University of Petroleum and Energy Studies

BATCH 2025 – 2027

Department : School of Computer Science

PROJECT

ON

(Linear Regression)

15 – Insurance Prediction

Submitted By :

Vivek Rawat & Vivek Patel

Submitted to :

Asst. Prof. Dhiviya Rose

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
from sklearn.preprocessing import StandardScaler

df = pd.read_csv("insurance.csv")
print("Initial shape:", df.shape)
print("\nMissing values:\n", df.isnull().sum())
dups = df.duplicated().sum()
print("\nDuplicates found:", dups)
if dups > 0:
    df = df.drop_duplicates()
    print("New shape after removing duplicates:", df.shape)

Q1 = df['expenses'].quantile(0.25)
Q3 = df['expenses'].quantile(0.75)
IQR = Q3 - Q1
lower = Q1 - 1.5 * IQR
upper = Q3 + 1.5 * IQR
df = df[(df['expenses'] >= lower) & (df['expenses'] <= upper)]
print("Shape after removing extreme outliers:", df.shape)

df['bmi'] = df['bmi'].clip(lower=15, upper=45)

num_cols = ['age', 'bmi', 'children', 'expenses']
```

```
for col in num_cols:
    plt.figure(figsize=(6,3))
    plt.hist(df[col], bins=30)
    plt.title(f"Histogram of {col}")
    plt.xlabel(col)
    plt.ylabel("Count")
    plt.show()
for col in num_cols:
    plt.figure(figsize=(6,3))
    plt.boxplot(df[col], vert=False)
    plt.title(f"Boxplot of {col}")
    plt.show()

print("\nMean expenses by smoker:\n", df.groupby('smoker')['expenses'].mean())
print("\nMean expenses by sex:\n", df.groupby('sex')['expenses'].mean())
print("\nMean expenses by region:\n", df.groupby('region')['expenses'].mean())
df['sex'] = df['sex'].map({'male':0, 'female':1})
df['smoker'] = df['smoker'].map({'no':0, 'yes':1})
df = pd.get_dummies(df, columns=['region'], drop_first=True)
X = df.drop('expenses', axis=1)
y = df['expenses']

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)
scaler = StandardScaler()
num_feats = ['age','bmi','children']
X_train_scaled = X_train.copy()
X_test_scaled = X_test.copy()
```

```
X_train_scaled[num_feats] = scaler.fit_transform(X_train[num_feats])
X_test_scaled[num_feats] = scaler.transform(X_test[num_feats])
lr = LinearRegression()
lr.fit(X_train_scaled, y_train)
y_pred_lr = lr.predict(X_test_scaled)

print("\n--- Linear Regression Results ---")
print("MSE:", mean_squared_error(y_test, y_pred_lr))
print("MAE:", mean_absolute_error(y_test, y_pred_lr))
print("R² :", r2_score(y_test, y_pred_lr))
plt.scatter(y_test, y_pred_lr, s=10)
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()])
plt.xlabel("Actual")
plt.ylabel("Predicted")
plt.title("Linear Regression: Actual vs Predicted")
plt.show()
rf = RandomForestRegressor(n_estimators=200, random_state=42)
rf.fit(X_train, y_train)
y_pred_rf = rf.predict(X_test)

print("\n--- Random Forest Results ---")
print("MSE:", mean_squared_error(y_test, y_pred_rf))
print("MAE:", mean_absolute_error(y_test, y_pred_rf))
print("R² :", r2_score(y_test, y_pred_rf))

plt.scatter(y_test, y_pred_rf, s=10)
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()])
plt.xlabel("Actual")
plt.ylabel("Predicted")
```

```
plt.title("Random Forest: Actual vs Predicted")

plt.show()
```