Assignment 1

1 Objectives & Workflow

In this assignment, our objective is to build a predictive model to determine the 'price per night' for Airbnb properties in Belgium. We're working with a dataset that includes 55 variables, along with the target variable - the price. Our approach employs a range of machine learning techniques to optimize model training, such as LightGBM, a Neural Network model featuring stochastic gradient descent (SGD), XGBoost, Support Vector Regression (SVR), and Random Forest. This diverse set of methodologies enhances the robustness of our analysis. During the preprocessing stage, we have adopted several techniques such as the K-Nearest Neighbors (KNN) method for imputing missing values and transforming new variables. Furthermore, we employ SHAP values to interpret the effect of each variable within our models.

2 Preprocessing & Exploratory Data Analysis

Preprocessing and exploratory data analysis do not occur in a linear sequence. Rather, these processes are iterative and cyclical, involving a constant back-and-forth between the two stages. We put the table below to show what we do in this section.

2.1Preprocessing and Feature Selection

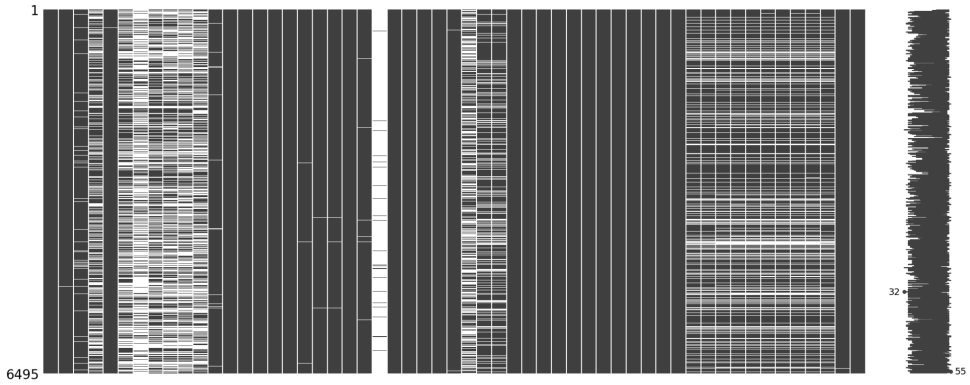
Variable	**What we do**
property_rules, property_zipcode, property_lat, property_lon, property_type, property_room_type, property_max_guests, property_bathrooms, property_bedrooms,	1. removed ‘property_rules’ since It is so massive 2. removed ‘property_zipcode’ since it is nom-informative 3. clustering using ’property_lat’ and ‘property_lon’, calculate the distance betweeneach location and its central of cluster 4. encoded ‘property_type’,’’property_room_type’ as nominal 5. filled new values using mean/ mode with a little number of missing values forvaroables ‘property_bathrooms’, and ‘property_bedrooms’
reviews_acc, reviews_cleanliness, reviews_checkin, reviews_communication, reviews_location, reviews_value, reviews_per_month	3-nearest neighbors imputation
Host Is Superhost, Host Has Profile Pic, Host Identity Verified, Is Location Exact, Instant Bookable, Require Guest Profile Picture, Require Guest Phone Verification	one-hot encode for ‘Extra’ variable

host_about, hsost_response_time, host_response_rate, host_nr_listing, host_nr_listings_total, host_verified, booking_price_covers, booking_min_nights, booking_max_nights, property_beds	1. remove ‘host_about’ due to noisy text and more than 3000 missingness. 2. impute missingness in ‘host_nr_listing’ and ‘host_nr_listings_total’ with median. 3. create a new column ‘logit_host_verified’ by recognizing the government id in ‘host_verified’ and the data type of the new column is boolean. 4. remove original ‘host_verified’. 5. encode ‘host_respinse_time’ with ordinal encoder. 6. impute missingness in the rest variables with 5-nearest neighbor imputation. 7. create a new column ‘extra_beds’ by ‘property_beds’ indicating the possibility of adding extra bed.
booking_availability_30 booking_availability_60 booking_availability_90 booking_availability_365 reviews_num reviews_last reviews_rating booking_cancel_policy	1. 4-nearest neighbors imputation for r’eviews_num’ ‘reviews_last’,’reviews_rating’,’booking_cancel_policy’ 2. Remove ‘reviews_last’ because ‘reviews_cleanliness’ has the same effect and more effective.
property_bed_type property_amenities property_sqfeet property_scraped_at property_last_updated host_id host_since host_location	1. remove ‘property_bed_type’ with creation of new features ‘Airbed’’Couch’’Futon’’Pull-out Sofa’’Real Bed’ 2. remove ‘property_amenities’ with creation of new feature according to the exact amenities 3. remove ‘property_sqfeet’ since it has lots of missing values 4. remove‘host_id’ 5. remove ‘host_location’ and create new features ’host_location_belgium_brussels’’host_location_belgium_flanders’’host_location_belgium_other_regions’’host_location_other_countries’

2.2 Exploratory Data Analysis

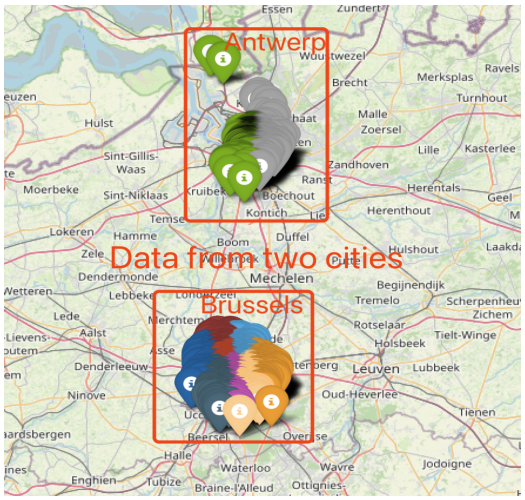
2.2.1 Missing Value Matrix of All Data

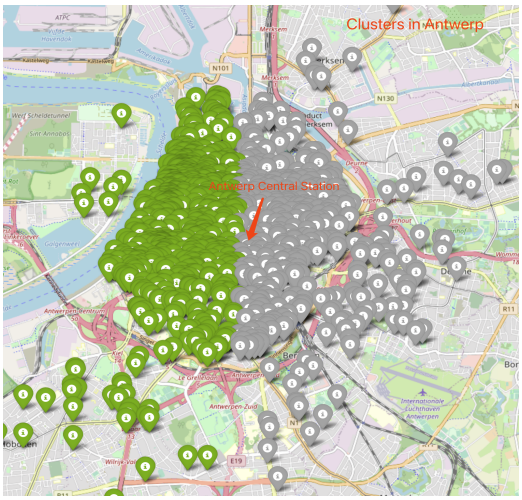
The review-related variables have similar missing patterns.

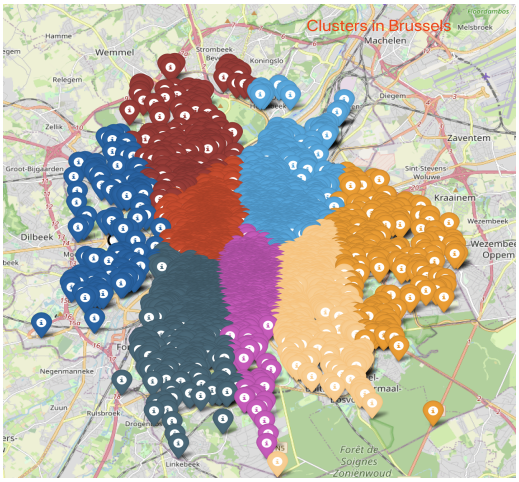


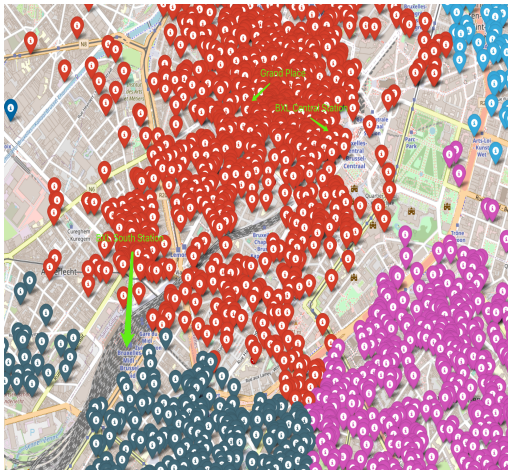
2.2.2 Clusters of All Data Points (House Location)

K-Means Clustering is used here on the 'property_lat' and 'property_lon' features to identify geographical concentrations. This revealed that our data primarily clusters around Brussels and Antwerp. Notably, the clustering differentiates Antwerp into two groups separated by the Central Station and City Park, and divides Brussels into eight sections reflecting distinct commercial and physical boundaries. These findings substantiate our clustering approach, implying that geographical location, represented by different clusters, may influence the property prices.









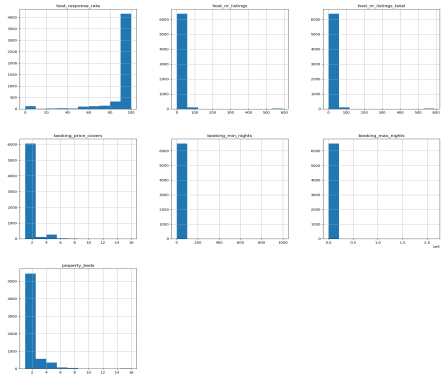
2.2.3 review-relevant variables

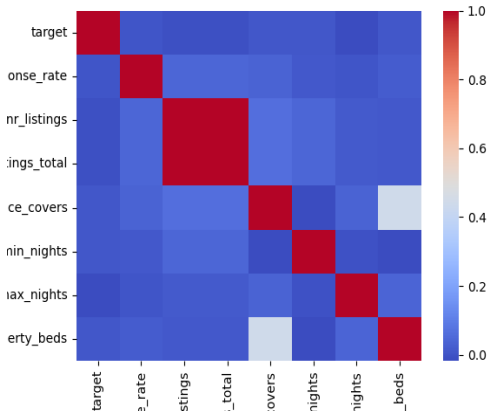
- The review-relevant variables are all skewed.
- The accuracy and cleanliness review scores are linearly-correlated to the overall review value. The communication review score is linearly-correlated to the check-in review score.

2.2.4 **Data exploration of host and booking related features**

It can be seen from the histogram of features that the ‘response_rate’ is left skewed. Most of the host repose rate achieve 100%. Furthermore, the distributions of 'host_nr_listing' and 'host_nr_listing_total' appear to be similar, suggesting that instances under these two features have the same values. Additionally, it seems that histograms of 'booking_min_nights' and 'booking_max_nights' demonstrate consistent patterns among hosts.

The correlation between the numeric variable and target are investigated. The plot of the correlation matrix shows slight direct correlation between target and those numeric features. Notably, a perfect correlation was identified between 'host_nr_listing' and 'host_nr_listing_total,' indicating that the data for these two features is identical.





3 Model Training & Model Evaluation with Different Algorithms

3.1 Random Forest

We use Random Forest as the model and perform 5-fold cross-validation. Through cross-validation, we are able to evaluate the model and find the optimal parameter combination {'max_depth': 80, 'min_samples_leaf': 5, 'min_samples_split': 12, 'n_estimators': 1000} to achieve the best performance.

3.2 Neural Network

A Neural Network model was developed using a MLPRegressor with hidden layer sizes (40,10), 'ReLU' activation, and 'sgd' solver. Stochastic gradient descent optimizes the model by iteratively minimizing the error function using a randomly selected data point. The data was divided into training and testing subsets (70%/30% split) and the model was trained for a maximum of 200 iterations. Performance was evaluated via RMSE, MAE.

3.3 Support vector regression

A support vector regression (SVR) model was constructed. The dataset was divided into training and test data, with 80% allocated to training and 20% to testing. Recursive feature selection was applied

to the training data to select features with high weights. To determine the optimal model, 10-fold cross-validation was performed. The SVR model with an RBF kernel, gamma of 1, and C of 10 was selected.The model performance is evaluated with RMSE and MAE.

3.4 XGBoost

The XGBoost algorithm was implemented to build a predictive model. XGBoost utilizes an ensemble of decision trees, with each subsequent tree attempting to correct the mistakes made by the previous trees.This iterative process continues until the model achieves optimal performance.The dataset was divided into five equal-sized subsets through 5-fold cross-validation. To evaluate the performance of the model, tthe average RMSE and MAE scores are calculated by dividing the sum of the scores by the number of folds.

3.6 Decision Tree

Starting from the root node, the best feature is chosen as the splitting criterion for the current node.Assuming review_rating is chosen as the first splitting criterion, the dataset is divided into subsets based on different value of rating.Termination Conditions:At each child node, a termination condition can be defined, such as a subset containing a sample size smaller than a certain threshold or a subset with similar price ranges.If the termination condition is met, further splitting stops at that branch, and the node becomes a leaf node. The price range of the subset is used as the predicted value at the leaf node.Tree Pruning:The constructed decision tree may suffer from overfitting. Pruning techniques can be employed to trim the decision tree and improve its generalization ability.Prediction:When a new hotel sample arrives, it is classified step by step based on the decision tree's splitting criteria until a leaf node is reached.At the leaf node, the price range associated with that leaf node is used as the prediction for the new sample's price.

4 Summary

Here is summary table of our model results using different algorithm, and we got top1 in leaderboard result in the final Leaderboard. (group 21).

Table 1. Performance with different methods

No	Method	Train Data		Test Data(Leaderboard)	
		RMSE	MAE	RMSE	MAE
1	Random Forest	50.44	30.86	58.59	32.65
2	Neural Network	49.83	29.56	46.74	27.78
3	lightGBM				
4	XGBoost	55.59	34.64	56.32	36.38
5	SVR	59.98	29.94	48.00	27.08
6	DecisionTree	60.11	35.53	60.69	33.23

Assignment 2

1 Objectives & Pipeline

In this assignment, we aimed to predict a restaurant's cuisine type from its image. We processed a dataset of over 117,000 images from 16,000+ Michelin Guide restaurants. The cuisine types were simplified into four primary categories, 'Japanese', 'Italian', 'Chinese', 'French', and 'Other' category. Each cuisine was then associated with its images, which were resized and stored locally. Using this processed dataset, *Keras* is used and Convolutional Neural Network (CNN) was developed for image classification. Furthermore, the model was deployed via a Flask application. Pipline is below:

Step 1: Data Preprocessing

- Convert JSON structure into a pandas DataFrame.
- Check the descriptive distribution of all cuisines.
- Merge and relabel sub-cuisines for four susines individually.
- Check distribution of four cuisines -> Unbalanced -> Undersampling Employed.
- Split training and test data in 70% and 30%

Step 2: Model Training and Test

01. Evaluation metrics

For our model, accuracy and AUC-ROC are selected. Accuracy determines the percentage of accurate predictions the model makes. Due to the new balance of our dataset, which was constructed using undersampling during step1 preprocessing and provides nearly equal representation for each cuisine, we particularly picked accuracy. We employed the AUC-ROC curve, a performance indicator for multi-class classification issues at different threshold settings. A confusion matrix could offer a thorough assessment of the model's performance when applied to our multi-class classification task, showing which classes are most frequently confused with one another. Our task's scope and dataset led us to conclude that the accuracy and AUC-ROC combination is best.

02. CNN model and the settings

a. Model Selection: Sequential model was adopted since its simplicity and efficiency in linear stacking of layers, which perfectly fits our application having a single input and output tensor. More complex structures involving multiple inputs/outputs or shared layers would necessitate a more flexible approach like the Functional API.

b. Layer Selection & Parameters: The architecture includes Conv2D layers for feature extraction, MaxPooling2D for down-sampling, Flatten to convert matrices into single vectors, Dense for decision making, and Dropout for regularization. The parameters were chosen based on empirical tuning. Conv2D (32, 64, 128 filters with 3x3 kernels) + MaxPooling2D: Helps in detecting hierarchical features and reducing spatial dimensions to control overfitting. Flatten + Dense (128 neurons): Transforms 2D feature maps to 1D for decision making. Dropout (0.5): Prevents overfitting by ignoring randomly selected neurons during training, promoting generalization.

c. Activation Functions: 'relu' is used in intermediate layers since it shows a good results, and 'softmax' in the final layer, which is Perfect for our multi-class classification problems (4 cuisines + 1 'other').

d. Regularization (L2 & Dropout): L2 regularization with lambda of 0.01 and 0.005, along with dropout of 0.5, are applied to prevent our model's overfitting and ensure our model generalization.

03. ResNet50 model and the settings

a. Model section: In the deep structure of CNNs, as the network layers deepen, the issues of gradient vanishing and gradient explosion arise, resulting in a decline in accuracy on the training set. This problem can be addressed by utilizing residual networks, which enhance the network's performance while increasing its depth.

b. Architecture of ResNet50: ResNet-50 is composed of 50 layers, including a combination of convolutional layers, pooling layers, fully connected layers, and shortcut connections (also known as skip connections or residual connections). The architecture follows a "bottleneck" design, which reduces the computational complexity of the network by using 1x1, 3x3, and 1x1 convolutional layers in a specific order. The use of shortcut connections helps to alleviate the vanishizg gradient problem and allows the network to learn more effectively.

c. ResNet-50 is a deep neural network architecture comprising 49 convolutional layers and 1 fully connected layer. It takes input images of size 224x224 pixels. The architecture consists of five stages:

Stage 1: This initial stage includes a 7x7 convolutional layer with 64 filters, followed by a 3x3 max-pooling layer, which captures low-level features.

Stage 2: Downsampling is performed using a 1x1 convolutional layer with stride 2 to reduce spatial dimensions. It is followed by residual blocks, each composed of three 3x3 convolutional layers with 64 filters.

Stage 3: Similar to Stage 2, it begins with downsampling and contains residual blocks with four 3x3 convolutional layers. The first layer uses 1x1 filters to reduce and then increase dimensions, while the remaining layers have 3x3 filters. Each block has 128 filters.

Stage 4: This stage follows the pattern of Stage 3, with additional residual blocks. Downsampling is performed using a 1x1 convolutional layer with stride 2. Each block consists of four 3x3 convolutional layers and has 256 filters.

Stage 5: The final stage starts with downsampling using a 1x1 convolutional layer with stride 2. It includes three residual blocks, each comprising four 3x3 convolutional layers. Each block has 512 filters.

d. Ways used to prevent overfitting: ResNet-50 employs multiple techniques, including batch normalization, which normalizes the output activations of each convolutional layer; dropout, a regularization technique that stochastically deactivates a fraction of neuron connections in the fully connected layers; and data augmentation, which entails applying random transformations, such as scaling, to the training data.

04. Models Test

a. After model training with 70% data, the model test is conducted with remaining 30% data. The model.*evaluate()* function of the Keras is used to assess the model performance on the test data. This function returns the loss value and accuracy values for the model in test mode. As well as, the *roc_curve()* function to output

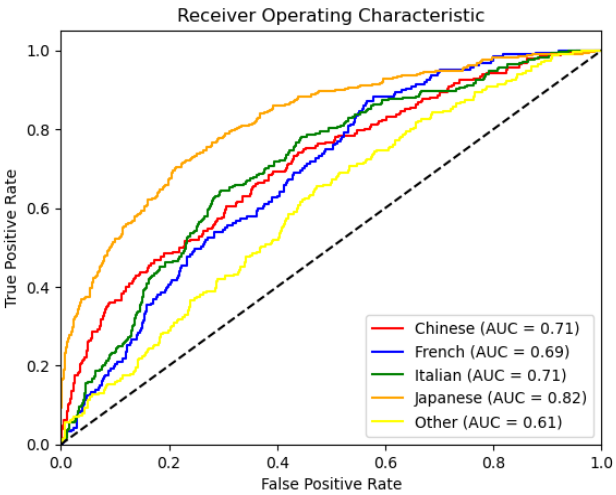
AUC scores provides an aggregated measure of the model's capability to distinguish between classes across various threshold levels.

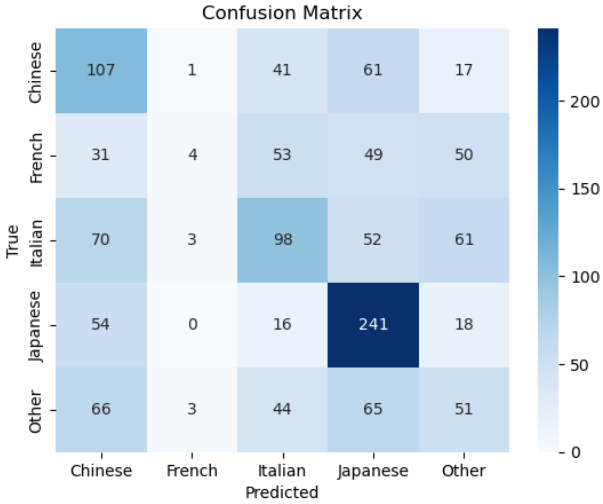
b. The ResNet50 model underwent an initial training phase of 15 epochs, during which the highest validation accuracy was achieved and preserved for further analysis. This selected model was then subjected to an additional 15 epochs of training.Throughout the training process, the model achieved a training loss of 1.1678 and exhibited a training accuracy of 50.14% on the training dataset. Evaluation using the validation dataset revealed an accuracy of 42.6%. The validation loss was measured at 1.4060, providing insight into the average error between predicted and actual values.

c. Based on the results obtained from the evaluation metrics such as AUC-ROC and the confusion matrix on the validation dataset, it is evident that the model exhibits excellent performance in classifying Japanese cuisine, followed by Chinese and Italian cuisine. However, the model shows relatively weaker performance in accurately classifying French cuisine. Other cuisines may originate from different countries or regions like the label 'contemporary', which can create confusion in classification. These results suggest that the model has room for improvement, as both the training and validation accuracies are relatively low.

Step 4: Model Deployment

- Flask is chosen for deployment due to its simplicity, flexibility, and compatibility with Python. This lightweight WSGI web application framework is also used here since Colab can not give a local web service and we need to use third-part server to get the deployed webpage.
- Once deployed, the model could receive images as inputs via an web page(HTTP request), classify the cuisine type of the provided image, and return the classification results (the type of cuisine, with a probability).



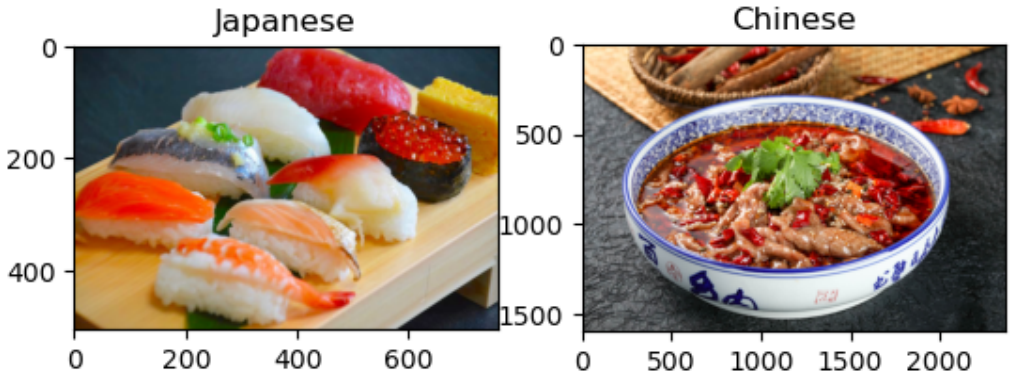


2 Model Results and Predictions

```
1/1 [==============================] - 0s 118ms/step
Predicted class: Chinese
```
Label in test data : Chinese

```
1/1 [==============================] - 0s 85ms/step
Predicted class: Japanese
```
Label in test data : Japanese

We selected two food images from the website to demonstrate the predictions made by the trained model. The accompanying figures depict typical Japanese and Chinese cuisine, along with the corresponding prediction label displayed in the title.



3 Model Interpretations (A CNN Case in Our Model)

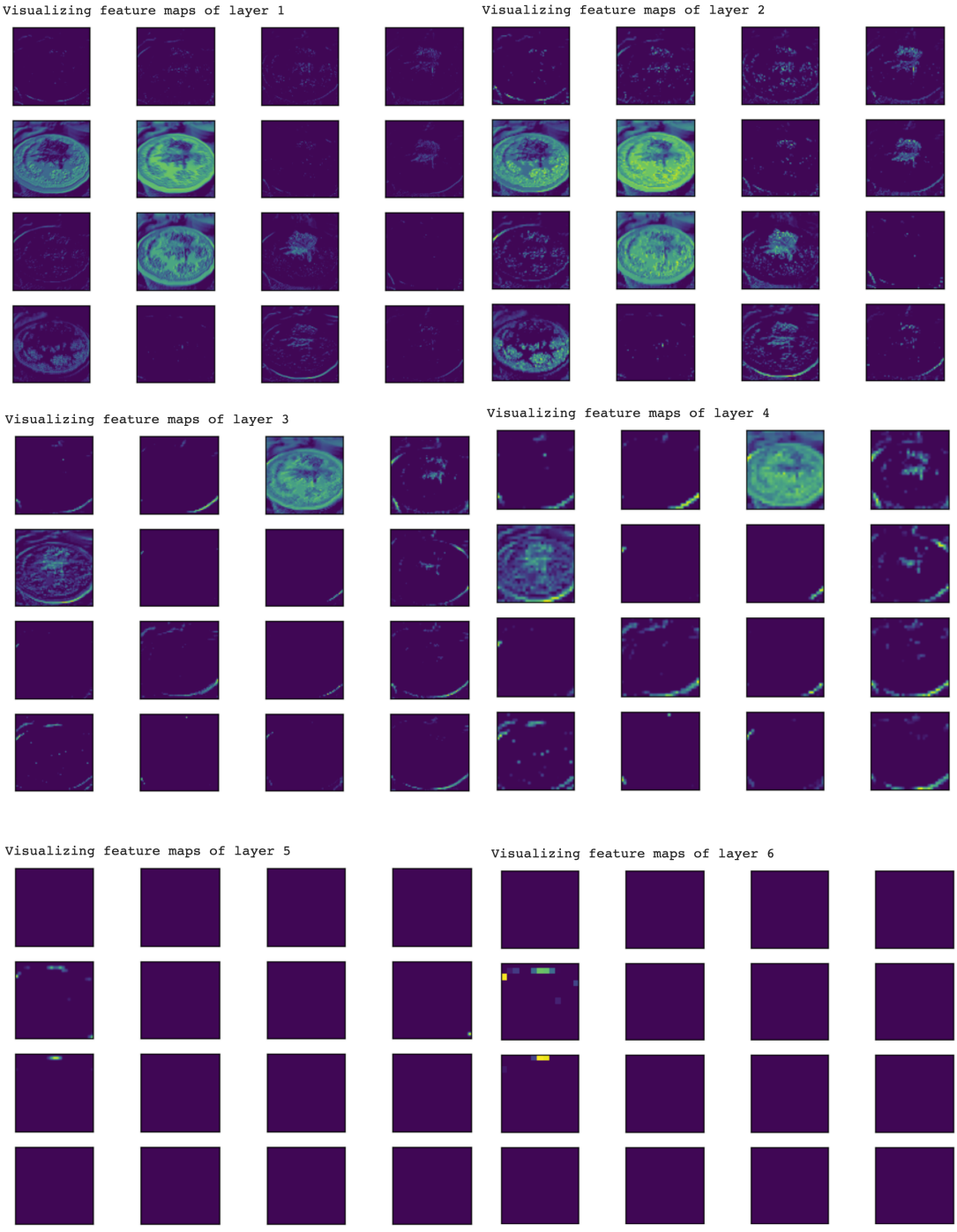
```
1/1 [==============================] - 0s 118ms/step
Predicted class: Chinese
```
Label in test data : Chinese

The visualization of feature maps was used to have a deeper understanding of the working principles and feature extraction process of CNN. Feature maps represent the visualized representation of image features extracted at various layers of the network. The process is as follows:

- Firstly choose layers that contain convolution and pooling operations, as these layers are responsible for extracting crucial features from the images.
- Construct a new model that takes the input image (see Chinese dish on the lleft hand side) and outputs the feature maps of the selected layers.
- By inputting the test images into this model, we obtain feature maps for each layer (n=6).

- Finally, through the visualization of feature maps for each layer, we can visually observe the different features extracted by the model from the images.

By examining the feature maps, we gain insights into the features extracted by the model at different levels. They are feature map for Conv2D_1, MaxPooling2D_1, Conv2D_2,MaxPooling2D_2, Conv2D_3,MaxPooling2D_3, orderly:



By analyzing feature maps changes, we can see how the model works:

In the first layer, it learns the dish's overall outline, especially the plate. This suggests the model can distinguish primary objects such as the plate, indicating the initial feature extraction process. The second layer refines the feature representation, specifically, the delineation between the plate and its contents, demonstrating the model's ability to segment and activate ingredient features individually. The third and fourth layers further activate features of the central ingredients. This shows the model's ability to discern finer ingredient characteristics and activate them independently. The fifth and sixth layers activate a small region beyond the plate and ingredients, showing the model's attention to detail.

In summary, these feature maps reveal the model's proficiency in recognizing and activating distinct feature regions of the plate and ingredients.

Assignment 3

- **Loading and Processing reviews from the stream with Pyspark**

  To initiate, the reviews from the stream are downloaded using the provided stream-saving procedure. The SparkContext is first created with all available GPUs on the computer. To load the downloaded data, a Spark session is then constructed. Considering the possible text preprocessing, the sparknlp is configured while constructing the spark session. Then, in order to extract information from the downloaded JSON file, the four schemas, specifically “review_id”, “app_id”, “review_text”, and “label”, are specified in StructType. After removing empty and duplicate JSON files, 481 reviews from the stream are successfully loaded.

review_id	app_id	review_text	label
138086736	1742020	Lemme get a Crunc...	1
138083242	1293460	[h1] Less of a ca...	1
138125042	1321440	Cassette Beasts i...	1
138076943	1575830	I'm really not su...	0
138082413	1159690	Voidtrain is a ne...	1
137685823	1321440	Playtime: 20 hour...	1
137905799	1321440	仅代表个人观点，浅浅的打个分吧-\...	1
123764026	2027560	It's close to bar...	1
138091114	1742020	Soratomo, Robosa,...	1
137542841	2176930	https://steamcomm...	1
138079463	1940340	The good:\n- Rela...	0
138082319	1940340	Alright, so this ...	0
138093108	2229260	I played Eden Ete...	1
137684311	1321440	I wasn't expectin...	1
138125985	1159690	A friend of mine ...	1
138126546	1159690	This game is incr...	1
138090218	1940340	Interesting seque...	1
138090218	1940340	Interesting seque...	1
138126222	2069040	I love puzzle gam...	1
138123827	1230170	I've largely enjo...	1