Importing essential libraries for EDA

In [50]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

Loading the data

In [2]:
```
df = pd.read_csv('insurance.csv')
```

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

Out[3]:

	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

The factors that might affect the health insurance cost are:-

- **Age**: Age of the beneficiary
- **Sex**: Gender of the beneficiary
- **BMI**: Body Mass Index (kg / m^2)
- **Children**: Number of children / Number of dependants
- **Smoker**: Does smoke or not
- **Region**: Residential area of the beneficiary

Analysing the data stats

In [4]:
```
df.shape
```

Out[4]:
```
(1338, 7)
```

In [5]:
```
df.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 [6]:
```
df.isnull().values.any()
```

Out[6]:
```
False
```

The datatype of the all the categories are correct, hence no changes needed. Also, there are no null valued elements

In [7]:
```
df['charges'].describe()
```

Out[7]:
```
count     1338.000000
mean     13270.422265
std      12110.011237
min       1121.873900
25%       4740.287150
50%       9382.033000
75%      16639.912515
max      63770.428010
Name: charges, dtype: float64
```

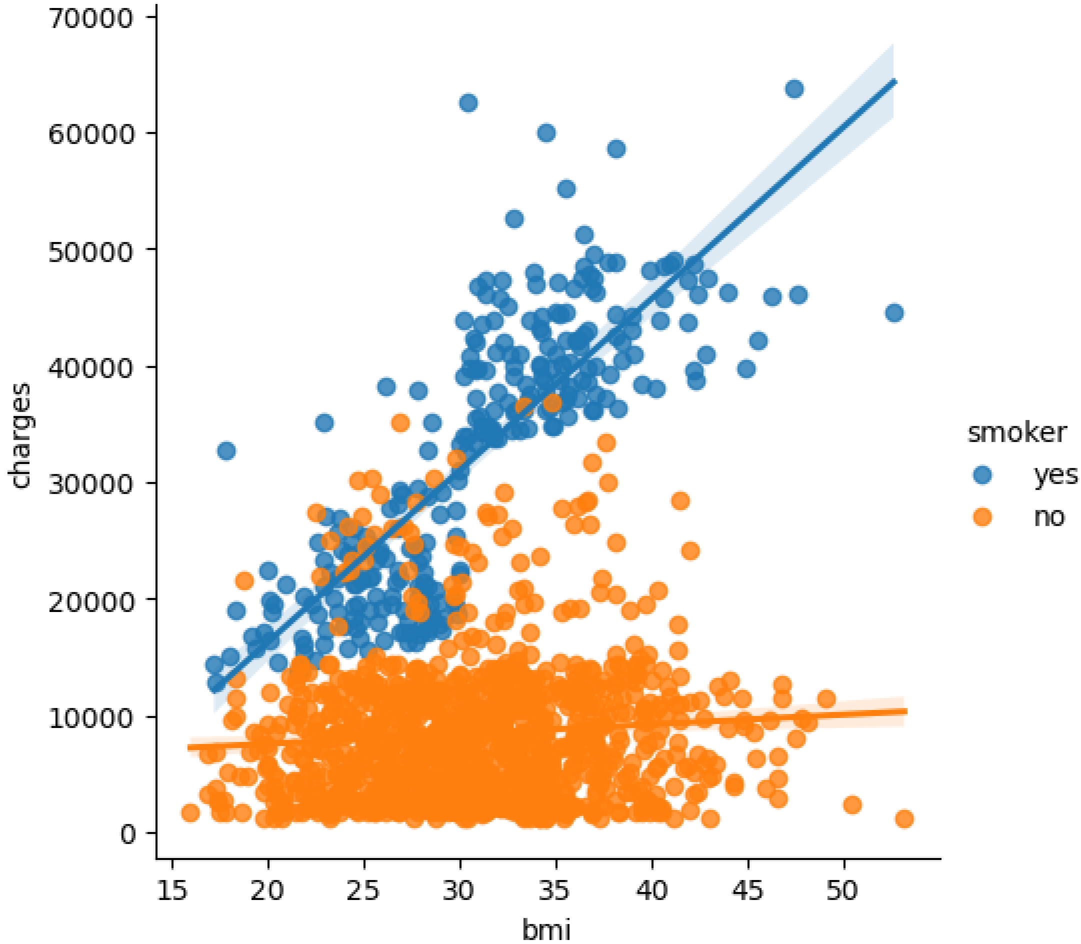
This shows a brief description of the Charges column.
Few observations:-

- The minimum cost of an insurance is 1122
- Around 75% of the charges are below 17,000
- The maximum cost of an insurance is 63770

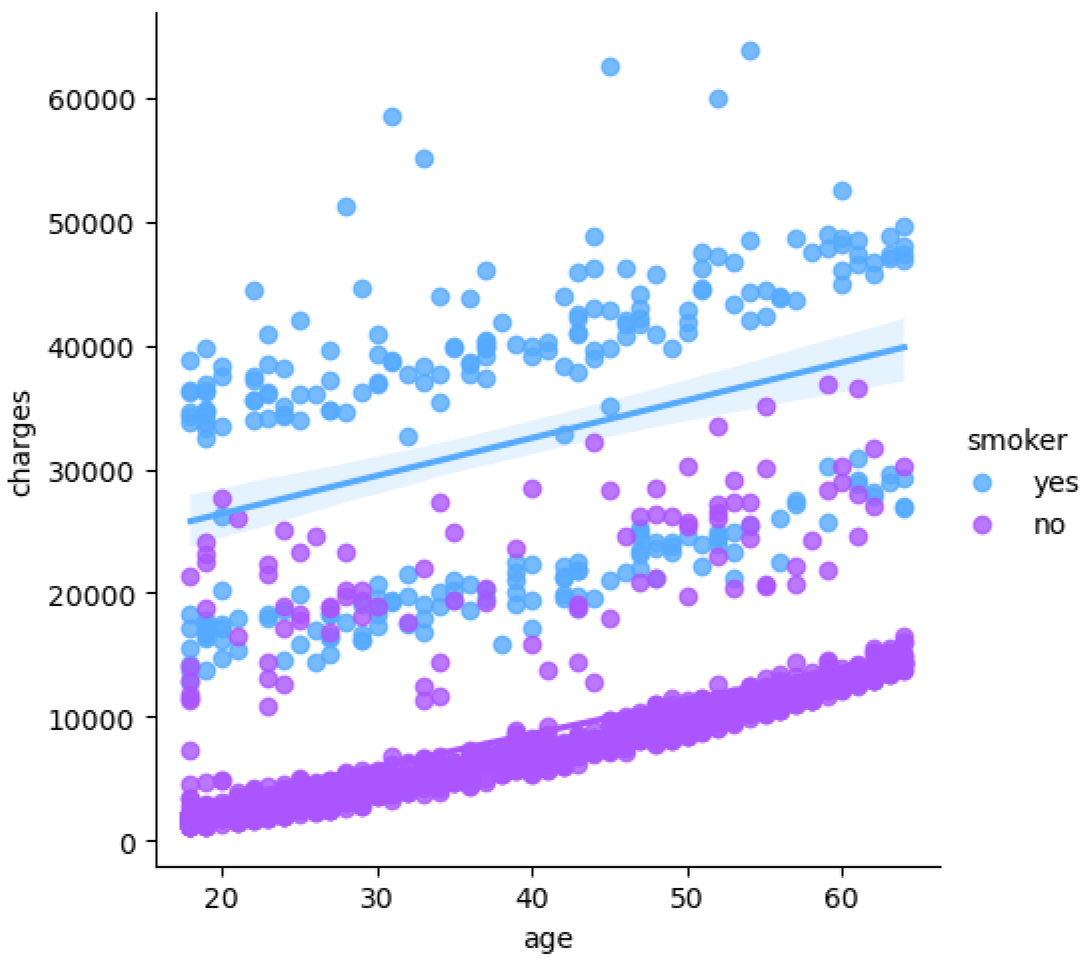
Exploratory Data Analysis

Let's analyze the medical insurance charges by age, bmi and children according to the smoking factor

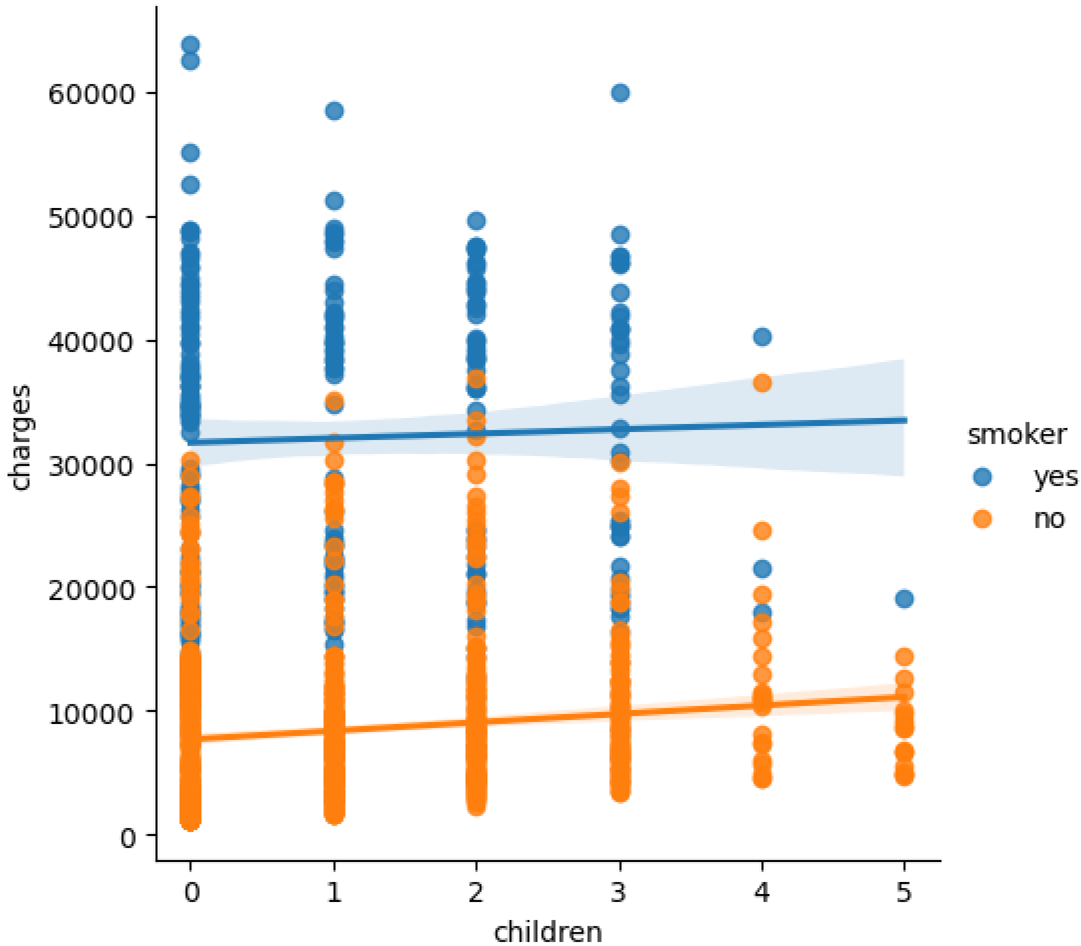
In [4]:
```
sns.lmplot( x='bmi',y='charges', data=df, hue='smoker')
plt.show()
```




```
In [7]: sns.lmplot(x='age', y='charges', hue='smoker', data= df, palette= 'cool')
        plt.show()
```

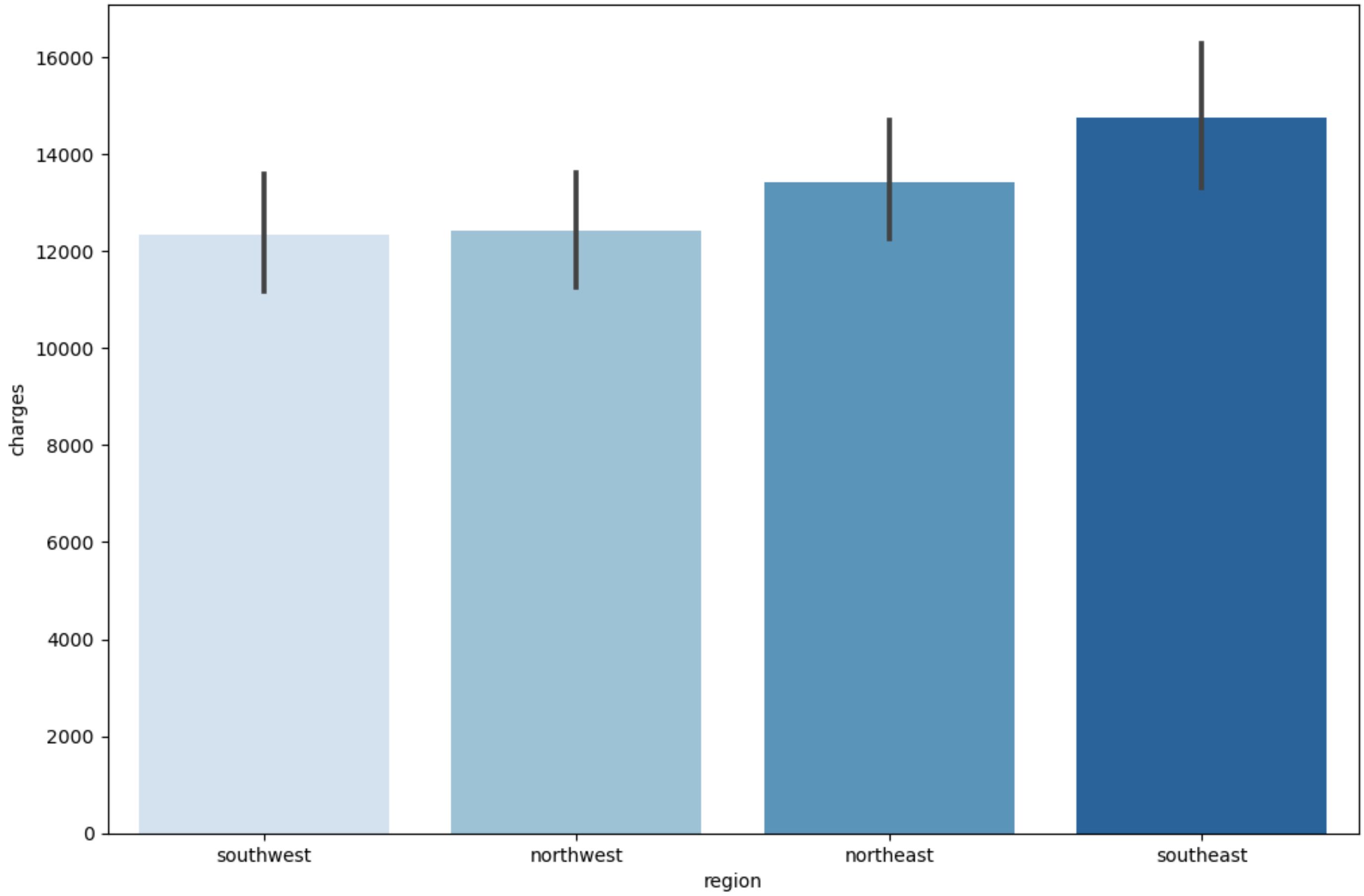


In [8]:
```
sns.lmplot(x='children', y='charges', hue='smoker', data=df)
plt.show()
```



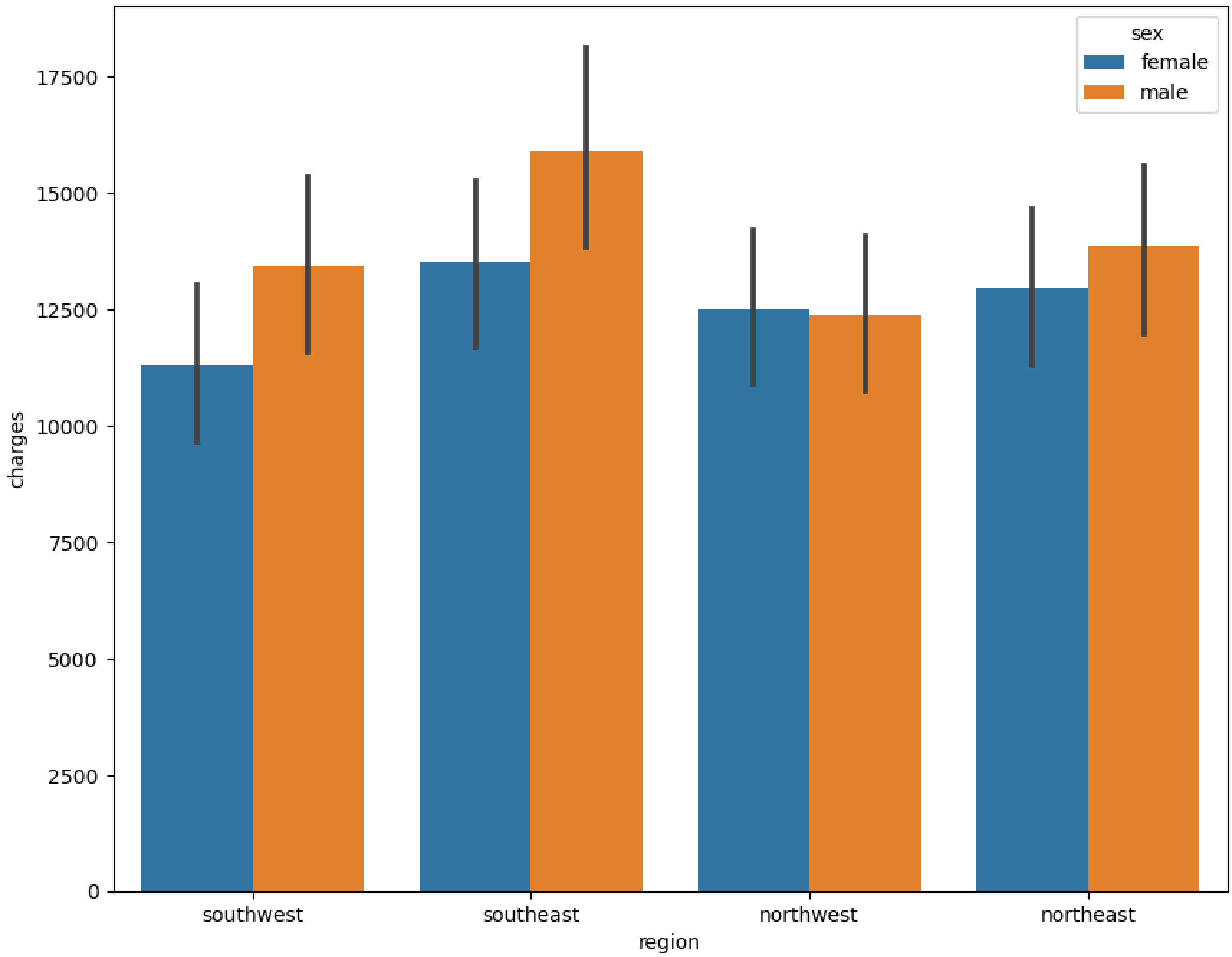
Smoking has the highest impact on the medical insurance costs, even though costs are increasing with age, bmi and children. Also people with children tend to smoke less.

In [11]:
```
f, axs = plt.subplots(1,1, figsize=(12,8))
plot_order = df.groupby('region')['charges'].sum().sort_values(ascending=True).index.val
sns.barplot(x='region', y='charges', data=df, palette='Blues', order=plot_order)
plt.show()
```

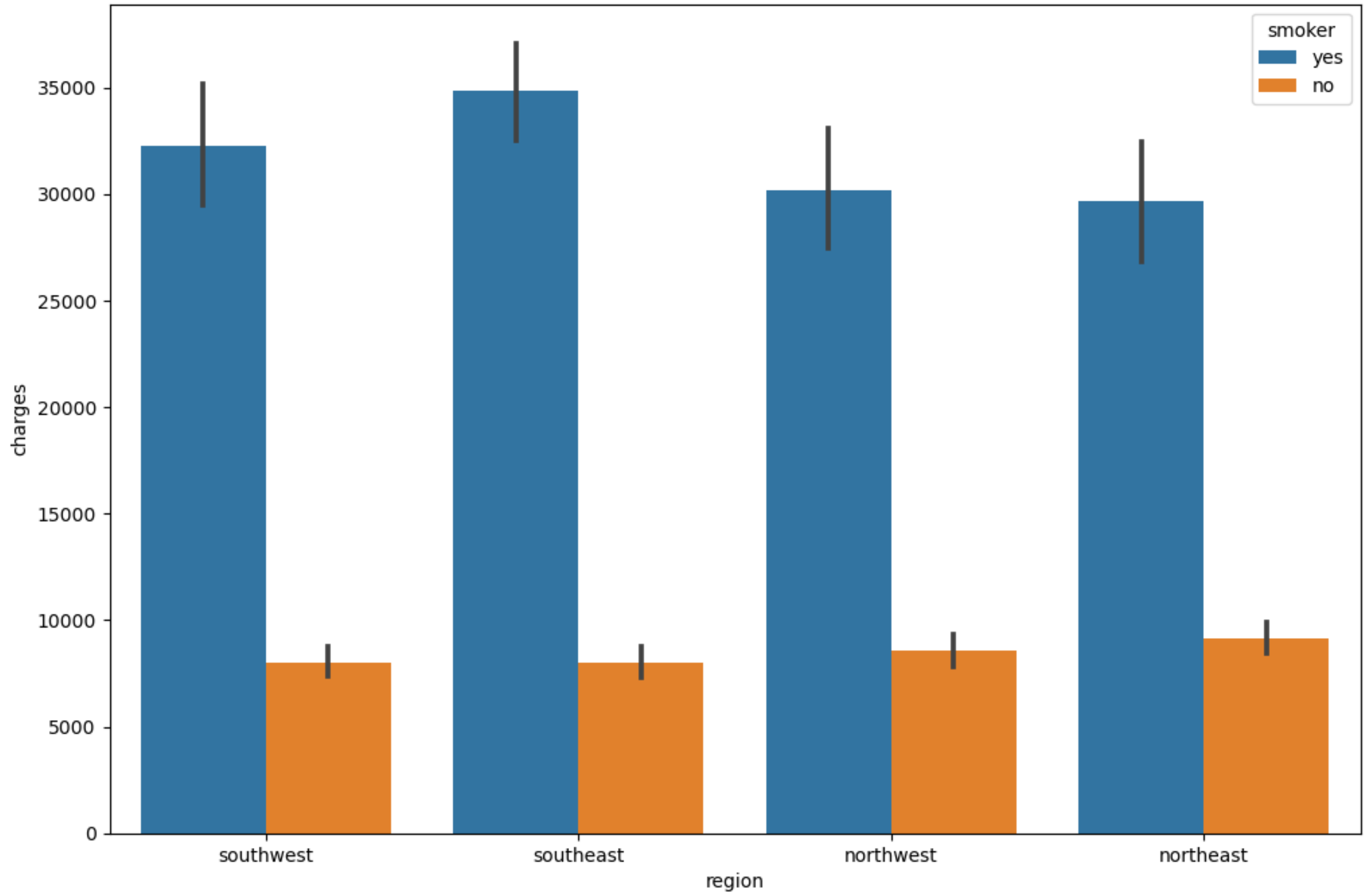


So overall the highest medical insurance charges are in the Southeast and the lowest are in the Southwest. Taking into account certain factors (sex, smoking, having children) let's see how it changes by region

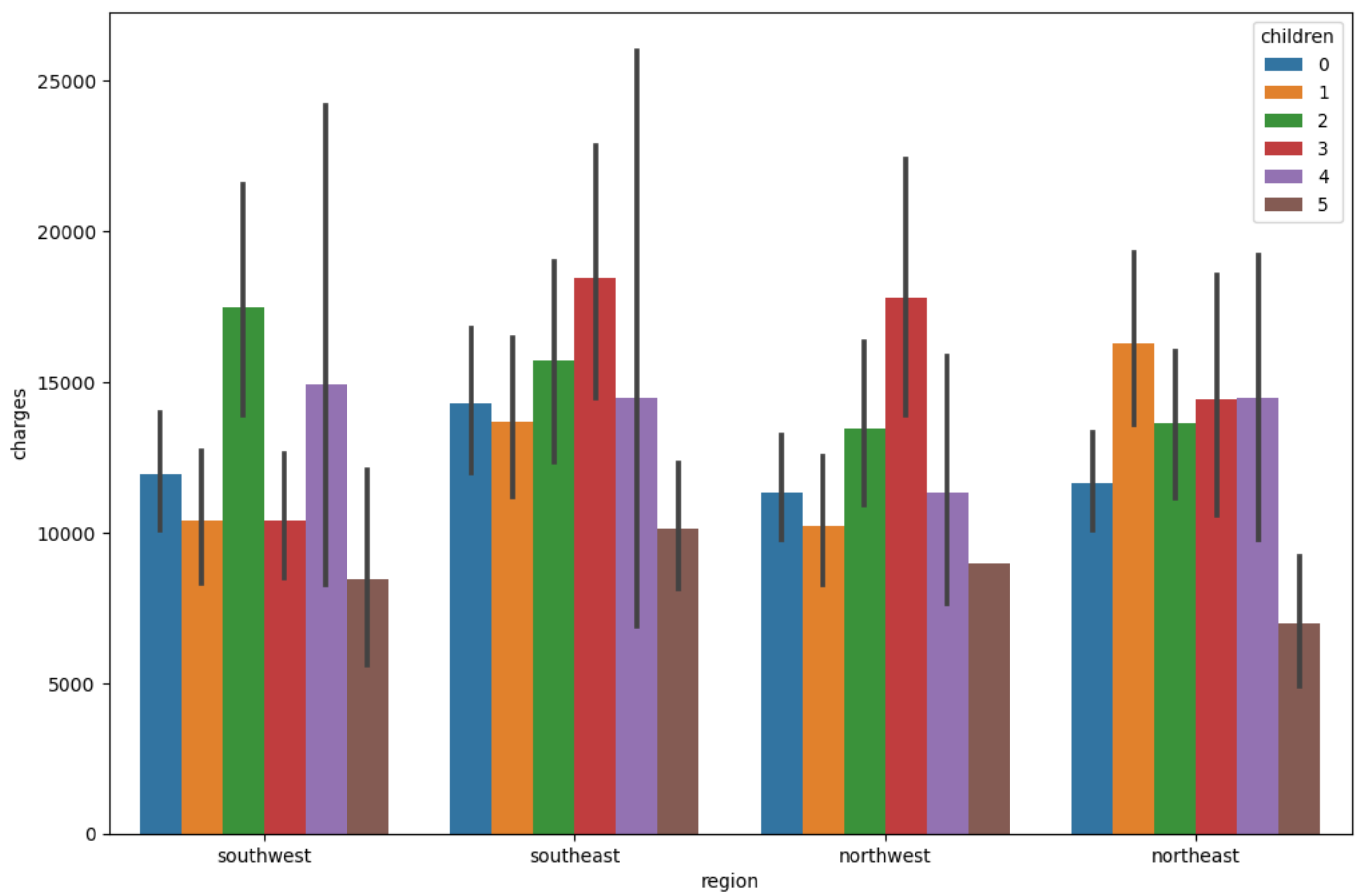
In [13]:
```
f, axs = plt.subplots(1,1, figsize=(10,8))
sns.barplot(x='region', y='charges', hue='sex', data=df)
plt.show()
```



In [14]:
```
f, axs = plt.subplots(1,1, figsize=(12,8))
sns.barplot(x='region', y='charges', hue='smoker', data=df)
plt.show()
```



In [15]:
```
f, ax = plt.subplots(1, 1, figsize=(12, 8))
sns.barplot(x='region', y='charges', hue='children', data=df)
plt.show()
```



From the above graphs we can say that the highest charges are still in Southeast. People in Southeast smoke more than people in Northeast, but people in Northeast have higher charges by gender than in Southwest and Northwest overall. And people with children tend to have higher medical costs overall as well.

Modifying the data for prediction

Converting object labels into categorical

In [17]:
```
df[['sex', 'smoker', 'region']] = df[['sex', 'smoker', 'region']].astype('category')
df.dtypes
```

Out[17]:
```
age            int64
sex         category
bmi          float64
children       int64
smoker      category
region      category
charges      float64
dtype: object
```

Converting category labels into numerical using LabelEncoder

In [20]:
```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()

le.fit(df.sex.drop_duplicates())
df.sex = le.transform(df.sex)

le.fit(df.smoker.drop_duplicates())
df.smoker = le.transform(df.smoker)

le.fit(df.region.drop_duplicates())
df.region = le.transform(df.region)

df.dtypes
```

Out[20]:
```
age           int64
sex           int64
bmi         float64
children      int64
smoker        int64
region        int64
charges     float64
dtype: object
```

In [21]:
```
f, axs = plt.subplots(1,1, figsize=(12,12))
sns.heatmap(df.corr(), annot=True, cmap='Blues')
plt.show()
```

No correlation, except with smoking

In [22]:
```
df
```

Out[22]:

	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 [23]:
```
df['region'].unique()
```
Out[23]:
```
array([3, 2, 1, 0], dtype=int64)
```

Region 0 -> Northeast
Region 1 -> Northwest
Region 2 -> Southeast
Region 3 -> Southwest

Multiple Linear Regression

This is commonly used in predictive analysis. This regression estimates are used to explain the relationship between one dependent variable and one or more independent variables

In [26]:
```
X = np.array(df.iloc[:,:-1]) # Independent variables
y = np.array(df.iloc[:,-1])   # Dependent variable
```

In [27]:
```
# Splitting the data into train and test
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=0
y_train = y_train.reshape(len(y_train),1)
y_test = y_test.reshape(len(y_test),1)
```

In [28]:
```
X.shape, y.shape
```
Out[28]:
```
((1338, 6), (1338,))
```

In [29]:
```
X_train.shape, X_test.shape, y_train.shape, y_test.shape
```
Out[29]:
```
((1003, 6), (335, 6), (1003, 1), (335, 1))
```

In [31]:
```
# Training the model
from sklearn.linear_model import LinearRegression
reg = LinearRegression()
reg.fit(X_train, y_train)
```
Out[31]:
```
▾ LinearRegression
LinearRegression()
```

In [33]:
```
y_predict = reg.predict(X_test)
```

In [36]:
```
accuracy = reg.score(X_test, y_test)
print('Accuracy = ', accuracy)
```
```
Accuracy =  0.7962732059725786
```

Ridge Regression

In [37]:
```
from sklearn.linear_model import Ridge
r = Ridge(alpha=0.5)
r.fit(X_train, y_train)
```

Out[37]:
```
▾    Ridge
Ridge(alpha=0.5)
```

In [38]:
```
ridge_accuracy = r.score(X_test, y_test)
print('Accuracy = ', ridge_accuracy)
```

```
Accuracy =  0.7961319557404143
```

Random Forest Regressor

In [39]:
```
from sklearn import metrics
from sklearn.ensemble import RandomForestRegressor as rfr
rf_reg = rfr(n_estimators = 10,random_state = 0)

rf_reg.fit(X_train, y_train)
```

Out[39]:
```
▾          RandomForestRegressor
RandomForestRegressor(n_estimators=10, random_state=0)
```

In [40]:
```
X_train.shape, X_test.shape, y_train.shape, y_test.shape
```

Out[40]:
```
((1003, 6), (335, 6), (1003, 1), (335, 1))
```

In [41]:
```
accuracy = rf_reg.score(X_test, y_test)
print(f'Accuracy : {accuracy:.3}')
```

```
Accuracy : 0.86
```

Saving the model

In [46]:
```
import pickle
fname = 'Model.pkl'
pickle.dump(rf_reg, open(fname, 'wb'))
```

Conclusion

- **Smoking** is the greatest factor that affects medical cost charges, then it's **bmi** and **age**.
- **Random Forest Regression** turned out to be the best model