**Zomato Rating Prediction System**

Welcome to the Zomato Rating Prediction System repository! This project focuses on developing an advanced Machine Learning model to predict the ratings of restaurants listed on Zomato, a popular food delivery platform. This README.md file provides an overview of the project and guides you on how to use the system effectively.

**Table of Contents**

* [About](https://github.com/code-with-praveen143/ineuron_internship_project#about)
* [Features](https://github.com/code-with-praveen143/ineuron_internship_project#features)
* [Dataset](https://github.com/code-with-praveen143/ineuron_internship_project#dataset)
* [Technologies Used](https://github.com/code-with-praveen143/ineuron_internship_project#technologies-used)
* [Usage](https://github.com/code-with-praveen143/ineuron_internship_project#usage)
* [Installation](https://github.com/code-with-praveen143/ineuron_internship_project#installation)
* [Model Training](https://github.com/code-with-praveen143/ineuron_internship_project#model-training)
* [Evaluation and Prediction](https://github.com/code-with-praveen143/ineuron_internship_project#evaluation-and-prediction)
* [Contributing](https://github.com/code-with-praveen143/ineuron_internship_project#contributing)
* [License](https://github.com/code-with-praveen143/ineuron_internship_project#license)
* [Contact](https://github.com/code-with-praveen143/ineuron_internship_project#contact)

**About**

The Zomato Rating Prediction System aims to provide accurate predictions of restaurant ratings using Machine Learning algorithms. By analyzing various factors such as cuisine, location, pricing, ambience, and customer reviews, the system leverages a trained model to predict ratings, enabling users to make informed decisions when selecting restaurants.

**Features**

* Predicts the ratings of restaurants listed on Zomato.
* Provides insights into factors influencing restaurant ratings.
* Offers a data-driven approach for making informed dining decisions.
* Helps restaurant owners identify areas for improvement based on predictive analytics.

**Dataset**

The system is trained on a comprehensive dataset that includes restaurant attributes, customer reviews, and historical ratings. However, please note that the dataset used in this project is not included in this repository due to licensing restrictions. You will need to obtain the dataset separately and ensure its compatibility with the provided code.

**Technologies Used**

The Zomato Rating Prediction System is built using the following technologies and tools:

* Python: Programming language used for development.
* Machine Learning Libraries: Scikit-learn, TensorFlow, or PyTorch (choose based on your preference) for building and training the ML model.
* Jupyter Notebook: Interactive environment for model development and experimentation.

**Usage**

To use the Zomato Rating Prediction System, follow these steps:

1. Clone or download this repository to your local machine.
2. Install the necessary dependencies mentioned in the requirements.txt file.
3. Obtain the dataset compatible with the system.
4. Launch Jupyter Notebook and open the provided Restaurant\_Ratin\_Prediction\_System.ipynb file.
5. Execute the code cells in the notebook sequentially, following the instructions and comments.
6. Customize the model and parameters as per your requirements.

Note: It is advisable to have a good understanding of Machine Learning concepts and libraries to effectively use and modify the Zomato Rating Prediction System.

**Installtion**

he Code is written in Python 3.8.8. If you don't have Python installed you can find it here. If you are using a lower version of Python you can upgrade using the pip package, ensuring you have the latest version of pip. To install the required packages and libraries, run this command in the project directory after cloning the repository:

pip install -r requirements.txt

Run on your Local Machine :

flask run app.py

This will start the run the server.py which will also trigger code for server\_app.py because of the use of asynchronous execution (threading) and will connect our ML model to Anvil application UI and will keep the server running till the web page rendered by flask application gets closed but to keep the server running forever, we used the heroku cloud to run our server continuously.

**Model Training**

The model training process involves loading the dataset, preprocessing, feature engineering, and training the ML model. The code provided in the Jupyter Notebook guides you through each step and includes necessary explanations and comments for better understanding. Feel free to experiment with different models and hyperparameters to achieve optimal results.

**Evaluation and Prediction**

After training the model, the notebook provides sections for evaluating the model's performance and making predictions. Use the evaluation section to assess the accuracy and performance metrics of the trained model. In the prediction section, you can apply the model to new restaurant data to predict their ratings.

**Contributing**

Contributions to the Zomato Rating Prediction System are welcome! If you would like to contribute, please follow the guidelines outlined in the CONTRIBUTING.md file. Your contributions can include bug fixes, enhancements, or new features that improve the functionality or performance of the system.

**License**

The Zomato Rating Prediction System is released under the [MIT License](https://github.com/code-with-praveen143/ineuron_internship_project/blob/master/LICENSE).

**Contact**

For any inquiries or questions regarding the Zomato Rating Prediction System, please feel free to reach out to me at [your contact information]. I'm open

**What techniques were you using for data pre-processing?**

* Removing unwanted attributes
* Visualizing relation of independent variables with each other and output variables
* Checking and changing Distribution of continuous values
* Removing outliers
* Cleaning data and imputing if null values are present.
* Converting categorical data into numeric values.
* Scaling the data

**How training was done or what models were used?**

Data pipeline was created to implement data scaling, one-hot encoding and an estimator to prevent any data leakage. CatBoost model was used as the best estimator which was then used for production followed by hyperparameter tuning.

**What are the different stages of deployment?**

* When the pipeline is ready, we connected the front-end made on Anvil with the backend code having the pipeline on the local machine via anvil uplink.
* A server is created in Flask which just displays a single web page and runs the uplink code via threading thereby creating a server.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.