4uqn1vued

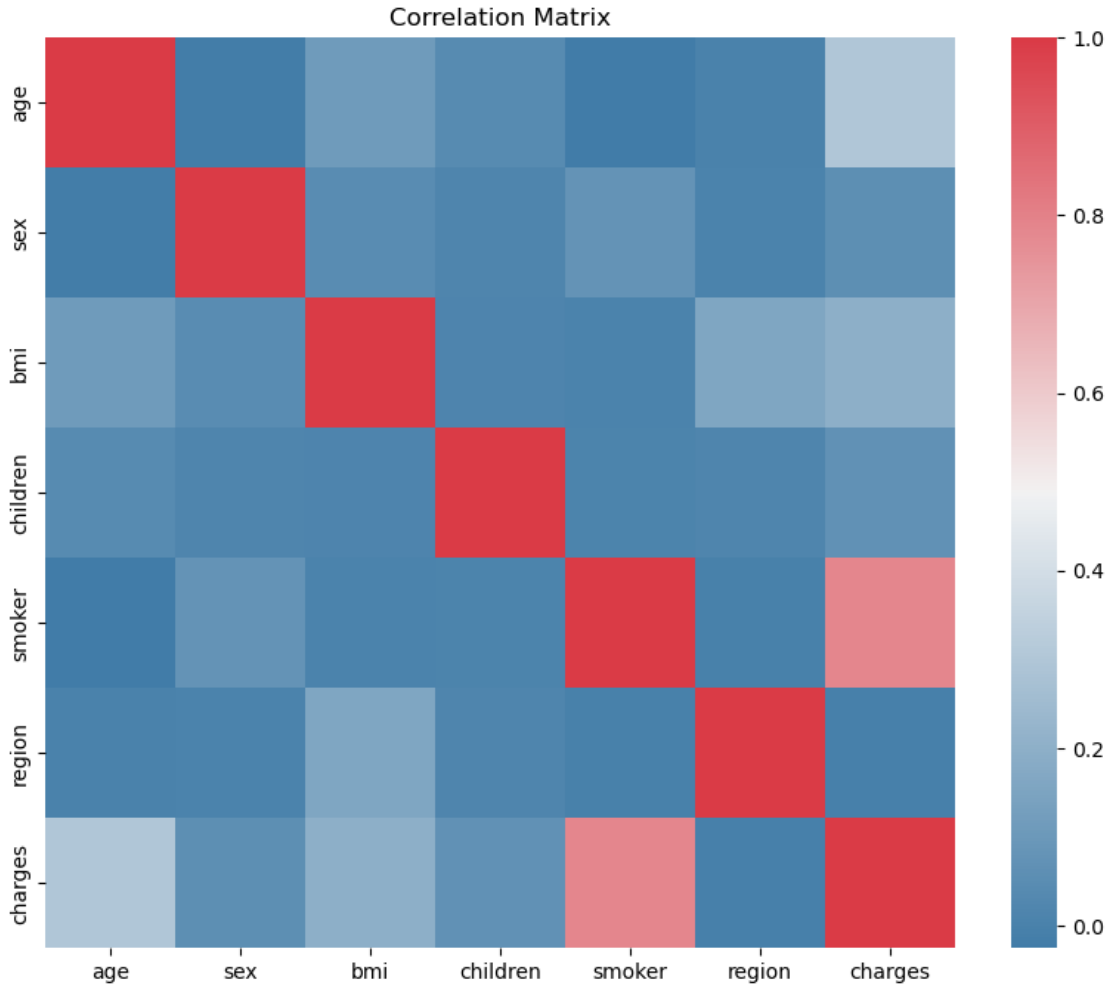
June 24, 2025

[37]:
```
import numpy as np
import pandas as pd
import os
import matplotlib.pyplot as pl
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
from sklearn.preprocessing import LabelEncoder
import matplotlib.pyplot as plt

from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import PolynomialFeatures
from sklearn.metrics import r2_score,mean_squared_error
from sklearn.ensemble import RandomForestRegressor


data = pd.read_csv("C:\\Users\\jpran\\Downloads\\Insuarance\\insurance.csv")

head = data.head()

data.isnull().sum()


# ======================================================================
# PART 1: data encoding
# ======================================================================

#sex
le = LabelEncoder()
le.fit(data.sex.drop_duplicates())
data.sex = le.transform(data.sex)

# smoker or not
le.fit(data.smoker.drop_duplicates())
data.smoker = le.transform(data.smoker)
```

```
#region
le.fit(data.region.drop_duplicates())
data.region = le.transform(data.region)
```

[39]:
```
# ========================================================================
# Part 2: Correlation Matrix
# ========================================================================

data.corr()['charges'].sort_values()


f, ax = pl.subplots(figsize=(10, 8))
corr = data.corr()
sns.heatmap(corr, mask=np.zeros_like(corr, dtype=np.bool), cmap=sns.
 ↪diverging_palette(240,10,as_cmap=True),
            square=True, ax=ax)

plt.title("Correlation Matrix")
```

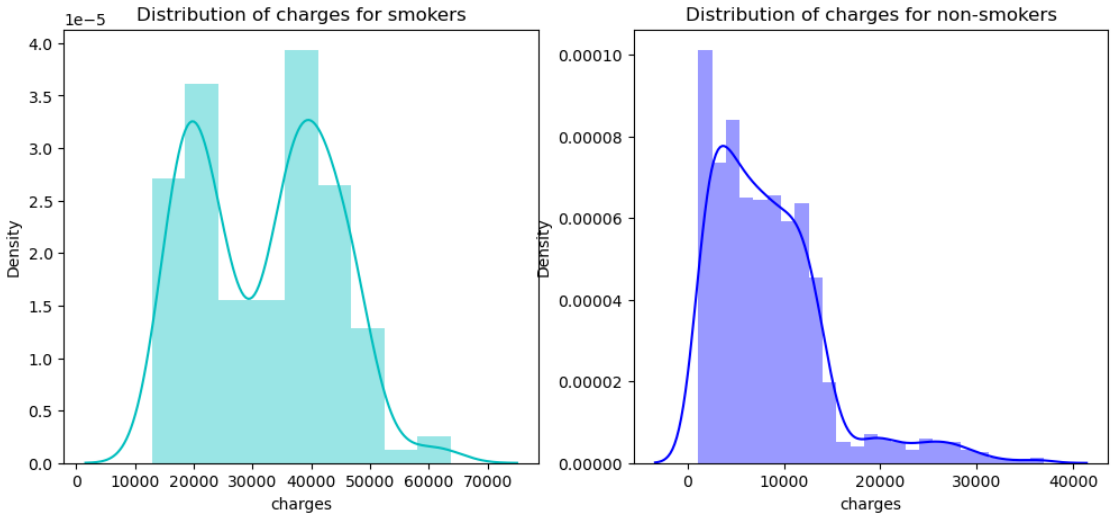
[39]:
```
Text(0.5, 1.0, 'Correlation Matrix')
```

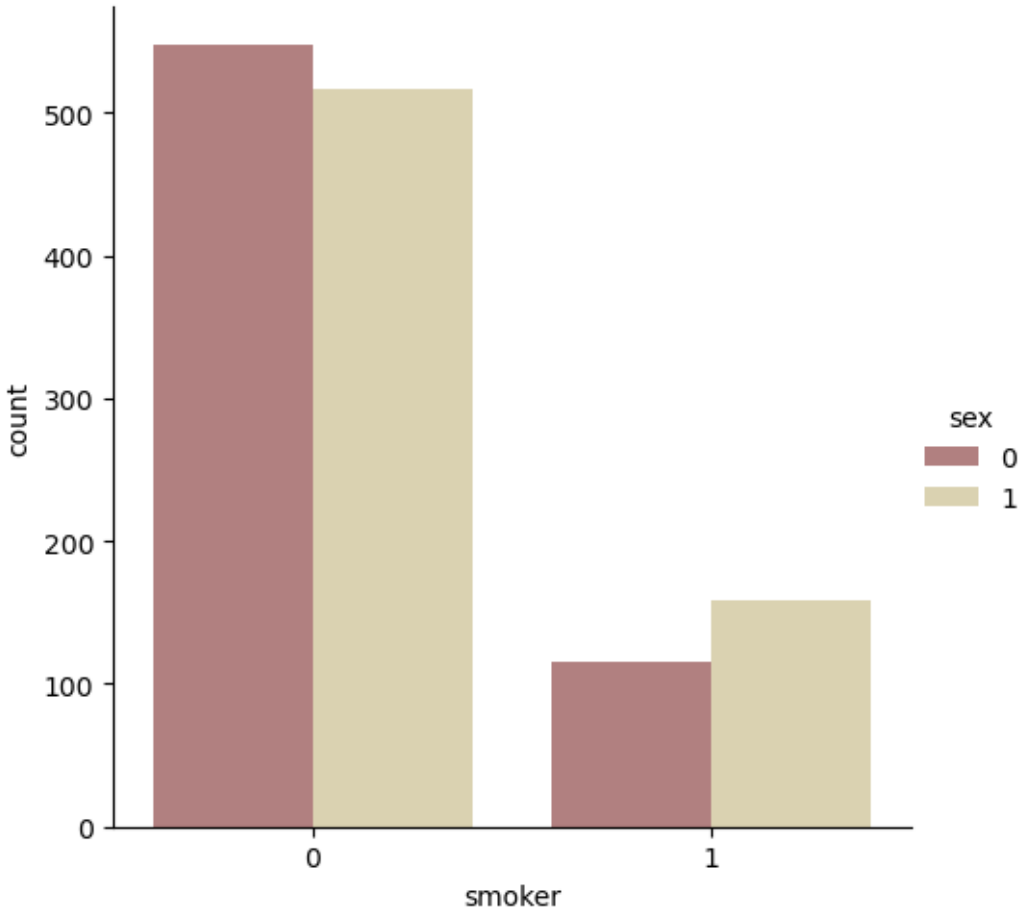


[40]:
```
# ========================================================================
# PART 4: Distribution of charges
# ========================================================================

f= pl.figure(figsize=(12,5))

ax=f.add_subplot(121)
sns.distplot(data[(data.smoker == 1)]["charges"],color='c',ax=ax)
ax.set_title('Distribution of charges for smokers')

ax=f.add_subplot(122)
sns.distplot(data[(data.smoker == 0)]['charges'],color='b',ax=ax)
ax.set_title('Distribution of charges for non-smokers')


# ========================================================================
```

```
# PART 5: Distribution of genders
# ================================================================================


sns.catplot(x="smoker", kind="count",hue = 'sex', palette="pink", data=data)
```

[40]: <seaborn.axisgrid.FacetGrid at 0x1fd7a2e5820>

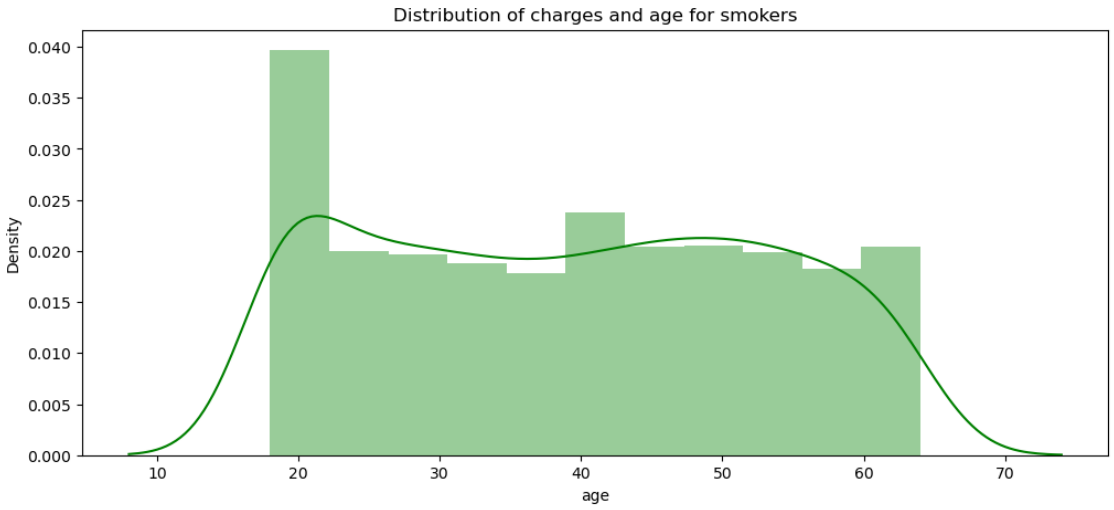


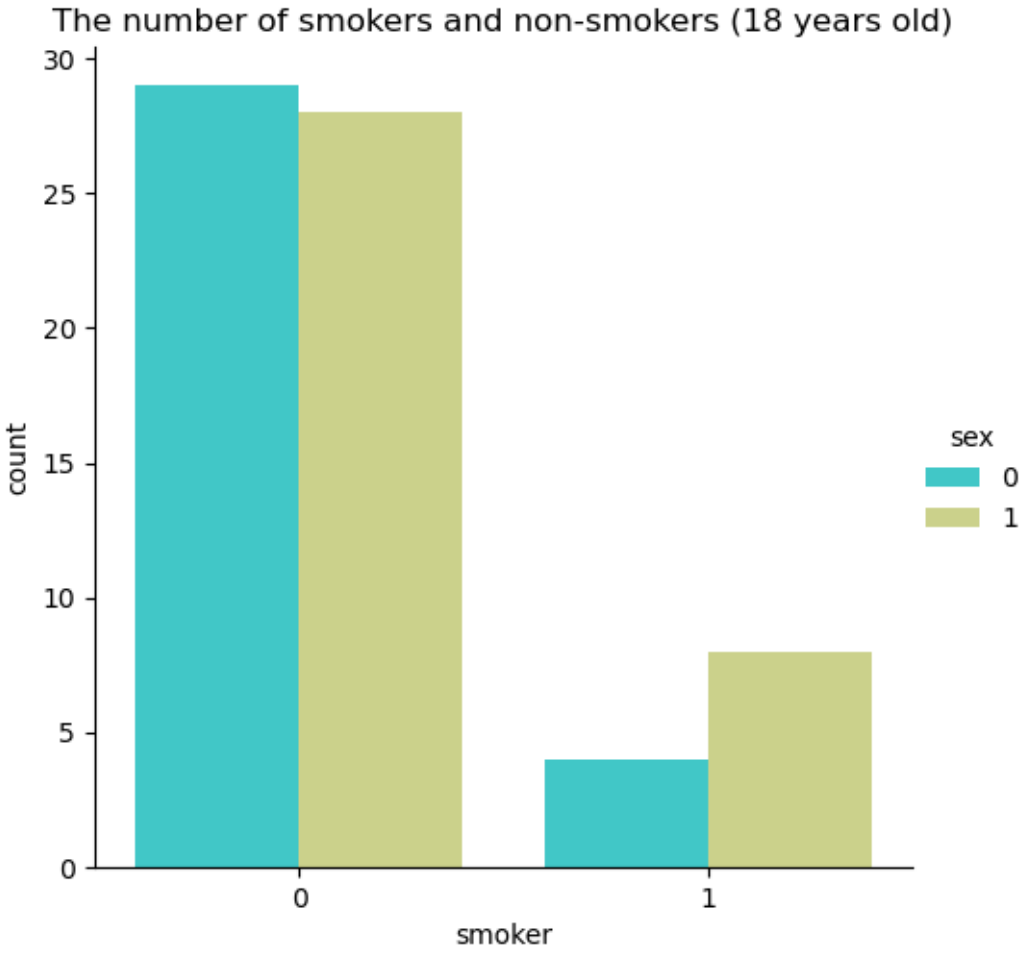


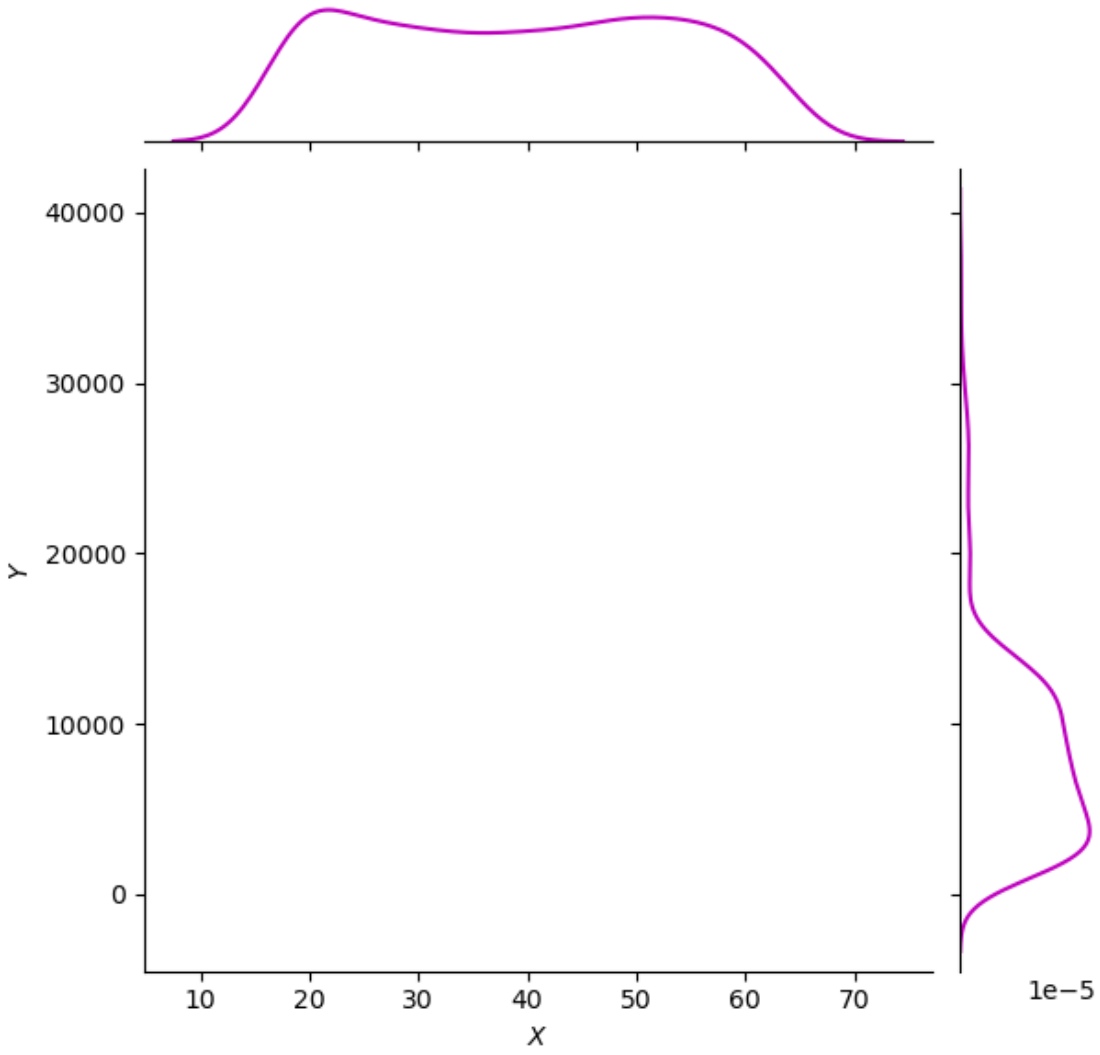
[42]:
```
# ================================================================================
# PART 5: Distribution of Ages
# ================================================================================


pl.figure(figsize=(12,5))
pl.title("Distribution of Age")
ax = sns.distplot(data["age"], color = 'g')


sns.catplot(x="smoker", kind="count",hue = 'sex', palette="rainbow",␣
 ↪data=data[(data.age == 18)])
pl.title("The number of smokers and non-smokers (18 years old)")
```

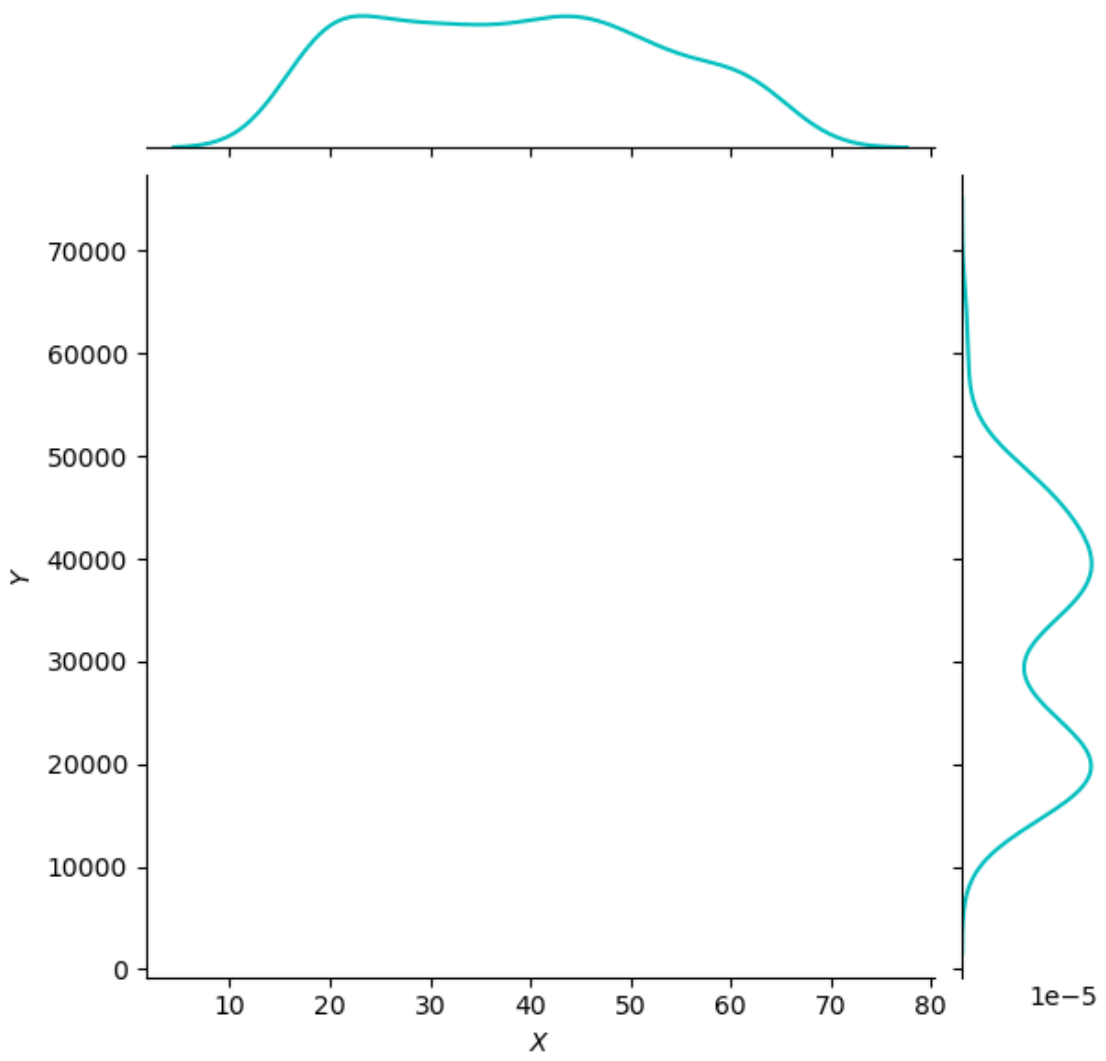
```
# ========================================================================
# PART 5-1: Distribution of Age VS Charges (non-smokers)
# ========================================================================



g = sns.jointplot(x="age", y="charges", data = data[(data.smoker ==␣
 ↪0)],kind="kde", color="m")
g.plot_joint(pl.scatter, c="w", s=30, linewidth=1, marker="+")
g.ax_joint.collections[0].set_alpha(0)
g.set_axis_labels("$X$", "$Y$")
ax.set_title('Distribution of charges and age for non-smokers')




# ========================================================================
# PART 5-2: Distribution of Age VS Charges (smokers)
# ========================================================================


g = sns.jointplot(x="age", y="charges", data = data[(data.smoker ==␣
 ↪1)],kind="kde", color="c")
g.plot_joint(pl.scatter, c="w", s=30, linewidth=1, marker="+")
g.ax_joint.collections[0].set_alpha(0)
g.set_axis_labels("$X$", "$Y$")
ax.set_title('Distribution of charges and age for smokers')
```

[42]: Text(0.5, 1.0, 'Distribution of charges and age for smokers')



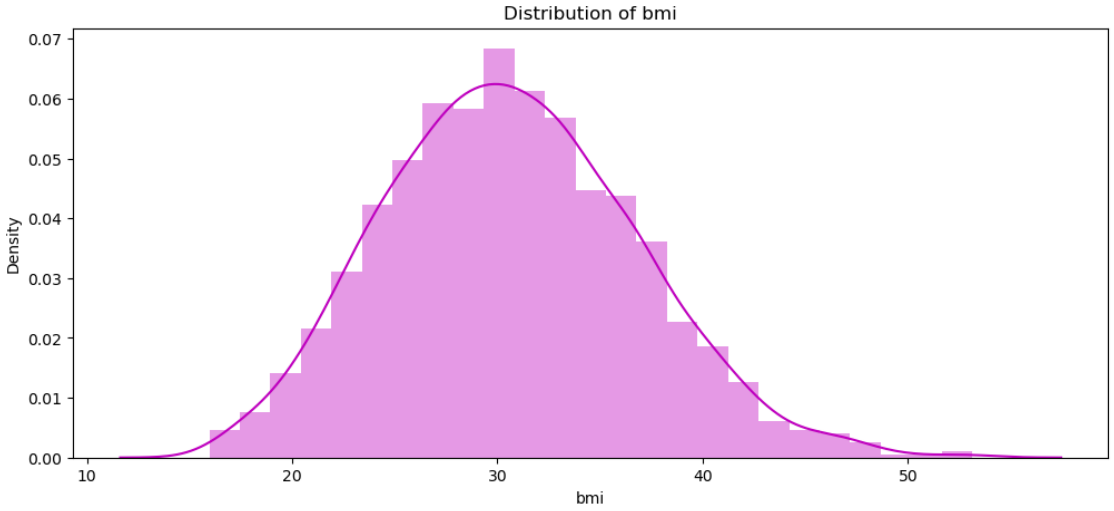

The number of smokers and non-smokers (18 years old)
30
25
20
15
10
5
0
count
0
1
smoker
sex
0
1

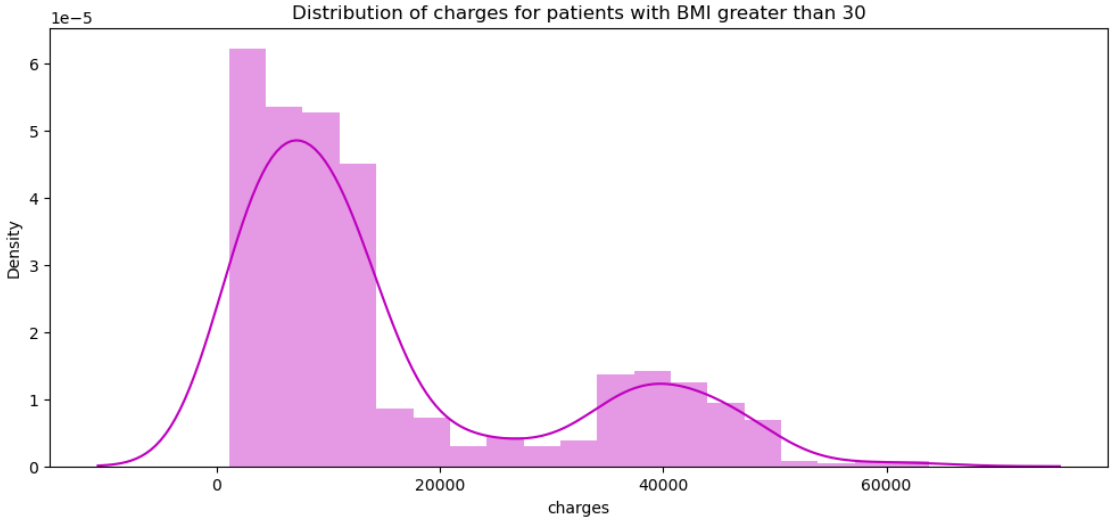

40000
30000
20000
10000
0
Y
10
20
30
40
50
60
70
X
1e−5

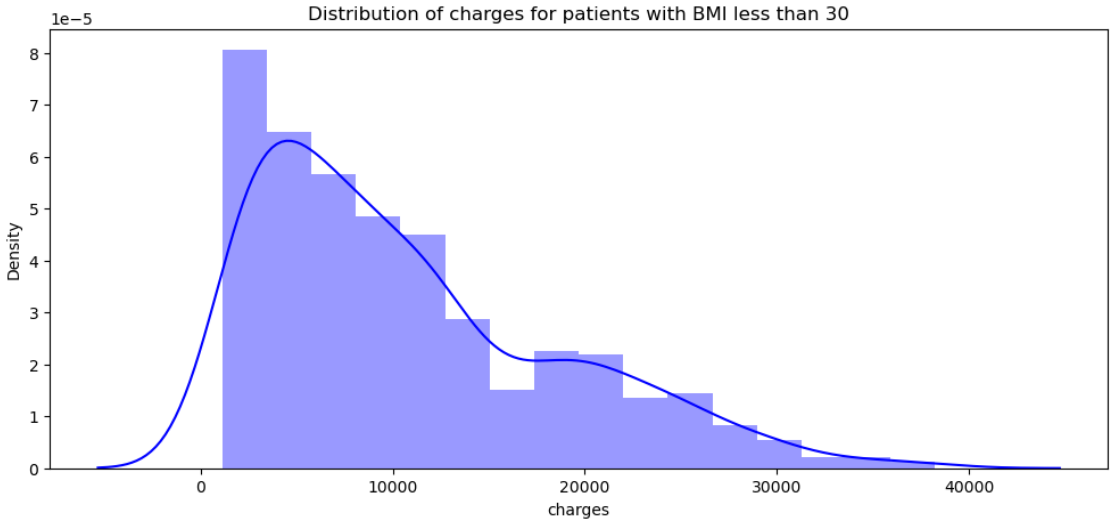


[43]:
```
# ================================================================================
# PART 6: Distribution of BMI
# ================================================================================

pl.figure(figsize=(12,5))
pl.title("Distribution of bmi")
ax = sns.distplot(data["bmi"], color = 'm')




pl.figure(figsize=(12,5))
pl.title("Distribution of charges for patients with BMI greater than 30")
ax = sns.distplot(data[(data.bmi >= 30)]['charges'], color = 'm')
```

```
pl.figure(figsize=(12,5))
pl.title("Distribution of charges for patients with BMI less than 30")
ax = sns.distplot(data[(data.bmi < 30)]['charges'], color = 'b')
```

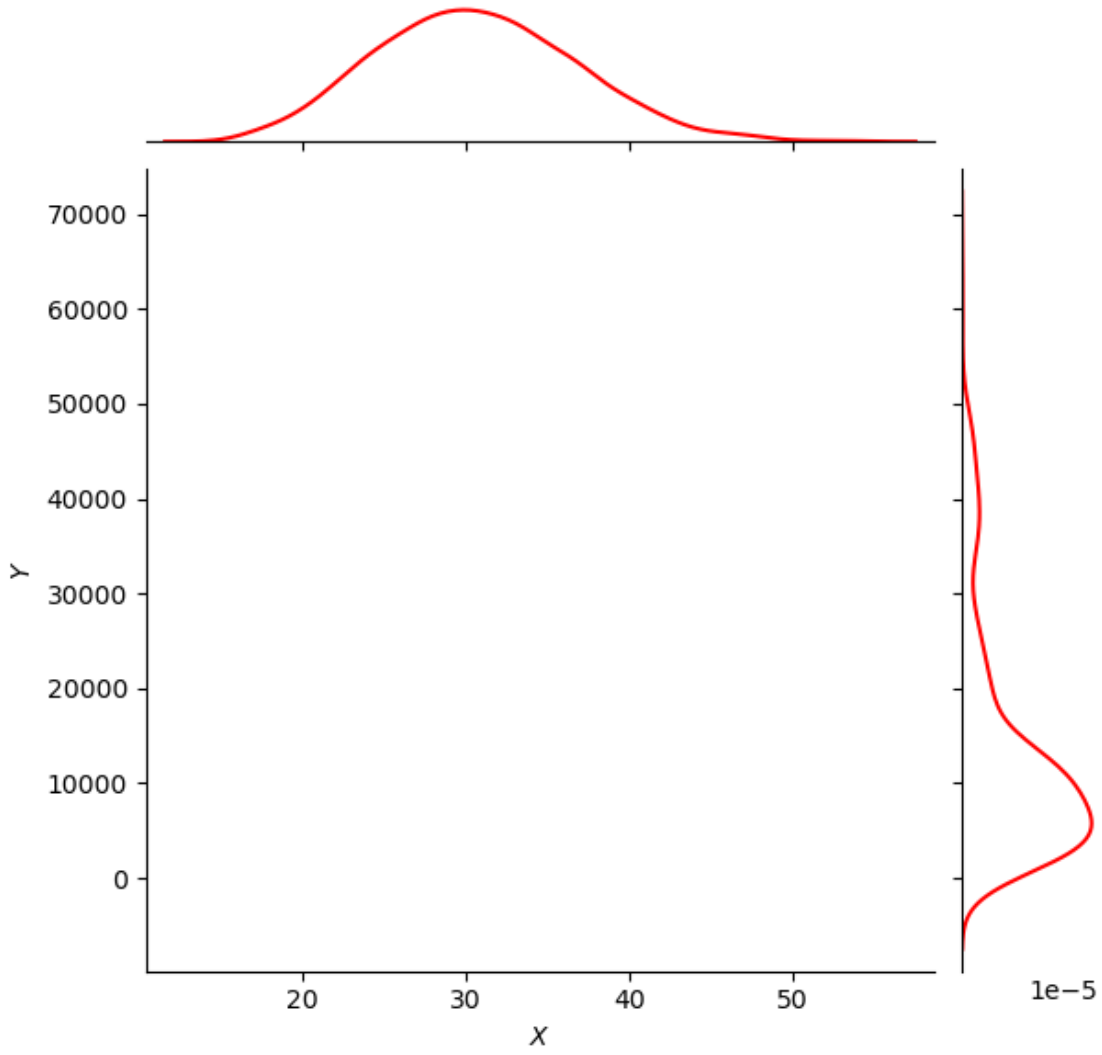






[45]:
```
# ================================================================================
# PART 7: Distribution of BMI and Charges
# ================================================================================



g = sns.jointplot(x="bmi", y="charges", data = data,kind="kde", color="r")
g.plot_joint(pl.scatter, c="w", s=30, linewidth=1, marker="+")
g.ax_joint.collections[0].set_alpha(0)
g.set_axis_labels("$X$", "$Y$")
ax.set_title('Distribution of bmi and charges')
```

[45]: Text(0.5, 1.0, 'Distribution of bmi and charges')

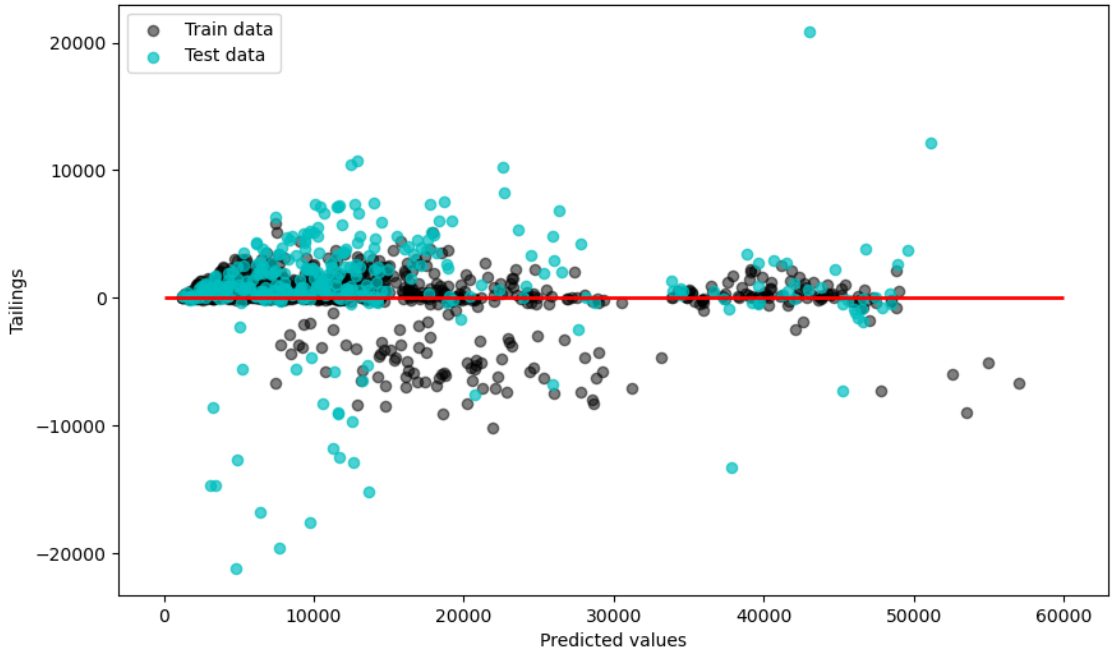


[46]:
```
# ================================================================================
# PART 8: ML Model
# ================================================================================

# LinearRegression

x = data.drop(['charges'], axis = 1)
y = data.charges

x_train,x_test,y_train,y_test = train_test_split(x,y, random_state = 0)
lr = LinearRegression().fit(x_train,y_train)

y_train_pred = lr.predict(x_train)
y_test_pred = lr.predict(x_test)
```

```
print(lr.score(x_test,y_test))


# LinearRegression (PolynomialFeatures)

X = data.drop(['charges','region'], axis = 1)
Y = data.charges



quad = PolynomialFeatures (degree = 2)
x_quad = quad.fit_transform(X)

X_train,X_test,Y_train,Y_test = train_test_split(x_quad,Y, random_state = 0)

plr = LinearRegression().fit(X_train,Y_train)

Y_train_pred = plr.predict(X_train)
Y_test_pred = plr.predict(X_test)

print(plr.score(X_test,Y_test))


# RandomForestRegressor

forest = RandomForestRegressor(n_estimators = 100,
                               # criterion = 'mse',
                               random_state = 1,
                               n_jobs = -1)
forest.fit(x_train,y_train)
forest_train_pred = forest.predict(x_train)
forest_test_pred = forest.predict(x_test)

print('MSE train data: %.3f, MSE test data: %.3f' % (
mean_squared_error(y_train,forest_train_pred),
mean_squared_error(y_test,forest_test_pred)))
print('R2 train data: %.3f, R2 test data: %.3f' % (
r2_score(y_train,forest_train_pred),
r2_score(y_test,forest_test_pred)))



pl.figure(figsize=(10,6))

pl.scatter(forest_train_pred,forest_train_pred - y_train,
          c = 'black', marker = 'o', s = 35, alpha = 0.5,
```

```
          label = 'Train data')
pl.scatter(forest_test_pred,forest_test_pred - y_test,
          c = 'c', marker = 'o', s = 35, alpha = 0.7,
          label = 'Test data')
pl.xlabel('Predicted values')
pl.ylabel('Tailings')
pl.legend(loc = 'upper left')
pl.hlines(y = 0, xmin = 0, xmax = 60000, lw = 2, color = 'red')
pl.show()
```

```
0.7962732059725786
0.8849197344147239
MSE train data: 3746684.434, MSE test data: 19965476.411
R2 train data: 0.974, R2 test data: 0.873
```



[ ]: