In [1]:
```
import pandas as pd
import numpy as np
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
```

In [2]:
```
insurance_data = pd.read_csv("insurance.csv")
insurance_data
```

Out[2]:

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	yes	southwest	16884.92400
1	18	male	33.770	1	no	southeast	1725.55230
2	28	male	33.000	3	no	southeast	4449.46200
3	33	male	22.705	0	no	northwest	21984.47061
4	32	male	28.880	0	no	northwest	3866.85520
...	...	...	...	...	...	...	...
1333	50	male	30.970	3	no	northwest	10600.54830
1334	18	female	31.920	0	no	northeast	2205.98080
1335	18	female	36.850	0	no	southeast	1629.83350
1336	21	female	25.800	0	no	southwest	2007.94500
1337	61	female	29.070	0	yes	northwest	29141.36030

1338 rows × 7 columns

In [3]:
```
insurance_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1338 entries, 0 to 1337
Data columns (total 7 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   age       1338 non-null   int64  
 1   sex       1338 non-null   object 
 2   bmi       1338 non-null   float64
 3   children  1338 non-null   int64  
 4   smoker    1338 non-null   object 
 5   region    1338 non-null   object 
 6   charges   1338 non-null   float64
dtypes: float64(2), int64(2), object(3)
memory usage: 73.3+ KB
```

In [4]:
```
np.unique(insurance_data["sex"])
```

Out[4]:
```
array(['female', 'male'], dtype=object)
```

In [5]:
```
np.unique(insurance_data["smoker"])
```

Out[5]:
```
array(['no', 'yes'], dtype=object)
```

In [6]:
```
np.unique(insurance_data["region"])
```

Out[6]:
```
array(['northeast', 'northwest', 'southeast', 'southwest'], dtype=object)
```

In [7]:
```
category_features = ["sex" , "smoker" , "region"]
```

In [8]:
```
one_hot = OneHotEncoder()
```

In [9]:
```
transformer = ColumnTransformer([("one hot" , one_hot , category_features )] ,
```

In [10]:
```
transformer
```

Out[10]:
```
ColumnTransformer(remainder='passthrough',
                  transformers=[('one hot', OneHotEncoder(),
                                 ['sex', 'smoker', 'region'])])
```

In [11]:
```
transformer_data = transformer.fit_transform(insurance_data)
```

In [12]:
```
transformer_data
```

Out[12]:
```
array([[1.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
        2.79000000e+01, 0.00000000e+00, 1.68849240e+04],
       [0.00000000e+00, 1.00000000e+00, 1.00000000e+00, ...,
        3.37700000e+01, 1.00000000e+00, 1.72555230e+03],
       [0.00000000e+00, 1.00000000e+00, 1.00000000e+00, ...,
        3.30000000e+01, 3.00000000e+00, 4.44946200e+03],
       ...,
       [1.00000000e+00, 0.00000000e+00, 1.00000000e+00, ...,
        3.68500000e+01, 0.00000000e+00, 1.62983350e+03],
       [1.00000000e+00, 0.00000000e+00, 1.00000000e+00, ...,
        2.58000000e+01, 0.00000000e+00, 2.00794500e+03],
       [1.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
        2.90700000e+01, 0.00000000e+00, 2.91413603e+04]])
```

In [13]:
```
pd.DataFrame(transformer_data)
```

Out[13]:

	0	1	2	3	4	5	6	7	8	9	10	11
0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	19.0	27.900	0.0	16884.92400
1	0.0	1.0	1.0	0.0	0.0	0.0	1.0	0.0	18.0	33.770	1.0	1725.55230
2	0.0	1.0	1.0	0.0	0.0	0.0	1.0	0.0	28.0	33.000	3.0	4449.46200
3	0.0	1.0	1.0	0.0	0.0	1.0	0.0	0.0	33.0	22.705	0.0	21984.47061
4	0.0	1.0	1.0	0.0	0.0	1.0	0.0	0.0	32.0	28.880	0.0	3866.85520
...	...	...	...	...	...	...	...	...	...	...	...	...
1333	0.0	1.0	1.0	0.0	0.0	1.0	0.0	0.0	50.0	30.970	3.0	10600.54830
1334	1.0	0.0	1.0	0.0	1.0	0.0	0.0	0.0	18.0	31.920	0.0	2205.98080
1335	1.0	0.0	1.0	0.0	0.0	0.0	1.0	0.0	18.0	36.850	0.0	1629.83350
1336	1.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	21.0	25.800	0.0	2007.94500
1337	1.0	0.0	0.0	1.0	0.0	1.0	0.0	0.0	61.0	29.070	0.0	29141.36030

1338 rows × 12 columns

In [14]:
```
dummies = pd.get_dummies(insurance_data , columns=["sex" , "smoker" , "region"
```

In [15]:
```
dummies
```

Out[15]:

	age	bmi	children	charges	sex_female	sex_male	smoker_no	smoker_yes	region
0	19	27.900	0	16884.92400	1	0	0	1	
1	18	33.770	1	1725.55230	0	1	1	0	
2	28	33.000	3	4449.46200	0	1	1	0	
3	33	22.705	0	21984.47061	0	1	1	0	
4	32	28.880	0	3866.85520	0	1	1	0	
...	...	...	...	...	...	...	...	...	
1333	50	30.970	3	10600.54830	0	1	1	0	
1334	18	31.920	0	2205.98080	1	0	1	0	
1335	18	36.850	0	1629.83350	1	0	1	0	
1336	21	25.800	0	2007.94500	1	0	1	0	
1337	61	29.070	0	29141.36030	1	0	0	1	

1338 rows × 12 columns

In [16]:
```
x = dummies.drop("charges" , axis=1)
y = dummies["charges"]
```

In [17]:
```
x_train , x_test , y_train , y_test = train_test_split( x , y , test_size=0.2)
```

In [18]:
```
model = RandomForestRegressor()
```

In [19]:
```
model.get_params()
```

Out[19]:
```
{'bootstrap': True,
 'ccp_alpha': 0.0,
 'criterion': 'squared_error',
 'max_depth': None,
 'max_features': 'auto',
 'max_leaf_nodes': None,
 'max_samples': None,
 'min_impurity_decrease': 0.0,
 'min_samples_leaf': 1,
 'min_samples_split': 2,
 'min_weight_fraction_leaf': 0.0,
 'n_estimators': 100,
 'n_jobs': None,
 'oob_score': False,
 'random_state': None,
 'verbose': 0,
 'warm_start': False}
```

In [20]:
```
model.fit( x_train , y_train )
```

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

In [21]:
```
y_predict = model.predict(x_test)
```

In [22]:
```
model.score(x_test,y_test)
```

Out[22]:
```
0.8284273630523311
```

In [23]:
```
np.random.seed()
```

In [24]:
```
for i in range(10,151,10):
    print(f'trying model with {i} estimators')
    model = RandomForestRegressor(n_estimators = i).fit( x_train , y_train )
    print(f'model accuracy on test :{model.score( x_test , y_test )*100}')
    print(" ")
```

```
trying model with 10 estimators
model accuracy on test :83.03915223968296

trying model with 20 estimators
model accuracy on test :83.18210260248703

trying model with 30 estimators
model accuracy on test :82.33733498600799

trying model with 40 estimators
model accuracy on test :82.78506705167929

trying model with 50 estimators
model accuracy on test :82.7460009712462

trying model with 60 estimators
model accuracy on test :82.81550312940821

trying model with 70 estimators
model accuracy on test :83.71793775154333

trying model with 80 estimators
model accuracy on test :83.46475705687597

trying model with 90 estimators
model accuracy on test :83.00614496743758

trying model with 100 estimators
model accuracy on test :83.69519360534935

trying model with 110 estimators
model accuracy on test :82.77080644590893

trying model with 120 estimators
model accuracy on test :83.56409817120209

trying model with 130 estimators
model accuracy on test :83.10321197098625

trying model with 140 estimators
model accuracy on test :83.3593011119682

trying model with 150 estimators
model accuracy on test :83.34956492225562
```

In [25]:
```
from sklearn import preprocessing
```

In [26]:
```
label_encoder = preprocessing.LabelEncoder()
```

In [27]:
```
insurance_data
```

Out[27]:

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	yes	southwest	16884.92400
1	18	male	33.770	1	no	southeast	1725.55230
2	28	male	33.000	3	no	southeast	4449.46200
3	33	male	22.705	0	no	northwest	21984.47061
4	32	male	28.880	0	no	northwest	3866.85520
...	...	...	...	...	...	...	...
1333	50	male	30.970	3	no	northwest	10600.54830
1334	18	female	31.920	0	no	northeast	2205.98080
1335	18	female	36.850	0	no	southeast	1629.83350
1336	21	female	25.800	0	no	southwest	2007.94500
1337	61	female	29.070	0	yes	northwest	29141.36030

1338 rows × 7 columns

In [28]:
```
insurance_data["sex"] = label_encoder.fit_transform(insurance_data["sex"])
```

In [29]:
```
insurance_data["smoker"] = label_encoder.fit_transform(insurance_data["smoker"
```

In [30]:
```
insurance_data["region"] = label_encoder.fit_transform(insurance_data["region"
```

In [31]:
```
insurance_data
```

Out[31]:

	age	sex	bmi	children	smoker	region	charges
0	19	0	27.900	0	1	3	16884.92400
1	18	1	33.770	1	0	2	1725.55230
2	28	1	33.000	3	0	2	4449.46200
3	33	1	22.705	0	0	1	21984.47061
4	32	1	28.880	0	0	1	3866.85520
...	...	...	...	...	...	...	...
1333	50	1	30.970	3	0	1	10600.54830
1334	18	0	31.920	0	0	0	2205.98080
1335	18	0	36.850	0	0	2	1629.83350
1336	21	0	25.800	0	0	3	2007.94500
1337	61	0	29.070	0	1	1	29141.36030

1338 rows × 7 columns

In [32]:
```
x2 = insurance_data.drop("charges" , axis=1)
y2 = insurance_data["charges"]
```

In [33]:
```
x_train2 , x_test2 , y_train2 , y_test2 = train_test_split( x2 , y2 , test_siz
```

In [34]:
```
model2 = RandomForestRegressor()
```

In [35]:
```
model2.fit( x_train2 , y_train2)
```

Out[35]:
```
RandomForestRegressor()
```

In [36]:
```
y2_predict = model2.predict(x_test2)
```

In [37]:
```
model2.score( x_test2 , y_test2 )
```

Out[37]:
```
0.8032168105357446
```

In [38]:
```
for i in range(10,151,10):
    print(f'trying model with {i} estimators')
    model2 = RandomForestRegressor(n_estimators = i).fit( x_train2 , y_train2
    print(f'model accuracy on test :{model2.score( x_test2 , y_test2 )*100}')
    print(" ")
```

```
trying model with 10 estimators
model accuracy on test :79.27819083678061

trying model with 20 estimators
model accuracy on test :79.7674572853045

trying model with 30 estimators
model accuracy on test :79.92144065331627

trying model with 40 estimators
model accuracy on test :79.83439685584864

trying model with 50 estimators
model accuracy on test :80.40981491557271

trying model with 60 estimators
model accuracy on test :80.54695834459228

trying model with 70 estimators
model accuracy on test :80.19962071292746

trying model with 80 estimators
model accuracy on test :80.52366575405279

trying model with 90 estimators
model accuracy on test :80.49737136953622

trying model with 100 estimators
model accuracy on test :80.27235778238767

trying model with 110 estimators
model accuracy on test :80.32817061914734

trying model with 120 estimators
model accuracy on test :80.1901300418803

trying model with 130 estimators
model accuracy on test :80.29436907737824

trying model with 140 estimators
model accuracy on test :80.36106558108176

trying model with 150 estimators
model accuracy on test :80.1232810581321
```

In [ ]: