Exploratory data analysis using statistical techniques

Leveraging customer information is paramount for most businesses. In the case of an insurance company, attributes of customers like the ones mentioned below can be crucial in making business decisions. Hence, knowing to explore and generate value out of such data can be an invaluable skill to have.

1. Import the necessary libraries (2 marks)

In [1]:
```
'''Importing Data Manipulattion Moduls'''
import numpy as np
import pandas as pd

'''Seaborn and Matplotlib Visualization'''
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

2. Read the data as a data frame (2 marks)

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

Out[2]: 1338

In [3]:
```
data.head(5)
```

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

Perform basic EDA which should include the following and print out your insights at every step. (28 marks)

In [4]:
```
#a) Shape of the data (2 marks)
print('Dimensions of train data:', data.shape)
```

```
Dimensions of train data: (1338, 7)
```

In [5]:
```
#b) Data type of each attribute (2 marks)
data.dtypes
```

Out[5]:
```
age           int64
sex          object
bmi         float64
children      int64
smoker       object
region       object
charges     float64
dtype: object
```

In [6]:
```
#c) Checking the presence of missing values (3 marks)
''' count of missing values column wise'''
data.isnull().sum()
```

Out[6]:
```
age         0
sex         0
bmi         0
children    0
smoker      0
region      0
charges     0
dtype: int64
```

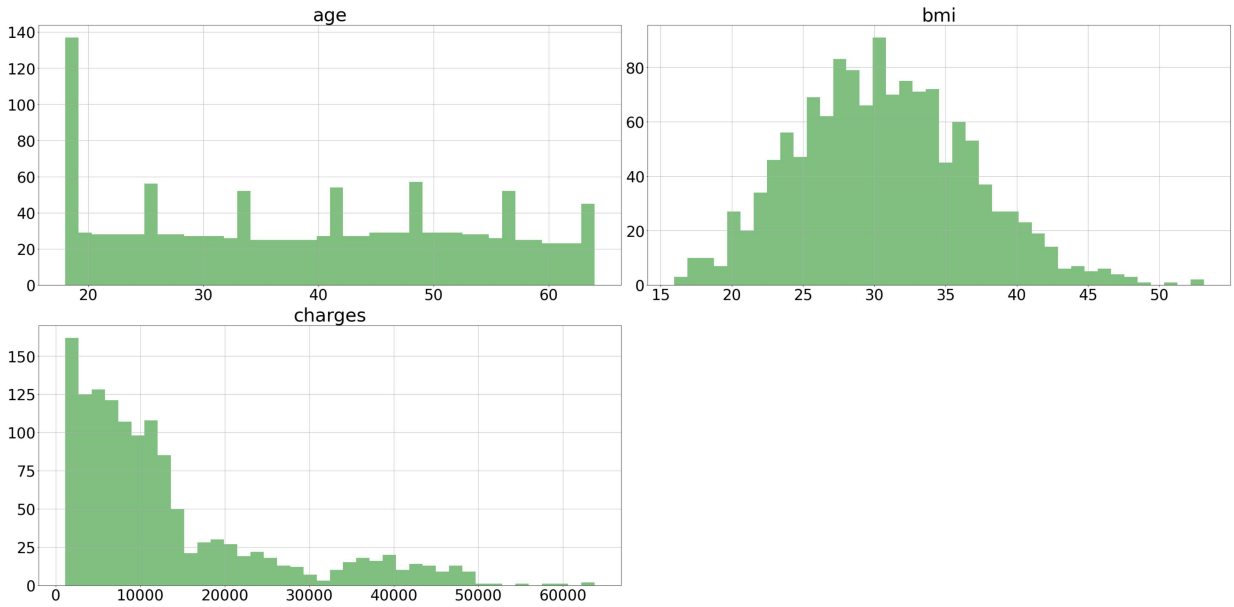
Hence, no missing value is present in this data

In [7]:
```
#d) 5 point summary of numerical attributes (3 marks)
```

In [8]:
```
#e) Distribution of 'bmi', 'age' and 'charges' columns. (4 marks)

def draw_histograms(df, variables, n_rows, n_cols):
    fig=plt.figure()
    for i, var_name in enumerate(variables):
        ax=fig.add_subplot(n_rows,n_cols,i+1)
        df[var_name].hist(bins=40,ax=ax,color = 'green',alpha=0.5, figsize = (40,
        ax.set_title(var_name, fontsize = 43)
        ax.tick_params(axis = 'both', which = 'major', labelsize = 35)
        ax.tick_params(axis = 'both', which = 'minor', labelsize = 35)
        ax.set_xlabel('')
    fig.tight_layout(rect = [0, 0.03, 1, 0.95])  # Improves appearance a bit.
    plt.show()


