Aim : Demonstrate how coefficient affected by increasing values of the lamda(alpha).

In [1]:
```
from sklearn.datasets import load_diabetes
```

In [2]:
```
data = load_diabetes()
```

In [7]:
```
data
```

Out[7]:
```
{'data': array([[ 0.03807591,  0.05068012,  0.06169621, ..., -0.00259226,
          0.01990842, -0.01764613],
        [-0.00188202, -0.04464164, -0.05147406, ..., -0.03949338,
         -0.06832974, -0.09220405],
        [ 0.08529891,  0.05068012,  0.04445121, ..., -0.00259226,
          0.00286377, -0.02593034],
        ...,
        [ 0.04170844,  0.05068012, -0.01590626, ..., -0.01107952,
         -0.04687948,  0.01549073],
        [-0.04547248, -0.04464164,  0.03906215, ...,  0.02655962,
          0.04452837, -0.02593034],
        [-0.04547248, -0.04464164, -0.0730303 , ..., -0.03949338,
         -0.00421986,  0.00306441]]),
 'target': array([151.,  75., 141., 206., 135.,  97., 138.,  63., 110., 310.,
101.,
         69., 179., 185., 118., 171., 166., 144.,  97., 168.,  68.,  49.,
         68., 245., 184., 202., 137.,  85., 131., 283., 129.,  59., 341.,
         87.,  65., 102., 265., 276., 252.,  90., 100.,  55.,  61.,  92.,
        259.,  53., 190., 142.,  75., 142., 155., 225.,  59., 104., 182.,
        128.,  52.,  37., 170., 170.,  61., 144.,  52., 128.,  71., 163.,
        150.,  97., 160., 178.,  48., 270., 202., 111.,  85.,  42., 170.,
        200., 252., 113., 143.,  51.,  52., 210.,  65., 141.,  55., 134.,
         42., 111.,  98., 164.,  48.,  96.,  90., 162., 150., 279.,  92.,
         83., 128., 102., 302., 198.,  95.,  53., 134., 144., 232.,  81.,
        104.,  59., 246., 297., 258., 229., 275., 281., 179., 200., 200.,
        173., 180.,  84., 121., 161.,  99., 109., 115., 268., 274., 158.,
        107.,  83., 103., 272.,  85., 280., 336., 281., 118., 317., 235.,
         60., 174., 259., 178., 128.,  96., 126., 288.,  88., 292.,  71.,
        197., 186.,  25.,  84.,  96., 195.,  53., 217., 172., 131., 214.,
         59.,  70., 220., 268., 152.,  47.,  74., 295., 101., 151., 127.,
        237., 225.,  81., 151., 107.,  64., 138., 185., 265., 101., 137.,
        143., 141.,  79., 292., 178.,  91., 116.,  86., 122.,  72., 129.,
        142.,  90., 158.,  39., 196., 222., 277.,  99., 196., 202., 155.,
         77., 191.,  70.,  73.,  49.,  65., 263., 248., 296., 214., 185.,
         78.,  93., 252., 150.,  77., 208.,  77., 108., 160.,  53., 220.,
        154., 259.,  90., 246., 124.,  67.,  72., 257., 262., 275., 177.,
         71.,  47., 187., 125.,  78.,  51., 258., 215., 303., 243.,  91.,
        150., 310., 153., 346.,  63.,  89.,  50.,  39., 103., 308., 116.,
        145.,  74.,  45., 115., 264.,  87., 202., 127., 182., 241.,  66.,
         94., 283.,  64., 102., 200., 265.,  94., 230., 181., 156., 233.,
         60., 219.,  80.,  68., 332., 248.,  84., 200.,  55.,  85.,  89.,
         31., 129.,  83., 275.,  65., 198., 236., 253., 124.,  44., 172.,
        114., 142., 109., 180., 144., 163., 147.,  97., 220., 190., 109.,
        191., 122., 230., 242., 248., 249., 192., 131., 237.,  78., 135.,
        244., 199., 270., 164.,  72.,  96., 306.,  91., 214.,  95., 216.,
        263., 178., 113., 200., 139., 139.,  88., 148.,  88., 243.,  71.,
         77., 109., 272.,  60.,  54., 221.,  90., 311., 281., 182., 321.,
         58., 262., 206., 233., 242., 123., 167.,  63., 197.,  71., 168.,
        140., 217., 121., 235., 245.,  40.,  52., 104., 132.,  88.,  69.,
        219.,  72., 201., 110.,  51., 277.,  63., 118.,  69., 273., 258.,
         43., 198., 242., 232., 175.,  93., 168., 275., 293., 281.,  72.,
        140., 189., 181., 209., 136., 261., 113., 131., 174., 257.,  55.,
         84.,  42., 146., 212., 233.,  91., 111., 152., 120.,  67., 310.,
         94., 183.,  66., 173.,  72.,  49.,  64.,  48., 178., 104., 132.,
        220.,  57.]),
 'frame': None,
 'DESCR': '.. _diabetes_dataset:\n\nDiabetes dataset\n----------------\n\nTen
```

```
baseline variables, age, sex, body mass index, average blood\npressure, and s
ix blood serum measurements were obtained for each of n =\n442 diabetes patie
nts, as well as the response of interest, a\nquantitative measure of disease
progression one year after baseline.\n\n**Data Set Characteristics:**\n\n  :N
umber of Instances: 442\n\n  :Number of Attributes: First 10 columns are nume
ric predictive values\n\n  :Target: Column 11 is a quantitative measure of di
sease progression one year after baseline\n\n  :Attribute Information:\n
- age     age in years\n      - sex\n      - bmi     body mass index\n      -
bp      average blood pressure\n      - s1      tc, total serum cholesterol\n
- s2      ldl, low-density lipoproteins\n      - s3      hdl, high-density li
poproteins\n      - s4      tch, total cholesterol / HDL\n      - s5      lt
g, possibly log of serum triglycerides level\n      - s6      glu, blood suga
r level\n\nNote: Each of these 10 feature variables have been mean centered a
nd scaled by the standard deviation times `n_samples` (i.e. the sum of square
s of each column totals 1).\n\nSource URL:\nhttps://www4.stat.ncsu.edu/~boos/
var.select/diabetes.html\n\nFor more information see:\nBradley Efron, Trevor
Hastie, Iain Johnstone and Robert Tibshirani (2004) "Least Angle Regression,"
Annals of Statistics (with discussion), 407-499.\n(https://web.stanford.edu/~
hastie/Papers/LARS/LeastAngle_2002.pdf)',
 'feature_names': ['age',
  'sex',
  'bmi',
  'bp',
  's1',
  's2',
  's3',
  's4',
  's5',
  's6'],
 'data_filename': 'diabetes_data.csv.gz',
 'target_filename': 'diabetes_target.csv.gz',
 'data_module': 'sklearn.datasets.data'}
```

In [4]:

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

In [5]:

```
df = pd.DataFrame(data.data,columns=data.feature_names)
```

In [6]:
```
df
```

Out[6]:

	age	sex	bmi	bp	s1	s2	s3	s4	
0	0.038076	0.050680	0.061696	0.021872	-0.044223	-0.034821	-0.043401	-0.002592	0.019
1	-0.001882	-0.044642	-0.051474	-0.026328	-0.008449	-0.019163	0.074412	-0.039493	-0.068
2	0.085299	0.050680	0.044451	-0.005671	-0.045599	-0.034194	-0.032356	-0.002592	0.002
3	-0.089063	-0.044642	-0.011595	-0.036656	0.012191	0.024991	-0.036038	0.034309	0.022
4	0.005383	-0.044642	-0.036385	0.021872	0.003935	0.015596	0.008142	-0.002592	-0.031
...	...	...	...	...	...	...	...	...	
437	0.041708	0.050680	0.019662	0.059744	-0.005697	-0.002566	-0.028674	-0.002592	0.031
438	-0.005515	0.050680	-0.015906	-0.067642	0.049341	0.079165	-0.028674	0.034309	-0.018
439	0.041708	0.050680	-0.015906	0.017282	-0.037344	-0.013840	-0.024993	-0.011080	-0.046
440	-0.045472	-0.044642	0.039062	0.001215	0.016318	0.015283	-0.028674	0.026560	0.044
441	-0.045472	-0.044642	-0.073030	-0.081414	0.083740	0.027809	0.173816	-0.039493	-0.004

442 rows × 10 columns

In [8]:
```
df.shape
```

Out[8]: (442, 10)

In [9]:
```
from sklearn.model_selection import train_test_split
```

In [11]:
```
x_train,x_test,y_train,y_test = train_test_split(data.data,data.target,tes
```

In [12]:
```
x_train.shape
```

Out[12]: (353, 10)

In [13]:
```
x_test.shape
```

Out[13]: (89, 10)

In [14]:
```
y_test.shape
```

Out[14]: (89,)

In [16]:
```
from sklearn.linear_model import Ridge
from sklearn.metrics import r2_score
```

In [17]:
```
coef = []
r2_scores = []

for i in [0,10,100,1000]:
    reg = Ridge(alpha = i)
    reg.fit(x_train,y_train)

    coef.append(reg.coef_.tolist())
    y_pred = reg.predict(x_test)
    r2_scores.append(r2_score(y_test,y_pred))
```

In [18]:
```
coef
```

Out[18]:
```
[[-9.16088483246257,
  -205.46225987708993,
  516.6846238313885,
  340.6273410788917,
  -895.5436086743589,
  561.2145330558977,
  153.88478595250436,
  126.73431596154738,
  861.1213995461836,
  52.41982835857518],
 [21.17400371774998,
  1.6597961347385322,
  63.6597719017997,
  48.493240031697546,
  18.421491990472816,
  12.875448426495627,
  -38.91543505723751,
  38.84246372206304,
  61.61240510619145,
  35.505355265613154],
 [2.8589794382553495,
  0.6294520371235343,
  7.5406044960945104,
  5.849996643873588,
  2.710878515266963,
  2.142134389296116,
  -4.834046968577792,
  5.108223239548697,
  7.4484662433551705,
  4.576128672131118],
 [0.29572556030095387,
  0.06929028636932691,
  0.769003806199464,
  0.5978292887031436,
  0.28289951335334357,
  0.22593550596063294,
  -0.4956069088303587,
  0.5270313419211984,
  0.7614974792951518,
  0.4710290658232608]]
```

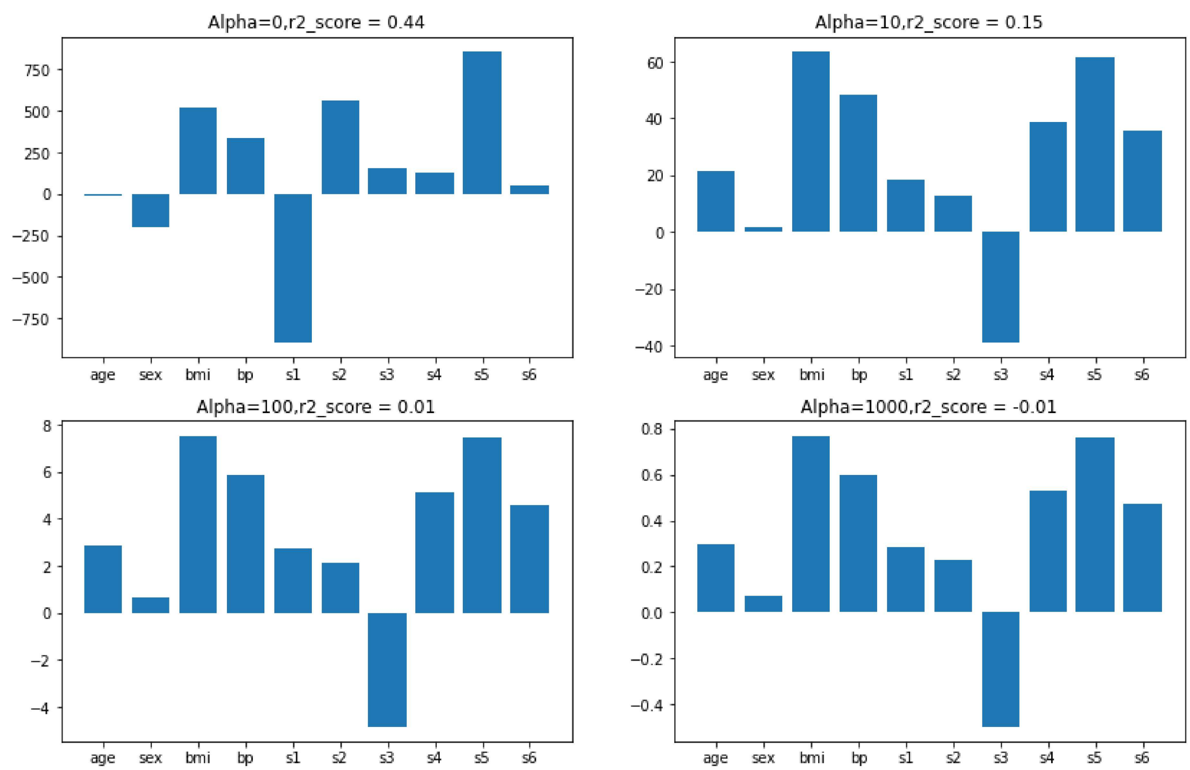
In [24]:
```
plt.figure(figsize=(14,9))
plt.subplot(221)
plt.bar(data.feature_names,coef[0])
plt.title('Alpha=0,r2_score = {}'.format(round(r2_scores[0],2)))

plt.subplot(222)
plt.bar(data.feature_names,coef[1])
plt.title('Alpha=10,r2_score = {}'.format(round(r2_scores[1],2)))

plt.subplot(223)
plt.bar(data.feature_names,coef[2])
plt.title('Alpha=100,r2_score = {}'.format(round(r2_scores[2],2)))

plt.subplot(224)
plt.bar(data.feature_names,coef[3])
plt.title('Alpha=1000,r2_score = {}'.format(round(r2_scores[3],2)))
```

