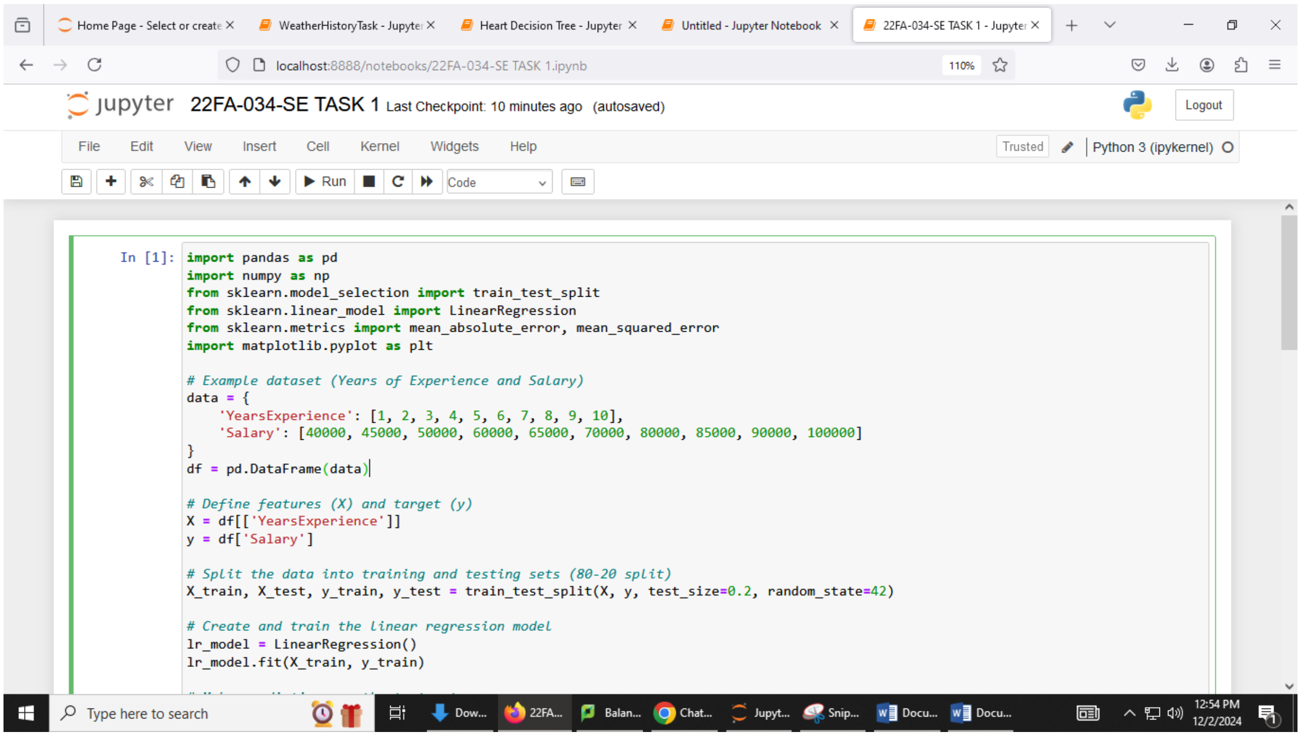
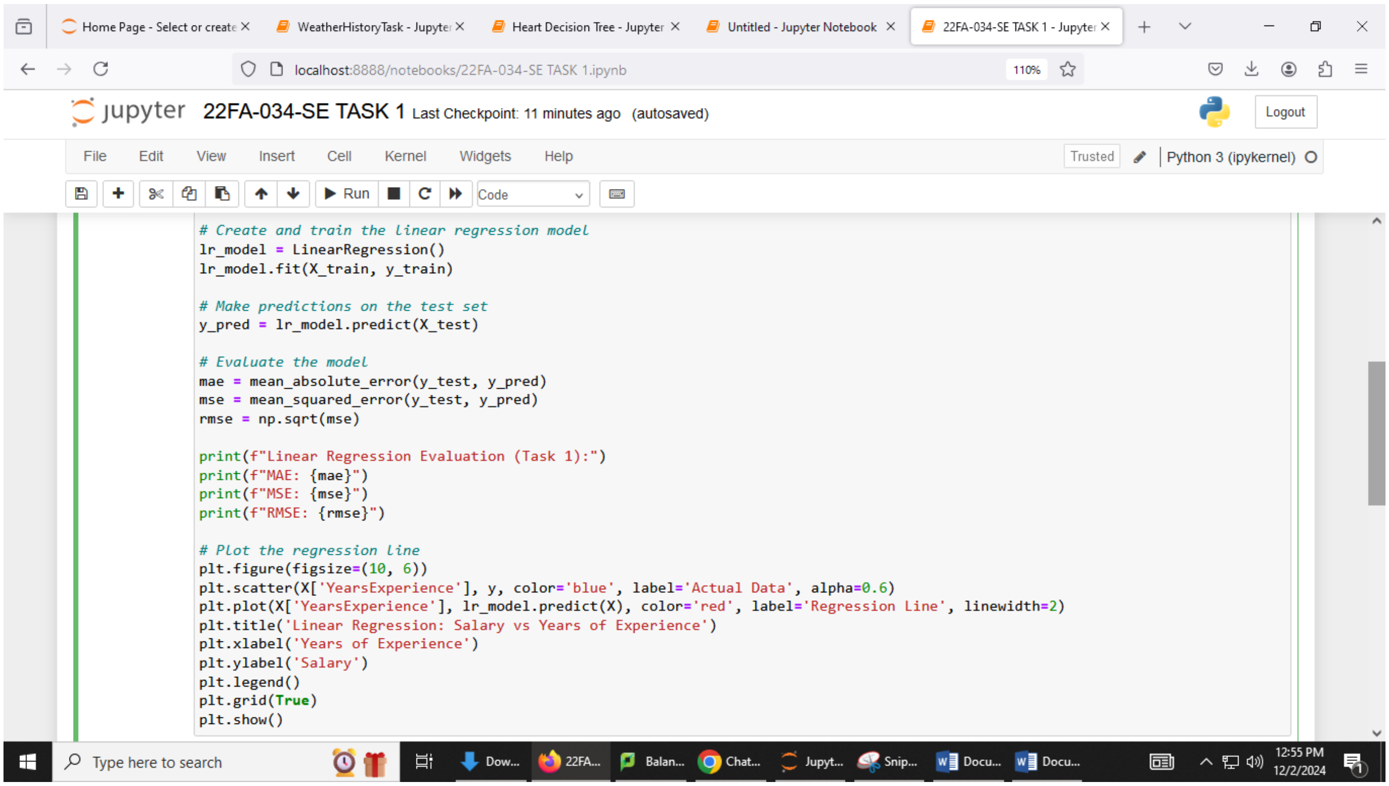
TASK 1 :

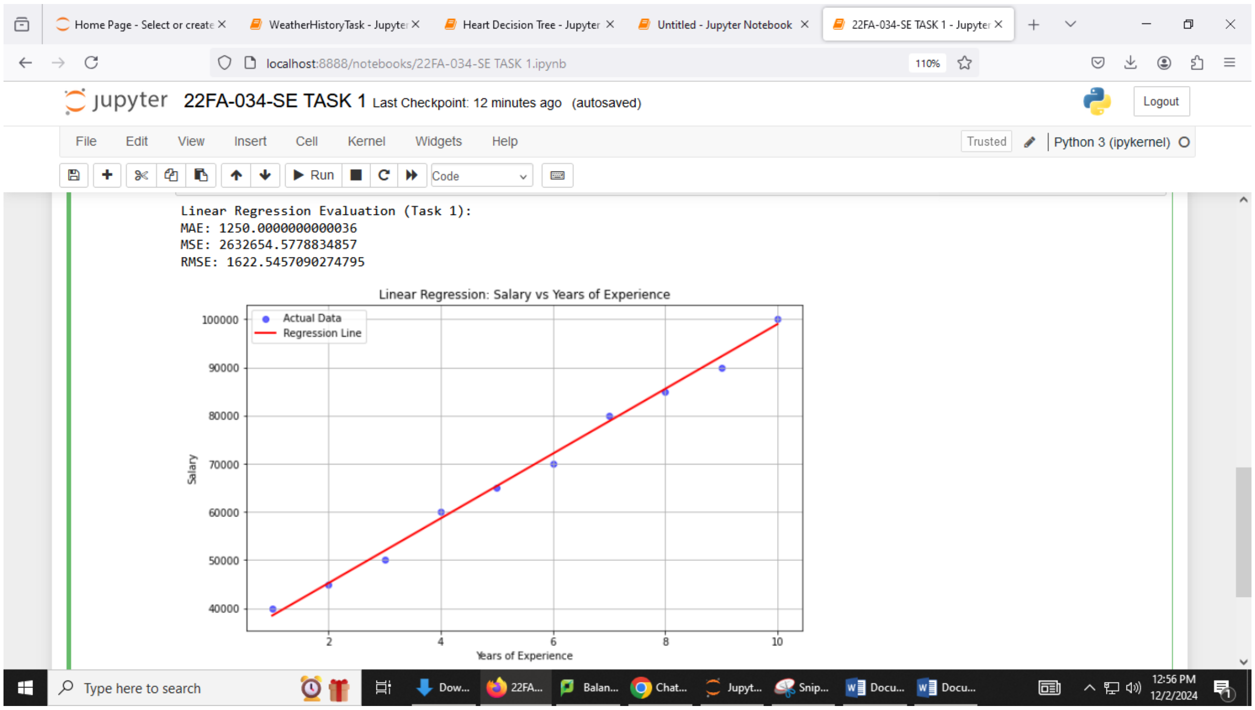

Home Page - Select or create
WeatherHistoryTask - Jupyter
Heart Decision Tree - Jupyter
Untitled - Jupyter Notebook
22FA-034-SE TASK 1 - Jupyter
localhost:8888/notebooks/22FA-034-SE TASK 1.ipynb
110%
jupyter 22FA-034-SE TASK 1 Last Checkpoint: 10 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code
Type here to search
12:54 PM
12/2/2024

```
In [1]: import pandas as pd
        import numpy as np
        from sklearn.model_selection import train_test_split
        from sklearn.linear_model import LinearRegression
        from sklearn.metrics import mean_absolute_error, mean_squared_error
        import matplotlib.pyplot as plt

        # Example dataset (Years of Experience and Salary)
        data = {
            'YearsExperience': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
            'Salary': [40000, 45000, 50000, 60000, 65000, 70000, 80000, 85000, 90000, 100000]
        }
        df = pd.DataFrame(data)

        # Define features (X) and target (y)
        X = df[['YearsExperience']]
        y = df['Salary']

        # Split the data into training and testing sets (80-20 split)
        X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

        # Create and train the linear regression model
        lr_model = LinearRegression()
        lr_model.fit(X_train, y_train)
```


Home Page - Select or create
WeatherHistoryTask - Jupyter
Heart Decision Tree - Jupyter
Untitled - Jupyter Notebook
22FA-034-SE TASK 1 - Jupyter
localhost:8888/notebooks/22FA-034-SE TASK 1.ipynb
110%
jupyter 22FA-034-SE TASK 1 Last Checkpoint: 11 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code
Type here to search
12:55 PM
12/2/2024

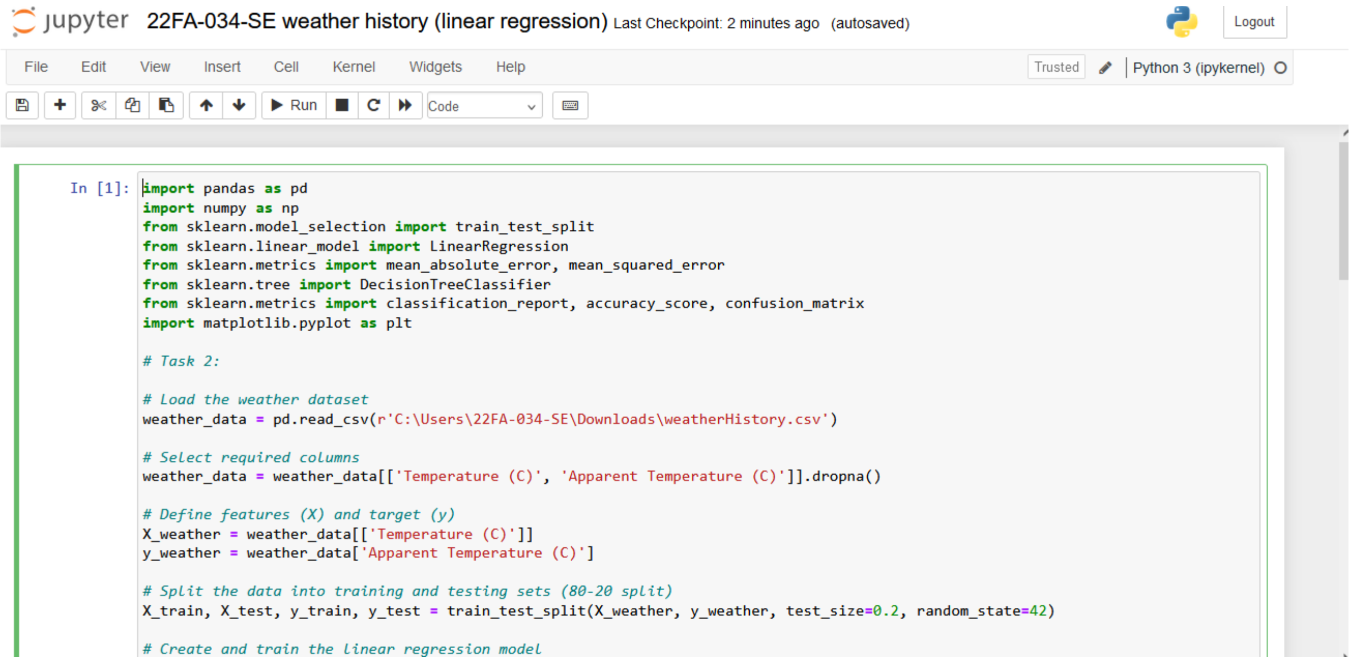
```
# Create and train the linear regression model
lr_model = LinearRegression()
lr_model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = lr_model.predict(X_test)

# Evaluate the model
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)

print(f"Linear Regression Evaluation (Task 1):")
print(f"MAE: {mae}")
print(f"MSE: {mse}")
print(f"RMSE: {rmse}")

# Plot the regression line
plt.figure(figsize=(10, 6))
plt.scatter(X['YearsExperience'], y, color='blue', label='Actual Data', alpha=0.6)
plt.plot(X['YearsExperience'], lr_model.predict(X), color='red', label='Regression Line', linewidth=2)
plt.title('Linear Regression: Salary vs Years of Experience')
plt.xlabel('Years of Experience')
plt.ylabel('Salary')
plt.legend()
plt.grid(True)
plt.show()
```

OUTPUT:

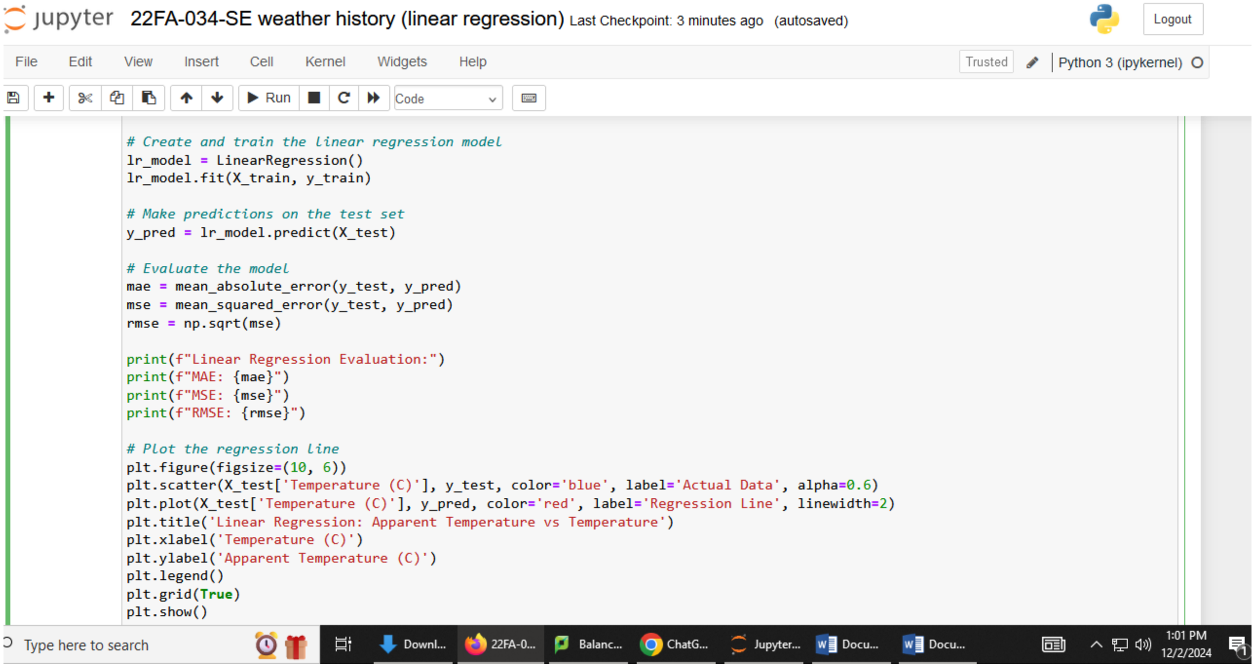

Home Page - Select or create
WeatherHistoryTask - Jupyter
Heart Decision Tree - Jupyter
Untitled - Jupyter Notebook
22FA-034-SE TASK 1 - Jupyter
localhost:8888/notebooks/22FA-034-SE TASK 1.ipynb
110%
jupyter 22FA-034-SE TASK 1 Last Checkpoint: 12 minutes ago (autosaved)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3 (ipykernel)
Run
Code
Linear Regression: Salary vs Years of Experience
Actual Data
Regression Line
100000
90000
80000
70000
60000
50000
40000
Salary
2
4
6
8
10
Years of Experience
Type here to search
Dow...
22FA...
Balan...
Chat...
Jupyt...
Snip...
Docu...
Docu...
12:56 PM
12/2/2024

```
Linear Regression Evaluation (Task 1):
MAE: 1250.0000000000036
MSE: 2632654.5778834857
RMSE: 1622.5457090274795
```

TASK 2:

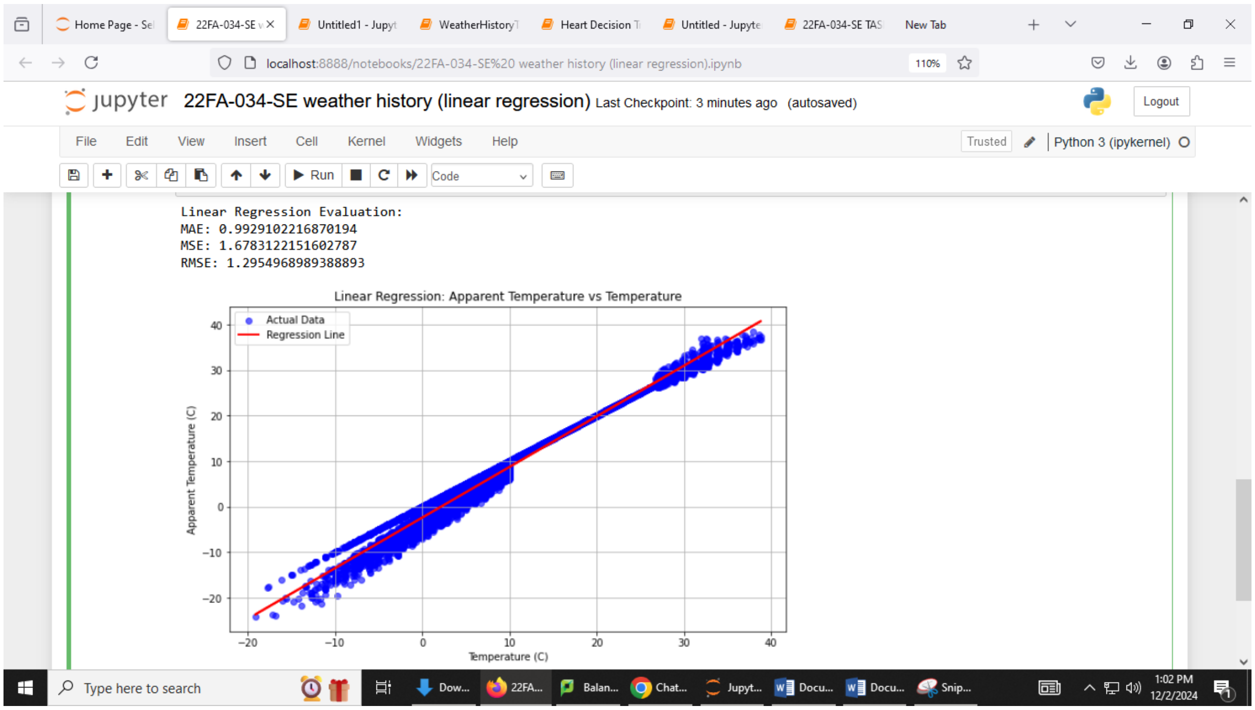

jupyter 22FA-034-SE weather history (linear regression) Last Checkpoint: 2 minutes ago (autosaved)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3 (ipykernel)
Run
Code
In [1]:

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import classification_report, accuracy_score, confusion_matrix
import matplotlib.pyplot as plt

# Task 2:

# Load the weather dataset
weather_data = pd.read_csv(r'C:\Users\22FA-034-SE\Downloads\weatherHistory.csv')

# Select required columns
weather_data = weather_data[['Temperature (C)', 'Apparent Temperature (C)']].dropna()

# Define features (X) and target (y)
X_weather = weather_data[['Temperature (C)']]
y_weather = weather_data['Apparent Temperature (C)']

# Split the data into training and testing sets (80-20 split)
X_train, X_test, y_train, y_test = train_test_split(X_weather, y_weather, test_size=0.2, random_state=42)

# Create and train the linear regression model
```


jupyter
22FA-034-SE weather history (linear regression) Last Checkpoint: 3 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code
Type here to search
1:01 PM
12/2/2024

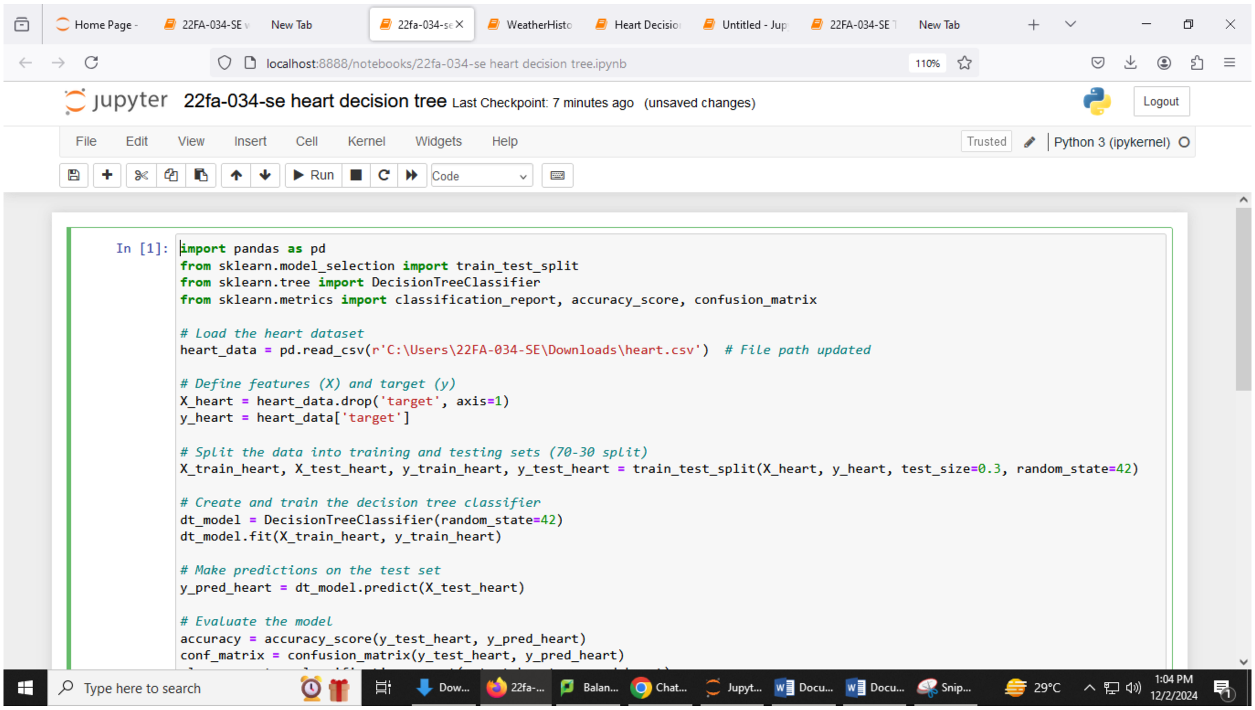
```
# Create and train the linear regression model
lr_model = LinearRegression()
lr_model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = lr_model.predict(X_test)

# Evaluate the model
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)

print(f"Linear Regression Evaluation:")
print(f"MAE: {mae}")
print(f"MSE: {mse}")
print(f"RMSE: {rmse}")

# Plot the regression line
plt.figure(figsize=(10, 6))
plt.scatter(X_test['Temperature (C)'], y_test, color='blue', label='Actual Data', alpha=0.6)
plt.plot(X_test['Temperature (C)'], y_pred, color='red', label='Regression Line', linewidth=2)
plt.title('Linear Regression: Apparent Temperature vs Temperature')
plt.xlabel('Temperature (C)')
plt.ylabel('Apparent Temperature (C)')
plt.legend()
plt.grid(True)
plt.show()
```

Output:

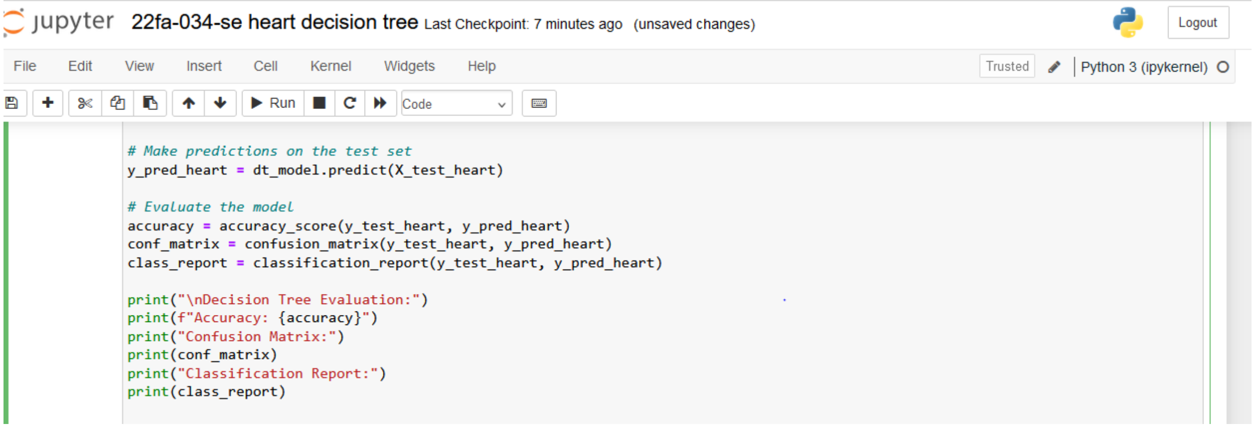


```
Linear Regression Evaluation:
MAE: 0.9929102216870194
MSE: 1.6783122151602787
RMSE: 1.2954968989388893
```

TASK 3 :

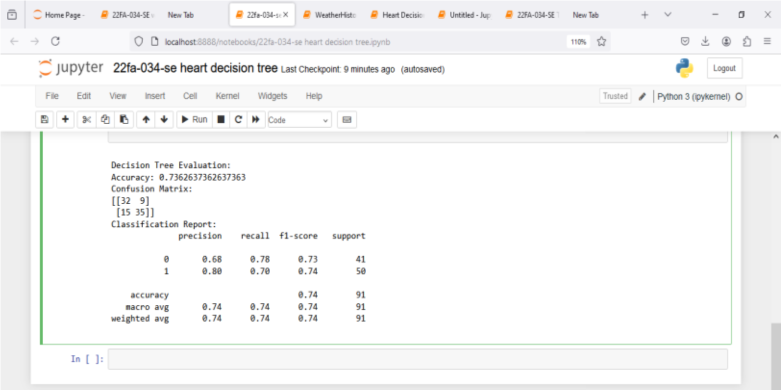


```
In [1]: import pandas as pd
        from sklearn.model_selection import train_test_split
        from sklearn.tree import DecisionTreeClassifier
        from sklearn.metrics import classification_report, accuracy_score, confusion_matrix

        # Load the heart dataset
        heart_data = pd.read_csv(r'C:\Users\22FA-034-SE\Downloads\heart.csv')  # File path updated

        # Define features (X) and target (y)
        X_heart = heart_data.drop('target', axis=1)
        y_heart = heart_data['target']

        # Split the data into training and testing sets (70-30 split)
        X_train_heart, X_test_heart, y_train_heart, y_test_heart = train_test_split(X_heart, y_heart, test_size=0.3, random_state=42)

        # Create and train the decision tree classifier
        dt_model = DecisionTreeClassifier(random_state=42)
        dt_model.fit(X_train_heart, y_train_heart)

        # Make predictions on the test set
        y_pred_heart = dt_model.predict(X_test_heart)

        # Evaluate the model
        accuracy = accuracy_score(y_test_heart, y_pred_heart)
        conf_matrix = confusion_matrix(y_test_heart, y_pred_heart)
```



```
# Make predictions on the test set
y_pred_heart = dt_model.predict(X_test_heart)

# Evaluate the model
accuracy = accuracy_score(y_test_heart, y_pred_heart)
conf_matrix = confusion_matrix(y_test_heart, y_pred_heart)
class_report = classification_report(y_test_heart, y_pred_heart)

print("\nDecision Tree Evaluation:")
print(f"Accuracy: {accuracy}")
print("Confusion Matrix:")
print(conf_matrix)
print("Classification Report:")
print(class_report)
```

OUTPUT:



```
Decision Tree Evaluation:
Accuracy: 0.7362637362637363
Confusion Matrix:
[[32  9]
 [15 35]]
Classification Report:
              precision    recall  f1-score   support

           0       0.68      0.78      0.73        41
           1       0.80      0.70      0.74        50

    accuracy                           0.74        91
   macro avg       0.74      0.74      0.74        91
weighted avg       0.74      0.74      0.74        91
```