Model Optimization and Tuning Phase Template

Date	6 June 2025
Name	Prajakta Arun Patil
Project Title	Restaurant Recommendation System
Maximum Marks	10 Marks

Model Optimization and Tuning Phase

The Model Optimization and Tuning Phase involves improving our machine learning recommendation model to get the best performance. This includes adjusting the model's parameters, experimenting with different algorithms, and selecting the most suitable model based on evaluation metrics such as accuracy, precision, recall, and RMSE (Root Mean Squared Error). Our restaurant recommendation system was designed to suggest similar restaurants based on location, user ratings, cuisines, and cost using collaborative filtering and content-based filtering techniques.

Hyperparameter Tuning Documentation (8 Marks):

Model	Tuned Hyperparameters
Model 1: Content-Based Filtering	- **Similarity Metric:** Cosine similarity was used as the primary metric to compute similarity between restaurants based on features like cuisines, rating, and cost. - **Top N Recommendations:** The number of top similar restaurants returned was tested with values like 5, 10, and 15.

```python
1   def recommend(name, cosine_similarities = cosine_similarities):
2
3       # Create a list to put top restaurants
4       recommend_restaurant = []
5
6       # Find the index of the hotel entered
7       idx = indices[indices == name].index[0]
8
9       # Find the restaurants with a similar cosine-sim value and order them from bigges number
10      score_series = pd.Series(cosine_similarities[idx]).sort_values(ascending=False)
11
12      # Extract top 30 restaurant indexes with a similar cosine-sim value
13      top30_indexes = list(score_series.iloc[0:31].index)
14
15      # Names of the top 30 restaurants
16      for each in top30_indexes:
17          recommend_restaurant.append(list(df_percent.index)[each])
18
19      # Creating the new data set to show similar restaurants
20      df_new = pd.DataFrame(columns=['cuisines', 'Mean Rating', 'cost'])
21
22      # Create the top 30 similar restaurants with some of their columns
23      for each in recommend_restaurant:
24          df_new = df_new.append(pd.DataFrame(df_percent[['cuisines','Mean Rating', 'cost']][df_percent.index == each].sample()))
25
26      # Drop the same named restaurants and sort only the top 10 by the highest rating
27      df_new = df_new.drop_duplicates(subset=['cuisines','Mean Rating', 'cost'], keep=False)
28      df_new = df_new.sort_values(by='Mean Rating', ascending=False).head(10)
29
30      print('TOP %s RESTAURANTS LIKE %s WITH SIMILAR REVIEWS: ' % (str(len(df_new)), name))
31      df_new.index = df_new.index.str.lower()
32      return df_new
```

Model 2:

Collaborative

Filtering

- **Algorithm:** SVD (Singular Value Decomposition) from the Surprise library.
- **Learning Rate:** Tuned values such as 0.005, 0.01, and 0.02 were tested.
- **Regularization:** Parameters such as 0.02, 0.05 were tried to avoid overfitting.
- **Number of Epochs:** Adjusted between 20 and 100 epochs.

```python
1   from surprise import SVD, Dataset, Reader
2   from surprise.model_selection import cross_validate
3
4   reader = Reader(rating_scale=(1, 5))
5   data = Dataset.load_from_df(df[['user_id', 'restaurant_name', 'rating']], reader)
6
7   svd = SVD()
8   cross_validate(svd, data, measures=['RMSE', 'MAE'], cv=5, verbose=True)
```

Final Model Selection Justification (2 Marks):

Final Model	Reasoning
Model 1: Content-Based Filtering	Selected due to its simplicity and good performance without requiring detailed user history. It gave interpretable and relevant results using restaurant features like cuisines, ratings, and cost.