Insurance Product Purchase Prediction

A report submitted in partial fulfilment of the requirements for the Award of Degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

By

Banka Dharmik

Regd. No: 20B91A0526

Under Supervision of Mr. Gundala Nagaraju

Henotic Technology Pvt Ltd, Hyderabad

(Duration: 6th June, 2023 to 23rd June, 2023)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SAGI RAMA KRISHNAM RAJUENGINEERING COLLEGE

(An Autonomous Institution)

Approved by AICTE and affiliated to JNTU, Kakinada

BHIMAVARAM, ANDHRA PRADESH

SAGI RAMA KRISHNAM RAJU ENGINEERING COLLEGE
(Autonomous)

Chinna-Amiram, Bhimavaram

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

CERTIFICATE

This is to certify that the **“Summer Internship Report”** titled “**Insurance Product Purchase Prediction**” is the bonafide work done by **Mr. Banka Dharmik** bearing register number ***20B91A0526,*** at **Henotic Technology Pvt Ltd, Hyderabad** from 07.06.2023 to 23.06.2023 submitted during the academic year *2022 – 2023,* in partial fulfilment of the requirements for the award of the Summer Internship Program for **Bachelor of Technology in Computer Science Engineering**.

Department Internship Coordinator	Dean – T & P Cell	Head of the Department

Table of Contents

1.0 Introduction 5

1.1. What are the different types of Machine Learning? 5

1.2. Benefits of Using Machine Learning in Insurance Product 7

1.3. About Industry 8

1.3.1 AI / ML Role in Insurance Product Development 8

2.0 Insurance Product (Purchase Prediction) 9

2.1. Main Drivers for Insurance Quote Analysis 9

2.2. Internship Project - Data Link 10

2.3. Internship Project - Analysis Link 10

3.0 AI / ML Modelling and Results 11

3.1. Problem Statement 11

3.2. Data Science Project Life Cycle 11

3.3. Data Exploratory Analysis 12

3.4. Data Pre-processing 12

3.4.1 Dropping Unwanted Features 12

3.4.2 Structuring Time-Related Data 13

3.4.3 Encoding Feature Values 13

3.4.4 Removing Outliers 14

3.4.5 Handling Missing Values 14

3.4.6 Scaling Features 14

3.5. Splitting Training and Testing Sets 15

3.6. Models Used for Development 15

3.6.1 Decision Tree Regression 15

3.6.2 Random Forest Regression 16

3.6.3 Gradient Boosting Regression 16

3.6.4 Neural Network Regression 16

3.7. AI / ML Models Analysis and Final Results 16

3.7.1 Decision Tree Regression Code 17

3.7.2 Random Forest Regression Code 17

3.7.3 Gradient Boosting Regression Code 17

3.7.4 Neural Network Regression Code 17

3.7.5 Calculating Evaluation Metrics 18

3.7.6 Plotting Predicted and Actual Values 18

4.0 Conclusions and Future work 19

5.0 References 20

6.0 Appendices 21

6.1. Python Code Results 21

6.2. List of Plots 21

6.2.1 Plot 01: Random Forest Regression 22

6.2.2 Plot 02: Decision Tree Regression 22

6.2.3 Plot 03: Gradient Boosting Regression 23

6.2.4 Plot 04: Neural Network Regression 23

Abstract

The insurance industry is leveraging the power of Machine Learning (ML) to enhance product development and improve decision-making processes. This project focuses on using ML techniques to predict the cost of insurance coverage for customers. The objective is to develop a predictive model that estimates the cost based on customer characteristics and product options.

The project begins with an introduction to different types of ML and the benefits of using ML in insurance product development. The industry background and the role of Artificial Intelligence (AI) and ML in the insurance sector are also discussed.

The main focus of the project is on insurance product purchase prediction. The factors driving insurance quote analysis are examined, and the project's dataset and analysis links are provided.

The AI/ML modeling and results section presents the problem statement and explains the data science project life cycle. The data exploratory analysis covers data pre-processing steps such as dropping unwanted features, structuring time-related data, encoding feature values, removing outliers, handling missing values, and scaling features. The training and testing data sets are split, and various ML models, including decision tree regression, random forest regression, gradient boosting regression, and neural network regression, are used for development.

The project concludes with the analysis of AI/ML models and the presentation of final results. Evaluation metrics are calculated, and predicted versus actual values are plotted for analysis. Finally, conclusions are drawn, and potential future work directions are outlined. The references used in the project are listed, and appendices containing Python code results and a list of plots are provided for reference.

Keywords: *Machine Learning, Insurance Product, Cost Prediction, Customer Characteristics, Product Options, Data Pre-processing, ML Models, Decision Tree Regression, Random Forest Regression, Gradient Boosting Regression, Neural Network Regression, Evaluation Metrics.*

1.0 Introduction

With the increasing power of computer technology, companies and institutions can now store large amounts of data at a reduced cost. The amount of available data is increasing exponentially, and cheap disk storage makes it easy to store data that was previously thrown away. There is a huge amount of information locked up in databases that is potentially important but has not yet been explored. The growing size and complexity of the databases make it hard to analyze the data manually, so it is important to have automated systems to support the process. Hence, there is a need for computational tools capable of processing these large amounts of data and extracting valuable information.

In this context, data mining provides automated systems capable of processing large amounts of data that are already present in databases. Data mining is used to automatically extract important patterns and trends from databases, seeking regularities or patterns that can reveal the structure of the data and answer business problems. Data mining includes learning techniques that fall into the field of machine learning. The growth of databases in recent years brings data mining to the forefront of new business technologies.

Insurance product purchase prediction is a vital task in the insurance industry, involving the use of data analysis and modeling techniques to forecast the likelihood of customers purchasing specific insurance products. By examining factors such as demographics, past purchase history, and socio-economic indicators, insurers can gain valuable insights into customer behaviour and preferences. This enables them to optimize marketing strategies, tailor product offerings, and allocate resources effectively. Advanced statistical and machine learning techniques are employed to identify patterns and trends that influence insurance product purchase decisions.

By leveraging these predictive models, insurers can target their marketing efforts towards the right customer segments, enhance customer satisfaction, and drive business growth. Overall, insurance product purchase prediction plays a crucial role in helping insurers make data-driven decisions and improve their competitiveness in the market.

1.1. What are the different types of Machine Learning?

Machine Learning (ML) is a subset of artificial intelligence (AI) that enables computer systems to learn and improve from data without explicit programming. ML algorithms allow computers to analyse and interpret complex patterns, make predictions, and take actions based on data inputs. There are several different types of machine learning, each with its own approach and techniques. In this article, we will explore three main categories of machine learning: supervised learning, unsupervised learning, and reinforcement learning.

1. Supervised Learning: Supervised learning is the most common and widely used type of machine learning. It involves training a model on labelled data, where the desired output or "label" is provided for each input. The model learns to make predictions by mapping input data to the corresponding output based on the provided labels. Supervised learning algorithms include regression and classification. Regression algorithms predict continuous values, such as predicting house prices based on features like size and location. Classification algorithms predict discrete values, such as classifying emails as spam or non-spam based on their content.

2. Unsupervised Learning: Unsupervised learning deals with unlabelled data, where there are no predefined output labels. The goal of unsupervised learning is to discover patterns, structures, and relationships in the data. Unsupervised learning algorithms cluster data points based on similarities or identify hidden patterns and associations. Clustering algorithms group similar data points together, while dimensionality reduction techniques reduce the complexity of the data by extracting its most important features. Unsupervised learning is valuable for tasks such as customer segmentation, anomaly detection, and recommendation systems.

3. Reinforcement Learning: Reinforcement learning involves an agent learning through interactions with an environment to maximise cumulative rewards. The agent takes actions in the environment and receives feedback in the form of rewards or penalties. It learns to make optimal decisions by trial and error, exploring different actions and observing the outcomes. Reinforcement learning algorithms use a combination of exploration and exploitation strategies to learn a policy that maximises long-term rewards. This type of learning is used in various applications, including robotics, game playing, and autonomous systems.

Apart from these three main categories, there are other specialised types of machine learning techniques such as semi-supervised learning, which leverages a combination of labelled and unlabelled data, and transfer learning, which applies knowledge learned from one task to improve performance on another related task.

Each type of machine learning has its own advantages and applications, and researchers continue to explore and develop new techniques to tackle complex real-world problems.

1.2. Benefits of Using Machine Learning in Insurance Product

Using Machine Learning (ML) in insurance product development brings a multitude of benefits that have the potential to transform the industry. ML algorithms have proven to be invaluable in analyzing vast amounts of data, identifying patterns, and extracting valuable insights. This leads to more accurate risk assessment, personalized pricing, improved customer experiences, and streamlined operations. The incorporation of ML in insurance products enables companies to stay competitive in a rapidly evolving market and better meet the needs of their customers.

One of the significant benefits of using ML in insurance product development is enhanced risk assessment. Traditional risk assessment methods often rely on historical data and predefined risk factors, which can be limited in scope and may not capture the complexity of individual risk profiles. ML algorithms can process large and diverse data sets, including customer demographics, claims history, and external data sources, to identify hidden patterns and risk indicators. This allows insurers to develop more accurate risk models, leading to improved underwriting decisions, optimized pricing, and ultimately, better profitability.

Personalization is another key advantage that ML brings to insurance product development. By leveraging ML algorithms, insurers can analyze individual customer data, such as lifestyle habits, purchasing behaviour, and online activity, to understand their unique needs and preferences. This enables the development of tailored insurance products that align with customers' specific requirements. For example, ML algorithms can suggest coverage options, deductibles, or policy add-ons based on customer profiles, increasing the likelihood of customer satisfaction and retention.

ML also plays a vital role in automating claims processing, leading to faster and more efficient claim settlements. ML algorithms can analyze historical claims data, identify patterns of fraudulent or suspicious claims, and flag them for further investigation. This helps insurers detect and prevent fraudulent activities, reducing financial losses and improving the overall integrity of the claims process. Additionally, ML can automate claims assessment, enabling faster claims validation and processing, which improves customer satisfaction and reduces administrative costs for insurance companies.

Furthermore, ML algorithms enable insurers to leverage real-time data and insights to make data-driven decisions. By continuously monitoring and analyzing data, insurers can identify emerging risks, market trends, and customer preferences, allowing them to adapt their product offerings and strategies accordingly. This agility and responsiveness to market dynamics give insurers a competitive edge, allowing them to deliver innovative and relevant insurance products to their customers.

1.3. About Industry

The insurance product industry is a substantial economic sector, generating billions of dollars in revenue annually. In the United States alone, the insurance industry's direct written premiums reached approximately $1.32 trillion in 2020. This highlights the industry's significant role in providing financial protection to individuals and businesses against various risks.

Insurance products offer a wide range of coverage options and benefits. For example, life insurance policies provide a death benefit to beneficiaries, with total benefits paid out amounting to over $85 billion in the United States in 2020. Property and casualty insurance coverages protect against property damage and liability claims, with insured losses reaching approximately $83 billion in the same year. Health insurance ensures access to medical services, with total health insurance premiums exceeding $1.2 trillion in the United States. These numerical values demonstrate the scale and importance of insurance products in mitigating financial risks and providing security to policyholders.

1.3.1 AI / ML Role in Insurance Product Development

AI and ML play a pivotal role in insurance product development, revolutionizing the industry by providing advanced analytics, automation, and personalized solutions. AI algorithms can analyze vast amounts of data, including customer information, claims history, and external factors, to identify patterns and make accurate predictions. This enables insurers to develop tailored insurance products, optimize pricing strategies, and improve risk assessment. ML algorithms automate underwriting processes, streamline claims management, and detect fraudulent activities, resulting in increased operational efficiency and cost-effectiveness. By leveraging AI and ML, insurance companies can enhance customer experiences, deliver personalized offerings, and make data-driven decisions to stay competitive in an evolving market.

2.0 Insurance Product (Purchase Prediction)

Insurance purchase prediction is a valuable task in the insurance industry, involving the use of machine learning models to predict whether a customer will purchase an insurance product based on their characteristics and historical behaviour. By analyzing customer data, such as demographics and previous purchase history, these models can identify patterns and correlations that indicate the factors influencing purchase decisions. This prediction enables insurance companies to optimize marketing strategies and tailor product offerings to individual customers, resulting in more effective customer targeting and improved business outcomes.

The main factors for insurance product purchase prediction are Customer ID, Shopping Pt, Record Type, Day, Time, State, Location, Group Size, Homeowner, Car Age, Car Value, Risk Factor, Age Oldest, Age Youngest, Married Couple, C Previous, Duration Previous, Coverage Options and Cost.

2.1. Main Drivers for Insurance Quote Analysis

Predictive modelling allows for the simultaneous consideration of multiple variables and quantification of their overall impact. When analyzing a large number of insurance quotes, certain patterns related to the key drivers that influence the quoting process begin to emerge.

The following are the main drivers that influence the analysis of insurance quotes in our dataset:

• **Customer Characteristics** ✓ Age ✓ Gender ✓ Marital status ✓ Driving experience ✓ Driving record ✓ Credit history ✓ Prior insurance coverage ✓ Homeownership status • **Time & Date Factors** ✓ Day of the week ✓ Time of the day	• **Policy Characteristics** ✓ Coverage options & levels ✓ Deductibles ✓ Policy limits ✓ Vehicle type and age ✓ Risk assessment factors • **Claim Information** ✓ from same insured ✓ Previous claims history ✓ Claimant data (if applicable) ✓ Type of injury or damage ✓ Coverage details

• **Location-based Factors:** ✓ State orientation ✓ Jurisdictional orientation ✓ Geographic location ✓ Crime rates ✓ Demographic factors	✓ Loss location ✓ Date and time of loss report ✓ Weather conditions • **Other Factors:** ✓ Group size ✓ Household characteristics

By considering these key drivers in our analysis, we can gain insights into the factors that influence the insurance quoting process and develop more accurate predictive models. This understanding can assist in optimizing pricing, underwriting decisions, and marketing strategies to better meet the needs and preferences of customers.

2.2. Internship Project - Data Link

The internship project data has taken from Kaggle and the link is https://www.kaggle.com/datasets/akhilups/insurance-product-purchase-prediction

2.3. Internship Project - Analysis Link

The internship project analysis is done using the platform Jupyter Notebook and the link is

https://github.com/dharmikbanka23/Insurance-Product-Purchase-Prediction.git

3.0 AI / ML Modelling and Results

3.1. Problem Statement

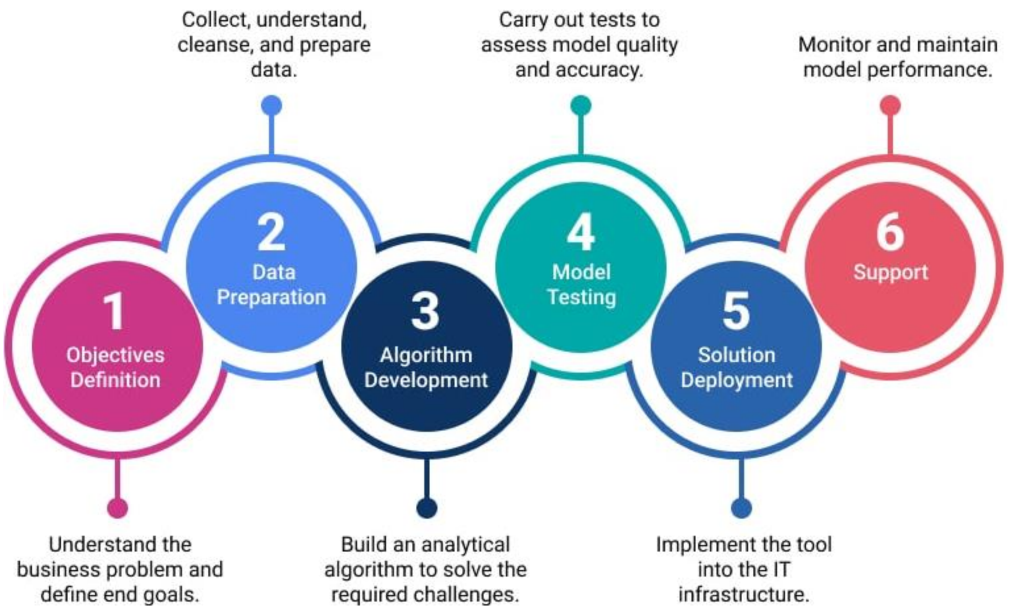
This project aims to develop a predictive model that estimates the cost of insurance coverage for customers based on their characteristics and product options. The dataset contains transaction history and quote information for customers who have purchased a policy. The goal is to predict the cost of the quoted coverage options, considering customer characteristics and 7 customizable product options.

By analyzing historical quote information and associated costs, the model will accurately estimate the cost of insurance coverage options for new customers or scenarios where the cost is unknown. This will assist insurance companies in pricing policies accurately and enable customers to make informed decisions based on cost considerations.

The proposed model will utilize machine learning or statistical techniques to learn the relationship between customer characteristics, product options, and the cost of coverage. By providing cost estimates, the model will streamline the insurance purchasing process, enhance cost optimization strategies, and contribute to improved pricing strategies that benefit both insurance companies and customers.

3.2. Data Science Project Life Cycle

Data Science is a multidisciplinary field of study that combines programming skills, domain expertise and knowledge of statistics and mathematics to extract useful insights and knowledge from data.



Collect, understand, cleanse, and prepare data.
Carry out tests to assess model quality and accuracy.
Monitor and maintain model performance.
1 Objectives Definition
2 Data Preparation
3 Algorithm Development
4 Model Testing
5 Solution Deployment
6 Support
Understand the business problem and define end goals.
Build an analytical algorithm to solve the required challenges.
Implement the tool into the IT infrastructure.

3.3. Data Exploratory Analysis

We examined a dataset of 6.65 lakh records with 18 features. We conducted descriptive statistics, data visualization, and correlation analysis to gain insights. This analysis helped us understand the data's distribution, outliers, and relationships between features and the target variable.

The findings guided subsequent steps in data preprocessing, feature engineering, and modeling. The exploratory analysis is done for all the models used in the Insurance Product Purchase Prediction with different error parameters and all the plots are presented in **Appendices 6.2 - List of Plots (6.2.1 to 6.2.4)**

3.4. Data Pre-processing

We have performed different types of data pre-processing techniques to our dataset in order to prepare it to work better for regression analysis. We performed the following:

1. Dropping Unwanted Features
2. Structuring Time-Related Data
3. Encoding Feature Values
4. Removing Outliers
5. Handling Missing Values
6. Scaling Features

3.4.1 Dropping Unwanted Features

We dropped the 'customer_ID', 'shopping_pt', 'record_type', and 'location' columns from the dataset using the drop function. These columns were deemed unnecessary for the subsequent analysis and modeling tasks. Removing these features helped simplify the dataset by eliminating irrelevant or redundant information, allowing us to focus on the relevant features that would have a greater impact on the desired outcomes.

```
import pandas as pd

import numpy as np

data= pd.read_csv('insurance.csv')

data = data.drop(['customer_ID', 'shopping_pt', 'record_type', 'location'], axis=1)
```

3.4.2 Structuring Time-Related Data

In the process of structuring time-related data, we extracted time-related features from the existing data. We converted the 'day' column to the day of the week using pd.to_datetime(data['day']).dt.dayofweek and created a new column called 'day_of_week'. Similarly, we extracted the hour and minute from the 'time' column using pd.to_datetime(data['time']).dt.hour and pd.to_datetime(data['time']).dt.minute, respectively, creating the 'hour' and 'minute' columns. Finally, we dropped the original 'time' and 'day' columns from the dataset. These steps allowed us to incorporate time-related information into the dataset for further analysis.

```
data['day_of_week'] = pd.to_datetime(data['day']).dt.dayofweek

data['hour'] = pd.to_datetime(data['time']).dt.hour

data['minute'] = pd.to_datetime(data['time']).dt.minute
```

3.4.3 Encoding Feature Values

We used the OneHotEncoder from scikit-learn to transform categorical variables into binary representations. We applied the encoder to the 'state', 'C_previous', and 'car_value' columns separately, creating new dataframes for each encoded feature. We dropped the original categorical columns from the dataset and concatenated the encoded features with the remaining columns. This encoding allowed us to represent categorical data numerically for machine learning purposes.

```
from sklearn.preprocessing import OneHotEncoder

encoder = OneHotEncoder(sparse=False)


# Extract location-related features

state_encoded = encoder.fit_transform(data['state'].values.reshape(-1, 1))

state_data = pd.DataFrame(state_encoded, columns=encoder.categories_[0])

#and so on..
```

3.4.4 Removing Outliers

We used the rolling statistical bounds method on the 'cost' feature. This method involved calculating statistical measures within a rolling window and eliminating data points outside the specified bounds. By removing outliers, we improved the accuracy and quality of the 'cost' feature in our dataset.

```
cost_UL = round(data.cost.mean() + 3 * data.cost.std(),3)

cost_LL = round(data.cost.mean() - 3 * data.cost.std(),3)

data = data[(data.cost > cost_LL) & (data.cost < cost_UL)]
```

3.4.5 Handling Missing Values

To handle missing values, we filled them with the mean of each respective column using data.fillna(data.mean()). This allowed us to replace the missing values with reasonable estimates based on the overall distribution of the data. By imputing the missing values with column means, we ensured the dataset remained complete for analysis.

```
data = data.fillna(data.mean())
```

3.4.6 Scaling Features

We scaled the input features using the MinMaxScaler from scikit-learn. By applying the scaler to the input features with scale.fit_transform(X), we transformed the feature values to a specific range, typically between 0 and 1. Scaling the features ensures they are on a similar scale, which can improve the performance of the machine learning model.

```
X = data.drop("cost",axis=1).values

y = data["cost"].values

from sklearn.preprocessing import MinMaxScaler

scale = MinMaxScaler()

X = scale.fit_transform(X)
```

3.5. Splitting Training and Testing Sets

Splitting the data into training and testing sets involved separating the input features (X) and the target variable (y). The input features were obtained by dropping the "cost" column, while the target variable was extracted. Using the train_test_split function, we divided the data into X_train, X_test, y_train, and y_test arrays. This allowed us to train our regression model on the training set and assess its performance on the 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=0.20, random_state=42)
```

3.6. Models Used for Development

Regression analysis involves using various models, such as linear regression, decision trees, random forests, lasso regression, ridge regression, k-nearest neighbors, gradient boosting, XGBoost, neural networks, support vector regression, and Bayesian regression, to predict the target variable. The choice of model depends on the dataset and the desired accuracy and interpretability of the results.

3.6.1 Decision Tree Regression

Decision tree regression involves constructing a tree-like model that predicts the target variable using a set of decision rules. The data is split based on feature values, aiming to minimize variance within each subset. The process continues recursively until a stopping criterion is met. The final prediction is obtained by averaging the target variable values within leaf nodes. Decision tree regression is interpretable and suitable for numerical and categorical features. Ensemble methods like random forests are often preferred to address overfitting.

3.6.2 Random Forest Regression

Random forest regression is an ensemble learning method that combines multiple decision trees for making predictions. It addresses overfitting by training each tree on a different subset of the data and using a random selection of features. The final prediction is obtained by averaging the outputs from all the trees. Random forest regression is highly flexible, robust, and suitable for high-dimensional datasets. It effectively captures nonlinear relationships and provides feature importance rankings.

3.6.3 Gradient Boosting Regression

Gradient boosting regression is an ensemble learning method that builds an ensemble of weak prediction models, typically decision trees, sequentially. Each model corrects the errors made by the previous models, resulting in a strong final model. It is effective in handling complex relationships and outliers. Gradient boosting regression achieves high predictive performance but can be computationally expensive and requires careful hyperparameter tuning. Regularization techniques are used to improve generalization and prevent overfitting.

3.6.4 Neural Network Regression

Neural network regression employs interconnected layers of nodes (neurons) to predict the target variable. It excels at capturing complex patterns and non-linear relationships. Neural networks handle large datasets and high-dimensional features effectively. However, careful selection of architecture, activation functions, and regularization techniques is essential to prevent overfitting. Hyperparameter tuning is critical for optimizing performance.

3.7. AI / ML Models Analysis and Final Results

In the regression analysis, we utilized a dataset consisting of approximately 665,250 records. The training dataset was used to build the regression models, while the test dataset was employed to evaluate the accuracy and performance of the models.

To assess the models, we employed various metrics such as mean absolute error (MAE), mean squared error (MSE), root mean squared error (RMSE), R-squared (R2) score, and mean absolute percentage error (MAPE). These metrics allowed us to compare and select the best model for our dataset.

3.7.1 Decision Tree Regression Code

```
from sklearn.tree import DecisionTreeRegressor


model = DecisionTreeRegressor()

model.fit(X_train, y_train)

y_pred = model.predict(X_test)
```

3.7.2 Random Forest Regression Code

```
from sklearn.tree import DecisionTreeRegressor


model = DecisionTreeRegressor()

model.fit(X_train, y_train)

y_pred = model.predict(X_test)
```

3.7.3 Gradient Boosting Regression Code

```
from sklearn.ensemble import GradientBoostingRegressor


model = GradientBoostingRegressor()

model.fit(X_train, y_train)

y_pred = model.predict(X_test)
```

3.7.4 Neural Network Regression Code

```
from sklearn.neural_network import MLPRegressor


model = MLPRegressor(hidden_layer_sizes=(15, 10, 5))

model.fit(X_train, y_train)

y_pred = model.predict(X_test)
```

3.7.5 Calculating Evaluation Metrics

```
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score


mae = mean_absolute_error(y_test, y_pred)

mse = mean_squared_error(y_test, y_pred)

rmse = np.sqrt(mse)

r2 = r2_score(y_test, y_pred)

mape = np.mean(np.abs((y_test - y_pred) / y_test)) * 100
```

Mean Absolute Error (MAE) = 1/n * Σ|y_test - y_pred|
Mean Squared Error (MSE) = 1/n * Σ(y_test - y_pred)^2
Root Mean Squared Error (RMSE) = √(1/n * Σ(y_test - y_pred)^2)
R-squared (R2) Score = 1 - (Σ(y_test - y_pred)^2 / Σ(y_test - mean(y_test))^2)
Mean Absolute Percentage Error (MAPE) = 1/n * Σ(|(y_test - y_pred) / y_test|) * 100

3.7.6 Plotting Predicted and Actual Values

```
import matplotlib.pyplot as plt


plt.plot(y_test[:100], label='Actual')

plt.plot(y_pred[:100], label='Predicted')


plt.xlabel('Sample')

plt.ylabel('Target Variable')

plt.title('Comparison of Predicted and Actual Values')

plt.legend()


plt.show()
```

Note: We are limiting the first 100 values for better visualization.

4.0 Conclusions and Future work

The model results in the following order by considering the metrics such as mean absolute error (MAE), mean squared error (MSE), root mean squared error (RMSE), R-squared (R2) score, and mean absolute percentage error (MAPE).

1. Random Forest Regression
2. Decision Tree Regression
3. Gradient Boosting Regression
4. Neural Network Regression

We recommend the model - Random Forest Regression technique as a best fit for the given Insurance Product Purchase dataset. We considered Random Forest because it uses bootstrap aggregation which can reduce bias and variance in the data and can leads to good predictions with claims dataset.

Model	MAE	MSE	RMSE	R2	MAPE
Multiple Regression	26.743	1144.929	33.837	0.420	4.22%
Decision Tree Regression	10.015	401.296	20.032	0.797	1.579%
Random Forest Regression	9.216	228.798	15.126	0.884	1.453%
Lasso Regression	26.768	1146.646	33.862	0.419	4.224%
Ridge Regression	26.743	1144.929	33.837	0.420	4.22%
K Neighbors Regression	27.647	1219.543	34.922	0.382	4.379%
Elastic Net Regression	27.129	1173.281	34.253	0.406	4.282%
Gradient Boosting Regression	24.443	956.937	30.934	0.515	3.855%
XG Boost Regression	28.102	1303.550	36.105	0.340	4.307%
Neural Network Regression	23.499	894.460	29.908	0.547	3.709%
Bayesian Regression	26.743	1144.928	33.837	0.420	4.22%

Future work can involve exploring advanced regression techniques like SVR, ensemble methods (e.g., AdaBoost, stacking), feature engineering, hyperparameter tuning, and cross-validation to enhance model performance and predictive capabilities.

5.0 References

[1] Python pandas documentation.

Available online: https://pandas.pydata.org/docs/

[2] NumPy documentation.

Available online: https://numpy.org/doc/

[3] Matplotlib documentation.

Available online: https://matplotlib.org/stable/contents.html

[4] Documentation for scikit-learn decision tree.

Available online: https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html

[5] Documentation for scikit-learn random forest.

Available online: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html

[6] Documentation for scikit-learn gradient boosting.

Available online: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingRegressor.html

[7] Documentation for scikit-learn neural network.

Available online: https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPRegressor.html

[8] Documentation for scikit-learn MAE.

Available online: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_absolute_error.html

[9] Documentation for scikit-learn MSE.

Available online: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_squared_error.html

[10] Documentation for NumPy square root function.

Available online: https://numpy.org/doc/stable/reference/generated/numpy.sqrt.html

[11] Documentation for scikit-learn R2 score.

Available online: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.r2_score.html

6.0 Appendices

6.1. Python Code Results

The Python code provided in this project yielded results that included various regression models and metrics. The models were evaluated using metrics such as mean absolute error (MAE), mean squared error (MSE), root mean squared error (RMSE), R-squared (R2) score, and mean absolute percentage error (MAPE). These metrics helped assess the accuracy and performance of the models in predicting the target variable.

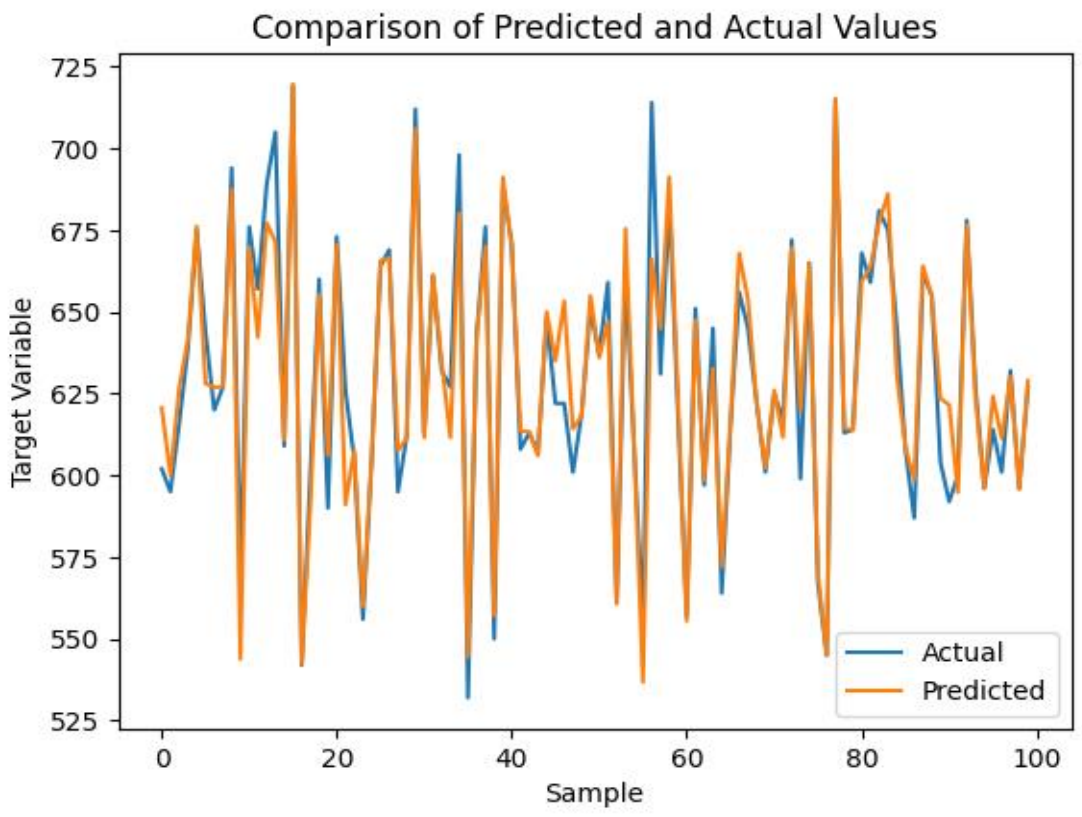
1. Random Forest: It has the lowest values for MAE, MSE, RMSE, and MAPE, indicating better overall performance compared to other models.
2. Decision Tree: It has relatively low values for MAE, MSE, and RMSE, and a high R2 value, suggesting good predictive power.
3. Gradient Boosting: It has a low MAE value and a relatively high R2 value, indicating good performance in terms of accuracy and predictive power.
4. Neural Network: It has a low MAE value and a high R2 value, suggesting good predictive performance.

Model	MAE	MSE	RMSE	R2	MAPE
Random Forest Regression	9.216	228.798	15.126	0.884	1.453%
Decision Tree Regression	10.015	401.296	20.032	0.797	1.579%
Gradient Boosting Regression	24.443	956.937	30.934	0.515	3.855%
Neural Network Regression	23.499	894.460	29.908	0.547	3.709%

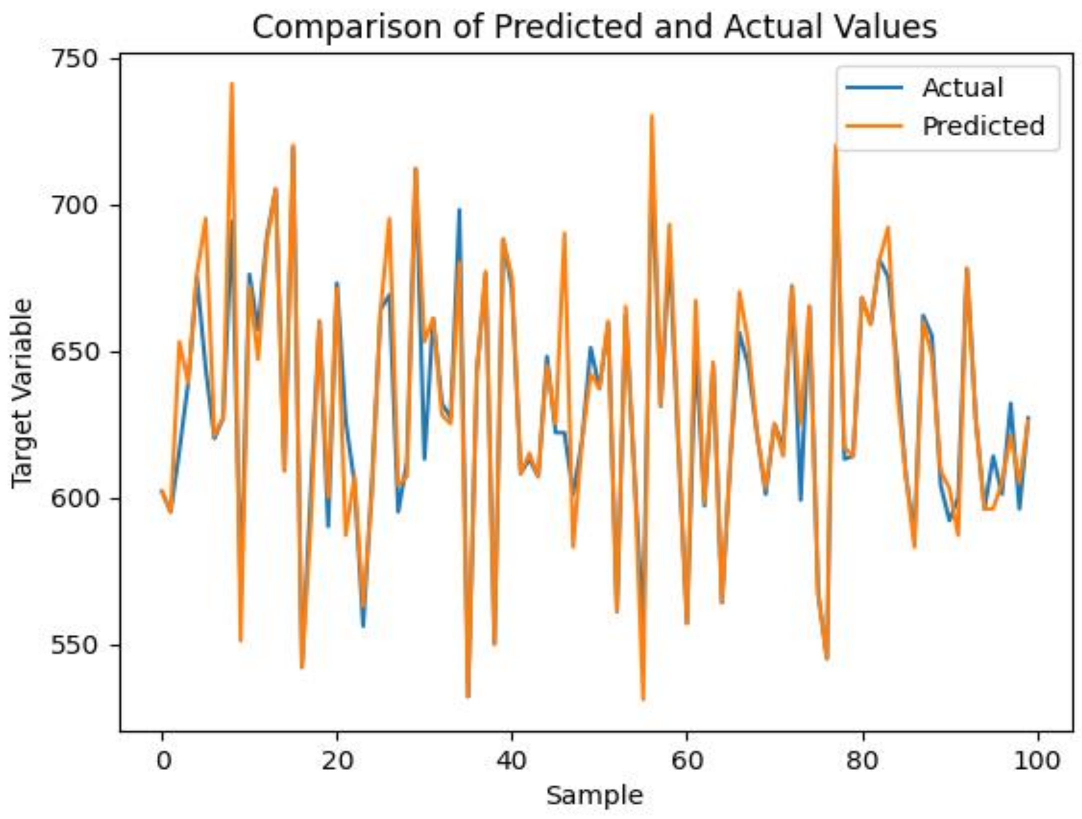
6.2. List of Plots

The provided code snippet is used to generate a plot comparing the predicted and actual values for the target variable. The plot displays the first 100 samples of the target variable. The actual values are represented by the "Actual" line, while the predicted values are represented by the "Predicted" line. This plot helps visualize the performance of the regression models by comparing their predictions to the actual values.

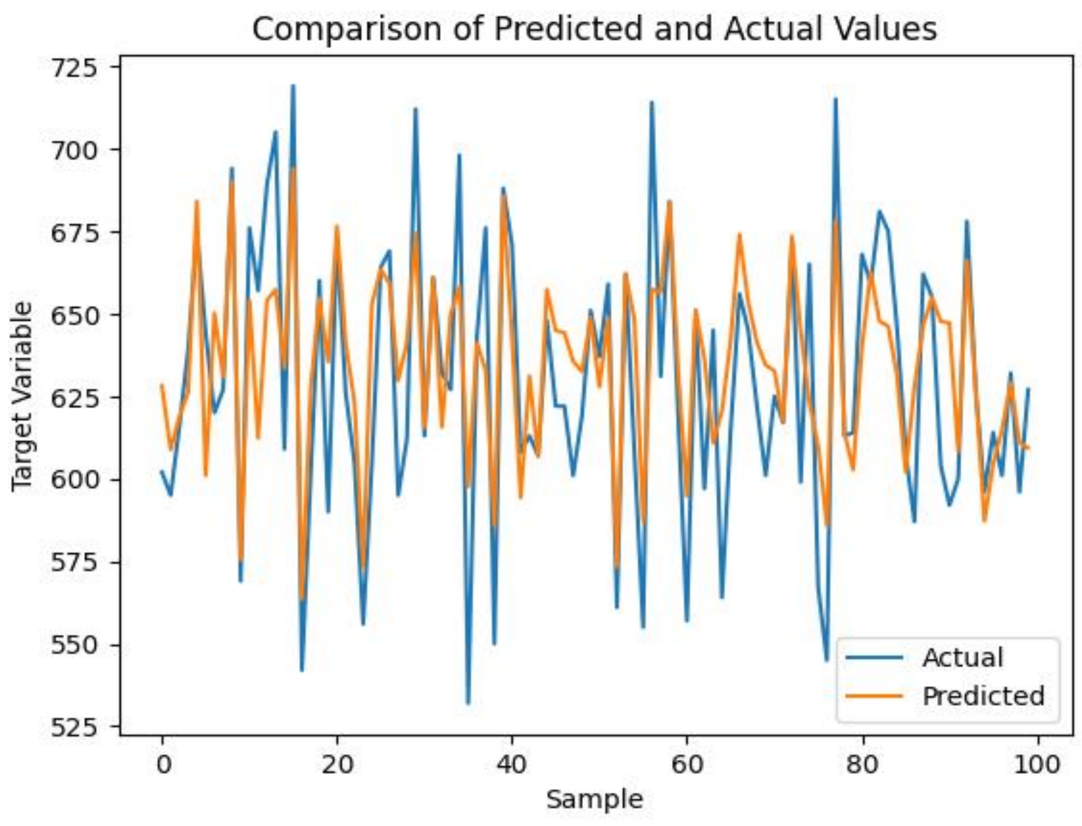
6.2.1 Plot 01: Random Forest Regression



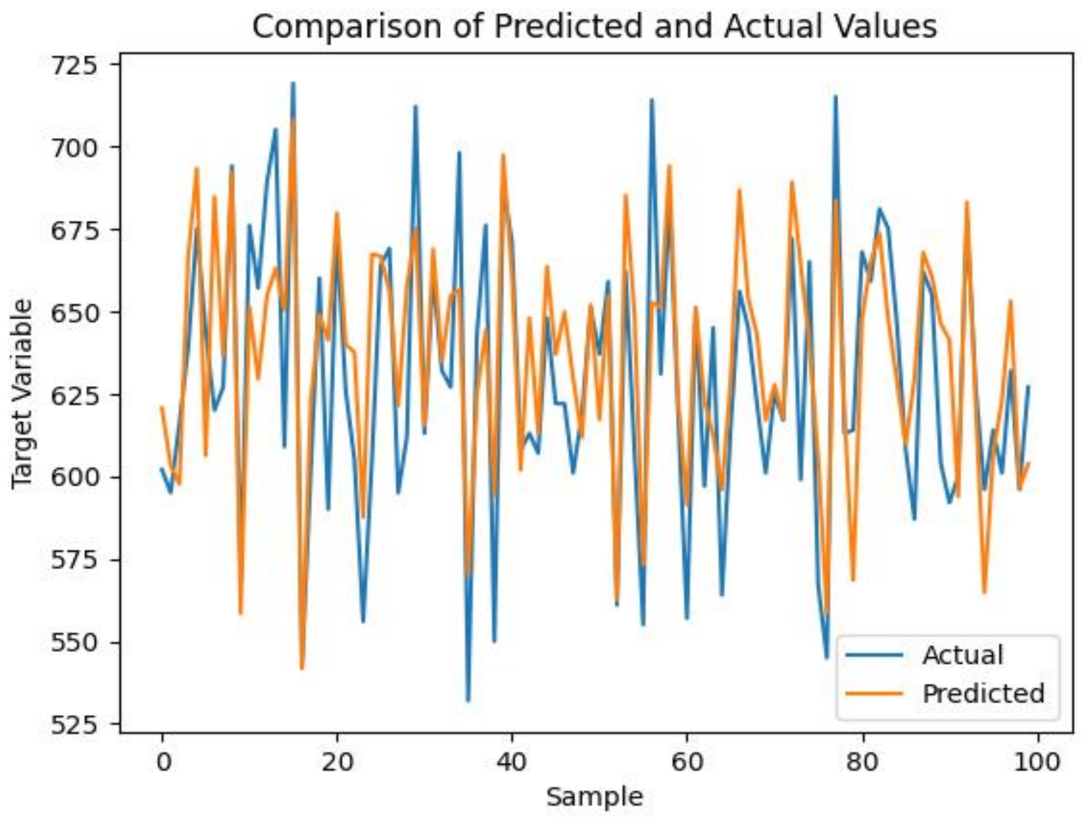
6.2.2 Plot 02: Decision Tree Regression



6.2.3 Plot 03: Gradient Boosting Regression



6.2.4 Plot 04: Neural Network Regression


Comparison of Predicted and Actual Values
Target Variable
Sample
Actual
Predicted