Medical Insurance Cost Prediction:

(Machine Learning Project) : -

Problem Statement : - Here I have to built a machine learning model ,which can predict anyone's insurance cost based on certain parameter. Here I have to use input (parameters) features like sex,age,region etc. The given dataset gives us the values of different individual’s age , sex, bmi , children ,smoker, region and charges. Our goal is to make a model which can predict the health insurance cost of new customer which is mentioned in the dataset as ‘charges’ based on the other features . Imposing this model the insurance company, health policy officials , even an individual can estimate their cost and can plan or able to anticipate better about their medical insurance cost. This models could be the best predictive tool for giving the insight about financial constraints presents in healthcare which can directly affect the cost of medical insurance ,so one can have a blueprint of potential insurance cost. It can help the policy companies to optimize the charge of health insurance ,building strategies to bring in traffic. A individual can be aware at the time of budgeting their healthcare cost. It will help the health policy officials to make the best decisions regarding the healthcare policy. Here the problem statement revolves around to make a suitable, accurate decision moreover prediction of health insurance by analyzing the key features based on their lifestyle and demographic factors. Thereby it is facilitating the decision making with knowledge and strategize good Ideas for companies and officials.

DATA EXPLORING AND ANALYZING : -

* The dataset comprises individuals' demographic and health-related information including age, sex, BMI, number of children, smoking status, region, and medical insurance charges. Age and number of children are represented as integers, while BMI and charges are floats. Sex, smoker status, and region are categorical variables.
* There are 1338 entries in the Dataframe, each with 7 columns. It has data types for objects, floats, and integers. The columns show attributes like age, gender, BMI, number of kids, smoking status, area, and insurance costs. The dataset is free of missing values.
* There is one duplicate value which I removed by .drop\_duplicates()
* Stats: Considering a standard deviation of about 14 years, the average age of those included in the sample is about 39 years old. The age range is 18 to 64 years old, with 25% of the population being under 27, 50% being under 39, and 75% being under 51.

With a standard deviation of roughly 6.10, the average BMI is roughly 30.66. The range of BMI values is 15.96 to 53.13. The average number of children, per person is 1.10 with a deviation of around 1.21. In terms of the distribution of children 25% of the population does not have any children 50% have one child and 75% have two or fewer children. The range for the number of children varies from 0 to 5.When it comes to insurance costs the average fee is $13,279, with a deviation of about $12,110. The charges typically fall between $1,121.87 and $63,770.43.

* With an average age of 39, people are comparatively youthful.
* Most individuals have 2 or fewer children, with a few having up to 5 children.

EDA :-

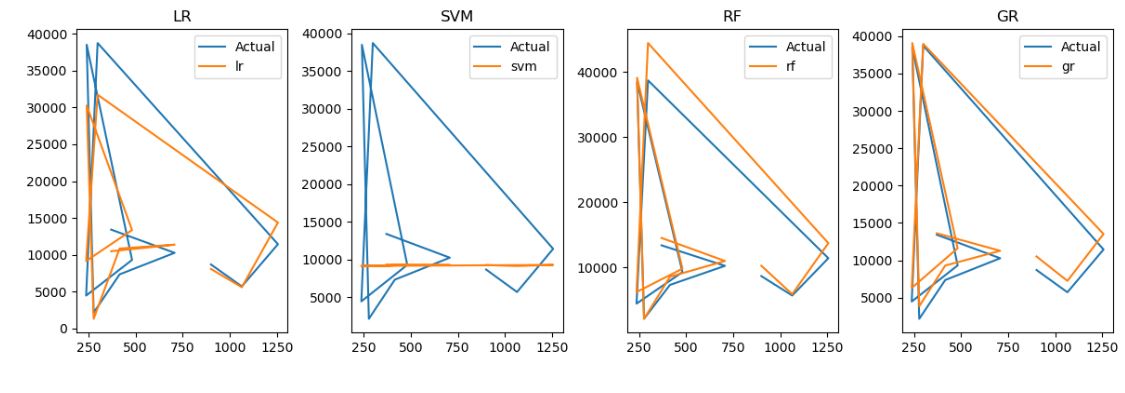
* We already know that there some of the categorical columns and some are numerical columns.
* We discriminate them by simple - select\_dtypes(include**=**"number") , select\_dtypes(exclude**=**"number").
* We get - Numerical columns in the data :['age', 'bmi', 'children', 'charges']  
  Categorical columns in the data : ['sex', 'smoker', 'region']
* Now we get the values of each unique category of every categorical column by - value\_counts()
* We get male 67 , female 662 for the column ‘sex’
* We get no 1063, yes 274 for the column of ‘smoker’
* We got southeast 364 , southwest 325, northwest 324,northeast 324 for the column region.
* Then we visualize the ctegorical data by pie-charts
* We got the ‘sex’ column distributed among 50.5% ‘male’ and 49.5% female , the ‘smoker’ column distributed among 79.5 ‘no’ and 20.5% ‘yes’ and the ‘region’ column distributed among ‘southeast’ 27.2% , ‘northeast’ 24.2% , ‘southwest’ 24.3% and ‘northwest’ 24.3%.
* We visualize the numerical columns distribution by various plot like – box, kde, strip, point and ecdf etc.
* From the ecdf graph of age distribution we can see - 20% or so of the population is under 30.
* From ecdf plot of bmi distribution we can see that – 60% individual have the bmi less or equal 25.
* The age distribution and children distribution shows staircase in ecdf plot.
* So we can conclude that the data set the distribution of above two reflects the probability of encountering a specific points or less than some perticular value of the data set .
* Here the data points are stored so new data arrives and we can see jump in y-axis for any new value.
* The strip plot of age distribution also shows the concentration in kind of younger group of ages.
* It shows the higher density of datapoints in the younger people more accurately around 20-30 years old compare to the older group of peoples.
* Here the possible inferences can be due to any of the following reasons – high birthrate in recent past, may be lots of younger people migrate here, also it can be due to low mortality rate at older ages.
* The strip plot on bmi distribution shows - The data points are distributed throughout the BMI range in the strip plot, with a potential concentration in the overweight (25–30 BMI) area. This may point to a high rate of overweight people in the community.
* This leads to Health Concerns: Obesity and overweight are associated with a number of health issues. In the event that this sample is typical of the broader community, encouraging good weight control techniques may be advantageous.
* The strip plot of charges does not gives a clear understanding or trend that's why we need to build a machine learning model to deal with it.
* The pair plot of Age distribution, Bmi distribution , Children distribution and charge distribution shows similar plots so we can say the numerical data are well distributed.
* The age distribution in kde plot shows positive skewness but for the children distribution , Bmi distribution and charges distribution we cannot comment on its skewness definitely . But these three may have a small or negligible positive skewness .

Data PreProcessing : We know machine only can process numbers but in our dataset we got categorical values . So we have to assign each individual unique feature to certain numeric values so that we can proceed. We can use pythons libraire's panda to do so. We can also use Label Encoding or get dummies methods to do it .But here’s how I did it -

* We have three categorical columns - ‘sex’, ‘smoker’ and ‘region’
* We have to process it so that our machine learning model could proceed.
* data**.**replace({'sex':{'female':0,'male':1}},inplace**=True**) for ‘sex’ column
* The replace() function in Pandas is used in this code to translate the categorical values in the 'sex' column to their numerical counterparts. The mapping of'male' to 1 and 'female' to 0. Training machine learning models and numerical analysis are made possible by this.
* data**.**replace({'smoker':{'yes':1,'no':0}},inplace**=True**)
* This bit of code changes the category values in the'smoker' column to numerical values by using Pandas' replace() method. 'No' is substituted with '0,' and 'yes' with 1. The dataset can now be used for machine learning model training and numerical analysis thanks to this change.
* data**.**replace({'region':{'southwest':1,'southeast':2,'northwest':3,'northeast':4}},inplace**=True**)
* The code supplied converts categorical'region' values to numerical values using Pandas' replace() function. 'Southwest' becomes 1,'southeast' 2, 'northwest' 3, and 'northeast' 4. The dataset can now be used for machine learning model training and numerical analysis thanks to this change.

Building & Saving Machine Learning Models:

* Splitting Feature and Target
* X**=**data**.**drop(['charges'],axis**=**1)
* Here our target is charges and all the other columns are our features.
* Thats why in splitting We split our dataset into features and target
* We store our features into X.
* y**=**data['charges']
* We store our Target into y
* We import **from** sklearn.model\_selection **import** train\_test\_split
* For splitting the trainning and testing data
* Splitting Tainning and testing data
* X\_train,X\_test,y\_train,y\_test**=**train\_test\_split(X,y,test\_size**=**0.2,random\_state**=**42)
* Here we consider the random state 42 as per our convenience which gives highest test accuracy.
* The test size splits 20% and 80%
* Importing necessary models
* **from** sklearn.linear\_model **import** LinearRegression  
  **from** sklearn.svm **import** SVR  
  **from** sklearn.ensemble **import** RandomForestRegressor  
  **from** sklearn.ensemble **import** GradientBoostingRegressor
* We choose these four models that is – Linear Regressor , Support Vectofr Regression, Random Forest Regressor, Gradient Boosting Regressor for this perticular problem set.
* Fitting
* lr**=**LinearRegression()  
  lr**.**fit(X\_train,y\_train)  
  svm**=**SVR()  
  svm**.**fit(X\_train,y\_train)  
  rf**=**RandomForestRegressor()  
  rf**.**fit(X\_train,y\_train)  
  gr**=**GradientBoostingRegressor()  
  gr**.**fit(X\_train,y\_train)
* Now we fit those one by one.
* Prediction Comparism
* We define the prediction of each models by - y\_pred1**=**lr**.**predict(X\_test)  
  y\_pred2**=**svm**.**predict(X\_test)  
  y\_pred3**=**rf**.**predict(X\_test)  
  y\_pred4**=**gr**.**predict(X\_test)
* Data Frame to compare seperate models performance
* Then we made a dataframe to compare the charges of random points using different models by
* df1**=**pd**.**DataFrame({'actual':y\_test,'lr':y\_pred1,'svm':y\_pred2,'rf':y\_pred3,'gr':y\_pred4})
* Visualisation of seperate models peformance
* **import** matplotlib.pyplot **as** plt  
    
    
  fig, axs **=** plt**.**subplots(1, 4, figsize**=**(12, 4))  
    
    
  axs[0]**.**plot(df1["actual"]**.**iloc[0:11], label**=**'Actual')  
  axs[0]**.**plot(df1["lr"]**.**iloc[0:11], label**=**'lr')  
  axs[0]**.**set\_title('LR')  
  axs[0]**.**legend()  
    
    
  axs[1]**.**plot(df1["actual"]**.**iloc[0:11], label**=**'Actual')  
  axs[1]**.**plot(df1["svm"]**.**iloc[0:11], label**=**'svm')  
  axs[1]**.**set\_title('SVM')  
  axs[1]**.**legend()  
    
    
  axs[2]**.**plot(df1["actual"]**.**iloc[0:11], label**=**'Actual')  
  axs[2]**.**plot(df1["rf"]**.**iloc[0:11], label**=**'rf')  
  axs[2]**.**set\_title('RF')  
  axs[2]**.**legend()  
    
    
  axs[3]**.**plot(df1["actual"]**.**iloc[0:11], label**=**'Actual')  
  axs[3]**.**plot(df1["gr"]**.**iloc[0:11], label**=**'gr')  
  axs[3]**.**set\_title('GR')  
  axs[3]**.**legend()  
    
  plt**.**tight\_layout()
* We got -



* We can see that the Svm and LR are not quite good for our model building.
* The best model should be either of the two RF and GR
* For that we need scoring of the models to fetch the best model
* For scoring first we have done
* R Squared Score
* Importing - **from** sklearn **import** metrics
* Definning the scores - score1**=**metrics**.**r2\_score(y\_test,y\_pred1)  
  score2**=**metrics**.**r2\_score(y\_test,y\_pred2)  
  score3**=**metrics**.**r2\_score(y\_test,y\_pred3)  
  score4**=**metrics**.**r2\_score(y\_test,y\_pred4)
* Printing - print(score1,score2,score3,score4)
* We got - 0.806846632262911 -0.13444607209628168 0.8801857051200719 0.900499400599332
* Hence according to r2 score the best model is Gradient Boosting Regressor
* Now we have done another score testing
* Mean Absolute Error
* Definning the scores - s1**=**metrics**.**mean\_absolute\_error(y\_test,y\_pred1)  
  s2**=**metrics**.**mean\_absolute\_error(y\_test,y\_pred2)  
  s3**=**metrics**.**mean\_absolute\_error(y\_test,y\_pred3)  
  s4**=**metrics**.**mean\_absolute\_error(y\_test,y\_pred4)
* Printing - print(s1,s2,s3,s4)
* We got - 4182.3531552883005 9249.561476303981 2596.904554489179 2526.0223170700056
* By this also we get Gradient Boosting Regressor as the best Model
* Now the testing
* Testing on existing data
* data**=**{'age':34,'sex':0,'bmi':31.92,'children':1,'smoker':1,'region':4}  
  df1**=**pd**.**DataFrame(data,index**=**[0])  
  df1
* We fetched the existing data and made a new data frame because we know this models are sensible only in pd data frame.
* Now the main part – we predict this by - pred**=**gr**.**predict(df1)  
  pred
* We get pretty good result - 39655.22344695
* Which is close to the original value
* Now we have to predict new customers charge,
* Thats the whole objective of this project
* We define the new customers data as - data2**=**{'age':41,'sex':0,'bmi':40.23,'children':4,'smoker':0,'region':1}
* made a dataframe - df2**=**pd**.**DataFrame(data2,index**=**[0])  
  df2
* pred1**=**gr**.**predict(df2) - predicting
* It gives - 7624.96025611
* Now saving the Model
* Saving Model Using Joblib
* We import joblib and save the best model for future use -
* gr**=**GradientBoostingRegressor()
* gr**.**fit(X,y) fitting target and features
* **import** joblib
* joblib**.**dump(gr,'model\_joblib\_gradientBoostingRegressor')
* model**=**joblib**.**load('model\_joblib\_gradientBoostingRegressor')
* By above method
* Now making prediction using the saved model
* model**.**predict(df2)
* Here we took the new customer data to predict
* It predicts the same - 7624.96025611
* Hence we able to make a fully working machine learning model to predict the medical insurance cost .

Conclusion:

* The heatmap shows interesting relationships between many elements that could affect charges. This is a look below the surface:
* Age-Charge Connection: It's noteworthy to note that age and charges have a positive link. It might imply:
* Accumulated Healthcare Needs: As people get older, their healthcare requirements and related expenses may increase, which could result in increased fees.
* Lifestyle Decisions: Age may serve as a stand-in for decisions made regarding a lifestyle that may have long-term health effects or result in greater costs down the road.
* Charges and the number of children: Here, the positive association calls for more research. Among the options are:
* Increasing Healthcare Costs: Families with more children may have greater medical costs, which could result in additional fees.
* Socioeconomic Factors: Lower socioeconomic groups may have fewer children in their family, which means they have less access to preventative care and so pay higher in the time of crisis.
* Smoking and Penalties: It makes no sense that there is a negative association. It might result from:
* Underlying Health Conditions: Smokers may already be charged with underlying health conditions, which reduces the likelihood of additional charges. (Warning: Causation is not implied by correlation. Smoking poses serious health hazards.)
* Data Limitations: Without more investigation, it is not advisable to generalise the negative association, which may be the result of particular features of the data set.
* These are the insights from the correlation heatmap.
* I have learned how to remove duplicate values from the dataframe using panda
* I explored the data visualization
* So many new techniques I have learned
* This is a good model which can be used by health professionals and policy maker to take informed decission and optimization.
* For a number of important reasons, Gradient Boosting Regressor (GBR) is the most effective model for estimating medical expenses. First off, GBR is an effective ensemble learning method that builds a strong predictive model by combining the predictive potential of several weak learners—usually decision trees. This makes it possible for GBR to efficiently capture intricate correlations between the target variable (medical costs) and input parameters (such as age, BMI, smoking status, etc.).
* Second, GBR works effectively with data that has both linear and non-linear relationships. Since complex interactions between several parameters are commonly involved in medical cost prediction, GBR's capacity to capture nonlinear patterns makes it especially useful in this setting.
* Additionally, GBR is resistant to overfitting because of methods like regularisation and shrinkage that stop the model from learning from noise in trainning data.