In [26]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.datasets import load_diabetes

data = load_diabetes()
df = pd.DataFrame(data['data'], columns=data['feature_names'])
df['target'] = data['target']

```

In [32]:
```
print(data['DESCR'])
```

```
.. _diabetes_dataset:

Diabetes dataset
----------------

Ten baseline variables, age, sex, body mass index, average blood
pressure, and six blood serum measurements were obtained for each of n =
442 diabetes patients, as well as the response of interest, a
quantitative measure of disease progression one year after baseline.

**Data Set Characteristics:**

  :Number of Instances: 442

  :Number of Attributes: First 10 columns are numeric predictive values

  :Target: Column 11 is a quantitative measure of disease progression one year after baseline

  :Attribute Information:
      - age     age in years
      - sex
      - bmi     body mass index
      - bp      average blood pressure
      - s1      tc, T-Cells (a type of white blood cells)
      - s2      ldl, low-density lipoproteins
      - s3      hdl, high-density lipoproteins
      - s4      tch, thyroid stimulating hormone
      - s5      ltg, lamotrigine
      - s6      glu, blood sugar level

Note: Each of these 10 feature variables have been mean centered and scaled by the standard deviation times `n_sampl
es` (i.e. the sum of squares of each column totals 1).

Source URL:
https://www4.stat.ncsu.edu/~boos/var.select/diabetes.html (https://www4.stat.ncsu.edu/~boos/var.select/diabetes.htm
l)

For more information see:
Bradley Efron, Trevor Hastie, Iain Johnstone and Robert Tibshirani (2004) "Least Angle Regression," Annals of Statis
```

tics (with discussion), 407-499.
(https://web.stanford.edu/~hastie/Papers/LARS/LeastAngle_2002.pdf)

In [27]:

```
df.head()
```

Out[27]:

	age	sex	bmi	bp	s1	s2	s3	s4	s5	s6	target
0	0.038076	0.050680	0.061696	0.021872	-0.044223	-0.034821	-0.043401	-0.002592	0.019908	-0.017646	151.0
1	-0.001882	-0.044642	-0.051474	-0.026328	-0.008449	-0.019163	0.074412	-0.039493	-0.068330	-0.092204	75.0
2	0.085299	0.050680	0.044451	-0.005671	-0.045599	-0.034194	-0.032356	-0.002592	0.002864	-0.025930	141.0
3	-0.089063	-0.044642	-0.011595	-0.036656	0.012191	0.024991	-0.036038	0.034309	0.022692	-0.009362	206.0
4	0.005383	-0.044642	-0.036385	0.021872	0.003935	0.015596	0.008142	-0.002592	-0.031991	-0.046641	135.0

In [28]:

```
## Question Number 1

print(df.describe())
```

```
                age           sex           bmi            bp            s1  \
count  4.420000e+02  4.420000e+02  4.420000e+02  4.420000e+02  4.420000e+02
mean  -3.634285e-16  1.308343e-16 -8.045349e-16  1.281655e-16 -8.835316e-17
std    4.761905e-02  4.761905e-02  4.761905e-02  4.761905e-02  4.761905e-02
min   -1.072256e-01 -4.464164e-02 -9.027530e-02 -1.123996e-01 -1.267807e-01
25%   -3.729927e-02 -4.464164e-02 -3.422907e-02 -3.665645e-02 -3.424784e-02
50%    5.383060e-03 -4.464164e-02 -7.283766e-03 -5.670611e-03 -4.320866e-03
75%    3.807591e-02  5.068012e-02  3.124802e-02  3.564384e-02  2.835801e-02
max    1.107267e-01  5.068012e-02  1.705552e-01  1.320442e-01  1.539137e-01

                 s2            s3            s4            s5            s6  \
count  4.420000e+02  4.420000e+02  4.420000e+02  4.420000e+02  4.420000e+02
mean   1.327024e-16 -4.574646e-16  3.777301e-16 -3.830854e-16 -3.412882e-16
std    4.761905e-02  4.761905e-02  4.761905e-02  4.761905e-02  4.761905e-02
min   -1.156131e-01 -1.023071e-01 -7.639450e-02 -1.260974e-01 -1.377672e-01
25%   -3.035840e-02 -3.511716e-02 -3.949338e-02 -3.324879e-02 -3.317903e-02
50%   -3.819065e-03 -6.584468e-03 -2.592262e-03 -1.947634e-03 -1.077698e-03
75%    2.984439e-02  2.931150e-02  3.430886e-02  3.243323e-02  2.791705e-02
max    1.987880e-01  1.811791e-01  1.852344e-01  1.335990e-01  1.356118e-01

           target
count  442.000000
mean   152.133484
std     77.093005
min     25.000000
25%     87.000000
50%    140.500000
75%    211.500000
max    346.000000
```

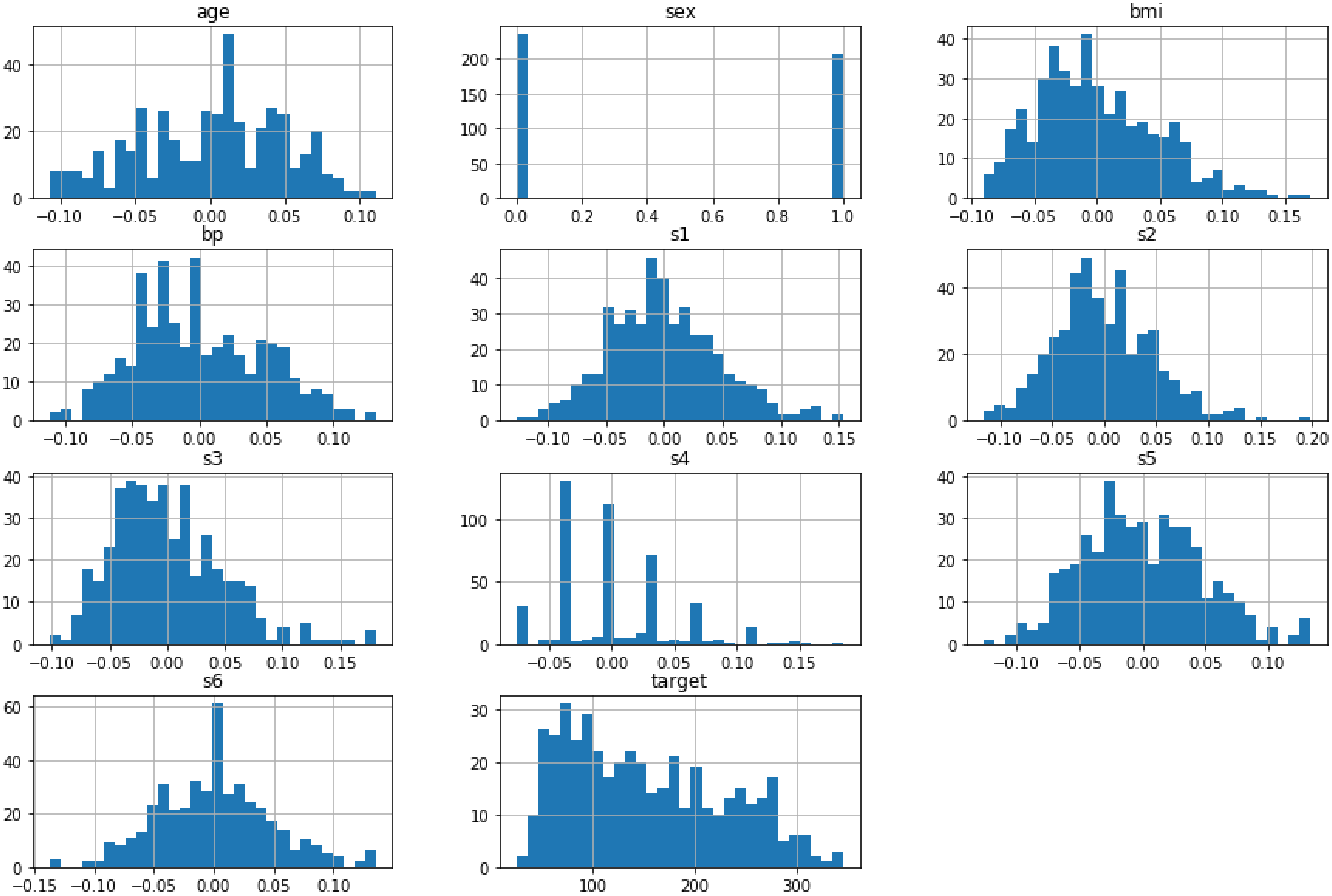
In [24]:
```
## Question 2

100*df.isna().sum()/df.shape[0]
```

Out[24]:
```
age       0.0
sex       0.0
bmi       0.0
bp        0.0
s1        0.0
s2        0.0
s3        0.0
s4        0.0
s5        0.0
s6        0.0
target    0.0
dtype: float64
```

In [6]:
```
#Question 3 without using the replace function

df['sex'] = (df['sex']-df['sex'].min())/(df['sex'].max()-df['sex'].min())
```

In [7]:
```
#Question 4
df.hist(bins=30, figsize=(15, 10))

```

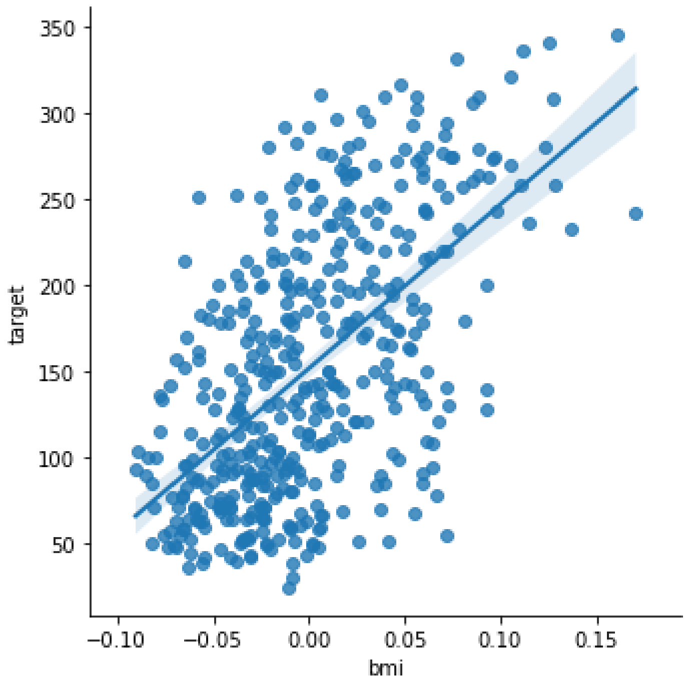
Out[7]:
```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x000001B2FF704C08>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000001B298ED07C8>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000001B2FF786688>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x000001B2FF7B3BC8>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000001B2FF7E2908>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000001B2FF7E29C8>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x000001B2FF811688>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000001B2FF8701C8>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000001B2FF899EC8>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x000001B2FF8C9C08>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000001B2FF8F9848>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000001B2FF92A588>]],
      dtype=object)
```


age
sex
bmi
bp
s1
s2
s3
s4
s5
s6
target

In [17]:
```
#Question 5
##vif code given which creates a dataframe of vif values for each variable
##algo: in a loop remove the variable with the maximum vif in each iteration till we get a dataframe where all vif v
## are uder the threshold




```

In [18]:
```
#Question 6
import seaborn as sns
sns.lmplot(x = 'bmi',y= 'target',data = df)

