Southampton SOLENT University

Solent University

Data Science Report

Health insurance using Machine Learning

Module Name: Data Science
Module Code: QHO636
Module Leader: Muhammad Parvez Jugon
Assessment Submission Date: 4th October 2023

Student Number:

Table of Contents

Introduction

Data-driven decision making is nothing new in the insurance industry. However, big data and AI are revolutionising the sector. AI has already been very successful in different stages of health sector including drug development, detecting lung cancer or stroke based on CT scans. The exponential growth of data has resulted in modernization and innovation in the health insurance industry (Machine Learning on Insurance Premium Prediction, 2023). This indicates that Machine learning will play a key role in dealing with massive volumes of data, automating, and speeding up the implementation process, and outperforming technical approaches. This also will provide the industry with a recursive approach to predictive modelling, enabling for substantial advancements in policy enrolment and claims settlement methods.

In this project, I will analyse and investigate a dataset for medical costs in Health Insurance to gain valuable insights and find solutions to cost, using a supervised learning. I will also evaluate how health insurance's cost could be influenced by several other things. Furthermore, I will be considering numerous criteria that influence how much an insurer charges for health insurance; Smoking for instance, is one lifestyle decision that could raise your monthly premium. Many providers offer lower health insurance premiums if you don’t smoke. Another is BMI, you may be offered a coverage at the standard premium if your less than 30, unless you have any linked health issues. Others includes Age, postcode. Who is added to your health insurance plan etc?

Research Question

How will machine learning assist to improve health care insurance?
How will algorithm predict outputs more accurate?
How to Identify relationships between variables in big dataset.

Aim

The aim of this research is to find a pattern that analyse the cost of life insurance more accurately. At the end of the project my aim is to provide a better model that predict a more accurate charges for life insurance

Objective

I will be exploring the supervised learning algorism to investigate the business model, data understanding, data preparation, modelling, evaluations, and deployment, that will reflect the relevant insurance specifications. (Azar, Ban and Mansour, 2016) The ability to foresee outcomes with more accuracy is the essence of an expert's skill.
Similarly, (Wang et al., 2016) Suggested in his project that fascinating applications, such as recommendations and forecast performance analysis, can benefit from mining relationships.

Methods and tools used.

Data is a commodity, but it’s worth is subject to debate until it has been processed. IBM (2018) defines data science as a multidisciplinary field whose purpose is to extract value from data in all its forms. The purpose of these techniques is to minimize the large tables and provide a simplified presentation, that justifies claims of the characteristics of the data. To achieve this, I will be using some on the useful analytical and visual tools in this project, they include Google Colad as my integrated development environment (IDE), Python as my programmer, and my dataset will be gathered from Kaggle.

This project will also explore the Crisp Methodology, that will help me structure the approach of the project into dynamic phases. The phases were iterated into Data Collection, Data Cleansing, Data description, Modelling, and algorithms, Data evaluations. To proceed I used quantitative analysis to collect, evaluate and measure the patterns of my dataset. I will also utilising the multivariate analysis technique to figure out multiple factors that affect insurance premium.

Collecting dataset

To collect the suitable data, I will be considering the quantitative analysis, this approach will focus more on a set of numerical data with statistical significance; utilizing the numerical values to quantify the attitudes, behaviours, patterns, and other characteristics that establish or disprove previous claims. Similar, it will help to find the adequate test for checking for errors in my decisions.

In Quantitative analysis there are several primary ways to collect data, but I will be using a third party, which is already generated set in Kaggle.

Data cleaning

First, I imported my file into colab and then import all the required libraries and then read my file into a csv file.

```python
from ast import increment_lineno
import numpy as np
import pandas as pd
import scipy.stats as stats
import seaborn as sns
%matplotlib inline
import statsmodels.api as sm
from sklearn.preprocessing import LabelEncoder
import copy

df=pd.read_csv("Health_insurance.csv")
df
```

(Tableau, 2023) Affirmed that when it comes to data for most developers, your insights and analyses are only as good as the data you use. In essence, junk data in equals rubbish analysis out. Data cleaning, also known as data cleansing and data scrubbing, is a critical step for fostering a culture of quality data decision-making.

I will be checking and removing incorrect, corrupt, incorrect formatted, and duplicated data from my dataset. To do this I will first check if there a missing value with count function.

```python
df.isna().apply(pd.value_counts)
```

	age	sex	bmi	children	smoker	region	charges
False	1338	1338	1338	1338	1338	1338	1338

As shown above there is no null value on my set.

Other ways I can look for null value is df.isnull().any().any(),df.isnull().any()
I also checked by column with the function df.isnull().sum()

```
[38] df.isnull().any()
```

```
age         False
sex         False
bmi         False
children    False
smoker      False
region      False
charges     False
dtype: bool
```

```
[39] df.isnull().sum()
```

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

Exploratory data analysis

Data Summary and overview

I used **df** as a variable that will contian my data frame all through this report

	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

Above is a comprehensive collection of my dataset, it contains 1338 rows and 7 columns. Though, this may not present the entire set we can also filter to any size

```
df.head(7)
```

	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
5	31	female	25.740	0	no	southeast	3756.62160
6	46	female	33.440	1	no	southeast	8240.58960

```
df.tail(7)
```

	age	sex	bmi	children	smoker	region	charges
1331	23	female	33.40	0	no	southwest	10795.93733
1332	52	female	44.70	3	no	southwest	11411.68500
1333	50	male	30.97	3	no	northwest	10600.54830
1334	18	female	31.92	0	no	northeast	2205.98080

Alt+A

First, I will be checking the data type of my dataset using the unique function.

```
df.dtypes
```

```
age           int64
sex          object
bmi         float64
children      int64
smoker       object
region       object
charges     float64
dtype: object
```

As shown above I 1338 instances of the data with 7 attributes are present. 3 objects ,2 float, and 2 integers.

Another way to understand the type of data is by checking the datatype.

Descriptive Statistic

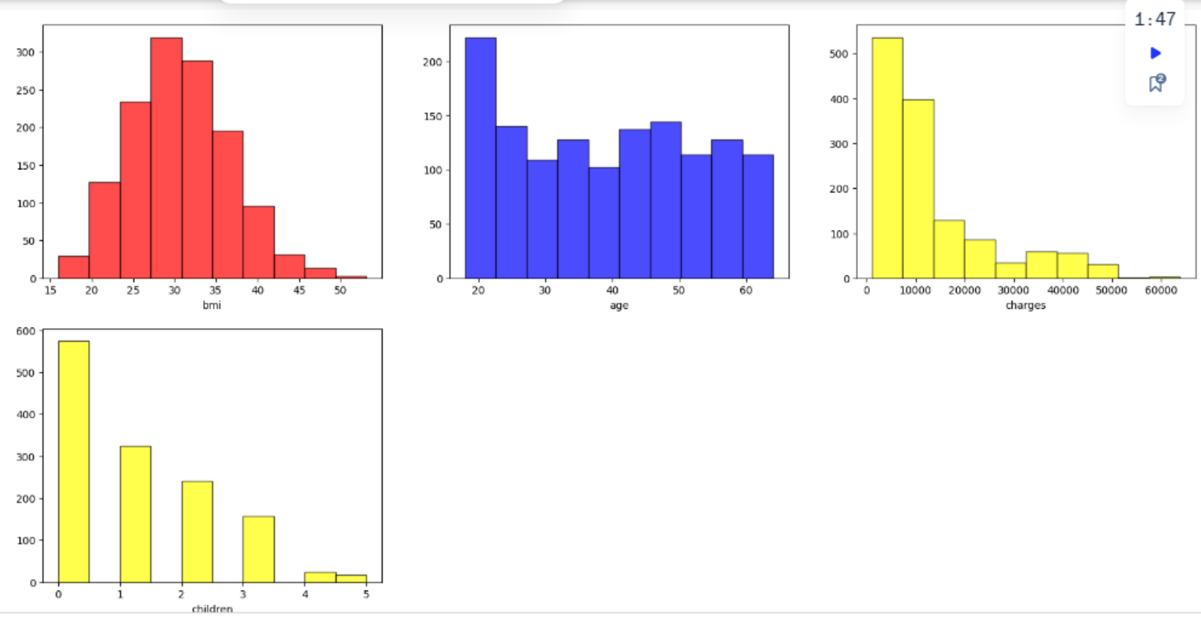
This will help me determine how to manipulate my dataset to get the desired results.
In the section, I will be using multivariate graphic and cross-tabulation or statistics.
First, let me see the basic statistics of my dataset.

```
[195] df.describe().transpose()
```

	count	mean	std	min	25%	50%	75%	max
age	1338.0	39.207025	14.049960	18.0000	27.00000	39.000	51.000000	64.00000
bmi	1338.0	30.663397	6.098187	15.9600	26.29625	30.400	34.693750	53.13000
children	1338.0	1.094918	1.205493	0.0000	0.00000	1.000	2.000000	5.00000
charges	1338.0	13270.422265	12110.011237	1121.8739	4740.28715	9382.033	16639.912515	63770.42801

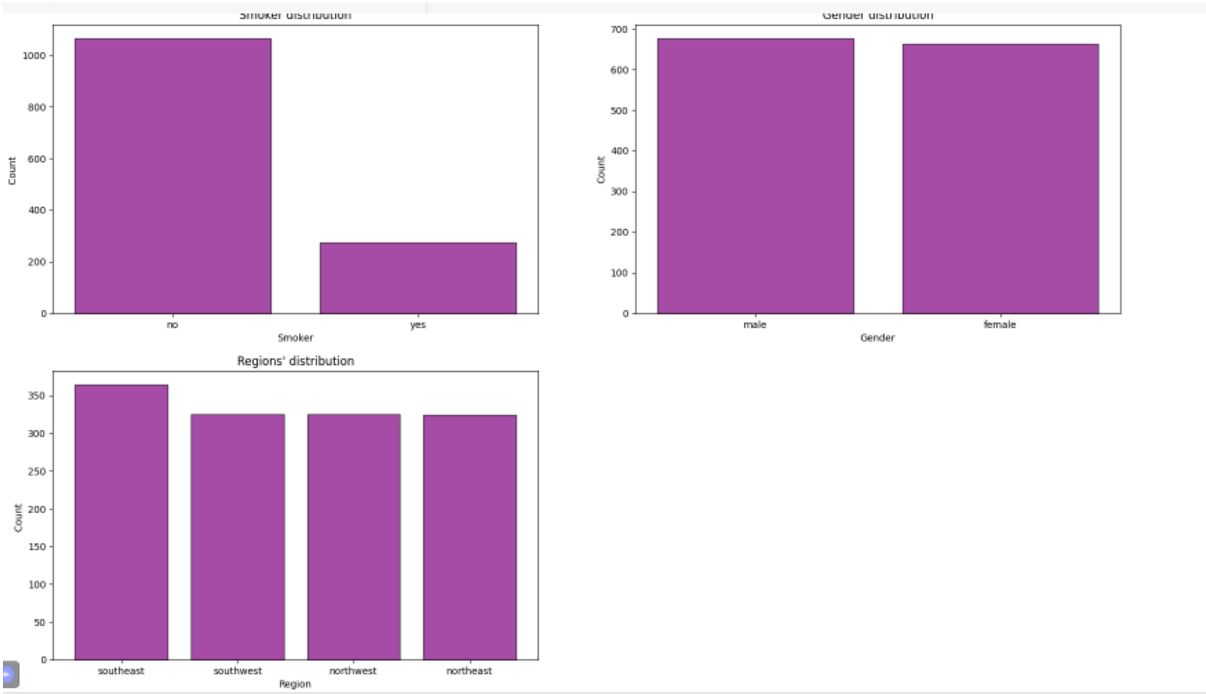
I transpose my dataset into diagonal to find the important statistics of each object. These will help me summarize the central tendencies and variabilities in each my numerical variables. For example, the "charges" variable with a high variation, indicates that the charges can vary from one instant to the other.
Looking at the age column, the data appears to be typical of the genuine age distribution of the adult population.
The mean of children suggests that 75% have 2 or less children.

Data Visualization



The chart above shows my numerical data:

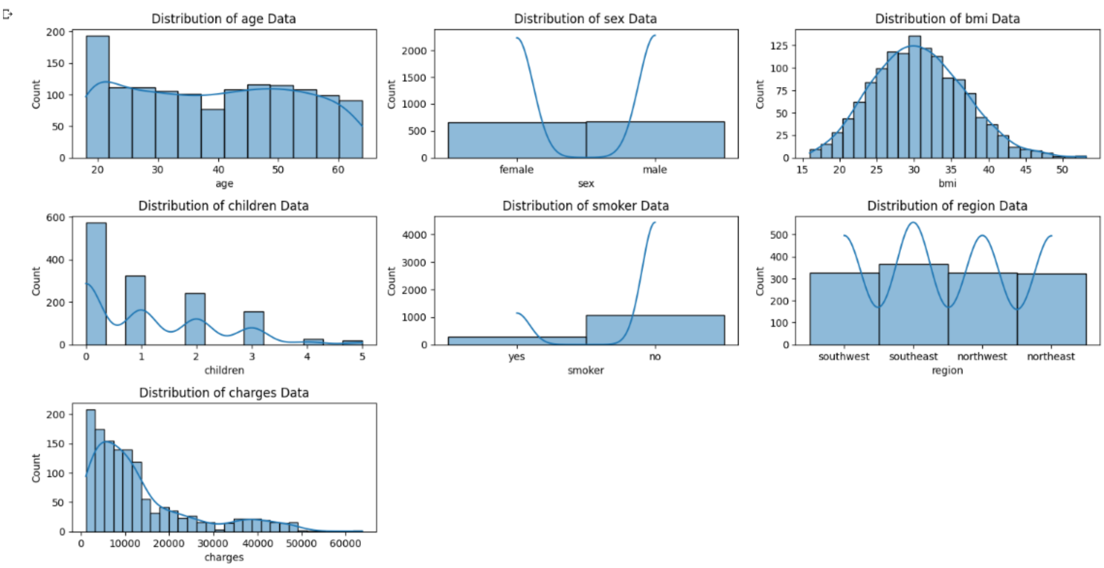
- BMI appears to be quite evenly distributed.
- Age appears to be spread rather uniformly.
- Charges are also extremely skewed, as was already evident in the previous description stage.
- Most instances have less than 2 children and very few have 4 or 5 children.



The chart above shows my categorical data:

- In the data, there are many more nonsmokers than smokers.
- Instances are evenly distributed throughout all regions.
- Gender is also evenly distributed.

Next, I will be plotting my data to see if my data is symmetric or not. This is a helpful visualization technique for understanding the data distribution of each feature in my targeted variable "Charges ".



As shown above, some of my data is skewed, however I will measure the skewness to see how I can manipulate the data to reduce the skewness.
The skew of "bmi" is very low, as can be seen in above diagram; the skew of "age" is equally distributed and barely noticeable; and the skew of "charges" is positively distributed.

```
[47] # Measure of skewness of 'bmi', 'age' and 'charges' columns
     Skewness = pd.DataFrame({'Skewness' : [stats.skew(df.bmi),stats.skew(df.age),stats.skew(df.charges)]},
                             index=['bmi','age','charges'])  # Measure the skeweness of the required columns
     Skewness
```

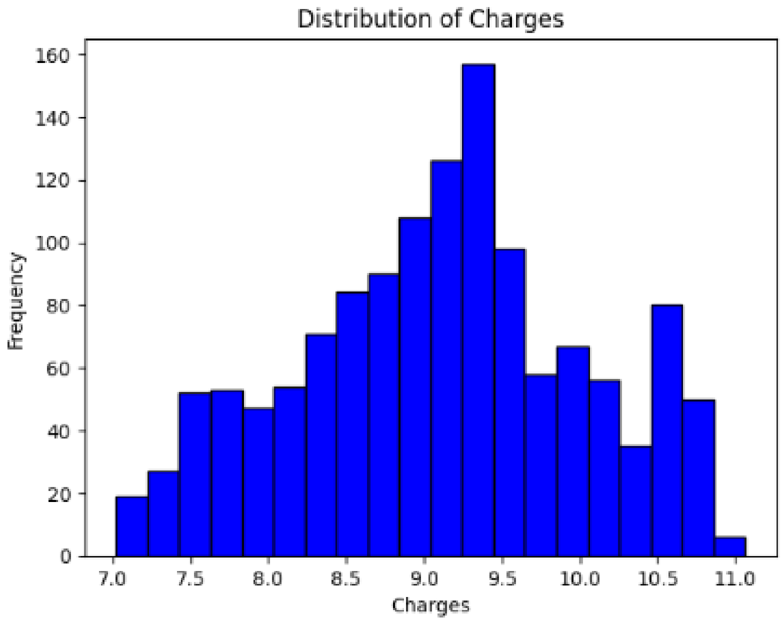
	Skewness
bmi	0.283729
age	0.055610
charges	1.514180

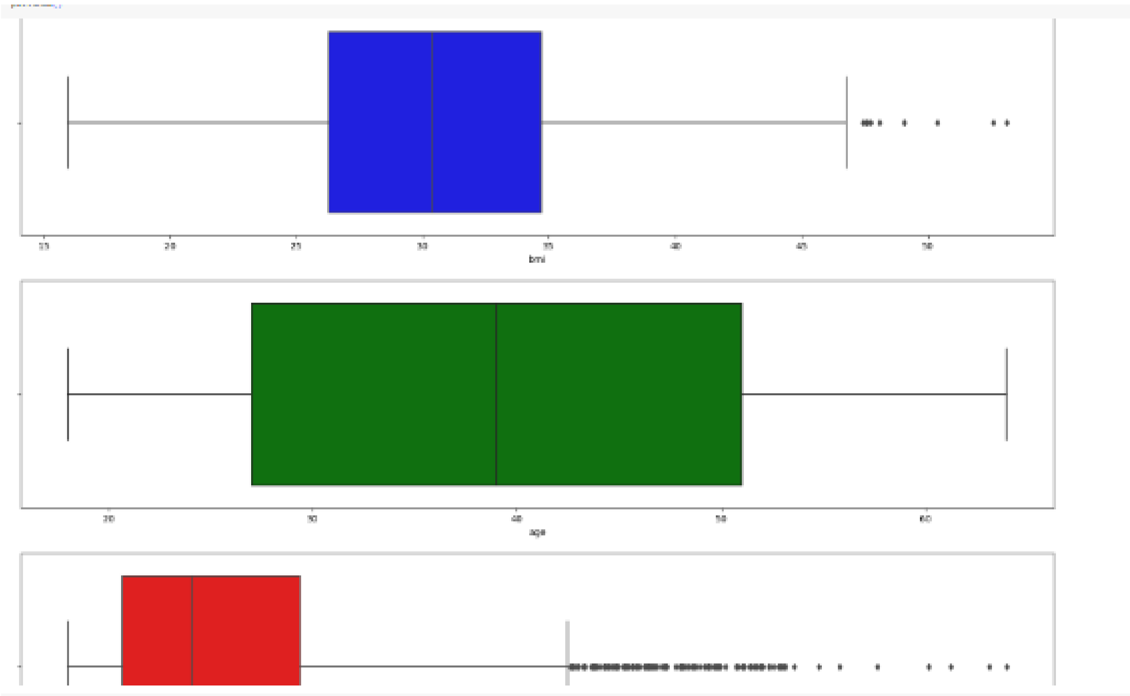
I checked how each variable is close to zero(positive/negative)

- As seen above the bmi at 0.2 is moderate skew.
- Age at 0.05 indicates a nearly symmetric data distribution with a very mild tendency, either to the right or left, but not strong enough to be considered a highly skewed distribution.
- Charges at 1.5 is highly asymmetric and could impact the result.

I normalise the charges to be normally distributed.

```
# Show the histogram
plt.show()
```





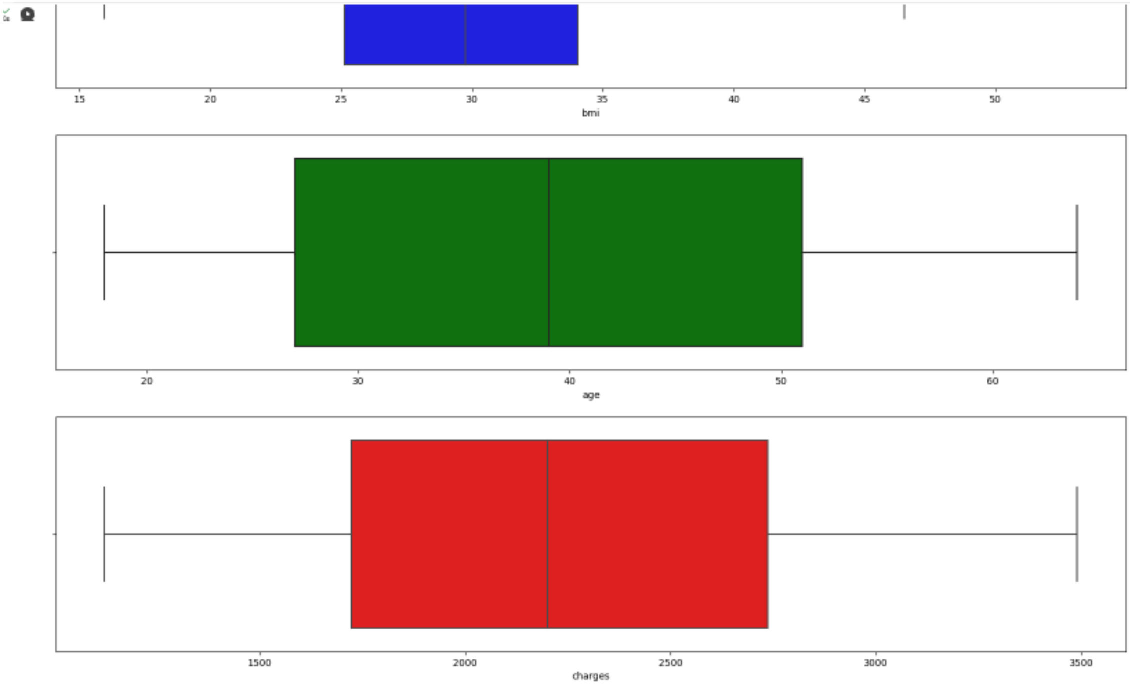
No outliers are detected in the 'age' variable.
The variable "bmi" reveals the presence of a few extreme values.

The "charges" variable is very skewed and contains several extreme values.

Handling outliers

At this stage, I will be removing the outliers, as this may affect my training modelling

Removing outlier



As seen above, I applied a condition to remove data that significantly deviate from the mean of my dataset.

```
df1=df[df['bmi']<45]
```

```
[ ] df1['bmi'].mean()
```

```
30.407143399089527
```

```
[ ]
```

```
[ ] df1=df[df['charges']<3500]
```

```
[ ] df1['charges'].mean()
```

```
2261.388053030303
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                charges   R-squared:                       0.117
Model:                            OLS   Adj. R-squared:                  0.116
Method:                 Least Squares   F-statistic:                     88.60
Date:                Sun, 01 Oct 2023   Prob (F-statistic):           7.39e-37
Time:                        09:07:20   Log-Likelihood:                -14394.
No. Observations:                1338   AIC:                         2.879e+04
Df Residuals:                    1335   BIC:                         2.881e+04
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const      -6424.8046   1744.091     -3.684      0.000   -9846.262   -3003.347
age          241.9308     22.298     10.850      0.000     198.187     285.674
bmi          332.9651     51.374      6.481      0.000     232.182     433.748
==============================================================================
Omnibus:                      321.874   Durbin-Watson:                   2.010
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              592.574
Skew:                           1.511   Prob(JB):                    2.11e-129
Kurtosis:                       4.223   Cond. No.                         287.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

This Ordinary Least Squares regression (OLS) regression result offers details about the model's effectiveness, the importance of specific variables, and other statistical diagnostics, to sum up. It indicates that the model is statistically significant overall and that "age" and "bmi" are strong predictors of "charges."

Hypothesis Testing

I also tried to perform a Chi-square test to check if smoking habits are different for people.

```python
# Chi_square test to check if smoking habits are different for different genders
sm = "Gender has no effect on smoking habits"   # Stating the Null Hypothesis
na = "Gender has an effect on smoking habits"   # Stating the Alternate Hypothesis

crosstab = pd.crosstab(df['sex'],df['smoker'])  # Contingency table of sex and smoker attributes

chi, p_value, dof, expected =  stats.chi2_contingency(crosstab)

if p_value < 0.05:  # Setting our significance level at 5%
    print(f'{sm} as the p_value ({p_value.round(3)}) < 0.05')
else:
    print(f'{na} as the p_value ({p_value.round(3)}) > 0.05')
crosstab
```

```
Gender has no effect on smoking habits as the p_value (0.007) < 0.05
```

smoker	0	1
sex		
0	115	547
1	159	517

```
# Chi_square test to check if smoking habits are different for people of different regions
rs = "Region has no effect on smoking habits"   # Stating the Null Hypothesis
na = "Region has an effect on smoking habits"   # Stating the Alternate Hypothesis

crosstab = pd.crosstab(df['smoker'], df['region'])  # Contingency table of sex and smoker attributes

chi, p_value, dof, expected =  stats.chi2_contingency(crosstab)

if p_value < 0.05:  # Setting our significance level at 5%
    print(f'{rs} as the p_value ({p_value.round(3)}) < 0.05')
else:
    print(f'{na} as the p_value ({p_value.round(3)}) > 0.05')
crosstab
```

```
Region has an effect on smoking habits as the p_value (0.062) > 0.05
```

region	1	2	3	4
smoker				
0	58	58	91	67
1	267	267	273	257

Hypothesis of smoker vs none smokers

```
nonsmokers = df[df['smoker'] == 1]
[662] charge_smokers = smokers['charges']
      charge_nonsmokers = nonsmokers['charges']

      print(f'Number of smokers: {smokers.shape[0]}')
      print(f'Variance in charges of smokers: {np.var(charge_smokers)}')
      print(f'Number of non - smokers: {nonsmokers.shape[0]}')
      print(f'Variance in charges of non - smokers: {np.var(charge_nonsmokers)}')
```

```
Number of smokers: 274
Variance in charges of smokers: 132721153.13625307
Number of non - smokers: 1064
Variance in charges of non - smokers: 35891656.00316426
```

```
[663] from scipy.stats import ttest_ind

      t_statistic, p_value = ttest_ind(charge_smokers, charge_nonsmokers, equal_var=False)
      print(f't_statistic: {t_statistic}\np_value: {p_value}')
```

```
t_statistic: 32.751887766341824
p_value: 5.88946444671698e-103
```

```
[664] print ("two-sample t-test p-value=", p_value)
```

```
two-sample t-test p-value= 5.88946444671698e-103
```

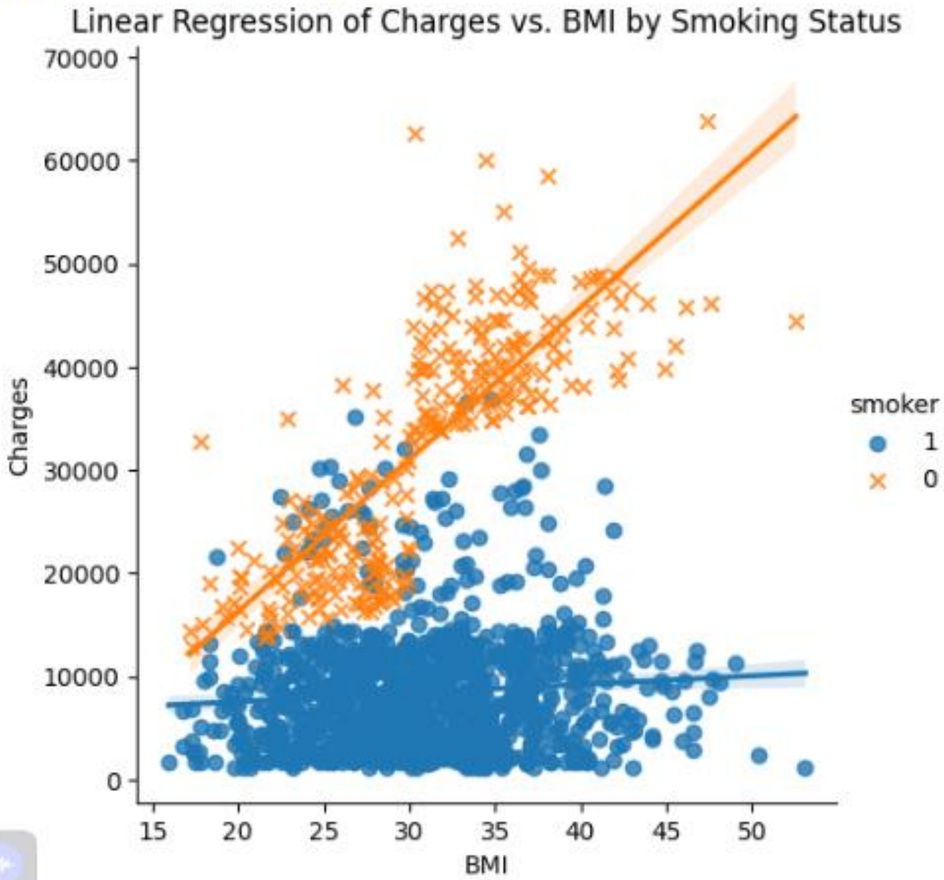
```
p_value > 0.05
```

```
False
```

I reject the Null Hypothesis and conclude that, at the 5% level of significance, the mean charges of smokers and non-smokers are not comparable.
As a result, costs for smokers differ dramatically from those for nonsmokers.

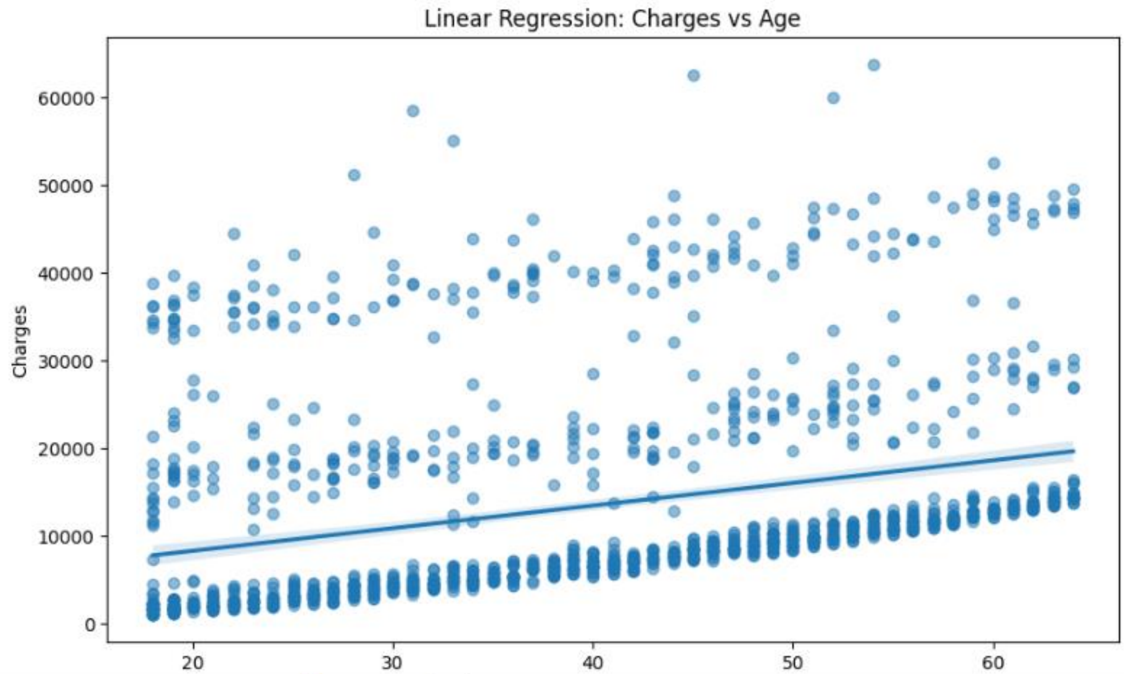
Furthermore, I checked if there is a link between BMI and medical costs for both smokers and non-smokers.

<Figure size 1000x600 with 0 Axes>



You can conclude the following by studying the scatter plot and the linear regression lines:

There is a link between BMI and medical costs for both smokers and non-smokers. Medical costs typically rise along with BMI.
The pace at which fees rise with BMI is indicated by the slope of the regression lines. Smokers may experience a greater average effect of BMI on medical costs than non-smokers due to the orange line's apparent steeper slope than the blue line.
There is some fluctuation in charges for a given BMI, as seen by the scatter plot and regression lines.



As indicated in the above diagram, the older the more likelihood of charges going up

The correlation map charges reveals the relationship of each variable to our targeted variable: Charges have a weak positive association with the insured's age and BMI, and a large positive link with smoking habit.

Data Modelling

To start, since machine learning does not easily manipulate categorical variables, I will convert all my categorical to numerical.

```
df['sex']=df['sex'].map({'female':0,'male':1})
```

```
[52] df['smoker']=df['smoker'].map({'yes':0,'no':1})
```

```
[53] df['region']=df['region'].map({'northwest':1,'southwest':2,'southeast':3,'northeast':4})
```

```
[54] df
```

	age	sex	bmi	children	smoker	region	charges
0	19	0	27.900	0	0	2	16884.92400
1	18	1	33.770	1	1	3	1725.55230
2	28	1	33.000	3	1	3	4449.46200
3	33	1	22.705	0	1	1	21984.47061
4	32	1	28.880	0	1	1	3866.85520
...	...	...	...	...	...	...	...
1333	50	1	30.970	3	1	1	10600.54830
1334	18	0	31.920	0	1	4	2205.98080
1335	18	0	36.850	0	1	3	1629.83350
1336	21	0	25.800	0	1	2	2007.94500
1337	61	0	29.070	0	0	1	29141.36030

Next, I will be storing my independent variable into matrix X and response(target) in Y. This is to allow me to train my model into a separate folder.
My independent variable will be stored in metric X.
My dependent variable will be stored in metric Y.

```python
X = df.drop(['charges'],axis=1)
X
```

	age	sex	bmi	children	smoker	region
0	19	0	27.900	0	0	2
1	18	1	33.770	1	1	3
2	28	1	33.000	3	1	3
3	33	1	22.705	0	1	1
4	32	1	28.880	0	1	1
...	...	...	...	...	...	...
1333	50	1	30.970	3	1	1
1334	18	0	31.920	0	1	4
1335	18	0	36.850	0	1	3
1336	21	0	25.800	0	1	2
1337	61	0	29.070	0	0	1

1338 rows × 6 columns

The y axis is also printed below.

```python
Y = df['charges']
Y
```

```
0       16884.92400
1        1725.55230
2        4449.46200
3       21984.47061
4        3866.85520
           ...
1333    10600.54830
1334     2205.98080
1335     1629.83350
1336     2007.94500
1337    29141.36030
Name: charges, Length: 1338, dtype: float64
```

Next, I will be training my model to evaluate the performance of the models and test this model. I spited 20% of my data for this test and used RANDOM_STATE to keep my sample constant.

```
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.2,random_state=42)

Y_train
```

```
560      9193.83850
1285     8534.67180
1142    27117.99378
969      8596.82780
486     12475.35130
           ...
1095     4561.18850
1130     8582.30230
1294    11931.12525
860     46113.51100
1126    10214.63600
Name: charges, Length: 1070, dtype: float64
```

Linear Regression Training model

```
[223] lr= LinearRegression()
      lr.fit(X_train,Y_train)
```

```
▾ LinearRegression
LinearRegression()
```

```
[224] y_pred =lr.predict(X_test)
```

```
[225] df=({'Actual':Y_test,'Lr':y_pred})
```

```
[226] print (lr.coef_)
```

```
[ 2.57334934e+02 -5.45160552e+00  3.26891659e+02  4.30247314e+02
 -2.36393995e+04  1.29887884e+02]
```

```
[227] print (lr.intercept_)
```

```
11205.606958531025
```

```
r2_score(Y_test,y_pred)
```

```
0.7809227350368882
```

- I trained linear regression model and stored it in lr variable this will be used to make predictions on fresh, unforeseen data and to examine the correlations between characteristics and the target variable. Typically, the predict technique would be used to make predictions using this trained model.
- I then created two columns to help me compare the actual and prediction.

- The coefficient reveals the relationship of each variable to the target. They reveal which characteristics have a bigger influence on the predictions and whether that influence is positive or negative) ranging from +1 to -1

- I used the R-score to see the performance of my model (The higher the better

As you can see in the above code, the r-score is 78% accurate. So, I will be Training different model to see the best accurate outcome, I will be training Linear Regression, Support Vector Regression, Gradient Boosting Regression, and Random Forest Regression

```
lr= LinearRegression()
lr.fit(X_train,Y_train)
svm = SVR()
svm.fit(X_train,Y_train)
rf = RandomForestRegressor()
rf.fit(X_train,Y_train)
gr = GradientBoostingRegressor()
gr.fit(X_train,Y_train)
```

```
▾ GradientBoostingRegressor
GradientBoostingRegressor()
```

As seen above I created the instant of each model.

Model Testing

Next, I will be testing my model with the actual model.

```
Y_pred2 = svm.predict(X_test)
[59] Y_pred3 = rf.predict(X_test)
     Y_pred4 =gr.predict(X_test)

     df1=pd.DataFrame({'Actual':Y_test,'Lr':Y_pred1,'svm':Y_pred2,'rf':Y_pred3,'gr':Y_pred4})
```

```
df1
```

	Actual	Lr	svm	rf	gr
764	9095.06825	8755.823138	9548.193152	10011.689040	10731.941685
887	5272.17580	6773.440540	9492.070844	5122.542870	5831.051454
890	29330.98315	36593.412898	9649.028325	28120.715163	28509.596550
1293	9301.89355	9234.618374	9554.934946	13034.087793	9682.279419
259	33750.29180	26653.788754	9419.856808	34641.952641	33531.443511
...	...	...	...	...	...
109	47055.53210	39272.548191	9649.412521	47190.600633	45700.985045
575	12222.89830	11503.167865	9625.661901	13151.871973	12624.789938
535	6067.12675	7450.420908	9503.815696	6278.223854	6922.538173
543	63770.42801	40989.290631	9605.276965	46550.067446	47820.270382
846	9872.70100	12560.006888	9591.327598	9886.510913	10749.368233

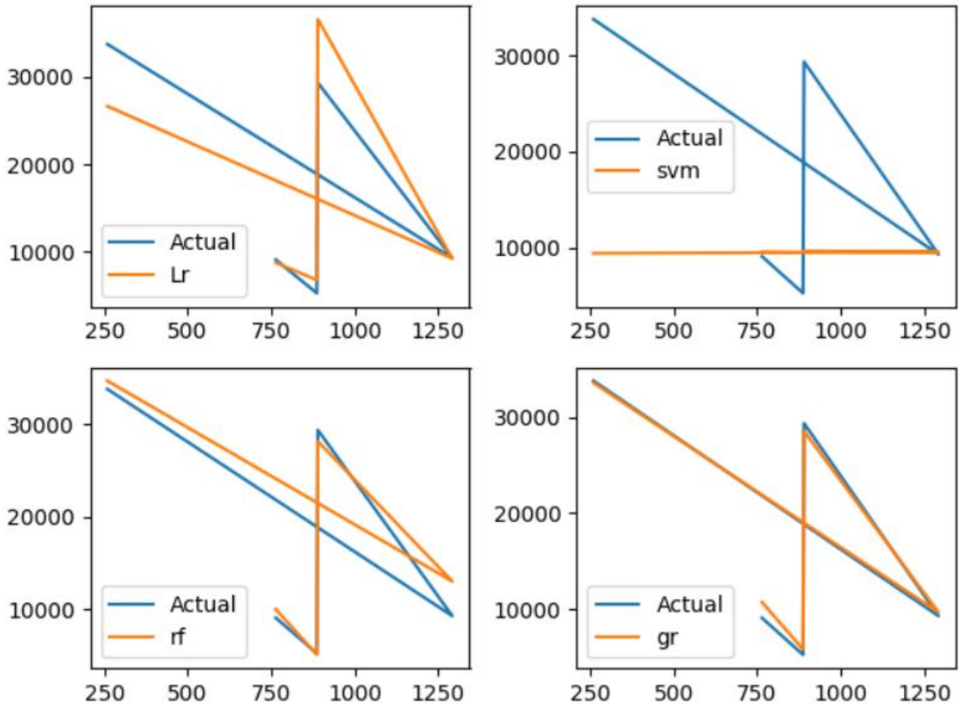
268 rows × 5 columns

As seen above, some of my predictions are very close to the actual value.
Let me but this into visual so we can make more sense of it.

```
plt.plot(df1['Actual'].iloc[0:5],label='Actual')
plt.plot(df1['gr'].iloc[0:5],label='gr')

plt.tight_layout()
plt.legend()
```

```
<matplotlib.legend.Legend at 0x7db62b9cca60>
```



Evaluation and Algorithm

As you can see in the diagram above model three and four is very close to the actual value Model one is not matching the actual value while model two is totally off. So, it means I can further investigate the models to see the best model. To achieve this, I will be using the R Square to determine the coefficients of each model. The higher the R Square the better

```
score1 = metrics.r2_score(Y_test,Y_pred1)
score2 = metrics.r2_score(Y_test,Y_pred2)
score3 = metrics.r2_score(Y_test,Y_pred3)
score4 = metrics.r2_score(Y_test,Y_pred3)
print(score1,score2,score3,score4)
```

```
0.7809227350368882 -0.07228434659803207 0.8587559488076063 0.8587559488076063
```

As seen above both model 3 and 4 is showing more values.

Another way we can check this is by Mean Absolute Error, in this case the lower the result the better

```
s1 =metrics.mean_absolute_error(Y_test,Y_pred1)
s2 =metrics.mean_absolute_error(Y_test,Y_pred2)
s3 =metrics.mean_absolute_error(Y_test,Y_pred3)
s4 =metrics.mean_absolute_error(Y_test,Y_pred4)
print(s1,s2,s3,s4)
```

```
4211.922392445529 8592.17909533713 2553.9356799204616 2394.793833807251
```

As we can see the fourth model has the lowest value, so our best model is Gradient Boosting Regression

Predictions for New Patient

```
data = {  'age':19,
         'sex': 0,'bmi':  27.900,'children':0,'smoker' :  0, 'region':  2
}
df =pd.DataFrame(data,index=[0])
df
```

	age	sex	bmi	children	smoker	region
0	19	0	27.9	0	0	2

```
[65] new_pred = gr.predict(df)
     print(new_pred)
```

```
[17921.72785527]
```

```
[66] gr =GradientBoostingRegressor()
     gr.fit(X,Y)
```

I now Used the best model to make new predictions.

```
[66] gr =GradientBoostingRegressor()
     gr.fit(X,Y)
```

```
▾ GradientBoostingRegressor
GradientBoostingRegressor()
```

```
[67] joblib.dump(gr,'model_joblib_gr')
```

```
['model_joblib_gr']
```

```
[68] model=joblib.load('model_joblib_gr')
```

```
[70] model.predict(df)
```

```
array([18456.13263041])
```

Finally, I trained my model on the entire dataset as seen above.

Conclusion

According to (Aakash Ganju et al., 2021) The healthcare sector encourages people to take more responsibility for their own health; but providers have been slow to equip patients with the tools they need to become active stakeholders in their health journeys. The rising adoption of mobile and digital platforms presents a chance to build highly targeted, engaging, and quantifiable health communication treatments that will result in more activated and engaged patients. This research I used some of the basic machine learning stools that could help to general a more accurate health insurance cover. There is other method that can be used to achieve the results.

https://colab.research.google.com/drive/1NsMmW7W8WC_QRMq6JKZ6wAQUfIPQv_b7?usp=sharing

Reference

Aakash Ganju, Sonia Rebecca Menezes, Schenelle Dlima and Santosh Shevade (2021). Machine learning-driven recommender systems to improve engagement with health content in a low-resource setting: Poster. [online] doi:https://doi.org/10.1145/3460112.3471976.

Azar, Y., Ban, A. and Mansour, Y. (2016). When Should an Expert Make a Prediction? *arXiv (Cornell University)*. [online] doi:https://doi.org/10.1145/2940716.2940729.

Fox, I., Ang, L., Jaiswal, M., Rodica Pop-Busui and Wiens, J. (2018). Deep Multi-Output Forecasting. *arXiv (Cornell University)*. [online] doi:https://doi.org/10.1145/3219819.3220102.

Amir Shareghi Najar and Davoudi, A. (2009). A new model for health care e-insurance using credit points and Service Oriented Architecture (SOA). [online] doi:https://doi.org/10.1145/1806338.1806441.

Goyal, A., Anubhav Elhence, Vinay Chamola and Biplab Sikdar (2021). A Blockchain and Machine Learning based Framework for Efficient Health Insurance Management. [online] doi:https://doi.org/10.1145/3485730.3493685.

Mehdi Bagherzadeh and Khatchadourian, R. (2019). Going big: a large-scale study on what big data developers ask. *CUNY Academic Works (City University of New York)*. [online] doi:https://doi.org/10.1145/3338906.3338939.

Machine learning basics with IBM data science experience. (2023). *Acm.org*. [online] doi:https://doi.org/10.5555/3172795.3172853.

Machine Learning on Insurance Premium Prediction. (2023). *Acm.org*. [online] doi:https://doi.org/10.1145/3605423.3605450.

Yi, B., Lin, W.T., Li, W., Gao, X., Bing Bing Zhou and Tang, J. (2023). Process Quality Prediction Algorithm of Multi output Workshop Based on ATT-CNN-TCN. [online] doi:https://doi.org/10.1145/3589572.3589590.

Appendix