Medical Insurance Cost Prediction Using Machine Learning

Pranjal Upadhyay, Abhishek Prasad, Satyam Singh

Introduction: Need of the Model?

The sudden financial crisis during the COVID-19 pandemic made us realize the importance of medical insurance globally. Calculating the exact price of insurance for an individual, as well as for family members, is crucial for better financial management. For insurance companies, offering insurance at a higher price leads to a drop in sales, while offering it at a lower price can lead to losses. Therefore, it is essential to offer insurance at optimized and customized prices for every individual according to their needs, ensuring better profitability for the company. We are thus trying to analyze the factors affecting the price of insurance and their correlation. After analyzing the factors and their correlation we will proceed to train a machine learning model to predict the insurance premium.

Problem Formulation: Why is this a Regression Problem?

In classification problems, the target variable is categorical, meaning that the goal is to assign an instance to one of several predefined classes (e.g., "smoker" or "non-smoker"). However, in this case, the target variable "charges" is continuous, representing the cost of medical insurance, which can take on a wide range of values. Our aim is to predict this continuous value based on the independent variables, making this a regression problem.

Dataset Description

Statistic	Age	BMI	Children	Charges
Count	1338	1338	1338	1338
Mean	39.21	30.66	1.09	13270.42
Std	14.05	6.1	1.21	12110.01
Min	18	15.96	0	1121.87
25%	27	26.3	0	4740.29
50%	39	30.4	1	9382.03
75%	51	34.69	2	16639.91
Max	64	53.13	5	63770.43

The dataset consists of 1338 instances and 7 variables, including:

Independent Variables:

- **Age:** The age of the insured individual.
- **BMI:** Body mass index of the individual.
- **Children:** Number of children covered by health insurance.
- **Sex:** Gender of the individual (categorical).
- **Smoker:** Smoking status of the individual (categorical).
- **Region:** The region where the individual resides (categorical).

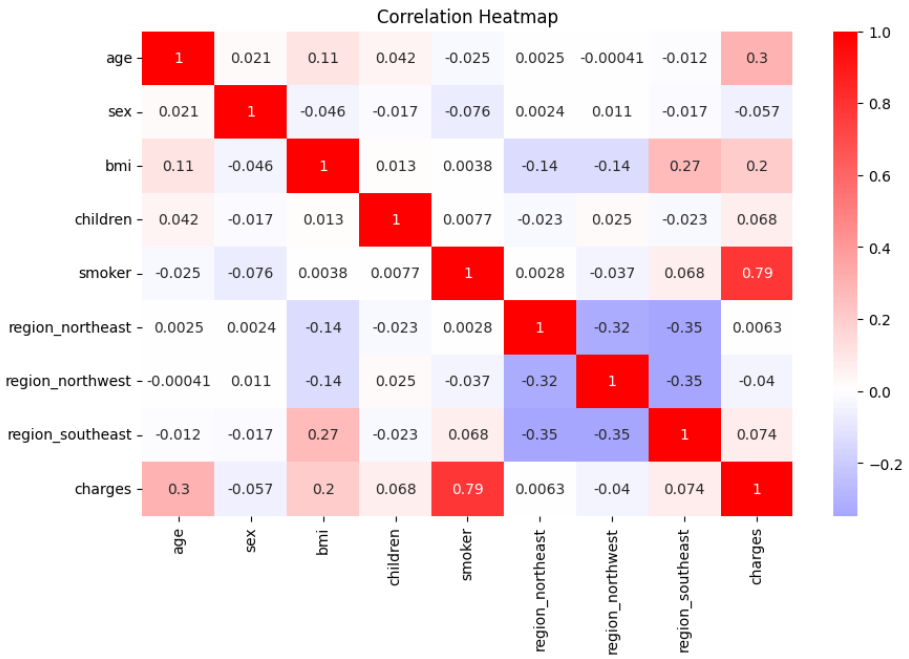
Dependent Variable:

- **Charges:** Medical insurance charges (in dollars).

Data Characteristics:

Column	Description	Non-Null Count	Data Type
age	Age of the individual	1338	Integer
sex	Gender of the individual	1338	Categorical
bmi	Body Mass Index (BMI)	1338	Float
children	Number of dependents	1338	Integer
smoker	Smoking status (Yes/No)	1338	Categorical
region	Residential region	1338	Categorical
charges	Medical charges incurred	1338	Float

- **Balanced Dataset:** All independent variables have the same number of instances, and there are no null values.
- **Categorical Variables:** "Sex," "Smoker," and "Region" are categorical and need to be encoded before model training.



From the above correlation heatmap we can see that smoker is showing highest correlation with the charges

Preprocessing Pipeline

In order to prepare the dataset for modeling, the following preprocessing steps are implemented:

1. Normalization

Min-Max normalization is applied to ensure that all numeric features are scaled between 0 and 1, as features such as age, BMI, and number of children have different ranges. This normalization improves runtime and ensures no value exceeds the maximum integer supported by the system.

2. Encoding Categorical Variables

Suppose the `region` column has four categories: `southwest`, `southeast`, `northwest`, and `northeast`. These categories are nominal, meaning there is no inherent order. If arbitrary numbers are assigned as follows:

```
southwest = 0
southeast = 1
northwest = 2
northeast = 3
```

this could suggest a false relationship, implying that `northeast` (3) is "greater than" `southeast` (1), which is not meaningful.

One-Hot Encoding: The categorical variables `Sex`, `Smoker`, and `Region` are transformed using one-hot encoding. This ensures that the data is suitable for machine learning algorithms. After encoding:

- If `southeast = 0`, `northwest = 0`, and `northeast = 0`, then the region must be `southwest`.
- If `southeast = 1`, it indicates the region is `southeast`.
- If `northwest = 1`, it indicates the region is `northwest`.
- If `northeast = 1`, it indicates the region is `northeast`.

southeast	northwest	northeast
0	0	0
1	0	0
0	1	0
0	0	1

	Variable	Pearson Coefficient	P-value
0	age	0.299008	4.886693e-29
1	bmi	0.198341	2.459086e-13
2	children	0.067998	1.285213e-02
3	sex	-0.057292	3.613272e-02
4	smoker	0.787251	8.271436e-283
5	region_northeast	0.006349	8.165264e-01
6	region_northwest	-0.039905	1.445970e-01
7	region_southeast	0.073982	6.782699e-03

In the above data, we are going to use two coefficients:

- **Pearson Coefficient:** It tells the strength and direction of linear relationships between two variables.
- **P-value:** It determines whether the observed correlation is statistically significant. We take values below 0.05 as significant.

Modeling Approach

We trained multiple machine learning models to predict insurance charges. The models include:

- **Linear Regression:** A statistical method used to model the relationship between a dependent variable and one or more independent variables by fitting a linear equation to observed data. It assumes a straight-line relationship between the inputs and the target variable.
- **Polynomial Regression:** An extension of linear regression that models the relationship between the dependent variable and the independent variable(s) as an n-degree polynomial. It allows the model to capture more complex, non-linear relationships.
- **Ridge Regression:** A regularized version of linear regression that adds a penalty term (L2 regularization) to the loss function to prevent overfitting. It shrinks the coefficients to zero but does not eliminate them, making it useful for multicollinearity.
- **Lasso Regression:** Another form of regularized linear regression, Lasso (Least Absolute Shrinkage and Selection Operator) adds an L1 regularization term to the loss function. Lasso can shrink some coefficients to zero, effectively performing feature selection.
- **Random Forest:** An ensemble learning method that combines multiple decision trees, each trained on a random subset of data. It aggregates the predictions from all the trees, improving accuracy and reducing overfitting compared to individual decision trees.

- **Decision Tree:** A non-linear model that splits data into subsets based on feature values, forming a tree-like structure. Each internal node represents a decision rule, and the leaf nodes represent the predicted outcome. It can overfit if not properly pruned.
- **AdaBoost:** Adaptive Boosting (AdaBoost) is an ensemble technique that combines multiple weak learners, typically decision trees, to create a strong learner. It adjusts the weights of incorrectly classified instances so that the model focuses on harder examples.
- **Gradient Boosting:** An ensemble learning method that builds decision trees sequentially. Each new tree tries to correct the errors of the previous trees by minimizing the residual errors through gradient descent. It's effective but can be slow to train.
- **XGBoost:** Extreme Gradient Boosting (XGBoost) is an optimized version of gradient boosting. It uses a more efficient implementation of gradient boosting and incorporates regularization to prevent overfitting, making it one of the most powerful algorithms for structured/tabular data.

Model Evaluation Metrics

The models evaluation will be based on the following metrics:

- **Mean Squared Error (MSE):**

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2$$

Where:

$\hat{y}_i$ = Predicted value for the i^{th} data point

y_i = Actual value for the i^{th} data point

n = number of observations

- **Mean Absolute Error (MAE):**

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|\hat{y}_i - y_i|$$

Where:

$\hat{y}_i$ = Predicted value for the i^{th} data point

y_i= Actual value for the i^{th} data point

n = number of observations

- **R^2 Score:**

$$R^2 = 1 - \frac{SS_{\text{res}}}{SS_{\text{tot}}}$$

$$SS_{\text{res}} = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

$$SS_{\text{tot}} = \sum_{i=1}^{n}(y_i - \bar{y})^2$$

Where:

SS_{res}= The sum of squares of the residual errors, representing the variation that the model fails to explain

SS_{tot} = The total sum of squares, representing the total variation in the data

$\hat{y}_i$ = Predicted value for the i^{th} data point

y_i= Actual value for the i^{th} data point

$\bar{y}$ = Mean value of the dependent variable

n =Number of observations

Hyperparameter Tuning

To optimize the performance of our models, we used GridSearchCV to perform an exhaustive search over specified hyperparameter values for each model. This helped in finding the best model configurations.

Feature Selection and Correlation Analysis

Once baseline models were trained, we performed correlation analysis to detect multicollinearity or redundant features. Highly correlated features might not provide additional information to the model, and removing them can help improve model interpretability and performance.

Results

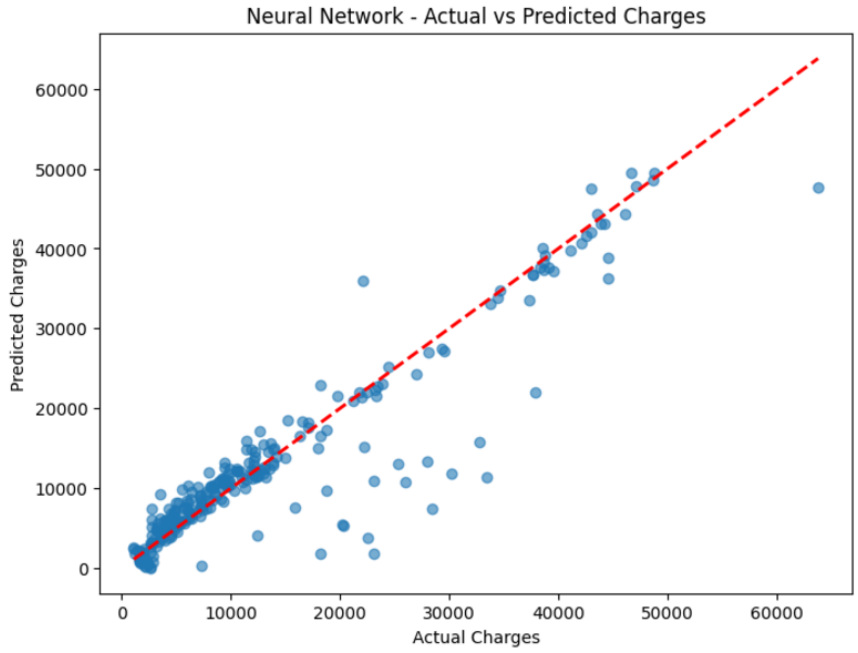
Here we used all the variables:

- **Before Optimization:**

Algorithm	MAE	MSE	R^2	MPE (%)
Linear Regression	4181.19	3.35969e+07	0.783593	46.8883
Polynomial Regression	2743.4	2.07246e+07	0.866507	30.5028
Ridge Regression	4182.74	3.36046e+07	0.783543	46.9274
Lasso Regression	4181.2	3.35993e+07	0.783578	46.8919
Random Forest	2493.08	2.15049e+07	0.861481	29.0193
Decision Tree	3055.37	4.19117e+07	0.730035	39.6185
AdaBoost	4192.6	2.57379e+07	0.834215	73.8829
Gradient Boosting	2381.91	1.86325e+07	0.879983	27.8649
XGBoost	2594.75	2.21705e+07	0.857193	32.4506

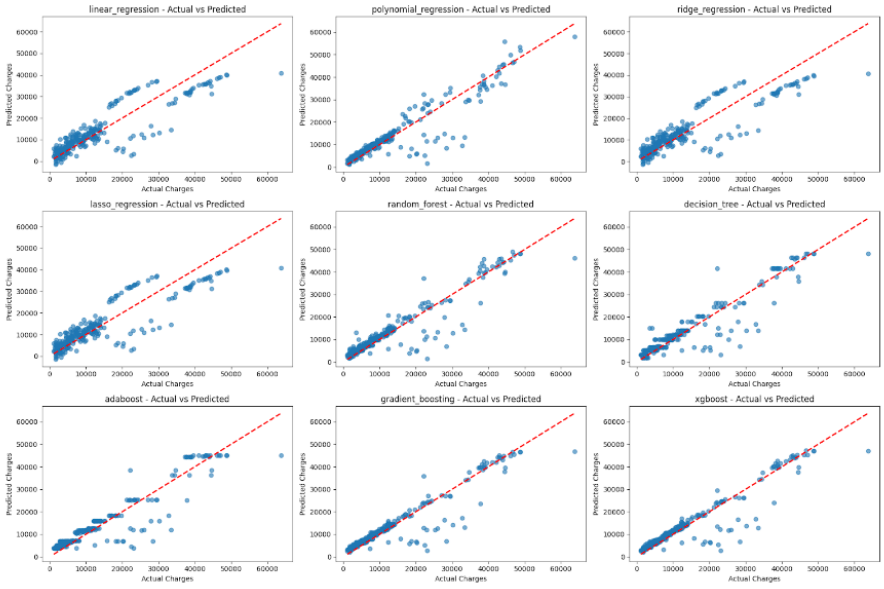
- **After Optimization:**

Algorithm	MAE	MSE	R^2	MPE (%)
linear_regression	4181.19	3.35969e+07	0.783593	46.8883
polynomial_regression	2743.4	2.07246e+07	0.866507	30.5028
ridge_regression	4197.12	3.36823e+07	0.783043	47.2946
lasso_regression	4181.52	3.36206e+07	0.78344	46.9303
random_forest	2495.43	1.99684e+07	0.871378	29.6439
decision_tree	2652.03	2.04428e+07	0.868322	33.6511
adaboost	3088.07	2.21614e+07	0.857252	42.609
gradient_boosting	2456.48	1.86802e+07	0.879676	32.2482
xgboost	2436.38	1.83533e+07	0.881781	31.1883



Deep Learning Model Result

With all variables:




```
Neural Network Metrics:
Mean Absolute Error (MAE): 2380.4830813634194
Mean Squared Error (MSE): 20912049.2347667
R² Score: 0.8652997094802362
Mean Percentage Error (MPE): 25.81043825851873%
```

Conclusion

This study demonstrates the application of various machine learning models to predict medical insurance costs. By using regression models and employing techniques like regularization, we can reduce overfitting and improve the generalization of the model. Through careful preprocessing, feature selection, and hyperparameter tuning, we have developed an accurate predictive model that can estimate insurance charges based on the given data.

Ensemble algorithms performed better than traditional regression methods and after evaluating machine learning regression models, we found that Random Forest gave the best result, with the mean percentage error being 29.6%. But the deep learning model outperformed all the machine learning algorithms as it gave 25.8% MPE. We choose MPE evaluation metric so that whenever our model predicts Insurance price we can tell the range of insurance price as: