In [1]:
```
import numpy as np
import pandas as pd
```

In [2]:
```
dataset=pd.read_csv(r"D:\ML_Course\Works_on_python\Multi_Linear_Regression\50_Startups.csv")
```

In [3]:
```
dataset.head()
```

Out[3]:

	R&D Spend	Administration	Marketing Spend	State	Profit
0	165349.20	136897.80	471784.10	New York	192261.83
1	162597.70	151377.59	443898.53	California	191792.06
2	153441.51	101145.55	407934.54	Florida	191050.39
3	144372.41	118671.85	383199.62	New York	182901.99
4	142107.34	91391.77	366168.42	Florida	166187.94

In [4]:
```
dataset["State"].unique()
```

Out[4]:
```
array(['New York', 'California', 'Florida'], dtype=object)
```

In [5]:
```
dataset.isnull().any()
```

Out[5]:
```
R&D Spend          False
Administration     False
Marketing Spend    False
State              False
Profit             False
dtype: bool
```

In [6]:
```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
dataset["State"]=le.fit_transform(dataset["State"])
```

In [7]:
```
dataset.head()
```

Out[7]:

	R&D Spend	Administration	Marketing Spend	State	Profit
0	165349.20	136897.80	471784.10	2	192261.83
1	162597.70	151377.59	443898.53	0	191792.06
2	153441.51	101145.55	407934.54	1	191050.39
3	144372.41	118671.85	383199.62	2	182901.99
4	142107.34	91391.77	366168.42	1	166187.94

In [8]:
```
x=dataset.iloc[:,0:4].values
y=dataset.iloc[:,4:5].values
```

In [9]:
```
x.shape
```

Out[9]:
```
(50, 4)
```

In [10]:
```
from sklearn.preprocessing import OneHotEncoder
one=OneHotEncoder()
z=one.fit_transform(x[:,3:4]).toarray()
x=np.delete(x,3,axis=1)
x=np.concatenate((z,x),axis=1)
```

In [10]:
```
x.shape
```

Out[10]:
```
(50, 4)
```

In [11]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)
```

In [12]:
```
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
x_train=sc.fit_transform(x_train)
x_test=sc.fit_transform(x_test)
```

In [13]:
```
import keras
from keras.layers import Dense
```

```
Using TensorFlow backend.
C:\Users\anikp\Anaconda3\lib\site-packages\tensorflow\python\framework\dtypes.py:516: FutureWarning: Passing (type, 1) or '1ty
pe' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint8 = np.dtype([("qint8", np.int8, 1)])
C:\Users\anikp\Anaconda3\lib\site-packages\tensorflow\python\framework\dtypes.py:517: FutureWarning: Passing (type, 1) or '1ty
pe' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_quint8 = np.dtype([("quint8", np.uint8, 1)])
C:\Users\anikp\Anaconda3\lib\site-packages\tensorflow\python\framework\dtypes.py:518: FutureWarning: Passing (type, 1) or '1ty
pe' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint16 = np.dtype([("qint16", np.int16, 1)])
C:\Users\anikp\Anaconda3\lib\site-packages\tensorflow\python\framework\dtypes.py:519: FutureWarning: Passing (type, 1) or '1ty
pe' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_quint16 = np.dtype([("quint16", np.uint16, 1)])
C:\Users\anikp\Anaconda3\lib\site-packages\tensorflow\python\framework\dtypes.py:520: FutureWarning: Passing (type, 1) or '1ty
pe' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint32 = np.dtype([("qint32", np.int32, 1)])
C:\Users\anikp\Anaconda3\lib\site-packages\tensorflow\python\framework\dtypes.py:525: FutureWarning: Passing (type, 1) or '1ty
pe' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  np_resource = np.dtype([("resource", np.ubyte, 1)])
C:\Users\anikp\Anaconda3\lib\site-packages\tensorboard\compat\tensorflow_stub\dtypes.py:541: FutureWarning: Passing (type, 1)
or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)typ
e'.
  _np_qint8 = np.dtype([("qint8", np.int8, 1)])
C:\Users\anikp\Anaconda3\lib\site-packages\tensorboard\compat\tensorflow_stub\dtypes.py:542: FutureWarning: Passing (type, 1)
or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)typ
e'.
  _np_quint8 = np.dtype([("quint8", np.uint8, 1)])
C:\Users\anikp\Anaconda3\lib\site-packages\tensorboard\compat\tensorflow_stub\dtypes.py:543: FutureWarning: Passing (type, 1)
or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)typ
e'.
  _np_qint16 = np.dtype([("qint16", np.int16, 1)])
C:\Users\anikp\Anaconda3\lib\site-packages\tensorboard\compat\tensorflow_stub\dtypes.py:544: FutureWarning: Passing (type, 1)
or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)typ
e'.
  _np_quint16 = np.dtype([("quint16", np.uint16, 1)])
C:\Users\anikp\Anaconda3\lib\site-packages\tensorboard\compat\tensorflow_stub\dtypes.py:545: FutureWarning: Passing (type, 1)
or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)typ
e'.
  _np_qint32 = np.dtype([("qint32", np.int32, 1)])
C:\Users\anikp\Anaconda3\lib\site-packages\tensorboard\compat\tensorflow_stub\dtypes.py:550: FutureWarning: Passing (type, 1)
```

```
or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)typ
e'.
  np_resource = np.dtype([("resource", np.ubyte, 1)])
```

In [14]:

```
from keras.models import Sequential
model=Sequential()
```

```
WARNING:tensorflow:From C:\Users\anikp\Anaconda3\lib\site-packages\keras\backend\tensorflow_backend.py:74: The name tf.get_def
ault_graph is deprecated. Please use tf.compat.v1.get_default_graph instead.
```

In [15]:

```
model.add(Dense(units=6,init="random_uniform",activation="relu"))
```

```
C:\Users\anikp\Anaconda3\lib\site-packages\ipykernel_launcher.py:1: UserWarning: Update your `Dense` call to the Keras 2 API:
`Dense(units=6, activation="relu", kernel_initializer="random_uniform")`
  """Entry point for launching an IPython kernel.
```

In [16]:

```
#more hidden layer means more accuracy
model.add(Dense(units=12,init="random_uniform",activation="relu"))#1sr hidden layer
```

```
C:\Users\anikp\Anaconda3\lib\site-packages\ipykernel_launcher.py:1: UserWarning: Update your `Dense` call to the Keras 2 API:
`Dense(units=12, activation="relu", kernel_initializer="random_uniform")`
  """Entry point for launching an IPython kernel.
```

In [18]:

```
model.add(Dense(units=24,init="random_uniform",activation="relu"))#2nd hidden layer
```

```
C:\Users\anikp\Anaconda3\lib\site-packages\ipykernel_launcher.py:1: UserWarning: Update your `Dense` call to the Keras 2 API:
`Dense(units=24, activation="relu", kernel_initializer="random_uniform")`
  """Entry point for launching an IPython kernel.
```

In [19]:

```
model.add(Dense(units=48,init="random_uniform",activation="relu"))#3rd hidden layers
```

```
C:\Users\anikp\Anaconda3\lib\site-packages\ipykernel_launcher.py:1: UserWarning: Update your `Dense` call to the Keras 2 API:
`Dense(units=48, activation="relu", kernel_initializer="random_uniform")`
  """Entry point for launching an IPython kernel.
```

In [20]:

```
model.add(Dense(units=1,init="random_uniform",activation="relu"))#if don't use activation then default is regression
```

```
C:\Users\anikp\Anaconda3\lib\site-packages\ipykernel_launcher.py:1: UserWarning: Update your `Dense` call to the Keras 2 API:
`Dense(units=1, activation="relu", kernel_initializer="random_uniform")`
  """Entry point for launching an IPython kernel.
```

In [21]:

```
model.compile(optimizer="adam",loss="mse",metrics=["mse"])
```

```
WARNING:tensorflow:From C:\Users\anikp\Anaconda3\lib\site-packages\keras\optimizers.py:790: The name tf.train.Optimizer is dep
recated. Please use tf.compat.v1.train.Optimizer instead.
```

In [30]:

```
model.fit(x_train,y_train,batch_size=64,epochs=1000)
```

```
Epoch 1/1000
40/40 [==============================] - 0s 50us/step - loss: 117396112.0000 - mean_squared_error: 117396112.0000
Epoch 2/1000
40/40 [==============================] - 0s 25us/step - loss: 117330408.0000 - mean_squared_error: 117330408.0000
Epoch 3/1000
40/40 [==============================] - 0s 50us/step - loss: 117265024.0000 - mean_squared_error: 117265024.0000
Epoch 4/1000
40/40 [==============================] - 0s 50us/step - loss: 117199896.0000 - mean_squared_error: 117199896.0000
Epoch 5/1000
40/40 [==============================] - 0s 25us/step - loss: 117134928.0000 - mean_squared_error: 117134928.0000
Epoch 6/1000
40/40 [==============================] - 0s 75us/step - loss: 117070128.0000 - mean_squared_error: 117070128.0000
Epoch 7/1000
40/40 [==============================] - 0s 50us/step - loss: 117005392.0000 - mean_squared_error: 117005392.0000
Epoch 8/1000
40/40 [==============================] - 0s 50us/step - loss: 116940672.0000 - mean_squared_error: 116940672.0000
Epoch 9/1000
40/40 [==============================] - 0s 25us/step - loss: 116876016.0000 - mean_squared_error: 116876016.0000
Epoch 10/1000
```

In [31]:

```
y_pred=model.predict(x_test)
```

In [32]:
```
y_pred
```

Out[32]:
```
array([[ 97709.5  ],
       [128517.85 ],
       [122988.92 ],
       [ 50916.523],
       [182800.84 ],
       [102247.03 ],
       [ 56163.426],
       [102314.06 ],
       [105245.3  ],
       [168785.5  ]], dtype=float32)
```

In [33]:
```
y_test
```

Out[33]:
```
array([[103282.38],
       [144259.4 ],
       [146121.95],
       [ 77798.83],
       [191050.39],
       [105008.31],
       [ 81229.06],
       [ 97483.56],
       [110352.25],
       [166187.94]])
```

In [34]:
```
from sklearn.metrics import r2_score
r2_score(y_test,y_pred)
```

Out[34]:
```
0.82040882255744
```

In [ ]: