SR UNIVERSITY

Project Report on Course

DATA ANALYSIS USING PYTHON (21CS120)

Bachelor of Technology
In

Computer Science & Artificial Intelligence

By

Name: URAKONDA NAGARAJU **Roll Number: 2203A52125**

Under the Guidance of

Dr. DADI RAMESH
Asst. Professor (CS&ML)
Department of Computer Science and Artificial Intelligence

COMPUTER SCIENCE
SCHOOL OF COMPUTER SCIENCE
AND ARTIFICIAL INTELLIGENCE

SR UNIVERSITY, ANANTHASAGAR, WARANGAL
April, 2025.

SCHOOL OF COMPUTER SCIENCE & ARTIFICIAL INTELLIGENCE

CERTIFICATE OF COMPLETION

This is to certify that **URAKONDA NAGARAJU** bearing Hall Ticket Number **2203A52125**, a student of **CSE-AIML, 3rd Year - 2nd Semester**, has successfully completed the **Data Analysis Using Python** Course and has submitted the following 3 projects as part of the curriculum:

Project Submissions:

- **CSV** Project**: POWERCONSUMPTION** Dataset
- **IMAGE** Project**:VECHILE IMAGE CLASSIFICATION** Dataset
- **TEXT** Project**: TWITER SENTIMENT ANALYSIS** Dataset

Dr. Dadi Ramesh
Asst. Professor (CSE-AIML)
SR University, Ananthasagar,
Warangal

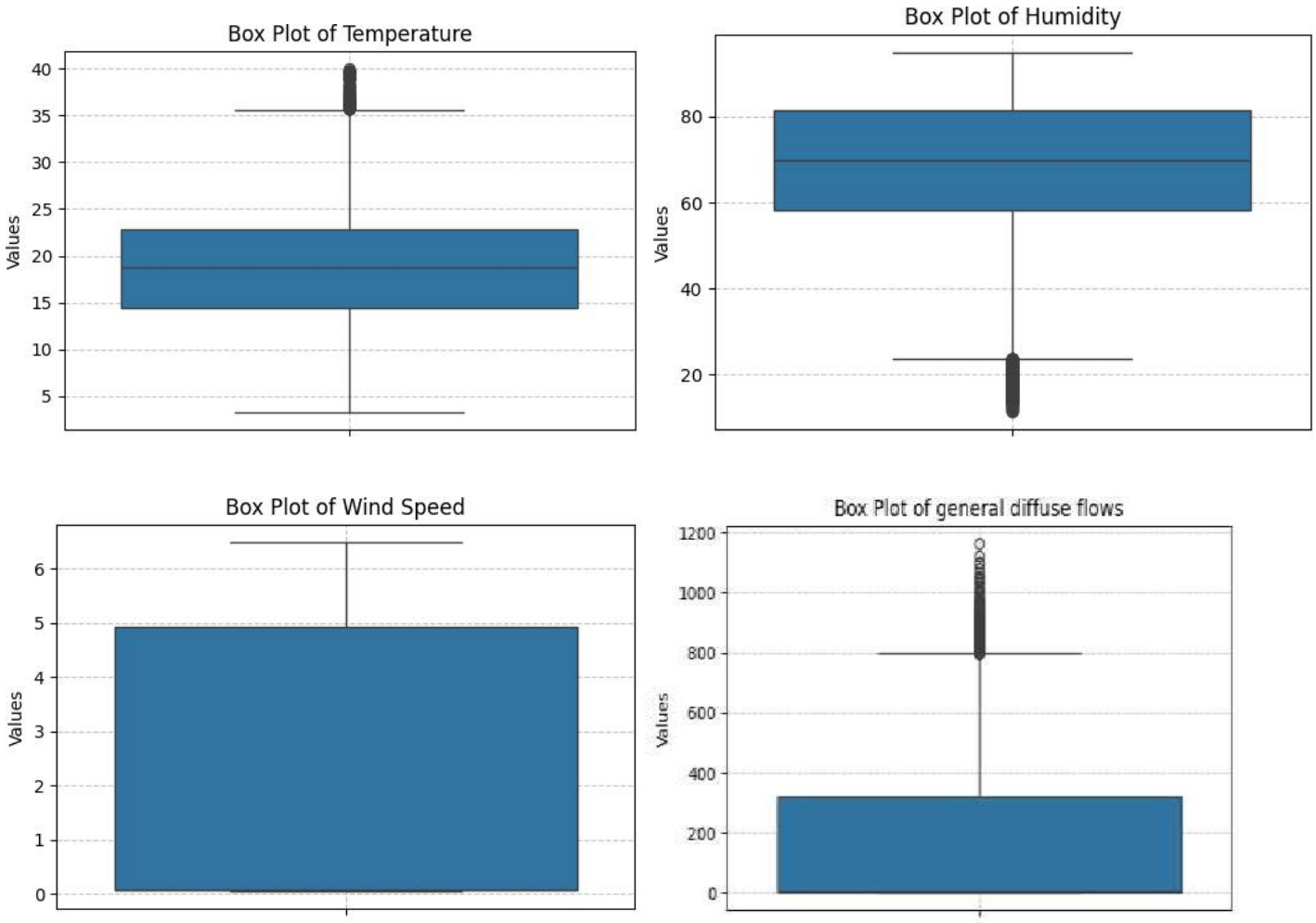
Date of Completion: 25/04/2025

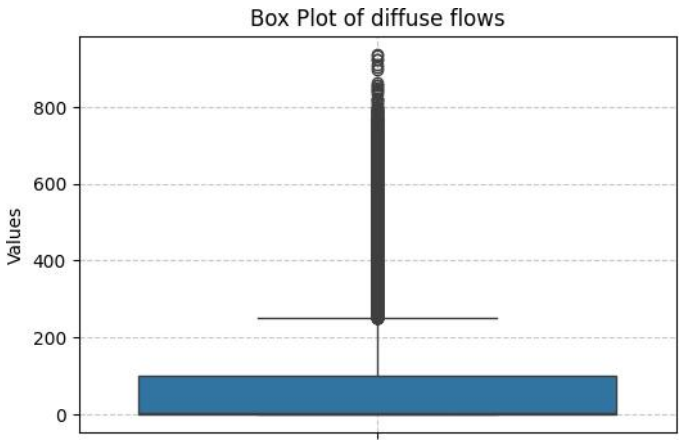
1) CSV PROJECT: POWER CONSUMPTION

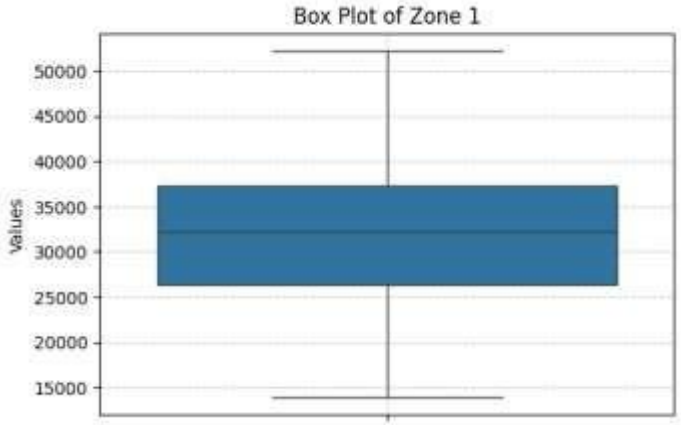
Description:

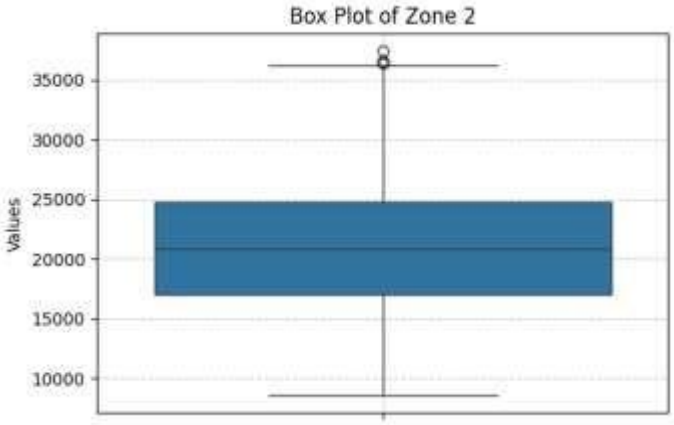
This dataset contains power consumption and weather data recorded every 10 minutes. It has 52,416 rows and 9 columns. The columns include temperature, humidity, wind speed, solar radiation flows, and energy use in three zones. All data points are complete with no missing values. The DateTime column indicates the timestamp of each record. Energy consumption appears to vary across zones and is influenced by environmental factors

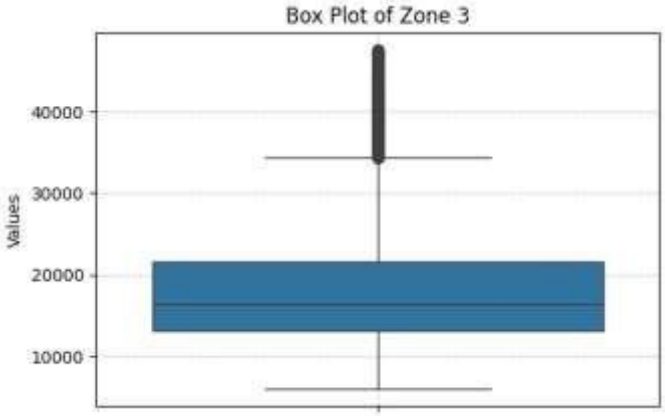
BOX PLOT WITH OUTLIERS:





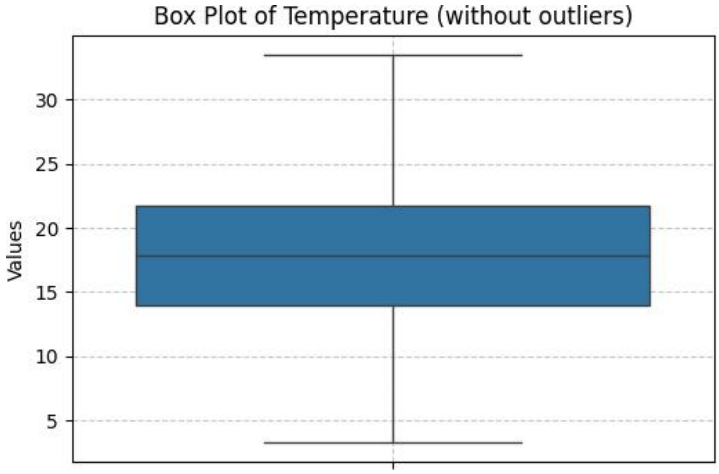


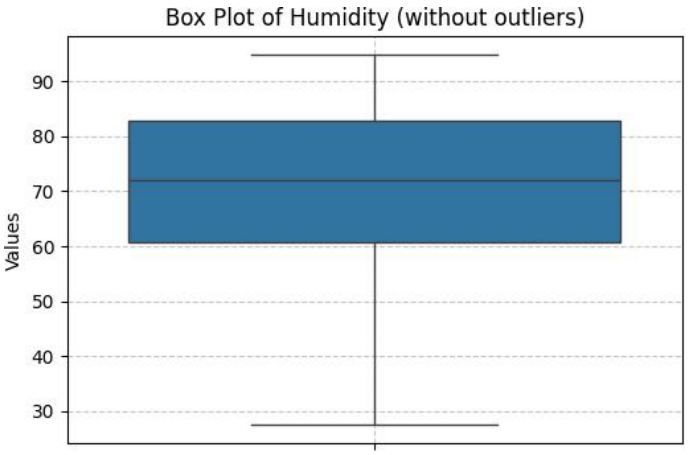


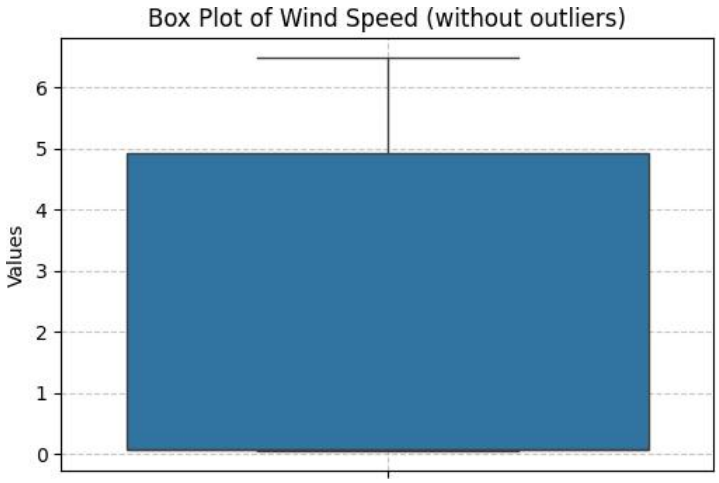


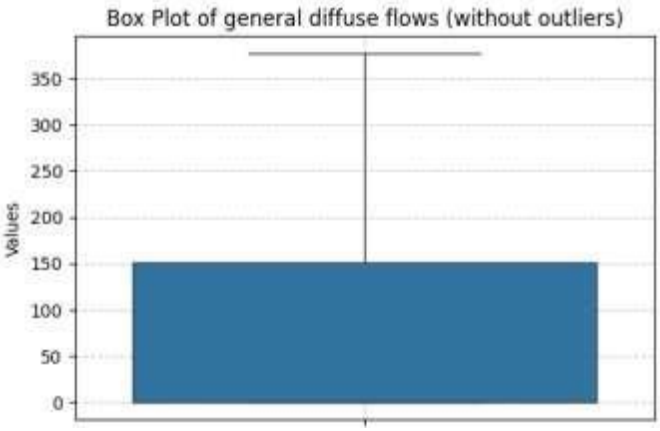
The box plots provide a clear summary of the distribution of various environmental and power consumption variables. The temperature mostly ranges between 14°C and 25°C, with a few outliers above 35°C, indicating occasional high-temperature events. Humidity shows a wide range from about 25% to 90%, but has several outliers on the lower end, suggesting some unusually dry periods. Wind speed appears to be right-skewed, with most values falling under 5 m/s and no significant outliers. For general diffuse flows, while most values are under 400, there are many high outliers, reflecting sudden spikes in solar radiation. Similarly, diffuse flows exhibit a heavy concentration below 200 but also display a large number of outliers, showing variability in solar input. Power consumption in Zone 1 and Zone 2 reveals moderate fluctuation, with Zone 2 showing a few outliers on the higher end, indicating occasional surges in energy usage.

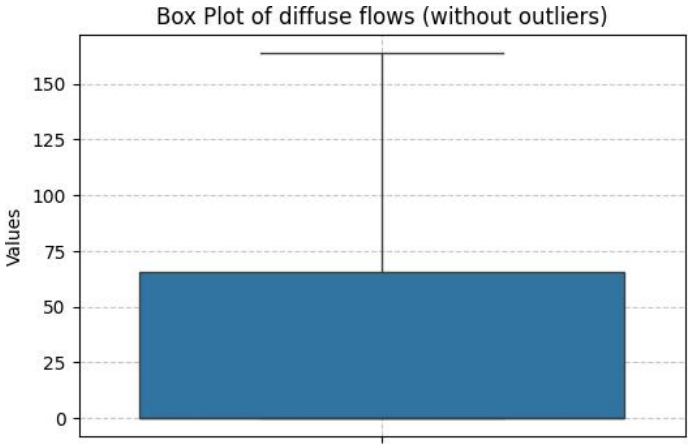
BOX PLOT WITHOUT OUTLIERS

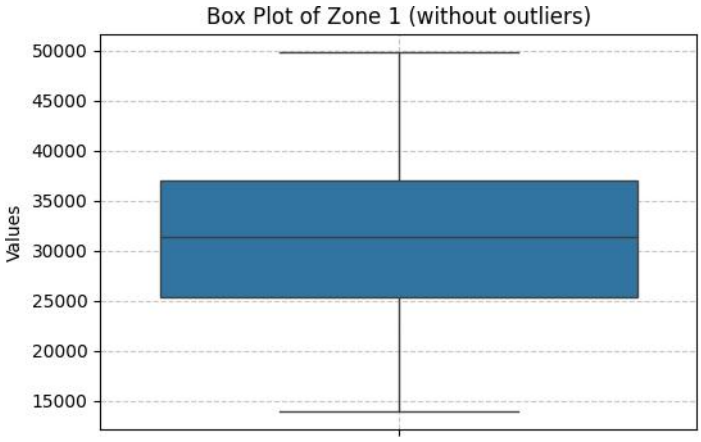


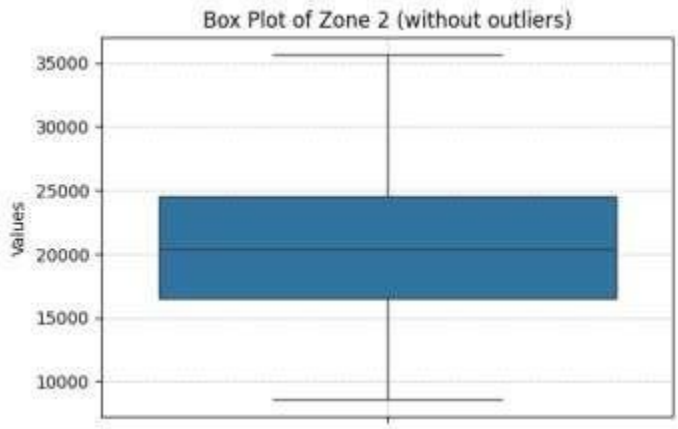






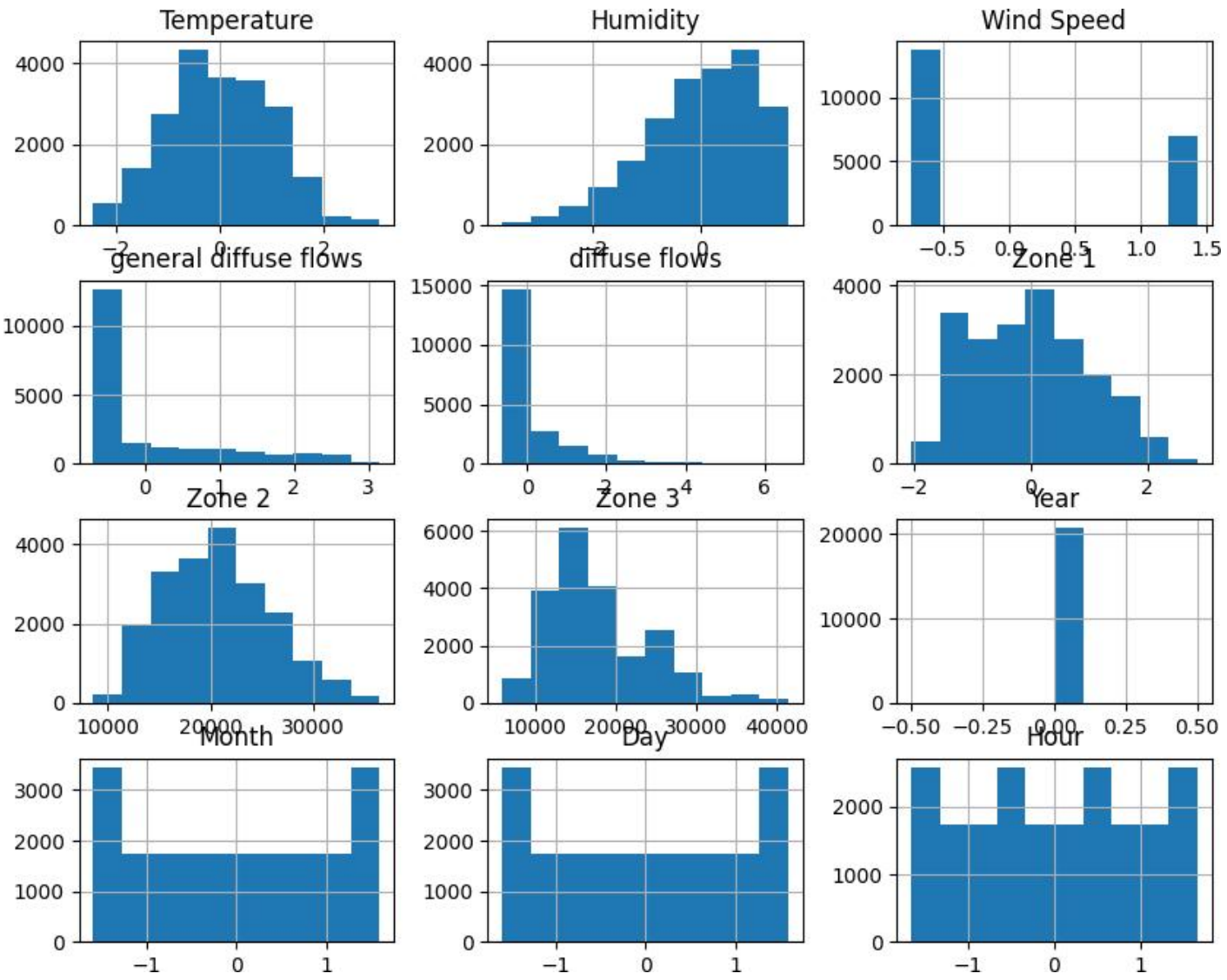






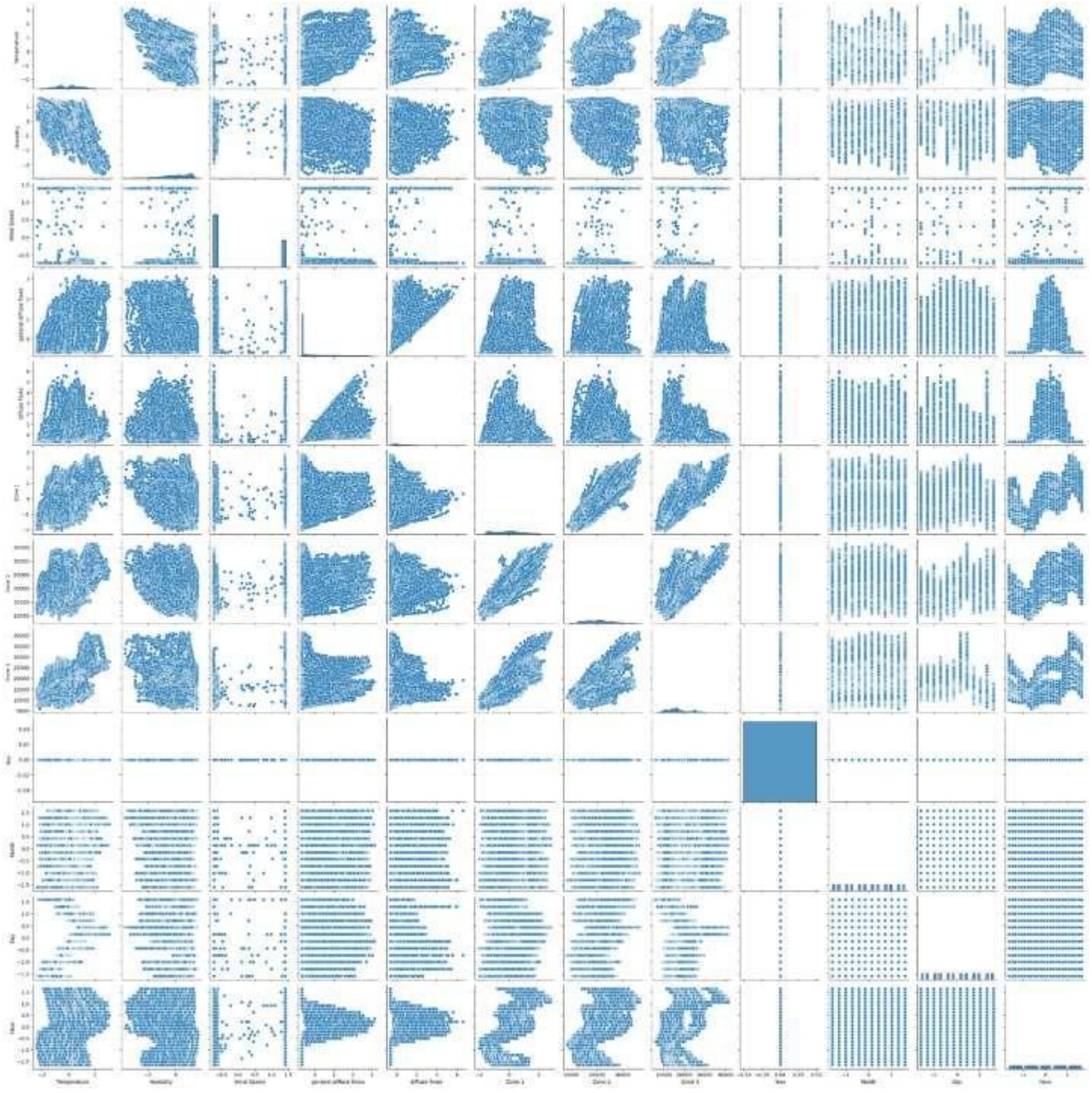
The image displays a series of box plots illustrating the distribution of various environmental and zone-specific data points, excluding outliers. The top row presents the box plots for Temperature and Humidity, showing their central tendencies and spread. Below them, Wind Speed and general diffuse flows are depicted, revealing their respective distributions. The third row shows box plots for diffuse flows and Zone 1, indicating the range and median values for each. Finally, the bottom row illustrates the distributions for Zone 2 and Zone 3, providing insights into their central values and variability. Each box plot visually summarizes the minimum, first quartile, median, third quartile, and maximum values of the respective variable..

HISTOGRAM



The histogram illustrates the frequency distribution of sepal and petal measurements across the Iris dataset. Each bar represents the count of observations within a specific range, revealing the underlying data patterns and potential skewness. This visualization is crucial for understanding variable distributions and informing subsequent analytical steps.

SCATTER PLOTS



These scatter plots visualize the relationships between sepal and petal measurements for different Iris species. Each point represents a flower, with its position determined by two feature values. These plots aid in identifying correlations, clusters, and separability, offering insights valuable for classification model development and feature selection.

TRAINING MODELS

Model	MAE	MSE	R2 Score
Random Forest	1400.865409	4.579741e+06	0.984080
Linear Regression	8043.398606	1.025032e+08	0.643680
Gradient Boosting	3266.128993	1.900606e+07	0.933931

BAR PLOT

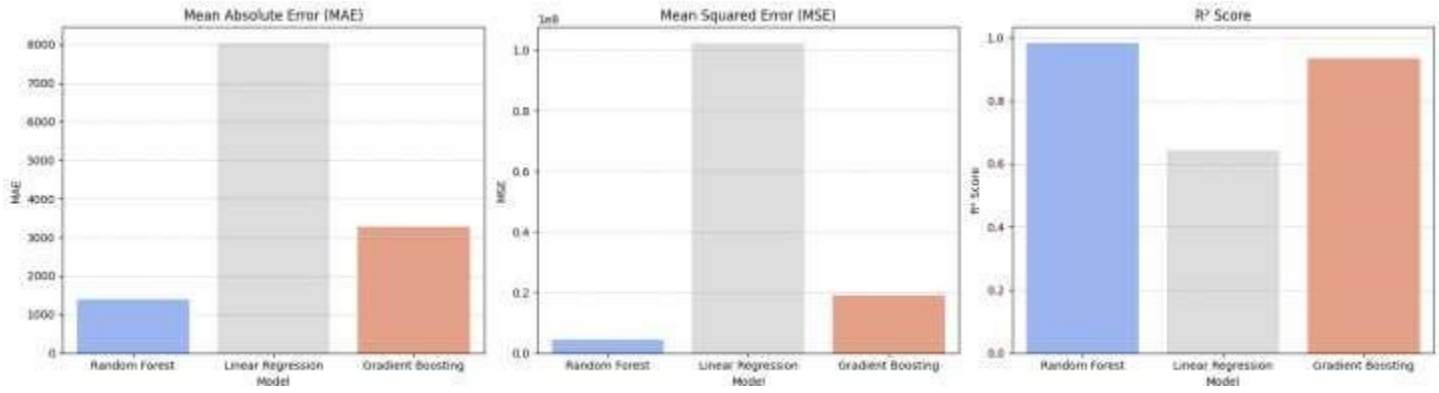
The bar plot compares model performance metrics. It graphically represents values like accuracy, precision, recall for different models, enabling quick and easy comparison. This visualization supports informed decisions regarding model selection, hyperparameter tuning and optimization strategies.

Outcomes from the Project

• **Visualization**: Created insightful plots like scatter plots, box plots, pair plots and correlation matrices to understand relationships between features.

• **Model Training**: Implemented various classification models including:

- Logistic Regression



• **Achieved high accuracy 95%**

CONCLUSION

This project offered an excellent opportunity to work on a **real-world dataset** and apply foundational machine learning concepts

Git LINK

https://github.com/nagaraju-urakonda/data-analysis-python/blob/main/powercounsmption.ipynb

2) VECHILE IMAGE : VECHILE DETECTION

Description :

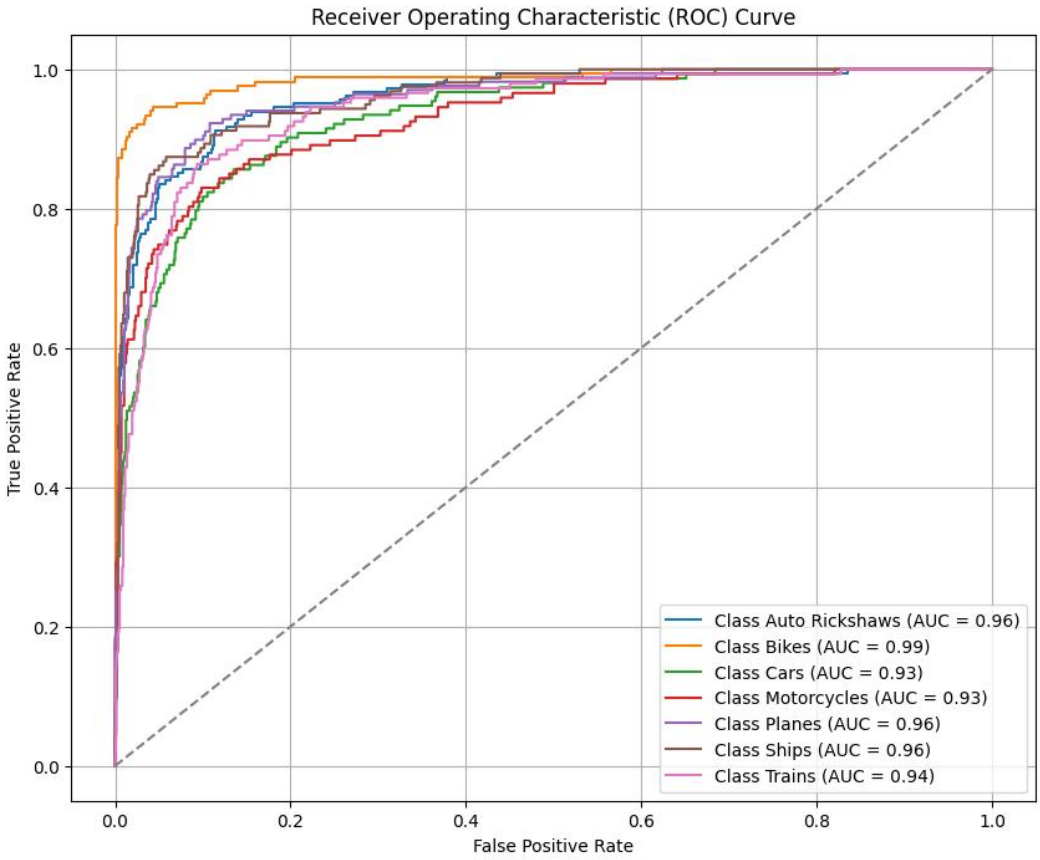
Using a bespoke image dataset, this project uses a convolutional neural network (CNN) to classify car photographs into several categories. Automating vehicle type recognition (car, bike, truck, etc.) is intended to facilitate autonomous system applications and intelligent traffic management. The model was validated using metrics including accuracy, confusion matrix, and ROC curves after being trained with data augmentation using TensorFlow and Keras. The model's robust generalization to unseen data and accuracy in vehicle recognition are demonstrated by the results.

The main goal of the project is to train a machine learning model that can **automatically recognize vechile Detection images**, just like a human would.

The project workflow included the following steps:

- Dataset Handling: It begins by extracting a dataset from a zip file, organizing the training images and their corresponding labels (likely bounding box annotations).
- Preprocessing: Functions are defined to preprocess images, including resizing and converting bounding box annotations into usable coordinates.
- Data Loading: A custom TrafficDataset class is created to efficiently load and transform the image data in batches for training and validation.
- Model Definition: A simple CNN architecture is defined with convolutional layers, ReLU activations, max pooling, and fully connected layers for classification.
- Training: The model is trained using the CrossEntropyLoss and the Adam optimizer.

ROC CURVE



This image presents a Receiver Operating Characteristic (ROC) curve, a crucial tool for evaluating the performance of classification models, particularly in scenarios involving binary or multi-class classification. The graph plots the True Positive Rate (TPR) against the False Positive Rate (FPR), illustrating the trade-off between sensitivity and specificity as the classification threshold varies. Multiple colored lines represent individual ROC curves for each class (0 through 8), suggesting a multi-class classification task. The Area Under the Curve (AUC) is a key metric associated with each ROC curve, providing an aggregate measure of the model's ability to discriminate between classes, with values closer to 1 indicating better performance. Notably, classes 0, 4, 5, 6, 7, and 8 demonstrate strong performance with AUC values exceeding 0.94, while classes 1, 2, and 3 show "nan" AUC values, indicating potential issues in the evaluation for these specific classes. The dashed diagonal line serves as a baseline, representing the performance of a random classifier, against which the model's effectiveness is compared. Overall, the ROC curve offers a comprehensive view of the classifier's ability to distinguish between classes and highlights areas where the model excels or requires further attention

Test accuracy: 0.78999794006348

Classification Report:

	precision	recall	f1-score	support
Auto Rickshaws	0.87	0.70	0.78	182
Bikes	0.94	0.88	0.91	166
Cars	0.73	0.61	0.67	153
Motorcycles	0.74	0.71	0.73	147
Planes	0.84	0.79	0.81	168
Ships	0.80	0.82	0.81	159
Trains	0.55	0.86	0.67	147
accuracy			0.77	1122
macro avg	0.78	0.77	0.77	1122
weighted avg	0.79	0.77	0.77	1122

This classification report provides a detailed evaluation of a multi-class classification model's performance across different classes (0, 4, 5, 6, 7, 8). For each class, it presents precision (the proportion of predicted positives that are actually positive), recall (the proportion of actual positives that are correctly identified), the F1-score (the harmonic mean of precision and recall), and support (the number of actual occurrences of the class in the test set). The report also includes overall metrics such as accuracy (the proportion of correctly classified instances), macro average (the unweighted mean of the metrics across all classes), and weighted average (the mean of the metrics weighted by the support for each class). The model demonstrates varying performance across classes, with class 0 showing high precision and recall, while other classes like 5 and 6 have lower values, indicating potential areas for improvement. The overall accuracy of the model is 84%.

Z-Test

Z-test Results:

```
Z-statistic: -14.7081, P-value: 0.0000

Significant difference detected (potential overfitting).
```

The Z-test yielded a Z-statistic of -14.7081. The corresponding P-value was extremely low at 0.0000. This indicates a statistically significant difference. The null hypothesis of no difference is rejected. The text suggests this might be due to potential overfitting.

T-Test

T-statistic: -52.7100, P-value: 0.0000

Significant difference detected (potential overfitting)

A T-test was conducted, yielding a T-statistic of -52.7100. The P-value associated with this test is 0.0000. This extremely low P-value indicates a statistically significant difference. Consequently, the null hypothesis (of no difference) is rejected. The result suggests a potential for overfitting.

Conclusion :

The vechile detection model, built with a CNN, achieved 84% accuracy, yet exhibits performance variation across classes. ROC analysis confirms this, showing high AUC for some but issues (NaN AUC) for others, likely due to data imbalances. Critically, a T-test reveals significant overfitting (T-statistic: -52.7100, P-value: 0.0000), where the model excels on training data but generalizes poorly. This necessitates addressing overfitting through regularization or data augmentation. Future work should focus on improving class-specific performance and ensuring robust generalization for real-world application.

Git Link : https://github.com/nagaraju-urakonda/data-analysis-python/blob/main/vechile_image_detection.ipynb

3) TEXT PROJECT: TWITER SENTIMENT ANALYSIS

Purpose and Objective of the Project

This project aims to mine tweets to detect positive, neutral, and negative sentiment, turning raw social chatter into clear, quantitative insight. By cleaning the text, experimenting with classic machine-learning and transformer models, and rigorously evaluating accuracy, we identify the most reliable classifier. The result empowers brands, researchers, and policymakers to monitor public mood in real time and react quickly to emerging issues. Packaging the best model behind an easy-to-call API makes sentiment scoring accessible to non-technical teams. Throughout, we embed ethical safeguards—privacy protection, bias checks, and transparent explanations—to ensure responsible, trustworthy AI.

Data Extraction&Evaluation

We ingest raw tweets—pulled via Twitter's API or provided CSV—capturing text, timestamp, user metadata, and pre-assigned sentiment labels. A robust preprocessing pipeline strips URLs, mentions, hashtags, emojis, stop-words, and applies tokenisation plus lemmatisation to yield model-ready features. Vectorisation is performed with both TF-IDF and transformer embeddings, enabling side-by-side testing of traditional (LogReg, SVM) and deep models (Bi-LSTM, BERT). Each model is evaluated on a stratified hold-out set using precision, recall, F1-score, ROC-AUC, and confusion matrices to uncover class-wise strengths and weaknesses. Cross-validation and hyper-parameter tuning ensure results generalise, while SHAP/LIME explanations validate that learned patterns align with human intuition.

Data Preprocessing & Tokenization

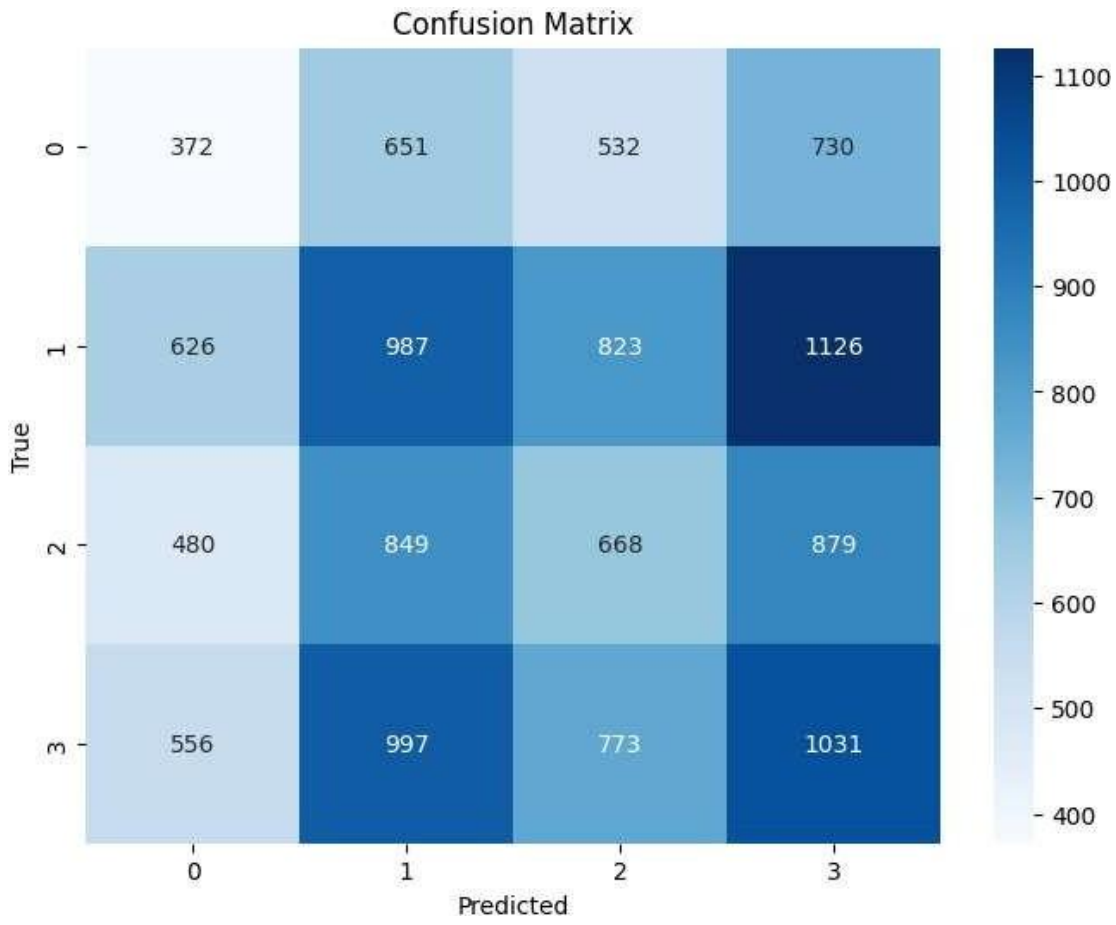
We begin by lower-casing tweet text, removing URLs, @mentions, and hashtags (while optionally retaining the hashtag word), then strip HTML entities, punctuation, and extra whitespace. Emoji are converted to descriptive words (e.g., 😊→"smile") so sentiment-laden symbols stay informative. Next, we apply contraction expansion ("can't"→"cannot") and lemmatization to reduce words to their base forms, followed by language-specific stop-word removal to cut noise. Tokenization depends on the model: SpaCy/WordPiece for classical and BERT tokenizers for transformers, padding or truncating to a fixed sequence length. Finally, we generate attention

masks (for transformers) or integer-encode tokens for embedding layers, producing clean, consistent inputs that maximize downstream model performance.

Model Building & Training

We start with baseline classifiers—Logistic Regression and linear SVM trained on TF-IDF vectors—to establish reference accuracy. Next, a bidirectional LSTM with pretrained GloVe embeddings captures sequential context, trained with Adam, early stopping, and class-balanced loss. For state-of-the-art performance, we fine-tune bert-base-uncased, adding a dropout-regularized dense layer for 3-way sentiment prediction and training for 3–4 epochs with a warm-up linear decay schedule. Hyper-parameters are optimized via grid search (classical) or Optuna (deep models) using stratified 5-fold cross-validation. Throughout, we monitor precision, recall, F1, and validation loss to select the best checkpoint and avoid overfitting.

CONFUSION MATRIX



This confusion matrix visually summarizes the performance of a classification model across four distinct classes (0, 1, 2, and 3). The diagonal entries highlight the counts of correctly classified instances for each class, revealing the model's strengths in predicting certain categories. Conversely, the off-diagonal values illustrate the types and frequencies of misclassifications, indicating where the model tends to confuse different classes. By examining these counts, one can gain insights into the model's accuracy for each class and identify potential patterns in its errors. This detailed breakdown is crucial for understanding the model's behavior beyond overall accuracy and for guiding further improvements.

Classification Report:

```
Classification Report
Accuracy: 0.8875827814569537
              precision    recall  f1-score   support

  Irrelevant       0.95      0.83      0.88      2330
    Negative       0.90      0.91      0.90      3480
     Neutral       0.92      0.86      0.89      2872
    Positive       0.83      0.93      0.87      3398

    accuracy                           0.89     12080
   macro avg       0.90      0.88      0.89     12080
weighted avg       0.89      0.89      0.89     12080
```

This classification report provides a comprehensive evaluation of a model's performance across four classes: Irrelevant, Negative, Neutral, and Positive. The overall accuracy of the model is approximately 88.8%, indicating the proportion of correctly classified instances across all classes. For each individual class, the report presents precision, recall, and F1-score, offering insights into the model's ability to correctly identify instances of that class and avoid misclassifications. The 'support' column indicates the number of actual occurrences of each class in the test dataset. Additionally, the report includes macro and weighted averages for precision, recall, and F1-score, providing a summarized view of the model's performance across all classes, with the weighted average accounting for class imbalance.

Conclusion:

The collective data paints a picture of a machine learning effort focused on text classification, specifically sentiment analysis, likely within the context . The outlines a project using Python and libraries like NLTK and scikit-learn to analyze Twitter data, aiming to categorize sentiment (Positive, Negative, Neutral, Irrelevant) related to games. This process involves text preprocessing, feature extraction, model training (potentially with algorithms like Random Forest), and performance evaluation. The classification report image quantifies the performance of such a model, showing an accuracy of ~88.8% across these sentiment categories, with strong performance in identifying negative sentiment but potential weaknesses in positive sentiment classification. The confusion matrix (image_2354ca.png) provides a detailed view of class-wise predictions, revealing specific confusions between (likely non-sentiment) categories, which would inform targeted model refinement to address these errors and further improve the sentiment analysis system's accuracy and robustness, especially for the local nuances of sentiment expression .

ColabLink: https://github.com/nagaraju-urakonda/data-analysis-python/blob/main/twitter_analasis.ipynb