Restaurant Rating Prediction System Project Report

by ***Ravi Kumar M***

Table of Contents

Table of Contents

1. Introduction

- The food and restaurant industry is a dynamic and rapidly growing sector that significantly impacts urban life and consumer choices. With the rise of food delivery platforms and online restaurant booking services, customer ratings and reviews play a crucial role in shaping consumer behavior and business success. Predicting restaurant ratings, therefore, has become an essential task for both restaurant owners and consumers to understand customer preferences, improve service quality, and optimize customer satisfaction.

- In this project, we aim to solve the problem of predicting restaurant ratings based on various factors, such as the availability of online orders, booking tables, the number of votes, location, cuisine type, and cost for two people. The idea is to analyze these features and develop a machine learning model that can predict how a restaurant is likely to be rated by customers. This can help restaurants make data-driven decisions to enhance their services and provide a better customer experience.

1. Introduction

- Machine learning is an ideal approach for this problem because it allows us to process large volumes of data, identify patterns, and make accurate predictions. By leveraging machine learning algorithms, we can learn from historical data to predict future outcomes. Traditional methods of manually analyzing customer feedback and business performance are time-consuming and often inaccurate, but with machine learning, we can automate this process to derive meaningful insights efficiently.

- The motivation for developing a web-based system for this project stems from the need for an easy-to-use, accessible platform for both restaurant owners and consumers. A web interface provides a user-friendly way for stakeholders to input data and receive predictions without requiring technical expertise. Furthermore, deploying the system as a web application using Flask allows for seamless integration with existing restaurant management systems and online platforms like Zomato.

1. Introduction

- By building a web-based restaurant rating prediction system, we can empower restaurant owners with the tools to make informed decisions and improve their business strategies. In turn, customers can benefit from more personalized and higher-quality dining experiences. This project demonstrates the power of machine learning in solving real-world business challenges and highlights the importance of technological solutions enhancing service quality in the food industry.

2. Objective

<u>Objective</u>

- The primary objective of this project is to develop a comprehensive system that predicts restaurant ratings based on various input features and provides a user-friendly interface for users to interact with the model. The system is designed to offer insights to restaurant owners about factors influencing their ratings and enable data-driven decision-making.

<u>The specific goals of the project are as follows:</u>

- To build a system : that can accurately predict restaurant ratings using machine learning algorithms based on key features such as location, type of restaurant, availability of online orders, the number of votes, cost for two people, and cuisines.

To create a simple and secure web interface using Flask, where users (restaurant owners, customers, or analysts) can easily interact with the prediction model by providing input data and receiving rating predictions.

2. Objective

- To implement a secure login system without using a database, ensuring basic authentication to allow only authorized users (e.g., admin or managers) to access the prediction tool, thereby safeguarding the system from unauthorized access.

- To optimize the machine learning model by exploring different algorithms, performing cross-validation, and selecting the best-performing model (such as XGBoost with Cross-Validation) to ensure high accuracy and reliability in predicting restaurant ratings.

- To lay the groundwork for future scalability by preparing the system for containerization using Docker and exploring the possibility of deploying the application on cloud platforms (such as AWS or Heroku) for live access and real-time predictions.

These objectives aim to deliver a well-rounded, functional, and scalable system that meets the needs of restaurant owners, customers, and data analysts while enhancing the overall user experience and system performance.

3. Project Architecture

3.1 <u>System Components</u>

- Outline the components in the architecture (Frontend, Backend, Model, Login, etc.).

- In the Restaurant Rating Prediction System, users interact with the system through a web-based interface. The interaction flow starts with the login process, followed by inputting data into the predictor, and concludes with receiving the predicted restaurant rating. Below is a detailed step-by-step description of how users interact with the system.

3. Project Architecture

3.2 Interaction Flow

In the Restaurant Rating Prediction System, users interact with the system through a web-based interface. The interaction flow starts with the login process, followed by inputting data into the predictor, and concludes with receiving the predicted restaurant rating. Below is a detailed step-by-step description of how users interact with the system.

User Interaction Steps:

1. Login Process:

- The user (admin or manager) navigates to the login page.
- The user enters the username and password.
- If the credentials are correct, the user is granted access to the prediction system. If invalid, the system displays an error message: "Invalid credentials! Please try again.

3. Project Architecture

3.2 Interaction Flow

2. Access the Prediction Page:

- Once logged in, the user can navigate to the prediction page by clicking on the Predictlink in the navigation bar.

- The page contains a form where the user can input features like **location, online order availability, table booking option, restaurant type, votes, cost for two people, and cuisines.

3. Input Data for Prediction:

- The user fills in the form with the required information, such as selecting the location, restaurant type, and cuisines from the dropdowns and entering the number of votes or cost for two people.

4. Submit Data:

- After filling in all the required fields, the user clicks the Predict button.

- The system processes the input data, encodes categorical variables using the pre-trained label encoders, and passes the data into the machine learning model.

3. Project Architecture

5. Receive Prediction:

- The model returns the predicted restaurant rating based on the inputs.
- The predicted rating is displayed on the same page, typically under the input form, as Prediction: [Predicted Rating].

6. Logout:

- Once the user is done using the system, they can choose to log out of the system or return to the homepage.

4. Technology Stack

The Restaurant Rating Prediction System utilizes a combination of frontend, backend, and machine learning technologies to build an efficient web-based application. This section details the key components of the technology stack used in the project.

4.1 Frontend Technologies

The frontend technologies form the user interface (UI) that allows users to interact with the system, providing input and receiving predictions.

- **HTML (HyperText Markup Language): HTML is used to structure the content on the web pages, such as forms for user input, buttons, and navigation links. The login page, home page, and prediction page are all built using HTML to display text, input fields, dropdowns, and buttons.**

4. Technology Stack

- CSS (Cascading Style Sheets): CSS is used to style the web pages and make the interface visually appealing. It defines the layout, color schemes, and fonts to ensure a clean, user-friendly experience. For example, CSS is responsible for making the form elements like buttons and input fields easy to navigate and interact with.
- JavaScript (Optional): Although not extensively used in this project, JavaScript can be implemented to add dynamic interactions, such as form validation, enhancing the responsiveness of the UI without reloading the page.

<u>Role in User Interaction:</u>

- Users interact with the system by entering data in HTML forms styled using CSS.
- CSS ensures that the forms and pages are responsive and visually structured, leading to a better user experience.

4. Technology Stack

4.2 Backend Technologies

The backend technologies are responsible for processing user requests, handling the machine learning model, and serving the final predictions.

- Flask (Python Web Framework): Flask is a lightweight web framework written in Python, used to manage the server-side logic of the application. It handles routing, HTTP requests, and integration with the machine learning model. Flask manages the following routes in the project:
    - home(): Loads the homepage for navigation.
    - show_predictor(): Displays the form for user input on the prediction page.
    - predict(): Processes the user's input and returns the predicted restaurant rating.
- Flask also handles rendering templates (HTML files) via the render_template() function and processes POST requests to handle user inputs for prediction.

4. Technology Stack

- **Python: Python is the core programming language used in this project. It is used both for the backend and for running the machine learning model. The Python environment integrates various libraries for preprocessing the data, transforming user input, and interacting with the trained model.**

Role in Backend:

- **Flask serves as the bridge between the user and the machine learning model, managing requests and rendering responses.**
- **Python executes the business logic, including encoding features, loading the model, and returning predictions.**

4. Technology Stack

4.3 Machine Learning Model

The machine learning component is responsible for predicting restaurant ratings based on input features. Various algorithms and libraries have been utilized in this project to build the model.

- **Machine Learning Algorithms:**
  - **XGBoost: The final deployed model was built using the XGBoost algorithm due to its superior performance in prediction accuracy. It is a powerful gradient-boosting algorithm that efficiently handles both classification and regression tasks.**
  - **Cross-Validation (CV): Cross-validation was implemented to ensure the model's robustness and to avoid overfitting during training.**

4. Technology Stack

- Scikit-learn:
  - This Python library was used for preprocessing tasks like label encoding of categorical variables, scaling numerical values, and evaluating model performance.
  - Label Encoders: Encoders were trained using Scikit-learn to convert categorical input (like restaurant type, location) into numerical values, which the machine learning model can process.
- Pandas:
  - Pandas, a Python data analysis library, was used for data manipulation and cleaning. It was essential in handling tabular data, performing operations like transforming input data into the correct format for predictions.

<u>Role in Machine Learning:</u>

- The model is pre-trained and saved as a pickle file, which is loaded during runtime using Flask. When a user provides input, the backend applies label encoding and scaling as required, then passes the data to the XGBoost model for predictions.

5. Dataset Overview

5.1 <u>Source of the Dataset</u>

- Downloaded dataset from Ineuron portal.

5.2 <u>Data Preprocessing</u>

Handling Missing Values: Missing data can lead to inaccurate or biased model predictions. In this dataset, certain columns had missing values, especially for ratings and restaurant details like cost.

- Missing Ratings: Rows with missing values in the target column (ratings) were removed because the absence of the target variable makes it impossible to train the model on those records.

5. Dataset Overview

- Imputation: For columns with missing values in features like "Cost for Two" or "Cuisine," mean or mode imputation was applied, filling in missing values with the most frequent or average value to maintain consistency across the dataset.

Categorical Variable Encoding: Many columns, such as Location, Cuisine, and Type of Service, are categorical in nature. Machine learning models cannot process categorical data directly, so these variables need to be converted into numerical form.

- Label Encoding: Categorical columns were label-encoded using Scikit-learn's LabelEncoder. Each unique value in the categorical columns was assigned a corresponding integer. Example:
    - For the "Online Order" column:
        - "Yes" → 1
        - "No" → 0
- One-Hot Encoding (Optional): In some cases, one-hot encoding could be applied to ensure that each category is represented as a binary column. This prevents the model from mistakenly assigning ordinal significance to the labels.

6. Model Development

6.1 Feature Selection

Feature selection is the process of identifying the most relevant data points (features) that contribute to the accuracy and efficiency of a machine learning model. For the Restaurant Rating Prediction project, the following features were selected for use in the model:

- online_order: Indicates whether the restaurant offers online ordering (Yes/No). This is a crucial feature as online availability can impact customer convenience and, subsequently, the restaurant's ratings.
- book_table: Shows if the restaurant allows table booking (Yes/No). This can be an important factor influencing customer satisfaction, especially in busy or premium locations.
- rate: The restaurant's average rating, which is the target variable we aim to predict. It reflects the overall performance based on customer feedback.
- votes: The number of votes (or reviews) the restaurant has received. More votes usually indicate a higher level of customer engagement.

6. Model Development

- location: The geographical location of the restaurant (e.g., Koramangala, BTM). Location is often a significant factor in determining a restaurant's rating due to demographic differences and customer preferences in different areas.
- rest_type: The type of restaurant (e.g., Casual Dining, Quick Bites). Different types of restaurants serve different customer bases and expectations, influencing ratings.
- dish_liked: The dish that the restaurant is known for. This is a text-based feature that can be transformed into numerical data using natural language processing (NLP) techniques such as TF-IDF (Term Frequency-Inverse Document Frequency) to assess its impact on ratings.
- cuisines: The types of cuisines the restaurant offers (e.g., North Indian, Chinese). The variety and popularity of cuisines can attract different customer groups, which affects the rating.
- approx_cost(for two people): The approximate cost for two people to dine at the restaurant. Price sensitivity often plays a role in how customers rate restaurants.
- listed_in(type): The type of listing (e.g., Delivery, Dine-Out), which shows whether the restaurant primarily focuses on delivery services or dine-in experiences.

6. Model Development

6.2 Label Encoding

Label Encoding is a crucial step when working with categorical features, as machine learning models typically work with numerical values. For this project, label encoding was applied to transform text-based categorical features into numerical representations:

- online_order:
  - 'Yes' → 1
  - 'No' → 0
- book_table:
  - 'Yes' → 1
  - 'No' → 0
- location: Each unique location was assigned a corresponding numerical label (e.g., 'Koramangala' → 1, 'BTM' → 2, and so on).
- rest_type: Similar to location, each restaurant type (e.g., 'Casual Dining', 'Quick Bites') was label-encoded. This allows the model to differentiate between different types of restaurants without imposing any ordinal relationship.

6. Model Development

- Training Process: Various machine learning algorithms were applied to build the predictive model, including:
  - Linear Regression: A simple algorithm that models the relationship between the input features and the target rating.
  - Decision Tree Regressor: A tree-based algorithm that splits the dataset into different nodes based on feature importance.
  - Random Forest Regressor: An ensemble technique that combines multiple decision trees to improve prediction accuracy.
  - Gradient Boosting Regressor: A boosting algorithm that sequentially trains weak models and combines them for a stronger predictive model.
  - XGBoost: A more advanced boosting technique that optimizes performance and accuracy using gradient boosting.
  - XGBoost with Cross-Validation: The most robust model in this project, where cross-validation was applied to further improve the accuracy and reduce overfitting.

6. Model Development

- Evaluation Metrics: After training, the models were evaluated using metrics like:
  - Mean Squared Error (MSE): Measures the average squared difference between predicted and actual ratings.
  - R-squared (R^2): Shows how well the model explains the variance in the target rating. A value closer to 1 indicates a better fit.
- Model Selection: The XGBoost with Cross-Validation model performed the best in terms of both accuracy and generalization. This model was selected for deployment due to its ability to handle complex patterns in the data effectively.

7. Flask Web Application

7.1 Application Structure

The folder structure of the Flask application is organized to separate concerns, making it easier to maintain and develop. Here’s an overview of the typical structure:

```
restaurant_rating_prediction/
│
├── app.py                # Main application file
├── requirements.txt      # File listing project dependencies
├── static/               # Folder for static files (CSS, JS, images)
│   ├── css/
│   │   └── styles.css    # CSS file for styling the application
│   ├── js/
│   │   └── scripts.js    # JavaScript file for interactive elements
│   └── images/
│       └── logo.png      # Logo or any images used in the app
│
├── templates/            # Folder for HTML templates
│   ├── login.html        # HTML template for the login page
│   ├── predict.html      # HTML template for the prediction form
```

7. Flask Web Application

```
│
├── templates/          # Folder for HTML templates
│   ├── login.html       # HTML template for the login page
│   ├── predict.html      # HTML template for the prediction form
│   └── result.html       # HTML template for displaying prediction results
│
└── model/             # Folder for model-related files
    ├── model.pkl         # Trained machine learning model file
    ├── label_encoder.pkl    # Label encoder for categorical features
    └── tfidf_vectorizer.pkl # TF-IDF vectorizer for text features
```

- app.py: The main application file where the Flask app is created, routes are defined, and the model is loaded.
- static/: This folder contains static files like CSS for styling, JavaScript for functionality, and images used in the application.
- templates/: This folder holds the HTML files used to render different pages in the application.
- model/: This folder contains the trained machine learning model and any related files required for making predictions.

7. Flask Web Application

7.2 Route Definitions

In a Flask application, routes are defined to handle incoming requests. Below are the key routes defined in this application:

- **/login:**
  - **Purpose: Displays the login page where users can enter their credentials to access the application.**
  - **Method: GET for rendering the login page, POST for processing the login form submission.**
- **/predict:**
  - **Purpose: Displays the prediction form and receives user input for making predictions. After the user submits the form, it displays the prediction results based on the input features.**
  - **Method: GET for rendering the prediction form, POST for processing the form submission, running the model, and displaying the results.**

7. Flask Web Application

7.3 Frontend Design (Login, Form, Results Page)

The frontend of the application is designed to be user-friendly and visually appealing. Below are descriptions of the key pages:

1. **Login Page (login.html):**
   - **Design: The login page includes a simple form with fields for username and password, along with a submit button. The layout is clean, with clear labels and error messages for invalid logins.**
   - **CSS Styling: Styles are applied to center the form on the page and improve usability.**
2. **Prediction Form Page (predict.html):**
   - **Design: This page contains a form for users to input features such as online order availability, restaurant type, location, etc. Each feature has clear instructions and dropdowns or input fields for easy data entry.**
   - **CSS Styling: The form is styled to be responsive, ensuring it looks good on both desktop and mobile devices.**

7. Flask Web Application

Results Page (result.html):

- **Design: After submitting the prediction form, the results page displays the predicted restaurant rating based on the user input. It includes a summary of the input features and the predicted rating.**
- **CSS Styling: Results are presented in a structured format, with emphasis on the predicted rating and a suggestion for further actions (e.g., trying different restaurants).**

10. Conclusion

Project Summary :

The project aimed to develop a web-based system for predicting restaurant ratings using machine learning. The successful achievement of the project's objectives can be summarized in several key areas:

1. Accurate Prediction Model: The system was designed to predict restaurant ratings based on various input features such as location, restaurant type, online order availability, and the cost for two people. The model utilized advanced machine learning algorithms, including XGBoost, which provided high accuracy in predicting ratings. The inclusion of cross-validation during model training ensured that the model generalizes well to unseen data.

10. Conclusion

2. User-Friendly Web Interface: A secure and intuitive web interface was created using Flask, allowing users to easily interact with the model. The interface included a simple login page for user authentication, followed by a prediction form where users can input the necessary features. This streamlined the process of making predictions, making it accessible even to those without technical expertise.

3. Seamless Interaction Flow: The interaction flow was designed to be straightforward. Users start by logging in, after which they are directed to the prediction form. Upon submitting their input, the system processes the data, runs the prediction model, and displays the results in a clear and organized manner. This ensures that users receive immediate feedback on their queries regarding restaurant ratings.

10. Conclusion

4. Integration of Machine Learning with Web Technologies: By combining machine learning with web development, the project showcased the potential of deploying predictive models in real-world applications. Users can make data-driven decisions based on the predictions generated by the model, thereby enhancing their dining experiences.

5. Future Work and Improvements: The project identified future enhancements, including containerization using Docker for easier deployment and potential migration to cloud platforms for live access. This could improve scalability and accessibility, allowing a broader audience to benefit from the system.

Overall, the project successfully met its objectives by delivering a functional and effective restaurant rating prediction system. The integration of a user-friendly web interface with a robust machine learning model has equipped users with valuable insights into restaurant ratings, helping them make informed dining choices. This success not only demonstrates the feasibility of machine learning applications in the restaurant industry but also opens avenues for further enhancements and features in the future.

11. Future Work

Discuss potential improvements and future work, such as:

- **Integrating a real database for user login and session management.**
- **Using more advanced machine learning models or fine-tuning the current model.**
- **Enhancing the frontend with better UI/UX design.**

12. References

List all the references, tutorials, libraries, and datasets you used in the project. Include:

- Documentation (Flask, Scikit-learn, Pandas, etc.)
- Dataset sources
- Any online tutorials or articles

11. Future Work

Discuss potential improvements and future work, such as:

- **Integrating a real database for user login and session management.**
- **Using more advanced machine learning models or fine-tuning the current model.**
- **Enhancing the frontend with better UI/UX design.**

Thank You