CAPSTONE PROJECT

Predicting Annual Medical Expenditures Using Machine Learning for Health Insurance Premium Estimation

PRESENTED BY

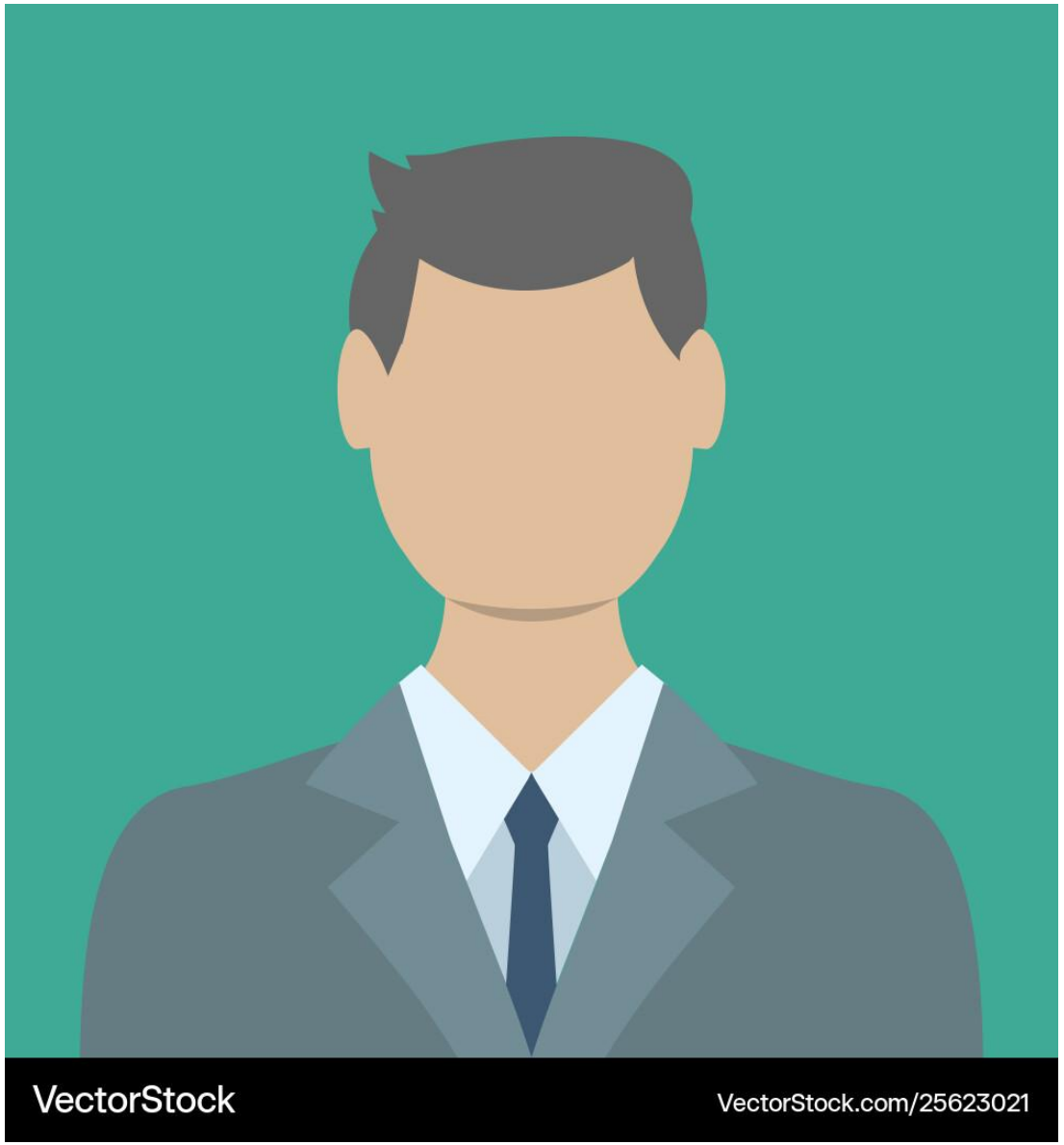
STUDENT NAME: Vedant Suhas Kale

COLLEGE NAME: PIMPRI CHINCHWAD UNIVERSITY

DEPARTMENT: COMPUTER SCIENCE

EMAIL ID: vedantskale3@gmail.com

AICTE STUDENT ID: STU678b41f7addfe1737179639



OUTLINE

- **Problem Statement**
- **Proposed System/Solution**
- **System Development Approach**
- **Algorithm & Deployment**
- **Result**
- **Conclusion**
- **Future Scope**
- **References**

PROBLEM STATEMENT

ACME Insurance Inc. offers affordable health insurance to thousands of customers across the United States. The company faces the challenge of accurately estimating the annual medical expenditure of a new customer, which is critical for determining appropriate insurance premium amounts. The estimation must be made using basic customer information such as age, gender, BMI, number of children, smoking habits, and region of residence. A reliable prediction system will help automate the premium calculation process, ensuring fairness, consistency, and efficiency.

PROPOSED SOLUTION

The proposed system will use machine learning techniques to predict a customer's annual medical expenses. The solution includes:

- **Data Collection:**
  Use historical data containing features like age, sex, BMI, number of children, smoking status, and region, along with actual medical charges.
- **Data Preprocessing:**
  - Encode categorical variables such as gender, smoking status, and region.
  - Handle missing values and normalize numeric features like age and BMI.
- **Feature Engineering:**
  - Convert categorical values to numerical codes or dummy variables using one-hot encoding.
  - Analyze correlations between features and the target (charges).
- **Model Training:**
  - Use regression algorithms like Linear Regression and Random Forest Regressor.
  - Evaluate models using RMSE (Root Mean Squared Error).
- **Prediction:**
  - Predict annual charges for a new customer based on their profile.
  - Use the prediction to assist in premium setting.

SYSTEM APPROACH

System Requirements:-

•Python (Jupyter Notebook)
•Required Libraries: pandas, numpy, sklearn, matplotlib, seaborn

Steps :-
1.Load and clean data
2.Encode categorical features
3.Train/test split
4.Train regression models
5.Evaluate model performance
6.Predict new customer charges

ALGORITHM & DEPLOYMENT

- **Algorithm Used: Linear Regression & RandomForestRegressor**

- **Input Features:**
  - age, bmi, children, sex_code, smoker_code, and one-hot encoded region (northeast, northwest, southeast, southwest)

- **Training:**
  - 80/20 train-test split
  - RMSE used to evaluate prediction error

- **Prediction:**
  - The model is used to estimate expenses for new individuals using a single input vector with the same structure.

- **Deployment Idea:**
  - The model can be embedded in a web interface for use by insurance agents or customers.

RESULT

- **Models Used:**

❖Linear Regression

❖Random Forest Regressor

- **Performance Metric:**

❖RMSE (Root Mean Squared Error) used to evaluate model accuracy.

Linear Regression using Scikit-learn

In practice, you'll never need to implement either of the above methods yourself. You can use a library like `scikit-learn` to do this for you.

[39]:
```
from sklearn.linear_model import LinearRegression
```

Let's use the `LinearRegression` class from `scikit-learn` to find the best fit line for "age" vs. "charges" using the ordinary least squares optimization technique.

First, we create a new model object.

[40]:
```
model = LinearRegression()
model
```

[40]:
```
LinearRegression()
```

Next, we can use the `fit` method of the model to find the best fit line for the inputs and targets.

Not that the input `X` must be a 2-d array, so we'll need to pass a dataframe, instead of a single column.

[41]:
```
inputs = non_smoker_df[['age']]
targets = non_smoker_df.charges
print('Inputs.Shape:',inputs.shape)
print('Targets.Shape:',targets.shape)
```

```
Inputs.Shape: (1064, 1)
Targets.Shape: (1064,)
```

Let's fit the model to the data.

[42]:
```
model.fit(inputs, targets)
```

[42]:
```
LinearRegression()
```

We can now make predictions using the model. Let's try predicting the charges for the ages 23, 37 and 61

[2]:
```
model.predict(np.array([[23], [50], [60]]))
```

RESULT

[45]:
```
predictions = model.predict(inputs)
```

[46]:
```
predictions
```

[46]:
```
array([2719.0598744 , 5391.54900271, 6727.79356686, ..., 2719.0598744 ,
       2719.0598744 , 3520.80661289])
```

Let's compute the RMSE loss to evaluate the model.

[47]:
```
rmse(targets, predictions)
```

[47]:
```
4662.505766636395
```

Seems like our prediction is off by $4000 on average, which is not too bad considering the fact that there are seve

The parameters of the model are stored in the `coef_` and `intercept_` properties.

[48]:
```
model.intercept_
```

[48]:
```
-2091.4205565650827
```

[49]:
```
model.coef_
```

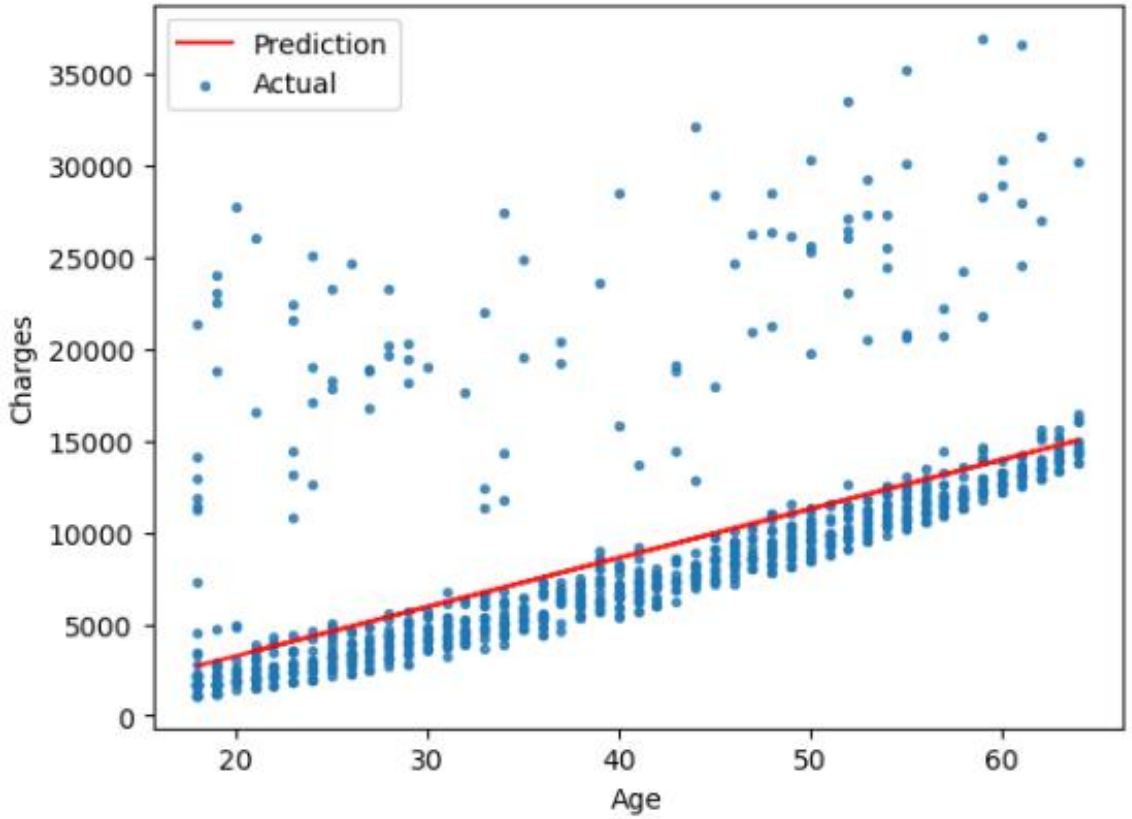
[49]:
```
array([267.24891283])
```

Are these parameters close to your best guesses?

Let's visualize the line created by the above parameters.

[50]:
```
try_parameters(model.coef_, model.intercept_)
```

```
RMSE LOSS: 4662.505766636395
```



RESULT

- **Sample Prediction:**

❖ **Input:- 19 years old, BMI: 24, 0 children, Non-smoker, Female, Southwest region**

- **OUTPUT:-**

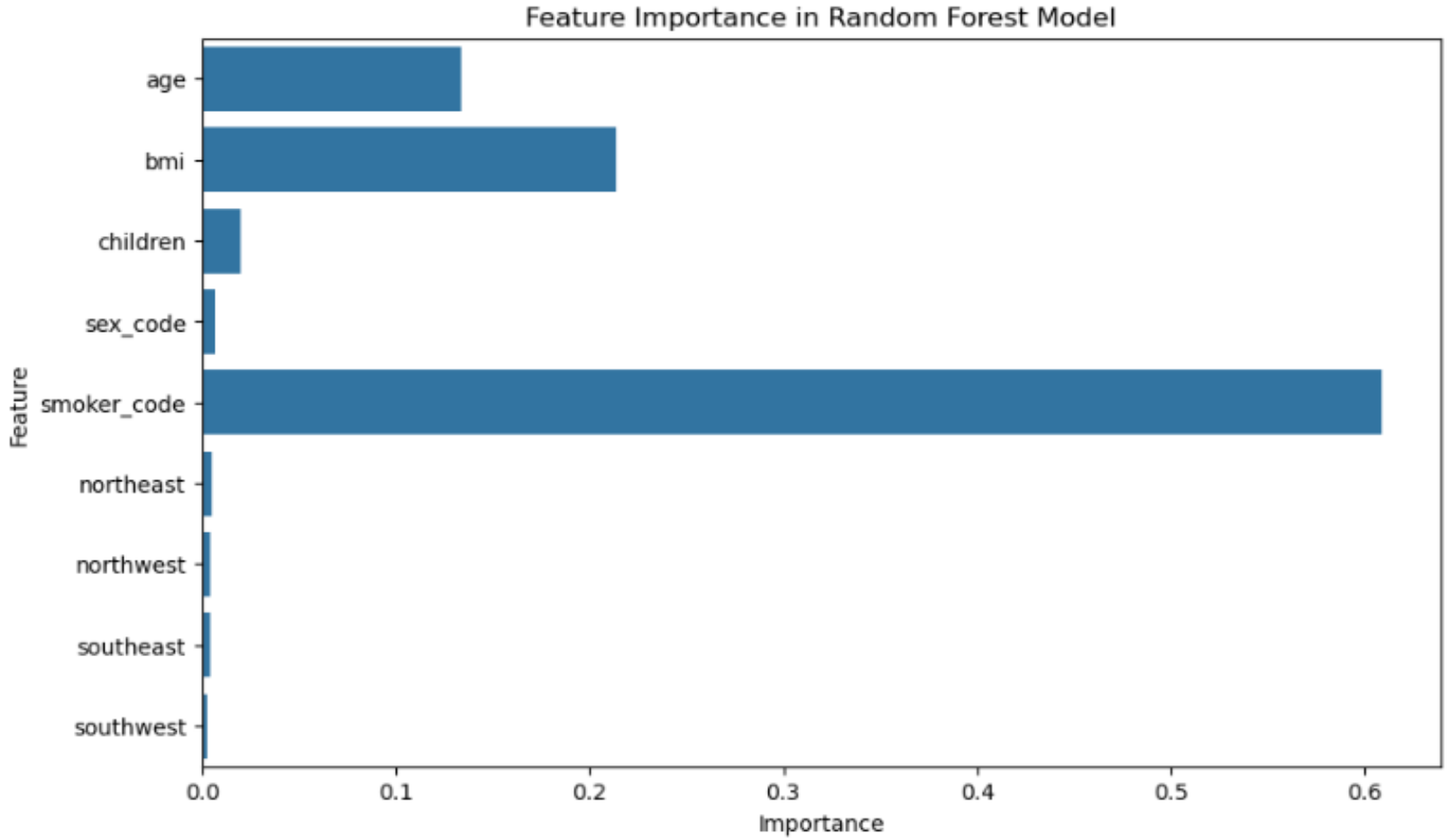
❖ **Predicted Annual Charge: $1759.61**

```
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split

# Prepare inputs and targets
inputs = medical_df[['age','bmi','children','sex_code','smoker_code',
                     'northeast','northwest','southeast','southwest']]
targets = medical_df['charges']

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(inputs, targets, test_size=0.2, random_state=42)

# Train model
model_rf = RandomForestRegressor(random_state=42)
model_rf.fit(X_train, y_train)

# Evaluate
predictions = model_rf.predict(X_test)
loss = rmse(y_test, predictions)
print('Test RMSE Loss:', loss)

# Predict for a new sample
new_data = pd.DataFrame([[19, 24, 0, 0, 0, 0, 0, 0, 1]],
    columns=['age', 'bmi', 'children', 'sex_code', 'smoker_code', 'northeast', 'northwest', 'southeast', 'southwest'])

print('New Prediction:', model_rf.predict(new_data)[0])
```

```
Test RMSE Loss: 4584.737145025518
New Prediction: 1765.829253999998
```

RESULT

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

feat_importances = pd.Series(model_rf.feature_importances_, index=inputs.columns)
plt.figure(figsize=(10,6))
sns.barplot(x=feat_importances.values, y=feat_importances.index)
plt.title("Feature Importance in Random Forest Model")
plt.xlabel("Importance")
plt.ylabel("Feature")
plt.show()
```



- **Evaluation Results:**

❖ **Linear Regression RMSE: ~4662.50**

❖ **Random Forest RMSE: ~4584.73**

(Random Forest outperforms Linear Regression in prediction accuracy.)

CONCLUSION

This project successfully demonstrates a machine learning-based approach to estimate annual medical expenditures using basic demographic and lifestyle data. The system can automate the insurance premium calculation process, making it more efficient and scalable.

FUTURE SCOPE

- Integrate additional features like medical history, income level, or lifestyle habits.

- Deploy the model using a Flask or Streamlit web app.

- Improve accuracy using advanced models like Gradient Boosting or Neural Networks.

- Enable real-time prediction for use in mobile apps or agent dashboards.

REFERENCES

- Machine Learning with Python – Coursera, Andrew Ng
- Dataset: Kaggle Medical Cost *Dataset*
- GitHub Link: *LINK*

Thank you