Noreen jamil

Roll #102 (5^{th})

Artificial intelligence project

Weather Forecasting Code Documentation

This document explains a Python script designed for weather forecasting. The script uses various libraries to load, explore, visualize, and analyze weather data, and applies machine learning models to predict temperature. Below is a simple, step-by-step explanation of the code and its purpose.

---

1. Import Required Libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn import preprocessing
```

Why These Libraries Are Used:

- **numpy**: For numerical operations and handling arrays.
- **pandas**: For data management and analysis in table format.
- **matplotlib.pyplot**: For creating visualizations (e.g., plots, histograms).
- **sklearn**: For machine learning tasks, including data splitting, model training, and evaluation.

---

2. Load the Dataset

```
weather_df = pd.read_csv('Lahore.csv', parse_dates=['date_time'],
index_col='date_time')
```

Purpose:

Loads weather data from a CSV file and prepares it for analysis.

- **`parse_dates=['date_time']`**: Converts the `date_time` column into a datetime format.
- **`index_col='date_time'`**: Sets `date_time` as the index to enable time-based operations.

---

3. Explore the Dataset

```
weather_df.head(5)
weather_df.columns
weather_df.shape
weather_df.describe()
weather_df.isnull().any()
```

Key Operations:

- **`head(5)`**: Displays the first 5 rows for a quick look.
- **`columns`**: Lists all column names.
- **`shape`**: Shows the dataset size (rows, columns).
- **`describe()`**: Provides statistical summaries for numerical columns.
- **`isnull().any()`**: Checks for missing data in columns.

---

4. Select Numerical Columns

```
weather_df_num = weather_df.loc[:, ['maxtempC', 'mintempC', 'cloudcover',
'humidity', 'tempC',
                                     'sunHour', 'HeatIndexC', 'precipMM',
'pressure', 'windspeedKmph']]
```

Purpose:

Focuses on numerical columns relevant for weather analysis and forecasting.

---

5. Visualize the Data

```
weather_df_num.plot(subplots=True, figsize=(25, 20))
weather_df_num['2019':'2020'].resample('D').fillna(method='pad').plot(subplot
s=True, figsize=(25, 20))
weather_df_num.hist(bins=10, figsize=(15, 15))
```

Key Visualizations:

- **Line plots**: Show trends for each feature.
- **Resampling**: Focuses on daily data for 2019–2020 and fills missing values.
- **Histograms**: Display the distribution of each numerical feature.

---

6. Prepare Data for Machine Learning

```
weather_y = weather_df_num.pop("tempC")
weather_x = weather_df_num
```

Purpose:

- **`weather_y`**: The target variable (`tempC`) to predict.
- **`weather_x`**: The remaining features used as input for prediction.

---

7. Split the Data

```
train_X, test_X, train_y, test_y = train_test_split(weather_x, weather_y,
test_size=0.2, random_state=4)
```

Purpose:

Divides data into training and testing sets.

- **Training data (80%)**: Used to train the model.
- **Testing data (20%)**: Used to evaluate the model.
- **`random_state=4`**: Ensures consistent results.

---

8. Visualize Relationships

```
plt.scatter(weather_df_num['mintempC'], weather_df_num['tempC'])
plt.xlabel("Minimum Temperature")
plt.ylabel("Temperature")
plt.show()
```

Purpose:

- Displays the relationship between `mintempC` and `tempC` using a scatter plot.
- Helps identify if this feature is predictive.

---

9. Train a Linear Regression Model

```
model = LinearRegression()
model.fit(train_X, train_y)
prediction = model.predict(test_X)
```

Purpose:

- Fits a linear regression model on the training data.
- Predicts temperature (`tempC`) on the test set.

---

10. Evaluate the Linear Regression Model

```
np.mean(np.absolute(prediction - test_y))
print('Variance score: %.2f' % model.score(test_X, test_y))
```

Metrics:

- **Mean Absolute Error (MAE)**: Average error between predicted and actual values.
- **Variance score (R^2)**: Measures how well the model explains the variation in `tempC`.

---

11. Train a Decision Tree Model

```
regressor = DecisionTreeRegressor(random_state=0)
regressor.fit(train_X, train_y)
prediction2 = regressor.predict(test_X)
```

Purpose:

Trains a decision tree regressor to predict temperature and tests it on the test set.

---

12. Train a Random Forest Model

```
regr = RandomForestRegressor(max_depth=90, random_state=0, n_estimators=100)
regr.fit(train_X, train_y)
prediction3 = regr.predict(test_X)
```

Purpose:

Uses an ensemble of decision trees (random forest) to improve accuracy.

- **`max_depth=90`**: Prevents overfitting by limiting tree depth.
- **`n_estimators=100`**: Uses 100 decision trees.

---

13. Compare Models

For each model, calculate:

Mean Absolute Error (MAE):

```
np.mean(np.absolute(prediction - test_y))
```

Mean Squared Error (MSE):

```
np.mean((prediction - test_y) ** 2)
```

R^2 Score:

```
from sklearn.metrics import r2_score
print("R2-score: %.2f" % r2_score(test_y, prediction))
```

Purpose:

- Compares model performance based on prediction errors and R^2 score.

Evaluation :

Multiple linear regression

Mean absolute error: 1.20
Residual sum of squares (MSE): 2.51
R2-score: 0.96

Decision tree

Mean absolute error: 0.56
Residual sum of squares (MSE): 1.12
R2-score: 0.98

Random forest

Mean absolute error: 0.47

Residual sum of squares (MSE): 0.63
R2-score: 0.99

As we see that according to these models random forest is best for weather forecasting.

Summary

This script:

1. Loads and explores weather data.
2. Visualizes trends and relationships in the data.
3. Builds and evaluates three machine learning models:
   - Linear Regression
   - Decision Tree Regressor
   - Random Forest Regressor
4. Compares model performance to determine the best approach for temperature prediction.

This simple and structured approach ensures accurate weather forecasting with actionable insights.

Thank you ☺