'''Extracting numerical variables first'''
num_merged = data.select_dtypes(include = ['int64', 'float64'])
del num_merged["children"]
display(num_merged.head(3))
print('\n')
display(num_merged.columns.values)
draw_histograms(num_merged, num_merged.columns, 19, 2)
```

	age	bmi	charges
0	19	27.90	16884.9240
1	18	33.77	1725.5523
2	28	33.00	4449.4620

```
array(['age', 'bmi', 'charges'], dtype=object)
```



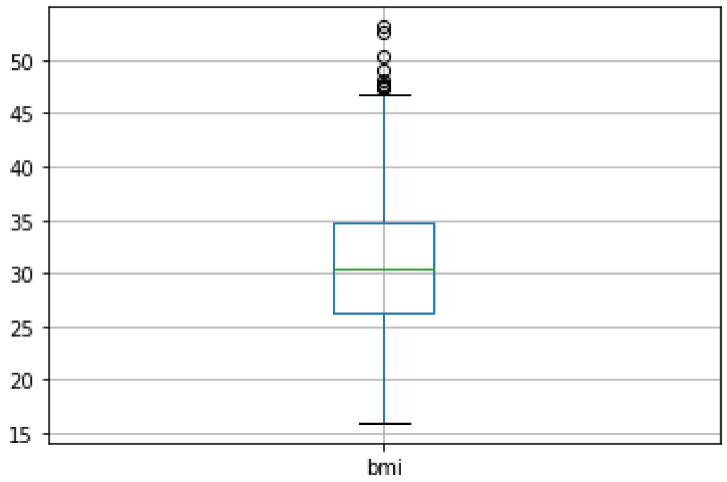
In [9]:
```
#f) Measure of skewness of 'bmi', 'age' and 'charges' columns (2 marks)
'''Skewness and Kurtosis of SalePrice'''
print("Skewness of bmi : %f" % data["bmi"].skew())
print("Skewness of age: %f" % data["age"].skew())
print("Skewness of charges: %f" % data["charges"].skew())
```

```
Skewness of bmi : 0.284047
Skewness of age: 0.055673
Skewness of charges: 1.515880
```

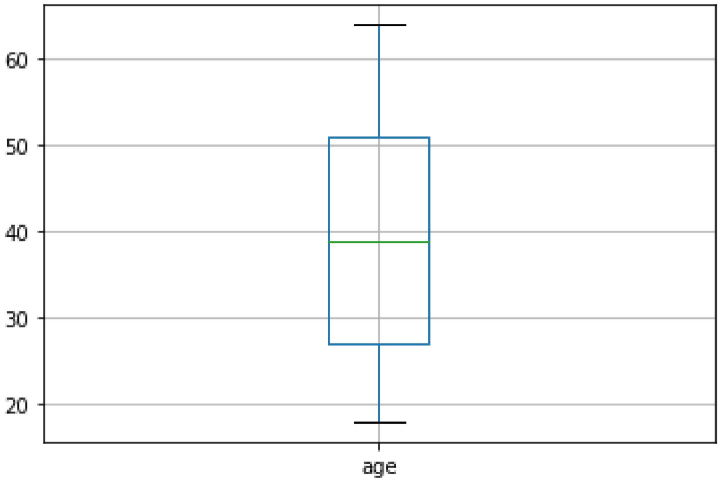
In [10]:
```
#g. Checking the presence of outliers in 'bmi', 'age' and 'charges columns (4 mar

bmi_low = data["bmi"].quantile(0.1)
bmi_hi  = data["bmi"].quantile(0.9)
print("Lower BMI threshold :", bmi_low)
print("Higher BMI threshold :",bmi_hi )
bmi_outliers = data["bmi"][(data["bmi"] < bmi_hi) & (data["bmi"] > bmi_low)]
print(bmi_outliers)
```

```
Lower BMI threshold : 22.99
Higher BMI threshold : 38.619499999999995
0       27.90
1       33.77
2       33.00
4       28.88
5       25.74
        ...
1333    30.97
1334    31.92
1335    36.85
1336    25.80
1337    29.07
Name: bmi, Length: 1068, dtype: float64
```

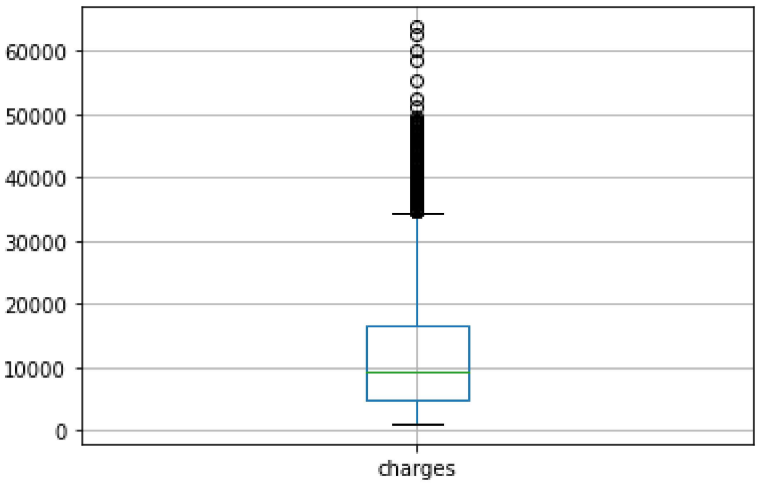
In [11]:
```
boxplot = data.boxplot(column=['bmi'])
```



In [12]:
```
boxplot = data.boxplot(column=['age'])
```



In [13]:
```
boxplot = data.boxplot(column=['charges'])
```



In [14]:
```
Q1 = data['bmi'].quantile(0.25)
Q3 = data['bmi'].quantile(0.75)
IQR = Q3 - Q1    #IQR is interquartile range.

filter = (data['bmi'] >= Q1 - 1.5 * IQR) & (data['bmi'] <= Q3 + 1.5 *IQR)
print("Number of outliers in BMI :",len(data)-len(data.loc[filter]))
```

```
Number of outliers in BMI : 9
```

In [15]:
```
Q1 = data['age'].quantile(0.25)
Q3 = data['age'].quantile(0.75)
IQR = Q3 - Q1    #IQR is interquartile range.

filter = (data['age'] >= Q1 - 1.5 * IQR) & (data['age'] <= Q3 + 1.5 *IQR)
print("Number of outliers in Age :",len(data)-len(data.loc[filter]))
```

```
Number of outliers in Age : 0
```

In [16]:
```
Q1 = data['charges'].quantile(0.25)
Q3 = data['charges'].quantile(0.75)
IQR = Q3 - Q1    #IQR is interquartile range.

filter = (data['charges'] >= Q1 - 1.5 * IQR) & (data['charges'] <= Q3 + 1.5 *IQR)
print("Number of outliers in Charges :",len(data)-len(data.loc[filter]))
```

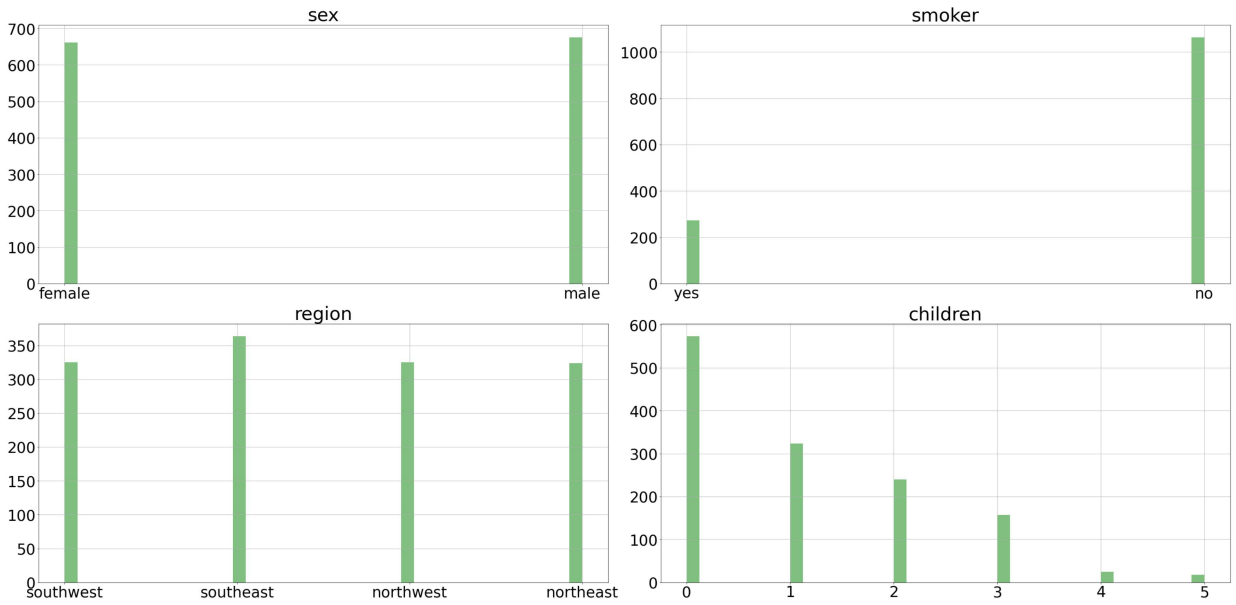
```
Number of outliers in Charges : 139
```

In [17]:
```
# Distribution of categorical columns (include children) (4 marks

'''Extracting categorical variables first'''
num_merged = data.select_dtypes(include = ['object'])
num_merged["children"] = data["children"]
display(num_merged.head(3))
print('\n')
display(num_merged.columns.values)
draw_histograms(num_merged, num_merged.columns, 19, 2)
```

```
<ipython-input-17-847313aaae0e>:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  num_merged["children"] = data["children"]
```

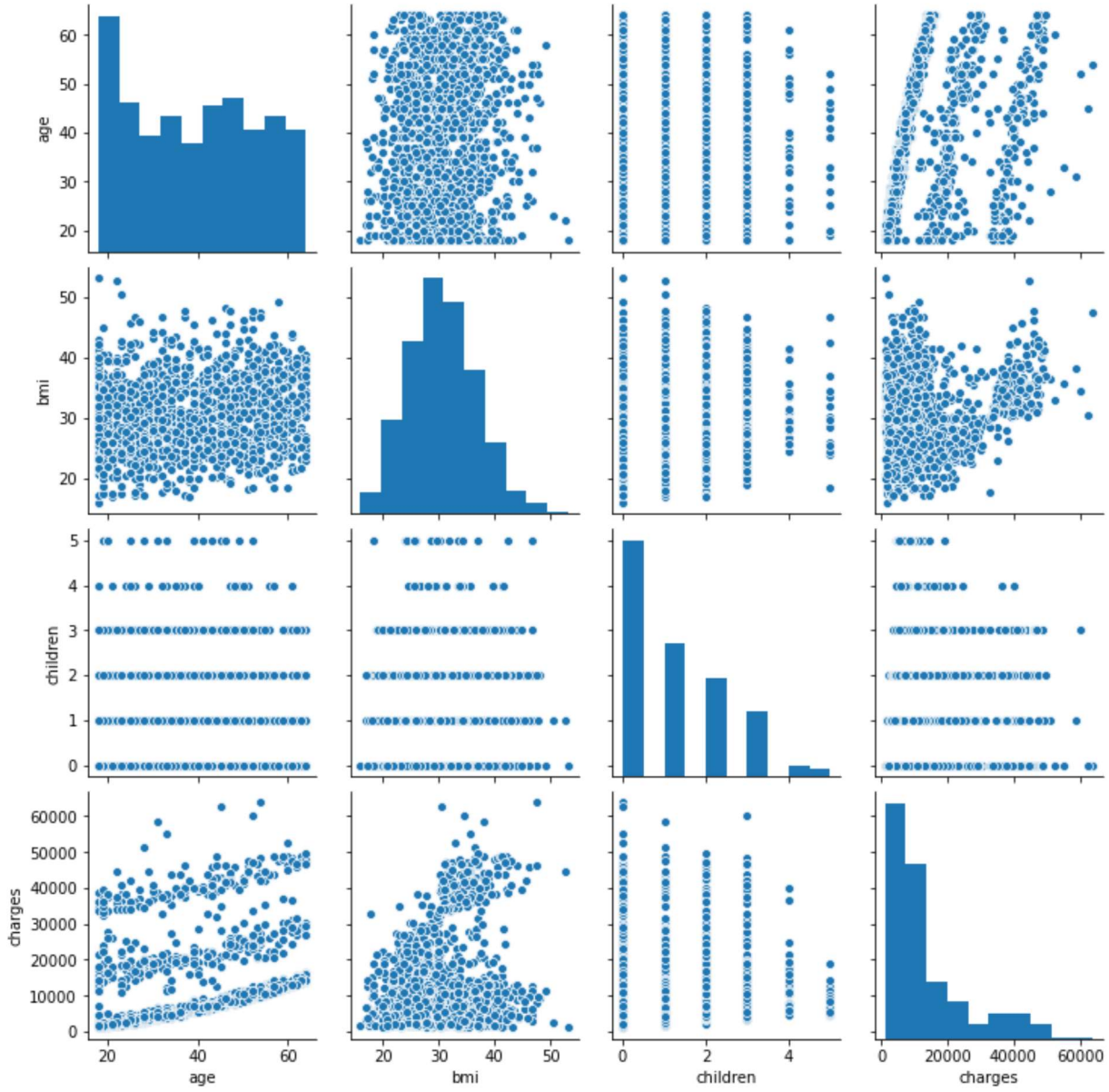
	sex	smoker	region	children
0	female	yes	southwest	0
1	male	no	southeast	1
2	male	no	southeast	3

```
array(['sex', 'smoker', 'region', 'children'], dtype=object)
```



In [18]:
```
# i.Pair plot that includes all the columns of the data frame (4 marks)
sns.pairplot(data)
```

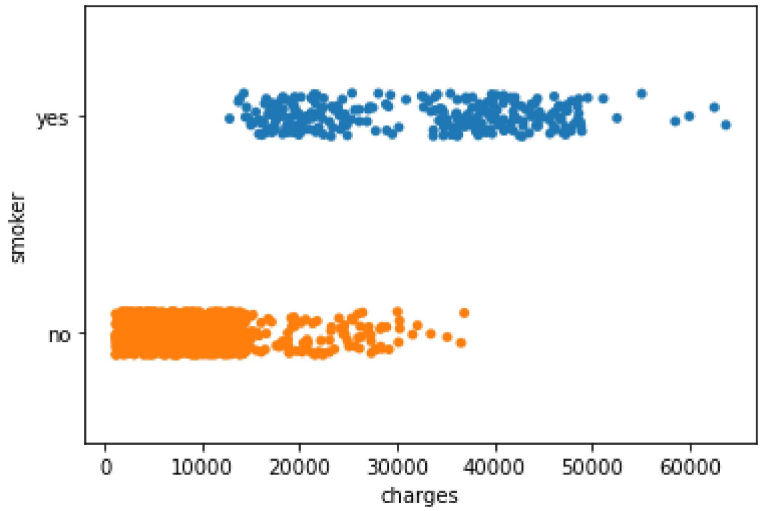
Out[18]: <seaborn.axisgrid.PairGrid at 0x2250d762580>



4. Answer the following questions with statistical evidence (28 marks)

In [23]:
```
#a) Do charges of people who smoke differ significantly from the people who don't
sns.stripplot(data['charges'], data['smoker'])
```

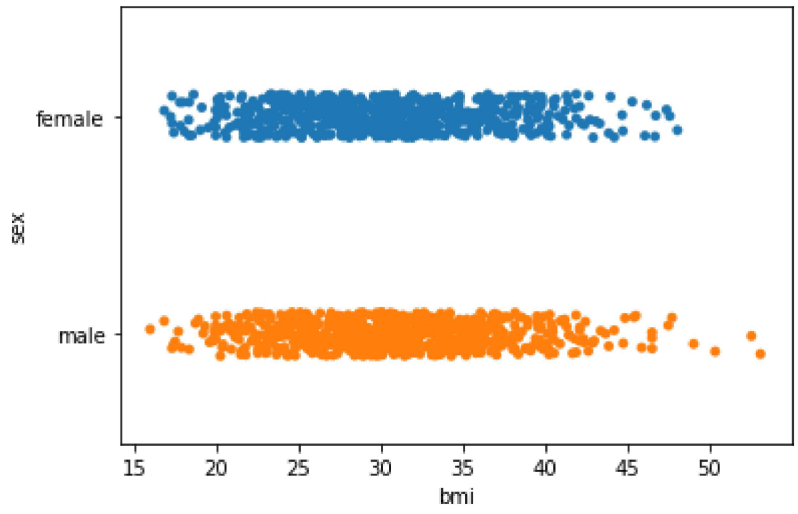
Out[23]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x225026824f0>
```



The above chart shows that the charges for people who smoke ranges from 15,000 -50,000 approximately but for rest of the people the charges lies in the range of 0 - 40,000 approximately. Hence, the charges for people who smoke is little high than people who don't. There is some difference which we can infer from the chart but it is not much significant as 2 groups have many common values together.

