Project 1 Statistics and Probability

In [ ]:

```
import pandas as pd
import numpy as np
import scipy.stats as stats
```

In [5]:

```
df = pd.read_csv('insurance .csv')
```

In [63]:

```
df.head()
```

Out[63]:

	age	sex	bmi	children	smoker	region	charges
0	19	0	27.900	0	1	0	16884.92400
1	18	1	33.770	1	0	1	1725.55230
2	28	1	33.000	3	0	1	4449.46200
3	33	1	22.705	0	0	1	21984.47061
4	32	1	28.880	0	0	1	3866.85520

Q2. Estimate the minimum sample size n to get the 99% accurate predictions.(precision = 0.02)

In [ ]:

```
## Q2
Given information:
= 99% or it is also called as confidence level
precision =E= or Margin of error
p= 0.5
When estimate about the population proportion is not known then we assumed it␣
↪is unbased means p=0.5
We need to find the sample size n
The sample size can be determine as
n= z_alpha/2^2*p(1-p)/ E^2
C= 99%
alpha= 1- C= 0.01
```

In [12]:

```
from scipy.stats import norm
```

In [13]:

```
critical=norm.ppf(0.995)
critical
```

Out[13]:

```
2.5758293035489004
```

In [14]:
```
n=(2.58**2*0.25)/(0.02**2)
```

In [15]:
```
n
```

Out[15]:
```
4160.25
```

In [19]:
```
n= 4161
#The_required_sample_size is 4161
```

In [20]:
```
 import pandas as pd
```

In [21]:
```
df = pd.read_csv('insurance .csv')
```

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

Out[22]:

	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
...	...	...	...	...	...	...	...
1333	50	male	30.970	3	no	northwest	10600.54830
1334	18	female	31.920	0	no	northeast	2205.98080
1335	18	female	36.850	0	no	southeast	1629.83350
1336	21	female	25.800	0	no	southwest	2007.94500
1337	61	female	29.070	0	yes	northwest	29141.36030

1338 rows × 7 columns

3. Check the data is cleaned or not. If not then clean it (Null values,Row/Column Duplicates, Outliers, Change the string into numbers)

In [23]:

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

Out[23]:

```
age         0
sex         0
bmi         0
children    0
smoker      0
region      0
charges     0
dtype: int64
```

In [25]:

```
from sklearn.preprocessing import LabelEncoder
model=LabelEncoder()
df['sex']=model.fit_transform(df['sex'])
df['region']=model.fit_transform(df['sex'])
df['smoker']=model.fit_transform(df['smoker'])
```

In [26]:

```
df.head()
```

Out[26]:

	age	sex	bmi	children	smoker	region	charges
0	19	0	27.900	0	1	0	16884.92400
1	18	1	33.770	1	0	1	1725.55230
2	28	1	33.000	3	0	1	4449.46200
3	33	1	22.705	0	0	1	21984.47061
4	32	1	28.880	0	0	1	3866.85520

In [27]:

```
df.drop_duplicates(inplace=True)
```

In [28]:

```
df.shape
```

Out[28]:

```
(1337, 7)
```

4. Check that sex and smoking are statistically independent or not.

In [29]:

```
obs_table=pd.crosstab(df['sex'],df['smoker'])
print("Observed frequencies")
print(obs_table)
```

```
Observed frequencies
smoker    0    1
sex
0       547  115
1       516  159
```

In [30]:

```
obs=obs_table.values
print(obs)
```

```
[[547 115]
 [516 159]]
```

In [32]:

```
exp_t=stats.chi2_contingency(obs_table)
print(exp_t)
```

```
(7.469139330086637, 0.0062765550120107375, 1, array([[526.33208676, 135.66791
324],
       [536.66791324, 138.33208676]]))
```

In [34]:

```
dof=exp_t[2]
exp=exp_t[3]
print("Expected frequencies")
print(exp)
```

```
Expected frequencies
[[526.33208676 135.66791324]
 [536.66791324 138.33208676]]
```

In [35]:

```
chi_squared_stat = (((obs-exp)**2)/exp).sum().sum()
print("chi square statistic")
print(chi_squared_stat)
```

```
chi square statistic
7.844077785733106
```

In [36]:

```
crit=stats.chi2.ppf(q = 0.95,df = dof)
print("critical value for respective confidence interval and degree of freedom
print(crit)
```

```
critical value for respective confidence interval and degree of freedom
3.841458820694124
```

In [37]:

```
p_value = 1 - stats.chi2.cdf(x=chi_squared_stat,
df=dof)
print("P value")
print(p_value)
```

```
P value
0.005098746217145678
```

In [41]:

```
if p_value < 0.05:
    print(" we are rejecting null hypothesis")
else:
    print("we are accepting null hypothesis")
```

```
 we are rejecting null hypothesis
```

4 Sex and smoking are not independent

5. Check that all regressor variables (independent variable) are independentof each other or not.

In [42]:

```
X=df.drop('charges',axis=1)
cnames=list(X.columns)
```

In [44]:

```
cnames
```

Out[44]:

```
['age', 'sex', 'bmi', 'children', 'smoker', 'region']
```

In [45]:

```
for i in cnames:
    for j in cnames:
        if(i!=j):
            obs_table=pd.crosstab(df[i],df[j])
            obs=obs_table.values
            exp_t=stats.chi2_contingency(obs_table)
            dof=exp_t[2]
            exp=exp_t[3]
            chi_squared_stat = (((obs-exp)**2)/exp).sum().sum()
            crit=stats.chi2.ppf(q = 0.95,df = dof)
            p_value = 1 - stats.chi2.cdf(x=chi_squared_stat,df=dof)
            if p_value < 0.05:
                print(f" {i} and {j} are not independent")
            else:
                print(f" {i} and {j} are independent")
```

```
 age and sex are independent
 age and bmi are independent
 age and children are not independent
 age and smoker are independent
 age and region are independent
 sex and age are independent
 sex and bmi are independent
 sex and children are independent
 sex and smoker are not independent
 sex and region are not independent
 bmi and age are independent
 bmi and sex are independent
 bmi and children are independent
 bmi and smoker are independent
 bmi and region are independent
 children and age are not independent
 children and sex are independent
 children and bmi are independent
 children and smoker are independent
 children and region are independent
 smoker and age are independent
 smoker and sex are not independent
 smoker and bmi are independent
 smoker and children are independent
 smoker and region are not independent
 region and age are independent
 region and sex are not independent
 region and bmi are independent
 region and children are independent
 region and smoker are not independent
```

In [ ]:

```

#Not all the regressor variables are independent of each other, only bmi is in
```

## 6. Check the dependency between response and regressors.

In [47]:
```
j="charges"
for i in cnames:
    obs_table=pd.crosstab(df[i],df[j])
    obs=obs_table.values
    exp_t=stats.chi2_contingency(obs_table)
    dof=exp_t[2]
    exp=exp_t[3]
    chi_squared_stat = (((obs-exp)**2)/exp).sum().sum()
    crit=stats.chi2.ppf(q = 0.95,df = dof)
    p_value = 1 - stats.chi2.cdf(x=chi_squared_stat,df=dof)
    if p_value < 0.05:
        print(f" {i} and {j} are not independent")
    else:
        print(f" {i} and {j} are independent")
```

```
 age and charges are independent
 sex and charges are independent
 bmi and charges are independent
 children and charges are independent
 smoker and charges are independent
 region and charges are independent
```

```
#response and regressors are independent of each other
```

7.Predict the regression Line to predict the charges for insurance using inde pendent variables.

In [48]:
```
X=df.drop('charges',axis=1)
y=df['charges']
```

In [49]:
```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=
```

In [50]:
```
from sklearn.linear_model import LinearRegression
model=LinearRegression()
model.fit(X_train,y_train)
pred=model.predict(X_test)
```

```
In [51]: model.intercept_
```

```
Out[51]: -12004.987401392225
```

```
In [52]: model.coef_
```

```
Out[52]: array([  261.02035669,  -113.22506984,   316.78476878,   464.61064522,
                24239.42001414,  -113.22506984])
```

8. Predict the accuracy of the regression Model.

```
In [53]:  model.score(X_test,y_test)
```

```
Out[53]: 0.7604691935331385
```

9. Predict insurance charge for Age = 29, Sex = F, bmi = 28, children = 1,Smoke = Yes, region = southeast.

```
In [54]:  pred1=model.predict([[29,0,28,1,1,0]])
```

```
C:\Users\91830\anaconda3\lib\site-packages\sklearn\base.py:450: UserWarning:
X does not have valid feature names, but LinearRegression was fitted with fea
ture names
  warnings.warn(
```

```
In [55]: pred1
```

```
Out[55]: array([29138.60712805])
```

```
In [57]: #insurance chagers : 29138.60712805
```

10. Give the percentage of error in regression model

```
In [59]: from sklearn.metrics import mean_absolute_percentage_error
```

```
In [60]: error=mean_absolute_percentage_error(y_test,pred)
```

```
In [61]: error
```

```
Out[61]: 0.39702898287144645
```

11. Give the 95% confidence interval for average charge insurance

In [62]:

```
y_mean=np.mean(y)
print(y_mean)
```

```
13279.121486655948
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: