HEALTHCARE INSURANCE ANALYSIS

INTRODUCTION:

The Healthcare Insurance Company provide financial support to those who have paid insurance premiums to the company. They provide basic medical expenses like hospital charges, medicine as well as critical illnesses like Cancer, Heart diseases, kidney ailments etc.

The purpose of the project is to analyze the data to predict the patients' healthcare costs. The various factors that contribute to higher healthcare costs are determined and the correlation between age, gender and medical history that leads to higher healthcare costs are analyzed and appropriate predictions made using different data science techniques.

The datasets which are used for the analysis are:

- ✓ Hospitalisation details.csv
- ✓ Medical Examinations.csv
- ✓ Names.xlsx

PROJECT TASKS:

DATA SCIENCE:

1) *Collate the files so that all the information is in one place*

Google Colab is an open source Jupyter Notebook environment. The code is written and executed in Colab.

All the 3 datasets are uploaded in the Colab environment.

The 3 files are read in the notebook using pd.read_csv() command.

2) *Check for missing values in the dataset*

DATA EXPLORATION: checked for any missing values in all 3 datasets using the code df.info()

No missing values found in the datasets

3) *Find the percentage of rows that have trivial value (for example?), and delete such rows if they do not contain significant information*

DATA CLEANING: The 3 files are merged into a single file using inner join on the column "Customer ID"

Trivial values such as wild character '?' searched using str.contains('?') and percentage of such rows calculated. The percentage is 0.428, significantly low value. Hence the corresponding rows are dropped from the dataframe.

4) *Use the necessary transformation methods to deal with the nominal and ordinal categorical variables in the dataset*

DATA PREPROCESSING:

To further data analysis, the categorical variables have to be transformed into nominal or ordinal variables. This can be achieved using ordinal encoding and one hot encoding.

1. Converted hospital_tier and city_tier columns using onehot encoder
2. Assigned numerical values to heart issues, any transplants, cancer history, smoker columns
3. Dropped the old columns from the dataframe

5) *The dataset has State ID, which has around 16 states. R1011, R1012, and R1013 are worth investingating*

State IDs R1011, R1012, and R1013 are major contributors, Using one hot encoding dummy variables are created for these state IDs and the original column dropped from the dataset

6) *The variable NumberOfMajorSurgeriesalso appears to have string values. Apply a suitable method to clean up this variable*.

1. Converted string to numerical value 0 for Nomajorsurgeries by applying lambda function to the column NumberOfMajorSurgeries.

7) *Age appears to be a significant factor in this analysis. Calculate the patients' ages based on their dates of birth.*

The year, month and date columns are combined in a single "dob" column. Age is calculated from the (current date – dob) and new column 'Age' added to the dataset

8) The gender of the patient may be an important factor in determining the cost of hospitalization. The salutations in a beneficiary's name can be used to determine their gender. Make a new field for the beneficiary's gender.

The name of the customer containing 'Mr' is categorized as Male and 'Mrs' or 'Ms' is categorized as Female. A new column 'Gender' is added to the dataset

DATA VISUALIZATION:

9) You should also visualize the distribution of costs using a histogram, box and whisker plot, and swarm plot.

Imported the required libraries for data visualization, Matplotlib and seaborn. Histogram and Box plot and swarm plots generated for the 'charges' column in the dataset.

10) State how the distribution is different across gender and tiers of hospitals

Box plots are plotted between charges and hospital tiers and hue = 'Gender'.

Observations:

1. Tier – 1 hospital charges are the highest among all the hospitals ranging from 20k to 40k

2. Median value of charges is the lowest for tier -2 hospitals

3. In comparison with tier 1 and 2 patient count admitted into tier -3 is the lowest

4. Gender also plays a contributing part in the hospital charges, Females admitted into hospital tier - 1 is very high compared to Male and also compared to tier 1 and 2.

11) Create a radar chart to showcase the median hospitalization cost for each tier of hospitals

Radar chart or spider chart is created from plotly.express library.

First the median value of hospital charges is calculated for all the 3 tiers of hospitals. This is done using groupby function.

Radar chart generated using r = median cost and theta = hospital tier

12) *Create a frequency table and a stacked bar chart to visualize the count of people in the different tiers of cities and hospitals*

Frequency table of people count across the city tiers and hospital tiers is created using crosstab() method

Stacked bar chart generated from the frequency table across the city tiers and hospital tiers.

13) Test the following null hypotheses:

a. The average hospitalization costs for the three types of hospitals are not significantly different.

NULL HYPOTHESIS: Mean of charges of tier 1, tier 2 and tier 3 hospitals is equal

ALTERNATE HYPOTHESIS: Mean of charges for 3 tiers of hospitals is significantly different

Created 3 subsets for the tier 1, 2 and 3 hospital charges using where condition

Imported scipy.stats library to perform Hypothesis testing.

Since there are more than 2 samples, we go for ANOVA (Analysis of Variance) test to check whether the means are equal.

Result: The p-test value is 6.215e-173 < 0.05, hence we reject the NULL hypotheses.

The average of hospitalization costs is significantly different for all 3 types of hospitals.

b. The average hospitalization costs for the three types of cities are not significantly different.

NULL HYPOTHESIS: Mean of charges of tier 1, tier 2 and tier 3 cities is equal

ALTERNATE HYPOTHESIS: Mean of charges for 3 tiers of cities is significantly different

Created 3 subsets for the tier 1, 2 and 3 cities using where condition

Since there are more than 2 samples, we go for ANOVA (Analysis of Variance) test to check whether the means are equal.

Result: The p-test value is 0.436 > 0.05, hence we fail to reject the NULL hypotheses.

The average of hospitalization costs is not significantly different for all 3 tiers of cities.

c. The average hospitalization cost for smokers is not significantly different from the average cost for nonsmokers.

NULL HYPOTHESIS: Mean of charges of smokers and nonsmokers are not significantly different

ALTERNATE HYPOTHESIS: Mean of charges for smokers and nonsmokers is significantly different

Created 2 subsets for the smokers and nonsmokers using where condition

Hypothesis testing done using independent 2 sample t-test.

Result: The p-test value is 7.521e-242 < 0.05, hence we reject the NULL hypotheses.

The average of hospitalization costs is significantly different for all smokers and nonsmokers.

d. Smoking and heart issues are independent.

H0: (null hypothesis) The two variables are independent.

H1: (alternative hypothesis) The two variables are not independent.

Created frequency table for smoker and heart issues columns.

To check for independence of the 2 variables chi2_contingency function from the SciPy library is applied.

Result: The p-test value is 0.935 > 0.05, hence we fail to reject the NULL hypotheses.

The 2 variables smoking and heart issues are independent of each other.

MACHINE LEARNING:

1) Examine the correlation between predictors to identify highly correlated predictors

Correlation between the different variables can be visualized using a heat map

The columns considered for studying correlation are selected in a list and Heat map is plotted using sns.heatmap()

Observations:

1. Age is positively correlated with having HBA1C
2. High positive correlation between smoking and hospitalization charges
3. Positive correlation between smoking and tier-1 hospitals
4. Positive correlation between age and transplants

2) Develop a regression model Linear or Ridge. Evaluate the model with k-fold cross validation. Also, ensure that you apply all the following suggestions:

•Implement the stratified 5-fold cross validation technique for both model building and validation

•Utilize effective standardization techniques and hyperparameter tuning

•Incorporate sklearn-pipelines to streamline the workflow

•Apply appropriate regularization techniques to address the bias-variance trade-off

•Create five folds in the data, and introduce a variable to identify the folds

•Develop Gradient Boost model and determine the variable importance scores,and identify the redundant variables

Linear Regression Model: Multiple linear regression model used to predict one dependent variable from multiple independent variables. Multicollinearity, correlation between independent variables can cause errors in the regression model and may lead to incorrect relationship between variables.

Ridge Regression Model: This model is used to reduce the complexity of data by shrinking the coefficients to zero. Ridge regression minimizes the variance of the model without increasing the bias, it helps with overfitting of the model. It decreases the multicollinearity between the features in the dataset.

The dataset is split into X (independent variables) and y (target variable). StandardScalar() is used for standardizing the data. KFold() cross validation with 5 folds is used.

GridSearcgCV() applied to search for the best parameters for alpha in ridge regressor.

Gradient Boosting Classifier ensemble learning model applied to the dataset and feature importances are plotted on a bar graph. Smoking, BMI, Age, Hospital tier-1, children, state ID R1011 are the top 6 important variable scores and the remaining variables have significantly low scores and can be considered redundant variables

3) Case scenario:

Estimate the cost of hospitalization for Christopher, Ms. Jayna (Dateof birth12/28/1988;height170 cm;and weight 85 kgs). She lives with her partner and two children in a tier-1 city,and her state's State ID is R1011.She was found to be nondiabetic (HbA1c = 5.8). She smokes but is otherwise healthy. She has had no transplants or major surgeries. Her father died of lung cancer. Hospitalization costs will be estimated using tier-1 hospitals.

The given details of the Case scenario is saved in a new data frame.

4) Find the predicted hospitalization cost using the best models

Gradient Boosting Regressor is used as best model to predict the hospital costs. The data is trained and then predicted using gb_model()

Predicted Hospitilization costs of Christopher, Ms. Jayna is 40134.12

SQL:

1.*To gain a comprehensive understanding of the factors influencing hospitalization costs*

a. Merge the two tables by first identifying the columns in the data tables that will help you in merging

b. In both tables, add a Primary Key constraint for these columns

Created a new database in Mysql workbench. Hospitalization_details and medical_examination files are imported into the Mysql database using Table import wizard.

The 2 tables are merged using inner join on column Customer ID and new table created mergeddata

The columns having null values from mergeddata are dropped using where condition

Primary key is added on the customer ID column

2) Retrieve information about people who are diabetic and have heart problems with their average age, the average number of dependent children, average BMI, and average hospitalization costs

Age is calculated from the current_date and year, month and date columns.

Average age, BMI, children and hospital charges of people having both diabetes and heart issues is calculated using where condition and AVG() in the query

3) Find the average hospitalization cost for each hospital tier and each city level

Average charges calculated by group by clause on Hospital tier and city tier columns

4) Determine the number of people who have had major surgery with a history of cancer

Count of people who had a cancer history and also had a major surgery is calculated using where clause and COUNT() function

5) Determine the number of tier-1 hospitals in each state

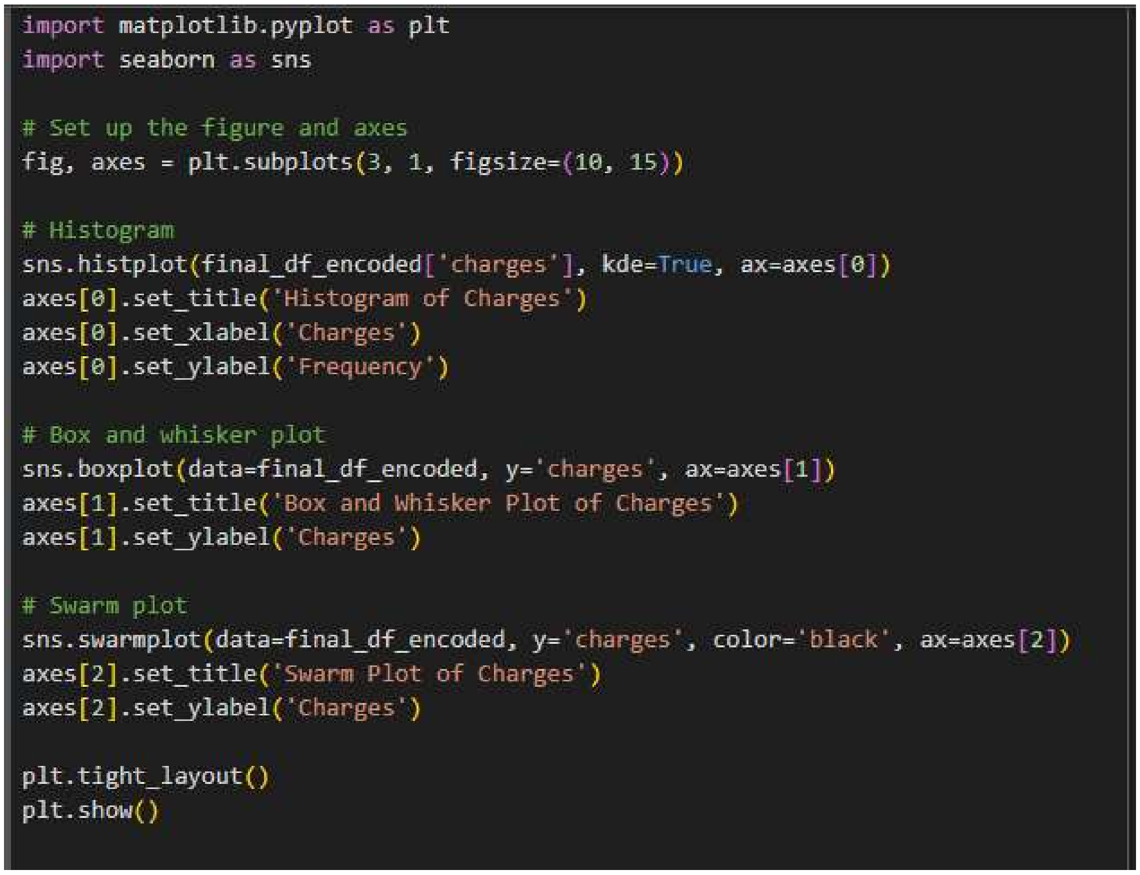
Count is calculated using where `Hospital tier` = 'tier - 1' group by `State ID` clause

TABLEAU

1) Create a dashboard in Tableau by selecting the appropriate chart types and business metrics

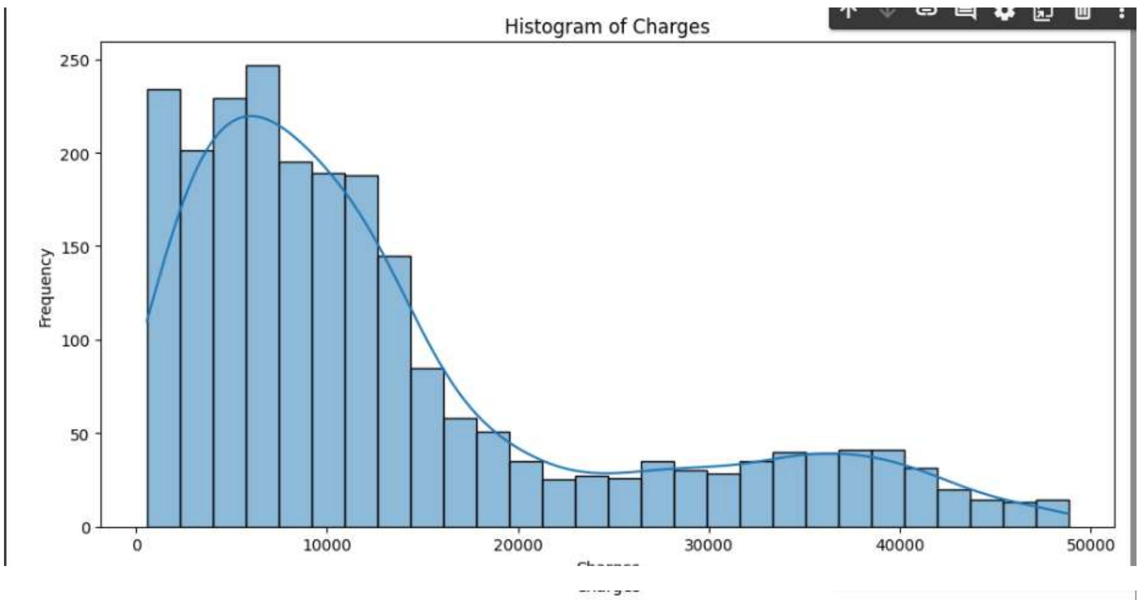
Dashboard created which depicts the distribution and the main factors for hospitalization charges

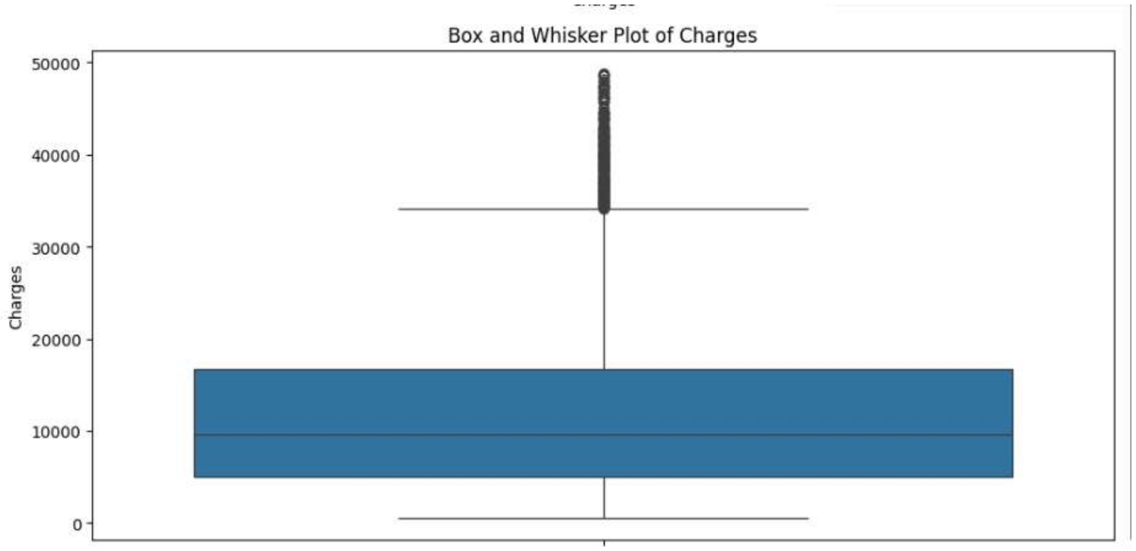
DATA VISUALIZATION:

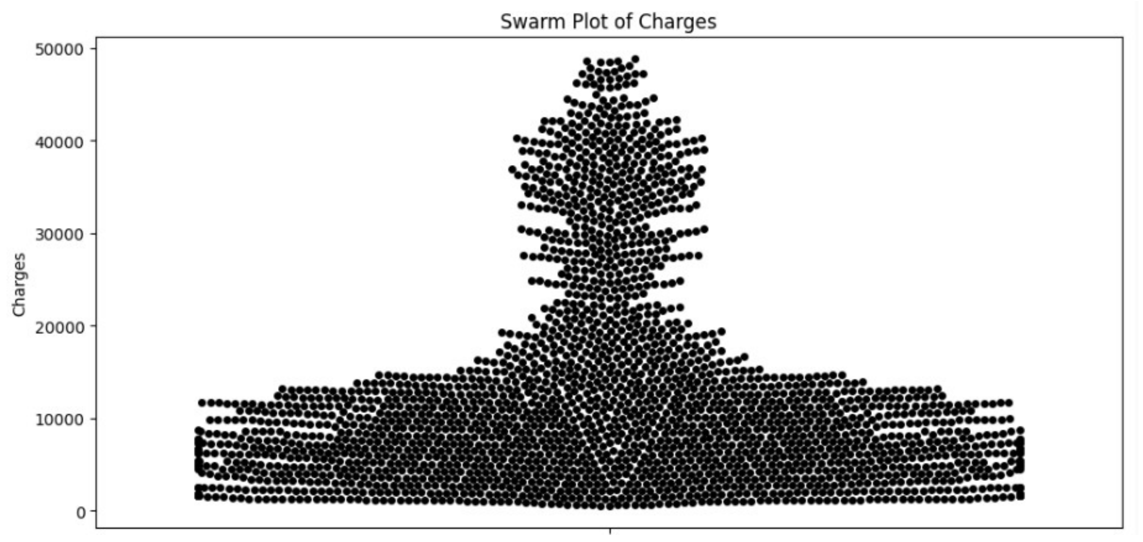


```python
import matplotlib.pyplot as plt
import seaborn as sns

# Set up the figure and axes
fig, axes = plt.subplots(3, 1, figsize=(10, 15))

# Histogram
sns.histplot(final_df_encoded['charges'], kde=True, ax=axes[0])
axes[0].set_title('Histogram of Charges')
axes[0].set_xlabel('Charges')
axes[0].set_ylabel('Frequency')

# Box and whisker plot
sns.boxplot(data=final_df_encoded, y='charges', ax=axes[1])
axes[1].set_title('Box and Whisker Plot of Charges')
axes[1].set_ylabel('Charges')

# Swarm plot
sns.swarmplot(data=final_df_encoded, y='charges', color='black', ax=axes[2])
axes[2].set_title('Swarm Plot of Charges')
axes[2].set_ylabel('Charges')

plt.tight_layout()
plt.show()
```

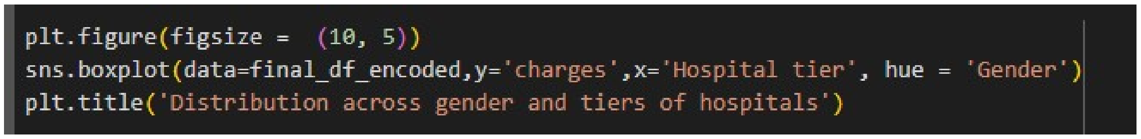
9) Distribution of costs using a histogram, box and whisker plot, and swarm plot:


Histogram of Charges
Frequency
250
200
150
100
50
0
0
10000
20000
30000
40000
50000

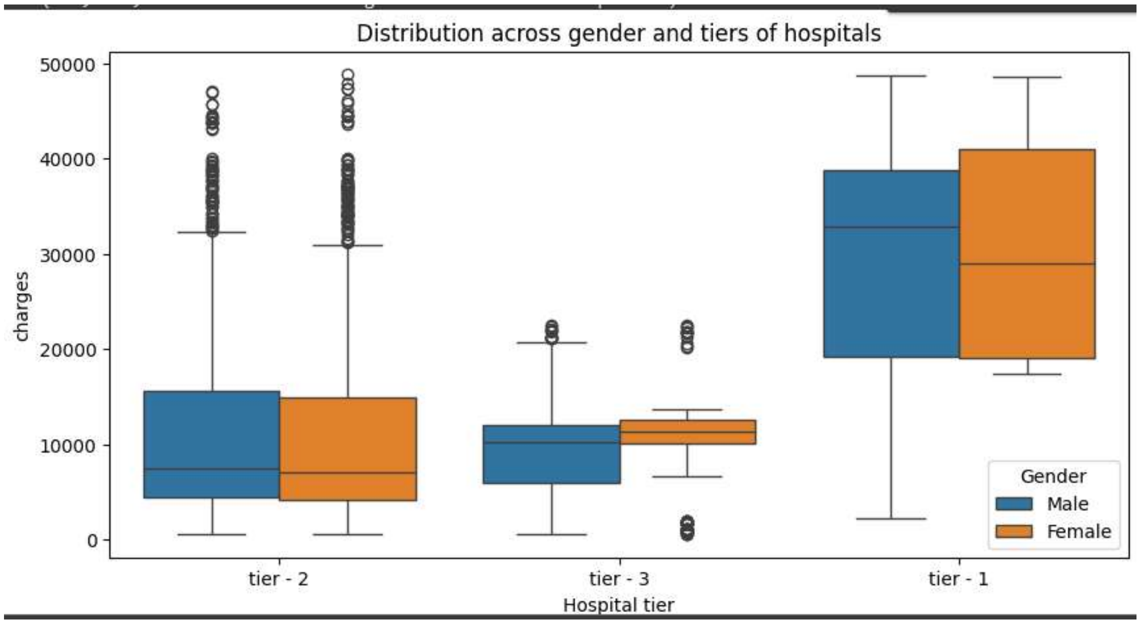

Box and Whisker Plot of Charges
Charges
50000
40000
30000
20000
10000
0



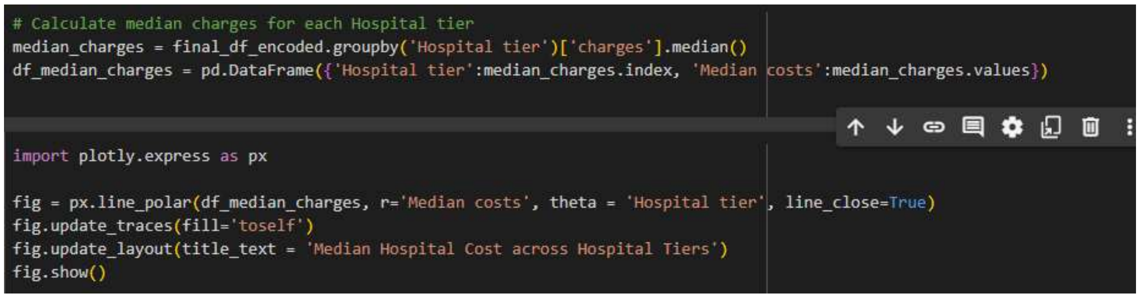
10) Distribution of costs across gender and tiers of hospitals:



```
plt.figure(figsize =  (10, 5))
sns.boxplot(data=final_df_encoded,y='charges',x='Hospital tier', hue = 'Gender')
plt.title('Distribution across gender and tiers of hospitals')
```

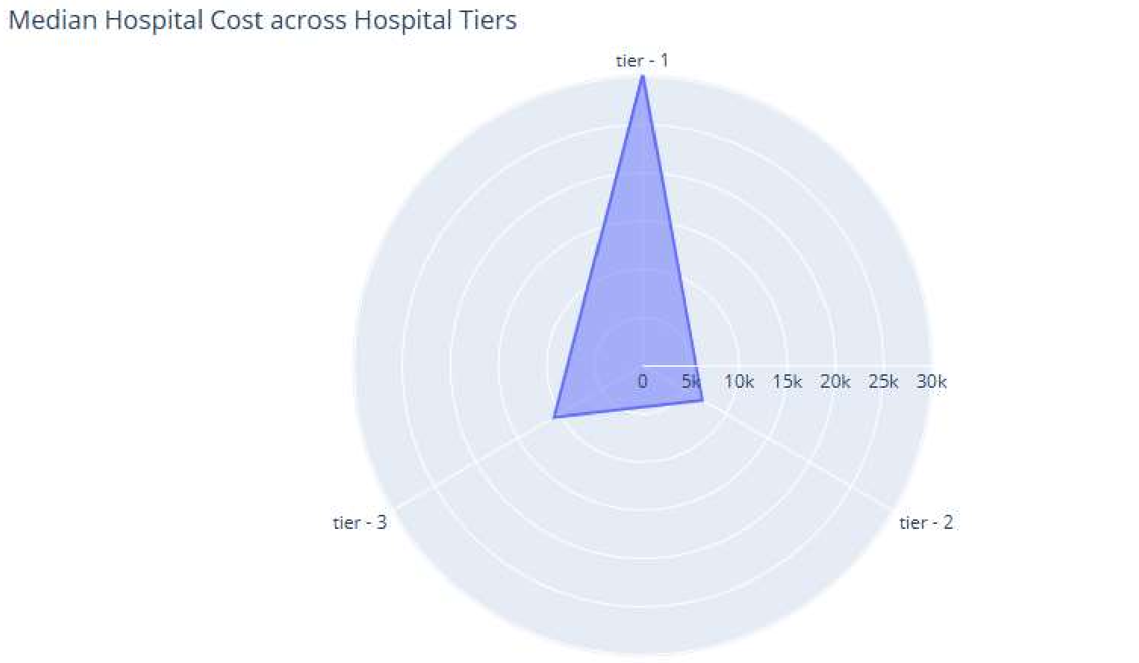


11) Radar chart to showcase the median hospitalization cost for each tier of hospitals:

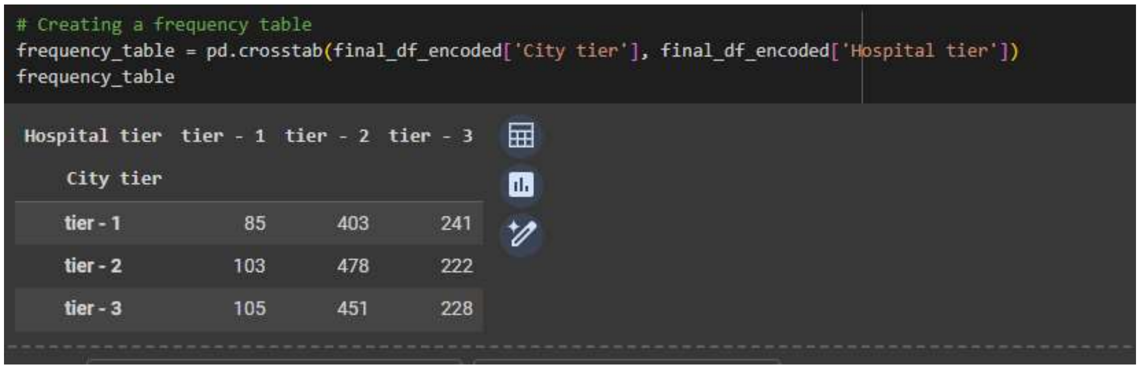


```
# Calculate median charges for each Hospital tier
median_charges = final_df_encoded.groupby('Hospital tier')['charges'].median()
df_median_charges = pd.DataFrame({'Hospital tier':median_charges.index, 'Median costs':median_charges.values})
```

```
import plotly.express as px

fig = px.line_polar(df_median_charges, r='Median costs', theta = 'Hospital tier', line_close=True)
fig.update_traces(fill='toself')
fig.update_layout(title_text = 'Median Hospital Cost across Hospital Tiers')
fig.show()
```



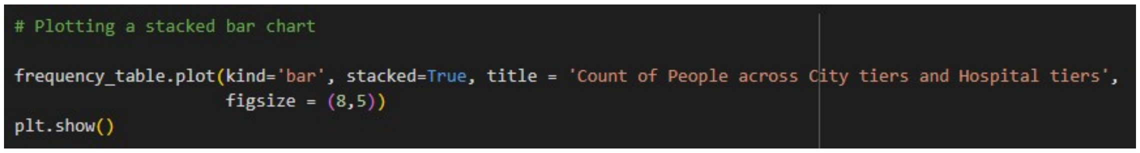
12) Frequency table visualize the count of people in the different tiers of cities and hospitals:



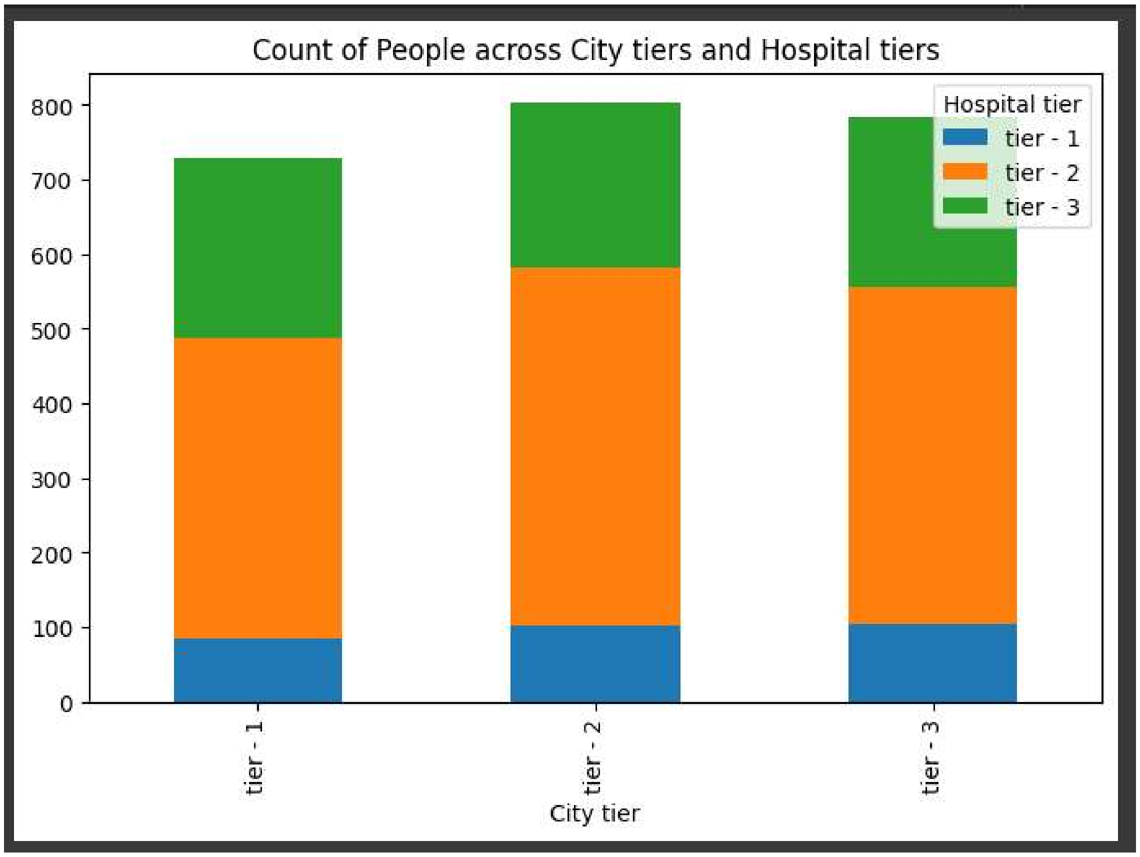
```
# Creating a frequency table
frequency_table = pd.crosstab(final_df_encoded['City tier'], final_df_encoded['Hospital tier'])
frequency_table
```

Hospital tier City tier	tier - 1	tier - 2	tier - 3
tier - 1	85	403	241
tier - 2	103	478	222
tier - 3	105	451	228

Stacked bar chart to visualize the count of people in the different tiers of cities and hospitals:

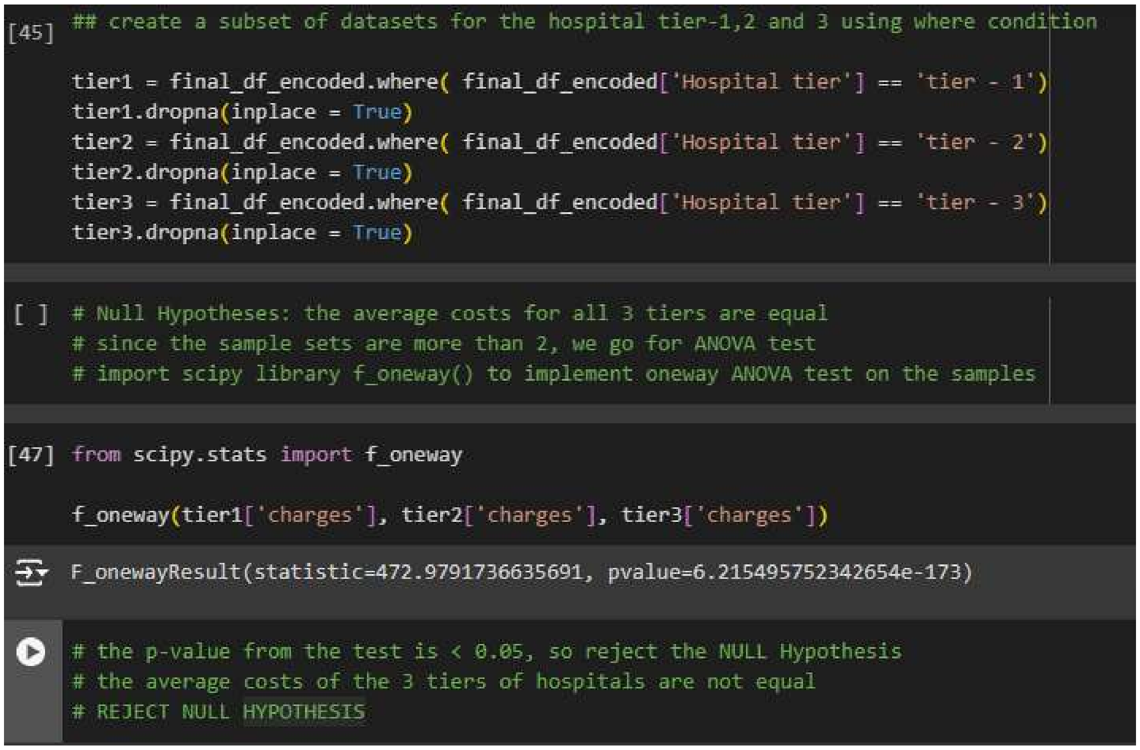


```
# Plotting a stacked bar chart

frequency_table.plot(kind='bar', stacked=True, title = 'Count of People across City tiers and Hospital tiers',
                     figsize = (8,5))
plt.show()
```



13) Test the following null hypotheses:

a. The average hospitalization costs for the three types of hospitals are not significantly different.



```
[45] ## create a subset of datasets for the hospital tier-1,2 and 3 using where condition

     tier1 = final_df_encoded.where( final_df_encoded['Hospital tier'] == 'tier - 1')
     tier1.dropna(inplace = True)
     tier2 = final_df_encoded.where( final_df_encoded['Hospital tier'] == 'tier - 2')
     tier2.dropna(inplace = True)
     tier3 = final_df_encoded.where( final_df_encoded['Hospital tier'] == 'tier - 3')
     tier3.dropna(inplace = True)
```

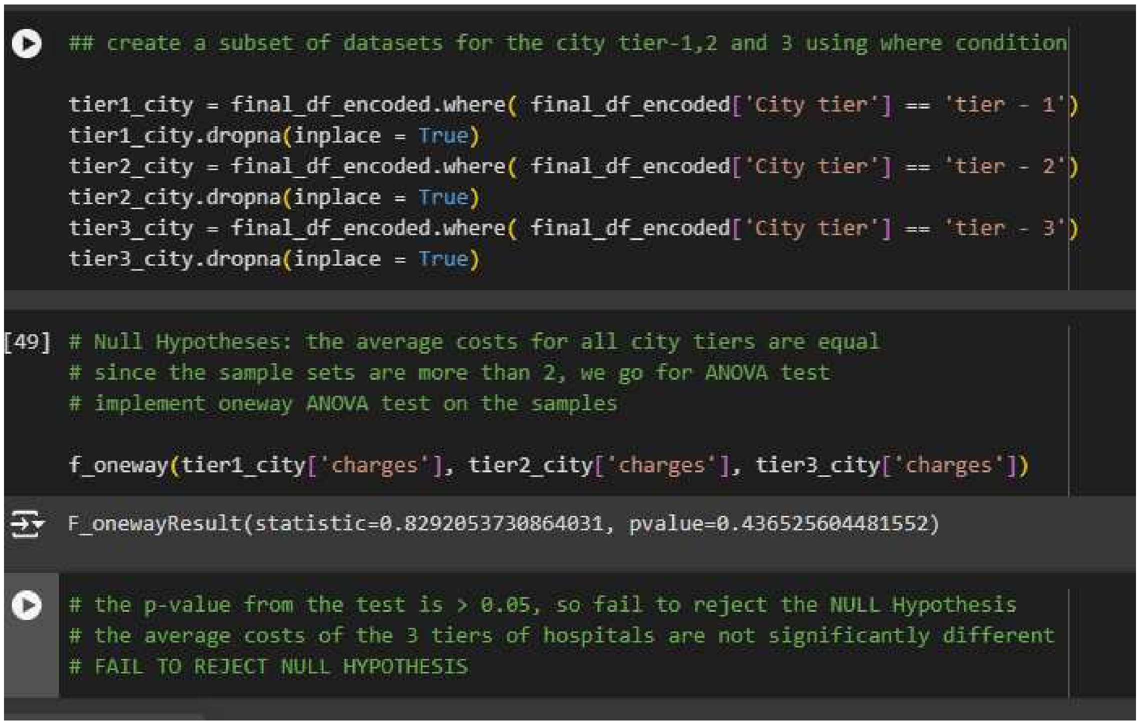
```
[ ]  # Null Hypotheses: the average costs for all 3 tiers are equal
     # since the sample sets are more than 2, we go for ANOVA test
     # import scipy library f_oneway() to implement oneway ANOVA test on the samples
```

```
[47] from scipy.stats import f_oneway

     f_oneway(tier1['charges'], tier2['charges'], tier3['charges'])
```

```
F_onewayResult(statistic=472.9791736635691, pvalue=6.215495752342654e-173)
```

```
# the p-value from the test is < 0.05, so reject the NULL Hypothesis
# the average costs of the 3 tiers of hospitals are not equal
# REJECT NULL HYPOTHESIS
```

13 b) The average hospitalization costs for the three types of cities are not significantly different



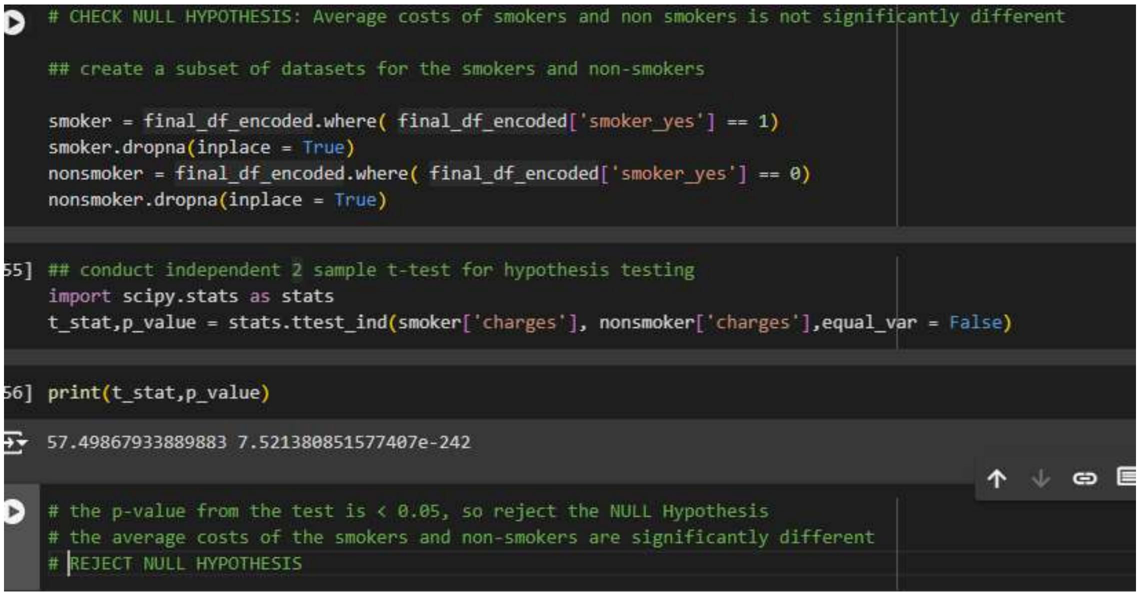
```
## create a subset of datasets for the city tier-1,2 and 3 using where condition

tier1_city = final_df_encoded.where( final_df_encoded['City tier'] == 'tier - 1')
tier1_city.dropna(inplace = True)
tier2_city = final_df_encoded.where( final_df_encoded['City tier'] == 'tier - 2')
tier2_city.dropna(inplace = True)
tier3_city = final_df_encoded.where( final_df_encoded['City tier'] == 'tier - 3')
tier3_city.dropna(inplace = True)
```

```
[49] # Null Hypotheses: the average costs for all city tiers are equal
     # since the sample sets are more than 2, we go for ANOVA test
     # implement oneway ANOVA test on the samples

     f_oneway(tier1_city['charges'], tier2_city['charges'], tier3_city['charges'])
```

```
F_onewayResult(statistic=0.8292053730864031, pvalue=0.436525604481552)
```

```
# the p-value from the test is > 0.05, so fail to reject the NULL Hypothesis
# the average costs of the 3 tiers of hospitals are not significantly different
# FAIL TO REJECT NULL HYPOTHESIS
```

13 c) The average hospitalization cost for smokers is not significantly different from the average cost for nonsmokers.



```
# CHECK NULL HYPOTHESIS: Average costs of smokers and non smokers is not significantly different

## create a subset of datasets for the smokers and non-smokers

smoker = final_df_encoded.where( final_df_encoded['smoker_yes'] == 1)
smoker.dropna(inplace = True)
nonsmoker = final_df_encoded.where( final_df_encoded['smoker_yes'] == 0)
nonsmoker.dropna(inplace = True)
```

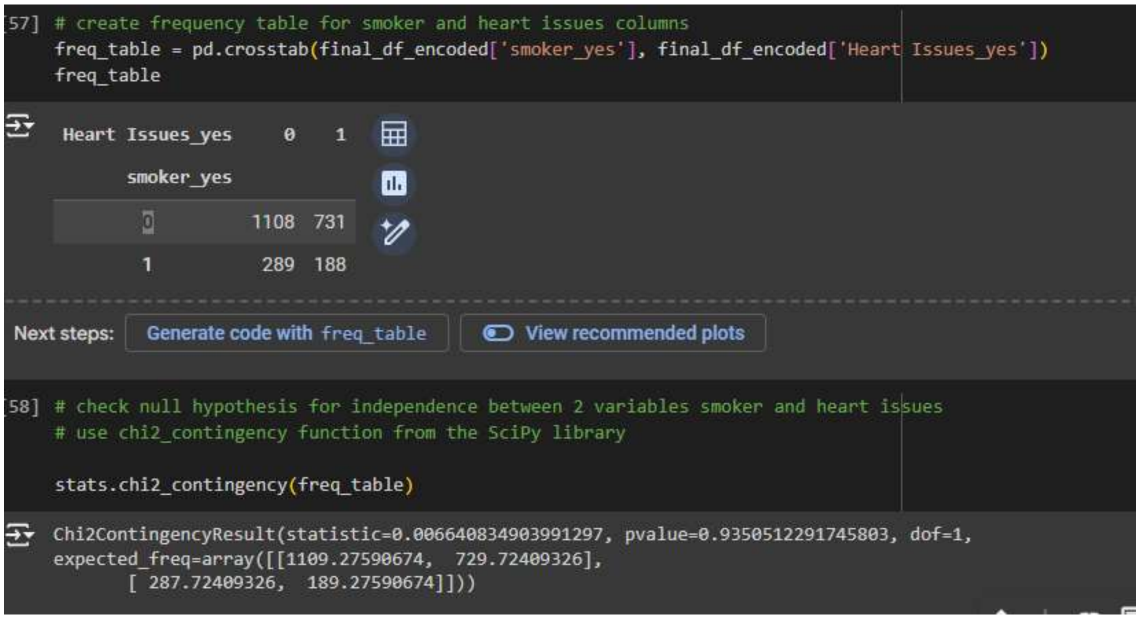
```
55] ## conduct independent 2 sample t-test for hypothesis testing
import scipy.stats as stats
t_stat,p_value = stats.ttest_ind(smoker['charges'], nonsmoker['charges'],equal_var = False)
```

```
56] print(t_stat,p_value)
```

```
57.49867933889883 7.521380851577407e-242
```

```
# the p-value from the test is < 0.05, so reject the NULL Hypothesis
# the average costs of the smokers and non-smokers are significantly different
# REJECT NULL HYPOTHESIS
```

13 d) Smoking and heart issues are independent.



```
[57] # create frequency table for smoker and heart issues columns
freq_table = pd.crosstab(final_df_encoded['smoker_yes'], final_df_encoded['Heart Issues_yes'])
freq_table
```

```
Heart Issues_yes     0    1
      smoker_yes
               0  1108  731
               1   289  188
```

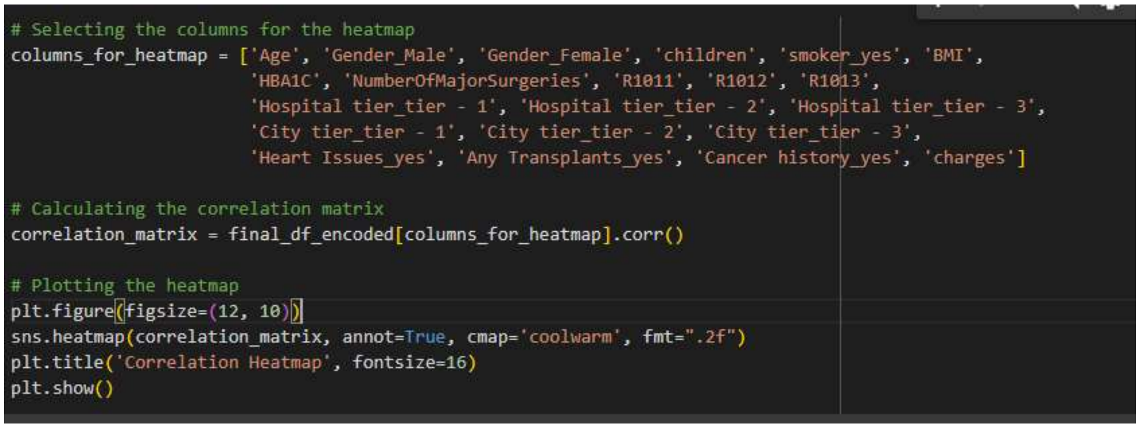
```
[58] # check null hypothesis for independence between 2 variables smoker and heart issues
# use chi2_contingency function from the SciPy library

stats.chi2_contingency(freq_table)
```

```
Chi2ContingencyResult(statistic=0.006640834903991297, pvalue=0.9350512291745803, dof=1,
expected_freq=array([[1109.27590674,  729.72409326],
       [ 287.72409326,  189.27590674]]))
```

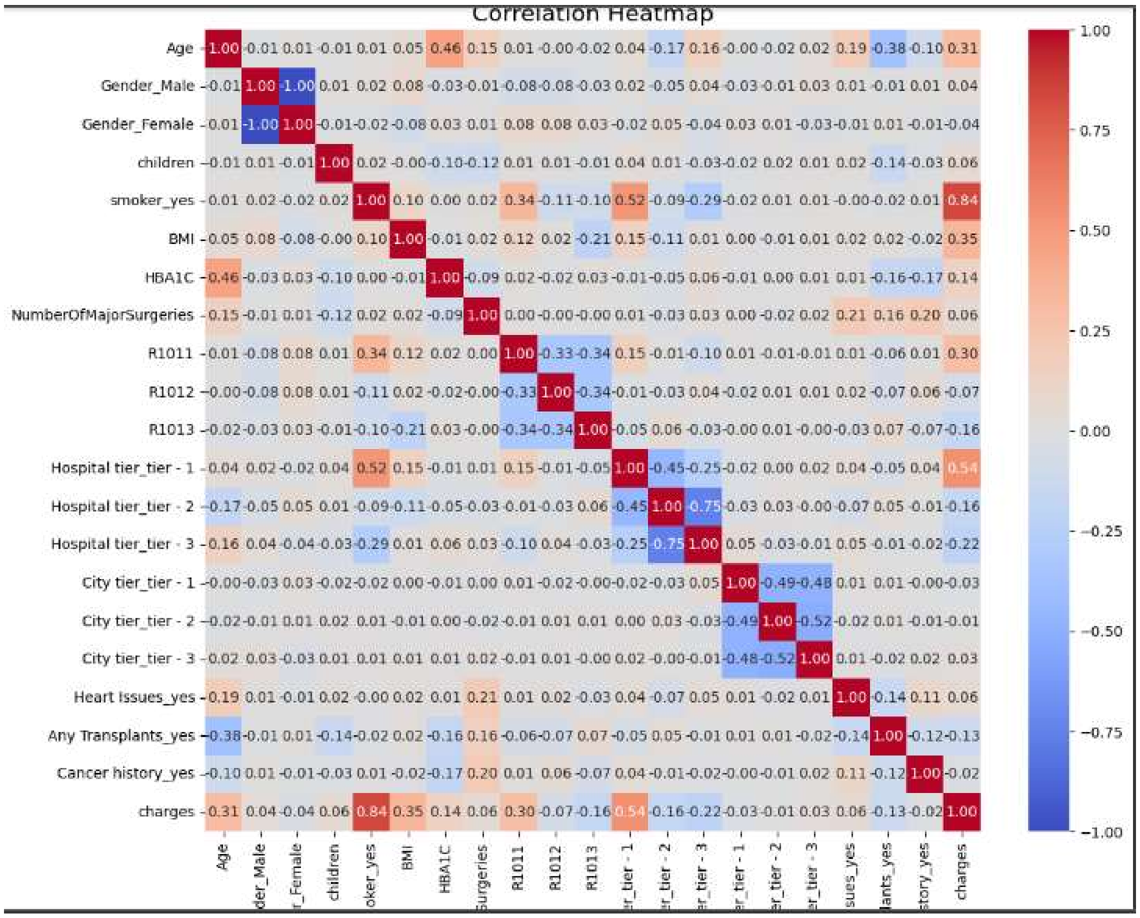
# H0: (null hypothesis) The two variables are independent.
H1: (alternative hypothesis) The two variables are not independent.
the p-value from the test is > 0.05, so fail to reject the NULL Hypothesis
# the 2 variables smoking and heart issues are independent of each other
# FAIL TO REJECT NULL HYPOTHESIS

MACHINE LEARNING:

1) Examine the correlation between predictors to identify highly correlated predictors:

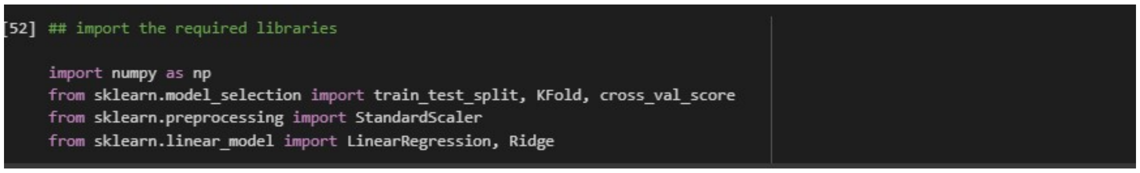


```
# Selecting the columns for the heatmap
columns_for_heatmap = ['Age', 'Gender_Male', 'Gender_Female', 'children', 'smoker_yes', 'BMI',
                       'HBA1C', 'NumberOfMajorSurgeries', 'R1011', 'R1012', 'R1013',
                       'Hospital tier_tier - 1', 'Hospital tier_tier - 2', 'Hospital tier_tier - 3',
                       'City tier_tier - 1', 'City tier_tier - 2', 'City tier_tier - 3',
                       'Heart Issues_yes', 'Any Transplants_yes', 'Cancer history_yes', 'charges']

# Calculating the correlation matrix
correlation_matrix = final_df_encoded[columns_for_heatmap].corr()

# Plotting the heatmap
plt.figure(figsize=(12, 10))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Heatmap', fontsize=16)
plt.show()
```



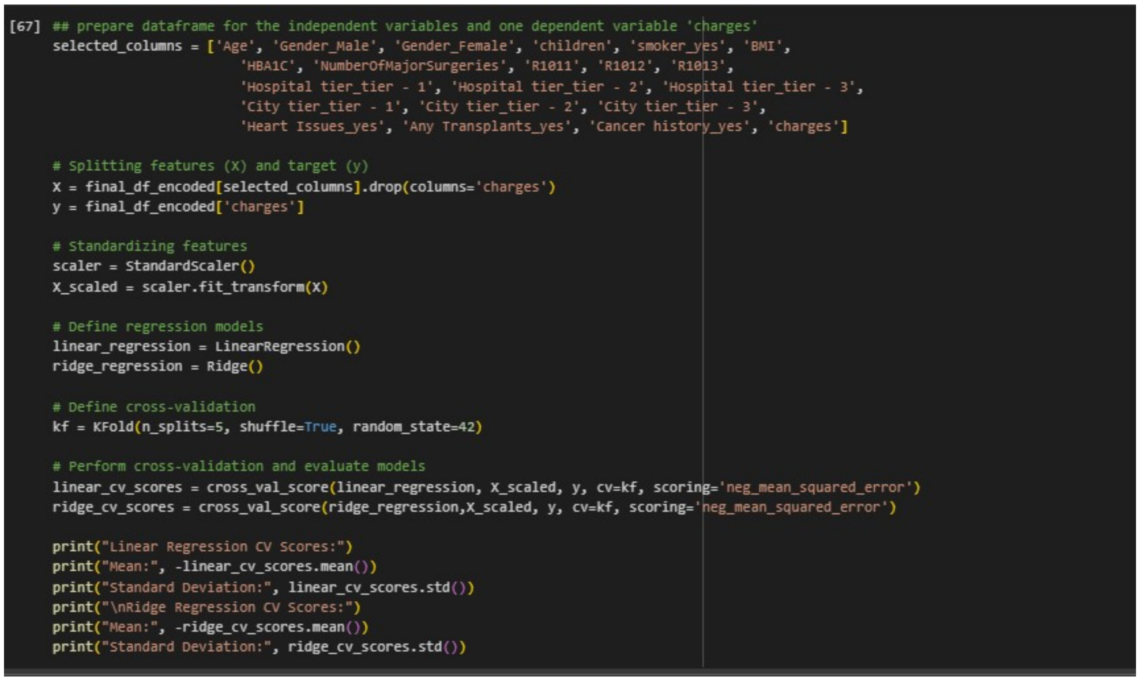
	Age	der_Male	r_Female	children	oker_yes	BMI	HBA1C	Surgeries	R1011	R1012	R1013	r_tier - 1	r_tier - 2	r_tier - 3	r_tier - 1	r_tier - 2	r_tier - 3	sues_yes	lants_yes	story_yes	charges
Age	1.00	-0.01	0.01	-0.01	0.01	0.05	0.46	0.15	0.01	-0.00	-0.02	0.04	-0.17	0.16	-0.00	-0.02	0.02	0.19	-0.38	-0.10	0.31
Gender_Male	-0.01	1.00	-1.00	0.01	0.02	0.08	-0.03	-0.01	-0.08	-0.08	-0.03	0.02	-0.05	0.04	-0.03	-0.01	0.03	0.01	-0.01	0.01	0.04
Gender_Female	-0.01	-1.00	1.00	-0.01	-0.02	-0.08	0.03	0.01	0.08	0.08	0.03	-0.02	0.05	-0.04	0.03	0.01	-0.03	-0.01	0.01	-0.01	-0.04
children	-0.01	0.01	-0.01	1.00	0.02	-0.00	-0.10	-0.12	0.01	0.01	-0.01	0.04	0.01	-0.03	-0.02	0.02	0.01	0.02	-0.14	-0.03	0.06
smoker_yes	0.01	0.02	-0.02	0.02	1.00	0.10	0.00	0.02	0.34	-0.11	-0.10	0.52	-0.09	-0.29	-0.02	0.01	0.01	-0.00	-0.02	0.01	0.84
BMI	0.05	0.08	-0.08	-0.00	0.10	1.00	-0.01	0.02	0.12	0.02	-0.21	0.15	-0.11	0.01	0.00	-0.01	0.01	0.02	0.02	-0.02	0.35
HBA1C	0.46	-0.03	0.03	-0.10	0.00	-0.01	1.00	-0.09	0.02	-0.02	0.03	-0.01	-0.05	0.06	-0.01	0.00	0.01	0.01	-0.16	-0.17	0.14
NumberOfMajorSurgeries	0.15	-0.01	0.01	-0.12	0.02	0.02	-0.09	1.00	0.00	-0.00	-0.00	0.01	-0.03	0.03	0.00	-0.02	0.02	0.21	0.16	0.20	0.06
R1011	0.01	-0.08	0.08	0.01	0.34	0.12	0.02	0.00	1.00	-0.33	-0.34	0.15	-0.01	-0.10	0.01	-0.01	-0.01	0.01	-0.06	0.01	0.30
R1012	-0.00	-0.08	0.08	0.01	-0.11	0.02	-0.02	-0.00	-0.33	1.00	-0.34	-0.01	-0.03	0.04	-0.02	0.01	0.01	0.02	-0.07	0.06	-0.07
R1013	-0.02	-0.03	0.03	-0.01	-0.10	-0.21	0.03	-0.00	-0.34	-0.34	1.00	-0.05	0.06	-0.03	-0.00	0.01	-0.00	-0.03	0.07	-0.07	-0.16
Hospital tier_tier - 1	0.04	0.02	-0.02	0.04	0.52	0.15	-0.01	0.01	0.15	-0.01	-0.05	1.00	-0.45	-0.25	-0.02	0.00	0.02	0.04	-0.05	0.04	0.54
Hospital tier_tier - 2	-0.17	-0.05	0.05	0.01	-0.09	-0.11	-0.05	-0.03	-0.01	-0.03	0.06	-0.45	1.00	-0.75	-0.03	0.03	-0.00	-0.07	0.05	-0.01	-0.16
Hospital tier_tier - 3	0.16	0.04	-0.04	-0.03	-0.29	0.01	0.06	0.03	-0.10	0.04	-0.03	-0.25	-0.75	1.00	0.05	-0.03	-0.01	0.05	-0.01	-0.02	-0.22
City tier_tier - 1	-0.00	-0.03	0.03	-0.02	-0.02	0.00	-0.01	0.00	0.01	-0.02	-0.00	-0.02	-0.03	0.05	1.00	-0.49	-0.48	0.01	0.01	-0.00	-0.03
City tier_tier - 2	-0.02	-0.01	0.01	0.02	0.01	-0.01	0.00	-0.02	-0.01	0.01	0.01	0.00	0.03	-0.03	-0.49	1.00	-0.52	-0.02	0.01	-0.01	-0.01
City tier_tier - 3	0.02	0.03	-0.03	0.01	0.01	0.01	0.01	0.02	-0.01	0.01	-0.00	0.02	-0.00	-0.01	-0.48	-0.52	1.00	0.01	-0.02	0.02	0.03
Heart Issues_yes	0.19	0.01	-0.01	0.02	-0.00	0.02	0.01	0.21	0.01	0.02	-0.03	0.04	-0.07	0.05	0.01	-0.02	0.01	1.00	-0.14	0.11	0.06
Any Transplants_yes	-0.38	-0.01	0.01	-0.14	-0.02	0.02	-0.16	0.16	-0.06	-0.07	0.07	-0.05	0.05	-0.01	0.01	0.01	-0.02	-0.14	1.00	-0.12	-0.13
Cancer history_yes	-0.10	0.01	-0.01	-0.03	0.01	-0.02	-0.17	0.20	0.01	0.06	-0.07	0.04	-0.01	-0.02	-0.00	-0.01	0.02	0.11	-0.12	1.00	-0.02
charges	0.31	0.04	-0.04	0.06	0.84	0.35	0.14	0.06	0.30	-0.07	-0.16	0.54	-0.16	-0.22	-0.03	-0.01	0.03	0.06	-0.13	-0.02	1.00

2) Develop a regression model Linear or Ridge. Evaluate the model with k-fold cross validation*:*



```
[52] ## import the required libraries

     import numpy as np
     from sklearn.model_selection import train_test_split, KFold, cross_val_score
     from sklearn.preprocessing import StandardScaler
     from sklearn.linear_model import LinearRegression, Ridge
```

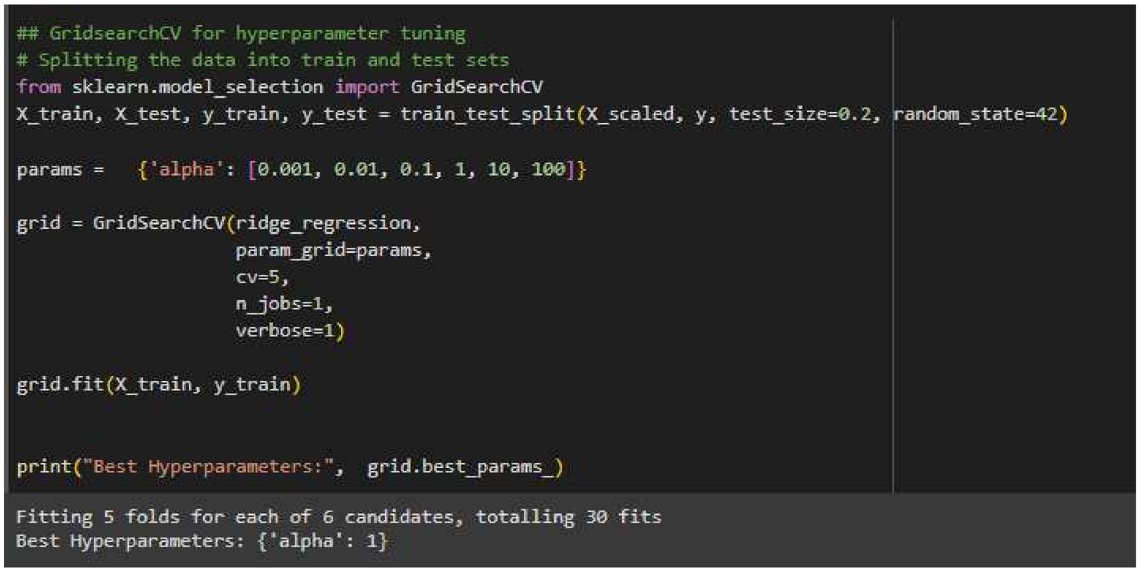
Standardization of data and KFold cross validation:



```
[67] ## prepare dataframe for the independent variables and one dependent variable 'charges'
     selected_columns = ['Age', 'Gender_Male', 'Gender_Female', 'children', 'smoker_yes', 'BMI',
                         'HBA1C', 'NumberOfMajorSurgeries', 'R1011', 'R1012', 'R1013',
                         'Hospital tier_tier - 1', 'Hospital tier_tier - 2', 'Hospital tier_tier - 3',
                         'City tier_tier - 1', 'City tier_tier - 2', 'City tier_tier - 3',
                         'Heart Issues_yes', 'Any Transplants_yes', 'Cancer history_yes', 'charges']

     # Splitting features (X) and target (y)
     X = final_df_encoded[selected_columns].drop(columns='charges')
     y = final_df_encoded['charges']

     # Standardizing features
     scaler = StandardScaler()
     X_scaled = scaler.fit_transform(X)

     # Define regression models
     linear_regression = LinearRegression()
     ridge_regression = Ridge()

     # Define cross-validation
     kf = KFold(n_splits=5, shuffle=True, random_state=42)

     # Perform cross-validation and evaluate models
     linear_cv_scores = cross_val_score(linear_regression, X_scaled, y, cv=kf, scoring='neg_mean_squared_error')
     ridge_cv_scores = cross_val_score(ridge_regression,X_scaled, y, cv=kf, scoring='neg_mean_squared_error')

     print("Linear Regression CV Scores:")
     print("Mean:", -linear_cv_scores.mean())
     print("Standard Deviation:", linear_cv_scores.std())
     print("\nRidge Regression CV Scores:")
     print("Mean:", -ridge_cv_scores.mean())
     print("Standard Deviation:", ridge_cv_scores.std())
```

```
Linear Regression CV Scores:
Mean: 18594113.85384363
Standard Deviation: 1019055.7484081907

Ridge Regression CV Scores:
Mean: 18581043.907581866
Standard Deviation: 1001070.2553831289
```

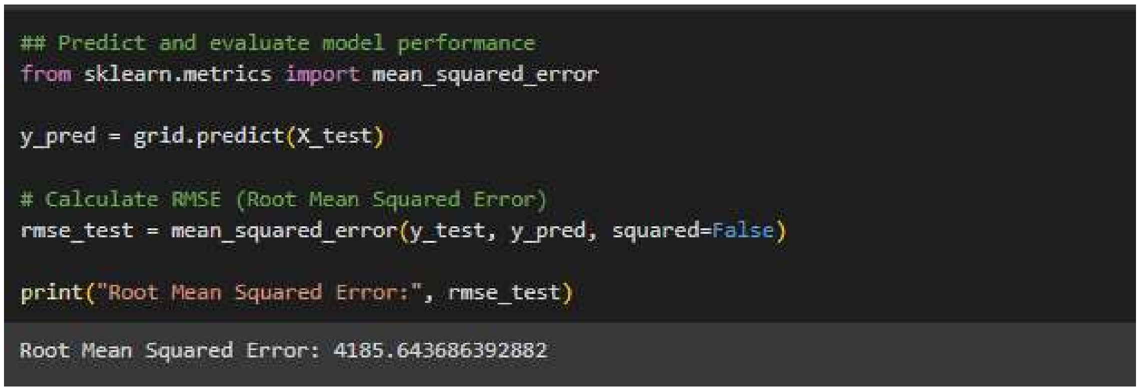
Grid Search for hyper parameter tuning:



```python
## GridsearchCV for hyperparameter tuning
# Splitting the data into train and test sets
from sklearn.model_selection import GridSearchCV
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)

params =   {'alpha': [0.001, 0.01, 0.1, 1, 10, 100]}

grid = GridSearchCV(ridge_regression,
                    param_grid=params,
                    cv=5,
                    n_jobs=1,
                    verbose=1)

grid.fit(X_train, y_train)


print("Best Hyperparameters:",  grid.best_params_)
```

```
Fitting 5 folds for each of 6 candidates, totalling 30 fits
Best Hyperparameters: {'alpha': 1}
```

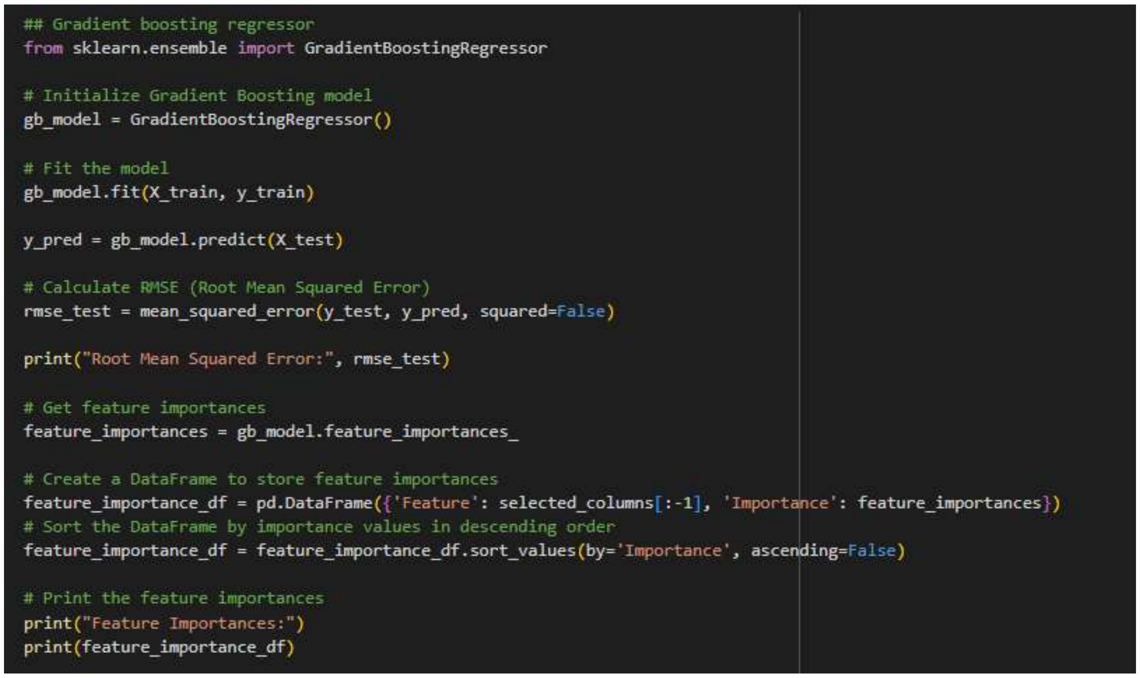
Model performance:



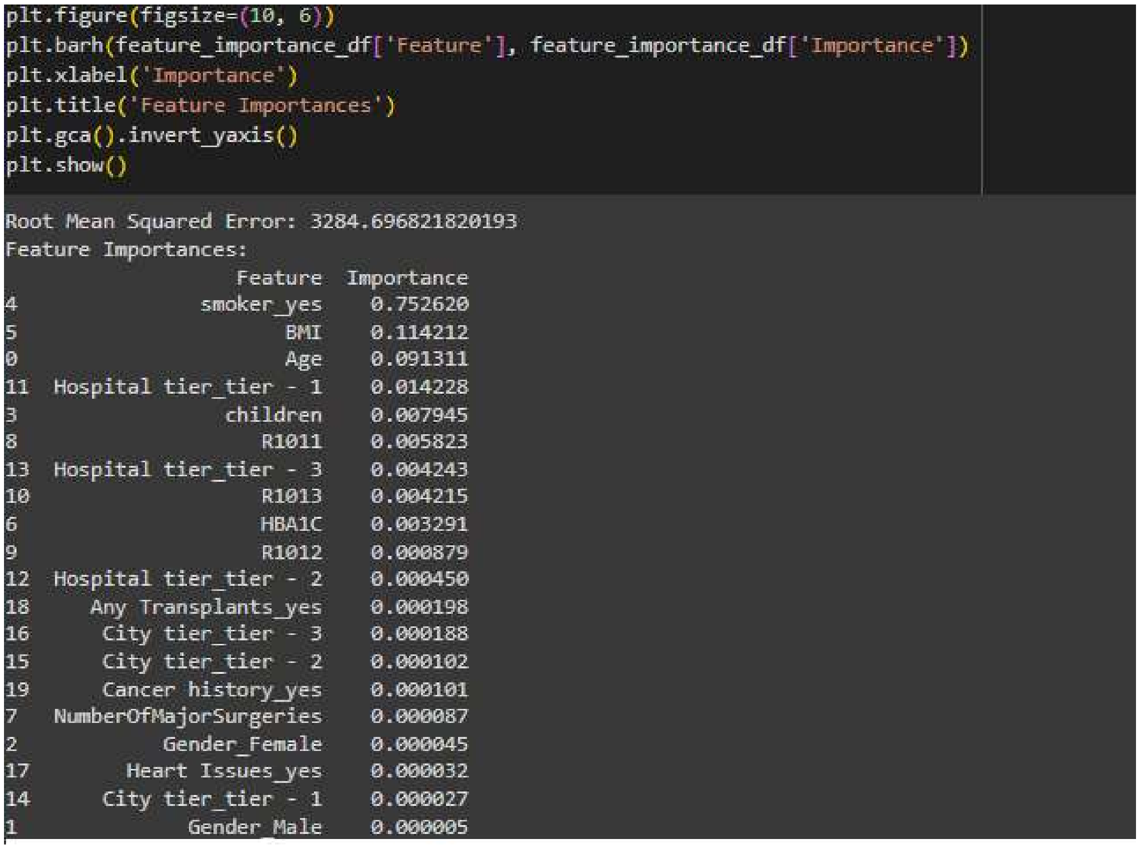
```python
## Predict and evaluate model performance
from sklearn.metrics import mean_squared_error

y_pred = grid.predict(X_test)

# Calculate RMSE (Root Mean Squared Error)
rmse_test = mean_squared_error(y_test, y_pred, squared=False)

print("Root Mean Squared Error:", rmse_test)
```

```
Root Mean Squared Error: 4185.643686392882
```

Gradient Boosting Classifier:

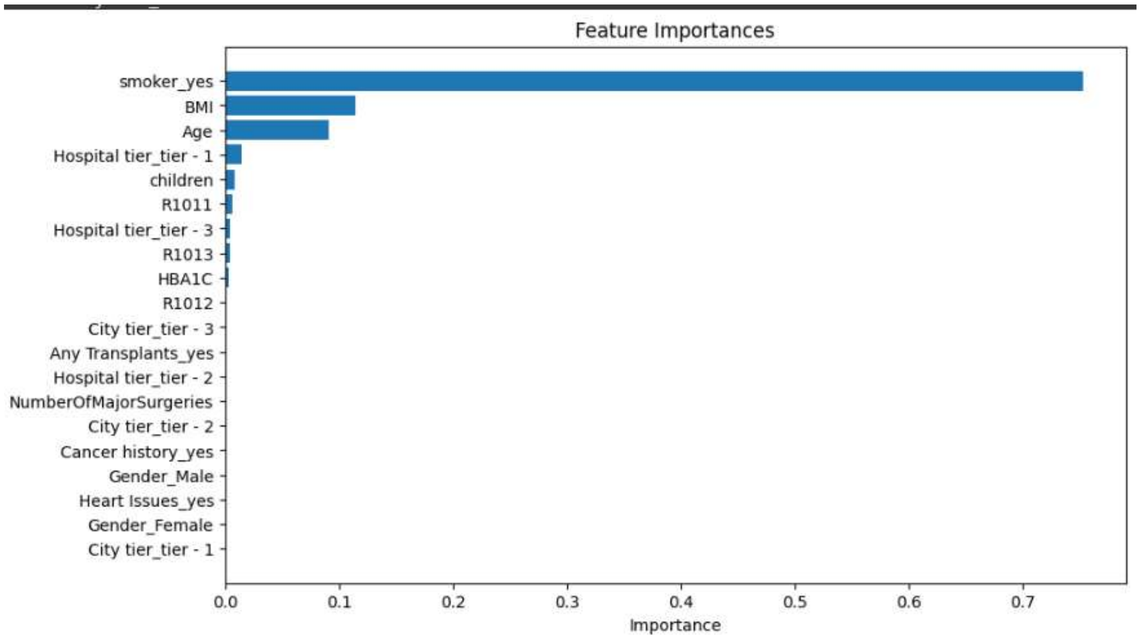


```python
## Gradient boosting regressor
from sklearn.ensemble import GradientBoostingRegressor

# Initialize Gradient Boosting model
gb_model = GradientBoostingRegressor()

# Fit the model
gb_model.fit(X_train, y_train)

y_pred = gb_model.predict(X_test)

# Calculate RMSE (Root Mean Squared Error)
rmse_test = mean_squared_error(y_test, y_pred, squared=False)

print("Root Mean Squared Error:", rmse_test)

# Get feature importances
feature_importances = gb_model.feature_importances_

# Create a DataFrame to store feature importances
feature_importance_df = pd.DataFrame({'Feature': selected_columns[:-1], 'Importance': feature_importances})
# Sort the DataFrame by importance values in descending order
feature_importance_df = feature_importance_df.sort_values(by='Importance', ascending=False)

# Print the feature importances
print("Feature Importances:")
print(feature_importance_df)
```

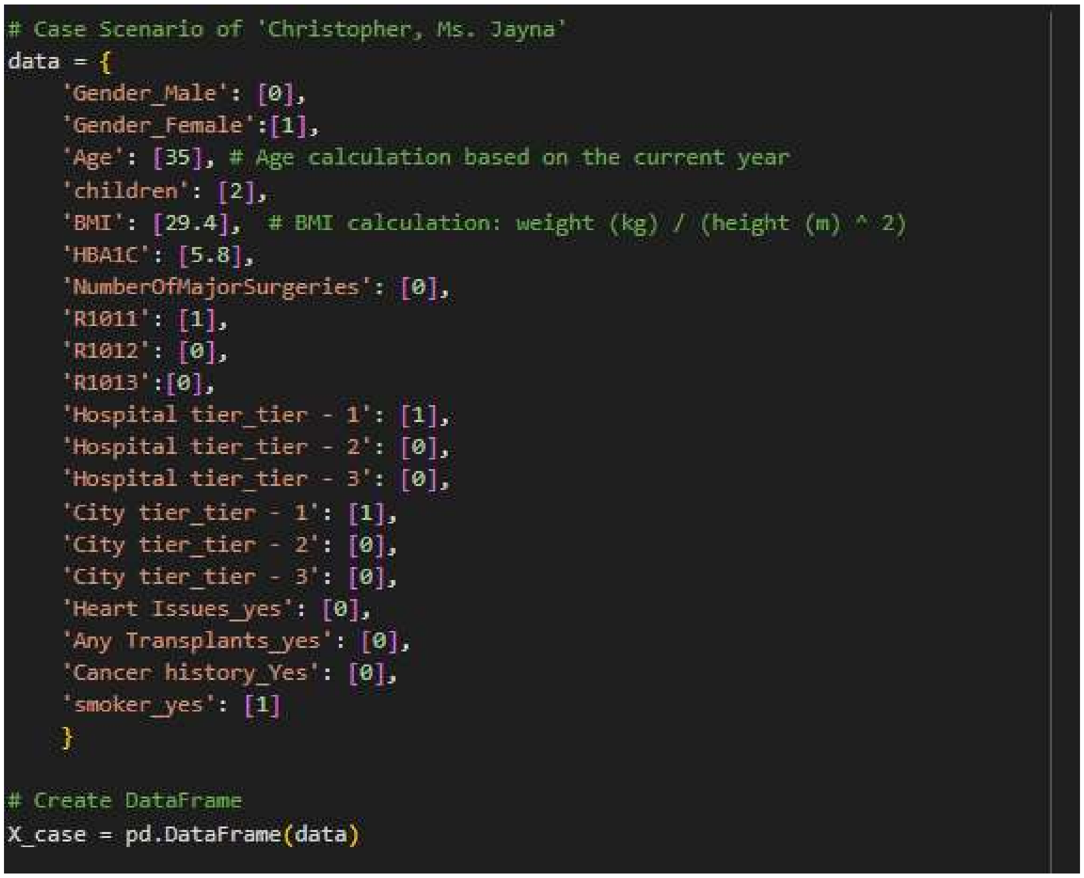


```python
plt.figure(figsize=(10, 6))
plt.barh(feature_importance_df['Feature'], feature_importance_df['Importance'])
plt.xlabel('Importance')
plt.title('Feature Importances')
plt.gca().invert_yaxis()
plt.show()
```

```
Root Mean Squared Error: 3284.696821820193
Feature Importances:
                   Feature  Importance
4               smoker_yes    0.752620
5                      BMI    0.114212
0                      Age    0.091311
11  Hospital tier_tier - 1    0.014228
3                 children    0.007945
8                    R1011    0.005823
13  Hospital tier_tier - 3    0.004243
10                   R1013    0.004215
6                    HBA1C    0.003291
9                    R1012    0.000879
12  Hospital tier_tier - 2    0.000450
18     Any Transplants_yes    0.000198
16      City tier_tier - 3    0.000188
15      City tier_tier - 2    0.000102
19      Cancer history_yes    0.000101
7   NumberOfMajorSurgeries    0.000087
2            Gender_Female    0.000045
17        Heart Issues_yes    0.000032
14      City tier_tier - 1    0.000027
1              Gender_Male    0.000005
```

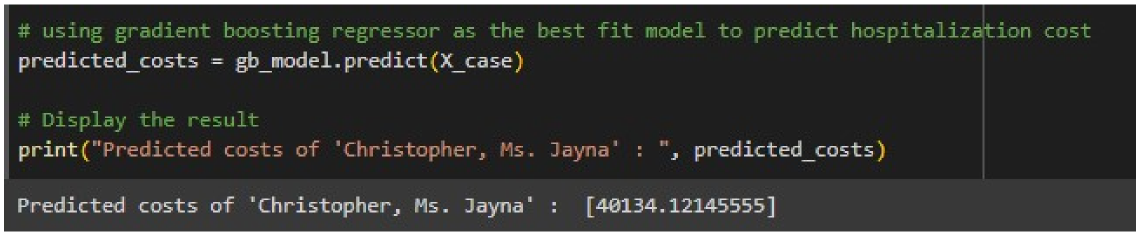


3) Case scenario:



```
# Case Scenario of 'Christopher, Ms. Jayna'
data = {
    'Gender_Male': [0],
    'Gender_Female':[1],
    'Age': [35], # Age calculation based on the current year
    'children': [2],
    'BMI': [29.4],  # BMI calculation: weight (kg) / (height (m) ^ 2)
    'HBA1C': [5.8],
    'NumberOfMajorSurgeries': [0],
    'R1011': [1],
    'R1012': [0],
    'R1013':[0],
    'Hospital tier_tier - 1': [1],
    'Hospital tier_tier - 2': [0],
    'Hospital tier_tier - 3': [0],
    'City tier_tier - 1': [1],
    'City tier_tier - 2': [0],
    'City tier_tier - 3': [0],
    'Heart Issues_yes': [0],
    'Any Transplants_yes': [0],
    'Cancer history_Yes': [0],
    'smoker_yes': [1]
    }

# Create DataFrame
X_case = pd.DataFrame(data)
```

4) Find the predicted hospitalization cost using the best models



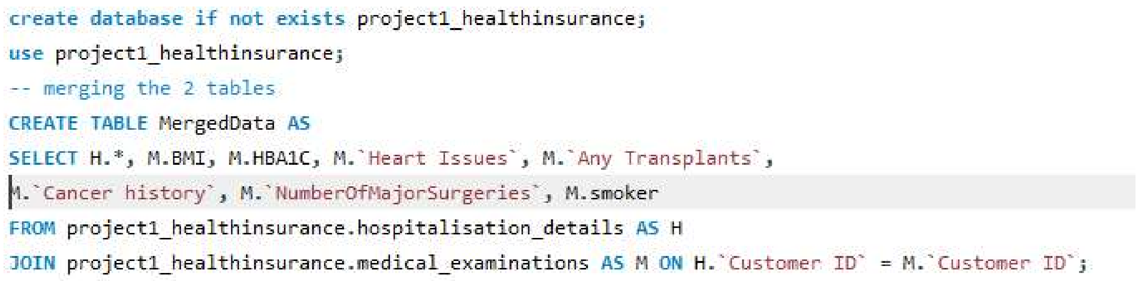
```
# using gradient boosting regressor as the best fit model to predict hospitalization cost
predicted_costs = gb_model.predict(X_case)

# Display the result
print("Predicted costs of 'Christopher, Ms. Jayna' : ", predicted_costs)
```

```
Predicted costs of 'Christopher, Ms. Jayna' :  [40134.12145555]
```

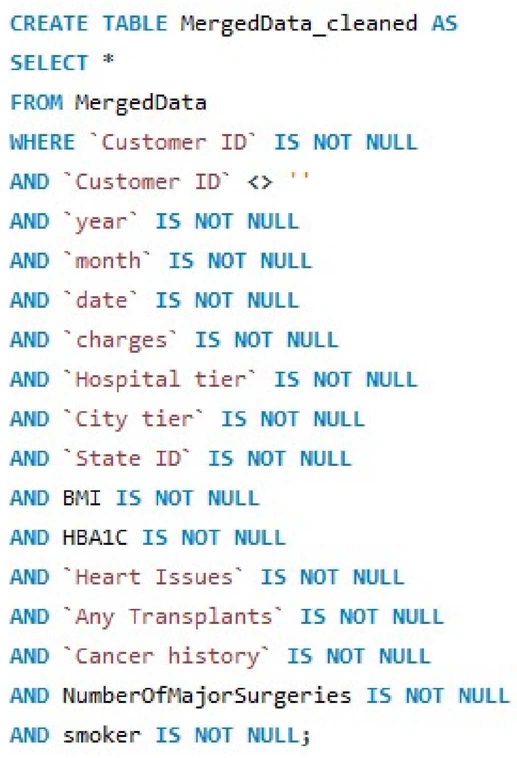
SQL:

1) a. Merge the two tables by first identifying the columns in the data tables that will help you in merging

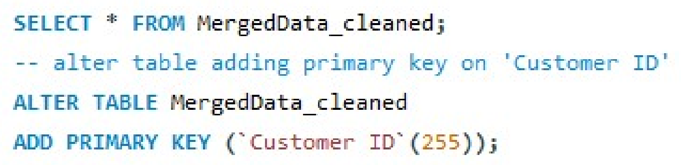
b. In both tables, add a Primary Key constraint for these columns



```
create database if not exists project1_healthinsurance;
use project1_healthinsurance;
-- merging the 2 tables
CREATE TABLE MergedData AS
SELECT H.*, M.BMI, M.HBA1C, M.`Heart Issues`, M.`Any Transplants`,
M.`Cancer history`, M.`NumberOfMajorSurgeries`, M.smoker
FROM project1_healthinsurance.hospitalisation_details AS H
JOIN project1_healthinsurance.medical_examinations AS M ON H.`Customer ID` = M.`Customer ID`;
```

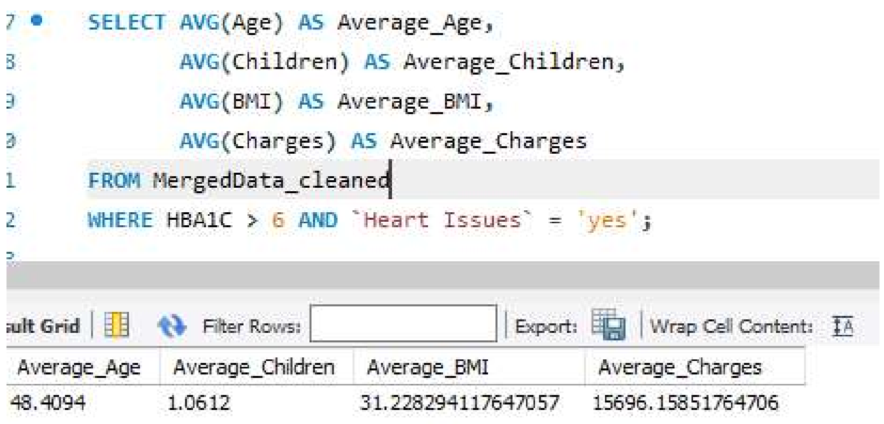


```
CREATE TABLE MergedData_cleaned AS
SELECT *
FROM MergedData
WHERE `Customer ID` IS NOT NULL
AND `Customer ID` <> ''
AND `year` IS NOT NULL
AND `month` IS NOT NULL
AND `date` IS NOT NULL
AND `charges` IS NOT NULL
AND `Hospital tier` IS NOT NULL
AND `City tier` IS NOT NULL
AND `State ID` IS NOT NULL
AND BMI IS NOT NULL
AND HBA1C IS NOT NULL
AND `Heart Issues` IS NOT NULL
AND `Any Transplants` IS NOT NULL
AND `Cancer history` IS NOT NULL
AND NumberOfMajorSurgeries IS NOT NULL
AND smoker IS NOT NULL;
```



```sql
SELECT * FROM MergedData_cleaned;
-- alter table adding primary key on 'Customer ID'
ALTER TABLE MergedData_cleaned
ADD PRIMARY KEY (`Customer ID`(255));
```

2) Retrieve information about people who are diabetic and have heart problems with their average age, the average number of dependent children, average BMI, and average hospitalization costs

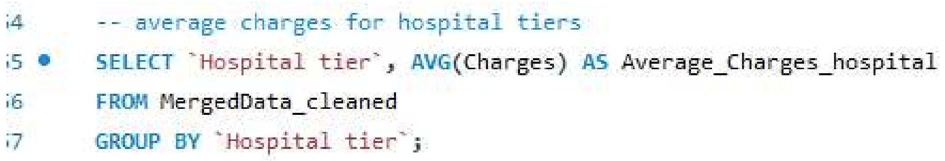


```sql
SELECT AVG(Age) AS Average_Age,
       AVG(Children) AS Average_Children,
       AVG(BMI) AS Average_BMI,
       AVG(Charges) AS Average_Charges
FROM MergedData_cleaned
WHERE HBA1C > 6 AND `Heart Issues` = 'yes';
```

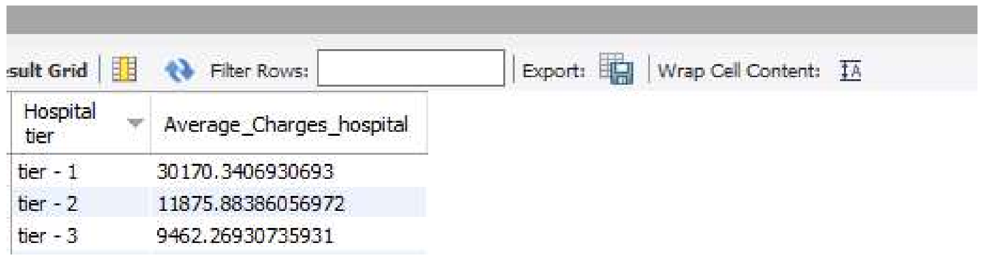
Average_Age	Average_Children	Average_BMI	Average_Charges
48.4094	1.0612	31.228294117647057	15696.15851764706

3) Find the average hospitalization cost for each hospital tier and each city level

Average costs for 3 hospital tiers:

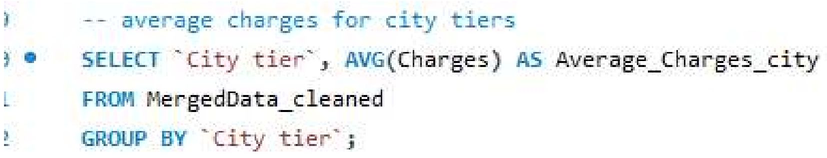


```sql
-- average charges for hospital tiers
SELECT `Hospital tier`, AVG(Charges) AS Average_Charges_hospital
FROM MergedData_cleaned
GROUP BY `Hospital tier`;
```

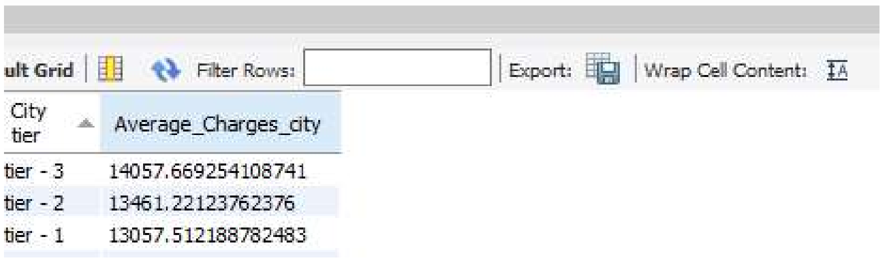


Hospital tier	Average_Charges_hospital
tier - 1	30170.3406930693
tier - 2	11875.88386056972
tier - 3	9462.26930735931

Average costs for 3 City tiers:

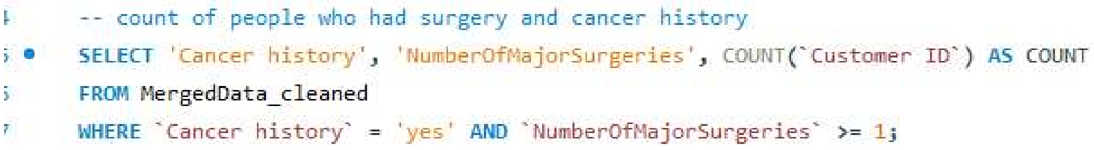


```sql
-- average charges for city tiers
SELECT `City tier`, AVG(Charges) AS Average_Charges_city
FROM MergedData_cleaned
GROUP BY `City tier`;
```

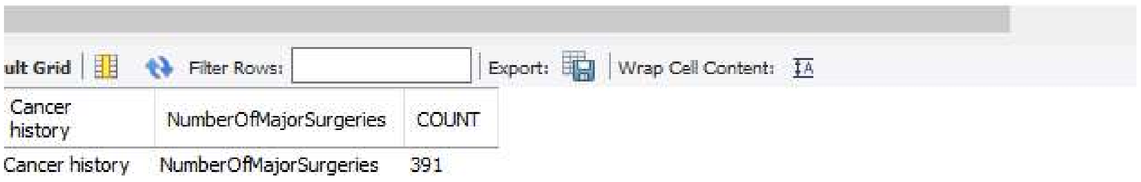


City tier	Average_Charges_city
tier - 3	14057.669254108741
tier - 2	13461.22123762376
tier - 1	13057.512188782483

4) Determine the number of people who have had major surgery with a history of cancer

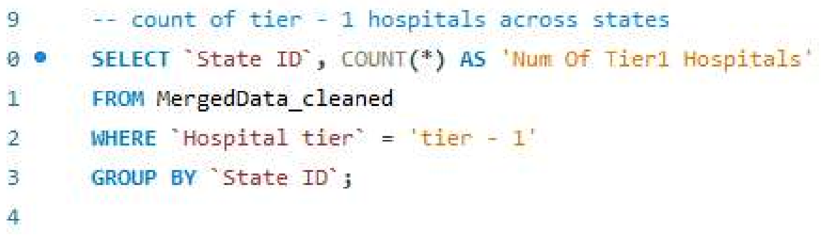


```sql
-- count of people who had surgery and cancer history
SELECT 'Cancer history', 'NumberOfMajorSurgeries', COUNT(`Customer ID`) AS COUNT
FROM MergedData_cleaned
WHERE `Cancer history` = 'yes' AND `NumberOfMajorSurgeries` >= 1;
```

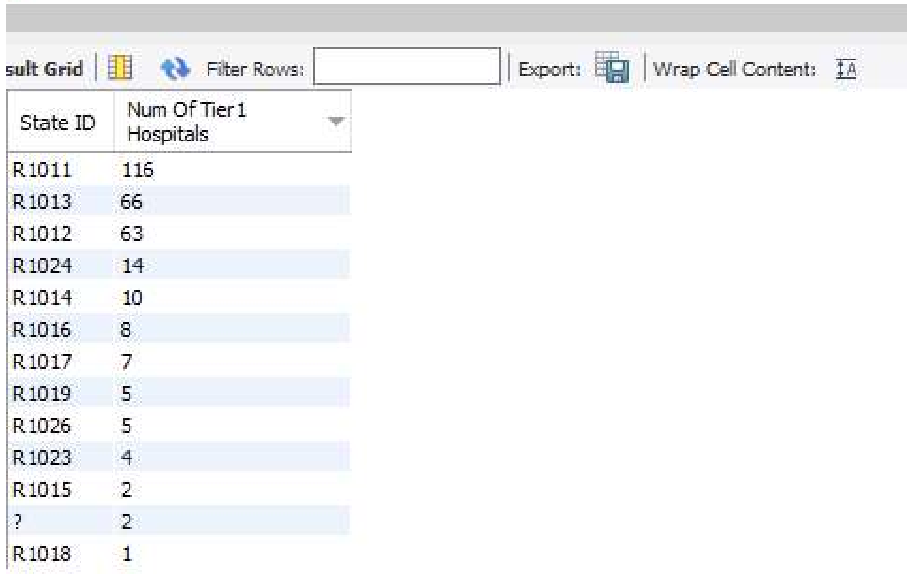


Cancer history	NumberOfMajorSurgeries	COUNT
Cancer history	NumberOfMajorSurgeries	391

5) Determine the number of tier-1 hospitals in each state



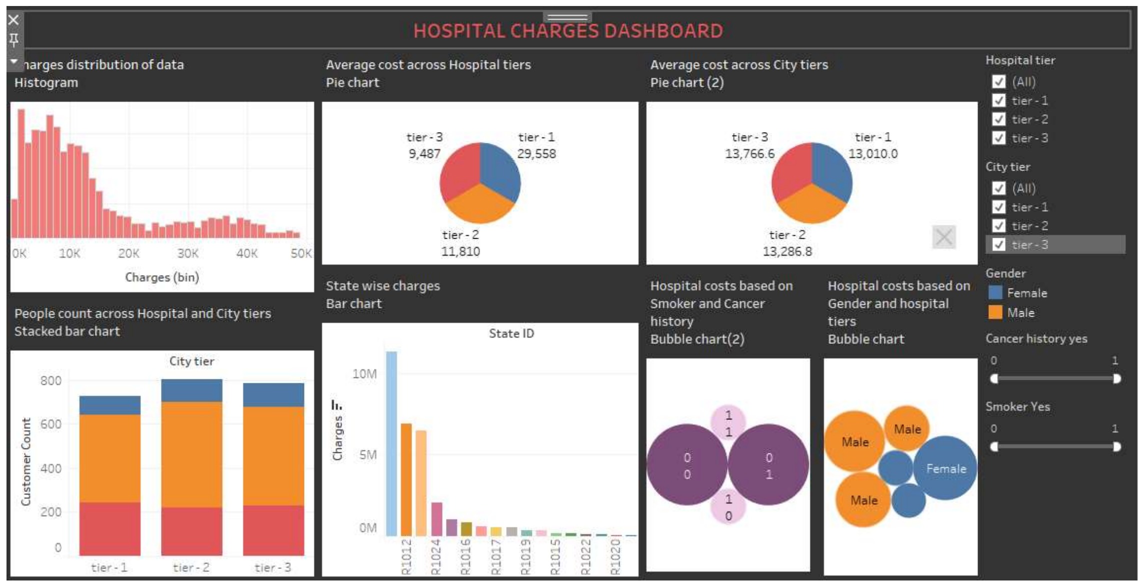
```
-- count of tier - 1 hospitals across states
SELECT `State ID`, COUNT(*) AS 'Num Of Tier1 Hospitals'
FROM MergedData_cleaned
WHERE `Hospital tier` = 'tier - 1'
GROUP BY `State ID`;
```



State ID	Num Of Tier 1 Hospitals
R1011	116
R1013	66
R1012	63
R1024	14
R1014	10
R1016	8
R1017	7
R1019	5
R1026	5
R1023	4
R1015	2
?	2
R1018	1

TABLEAU

1) *Create a dashboard in Tableau by selecting the appropriate chart types and business metrics*



Project submitted by

Phebe Prasanthi