Out[24]:
```
Text(0.5, 1.0, 'Alpha=1000,r2_score = -0.01')
```



Aim : In Ridge regression prove that "The more higher coefficient are affected more".

In [25]:

```python
coef = []


alphas = [0,0.0001,0.001,0.01,0.1,1,10,100,1000,10000]

for i in alphas:

    reg = Ridge(alpha = i)
    reg.fit(x_train,y_train)
    coef.append(reg.coef_.tolist())
```

In [27]:

```python
import numpy as np
np_arr = np.array(coef)
```

In [28]:

```python
coef_df = pd.DataFrame(np_arr,columns = data.feature_names)
coef_df
```

Out[28]:

	age	sex	bmi	bp	s1	s2	s3	
0	-9.160885	-205.462260	516.684624	340.627341	-895.543609	561.214533	153.884786	126.734
1	-9.118336	-205.337133	516.880570	340.556792	-883.415291	551.553259	148.578680	125.355
2	-8.763583	-204.321125	518.371729	339.975385	-787.690766	475.274718	106.786540	114.632
3	-6.401088	-198.669767	522.048548	336.348363	-383.709187	152.663678	-66.060583	75.611
4	6.642753	-172.242166	485.523872	314.682122	-72.939323	-80.590053	-174.466515	83.616
5	42.242217	-57.305508	282.170831	198.061386	14.363544	-22.551274	-136.930053	102.023
6	21.174004	1.659796	63.659772	48.493240	18.421492	12.875448	-38.915435	38.842
7	2.858979	0.629452	7.540604	5.849997	2.710879	2.142134	-4.834047	5.108
8	0.295726	0.069290	0.769004	0.597829	0.282900	0.225936	-0.495607	0.527
9	0.029674	0.006995	0.077054	0.059915	0.028412	0.022715	-0.049686	0.052

In [30]:
```
coef_df['alpha'] = alphas
coef_df.set_index('alpha')
```

Out[30]:

	age	sex	bmi	bp	s1	s2	s3
alpha							
0.0000	-9.160885	-205.462260	516.684624	340.627341	-895.543609	561.214533	153.884786
0.0001	-9.118336	-205.337133	516.880570	340.556792	-883.415291	551.553259	148.578680
0.0010	-8.763583	-204.321125	518.371729	339.975385	-787.690766	475.274718	106.786540
0.0100	-6.401088	-198.669767	522.048548	336.348363	-383.709187	152.663678	-66.060583
0.1000	6.642753	-172.242166	485.523872	314.682122	-72.939323	-80.590053	-174.466515
1.0000	42.242217	-57.305508	282.170831	198.061386	14.363544	-22.551274	-136.930053
10.0000	21.174004	1.659796	63.659772	48.493240	18.421492	12.875448	-38.915435
100.0000	2.858979	0.629452	7.540604	5.849997	2.710879	2.142134	-4.834047
1000.0000	0.295726	0.069290	0.769004	0.597829	0.282900	0.225936	-0.495607
10000.0000	0.029674	0.006995	0.077054	0.059915	0.028412	0.022715	-0.049686

In [ ]:
```

```