Model reporting

Introduction

Objective: The aim is to predict whether a restaurant accepts online orders (online_order) based on various features like location, cuisine, rating, and others. The dataset used in this study is derived from a restaurant listing platform, containing detailed information about various restaurants in Bangalore. The dataset provides insights into customer preferences, restaurant characteristics, and their service offerings, with a focus on predicting whether a restaurant offers **online ordering** (online_order).

Dataset Overview: The dataset contains 51,717 entries with features like address, ratings, votes, cost, cuisines, and more. The target variable is categorical (Yes or No for online orders). This dataset is valuable for analyzing customer behavior and restaurant performance. Predicting online ordering availability can help restaurants optimize their services and better cater to customer needs.

Implementation part.

The implementation begins by importing necessary libraries for data analysis, visualization, preprocessing, and model evaluation.

During the data preprocessing step, the unique values in the online_order and book_table columns were analyzed. Both columns contained only two categories: Yes and No. o prepare the dataset for modeling, the categorical values in these columns were mapped to integers:

Yes → 1

No → 0

The columns were then converted to integer format to ensure compatibility with machine learning algorithms.

The **rate** column initially contained non-numerical values (e.g., NEW, -, and /5 suffix). These were cleaned by:

Removing the /5 suffix and replacing invalid entries ('NEW', '-') with None (NaN). Converting the column to a float type. Filling missing values with the mean of the column. This step ensures that the rate column is numerical and ready for modelling

The B column contained multiple formatting issues, including entries with line breaks and different phone number formats:

The unique values in the phone column included phone numbers with mixed formats such as:

'080 42297555\r\n+91 9743772233'

'+91 9663487993'

'+91 9663517066\n+91 9686861135'

Since the phone column is not important for the prediction task, it was decided to remove it entirely. The column had multiple inconsistencies and cleaning it was not deemed necessary, as it does not contribute to the prediction model. Removing irrelevant or redundant features is a common preprocessing step to improve model efficiency.

Categorical variables in the dataset were identified and encoded into numerical values using the **LabelEncoder**. This transformation was applied to all columns with object or category data types to ensure compatibility with machine learning algorithms. Each unique category was replaced by an integer, allowing for seamless model training.

The numerical features were standardized using **StandardScaler** to ensure all features have a mean of 0 and a standard deviation of 1. This scaling was applied to the training, validation, and test datasets to improve model performance and convergence. The scaler was fitted on the training data and then used to transform the validation and test data.

The Logistic Regression model was trained on the scaled training data with a maximum of 1000 iterations and the 'lbfgs' solver. The model's performance was evaluated on both the validation and test datasets:

Validation Accuracy: 73.34%

Test Accuracy: 73.34%

These results indicate the model's ability to generalize to unseen data, with a relatively strong performance on both validation and test sets.

The classification report for the Logistic Regression model on the test dataset is as follows:

Precision:

Class 0: 0.65, Class 1: 0.80

Recall:

Class 0: 0.71, Class 1: 0.75

F1-Score:

Class 0: 0.68, Class 1: 0.77

Accuracy: 73.34%

Macro Average:

Precision: 0.72, Recall: 0.73, F1-Score: 0.73

Weighted Average:

Precision: 0.74, Recall: 0.73, F1-Score: 0.74

The model performs better in predicting the positive class (1) with higher precision and recall, while the negative class (0) shows lower performance.

Test Confusion Matrix:

The confusion matrix shows the model's ability to distinguish between classes, with a higher number of true positives for Class 1.

Cross-Validation: The final model achieved a cross-validation accuracy of 72.86% across five folds.

The tuned model demonstrates improved performance compared to the initial model, with better classification metrics and higher overall accuracy on the validation set.

To improve the model's performance, regularization was applied by tuning the C parameter using GridSearchCV. The grid search explored different values of C and solvers, and the best parameters were found as:

Best Parameters: C=100, solver='lbfgs'

Using the best model from the grid search, the accuracy on the test data was evaluated:

Test Accuracy: 51.51%

Although the model was tuned for regularization, the performance on the test set is lower than expected, indicating that further optimization or feature adjustments might be needed.

To evaluate the performance of different models, the following classifiers were trained and tested on the validation set:

Decision Tree Classifier: Achieved a validation accuracy of 85.98%.

Random Forest Classifier: Achieved the highest validation accuracy of 92.32%.

K-Nearest Neighbors Classifier: Achieved a validation accuracy of 79.71%.

Among the models tested, Random Forest performed the best, demonstrating the highest accuracy on the validation set, followed by Decision Tree. K-Nearest Neighbors showed relatively lower performance. Based on these results, Random Forest is the most suitable model for this task.

Model Comparison and Conclusion

After evaluating multiple models and performing cross-validation, the following results were observed:

Decision Tree Classifier:

Mean Cross-Validation Score: 84.46%

Decision Trees provided decent performance but showed slightly less stability compared to Random Forest.

Random Forest Classifier:

Mean Cross-Validation Score: 91.21%

Random Forest performed the best overall, with the highest cross-validation score, indicating strong performance and generalization on the training data.

K-Nearest Neighbors Classifier:

Mean Cross-Validation Score: 78.74%

KNN performed the worst among the models tested, with a lower mean score and less consistency.

Final Conclusion:

Based on the cross-validation results, Random Forest is the best-performing model for this dataset. It consistently outperforms the other models (Decision Tree and KNN) in terms of accuracy and stability. Therefore, Random Forest was selected as the final model for this task.