HEALTH INSURANCE COST PREDICTOR

A Project report
Submitted in partial fulfillment of requirements for the award of degree

Bachelor of Technology
In
Computer Science and Engineering

Submitted By

NAME	ROLL NO.
Anoushka Ghosh	11900119057
Didhiti Raj Chakraborty	11900120096
Pawan Kumar Gupta	11900119011
Nayan Kumar Sinha	11900120101

Under the Project Guidance of

Mr. Mahendra Datta

SILIGURI INSTITUTE OF TECHNOLOGY
Siliguri, West Bengal, India

In Association with

ARDENT
COMPUTECH PVT. LTD.
High-End Technology Training and Project

TITLE OF THE PROJECT: HEALTH INSURANCE COST PREDICTOR

PROJECT MEMBERS: Anoushka Ghosh,
Didhiti Raj Chakraborty,
Pawan Kumar Gupta,
Nayan Kumar Sinha

NAME OF THE GUIDE: Mr. Mahendra Datta

PROJECT VERSION CONTROL:

Version	Primary Authors	Description of version	Date of completion
Final	Anoushka Ghosh, Didhiti Raj Chakraborty, Pawan Kumar Sah, Nayan Kumar Sinha.	Project Report	28-08-2021

Anoushka Ghosh
Didhiti Raj Chakraborty
Pawan Kumar Gupta
Nayan Kr. Sinha.

Signature of the Students

Signature of the Supervisor

Date: 28-08-2021

Date:

For Office Use Only

M. Datta

Approved Not Approved

MAHENDRA DATTA
Project proposal / Evaluator

SILIGURI INSTITUTE OF TECHNOLOGY
A Unit of Techno India Group
Affiliated to MAKAUT, WBSCT, VE & SD
Approved by AICTE

DECLARATION

We hereby declare that the project work being presented in the project proposal entitled **“HEALTH INSURANCE COST PREDICTOR”** in partial fulfilment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE AND ENGINEERING** in association with **ARDENT COMPUTECH PVT LTD, SALTLAKE, KOLKATA, WEST BENGAL**, is an authentic work carried out under the guidance of **MR. MAHENDRA DATTA**. The matter embodied in this project work has not been submitted elsewhere for the award of any degree of our knowledge and belief.

Date:	28-08-2021
Name of the Students:	Anoushka Ghosh, Didhiti Raj Chakraborty, Pawan Kumar Gupta, Nayan Kumar Sinha
Signature of the Students:	Anoushka Ghosh Didhiti Raj Chakraborty Pawan Kumar Gupta Nayan Kr. Sinha.

ARDENT
COMPUTECH PVT. LTD.
High-End Technology Training and Project

CERTIFICATE

This is to certify that this proposal of minor project entitled **“HEALTH INSURANCE COST PREDICTOR”** is a record of bonafide work, carried out by **Anoushka Ghosh, Didhiti Raj Chakraborty, Pawan Kumar Gupta** and **Nayan Kumar Sinha** under my guidance at **ARDENT COMPUTECH PVT LTD**. In my opinion, the report in its present form is in partial fulfilment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE AND ENGINEERING** and as per regulations of the **ARDENT COMPUTECH PRIVATE LIMITED**. To the best of my knowledge, the results embodied in this report, are original in nature and worthy of incorporation in the present version of the report.

Guide/Supervisor

M. Datta

MR. MAHENDRA DATTA
Project Engineer
ARDENT COMPUTECH PVT. LTD.

ACKNOWLEDGEMENT

Success of any project depends largely on the encouragement and guidelines of many others. I take this sincere opportunity to express my gratitude to the people who have been instrumental in the successful completion of this project work.

I would like to show our greatest appreciation to **Mr. Mahendra Datta**, Project Engineer at **ARDENT COMPUTECH PRIVATE LIMITED**, Kolkata. I always feel motivated and encouraged every time by his valuable advice and constant inspiration; without his encouragement and guidance this project would not have materialized.

Words are inadequate in offering our thanks to the other trainees, project assistants and other members at **Ardent Computech Pvt. Ltd.** for their encouragement and cooperation in carrying out this project work. The guidance and support received from all the members and who are contributing to this project, was vital for the success of this project.

Anoushka Ghosh	**11900119057**
Didhiti Raj Chakraborty	**11900120096**
Pawan Kumar Gupta	**11900119011**
Nayan Kumar Sinha	**11900120101**

Table of Contents

Abstract

In this project, we analyze the personal health data to predict insurance amount for individuals on the basis of the dataset with seven attributes that we have. Here five regression models naming Linear Regression, Random Forest Regression, Decision Tree Regression, Support Vector Regression, K Nearest Neighbor Regression have been used to compare and contrast the performance of these algorithms. The dataset was utilized to train the models, and the results of that training were used to make some predictions. The model was then tested and verified by comparing the projected quantity to the actual data. After using five regression models, we found Random Forest Regression is performing better (with maximum accuracy) than the other regression models.

Keywords: Machine Learning, Regression models (KNN, SVM, Decision Tree, Random Forest, Linear.

CHAPTER 1

INTRODUCTION

The purpose of this project is to provide people an indication of how much they'll need based on their current health status. After that, they can get touch with any health insurance company and their plans and benefits while keeping in mind the projected budget from our project. This can assist a person in concentrating more on the health side of an insurance policy rather than the ineffective part.

Health insurance is a necessity nowadays, and almost every individual is linked with a government or private health insurance company. Factors determining the amount of insurance vary from company to company. Also people in rural areas are unaware of the fact that the government of India provide free health insurance to those below poverty line. It is very complex method and some rural people either buy some private health insurance or do not invest money in health insurance at all. Apart from this people can be fooled easily about the amount of the insurance and may unnecessarily buy some expensive health insurance.

Our study does not provide the exact amount required by any health insurance provider, but it does provide a reasonable estimate of the sum involved with an individual's own health insurance.

Prediction is premature and does not apply to any specific organization, so it should not be the sole criterion for choosing a health insurance policy. Predicting the amount of health insurance, early on can help you think about how much you'll need. Where a person can assure that the quantity he or she chooses is appropriate. It might also give you an idea of how to get more out of your health insurance.

CHAPTER 2

MACHINE LEARNING

2.1 What is Machine Learning?

Machine Learning is the science of getting computers to learn and act like humans do, and improve their learning over time in autonomous fashion, by feeding them data and information in the form of observations and real-world interactions.

It can be defined as the process of teaching a computer system which allows it to make accurate predictions after the data is fed. However, training has to be done first with the data associated. By filtering and various machine learning models accuracy can be improved.

2.2 Types of Machine Learning.

1. Supervised Learning.

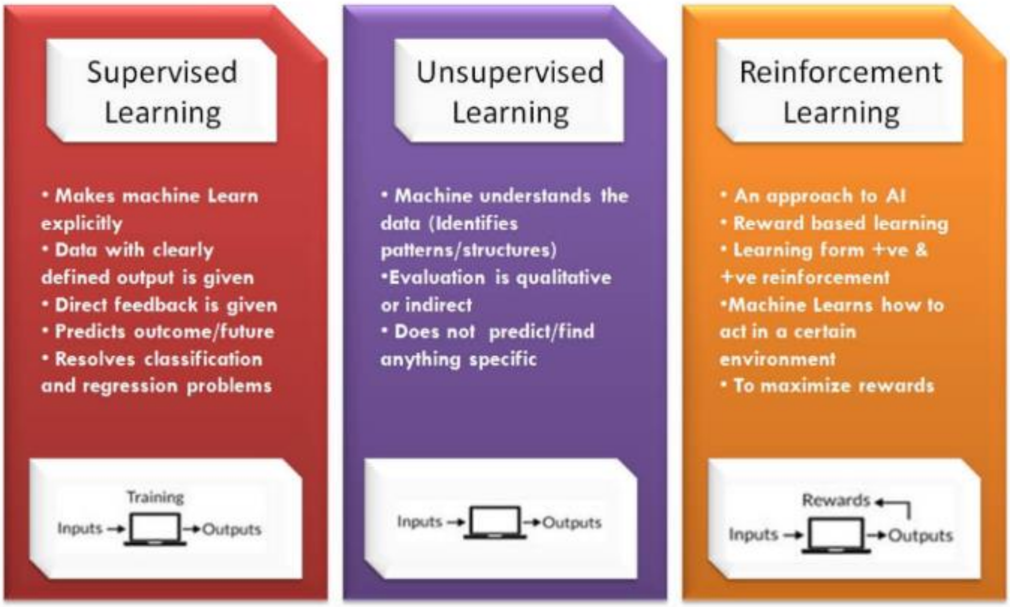
Supervised learning algorithms create a mathematical model according to a set of data that contains both the inputs and the desired outputs. Usually a random part of data is selected from the complete dataset known as training data, or in other words a set of training examples. Training data has one or more inputs and a desired output, called as a supervisory signal. What's happening in the mathematical model is each training dataset is represented by an array or vector, known as a feature vector. A matrix is used for the representation of training data. Supervised learning algorithms learn from a model containing function that can be used to predict the output from the new inputs through iterative optimization of an objective function. The algorithm correctly determines the output for inputs that were not a part of the training data with the help of an optimal function.

2. Unsupervised Learning.

In this learning, algorithms take a set of data that contains only inputs, and find structure in the data, like grouping or clustering of data points. Test data that has not been labeled, classified or categorized helps the algorithm to learn from it. What actually happens is unsupervised learning algorithms identify commonalities in the data and react based on the presence or absence of such commonalities in each new piece of data. The main application of unsupervised learning is density estimation in statistics. Though unsupervised learning, encompasses other domains involving summarizing and explaining data features also.

3. Reinforcement Learning.

Reinforcement learning is class of machine learning which is concerned with how software agents ought to make actions in an environment. These actions must be in a way so they maximize some notion of cumulative reward. Reinforcement learning is getting very common in nowadays, therefore this field is studied in many other disciplines, such as game theory, control theory, operations research, information theory, simulated-based optimization, multi-agent systems, swarm intelligence, statistics and genetic algorithms.



Supervised Learning

- Makes machine Learn explicitly
- Data with clearly defined output is given
- Direct feedback is given
- Predicts outcome/future
- Resolves classification and regression problems

Unsupervised Learning

- Machine understands the data (Identifies patterns/structures)
- Evaluation is qualitative or indirect
- Does not predict/find anything specific

Reinforcement Learning

- An approach to AI
- Reward based learning
- Learning form +ve & +ve reinforcement
- Machine Learns how to act in a certain environment
- To maximize rewards

Fig: 2.1 *Types of Machine Learning*

2.3 Common Machine Learning Algorithms.

Here is the list of commonly used machine learning algorithms. These algorithms can be applied to almost any data problem:

a) Linear Regression
b) Logistic Regression
c) Decision Tree
d) Support Vector Machine (SVM)
e) Naive Bayes
f) K-Nearest Neighbors (KNN)
g) Random Forest

a) Linear Regression:

It is used to estimate real values (cost of houses, number of calls, total sales etc.) based on continuous variable(s). Here, we establish relationship between independent and dependent variables by fitting a best line. This best fit line is known as regression line and represented by a linear equation Y= a *X + b.

The best way to understand linear regression is to relive this experience of childhood. Let us say, you ask a child in fifth grade to arrange people in his class by increasing order of weight, without asking them their weights! What do you think the child will do? He / she would likely look (visually analyze) at the height and build of people and arrange them using a combination of these visible parameters. This is linear regression in real life! The child has actually figured out that height and build would be correlated to the weight by a relationship, which looks like the equation above.

In this equation:

- Y – Dependent Variable
- a – Slope
- X – Independent variable
- b – Intercept

These coefficients a and b are derived based on minimizing the sum of squared difference of distance between data points and regression line.

Look at the below example. Here we have identified the best fit line having linear equation y=0.2811x+13.9. Now using this equation, we can find the weight, knowing the height of a person.

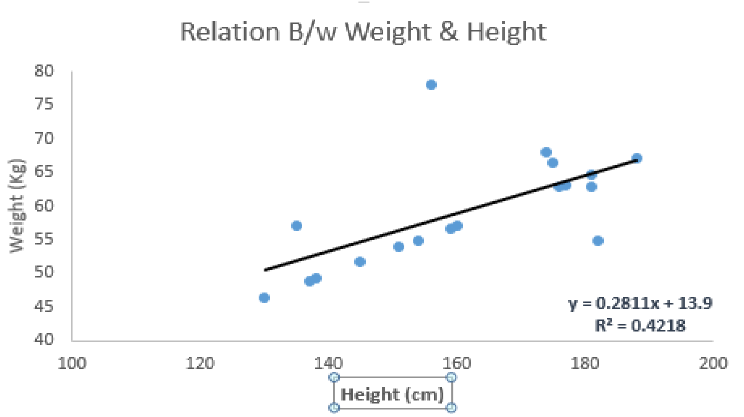


***Fig: 2.2** Linear Regression*

b) Logistic Regression:

Logistic regression, despite its name, is a classification model rather than regression model. Logistic regression is a simple and more efficient method for binary and linear classification problems. It is a classification model, which is very easy to realize and achieves very good performance with linearly separable classes. It is an extensively employed algorithm for classification in industry. Scikit-learn has a highly optimized version of logistic regression implementation, which supports multiclass classification task.

A solution for classification is logistic regression. Instead of fitting a straight line or hyperplane, the logistic regression model uses the logistic function to squeeze the output of a linear equation between 0 and 1. The logistic function is defined as:

$$\text{logistic}(\eta) = \frac{1}{1 + exp(-\eta)}$$

And it looks like this:

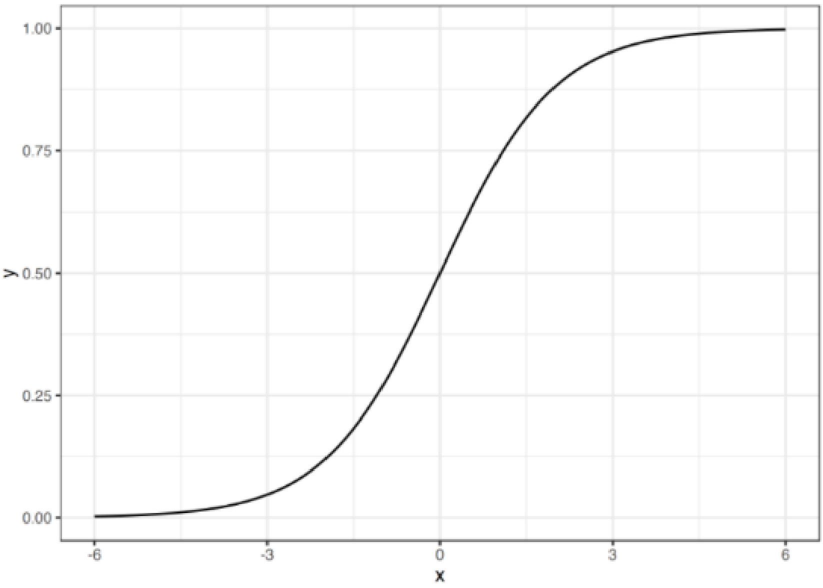


***Fig: 2.3** Logistic Regression*

c) Decision Tree:

Decision tree is the most powerful and popular tool for classification and prediction. A Decision tree is a flowchart like tree structure, where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (terminal node) holds a class label.

It's a predictive modelling tool that has applications spanning a number of different areas. In general, decision trees are constructed via an algorithmic approach that identifies ways to split a data set based on different conditions. It is one of the most widely used and practical methods for supervised learning. Decision Trees are a non-parametric supervised learning method used for both classification and regression tasks. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. The decision rules are generally in form of if-then-else statements. The deeper the tree, the more complex the rules and fitter the model.

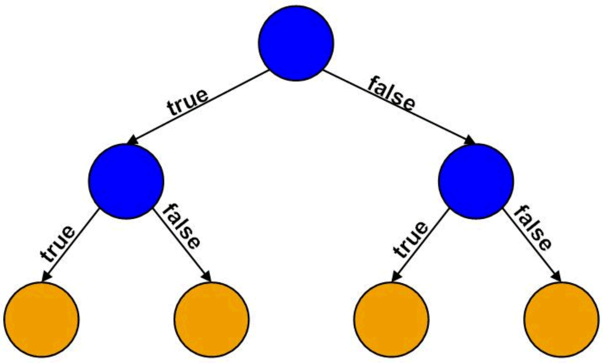


Fig: 2.4 *Decision Tree*

d) Support Vector Machine (SVM):

Support vector machines (SVMs) are powerful yet flexible supervised machine learning methods used for classification, regression, and, outliers' detection. SVMs are very efficient in high dimensional spaces and generally are used in classification problems. SVMs are popular and memory efficient because they use a subset of training points in the decision function.

Some important concepts in SVM are as follows:

i. **Support Vectors** − they may be defined as the data points which are closest to the hyperplane. Support vectors help in deciding the separating line.

ii. **Hyperplane** − the decision plane or space that divides set of objects having different classes.

iii. **Margin** − the gap between two lines on the closet data points of different classes is called margin.

Following diagrams will give you an insight about these SVM concepts –

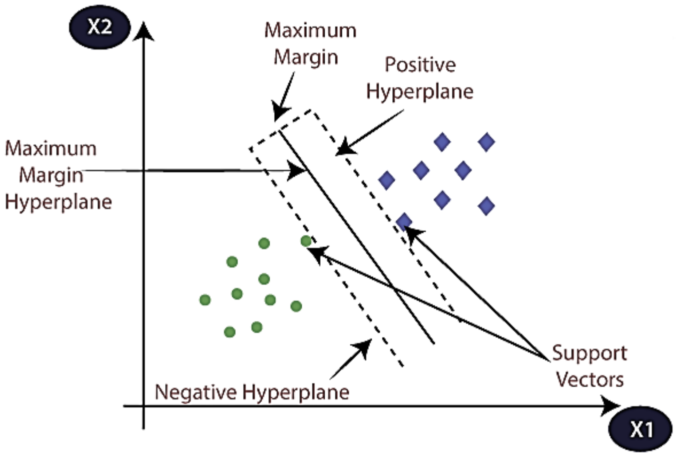


Fig: 2.5 *Support Vector Machine*

e) Naive Bayes:

Naive Bayes classifiers are a collection of classification algorithms based on Bayes' Theorem. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other.

Bayes' Theorem finds the probability of an event occurring given the probability of another event that has already occurred. Bayes' theorem is stated mathematically as the following equation:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

where A and B are events and P(B)? 0.

- Basically, we are trying to find probability of event A, given the event B is true. Event B is also termed as evidence.
- P(A) is the priori of A (the prior probability, i.e. Probability of event before evidence is seen). The evidence is an attribute value of an unknown instance (here, it is event B).
- P(A|B) is a posteriori probability of B, i.e. probability of event after evidence is seen.

f) K-Nearest Neighbors (KNN):

K-Nearest Neighbor is one of the simplest Machine Learning algorithms based on Supervised Learning technique. The algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories. It stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K-NN algorithm.

K-NN is a non-parametric algorithm, which means it does not make any assumption on underlying data. It is also called a lazy learner algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset. KNN algorithm at the training phase just stores the dataset and when it gets new data, and then it classifies that data into a category that is much similar to the new data.

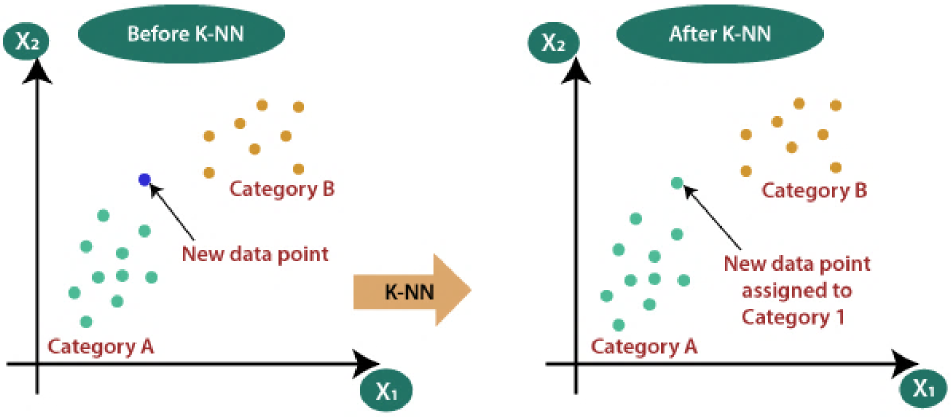


***Fig: 2.6** K Nearest Neighbors*

g) Random Forest:

Random Forest classifier is a learning method that operates by constructing multiple decision trees and the final decision is made based on the majority of the trees and is chosen by the random forest. It is a tree-shaped diagram used to determine a course of action. Each branch of the tree represents a possible decision, instance, or reaction.

Using of Random Forest Algorithm is one of the main advantages is that it reduces the risk of over fitting and the required training time. It offers a high level of accuracy and runs efficiently in large databases and produces almost accurate predictions by approximating missing data. It is having one of the main advantages is that it reduces the risk of over-fitting and the required training time.

2.4 Regression Models.

Regression analysis is a fundamental concept in the field of machine learning. It falls under supervised learning wherein the algorithm is trained with both input features and output labels. It helps in establishing a relationship among the variables by estimating how one variable affects the other.

Regression in machine learning consists of mathematical methods that allow data scientists to predict a continuous outcome (y) based on the value of one or more predictor variables (x).

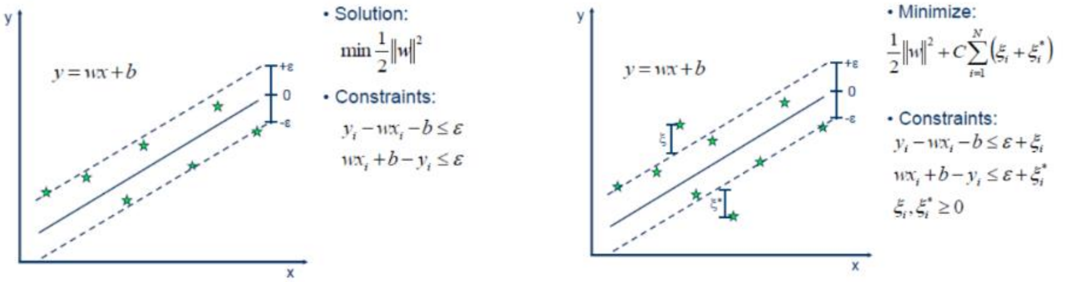
Regression analysis allows us to quantify the relationship between outcome and associated variables. Many techniques for performing statistical predictions have been developed, but, in this project, five models – Linear Regression, Random Forest Regression, Decision Tree Regression, Support Vector Regression, K Nearest Neighbor Regression.

1. **Linear Regression:**
   Linear regression is a basic and commonly used type of predictive analysis. The overall idea of regression is to examine two things:
   (1) does a set of predictor variables do a good job in predicting an outcome (dependent) variable?
   (2) Which variables in particular are significant predictors of the outcome variable, and in what way do they–indicated by the magnitude and sign of the beta estimates–impact the outcome variable? These regression estimates are used to explain the relationship between one dependent variable and one or more independent variables. The simplest form of the regression equation with one dependent and one independent variable is defined by the formula y = c + b*x, where y = estimated dependent variable score, c = constant, b = regression coefficient, and x = score on the independent variable.

2. **Random Forest Regression:**
   A random forest regression follows the concept of simple regression. Values of dependent (features) and independent variables are passed in the random forest model. In a random forest regression, each tree produces a specific prediction. The mean prediction of the individual trees is the output of the regression. This is contrary to random forest classification, whose output is determined by the mode of the decision trees' class. Although random forest regression and linear regression follow the same concept, they differ in terms of functions. The function of linear regression is y=bx + c, where y is the dependent variable, x is the independent variable, b is the estimation parameter, and c is a constant.

3. **Decision Tree Regression:**
   Decision tree builds regression or classification models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes. A decision node (e.g. Age) has two or more branches (e.g. BMI, Sex, Smokers, Region, Children), each representing values for the attribute tested. Leaf node (e.g. Charges) represents a decision on the numerical target. The topmost decision node in a tree which corresponds to the best predictor called root node. Decision trees can handle both categorical and numerical data.

4. **Support Vector Regression:**
   Support Vector Machine can also be used as a regression method, maintaining all the main features that characterize the algorithm (maximal margin). The Support Vector Regression (SVR) uses the same principles as the SVM for classification, with only a few minor differences. First of all, because output is a real number it becomes very difficult to predict the information at hand, which has infinite possibilities. In the case of regression, a margin of tolerance (epsilon) is set in approximation to the SVM which would have already requested from the problem. But besides this fact, there is also a more complicated reason, the algorithm is more complicated therefore to be taken in consideration. However, the main idea is always the same: to minimize error, individualizing the hyperplane which maximizes the margin, keeping in mind that part of the error is tolerated.


y
$y = wx + b$
+ε
0
-ε
x
• Solution:
$\min \frac{1}{2}\|w\|^2$
• Constraints:
$y_i - wx_i - b \le \varepsilon$
$wx_i + b - y_i \le \varepsilon$
y
$y = wx + b$
ξ
ξ*
+ε
0
-ε
x
• Minimize:
$\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}(\xi_i + \xi_i^*)$
• Constraints:
$y_i - wx_i - b \le \varepsilon + \xi_i$
$wx_i + b - y_i \le \varepsilon + \xi_i^*$
$\xi_i, \xi_i^* \ge 0$

5. **K Nearest Neighbor Regression:**
   KNN regression is a non-parametric method that, in an intuitive manner, approximates the association between independent variables and the continuous outcome by averaging the observations in the same neighborhood. The size of the neighborhood needs to be set by the analyst or can be chosen using cross-validation to select the size that minimizes the mean-squared error.

CHAPTER 3

DESIGNING AND IMPLEMENTATION

3.1 Data Preparation and Cleaning

The data has been imported from kaggle website. The website provides with a variety of data and the data used for the project is an insurance amount data. The data included various attributes such as age, sex, body mass index (bmi), smoker and the charges attribute which will work as the label for the project. The data was in structured format and was stores in a csv file format. The data was imported using pandas library.

The presence of missing, incomplete, or corrupted data leads to wrong results while performing any functions such as count, average, mean etc. These inconsistencies must be removed before doing any analysis on data. The data included some ambiguous values which were needed to be removed.

3.2 Training

The model's training and testing phases can begin after the training data is in a suitable format for feeding to the model. Here we have taken 80% of training data and 20% test data. After that the model selection is the most important aspect of the training phase. This entails deciding on the appropriate modelling approach for the job or the optimum parameter values for a particular model.

In fact, the term model selection often refers to both of these processes, as, in many cases, various models were tried first and best performing model (with the best performing parameter settings for each model) was selected.

3.3 Prediction

The model was used to estimate the amount of insurance that would be spent on their health. The amount was predicted using the relationship between the features and the label.

The degree of correctness of the expected value of the insurance amount is defined by accuracy. Using various techniques, features, and train test split sizes, the model predicted the accuracy of the model. The amount of the data used for data training has a significant impact on data correctness. The greater the train's size, the more accurate it is. Using numerous algorithms, the model forecasts the premium amount and displays the impact of each attribute on the forecasted value.

CHAPTER 4

PROJECT DETAILS

4.1 Objective

The main objectives of the proposed project are to:

a) Train and Test the dataset.
b) Regression models to be used are Linear Regression, Random Forest Regression, Decision Tree Regression, Support Vector Regression, K Nearest Neighbor Regression.
c) Plotting the graph of 6 attributes that are in the dataset.
d) Maximum accuracy will link up in the web model.
e) And predict what can be the predicted amount that an individual can get.

4.2 Justification of Topic

a) This project is based on the comparative study of machine learning regression algorithms, as the values are in continuous variable in the dataset.
b) Most of the supervised machine learning algorithm that will be using for checking the comparativeness and the accuracy.
c) And choose the best fit to implement for the prediction.
d) Flask framework will be using for the front end interface to show the predicted output.

4.3 Problem Statement

Predict the charges to the Insurance policyholders. We can analyze all relevant data for the health insurance of a group of insurance policies and develop focused amount charge program.

4.4 Data Collection

The primary source of data for this project was from Kaggle and the dataset is comprised of 3630 records with 7 attributes.

4.5 About the dataset

The data was in structured format and was stores in a .csv file. The data set is not suitable for regression to be performed directly. As a result, cleaning a dataset is necessary before using it in various regression algorithms. Not every attribute in a dataset has an effect on the prediction.

Some attributes even degrade the accuracy, necessitating the removal of these attributes from the code's features. Removing these qualities improves not only accuracy but also overall performance and quickness. Many factors influence the cost of health insurance, including pre-existing medical conditions, family medical history, Body Mass Index (BMI), marital status, location, previous insurances, and so on.

According to our research, age and smoking status had the most influence on amount prediction, with smoker being the most influential factor. Because the children characteristic had absolutely little effect on the prediction, it was removed from the input to the regression model to allow for faster computation.

4.6 Modules used in this project

a) **Scikit-learn:**
Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use. The library is built upon the SciPy (Scientific Python) that must be installed before you can use scikit-learn.

b) **NumPy:**
NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more. At the core of the NumPy package, is the ndarray object. This encapsulates n-dimensional arrays of homogeneous data types, with many operations being performed in compiled code for performance.

c) **Pandas:**

Pandas is an open source Python package that is most widely used for data science/data analysis and machine learning tasks. It is built on top of another package named Numpy, which provides support for multi-dimensional arrays. As one of the most popular data wrangling packages, Pandas works well with many other data science modules inside the Python ecosystem, and is typically included in every Python distribution, from those that come with your operating system to commercial vendor distributions like Active State's Active Python.

d) **Matplotlib:**

Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack. It was introduced by John Hunter in the year 2002. One of the greatest benefits of visualization is that it allows us visual access to huge amounts of data in easily digestible visuals. Matplotlib consists of several plots like line, bar, scatter, histogram etc.

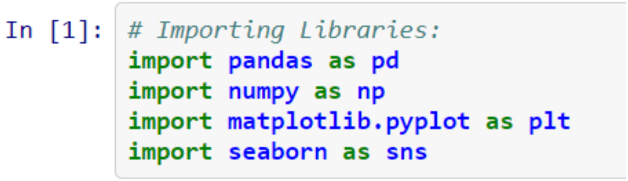
e) **Seaborn:**

Seaborn is a library for making statistical graphics in Python. It builds on top of matplotlib and integrates closely with pandas data structures. Seaborn helps you explore and understand your data. Its plotting functions operate on dataframes and arrays containing whole datasets and internally perform the necessary semantic mapping and statistical aggregation to produce informative plots. Its dataset-oriented, declarative API lets you focus on what the different elements of your plots mean, rather than on the details of how to draw them.

CHAPTER 5

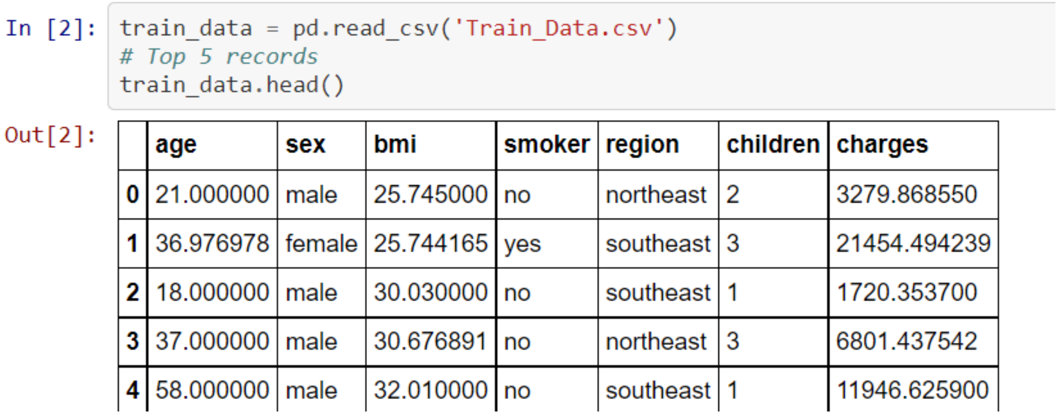
CODES WITH OUTPUT (Explanation)

- **Importing necessary libraries:**
  We have imported the necessary libraries i.e numpy, pandas, matplotlib and seaborn (provides high-level interface for attractive and informative statistical graphics).



```
In [1]: # Importing Libraries:
        import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        import seaborn as sns
```

- **Importing the dataset and head():**
  With the help of pandas as pd we have import our dataset and head() gives us a quick look at our dataset.

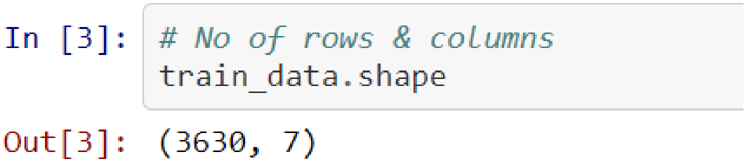


```
In [2]: train_data = pd.read_csv('Train_Data.csv')
        # Top 5 records
        train_data.head()
```

Out[2]:

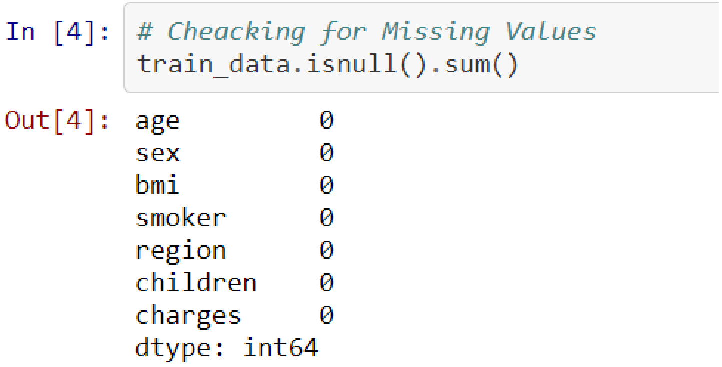
	age	sex	bmi	smoker	region	children	charges
0	21.000000	male	25.745000	no	northeast	2	3279.868550
1	36.976978	female	25.744165	yes	southeast	3	21454.494239
2	18.000000	male	30.030000	no	southeast	1	1720.353700
3	37.000000	male	30.676891	no	northeast	3	6801.437542
4	58.000000	male	32.010000	no	southeast	1	11946.625900

- **Viewing total rows and columns of the dataset:**
  .shape will help us to know total number of rows and columns of the dataset.



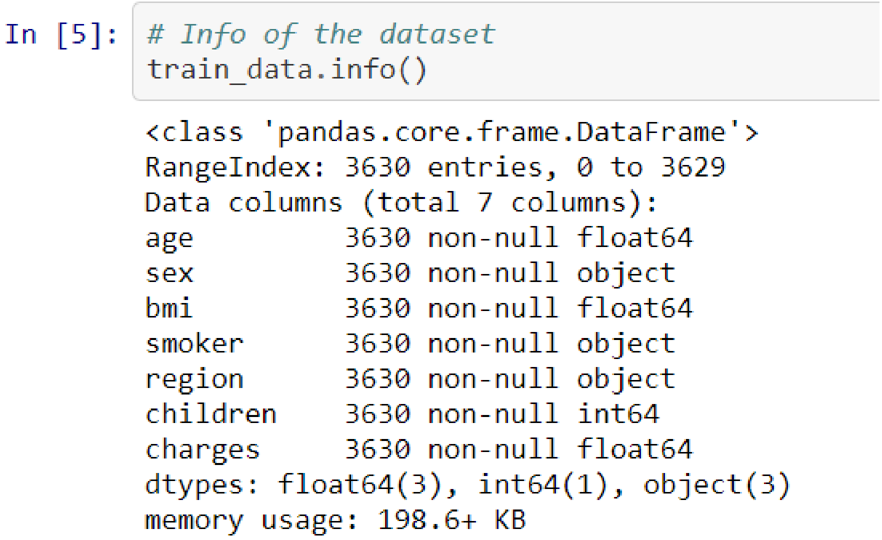
```
In [3]: # No of rows & columns
        train_data.shape

Out[3]: (3630, 7)
```

- **Checking for Missing Values:**
  For finding the number of missing values in a dataset we use .isnull(). We call .isnull().sum() to give the output of series containing data about count of NaN in each column. We get no null values…thus the data set has no missing values.



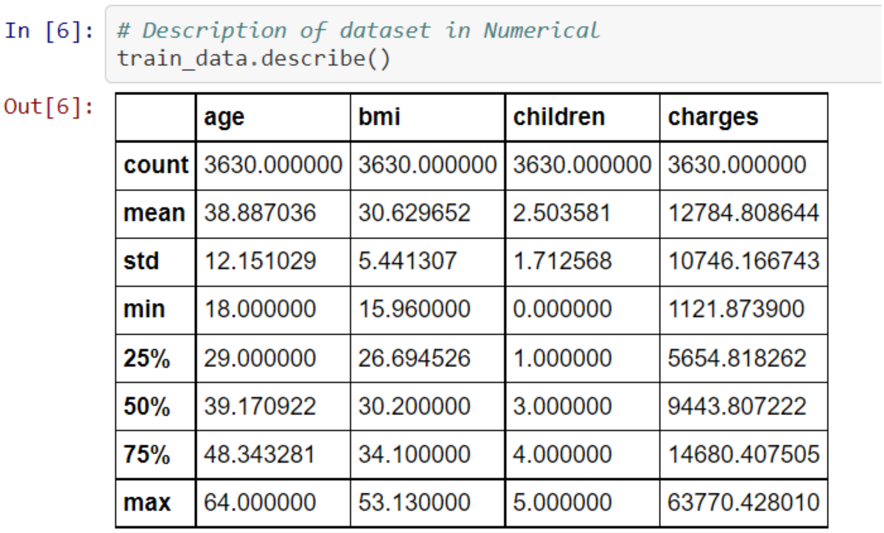
```
In [4]: # Cheacking for Missing Values
        train_data.isnull().sum()

Out[4]: age         0
        sex         0
        bmi         0
        smoker      0
        region      0
        children    0
        charges     0
        dtype: int64
```

- **Information of the dataset:**
  .info() help us to view the datatype.



```
In [5]: # Info of the dataset
        train_data.info()

        <class 'pandas.core.frame.DataFrame'>
        RangeIndex: 3630 entries, 0 to 3629
        Data columns (total 7 columns):
        age         3630 non-null float64
        sex         3630 non-null object
        bmi         3630 non-null float64
        smoker      3630 non-null object
        region      3630 non-null object
        children    3630 non-null int64
        charges     3630 non-null float64
        dtypes: float64(3), int64(1), object(3)
        memory usage: 198.6+ KB
```

- **Description of dataset in Numericals:**
  Pandas describe() is used to view some basic statistical details like percentile, mean, std etc. of a data frame or a series of numeric values.



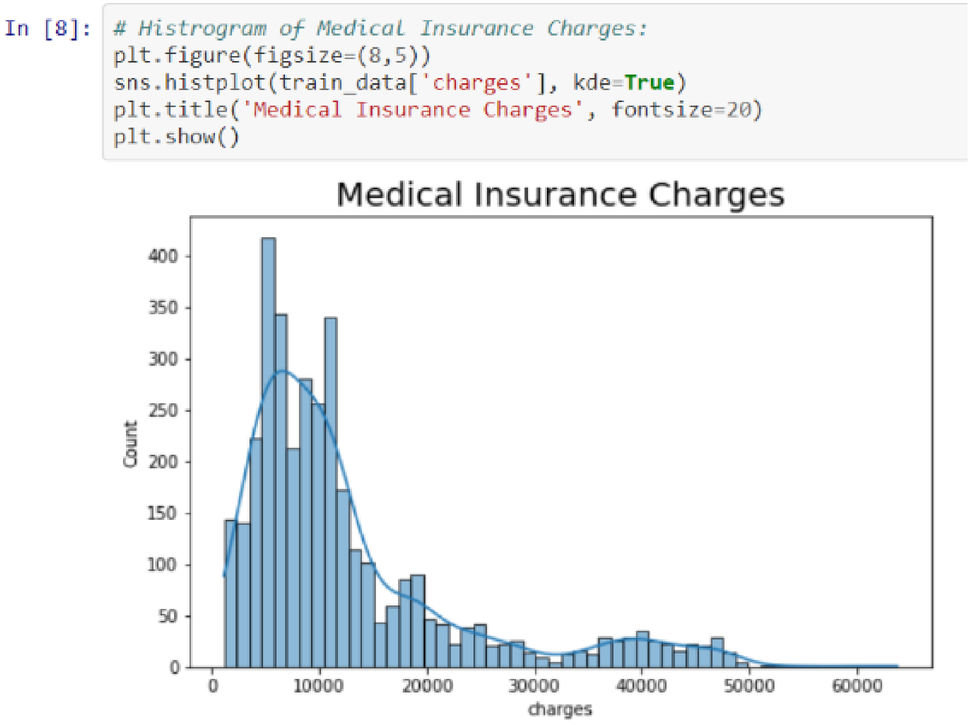
```
In [6]: # Description of dataset in Numerical
        train_data.describe()
```

Out[6]:

	age	bmi	children	charges
count	3630.000000	3630.000000	3630.000000	3630.000000
mean	38.887036	30.629652	2.503581	12784.808644
std	12.151029	5.441307	1.712568	10746.166743
min	18.000000	15.960000	0.000000	1121.873900
25%	29.000000	26.694526	1.000000	5654.818262
50%	39.170922	30.200000	3.000000	9443.807222
75%	48.343281	34.100000	4.000000	14680.407505
max	64.000000	53.130000	5.000000	63770.428010

- **Plotting the Histogram:**
  A histogram is used to summarize discrete or continuous data. In other words, it provides a visual interpretation of numerical data by showing number of data points that fall within a specified range of values.

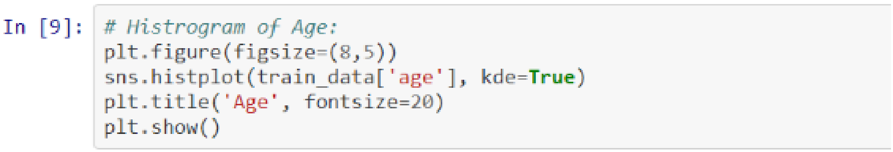
  a) Histogram of Charges.



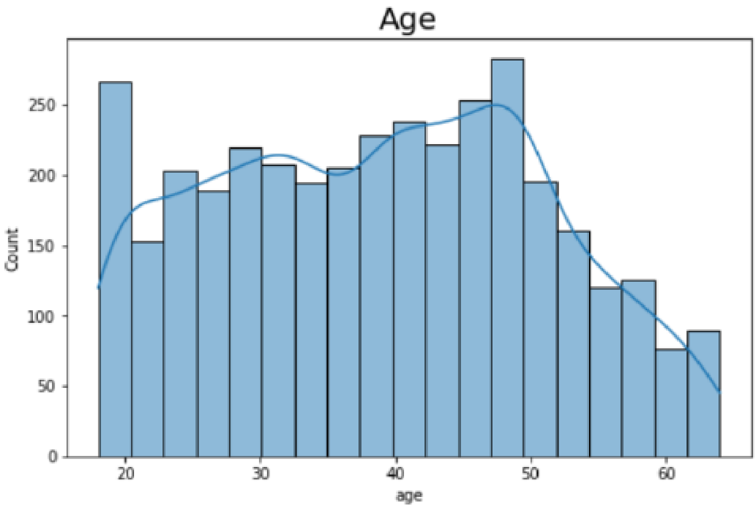
```
In [8]: # Histrogram of Medical Insurance Charges:
        plt.figure(figsize=(8,5))
        sns.histplot(train_data['charges'], kde=True)
        plt.title('Medical Insurance Charges', fontsize=20)
        plt.show()
```

b) Histogram of Age.

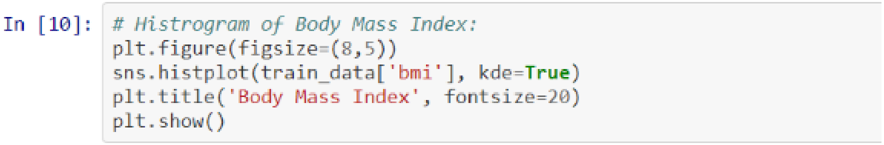
Age:



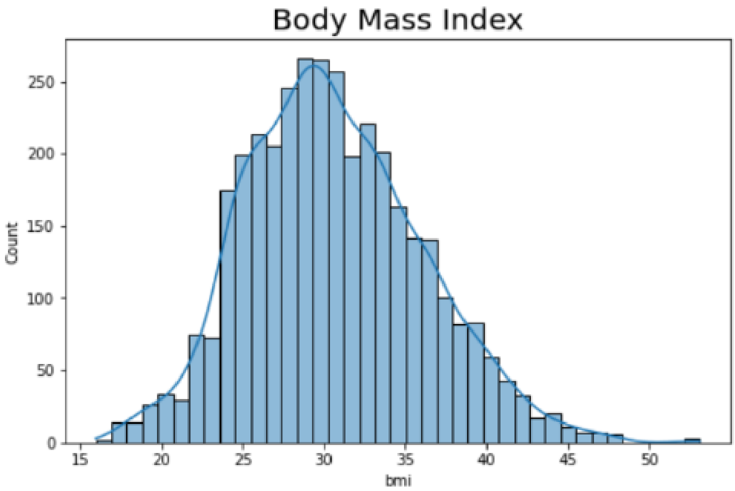
```
In [9]: # Histrogram of Age:
        plt.figure(figsize=(8,5))
        sns.histplot(train_data['age'], kde=True)
        plt.title('Age', fontsize=20)
        plt.show()
```



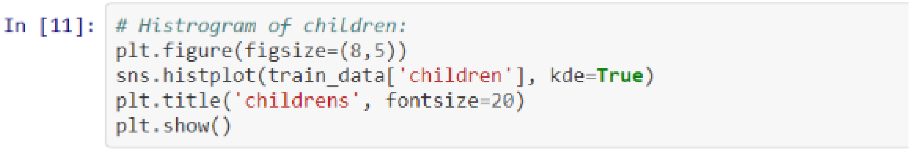
c) Histogram of Body Mass Index.



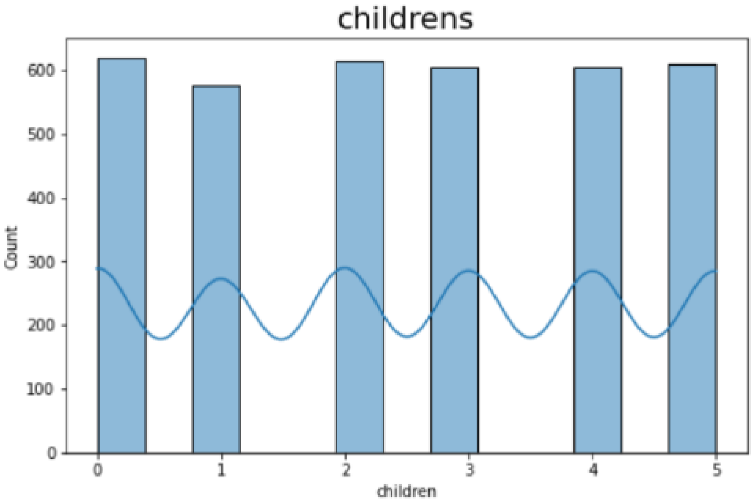
```
In [10]: # Histrogram of Body Mass Index:
         plt.figure(figsize=(8,5))
         sns.histplot(train_data['bmi'], kde=True)
         plt.title('Body Mass Index', fontsize=20)
         plt.show()
```



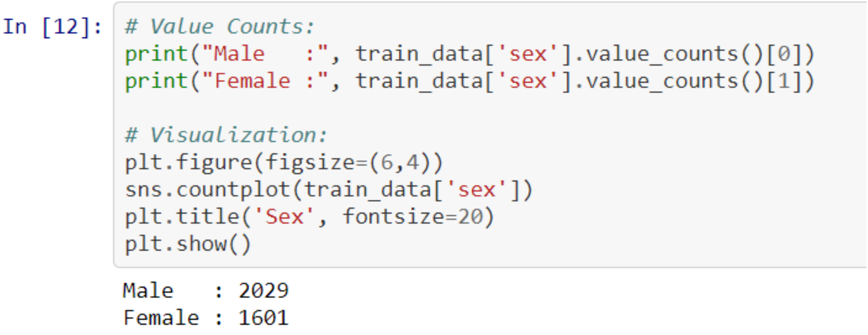
d) Histogram of Children’s.



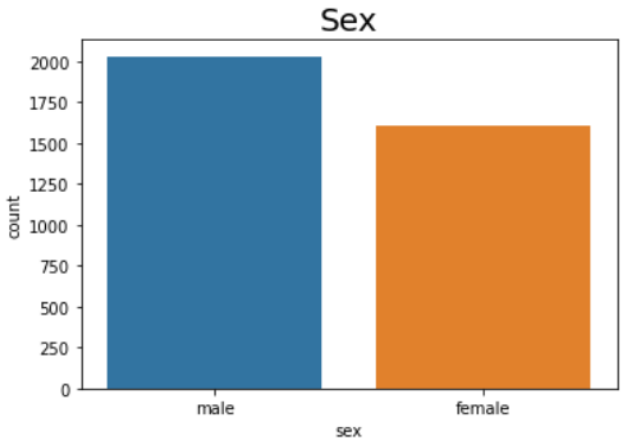
```
In [11]: # Histrogram of children:
         plt.figure(figsize=(8,5))
         sns.histplot(train_data['children'], kde=True)
         plt.title('childrens', fontsize=20)
         plt.show()
```



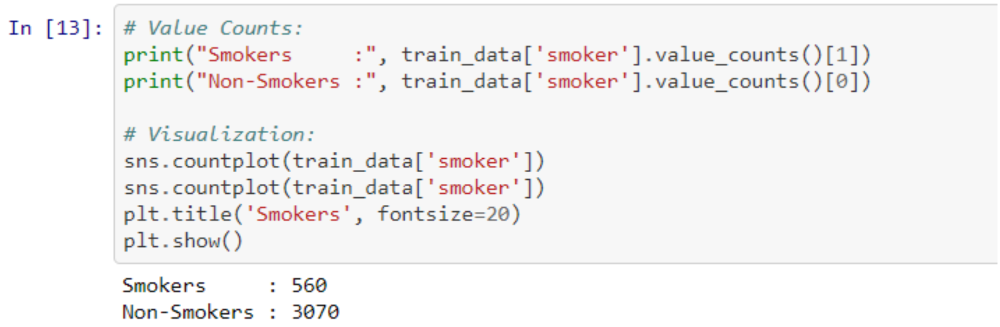
- **Plotting the graph of Gender/Sex:**
  Here the graph shows the total number of Male and Female, present in the dataset.



```
In [12]: # Value Counts:
         print("Male   :", train_data['sex'].value_counts()[0])
         print("Female :", train_data['sex'].value_counts()[1])

         # Visualization:
         plt.figure(figsize=(6,4))
         sns.countplot(train_data['sex'])
         plt.title('Sex', fontsize=20)
         plt.show()

         Male   : 2029
         Female : 1601
```

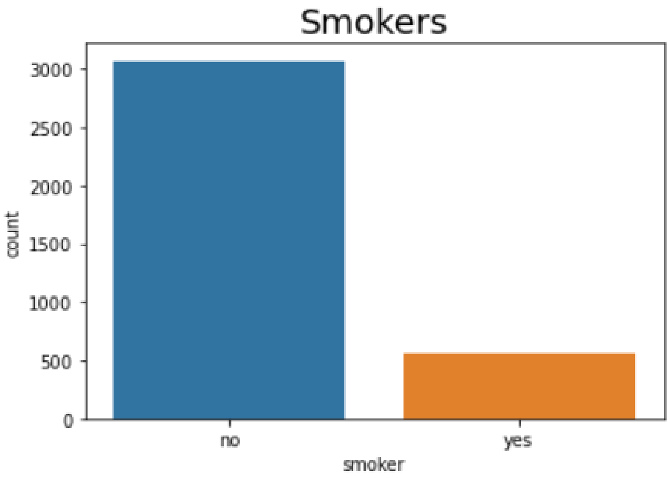


- **Plotting the graph of Smokers and Non Smokers:**
  Here the graph shows the total number of Smokers and Non Smokers, present in the dataset.

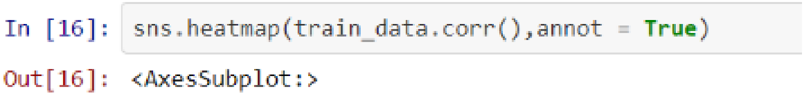


```
In [13]: # Value Counts:
         print("Smokers     :", train_data['smoker'].value_counts()[1])
         print("Non-Smokers :", train_data['smoker'].value_counts()[0])

         # Visualization:
         sns.countplot(train_data['smoker'])
         sns.countplot(train_data['smoker'])
         plt.title('Smokers', fontsize=20)
         plt.show()
```

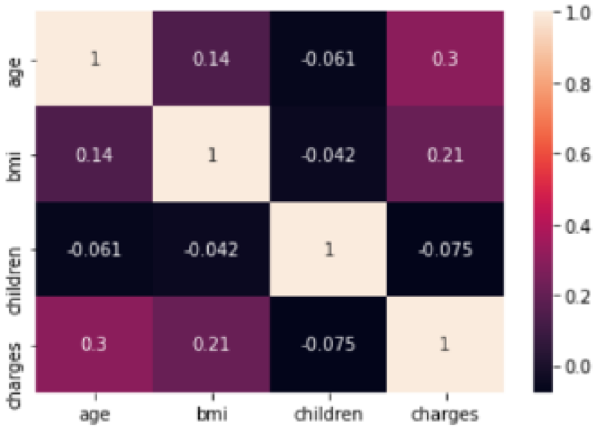
```
Smokers     : 560
Non-Smokers : 3070
```



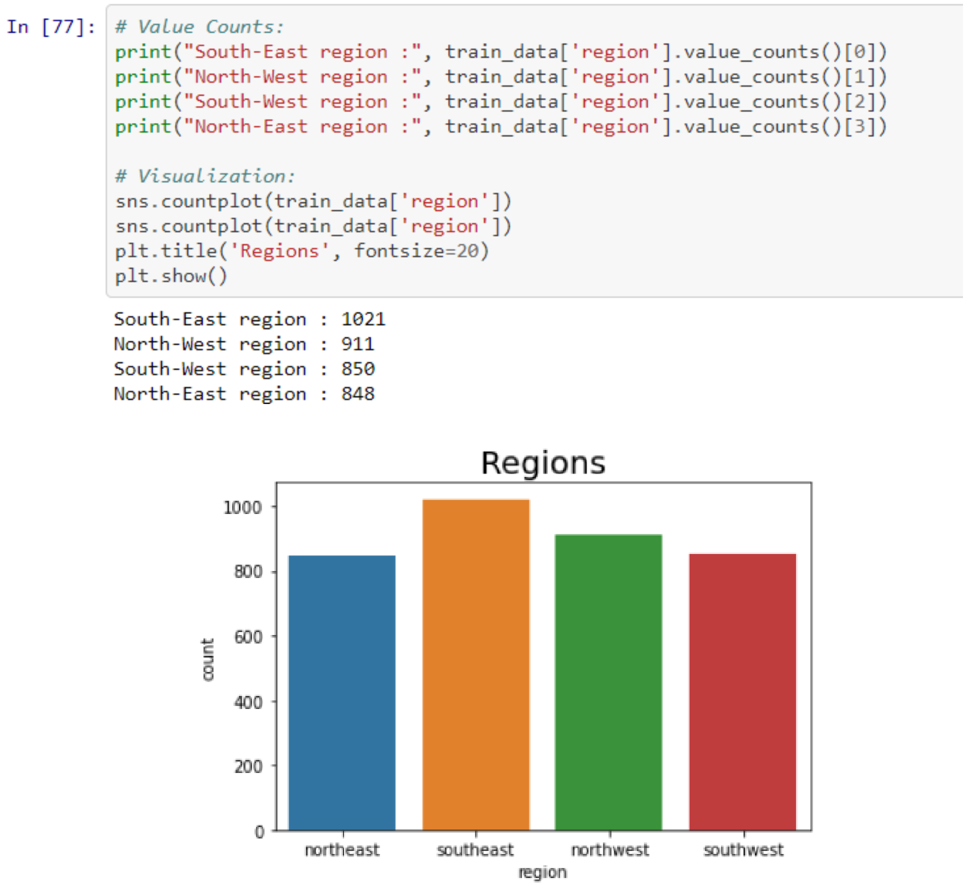
- **Plotting the Correlation Heatmap:**
  A correlation heatmap is a heatmap that shows a 2D correlation matrix between two discrete dimensions, using colored cells to represent data from usually a monochromatic scale. The values of the first dimension appear as the rows of the table while of the second dimension as a column.



```
In [16]: sns.heatmap(train_data.corr(),annot = True)
Out[16]: <AxesSubplot:>
```



- **Plotting the graph of Regions:**
  Here the graph shows the total number of regions and their counts, present in the dataset.



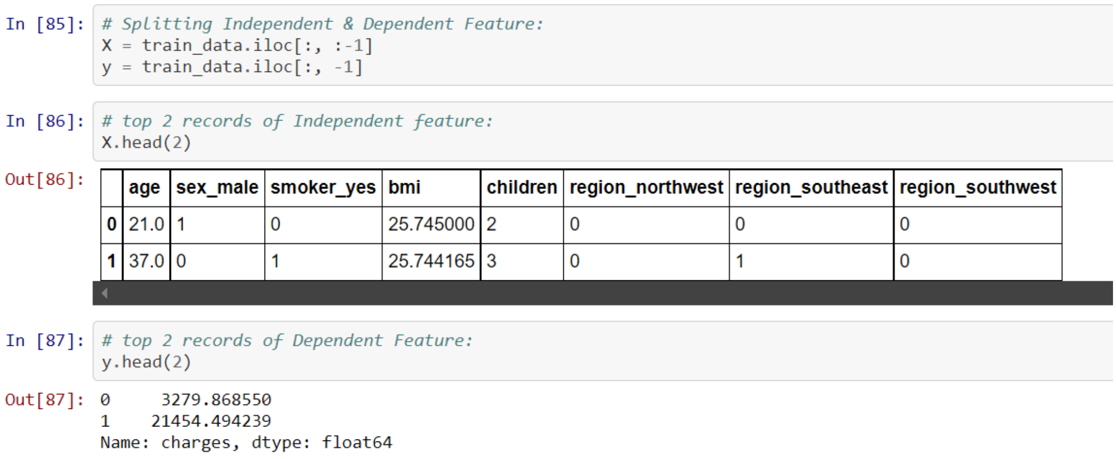
In [77]:

```
# Value Counts:
print("South-East region :", train_data['region'].value_counts()[0])
print("North-West region :", train_data['region'].value_counts()[1])
print("South-West region :", train_data['region'].value_counts()[2])
print("North-East region :", train_data['region'].value_counts()[3])

# Visualization:
sns.countplot(train_data['region'])
sns.countplot(train_data['region'])
plt.title('Regions', fontsize=20)
plt.show()
```

```
South-East region : 1021
North-West region : 911
South-West region : 850
North-East region : 848
```

- **Splitting up the Dependent and Independent variables:**
  Independent variables (also referred to as Features) are the input for a process that is being analyzes. Dependent variables are the output of the process.
  Any predictive mathematical model tends to divide the observations (data) into dependent/ independent features in order to determine the causal effect. It should be noted that relationship between dependent and independent variables need not be linear, it can be polynomial.
  It is common practice while doing experiments to change one independent variable while keeping others constant to see the change caused on the dependent variable.

The concept of dependent and independent variables is important to understand. In the above dataset, if you look closely, the first six columns (age, sex, bmi, smoker, region, children) determine the outcome of the seventh, or last, column (i.e. Charges).
With this in mind, we need to split our dataset into the matrix of independent variables and the vector or dependent variable. Mathematically, Vector is defined as a matrix that has just one column.

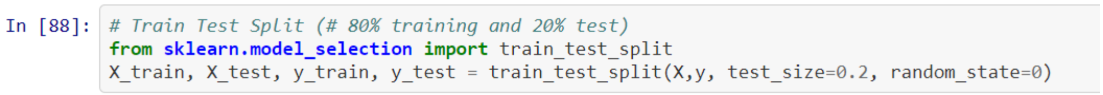


```
In [85]: # Splitting Independent & Dependent Feature:
         X = train_data.iloc[:, :-1]
         y = train_data.iloc[:, -1]

In [86]: # top 2 records of Independent feature:
         X.head(2)

Out[86]:
```

	age	sex_male	smoker_yes	bmi	children	region_northwest	region_southeast	region_southwest
0	21.0	1	0	25.745000	2	0	0	0
1	37.0	0	1	25.744165	3	0	1	0

```
In [87]: # top 2 records of Dependent Feature:
         y.head(2)

Out[87]: 0     3279.868550
         1    21454.494239
         Name: charges, dtype: float64
```

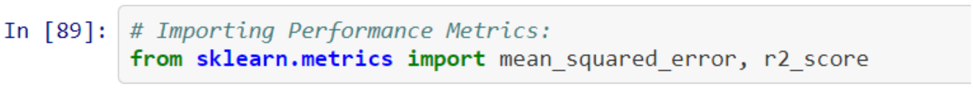
- **Training and Testing Data Bifurcation:**
  Here we have taken 80% of training data and 20% of test data.
  We will be dividing the dataset into two main groups. One for training the model and the other for Testing our trained model's performance.



```
In [88]: # Train Test Split (# 80% training and 20% test)
         from sklearn.model_selection import train_test_split
         X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.2, random_state=0)
```

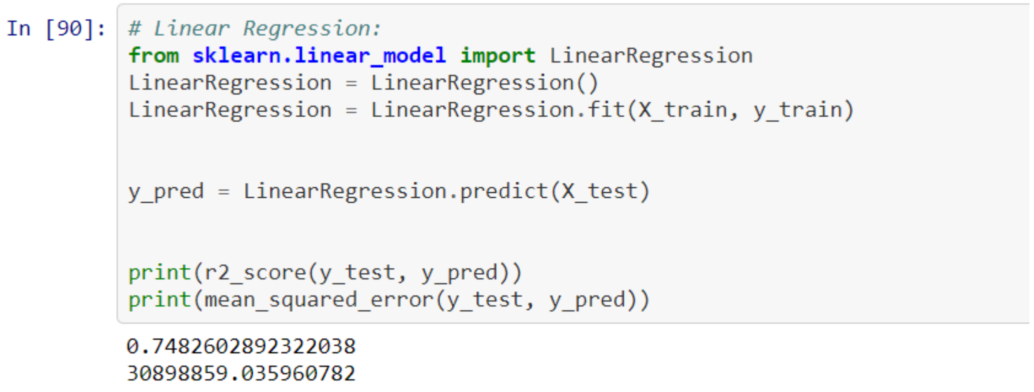
- **Importing the performance matrix:**

Mean Squared Error (MSE): It is quite similar to Mean Absolute Error, the only difference being that MSE takes the average of the square of the difference between the original values and the predicted values. The advantage of MSE being that it is easier to compute the gradient.
R squared: It is a statistical measure that represents the goodness of fit of a regression model. The ideal value for r-square is 1. The closer the value of r-square to 1, the better is the model fitted.



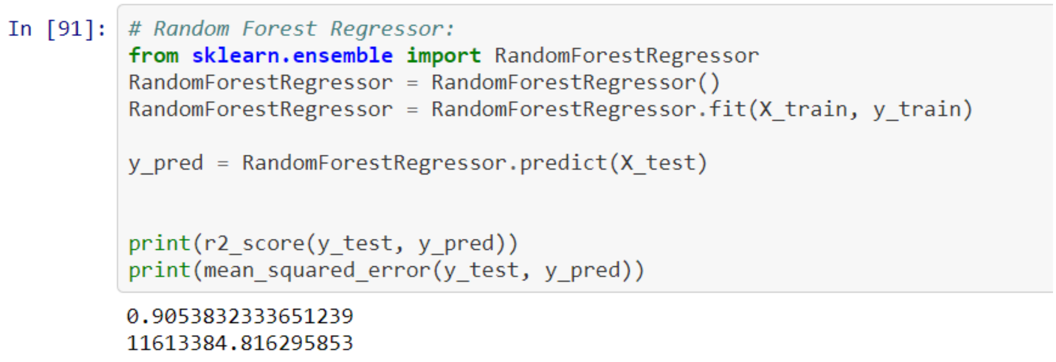
```
In [89]: # Importing Performance Metrics:
         from sklearn.metrics import mean_squared_error, r2_score
```

- **Building a Linear Regression model using Scikit Learn:**



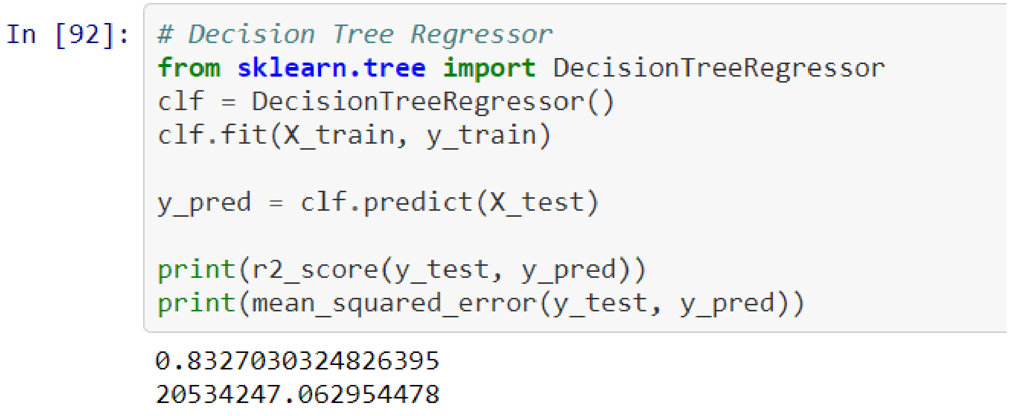
```
In [90]: # Linear Regression:
         from sklearn.linear_model import LinearRegression
         LinearRegression = LinearRegression()
         LinearRegression = LinearRegression.fit(X_train, y_train)


         y_pred = LinearRegression.predict(X_test)


         print(r2_score(y_test, y_pred))
         print(mean_squared_error(y_test, y_pred))

         0.7482602892322038
         30898859.035960782
```

- **Building a Random Forest Regression model using Scikit Learn:**



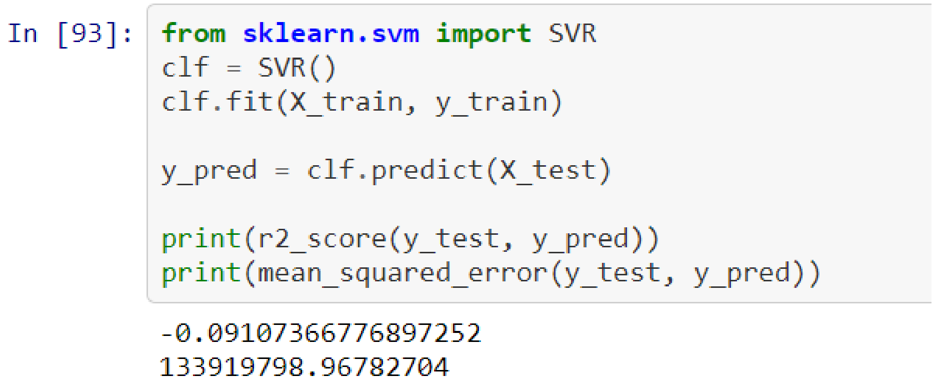
```
In [91]: # Random Forest Regressor:
         from sklearn.ensemble import RandomForestRegressor
         RandomForestRegressor = RandomForestRegressor()
         RandomForestRegressor = RandomForestRegressor.fit(X_train, y_train)

         y_pred = RandomForestRegressor.predict(X_test)


         print(r2_score(y_test, y_pred))
         print(mean_squared_error(y_test, y_pred))

         0.9053832333651239
         11613384.816295853
```

- **Building a Decision Tree Regression model using Scikit Learn:**



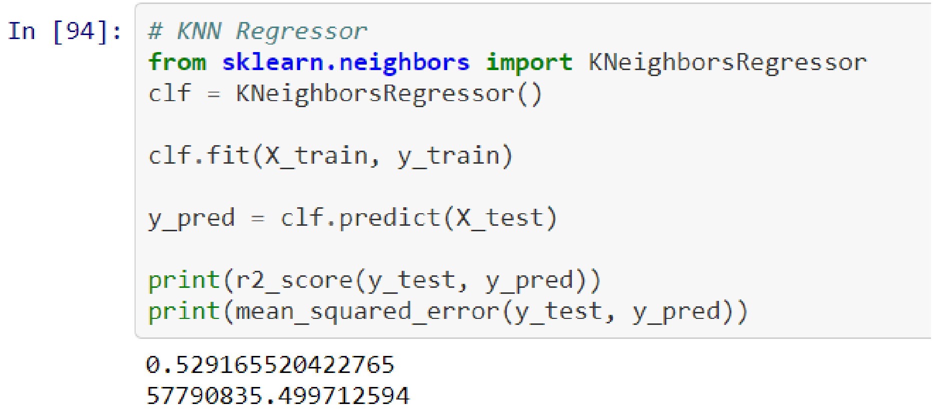
```
In [92]: # Decision Tree Regressor
         from sklearn.tree import DecisionTreeRegressor
         clf = DecisionTreeRegressor()
         clf.fit(X_train, y_train)

         y_pred = clf.predict(X_test)

         print(r2_score(y_test, y_pred))
         print(mean_squared_error(y_test, y_pred))

         0.83270303248263951
         20534247.062954478
```

- **Building a Support Vector Regression model using Scikit Learn:**



```
In [93]: from sklearn.svm import SVR
         clf = SVR()
         clf.fit(X_train, y_train)

         y_pred = clf.predict(X_test)

         print(r2_score(y_test, y_pred))
         print(mean_squared_error(y_test, y_pred))

         -0.09107366776897252
         133919798.96782704
```

- **Building a K Nearest Neighbor Regression model using Scikit Learn:**



```
In [94]: # KNN Regressor
         from sklearn.neighbors import KNeighborsRegressor
         clf = KNeighborsRegressor()

         clf.fit(X_train, y_train)

         y_pred = clf.predict(X_test)

         print(r2_score(y_test, y_pred))
         print(mean_squared_error(y_test, y_pred))

         0.529165520422765
         57790835.499712594
```

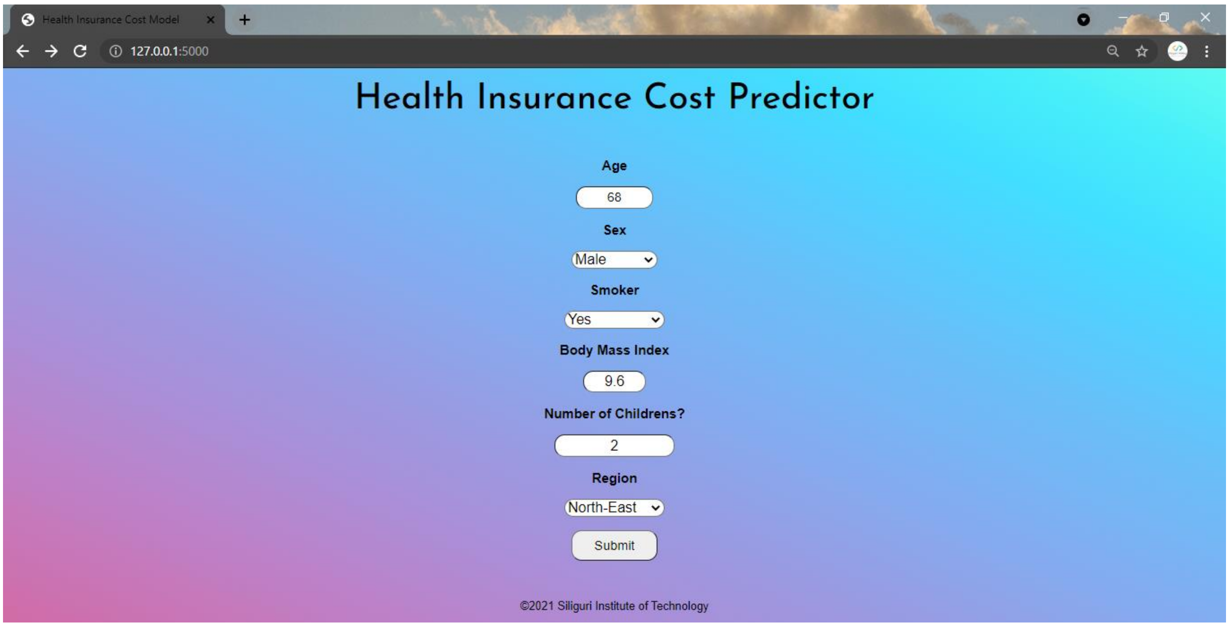
- **Results:**

We see that the accuracy of predicted amount was seen better i.e. 90.5% in Random Forest Regression. And therefore we have taken this model and display the predicted amount in a User Interface i.e. deploying the model using flask framework.

Here the individual will fill the following attributes and get the idea of the predicted amount for the Health Insurance.

- **Output User Interface (UI):**

  Using Flask framework, we have created the user interface, with attributes to be filled and the predicted output will be shown.



Health Insurance Cost Predictor

Age: 68

Sex: Male

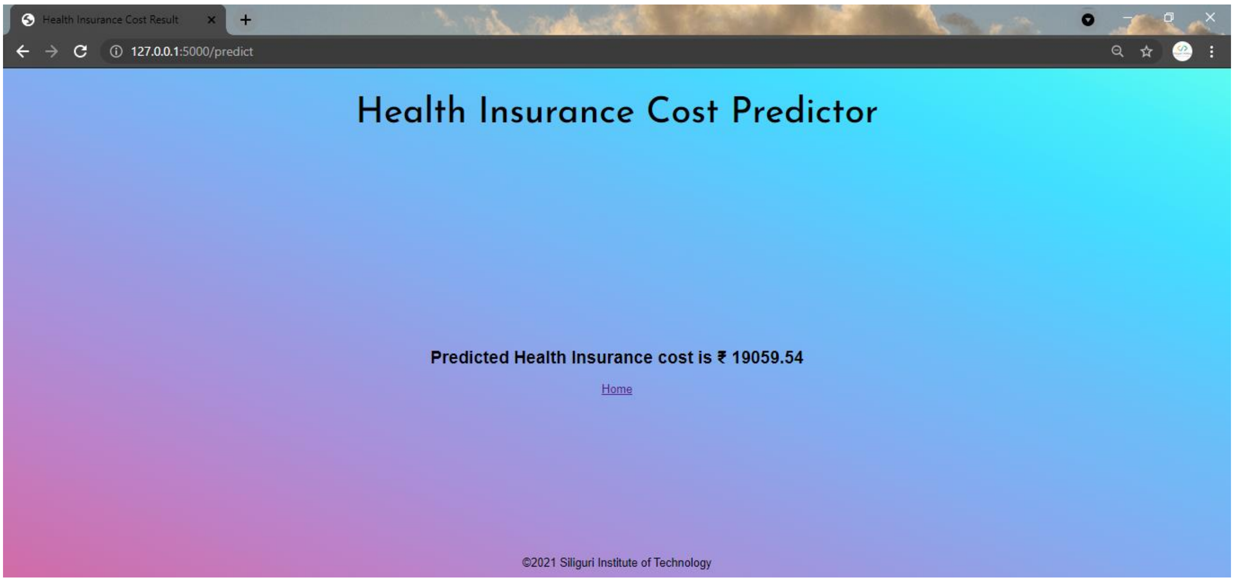
Smoker: Yes

Body Mass Index: 9.6

Number of Childrens?: 2

Region: North-East

©2021 Siliguri Institute of Technology



Health Insurance Cost Predictor

Predicted Health Insurance cost is ₹ 19059.54

©2021 Siliguri Institute of Technology

CHAPTER 6

CONCLUSION AND FUTURE SCOPE

Conclusion:

Background in this project, five regression models are evaluated for individual health insurance data. The health insurance data was used to develop the five regression models, and the predicted premiums from these models were compared with actual premiums to compare the accuracies of these models. It has been found that Random Forest Regression model is the best performing model out here.

Future Scope:

Predicting premium amounts is based on an individual's health rather than the terms and circumstances of another company's insurance. The models can be used to forecast premiums based on data collected in subsequent years. Also further, we can use the Gradient boosting regression technique to get a better predicted output. This can help people and insurance companies work together to provide better and more health-focused insurance coverage.

CHAPTER 7

SYSTEM REQUIREMENTS

Software Requirements:

- Operating System (OS): Windows or Linux
- Python 3
- Anaconda software
- Jupyter Notebook

Hardware Requirements:

- RAM: 4 GB or above
- Processor: Intel Core i3 or above
- HDD: 1 TB or above

REFERENCES

1) https://www.researchgate.net/publication/348559741_Predict_Health_Insurance_Cost_by_using_Machine_Learning_and_DNN_Regression_Models

2) https://www.ijert.org/health-insurance-amount-prediction

3) https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5977561/

THANK YOU