In [8]:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.datasets import make_regression
```

In [31]:

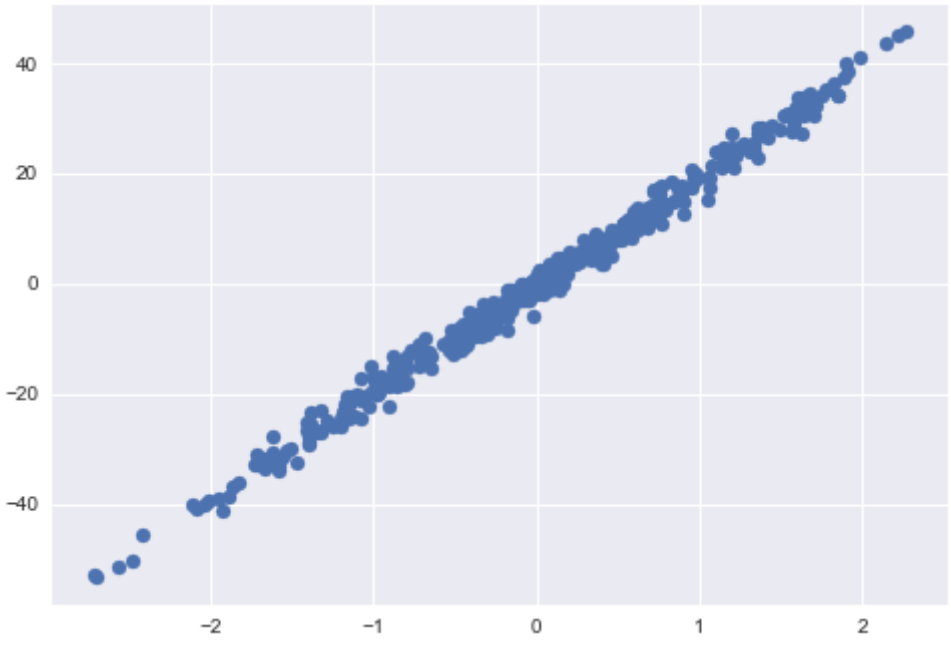
```
# Generate Dataset
X,Y = make_regression(n_samples=400, n_features=1, n_informative=1, noise=1.8, random_s
print(X.shape,Y.shape)
Y = Y.reshape((-1,1))
#print(X,Y)
print(X.shape,Y.shape)
#print(X,Y)
```

```
(400, 1) (400,)
(400, 1) (400, 1)
```

In [32]:

```
# Normalise
X = (X - X.mean())/X.std()

# Visualize
plt.style.use("seaborn")
plt.scatter(X,Y)
plt.show()
```



In [41]:

```
ones = np.ones((X.shape[0],1))
X_ = np.hstack((X, ones))
#print(X_.shape)
#print(X_[:5,])


print(Y.shape)
print(Y[:5,:])
print(type(Y))

Y_ = np.mat(Y)

print(Y.shape)
print(Y_[:5,])
print(type(Y_))
```

```
(400, 1)
[[-14.7699111 ]
 [ 15.19790363]
 [ -4.62367849]
 [ -9.6360043 ]
 [ -1.16210302]]
<class 'numpy.ndarray'>
(400, 1)
[[-14.7699111 ]
 [ 15.19790363]
 [ -4.62367849]
 [ -9.6360043 ]
 [ -1.16210302]]
<class 'numpy.matrix'>
```

In [42]:

```
# Predict
def predict(X, theta):
    return np.dot(X,theta)

def getThetaClosedForm(X,Y):
    Y = np.mat(Y)
    first = np.dot(X.T,X)
    second = np.dot(X.T,Y)

    theta = np.linalg.pinv(first)*second

    return theta
```

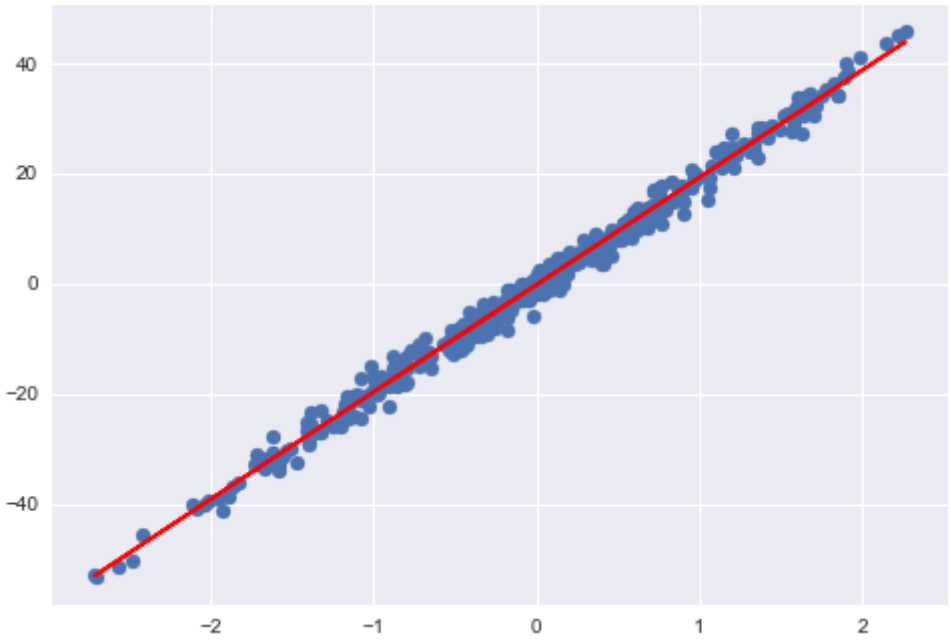
In [43]:

```
theta = getThetaClosedForm(X_,Y)
print(theta)
```

```
 [[19.45877082]
  [-0.29187892]]
```

In [44]:

```
plt.style.use("seaborn")
plt.scatter(X,Y)
plt.plot(X,predict(X_, theta), color='red')
plt.show()
```



In [ ]: