Numerical Dataset Description:

The data pertains to the houses found in a given California district and some summary stats about them based on the 1990 census data. Be warned that the data aren't cleaned so there are some preprocessing steps required! The columns are as follows; their names are pretty self explanitory:

1. longitude: A measure of how far west a house is; a higher value is farther west

2. latitude: A measure of how far north a house is; a higher value is farther north

3. housingMedianAge: Median age of a house within a block; a lower number is a newer building

4. totalRooms: Total number of rooms within a block

5. totalBedrooms: Total number of bedrooms within a block

6. population: Total number of people residing within a block

7. households: Total number of households, a group of people residing within a home unit, for a block

8. medianIncome: Median income for households within a block of houses (measured in tens of thousands of US Dollars)

9. medianHouseValue: Median house value for households within a block (measured in US Dollars)

10. oceanProximity: Location of the house w.r.t ocean/sea

Data.info:

#	Column	Non-Null Count	Dtype
0	longitude	20640 non-null	float64
1	latitude	20640 non-null	float64
2	housing_median_age	20640 non-null	float64
3	total_rooms	20640 non-null	float64
4	total_bedrooms	20433 non-null	float64

5	population	20640 non-null	float64
6	households	20640 non-null	float64
7	median_income	20640 non-null	float64
8	median_house_value	20640 non-null	float64
9	ocean_proximity	20640 non-null	object

dtypes: float64(9), object (1)

The Sum of the Null values:

longitude	0
latitude	0
housing_median_age	0
total_rooms	0
total_bedrooms	207
population	0
households	0
median_income	0
median_house_value	0
ocean_proximity	0

dtype: int64

What I did to work with the Data:

1- Replace The Nan values with the mean of its column.
2- Then I replace the values in ocean_proximity column to be numbers instead of string.
3- I got the histogram of the data, and found that some columns are skewed left, so I used the log to solve this problem.
4- Then I extracted the features in X and the target in Y.
5- After that I split the data to train data and test data.
6- Then I train the models.

The Evaluation Metrics:

	Linear Regression	KNN
Accuracy	0.65	0.74
Mean Squared Error	4626715781.93	3459435924.5
R2 Score	0.6473	0.7363
RMSE	68019.97	58816.97
Mean Absolute Error	48879.1	3926.2

The Image Dataset Description:

Columbia Object Image Library (COIL-100) is a database of color images of 100 objects. The objects were placed on a motorized turntable against a black background. The turntable was rotated through 360 degrees to vary object pose with respect to a fixed color camera. Images of the objects were taken at pose intervals of 5 degrees. This corresponds to 72 poses per object. The images were size normalized.

The Columbia Object Image Library COIL contains 7200 images of 100 objects. The objects have a wide variety of complex geometric and reflectance characteristics.

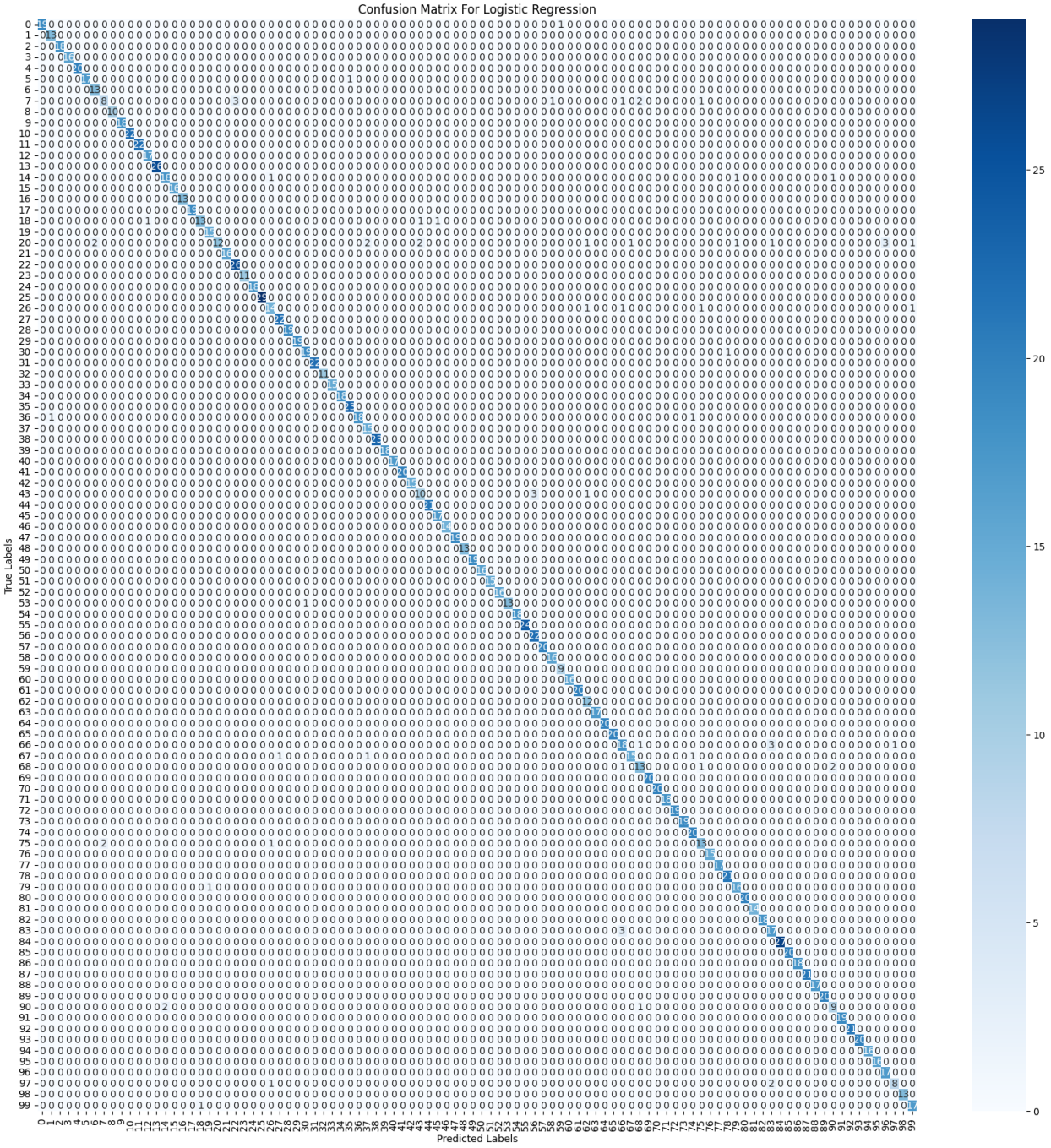
What I did to work with the Data:

1- I used a function to convert the images to a csv file with 64*64 pixels.
2- After that I choose the features into X and target into Y.
3- Then I convert them to np array.
4- Then I split the data into train and test data.
5- And to make sure that the data is correct I display some images.
6- Then I convert the data to 1D vector to simplify the work with logistic regression.
7- Then I train the models.

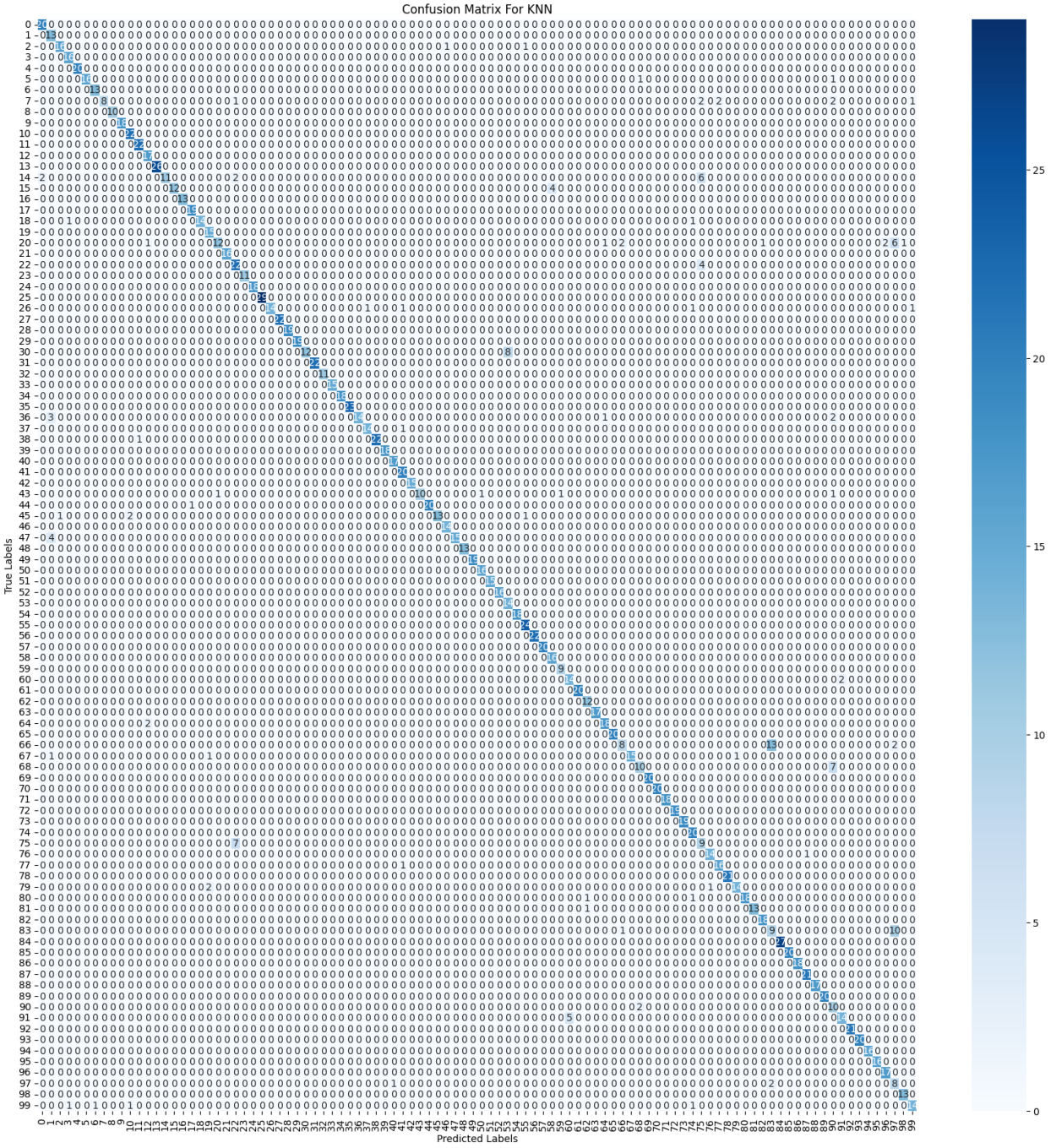
For 100 Classes:

	Logistic Regression	KNN
Accuracy	0.96	0.92
Recall	0.96	0.92
Precision	0.96	0.93
Loss	0.2622	0.3107

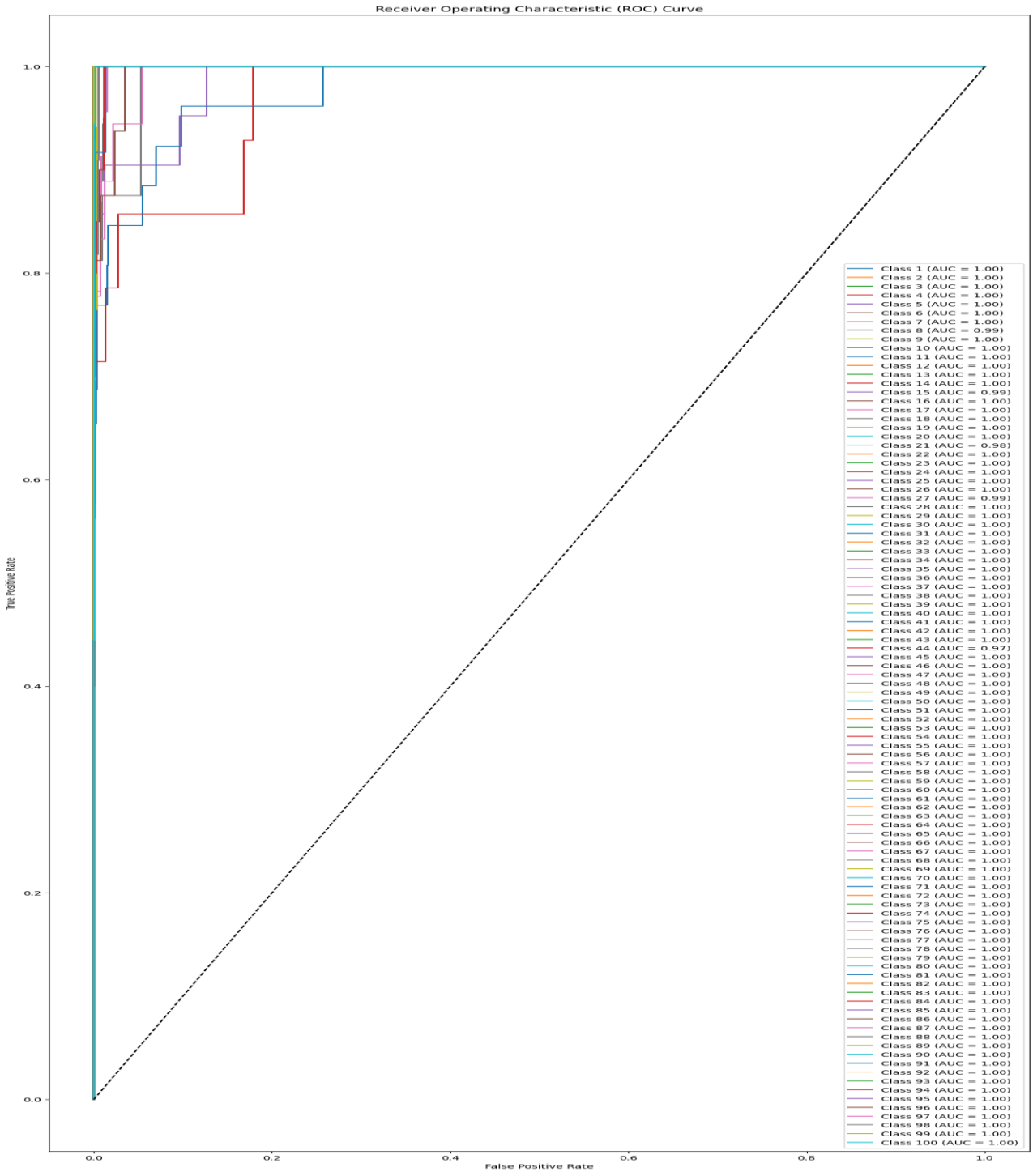
Confusion Matrix for Logistic Regression:



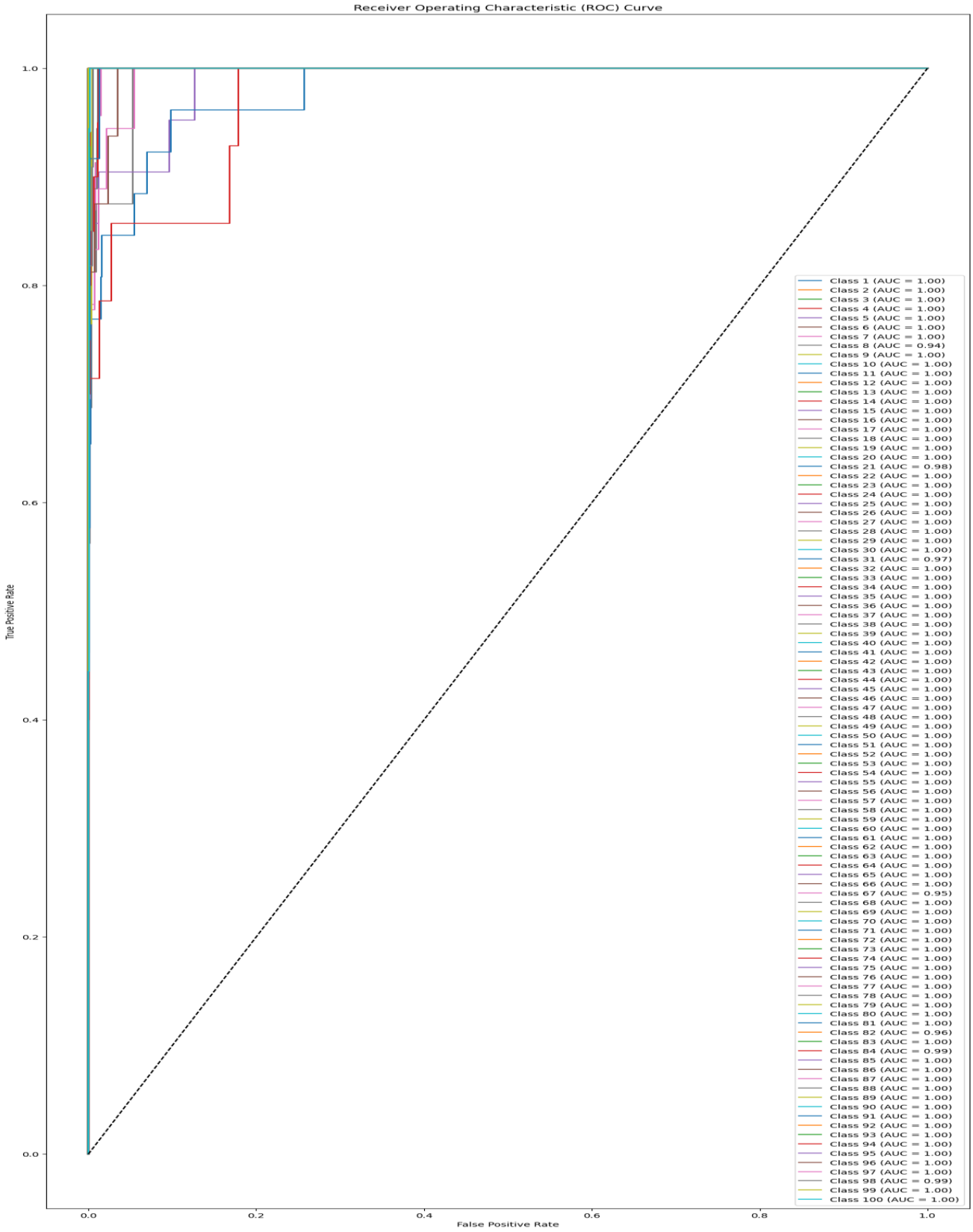
Confusion Matrix for KNN:



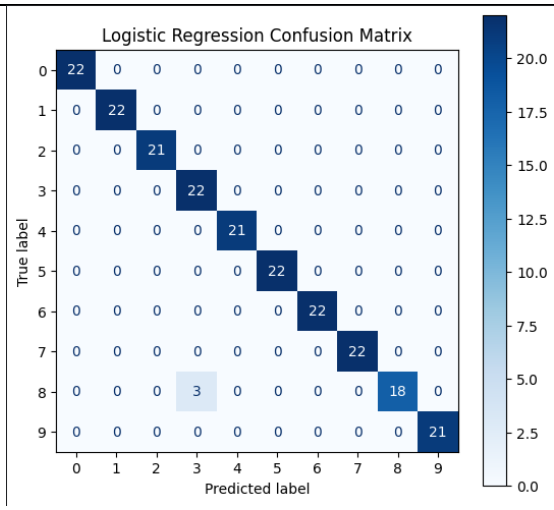
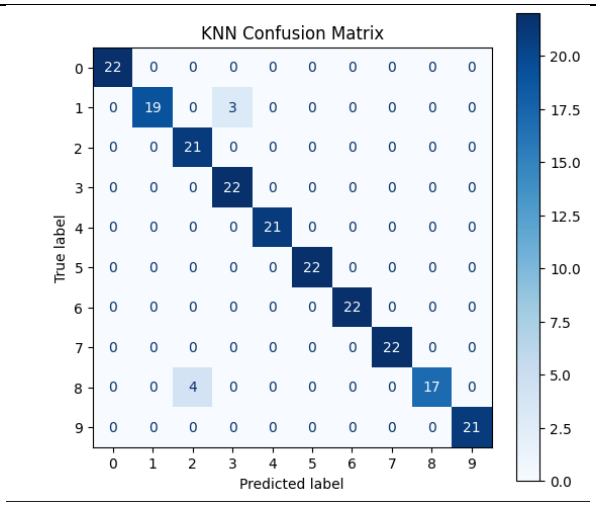
ROC and AUC for Logistic Regression:



ROC and AUC for KNN:



For 10 Classes:

	Logistic Regression	KNN
Accuracy	0.99	0.97
Recall	0.99	0.97
Precision	0.99	0.97
Loss	0.2886	0.0784
Confusion Matrix		
ROC and AUC