In [2]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np

#분류 알고리즘
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier

#예측 알고리즘
from sklearn.linear_model import LinearRegression
from sklearn.svm import SVR
from sklearn.ensemble import RandomForestRegressor
from xgboost import XGBRegressor

#평가 (분류)
from sklearn.metrics import accuracy_score,confusion_matrix, classification_report

#평가 (예측)
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score

#데이터 스케일링
from sklearn.preprocessing import StandardScaler,LabelEncoder

#train,test 나누기
from sklearn.model_selection import train_test_split
```

In [3]:
```
#예측할 데이터셋 불러오기
df1 = pd.read_csv("/Users/youngjinseo/Desktop/파이썬/insurance.csv")

#분류할 데이터셋 불러오기
df2 = pd.read_csv("/Users/youngjinseo/Desktop/파이썬/mobile_price_classification.csv")
```

분류

In [25]:
```
#데이터셋 내부 보기
df2.head(8)
```

Out[25]:

	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	n_cores	...	px_height	px_width	ram	sc_h	sc_w	talk_time	three_g	touch_screen	wifi	price_range
0	842	0	2.2	0	1	0	7	0.6	188	2	...	20	756	2549	9	7	19	0	0	1	1
1	1021	1	0.5	1	0	1	53	0.7	136	3	...	905	1988	2631	17	3	7	1	1	0	2
2	563	1	0.5	1	2	1	41	0.9	145	5	...	1263	1716	2603	11	2	9	1	1	0	2
3	615	1	2.5	0	0	0	10	0.8	131	6	...	1216	1786	2769	16	8	11	1	0	0	2
4	1821	1	1.2	0	13	1	44	0.6	141	2	...	1208	1212	1411	8	2	15	1	1	0	1
5	1859	0	0.5	1	3	0	22	0.7	164	1	...	1004	1654	1067	17	1	10	1	0	0	1
6	1821	0	1.7	0	4	1	10	0.8	139	8	...	381	1018	3220	13	8	18	1	0	1	3
7	1954	0	0.5	1	0	0	24	0.8	187	4	...	512	1149	700	16	3	5	1	1	1	0

8 rows × 21 columns

In [26]:
```
#데이터 정보
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2000 entries, 0 to 1999
Data columns (total 21 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   battery_power  2000 non-null   int64
 1   blue           2000 non-null   int64
 2   clock_speed    2000 non-null   float64
 3   dual_sim       2000 non-null   int64
 4   fc             2000 non-null   int64
 5   four_g         2000 non-null   int64
 6   int_memory     2000 non-null   int64
 7   m_dep          2000 non-null   float64
 8   mobile_wt      2000 non-null   int64
 9   n_cores        2000 non-null   int64
 10  pc             2000 non-null   int64
 11  px_height      2000 non-null   int64
 12  px_width       2000 non-null   int64
 13  ram            2000 non-null   int64
 14  sc_h           2000 non-null   int64
 15  sc_w           2000 non-null   int64
 16  talk_time      2000 non-null   int64
 17  three_g        2000 non-null   int64
 18  touch_screen   2000 non-null   int64
 19  wifi           2000 non-null   int64
 20  price_range    2000 non-null   int64
dtypes: float64(2), int64(19)
memory usage: 328.2 KB
```

In [27]:
```
#데이터 결측치 확인
df2.isnull().sum()
```

Out[27]:
```
battery_power    0
blue             0
clock_speed      0
dual_sim         0
fc               0
four_g           0
int_memory       0
m_dep            0
mobile_wt        0
n_cores          0
pc               0
px_height        0
px_width         0
ram              0
sc_h             0
sc_w             0
talk_time        0
three_g          0
touch_screen     0
wifi             0
price_range      0
dtype: int64
```

In [28]:
```
#x, y로 나누기
y = df2['price_range']
x = df2.drop('price_range',axis = 1)
```

In [29]:
```
#train_test_split
xtrain,xtest,ytrain,ytest = train_test_split(x,y, test_size = 0.3, random_state = 42)

print(xtrain.shape)
print(xtest.shape)
print(ytrain.shape)
print(ytest.shape)
```

```
(1400, 20)
(600, 20)
(1400,)
(600,)
```

In [30]:
```
#데이터 스케일링
scaler = StandardScaler()

xtrain = scaler.fit_transform(xtrain)
xtest = scaler.fit_transform(xtest)
```

In [31]:
```
# 모델 튜닝하기

rf = RandomForestClassifier(random_state = 42)
dt = DecisionTreeClassifier(random_state = 42)
lr = LogisticRegression(max_iter = 1000,random_state = 42)
knn = KNeighborsClassifier()
svc = SVC(random_state = 42)
```

In [32]:
```
# 모델 학습 및 평가
models = {'RandomForest':rf, 'DecisionTree': dt, 'LogisticRegression':lr,'KNN':knn, 'SVC':svc}
results = {}

for name, model in models.items():
    model.fit(xtrain,ytrain)
    ypred = model.predict(xtest)
    accuracy = accuracy_score(ytest,ypred)
    results[name]=accuracy

