CAPSTONE PROJECT

FINAL REPORT

SHANKAR SUNDAR

PGPDSBA – ONLINE APRIL_B-2021

15th May, 2022

Table of Contents

LIST OF FIGURES:

LIST OF TABLES:

PROBLEM STATEMENT:

We all know that Health care is very important domain in the market. It is directly linked with the life of the individual; hence we have to be always be proactive in this particular domain. Money plays a major role in this domain, because sometime treatment becomes super costly and if any individual is not covered under the insurance then it will become a pretty tough financial situation for that individual. The companies in the medical insurance also want to reduce their risk by optimizing the insurance cost, because we all know a healthy body is in the hand of the individual only. If individual eat healthy and do proper exercise the chance of getting ill is drastically reduced.

Goal & Objective: The objective of this exercise is to build a model, using data that provide the optimum insurance cost for an individual. You have to use the health and habit related parameters for the estimated cost of insurance.

1. INTRODUCTION

- ### BRIEF INTRODUCTION ABOUT THE PROBLEM STATEMENT AND THE NEED OF SOLVING IT.

A. Defining Problem Statement

As we all know Health Care is an important Domain in the Market. Since it is directly linked with the life of an individual the companies have to be proactive in this domain. As there is a lot of money involved, the insurance company wants to reduce their risks by optimizing the insurance costs. By taking the health and habit related parameters of the individual, we have to build a model which will estimate the cost of insurance for the individual.

B. Need of the Study/Project

The **Sustainable Development Goals** (SDGs) reaffirm a global commitment to achieve **Universal Health Coverage** (UHC) by 2030. This means that all people and communities, everywhere in the world, should have access to the high-quality health services they need – promotive, preventive, curative, rehabilitative, or palliative – without facing financial hardship. A few initiatives have been taken across the world in order make the Universal Health Coverage successful.

The Need of this study is that by analysing the individual we can have a better knowledge of their health which can helps the company in achieving the insurance cost and the risk involved. This helps the company to bring up new policies which can benefit both the company and the individual involved.

C. Understanding business/social opportunity

There are around 8 Billion (2023) of people around the world in which only half of the population is covered with insurance. By optimizing the insurance cost and with less risk involved, we can cover the entire population under insurance which can be profitable to both the company and the individual. This will have a huge economic impact as we are covering the individuals at the time of medical emergency.

2. EDA AND BUSINESS IMPLICATION

- ### UNIVARIATE / BIVARIATE / MULTIVARIATE ANALYSIS TO UNDERSTAND RELATIONSHIP B/W VARIABLES. HOW YOUR ANALYSIS IS IMPACTING THE BUSINESS?

A. UNDERSTANDING DATA.

The company has collected data at the time of drafting insurance policies to the individual by collecting their entire daily and health habits like age, occupation, cholesterol level, bmi etc. We can also notice the years of insurance of the individual with the company.

B. VISUAL INSPECTION AND UNDERSTANDING OF DATA.

The head of the data is shown below.

	applicant_id	years_of_insurance_with_us	regular_checkup_lasy_year	adventure_sports	Occupation	visited_doctor_last_1_year	cholesterol_level	daily_avg_st
0	5000	3	1	1	Salried	2	125 to 150	4
1	5001	0	0	0	Student	4	150 to 175	6
2	5002	1	0	0	Business	4	200 to 225	4
3	5003	7	4	0	Business	2	175 to 200	6
4	5004	3	1	0	Student	2	150 to 175	4

Figure 1: Head of the data

The tail of the data is shown below.

	applicant_id	years_of_insurance_with_us	regular_checkup_lasy_year	adventure_sports	Occupation	visited_doctor_last_1_year	cholesterol_level	daily_av
24995	29995	3	0	0	Salried	4	225 to 250	
24996	29996	6	0	0	Business	4	200 to 225	
24997	29997	7	0	1	Student	2	150 to 175	
24998	29998	1	0	0	Salried	2	225 to 250	
24999	29999	8	2	0	Business	4	150 to 175	

Figure 2: Tail of the data

The descriptive stat for the given dataset is shown below.

	count	mean	std	min	25%	50%	75%	max
applicant_id	25000.0	17499.500000	7217.022701	5000.0	11249.75	17499.5	23749.25	29999.0
years_of_insurance_with_us	25000.0	4.089040	2.606612	0.0	2.00	4.0	6.00	8.0
regular_checkup_lasy_year	25000.0	0.773680	1.199449	0.0	0.00	0.0	1.00	5.0
adventure_sports	25000.0	0.081720	0.273943	0.0	0.00	0.0	0.00	1.0
visited_doctor_last_1_year	25000.0	3.104200	1.141663	0.0	2.00	3.0	4.00	12.0
daily_avg_steps	25000.0	5215.889320	1053.179748	2034.0	4543.00	5089.0	5730.00	11255.0
age	25000.0	44.918320	16.107492	16.0	31.00	45.0	59.00	74.0
heart_decs_history	25000.0	0.054640	0.227281	0.0	0.00	0.0	0.00	1.0
other_major_decs_history	25000.0	0.098160	0.297537	0.0	0.00	0.0	0.00	1.0
avg_glucose_level	25000.0	167.530000	62.729712	57.0	113.00	168.0	222.00	277.0
bmi	24010.0	31.393328	7.876535	12.3	26.10	30.5	35.60	100.6
Year_last_admitted	13119.0	2003.892217	7.581521	1990.0	1997.00	2004.0	2010.00	2018.0
weight	25000.0	71.610480	9.325183	52.0	64.00	72.0	78.00	96.0
weight_change_in_last_one_year	25000.0	2.517960	1.690335	0.0	1.00	3.0	4.00	6.0
fat_percentage	25000.0	28.812280	8.632382	11.0	21.00	31.0	36.00	42.0
insurance_cost	25000.0	27147.407680	14323.691832	2468.0	16042.00	27148.0	37020.00	67870.0

Figure 3: Description for the dataset.

As we can the average insurance costs of the individuals are 27148 with 2468 as the minimum insurance costs and 67870 as the maximum insurance costs covered by the company.

The Info about the dataset is given below.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 25000 entries, 0 to 24999
Data columns (total 24 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   applicant_id                    25000 non-null  int64
 1   years_of_insurance_with_us      25000 non-null  int64
 2   regular_checkup_lasy_year       25000 non-null  int64
 3   adventure_sports                25000 non-null  int64
 4   Occupation                      25000 non-null  object
 5   visited_doctor_last_1_year      25000 non-null  int64
 6   cholesterol_level               25000 non-null  object
 7   daily_avg_steps                 25000 non-null  int64
 8   age                             25000 non-null  int64
 9   heart_decs_history              25000 non-null  int64
 10  other_major_decs_history        25000 non-null  int64
 11  Gender                          25000 non-null  object
 12  avg_glucose_level               25000 non-null  int64
 13  bmi                             24010 non-null  float64
 14  smoking_status                  25000 non-null  object
 15  Year_last_admitted              13119 non-null  float64
 16  Location                        25000 non-null  object
 17  weight                          25000 non-null  int64
 18  covered_by_any_other_company    25000 non-null  object
 19  Alcohol                         25000 non-null  object
 20  exercise                        25000 non-null  object
 21  weight_change_in_last_one_year  25000 non-null  int64
 22  fat_percentage                  25000 non-null  int64
 23  insurance_cost                  25000 non-null  int64
dtypes: float64(2), int64(14), object(8)
memory usage: 4.6+ MB
```

Figure 4: Info of the dataset.

C. UNIVARIATE ANALYSIS.

Univariate Analysis has to be performed for Categorical variables and Numerical Variables. The Univariate Analysis for numerical variables is shown below.

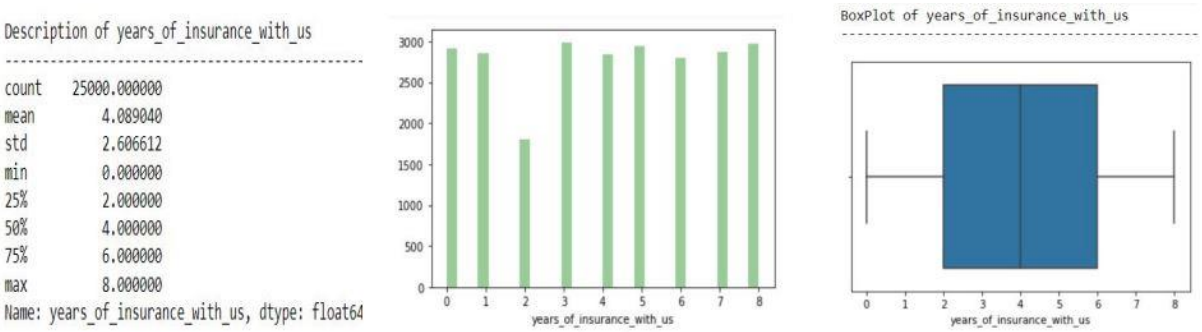

Figure 5: Univariate Analysis for Years of Insurance with us.

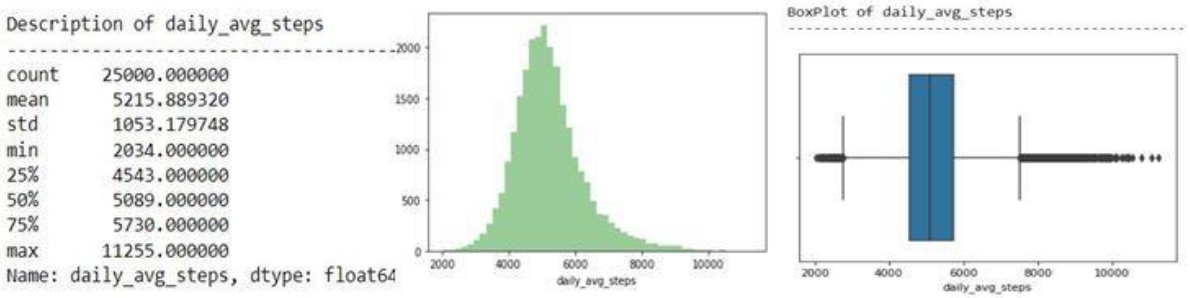

Figure 6: Univariate Analysis for Daily Avg Steps.

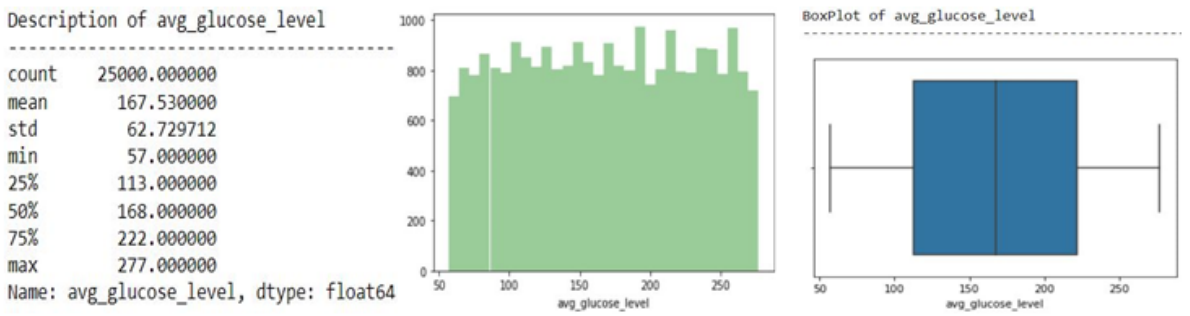

Figure 7: Univariate Analysis for Avg_Glucose_level.

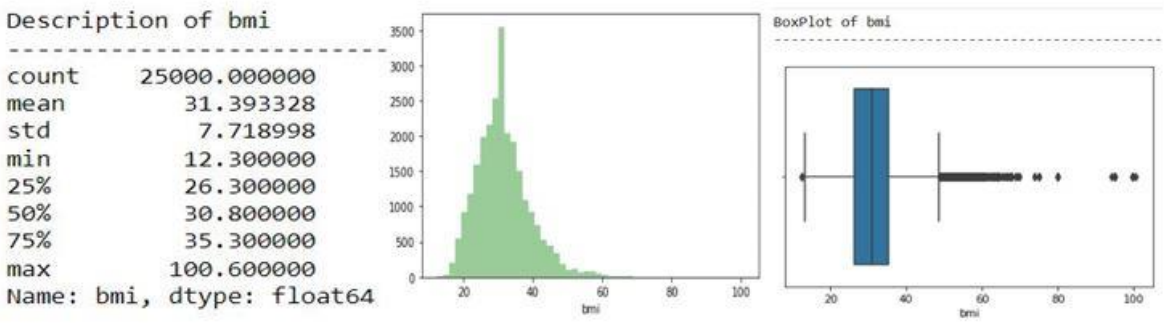

Figure 8: Univariate Analysis for BMI.

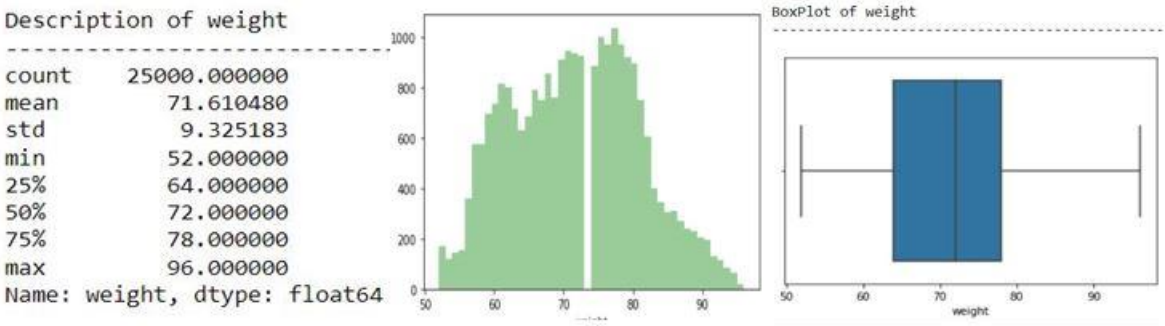

Figure 9: Univariate Analysis for Weight.

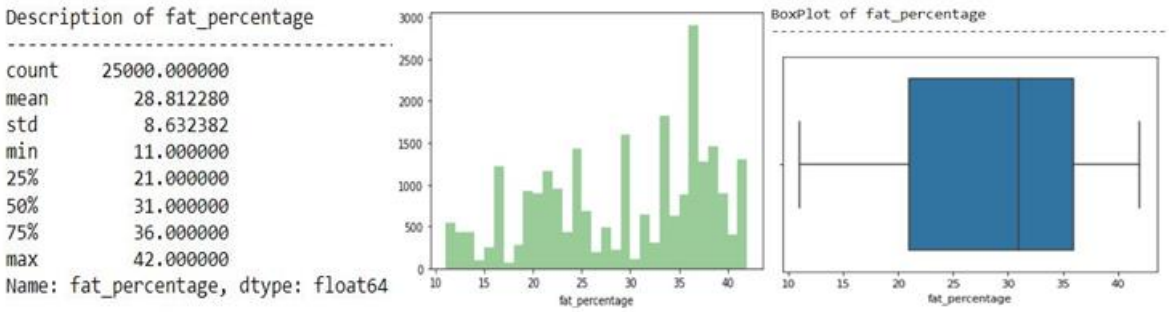

Figure 10: Univariate Analysis for Fat Percentage

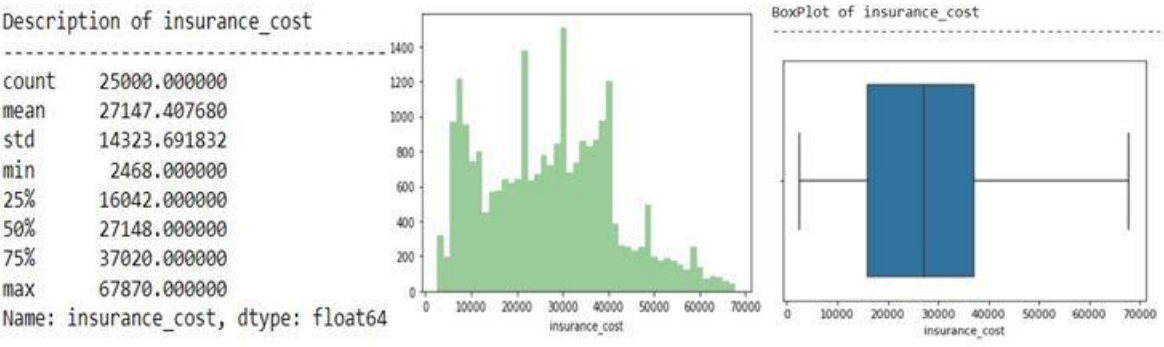

Figure 11: Univariate Analysis for Insurance Cost

The Univariate Analysis for the Categorical variable is plotted and the results are given below.

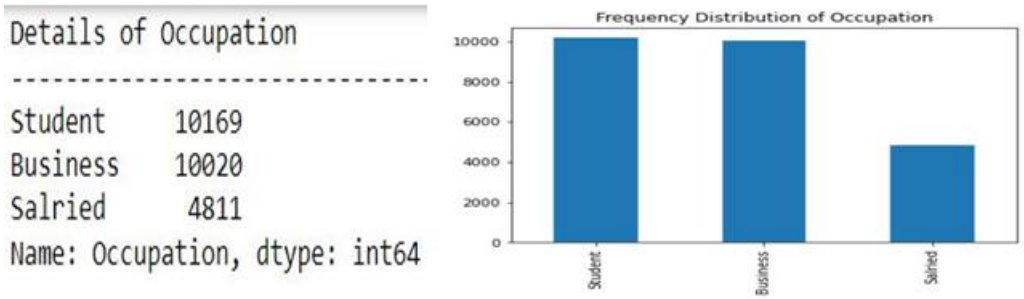

Figure 12: Univariate Analysis for Occupation.

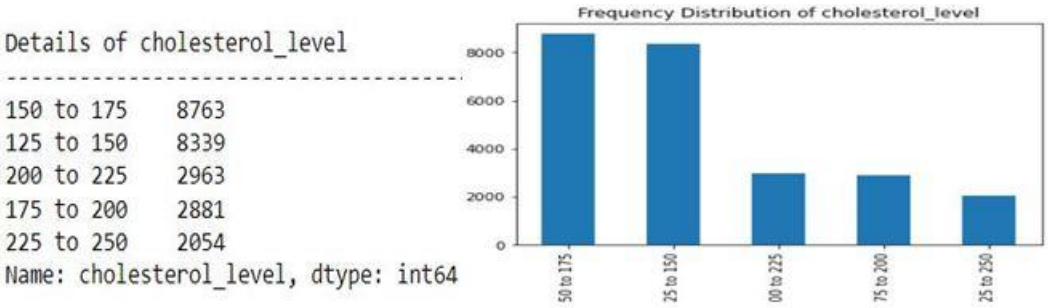

Figure 13: Univariate Analysis for Cholesterol Level.

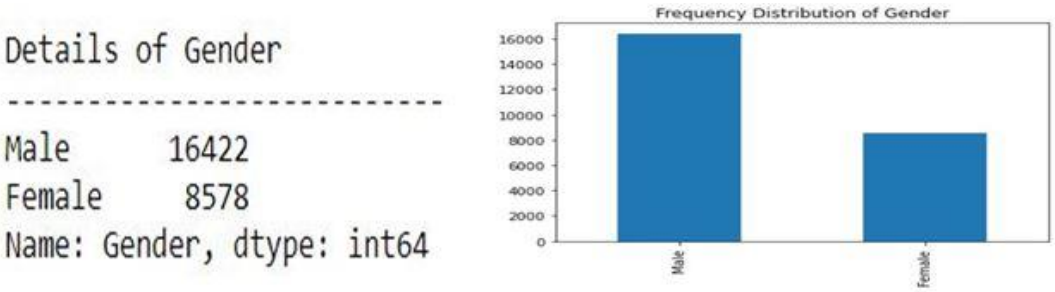

Figure 14: Univariate Analysis for Gender.

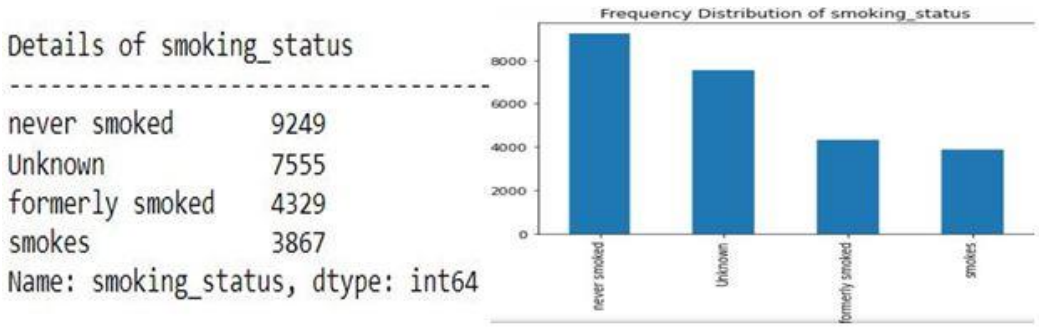

Figure 15: Univariate Analysis for Smoking Status.

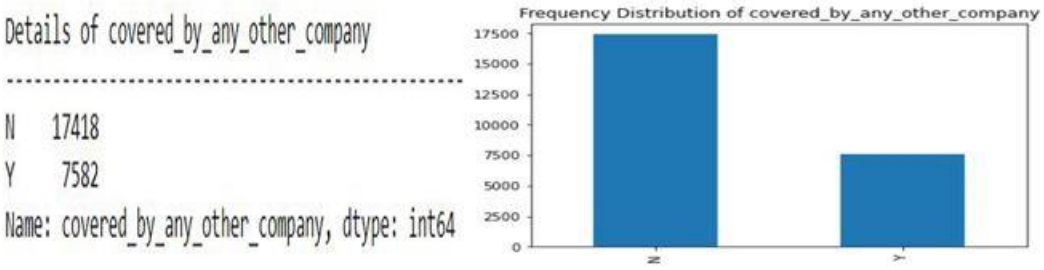

Figure 16: Univariate Analysis for Insurance Covered by other company.

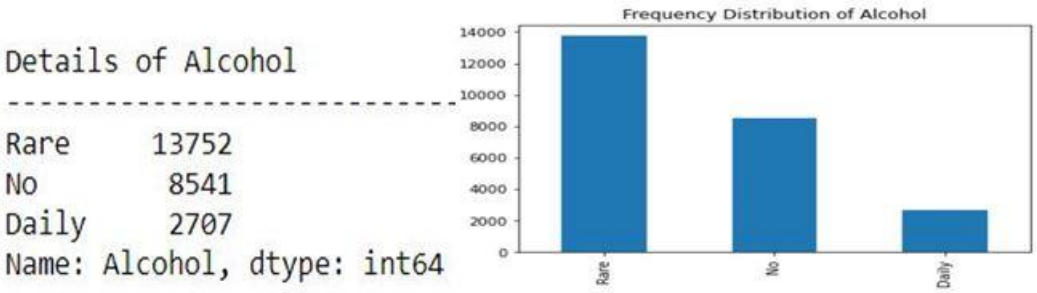

Figure 17: Univariate Analysis for Alcohol.

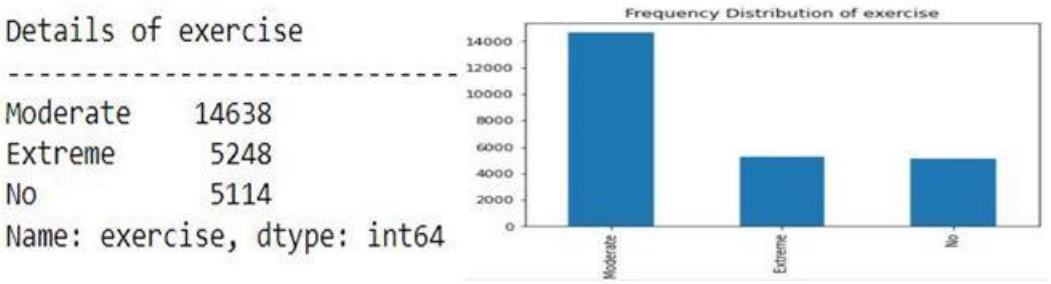

Figure 18: Univariate Analysis for Exercise.

INFERENCE:

- Average insurance costs of the individuals are **27148** with **2468** as the minimum insurance costs and **67870** as the maximum insurance costs covered by the company.
- Most of the customers are having 4 years of contract with the company.
- Customers are having 5089 as average for the daily steps where as there are some customers who are having 11255 as their daily average steps.
- 168 is the average Glucose Level for the customers in the dataset.
- Around 50% of the customers in the dataset are obese.
- We are having more number of students compared to working professionals.
- Most number of people have the cholesterol level from 150 – 175 with **8763**.
- The dataset contains more number of male customers.

- More than 50% of the data are loyal to the company as they are staying with the company for more than 4 years.
- Most of the customers in the dataset have never smoked whereas only 15% of the customers smoke regularly.
- Customers have a habit of social drinking as more than 50% of the customers have rare alcohol consumption.

These are the inferences from the Univariate Analysis. Now we can go to Bivariate Analysis and we can see if there is any relation between the variable.

D. BIVARIATE ANALYSIS.

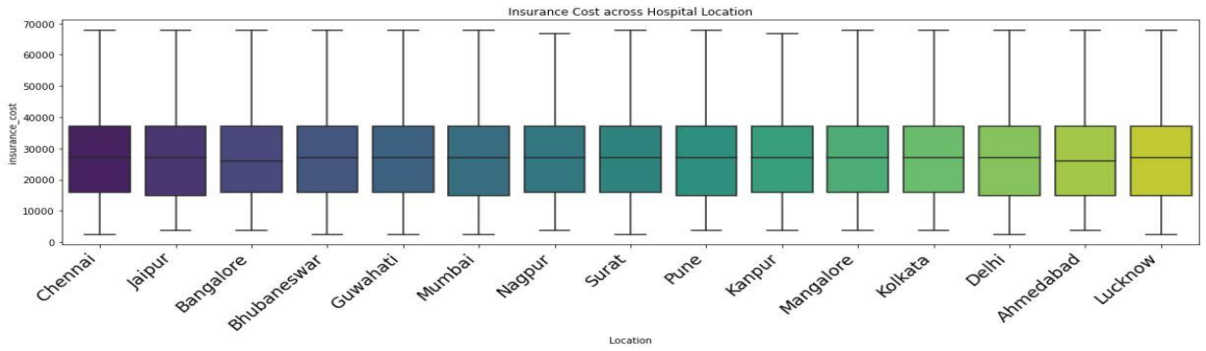

Figure 19: Bivariate Analysis for Location VS Insurance Cost.

From this we can say that Location doesn't have any impact on the target variable. Here the insurance cost across location is more or less the same.

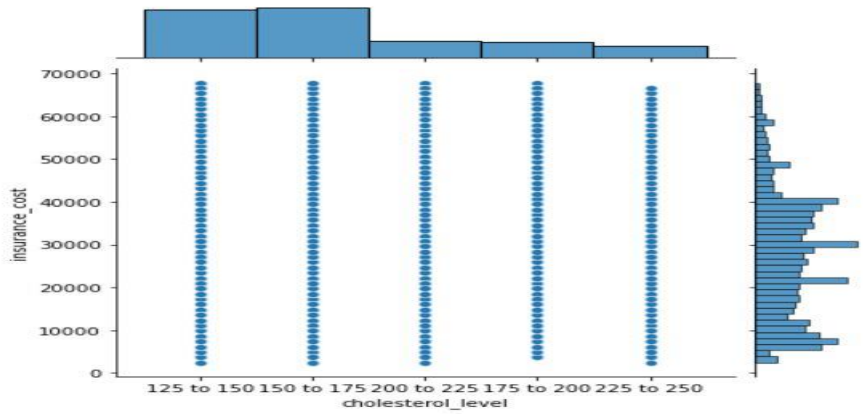

Figure 20: Bivariate Analysis for Cholesterol Level VS Insurance Cost.

Here customers with Cholesterol level of 150 – 175 are the highest and only very few customers are claiming high insurance cost whereas most of the customers are claiming in between 20,000 to 40,000.

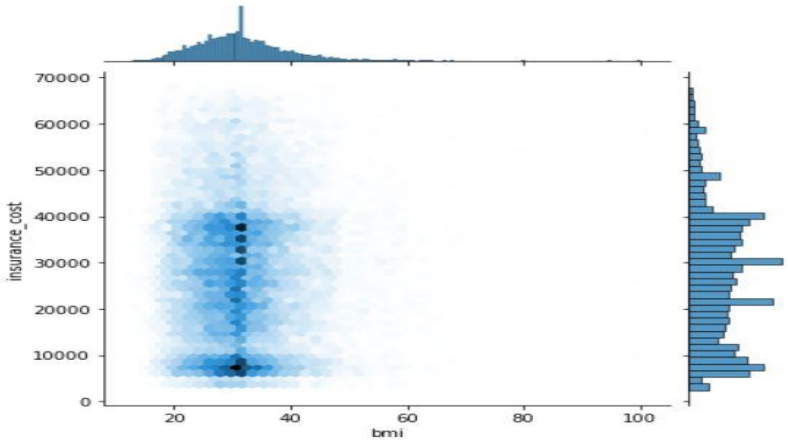

Figure 21: Bivariate Analysis for BMI VS Insurance Cost.

Here we can see that there are more number of customers with BMI more than 30 and the insurance cost are most claimed between the ranges of 30,000 to 40,000.

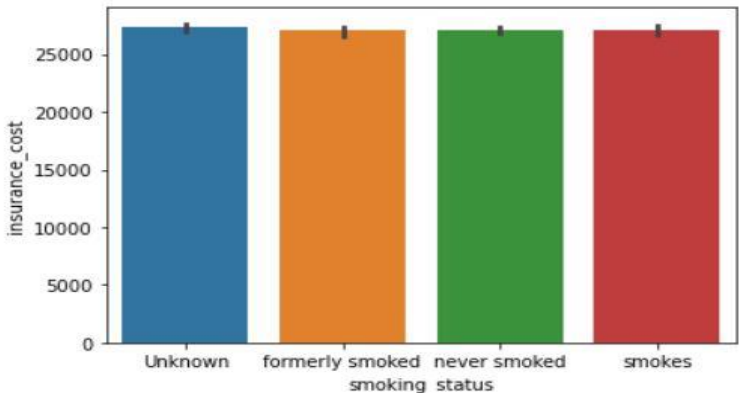

Figure 22: Bivariate Analysis for Smoking status VS Insurance Cost.

We can see that smoking habits doesn't have any impact on predicting the insurance cost.

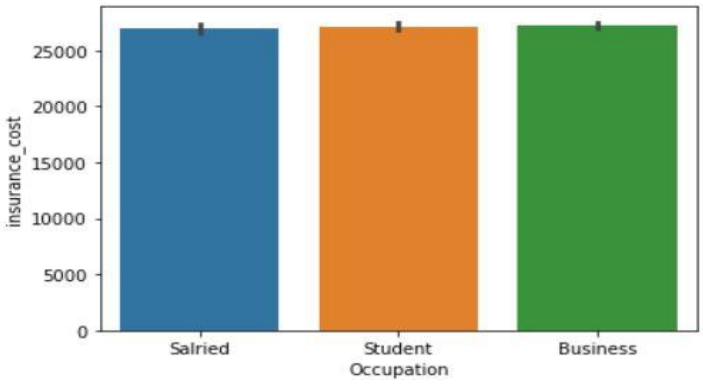

Figure 23: Bivariate Analysis for Occupation VS Insurance Cost.

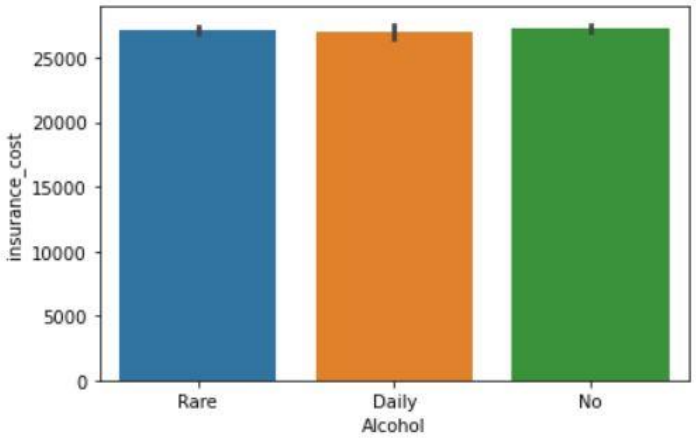

Figure 24: Bivariate Analysis for Alcohol VS Insurance Cost.

The Alcohol habits don't have any impact on the target variable. However there is a higher probability that this person will be prone to diseases.

E. MULTIVARIATE ANALYSIS.

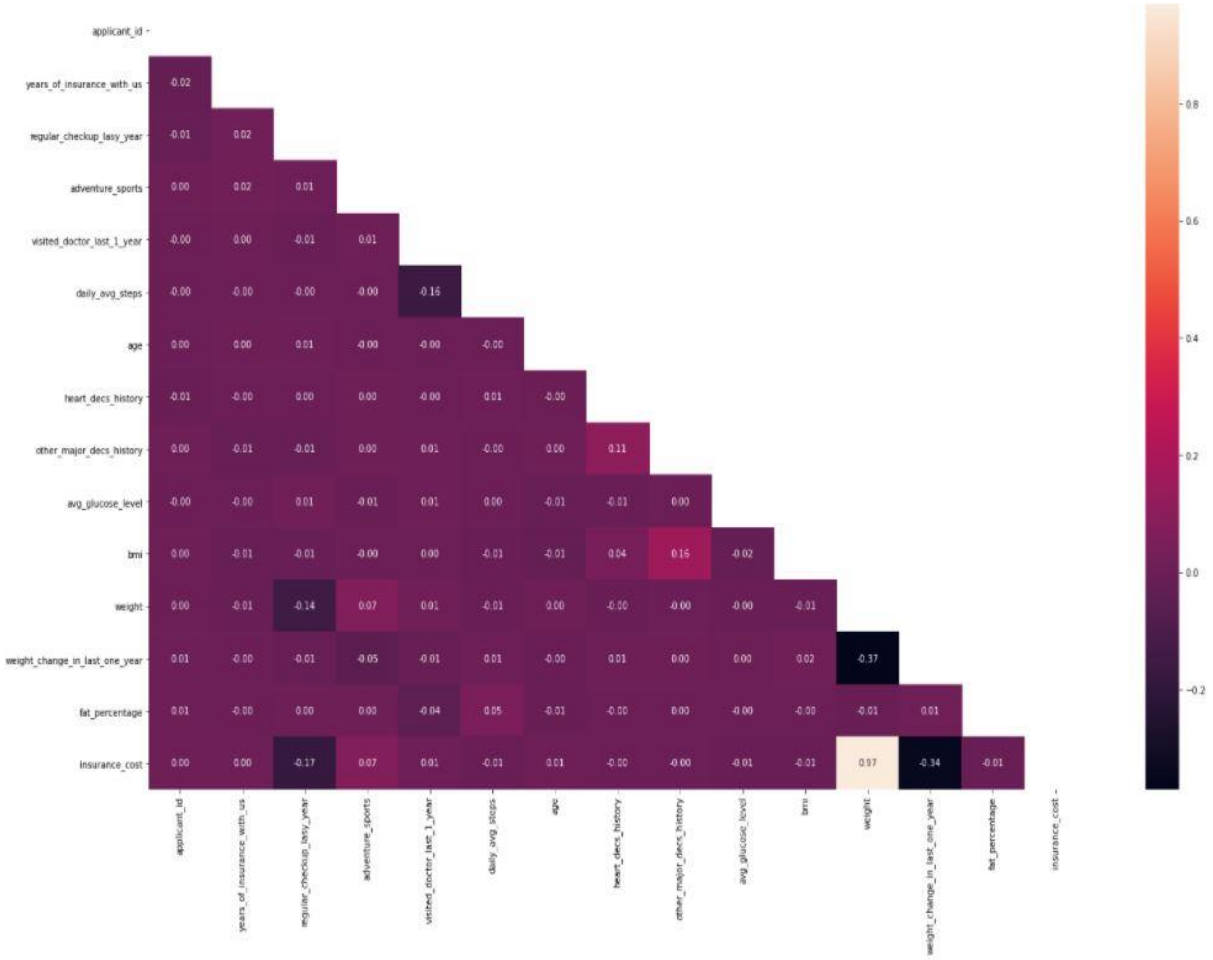

Figure 25: Heatmap for the dataset.

From the Heatmap we can observe that there are very few variables with high correlation with the target variable.

Weight has the high correlation with the target variable which says that as the weight increases, insurance cost for the individual increases.

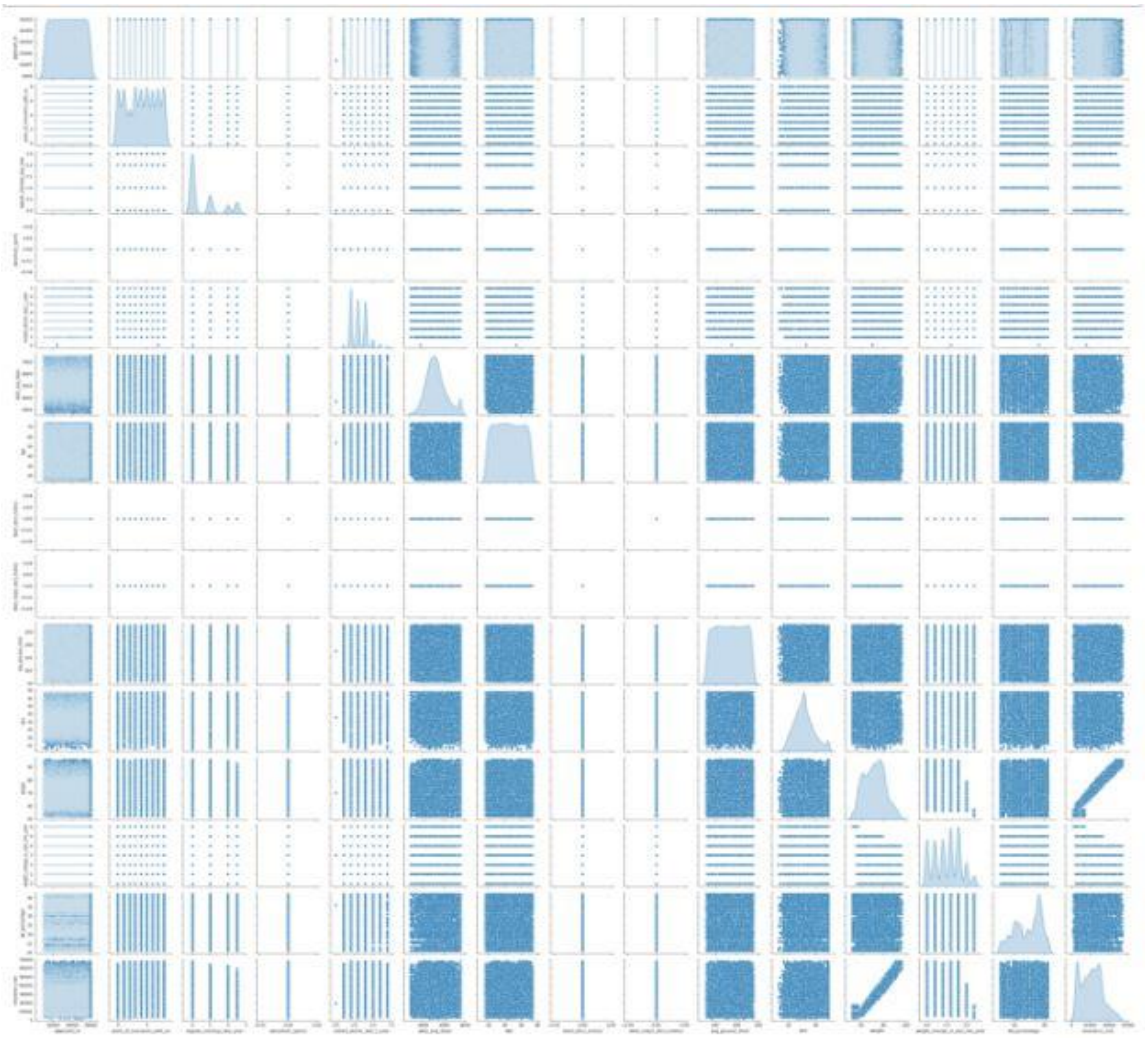

Figure 26: Pairplot for the dataset.

From the Pairplot we can understand the relationship of variable with the other.

NOTE:

Weight has high correlation with the target variable. We can say that BMI will also have an indirect impact on target variable as weight is related to BMI.

3. DATA CLEANING AND PRE-PROCESSING.

- **APPROACH USED FOR IDENTIFYING AND TREATING MISSING VALUES AND OUTLIER TREATMENT (AND WHY)**
- **MISSING VALUE TREATMENT:**

From the given dataset we can see that Year Last Admitted and BMI have null values present in it. In Year Last Admitted we can see that there are more than **50%** of null values present in it. So we can drop the variable.

We can see that there are only 4% of null values present in the BMI variable. This can be treated since only very few data points are missing in the BMI variable.

The null value which is present in BMI has been imputed with the mean of the variable which is **31.39**.

```
applicant_id                      0
years_of_insurance_with_us        0
regular_checkup_lasy_year         0
adventure_sports                  0
visited_doctor_last_1_year        0
daily_avg_steps                   0
age                               0
heart_decs_history                0
other_major_decs_history          0
avg_glucose_level                 0
bmi                               0
weight                            0
weight_change_in_last_one_year    0
fat_percentage                    0
insurance_cost                    0
Occupation_Salried                0
Occupation_Student                0
cholesterol_level_150 to 175      0
cholesterol_level_175 to 200      0
cholesterol_level_200 to 225      0
cholesterol_level_225 to 250      0
Gender_Male                       0
smoking_status_formerly smoked    0
smoking_status_never smoked       0
smoking_status_smokes             0
Location_Bangalore                0
Location_Bhubaneswar              0
Location_Chennai                  0
Location_Delhi                    0
Location_Guwahati                 0
Location_Jaipur                   0
Location_Kanpur                   0
Location_Kolkata                  0
Location_Lucknow                  0
Location_Mangalore                0
Location_Mumbai                   0
Location_Nagpur                   0
Location_Pune                     0
Location_Surat                    0
covered_by_any_other_company_Y    0
Alcohol_No                        0
Alcohol_Rare                      0
exercise_Moderate                 0
exercise_No                       0
dtype: int64
```

Figure 27: After Missing Value Treatment.

From the above figure, we can see that all the null values have been treated with the mean which leads to no null values in the dataset.

- **OUTLIER TREATMENT:**

From the Univariate Analysis we can see that there are outliers present in the dataset. Having outliers will have a huge impact on model performance and accuracy.

Since we are dealing with the regression problems, the model is sensitive to outliers which will affect in optimizing the insurance cost for the company. The performance metrics which is used to analyse the performance and accuracy of the model is also sensitive to outliers.

So we are treating the outliers by adjusting the Inter Quartile Range by changing the upper limit and lower limit range such that there are no outliers present in the dataset.

The Boxplot after Outlier Treatment is given below.

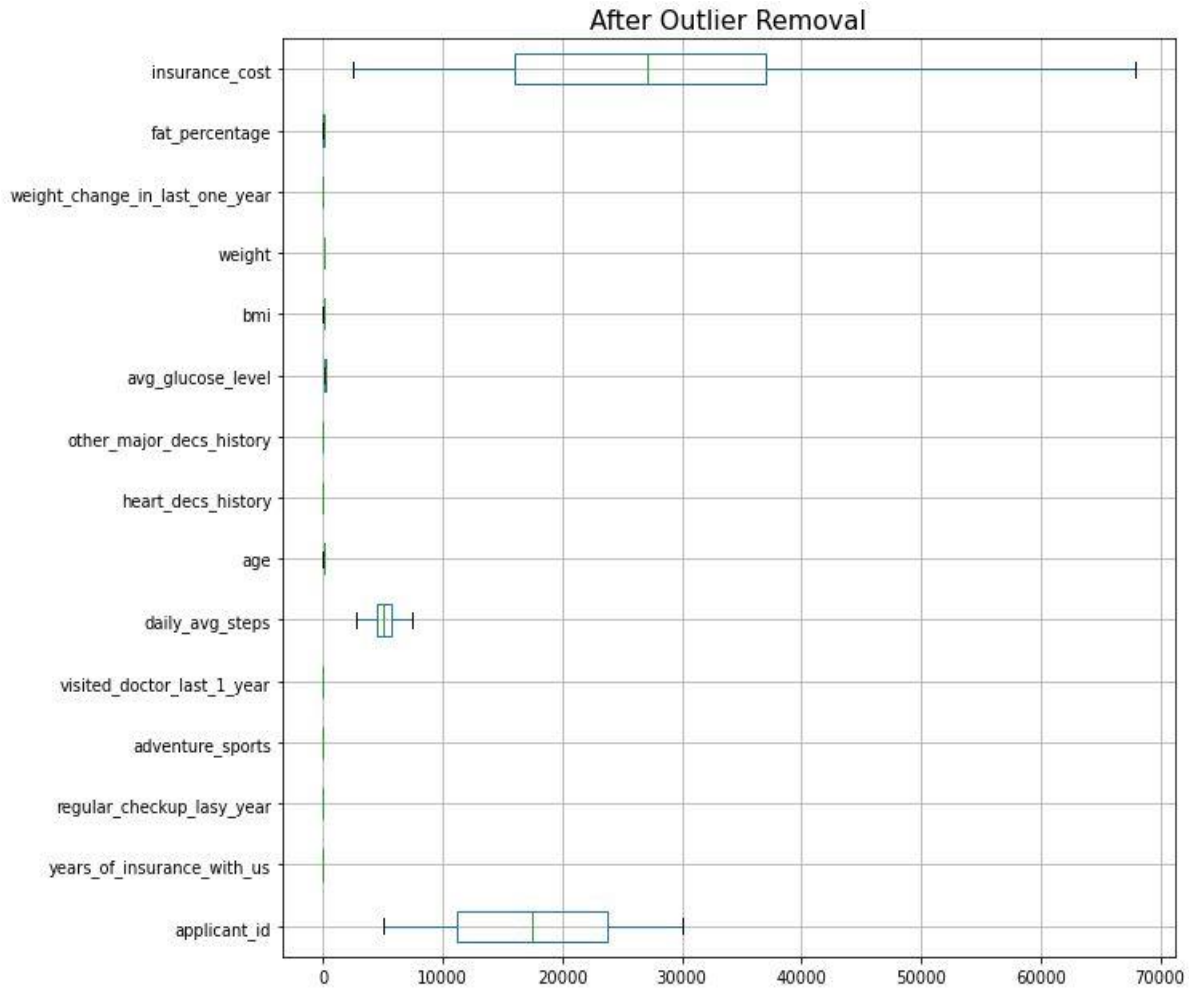

Figure 28: Boxplot after Outlier Treatment.

- ## NEED FOR VARIABLE TRANSFORMATION (IF ANY)

Here we have a Variable Adventure Sports which is detected as a numerical variable by python. Even though we can see numerical values present in the dataset we know that this is a categorical variable since it has only 1 and 0.

Also we need to transform Categorical variable to Numerical variable since we are dealing with the regression model. So we are converting all the categorical variables to numeric.

- ## VARIABLES REMOVED OR ADDED AND WHY (IF ANY)

Here there are some new variables added to the dataset since we are converting the categorical variables to numerical variables using Get Dummies. All the new variables added is a sub category of the categorical variables which are converted to numerical variable.

Also we are adding a variable called KM_Clusters which is a result of K-Means Clustering to further analyse the data.

4. MODEL BUILDING

- ## CLEAR ON WHY WAS A PARTICULAR MODEL(S) CHOSEN.

The given problem is to predict and optimize the insurance cost for the company. We have to build regression models since the target variable is numeric. The models that can be built for the given set of data are shown below.

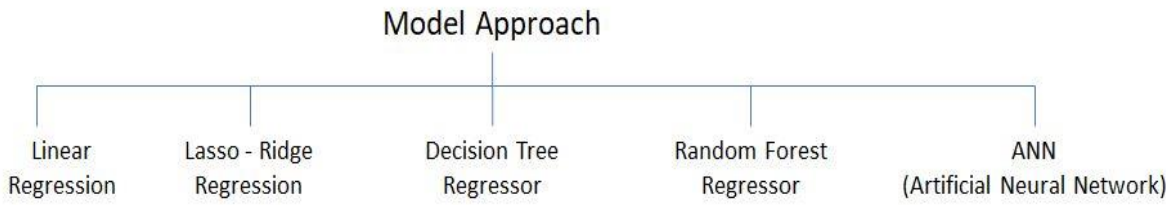

Figure 29: Regression Models.

Before we jump to model building, we have to split the data into train and test set which are used to train and test the model.

The head of the training data is shown below.

applicant_id	years_of_insurance_with_us	regular_checkup_lasy_year	adventure_sports	visited_doctor_last_1_year	daily_avg_steps	age	heart_decs_hist
9289.0	0.0	2.0	0.0	4.0	5245.0	45.0	
24621.0	6.0	2.0	0.0	3.0	7510.5	60.0	
19965.0	6.0	0.0	0.0	4.0	5828.0	41.0	
17321.0	6.0	1.0	0.0	3.0	4463.0	55.0	
11269.0	4.0	2.5	0.0	3.0	7510.5	25.0	

Figure 30: Head of training data.

The tail of the training data is given below.

applicant_id	years_of_insurance_with_us	regular_checkup_lasy_year	adventure_sports	visited_doctor_last_1_year	daily_avg_steps	age	heart_decs_hist
15955.0	0.0	0.0	0.0	2.0	5703.0	25.0	
22289.0	4.0	2.5	0.0	2.0	5355.0	56.0	
10192.0	8.0	1.0	0.0	2.0	4427.0	69.0	
17172.0	6.0	0.0	0.0	2.0	3485.0	59.0	
5235.0	4.0	2.5	0.0	4.0	6168.0	69.0	

Figure 31: Tail of training data.

The head of the testing data is given below.

applicant_id	years_of_insurance_with_us	regular_checkup_lasy_year	adventure_sports	visited_doctor_last_1_year	daily_avg_steps	age	heart_decs_hist
26492.0	5.0	0.0	0.0	2.0	3099.0	32.0	
14488.0	1.0	1.0	0.0	4.0	5119.0	26.0	
21933.0	5.0	0.0	0.0	3.0	4649.0	60.0	
17604.0	6.0	2.5	0.0	3.0	5837.0	55.0	
13222.0	7.0	0.0	0.0	3.0	4453.0	37.0	

Figure 32: Head of testing data.

SUMMARY:

- Missing values have been treated and the outliers have been removed by performing the outlier treatment.
- All categorical variables have been converted to Numerical variables using Get Dummies.
- Data has been split to train and test data with the ratio of **70:30**.

Now we can continue to model building.

- **LINEAR REGRESSION**:

Linear Regression model can be built using two libraries. One is using Statsmodels and the other method is using Sklearn. The two models have been built and the results are shown below.

Linear Regression using Sklearn library has been built and the model it's been trained using the training dataset.

Once the model is trained we can check the model accuracy and other metrics to measure the performance of the model.

The Coefficients of variables is calculated after the model is built and trained using training dataset and it is shown below.

```
The coefficient for applicant_id is 0.002105759049567801
The coefficient for years_of_insurance_with_us is -13.689477877823894
The coefficient for regular_checkup_lasy_year is -621.9140506283179
The coefficient for adventure_sports is -8.321308087033685e-10
The coefficient for visited_doctor_last_1_year is -35.42786474040338
The coefficient for daily_avg_steps is -0.03174877297199258
The coefficient for age is 2.839553495819824
The coefficient for heart_decs_history is -3.2514435588382185e-10
The coefficient for other_major_decs_history is 3.275317794759758e-10
The coefficient for avg_glucose_level is 0.36858826333210054
The coefficient for bmi is -0.3321215020905373
The coefficient for weight is 1489.6101112212457
The coefficient for weight_change_in_last_one_year is 172.1939053765156
The coefficient for fat_percentage is -1.2017480474388587
The coefficient for Occupation_Salried is -1.0450496001072667
The coefficient for Occupation_Student is 40.453028842097545
The coefficient for cholesterol_level_150 to 175 is -61.61711104800862
The coefficient for cholesterol_level_175 to 200 is -22.47797108454849
The coefficient for cholesterol_level_200 to 225 is 48.0257931340716
The coefficient for cholesterol_level_225 to 250 is 161.28538334617565
The coefficient for Gender_Male is 52.754473326514315
The coefficient for smoking_status_formerly smoked is -39.862686616458134
The coefficient for smoking_status_never smoked is 20.438417580473345
The coefficient for smoking_status_smokes is -40.7371656515802
The coefficient for Location_Bangalore is 339.9395914714865
The coefficient for Location_Bhubaneswar is 252.06673723956362
The coefficient for Location_Chennai is 284.9241204864336
The coefficient for Location_Delhi is 427.4026038414833
The coefficient for Location_Guwahati is 301.3136587266063
The coefficient for Location_Jaipur is 392.8130483850008
The coefficient for Location_Kanpur is 317.8868502870905
The coefficient for Location_Kolkata is 227.14254787496083
The coefficient for Location_Lucknow is 408.35214662701003
The coefficient for Location_Mangalore is 238.83175824147884
The coefficient for Location_Mumbai is 256.7045576715103
The coefficient for Location_Nagpur is 444.77074401447896
The coefficient for Location_Pune is 309.06565774212436
The coefficient for Location_Surat is 339.348830950629
The coefficient for covered_by_any_other_company_Y is 1214.5692881434652
The coefficient for Alcohol_No is -16.991962934530015
The coefficient for Alcohol_Rare is 0.24894015134779715
The coefficient for exercise_Moderate is 14.621308382012979
The coefficient for exercise_No is 6.734913855325543
```

Figure 33: Coefficients of model in training data.

The intercept of the model is calculated and it is shown below.

```
The intercept for our model is -80105.05064290411
```

Figure 34: Intercept for training data.

The accuracy of the model for the training dataset is **0.9447**.

The performance of the model can be examined by using the following metrics.

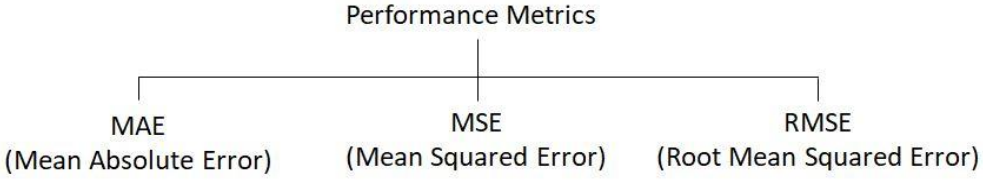

Figure 35: Performance Metrics.

The performance of the model on training data is given below.

	ACCURACY	MAE	MSE	RMSE
LR – Training model.	**0.9447**	**2721.016**	**11408227.133**	**3377.606**

Table 1: Performance metric on Training data - LR.

```
MAE: 2721.016248851743

MSE: 11408227.133204147

RMSE: 3377.6067167750816
```

Figure 36: Code Result - Performance Metrics for Linear Regression Training model.

Now we can use the test data to test the model and check the performance of the model in optimizing the insurance cost.

The Coefficients of the model in testing data is calculated and it is displayed below.

```
The coefficient for applicant_id is -2.1605606920997324e+16
The coefficient for years_of_insurance_with_us is 18.934412456837165
The coefficient for regular_checkup_lasy_year is -40.47594347254871
The coefficient for adventure_sports is -637.3466824712933
The coefficient for visited_doctor_last_1_year is 197974225844727.3
The coefficient for daily_avg_steps is -25.397419897905763
The coefficient for age is -25.74833463828969
The coefficient for heart_decs_history is 31.099622033881303
The coefficient for other_major_decs_history is -2.8493967140790652e+16
The coefficient for avg_glucose_level is -1.452365545696681e+16
The coefficient for bmi is 55.625
The coefficient for weight is -19.21875
The coefficient for weight_change_in_last_one_year is 13989.75
The coefficient for fat_percentage is 119.75
The coefficient for Occupation_Salried is 21.125
The coefficient for Occupation_Student is -2.2430753952331268e+16
The coefficient for cholesterol_level_150 to 175 is 1.2913914485949356e+16
The coefficient for cholesterol_level_175 to 200 is 8711733942742878.0
The coefficient for cholesterol_level_200 to 225 is 1.5961489888541128e+16
The coefficient for cholesterol_level_225 to 250 is 1.1106245837688718e+16
The coefficient for Gender_Male is -1.3874733859765716e+16
The coefficient for smoking_status_formerly smoked is 38.625
The coefficient for smoking_status_never smoked is -3294276579598284.5
The coefficient for smoking_status_smokes is -8409116957693993.0
The coefficient for Location_Bangalore is 2842539969958395.0
The coefficient for Location_Bhubaneswar is -417212577367976.6
The coefficient for Location_Chennai is 1.0994411179682606e+16
The coefficient for Location_Delhi is 5581612852552772.0
The coefficient for Location_Guwahati is -1674823257507606.8
The coefficient for Location_Jaipur is 1.5119547683905438e+16
The coefficient for Location_Kanpur is 5107494080118760.0
The coefficient for Location_Kolkata is 2398127422630625.0
The coefficient for Location_Lucknow is -1.4485133580216726e+16
The coefficient for Location_Mangalore is -1881714533896838.2
The coefficient for Location_Mumbai is -996787266268802.9
The coefficient for Location_Nagpur is -9662037377383398.0
The coefficient for Location_Pune is 1349035676065691.8
The coefficient for Location_Surat is -6886939079854346.0
The coefficient for covered_by_any_other_company_Y is 745370370871553.0
The coefficient for Alcohol_No is 533.46875
The coefficient for Alcohol_Rare is 2.283473932440554e+16
The coefficient for exercise_Moderate is 1.4876590839411658e+16
The coefficient for exercise_No is -7441081042163864.0
```

Figure 37: Coefficients of model in testing data.

The model score for the testing dataset is **0.9449**.

The performance metrics of the model in testing data is given below.

	ACCURACY	MAE	MSE	RMSE
LR – Testing model.	**0.9449**	**2709.381**	**11131896.631**	**3336.449**

Table 2: Performance Metrics testing data - LR

```
MAE: 2709.3817087074876

MSE: 11131896.631153185

RMSE: 3336.4497045741878
```

Figure 38: Code result - Performance metrics of model in testing data.

Now we can use the Statsmodels library to create the Linear Regression model and we can use the train data to train the model.

The model parameters are shown below.

Intercept	-8.007003e+04	Gender_Male	5.264966e+01
years_of_insurance_with_us	-1.372820e+01	smoking_status_formerly_smoked	-4.047469e+01
regular_checkup_lasy_year	-6.219187e+02	smoking_status_never_smoked	2.031204e+01
adventure_sports	1.088230e-09	smoking_status_smokes	-4.076494e+01
visited_doctor_last_1_year	-3.548102e+01	Location_Bangalore	3.399728e+02
daily_avg_steps	-3.180821e-02	Location_Bhubaneswar	2.522238e+02
age	2.844453e+00	Location_Chennai	2.839197e+02
heart_decs_history	-1.394459e-10	Location_Delhi	4.282567e+02
other_major_decs_history	-5.214331e-11	Location_Guwahati	3.015159e+02
avg_glucose_level	3.673097e-01	Location_Jaipur	3.921572e+02
bmi	-3.066434e-01	Location_Kanpur	3.182858e+02
weight	1.489622e+03	Location_Kolkata	2.262297e+02
weight_change_in_last_one_year	1.722696e+02	Location_Lucknow	4.082907e+02
fat_percentage	-1.173985e+00	Location_Mangalore	2.392263e+02
Occupation_Salried	-6.049516e-01	Location_Mumbai	2.569246e+02
Occupation_Student	4.069402e+01	Location_Nagpur	4.442287e+02
cholesterol_level_150_to_175	-6.081397e+01	Location_Pune	3.088729e+02
cholesterol_level_175_to_200	-2.231646e+01	Location_Surat	3.388740e+02
cholesterol_level_200_to_225	4.828971e+01	covered_by_any_other_company_Y	1.214503e+03
cholesterol_level_225_to_250	1.617918e+02	Alcohol_No	-1.735976e+01
		Alcohol_Rare	-6.486472e-02
		exercise_Moderate	1.471346e+01
		exercise_No	6.392007e+00

Figure 39: Model Parameters for training data.

The RMSE value for the model in train data is **3377.640**.

For OLS model, we can check the R-Squared and Adjusted R-Squared for the model.

The summary of the model in training data is given below.

OLS Regression Results

Dep. Variable:	insurance_cost	R-squared:	0.945
Model:	OLS	Adj. R-squared:	0.945
Method:	Least Squares	F-statistic:	7655.
Date:	Sun, 24 Apr 2022	Prob (F-statistic):	0.00
Time:	11:44:15	Log-Likelihood:	-1.6702e+05
No. Observations:	17500	AIC:	3.341e+05
Df Residuals:	17460	BIC:	3.344e+05
Df Model:	39		
Covariance Type:	nonrobust		

Figure 40: OLS model - training data summary.

Now we can use the test data to check the performance of the model and the it is given below.

Intercept	-7.920330e+04	Gender_Male	-6.631715e+01
years_of_insurance_with_us	-4.766544e+00	smoking_status_formerly_smoked	1.065836e+02
regular_checkup_lasy_year	-5.443343e+02	smoking_status_never_smoked	-5.795219e+01
adventure_sports	-6.386057e-10	smoking_status_smokes	-5.880065e+01
visited_doctor_last_1_year	-5.236997e+01	Location_Bangalore	6.092729e+01
daily_avg_steps	-1.073874e-02	Location_Bhubaneswar	2.186051e+02
age	4.883556e+00	Location_Chennai	4.540830e+02
heart_decs_history	1.744445e-10	Location_Delhi	6.734482e+02
other_major_decs_history	-1.344132e-10	Location_Guwahati	1.981085e+02
avg_glucose_level	-7.354842e-01	Location_Jaipur	2.089765e+02
bmi	-7.865194e+00	Location_Kanpur	6.561601e+01
weight	1.484994e+03	Location_Kolkata	2.228169e+02
weight_change_in_last_one_year	1.141068e+02	Location_Lucknow	1.874169e+02
fat_percentage	-8.895684e+00	Location_Mangalore	2.003016e+02
Occupation_Salried	-5.892302e+01	Location_Mumbai	3.592173e+02
Occupation_Student	7.626113e+00	Location_Nagpur	7.936114e+01
cholesterol_level_150_to_175	-3.693586e+01	Location_Pune	1.071312e+02
cholesterol_level_175_to_200	1.567595e+02	Location_Surat	2.562049e+02
cholesterol_level_200_to_225	-3.415402e+01	covered_by_any_other_company_Y	1.206109e+03
cholesterol_level_225_to_250	1.807089e+02	Alcohol_No	1.231102e+02
		Alcohol_Rare	9.970688e+01
		exercise_Moderate	6.324378e+01
		exercise_No	1.131585e+02

Figure 41: Model parameters - testing data.

The Root Mean Squared Error of the model on test data is **3336.740**.

As we can see, both the models show the same RMSE value which explains that either of those models can be taken into consideration.

Once this is done we can look in to the summary of the model on test data.

The summary of the model in test data is shown below.

OLS Regression Results

Dep. Variable:	insurance_cost	R-squared:	0.945
Model:	OLS	Adj. R-squared:	0.945
Method:	Least Squares	F-statistic:	3305.
Date:	Sun, 24 Apr 2022	Prob (F-statistic):	0.00
Time:	18:32:28	Log-Likelihood:	-71461.
No. Observations:	7500	AIC:	1.430e+05
Df Residuals:	7460	BIC:	1.433e+05
Df Model:	39		
Covariance Type:	nonrobust		

Figure 42: Summary of the model in testing data.

Now we can look in to the distribution of Insurance Cost of the model in test data and the scatter plot is shown below.

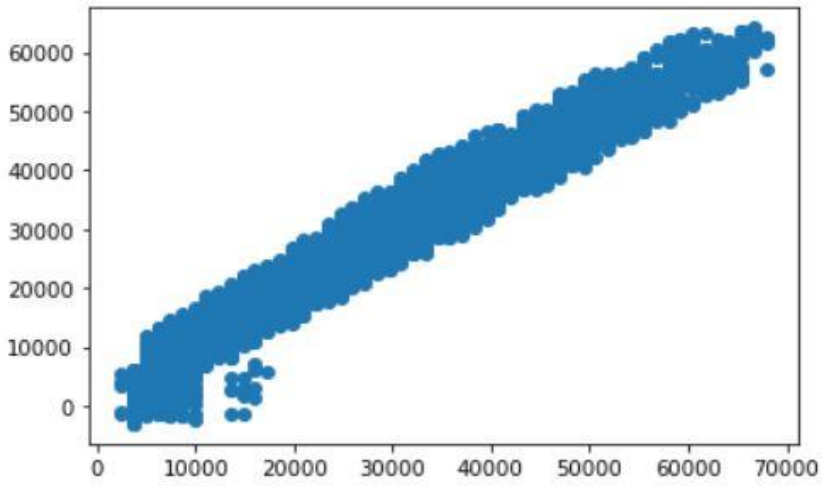

Figure 43: Scatter Plot on Insurance Cost - predicted.

- **DECISION TREE REGRESSOR**:

The Decision Tree Regressor observes features of an object and trains a model in the structure of a tree to predict data in the future to produce meaningful continuous output.

Decision Tree Regressor is imported from the library of Sklearn and the model is trained using the training dataset.

The model has been built and it is trained using the train dataset. The model score for the training dataset is **1.0**.

	ACCURACY	MAE	MSE	RMSE
Decision Tree- Training model	**1.0**	**0.0**	**0.0**	**0.0**

Table 3: Performance Metric for training data - DTree.

Here as we can able to analyse that the model score for the training data is 1.0 and the RMSE is 0.0. We can say that the model has been over fitted and requires some model tuning measures in order to regularize the model.

To regularize the model, we are performing GridSearchCV in order to get the best parameters to build the model for both train and test data.

GridSearchCV is imported from the library of Sklearn Model Selection. The grid search CV is taken all the parameters and it is fitted for the training data in order to get the best parameters.

The best parameters taken from GridSearchCV are shown below.

```
{'max_depth': 10, 'min_samples_leaf': 30, 'min_samples_split': 15}
```

Figure 44: Best Parameters for Decision Tree

Using these parameters the model has been built and trained using the train dataset.

The model score for the train data is **0.9601**.

	ACCURACY	MAE	MSE	RMSE
Decision Tree- Training model- Best Params	**0.9601**	**2286.663**	**8233782.850**	**2869.456**

Table 4: Performance Metric for training data with Best Params - DTree.

```
MAE: 2286.663299504258
MSE: 8233782.850781733
RMSE: 2869.4568912569034
```

Figure 45: Code Result - Performance metrics - training model with best params.

The Decision Tree Regressor model has been built and the model is trained and measured the performance on the training dataset.

Now we can use the model to check the performance and accuracy on the test data.

The model score for the test data is 0.9069.

	ACCURACY	MAE	MSE	RMSE
Decision Tree- Testing model	**0.9069**	**3402.713**	**18808879.077**	**4336.920**

Table 5: Performance Metric for test data - DTree.

Now we can use the model built with the best parameters taken from Grid Search CV to check the performance with the test dataset.

The model score in test data is **0.9496**.

	ACCURACY	MAE	MSE	RMSE
Decision Tree- Testing model – Best Params	**0.9496**	**2549.037**	**10176095.091**	**3189.999**

Table 6: Performance Metric - Test data with best params - DTree.

```
MAE: 2549.0377111789358
MSE: 10176095.091259174
RMSE: 3189.9992306047934
```

Figure 46: Code Result - Performance Metrics - Testing model with best params.

- **LASSO – RIDGE REGRESSION**:

Ridge regression is a method of estimating the coefficients of Multiple-Regression models in scenarios where linearly independent variables are highly correlated.

The Ridge Regression is imported from the library of Sklearn of Linear Model and the model is trained using the training dataset.

Lasso regression is a type of Linear Regression that uses shrinkage. Shrinkage is where data values are shrunk towards a central point, like the mean.

Lasso Regression is imported from the library of Sklearn Linear Model and the model is trained with the training dataset.

The lasso and ridge model has been built and the model is trained using the training dataset.

Ridge mode has been built and trained with the training dataset and the performance is given below.

The model score for the training dataset is **0.9447**.

The Coefficients of Ridge model in training data is shown below.

```
Ridge model: [[ 1.06109717e-03 -2.49138410e-03 -3.98674872e-02  0.00000000e+00
  -2.77177781e-03 -2.14904994e-03  3.19302941e-03  0.00000000e+00
   0.00000000e+00  1.61404919e-03 -1.65365579e-04  9.69764116e-01
   2.03127437e-02 -7.24255875e-04 -2.90319623e-05  1.38713729e-03
  -2.05260658e-03 -5.01451314e-04  1.08379209e-03  3.09191573e-03
   1.74857714e-03 -1.05315047e-03  6.88826189e-04 -1.02852716e-03
   6.04126697e-03  4.43377860e-03  4.96390052e-03  7.46950060e-03
   5.25402261e-03  6.91379880e-03  5.53043815e-03  3.90270726e-03
   7.05093423e-03  4.19304929e-03  4.45845830e-03  7.73620709e-03
   5.31330430e-03  5.77863566e-03  3.89789903e-02 -5.62376961e-04
   8.82985016e-06  5.02916651e-04  1.89398967e-04]]
```

Figure 47: Coefficients of Ridge model - training data.

	ACCURACY	MAE	MSE	RMSE
Ridge Model – Training data	**0.9443**	**0.1903**	**0.0560**	**0.2366**

Table 7: Performance Metric - Training data - Ridge.

```
MAE: 0.19035147780611125
MSE: 0.056001365183515166
RMSE: 0.23664607578304603
```

Figure 48: Code Result - Performance Metrics - Ridge model Training data

The Coefficients of the Ridge model for the test dataset is shown below.

```
Ridge model: [[ 5.21093821e-03 -6.62466704e-04 -3.47598100e-02  0.00000000e+00
  -4.11940815e-03 -7.43037840e-04  5.54383109e-03  0.00000000e+00
   0.00000000e+00 -3.26911847e-03 -3.80669027e-03  9.66795074e-01
   1.34224229e-02 -5.34650769e-03 -1.56541230e-03  3.51705753e-04
  -1.19720462e-03  3.53042034e-03 -7.00089191e-04  3.40149861e-03
  -2.25664777e-03  2.68595980e-03 -2.14110952e-03 -1.52830155e-03
   1.18469407e-03  3.85138985e-03  8.09610682e-03  1.19470332e-02
   3.47745930e-03  3.71419497e-03  1.20548144e-03  3.87903658e-03
   3.41837418e-03  3.60756362e-03  6.35772400e-03  1.57272554e-03
   1.98592762e-03  4.44475908e-03  3.86861472e-02  4.19896704e-03
   3.51314352e-03  2.09963545e-03  3.19635424e-03]]
```

Figure 49: Coefficients of Ridge model - test data

The Ridge model score for the test dataset is **0.9453**.

	ACCURACY	MAE	MSE	RMSE
Ridge Model – Test data	**0.9453**	**0.1880**	**0.0538**	**0.2320**

Table 8: Performance Metric - Testing data - Ridge.

```
MAE: 0.18806241471435312
MSE: 0.053858312384192086
RMSE: 0.23207393732212173
```

Figure 50: Code Result - Performance Metrics - Ridge model testing data

Lasso model has been built and trained using the training data and the performance is given below.

The coefficients of Lasso model on training data is shown below.

```
Lasso model: [ 0.   0.  -0.   0.  -0.  -0.
  0.   0.   0.   0.   0.   0.8710134
 -0.  -0.  -0.   0.  -0.  -0.
  0.   0.   0.  -0.   0.  -0.
 -0.   0.  -0.   0.   0.   0.
  0.  -0.   0.   0.  -0.   0.
  0.   0.   0.  -0.   0.   0.
 -0.  ]
```

Figure 51: Coefficients of Lasso model on training data.

The model score for the training data **is 0.9314**.

	ACCURACY	MAE	MSE	RMSE
Lasso Model – Train data	**0.9309**	**0.2084**	**0.0694**	**0.2635**

Table 9: Performance Metric - Train data - Lasso.

```
MAE: 0.2084928769282779
MSE: 0.06945937730425245
RMSE: 0.263551469933773
```

Figure 52: Code Result - Performance Metrics - Lasso model train data.

The coefficients of the Lasso model on test data is shown below.

```
Lasso model: [ 0.          0.         -0.          0.         -0.         -0.
  0.          0.          0.         -0.         -0.          0.86878281
 -0.         -0.          0.         -0.         -0.          0.
  0.          0.         -0.          0.         -0.         -0.
 -0.          0.          0.          0.          0.         -0.
 -0.         -0.          0.         -0.          0.          0.
  0.          0.          0.          0.         -0.          0.
  0.        ]
```

Figure 53: Coefficients of Lasso model - test data.

The Lasso model score for the test data is 0.9319.

	ACCURACY	MAE	MSE	RMSE
Lasso Model – Test data	**0.9319**	**0.2055**	**0.0670**	**0.2589**

Table 10: Performance Metric - Test data - Lasso.

```
MAE: 0.20555985385376999
MSE: 0.06704423533532894
RMSE: 0.2589290160166081 7
```

Figure 54: Code Result - Performance Metrics - Lasso model test data.

The above build regression model is in unregularized way. Now we can use the Polynomial Features from Sklearn Pre-processing to regularize the model.

Once the model has been regularized we can use the train data to train the Ridge and Lasso model.

The Ridge model score on the regularized model for the training data is **0.9423**.

The coefficients of the Ridge model for the training data is shown below.

```
Ridge model: [[ 0.00000000e+00  2.23961421e+01 -3.68271330e+01 -5.92716851e+02
    0.00000000e+00 -3.40399560e+01 -3.11283956e+01  6.16108323e+01
    0.00000000e+00  0.00000000e+00  2.23844876e+01  2.12731090e+00
    1.39566132e+04  1.15412785e+02 -7.58242487e+00  2.77188238e+01
   -2.15367717e+00 -3.41541033e+01  8.51887415e+00 -3.86842492e+00
   -8.42792916e-01  8.60010207e+00 -2.02443663e+01  1.45518878e+01
   -7.68332724e+00  2.33621889e+01  1.05646837e+01  1.47833094e+01
    3.75863614e+01  2.64035649e+01  4.22297262e+01  2.80059285e+01
    1.76014864e+01  3.61576451e+01  1.14057117e+01  1.05427372e+01
    4.16240258e+01  2.08540749e+01  2.26603035e+01  5.59721420e+02
   -9.70797748e+00  2.56288802e+00  7.45104716e+00 -5.95820900e+00
    9.91047436e+00 -6.21256132e+00  0.00000000e+00  6.55893904e+00
   -1.36100998e+00  1.04259102e+01  0.00000000e+00  0.00000000e+00
    3.26159792e+01  1.76317649e+00  2.35845973e+01  3.51952673e+01
    4.72717017e+01  1.02956769e+02  5.17110131e+01  3.28966105e+01
    2.20745166e+01  5.22901990e+01 -5.60719714e+00 -2.26417733e+00
    2.44605681e+01 -2.01460198e+01 -3.41773834e+01 -2.19748007e+01
   -5.37748588e+01 -3.15497920e+01  3.49184028e+01 -5.42533880e+01
    1.25256643e+01 -3.56425337e+01 -3.90838497e+01 -5.72893015e+01
```

Figure 55: Coefficients of Ridge Model train data – Regularized.

	ACCURACY	MAE	MSE	RMSE
Ridge Model – Train data - regularized	**0.9423**	**2780.501**	**11903472.264**	**3450.140**

Table 11: Performance Metric - Train data – Regularized - Ridge.

```
MAE: 2780.5019018802873
MSE: 11903472.264464037
RMSE: 3450.140905015915
```

Figure 56: Code Result - Performance Metrics - Ridge Model train data – Regularized

The coefficients of the Ridge model for the test data is shown below.

```
Ridge model: [[ 0.00000000e+00  6.22527501e+01 -5.05938258e+01 -5.36293537e+02
    0.00000000e+00 -5.52662686e+01 -1.74036171e+00  7.75411504e+01
    0.00000000e+00  0.00000000e+00 -5.45979493e+01 -6.23521468e+01
    1.39190868e+04  3.71671509e+01 -1.19717800e+02 -2.24528852e+01
    1.77828871e+01 -2.80562655e+01  5.85176790e+01 -6.60524453e+00
    1.00722862e+01  5.45725523e+00 -1.37937936e+00 -2.42457007e+01
   -1.96504527e+00 -1.47363719e+01  1.42791929e+01  4.17405755e+01
    6.68567587e+01  6.37150551e+00  1.69036392e+01 -1.87151924e+00
    1.41322334e+01 -1.32505656e+01  3.27564144e+00  3.53422580e+01
    9.02157704e+00  6.13481808e+00  2.96980051e+00  5.96549932e+02
    3.43267243e+01  6.03502417e+01  2.59443938e+01  1.03087607e+01
    1.09976215e+02 -6.28270171e+01  0.00000000e+00  1.37811873e+01
    3.30391604e+01 -4.29726107e+01  0.00000000e+00  0.00000000e+00
    2.51681835e+01 -6.72425637e+01  1.16267464e+00 -3.26539619e+00
    2.96296439e+01  2.25817789e+01  6.12999049e+01 -2.55411163e+01
    5.33830695e+01 -2.36891250e+01 -1.38307663e+01  1.04940306e+02
   -9.89081082e+01 -3.27228627e+01 -4.23254447e+01 -7.07159438e+00
    9.49141047e+01  2.70436200e+01  8.31997231e+01  8.86026017e+01
    5.07255729e+01  3.14353883e+01  9.75828322e-01  4.83553025e+01
```

Figure 57: Coefficients of Ridge Model test data – Regularized

The Ridge model score for the test data is 0.9531.

	ACCURACY	MAE	MSE	RMSE
Ridge Model – Test data - regularized	**0.9531**	**2499.232**	**9464254.403**	**3076.402**

Table 12: Performance Metric - Test data – Regularized - Ridge.

```
MAE: 2499.232300631816
MSE: 9464254.403259903
RMSE: 3076.40283501038
```

Figure 58: Code Result - Performance Metrics - Ridge Model test data – Regularized.

The Lasso model has been built and the model has been regularized using the Polynomial features.

The model is trained using the training dataset and the results are given below.

The Coefficients of the Lasso model for the training data is displayed below.

```
Lasso model: [ 0.00000000e+00  2.23853550e+01 -3.68079692e+01 -5.92661920e+02
  0.00000000e+00 -3.40251845e+01 -3.11267748e+01  6.16052463e+01
  0.00000000e+00  0.00000000e+00  2.23678080e+01  2.11556208e+00
  1.39569135e+04  1.15502941e+02 -7.55111814e+00  3.53123347e+01
  3.56551042e+00 -3.67474728e+01  1.25609776e+01 -1.54158963e+00
  7.41186186e-03  8.59210102e+00 -2.91111393e+01  1.15626105e+01
 -1.12628047e+01  3.45543196e+01  6.58753328e+00  1.73671724e+01
  5.84609669e+01  2.44815829e+01  3.63861151e+01  2.21303960e+01
  5.65244530e+00  3.72534040e+01  7.11361125e+00  1.42886887e+01
  5.16062910e+01  3.17845070e+01  4.33266971e+01  5.59694726e+02
 -1.36446120e+01  0.00000000e+00  8.68931688e+00 -3.04407957e+00
  9.89801459e+00 -6.20709976e+00  0.00000000e+00  6.54813773e+00
 -1.34547419e+00  1.04165173e+01  0.00000000e+00  0.00000000e+00
  3.25974102e+01  1.73694261e+00  2.35714168e+01  3.51828896e+01
  4.72395030e+01  1.02790083e+02  5.15940745e+01  3.28016677e+01
  2.19735686e+01  5.21803348e+01 -5.56123491e+00 -2.26523820e+00
  2.44656378e+01 -2.01160341e+01 -3.41580953e+01 -2.18854836e+01
 -5.36907116e+01 -3.14704391e+01  3.49841530e+01 -5.41693551e+01
  1.25882430e+01 -3.55641788e+01 -3.90038682e+01 -5.72080263e+01
```

Figure 59: Coefficients of Lasso Model train data – Regularized.

	ACCURACY	MAE	MSE	RMSE
Lasso Model – Train data - regularized	**0.9423**	**2780.452**	**11902880.600**	**3450.055**

Table 13: Performance Metric - Train Data – Regularized - Lasso.

```
MAE: 2780.4523378433387
MSE: 11902880.600722285
RMSE: 3450.055159084023
```

Figure 60: Code Result - Performance Metrics - Lasso model train data - Regularized.

The Coefficients of the Lasso model on test data is displayed below.

```
Lasso model: [ 0.00000000e+00  6.22608569e+01 -5.05803454e+01 -5.36178837e+02
  0.00000000e+00 -5.52492958e+01 -1.73621845e+00  7.75116573e+01
  0.00000000e+00  0.00000000e+00 -5.46002056e+01 -6.23165957e+01
  1.39198405e+04  3.73899111e+01 -1.19646212e+02 -4.65722906e+01
  1.08818250e+01 -2.04437808e+01  7.28368356e+01 -2.38327661e+01
  3.82112053e+01  5.43249773e+00  1.42737260e+01 -1.95860749e+01
 -1.17112372e+01 -2.10268487e+01  1.27699404e+00  8.16731761e+01
  1.29545226e+02  1.62725497e+01  4.56902465e+00 -7.68003888e+00
  1.01830086e+01 -9.57275038e+00  1.18010534e+01  5.13225751e+01
  4.54278831e+00  1.63720757e+01  2.04148889e+01  5.96521737e+02
  0.00000000e+00  3.79704935e+01  3.14707133e+01  2.33336110e+01
  1.09955918e+02 -6.28004791e+01  0.00000000e+00  1.37582482e+01
  3.30145617e+01 -4.29529530e+01  0.00000000e+00  0.00000000e+00
  2.51401100e+01 -6.72080380e+01  1.16629399e+00 -3.25690464e+00
  2.95855608e+01  2.24842513e+01  6.12661745e+01 -2.55419353e+01
  5.33669697e+01 -2.37055296e+01 -1.37870846e+01  1.04903358e+02
 -9.88798237e+01 -3.27027121e+01 -4.22981564e+01 -7.12646306e+00
  9.48312539e+01  2.69660092e+01  8.31160526e+01  8.85069640e+01
  5.06438970e+01  3.13522309e+01  8.97953377e-01  4.82752367e+01
```

Figure 61: Coefficients of Lasso model test data - Regularized.

The Lasso model score for the test data is **0.9531**.

	ACCURACY	MAE	MSE	RMSE
Lasso Model – Test data - regularized	**0.9531**	**2499.248**	**9464254.721**	**3076.402**

Table 14: Performance Metric - Test data – Regularized - Lasso.

```
MAE: 2499.2482550630975
MSE: 9464254.721786508
RMSE: 3076.402886779706
```

Figure 62: Code Result - Performance Metrics - Lasso model test data - Regularized.

- **RANDOM FOREST REGRESSOR**:

  A Random Forest is an ensemble technique capable of performing both regression and classification tasks with the use of multiple decision trees.

  Random Forest Regressor is imported from the library of Sklearn of Ensemble Techniques.

  The model has been built and the trained using the training dataset.

The model score for the training dataset is **0.9933**.

	ACCURACY	MAE	MSE	RMSE
Random Forest – Train data	**0.9933**	**927.56**	**1369948.30.**	**1170.44**

Table 15: Performance Metric - Train data - RF

```
MAE: 927.563248
MSE: 1369948.3055462856
RMSE: 1170.4479080874492
```

Figure 63: Code Result - Performance Metrics - Train data.

Here we can see that the model has been overfitted and we can rectify that by using GridSearchCV and find out the best parameters to build the model and train with the train data.

The best parameters have been found out by GridSearchCV and the results are shown below.

```
{'max_depth': 10, 'max_features': 6, 'min_samples_leaf': 3, 'min_samples_split': 30, 'n_estimators': 500}
```

Figure 64: Best Params - Random Forest Model.

The model has been build using the best parameters and the model is trained using the training dataset.

	ACCURACY	MAE	MSE	RMSE
Random Forest – Train data – Best params	**0.8451**	**4378.624**	**31975701.717**	**5654.706**

Table 16: Performance Metric - Train data with best params - RF.

```
MAE: 4378.624527454361

MSE: 31975701.717022922

RMSE: 5654.706156558705
```

Figure 65: Code Result - Performance Metrics - train data - Best Params.

Now the Random Forest model has been tested using the test dataset and the performance is given below.

The model score for the test dataset is **0.9512**.

	ACCURACY	MAE	MSE	RMSE
Random Forest – Test data	**0.9512**	**2509.442**	**9846790.533**	**3137.959**

Table 17: Performance Metric - Test data - RF

```
MAE: 2509.442656

MSE: 9846790.533115413

RMSE: 3137.9596130472128
```

Figure 66: Code Result - Performance Metrics - Random Forest train data.

The model is now built using the Best parameters and the model is trained and performance is shown. Now we can use the model to check the performance on test data.

The model score for the test dataset is **0.8302**.

	ACCURACY	MAE	MSE	RMSE
Random Forest – Test data – Best Params	**0.8302**	**4533.151**	**34312367.861**	**5857.675**

Table 18: Performance Metric - Test data - best params - RF

```
MAE: 4533.151249501726

MSE: 34312367.86159958

RMSE: 5857.675977860125
```

Figure 67: Code Result - Performance Metrics - test data - Best Params.

- **ARTIFICIAL NUERAL NETWORK (MULTILAYER PERCEPTRON REGRESSION)**:

  Regression ANNs predict an output variable as a function of the inputs. The input features (independent variables) can be categorical or numeric types; however, for regression ANNs, we require a numeric dependent variable.

  MLP Regressor can be imported from the library of Sklearn of Neural Network and the model can be built.

  Before building the model we have to scale the train and test dataset since the model is sensitive to range of values. If we don't normalize our inputs between (0,1) or (-1,1) we could not equally distribute importance of each input thus naturally large values become dominant according to less values during ANN training.

  Once the model is built, we can train the model with the training dataset and the results are shown below.

  The model score for the training dataset is 0.9497.

	ACCURACY	MAE	MSE	RMSE
ANN – Train data	**0.9497**	**2602.005**	**10367713.220**	**3219.893**

Table 19: Performance Metric - Train data - ANN

```
MAE: 2602.005183631221

MSE: 10367713.220178366

RMSE: 3219.893355404549
```

Figure 68: Code Result - Performance Metrics - Random Forest train data.

Now we can use GridSearchCV in order to maximize the accuracy of the model by getting the best parameters. The Grid Search has been performed and the best parameters are shown below.

```
{'activation': 'relu', 'hidden_layer_sizes': 100, 'solver': 'adam'}
```

Figure 69: Best Parameters – ANN.

After getting the best parameters from Grid Search, we can implement the parameters to build a new model and we can train the model using the training dataset.

Once the model is trained we can check the model score for the training dataset.

The model Score for the training dataset is 0.9497.

	ACCURACY	MAE	MSE	RMSE
ANN – Train data – Best Params	**0.9497**	**2602.005**	**10367713.220**	**3219.893**

Table 20: Performance Metric - Train data - best params - ANN

```
MAE: 2602.005183631221

MSE: 10367713.220178366

RMSE: 3219.893355404549
```

Figure 70: Code Result - Performance Metrics - Best Params.

ANN model has been built and the model has been trained and its performance for the training data has been measured. Now we can use the model to measure the performance on test data.

The model score for the test data is 0.9458.

	ACCURACY	MAE	MSE	RMSE
ANN – Test data	**0.9458**	**2693.330**	**10934579.305**	**3306.747**

Table 21: Performance Metric - Test data - ANN

Figure 71: Code Result - Performance Metrics - ANN Model test data.

The model has been built by taking the best parameters which is observed from Grid Search and the model has been tested with the test data and the observations are given below.

The model score for the test data is 0.9458.

	ACCURACY	MAE	MSE	RMSE
ANN – Test data – Best Params	**0.9458**	**2693.330**	**10934579.305**	**3306.747**

Table 22: Performance Metric - test data - best params - ANN

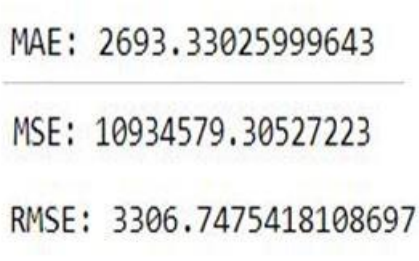

Figure 72: Performance Metrics - ANN Model test data - Best Params.

- ## EFFORT TO IMPROVE MODEL PERFORMANCE.

Ensemble modelling techniques are Bagging and Boosting. Here after building all the models and training and testing them with the train and test dataset we can see that the Model score for the Decision tree is 1.0 which means the model is overfitted. Here we can use Bagging technique to regularize the decision tree in order to get the maximum results.

- ### BAGGING:

Bagging is imported from Sklearn of Ensemble technique and the model is created with the decision tree as its base parameter.

The model is trained and the model score for the train data is 0.9604.

	ACCURACY	MAE	MSE	RMSE
Bagging Model – Train data	**0.9604**	**2295.326**	**8168464.910**	**2858.052**

Table 23: Performance Metric - train data - Bagging model

```
MAE: 2295.326559478785

MSE: 8168464.91083502

RMSE: 2858.052643118216
```

Figure 73: Code Result - Performance Metrics - Bagging Model train data.

Now we can test the model with the test data.

The model score for the test data is 0.9533.

	ACCURACY	MAE	MSE	RMSE
Bagging Model – Test data	**0.9533**	**2472.701**	**9423517.010**	**3069.774**

Table 24: Performance Metric - test data - Bagging model

```
MAE: 2472.7010008940447

MSE: 9423517.01091113

RMSE: 3069.7747492138783
```

Figure 74: Performance Metrics - Bagging model test data.

- **BOOSTING**:

Here we can able to see that after the model built using the best parameters for the random forest Regressor we can see that the model score is less compared to other models.

Here we can able to perform ADA Boosting to increase its performance.

ADA Boosting is imported from the library of Sklearn of Ensemble Technique and the Random Forest Regressor is given as the base estimator for the Boosting Regressor.

Once the model is built, we can able to train the model using the training dataset and the model accuracy is 0.8936.

	ACCURACY	MAE	MSE	RMSE
Boosting Model – Train data	**0.8936**	**3763.903**	**21961601.557**	**4686.320**

Table 25: Performance Metric - train data - Boosting

```
MAE: 3763.90340590268

MSE: 21961601.557599172

RMSE: 4686.320684460163
```

Figure 75: Code Result - Performance Metrics - Boosting model train data.

Now we can use the model to test the performance on test data and the results are given below.

The model score for the test data is 0.8743.

	ACCURACY	MAE	MSE	RMSE
Boosting Model – Test data	**0.8743**	**3986.036**	**25386411.480**	**5038.492**

Table 26: Performance Metric - test data - Boosting.

```
MAE: 3986.036496701663

MSE: 25386411.4804155

RMSE: 5038.492977112849
```

Figure 76: Code Result - Performance Metrics - Boosting model test data.

- **MODEL TUNING**:

Here we are using Grid Search CV as the model tuning measure for the Decision Tree Regressor, Random Forest Regressor and ANN Regressor.

By using GridSearchCV we are getting the best parameters in which the model can be built and trained and tested to get the most optimum results.

The Model tuning measures has been performed and the results are shown on model building for both train and test dataset.

The results obtained from Grid Search CV for Decision tree, Random Forest and ANN are displayed below.

GridSearchCV for Decision Tree:

```
{'max_depth': 10, 'min_samples_leaf': 30, 'min_samples_split': 15}
```

Figure 77: GridSearchCV for Decision Tree.

GridSearchCV for Random Forest:

```
GridSearchCV(cv=3, estimator=RandomForestRegressor(random_state=123),
             param_grid={'max_depth': [7, 10], 'max_features': [4, 6],
                         'min_samples_leaf': [3, 15, 30],
                         'min_samples_split': [30, 50, 100],
                         'n_estimators': [300, 500]})
```

Figure 78: GridSearchCV for Random Forest.

The best parameters for Random forest Regressor from Grid Search CV is shown below.

```
{'max_depth': 10, 'max_features': 6, 'min_samples_leaf': 3, 'min_samples_split': 30, 'n_estimators': 500}
```

Figure 79: Best Params Random Forest.

GridSearchCV for ANN:

```
GridSearchCV(cv=3, estimator=MLPRegressor(max_iter=500, random_state=123),
             param_grid={'activation': ['relu'], 'hidden_layer_sizes': [100],
                         'solver': ['adam']})
```

Figure 80: GridSearchCV for ANN.

- **VARIANCE INFLATION FACTOR**:

Variance inflation factor (VIF) is a measure of the amount of multi collinearity in a set of multiple regression variables.

VIF is calculated for the given dataset and the result is given below.

```
applicant_id ---> 6.719107803381725
years_of_insurance_with_us ---> 3.676119291715797
regular_checkup_lasy_year ---> 1.540813382065002
adventure_sports ---> nan
visited_doctor_last_1_year ---> 8.35123141342952
daily_avg_steps ---> 27.467122820470536
age ---> 8.478415109773302
heart_decs_history ---> nan
other_major_decs_history ---> nan
avg_glucose_level ---> 7.873523832968523
bmi ---> 22.207821925969643
weight ---> 46.177881685133165
weight_change_in_last_one_year ---> 3.411773323666729
fat_percentage ---> 13.8159238091584
Occupation_Salried ---> 5.135180256683086
Occupation_Student ---> 4.437172029891279
cholesterol_level_150 to 175 ---> 3.3717580946172485
cholesterol_level_175 to 200 ---> 2.7968879321496893
cholesterol_level_200 to 225 ---> 2.7926265171622164
cholesterol_level_225 to 250 ---> 1.8823012452235164
Gender_Male ---> 3.5991856282548276
smoking_status_formerly smoked ---> 1.7628592894145747
smoking_status_never smoked ---> 2.4635555086932523
smoking_status_smokes ---> 1.6450836143964878
Location_Bangalore ---> 1.9806467048841416
Location_Bhubaneswar ---> 1.9590101733575072
Location_Chennai ---> 1.9341529303209384
Location_Delhi ---> 1.9450399208861815
Location_Guwahati ---> 1.9436588641557704
Location_Jaipur ---> 1.9564014643647143
Location_Kanpur ---> 1.9315194503311077
Location_Kolkata ---> 1.9098377012015972
Location_Lucknow ---> 1.917759080606349
Location_Mangalore ---> 1.950648771714848
Location_Mumbai ---> 1.9340831068150488
Location_Nagpur ---> 1.9278742571724095
Location_Pune ---> 1.9098366565627847
Location_Surat ---> 1.8874593647522993
covered_by_any_other_company_Y ---> 1.5540748123469843
Alcohol_No ---> 4.202869775571303
Alcohol_Rare ---> 6.011231871563572
exercise_Moderate ---> 3.7785192241174013
exercise_No ---> 1.992802176232025
```

Figure 81: VIF for the dataset.

There are some variables which has high multi collinearity with the other variables. Having variables with high multi collinearity would affect the performance of the model. So we are dropping the variables and we are building the Linear Regression model using Statsmodels and the results are given below.

```
Intercept                          3.579177e+04
years_of_insurance_with_us        -1.323210e+02
regular_checkup_lasy_year         -2.883826e+03
adventure_sports                  -1.895682e-10
visited_doctor_last_1_year         1.249746e+02
age                               -8.679229e-01
heart_decs_history                -3.976200e-11
other_major_decs_history          -6.947669e-13
avg_glucose_level                 -1.029097e+00
weight_change_in_last_one_year    -2.899587e+03
Occupation_Salried                -4.669387e+02
Occupation_Student                -2.230288e+02
cholesterol_level_150_to_175      -1.465924e+02
cholesterol_level_175_to_200      -7.706066e+02
cholesterol_level_200_to_225       1.576730e+02
cholesterol_level_225_to_250      -3.035170e+02
Gender_Male                        1.213425e+02
smoking_status_formerly_smoked    -2.163470e+02
smoking_status_never_smoked       -1.202282e+02
smoking_status_smokes             -2.758276e+02
Location_Bangalore                 5.227180e+02
Location_Bhubaneswar               2.551349e+02
Location_Chennai                   3.768618e+02
Location_Delhi                     6.237328e+02
Location_Guwahati                  6.787384e+02
Location_Jaipur                    4.293810e+01
Location_Kanpur                   -9.358714e+01
Location_Kolkata                   7.631722e+02
Location_Lucknow                   7.213152e+01
Location_Mangalore                 6.373616e+02
Location_Mumbai                    5.127472e+02
Location_Nagpur                   -2.082709e+02
Location_Pune                     -4.461082e+02
Location_Surat                     3.244162e+00
covered_by_any_other_company_Y     2.908617e+03
Alcohol_No                         2.908485e+02
Alcohol_Rare                       2.251679e+02
exercise_Moderate                  8.477239e+01
exercise_No                       -4.600860e+02
dtype: float64
```

Figure 82: Parameters for Linear Regression model after VIF.

The summary of the model after VIF is given below.

OLS Regression Results

Dep. Variable:	insurance_cost	R-squared:	0.162
Model:	OLS	Adj. R-squared:	0.160
Method:	Least Squares	F-statistic:	96.29
Date:	Fri, 06 May 2022	Prob (F-statistic):	0.00
Time:	14:11:11	Log-Likelihood:	-1.9081e+05
No. Observations:	17500	AIC:	3.817e+05
Df Residuals:	17464	BIC:	3.820e+05
Df Model:	35		
Covariance Type:	nonrobust		

Figure 83: Summary of the model.

The parameters for the model in test data after VIF are given below.

Intercept	3.410731e+04	Location_Bangalore	4.364571e+02
years_of_insurance_with_us	-3.921224e+01	Location_Bhubaneswar	1.949262e+03
regular_checkup_lasy_year	-3.086141e+03	Location_Chennai	1.730612e+03
adventure_sports	1.082907e-09	Location_Delhi	7.954169e+02
visited_doctor_last_1_year	-1.389928e+02	Location_Guwahati	8.937092e+02
age	1.629591e+01	Location_Jaipur	8.260172e+02
heart_decs_history	-1.747625e-10	Location_Kanpur	6.533436e+02
other_major_decs_history	-1.952740e-11	Location_Kolkata	1.495710e+03
avg_glucose_level	4.008980e-01	Location_Lucknow	9.634925e+02
weight_change_in_last_one_year	-2.848690e+03	Location_Mangalore	7.314288e+02
Occupation_Salried	-3.183561e+02	Location_Mumbai	1.681614e+03
Occupation_Student	-3.166286e+02	Location_Nagpur	1.507131e+03
cholesterol_level_150_to_175	-7.536745e+02	Location_Pune	1.226099e+03
cholesterol_level_175_to_200	-8.006172e+02	Location_Surat	6.957491e+02
cholesterol_level_200_to_225	-5.380937e+02	covered_by_any_other_company_Y	2.525125e+03
cholesterol_level_225_to_250	-1.233383e+02	Alcohol_No	8.929636e+02
Gender_Male	-1.689325e+01	Alcohol_Rare	4.320479e+02
smoking_status_formerly_smoked	-3.763162e+02	exercise_Moderate	6.741120e+02
smoking_status_never_smoked	-1.646647e+02	exercise_No	3.405863e+02
smoking_status_smokes	-4.324308e+02	dtype: float64	

Figure 84: Parameters of the model test data after VIF.

OLS Regression Results

Dep. Variable:	insurance_cost	R-squared:	0.162
Model:	OLS	Adj. R-squared:	0.158
Method:	Least Squares	F-statistic:	41.17
Date:	Fri, 06 May 2022	Prob (F-statistic):	8.23e-255
Time:	14:13:00	Log-Likelihood:	-81696.
No. Observations:	7500	AIC:	1.635e+05
Df Residuals:	7464	BIC:	1.637e+05
Df Model:	35		
Covariance Type:	nonrobust		

Figure 85: Summary of the data after VIF - test data.

Here we can see that the R Squared and Adjusted R Squared of the model for both train and test data is very low.

This is because of the variables that are dropped after VIF. By dropping one variable at a time we can see that weight carries high variance towards the dataset. By dropping it the model loses its accuracy and therefore we are dropping the other three variables which will reduce the complexity of the data and reduce the run time of the model.

By dropping the other three variables, we are building the model and testing it with the test data and the summary of the data is given below.

OLS Regression Results

Dep. Variable:	insurance_cost	R-squared:	0.945
Model:	OLS	Adj. R-squared:	0.945
Method:	Least Squares	F-statistic:	3579.
Date:	Fri, 06 May 2022	Prob (F-statistic):	0.00
Time:	21:33:21	Log-Likelihood:	-71464.
No. Observations:	7500	AIC:	1.430e+05
Df Residuals:	7463	BIC:	1.433e+05
Df Model:	36		
Covariance Type:	nonrobust		

Figure 86: Summary of the data after VIF - dropping 3 variables.

5. MODEL VALIDATION

- ### HOW WAS THE MODEL VALIDATED? JUST ACCURACY, OR ANYTHING ELSE TOO?

The model is validated not just by using the accuracy but also the performance metrics.

The different performance metrics are

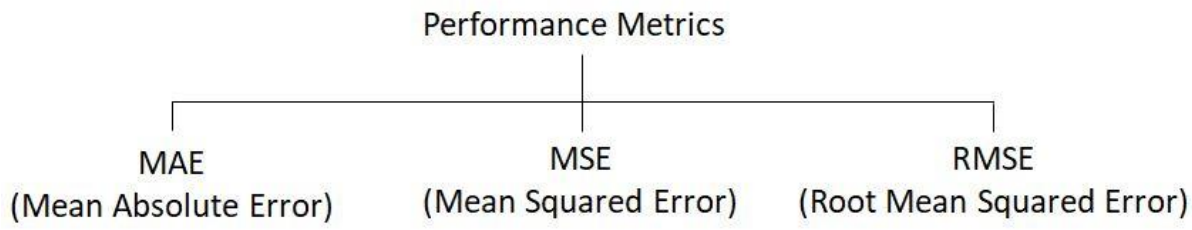

Figure 87: Performance Metrics.

- **MEAN ABSOLUTE ERROR**:

The MAE measures the average magnitude of the errors in a set of forecasts, without considering their direction.

- **MEAN SQUARED ERROR**:

In statistics, the mean squared error (MSE) of an estimator measures the average of the squares of the errors that is, the average squared difference between the estimated values and the actual value.

- **ROOT MEAN SQUARED ERROR**:

Root Mean Square Error (RMSE) is a frequently used measure of the differences between values predicted by a model or an estimator and the values observed.

The performance metrics for the entire model with train and test data is given below.

	Model Score		MAE		MSE		RMSE	
	Train	Test	Train	Test	Train	Test	Train	Test
Linear Regression	0.9447	0.9449	2721.016	2709.381	11408227.133	11131896.631	3377.606	3336.449
Decision Tree Regressor	0.9601	0.9496	2286.663	2549.037	8233782.850	10176095.091	2869.456	3189.999
Ridge Regressor	0.9423	0.9531	2780.501	2499.232	11903472.264	9464254.403	3450.140	3076.402
Lasso Regressor	0.9423	0.9531	2780.452	2499.248	11902880.600	9464254.721	3450.055	3076.402
Random Forest Regressor	0.8451	0.8302	4378.624	4533.151	31975701.717	34312367.861	5654.706	5857.675
ANN Regressor	0.9497	0.9458	2602.005	2693.330	10367713.220	10934579.305	3219.893	3306.747

Figure 88: Performance Metrics for the entire model with train and test data.

6. FINAL INTERPRETATION / RECOMMENDATION.

- ## DETAILED RECOMMENDATIONS FOR THE MANAGEMENT/CLIENT BASED ON THE ANALYSIS DONE.

  - Root Mean Squared Error is the best metric measure that can be used to measure the performance of the models.
  - ANN regression model will be the optimum model in order to optimize the insurance cost as the difference between the train RMSE and test RMSE is small and the model is capable of handling large dataset.
  - ANN Regressor model is capable of handling large and complex datasets which will make a huge impact in optimizing the insurance cost.
  - Customers with smoking habits and alcohol habits are easily prone to diseases and major heart diseases. So we can come up with new insurance policies for the people with smoking habits and alcohol consumption by taking higher prices.
  - Customers who are KM_Cluster 3 are claiming the most insurance. The company can draft a new policy for customers under KM_Cluster 3.

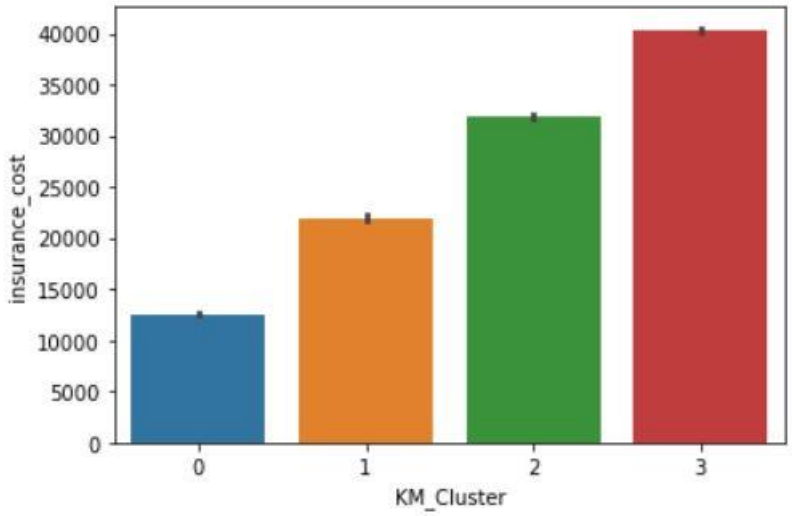

Figure 89: KM_Clusters.

  - Females are not taking insurance compared to Male. So the company can bring up some new policies which can attract females in getting insurance.
  - Most of the customers to the company are students. The company can come up with special policies for students which will increase the sales and also benefit the company.
  - Also from the analysis we can say that as the weight increases the insurance cost for the individual also increases which can be confirmed from the scatter plot shown below.

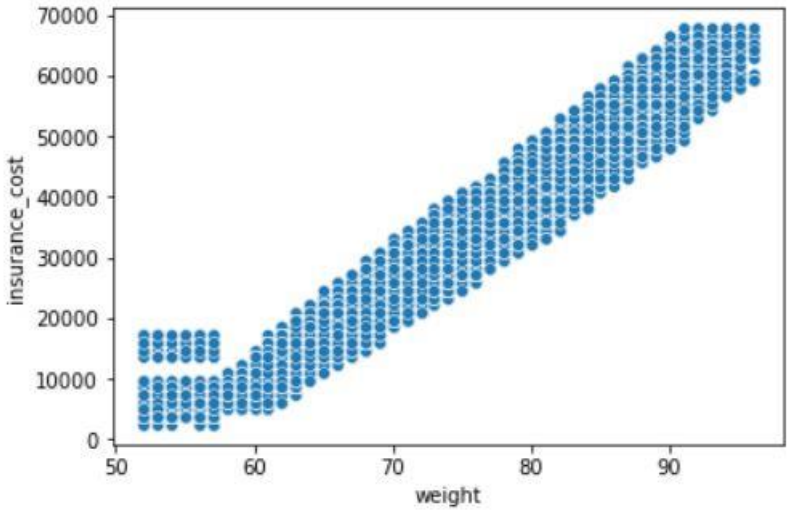

Figure 90: Scatter plot between Weight VS Insurance Cost.

- Most of the customers for the company are having average BMI of 30.8 which means that more than 50% of the customers are obese. Also weight is directly connected with BMI.
- Hence the company should draft policies taking BMI and weight into consideration by making the customers chose premium policies as they are easily prone to diabetic and heart diseases.