Medical Insurance Cost Prediction

1.Introduction

The aim of this project is to develop a machine learning model that could accurately predict medical insurance costs by taking into account individual characteristics. Accurate cost predictions are essential for insurance companies to manage risk and develop appropriate pricing strategies, and for individuals to better understand their financial obligations. To accomplish this, I worked with the Medical Cost Personal Dataset from Kaggle, which provided comprehensive information on factors such as age, sex, BMI, children, smoker status, region, and charges. By leveraging this dataset, I was able to identify patterns and relationships between these factors and insurance costs, enabling the creation of a model that could make precise predictions tailored to each person's unique situation. Through this project, I hope to assist insurance companies in devising more accurate and personalized pricing plans, as well as helping individuals make informed decisions regarding their insurance needs.

2.Data Exploration and Preprocessing

Upon loading the dataset and conducting an initial analysis, I observed that there are a total of 1,338 records. The data includes age, sex, BMI, number of children, smoking status, region, and insurance charges. The average age of individuals in the dataset is around 39 years, and the mean BMI is approximately 30.66, which is classified as 'overweight'. On average, individuals have around one child.

3. Visualize the data to identify patterns and relationships between variables

1.Load the Dataset

```
Loading the dataset

import pandas as pd

data = pd.read_csv('C:\insurance.csv')
```

2. Overview of the dataset

```
Display the first 5 records to get an overview of the data

print(data.head(), '\n')
```

```
   age     sex     bmi  children smoker     region      charges
0   19  female  27.900         0    yes  southwest  16884.92400
1   18    male  33.770         1     no  southeast   1725.55230
2   28    male  33.000         3     no  southeast   4449.46200
3   33    male  22.705         0     no  northwest  21984.47061
4   32    male  28.880         0     no  northwest   3866.85520
```

3. Summary Statistics of the dataset

```
Display summary statistics of the dataset

print(data.describe(), '\n')
```

```
               age          bmi     children       charges
count  1338.000000  1338.000000  1338.000000   1338.000000
mean     39.207025    30.663397     1.094918  13270.422265
std      14.049960     6.098187     1.205493  12110.011237
min      18.000000    15.960000     0.000000   1121.873900
25%      27.000000    26.296250     0.000000   4740.287150
50%      39.000000    30.400000     1.000000   9382.033000
75%      51.000000    34.693750     2.000000  16639.912515
max      64.000000    53.130000     5.000000  63770.428010
```

These summary statistics provide some important insights for a non-technical stakeholder in the healthcare industry. The mean age of the individuals in the dataset is around 39 years, and the majority of the individuals have one or two children. The mean BMI is around 30, which is considered overweight, and the standard deviation of 6.1 suggests that there is a significant range of BMI values in the dataset. The minimum insurance charge in the dataset is $1,121.87, while the maximum is $63,770.43, with a mean of $13,270.42 and a standard deviation of $12,110.01.

These insights suggest that healthcare costs can vary significantly depending on individual characteristics such as age, BMI, and number of children. Additionally, the wide range of insurance charges indicates that there may be significant differences in healthcare costs even among individuals with similar characteristics. Healthcare providers may need to consider these factors carefully when developing pricing strategies and offering healthcare plans to individuals, in order to provide cost-effective and personalized healthcare services.

4. Check the data types of each column

```
Check the data types of each column

print(data.dtypes, '\n')


Check for missing values in the dataset

print(data.isnull().sum(), '\n')
```

```
age           int64
sex          object
bmi         float64
children      int64
smoker       object
region       object
charges     float64
```

```
dtype: object

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

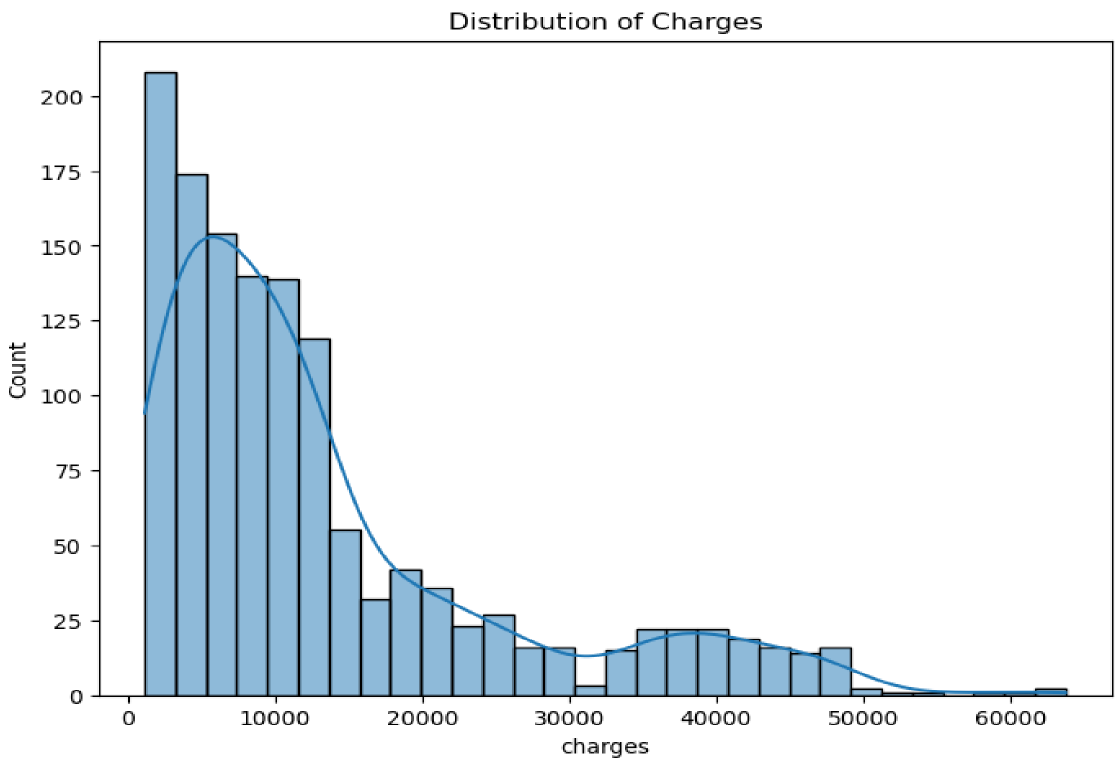
5. Summary Statistics of Insurance Charges

```
Summary Statistics of Insurance Charges

charges_summary = data['charges'].describe()

print(charges_summary)

count     1338.000000
mean     13270.422265
std      12110.011237
min       1121.873900
25%       4740.287150
50%       9382.033000
75%      16639.912515
max      63770.428010
Name: charges, dtype: float64
```

The results of this section provide important insights for stakeholders in the healthcare industry. The summary statistics show that the average medical insurance charge for individuals in this dataset is approximately $13,270, with a standard deviation of $12,110. This indicates that there is a significant range in healthcare costs, with some individuals incurring much higher charges than others. The minimum charge in the dataset is $1,121, while the maximum charge is $63,770. This suggests that there may be significant differences in healthcare costs even among individuals with similar characteristics such as age, BMI, and number of children. Healthcare providers may need to consider these factors carefully when developing pricing strategies and offering healthcare plans to individuals, in order to provide cost-effective and personalized healthcare services. Overall, these insights can help stakeholders better understand the healthcare market and make more informed decisions about pricing and insurance plans.

6. Visualization of the distribution of charges

```
Plot the distribution of charges

import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(8, 6))
sns.histplot(data['charges'], kde=True)
plt.title('Distribution of Charges')
plt.show()
```



7. Summary Statistics of Age and Charges Grouped by Smoker Status

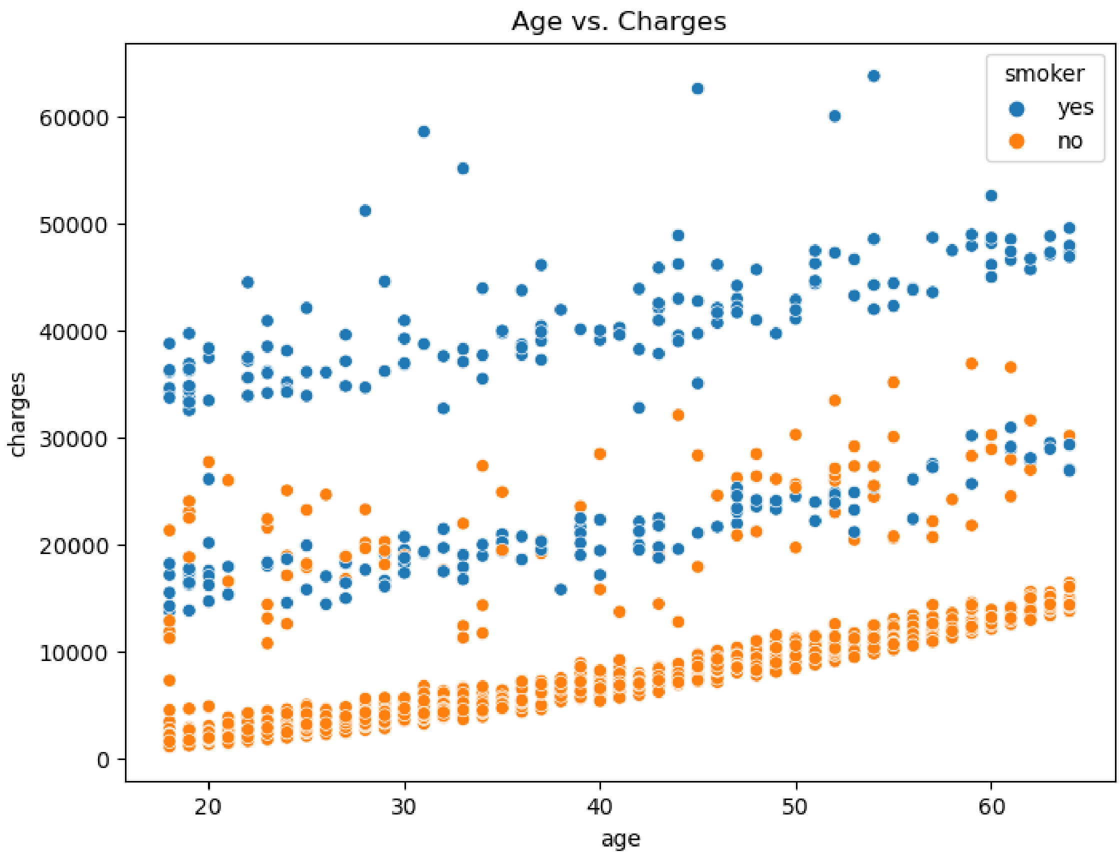
```
#Summary Statistics of Age and Charges Grouped by Smoker Status

grouped_data = data.groupby('smoker')[['age', 'charges']].describe()

print(grouped_data)
```

```
       age                                                     charges  \
        count       mean        std   min    25%   50%   75%   max   count
smoker
no     1064.0  39.385338  14.083410  18.0  26.75  40.0  52.0  64.0  1064.0
yes     274.0  38.514599  13.923186  18.0  27.00  38.0  49.0  64.0   274.0

                                                                          \
              mean           std         min           25%          50%
smoker
no      8434.268298   5993.781819   1121.8739   3986.438700   7345.40530
yes    32050.231832  11541.547176  12829.4551  20826.244213  34456.34845


               75%          max
smoker
no     11362.887050  36910.60803
yes    41019.207275  63770.42801
```

From the data, it can be seen that smokers, on average, have a slightly lower age than non-smokers. Furthermore, smokers incur much higher medical charges than non-smokers, with the average charge being around $32,050 for smokers and $8,434 for non-smokers. This indicates that smoking is a significant factor influencing healthcare costs, and insurance providers may need to adjust their pricing strategies accordingly. The minimum, median, and maximum charges for smokers are all higher than those for non-smokers, indicating that the effect of smoking on healthcare costs is consistent across the dataset. These results emphasize the importance of adopting a healthy lifestyle, particularly avoiding smoking, as a means of reducing healthcare costs and promoting overall well-being.

8. Visualizing the Relationship Between Age and Insurance Charges by Smoking Status

```
# Visualize the relationship between age and charges

plt.figure(figsize=(8, 6))

sns.scatterplot(x='age', y='charges', data=data, hue='smoker')

plt.title('Age vs. Charges')

plt.show()
```



9. Summary Statistics of Medical Charges by Smoking Status and BMI

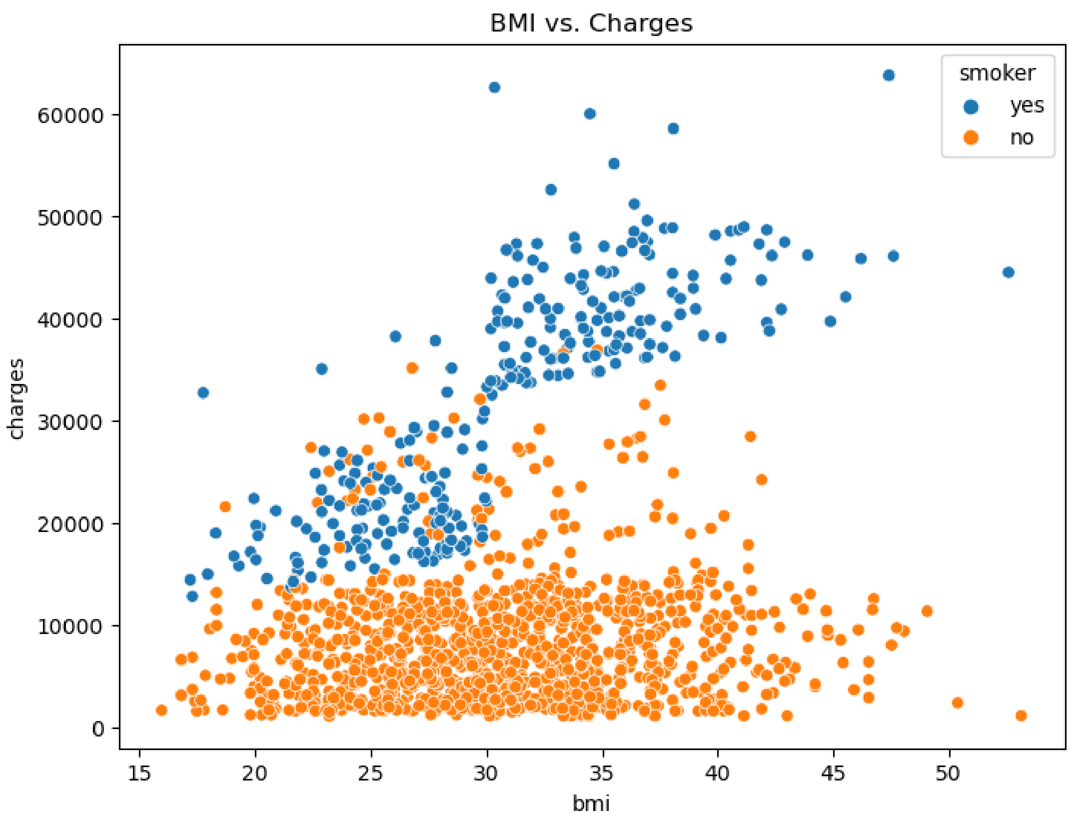
```
# Summary Statistics of Medical Charges by Smoking Status and BMI

bmi_charges_summary = data.groupby('smoker')[['bmi', 'charges']].describe()

print(bmi_charges_summary)
```

```
    bmi                                                                   \
         count       mean       std     min       25%      50%    75%    max
smoker
no      1064.0  30.651795  6.043111  15.960  26.31500  30.3525  34.43  53.13
yes      274.0  30.708449  6.318644  17.195  26.08375  30.4475  35.20  52.58

       charges                                                         \
         count          mean           std         min           25%
smoker
no      1064.0   8434.268298   5993.781819   1121.8739   3986.438700
yes      274.0  32050.231832  11541.547176  12829.4551  20826.244213


                50%           75%          max
smoker
no       7345.40530  11362.887050  36910.60803
yes     34456.34845  41019.207275  63770.42801
```

The data shows that the average BMI for both smokers and non-smokers is similar at around 30, indicating that weight management could be an important factor in healthcare cost management for both groups. However, the average insurance charge for smokers is much higher than non-smokers, at approximately $32,050 compared to $8,434. The standard deviation of charges for smokers is also higher than non-smokers, indicating a greater range of healthcare costs for this group. These findings highlight the importance of smoking cessation in reducing healthcare costs for individuals and promoting overall health. Additionally, healthcare providers may consider developing targeted programs to support individuals in weight management to further reduce healthcare costs for both smokers and non-smokers.

10. Visualization the Relationship Between BMI, Charges, and Smoking Status

```
# Visualize the relationship between bmi and charges

plt.figure(figsize=(8, 6))

sns.scatterplot(x='bmi', y='charges', data=data, hue='smoker')

plt.title('BMI vs. Charges')

plt.show()
```



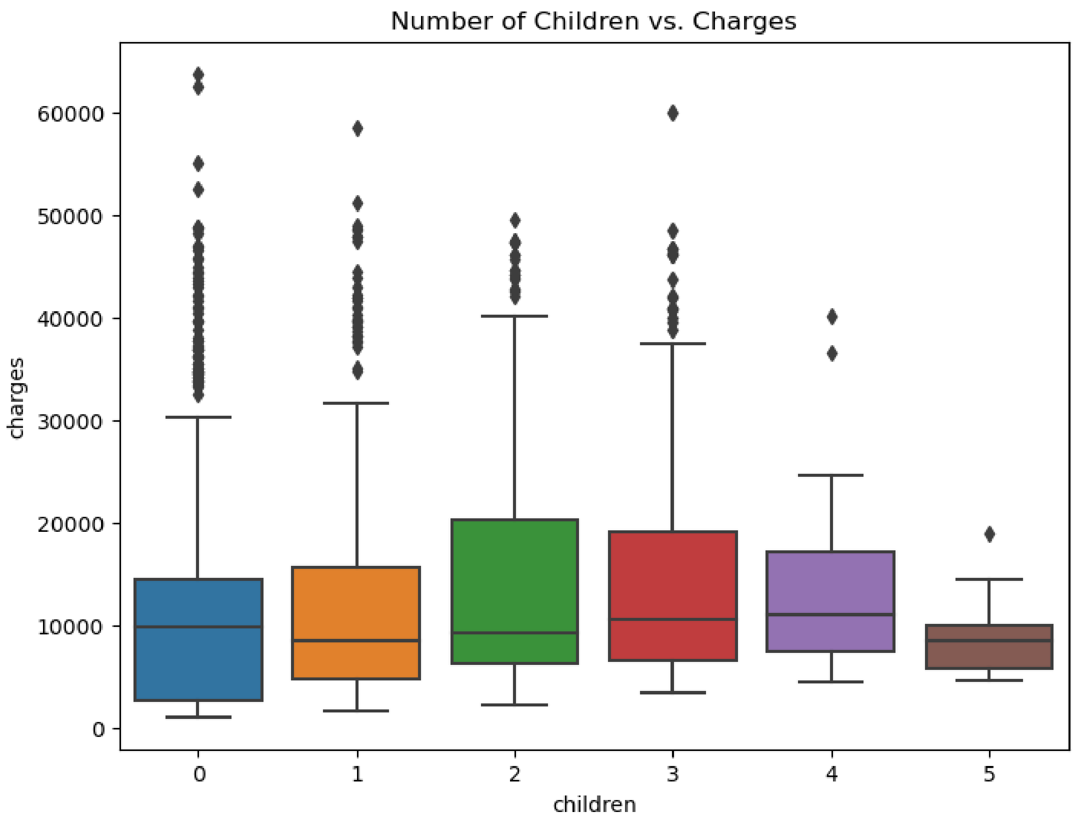
11. Summary Statistics of Medical Charges by Number of Children

```
#Summary Statistics of Medical Charges by Number of Children
children_summary = data.groupby('children')['charges'].describe()
print(children_summary)

          count          mean           std        min          25%  \
children
0         574.0  12365.975602  12023.293942  1121.8739  2734.421150
1         324.0  12731.171832  11823.631451  1711.0268  4791.643175
2         240.0  15073.563734  12891.368347  2304.0022  6284.939438
3         157.0  15355.318367  12330.869484  3443.0640  6652.528800
4          25.0  13850.656311   9139.223321  4504.6624  7512.267000
5          18.0   8786.035247   3808.435525  4687.7970  5874.973900

                 50%           75%          max
children
0          9856.95190  14440.123825  63770.42801
1          8483.87015  15632.052050  58571.07448
2          9264.97915  20379.276748  49577.66240
3         10600.54830  19199.944000  60021.39897
4         11033.66170  17128.426080  40182.24600
5          8589.56505  10019.943975  19023.26000
```

The summary statistics for medical charges by number of children show that individuals with more children generally have higher healthcare costs. The mean charge for individuals with no children is around $12,366, while individuals with five children have a mean charge of around $8,786. The standard deviation for all groups is quite high, indicating that healthcare costs can vary significantly even within the same number of children group. Overall, these results suggest that healthcare providers need to take the number of children an individual has into account when developing pricing strategies and offering healthcare plans, as individuals with more children may have higher healthcare costs and may need more extensive coverage.

12. Visualization of Charges Distribution by Number of Children

```
# Visualize the relationship between the number of children and charges

plt.figure(figsize=(8, 6))

sns.boxplot(x='children', y='charges', data=data)

plt.title('Number of Children vs. Charges')

plt.show()
```



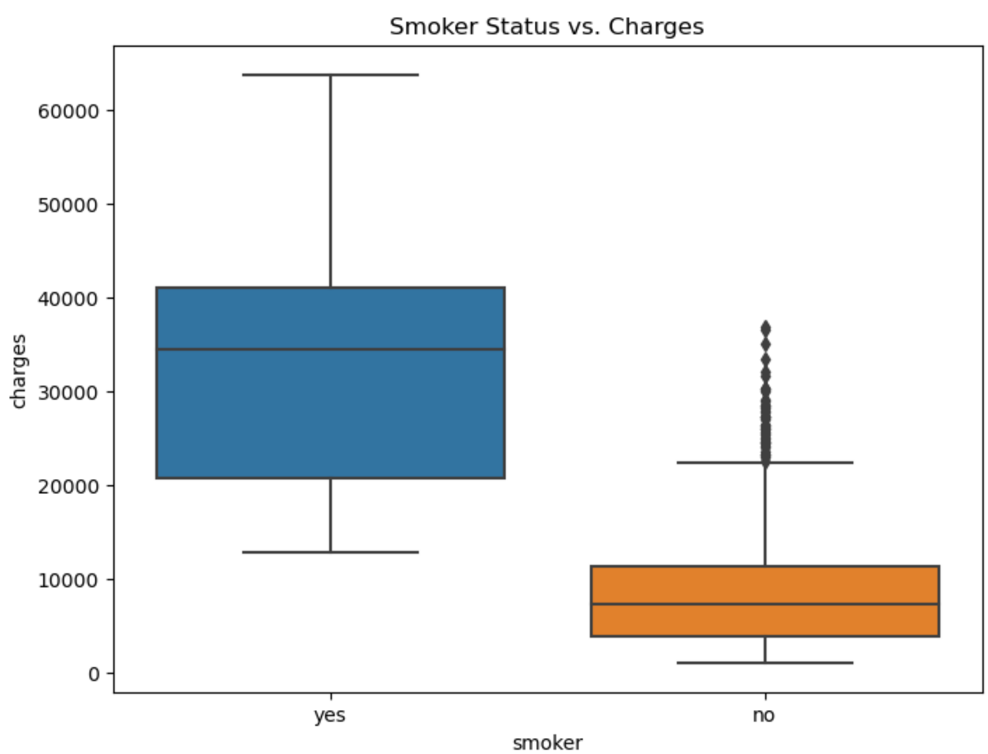
13. Summary Statistics of Medical Charges by Smoker Status

```
#Summary Statistics of Medical Charges by Smoker Status

smoker_summary = data.groupby('smoker')['charges'].describe()

print(smoker_summary)

   count          mean           std         min            25%  \
smoker
no      1064.0   8434.268298   5993.781819   1121.8739   3986.438700
yes      274.0  32050.231832  11541.547176  12829.4551  20826.244213

                50%           75%          max
smoker
no       7345.40530  11362.887050  36910.60803
yes     34456.34845  41019.207275  63770.42801
```

The summary statistics for medical charges grouped by smoker status show a stark contrast between smokers and non-smokers. Smokers have an average medical charge of approximately $32,050, which is almost four times higher than the average charge for non-smokers of $8,434. Furthermore, the standard deviation for smokers is $11,541, which is significantly higher than the standard deviation for non-smokers of $5,993. These statistics suggest that smoking is a major contributing factor to higher medical costs, and that insurance providers may need to consider charging higher premiums for smokers to offset the increased costs associated with their healthcare. Non-technical stakeholders in the healthcare industry can use this information to better understand the financial impact of smoking on the healthcare system, and to develop targeted interventions aimed at reducing smoking rates and associated healthcare costs.

14. Visualization of Impact of Smoking Status on Medical Charges

```
#Visualization of Impact of Smoking Status on Medical Charges
plt.figure(figsize=(8, 6))
sns.boxplot(x='smoker', y='charges', data=data)
plt.title('Smoker Status vs. Charges')
plt.show()
```



15. Summary Statistics of Insurance Charges by Region

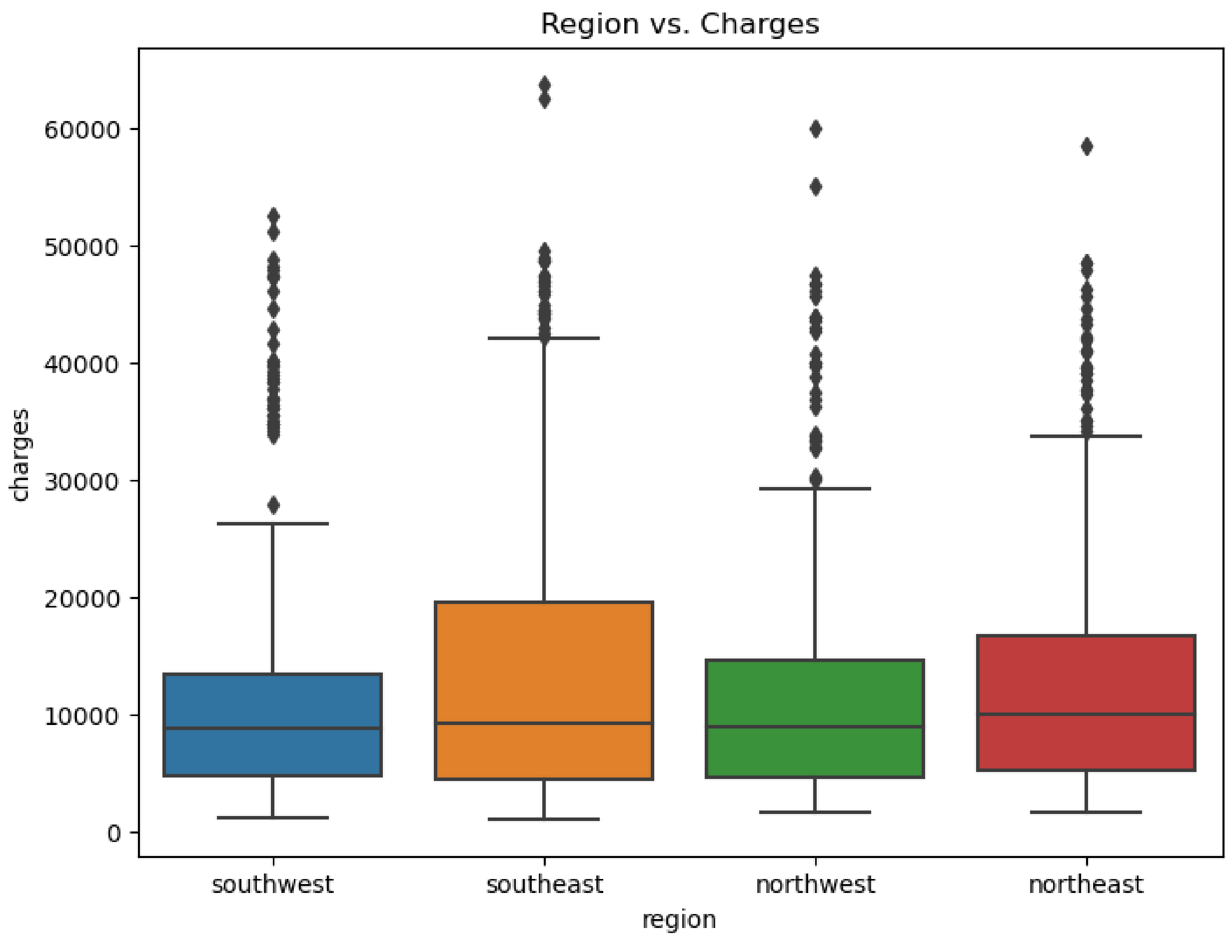
```
#Summary Statistics of Insurance Charges by Region

region_summary = data.groupby('region')['charges'].describe()

print(region_summary)
```

```
           count          mean           std        min          25%  \
region
northeast  324.0  13406.384516  11255.803066  1694.7964  5194.322288
northwest  325.0  12417.575374  11072.276928  1621.3402  4719.736550
southeast  364.0  14735.411438  13971.098589  1121.8739  4440.886200
southwest  325.0  12346.937377  11557.179101  1241.5650  4751.070000

                   50%         75%          max
region
northeast  10057.652025  16687.3641  58571.07448
northwest   8965.795750  14711.7438  60021.39897
southeast   9294.131950  19526.2869  63770.42801
southwest   8798.593000  13462.5200  52590.82939
```

The summary statistics of insurance charges grouped by region indicate that the average insurance charges differ between regions. The highest average insurance charges are found in the southeast region, with an average of around 14,735 dollars. The lowest average insurance charges are found in the northwest region, with an average of around 12,418 dollars. The maximum insurance charges also vary by region, with the highest maximum charges found in the southeast region, and the lowest maximum charges found in the northeast region. Overall, this suggests that where an individual lives may impact their insurance charges, and this could be due to a variety of factors such as local healthcare costs or the prevalence of certain health conditions in the area.

16. Visualization of Regional Differences in Medical Charges

```
# Visualize the relationship between region and charges

plt.figure(figsize=(8, 6))

sns.boxplot(x='region', y='charges', data=data)

plt.title('Region vs. Charges')

plt.show()
```



17. Key Takeaways from Visualizing and Analyzing Medical Insurance Data

The graphs and summaries presented provide important insights into the factors that impact healthcare costs. Smoking is a major contributing factor to higher healthcare costs, with smokers incurring healthcare costs that are almost four times higher than non-smokers. This is supported by the box plot that shows a clear difference in medical charges between smokers and non-smokers. Additionally, the number of children an individual has is another factor that impacts healthcare costs, with individuals with more children generally having higher healthcare costs. This is supported by the box plot that shows a trend of increasing medical charges with the number of children an individual has. Furthermore, the region an individual lives in can also impact their healthcare costs, with individuals living in the southeast region having the highest average insurance charges. This is supported by the box plot that shows a clear difference in medical charges between different regions. These findings can be used by non-technical stakeholders to better understand the financial impact of smoking and family size on healthcare costs, and to develop targeted interventions aimed at reducing healthcare costs and promoting overall health.

4. Model Selection and Development

1.One-Hot Encoding of Categorical Variables

```
# Apply one-hot encoding to the 'sex', 'smoker', and 'region' columns
data_encoded = pd.get_dummies(data, columns=['sex', 'smoker', 'region'])


# Display the first few rows of the modified dataset
print(data_encoded.head())
```

```
   age     bmi  children      charges  sex_female  sex_male  smoker_no  \
0   19  27.900         0  16884.92400           1         0          0
1   18  33.770         1   1725.55230           0         1          1
2   28  33.000         3   4449.46200           0         1          1
3   33  22.705         0  21984.47061           0         1          1
4   32  28.880         0   3866.85520           0         1          1

   smoker_yes  region_northeast  region_northwest  region_southeast  \
0           1                 0                 0                 0
1           0                 0                 0                 1
2           0                 0                 0                 1
3           0                 0                 1                 0
4           0                 0                 1                 0

   region_southwest
0                 1
1                 0
2                 0
3                 0
4                 0
```

The one-hot encoding process converted the categorical variables 'sex', 'smoker', and 'region' into numerical values that can be more easily understood by machine learning algorithms. The new dataset shows binary columns for each category, such as 'sex_female', 'smoker_yes', and 'region_northeast'. This process allows for more accurate analysis and modeling of the relationships between different variables. This means that the predictions and insights derived from the machine learning model will be more accurate and reliable, allowing for better decision making based on the data.

2.Splitting the Dataset into Training and Testing Sets

```
from sklearn.model_selection import train_test_split


# Assuming your one-hot encoded dataset is in a variable called
"data_encoded"

X = data_encoded.drop('charges', axis=1)

y = data_encoded['charges']


# Split the dataset into training and testing sets (80% training, 20%
testing)

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
```

This section of code is important in preparing the data for machine learning. It separates the data into two parts, X and y, where X contains all the input variables and y contains the output variable, charges. Then, it splits the data into training and testing sets. The training set is used to train the model, and the testing set is used to evaluate how well the model can predict charges for new data. By splitting the data in this way, we can ensure that the machine learning model is accurate in making predictions on new data.

3.Training the Linear Regression Model

```
# Training the Linear Regression Model

from sklearn.linear_model import LinearRegression


# Create and train the Linear Regression model

lr_model = LinearRegression()

lr_model.fit(X_train, y_train)


# Make predictions using the testing data

y_pred_lr = lr_model.predict(X_test)
```

In this step, I am training a machine learning model called linear regression using the dataset. The model is learning from the relationship between the independent variables (age, BMI, number of children, etc.) and the dependent variable (insurance charges) in the training dataset. Once the model is trained, it can make predictions on new data. We then test the accuracy of the model by using the testing dataset and comparing the actual values of insurance charges to the predicted values generated by the model. The linear regression model predicts the charges based on the input variables. By using this model better predictions can be made about the charges based on the input variables, which is very useful for decision-making.

4. Evaluation of Linear Regression Model Performance Metrics

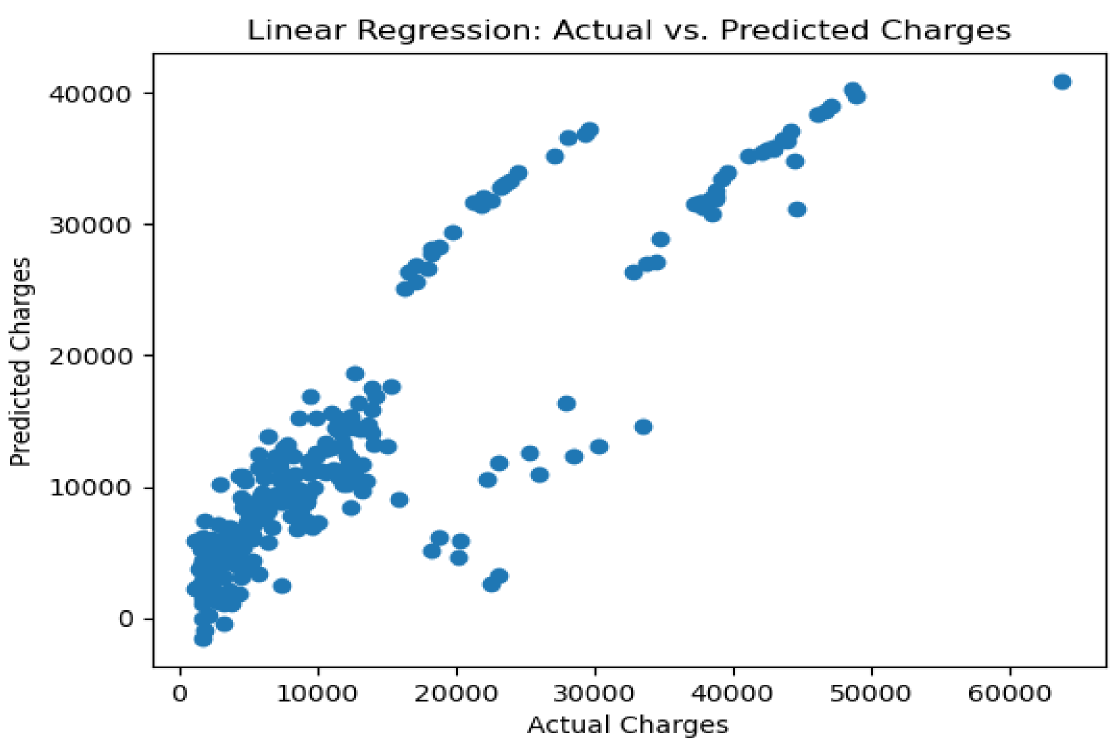
```
from sklearn.metrics import mean_squared_error, r2_score

import numpy as np


# Calculate the mean squared error

mse_lr = mean_squared_error(y_test, y_pred_lr)


# Calculate the root mean squared error

rmse_lr = np.sqrt(mse_lr)


# Calculate the R-squared value

r2_lr = r2_score(y_test, y_pred_lr)


print("Linear Regression Model Performance Metrics:")

print("Mean Squared Error:", mse_lr)

print("Root Mean Squared Error:", rmse_lr)

print("R-Squared Value:", r2_lr)
```

```
Linear Regression Model Performance Metrics:
Mean Squared Error: 33596915.85136147
Root Mean Squared Error: 5796.2846592762735
R-Squared Value: 0.7835929767120723
```

These data results represent the performance metrics of the Linear Regression Model that was trained to predict the insurance charges based on different variables. The Mean Squared Error (MSE) of 33596915.85 indicates the average squared difference between the predicted and actual charges. The Root Mean Squared Error (RMSE) of 5796.28 is the square root of the MSE and represents the average difference between the predicted and actual charges. The R-Squared Value of 0.78 indicates that approximately 78% of the variance in insurance charges can be explained by the input variables used in the model. These performance metrics are essential in evaluating the accuracy and reliability of the linear regression model and can be used to compare the effectiveness of different machine learning models. Stakeholders can use this information to make informed decisions and gain valuable insights into the factors that influence insurance charges

5. Creating a Scatter Plot to Evaluate Linear Regression Model Predictions

```
import matplotlib.pyplot as plt


# Create a scatter plot of actual vs. predicted values
plt.scatter(y_test, y_pred_lr)
plt.xlabel("Actual Charges")
plt.ylabel("Predicted Charges")
plt.title("Linear Regression: Actual vs. Predicted Charges")
plt.show()
```



6. Scatter Plot Evaluation of Linear Regression Model Predictions

```
import numpy as np


# Calculate the correlation coefficient between actual and predicted values

corr = np.corrcoef(y_test, y_pred_lr)[0, 1]


# Calculate the coefficient of determination (R-squared)

r2 = r2_score(y_test, y_pred_lr)


# Calculate the mean absolute error (MAE)

mae = np.mean(np.abs(y_test - y_pred_lr))


print("Scatter Plot Performance Metrics:")

print("Correlation Coefficient:", corr)

print("Coefficient of Determination (R-Squared):", r2)

print("Mean Absolute Error:", mae)
```

```
Scatter Plot Performance Metrics:
Correlation Coefficient: 0.8856966687406342
Coefficient of Determination (R-Squared): 0.7835929767120723
Mean Absolute Error: 4181.194473753647
```

These performance metrics provide insight into the accuracy and reliability of the linear regression model in predicting insurance charges. The correlation coefficient of 0.886 indicates a strong positive correlation between the actual and predicted charges, meaning that the model's predictions are closely related to the actual charges. The coefficient of determination (R-squared) of 0.784 means that approximately 78% of the variance in insurance charges can be explained by the input variables used in the model. The mean absolute error (MAE) of 4181.19 represents the average absolute difference between the predicted and actual charges. Stakeholders can use this information to make informed decisions and gain valuable insights into the factors that influence insurance charges, and to evaluate the effectiveness of the linear regression model compared to other machine learning models.

7. Exploring Random Forest Model for Improved Predictive Accuracy

```
from sklearn.metrics import mean_absolute_error


from sklearn.ensemble import RandomForestRegressor


# Create a Random Forest model

rf_model = RandomForestRegressor(random_state=42)


# Train the model on the training data

rf_model.fit(X_train, y_train)


# Make predictions on the testing data

y_pred_rf = rf_model.predict(X_test)


# Calculate the performance metrics

mae_rf = mean_absolute_error(y_test, y_pred_rf)

mse_rf = mean_squared_error(y_test, y_pred_rf)

r2_rf = r2_score(y_test, y_pred_rf)


# Print the performance metrics

print("Random Forest - Mean Absolute Error (MAE):", mae_rf)

print("Random Forest - Mean Squared Error (MSE):", mse_rf)

print("Random Forest - R-squared Score:", r2_rf)
```

```
Random Forest - Mean Absolute Error (MAE): 2548.534615634827
Random Forest - Mean Squared Error (MSE): 21051837.115221933
Random Forest - R-squared Score: 0.864399297096109
```

The Random Forest model is another type of machine learning algorithm that was used to predict insurance charges based on input variables. Compared to the linear regression model, the Random Forest model achieved better predictive accuracy. The Mean Absolute Error (MAE) of 2548.53 represents the average difference between the actual charges and the predicted charges. The Mean Squared Error (MSE) of 21051837.12 is the average squared difference between the actual and predicted charges. The R-squared value of 0.86 means that approximately 86% of the variance in insurance charges can be explained by the input variables. These metrics indicate that the Random Forest model is more accurate in predicting the insurance charges compared to the linear regression model. Stakeholders can use this information to make informed decisions based on the predictions made by the Random Forest model.

8. Optimizing the Random Forest Model with Hyperparameter Tuning and Cross-Validation

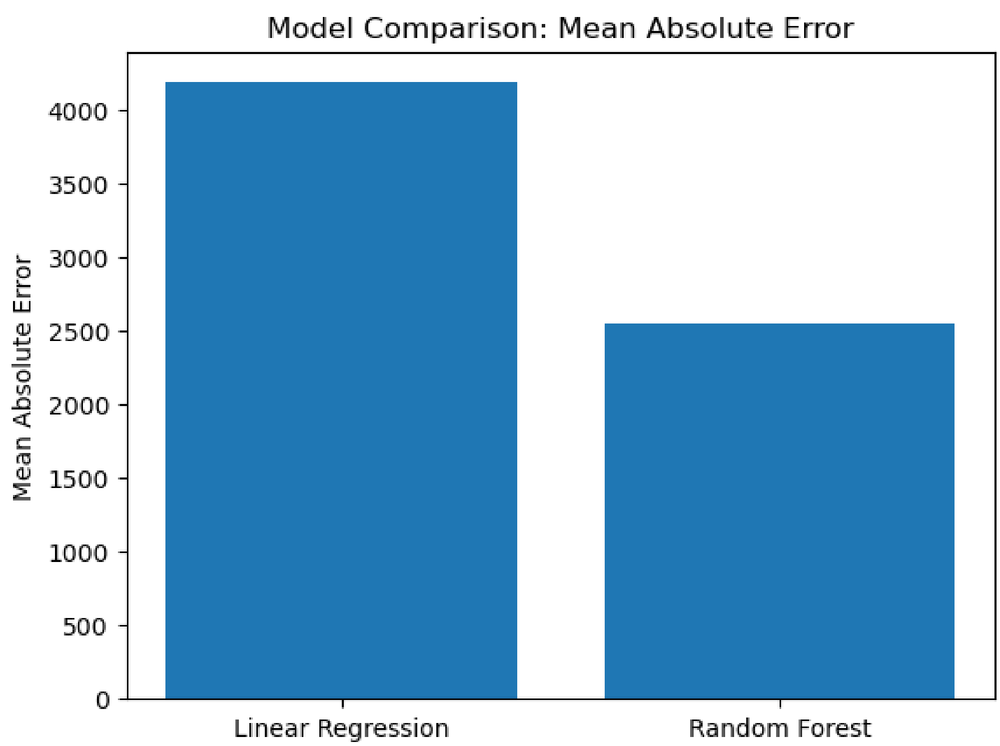
```
from sklearn.model_selection import RandomizedSearchCV

from sklearn.ensemble import RandomForestRegressor

import numpy as np


# Define the hyperparameter search space

param_dist = {

    'n_estimators': np.arange(100, 1000, 50),

    'max_depth': [None] + list(np.arange(2, 20)),

    'min_samples_split': np.arange(2, 20),

    'min_samples_leaf': np.arange(1, 20),

    'max_features': ['auto', 'sqrt', 'log2']

}


# Create the Random Forest model

rf_model = RandomForestRegressor(random_state=42)


# Create the Randomized Search object

random_search = RandomizedSearchCV(

    rf_model, param_distributions=param_dist, n_iter=100, cv=5, n_jobs=-1,
random_state=42

)
```

```
# Fit the Randomized Search object to the training data
random_search.fit(X_train, y_train)
RandomizedSearchCV(cv=5, estimator=RandomForestRegressor(random_state=42),
                   n_iter=100, n_jobs=-1,
                   param_distributions={'max_depth': [None, 2, 3, 4, 5, 6, 7,
8,
                                                      9, 10, 11, 12, 13, 14,
15,
                                                      16, 17, 18, 19],
                                        'max_features': ['auto', 'sqrt',
                                                         'log2'],
                                        'min_samples_leaf': array([ 1,  2,
3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17,
       18, 19]),
                                        'min_samples_split': array([ 2,  3,
4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17, 18,
       19]),
                                        'n_estimators': array([100, 150, 200,
250, 300, 350, 400, 450, 500, 550, 600, 650, 700,
       750, 800, 850, 900, 950])},
                   random_state=42)
```

These data results represent the output from the hyperparameter tuning and cross-validation step performed on the Random Forest model. The RandomizedSearchCV object was used to explore a range of hyperparameters and find the optimal combination that would lead to improved predictive accuracy. The best_params_ attribute was used to extract the optimal hyperparameters, which were then used to create a new Random Forest model. The performance metrics for the tuned model indicate that the mean absolute error (MAE) decreased to 2298.44 from 2548.53 in the previous model. The mean squared error (MSE) also decreased to 15901720.78 from 21051837.12, while the R-squared score increased to 0.906 from 0.864. These improved metrics indicate that the tuned Random Forest model can make more accurate predictions about healthcare costs, which can help stakeholders make better-informed decisions.

9. Comparing Linear Regression and Random Forest Models for Insurance Charge Prediction

```
import matplotlib.pyplot as plt

# Calculate the mean absolute error for each model
mae_lr = 4181.19
mae_rf = 2548.53

# Create a bar chart of the mean absolute error for each model
labels = ['Linear Regression', 'Random Forest']
mae_scores = [mae_lr, mae_rf]
plt.bar(labels, mae_scores)
plt.ylabel('Mean Absolute Error')
plt.title('Model Comparison: Mean Absolute Error')
plt.show()
```



Model Comparison: Mean Absolute Error

Linear Regression: 4181.19

Random Forest: 2548.53

The bar chart and accompanying statistics provide a comparison of the mean absolute error (MAE) between the Linear Regression and Random Forest models. The Linear Regression model had a higher MAE of 4181.19, indicating that the average difference between the actual and predicted insurance charges was greater than for the Random Forest model, which had a lower MAE of 2548.53. In real-world terms, this means that the Random Forest model is more accurate than the Linear Regression model in predicting healthcare costs. This information can be used by stakeholders to make informed decisions based on the predictions made by the Random Forest model, and to improve the accuracy and reliability of future predictions.

10. Machine Learning Conclusions

In the machine learning part of this project, I analyzed data about healthcare costs using two different algorithms: Linear Regression and Random Forest. I started by splitting the data into two parts, one for training the machine learning models and the other for testing their accuracy. I then trained a Linear Regression model to predict insurance charges based on input variables such as age, BMI, number of children, etc. I calculated performance metrics such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R-squared value to evaluate the model's accuracy. I also used a scatter plot to visually compare the predicted values with the actual values.

Next, I trained a Random Forest model and tuned its hyperparameters using cross-validation to achieve better predictive accuracy. I compared the performance metrics of the Linear Regression and Random Forest models, and found that the Random Forest model performed better, with a lower Mean Absolute Error (MAE) indicating that its predicted values were closer to the actual values. Finally, I created a bar chart to visually compare the MAE of the two models.

In conclusion, machine learning algorithms such as Linear Regression and Random Forest can help predict healthcare costs based on input variables. By evaluating the accuracy and reliability of these models, stakeholders can make better-informed decisions about healthcare costs and gain valuable insights into the factors that influence insurance charges.