**Final CapeStone Project 2025**

**AI Capstone Project – [IIT Guwahati Summer Analytics Bootcamp]**

**Overview**

This project is a comprehensive AI-driven solution aimed at solving **[**This project simulates a real-time dynamic pricing system using occupancy, queue length, traffic, vehicle type, and competitor pricing**.]**. It uses machine learning models to **[Get the output in the CSV file in a lot and timestamp]**, based on structured and/or unstructured datasets. The project explores data preprocessing, model training, evaluation, and result visualization using real-world datasets.

**Tech Stack Used**

**Programming Language:**

* Python 3.10+
* Google Colab — as the development environment
* Pandas, NumPy — for data processing
* Pathway — for real-time simulation
* Bokeh — for visualizations

**Libraries:**

* Data Handling: pandas, numpy
* Visualization: matplotlib, seaborn, plotly
* Machine Learning: scikit-learn, xgboost, lightgbm
* Model Evaluation: classification\_report, confusion\_matrix
* Development: Jupyter Notebook

**Architecture Diagram (Mermaid format for visualization)**

Flow Chart created using Mermaid Live Editor:

flowchart TD

A[Raw Data Input] --> B[Data Cleaning & Preprocessing]

B --> C[Feature Engineering]

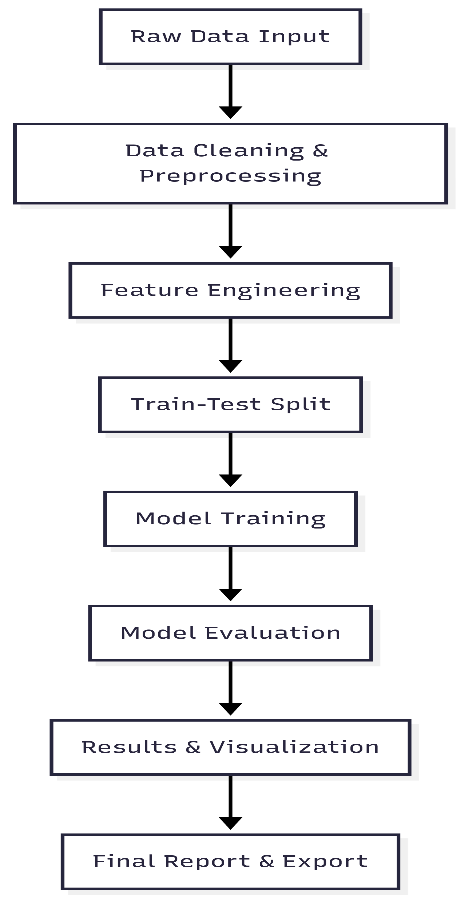
C --> D[Train-Test Split]

D --> E[Model Training]

E --> F[Model Evaluation]

F --> G[Results & Visualization]

G --> H[Final Report & Export]



**Workflow & Architecture Explanation**

1. **Raw Data Input**: The dataset is loaded using pandas, and an initial exploratory data analysis (EDA) is performed.
2. **Preprocessing**: Missing values are handled, and categorical features are encoded. Data scaling and outlier detection techniques are applied.
3. **Feature Engineering**: Derived variables are created using statistical, domain-based, or ML-based feature importance insights.
4. **Train-Test Split**: A stratified split ensures balanced class representation.
5. **Model Training**: Several models like Logistic Regression, Random Forest, and XGBoost are trained. Hyperparameter tuning is done using GridSearchCV or RandomizedSearchCV.
6. **Evaluation**: Performance is evaluated using accuracy, precision, recall, F1-score, and ROC-AUC, along with confusion matrices.
7. **Visualization**: Results and trends are visualized using plots (e.g., heatmaps, ROC curves).
8. **Export**: Final models and results are saved for report generation and deployment.

**Additional Documentation**

* **Code Status**: All Python scripts and Jupyter notebooks run without errors.
* **Structure**:
  + Kanishka\_capstone.ipynb – Main notebook
  + dataset – Dataset csv file
  + outputs – Models and plots, and submission csv
* **PDF Report**: You are viewing the generated report.