## create an lm fit and plot y = mx + c
```

Out[18]: <seaborn.axisgrid.FacetGrid at 0x1b2836953c8>



In [20]:
```
#Question 7

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression

X = data.drop('target',1)
y = data['target']
X_train,X_test,y_train,y_test = train_test_split(X,y,random_state = 0)
lm = LinearRegression()
lm.fit(X_train,y_train)
```

Out[20]:
```
LinearRegression()
```

In [33]:
```
lm.coef_
```

Out[33]:
```
array([ -43.26774487,  -19.89084209,  593.39797213,  302.89814903,
       -560.27689824,  261.47657106,   -8.83343952,  135.93715156,
        703.22658427,   28.34844354])
```

In [53]:
```
lm.intercept_
```

Out[53]:
```
162.38337655718004
```

In [47]:
```
df.corr().iloc[-1,:][:-1]
```

Out[47]:
```
age    0.187889
sex    0.043062
bmi    0.586450
bp     0.441484
s1     0.212022
s2     0.174054
s3    -0.394789
s4     0.430453
s5     0.565883
s6     0.382483
Name: target, dtype: float64
```

In [50]:
```
coeff_data = pd.DataFrame({'Coeffs':lm.coef_,'Corr':df.corr().iloc[-1,:][:-1]})
```

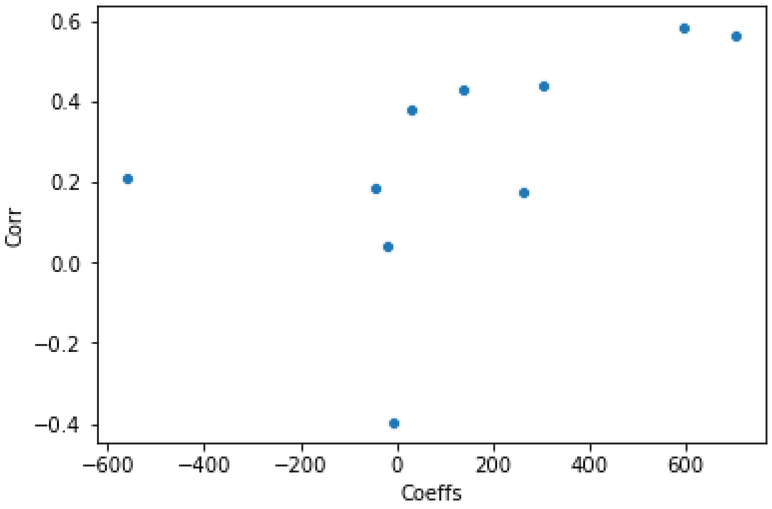
In [54]:
```
coeff_data
```

Out[54]:

	Coeffs	Corr
age	-43.267745	0.187889
sex	-19.890842	0.043062
bmi	593.397972	0.586450
bp	302.898149	0.441484
s1	-560.276898	0.212022
s2	261.476571	0.174054
s3	-8.833440	-0.394789
s4	135.937152	0.430453
s5	703.226584	0.565883
s6	28.348444	0.382483

In [52]:
```
sns.scatterplot(x = 'Coeffs',y = 'Corr',data = coeff_data)
```

Out[52]: <matplotlib.axes._subplots.AxesSubplot at 0x1b283cf2cc8>



In [ ]:
```
#Question 9
```

In [55]:
```
# Question 10
## Find out how to calc rmse
from sklearn.metrics import mean_absolute_error,mean_squared_error,r2_score
y_pred = lm.predict(X_test)
print(f'Mean Absolute Error : {mean_absolute_error(y_pred,y_test)}')
print(f'Mean Squared Error : {mean_squared_error(y_pred,y_test)}')
print(f'R2 Score : {r2_score(y_pred,y_test)}')


```

```
Mean Absolute Error : 45.12098768325099
Mean Squared Error : 3180.1988368427274
R2 Score : -0.20962680915904186
```

In [ ]:
```

```