PROJECT 2

Version 2

Axis Insurance

Prepared by: Vincent Taylor

February 26, 2021

Version History

Date	Version #	Description	Author
02/26/2021	1.0a	Final Version	Vincent Taylor

TABLE OF CONTENTS

1 Introduction

This work request was submitted by Great Learning Facility

1.1 Objectives

- Explore the dataset and extract insights using Exploratory Data Analysis.
- Prove (or disprove) that the medical claims made by the people who smoke is greater than those who don't? [Hint- Formulate a hypothesis and prove/disprove it]
- Prove (or disprove) with statistical evidence that the BMI of females is different from that of males.
- Is the proportion of smokers significantly different across different regions? [Hint : Create a contingency table/cross tab, Use the function : stats.chi2_contingency()]
- Is the mean BMI of women with no children, one child, and two children the same? Explain your answer with statistical evidence.

*Consider a significance level of 0.05 for all tests.

1.2 System Description

Not Applicable

1.3 Scope

Use data from the Ais Insurance dataset.

- Age - This is an integer indicating the age of the primary beneficiary (excluding those above 64 years, since they are generally covered by the government).
- Sex - This is the policy holder's gender, either male or female.
- BMI - This is the body mass index (BMI), which provides a sense of how over or under-weight a person is relative to their height. BMI is equal to weight (in kilograms) divided by height (in meters) squared. An ideal BMI is within the range of 18.5 to 24.9.
- Children - This is an integer indicating the number of children / dependents covered by the insurance plan.
- Smoker - This is yes or no depending on whether the insured regularly smokes tobacco.
- Region - This is the beneficiary's place of residence in the U.S., divided into four geographic regions - northeast, southeast, southwest, or northwest.
- Charges - Individual medical costs billed to health insurance

1.4 Issues

None foreseen at this time.

1.5 Dependencies

N/A.

2 Execution

2.1 Jupyter Notebook

3 Environment

3.1 Hardware

Laptop and Desktop work-stations

3.2 Servers

Not Applicable

3.3 Special Notes and Considerations

Not Applicable

4 Sources use to complete the project

- Course material provided by Great Learning
- 2nd Edition Python Crash Course
- www.kaggle.com [Studied concepts only]
- 2nd Edition Powerful Python, Author Aaron Maxwell
- White paper T Test Non and Parametric by Bonnie Jie Ma
- White paper Chi-Square Test by George Pipis

EXPLORATORY DATA ANALYSIS CHARTS

Objectives

Explore the dataset and extract insights using Exploratory Data Analysis.
Prove (or disprove) that the medical claims made by the people who smoke is greater than those who don't? [Hint- Formulate a hypothesis and prove/disprove it]
Prove (or disprove) with statistical evidence that the BMI of females is different from that of males.
Is the proportion of smokers significantly different across different regions? [Hint : Create a contingency table/cross tab, Use the function : stats.chi2_contingency()]
Is the mean BMI of women with no children, one child, and two children the same? Explain your answer with statistical evidence.

*Consider a significance level of 0.05 for all tests.

```
#Import liabraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import statsmodels.api as sm
import scipy.stats as stat
from scipy.stats import ttest_ind, mannwhitneyu, f_oneway, f, shapiro, levene, chi2_contingency
import warnings
warnings.filterwarnings('ignore') # To supress warnings
pd.set_option('display.float_format', lambda x: '%.5f' % x) # To supress numerical display in scientific notations

sns.set(style="ticks", color_codes=True)
sns.set_color_codes()

from statsmodels.stats.power import ttest_power
from statsmodels.formula.api import ols
from statsmodels.stats.proportion import proportions_ztest
from statsmodels.stats.multicomp import pairwise_tukeyhsd

#setting up for customized printing
from IPython.display import Markdown, display
from IPython.display import HTML

def printfunc(string, color=None):
    colorset = "<span style='color:{}'>{}</span>".format(color, string)
    display(Markdown(colorset))
```

```
# Load the dataset
df = pd.read_csv('Axisinsurance.csv')
df1 = df.copy()
df1

# Set categorical variables to numbers
df1['sex'] = df1['sex'].replace('male', 0)
df1['sex'] = df1['sex'].replace('female', 1)
df1['smoker'] = df1['smoker'].replace('no', 0)
df1['smoker'] = df1['smoker'].replace('yes', 1)

# Sklearn Label Encoding
from sklearn.preprocessing import LabelEncoder # import Label encoder
labelencoder = LabelEncoder()

# Returns label encodes variable. Change column name Region to RegionId
df1['regionId'] = labelencoder.fit_transform(df1.region)

df1.drop(['region'], axis=1,inplace=True)

df1['regionId'].head(5)

# Number of entries of each class
df1['regionId'].value_counts()

# All the unique class
df1['regionId'].unique

# Number of unique classes
df1['regionId'].nunique()

# Sklearn One HotEncoder- This function One-Hot Encoding on categorical numbers
from sklearn.preprocessing import OneHotEncoder
hotencoder = OneHotEncoder()
encoded = hotencoder.fit_transform(df1.regionId.values.reshape(-1,1)).toarray()

# Returns a numpy array of one hot encoded variables
encoded

# Convert the array into a dataframe. Specifically, ne hot encoded dataframe
df1_encoded = pd.DataFrame(encoded, columns = ["regionId_"+str(int(i)) for i in range(encoded.shape[1])])
df1_encoded.head()
df1_encoded.shape

# concats two dataFrames
df1_dummies = pd.concat([df1, df1_encoded], axis=1)
df1_dummies.head(5)
```

#Display the first ten rows of the data
df.head(10)

Observations and analysis:
There are male and female
There smokers and non-smokers
There are persons with and without children

age	sex	bmi	children	smoker	charges	regionId
19	1	27.9	0	1	16884.92	3
18	0	33.77	1	0	1725.552	2
28	0	33	3	0	4449.462	2
33	0	22.705	0	0	21984.47	1
32	0	28.88	0	0	3866.855	1
...	...	...	...	...	...	...
50	0	30.97	3	0	10600.55	1
18	1	31.92	0	0	2205.981	0
18	1	36.85	0	0	1629.834	2
21	1	25.8	0	0	2007.945	3
61	1	29.07	0	1	29141.36	1

# Check for missing values (NaN) and counting the number of NaN
df.isna().any()
df.isna().sum()

Observation and analysis
There are no NaN values

age	False
sex	False
bmi	False
children	False
smoker	False
charges	False
regionId	False
dtype:	
bool	

printfunc('Observation - There are 1338 rows and 7 columns', color="Black")
printfunc('\nObservation - There are no missing values in dataset', color="Black")
printfunc('\nObservation - There are male and female smokers', color="Black")

#Display last ten rows of the data
df.tail(10)

Observation and analysis

Display last ten rows of the dataset

age	sex	bmi	children	smoker	charges	regionId
23	1	24.225	2	0	22395.74	0
52	0	38.6	2	0	10325.21	3
57	1	25.74	2	0	12629.17	2
23	1	33.4	0	0	10795.94	3
52	1	44.7	3	0	11411.69	3
50	0	30.97	3	0	10600.55	1
18	1	31.92	0	0	2205.981	0
18	1	36.85	0	0	1629.834	2
21	1	25.8	0	0	2007.945	3
61	1	29.07	0	1	29141.36	1

# Display information regarding the data
df.info()

Observations and analysis

Of the 1338 rows there are no NaN (null) values
There are three data types uint8 and int64

```
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   int64
 2   bmi       1338 non-null   float64
 3   children  1338 non-null   int64
 4   smoker    1338 non-null   int64
 5   charges   1338 non-null   float64
 6   regionId  1338 non-null   int32
dtypes: float64(2), int32(1), int64(4)
```

# Generate description of the data in all columns and then transposing the columns
df.describe()
df.describe().T

Observations and analysis

```
AGE:      Mean=39.20    Min=18.00   Max=64.00
BMI:      Mean=30.66    Min=15.96   Max=53.13
CHILDREN: Mean=01.09    Min=00.00   Max=05.00
CHARGES:  Mean=13270.00 Min=1121.87 Max=63700.00
```

	count	mean	std	min	25%	50%	75%	max
age	1338	39.20703	14.04996	18	27	39	51	64
sex	1338	0.49477	0.50016	0	0	0	1	1
bmi	1338	30.6634	6.09819	15.96	26.29625	30.4	34.69375	53.13
children	1338	1.09492	1.20549	0	0	1	2	5
smoker	1338	0.20478	0.40369	0	0	0	0	1
charges	1338	9.09883	0.91938	7.02365	8.46406	9.14666	9.71962	11.06306
regionId	1338	1.5157	1.10488	0	1	2	2	3

Generate description of the dataset
Df.shape

Observations and analysis

(1338, 7)

# Count the datatypes
df.dtypes.value_counts()

Observations and analysis:

```
int64      4
float64    2
int32      1
dtype: int64
```

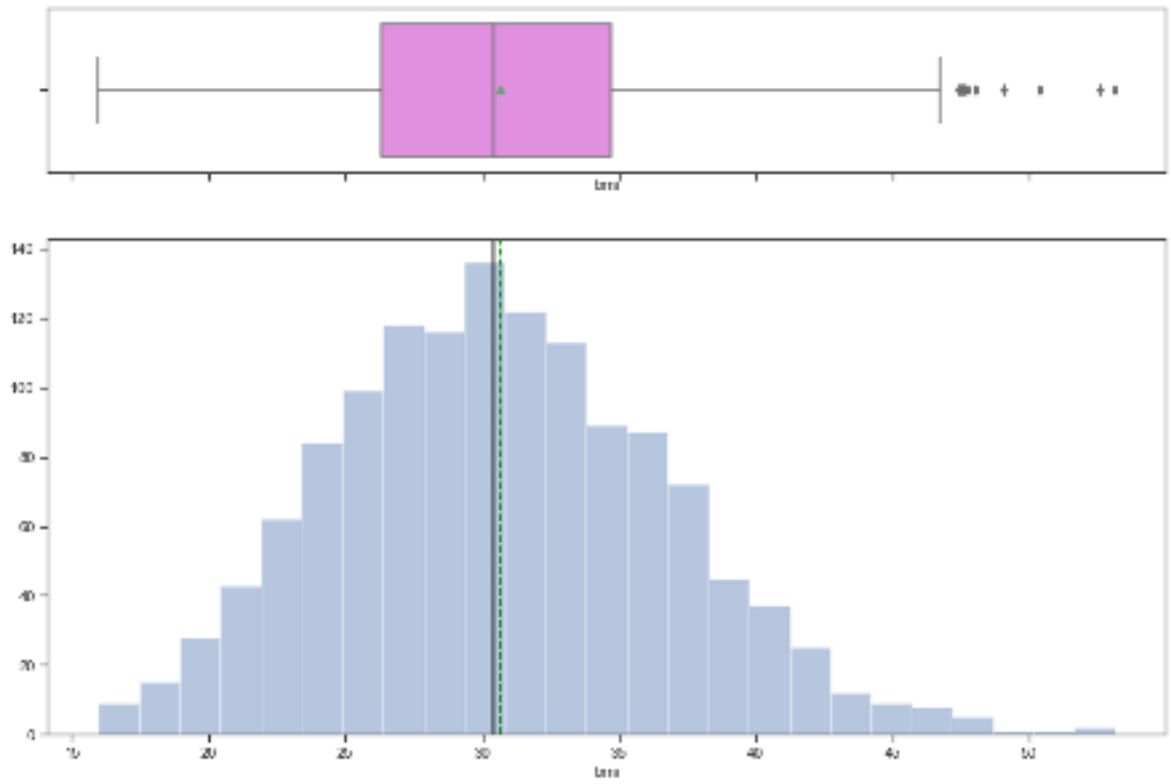
```
print('AGE:      Mean=39.20    Min=18.00   Max=64.00')
print('BMI:      Mean=30.66    Min=15.96   Max=53.13')
print('CHILDREN: Mean=01.09    Min=00.00   Max=05.00')
print('CHARGES:  Mean=13270.00 Min=1121.87 Max=63700.00')
```

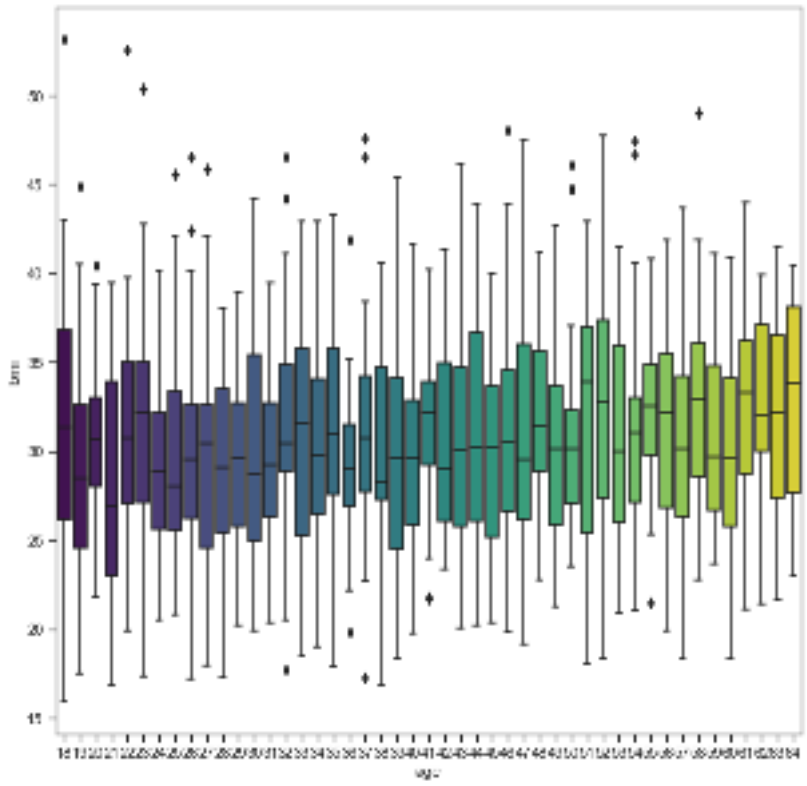
```
# To study central tendency and dispersion I write a function that will help us create boxplot and histogram for any
# input numerical variable. The function takes the numerical column as the input and returns the boxplots and histograms
# for the variable.

def histogram_boxplot(feature, figsize=(15,10), bins = None):
    """ Boxplot and histogram combined
    feature: 1-d feature array
    figsize: size of fig (default (9,8))
    bins: number of bins (default None / auto)
    """
    f2, (ax_box2, ax_hist2) = plt.subplots(nrows = 2, # Number of rows of the subplot grid= 2
                                           sharex = True, # x-axis will be shared among all subplots
                                           gridspec_kw = {"height_ratios": (.25, .75)},
                                           figsize = figsize
                                           ) # creating the 2 subplots
    sns.boxplot(feature, ax=ax_box2, showmeans=True, color='violet') # boxplot will be created and a triangle will indicate the mean value of the column
    sns.distplot(feature, kde=F, ax=ax_hist2, bins=bins,palette="winter") if bins else sns.distplot(feature, kde=False, ax=ax_hist2) # For histogram
    ax_hist2.axvline(np.mean(feature), color='green', linestyle='--') # Add mean to the histogram
    ax_hist2.axvline(np.median(feature), color='black', linestyle='-') # Add median to the histogram

histogram_boxplot(df1["bmi"])
plt.figure(figsize=(10, 10))
sns.boxplot(x = "age", y = "bmi", data = df1, palette = "viridis")
plt.show()

print('Histogram reveals BMI has a Mean: 30.6')
print('Histogram reveals BMI has a Median: 30.4')
print('Boxplot reveals BMI ages 26 have upper outlers')
print('Boxplot reveals BMI ages 32 have upper and lower outliers')
print('Boxplot reveals BMI ages 36 have upper and lower outliers')
print('Boxplot reveals BMI ages 37 and 41 has lower outliers')
print('Boxplot reveals BMI ages 50 has upper outliers')
print('Boxplot reveals BMI ages 55 has lower outliers')
```

#Create histogram boxplot of BMI

Observations and analysis

Histogram reveals BMI has a Mean: 30.6
Histogram reveals BMI has a Median: 30.4
Boxplot reveals BMI ages 26 have upper outlers
Boxplot reveals BMI ages 32 have upper and lower outliers
Boxplot reveals BMI ages 36 have upper and lower outliers
Boxplot reveals BMI ages 37 and 41 has lower outliers
Boxplot reveals BMI ages 50 has upper outliers
Boxplot reveals BMI ages 55 has lower outliers





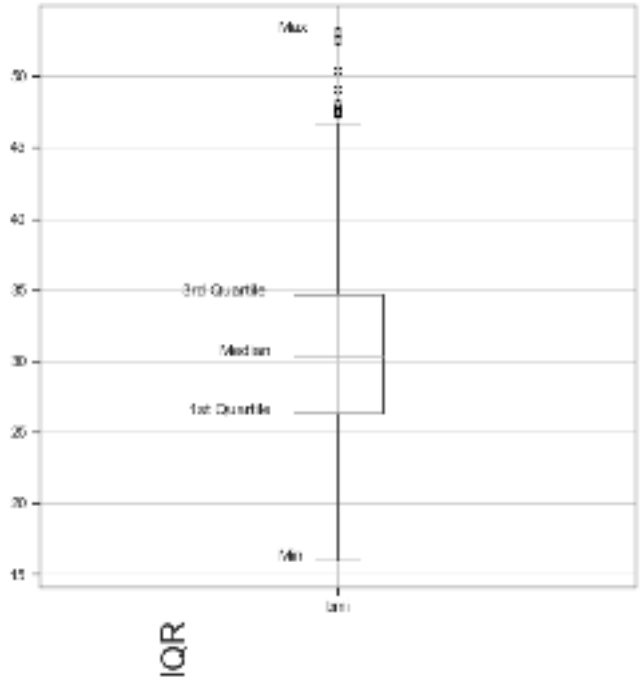
```
# Now let us have a closer look at the distribution of the column Age
mean = df1.mean()     # Arithmetic average of all observances
median = df1.median() # Middle valueS
mode = df1.mode()     # Maximum frequeny of occurence

print('Mean: ',mean,'\nMedian: ',median,'\nMode: ',mode)


print("Data_quantile(25%):",df1.quantile(q=0.25))  # Show values below for 25% of the data
print("Data_quantile(50%):",df1.quantile(q=0.50))  # Show values below for 50% of the data
print("Data_quantile(75%):",df1.quantile(q=0.75))  # Show values below for 75% of the data


# IQR Value for BMI
print(df1["bmi"].quantile(0.75) - df1["bmi"].quantile(0.25))

# Evaluate the Range
print('Difference between the highest value and lowest values for individual attributes','\nDiff:', df1.max() -
df1.min())

# Evaluate Variance
print(df1.var())
```

```
# Evaluate Standard Deviation
print(df1.std())

# Evaluate Convariance and Correlation
print('The convariance of each attribute against every other attribute')
df1.cov()

# Evaluate Correlation
print('The the correlation coefficient between every pair of attributes')
df1.corr()

print("Quantile(25%)for BMI is 26.29")
print("Quantile(50%)for BMI is 30.40")
print("Quantile(75%)for BMI is 34.69")
print("There are a few outlier above the 75% quantile")

# Plotting the summary mean,mode,median using histogram
mean=df1['bmi'].mean()
median=df1['bmi'].median()
mode=df1['bmi'].mode()

print('Mean: ',mean,'\nMedian: ',median,'\nMode: ',mode[0])

plt.figure(figsize=(20,10)) # makes the plot wider
plt.hist(df1['age'],bins=100,color='lightblue') #Plot the histogram

# Draw lines on the plot for mean median and the two modes we have in Age
plt.axvline(mean,color='green',label='Mean')
plt.axvline(median,color='blue',label='Median')
plt.axvline(mode[0],color='red',label='Mode')

plt.xlabel('bmi')         # label the x-axis
plt.ylabel('Frequency')   # label the y-axis
plt.legend()              # Plot the legend
plt.show()

# Distribution of AGE, Charges and BMI values.
f, axes = plt.subplots(1, 3, figsize=(15, 10))

age = sns.distplot(df1['age'], color='red', ax = axes[1], kde=True, hist_kws={"edgecolor":"k"})
age.set_xlabel("Age",fontsize=10)

charges = sns.distplot(df1['charges'], color="blue", ax = axes[2], kde=True, hist_kws={"edgecolor":"k"})
charges.set_xlabel("Charges",fontsize=10)

bmi = sns.distplot(df1['bmi'], color='olive', ax=axes[0], kde=True, hist_kws={"edgecolor":"k"})
bmi.set_xlabel("BMI",fontsize=10)
```

```
# Create boxplot for column="BMI Score"
df.boxplot(column="bmi",return_type='axes',figsize=(8,8))

# create text(x=0.74, y=22.25, s="3rd Quartile")like Median, 1st Quartile,Min,Max,IQR:
plt.text(x=0.74, y=34.69, s="3rd Quartile")
plt.text(x=0.8, y=30.40, s="Median")
plt.text(x=0.75, y=26.29, s="1st Quartile")
plt.text(x=0.9, y=15.96, s="Min")
plt.text(x=0.9, y=53.13, s="Max")
plt.text(x=0.7, y=8.39, s="IQR", rotation=90, size=25)
```

Observations and analysis:

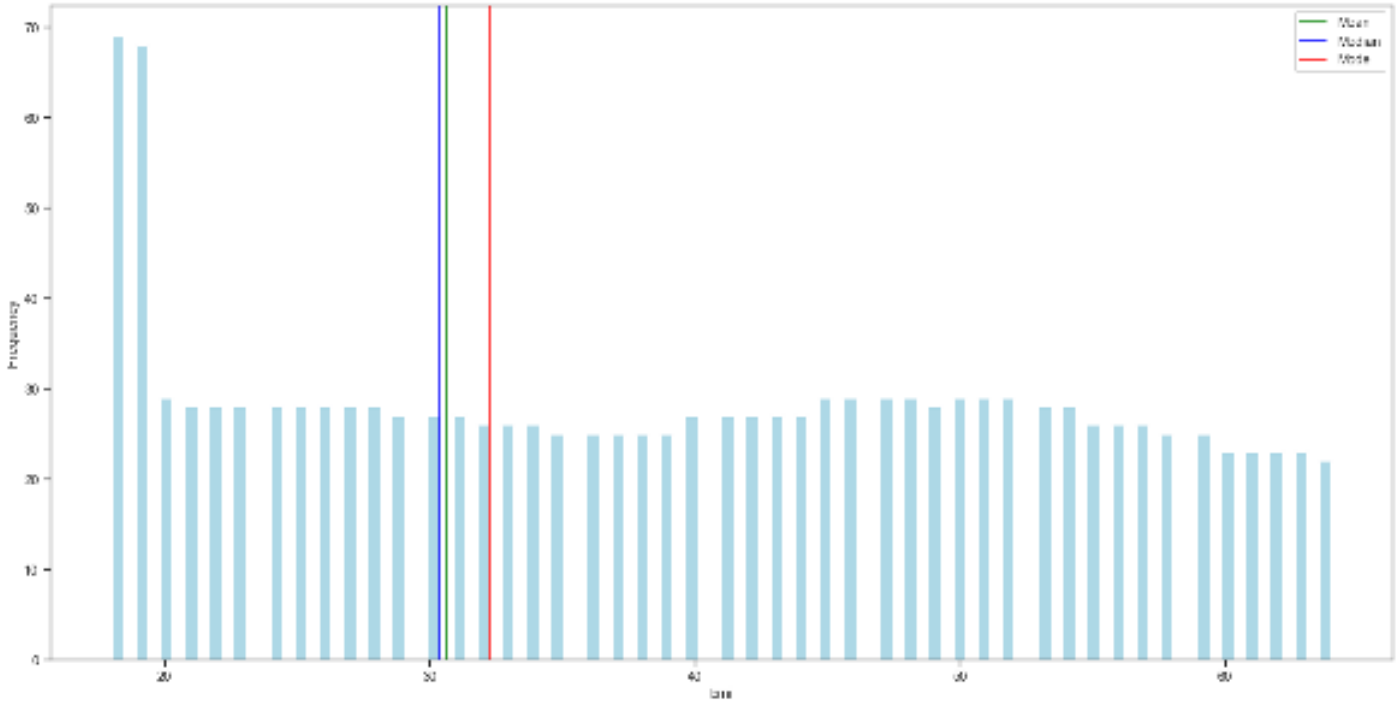
Quantile(25%)for BMI is 26.29
Quantile(50%)for BMI is 30.40
Quantile(75%)for BMI is 34.69
There are a few outlier above the 75% quantile



```
# Plotting the summary mean,mode,median using histogram
mean=df['bmi'].mean()
median=df['bmi'].median()
mode=df['bmi'].mode()

print('Mean: ',mean,'\nMedian: ',median,'\nMode: ',mode[0])

plt.figure(figsize=(20,10)) # makes the plot wider
plt.hist(df['age'],bins=100,color='lightblue') #Plot the histogram

# Draw lines on the plot for mean median and the two modes we have in Age
plt.axvline(mean,color='green',label='Mean')
plt.axvline(median,color='blue',label='Median')
plt.axvline(mode[0],color='red',label='Mode')

plt.xlabel('bmi')         # label the x-axis
plt.ylabel('Frequency')   # label the y-axis
plt.legend()              # Plot the legend
plt.show()
```

Observations and analysis:

Mean: 30.663396860986538
Median: 30.4
Mode: 32.3



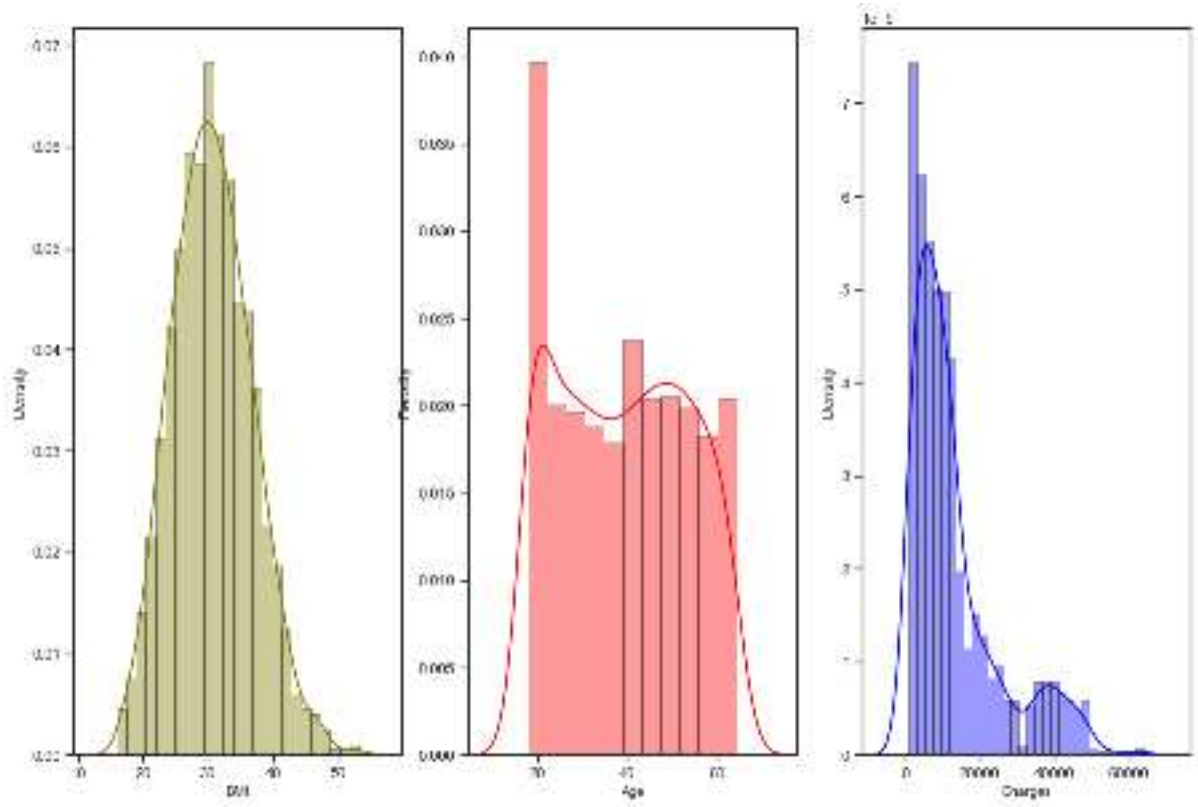
```
# Distribution of AGE, Charges and BMI values.
f, axes = plt.subplots(1, 3, figsize=(15, 10))

age = sns.distplot(df['age'], color='red', ax = axes[1], kde=True, hist_kws={"edgecolor":"k"})
age.set_xlabel("Age",fontsize=10)

charges = sns.distplot(df['charges'], color="blue", ax = axes[2], kde=True, hist_kws={"edgecolor":"k"})
charges.set_xlabel("Charges",fontsize=10)

bmi = sns.distplot(df['bmi'], color='olive', ax=axes[0], kde=True, hist_kws={"edgecolor":"k"})
bmi.set_xlabel("BMI",fontsize=10)
```

Observations and analysis:

Age Relative distribution

CHARGES Right skewed distribution

BMI Normal distribution



```
# Evaluate skewness of AGE, BMI and CHARGE
pd.DataFrame.from_dict(dict(
    {
     'age':df.age.skew(),
     'bmi': df.bmi.skew(),
     'charges': df.charges.skew()
    }), orient='index', columns=['Skewness'])
```

Observations and analysis:

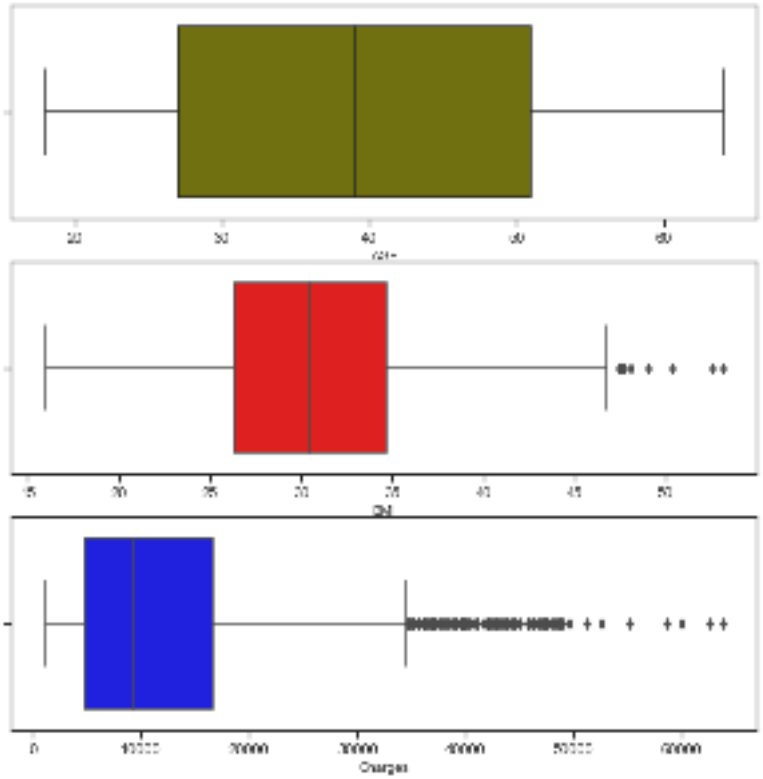
	Skewness
age	0.05567
bmi	0.28405
charges	1.51588

```
print('Age Relative distribution')
print('CHARGES Right skewed distribution')
print('BMI Normal distribution')
```

```
# Evaluation of outliers
f, axes = plt.subplots(3, 1, figsize=(10, 10))
bmi = sns.boxplot(df['age'], color='olive', ax=axes[0])
bmi.set_xlabel("AGE",fontsize=10)

age = sns.boxplot(df['bmi'], color='red', ax=axes[1])
age.set_xlabel("BMI",fontsize=10)

charges = sns.boxplot(df['charges'], color='blue', ax=axes[2])
charges.set_xlabel("Charges",fontsize=10)
```

Observations and analysis:

AGE outliers not present
BMI a few outliers
CHARGES several outliers

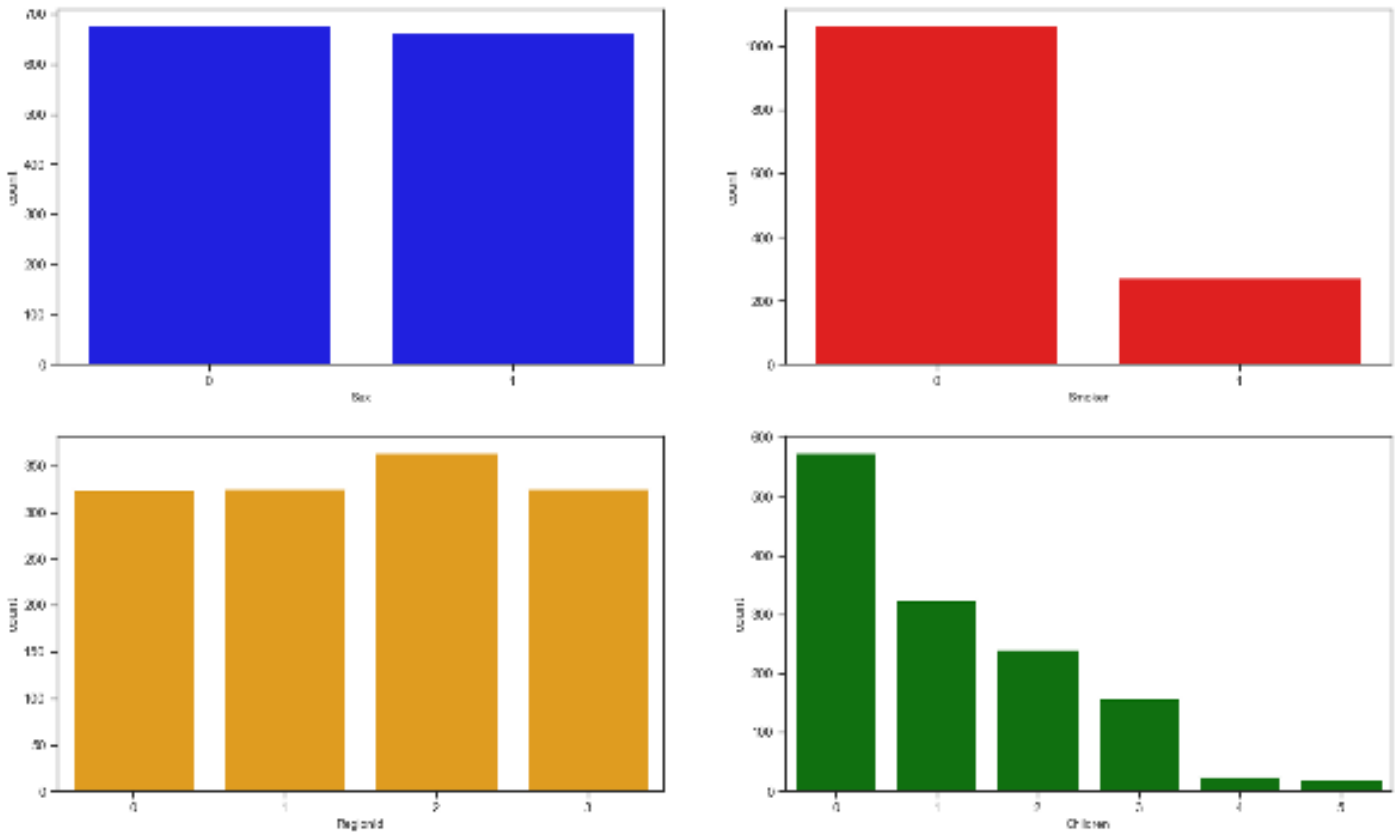


```
print('AGE outliers not present')
print('BMI a few outliers')
print('CHARGES several outliers')
```

```
#Evaluation of previous categorical columns SEX, SMOKER and REGION
f, axes = plt.subplots(2, 2, figsize=(20, 12))
sex = sns.countplot(df['sex'], color='blue', ax=axes[0,0])
sex.set_xlabel("Sex",fontsize=10)

smoker = sns.countplot(df['smoker'], color='red', ax = axes[0,1])
smoker.set_xlabel("Smoker",fontsize=10)

region = sns.countplot(df['regionId'], color='orange', ax = axes[1,0])
region.set_xlabel("RegionId",fontsize=10)

children = sns.countplot(df['children'], color='green', ax = axes[1,1])
children.set_xlabel("Children",fontsize=10)
```

Observations and analysis:

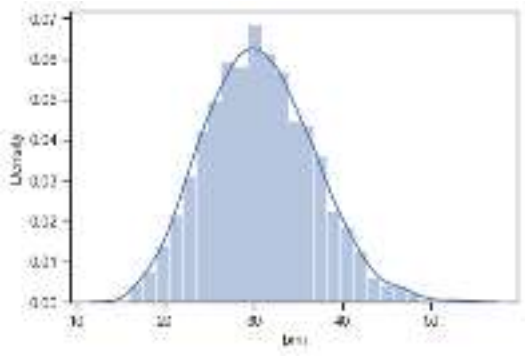
Sex: is relative equal between males(0) and female(1)
Smokers: there are far more non-smokers(0) than smokers(1)
Insurance: in the SW, NW and NE are basically equal. There are a few more insurered in the SE
Children: Many insured have between 1-3 children, majority have no children, minority have 4 or 5 children.



```
print('Sex: is relative equal between males(0) and female(1)')
print('Smokers: there are far more non-smokers(0) than smokers(1)')
print('Insurance: in the SW, NW and NE are basically equal. There are a few more insurered in the SE')
print('Children: Many insured have between 1-3 children, majority have no children, minority have 4 or 5 children.')

# Plot multiple pairwise scatterplots in the dataset, using the pairplot() function. This creates a matrix of axes
# and shows the relationship for each pair of columns in a DataFrame, it also draws the histogram of each # #
# variable on  the diagonal Axes:
sns.pairplot(df1, kind='reg')

# For displot note the following:
# If True, then a histogram is computed where each bin gives the counts in that bin plus all bins for smaller
# values.  # The last bin gives the total number of datapoints.
# If density is also True then the histogram is normalized such that the last bin equals 1.
# If cumulative is a number less than 0 (e.g., -1), the direction of accumulation is #reversed.
# In this case, if density is also True, then the histogram is normalized such that the first bin equals 1.

# Displot
sns.distplot(df.bmi)
ptl.show()
```

 # Observations and analysis

Quantile(25%)for BMI is 26.29
Quantile(50%)for BMI is 30.40
Quantile(75%)for BMI is 34.69
There are a few outlier above the 75% quantile

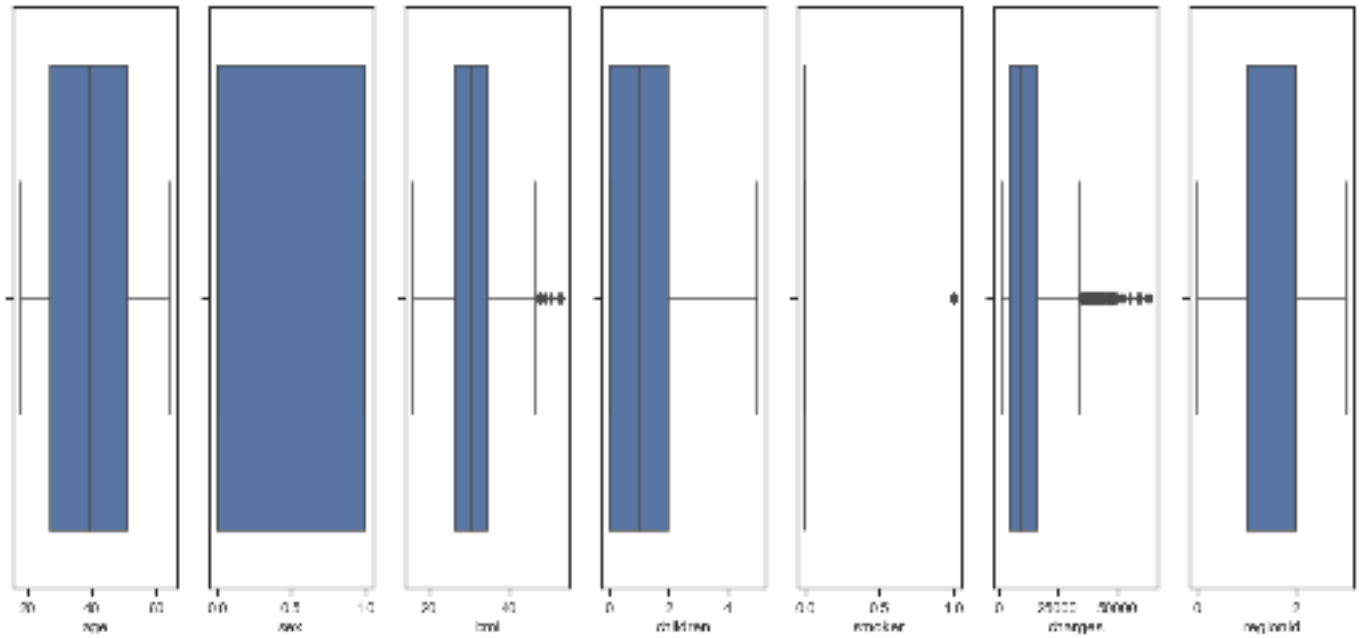


```
print("Quantile(25%)for BMI is 26.29")
print("Quantile(50%)for BMI is 30.40")
print("Quantile(75%)for BMI is 34.69")
print("There are a few outlier above the 75% quantile")
```

```
# Boxplot show distributions with respect to categories. A box plot (or box-and-whisker plot) shows the # #
# distribution  of quantitative data in a way that facilitates comparisons between variables or across levels of
# a categorical  variable. The box shows the quartiles of the dataset while the whiskers extend to show the #
# rest of the distribution,  except for points that are determined to be “outliers” using a method that is a # # #
# function of the inter-quartile range.

features_to_analyse = df.columns[:7]
current_palette = sns.color_palette("Blues")

fig, axes = plt.subplots(round(len(features_to_analyse)/7), 7, figsize = (18, 8))

for i, ax in enumerate(fig.axes):
    if i < len(features_to_analyse):
        sns.boxplot(x=features_to_analyse[i], data = df, ax = ax, orient = 'v')
```

 # Observations and analysis

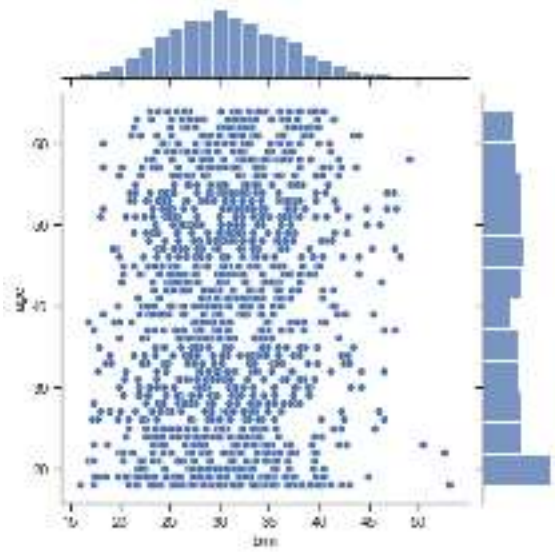
AGE: very few below age 20 and over 60
BMI: outliers start above 53%
CHILDREN: majority of person have 2 to 4 chidren with very few exceptions above 4
CHARGES: outliers start around the 30000 dollar mark



```
print('AGE: very few below age 20 and over 60')
print('BMI: outliers start above 53%')
print('CHILDREN: majority of person have 2 to 4 chidren with very few exceptions above 4')
print('CHARGES: outliers start around the 30000 dollar mark')
```

# Using bivariate distribution draw a histogram. Draws a plot of two variables with bivariate and univariate graphs.
sns.jointplot(df['bmi'], df['age']);
plt.show()

 # Observations and analysis
There are significant BMI outliers for all ages particularily person under age 25



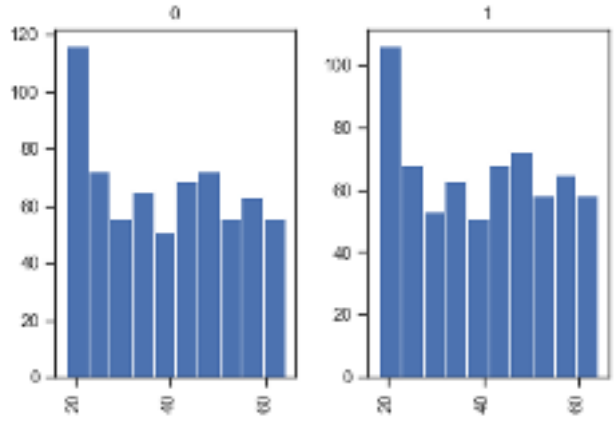
print('There are significant BMI outliers for all ages particularily person under age 25')

Crosstab compute a simple cross tabulation of two (or more) factors. By default computes a # frequency table of the factors unless an array of values and an aggregation function are # # passed.

# Male = 0 Female = 1
pd.crosstab(df1.age, df1.sex)

# The gender of person by age
df.hist(by='sex',column = 'age')

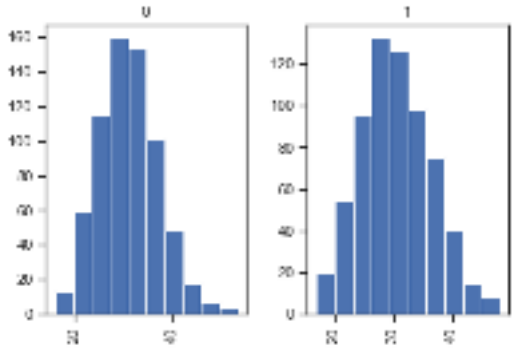
 # Observations and analysis
Majority of males are 20 years old the minority are 40 year old
Majority of females are 20 years old the minority are 40 year old



print('Majority of males are 20 years old the minority are 40 year old')
print('Majority of females are 20 years old the minority are 40 year old')

# The gender of person by bmi
df.hist(by='sex',column = 'bmi')

 # Observations and analysis
Males ages 25-30 have higher BMI
Females ages 25-30 have higher BMI



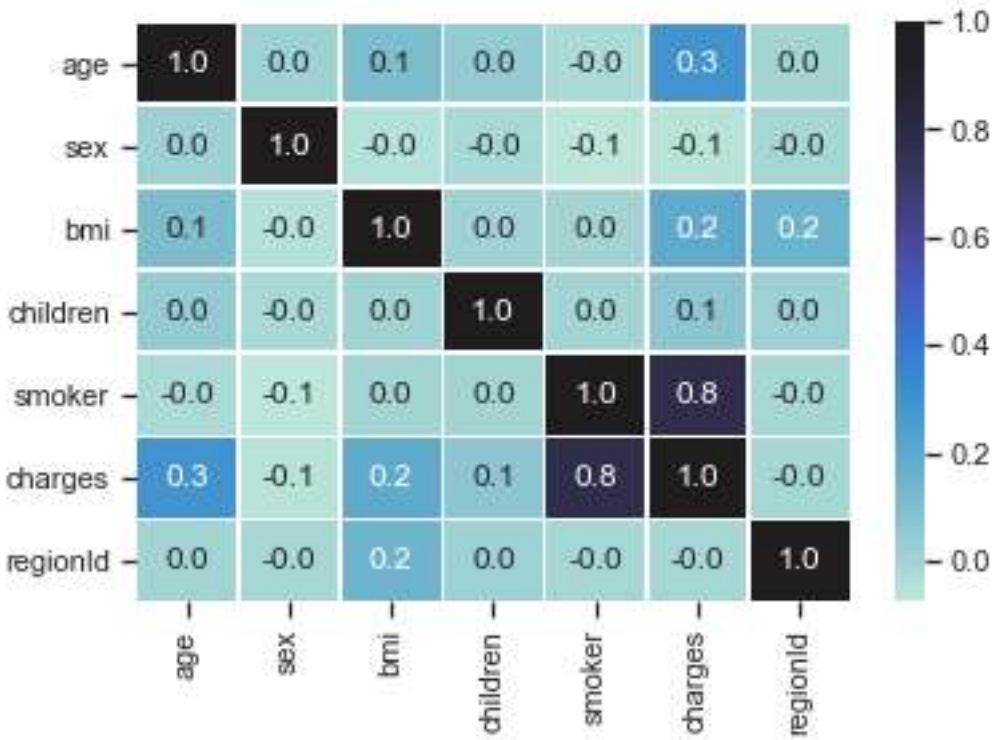
print('Males ages 25-30 have higher BMI')
print('Females ages 25-30 have higher BMI')

# The gender of person by smoking status
df1.hist(by='sex',column = 'smoker')

```
# Correlation of the different variables
# Correlation among pairs of continous variables using two heatmap constructs
sns.heatmap(df.corr(), annot=True, linewidths=.5, fmt= '.1f', center = 1 )  # heatmap
```

Observations and analysis

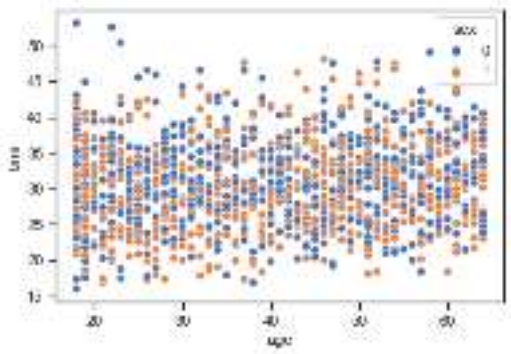
There is a positive 0.0 correlation between Age and Sex
There is a positive 0.3 correlation between Age and BMI
There is a positive 0.0 correlation between Age and Children
As expected there is a 0.8 significant correlation between Charges and Smoker




```
print('There is a positive 0.0 correlation between Age and Sex')
print('There is a positive 0.3 correlation between Age and BMI')
print('There is a positive 0.0 correlation between Age and Children')
print('As expected there is a 0.8 significant correlation between Charges and Smokers')
```

# A scatter plot is a diagram where each value in the data set is represented by a dot.
sns.scatterplot(x='age', y='bmi',data=df, hue = 'sex')

Observations and analysis:
The BMI for both sex are relatively close at the 35% range
The BMI for males for all age is higher than females at and above 40% except for ages 35 and 45

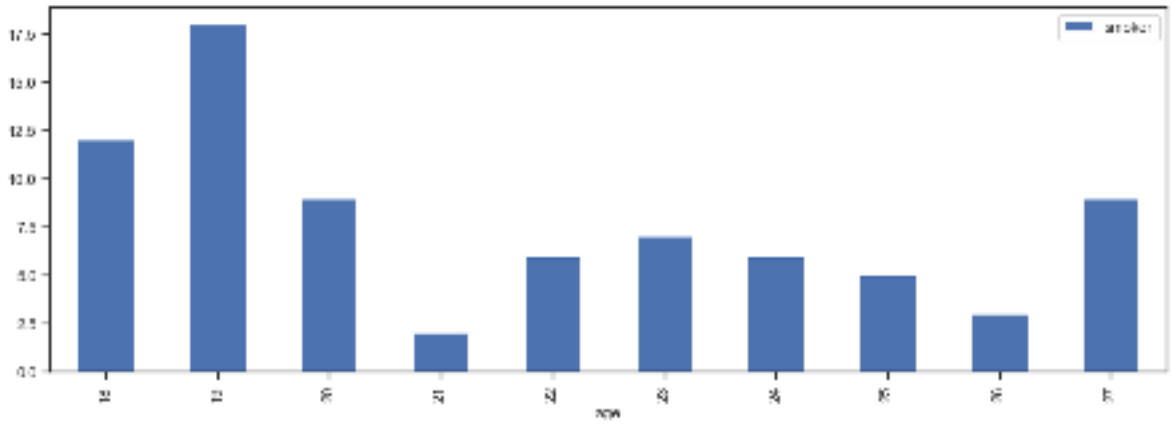


print('The BMI for both sex are relatively close at the 35% range')
print('The BMI for males for all age is higher than females at and above 40% except for ages 35 and 45')

Reset the index of the dataframe, sort data by Age, groupby Age and Sex, using first 10 # # # rows of data generate a bar graph

# Bar plot to check Number of Age by Children
df.groupby(by=['age'])['smoker'].sum().reset_index().sort_values(['age']).head(10).plot(x='age', y='smoker',kind='bar', figsize=(15,5))

Observations and analysis:
Majority of smokers are age 19
Minority of smokers are age 21

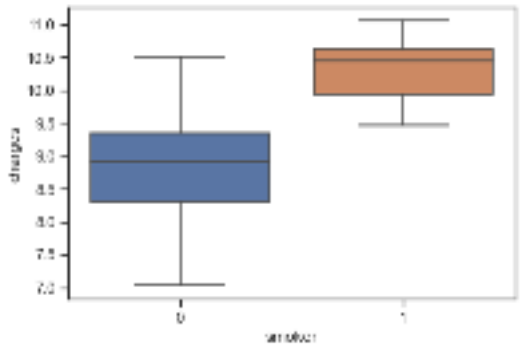


. HYPOTHESIS ANALYSIS CHARTS

```
# Prove (or disprove) that the medical claims made by the people who smoke is greater thatn those who
don't?
# For discussion see PDF PANDAS Intro to Data Structures
pd.DataFrame.from_dict(dict(
   {
    'smoker':df[df.smoker == '1'].charges.skew(),      # smokers -  charges
    'nonsmoker':df[df.smoker == '0'].charges.skew(),   # non-smokers - charges
   }), orient='index', columns=['Skewness'])

df.charges  = np.log1p(df.charges) # log1p used when x is small, e.g., when 1+x=1 in floating point accuracy

pd.DataFrame.from_dict(dict(
   {
    'smoker':df[df.smoker == '1'].charges.skew(),     # smokers - charges
    'nonsmoker':df[df.smoker == '0'].charges.skew(),  # non-smokers - charges
   }), orient='index', columns=['Skewness'])

sns.boxplot('smoker', 'charges', data=df)
```

Observations and analysis:

Medical charges for males is significantly lower than females



Evaluate the NULL and Alternate Hypothesis For BMI as it relates to Independence of the observations
Independence of the observations - The sample is a random sample, i.e. the observations are collected # # # # independently of each other. We cannot really test this assumption here. We will assume that this # # # # # # assumption holds for our experiment.

#Normality - All populations under consideratio nhave normal distribution

```
# normality test
stat, p = shapiro(df1["bmi"])

print('Statistics = %.2f, p = %.2f' % (stat*100, p*100))

if p < 0.05:
   print('Sample looks Gaussian fail to reject H0')
else:
   print('Sample does not look Gaussian reject H0')
```

Observations and analysis:
Statistics = 99.39, p = 0.00
Sample looks Gaussian fail to reject H0

Evaluate the NULL and Alternate Hypothesis For Smokers Using NonParametric
Null hypothesis
H0 : Charges of smokers are same as Charges of Non-smokers

Alternate hypothesis
H1 : Charges of non-smokers are significantly different from Charges of smokers

```
#Separate the charges column
x = np.array(df[df.smoker == '1'].charges) #Smokers
y = np.array(df[df.smoker == '0'].charges) #Non-Smokers

# t-test
t, p_value = stat.mannwhitneyu(x,y,alternative=None)
```

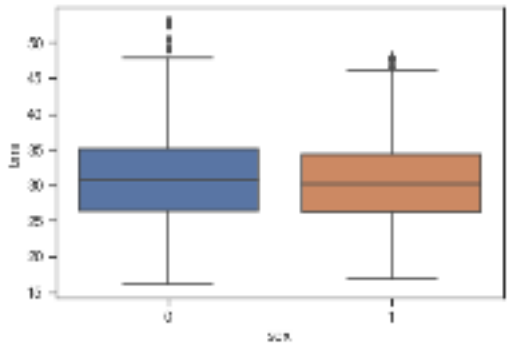
Observations and analysis:
p_value < 0.05, we reject the Null Hypothesis. Thus charges of smokers differ significantly from non-smokers

Evaluate the NULL and Alternate BMI of Genders Using NonParametric

```
# Prove (or disprove) with statistical evidence that the BMI of females is different from that of males
sns.boxplot('sex', 'bmi', data=df)

# Null hypothesis
# H0 : BMI of Males are similar to that of Females

# Alternate hypothesis
# H1 : BMI of Males are significantly different from that of Females

# Separate the bmi column

x = np.array(df[df.sex == '0'].bmi) #Males
y = np.array(df[df.sex == '1'].bmi) #Females

#  t-test
t, p_value = stat.mannwhitneyu(x,y, alternative=None)
```

Observations and analysis:

p_value < 0.05, we reject the Null Hypothesis. Thus the BMI of females are significantly different from males

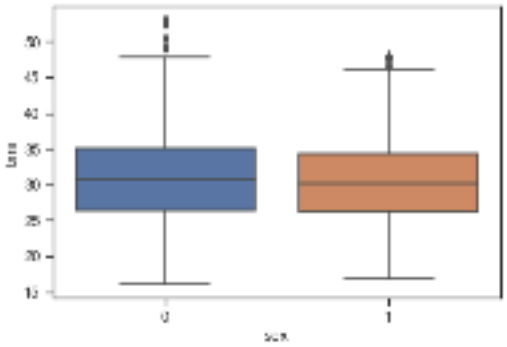


Evaluate the NULL and Alternate BMI of Genders Using Parametric

```
# Prove (or disprove) with statistical evidence that the BMI of females is different from that of males
sns.boxplot('sex', 'bmi', data=df)

# Null hypothesis
# H0 : BMI of Males are similar to that of Females

# Alternate hypothesis
# H1 : BMI of Males are significantly different from that of Females

# Separate the bmi column

x = np.array(df[df.sex == '0'].bmi) #Males
y = np.array(df[df.sex == '1'].bmi) #Females

#  t-test
t_stat, p_value = stat.ttest_ind(x,y)
```

Observations and analysis:

p_value > 0.05, we fail to reject Null Hypothesis. Thus the BMI of females are similar to males

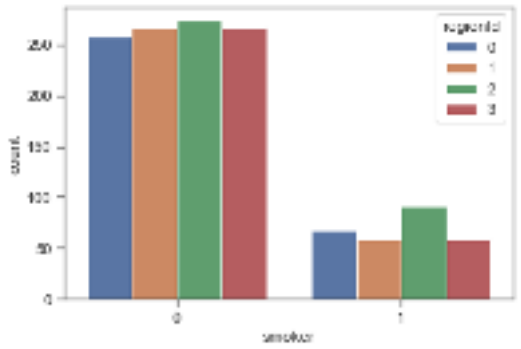


Evaluate the NULL and Alternate of Smokers Across Regions Pre-Verification

```
# Is the proportion of smokers significally different across different regions?
sns.countplot(df['smoker'],hue=df['regionId']);

# Chi-Square test of indepedence

contigency = pd.crosstab(df['smoker'],df['regionId'])
contigency

# chi2: The test statistic
# p: The p-value of the test
# dof: Degree of freedome
# expected The expected frequencies, based on the marginal sums of the table

c, p, dof, expected =  stat.chi2_contingency(contigency)

# Print the p-value
print(p)
```

Observations and analysis:

```
0.06171954839170547
```

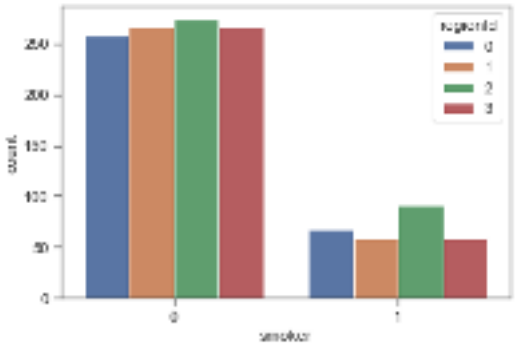


Evaluate the NULL and Alternate of Smokers Across Different Regions

```
# Is the proportion of smokers significally different across different regions?
sns.countplot(df['smoker'],hue=df['regionId']);

# State null hypothesis
# H0 : Proportion of Smokers are similar in Regions

# State alternate hypothesis
# H1 : Proportion of Smokers are significantly different Regions

# Chi-Square test for testing

contigency = pd.crosstab(df['smoker'],df['regionId'])

# chi2: The test statistic
# p_value: The p-value of the test
# dof: Degree of freedome
# expected The expected frequencies, based on the marginal sums of the table

c, p_value, dof, expected  =  stat.chi2_contingency(contigency)
```

Observations and analysis:

p_value > 0.05, we fail to reject Null Hypothesis. Thus proporations are similiar across regions



Evaluate the NULL and Alternate of BMI of Women With and Without Chidren

#Is the distribution of bmi across women with no children, one child and two children, the same ?
sns.boxplot('children', 'bmi', data=df[(df.children>0) & (df.children<4)])

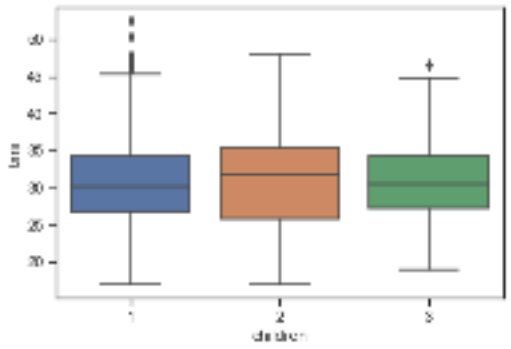
State null hypothesis
H0 : BMI is uniform across women with different number of children

#State alternate hypothesis
H1 : BMI is different across women with different number of children

# Get the female data
females = df[df['sex'] == '1']

# Get the bmi samples based on number of children
no_child_bmi = females[females['children'] == 0].bmi
one_child_bmi = females[females['children'] == 1].bmi
two_child_bmi = females[females['children'] == 2].bmi

Observations and analysis:
BMI of women with 1 and 3 chidren are very similiar both have outliers
BMI of women with 2 children has no outliers and is higher than women with 1 and 3 children



#Since there are multiple samples and we need to check the variances of multiple samples, choosing
ANOVA testing for this

f, p_value = stat.f_oneway(no_child_bmi,one_child_bmi ,two_child_bmi)

Observations and analysis:
As the p_value > 0.05, we fail to reject Null Hypothesis. Thus BMI is uniform across women with different number of children