In [24]:
```
#b) Does bmi of males differ significantly from that of females? (7 marks)
sns.stripplot(data['bmi'], data['sex'])
```

Out[24]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x225026453d0>
```



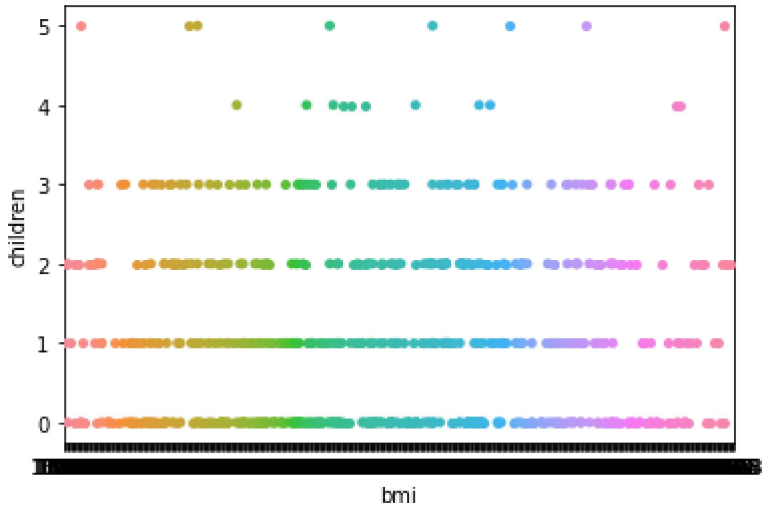
In [26]:
```
#c) Is the proportion of smokers significantly different in different genders? (7
print("Total count of smokers is ", data[data['smoker']=='yes'].shape[0])
print("Total count of male smokers is ", data[data['smoker']=='yes'][data['sex']=
print("Total count of female smokers is ", data[data['smoker']=='yes'][data['sex'
print("Proportion of smokers who are male is ", (data[data['smoker']=='yes'][data
print("Proportion of smokers who are female is ", (data[data['smoker']=='yes'][da
```

```
Total count of smokers is  274
Total count of male smokers is  159
Total count of female smokers is  115
Proportion of smokers who are male is  0.5802919708029197
Proportion of smokers who are female is  0.4197080291970803

<ipython-input-26-90de4538ab3d>:3: UserWarning: Boolean Series key will be rein
dexed to match DataFrame index.
  print("Total count of male smokers is ", data[data['smoker']=='yes'][data['se
x']=='male'].shape[0])
<ipython-input-26-90de4538ab3d>:4: UserWarning: Boolean Series key will be rein
dexed to match DataFrame index.
  print("Total count of female smokers is ", data[data['smoker']=='yes'][data
['sex']=='female'].shape[0])
<ipython-input-26-90de4538ab3d>:5: UserWarning: Boolean Series key will be rein
dexed to match DataFrame index.
  print("Proportion of smokers who are male is ", (data[data['smoker']=='yes']
[data['sex']=='male'].shape[0])/data[data['smoker']=='yes'].shape[0])
<ipython-input-26-90de4538ab3d>:6: UserWarning: Boolean Series key will be rein
dexed to match DataFrame index.
  print("Proportion of smokers who are female is ", (data[data['smoker']=='ye
s'][data['sex']=='female'].shape[0])/data[data['smoker']=='yes'].shape[0])
```

From the above chart, it is clear that there is no significant difference in BMI for male and female genders, so no relationship exists between the two.

In [27]:
```
#d) Is the distribution of bmi across women with no children, one child and two c
sns.stripplot(data['bmi'], data[data['sex']=='female']['children'])
```

Out[27]: <matplotlib.axes._subplots.AxesSubplot at 0x22502d78070>



Yes, from the above chart, we can infer that the distributions of 'bmi' are nearly same across women with 0, 1 or 2 children.

In [ ]:

In [ ]: