#Q1

Load the dataset using a library such as pandas.

Extract the number of instances by using the shape attribute of the DataFrame.

Extract the number of input attributes by using the shape attribute of the DataFrame and subtracting 1 to account for the output attribute.

Extract the number of possible values for the output attribute by using the unique function on the output column and counting the number of unique values.

Extract the number of categorical input attributes by counting the number of columns with the data type "object".

Extract the class ratio by using the value_counts function on the output column and dividing the number of instances belonging to each class by the total number of instances.

```
import pandas as pd


# Load the dataset

df = pd.read_csv("gender_prediction.csv")


# Extract the number of instances

num_instances = df.shape[0]


# Extract the number of input attributes

num_inputs = df.shape[1] - 1


# Extract the number of possible values for the output attribute

num_output_values = len(df["output"].unique())


# Extract the number of categorical input attributes

num_categorical = sum(df.dtypes == "object")


# Extract the class ratio

class_ratio = df["output"].value_counts() / num_instances


# Print the results
```

```
print(f"Number of instances: {num_instances}")

print(f"Number of input attributes: {num_inputs}")

print(f"Number of possible values for the output attribute: {num_output_values}")

print(f"Number of categorical input attributes: {num_categorical}")

print(f"Class ratio: {class_ratio}")


#Q2

import pandas as pd

from sklearn.ensemble import RandomForestClassifier

from sklearn.svm import SVC

from sklearn.neural_network import MLPClassifier

from sklearn.model_selection import train_test_split


# Load the dataset

df = pd.read_csv("gender_prediction.csv")


# Extract the input and output attributes

X = df.drop("output", axis=1)

y = df["output"]


# Split the dataset into train and test sets

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)


# Initialize the classifiers

rf = RandomForestClassifier()

svm = SVC()

mlp = MLPClassifier()


# Train the classifiers

rf.fit(X_train, y_train)

svm.fit(X_train, y_train)
```

```
mlp.fit(X_train, y_train)

# Predict on the test set
y_pred_rf = rf.predict(X_test)
y_pred_svm = svm.predict(X_test)
y_pred_mlp = mlp.predict(X_test)

# Calculate the number of instances that are incorrectly classified
incorrect_rf = sum(y_pred_rf != y_test)
incorrect_svm = sum(y_pred_svm != y_test)
incorrect_mlp = sum(y_pred_mlp != y_test)

print(f"Number of incorrect predictions (Random Forest): {incorrect_rf}")
print(f"Number of incorrect predictions (SVM): {incorrect_svm}")
print(f"Number of incorrect predictions (MLP): {incorrect_mlp}")
```