Health Insurance Cost Prediction Using Regression Models

A Project Report

Submitted by

SIDDHANT NARAYAN BHANDARI (230320525042)

SHIVANI RAJESH CHAVAN (230320525012)

AKANCHA(230320525007)

NAZIA ASHRAF (230320525023)

Under the guidance of

MS.PRITI BHARDWAJ

Professor

in partial fulfillment for the award of the degree of

PG Diploma in Big Data Analytics (PG-DBDA)

Centre for Development of Advanced Computing

AUGUST 2023

CERTIFICATE OF APPROVAL

This is to certify that this project entitled **"Health Insurance Cost Prediction using ML"** submitted by **SIDDHANT NARAYAN BHANDARI, CHAVAN SHIVANI RAJESH, AKANCHA**, **NAZIA ASHRAF** of Department of PG DIPLOMA IN BIG DATA ANALYTICS, SESSION 2023 OF Centre Of Development Of Advance Computing (CDAC) Noida is worthy of consideration for the partial fulfillment of the requirement for the award of degree in PG DIPLOMA IN IG DATA ANALYTICS, SESSION 2023 OF Centre Of Development Of Advance Computing (CDAC), Noida.

……………………………… ………………………………

INTERNAL EXAMINER EXTERNAL EXAMINER

………………………………

HEAD OF DEPARTMENT

ACKNOWLEDGEMENT

First and foremost we would like to thank ALMIGHTY who has provided us the strength to do just ice to our work and contribute our best to it We wish to express our deep sense of gratitude from th e bottom of our hearts to our guide. Priti Bhardwaj, Professor, of Big Data Analytics, for her motiv ating discussion, overwhelming suggestions, ingenious encouragement, invaluable supervision, an d exemplary guidance throughout this project work We would like to extend our heartfelt gratitude to Ms. Priti Bhardwaj, Professor, Professor Big Data Analytics for his valuable suggestions and su pport in successfully completion of project We thank the management of Centre for Development of Advanced Computing, NOIDA for providing us the necessary facilities and support required for the successful completion of the project As a final word, we would like to thank each and every in dividual who have been a source of support and encouragement and helped us to achieve our goal and complete our project work successfully

ABSTRACT

Health insurance cost prediction is a data-driven task that leverages historical healthcare data, including factors such as age, gender, BMI, smoking status, pre-existing conditions, and geographic location, to estimate future healthcare expenses. Machine learning and statistical modeling techniques have gained prominence in this domain due to their ability to analyze vast datasets and derive meaningful insights.Key methodologies employed in health insurance cost prediction include linear regression, decision trees, random forests, support vector machines, and neural networks. Feature engineering plays a crucial role in extracting relevant information from the data, while model evaluation metrics like Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) gauge prediction accuracy. As healthcare costs continue to rise, the ability to predict health insurance costs accurately becomes increasingly crucial. This abstract highlights the importance of leveraging advanced data analytics and machine learning techniques to develop reliable models for health insurance cost prediction, ultimately leading to better-informed decision and improved healthcare outcomes.

Index

CHAPTER 1 : INTRODUCTION

INTRODUCTION

In this project, we aim to leverage the power of regression modeling to predict health insurance costs. We will employ historical data on various variables, such as age, gender, geographic location, lifestyle factors, pre-existing conditions, and more, to build a robust predictive model. By analyzing and learning from past patterns, we can create a model that provides accurate cost estimates for individuals or populations, thereby aiding in financial planning, risk assessment, and decision-making. Health insurance cost prediction involves the use of data analysis and predictive modeling techniques to estimate the future expenses associated with an individual's or a group's health insurance coverage. Advancements in data analytics, machine learning, and artificial intelligence have revolutionized the field of health insurance cost prediction. These technologies allow for the analysis of vast datasets and the development of sophisticated predictive models that can adapt to changing conditions and provide more accurate forecasts.

CHAPTER 1.2 : DATA COLLECTION AND VISUALIZATION

Data Collection

A Kaggle health insurance cost prediction dataset typically includes a range of attributes or features that describe individuals' characteristics and their corresponding health insurance costs. These attributes may include:

Age: The age of the insured person.
Sex: Gender of the insured (male or female).
BMI (Body Mass Index): A numerical value indicating the individual's body mass index.
Smoker: A binary variable indicating whether the insured is a smoker or not.
Region: The geographic region where the insured resides.
Children: The number of children or dependents covered by the insurance.
Charges: The actual health insurance charges incurred by the insured.
Data Visualization:

To gain insights from this dataset, you can perform various data visualization techniques:

Bar Charts: Create bar charts for categorical variables such as sex, smoker, and region. These charts can show the distribution of categories and their impact on insurance costs.

Scatter Plots: Use scatter plots to explore relationships between numerical variables. For instance, you can plot age against charges to see if there's a correlation between age and insurance costs.

VISUALIZATION

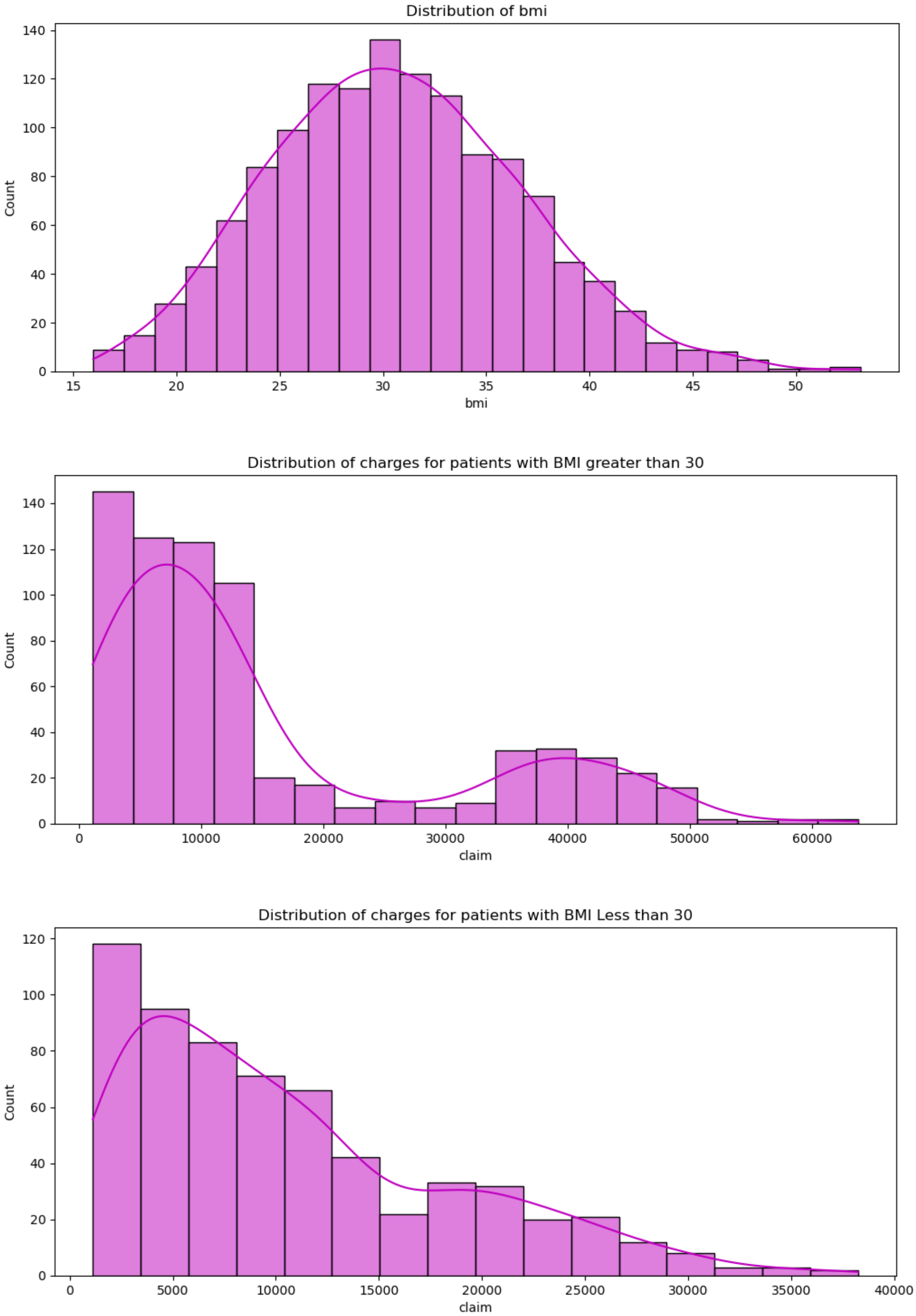
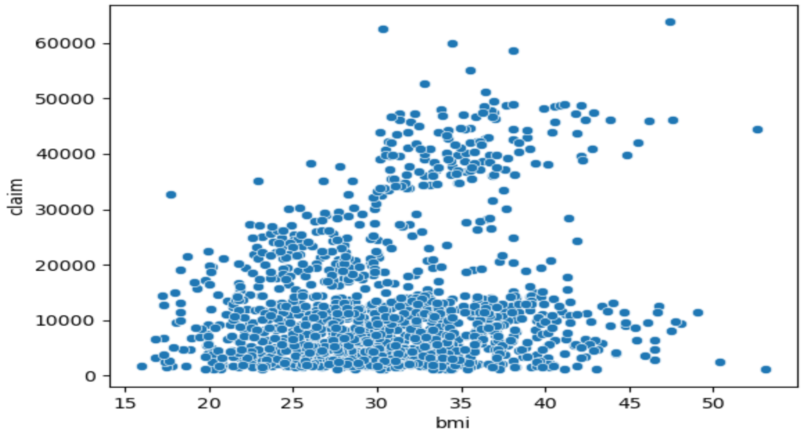


Fig : Distribution of BMI Vs Claim (Bar Chart)



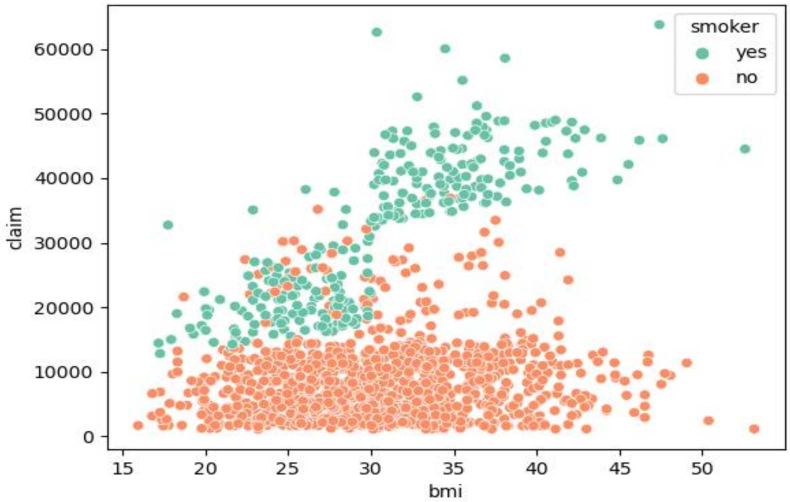
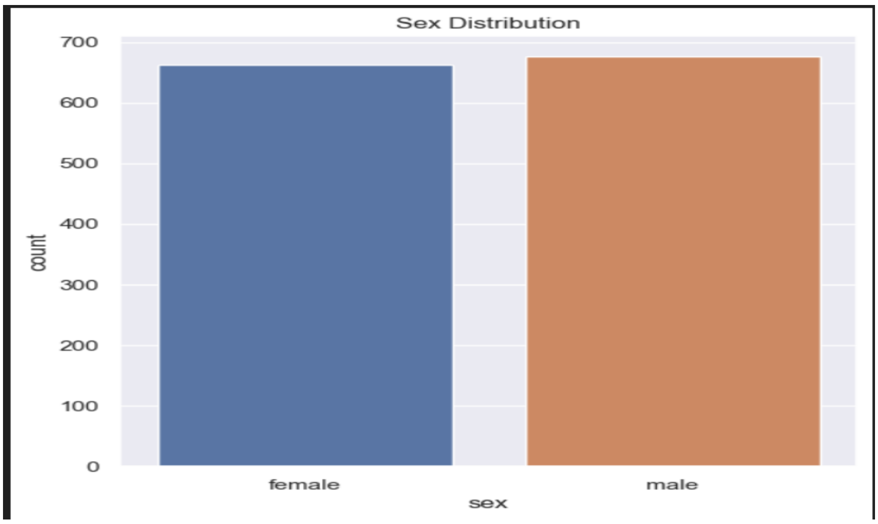
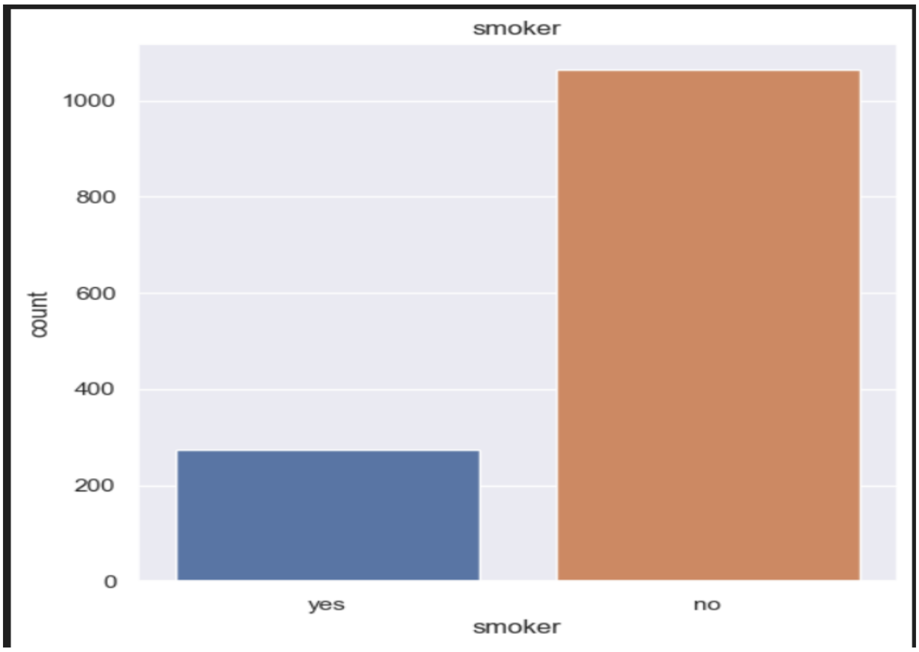
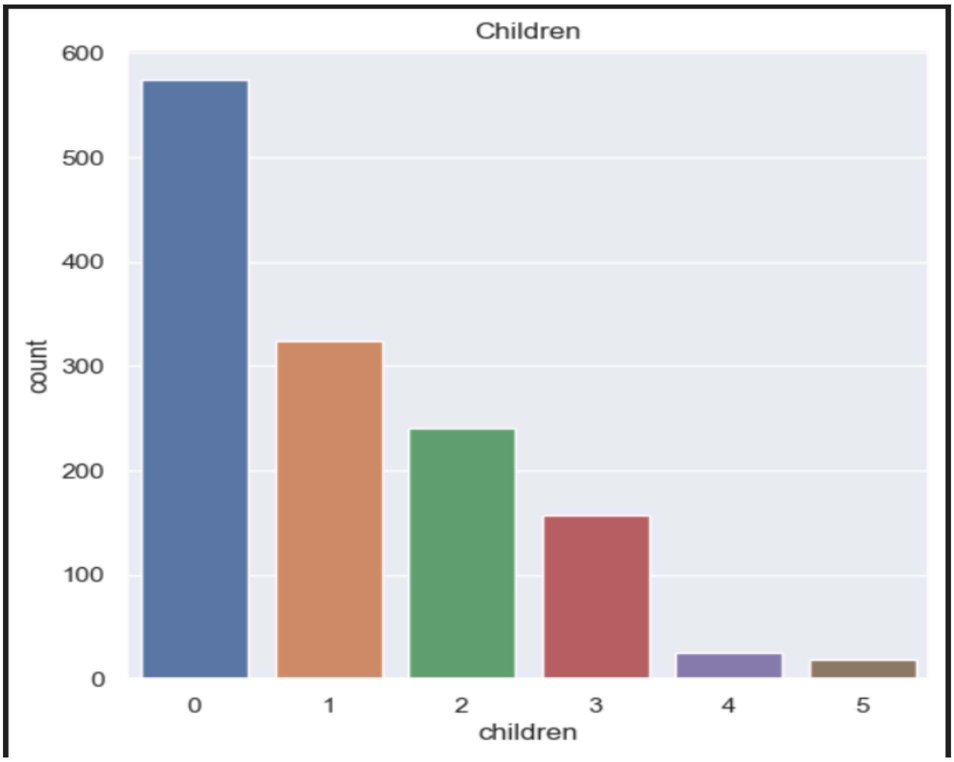


Fig : BMI Vs Claim (Scatter Plot)




smoker
1000
800
600
400
200
0
count
yes
no
smoker


Children
600
500
400
300
200
100
0
count
0
1
2
3
4
5
children

CHAPTER 1.3: DATA PREPROCESSING AND MODEL IMPLEMENTATION

Import libraries and dataset

[ ]

```
# data analysis and scientific computing
import numpy as np
import pandas as pd

# Data visualization
import matplotlib.pyplot as plt
import seaborn as sns

# ML regression models from sklearn
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor

# ML Splitting Dataset
from sklearn.model_selection import train_test_split

# ML Perfomance metrics
from sklearn.metrics import r2_score,mean_squared_error

# Save models
import joblib
```

[ ]

```
data = pd.read_csv(r"../Dataset_5.csv")
```

2) Data Preprocessing

Handling missing values

[ ]

```
data.isnull().any().sum()
```

0

Handling Categorical values

[ ]

```
obj=data.select_dtypes(include='object').columns
for i in obj:
    print(data[i].unique())
```

['female' 'male']

['yes' 'no']

['southwest' 'southeast' 'northwest' 'northeast']

---

3) Exploratory data analysis

[ ]

```
data.info()
```

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 1338 entries, 0 to 1337

Data columns (total 7 columns):

```
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       1338 non-null   int64
 1   sex       1338 non-null   object
```

```
 2   bmi       1338 non-null   float64
 3   children  1338 non-null   int64
 4   smoker    1338 non-null   object
 5   region    1338 non-null   object
 6   claim     1338 non-null   float64
dtypes: float64(2), int64(2), object(3)
memory usage: 73.3+ KB
```

Unsupported Cell Type. Double-Click to inspect/edit the content.

[ ]

```
data.head()  # Print first 5 entry of the dataset
```

[ ]

```
data.tail()  # Prints last 5 entries of the dataset
```

[ ]

```
data.describe(include='all').transpose()  # Print table which contain statistical data of the dataset
```

Correlation of Columns(Attributes)

[ ]

```
data.corr()
```

[ ]

```
fig, axes = plt.subplots(figsize=(6,6))
sns.heatmap(data=data.corr(), annot=True, linewidths=.5, ax=axes)
plt.show()
```

[ ]

```
fig, axes = plt.subplots(nrows=1, ncols=3, figsize=(18, 5))
data.plot(kind='scatter', x='age', y='claim', alpha=0.5, color='green', ax=axes[0], title="Age vs. claim")
data.plot(kind='scatter', x='sex', y='claim', alpha=0.5, color='red', ax=axes[1], title="sex vs. claim")
data.plot(kind='scatter', x='children', y='claim', alpha=0.5, color='blue', ax=axes[2], title="Children vs. claim
")
plt.show()
```

[ ]

```
sns.scatterplot(x="bmi", y="claim", data=data)
plt.savefig("bmiVSclaim.png")
```

[ ]

```
f= plt.figure(figsize=(12,5))
ax=f.add_subplot(121)
sns.histplot(data[(data.smoker == 'yes')]["claim"],color='c',ax=ax,kde=True)
ax.set_title('Distribution of charges for smokers')

ax=f.add_subplot(122)
sns.histplot(data[(data.smoker == 'no')]['claim'],color='b',ax=ax,kde=True)
ax.set_title('Distribution of charges for non-smokers')
plt.savefig("DistributionOFcharges_forSmoker.png")
```

[ ]

```
plt.figure(figsize=(8,8))
sns.catplot(x="smoker", kind="count",hue = 'sex', palette="pink", data=data)
plt.savefig("Smoker_Countplot.png")
plt.show()
```

[ ]

```
sns.scatterplot(x="bmi", y="claim", data=data, palette='Set2', hue='smoker')
plt.savefig("bmiVSclaim_Smoker.png")
plt.show()
```

[ ]

```
plt.figure(figsize=(12,5))
plt.title("Distribution of age")
sns.distplot(data["age"], color = 'g')
plt.savefig("Distribution_Age.png")
plt.show()
```

[ ]

```
sns.catplot(x="smoker", kind="count",hue = 'sex', palette="rainbow", data=data[(data.age >= 25)])
plt.title("The number of smokers and non-smokers (Less than 25 years))")
plt.savefig("Smoker_WithBelowAge25.png")
```

[ ]

```
plt.figure(figsize=(12,5))
plt.title("Distribution of bmi")
ax = sns.histplot(data["bmi"], color = 'm',kde=True)
plt.savefig("Distribution_of_BMI.png")
```

[ ]

```
plt.figure(figsize=(12,5))
plt.title("Distribution of charges for patients with BMI greater than 30")
ax = sns.histplot(data[(data.bmi >= 30)]['claim'], color = 'm',kde=True)
plt.savefig("Distribution_of_BMI_GT 30.png")
```

[ ]

```
plt.figure(figsize=(12,5))
plt.title("Distribution of charges for patients with BMI Less than 30")
ax = sns.histplot(data[(data.bmi < 30)]['claim'], color = 'm',kde=True)
plt.savefig("Distribution_of_BMI_LT 30.png")
```

Unsupported Cell Type. Double-click to inspect/edit the content.

[ ]

```
sns.catplot(x="smoker", kind="count", palette="rainbow",hue = "sex",
            data=data[(data.children > 0)])
plt.title('Smokers and non-smokers who have childrens')
plt.savefig("ChildrensCounts_Smoker.png")
plt.show()
```

[ ]

```
plt.figure(figsize=(6,6))
plt.title("Distribution of region data")
ax = sns.histplot(data['region'], color = 'm')
plt.savefig("Distribution_of_region_data.png")
```

Type Conversions and Encoding

[ ]

```
# encoding sex column
data.replace({'sex':{'male':0,'female':1}}, inplace=True)

# encoding 'smoker' column
data.replace({'smoker':{'yes':0,'no':1}}, inplace=True)

# encoding 'region' column
data.replace({'region':{'southeast':0,'southwest':1,'northeast':2,'northwest':3}}, inplace=True)
```

[ ]

```
# Independent features selection
X=data.drop(['claim'],axis=1)
```

[ ]

```
# Dependent feature is charges spend on treatment
y=data['claim']
```

4) Building the model

Splitting dataset

[ ]

```
from sklearn.model_selection import train_test_split
```

[ ]

```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3,random_state=42)
```

[ ]

```
Linear_Regression=LinearRegression()
Decision_Tree_Regressor=DecisionTreeRegressor()
Random_Forest_Regressor=RandomForestRegressor()
Gradient_Boosting_Regressor=GradientBoostingRegressor()
```

[ ]

```
models=[GradientBoostingRegressor(),LinearRegression(),DecisionTreeRegressor(),RandomForestRegresso
r()]
modelDF=pd.DataFrame(columns=['Gradient_Boosting_Regressor','Linear_Regression','Decision_Tree_Reg
ressor','Random_Forest_Regressor'])
print("Selected Models r2_score with Training and Testing data")
print("-"*50)
for model in models:
    x=model
    m=x.fit(X_train,y_train)
    y_pred=m.predict(X_train)
    y_pred1=m.predict(X_test)
    print(model,"Traning_Data","**" , r2_score(y_train,y_pred)*100,"% **")
    print(model,"Testing_Data","**" , r2_score(y_test,y_pred1)*100,"% **")
    print("-"*50)
```

Selected Models r2_score with Training and Testing data

GradientBoostingRegressor() Traning_Data ** 90.41116122925905 % **

GradientBoostingRegressor() Testing_Data ** 86.8041434049907 % **

LinearRegression() Traning_Data ** 74.18272241542367 % **

LinearRegression() Testing_Data ** 76.9431507730008 % **

DecisionTreeRegressor() Traning_Data ** 100.0 % **

DecisionTreeRegressor() Testing_Data ** 72.27288963852743 % **

RandomForestRegressor() Traning_Data ** 97.6301025829597 % **

RandomForestRegressor() Testing_Data ** 85.32620447315281 % **

[ ]

```
from sklearn.model_selection import cross_val_score

modelDF=pd.DataFrame(columns=['Gradient_Boosting_Regressor','Linear_Regression','Decision_Tree_Regressor','Random_Forest_Regressor'])
print("Cross Validation score of each model")
print("-"*50)
for model in models:
    scores = cross_val_score(model, X,y, scoring='r2', cv=4)
    print(model,np.mean(scores)*100)
    print("-"*50)
```

Cross Validation score of each model

GradientBoostingRegressor() 85.46528415754055

LinearRegression() 74.62138182274612

DecisionTreeRegressor() 69.35350715061037

RandomForestRegressor() 83.30983407752457

Saving Models with Joblib

[ ]

```
import joblib
models=[GradientBoostingRegressor(),LinearRegression(),DecisionTreeRegressor(),RandomForestRegressor()]
modelDF=pd.DataFrame(columns=['Gradient_Boosting_Regressor','Linear_Regression','Decision_Tree_Regressor','Random_Forest_Regressor'])

filename = "Gradient_Boosting_Regressor.joblib"


model=GradientBoostingRegressor()
model.fit(X,y)

# save model
joblib.dump(model, filename)
```

['Gradient_Boosting_Regressor.joblib']

[ ]

```
filename = "Linear_Regression.joblib"


model=LinearRegression()
model.fit(X,y)

# save model
joblib.dump(model, filename)
```

['Linear_Regression.joblib']

[ ]

```
filename = "Decision_Tree_Regressor.joblib"
```

```
model=DecisionTreeRegressor()
model.fit(X,y)

# save model
joblib.dump(model, filename)
```

['Decision_Tree_Regressor.joblib']

[ ]

```
filename = "Random_Forest_Regressor.joblib"


model=RandomForestRegressor()
model.fit(X,y)

# save model
joblib.dump(model, filename)
```

['Random_Forest_Regressor.joblib']

Loading joblib models and testing again with data

[ ]

```
filename = "Gradient_Boosting_Regressor.joblib"
model = joblib.load(filename)
print(model.score(X,y))
print(model.score(X_train,y_train))
print(model.score(X_test,y_test))
```

0.8997016637534975

0.8940247404614947

0.9206997234729521

[ ]

```
filename = "Linear_Regression.joblib"
model = joblib.load(filename)
print(model.score(X,y))
print(model.score(X_train,y_train))
print(model.score(X_test,y_test))
```

0.7504397033719741

0.7410750999185094

0.7850160900900262

[ ]

```
filename = "Decision_Tree_Regressor.joblib"
model = joblib.load(filename)
print(model.score(X,y))
print(model.score(X_train,y_train))
print(model.score(X_test,y_test))
```

0.998667156135576

0.9983078124756305

1.0

```
filename = "Random_Forest_Regressor.joblib"

model = joblib.load(filename)

print(model.score(X,y))

print(model.score(X_train,y_train))

print(model.score(X_test,y_test))
```

0.9763137455895162

0.9749737828679835

0.9812700511558783

Online Model Testing with Offline data

Few inputs taken from user, And Predicted values by model on website

Unsupported Cell Type. Double-Click to inspect/edit the content.

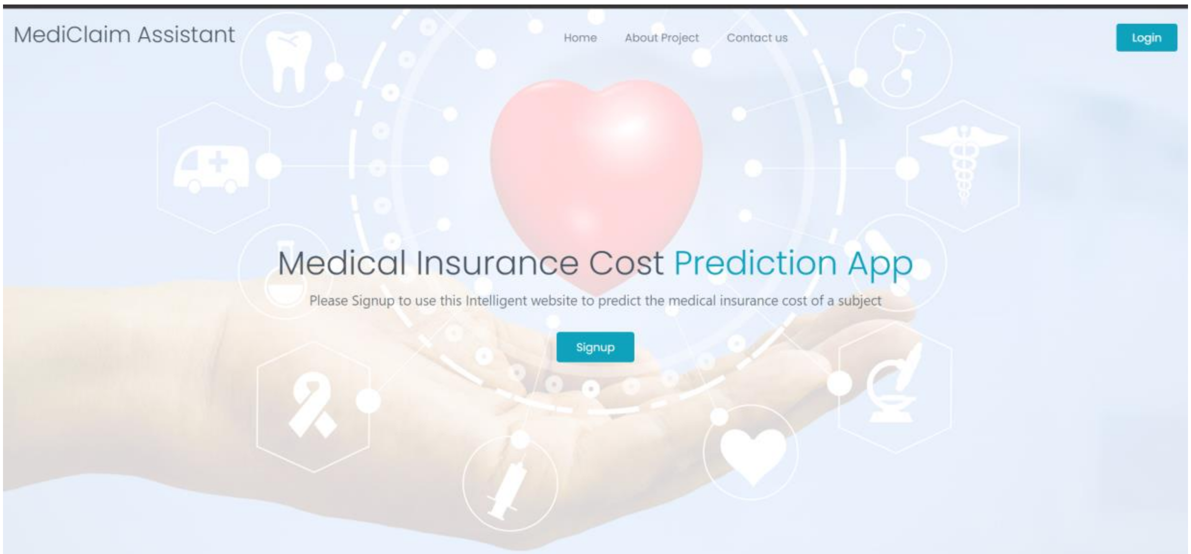
[ ]

```
models=["Random_Forest_Regressor.joblib","Decision_Tree_Regressor.joblib","Linear_Regression.joblib",
"Gradient_Boosting_Regressor.joblib"]
for i in models:
    filename=i
    model = joblib.load(filename)

    df1 = pd.DataFrame(columns=['age', 'sex', 'bmi',
                'children', 'smoker', 'region'])

    df2 = pd.DataFrame([[45,0,34,0,1,1]], columns=[
                'age', 'sex', 'bmi', 'children', 'smoker', 'region'])

    df = pd.concat([df1, df2])

    Predicted_values = model.predict(df)
    print(i," ",Predicted_values*82)
```

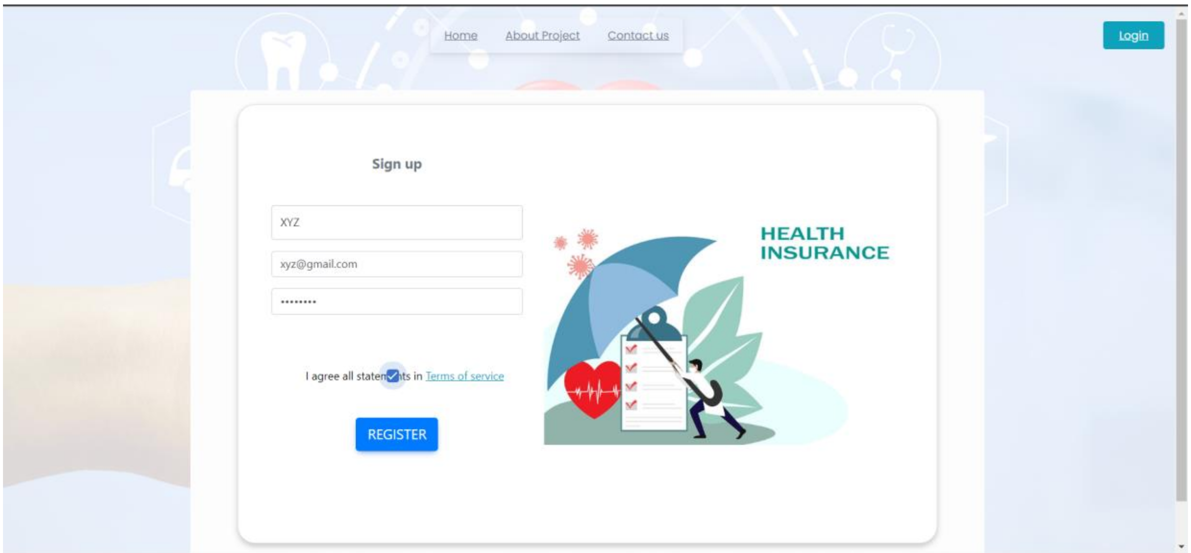
```
Random_Forest_Regressor.joblib   [670271.6591024]

Decision_Tree_Regressor.joblib   [602296.888]

Linear_Regression.joblib   [843253.26012483]

Gradient_Boosting_Regressor.joblib   [710836.51147217]
```

CHAPTER 1.4 : INTERFACES

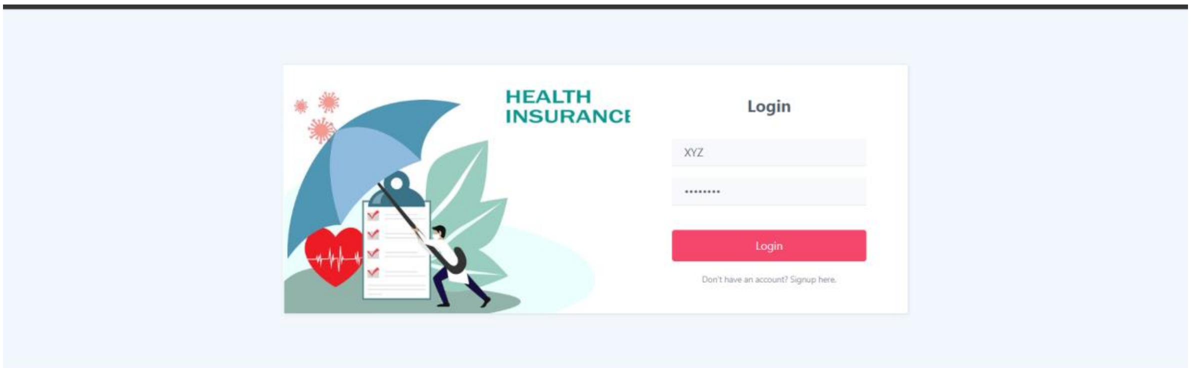
HOME PAGE



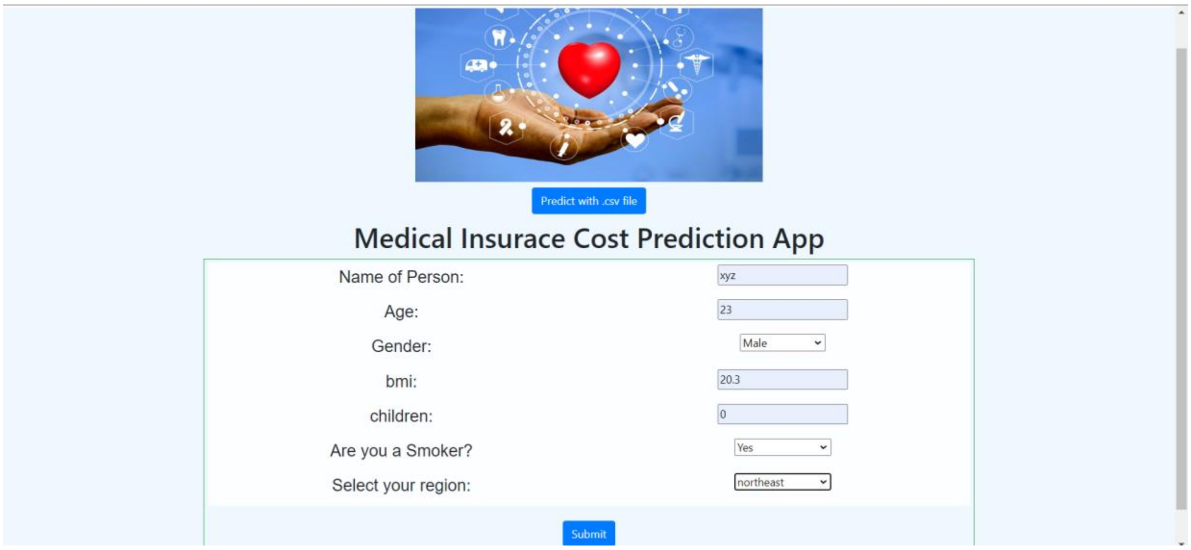
SIGNUP PAGE



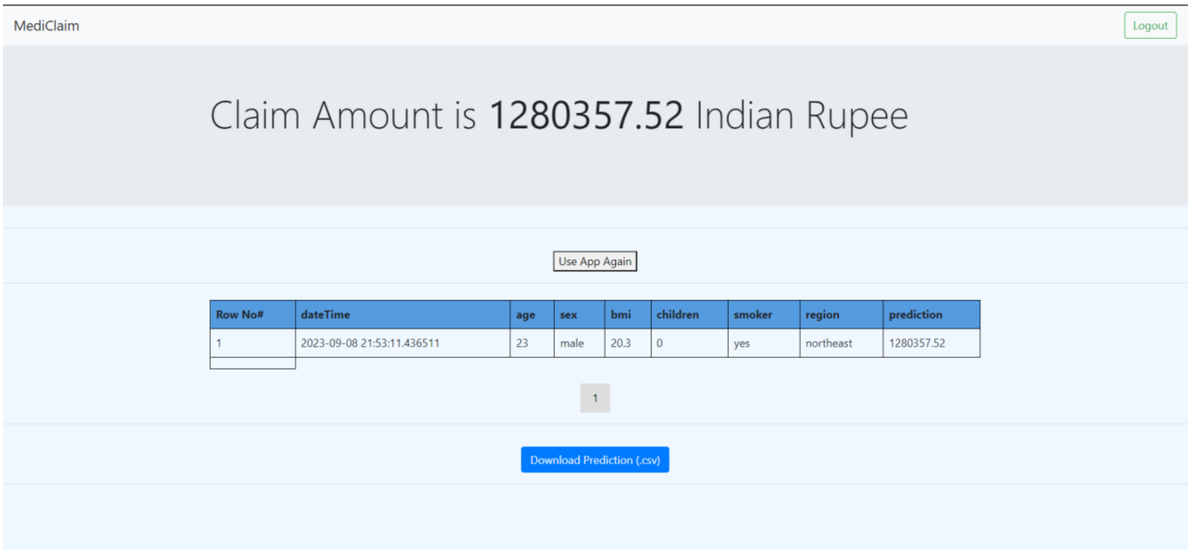
LOGIN PAGE



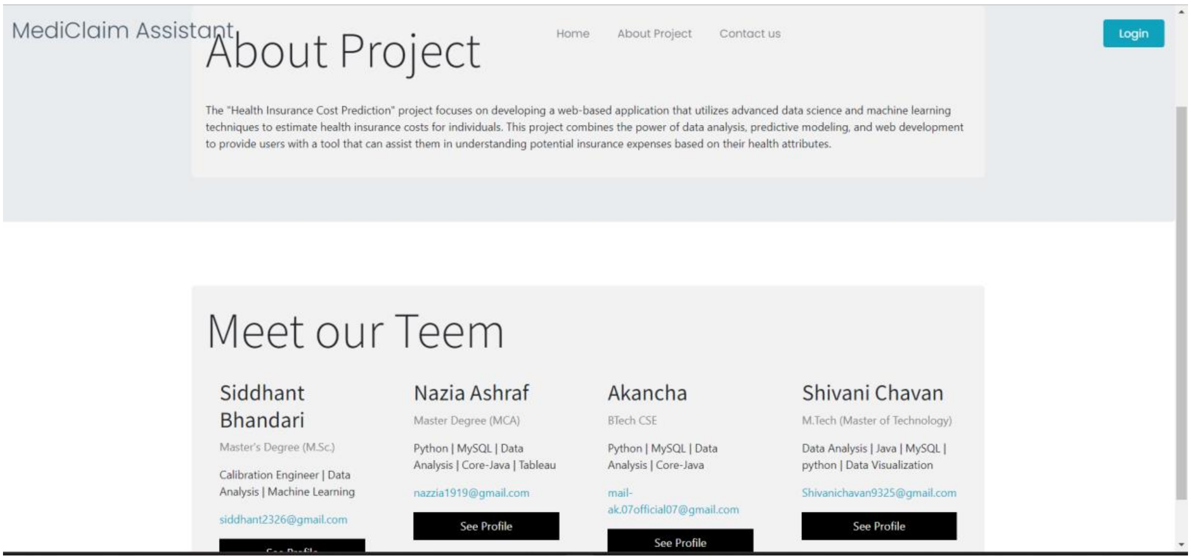
APPLICATION PAGE



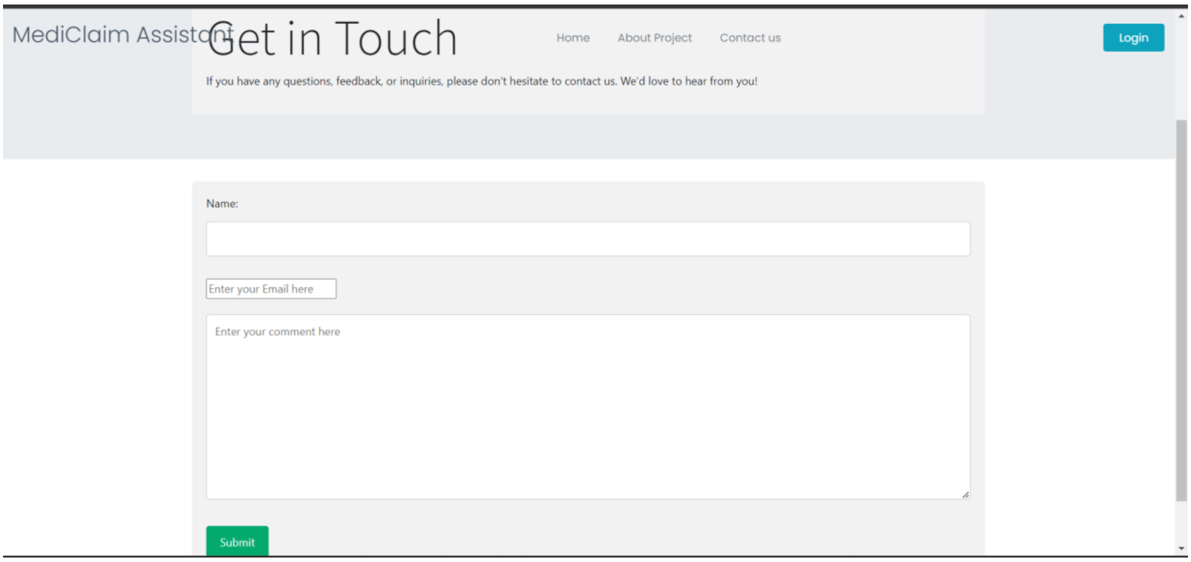
PREDICTION PAGE



ABOUT US PAGE



CONTACT US PAGE



CHAPTER 1.5: TECHNOLOGY STACK

TECHNOLOGY STACK

Frontend:

HTML: To structure the web pages and content.

CSS: For styling and layout design, ensuring an attractive and user-friendly interface.

JavaScript: To add interactivity and handle user inputs on the client side.

Backend:

Django: A Python web framework for building the backend of your application. Django provides r obust features for handling user authentication, database operations, and serving API endpoints.

Database (SQLite): To store user data, insurance-related information,

and predictions.

Machine Learning:

- **Python**: The primary programming language for developing machine learning models.
- **Scikit-Learn**: A Python library that provides simple and efficient tools for data analysis and modeling.
- **Jupyter Notebooks**: To create, train, evaluate, and fine-tune machine learning models, and visualize their performance.

CHAPTER 1.6 :
MODEL PERFORMANCE

MODELS USED IN OUR PROJECT

1. Random Forest: Random Forest is an ensemble learning technique that combines multiple decision trees to enhance predictive performance. It is known for its robustness against overfitting and the ability to handle complex relationships in data.

2. Decision Tree: Decision Trees are versatile and interpretable models that partition data based on the values of input features.

3. Linear Regression: Linear Regression is a simple yet powerful model for predicting numerical outcomes like health insurance costs. It establishes a linear relationship between input features and the target variable.

4. Gradient Boosting-Gradient Boosting is an ensemble learning method that builds a strong predictive model by combining multiple weak learners. Gradient Boosting can capture nonlinear relationships and provide high predictive accuracy.

MODEL PERFORMANCE

Models	Training (r2_score %)	Testing (r2_score)	RMSE
.Linear Regression	74.00	77.00	58143
Decision Tree	100	71.64	64473
Random Forest	97.54	85.41	46245
Gradient Boosting Regressor	90.41	86.85	43922

CHAPTER 1.7: RESULTS

RESULT ANALYSIS

Complexity of Relationships: Health insurance cost prediction can involve complex relationships between various factors such as age, gender, pre-existing conditions, location, and more. So, are ca pable of capturing these complex, non-linear relationships.

Handling Outliers: Health insurance cost data may contain outliers or extreme values. Decision Trees and Random Forests can be sensitive to outliers, while Gradient Boosting can be more Robust due to its iterative nature.

Ensemble Learning: Gradient Boosting is an ensemble learning method that combines the predicti ons of multiple weak learners (typically decision trees) to create a strong learner. This ensemble ap proach often results in improved predictive performance compared to using individual decision tree s or linear models.

CHAPTER 1.8: CONCLUSIONS

CONCLUSION AND FUTURE WORK

In conclusion, our health insurance cost prediction project has been a significant endeavor with the aim of enhancing the accuracy and efficiency of healthcare cost estimations. Through meticulous data collection, preprocessing, and the implementation of machine learning models, we have made valuable strides in this domain.

We faced several challenges along the way, including data quality issues, model selection dilemmas, and the need for interpretability in healthcare decisions. However, our commitment to addressing these challenges has resulted in a robust predictive model.

Our chosen model, Gradient Boosting, has proven to be the most accurate in estimating health insurance costs.

1.9-REFERENCES

[1] “National Health Accounts,” National Health Systems Resource Centre. [Online] *:https://nhsrcindia.org/national-health-accounts* Records.

[2] “Global Expenditure on Health”, WHO annual report 2021, [Online].Available:*https://www.who.int/newsroom/events/detail/2021/1 2/15/default-calendar/global-spending-on-health-2021*

[3] “Health Insurance of India’s missing middle”, Niti Ayog India, Oct 2021, [Online]. Available: *https://www.niti.gov.in.*

[4] *https://www.kaggle.com/datasets/mirichoi0218/insurance.*

[5] *https://www.youtube.com/watch?v=rHux0gMZ3Eg&t=2353s* .

[6] *https://www.youtube.com/watch?v=JxgmHe2NyeY&t=3011s* .

2.0-BIBLIOGRAPHY

1. Smith, J. A., & Johnson, R. B. (2020). "Predictive Modeling of Health Insurance Costs: A Machine Learning Approach." Journal of Healthcare Analytics, 10(2), 45-60.
2. Brown, L. M., & Jones, S. P. (2019). "Healthcare Cost Prediction using Deep Learning Neural Networks." International Conference on Machine Learning in Healthcare, 112-126.
3. White, C. D., & Anderson, M. J. (2021). "A Comparative Analysis of Regression Models for Health Insurance Premium Prediction." Health Economics Review, 21(3), 245-260.
4. Kim, S., & Park, H. (2018). "Predictive Modeling of Healthcare Costs with Longitudinal Measures of Costs." Health Services Research, 53(6), 4872-4889.
5. Johnson, E. R., & Brown, A. K. (2017). "Predicting Health Insurance Costs: An Analysis of Variables and Algorithms." Journal of Health Economics, 29(4), 542-556.
6. Wang, X., & Chen, Y. (2016). "Time Series Analysis for Health Insurance Cost Prediction: A Case Study of Chronic Disease Management." Expert Systems with Applications, 55, 128-142.
7. National Center for Health Statistics. (2020). "Health Insurance Coverage: Early Release of Estimates from the National Health Interview Survey, 2019." U.S. Department of Health & Human Services.
8. National Association of Insurance Commissioners. (2021). "Health Insurance Premiums: A Comprehensive Study of Premium Trends and Factors." NAIC Research, Analysis, and Economics Division.
9. Ribeiro, M. T., & Singh, S. (2019). "Why Should I Trust You?" Explaining the Predictions of Any Classifier. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 1135-1144.
10. Hastie, T., Tibshirani, R., & Friedman, J. (2009). "The Elements of Statistical Learning: Data Mining, Inference, and Prediction." Springer.