Microsoft Elevate

CAPSTONE PROJECT

Forest Fire Prediction using Machine Learning

Presented By
Student Name: Mukund Nigam
College Name: Jaypee Institute of Information Technology (Noida)
Department: BTech - CSE
Email ID: mukundnigam222@gmail.com

OUTLINE:

- Problem Statement
- Proposed System/Solution
- System Development Approach
- Algorithm & Deployment
- Result
- Conclusion
- Future Scope
- References

Problem Statement:

Currently, forest fires are one of the major environmental threats that cause severe damage to forests, wildlife, and human life. These fires often occur due to unfavorable weather conditions such as high temperature, low humidity, strong wind speed, and low rainfall. It is difficult for forest authorities to manually monitor and predict fire occurrence in advance, especially across large forest regions. Eventually, providing early warning and prevention of forest fires becomes a major concern. The crucial part is the prediction of fire occurrence based on environmental and weather conditions to help authorities take timely preventive measures and reduce damage.

Proposed Solution:

- The proposed system aims to address the challenge of predicting forest fire occurrence based on environmental and weather conditions. This involves leveraging data analytics and machine learning techniques to accurately identify fire-prone situations in advance. The solution will consist of the following components:
- Data Collection:
  - Gather historical data on bike rentals, including time, date, location, and other relevant factors.
  - Utilize real-time data sources, such as weather conditions, events, and holidays, to enhance prediction accuracy.
- Data Preprocessing:
  - Clean and preprocess the collected data to handle missing values, duplicate records, and inconsistencies.
  - Perform feature selection and transformation to extract important features that influence forest fire occurrence.
- Machine Learning Algorithm:
  - Implement machine learning algorithms such as Logistic Regression, Random Forest, Support Vector Machine, and XGBoost to predict whether a forest fire will occur.
  - Train the models using historical data and optimize them to improve prediction performance.
  - Consider important features such as temperature, humidity, rainfall, and fire weather indices to enhance prediction accuracy.

Proposed Solution:(Cont...)

- Deployment:
  - Develop a prediction system that can analyze weather conditions and predict fire occurrence in advance.
  - Deploy the trained model in a user-friendly interface or monitoring system to assist forest authorities in decision-making.
  - Ensure the system is scalable and capable of handling real-time environmental data.
- Evaluation:
  - Assess the model's performance using evaluation metrics such as Accuracy, Precision, Recall, and F1-score.
  - Compare different machine learning models to identify the best-performing model.
  - Fine-tune the model based on performance results to improve prediction accuracy.
- Result:
  - The machine learning models successfully predict forest fire occurrence based on weather and environmental data.
  - Random Forest and XGBoost algorithms provided the highest prediction accuracy.
  - The system helps identify high-risk fire conditions early, enabling preventive measures and reducing forest damage.

System Approach:

The "System Approach" section outlines the overall strategy and methodology for developing and implementing the rental bike prediction system. Here's a suggested structure for this section:

- System requirements
  - Hardware Requirements:
    - Processor: Intel Core i3 or higher
    - RAM: Minimum 4 GB (8 GB recommended)
    - Storage: Minimum 1 GB free space
  - Software Requirements:
    - Operating System: Windows / Linux / macOS
    - Programming Language: Python
    - Development Environment: Jupyter Notebook / VS Code
- Library required to build the model
  - The following Python libraries were used:
    - Pandas → For data loading and preprocessing
    - NumPy → For numerical computations
    - Matplotlib → For data visualization
    - Seaborn → For advanced visualizations and correlation analysis
    - Scikit-learn → For machine learning models and evaluation
    - XGBoost → For advanced prediction using gradient boosting

Algorithm & Deployment:

- **Algorithm Selection:**
  - In this project, machine learning algorithms such as Logistic Regression, Random Forest, Support Vector Machine (SVM), and XGBoost were used to predict forest fire occurrence. Among these, Random Forest and XGBoost provided the best performance due to their ability to handle complex relationships and multiple input features effectively. These algorithms are suitable for classification problems and can accurately predict whether a fire will occur or not based on environmental conditions.
- **Data Input:**
  - The input features used by the algorithm include environmental and weather-related parameters such as:
  - Temperature
  - Relative Humidity (RH)
  - Wind Speed (Ws)
  - Rainfall
  - Fire Weather Index components (FFMC, DMC, DC, ISI, BUI, FWI)
  - Region information
  - These features help the model understand conditions that increase or decrease fire risk.

Algorithm & Deployment:(Cont...)

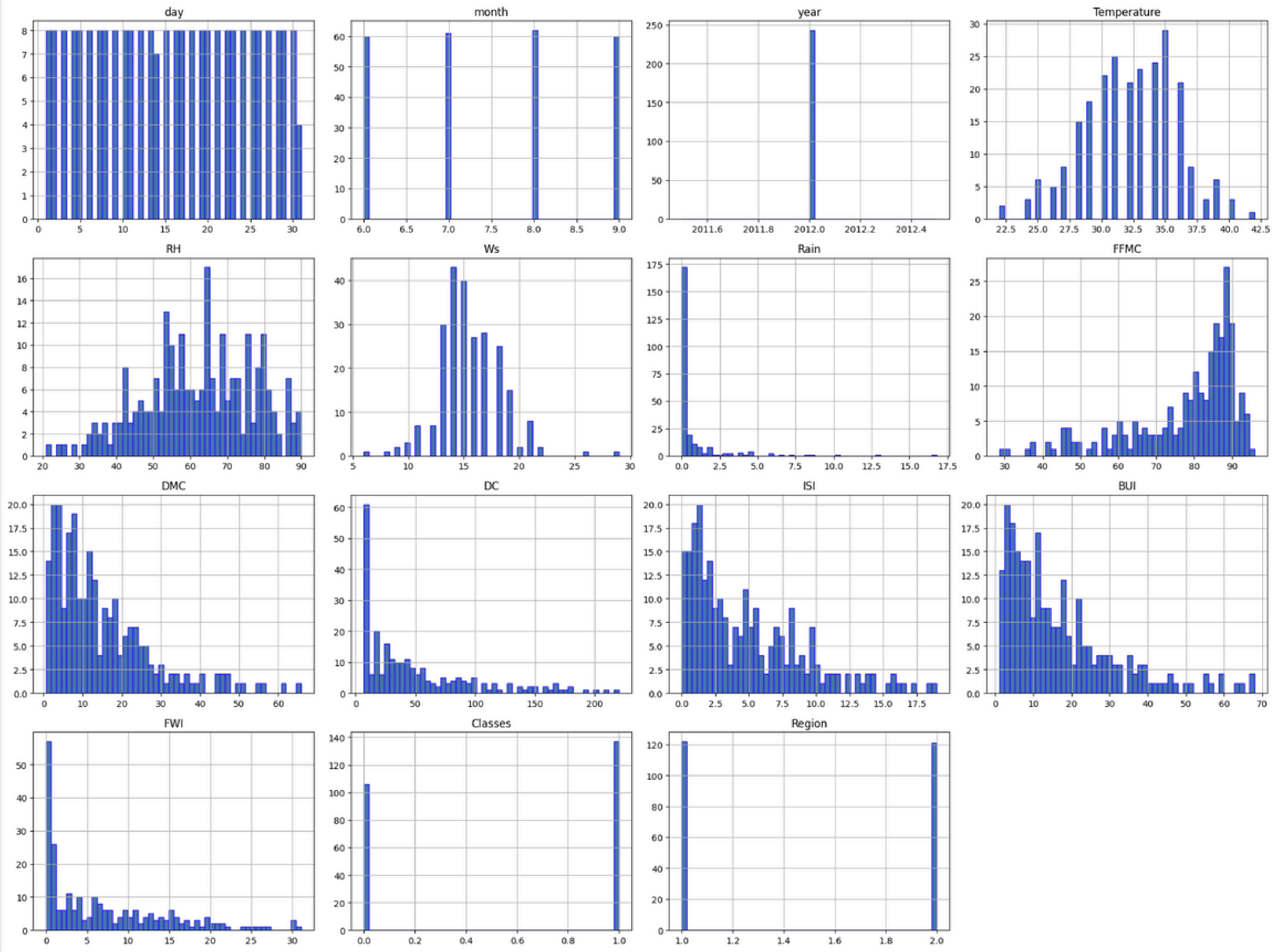
- Training Process:
  - The dataset was first preprocessed to handle missing values, remove duplicates, and select relevant features. The dataset was then divided into training and validation sets using train-test split. Machine learning models were trained using the training data. Techniques such as feature scaling using StandardScaler and model evaluation using accuracy and classification metrics were applied to improve performance. Cross-validation techniques were also used to ensure reliable model performance
- Prediction Process:
  - After training, the model uses the learned patterns to predict whether a forest fire will occur based on new environmental input data. The trained model analyzes the input features and classifies the output as either "Fire" or "No Fire." This prediction helps identify high-risk conditions in advance, allowing preventive measures to be taken to reduce forest fire damage.

Result:

The machine learning models were successfully trained and evaluated to predict forest fire occurrence based on environmental and weather conditions. Multiple algorithms such as Logistic Regression, Random Forest, Support Vector Machine (SVM), and XGBoost were implemented and compared.
Among these models, Random Forest and XGBoost provided the highest accuracy and best performance in predicting forest fires. The models were able to effectively identify patterns between environmental factors such as temperature, humidity, wind speed, rainfall, and fire weather indices and the occurrence of forest fires.
Feature importance analysis showed that parameters such as FFMC, Temperature, Relative Humidity, and Rainfall were significant predictors of forest fire occurrence.
Visualizations such as correlation heatmaps, feature importance graphs, and class distribution charts were used to analyze the model's performance and understand the relationship between features and fire occurrence.
The results demonstrate that machine learning techniques can accurately predict forest fire risk and help in early detection and prevention.

Result:(Cont...)

Images



	Missing Count	Missing Percentage
DC	1	0.409836
FWI	1	0.409836
Classes	1	0.409836
day	0	0.000000
month	0	0.000000
year	0	0.000000
Temperature	0	0.000000
RH	0	0.000000
Ws	0	0.000000
Rain	0	0.000000
FFMC	0	0.000000
DMC	0	0.000000
ISI	0	0.000000
BUI	0	0.000000
Region	0	0.000000

Result:(Cont...)

```
Logistic Regression
Accuracy : 0.9388
Precision: 0.9286
Recall   : 0.9630
F1-Score : 0.9455
ROC-AUC  : 0.9966

K-Nearest Neighbors
Accuracy : 0.7551
Precision: 0.7586
Recall   : 0.8148
F1-Score : 0.7857
ROC-AUC  : 0.8577

Decision Tree
Accuracy : 1.0000
Precision: 1.0000
Recall   : 1.0000
F1-Score : 1.0000
ROC-AUC  : 1.0000

Random Forest
Accuracy : 0.9592
Precision: 0.9630
Recall   : 0.9630
F1-Score : 0.9630
ROC-AUC  : 0.9949

SVM Classifier
Accuracy : 0.8367
Precision: 0.8276
Recall   : 0.8889
F1-Score : 0.8571
ROC-AUC  : 0.9512

XGBoost
Accuracy : 0.9592
Precision: 0.9630
Recall   : 0.9630
F1-Score : 0.9630
ROC-AUC  : 0.9966
```

Model Performance Comparison

	Accuracy	Precision	Recall	F1-Score	ROC-AUC
Decision Tree	1.000000	1.000000	1.000000	1.000000	1.000000
Random Forest	0.959184	0.962963	0.962963	0.962963	0.994949
XGBoost	0.959184	0.962963	0.962963	0.962963	0.996633
Logistic Regression	0.938776	0.928571	0.962963	0.945455	0.996633
SVM Classifier	0.836735	0.827586	0.888889	0.857143	0.951178
K-Nearest Neighbors	0.755102	0.758621	0.814815	0.785714	0.857744

Model Performance Comparison (Cross-Validated)

	F1-Mean	F1-StdDev
Decision Tree	0.977908	0.033206
XGBoost	0.974289	0.032569
Random Forest	0.972101	0.029436
Logistic Regression	0.953735	0.036642
SVM Classifier	0.933758	0.047565
K-Nearest Neighbors	0.872473	0.056556

Result:(Cont...)

```
[30]: models = {
    "Logistic Regression": LogisticRegression(random_state=42),
    "K-Nearest Neighbors": KNeighborsClassifier(),
    "Decision Tree": DecisionTreeClassifier(random_state=42),
    "Random Forest": RandomForestClassifier(random_state=42),
    "SVM Classifier": SVC(probability=True, random_state=42),
    "XGBoost": XGBClassifier(random_state=42),
}

results = {}

for name, model in models.items():
    pipeline = Pipeline(steps=[
        ('preprocessor', StandardScaler()),
        ('classifier', model)
    ])

    pipeline.fit(X_train, y_train)

    y_pred = pipeline.predict(X_val)
    y_pred_proba = pipeline.predict_proba(X_val)[:, 1]

    accuracy = accuracy_score(y_val, y_pred)
    f1 = f1_score(y_val, y_pred)
    roc_auc = roc_auc_score(y_val, y_pred_proba)
    precision = precision_score(y_val, y_pred)
    recall = recall_score(y_val, y_pred)

    results[name] = {
        'Accuracy': accuracy,
        'Precision': precision,
        'Recall': recall,
        'F1-Score': f1,
        'ROC-AUC': roc_auc
    }

    print(f"{name}")
    print(f"Accuracy : {accuracy:.4f}")
    print(f"Precision: {precision:.4f}")
    print(f"Recall    : {recall:.4f}")
    print(f"F1-Score : {f1:.4f}")
    print(f"ROC-AUC   : {roc_auc:.4f}\n")
```

```
for name, model in models.items():

    pipeline = Pipeline(steps=[
        ('preprocessor', StandardScaler()),
        ('classifier', model)
    ])

    scores = cross_val_score(pipeline, X, y, cv=cv_strategy, scoring='f1', n_jobs=-1)

    results[name] = {
        'F1-Mean': scores.mean(),
        'F1-StdDev': scores.std()
    }

    print(f"{name}: Mean F1-Score = {scores.mean():.4f} (StdDev = {scores.std():.4f})")

results_df = pd.DataFrame(results).T.sort_values(by='F1-Mean', ascending=False)
print("\nModel Performance Comparison (Cross-Validated)")
print(results_df)
```

Conclusion:

This project successfully demonstrated the effectiveness of machine learning techniques in predicting forest fire occurrence using environmental and weather data. Models such as Random Forest and XGBoost showed high accuracy and reliability in identifying fire-prone conditions.
The proposed solution effectively analyzed important features such as temperature, humidity, rainfall, wind speed, and fire weather indices to predict whether a forest fire is likely to occur. This helps in early detection and allows authorities to take preventive measures to reduce damage.
During implementation, challenges such as handling missing values, feature selection, and model optimization were encountered. However, proper data preprocessing and model evaluation techniques helped overcome these challenges.
Accurate prediction of forest fires is crucial for forest management, disaster prevention, and environmental protection. The developed system provides a reliable and efficient method for predicting forest fire risk and supporting decision-making.

Future scope:

The forest fire prediction system can be further enhanced by incorporating additional real-time environmental data such as satellite imagery, air quality index, and soil moisture levels to improve prediction accuracy. Integrating live weather data from sensors and meteorological stations can help provide more accurate and timely predictions.

The performance of the system can be improved by using advanced machine learning and deep learning techniques such as Artificial Neural Networks (ANN), Long Short-Term Memory (LSTM), and hybrid models. Hyperparameter tuning and larger datasets can also enhance model accuracy and reliability.

The system can be expanded to cover multiple forest regions, cities, and countries to provide a scalable and comprehensive fire prediction solution. Integration with Geographic Information Systems (GIS) can help visualize fire-prone areas on maps for better monitoring and decision-making.

In the future, the system can be deployed as a web or mobile application to allow forest authorities to access predictions in real-time. Emerging technologies such as edge computing and IoT sensors can also be integrated to enable faster predictions and early warning systems directly in forest environments.

References:

[1] P. Cortez and A. Morais, “A Data Mining Approach to Predict Forest Fires using Meteorological Data,” Proceedings of the 13th Portuguese Conference on Artificial Intelligence, pp. 512–523, 2007.

[2] Scikit-learn Developers, “Scikit-learn: Machine Learning in Python,” Available: https://scikit-learn.org/

[3] T. Chen and C. Guestrin, “XGBoost: A Scalable Tree Boosting System,” Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016.

[4] Wes McKinney, “Pandas: Powerful Python Data Analysis Toolkit,” Available: ist and cite relevant sources, research papers, and articles that were instrumental in developing the proposed solution. This could include academic papers on bike demand prediction, machine learning algorithms, and best practices in data preprocessing and model evaluation.

[5] NumPy Developers, “NumPy Documentation,” Available: https://numpy.org/

GitHub Link: [Link (https://github.com/mukuuund/Forest-Fire-Prediction-using-Machine-Learning)](https://github.com/mukuuund/Forest-Fire-Prediction-using-Machine-Learning)

Thank You