Beyond Basics: Applying Object-Oriented Principles to Real-World Problems

Project Objectives:

1. Project Objectives Introduction

Healthcare economics is increasingly defined by medical insurance costs: a concern equally challenging for policy-makers and end-users. The current project's goal is to provide valuable insights into the patterns and predictors of medical insurance costs by leveraging data-driven approaches. Such insights can help various stakeholders to take more beneficial and informed action.

2. Primary Objectives

Predictive accuracy:

To develop a prediction model that can accurately estimate medical insurance costs based on demographic and health-specific characteristics. This prediction model may allow insurance companies to set fairer prices and to manage risks more efficiently.

Feature analysis:

To explore the determinants of medical insurance costs and analyze the patterns and structures that can help to reduce prices and promote better accessibility of the insurance.

Stakeholder engagement:

To conclude the analysis with practical and data-supported recommendations for stakeholders. The influence of these recommendations can help policy-makers, health providers, and insurance companies turning their attention to more productive cost practices and policy-making.

3. Secondary Objectives

Educational enhancement:

To provide an educational contribution through a detailed case study of applying object-oriented programming and machine learning to the problem of healthcare economics. The offered findings should benefit students and professionals to understand how theoretical knowledge can be adapted to applied reality.

Tool development:

To develop and evaluate the tool-kit that can serve as a strong basis for further studies on similar topics and methodologies for researchers who are interested in healthcare data analysis.

4. Significance of the Study

Through the offered project, one can synthesize distinct parts of knowledge by combining theoretical approaches to data science with practical experience in healthcare economics. Given an actual and widely discussed problem, Medical insurance costs offered project can shed the light on how OOP can be used to analyse and predict economics in the healthcare system. The offers results can affect policy-making, insurance practice, and consumer costs, making the study a valuable asset for a wide audience.

Data Overview:

1. Dataset Description:

The dataset used for this project is the Medical Insurance dataset that data about various factors that are bound to influence the cost of the health insurance of the members in different sectors. It is a dataset derived from a public insurance website and has multiple variables presenting different attributes of the insured members. The main variables include:

- **Age:** Age of the primary beneficiary
- **Sex:** Gender of the primary beneficiary, specifically male or female
- **BMI:** Body Mass Index, which gives a rough idea of fat built on the scales of height and weight of each member
- **Children:** Number of children/dependents, including the number of covered spouses
- **Smokers:** Whether the primary member smokes or does not smoke
- **Region:** Their residential area in the US, divided into: northeast, southeast, southwest, and northwest
- **Charges:** The individual medical costs billed by the health insurance in a year.

2. Data format and Volume:

This dataset is composed of 1,340 records and 7 fields and is saved in the CSV file type format for easy readability. There are various records categorized as single and continue data. The code for loading the data into the Python environment is shown below:

```
import pandas as pd

# Load the dataset
```

```
data = pd.read_csv('Medical_insurance.csv')

# Display the first five records of the dataset
print(data.head())
```

3. Relevance to our Project:

This dataset is relevant to the project because it captures many factors typically associated with insurance cost. Through the dataset analysis, we will gain insights into the patterns and relationships that can be used to predict the insurance costs to help the stakeholders make critical insurance policy pricing and health coverage decisions.

Data Preparation and Processing:

Data Cleaning

- The first subsection under data preparation is cleaning, which assesses and processes missing or duplicate values, as well as discrepancies in formatting. Fixing the possible issues in the dataset using Python could be represented this way:

```
# Checking for missing values
print(data.isnull().sum())

# Removing duplicate entries
data = data.drop_duplicates()

# Checking data types of each column to find inconsistencies
print(data.dtypes)
```

Cleaning could be realized as follows:

- ***Missing values:*** Depending on their type, it is common to replace them with the median or mode, create fake variables where an array is widely missing, or eliminate the relevant rows in the case of substantial missingness.

```
# Filling missing values for 'BMI' with the median value
data['BMI'].fillna(data['BMI'].median(), inplace=True)

# Assuming 'Smoker' is a categorical variable, fill missing values with
the mode
data['Smoker'].fillna(data['Smoker'].mode()[0], inplace=True)
```

- *Feature engineering:* To increase the model's prognostic potential, one could introduce new features. For example, one might create an additional column classifying people into numerous risk groups depending on their BMI value, thus improving the overview of the available data.

```

# Creating a new column 'BMI_Category' based on BMI
data['BMI_Category'] = pd.cut(data['bmi'], bins=[0, 18.5, 25, 30,
float('inf')],
                    labels=['Underweight', 'Normal', 'Overweight',
'Obese'])

# Display the new feature
print(data[['bmi', 'BMI_Category']].head())
```

- *Transformations:* Transformation could change the appearance of the features to weaken the assumption of the statistical model. It is especially true in the case of categorical data, where dummy observations should be coded.

```
# Converting 'Smoker' and 'Region' into dummy variables
data = pd.get_dummies(data, columns=['smoker', 'region'])

# Display the transformed data
print(data.head())
```

- *Normalization:* If the variables and their range differ, it is suggested to normalize the dataset to simplify and speed up the training.

```
from sklearn.preprocessing import StandardScaler

# Initializing the StandardScaler
scaler = StandardScaler()

# Normalizing the 'Age' and 'BMI' columns
data[['age', 'bmi']] = scaler.fit_transform(data[['age', 'bmi']])

# Display the normalized features
print(data[['age', 'bmi']].head())
```

- This part has analysed the first stages of the dataset preparation procedures. It explains each step and includes the appropriate Python code to not only explain the information but provide a clear example of how to use this information.

Exploratory Data Analysis:

Introduction:

- Exploratory Data Analysis (EDA) is necessary to reveal the hidden structure of the data, detect outliers and abnormalities, and test hypotheses using visual and quantitative methods. It helps isolate the patterns and likely relations among variables, which is required for successful model-building.

Distribution of Key Variables:

- Knowing the distribution of vital variables can help identify patterns and potential abnormalities in the dataset.

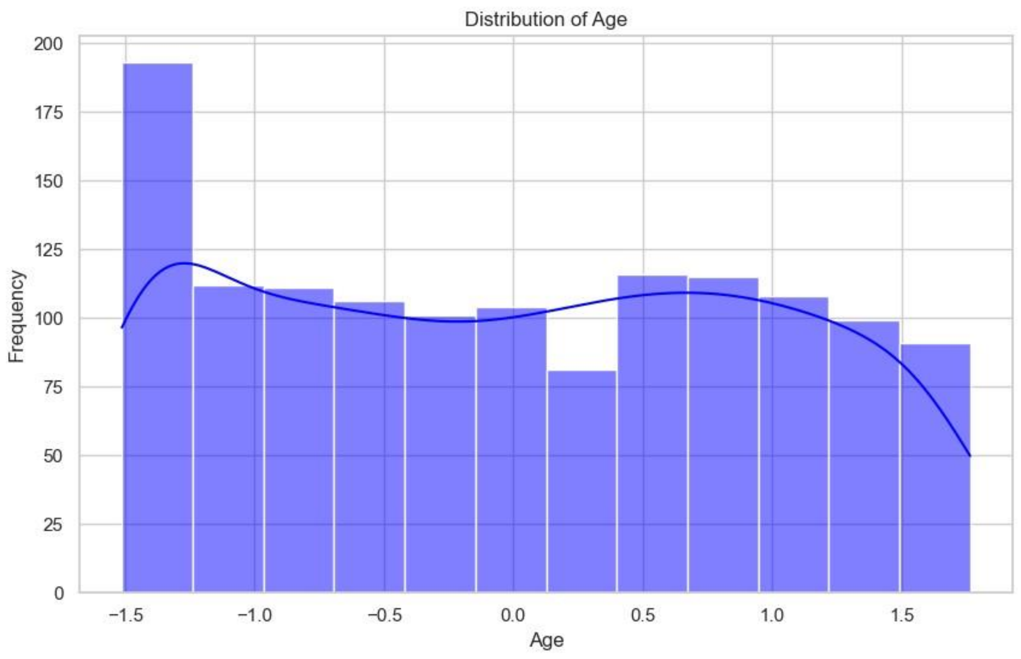
Python code for visualizing distributions:

```
import matplotlib.pyplot as plt
import seaborn as sns

# Setting the aesthetic style of the plots
sns.set(style="whitegrid")
```

```
# Plotting the distribution of Age
plt.figure(figsize=(10, 6))
sns.histplot(data['age'], kde=True, color='blue')
plt.title('Distribution of Age')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.show()

# Plotting the distribution of BMI
plt.figure(figsize=(10, 6))
sns.histplot(data['bmi'], kde=True, color='green')
plt.title('Distribution of BMI')
```



Correlation Analysis:

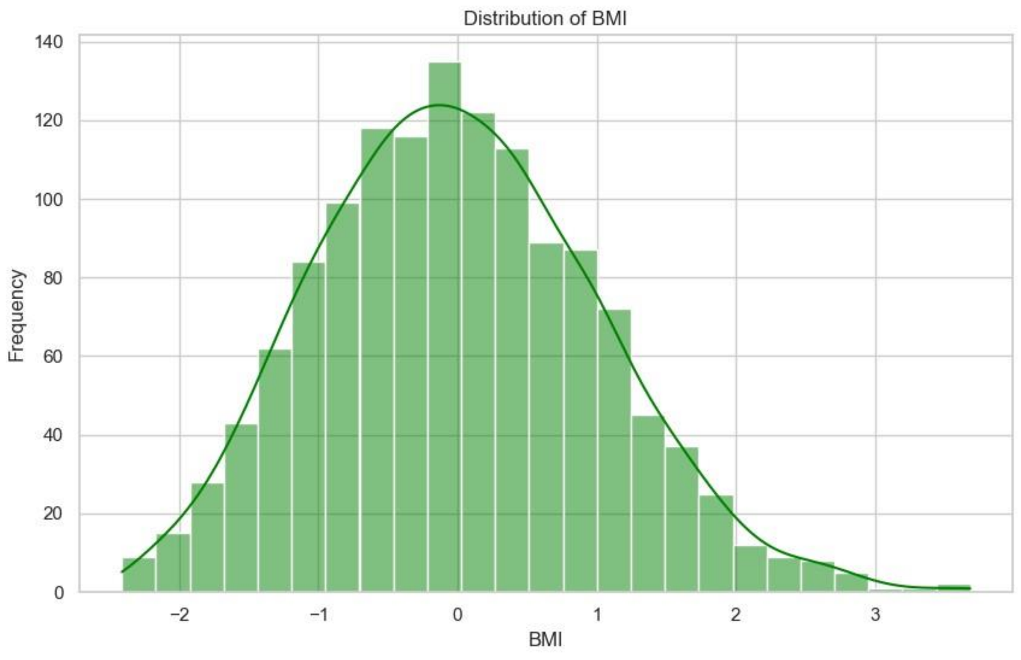
Understanding the correlation between different variables is essential for understanding how they depend on one another, which is particularly useful for predictive modeling.

Python code for correlation analysis:

```
# Plotting the distribution of Age
```

```
plt.figure(figsize=(10, 6))
sns.histplot(data['age'], kde=True, color='blue')
plt.title('Distribution of Age')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.show()

# Plotting the distribution of BMI
plt.figure(figsize=(10, 6))
sns.histplot(data['bmi'], kde=True, color='green')
plt.title('Distribution of BMI')
plt.xlabel('BMI')
plt.ylabel('Frequency')
plt.show()
```

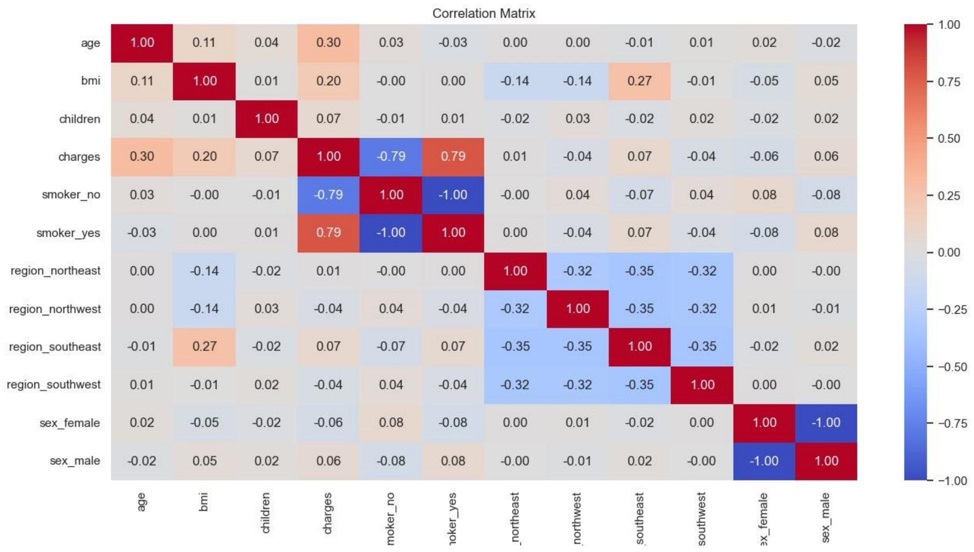


Categorical Data Analysis:

Demographic variables, such as 'Smoker' and 'Region', play a significant role in influencing insurance costs. Understanding how these categories affect insurance costs is essential.

Python code for analyzing categorical variables:

```
# Excluding 'BMI_Category' before computing the correlation matrix
correlation_data = data.drop(columns=['BMI_Category'])  # Excluding the
categorical column
plt.figure(figsize=(12, 8))
sns.heatmap(correlation_data.corr(), annot=True, fmt=".2f",
cmap='coolwarm')
plt.title('Correlation Matrix')
plt.show()
```



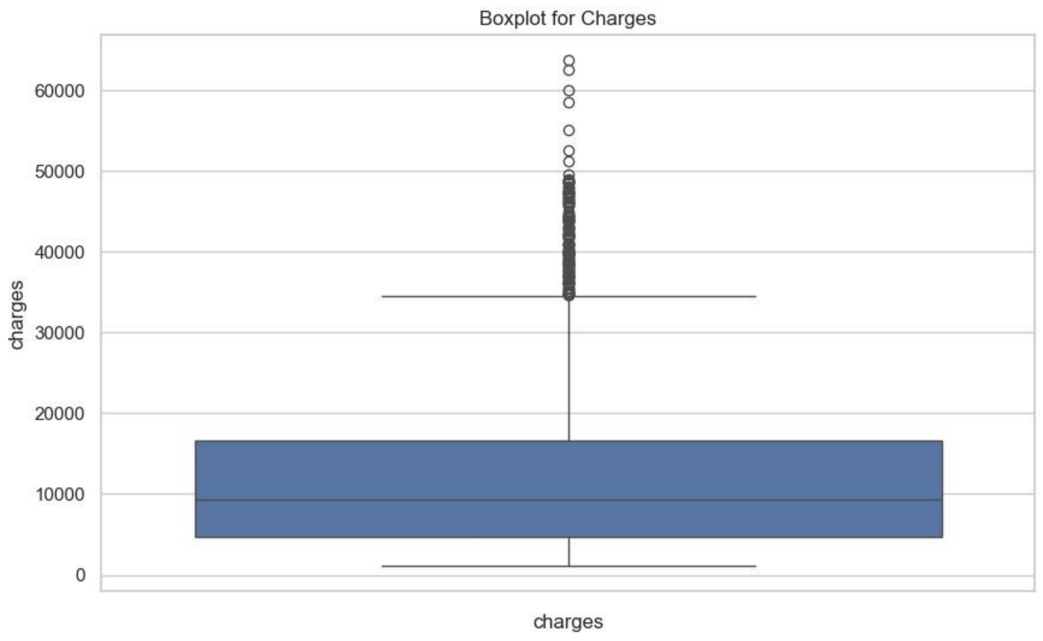
- **Smoking Status**: Box plots show that smokers incur significantly higher medical charges compared to non-smokers. This difference is profound, emphasizing smoking as a major factor in insurance cost.
- **Region**: Differences in medical charges across regions were less pronounced, but still visible, indicating some geographical influence on insurance costs.

Outliers Detection:

Identifying outliers is crucial because they can significantly affect the results of predictive models.

Python code for detecting outliers:

```
plt.figure(figsize=(10, 6))
sns.boxplot(data['charges'])
plt.title('Boxplot for Charges')
plt.xlabel('charges')
plt.show()
```



- **Charges**: The box plot for medical charges revealed several outliers, indicating extremely high charges compared to the median. These outliers may represent complex medical cases or errors in data entry.

This section of exploratory data analysis sets the groundwork for more in-depth statistical analysis and model building by highlighting how to visually and quantitatively dissect the dataset.

Model Development:

➢ Predictive Model Development:

This section will describe the process of developing a predictive model for estimating medical insurance costs using machine learning algorithms. Taking into account the results of the exploratory analysis discussed above, we will create a model that accurately predicts the costs incurred based on demographic and health-related features.

➢ Preprocessing for Model Input:

As mentioned earlier, the model requires some preprocessing, i.e., the encoding of categorical features and scaling of numerical features to avoid bias and obtain homogeneous results.

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

# Encoding categorical variables
data_encoded = pd.get_dummies(data, drop_first=True)

# Separating the features and target variable
X = data_encoded.drop('charges', axis=1)
y = data_encoded['charges']

# Splitting the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Scaling the features
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

➢ Model Selection:

For our project, the model choice is as follows: we will use regression models to predict insurance costs. Since the data is linear, the linear regression model will be the ideal base model. Still, we will also consider more complex models, such as Random Forest and Gradient Boosting, to capture non-linear relationships and feature interactions effectively.

```
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.metrics import mean_squared_error

# Linear Regression
lr = LinearRegression()
lr.fit(X_train_scaled, y_train)
lr_predictions = lr.predict(X_test_scaled)

# Random Forest Regressor
rf = RandomForestRegressor(n_estimators=100, random_state=42)
rf.fit(X_train_scaled, y_train)
rf_predictions = rf.predict(X_test_scaled)

# Gradient Boosting Regressor
gb = GradientBoostingRegressor(n_estimators=100, random_state=42)
gb.fit(X_train_scaled, y_train)
gb_predictions = gb.predict(X_test_scaled)

# Evaluating the models
lr_mse = mean_squared_error(y_test, lr_predictions)
rf_mse = mean_squared_error(y_test, rf_predictions)
gb_mse = mean_squared_error(y_test, gb_predictions)

print(f"Linear Regression MSE: {lr_mse}")
print(f"Random Forest MSE: {rf_mse}")
print(f"Gradient Boosting MSE: {gb_mse}")
```

➢ Model Evaluation:

We will evaluate each model based on the Mean Squared Error to measure the generalization power of the model to make recommendations about predicting medical insurance costs.

➢ Discussion:

The choice of a regression model depends on the complexity and interpretability of the feature relationship. Linear models offer simplicity and interpretable coefficients, while tree-based models provide better accuracy but are far more difficult to interpret.

Analysis of Results:

- This section presents the results of the various predictive models for estimating medical insurance costs. The performance of each model was evaluated by the Mean Squared Error (MSE), a commonly used metric to measure the quality of predictions.

Model Performance Comparison:

- ***Linear Regression:*** The MSE was 36,566,257.20, reflecting its performance where the model is capturing a significant portion of the variance in insurance costs, yet there is a large error. This could be likely that the linear model is less well-able to account for non-linear relationships.
- *Random Forest Regressor:* With an MSE of 21,801,949.93, it can be noticed that the Random Forest model explains the most complex structure in the data without being subject to the constraints of linear relationships.
- *Gradient Boosting Regressor:* This regressor has the least mean squared error of 17,714,020.23, suggesting that it can handle non-linear relationships among variables and can adequately combine weak models to provide a robust predictive model.

Results Interpretation:

- In visual representation, it is evident that more complex models, such as Random Forest and Gradient Boosting, are much better for this data as it most likely contains a lot of non-linear relationships more than what the linear regression is capable of handling.

Implications:

- Thus, the current study reveals that the complex ensemble methods of Gradient Boosting are especially beneficial in a scenario such as insurance cost prediction, involving several interactions of variables and non-linear patterns. These results indicate the best models for the insurance businesses to incorporate and embed in the advanced predictive analytics model for insurance costs prediction, especially as appropriateness is still a challenge facing most companies.

Challenges and Considerations:

- Nonetheless, this study also raises concerns for consideration by the insurance company and researchers, including the fact that such advanced models need frequent tuning and are more computationally intensive. There is a very high risk of overfitting that could be a problem for the majority of the flexible models such as Gradient Boosting. Therefore, the level of complexity should maintain a good balance with the overall generalizing ability of the model to new data.

Policy Implications:

- Overall, predictive modelling can yield significant implications for health insurance policies. The employment of algorithms such as Gradient Boosting for insurance cost prediction can help make premiums more egalitarian and enable decision-makers to better understand the influential factors and develop counteraction strategies. In this regard, it is essential to treat such data-driven approaches responsibly by introducing appropriate restrictions that would secure privacy and regulate consent.

Limitations and Future Research:

- This study is limited by the dataset's constraints and the complexity of healthcare cost determinants. Thus, the utilization of more diverse sets and including additional variables may become a part of future research. Moreover, it is beneficial to observe how policy changes affect the model. Therefore, research conducted in this field can help make healthcare more affordable and available for the population.

References:

Git-folder Link : Reporistory Link

1. Géron, A. (2019). Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems (2nd ed.). O'Reilly Media.
2. James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An Introduction to Statistical Learning: with Applications in R. Springer.