# 결과 출력
print("*모델별 정확도 결과:*")
for model,acc in results.items():
    print(f"{model}:{acc:.2%}")
```

모델별 정확도 결과:
RandomForest:88.83%
DecisionTree:82.17%
LogisticRegression:96.17%
KNN:50.83%
SVC:87.33%

In [33]:
```
#LogisticRegression으로 xtest 예측하기
ypred = lr.predict(xtest)

#정확도
al = accuracy_score(ytest,ypred)
print('Accuracy Score:',al)


#혼동 행렬
print('\nConfusion Matrix:\n',confusion_matrix(ytest,ypred))
```

```
#정밀도(Precision), 재현율(Recall), f1점수 (classification_report)
print('\nClassification REport:\n',classification_report(ytest,ypred))
```

```
Accuracy Score: 0.9616666666666667

Confusion Matrix:
 [[146   5   0   0]
 [  3 143   0   0]
 [  0   8 136   4]
 [  0   0   3 152]]

Classification REport:
               precision    recall  f1-score   support

           0       0.98      0.97      0.97       151
           1       0.92      0.98      0.95       146
           2       0.98      0.92      0.95       148
           3       0.97      0.98      0.98       155

    accuracy                           0.96       600
   macro avg       0.96      0.96      0.96       600
weighted avg       0.96      0.96      0.96       600
```

예측

In [34]:
```
#데이터셋 보기
df1.head(8)
```

Out[34]:

	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
5	31	female	25.740	0	no	southeast	3756.62160
6	46	female	33.440	1	no	southeast	8240.58960
7	37	female	27.740	3	no	northwest	7281.50560

In [35]:
```
#데이터 정보보기
df1.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 [36]:
```
#결측치 확인
df1.isnull().sum()
```

Out[36]:
```
age         0
sex         0
bmi         0
children    0
smoker      0
region      0
charges     0
dtype: int64
```

In [4]:
```
#LabelEncoder 문자데이터 숫자로

le = LabelEncoder()

df1['sex']= le.fit_transform(df1['sex'])
df1['smoker']= le.fit_transform(df1['smoker'])
df1['region']= le.fit_transform(df1['region'])

df1.head(8)
```

Out[4]:

	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
5	31	0	25.740	0	0	2	3756.62160
6	46	0	33.440	1	0	2	8240.58960
7	37	0	27.740	3	0	1	7281.50560

In [5]:
```
#x,y 나누기
x = df1.drop('charges',axis = 1 )
y = df1['charges']
```

In [6]:
```
#train_test_split

xtrain,xtest,ytrain,ytest = train_test_split(x,y,test_size = 0.2, random_state = 42)
```

In [7]:
```
#데이터 스케일링
scaler = StandardScaler()

xtrain = scaler.fit_transform(xtrain)
xtest = scaler.fit_transform(xtest)
```

In [8]:
```
#모델 튜닝하기
lr = LinearRegression()
svr = SVR(C=100, epsilon = 0.1, kernel = 'rbf')
rfr = RandomForestRegressor(n_estimators = 100, random_state = 42)
xgbr = XGBRegressor(n_estimators = 100, learning_rate = 0.1, max_depth = 61)
```

In [11]:
```
#모델 평가 점수보기
lr.fit(xtrain,ytrain)
print('LinearRegression:',lr.score(xtest,ytest))

svr.fit(xtrain,ytrain)
print('SVR:',svr.score(xtest,ytest))

rfr.fit(xtrain,ytrain)
print('RandomForestRegressor:',rfr.score(xtest,ytest))

xgbr.fit(xtrain,ytrain)
print('XGBRegressor:',xgbr.score(xtest,ytest))
```

```
LinearRegression: 0.7833237659369187
SVR: 0.13201493830493937
RandomForestRegressor: 0.8617757870630031
XGBRegressor: 0.8315286678548098
```

In [12]:
```
#RandomForestRegressor으로 예측하기
ypred = rfr.predict(xtest)

#평균 제곱 오차 (MSE),평균 절대 오차(MAE),R2점수 (결정 계수)보기
mse = mean_squared_error(ytest, ypred)
mae = mean_absolute_error(ytest, ypred)
r2 = r2_score(ytest, ypred)

print(f'MSE: {mse:}')
print(f'MAE: {mae:}')
print(f'R² Score: {r2:}')
```

```
MSE: 21459133.719911756
MAE: 2721.2658783747215
R² Score: 0.8617757870630031
```

In [ ]: