Insurance – Health Cost Prediction using Multivariate Regression

Importing the libraries:

**Import numpy as np - Importing the numpy package.**  
NumPy is a Python library for numerical computing that enables efficient handling of large arrays and matrices. It provides essential tools for mathematical operations, array manipulation.

**Import pandas as pd - Importing the pandas package.**  
Pandas is a powerful Python library for data manipulation and analysis. It provides data structures like DataFrames and Series, which make it easy to work with structured data. Pandas is widely used in data science, data analysis, and machine learning for tasks such as cleaning, exploring, and transforming datasets. Its key features include handling missing data, merging and grouping data, and providing flexible indexing and slicing capabilities.

**import matplotlib.pyplot as plt – Importing the matplot package.**

Matplotlib is a popular Python library for creating static, animated, and interactive visualizations in a variety of formats. It provides a flexible and comprehensive set of plotting tools, making it suitable for a wide range of data visualization tasks. Matplotlib can be used to generate line plots, scatter plots, bar plots, histograms, pie charts, and more**.**

**importing dataset**

**data=pd.read\_csv("insurance\_heartcost.csv")**

pd.read\_csv("insurance\_heartcost.csv"): This part of the code uses the read\_csv function from Pandas to read data from a CSV file named "insurance\_heartcost.csv". This function reads the data and creates a DataFrame, which is a two-dimensional, tabular data structure in Pandas.

**Preprocessing**

**from sklearn import preprocessing**

**l=preprocessing.LabelEncoder()**

**sex\_encoded=l.fit\_transform(data['sex'])**

**data['sex']=sex\_encoded**

**smoker\_encoded=l.fit\_transform(data['smoker'])**

**data['smoker']=smoker\_encoded**

1. **from sklearn import preprocessing**: Imports the preprocessing module from scikit-learn, which includes various tools for data preprocessing.
2. **l = preprocessing.LabelEncoder()**: Creates a **LabelEncoder** object named 'l'. The **LabelEncoder** is used to transform categorical labels (such as 'male' and 'female') into numerical values.
3. **sex\_encoded = l.fit\_transform(data['sex'])**: Applies the **fit\_transform** method of the **LabelEncoder** to the 'sex' column in the 'data' DataFrame. This method fits the encoder to the unique values in the 'sex' column and transforms the labels into numerical values. The encoded values are stored in the 'sex\_encoded' variable.
4. **data['sex'] = sex\_encoded**: Updates the 'sex' column in the 'data' DataFrame with the newly encoded values.
5. **smoker\_encoded = l.fit\_transform(data['smoker'])**: Applies the **fit\_transform** method of the previously created **LabelEncoder** object ('l') to the 'smoker' column in the 'data' DataFrame. This method fits the encoder to the unique values in the 'smoker' column and transforms the labels into numerical values. The encoded values are stored in the 'smoker\_encoded' variable.
6. **data['smoker'] = smoker\_encoded**: Updates the 'smoker' column in the 'data' DataFrame with the newly encoded values.

# Assigning dependent and independent variable

# x=data.iloc[:,:5] // input data

# y=data['charges'] // output data

# Splitting the Dataset into Training and Testing set

# from sklearn.model\_selection import train\_test\_split

# x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.3,random\_state=0)

# This line imports the train\_test\_split function from scikit-learn's model\_selection module.

* **train\_test\_split(x, y, test\_size=0.3, random\_state=0): This function takes your feature matrix x and target variable y and splits them into training and testing sets. The test\_size=0.3 parameter specifies that 30% of the data should be used for testing, and the remaining 70% for training. The random\_state=0 ensures reproducibility by fixing the random seed.**
* **x\_train, x\_test, y\_train, y\_test = ...: The function returns four sets of data: x\_train (training features), x\_test (testing features), y\_train (training target), and y\_test (testing target). These variables will be used in the subsequent training and evaluation of a machine learning model.**

# fitting the Model

**from sklearn.linear\_model import LinearRegression**

**hcr=LinearRegression()**

**hcr.fit(x\_train,y\_train)**

* This line imports the LinearRegression class from scikit-learn's linear\_model module.
* hcr = LinearRegression(): This line creates an instance of the LinearRegression model and assigns it to the variable hcr. This instance will be used to train the linear regression model.
* hcr.fit(x\_train, y\_train): This line trains the linear regression model (hcr) using the training data. The x\_train variable contains the training features, and y\_train contains the corresponding target values.

**y\_predict=hcr.predict([[23,0,34.67,0,1]])**

**y\_predict**

**hcr.predict([[23, 0, 34.67, 0, 1]])**: This line uses the **predict** method of the trained linear regression model (**hcr**) to make predictions. The input to the **predict** method is a 2D array or list containing the features for which you want to predict the target variable.