: ", dataset.shape)
```

```
Dataset Length:  489
Dataset Shape:  (489, 4)
```

[4]:

```
#check for missing values
print(dataset.__eq__('?').sum())
```

```
RM         0
LSTAT      0
PTRATIO    0
MEDV       0
dtype: int64
```

```
c:\users\user\appdata\local\programs\python\python38\lib\site-
packages\pandas\core\ops\array_ops.py:253: FutureWarning: elementwise comparison
failed; returning scalar instead, but in the future will perform elementwise
comparison
  res_values = method(rvalues)
```

[5]:

```
#count the NaN values
print(dataset.isnull().sum())
```

```
RM         0
LSTAT      0
PTRATIO    0
MEDV       0
dtype: int64
```

[6]:

```
print(dataset.dtypes)
```

```
RM         float64
LSTAT      float64
PTRATIO    float64
MEDV       float64
dtype: object
```

[7]:

```
#split 80% train 20% test
data_train = dataset.values[0:round(len(dataset)*0.8),0:4]
data_test = dataset.values[round(len(dataset)*0.8):,0:4]
```

[8]:

```
print("data_train shape = ",data_train.shape)
print("data_test shape = ",data_test.shape)
```

```
data_train shape =  (391, 4)
data_test shape =  (98, 4)
```

[22]:

```
X_train = data_train[:,0:3]
Y_train = data_train[:,3]
X_test = data_test[:,0:3]
Y_test = data_test[:,3]
X_train = np.hstack((np.ones((391,1)),X_train))
X_test = np.hstack((np.ones((98,1)),X_test))
```

[23]:

```
b_hat = np.dot(np.dot(np.linalg.inv(np.dot(X_train.T,X_train)),X_train.
 ↪T),Y_train)
```

[24]:

```
print("beta_hat = ",b_hat)
```

```
beta_hat =  [178553.97347306 111079.91663977  -9267.33228459 -15607.43116324]
```

[25]:

```
y_hat_X_train = np.dot(X_train,b_hat)
y_hat_X_test = np.dot(X_test,b_hat)
```

[26]:

```
print("Predictions for training set = \n",y_hat_X_train)
```

```
Predictions for training set =
 [623959.41380481 529282.42643031 661503.55071734 636786.31044899
```

```
631188.29386815 552656.07351159 493940.53833276 449436.85204269
289440.77511272 449773.45723062 460160.70298388 485822.73946379
449880.85869262 435064.17846442 432858.26164129 420343.84827108
449078.17786957 380214.85510244 348514.82982488 382417.09347094
274713.7301066  385222.4163055  359566.30667911 312270.9086545
357777.82898034 319730.71629265 359256.28333736 362358.67308814
453640.12437771 481122.64192959 275955.77317644 404429.15989078
254925.64544582 314008.97638626 339443.55221264 448220.66404789
421968.8268647  447434.30334577 447716.00176894 585475.15795573
654813.99974013 606338.103045   530589.76082844 520149.36570343
484821.94642011 435717.57937269 410756.60150195 394767.00603872
213374.91790112 371319.86565682 454073.05362596 508211.87673925
590658.95270964 504482.85877698 366119.20729187 659823.40308399
564096.01497868 663396.51220077 490099.76783645 444013.44181735
386931.96144392 399970.08404075 525850.37510841 534170.31917187
602765.45675489 582688.56564011 475196.32674515 461435.88299133
383645.19856554 455818.98313144 529414.64726393 449477.43525689
501435.31534823 502709.68912852 520666.81491234 502093.41609395
473235.83985517 473550.19632604 464586.1708278  454845.71727283
580223.19282186 550340.61051551 519761.943083   497444.9618318
510354.34778697 565760.66445653 438784.30236163 491520.38222333
628108.34709712 634252.63463826 531896.32641767 536216.44011159
534454.11785335 526865.95506451 490496.07649032 571896.9005808
477114.19067851 754908.59098303 733180.78440186 664024.08749377
512294.99127637 534511.13827294 465313.78605401 409503.16467486
423122.30100994 349654.29169358 327692.63524074 402368.59952348
457779.87535543 400168.44588524 420023.42604518 552487.24299212
407333.78952527 419061.84219335 498591.87505866 413170.28780293
475192.58322826 474100.35432428 410564.07206209 411212.30863676
399319.58400119 415131.02605938 376436.15348228 295453.11674639
370571.16661742 408127.22812619 251316.13227643 320748.95627062
419407.13286303 303871.04881152 448262.14768631 436750.49166662
452424.92663648 355095.70325186 326745.99106142 394193.74917881
351095.38187646 429375.83677342 300691.9802642  359854.04609021
309585.09015077  86297.63051481 300739.67310554 311666.80060096
222309.40404272 372412.78686375 419981.42368374 222640.08237713
262734.33276782 372054.75130189 498486.59982949 426330.4321555
393537.21028283 436948.80269631 489811.48031569 493293.05162686
385163.31282513 677815.54141718 563346.56312046 603769.40711531
592403.88680684 491514.81959012 535910.77708094 489435.99153054
548171.34026129 555352.16023978 467991.64076555 490788.63783176
402270.80041956 548382.67747145 480950.76453235 597204.86941045
494478.98493759 562648.76967345 617348.90490284 629372.16219876
693216.21940381 495640.41651287 650849.96071332 577120.74429783
393676.24627821 462351.00630965 632759.23478177 627301.91806451
689479.29983472 666174.12570354 646424.78961259 712055.4177754
643806.79908556 628058.90140915 753528.97764919 691553.57740485
728545.91473002 645751.02704275 664543.22025642 564742.90283329
```

```
765621.4975971  786126.78474135 441891.64082831 489284.67133815
363262.89070502 425985.2770484  267884.12590868 390245.22845509
266300.99517379 384741.49324209 509462.64558599 215570.59368175
487190.1617689  451428.99240619 570584.45887971 417558.09177946
533197.42215309 589115.37531247 392804.32657306 579078.8082118
571679.83419191 786804.50651893 771060.45096577 643709.88079347
724416.96797047 599935.1156665  463389.23153979 681655.51887267
810167.62551561 786454.78123684 579506.16566094 482188.4686461
555145.24847612 680476.21616211 580439.32257573 584964.30654827
580126.55982925 481587.78775387 521737.25792594 581507.0686844
385880.35846416 331980.01704769 474039.40536416 477972.17656592
506804.13864964 565893.21845435 546350.1970184  562314.81868044
620521.33535356 765054.7134957  540181.86012198 489574.86893886
729564.11320541 718013.87857351 671721.56523666 686892.29129808
743697.50909598 853736.57448989 685282.429369   701033.85615203
496418.08249419 617708.0591733  776139.5756308  419349.7746597
418264.42597974 520322.71391362 542859.77026785 681592.34628078
621827.5786871  637588.28344104 655013.57889958 623542.59367931
557251.08553293 657733.07971073 779803.02787382 677471.05310053
767314.5416753  654346.16738425 580288.83404985 466700.00052823
542997.06606774 550508.56438064 560577.93887702 610150.38669674
639898.8363469  571794.68072306 529908.01311141 499934.03818997
612520.58475732 587811.86335205 425193.2488177  606307.42188213
685750.53028886 654541.3925216  571251.32569748 568390.61941291
657795.07482076 630928.90414272 543524.69125689 655630.58165535
582380.57703238 586318.79840239 462351.1993649  326638.58544192
515989.84267638 451798.4436155  514192.07268609 534838.20904193
418513.84322666 378433.66630064 385920.04146436 504279.70559622
452435.63050565 519723.07455179 517227.29837371 472323.64050318
397894.0058324  528509.9143361  539925.21857831 516790.66295367
429818.2756735  474210.03477922 550235.27992518 513023.34418224
433417.45086349 512148.15228052 512226.17009604 501643.6450517
459641.98913046 424392.03834556 420236.94364191 457452.08115424
437833.34861285 440115.28955804 690090.81258911 576696.04854004
581114.86101669 624704.13014996 455583.87731911 422866.51589988
564037.45914592 594731.57021826 587286.53396448 536577.59148761
572855.10051652 474339.69419733 618870.3275698  389634.03092197
442941.83223041 390207.25793322 450662.43147056 437481.53860052
424787.21443226 501780.65214002 426140.97777261 364460.26101821
372206.87558993 192856.04126366 284830.83827466 168852.04260158
 86016.7914753  -28948.04781473 551058.34845749 386481.42741972
421122.67990167 352430.63031556 352691.68806668 477797.93450388
394946.5538239  259513.2405772  248839.32291776  64622.55198129
163926.07639562 117947.47917577 122221.48739059 121588.1426237
268780.73653041 339319.37234942 361573.28590409 184697.23110537
410616.35696861 365690.45038103 423425.27583952 395242.50370097
317054.88579548 185514.95482683 235581.58758255 280232.8396396
379551.58319895 386420.13143682 274235.17881554 223927.32495826
```

```
 281587.73433969 106633.02350879 373807.29587949]
```

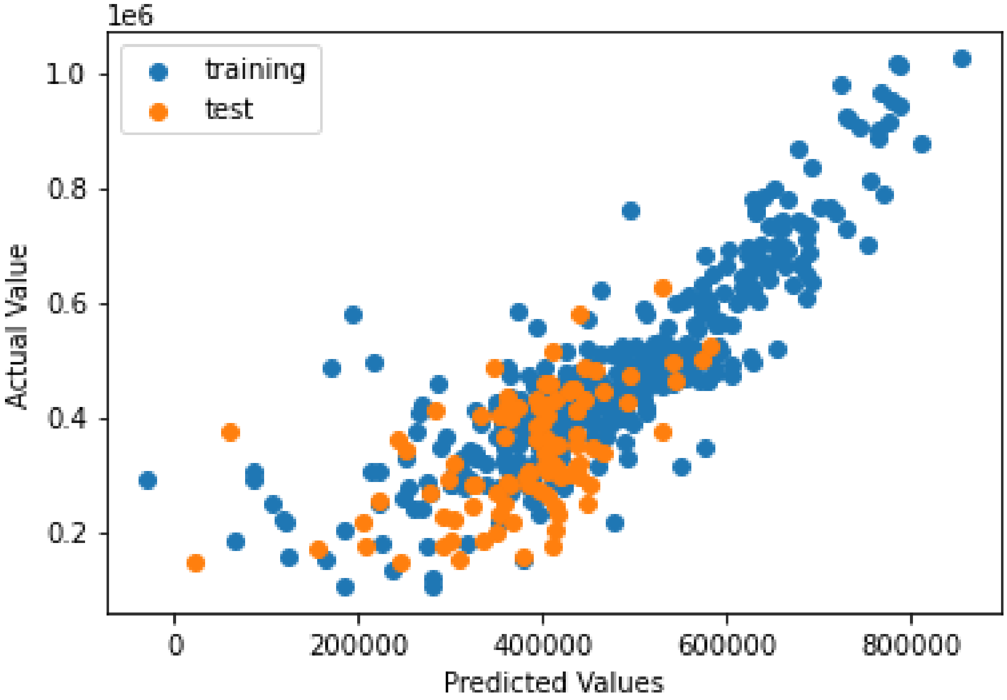
[27]:

```
print("Predictions for test set = \n",y_hat_X_test)
```

```
Predictions for test set =
 [242562.18342818 441095.62020227 409078.21467369 406090.07796771
  58843.50756328 249812.8019792   22548.05938677 308756.04476871
 377623.35900711 205570.00977163 333894.53569062 410554.07897398
 436221.87863877 384932.72644614 359993.84794444 325368.42632018
 322416.1780725  292180.8180627  366252.89385687 417639.84220335
 351768.79696148 348816.37072457 404943.43330484 439819.16169173
 465486.32100265 427878.07781326 412282.55571821 383981.80911839
 413695.85764846 301526.57221623 207268.52491107 276293.99223633
 304646.10217366 393963.48377493 400344.82295713 408947.90982014
 293075.86093458 358518.41736757 402775.77404568 405289.48013229
 382296.41407311 397131.50263773 451339.94633431 438210.90780764
 402707.27050151 529362.54524954 437237.75608345 420063.58573064
 350925.45908496 365665.64024847 412789.61574404 402531.05247871
 455460.78926588 435762.99450168 435325.71872776 491386.51184213
 430464.93358979 372045.69872292 365496.77913494 332517.07925684
 353526.71566341 361097.6032182  397348.86697724 435930.0982222
 445226.39577923 530564.58768302 298005.16393732 324415.60225141
 410412.27639188 221602.49462397 383223.14730157 433705.93847389
 457113.55490469 541344.14941145 582655.15605417 406760.74124866
 391807.8418914  466371.922235   403601.83668814 413099.81707502
 303306.45138568 244093.32091818 155520.20033181 361975.05396528
 405716.86234853 401523.71866422 411207.83539868 345609.3742777
 281700.64133112 391818.94693709 427747.38027685 357558.63340855
 415569.0810887  493532.70625917 446459.63173646 573423.66343909
 545422.5594916  447583.23798039]
```

[28]:

```
plt.scatter(y_hat_X_train,Y_train,label='training')
plt.scatter(y_hat_X_test,Y_test,label='test')
plt.xlabel("Predicted Values")
plt.ylabel("Actual Value")
plt.legend()
plt.show()
```


1e6
1.0
0.8
0.6
0.4
0.2
Actual Value
0
200000
400000
600000
800000
Predicted Values
● training
● test

```
[ ]:
```