Support Vector Machine Model(regression):

Data set information:

The dataset utilized in this project is [California Housing Prices]. The total number of samples in the dataset is [20640] For the training, and testing phases, [1840] samples are allocated for training, [15480] for validation, and [5160] for testing.

Feature Extraction Phase:

1. Number of Features Extracted: After preprocessing, feature engineering, and encoding categorical variables, the dataset contains 13 features.

2. Feature Names: The extracted features are:

- Longitude
- Latitude
- Housing Median Age
- Total Rooms
- Total Bedrooms
- Population
- Households
- Median Income
- median house value
- Ocean Proximity
- Rooms per Household
- Bedrooms per Room
- Population per Household

3. Dimension of Resulted Features: The dimension of the dataset after feature extraction is (x, 13), where 'x' represents the number of samples.

Cross-Validation:

Cross-validation is used in the Support Vector Regression (SVR) model for model evaluation. The cross_val_score function with 5-fold cross-validation is employed to generate cross-validated estimates for each input data point. The training data is split into 5 folds, and the model is trained and evaluated 5 times, using a different fold for evaluation each time.

Hyperparameters:

The hyperparameters used in the SVR model are as follows:

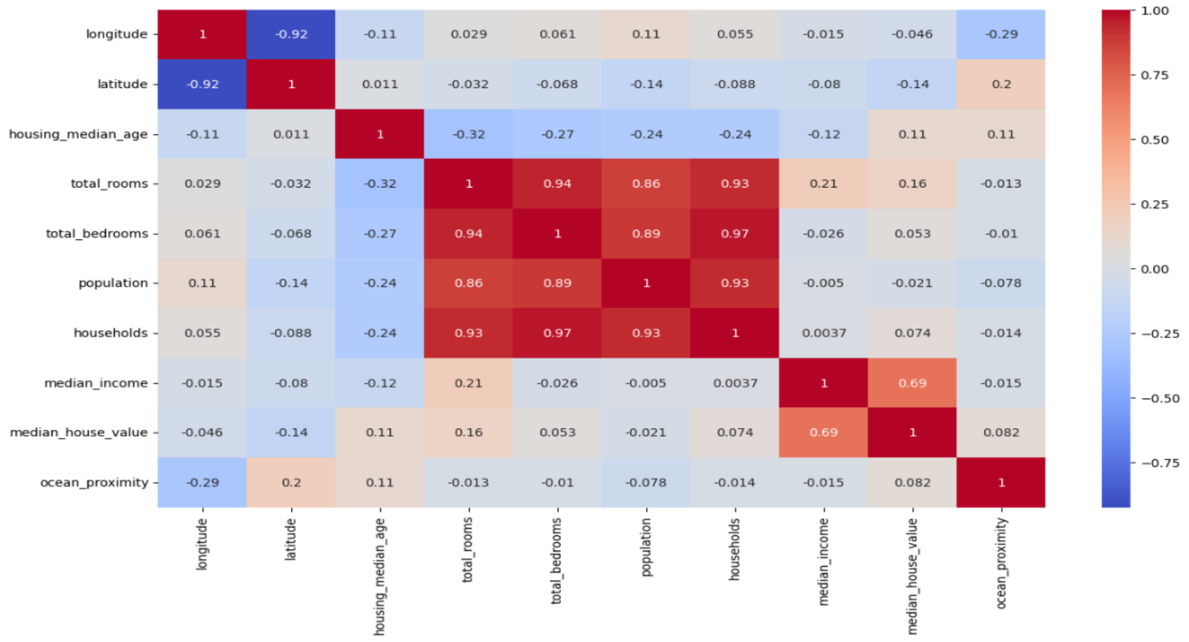
- degree: The degree of the polynomial kernel function. It's set to 2.

- C: Penalty parameter of the error term. It's tuned using grid search with values [1, 10, 50, 100].

- kernel: The kernel function used in the algorithm. It's set to 'linear'.

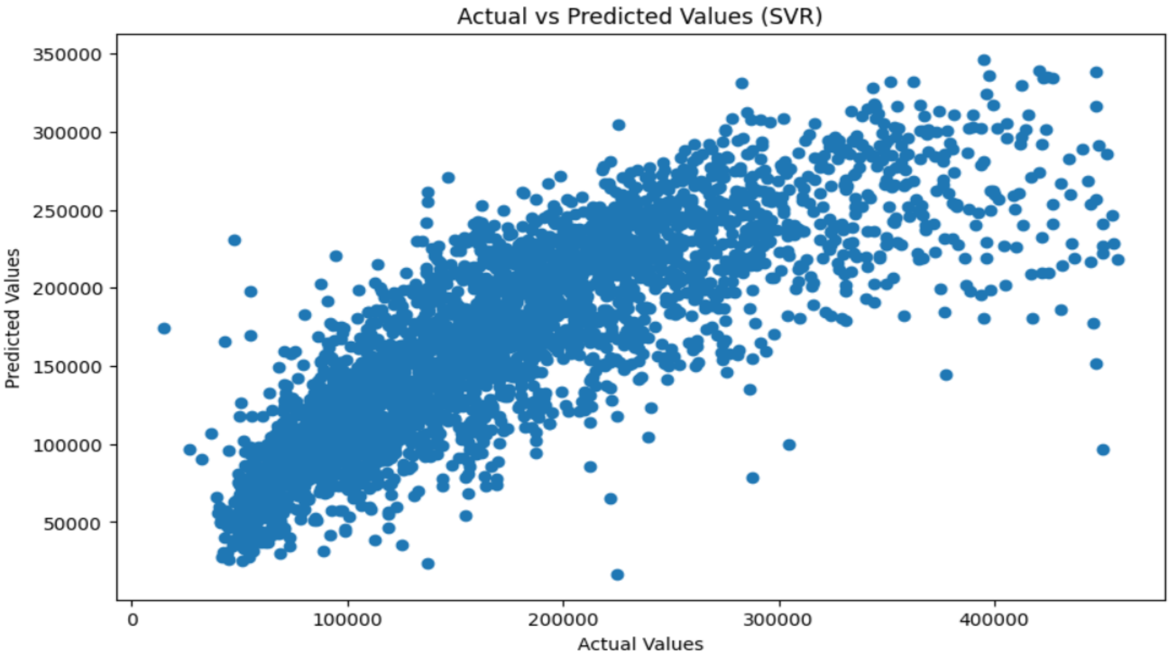
Plots:

Collaboration curve:



	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_income	median_house_value	ocean_proximity
longitude	1	-0.92	-0.11	0.029	0.061	0.11	0.055	-0.015	-0.046	-0.29
latitude	-0.92	1	0.011	-0.032	-0.068	-0.14	-0.088	-0.08	-0.14	0.2
housing_median_age	-0.11	0.011	1	-0.32	-0.27	-0.24	-0.24	-0.12	0.11	0.11
total_rooms	0.029	-0.032	-0.32	1	0.94	0.86	0.93	0.21	0.16	-0.013
total_bedrooms	0.061	-0.068	-0.27	0.94	1	0.89	0.97	-0.026	0.053	-0.01
population	0.11	-0.14	-0.24	0.86	0.89	1	0.93	-0.005	-0.021	-0.078
households	0.055	-0.088	-0.24	0.93	0.97	0.93	1	0.0037	0.074	-0.014
median_income	-0.015	-0.08	-0.12	0.21	-0.026	-0.005	0.0037	1	0.69	-0.015
median_house_value	-0.046	-0.14	0.11	0.16	0.053	-0.021	0.074	0.69	1	0.082
ocean_proximity	-0.29	0.2	0.11	-0.013	-0.01	-0.078	-0.014	-0.015	0.082	1

Mean Squared Error curve:



Decision tree Model:

Data set information:

The dataset utilized in this project is [loan approval]. The total number of samples in the dataset is [4269] For the training, and testing phases, [1840] samples are allocated for training, [15480] for validation, and [5160] for testing.

Feature Extraction Phase:

1. Number of Features Extracted: The dataset consists of 11 features.

2. Feature Names: The extracted features are:

- NO Of Dependents
- Income Annum
- Loan Amount
- Loan Term

- Cibil Score

- Residential Assets Value

- Commercial Assets Value

- Luxury Assets Value

- Bank Asset Value

- Education

- Self Employed

3. Dimension of Resulted Features: The dimension of the dataset after feature extraction is (100, 11), where 100 represents the number of samples and 11 represents the number of features.

Cross-Validation:

Cross-validation is used in the Decision Tree Classifier model for model evaluation. The cross_val_predict function with 5-fold cross-validation is employed to generate cross-validated estimates for each input data point. The training data is split into 5 folds, and the model is trained and evaluated 5 times, using a different fold for evaluation each time.

Hyperparameters:

The hyperparameters used in the Decision Tree Classifier model are as follows:

- criterion: The function to measure the quality of a split. The default is 'gini', but in this case, it's not specified, so it's 'gini' by default.

- splitter: The strategy used to choose the split at each node. The default is 'best', but in this case, it's not specified, so it's 'best' by default.

- max_depth: The maximum depth of the tree. It's not specified, so it grows until all leaves are pure or until all leaves contain less than min_samples_split samples.

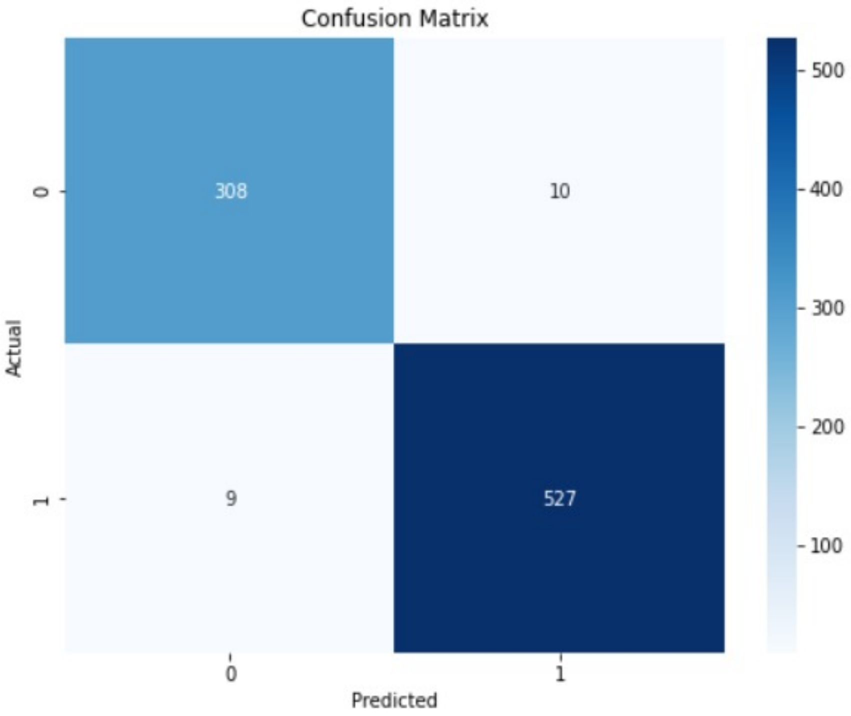
- min_samples_split: The minimum number of samples required to split an internal node. It's not specified, so it's 2 by default.

- min_samples_leaf: The minimum number of samples required to be at a leaf node. It's not specified, so it's 1 by default.

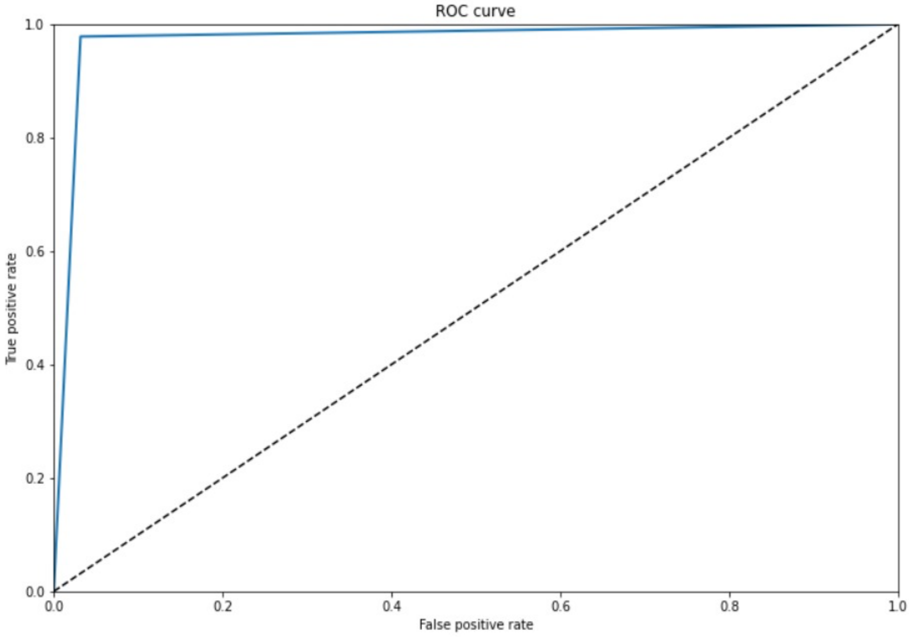
- random_state: The seed used by the random number generator. It's specified as 42 to ensure reproducibility.

Plots:

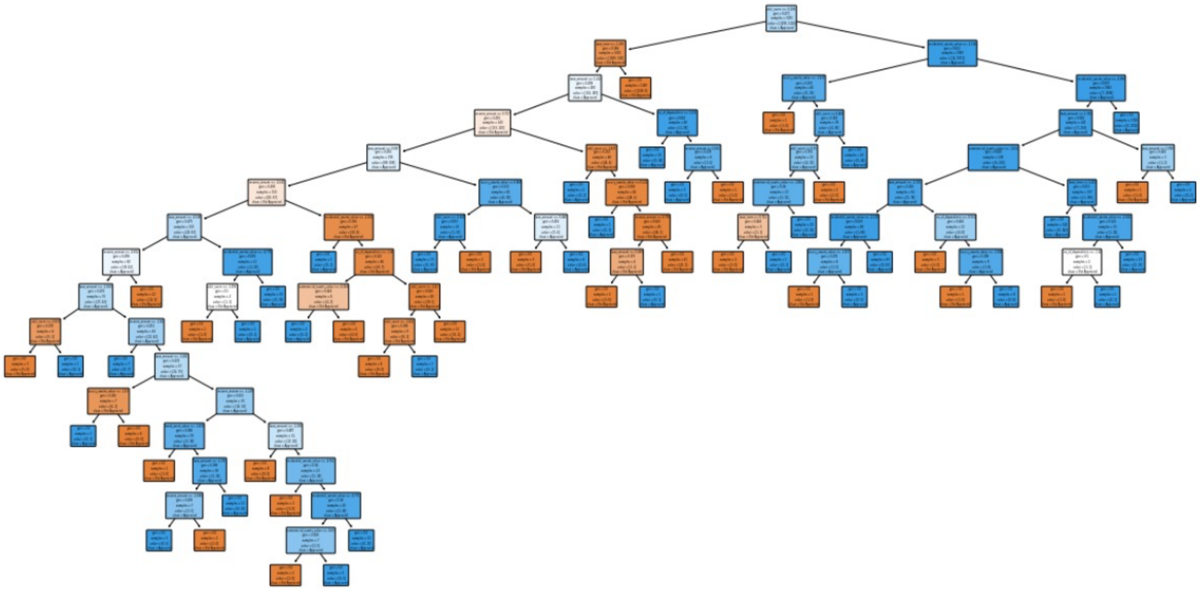
Confusion matrix:



