Contents

Healthcare Insurance Premium Prediction

Problem Statement

The Healthcare insurance industry faces the challenge of balancing fair premium rates with financial sustainability. This report explores the development of a predictive model to estimate health insurance premiums using demographic and lifestyle data. The model leverages machine learning techniques to provide personalized premium recommendations, enhancing both customer satisfaction and insurer efficiency.

Dataset Structure

Insurance Dataset for Predicting Health Insurance Premiums in the US" is a collection of data on various factors that can influence medical costs and premiums for health insurance in the United States. The dataset includes information on 10 variables,

Age: Age of primary beneficiary

Gender: Insurance contractor gender, female, male

Body mass index (BMI): Body mass index, providing an understanding of body, weights that are relatively high or low relative to height

Number of children: Number of children covered by health insurance / Number of dependents

Smoking status: Yes/Not

Region: the beneficiary's residential area in the US, northeast, southeast, southwest, northwest

Income: Earnings of the insurance holder

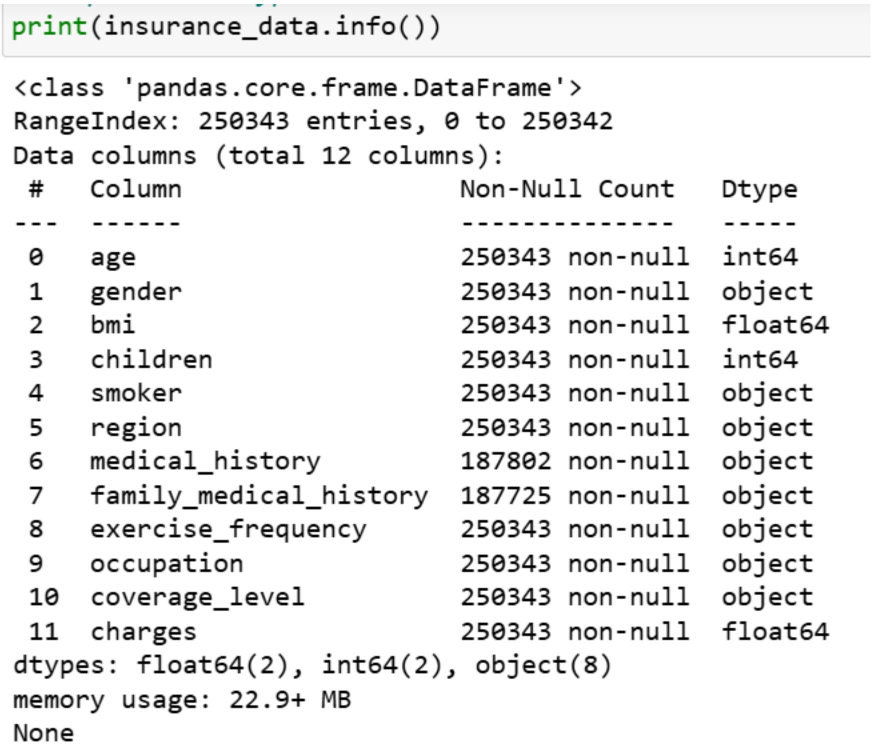
Education: Degree/Education of the insurance holder

Occupation: Profession of the insurance holder

Type of insurance plan: The category of health insurance plan selected

Charges: Premium amount paid for healthcare coverage

The dataset was created using a script that generated a million records of randomly sampled data points, ensuring that the data represented the population of insured individuals in the US. The dataset can be used to build and test machine learning models for predicting insurance premiums and exploring the relationship between different factors and medical costs.



```
print(insurance_data.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 250343 entries, 0 to 250342
Data columns (total 12 columns):
 #   Column                 Non-Null Count   Dtype  
---  ------                 --------------   -----  
 0   age                    250343 non-null  int64  
 1   gender                 250343 non-null  object 
 2   bmi                    250343 non-null  float64
 3   children               250343 non-null  int64  
 4   smoker                 250343 non-null  object 
 5   region                 250343 non-null  object 
 6   medical_history        187802 non-null  object 
 7   family_medical_history 187725 non-null  object 
 8   exercise_frequency     250343 non-null  object 
 9   occupation             250343 non-null  object 
 10  coverage_level         250343 non-null  object 
 11  charges                250343 non-null  float64
dtypes: float64(2), int64(2), object(8)
memory usage: 22.9+ MB
None
```

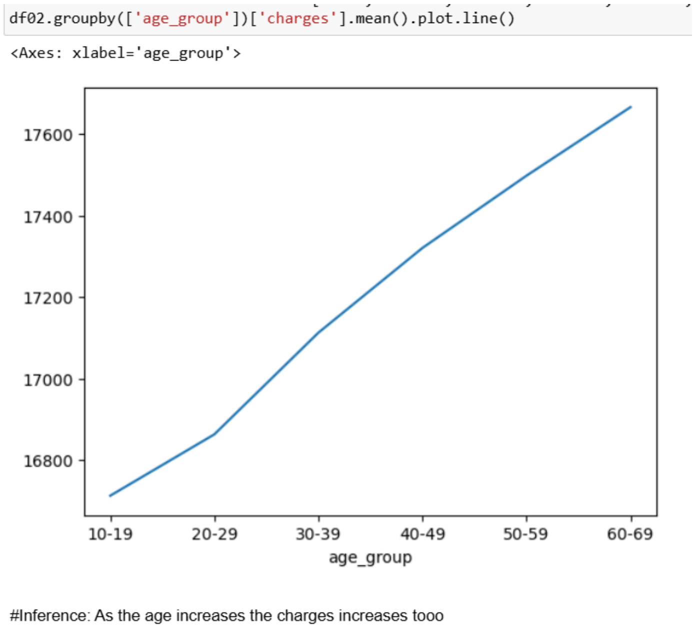
Data Wrangling

We took the below steps for Data Cleaning Data Transformation:

- Checked for duplicates, we found no duplicates in out dataset
- Checked for missing values, we found that medical_history & family_medical_history had missing values, therefore we filled these values with 'No history'
- We checked if columns 'BMI' or 'Charges' have outliers, we found no outliers in these columns
- We derived is_active column & weight_status column from exercise_frequency & bmi columns to better get the insights from data

Exploratory Data Analysis

Below are the Exploratory Data Analysis we performed on Healthcare Insurance dataset:

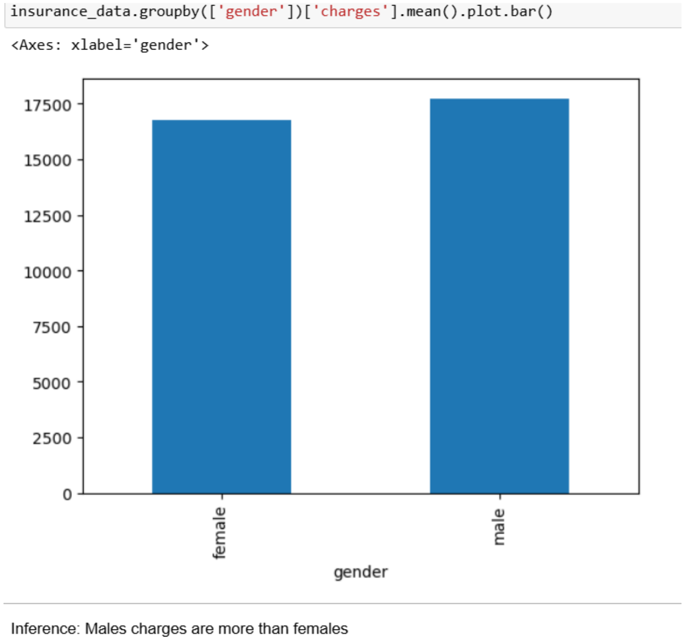


```
df02.groupby(['age_group'])['charges'].mean().plot.line()
```

```
<Axes: xlabel='age_group'>
```

#Inference: As the age increases the charges increases tooo

As the age increases of population the premiums/charges too go up.

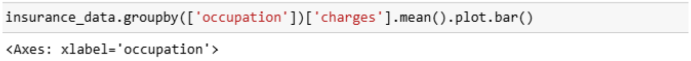


```
insurance_data.groupby(['gender'])['charges'].mean().plot.bar()
```

```
<Axes: xlabel='gender'>
```

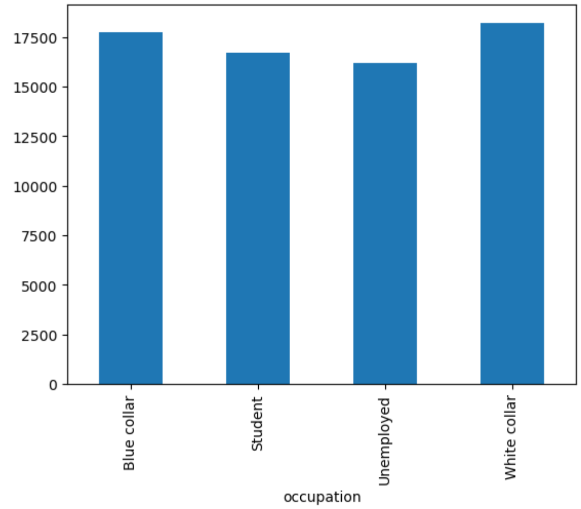
Inference: Males charges are more than females

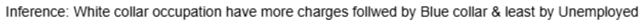
Charges charged to male's are more than females



```
insurance_data.groupby(['occupation'])['charges'].mean().plot.bar()
```

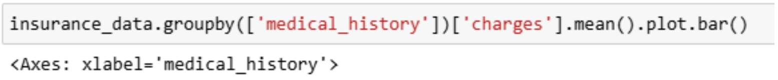
```
<Axes: xlabel='occupation'>
```





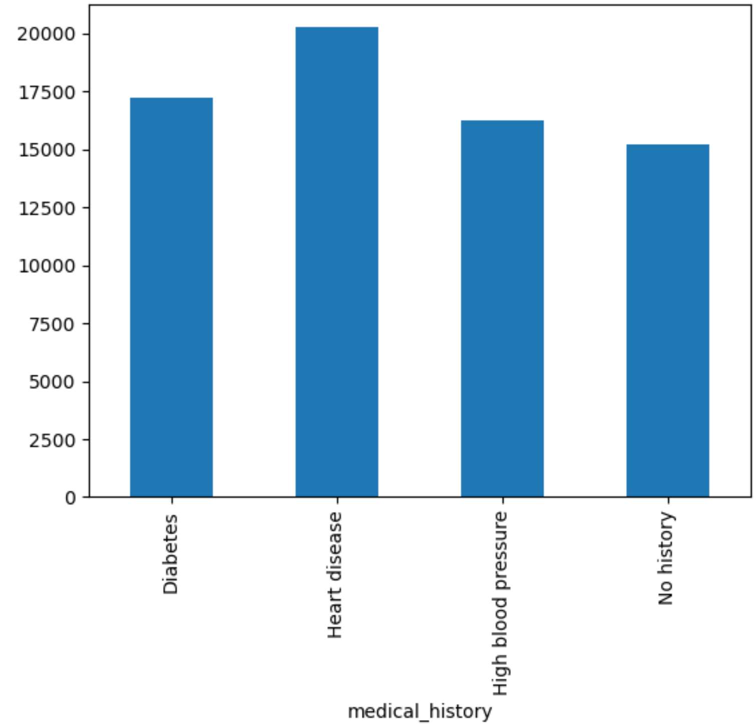
Inference: White collar occupation have more charges follwed by Blue collar & least by Unemployed

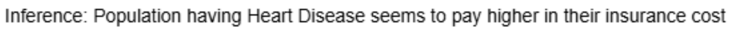
White collar occupation people pay Highest charges & Unemployed & Student pays the lowest.



```
insurance_data.groupby(['medical_history'])['charges'].mean().plot.bar()
```

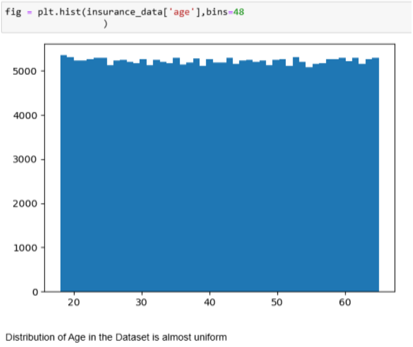
```
<Axes: xlabel='medical_history'>
```





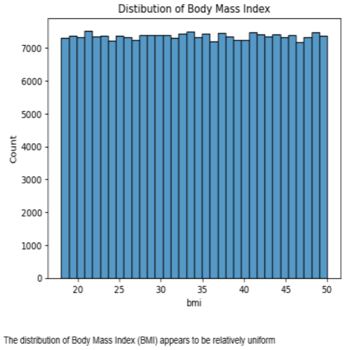
Inference: Population having Heart Disease seems to pay higher in their insurance cost

