Big Data Analytics

A Project Submitted

By
Batch 2

Under the Guidance of

Kunal

BMS INSTITUTE OF TECHNOLOGY AND MANAGEMENT

YELAHANKA BANGALORE

Phase 2 Documentation

Project: Big Data Analytics – Rainfall Analysis
Phase 2 Submission by: Batch 2

Year: 4th Year, Information Science and Engineering (ISE)
Institute: BMS Institute of Technology and Management, Yelahanka, Bangalore

---

1. Data Preparation and Preprocessing

Phase 2 involved preparing the dataset for machine learning applications. The raw rainfall dataset (1901–2015) was cleaned and processed to be compatible with both traditional and deep learning models.

Steps Involved:

- Handled missing values and invalid entries.
- Selected key features such as Year, Month, Region, and Rainfall.
- Applied train-test splitting (80:20 ratio) using train_test_split() from Scikit-learn.

Python

```
train_data, test_data = train_test_split(data, test_size=0.2, random_state=42)
```

---

2. Algorithm Selection

Two major models were selected for analysis:

a) Random Forest Regressor
A tree-based ensemble algorithm known for robustness and performance on structured data.

Python

```
random_forest_model = RandomForestRegressor()
```

b) LSTM Neural Network
A deep learning architecture for modeling time-series data, using Keras:

python

```
lstm_model = keras.Sequential([
  layers.LSTM(units=64, activation='relu', input_shape=(n_steps, n_features)),
  layers.Dense(1)])
```

- n_steps: number of past time steps used as input.
- n_features: number of features per time step.

---

3. Predictive Modeling with Linear Regression

To illustrate a basic regression pipeline, synthetic data was generated and evaluated using Linear Regression.

Python

```
model = LinearRegression()
model.fit(X_train, y_train)
```

Evaluation Metrics:

- **Mean Squared Error (MSE)**: Indicates average squared prediction error.
- **R^2 Score**: Measures how well the model explains the variability of the target.

Python

```
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
```

The prediction results were visualized using scatter plots, comparing actual and predicted values.

---

4. Anomaly Detection with Residual Analysis

Detecting anomalies in rainfall patterns is critical for understanding extreme events like floods and droughts. This was done by:

- Training a Random Forest on features like Year, Month, and Region.
- Computing residuals between actual and predicted values.
- Flagging entries with residuals greater than a defined threshold as anomalies.

python

```
residuals = y_test - y_pred
anomalies = np.where(np.abs(residuals) > threshold)[0]
```

Anomalies were printed and analyzed for further interpretation.

---

5. SQL and IBM Db2 Connectivity

A connection to **IBM Db2 Cloud Database** was maintained for storing processed and analyzed data.
Example SQL Command for Connection in VS Code:
Sql

```
CONNECT TO your_database_name
USER your_username
USING your_password
HOSTNAME your_hostname
PORT your_port
DATABASE your_database_name
```

This ensures a persistent backend to store results, models, or logs for further usage.

---

6. Summary of Phase 2

Task	Status
Data Preprocessing	✅ Completed
Train-Test Splitting	✅ Completed
Model Initialization (RF, LSTM)	✅ Completed
Linear Regression Evaluation	✅ Completed
Residual-based Anomaly Detection	✅ Completed
SQL Integration with Db2	✅ Completed

---

7. Tools and Libraries Used

- **Python** – Core programming and analysis.
- **Scikit-learn** – ML models and evaluation.
- **Keras/TensorFlow** – Deep learning (LSTM).
- **Pandas & NumPy** – Data manipulation.
- **Matplotlib** – Visualization.
- **IBM Db2** – Cloud database backend.