1766

RUTGERS

THE STATE UNIVERSITY
OF NEW JERSEY

Probability and Statistical Inference for Data Science (581)

Final Project

Submitted to Rutgers, the State University of New Jersey
Toward the partial fulfillment of the Award of the Degree of

MSCS/MSDS

2022-2023

By

Shourya Thatha Ravi (st1037)
Rahul Rajesh Singh (rs2050)
Dhruv Patel (dp1224)

Under the Guidance of
Mr. Michael LuValle

Department of Statistics
Rutgers University - New Brunswick

Introduction

Welcome to our Exploratory Data Analysis to Forecast Medical Insurance Prices using Lazy Regressor, Decision Trees, Boosting Methods, etc. In this project, we will be using machine learning techniques to analyze and forecast medical insurance prices. We will be using a dataset containing various features such as age, sex, BMI, number of children, smoker, region, and charges status to predict the cost of medical insurance.

To begin, we will start by performing an exploratory data analysis (EDA) on the dataset to understand the relationships between the different features and the target variable (medical insurance cost). This will involve visualizing the data, cleaning and preprocessing the data, and identifying any potential outliers or anomalies.

After completing the EDA, we will move on to building and training machine learning models using XGBoost, Lazy Regressor, and Decision Trees. We will evaluate the performance of each model and compare their results in order to determine the best approach for predicting medical insurance prices. Overall, this project aims to comprehensively analyze medical insurance prices and demonstrate the potential of machine learning techniques in forecasting such prices.

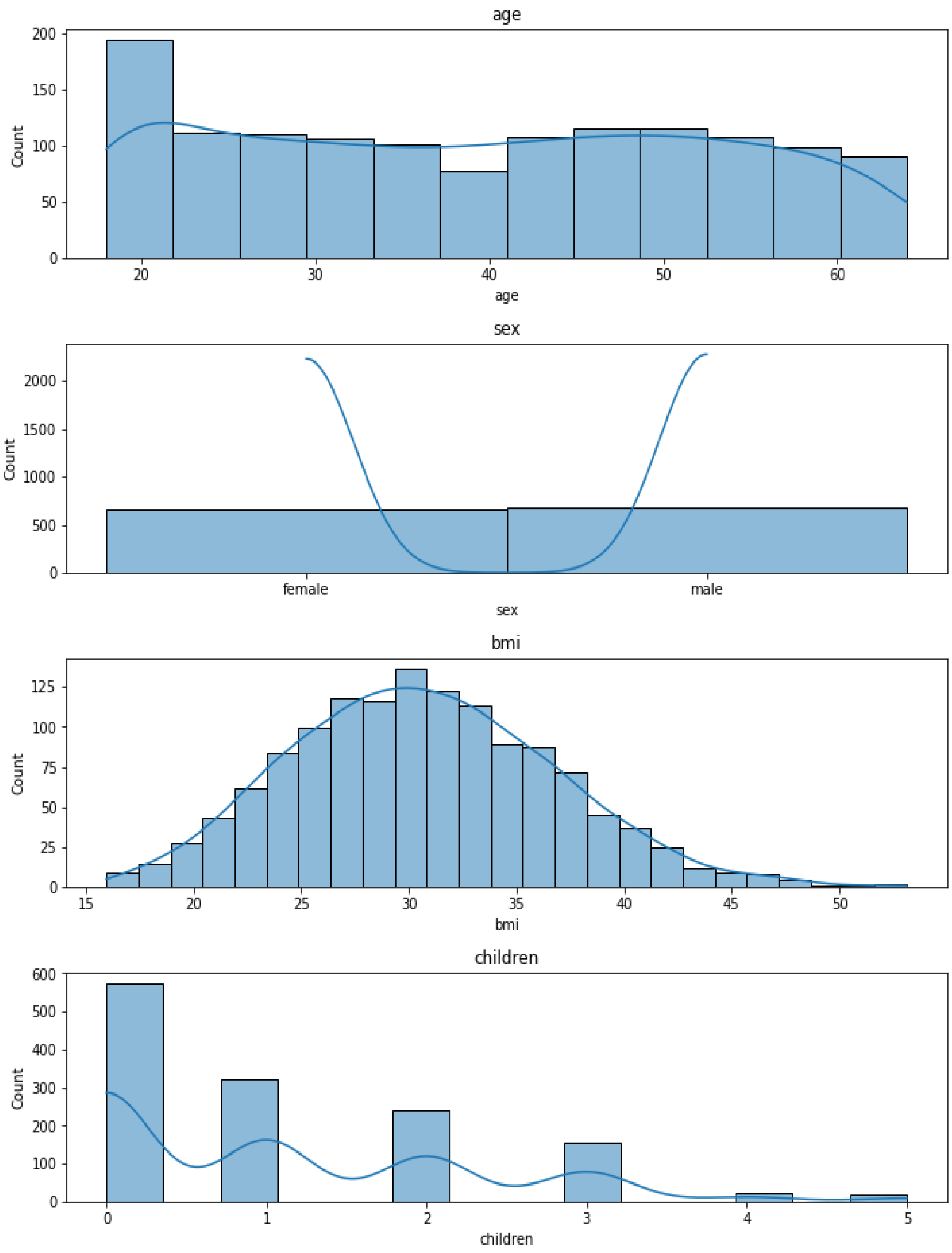
Review of Dataset

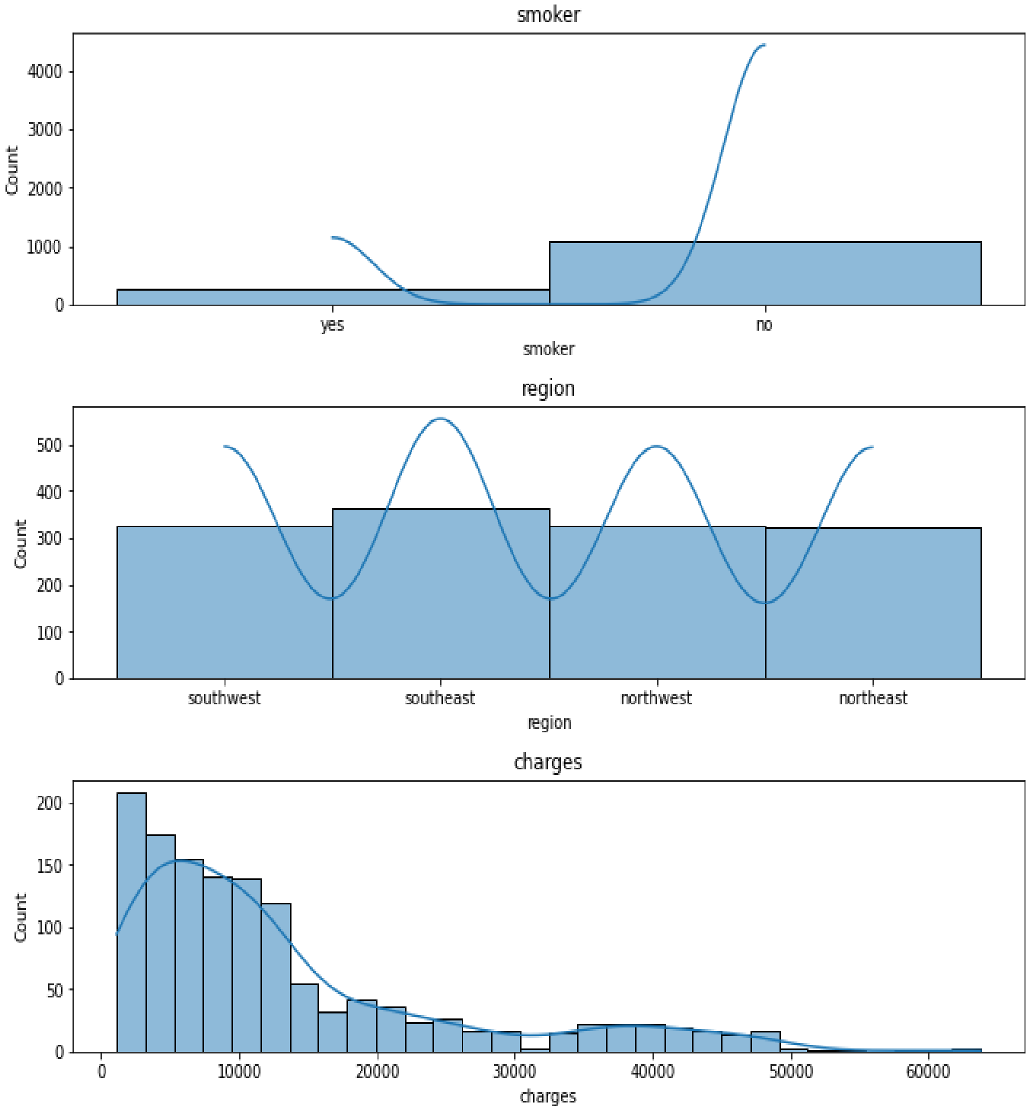
The data points cover 1338 individuals across various regions, ages, and several other factors total of 7 variables.

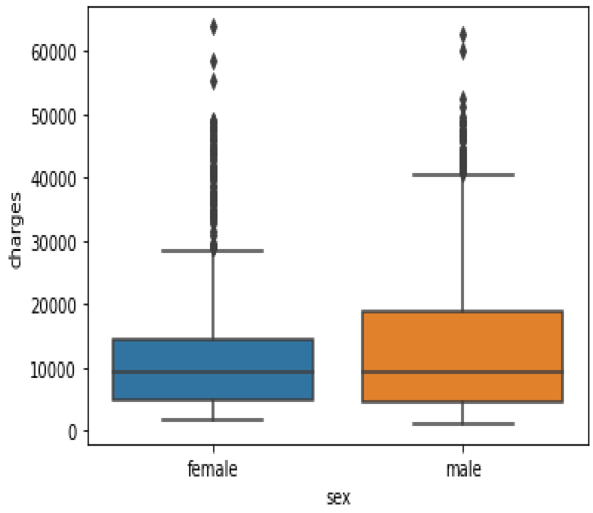
- **Age**: age of primary beneficiary
- **Sex**: insurance contractor gender, female, male
- **BMI**: Body mass index, providing an understanding of body, weights that are relatively high or low relative to height, objective index of body weight (kg / m ^ 2) using the ratio of height to weight, ideally 18.5 to 24.9
- **Children**: Number of children covered by health insurance / Number of dependents
- **Smoker**: Smoking
- **Region**: the beneficiary's residential area in the US, northeast, southeast, southwest, northwest.
- **Charges**: Individual medical costs billed by health insurance

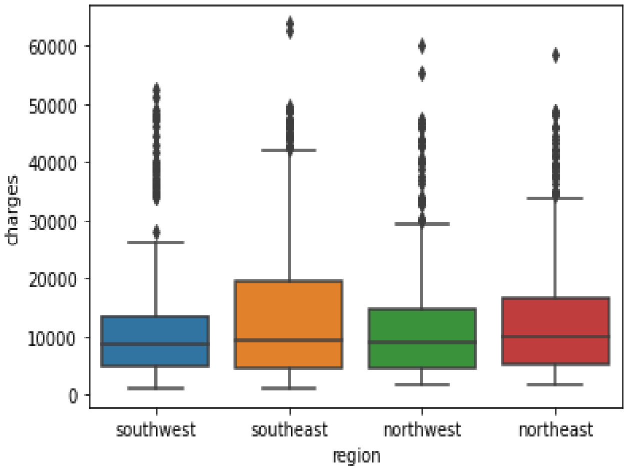
Exploratory Data Analysis

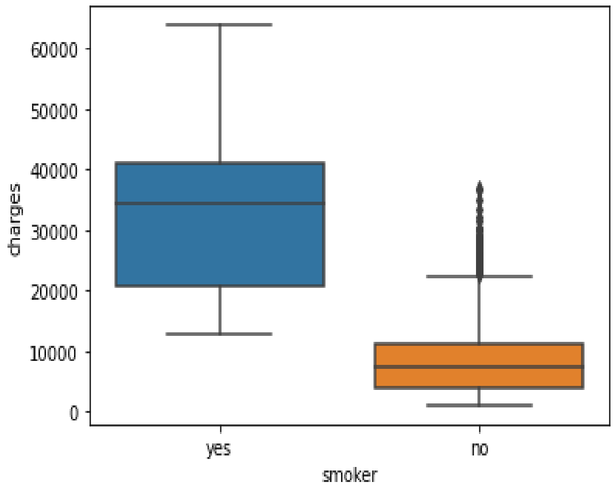
- Plotting the data set in the form of a graph.



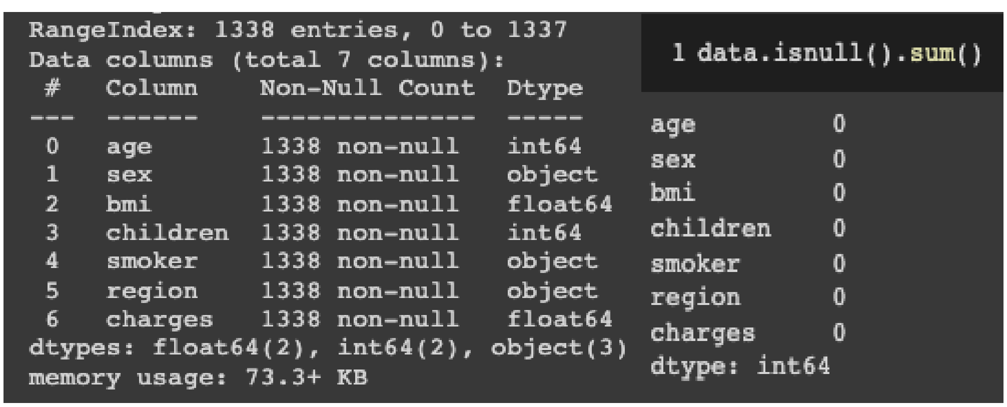

smoker
Count
4000
3000
2000
1000
0
yes
no
smoker
region
Count
500
400
300
200
100
0
southwest
southeast
northwest
northeast
region
charges
Count
200
150
100
50
0
0
10000
20000
30000
40000
50000
60000
charges







- Huge range and values in general for smokers when compared to non-smokers
- Below is a glimpse of information of our data and no missing values:



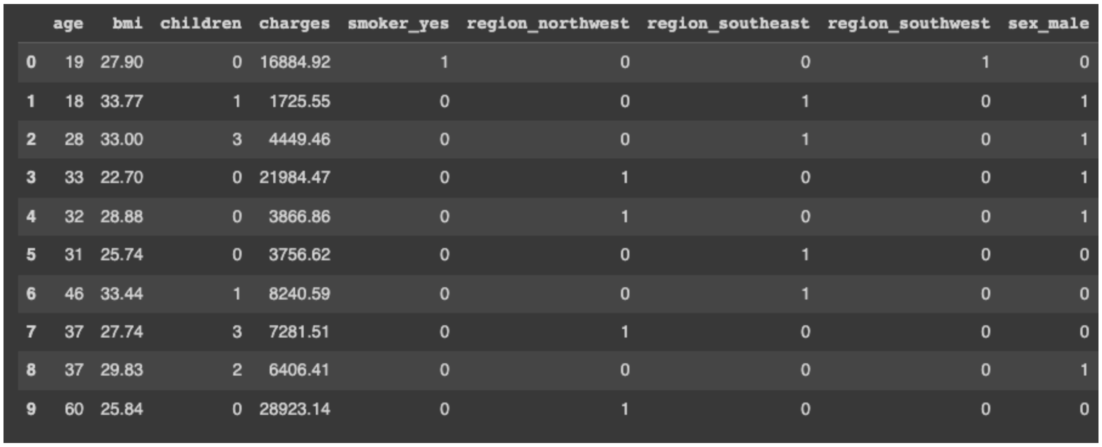
```
RangeIndex: 1338 entries, 0 to 1337
Data columns (total 7 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       1338 non-null   int64
 1   sex       1338 non-null   object
 2   bmi       1338 non-null   float64
 3   children  1338 non-null   int64
 4   smoker    1338 non-null   object
 5   region    1338 non-null   object
 6   charges   1338 non-null   float64
dtypes: float64(2), int64(2), object(3)
memory usage: 73.3+ KB
```

```
1 data.isnull().sum()
```

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

One Hot Encoding

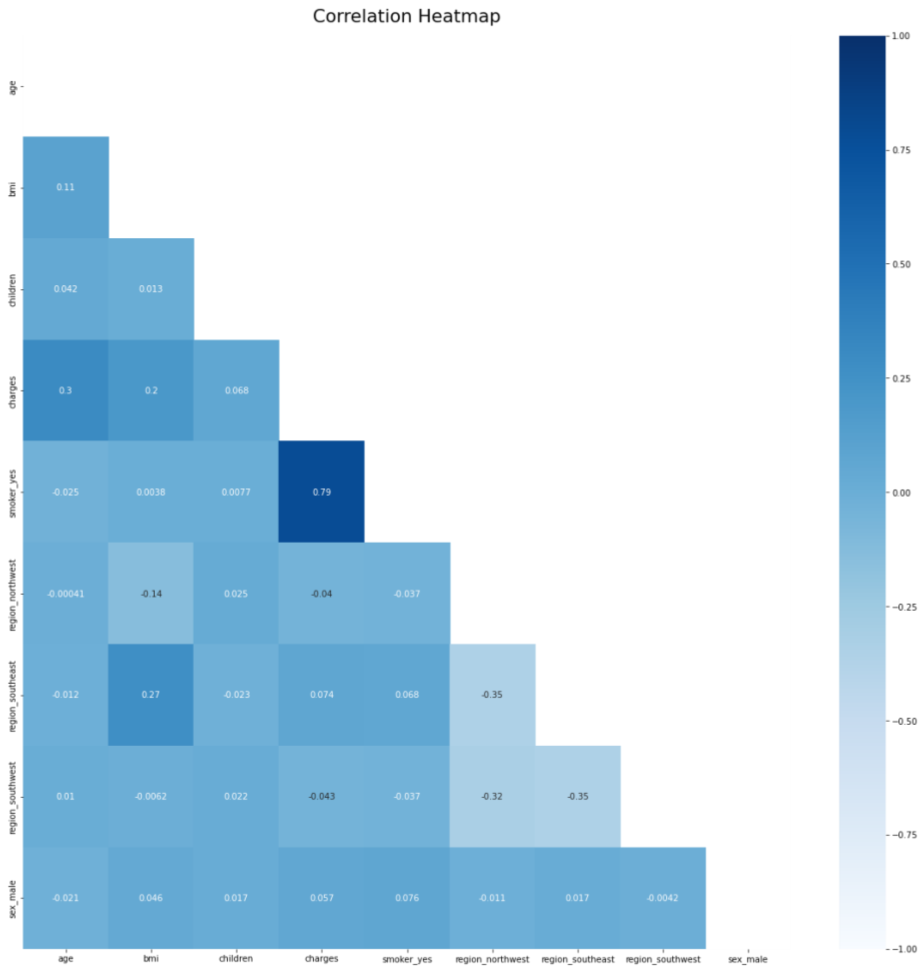
- Using One Hot Encoding to convert the categorical data variables to improve predictions and classification accuracy of a model. Dropping smoker_no , region_northeast, sex_female to remove multicollinearity.



	age	bmi	children	charges	smoker_yes	region_northwest	region_southeast	region_southwest	sex_male
0	19	27.90	0	16884.92	1	0	0	1	0
1	18	33.77	1	1725.55	0	0	1	0	1
2	28	33.00	3	4449.46	0	0	1	0	1
3	33	22.70	0	21984.47	0	1	0	0	1
4	32	28.88	0	3866.86	0	1	0	0	1
5	31	25.74	0	3756.62	0	0	1	0	0
6	46	33.44	1	8240.59	0	0	1	0	0
7	37	27.74	3	7281.51	0	1	0	0	0
8	37	29.83	2	6406.41	0	0	0	0	1
9	60	25.84	0	28923.14	0	1	0	0	0

Correlation Heatmap

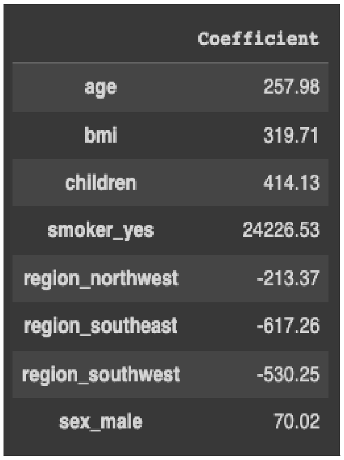
- Almost no correlation for most of the variables
- Smoker_Yes highly correlated with charges i.e. the target variable
- The least correlated variables are the region ones, which is intuitive



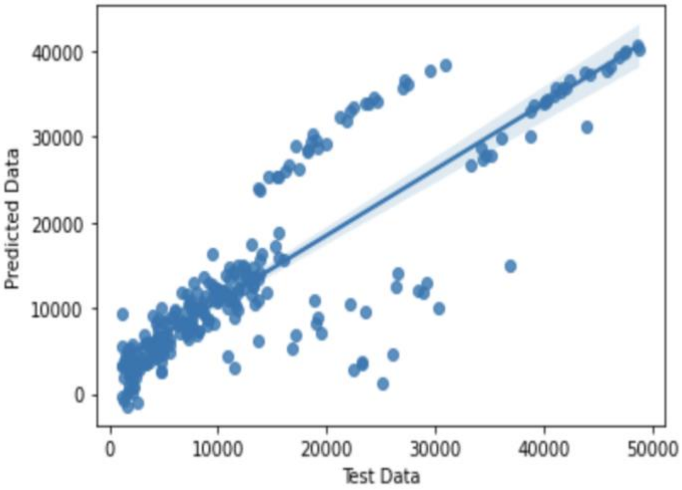
Modeling Methodologies

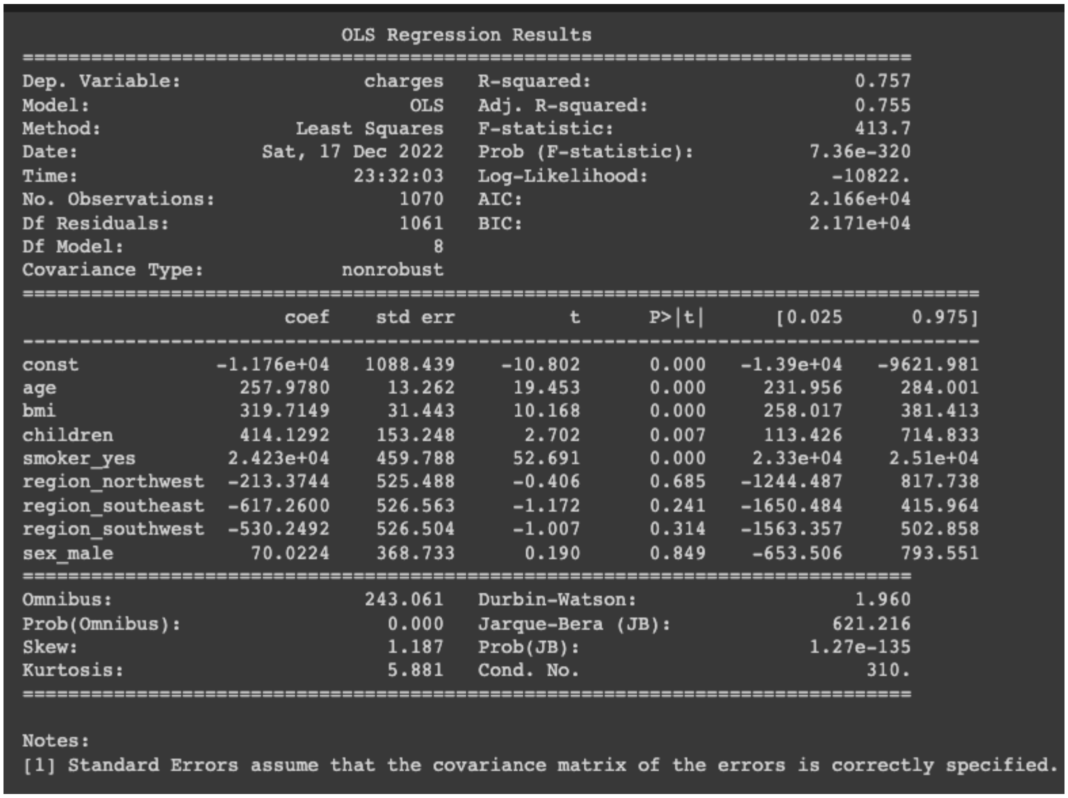
1. Multilinear Regression

- The R Squared and adj. R2 for the linear model are ~0.75
- Based on the p values, it can be noted that the regions and sex of an individual are statistically the least/ not significant
- Other variables like age, bmi etc. are significant
- Low Value of Prob F-Statistic points out that the model in itself is significant
- Among the variables, Smoker_yes seems to have the highest coefficient, implying that this binary variable heavily influences our data. The other remaining variables vary positively with the target variable



	Coefficient
age	257.98
bmi	319.71
children	414.13
smoker_yes	24226.53
region_northwest	-213.37
region_southeast	-617.26
region_southwest	-530.25
sex_male	70.02





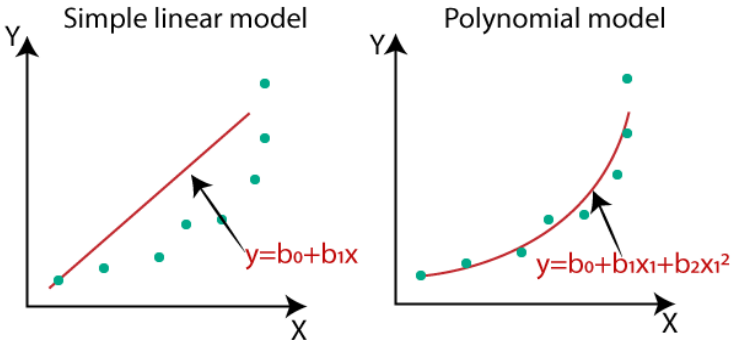
```
                            OLS Regression Results
==============================================================================
Dep. Variable:                charges   R-squared:                       0.757
Model:                            OLS   Adj. R-squared:                  0.755
Method:                 Least Squares   F-statistic:                     413.7
Date:                Sat, 17 Dec 2022   Prob (F-statistic):          7.36e-320
Time:                        23:32:03   Log-Likelihood:                -10822.
No. Observations:                1070   AIC:                         2.166e+04
Df Residuals:                    1061   BIC:                         2.171e+04
Df Model:                           8
Covariance Type:            nonrobust
====================================================================================
                           coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------------
const            -1.176e+04   1088.439    -10.802      0.000   -1.39e+04   -9621.981
age                257.9780     13.262     19.453      0.000     231.956     284.001
bmi                319.7149     31.443     10.168      0.000     258.017     381.413
children           414.1292    153.248      2.702      0.007     113.426     714.833
smoker_yes        2.423e+04    459.788     52.691      0.000     2.33e+04    2.51e+04
region_northwest  -213.3744    525.488     -0.406      0.685   -1244.487     817.738
region_southeast  -617.2600    526.563     -1.172      0.241   -1650.484     415.964
region_southwest  -530.2492    526.504     -1.007      0.314   -1563.357     502.858
sex_male            70.0224    368.733      0.190      0.849    -653.506     793.551
==============================================================================
Omnibus:                      243.061   Durbin-Watson:                   1.960
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              621.216
Skew:                           1.187   Prob(JB):                    1.27e-135
Kurtosis:                       5.881   Cond. No.                         310.
==============================================================================

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

2. Polynomial Regression

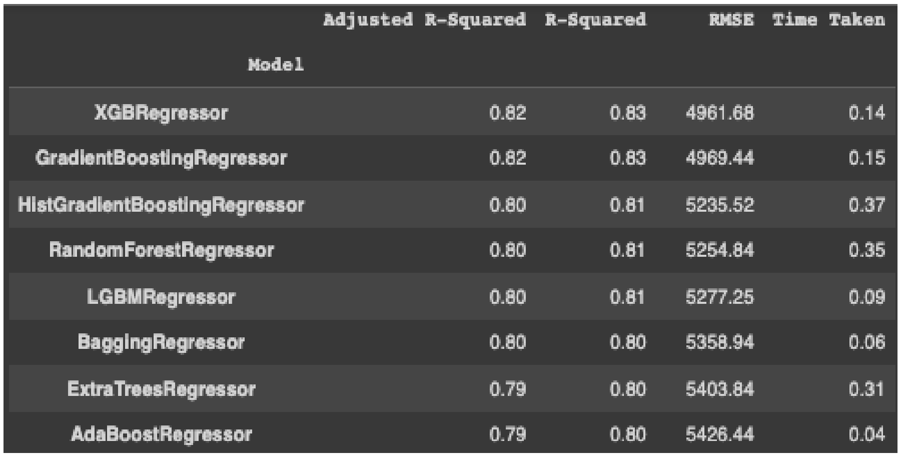
- In attempts to achieve a better model, polynomial regression is implemented after linear regression
- Degrees upto 2 is used which yields a model with 45 variables, after all the possible combinations of variables
- After which, Recursive Feature Elimination(RFE) is applied as the feature selection model, and a reduced dataset is retrieved

- Thus, after applying Linear Regression to this dataset, we achieve polynomial Regression



3. Lazy Regressor

- It is one of the best python libraries that helps you to semi-automate your Machine Learning Task. It builds a lot of basic models without much code and helps understand which models work better without any parameter tuning.



Model	Adjusted R-Squared	R-Squared	RMSE	Time Taken
XGBRegressor	0.82	0.83	4961.68	0.14
GradientBoostingRegressor	0.82	0.83	4969.44	0.15
HistGradientBoostingRegressor	0.80	0.81	5235.52	0.37
RandomForestRegressor	0.80	0.81	5254.84	0.35
LGBMRegressor	0.80	0.81	5277.25	0.09
BaggingRegressor	0.80	0.80	5358.94	0.06
ExtraTreesRegressor	0.79	0.80	5403.84	0.31
AdaBoostRegressor	0.79	0.80	5426.44	0.04

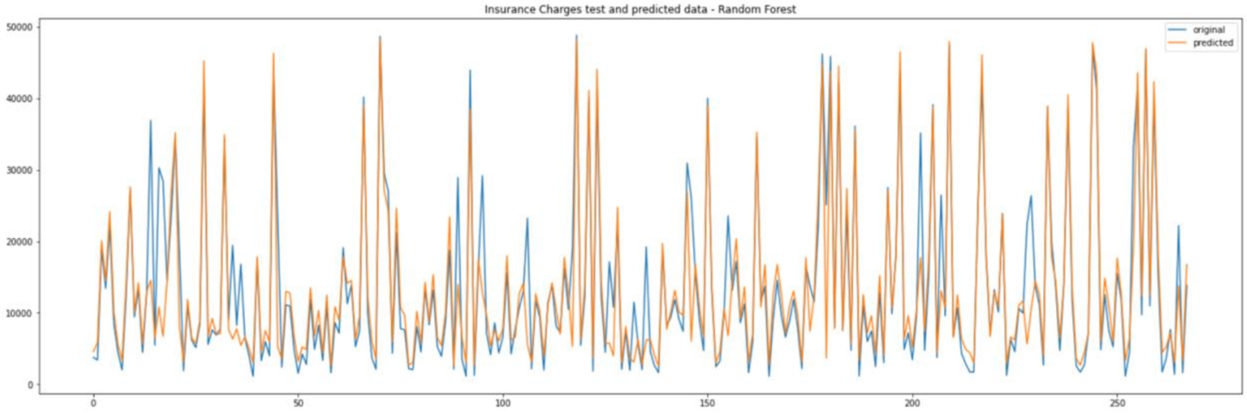
HyperParameter Tuning using Gridsearch

- Hyperparameter tuning is the process of finding the optimal values for the hyperparameters of a machine learning model. Hyperparameters are the parameters of the model that are set before training, and they can significantly affect the performance of the model.
- Grid search is a method of hyperparameter tuning that involves specifying a grid of hyperparameter values and training the model for each combination of hyperparameter values. The performance of the model is then evaluated using a validation set, and the combination of hyperparameters that results in the best performance is chosen as the optimal set of hyperparameters.

- A Grid Search is an exhaustive search over every combination of specified parameter values. If you specify 2 possible values for max_depth and 3 for n_estimators, Grid Search will iterate over 6 possible combinations. Sometimes tuning our model using our intuition will suffice. Additionally, optimization algorithms like GridSearch and RandomSearch are worthwhile to attempt. Most RandomizedSearchCV's parameters are similar to GridSearchCV's.

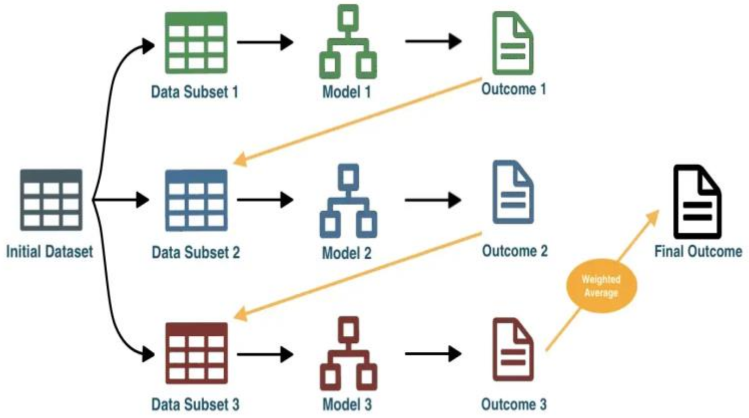
4. Random Forest Regression

- Random forest regression is an ensemble machine learning technique that is used to build predictive models by training multiple decision trees on different subsets of the training data and then averaging the predictions of the individual trees. It is a type of supervised learning, which means that the model is trained on labeled data that includes both the input features and the target variable.
- In a random forest regression model, each decision tree is trained on a random subset of the training data, and the predictions of the individual trees are combined using a mean or median aggregation. This approach is used to reduce the variance of the model and improve the overall accuracy of the predictions
- We used the sklearn module for training our random forest regression model, specifically the Random Forest Regressor function. The Random Forest Regressor documentation shows many different parameters we can select for our model. Some of the important parameters are highlighted below:
    - **a)** **n_estimators** — the number of decision trees you will be running in the model.
    - **b)** **criterion** — this variable allows you to select the criterion (loss function) used to determine model outcomes. We can select from loss functions such as mean squared error (MSE) and mean absolute error (MAE). The default value is MSE.
    - **c)** **max_depth** — this sets the maximum possible depth of each tree.



5. Boosting

- Boosting merely employs the ensemble learning concept in a sequential manner. The methodology integrates judgments from various underlying models and makes a final forecast by using a voting method.
- Bagging and random forests are two well-known ensemble learning techniques.
  Boosting is a sort of ensemble learning where the output from one model is fed into the next. Boosting trains models sequentially rather than separately, with each new model being trained to fix the flaws of the preceding ones.
- The outcomes that were successfully predicted are given a lower weight at each iteration, whereas the ones that were incorrectly forecasted are given a higher weight. It then calculates a final result using a weighted average.



$$MSE = \frac{1}{n}\Sigma\underbrace{\left(y - \hat{y}\right)^2}_{\text{The square of the difference between actual and predicted}}$$

6. Gradient Boosting:

- Gradient Boosting is a boosting technique that uses a gradient descent approach to decrease mistakes. Gradient descent is a straightforward iterative optimization approach for reducing a loss function.
- The loss function measures how far off from the actual result our forecast is for a particular data point. The output of our loss function will be lower the better the forecasts are.
- Our model's construction aims to minimize the loss function over each data point. For instance, the most frequently used loss function for regression is Mean Squared Error (MSE).
- Gradient boosting increases the weight of incorrectly anticipated results and modifies those weights according to a gradient, which is determined by the direction in which the loss "decreases the fastest" in the loss function.
- The process of training a gradient boosting model involves minimizing a loss function

using gradient descent. Gradient descent is an optimization algorithm used to find the optimal set of parameters for a model by iteratively updating the model's parameters in a direction that minimizes the loss function.

7. Extreme Gradient Boosting (XGBoost):

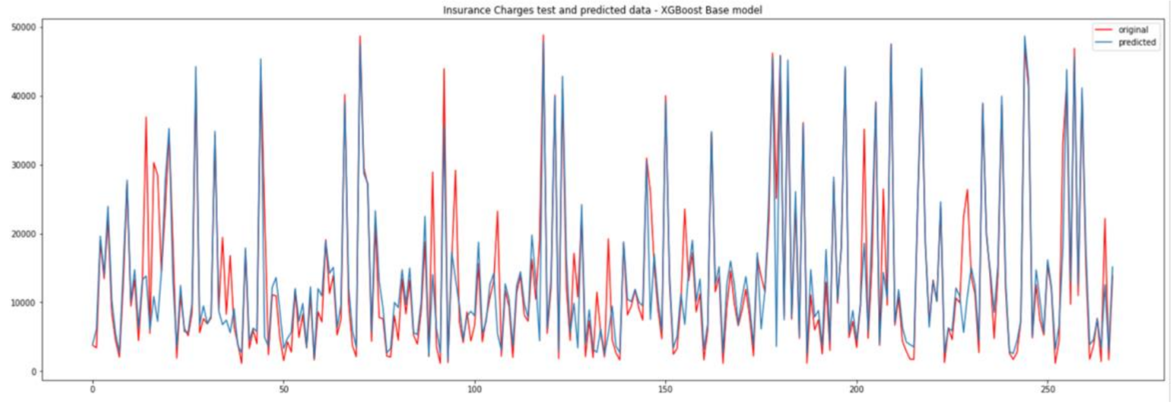
- Extreme Gradient Boosting, or XGBoost, is an open-source library as well as an algorithm that was created as an improved version of the Gradient Boosting framework. It emphasizes efficiency, adaptability, and model performance. Its strength comes from the entire underlying system optimization (parallelization, caching, hardware optimization, etc.), not just the method itself.
- At a high level, XGBoost works by training an ensemble of decision trees, each of which is trained to make predictions that are based on the mistakes made by the previous trees.
- This is similar to how gradient boosting works, but XGBoost includes several additional features that make it more efficient and effective at building accurate models.
- The most commonly used are:

  a) **reg:logistic:** for logistic regression.

  b) **binary:logistic:** for logistic regression with output of the probabilities.

  c) **reg:squarederror:** for linear regression.

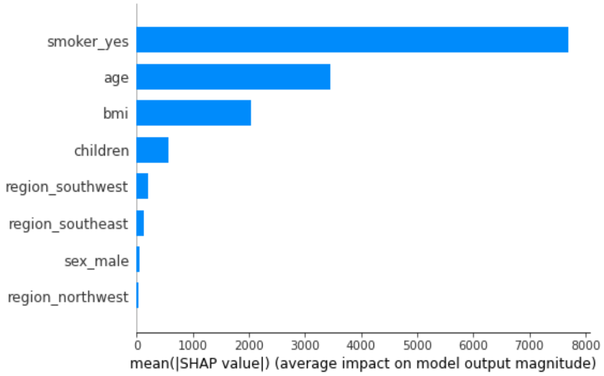
Plotting insurance charges test and predicted data:

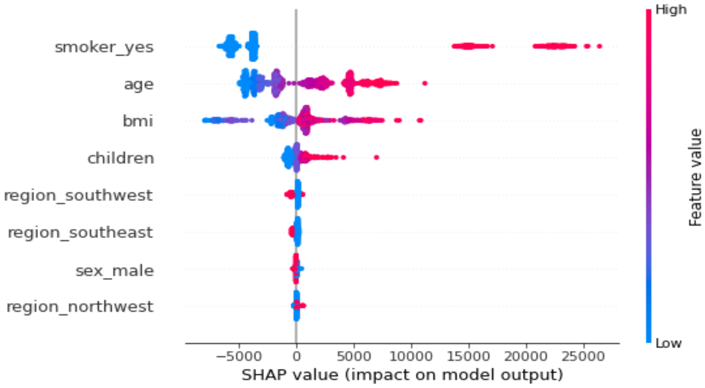


SHAP PLOTS

- The above plots are based on SHapley Additive exPlanations (SHAP) which explain the feature importance for the calculated XGBoost Model
- As seen in the left bar, Smoking is the feature which affects the insurance costs the most, followed by age and bmi whereas regions and sex are almost insignifcant

- The beeswarm plot indicates positive shap value for smoking and negative for not smoking





RESULTS COMPARISON

Model	RMSE	MAPE	R Squared
Multilinear Regression	6333.25	0.19	0.75
Polynomial Regression	5197.51	0.12	0.79
Decision Trees	5284.07	0.14	0.86
Random Forest	4994.38	0.18	0.87
XGBoost	4961.68	0.17	0.89
XGBoost using Randomized Search	5193.12	0.13	0.91
XGBoost using Grid Search	5034.57	0.19	0.88

- From the above seen models, we can note that Boosting Models Performed the best, especially XGBoost
- Among all the factors, Smoking seems to be the most influential one being positively correlated with insurance charges
- It is followed by Age and BMI whereas, surprisingly, the sex of an individual doesn't contribute much to the costs

FUTURE SCOPE

We can try out class balancing – selecting certain rows from smoker class and balancing it equally in the test dataset. We can also select features based on SHAP values and model further to observe better MAPE and lower RMSE.