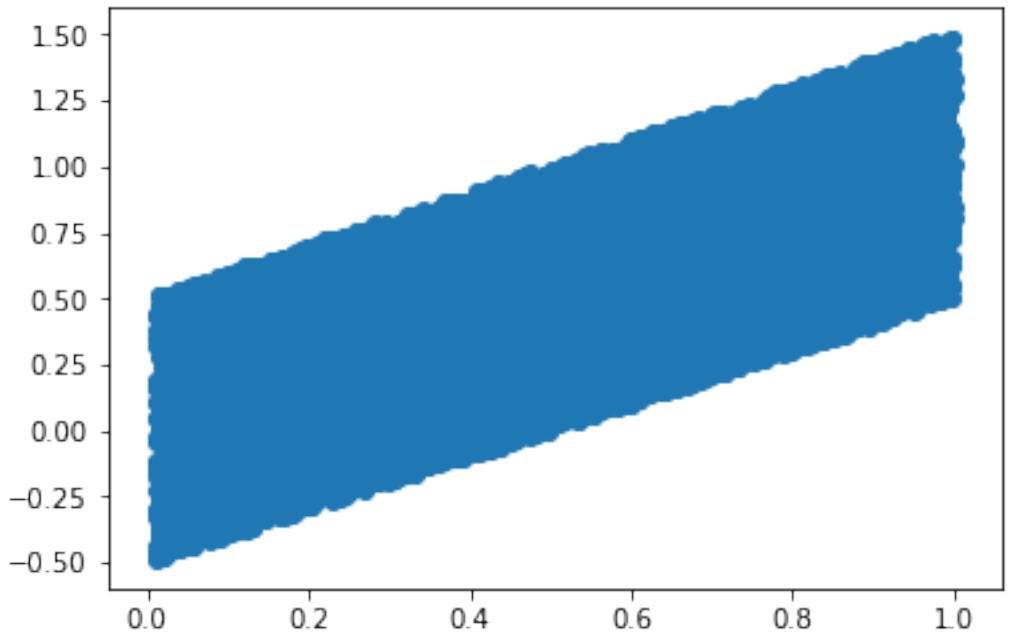
# 5-FINAL

November 7, 2018

In [1]:
```
from sklearn.linear_model import LinearRegression
import numpy as np
import matplotlib.pylab as plt
%matplotlib inline
from sklearn import linear_model
```

In [24]:
```
n = 10000
x = np.linspace(0.01, 1, n).reshape(-1, 1)
y = np.linspace(0.01, 1, n)  + np.random.rand(n) - .5


plt.scatter(x,y)
```

Out[24]: <matplotlib.collections.PathCollection at 0x2a0764fa4a8>



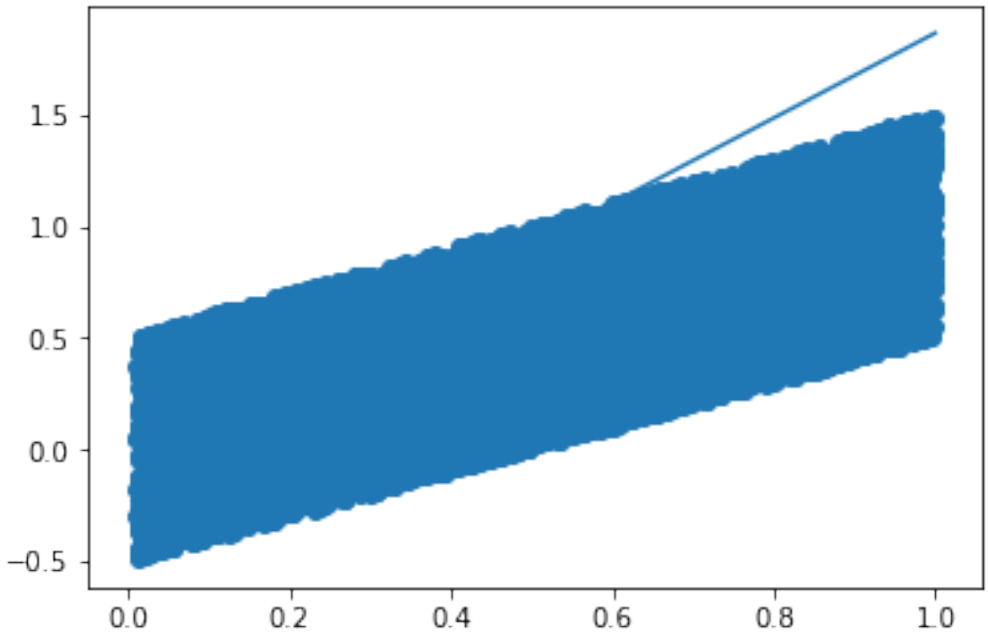
# 1 Assignment 5

## 1.1 1. Create and fit a Linear Regression Model

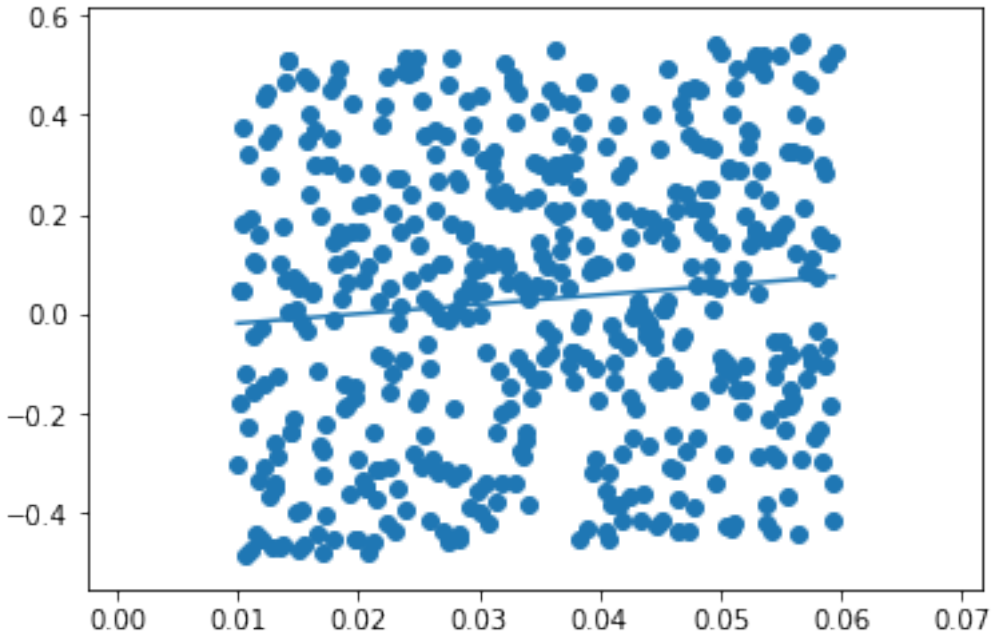
## 1.2 Calculate the Training error and Testing error using sklearn with a .50 split

For error, use `mean_squared`, but if you want to experiment with other mean errors, please do!

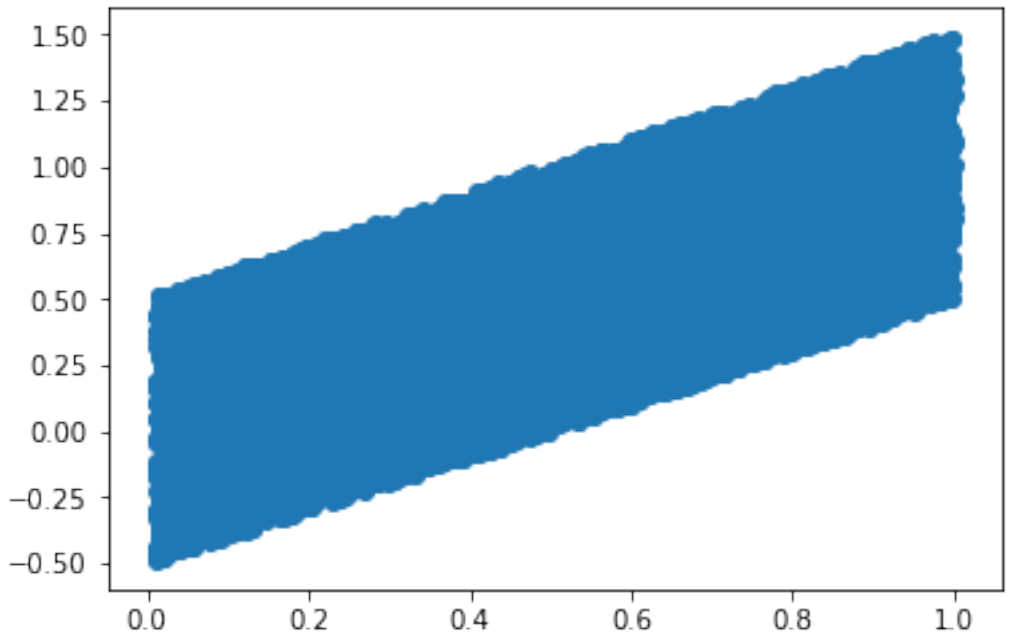
```
In [25]: model= LinearRegression()
         model.fit(x[:500], y[:500])
         model.coef_, model.intercept_

Out[25]: (array([1.90164331]), -0.037187125573919594)

In [26]: plt.scatter(x,y)
         plt.plot(x, np.dot(x,model.coef_) + model.intercept_)

Out[26]: [<matplotlib.lines.Line2D at 0x2a076528630>]
```



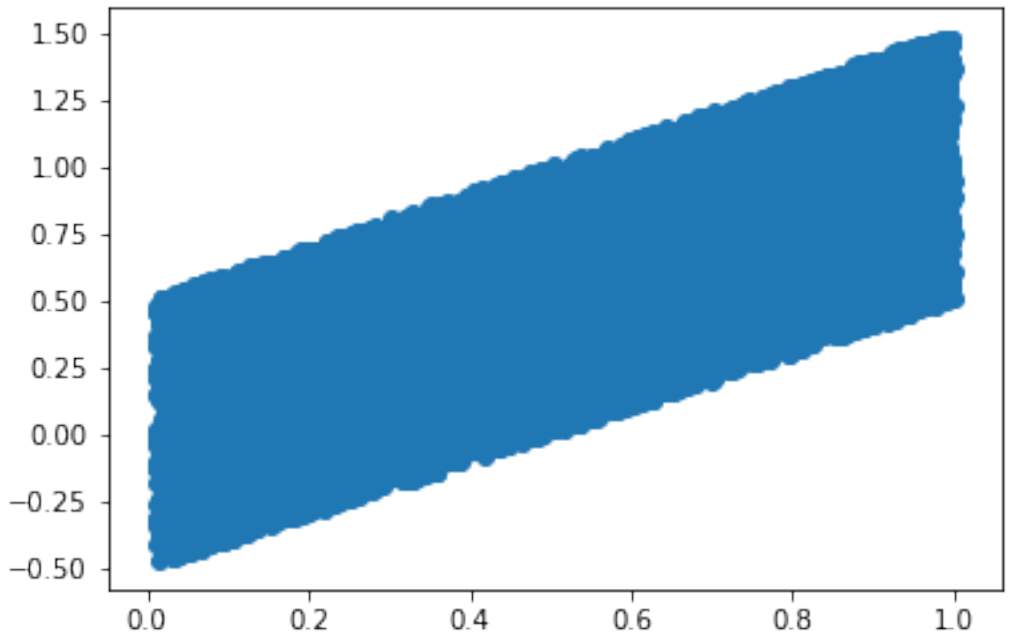

```
In [27]: plt.scatter(x[:500],y[:500])
         plt.plot(x[:500], np.dot(x[:500], model.coef_) + model.intercept_)

Out[27]: [<matplotlib.lines.Line2D at 0x2a07655d8d0>]
```




```
In [28]: def shuffle(a, b):
             assert len(a) == len(b)
             p = np.random.permutation(len(a))
             return p

In [29]: p = shuffle(x, y)

In [30]: p

Out[30]: array([8349, 4867, 7776, ..., 9075, 2807, 4911])

In [31]: x[p], y[p]

Out[31]: (array([[0.83663366],
                 [0.49188119],
                 [0.77990099],
                 ...,
                 [0.90851485],
                 [0.28792079],
                 [0.49623762]]),
          array([0.82911864, 0.93803698, 1.00946723, ..., 0.85325909, 0.60769757,
                 0.56167243]))

In [32]: plt.scatter(x[p],y[p])

Out[32]: <matplotlib.collections.PathCollection at 0x2a076616198>
```




```
In [33]: model = LinearRegression()
         model.fit(x[p][:500], y[p][:500])
         model.coef_, model.intercept_

Out[33]: (array([0.98756628]), -0.007338253751633661)

In [22]: plt.scatter(x,y)
         plt.plot(x, np.dot(x, model.coef_) + model.intercept_)

Out[22]: [<matplotlib.lines.Line2D at 0x2a0763abf60>]
```

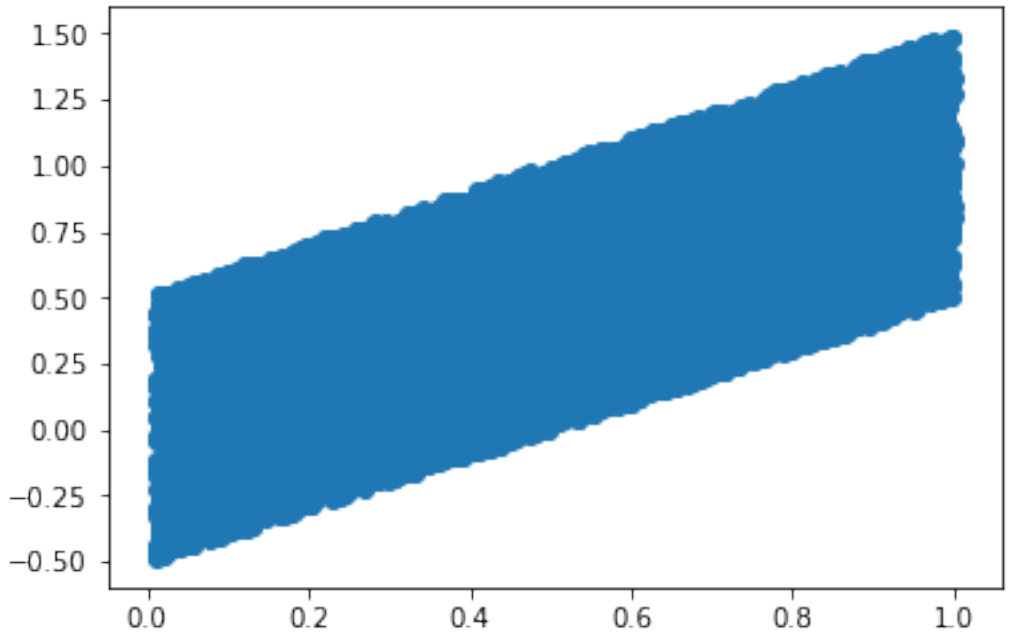



```
In [42]: x[500:]

Out[42]: array([[0.05950495],
                [0.05960396],
                [0.05970297],
                ...,
                [0.99980198],
                [0.99990099],
                [1.        ]])

In [43]: from sklearn.metrics import mean_squared_error

In [44]: mean_squared_error(y[p][500:], np.dot(x[p][500:], model.coef_) + model.intercept_)

Out[44]: 0.0839052218021657
```

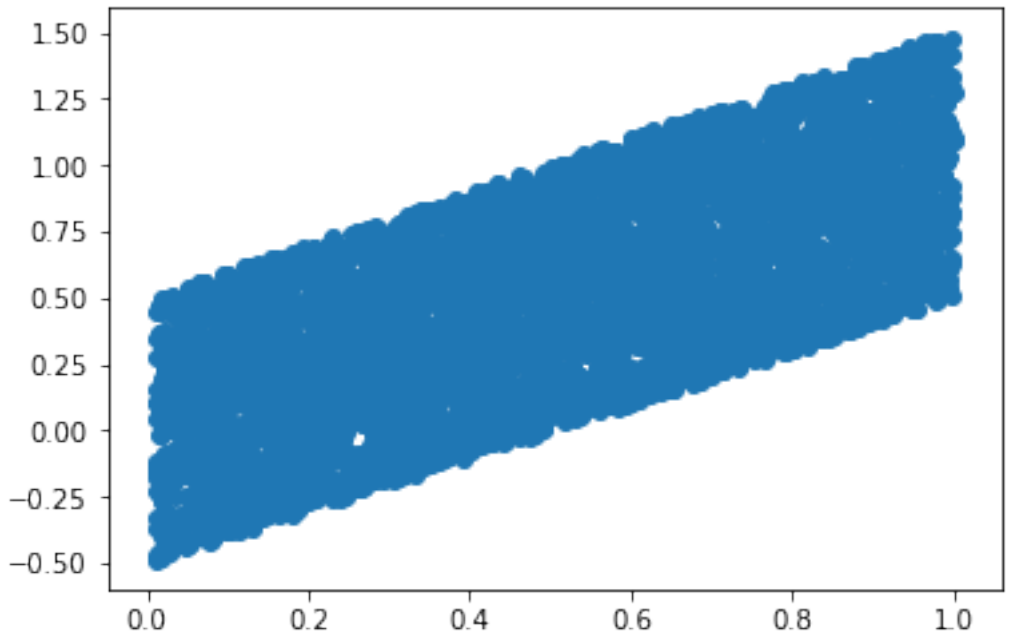
### 1.3 2. Repeat #1 for a Ridge Regression

```
In [45]: from sklearn.model_selection import train_test_split

In [46]: x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=.5)

In [47]: from sklearn.linear_model import Ridge

In [48]: model = Ridge()
         model.fit(x_train, y_train)
         model.coef_, model.intercept_
```

```
Out[48]: (array([1.00099457]), -0.007324920200704965)

In [49]: model = LinearRegression()
         model.fit(x_train, y_train)
         model.coef_, model.intercept_

Out[49]: (array([1.00343985]), -0.008561100269865007)

In [50]: plt.scatter(x,y)
         plt.plot(x, np.dot(x, model.coef_) + model.intercept_)

Out[50]: [<matplotlib.lines.Line2D at 0x2a076629da0>]
```




```
In [51]: mean_squared_error(y_test, np.dot(x_test, model.coef_) + model.intercept_)

Out[51]: 0.0831254978695135

In [52]: mean_squared_error(y_train, np.dot(x_train, model.coef_) + model.intercept_)

Out[52]: 0.08411865340717802

In [53]: plt.scatter(x_test,y_test)
         plt.plot(x_test, np.dot(x_test, model.coef_) + model.intercept_)

Out[53]: [<matplotlib.lines.Line2D at 0x2a076642c50>]
```




```
In [54]: mean_squared_error(y_test, np.dot(x_test, model.coef_) + model.intercept_)
```

```
Out[54]: 0.0831254978695135
```

```
In [55]: mean_squared_error(y_train, np.dot(x_train, model.coef_) + model.intercept_)
```

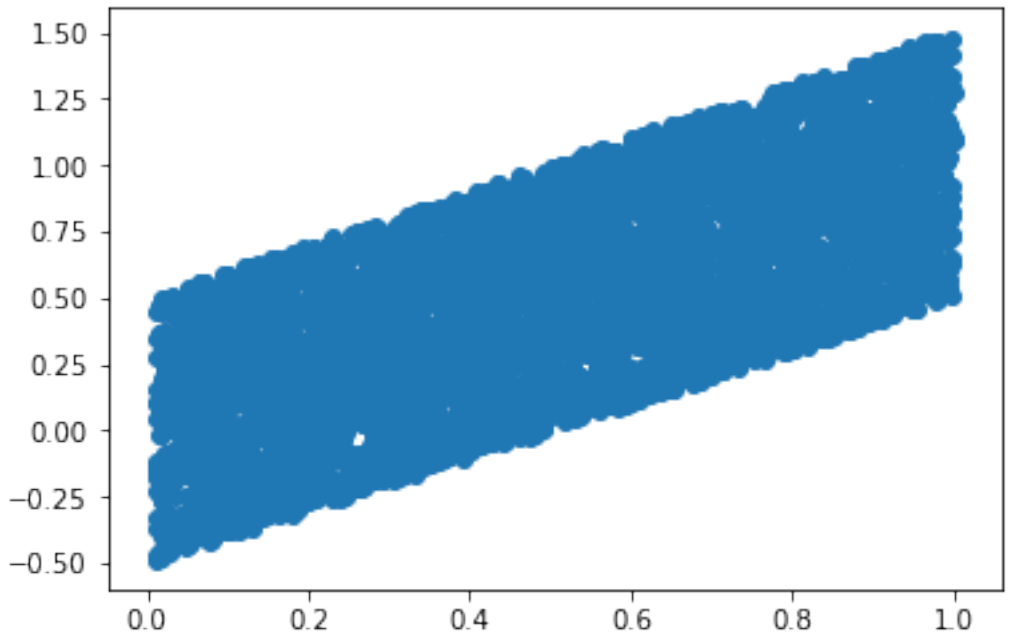
```
Out[55]: 0.08411865340717802
```

### 1.4 3. Vary the `split` size from .01 to .99 with at least 10 values (the more the merrier!). Plot the resulting Training error and Testing error vs. `split` size. Create separate plots for Linear and Ridge

```
In [ ]: x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=.01)
```

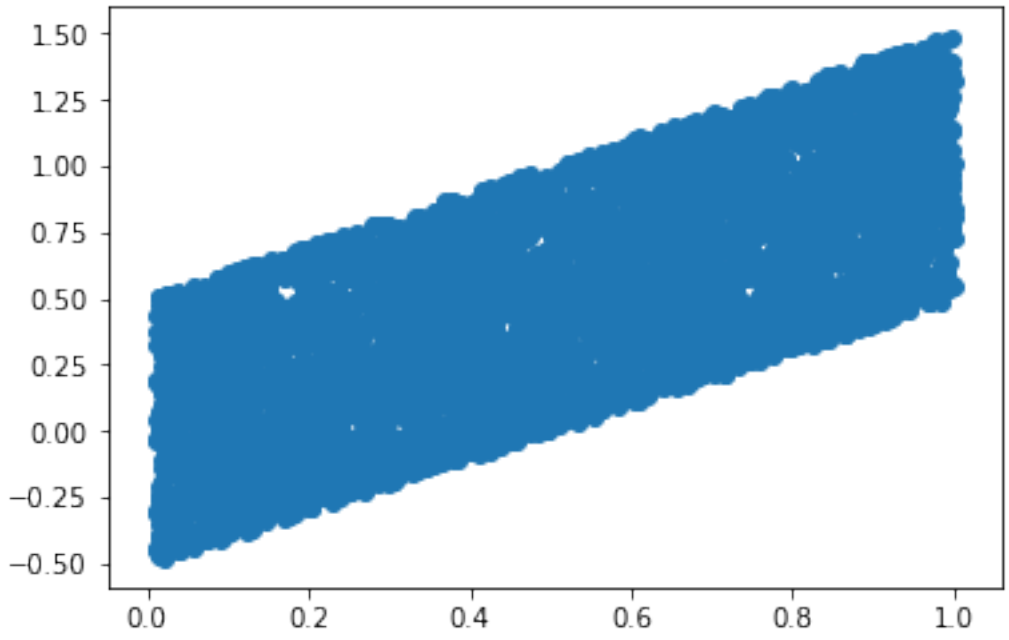
```
In [56]: model = Ridge()
         model.fit(x_train, y_train)
         model.coef_, model.intercept_

         model = LinearRegression()
         model.fit(x_train, y_train)
         model.coef_, model.intercept_

         plt.scatter(x_test,y_test)
         plt.plot(x_test, np.dot(x_test, model.coef_) + model.intercept_)
```

```
Out[56]: [<matplotlib.lines.Line2D at 0x2a07670d588>]
```




In [57]:
```
plt.scatter(x_train,y_train)
plt.plot(x_train, np.dot(x_train, model.coef_) + model.intercept_)
```

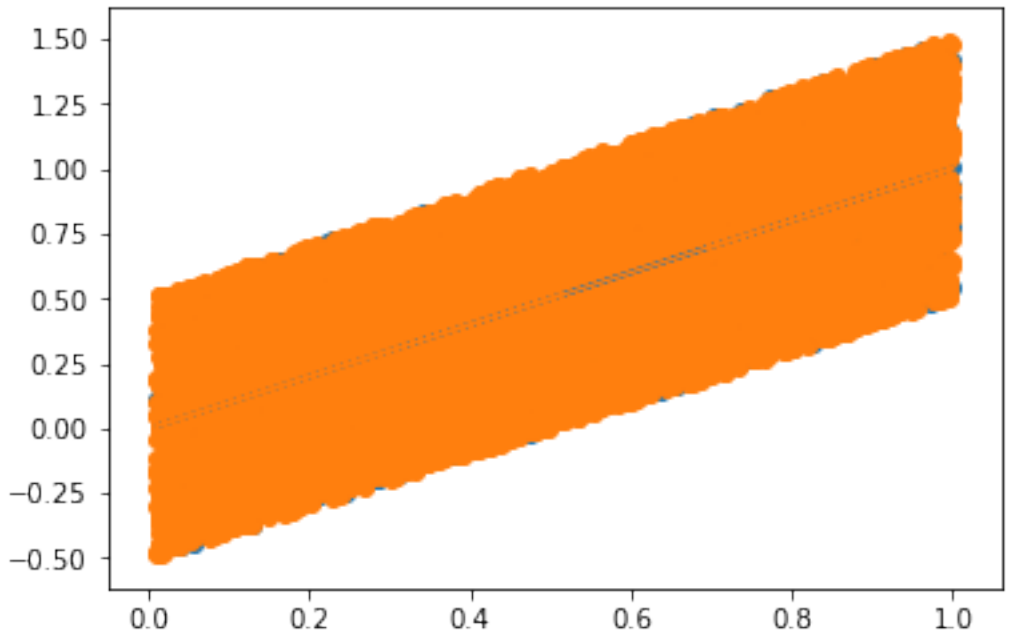
Out[57]: [<matplotlib.lines.Line2D at 0x2a0766f3b38>]



In [58]:
```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=.10)
```

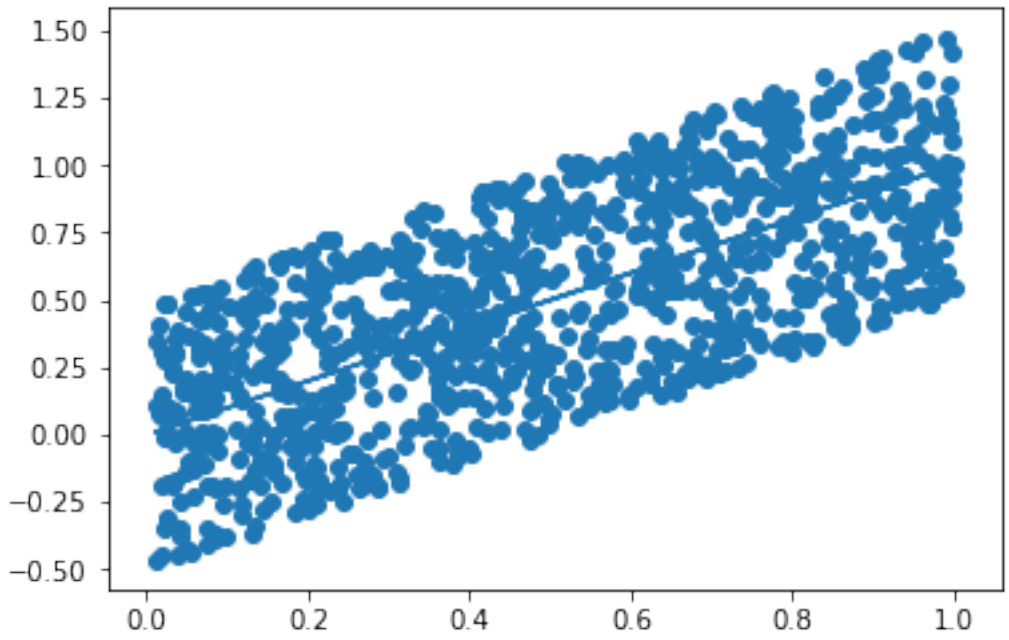
In [59]:
```
model = Ridge()
model.fit(x_train, y_train)
model.coef_, model.intercept_

model = LinearRegression()
model.fit(x_train, y_train)
model.coef_, model.intercept_

plt.scatter(x_test,y_test)
plt.plot(x_test, np.dot(x_test, model.coef_) + model.intercept_)

plt.scatter(x_train,y_train)
plt.plot(x_train, np.dot(x_train, model.coef_) + model.intercept_)
```

Out[59]:
```
[<matplotlib.lines.Line2D at 0x2a077c925c0>]
```



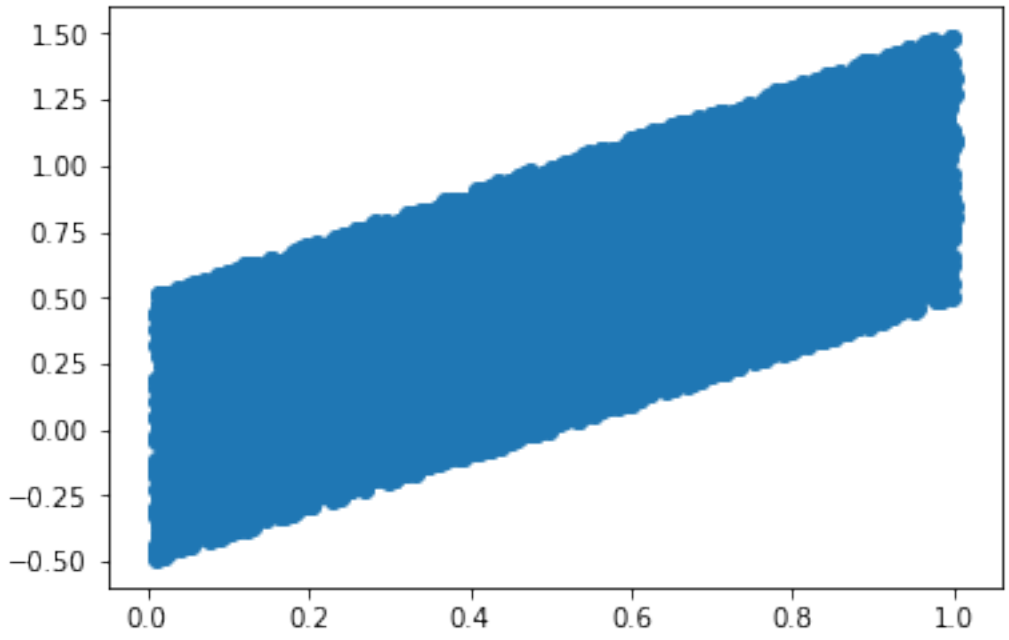

In [60]:
```
plt.scatter(x_test,y_test)
plt.plot(x_test, np.dot(x_test, model.coef_) + model.intercept_)
```

Out[60]:
```
[<matplotlib.lines.Line2D at 0x2a077cbc240>]
```




```
In [61]: plt.scatter(x_train,y_train)
         plt.plot(x_train, np.dot(x_train, model.coef_) + model.intercept_)
```

```
Out[61]: [<matplotlib.lines.Line2D at 0x2a077d25908>]
```



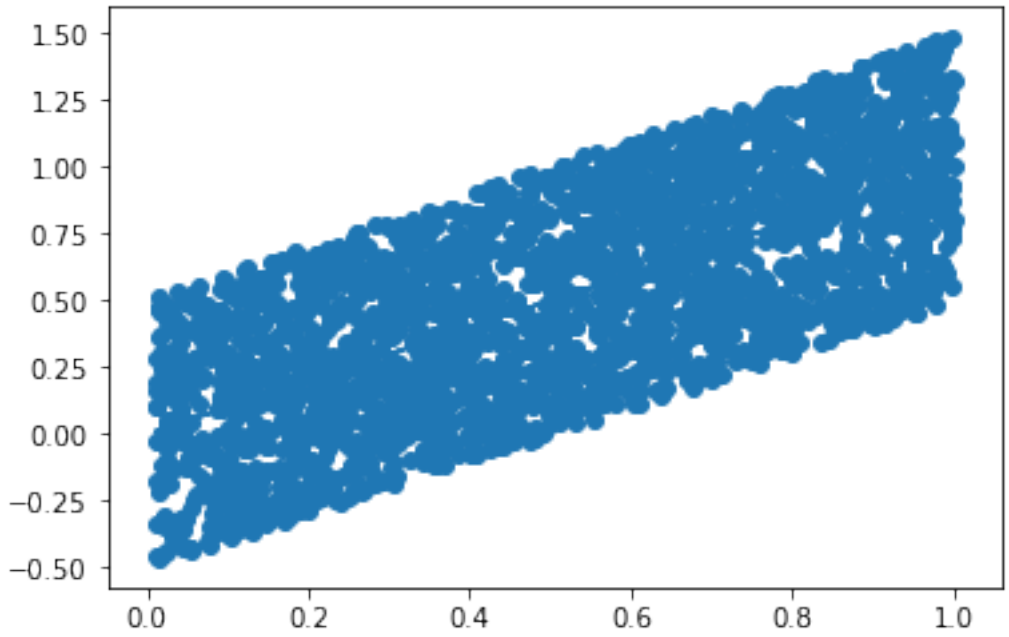
In [62]:
```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=.25)
```

In [63]:
```
model = Ridge()
model.fit(x_train, y_train)
model.coef_, model.intercept_

model = LinearRegression()
model.fit(x_train, y_train)
model.coef_, model.intercept_
```

Out[63]:
```
(array([0.99648997]), 0.0008303825746382998)
```

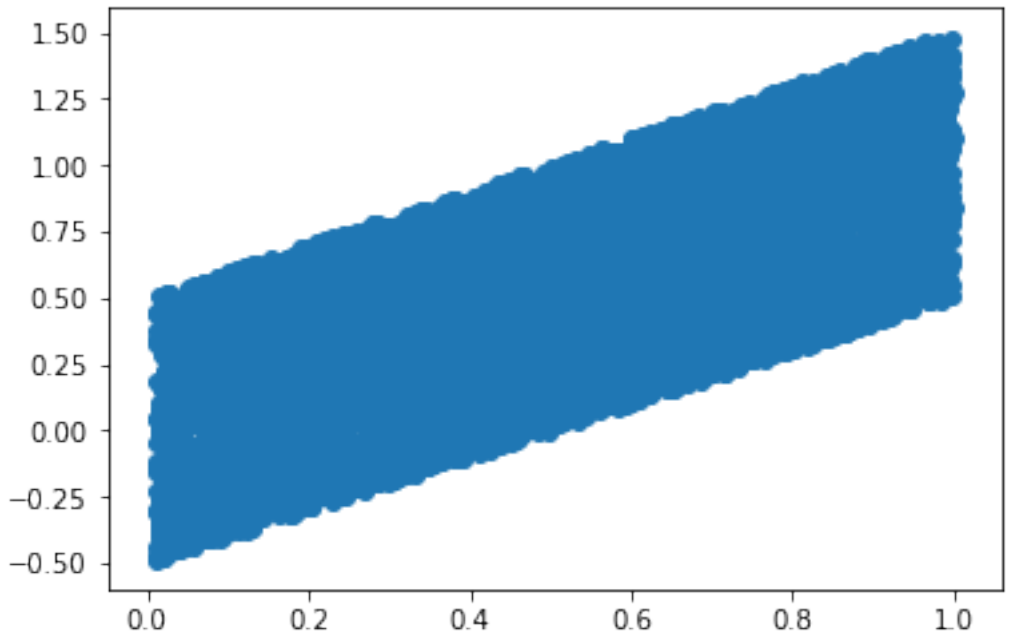
In [64]:
```
plt.scatter(x_test,y_test)
plt.plot(x_test, np.dot(x_test, model.coef_) + model.intercept_)
```

Out[64]:
```
[<matplotlib.lines.Line2D at 0x2a077d842b0>]
```

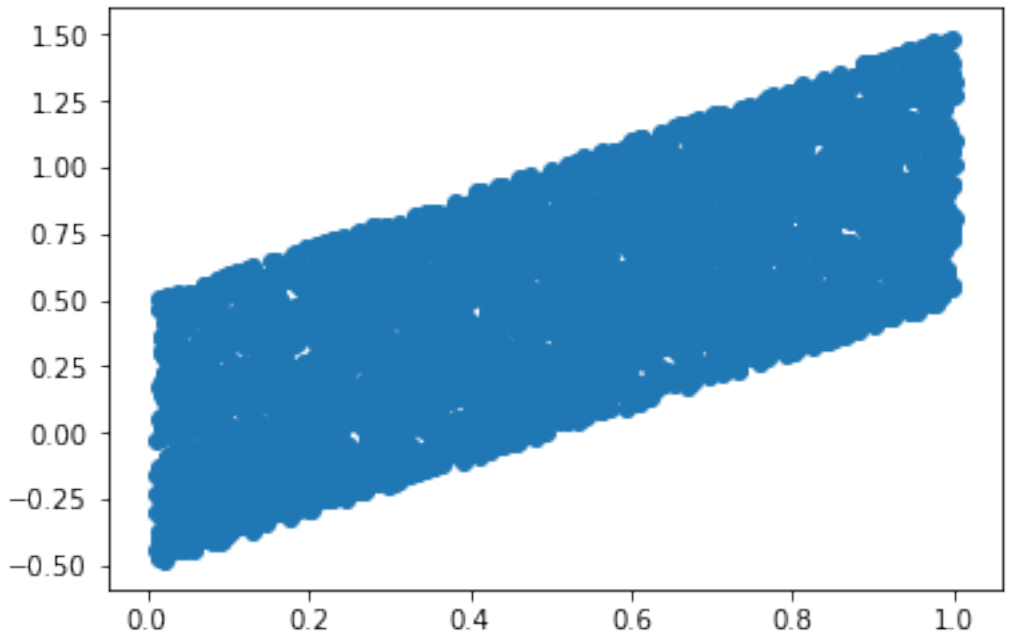



In [65]:
```
plt.scatter(x_train,y_train)
plt.plot(x_train, np.dot(x_train, model.coef_) + model.intercept_)
```

Out[65]:
```
[<matplotlib.lines.Line2D at 0x2a077dec588>]
```

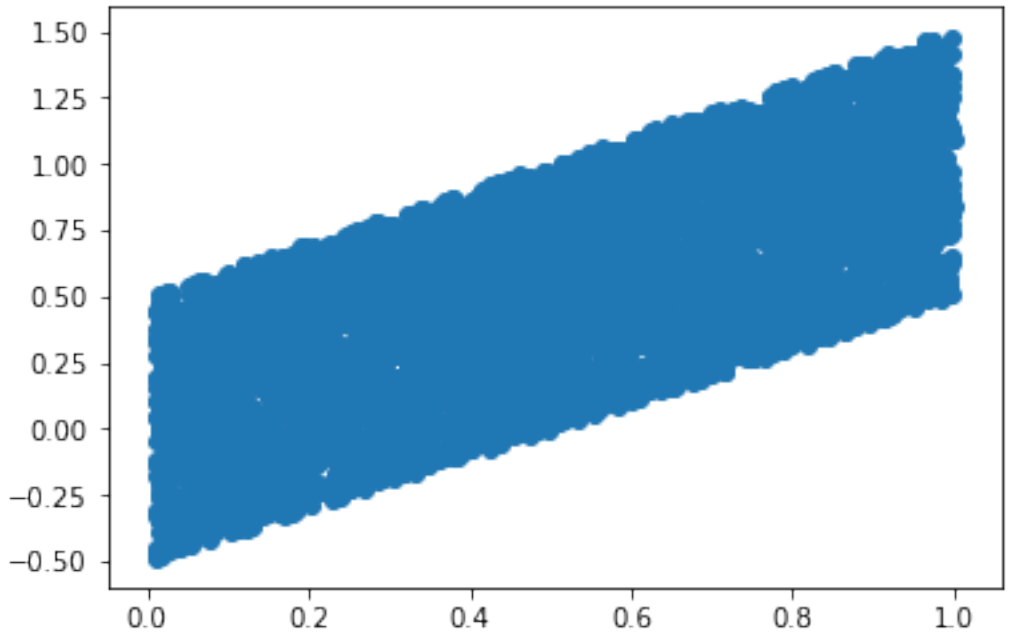



```
In [66]: x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=.45)

In [67]: model = Ridge()
         model.fit(x_train, y_train)
         model.coef_, model.intercept_

         model = LinearRegression()
         model.fit(x_train, y_train)
         model.coef_, model.intercept_

Out[67]: (array([0.99266303]), 0.0036758251788197027)

In [68]: plt.scatter(x_test,y_test)
         plt.plot(x_test, np.dot(x_test, model.coef_) + model.intercept_)

Out[68]: [<matplotlib.lines.Line2D at 0x2a077e2d0f0>]
```




In [69]:
```
plt.scatter(x_train,y_train)
plt.plot(x_train, np.dot(x_train, model.coef_) + model.intercept_)
```

Out[69]: [<matplotlib.lines.Line2D at 0x2a077eadcf8>]



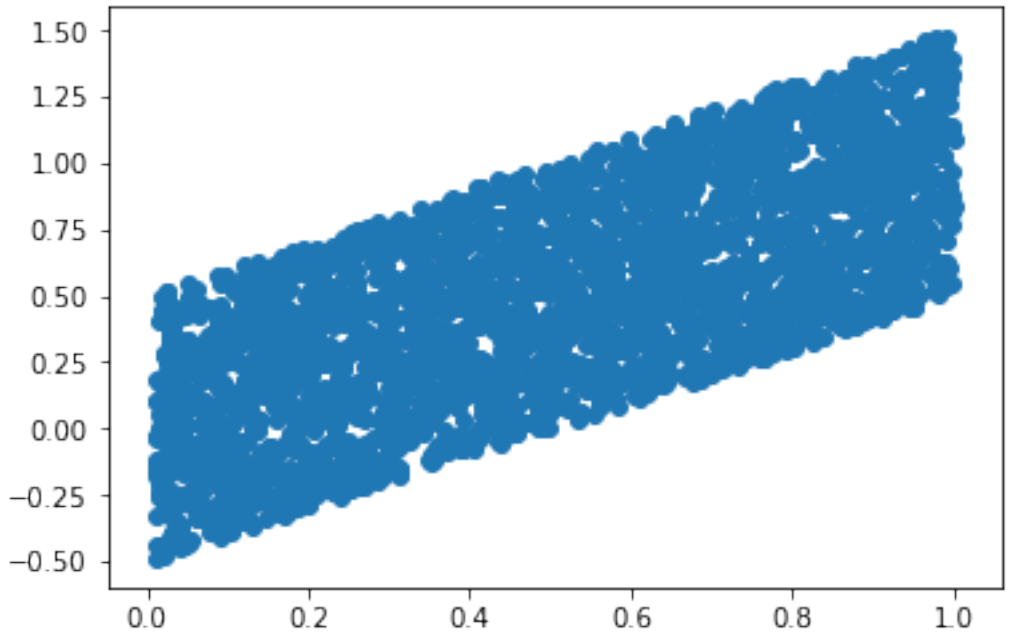
In [70]:
```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=.30)
```

In [71]:
```
model = Ridge()
model.fit(x_train, y_train)
model.coef_, model.intercept_

model = LinearRegression()
model.fit(x_train, y_train)
model.coef_, model.intercept_
```

Out[71]:
```
(array([1.00137286]), 0.0011350146406943207)
```

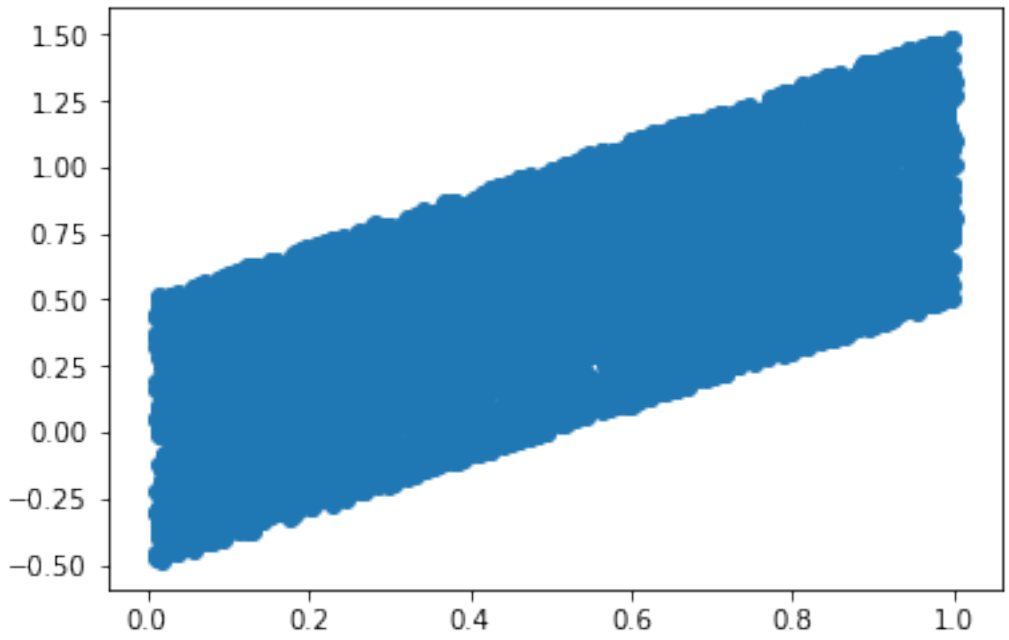
In [72]:
```
plt.scatter(x_test,y_test)
plt.plot(x_test, np.dot(x_test, model.coef_) + model.intercept_)
```

Out[72]:
```
[<matplotlib.lines.Line2D at 0x2a077d0beb8>]
```

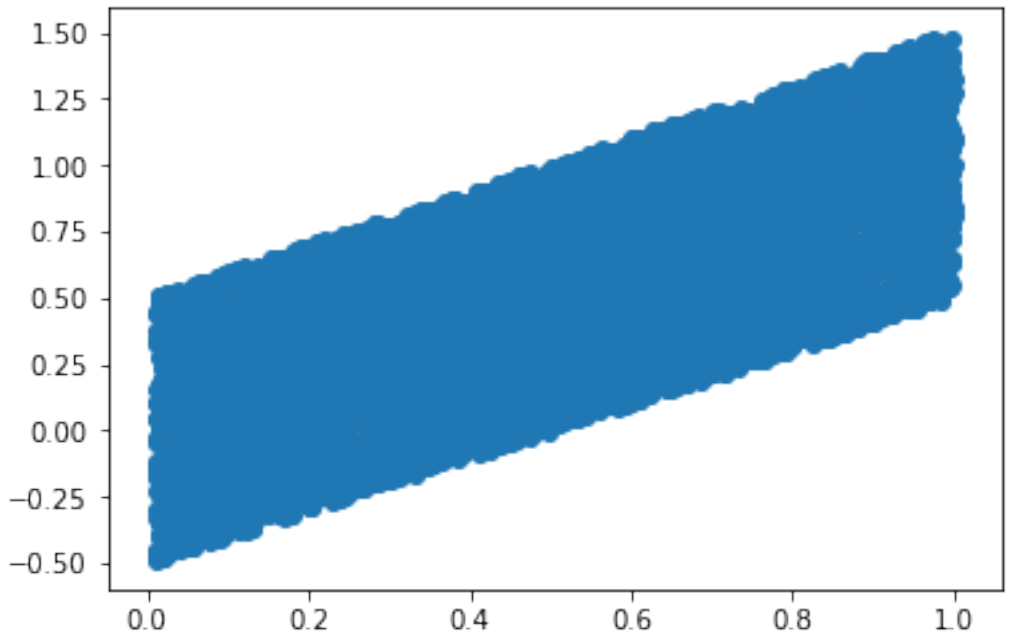



In [73]:
```
plt.scatter(x_train,y_train)
plt.plot(x_train, np.dot(x_train, model.coef_) + model.intercept_)
```

Out[73]:
```
[<matplotlib.lines.Line2D at 0x2a077ef5320>]
```

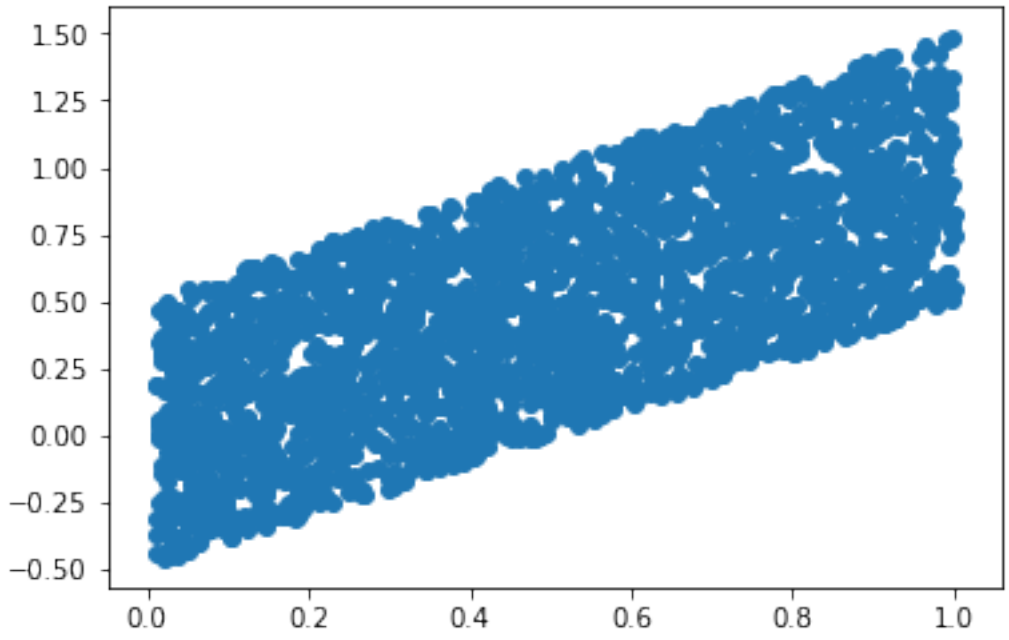



```
In [74]: x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=.75)

In [75]: model = Ridge()
         model.fit(x_train, y_train)
         model.coef_, model.intercept_

         model = LinearRegression()
         model.fit(x_train, y_train)
         model.coef_, model.intercept_

Out[75]: (array([0.98500536]), 0.002216741199578043)

In [76]: plt.scatter(x_test,y_test)
         plt.plot(x_test, np.dot(x_test, model.coef_) + model.intercept_)

Out[76]: [<matplotlib.lines.Line2D at 0x2a077f11e10>]
```

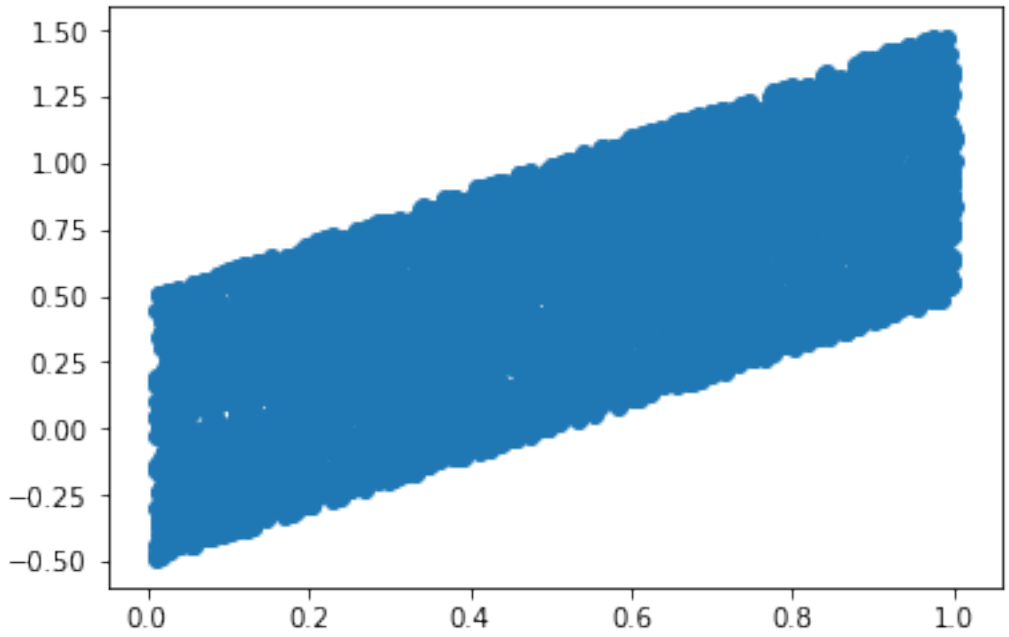



```
In [77]: plt.scatter(x_train,y_train)
         plt.plot(x_train, np.dot(x_train, model.coef_) + model.intercept_)
```

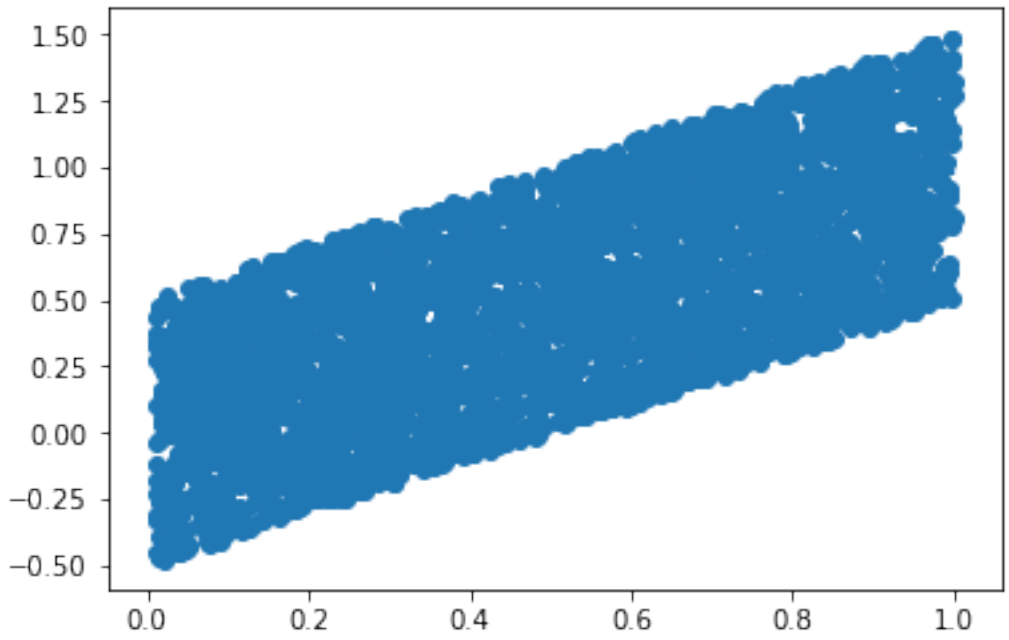
```
Out[77]: [<matplotlib.lines.Line2D at 0x2a07803dcc0>]
```



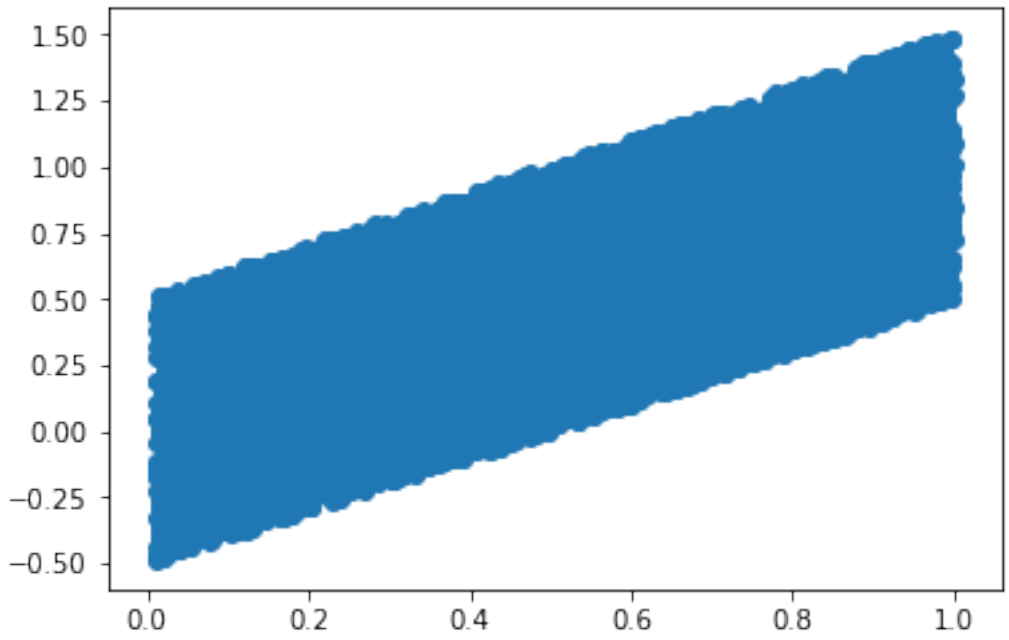
```
In [78]: x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=.60)

In [79]: model = Ridge()
         model.fit(x_train, y_train)
         model.coef_, model.intercept_

         model = LinearRegression()
         model.fit(x_train, y_train)
         model.coef_, model.intercept_

Out[79]: (array([0.97193257]), 0.008861328315733608)

In [80]: plt.scatter(x_test,y_test)
         plt.plot(x_test, np.dot(x_test, model.coef_) + model.intercept_)

Out[80]: [<matplotlib.lines.Line2D at 0x2a077fe6da0>]
```




```
In [81]: plt.scatter(x_train,y_train)
         plt.plot(x_train, np.dot(x_train, model.coef_) + model.intercept_)

Out[81]: [<matplotlib.lines.Line2D at 0x2a0780e3588>]
```

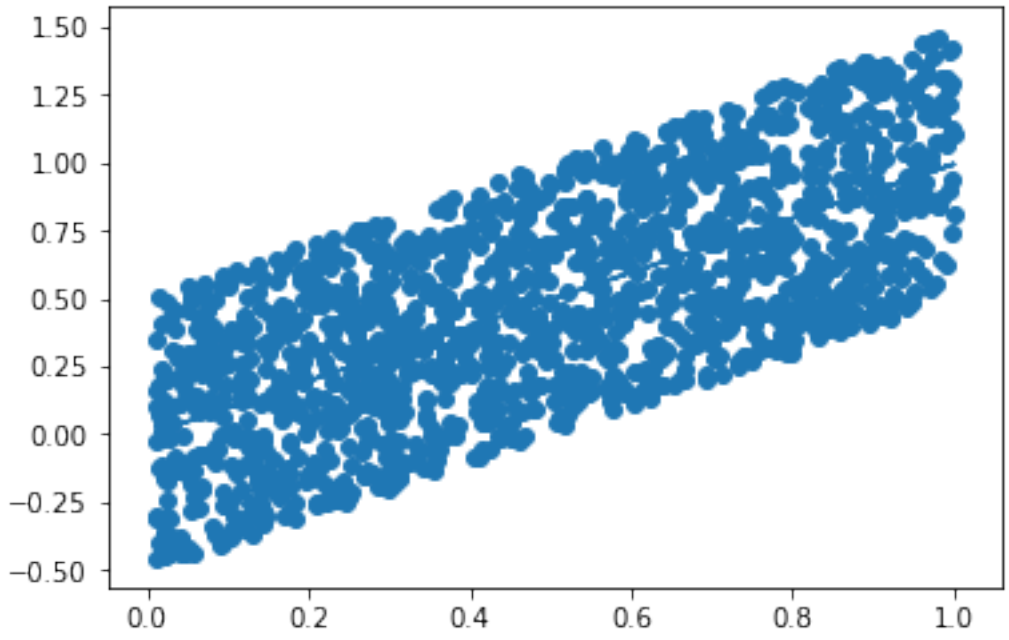



```
In [82]: x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=.85)

In [83]: model = Ridge()
         model.fit(x_train, y_train)
         model.coef_, model.intercept_

         model = LinearRegression()
         model.fit(x_train, y_train)
         model.coef_, model.intercept_

Out[83]: (array([0.9979618]), -0.0030101497311684766)

In [84]: plt.scatter(x_test,y_test)
         plt.plot(x_test, np.dot(x_test, model.coef_) + model.intercept_)

Out[84]: [<matplotlib.lines.Line2D at 0x2a078168a90>]
```

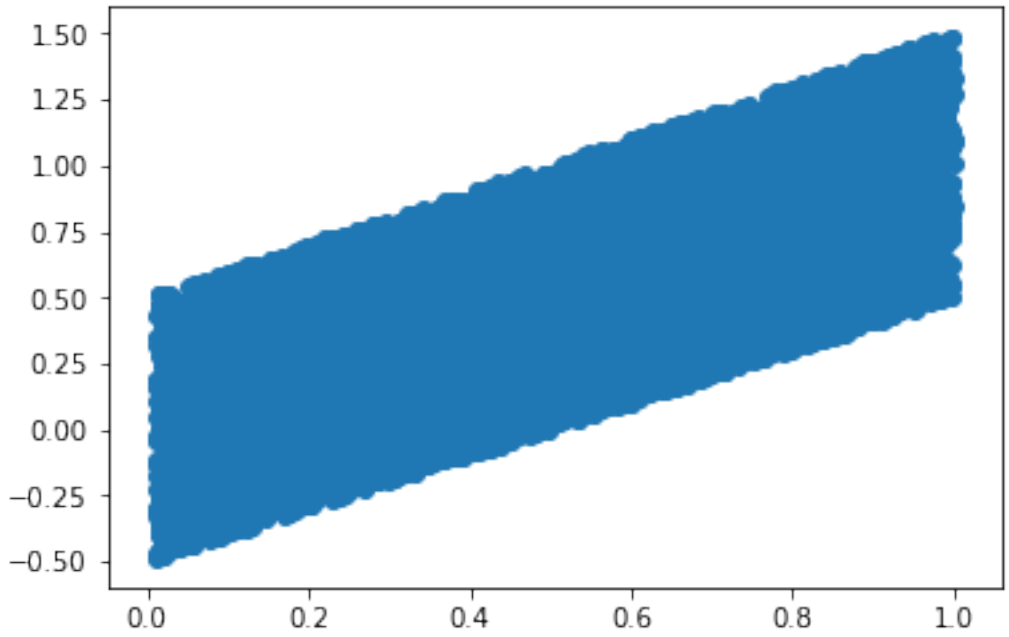



```
In [85]: plt.scatter(x_train,y_train)
         plt.plot(x_train, np.dot(x_train, model.coef_) + model.intercept_)
```

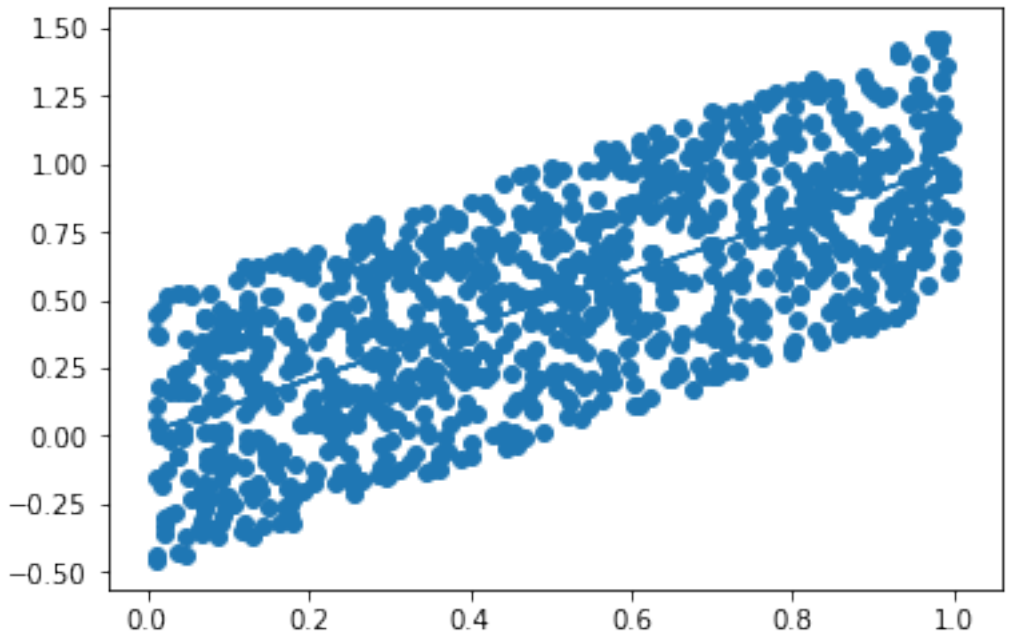
```
Out[85]: [<matplotlib.lines.Line2D at 0x2a078168fd0>]
```



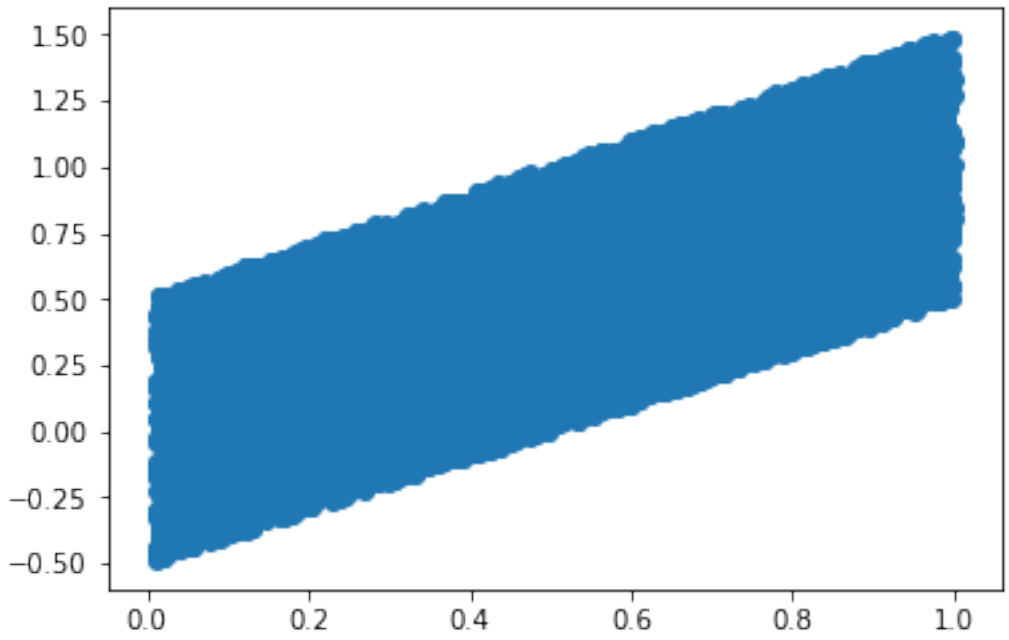
```
In [86]: x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=.90)

In [87]: model = Ridge()
         model.fit(x_train, y_train)
         model.coef_, model.intercept_

         model = LinearRegression()
         model.fit(x_train, y_train)
         model.coef_, model.intercept_

Out[87]: (array([0.98330901]), 0.011307609946766328)

In [88]: plt.scatter(x_test,y_test)
         plt.plot(x_test, np.dot(x_test, model.coef_) + model.intercept_)

Out[88]: [<matplotlib.lines.Line2D at 0x2a0791e7400>]
```




```
In [89]: plt.scatter(x_train,y_train)
         plt.plot(x_train, np.dot(x_train, model.coef_) + model.intercept_)

Out[89]: [<matplotlib.lines.Line2D at 0x2a07803d2e8>]
```

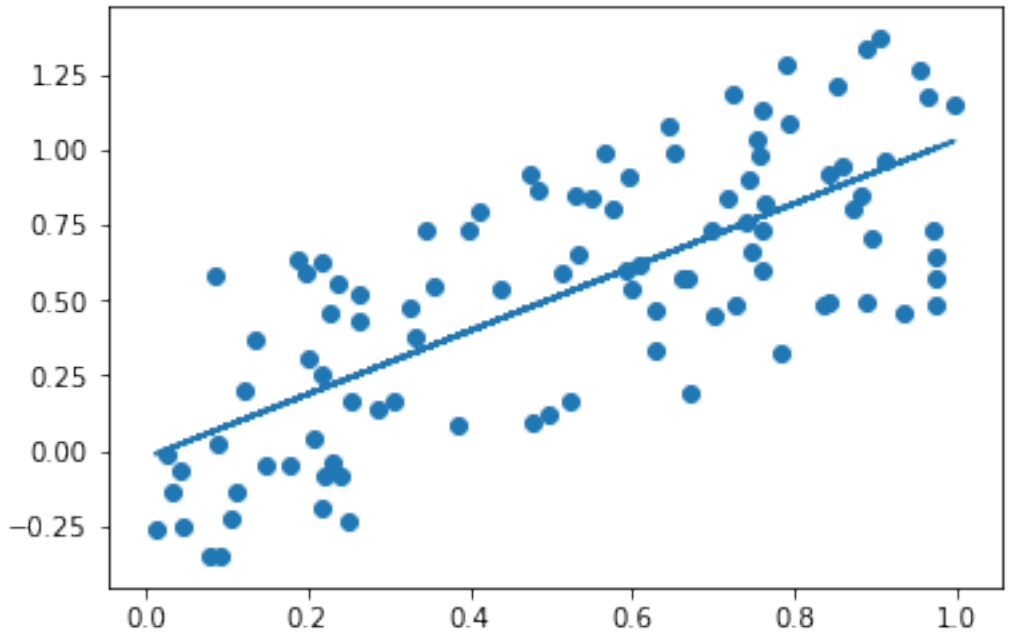



```
In [90]: x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=.99)

In [92]: model = Ridge()
         model.fit(x_train, y_train)
         model.coef_, model.intercept_

         model = LinearRegression()
         model.fit(x_train, y_train)
         model.coef_, model.intercept_

Out[92]: (array([1.05528512]), -0.02374212640883344)

In [93]: plt.scatter(x_test,y_test)
         plt.plot(x_test, np.dot(x_test, model.coef_) + model.intercept_)

Out[93]: [<matplotlib.lines.Line2D at 0x2a079270f98>]
```




```
In [94]: plt.scatter(x_train,y_train)
         plt.plot(x_train, np.dot(x_train, model.coef_) + model.intercept_)
```

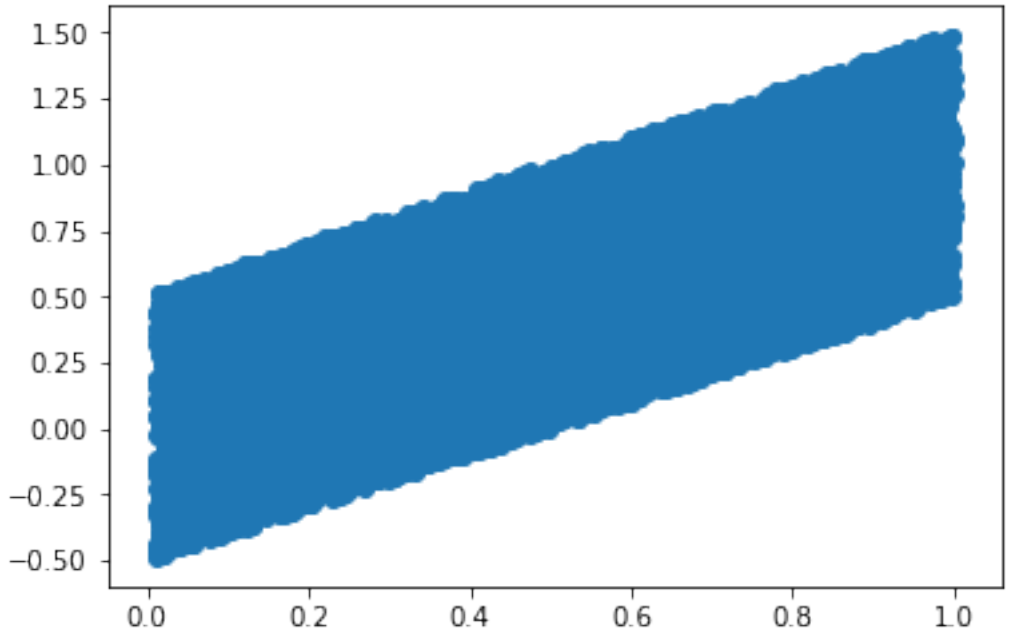
```
Out[94]: [<matplotlib.lines.Line2D at 0x2a0792500b8>]
```



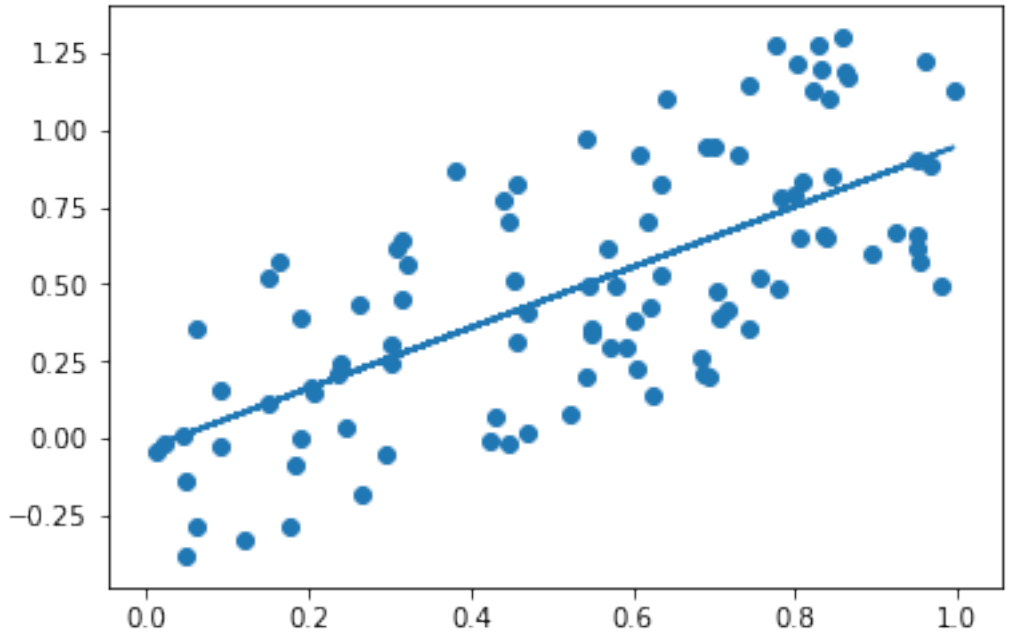
### 1.5 4. Chose an ideal split size based on the previous plot for Ridge.

### 1.6 Vary the Ridge parameter `alpha` from 0 to any value you'd like above 1. Plot the Train and Test error. Describe what you see based on the `alpha` parameter's stiffness.

```
In [95]: x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=.99)

In [96]: model = Ridge(alpha=.50)
         model.fit(x_train, y_train)
         model.coef_, model.intercept_

Out[96]: (array([0.98308587]), -0.0351552846399561)

In [97]: plt.scatter(x_test,y_test)
         plt.plot(x_test, np.dot(x_test, model.coef_) + model.intercept_)

Out[97]: [<matplotlib.lines.Line2D at 0x2a079406208>]
```




```
In [98]: plt.scatter(x_train,y_train)
         plt.plot(x_train, np.dot(x_train, model.coef_) + model.intercept_)

Out[98]: [<matplotlib.lines.Line2D at 0x2a079471860>]
```




**1.7 Bonus. Either: Generate data with a polynomial shape or use real data that you find on your own. Choose whatever regression model and process you'd like (Ridge, polynomial, etc.) and plot the Train-Test errors vs. any parameter your Model depends on (e.g. `alpha`, `degree`, etc.)**