Population Having Heart disease pays higher charges

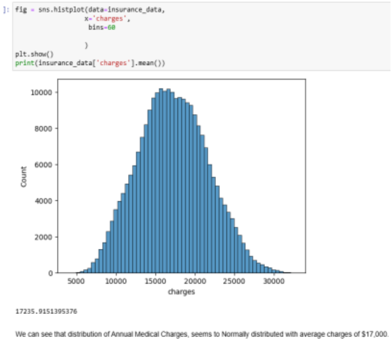


```
fig = plt.hist(insurance_data['age'],bins=48
              )
```

Distribution of Age in the Dataset is almost uniform



The distribution of Body Mass Index (BMI) appears to be relatively uniform



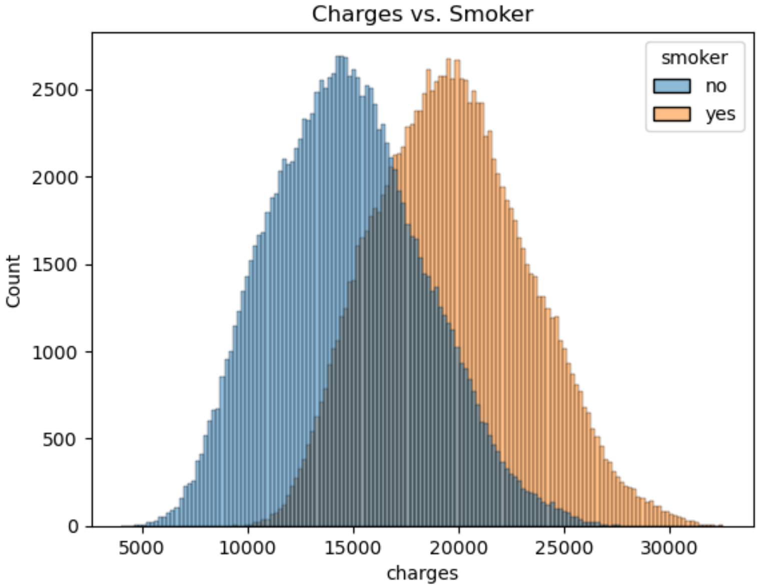
```
]: fig = sns.histplot(data=insurance_data,
                  x='charges',
                   bins=60

                  )
   plt.show()
   print(insurance_data['charges'].mean())
```

```
17235.9151395376
```

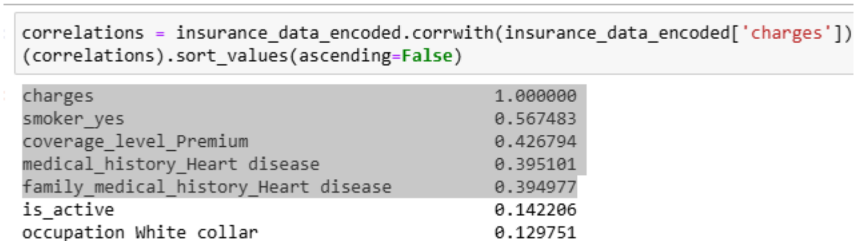
We can see that distribution of Annual Medical Charges, seems to Normally distributed with average charges of $17,000.

Distribution of Age & BMI in the dataset appears to be uniform as per above visualization & charges have uniform normal distribution with average charges to $17,000



The above visualization shows smoker vs non-smoker graph, from the above visualization there is a significant difference in median charges of the people who smoke & who don’t i.e

- Median charges of population who Smokes = $19000
- Median charges of population who doesn’t smoke = $14000



```
correlations = insurance_data_encoded.corrwith(insurance_data_encoded['charges'])
(correlations).sort_values(ascending=False)
```

```
charges                                     1.000000
smoker_yes                                  0.567483
coverage_level_Premium                      0.426794
medical_history_Heart disease               0.395101
family_medical_history_Heart disease        0.394977
is_active                                   0.142206
occupation White collar                     0.129751
```

People who smoke, takes premium coverage level, family/medical history of heart disease make most contribution to charges

Summary of the EDA:

- Performed basic EDA, results are as mentioned below:
    - Advanced age, male gender, heart disease & white-collar job occupation are associated with higher insurance cost
    - Medical/family history of high blood pressure has least correlation with heart problem people. Family/medical history of High blood pressure do not have heart problems
    - People who smoke, takes premium coverage level, family/medical history of heart disease make most contribution to charges

Feature Engineering

Feature engineering is the process of transforming raw data into meaningful features that improve the performance of a predictive model. It involves selecting, modifying, and creating new features to enhance the model's accuracy and efficiency. This step is crucial because it helps remove redundant information, reduces noise, and ensures that only relevant variables contribute to predictions. By eliminating multicollinearity and statistically insignificant features, we can build a more robust and interpretable model.

Feature Engineering Steps:

1. **Categorical Data Processing:**
    - Converted categorical variables (e.g., gender, smoker, region, children, occupation, etc.) into categorical data types.
    - Applied one-hot encoding to convert categorical variables into numerical representations, dropping the first category to avoid multicollinearity.

2. **Correlation & Multicollinearity Analysis:**

- Correlation measures the strength of the relationship between two variables, with values above **±0.75** indicating strong correlation
- Multicollinearity, detected using the Variance Inflation Factor (VIF), is considered problematic when **VIF > 5**, leading to the removal of highly correlated features to improve model stability and interpretability.
- Identified highly correlated independent variables such as *weight_status_obese & BMI* and *is_active_True & exercise_frequency_Never*.
- Used the Variance Inflation Factor (VIF) to detect multicollinearity and removed highly correlated features (*is_active_True* and *exercise_frequency_Rarely*).

3. **Feature Selection:**
   - Feature selection is the process of identifying and retaining the most relevant variables while removing redundant or insignificant ones to improve model performance. It is performed to enhance prediction accuracy, reduce overfitting, improve interpretability, and optimize computational efficiency by ensuring that only meaningful features contribute to the model.
   - Built an Ordinary Least Squares (OLS) regression model to determine statistical significance.
   - Removed features with high p-values (e.g., *weight_status_obese* and *weight_status_overweight*) to improve model performance.
4. **Final Feature Set:**
   - Dropped unnecessary columns: *weight_status_obese*, *weight_status_overweight*, *is_active_True*, and *exercise_frequency_Rarely*.
   - Ensured that the remaining variables effectively contribute to the prediction of insurance charges.

Data Modelling

Data modeling is the process of creating a structured representation of data and its relationships to facilitate analysis, storage, and retrieval. It plays a crucial role in data science, machine learning, and database management by defining how data should be organized, processed, and utilized for decision-making. Various techniques are used in data modeling, including regression analysis, decision trees, ensemble learning (such as Random Forest and XGBoost), and boosting methods like Gradient Boosting. Each method has its strengths and limitations; for instance, linear regression is suitable for linear relationships, while tree-based models handle complex, non-linear patterns. Hyperparameter tuning further optimizes these models for better accuracy and generalization. Effective data modeling ensures that data-driven insights are reliable, leading to improved business strategies and operational efficiencies.

1. Linear Regression

Linear Regression is a statistical technique used for predictive modeling by establishing a linear relationship between the dependent and independent variables.

Accuracy:

The North-East region has a Train Accuracy of 98.12% and a Test Accuracy of 98.11%.

The other regions' accuracy values are missing, but generally, linear regression tends to work well for problems with linear relationships but may underperform in cases with complex, non-linear relationships.

Comparison:

Linear Regression performs well in the North-East region but may not be as effective for other regions, given the absence of reported accuracies.

It is outperformed by tree-based models when data is non-linear.

2. Decision Tree

Decision Trees split the dataset into hierarchical decisions based on the features, making it highly interpretable but prone to overfitting.

Accuracy:

North-East: 99.99% (Train), 95.39% (Test)

North-West: 92.95% (Test)

South-East: 94.19% (Test)

South-West: 92.16% (Test)

Comparison:

Overfitting is evident as the training accuracy is significantly higher than test accuracy.

Performs worse than hyperparameter-tuned models in terms of test accuracy.

3. Decision Tree with Hyperparameter Tuning

This version of Decision Tree involves optimizing parameters (e.g., depth, splitting criteria) to reduce overfitting.

Accuracy:

North-East: 98.26% (Train), 96.99% (Test)

North-West: 94.53% (Test)

South-East: 95.77% (Test)

South-West: 94.51% (Test)

Comparison:

Improvement in test accuracy compared to the default Decision Tree.

Less overfitting, leading to better generalization.

4. Random Forest

Random Forest is an ensemble learning technique using multiple decision trees to improve model stability and accuracy.

Accuracy:

North-East: 99.63% (Train), 97.41% (Test)

North-West: 94.96% (Test)

South-East: 96.20% (Test)

South-West: 94.21% (Test)

Comparison:

Higher accuracy than Decision Trees.

More robust, reducing overfitting by averaging multiple trees.

5. Random Forest with Hyperparameter Tuning

An optimized version of Random Forest with tuned parameters.

Accuracy:

North-East: 99.32% (Train), 97.63% (Test)

North-West: 95.11% (Test)

South-East: 96.37% (Test)

South-West: 94.35% (Test)

Comparison:

Slight improvement over default Random Forest.

More stability in test accuracy.

6. XGBoost Model

XGBoost (Extreme Gradient Boosting) is a high-performance boosting algorithm that iteratively improves model predictions.

Accuracy:

North-East: 98.10% (Train), 98.03% (Test)

North-West: 95.49% (Test)

South-East: 96.75% (Test)

South-West: 94.74% (Test)

Comparison:

High generalization ability and better test accuracy than Random Forest.

Lower overfitting risk compared to single decision trees.

7. XGBoost with Hyperparameter Tuning

A refined version of XGBoost with tuned hyperparameters.

Accuracy:

North-East: 98.08% (Train), 98.06% (Test)

North-West: 95.52% (Test)

South-East: 96.77% (Test)

South-West: 94.77% (Test)

Comparison:

Marginal improvement over default XGBoost.

Better generalization than Random Forest.

8. Gradient Boosting Regression

Gradient Boosting is another boosting technique that builds models sequentially to minimize errors.

Accuracy:

North-East: 97.83% (Train), 97.81% (Test)

North-West: 95.26% (Test)

South-East: 96.53% (Test)

South-West: 94.51% (Test)

Comparison:

Performs similarly to XGBoost but may be slightly slower.

Well-suited for structured data.

9. Gradient Boosting with Hyperparameter Tuning

Optimized Gradient Boosting model.

Accuracy:

North-East: 98.05% (Train), 98.03% (Test)

North-West: 95.49% (Test)

South-East: 96.75% (Test)

South-West: 94.74% (Test)

Comparison:

Slight improvement over default Gradient Boosting.

Performs comparably to XGBoost.

		North-East	North-West	South-East	South-West
Linear Regression	Train Accuracy	98.12%			
	Test Accuracy	98.11%	95.58%	96.83%	94.83%
Decision Tree	Train Accuracy	99.99%			
	Test Accuracy	95.39%	92.95%	94.19%	92.16%
Decision Tree with Hyperparameter Tuning	Train Accuray	98.26%			
	Test Accuracy	96.99%	94.53%	95.77%	94.51%
Random Forest	Train Accuracy	99.63%			
	Test Accuracy	97.41%	94.96%	96.20%	94.21%
Random Forest with Hyperparameter Tuning	Train Accuracy	99.32%			
	Test Accuracy	97.63%	95.11%	96.37%	94.35%
XGBoost Model	Train Accuracy	98.10%			
	Test Accuracy	98.03%	95.49%	96.75%	94.74%
XGBoost with Hyperparameter Tuning	Train Accuracy	98.08%			
	Test Accuracy	98.06%	95.52%	96.77%	94.77%

Gradient Boosting Regression	Train Accuracy	97.83%			
	Test Accuracy	97.81%	95.26%	96.53%	94.51%
Gradient Boosting withHyperparameter Tuning	Train Accuracy	98.05%			
	Test Accuracy	98.03%	95.49%	96.75%	94.74%

- XGBoost (Tuned) and Gradient Boosting (Tuned) are the best models, with consistently high-test accuracy across all regions.
- Random Forest (Tuned) also performs well but is slightly less generalizable than boosting methods.
- Decision Trees have high training accuracy but suffer from overfitting, making them less reliable.
- Linear Regression is only effective if the data follows a linear trend.

Conclusion

This study successfully developed and evaluated machine learning models to predict healthcare insurance premiums using demographic and lifestyle factors. By leveraging techniques such as Linear Regression, Decision Trees, Random Forest, XGBoost, and Gradient Boosting, we identified that boosting-based models (XGBoost and Gradient Boosting with hyperparameter tuning) provided the best generalization across different US regions. The analysis revealed key insights: older individuals, smokers, and those with a history of heart disease tend to pay higher premiums, while white-collar professionals incur higher charges than unemployed individuals and students. Additionally, categorical variables such as smoking status and occupation significantly influence premium amounts. The study also highlighted the importance of feature engineering and multicollinearity analysis in improving model stability and interpretability.