Roc:



Decision tree visualization:



Neural network Model:

Data set information:

The dataset utilized in this project is [natural images]. It comprises [8] classes, with their corresponding labels being ['motorbike', 'airplane', 'flower', 'dog', 'fruit', 'car', 'cat', 'person']. The total number of samples in the dataset is [6899] each of size [56x56 pixels]For the training, validation, and testing phases, [5519] samples are allocated for training, and [1380] for testing.

Feature Extraction Phase:

1- Number of Features Extracted*: In the feature extraction phase, the images are resized to 56x56 pixels, resulting in a total of 56 * 56 = 3136 features per image.

2 -Names of Features: Each pixel value in the resized image can be considered as a feature. Since the images are grayscale, there is only one channel (intensity) per pixel.

- Dimension of Resulted Features: After flattening the resized images, the dimension of the resulted features is (56 * 56) = 3136.

Cross-validation Usage:

Cross-validation is used in the implemented model. Specifically, the model.fit() function includes a parameter validation_split =0.2, which indicates that 20% of the training data is used as validation data during training. This technique splits the training data into training and validation sets internally.

Hyperparameters Used:

- Initial Learning Rate*: The initial learning rate used is 0.0001.

- Optimizer*: Adam optimizer is used with the specified learning rate.

- Loss Function*: Sparse categorical cross-entropy loss function is used.

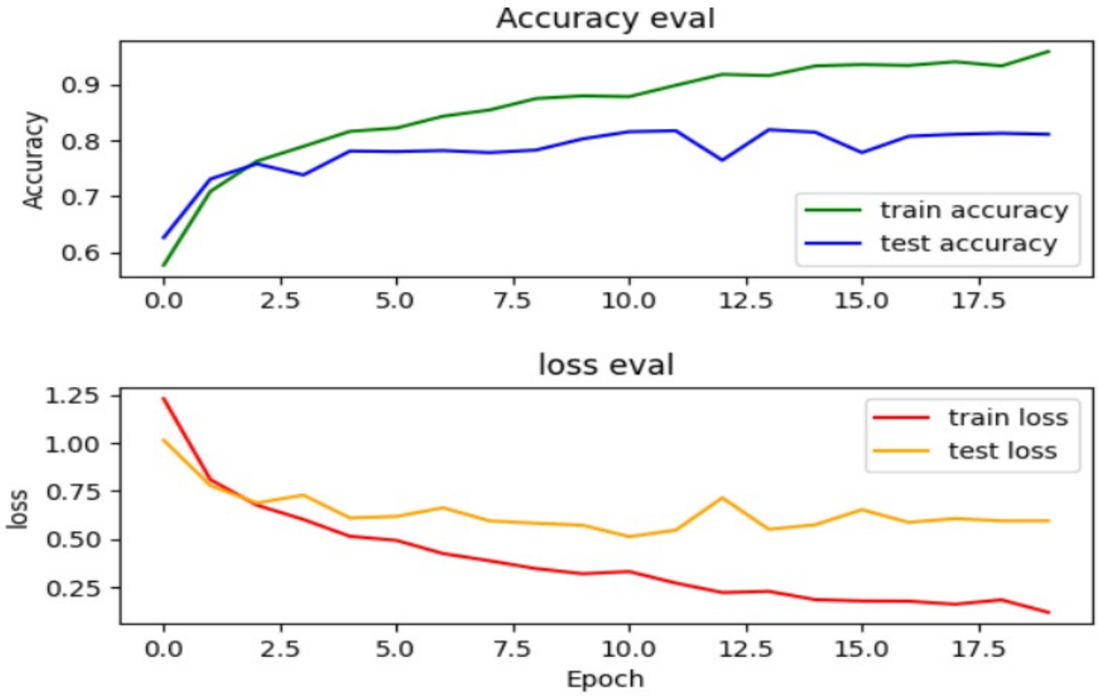
- Metrics*: Accuracy is used as the evaluation metric.

- Batch Size*: The batch size is not explicitly specified in the provided code. It defaults to the value set by the Keras framework.

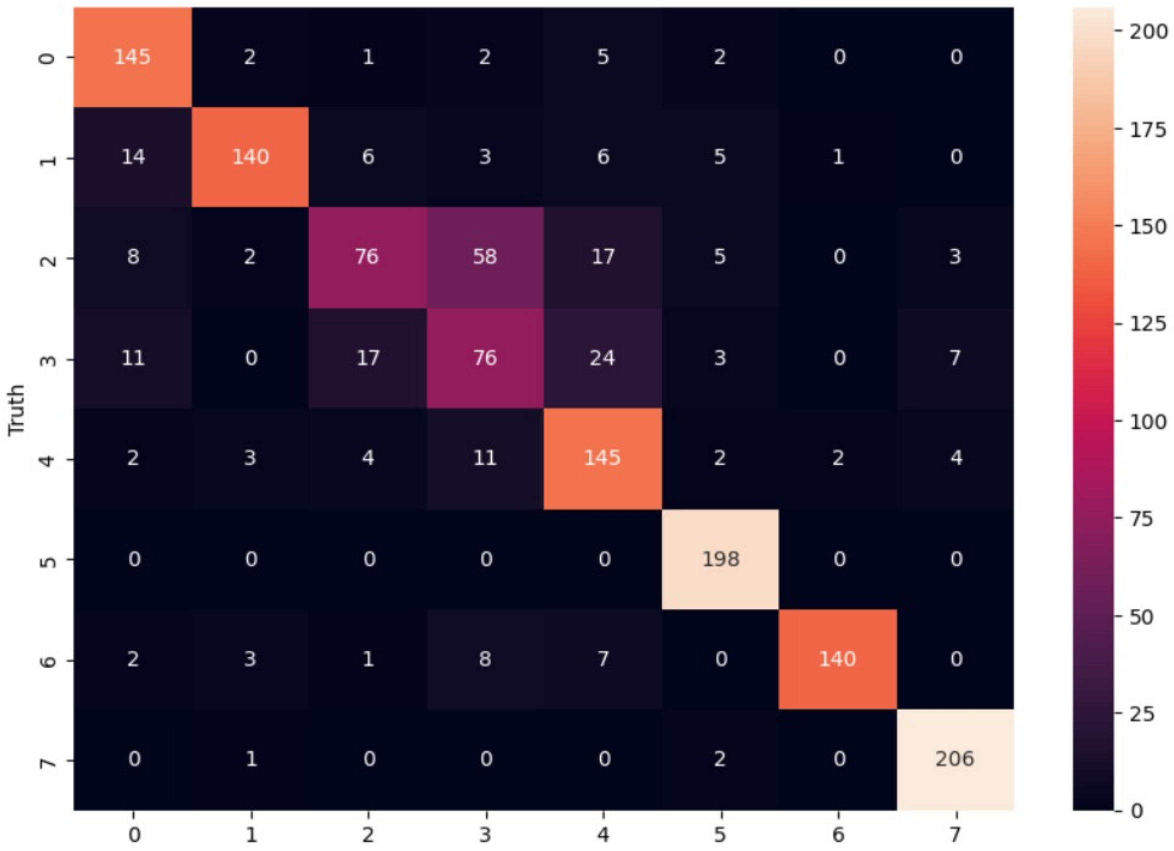
- Number of Epochs*: The model is trained for 20 epochs, as specified in model.fit().

Plots:

Accuracy curve, loss curve:

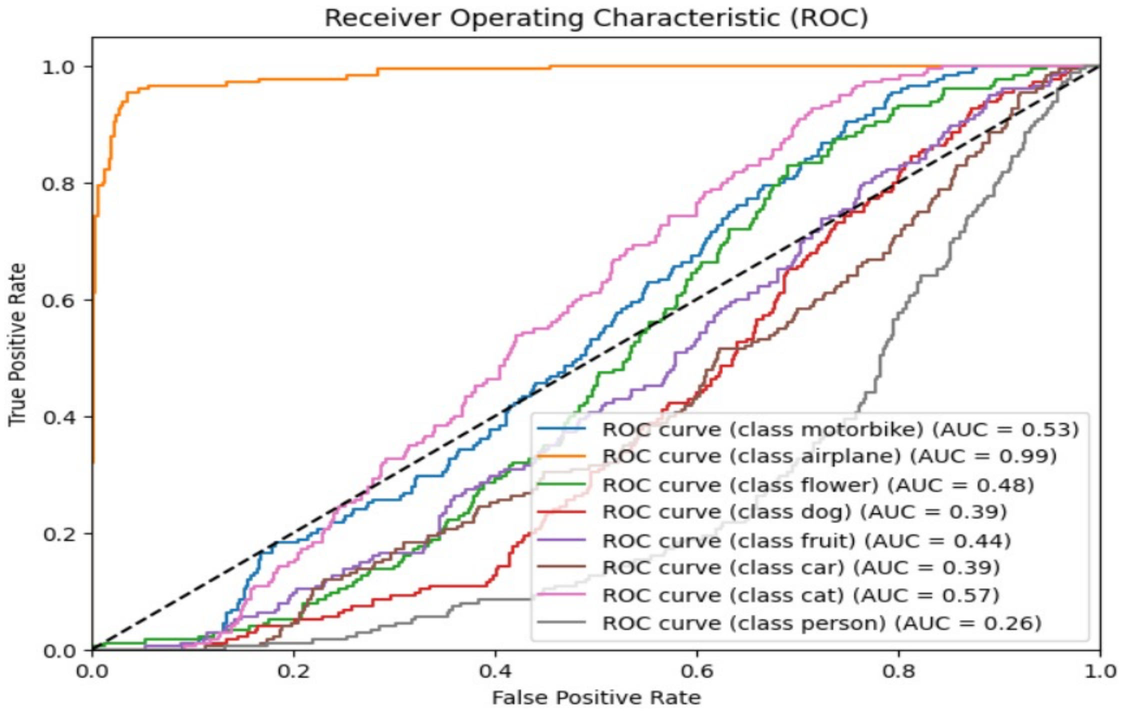


Confusion matrix:



	0	1	2	3	4	5	6	7
0	145	2	1	2	5	2	0	0
1	14	140	6	3	6	5	1	0
2	8	2	76	58	17	5	0	3
3	11	0	17	76	24	3	0	7
4	2	3	4	11	145	2	2	4
5	0	0	0	0	0	198	0	0
6	2	3	1	8	7	0	140	0
7	0	1	0	0	0	2	0	206

ROC curve:



Receiver Operating Characteristic (ROC)
True Positive Rate
False Positive Rate
ROC curve (class motorbike) (AUC = 0.53)
ROC curve (class airplane) (AUC = 0.99)
ROC curve (class flower) (AUC = 0.48)
ROC curve (class dog) (AUC = 0.39)
ROC curve (class fruit) (AUC = 0.44)
ROC curve (class car) (AUC = 0.39)
ROC curve (class cat) (AUC = 0.57)
ROC curve (class person) (AUC = 0.26)