```
import numpy as np
import pandas as pd
import seaborn as sb
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.metrics import mean_absolute_percentage_error as mape
from sklearn.linear_model import LinearRegression, Lasso, Ridge
from xgboost import XGBRegressor
from sklearn.ensemble import RandomForestRegressor, AdaBoostRegressor

import warnings
warnings.filterwarnings('ignore')
```

```
from google.colab import files
uploaded = files.upload()
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving Medical Price Dataset.csv to Medical Price Dataset.csv
```

```
df = pd.read_csv('Medical Price Dataset.csv')
df.head()
```

	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

```
df.shape
```

```
(1338, 7)
```

```
df.describe()
```

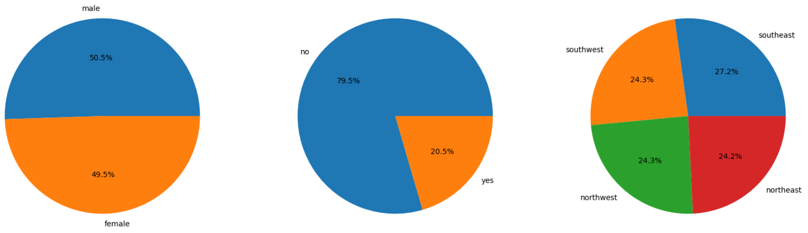
	age	bmi	children	charges
count	1338.000000	1338.000000	1338.000000	1338.000000
mean	39.207025	30.663397	1.094918	13270.422265
std	14.049960	6.098187	1.205493	12110.011237
min	18.000000	15.960000	0.000000	1121.873900
25%	27.000000	26.296250	0.000000	4740.287150
50%	39.000000	30.400000	1.000000	9382.033000
75%	51.000000	34.693750	2.000000	16639.912515
max	64.000000	53.130000	5.000000	63770.428010

```
df.isnull().sum()
```

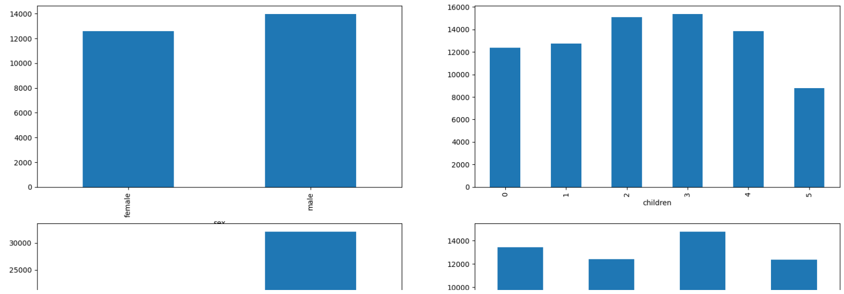
```
age         0
sex         0
bmi         0
children    0
smoker      0
```

```
    region      0
    charges     0
    dtype: int64
```

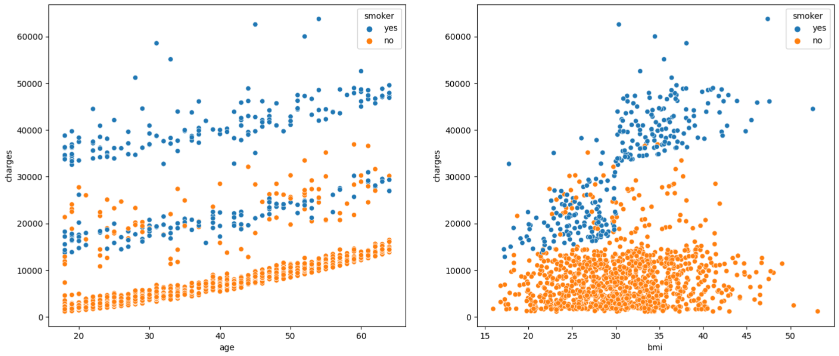
```
features = ['sex', 'smoker', 'region']

plt.subplots(figsize=(20, 10))
for i, col in enumerate(features):
    plt.subplot(1, 3, i + 1)

    x = df[col].value_counts()
    plt.pie(x.values,
            labels=x.index,
            autopct='%1.1f%%')

plt.show()
```



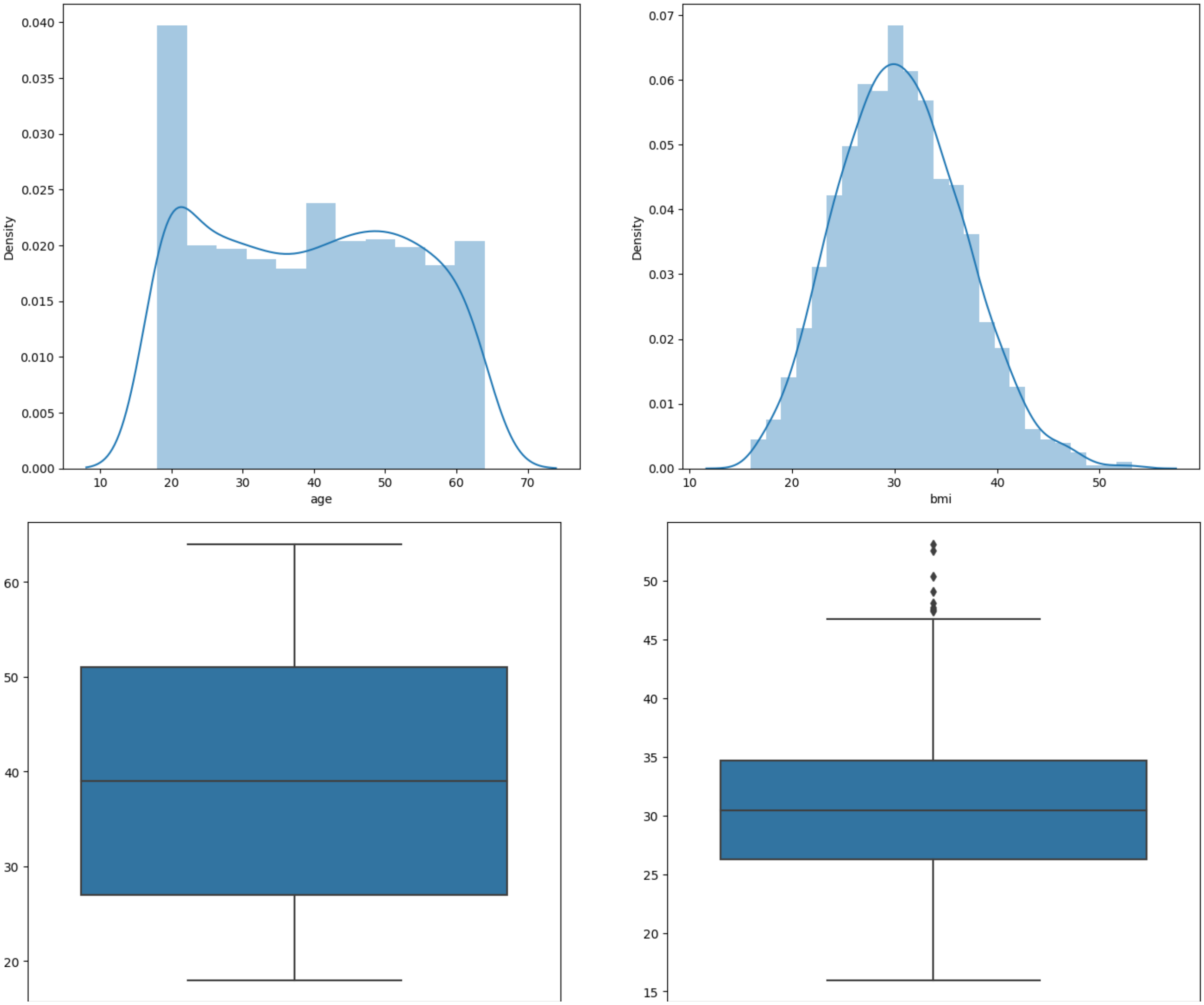
```
features = ['sex', 'children', 'smoker', 'region']

plt.subplots(figsize=(20, 10))
for i, col in enumerate(features):
    plt.subplot(2, 2, i + 1)
    df.groupby(col).mean()['charges'].plot.bar()
plt.show()
```



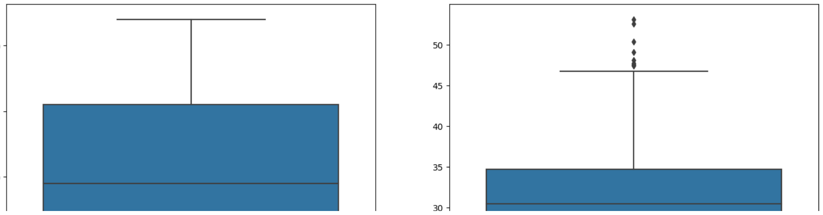
```
features = ['age', 'bmi']

plt.subplots(figsize=(17, 7))
for i, col in enumerate(features):
    plt.subplot(1, 2, i + 1)
    sb.scatterplot(data=df, x=col,
                  y='charges',
                  hue='smoker')
plt.show()
```



```
features = ['age', 'bmi']

plt.subplots(figsize=(17, 7))
for i, col in enumerate(features):
    plt.subplot(1, 2, i + 1)
    sb.distplot(df[col])
plt.show()
features = ['age', 'bmi']

plt.subplots(figsize=(17, 7))
for i, col in enumerate(features):
    plt.subplot(1, 2, i + 1)
    sb.boxplot(df[col])
plt.show()
```



```
features = ['age', 'bmi']

plt.subplots(figsize=(17, 7))
for i, col in enumerate(features):
    plt.subplot(1, 2, i + 1)
    sb.boxplot(df[col])
plt.show()
```



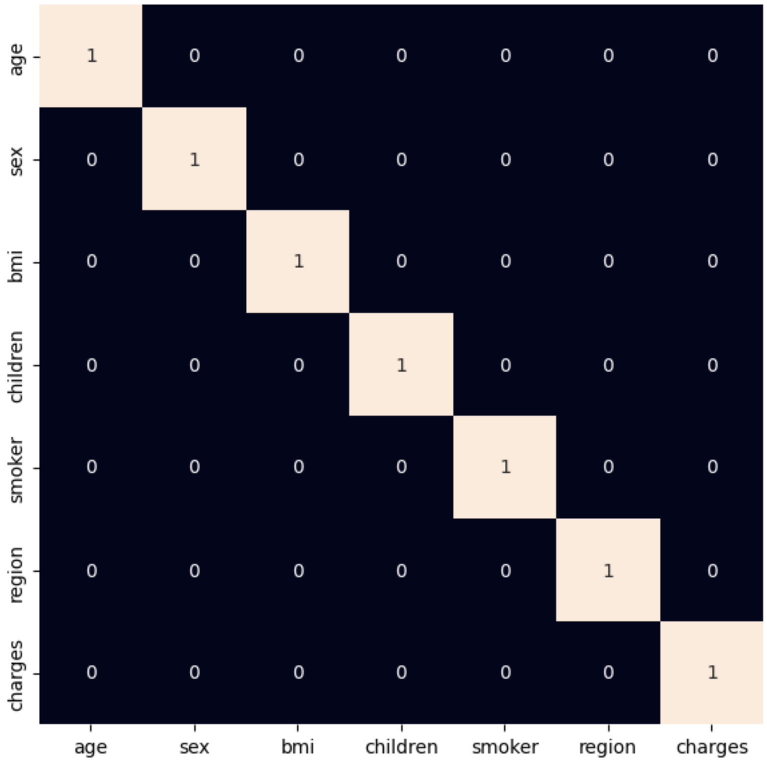
```
df.shape, df[df['bmi']<45].shape
```

```
((1338, 7), (1318, 7))
```

```
df = df[df['bmi']<45]
```

```
for col in df.columns:
    if df[col].dtype == object:
        le = LabelEncoder()
        df[col] = le.fit_transform(df[col])
```

```
plt.figure(figsize=(7, 7))
sb.heatmap(df.corr() > 0.8,
        annot=True,
        cbar=False)
plt.show()
```



```
features = df.drop('charges', axis=1)
target = df['charges']

X_train, X_val,\
Y_train, Y_val = train_test_split(features, target,
                                  test_size=0.2,
                                  random_state=22)
X_train.shape, X_val.shape
```

```
((1054, 6), (264, 6))
```

```
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_val = scaler.transform(X_val)
```

```
models = [LinearRegression(), XGBRegressor(),
        RandomForestRegressor(), AdaBoostRegressor(),
        Lasso(), Ridge()]

for i in range(6):
    models[i].fit(X_train, Y_train)

    print(f'{models[i]} : ')
    pred_train = models[i].predict(X_train)
    print('Training Error : ', mape(Y_train, pred_train))

    pred_val = models[i].predict(X_val)
    print('Validation Error : ', mape(Y_val, pred_val))
    print()
```

```
LinearRegression() :
Training Error :  0.4188805629224119
Validation Error :  0.4504495878121591

XGBRegressor(base_score=None, booster=None, callbacks=None,
             colsample_bylevel=None, colsample_bynode=None,
             colsample_bytree=None, early_stopping_rounds=None,
             enable_categorical=False, eval_metric=None, feature_types=None,
             gamma=None, gpu_id=None, grow_policy=None, importance_type=None,
             interaction_constraints=None, learning_rate=None, max_bin=None,
             max_cat_threshold=None, max_cat_to_onehot=None,
             max_delta_step=None, max_depth=None, max_leaves=None,
             min_child_weight=None, missing=nan, monotone_constraints=None,
             n_estimators=100, n_jobs=None, num_parallel_tree=None,
             predictor=None, random_state=None, ...) :
Training Error :  0.0697883333923925
Validation Error :  0.36004392100129423

RandomForestRegressor() :
Training Error :  0.1162169722343163
Validation Error :  0.25548880359615406

AdaBoostRegressor() :
Training Error :  0.5994569067722977
Validation Error :  0.6203643955021714

Lasso() :
Training Error :  0.418841845707845
Validation Error :  0.45044188913851757

Ridge() :
Training Error :  0.4190871910460788
Validation Error :  0.45082076456283665
```