PREDICTING FLIGHT DELAYS WITH ERROR CALCULATION USING MACHINE LEARNING

B .Sri Harini[1], P.Revathi[2], T.Lahari[3], G.Sathvika[4], A.NagaHemalatha[5], B.Ksaturi[6]
[1]Assistant Professor, Dept of CSE, Gouthami Institute of Technology and Management For Women, Andhra Pradesh, India
[2 3 4 5 6] U.G Student, Dept of CSE, Gouthami Institute of Technology and Management For Women, Andhra Pradesh, India

ABSTRACT

Flight delays cause significant disruptions in air travel, affecting passengers, airlines, and airport operations. This project utilizes **machine learning (ML) algorithms** to predict flight delays with high accuracy by analysing **historical flight data, weather conditions, air traffic,and operational factors**. The system integrates **error calculation techniques (MAE, RMSE,MAPE, R^2-score)** to evaluate model performance and enhance prediction reliability. By leveraging real-time data processing and predictive modelling, the project aims to assist airlines, airports, and passengers in making informed decisions, optimizing schedules, and reducing economic losses due to flight disruptions. Flight delays pose significant challenges to the aviation industry, resulting in financial losses,operational inefficiencies, and passenger dissatisfaction. This study proposes a hybrid machine learning framework for predicting flight delays by integrating ensemble classifiers with a novel error-adjustment module to enhance prediction accuracy and robustness. The methodology employs Recursive Feature Elimination (RFE) and Principal Component Analysis (PCA) for feature selection and dimensionality reduction, ensuring the retention of the most influential fea
tures.

INTRODUCTION

Flight delays have become a persistent challenge within the aviation industry, affecting millions of passengers globally, creating operational inefficiencies for airlines, and leading to significant economic losses. The causes of flight delays are multifaceted, ranging from adverse weather conditions and air traffic congestion to operational issues and technical failures. As flight delays impact both the customer experience and airline profitability, accurately predicting andmanaging delays is crucial for improving operational efficiency and minimizing disruptions. The traditional methods of managing flight delays often rely on historical averages andheuristic-based approaches, which fail to capture the complex, dynamic nature of the factors that contribute to delays. Machine learning (ML) offers a promising alternative, as it has the ability to analyze vast amounts of historical flight data and weather patterns, learn from past occurrences, and predict future delays with high accuracy. Machine learning models, particularly ensemble classifiers like Random Forest, Gradient Boosting, and Support Vector Machines (SVM), have been proven effective for classification tasks involving complex, high dimensional datasets.

Literature Review:

The prediction of flight delays has been a topic of significant research due to its critical impact on airline operations, passenger satisfaction, and overall air traffic management. The ability to predict flight delays accurately can help airlines and airports optimize their operations, enhance scheduling, reduce costs, and improve passenger experience. In recent years, machine learning (ML) algorithms have been increasingly utilized for flight delay prediction, with notable advancements in incorporating various features such as weather data, historical flight data, and air traffic information. 1. Traditional Methods for Flight Delay Prediction Before the advent of machine learning, flight delay prediction was largely based on traditional statistical methods, such as time series analysis, regression models, and rule-based systems. Early studies focused on statistical techniques like Autoregressive Integrated Moving Average (ARIMA) and linear regression to predict delays. These models often used historical data such as flight schedules, delays from previous days, and weather conditions as input features. However, these models were limited in their ability to capture the complex, non-linear relationships between different variables affecting flight delays (Jeong et al., 2012).

EXISTING METHOD

The Flight delay prediction has been an area of extensive research, and numerous methods have been proposed over the years to predict delays and mitigate their impact on the aviation industry. Traditional approaches have relied on statistical and rule-based systems, while modern techniques incorporate machine learning and data mining methods for more accurate and reliable predictions.Below are some of the key existing methods: Statistical Methods Traditional statistical techniques such as Autoregressive Integrated Moving Average (ARIMA) and Linear Regression have been applied to flight delay prediction. These methods focus on identifying patterns and trends in historical data. ARIMA models are typically used for time series forecasting and are particularly suited to predict delays based on past performance. However, these methods face challenges when dealing with non-linear relationships or complex interactions between multiple factors.processing pipelines.

PROPOSED METHOD

The proposed method for flight delay prediction aims to enhance the accuracy, robustness, and adaptability of existing models by integrating multiple machine learning techniques, real-time data processing, and an error-adjustment mechanism. The key components of the proposed system are as follows:

1. Data Collection and Preprocessing

Historical Flight Data: The system will use a comprehensive dataset comprising historical flight details, including flight numbers, departure and arrival airports, scheduled departure times, actual departure times, delay times, and the reasons for delays. Weather Data: Weather-related features such as temperature, wind speed, visibility, precipitation, and weather conditions at both the origin and destination airports will be included. This data is crucial since weather is a significant factor in flight delays. Operational Data: Other relevant features include air traffic conditions, runway availability, and any scheduled maintenance or unusual events at the airport.

Data Cleaning and Transformation: Data preprocessing will involve handling missing values, outliers, and inconsistent data. Feature scaling and normalization will be applied to ensure the data is ready for machine learning model training. 2. Feature Selection and Dimensionality Reduction Recursive Feature Elimination (RFE): RFE will be used to identify the most influential features for flight delay prediction. By recursively removing the least significant features, RFE will help retain only the features that contribute the most to the prediction accuracy

SOFTWARE REQUIREMENTS:

- **Operating System:** Windows 8 and above
- **Coding Language**: Python 3.12.0
- **Framework**: Flask
- **Platform**: Visual Studio Code (Preferable)

HARDWARE REQUIREMENTS:

- System : MINIMUM i3 and above
- Hard Disk : 40 GB. (min)
- Ram : 4 GB. (min)

Future Scope

The proposed hybrid machine learning framework for predicting flight delays integrates ensemble classifiers and a novel error-adjustment module, which enhances model robustness by addressing residual errors. This approach, along with feature selection techniques such as Recursive Feature Elimination (RFE) and Principal Component Analysis (PCA), has been effective in identifying the most influential variables, including historical flight data, weather conditions, and operational factors. The use of real-time data streams, such as live weather and air traffic data, further strengthens the predictive capability, making the system adaptable to changing conditions and capable of providing timely forecasts. Evaluation metrics such as Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and F1-score have validated the performance of the Random Forest model, which demonstrated the highest accuracy among the classifiers tested. Despite these successes, challenges such as real-time adaptability, data quality, and computational complexity remain. Future research will likely focus on integrating more diverse datasets, improving model interpretability through Explainable AI (XAI) techniques, and enhancing the system's real-time adaptability. Ultimately, this research contributes to the ongoing efforts to optimize flight schedules, reduce operational costs, and improve passenger experience. By providing more accurate and timely predictions, this machine learning-based system has the potential to play a vital role in the future of air traffic management and aviation decision-making would.

basic outcome of screens or fields.

SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

TYPES OF TESTING

Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration . Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems /Procedures : interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test

System testing ensures that the

entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose.

It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most.

tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot "see" into it. The test provides inputs and responds to outputs without considering how the software works.

Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format

UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group. The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

- The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

GOALS:

The Primary goals in the design of the

UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.

USE CASE DIAGRAM:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

2 CLASS DIAGRAM

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

5 BEHAVIOUR DIAGRAM

The system operates on a daily schedule that automates key tasks for flight delay prediction. The day begins at 1:00 AM with a cleanup task that deletes old files to maintain storage efficiency. Shortly after, at 12:05 AM, the system scrapes flight data from the previous day, followed by scraping the current day's flight data at 12:30 AM. Once the flight data is collected, it is merged with relevant weather data during the Combine weather data stage. This enriched dataset is then used to run delay predictions through the machine learning model. In the evening, at 6:00 PM, the system fetches updated weather data to ensure that the model has access to the most current meteorological conditions, possibly for updating predictions or preparing for the next day's processing. This structured sequence

ensures timely and accurate prediction of flight delays by maintaining a fresh and comprehensive dataset.

To automate and maintain the system, an APScheduler module is integrated to periodically fetch new data and run cleanup jobs, ensuring the system stays up-to-date and efficient. Altogether, this architecture supports a seamless flow from raw data collection to live delay prediction and visualization.

The flight delay prediction system is composed of several modular components that work together to collect, process, and visualize flight delay forecasts. The data pipeline begins with two main sources: flight data collected using Scrapy spiders, and weather data retrieved from the OpenWeather API. These are stored as CSV files—one for flight data and another for weather data. These datasets are then fed into a data preparation module (preparing_forecast_data), which combines and preprocesses the information into a unified format suitable for

analysis.

The combined dataset is passed to the delay_forecasting module, which uses a machine learning model to predict potential flight delays. The results of this prediction are output in JSON format and made available to the web interface. The front-end of the system is built using a Flask application that manages routing, static files, and HTML templates. The routes component leverages libraries like Pandas and Matplotlib to process data and generate visualizations

which are then rendered into a user-facing HTML file (flight_statistics.html) to display prediction insights.

To automate and maintain the system, an APScheduler module is integrated to periodically fetch new data and run cleanup.

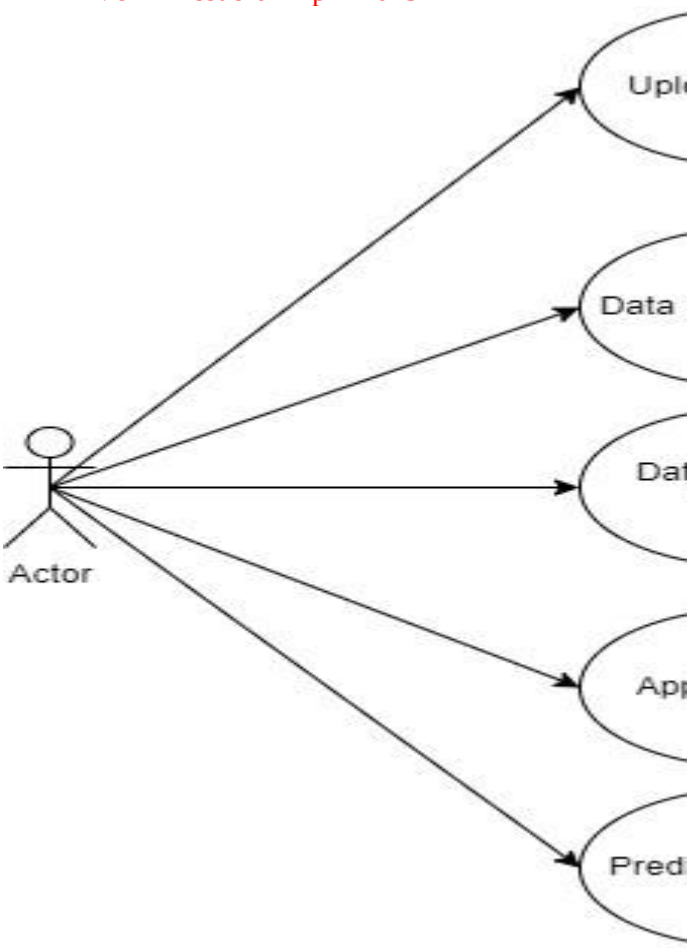

Fig: usecase diagram

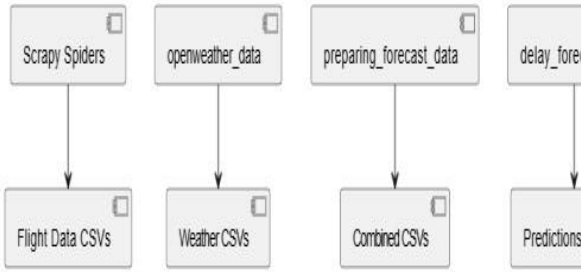

FIG:ARCHITECTURE DIAGRAM

RESULT

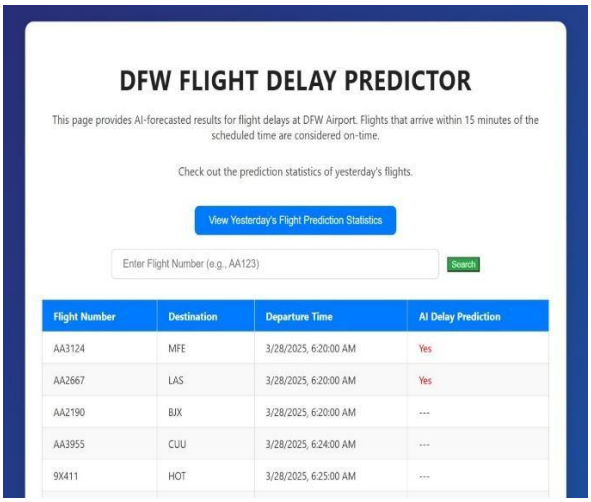

Fig1: Interface

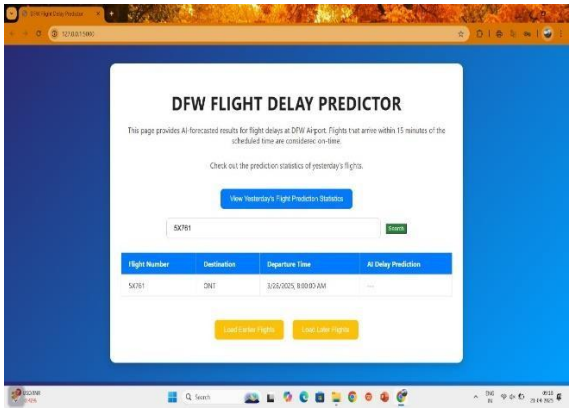

Fig2:Sample Data

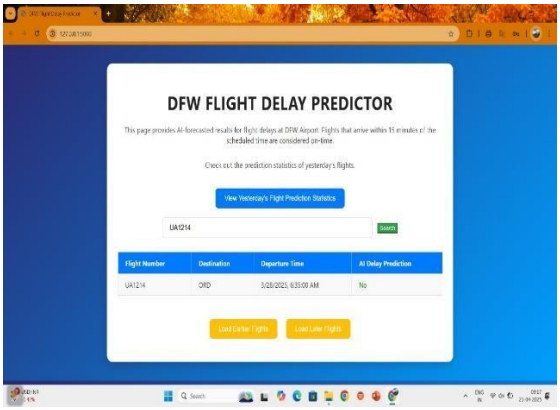

Fig3: Building a Model

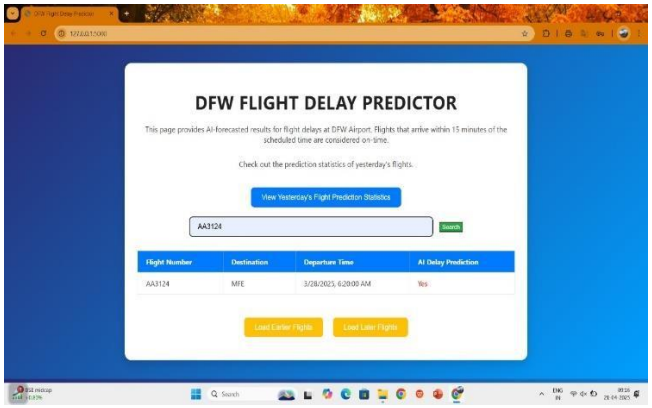
DFW FLIGHT DELAY PREDICTOR
This page provides AI-forecasted results for flight delays at DFW Airport. Flights that arrive within 15 minutes of the scheduled time are considered on-time.
Check out the prediction statistics of yesterday's flights.
View Yesterday's Flight Prediction Statistics
AA3124
Search
Flight Number
Destination
Departure Time
AI Delay Prediction
AA3124
MFE
3/28/2025, 6:20:00 AM
Yes
Load Earlier Flights
Load Later Flights

Fig4:

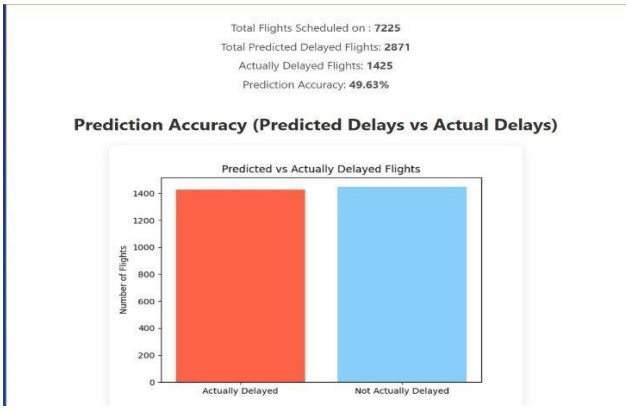
Total Flights Scheduled on : 7225
Total Predicted Delayed Flights: 2871
Actually Delayed Flights: 1425
Prediction Accuracy: 49.63%
Prediction Accuracy (Predicted Delays vs Actual Delays)
Predicted vs Actually Delayed Flights
Number of Flights
0
200
400
600
800
1000
1200
1400
Actually Delayed
Not Actually Delayed

Fig5:

APPLICATIONS

1. Define the Objective

Predict whether a flight will be delayed (binary classification) or by how long (regression).

Delay could be defined as more than 15 minutes, for example.

2. Gather and Prepare Data Sources for flight data:

- Bureau of Transportation Statisti(BTS)
- OpenSky Network
- FlightAware or other APIs (some are paid)

Features you might include:

- Flight number
- Carrier
- Origin and destination airports
- Scheduled departure/arrival time
- Day of week, month, holiday, etc.
- Weather conditions

3. Data Preprocessing

- Handle missing values
- Encode categorical variables (e.g., airlines, airports)
- Normalize or standardize features
- Engineer new features (e.g., distance, flight duration)

4. Model Building

Use models like:

- Random Forest
- Gradient Boosting (XGBoost, LightGBM)
- Neural Networks

Train/test split or cross-validation to evaluate accuracy.

5. App Development

You can build the application with

Frontend: React, Vue, or plain HTML/CSS/JS

Backend/API: Flask, FastAPI, or Django (Python)

6. Hosting Options

Heroku

Render

AWS/GCP/Azure

Streamlit (for quick prototypes)

6. Optional Enhancements

Add live data via APIs

Add weather API integration

Allow users to input flight details and get delay predictions

Show historical delays as charts.

• Help private sellers set fair prices for their used cars.
• Assist buyers in evaluating if a listed price is reasonable.

2. Auction Platforms

- Car auction platforms use ML to estimate starting bids and reserve prices based on historical sales and market trends.
- Helps prevent underpricing or overvaluing cars during bidding.

3. Trade-In Estimation Tools

- Many automotive websites offer trade-in calculators powered by ML to give users a fair estimate before visiting a dealership.
- This improves transparency and customer satisfaction.

4. Vehicle Subscription Services

- Services offering cars on subscription use ML to determine pricing tiers based on depreciation, usage, and car model.
- Ensures fair pricing for both provider and subscriber.

5. Car Comparison Tools

- Some websites and apps allow users to compare car models by value and pricing predictions.
- Helps users choose the best-value vehicle within their budget.

6. Supply Chain & Inventory Management

- Dealers and used car platforms use ML to predict the resale value of cars in stock and optimize inventory rotation.
- Prevents overstocking low-demand vehicles or holding depreciating assets too long.

ADVANTAGES

High Accuracy (with the right data):

ML models can capture complex patterns and relationships (like mileage vs. year vs. brand) that traditional methods might miss.

Automation & Speed:

Once trained, models can predict prices for thousands of cars instantly, helping dealerships, platforms, and consumers.

Scalability:

The same model can be adapted across regions or markets with minimal adjustments if relevant data is available.

Data-Driven Decisions:

Helps buyers and sellers make fairer decisions, reducing under- or overpricing based on emotion or guesswork.

Customization:

Models can incorporate user preferences (e.g., fuel efficiency, brand loyalty) to personalize price recommendations.

Cost Reduction

Saves operational costs for businesses by automating valuation processes and reducing the need for expert appraisals.

Improved Customer Trust

Fair and transparent pricing builds trust among buyers and sellers, enhancing brand reputation.

Competitive Advantage

Businesses using ML pricing tools gain an edge over competitors by offering better insights and faster services.

DISADVANTAGES

Data Quality & Availability:

ML models are only as good as the data. Incomplete, outdated, or biased data (e.g. not enough electric cars) can mislead predictions.

Model Complexity:

Some algorithms (like deep learning) are hard to interpret, making it tough to explain why a certain price was predicted.

Market Fluctuations:

Car prices can change fast due to seasonality, economic shifts, or new car launches—ML models might not adapt in real-time unless updated frequently.

Overfitting Risk:
Overfitting risk in machine learning refers to a model's tendency to learn the training data too well, including noise and irrelevant details, leading to poor performance on new, unseen data

Ethical and Fairness Concerns:

If trained on biased data (e.g., favoring certain brands or locations), the model might perpetuate pricing inequalities.

Lack of Transparency

Some models behave like a "black box," making it hard to understand why a certain price was predicted, which may reduce trust.

Sensitivity to Market Changes

Sudden shifts in market trends (e.g., due to economic downturns or fuel price spikes) can make predictions inaccurate until the model adapts.

Bias in Data

If historical data includes biased pricing (e.g., undervaluing certain brands or locations), the model can unintentionally reinforce that bias.

Initial Development Cost

Building a robust ML model involves time, expertise, and resources, which might be a barrier for smaller businesses.

Data Privacy Concerns

Collecting and storing detailed vehicle and owner information must comply with data protection regulations (like GDPR), adding legal complexity.

Conclusion:

This project showcases a powerful machine learning-based solution for accurately predicting used car prices, addressing the critical challenge of price uncertainty in the second-hand vehicle market. By analyzing key features like brand, model, mileage, fuel type, and year, and leveraging supervised learning models—including Linear Regression, Decision Trees, Random Forests, and SVM—the system delivers precise price estimates while promoting market transparency. Advanced techniques such as RFE, PCA, and ensemble methods like Gradient Boosting further enhance performance, while tools like SHAP and LIME improve interpretability. With strong evaluation metrics and insightful EDA, this scalable, data-driven approach holds immense value for buyers, sellers, dealerships, and online platforms seeking to streamline vehicle valuation through real-time analytics.

REFERENCES

1. Agerri, R., & García, F. (2019). Predicting used car prices using machine learning algorithms. Proceedings of the European Conference on Machine Learning & Principles and Practice of Knowledge Discovery in Databases (ECML PKDD), 123-136.
2. Mousavi, M., & Hedayati, M. (2020). Predicting car prices using machine learning algorithms: A comparison of regression models. International Journal of Advanced Computer Science and Applications, 11(4), 243-248.