assignment-03

February 13, 2022

1 Assignment 3

This assigment focuses on getting comfortable with working with multidimensional data and linear regression. Key items include: - Creating random n-dimensional data - Creating a Model that can handle the data - Plot a subset of the data along with the prediction - Using a Dataset to read in and choose certain columns to produce a model - Create several models from various combinations of columns - Plot a few of the results

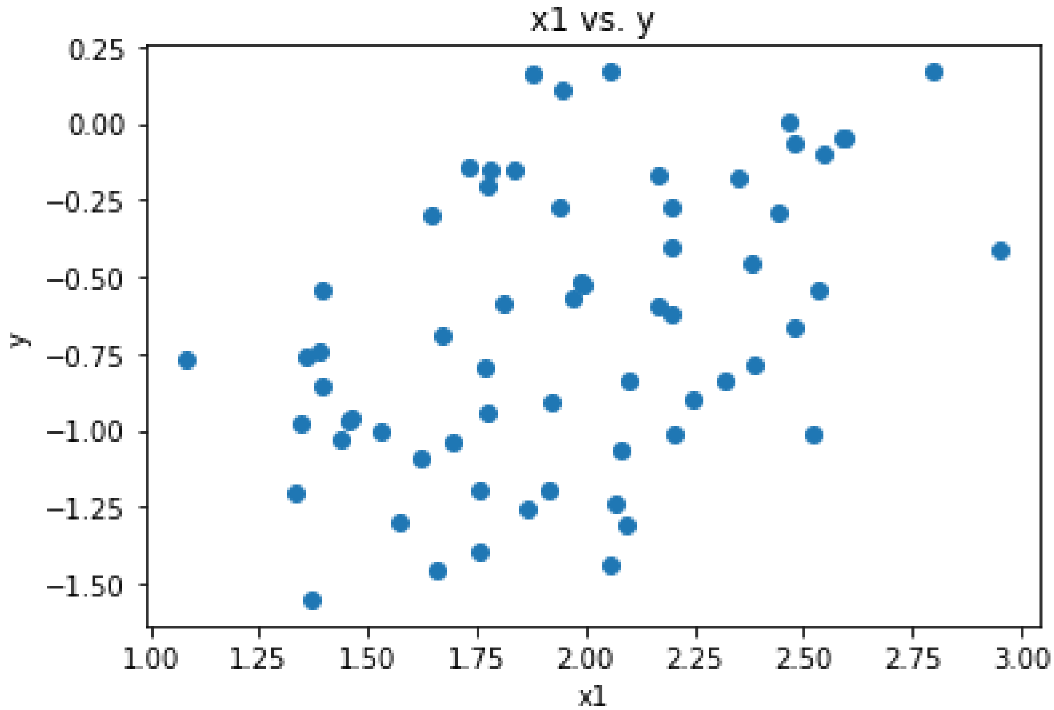
1.1 1. Create a 4 dimensional data set with 64 elements and show all 4 scatter 2D plots of the data x_1 vs. y, x_2 vs. y, x_3 vs. y, x_4 vs. y

[72]:
```
import numpy as np
import matplotlib.pylab as plt
%matplotlib inline
```

[73]:
```
n = 64
x = np.linspace(0, 1, n) + np.random.rand(4, n)
x = np.vstack([x, np.ones(len(x.T))]).T + 1
y = np.linspace(0, 1, n) + np.random.rand(n) - 1.7
```

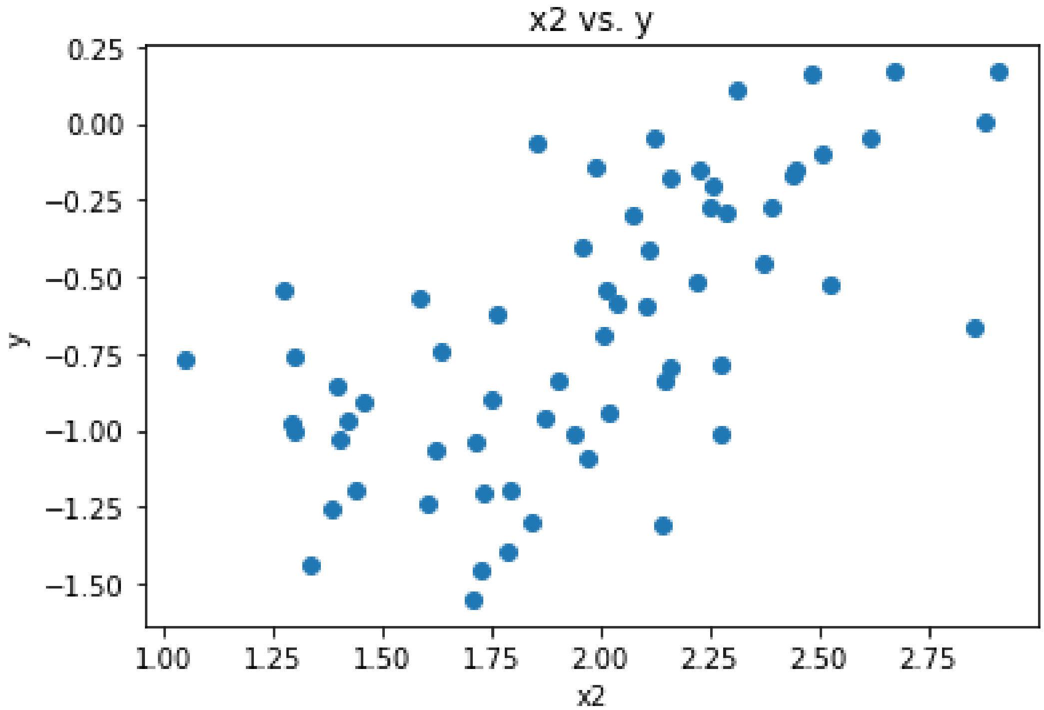
[74]:
```
fig1, ax1 = plt.subplots()
ax1.scatter(x.T[0], y)
ax1.set(xlabel='x1', ylabel='y', title='x1 vs. y')
```

[74]:
```
[Text(0.5, 0, 'x1'), Text(0, 0.5, 'y'), Text(0.5, 1.0, 'x1 vs. y')]
```



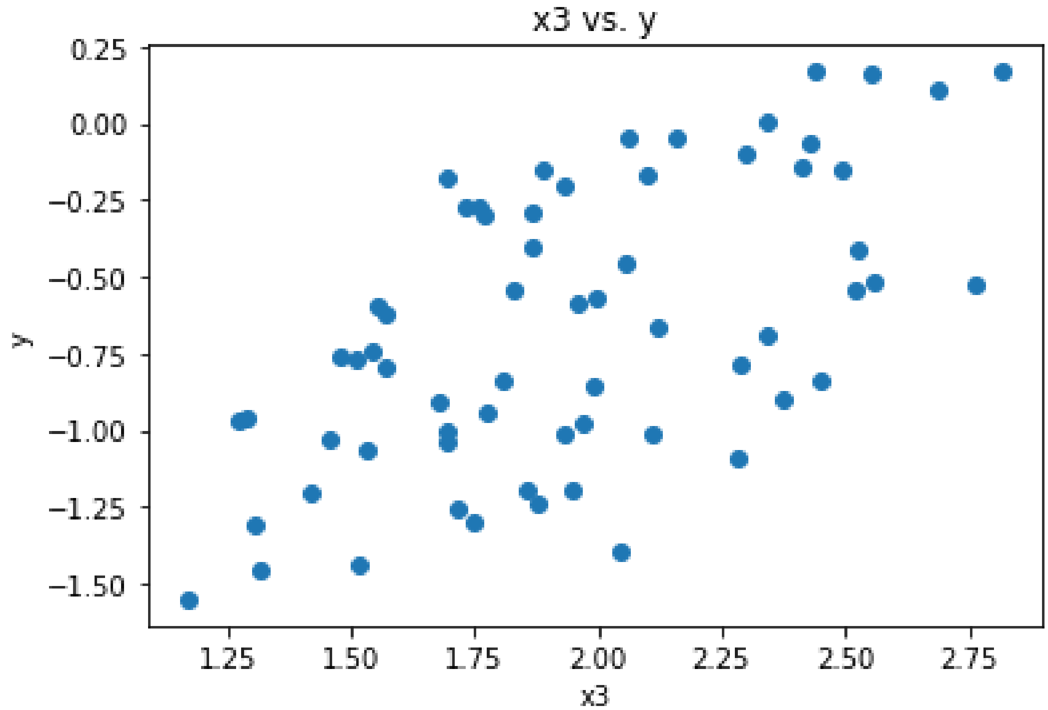
[75]:
```python
fig2, ax2 = plt.subplots()
ax2.scatter(x.T[1], y)
ax2.set(xlabel='x2', ylabel='y', title='x2 vs. y')
```

[75]: [Text(0.5, 0, 'x2'), Text(0, 0.5, 'y'), Text(0.5, 1.0, 'x2 vs. y')]



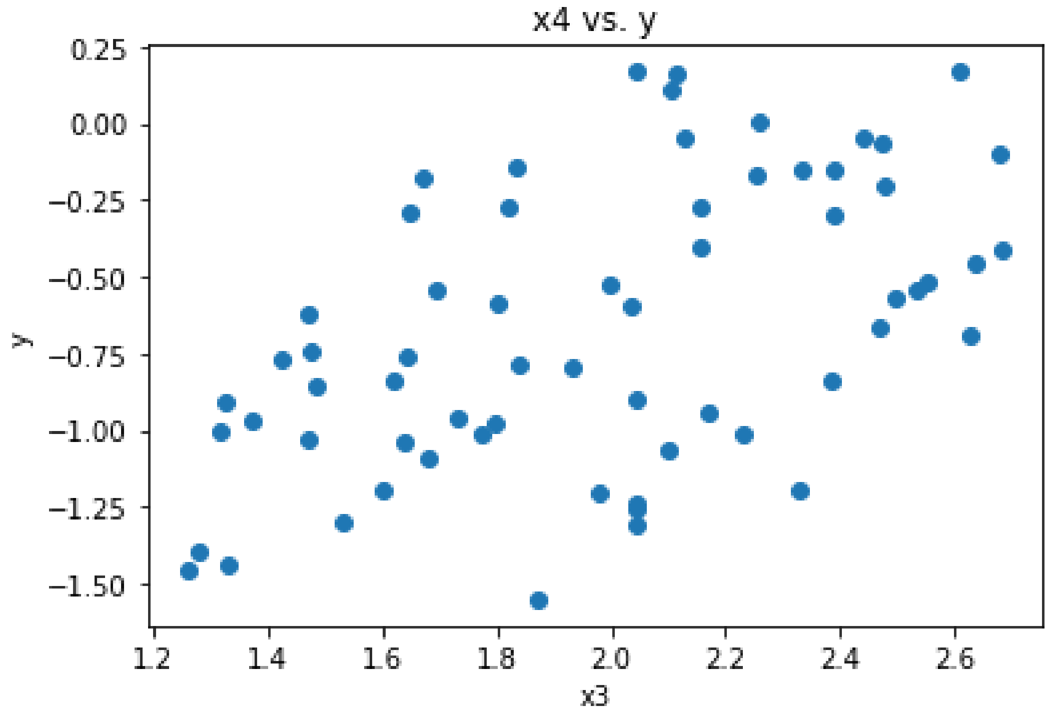
[76]:
```
fig3, ax3 = plt.subplots()
ax3.scatter(x.T[2], y)
ax3.set(xlabel='x3', ylabel='y', title='x3 vs. y')
```

[76]: [Text(0.5, 0, 'x3'), Text(0, 0.5, 'y'), Text(0.5, 1.0, 'x3 vs. y')]



[78]:
```
fig4, ax4 = plt.subplots()
ax4.scatter(x.T[3], y)
ax4.set(xlabel='x3', ylabel='y', title='x4 vs. y')
```

[78]: [Text(0.5, 0, 'x3'), Text(0, 0.5, 'y'), Text(0.5, 1.0, 'x4 vs. y')]



1.2 2. Create a Linear Regression model (like we did in class) to fit the data. *Use the example from Lesson 3 and do not use a library that calculates automatically.* We are expecting 5 coefficients to describe the linear model.

1.3 After creating the model (finding the coefficients), create a new column $y_p = \Sigma\beta_n \cdot x_n$

[79]:
```
left = np.linalg.inv(np.dot(x.T, x))
right = np.dot(y.T, x)
beta = np.dot(left, right)

yp = np.dot(x, beta)
yp
```

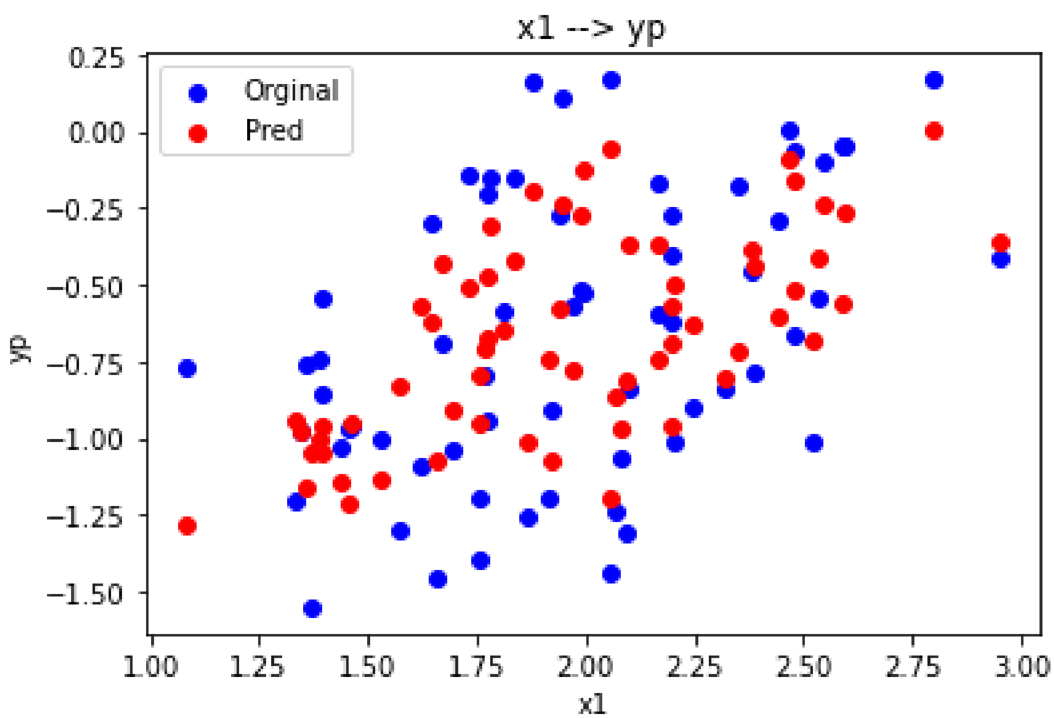
[79]:
```
array([-1.05130587, -1.28325934, -1.20966498, -0.9051003 , -0.95324249,
       -1.07301048, -0.93981536, -1.13128474, -1.01455521, -0.80872152,
       -1.16387332, -1.19377784, -1.07536552, -0.64820655, -0.9532798 ,
       -1.04897012, -0.70830873, -0.7933212 , -0.97604168, -0.50115522,
       -1.00344669, -0.86365234, -0.9579664 , -0.74154068, -1.14306392,
       -0.83272528, -0.74285088, -0.96606662, -0.96213385, -0.80156523,
       -0.57323347, -0.58115791, -0.60276789, -0.71570281, -0.62795027,
       -0.77513201, -0.30634602, -0.68219749, -0.62289382, -0.50759891,
       -0.57299885, -0.27381677, -0.42983934, -0.38306709, -0.47747891,
```

```
-0.51942695, -0.37158219, -0.6771009 , -0.41887962, -0.43899939,
-0.56273734, -0.41028019, -0.23582602, -0.69343279, -0.37084504,
-0.19891985, -0.2613792 , -0.23703012,  0.00215549, -0.15678061,
-0.12501621, -0.05375599, -0.09123372, -0.35973604])
```

1.4 3. Plot the model's prediction as a different color on top of the scatter plot from Q1 in 2D for all 4 of the dimensions ($x_1 \to y_p, x_2 \to y_p, x_3 \to y_p, x_4 \to y_p$)

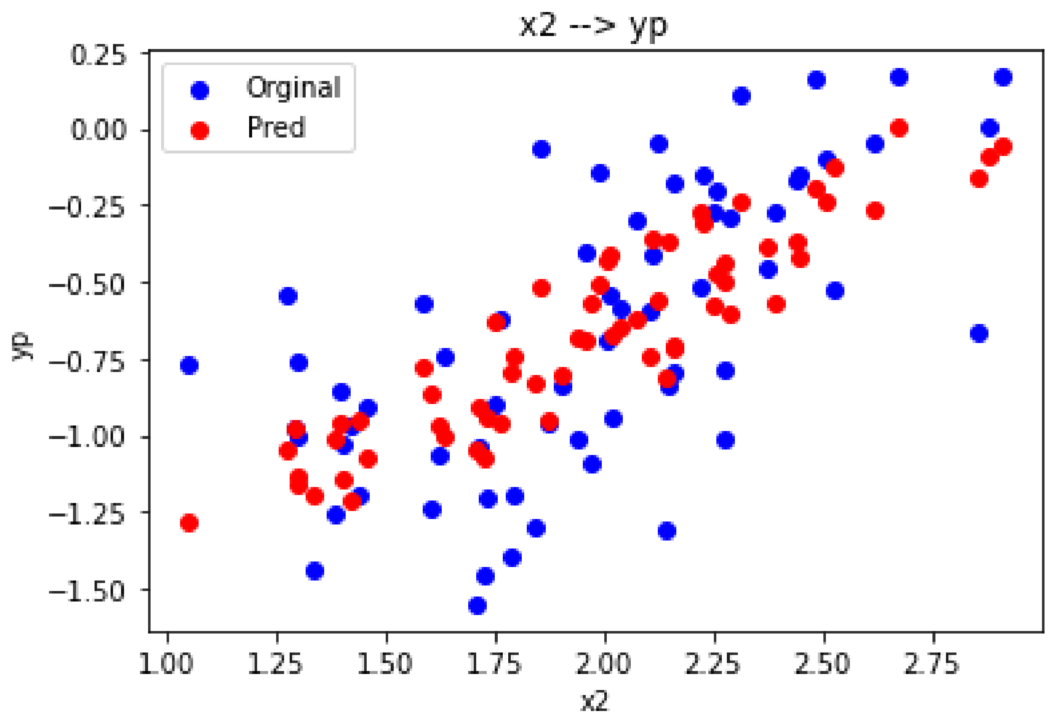
[80]:
```
fig1, ax1 = plt.subplots()
ax1.scatter(x.T[0], y, c ='blue', label = 'Orginal')
ax1.scatter(x.T[0], yp, c='red', label = 'Pred')
ax1.set(xlabel='x1', ylabel='yp', title='x1 --> yp')
ax1.legend()
```

[80]: <matplotlib.legend.Legend at 0x7fb36f0883a0>



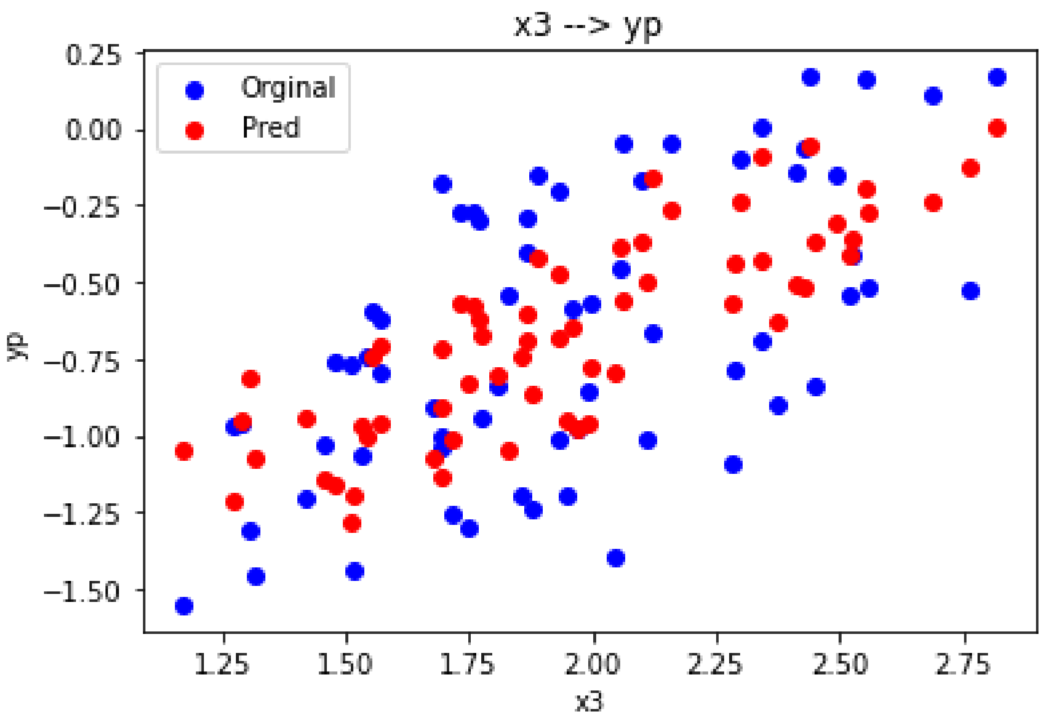
[81]:
```
fig2, ax2 = plt.subplots()
ax2.scatter(x.T[1], y, c ='blue', label = 'Orginal')
ax2.scatter(x.T[1], yp, c='red', label = 'Pred')
ax2.set(xlabel='x2', ylabel='yp', title='x2 --> yp')
ax2.legend()
```

[81]: <matplotlib.legend.Legend at 0x7fb36eda2460>



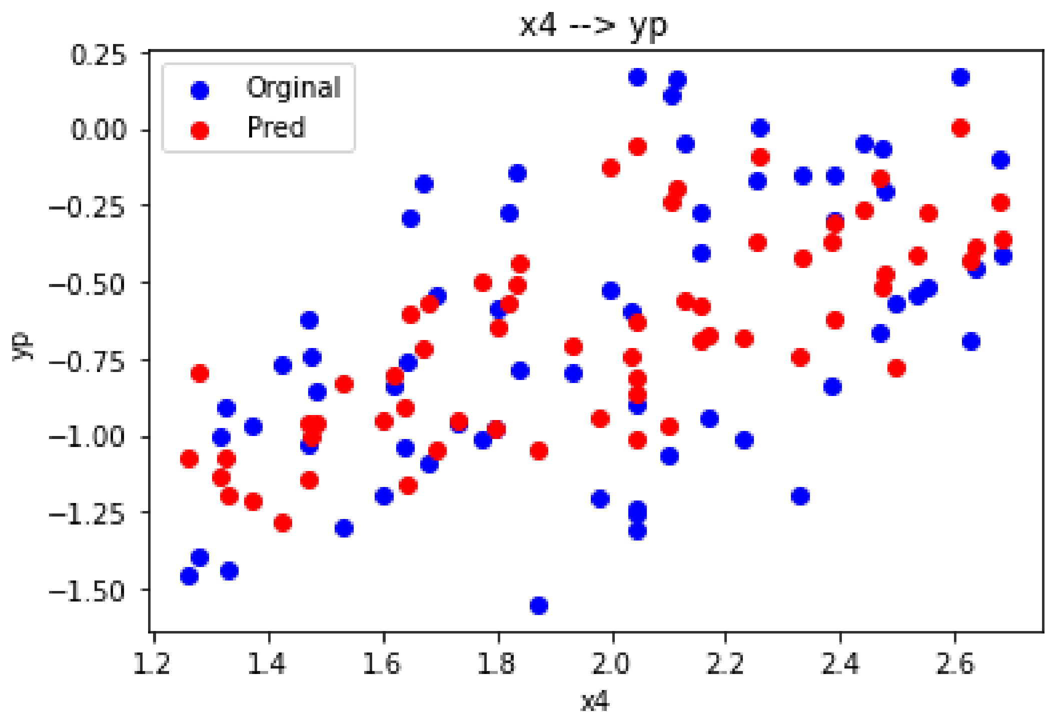
[82]:
```
fig3, ax3 = plt.subplots()
ax3.scatter(x.T[2], y, c ='blue', label = 'Orginal')
ax3.scatter(x.T[2], yp, c='red', label = 'Pred')
ax3.set(xlabel='x3', ylabel='yp', title='x3 --> yp')
ax3.legend()
```

[82]: <matplotlib.legend.Legend at 0x7fb36f1133d0>



[83]:
```
fig4, ax4 = plt.subplots()
ax4.scatter(x.T[3], y, c ='blue', label = 'Orginal')
ax4.scatter(x.T[3], yp, c='red', label = 'Pred')
ax4.set(xlabel='x4', ylabel='yp', title='x4 --> yp')
ax4.legend()
```

[83]: <matplotlib.legend.Legend at 0x7fb36ee9b370>



1.5 4. Read in `mlnn/data/Credit.csv` with Pandas and build a Linear Regression model to predict Credit Rating (`Rating`). Use only the numeric columns in your model, but feel free to experiment which which columns you believe are better predicters of Credit Rating (Column `Rating`)

[84]:
```
import pandas as pd
import numpy as np
credit = pd.read_csv('../data/Credit.csv')
credit.head()
```

[84]:
```
   Unnamed: 0   Income  Limit  Rating  Cards  Age  Education  Gender Student  \
0           1   14.891   3606     283      2   34         11    Male      No
1           2  106.025   6645     483      3   82         15  Female     Yes
2           3  104.593   7075     514      4   71         11    Male      No
3           4  148.924   9504     681      3   36         11  Female      No
4           5   55.882   4897     357      2   68         16    Male      No

  Married  Ethnicity  Balance
0     Yes  Caucasian      333
1     Yes      Asian      903
2      No      Asian      580
3      No      Asian      964
4     Yes  Caucasian      331
```

1.6 Choose multiple columns as inputs beyond `Income` and `Limit` but clearly, don't use `Rating`

[85]:
```
columns = ['Income', 'Limit', 'Age', 'Education']
X = credit[columns].values

X = np.vstack([X.T, np.ones(len(X))]).T
X
```

[85]:
```
array([[1.48910e+01, 3.60600e+03, 3.40000e+01, 1.10000e+01, 1.00000e+00],
       [1.06025e+02, 6.64500e+03, 8.20000e+01, 1.50000e+01, 1.00000e+00],
       [1.04593e+02, 7.07500e+03, 7.10000e+01, 1.10000e+01, 1.00000e+00],
       …,
       [5.78720e+01, 4.17100e+03, 6.70000e+01, 1.20000e+01, 1.00000e+00],
       [3.77280e+01, 2.52500e+03, 4.40000e+01, 1.30000e+01, 1.00000e+00],
       [1.87010e+01, 5.52400e+03, 6.40000e+01, 7.00000e+00, 1.00000e+00]])
```

[86]:
```
y = credit['Rating']
y
```

[86]:
```
0      283
1      483
2      514
3      681
4      357
      …
395    307
396    296
397    321
398    192
399    415
Name: Rating, Length: 400, dtype: int64
```

[87]:
```
left = np.linalg.inv(np.dot(X.T, X))
right = np.dot(y.T, X)
beta = np.dot(left, right)

income_pred = np.dot(X, beta)
income_pred
```

[87]:
```
array([279.55784695, 483.16981754, 512.86761648, 674.66571537,
       365.28650184, 577.61365527, 264.8812149 , 515.86848491,
       259.18192022, 491.99265106, 579.70166982, 125.71374961,
       395.8207332 , 501.8239206 , 258.83654504, 206.82327789,
       286.24857756, 330.73873099, 465.6751085 , 481.98443129,
       228.88140885, 463.3290974 , 215.60740247, 386.42867605,
       155.52186342, 325.45818626, 281.18033018, 340.22246323,
```

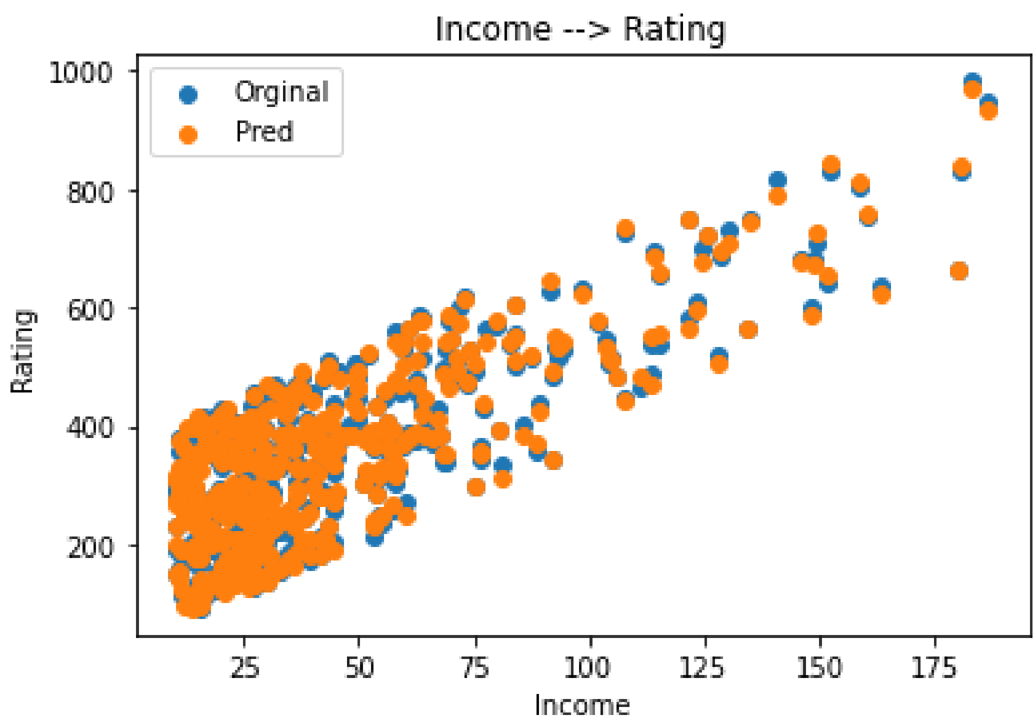
```
934.79156902, 412.05545344, 419.29969488, 220.213408  ,
563.0629752 , 161.76983093, 214.55712916, 209.90087447,
470.98013232, 472.62854708, 299.15859979, 269.15990555,
260.75262315, 550.49495945, 357.10471331, 455.27194238,
462.40323495, 544.52803582, 374.29097082, 334.00216811,
189.37850016, 343.82828977, 383.92999439, 301.06002944,
401.88589004, 404.41420844, 140.19187678, 160.20351018,
354.55376066, 358.38226613, 269.85028796, 391.58745705,
382.98407589, 246.08280397, 153.18277924, 237.92392277,
234.91668031, 315.28701262, 686.75255011, 377.97594339,
413.4043518 , 493.54558549, 303.41229611, 533.71558745,
365.35569439, 339.3996829 , 398.92166589, 249.74251709,
258.04104544, 254.82832557, 484.94853985, 179.0001237 ,
270.10207039, 323.66391204, 335.36967104, 134.0374497 ,
234.16265711, 845.09111786, 460.97570837, 190.79180939,
327.70209438, 541.84740972, 422.06184254, 440.83667548,
227.98647093, 399.56394039, 243.51396563,  94.63764046,
396.3415233 , 264.08068167, 236.82451026, 604.19897611,
288.46632154, 200.59686104, 544.04925034, 676.31171075,
357.00628309, 249.46330345, 128.00887203, 252.66142515,
442.41503905, 253.99301475, 256.09131342, 237.24646416,
481.10058438, 465.20664913, 259.76910316, 363.31805854,
179.69315724, 645.87581695, 181.70294617, 132.79191075,
132.37631754, 585.26102867, 511.13368527, 125.84763641,
207.69082369, 206.88440608, 407.08180798, 268.0886098 ,
598.2172571 , 269.09334757, 294.14951725, 141.39706727,
401.67136581, 428.30296183, 427.88568447, 272.3317319 ,
315.93300688, 281.07723186, 178.41108531, 734.89912804,
442.02734624, 490.82467027, 533.8534712 , 366.47765021,
221.44351555, 350.0465315 , 378.85413527, 138.35589057,
201.30618109,  98.95161787, 420.6016183 , 361.48756862,
180.87080975, 345.87945663, 248.52435222, 130.87783238,
325.17783827, 406.92429672, 410.15208707, 239.60855391,
364.01654252, 153.23824024, 541.81637206, 193.8673833 ,
438.3903074 , 341.31178409, 230.28513624, 193.72281744,
227.40037111, 450.81294929, 175.49361223, 325.40565666,
351.50618582, 355.56006429, 750.75426563, 182.46797235,
210.85596589, 302.6066164 , 332.28931862, 540.20439316,
277.08856538, 385.53994047, 469.01118409, 311.52135001,
812.22918055, 334.42706786, 292.96192504, 184.20908692,
552.16907668, 333.3157646 , 394.87485164, 676.3173879 ,
301.03190907, 710.65532691, 180.81404942, 397.25650267,
533.76245413, 303.9344568 , 171.54130614, 317.80038737,
393.19662144, 528.11354694, 135.80485401, 490.57831873,
395.30210309, 296.83790672, 202.97424146, 332.14708689,
327.13312188, 652.66889869, 253.33469066, 391.71133021,
329.58366431, 386.69447391, 390.89913514, 319.78532163,
```

```
221.59823277, 398.2836128 , 150.12521137, 387.7807416 ,
423.57062977, 625.10153851, 453.93384093, 345.63293315,
563.85584819, 415.75656399, 499.66044248, 413.45196353,
351.22091591, 544.98283262, 382.98903466, 357.21145268,
357.39284285, 191.82918346, 588.49723171, 231.51747786,
371.99083271, 381.86496623, 231.27658891, 275.88620285,
262.60587438,  99.82899519, 119.61817488, 480.92472675,
157.90969481, 171.6671307 , 252.86212695, 185.34091662,
 98.16107883, 142.38399059, 199.13705301, 253.21868409,
614.901372  , 385.93203372, 467.6070625 , 322.57587769,
157.53392955, 205.00460426, 209.49325053, 453.4296915 ,
385.82070633, 663.91694513, 308.97549106, 274.20810973,
380.48984264, 373.2858955 , 369.89198575, 426.68523232,
130.11791606, 408.40601349, 244.32796931, 364.22932731,
283.92345129, 352.1340914 , 432.13228715, 622.81234696,
270.0564343 , 372.84203667, 507.22783833, 245.58752794,
393.98437476, 162.45950498, 577.99898421, 464.80875566,
178.06469037, 147.9317333 , 141.81560928, 248.22924256,
389.17412968, 300.10565541, 255.95292486, 283.4384346 ,
378.03492508, 788.31295226, 208.03999926, 133.15539188,
380.92543096, 330.70726975, 215.90245522, 376.80685244,
346.34906797, 275.91537092, 370.35480379, 371.80923052,
540.38305976, 167.06752514, 292.39879889, 297.61710708,
350.03261047, 504.96991883, 364.17263789, 400.24634724,
388.31152257, 551.05146582, 658.32068818, 298.13250962,
526.06055424, 353.24922785, 138.72176063, 210.62610498,
116.95299555, 244.31893786, 271.99692186, 968.12625112,
229.04655872, 377.5472207 , 721.89175036, 483.35097553,
283.2428234 , 543.85030335, 340.84767639, 302.79578413,
388.07403732, 264.80207223, 355.59177512, 267.69600895,
431.59080914,  95.96693263, 392.25498017, 725.46375159,
294.17408051, 299.47446008, 236.92408664, 287.03829477,
386.64273228, 138.33121257, 395.84283201, 756.39668066,
114.67994231, 382.71982247, 143.7306246 , 383.93022121,
515.59515395, 351.90330438, 295.1891842 , 840.59090875,
444.35413027, 208.07866868, 327.08886746, 337.69939507,
433.69110249, 367.85396783, 377.24157591, 449.55000392,
694.60753026, 469.52303608, 563.81662545, 280.06081361,
425.91830272, 573.81889473, 448.10953053, 182.55344209,
289.96318175, 395.07552746, 360.5337389 , 415.85725936,
518.51561781, 144.69646326, 365.87698987, 231.90535882,
557.92144158, 576.60684597, 406.14962122, 258.71063413,
163.26831386, 414.36927016, 285.55935427, 131.0636467 ,
494.71401466, 511.31863084, 745.5405744 , 475.80969948,
193.44780137, 128.35840717, 424.56989858, 311.70398152,
293.63865727, 318.27479853, 207.52403116, 409.26377756])
```

1.6.1 5. Plot your results using scatter plots (just like in class). Show as many of your columns vs. credit rating that you can.

[88]:
```
fig_in, ax_in = plt.subplots()
ax_in.scatter(X.T[0], y, label = 'Orginal')
ax_in.scatter(X.T[0], income_pred, label = 'Pred')
ax_in.set(xlabel='Income', ylabel='Rating', title='Income --> Rating')
ax_in.legend()
```

[88]: <matplotlib.legend.Legend at 0x7fb36f556760>



[ ]: