**Insurance Premium Prediction Project Report**

**1. Summary of Results and Findings**

The notebook evaluates several models to predict insurance premiums. The models considered include:

* **Linear Regression**
* **Decision Tree Regressor**
* **Random Forest Regressor**
* **Polynomial Regression**

After comparing the models, it was found that **Polynomial Regression** achieved the highest accuracy, with a score of **85.28%**.

**2. Highlight of the Best Performing Model**

The **Polynomial Regression** model is identified as the best performing model for predicting insurance premiums, achieving an accuracy of **85.28%**. This model was selected over others due to its ability to capture non-linear relationships between the features and the target variable, leading to better performance compared to simpler models like Linear Regression.

**3. Potential Impact and Benefits of Accurate Insurance Premium Prediction**

Accurate insurance premium prediction can have significant benefits, including:

* **Risk Assessment:** Insurers can more precisely assess the risk associated with individual policyholders, leading to fairer premium pricing.
* **Customer Satisfaction:** More accurate premiums can lead to increased customer satisfaction, as policyholders are charged rates that more closely match their risk profile.
* **Profitability:** Accurate predictions help insurers maintain profitability by avoiding undercharging high-risk clients and overcharging low-risk clients.
* **Market Competitiveness:** Insurers with more accurate prediction models can offer competitive rates, attracting more customers while managing risks effectively.

**4. Step-by-Step Code Outline for the Insurance Premium Prediction Project**

Below is a high-level code outline for developing an insurance premium prediction model:

**Step 1: Import Necessary Libraries**

**import numpy as np**

**import pandas as pd**

**from sklearn.model\_selection import train\_test\_split**

**from sklearn.preprocessing import PolynomialFeatures, StandardScaler**

**from sklearn.linear\_model import LinearRegression**

**from sklearn.metrics import mean\_squared\_error, r2\_score**

**from sklearn.tree import DecisionTreeRegressor**

**from sklearn.ensemble import RandomForestRegressor**

**import matplotlib.pyplot as plt**

**import seaborn as sns**

**Step 2: Load and Explore the Dataset**

**# Load the dataset**

**df = pd.read\_csv('insurance.csv')**

**# Explore the dataset**

**print(df.head())**

**print(df.describe())**

**print(df.info())**

**Step 3: Data Preprocessing**

**# Handle categorical variables using one-hot encoding**

**df = pd.get\_dummies(df, drop\_first=True)**

**# Split the dataset into features and target variable**

**X = df.drop('charges', axis=1)**

**y = df['charges']**

**# Split 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)**

**Step 4: Model Training and Evaluation**

Linear Regression:

lr = LinearRegression()

lr.fit(X\_train, y\_train)

y\_pred = lr.predict(X\_test)

print(f'Linear Regression R2 Score: {r2\_score(y\_test, y\_pred)}')

Polynomial Regression:

poly = PolynomialFeatures(degree=2)

X\_poly = poly.fit\_transform(X)

X\_train\_poly, X\_test\_poly, y\_train, y\_test = train\_test\_split(X\_poly, y, test\_size=0.2, random\_state=42)

poly\_lr = LinearRegression()

poly\_lr.fit(X\_train\_poly, y\_train)

y\_pred\_poly = poly\_lr.predict(X\_test\_poly)

print(f'Polynomial Regression R2 Score: {r2\_score(y\_test, y\_pred\_poly)}')

Decision Tree Regressor:

dt = DecisionTreeRegressor(random\_state=42)

dt.fit(X\_train, y\_train)

y\_pred\_dt = dt.predict(X\_test)

print(f'Decision Tree Regressor R2 Score: {r2\_score(y\_test, y\_pred\_dt)}')

Random Forest Regressor:

rf = RandomForestRegressor(n\_estimators=100, random\_state=42)

rf.fit(X\_train, y\_train)

y\_pred\_rf = rf.predict(X\_test)

print(f'Random Forest Regressor R2 Score: {r2\_score(y\_test, y\_pred\_rf)}')

**Step 5: Model Comparison and Selection**

# Comparing model performances and selecting the best one

models = pd.DataFrame({

'Model': ['Linear Regression', 'Polynomial Regression', 'Decision Tree', 'Random Forest'],

'R2 Score': [r2\_score(y\_test, y\_pred), r2\_score(y\_test, y\_pred\_poly), r2\_score(y\_test, y\_pred\_dt), r2\_score(y\_test, y\_pred\_rf)]

})

print(models)

# Plotting the results for better visualization

sns.barplot(x='R2 Score', y='Model', data=models)

plt.title('Model Comparison')

plt.show()

Step 6: Conclusion

# Final model recommendation

best\_model = 'Polynomial Regression' if models.loc[1, 'R2 Score'] > max(models['R2 Score']) else 'Other Model'

print(f'The recommended model is: {best\_model}')

This project identified Polynomial Regression as the most effective model for predicting insurance premiums, achieving the highest accuracy. Accurate predictions enhance risk management and fair pricing, benefiting both insurers and customers. The findings highlight the value of advanced machine learning techniques in improving decision-making and business outcomes in the insurance industry.