Data Collection and Preprocessing Phase

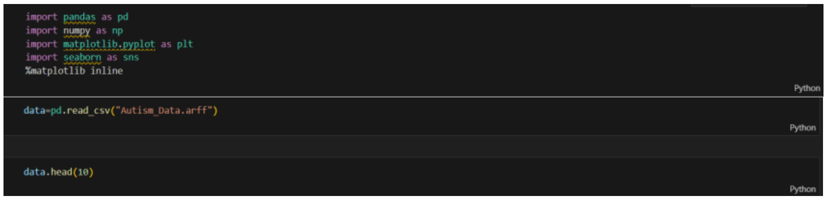
Date	16 July 2024
Team ID	xxxxxx
Project Title	Detection of Autistic Spectrum Disorder: Classification
Maximum Marks	6 Marks

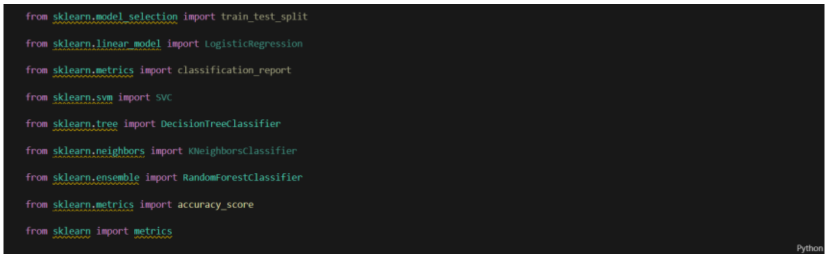
Preprocessing Template

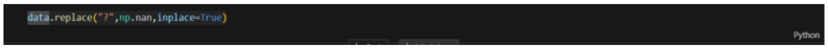
The images will be preprocessed by resizing, normalizing, augmenting, denoising, adjusting contrast, detecting edges, converting color space, cropping, batch normalizing, and whitening data. These steps will enhance data quality, promote model generalization, and improve convergence during neural network training, ensuring robust and efficient performance across various computer vision tasks.

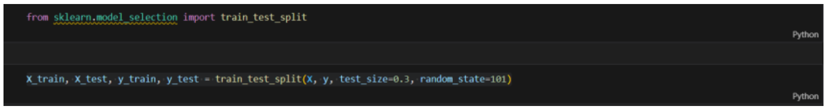
Section	**Description**
Data Overview	The dataset consists of behavioral features and individual characteristics for autism screening. It includes columns for age, gender, various scores, and binary features related to ASD detection.
Normalization	Normalize numerical feature values to a common scale, e.g., between 0 and 1 or standardize to have a mean of 0 and a standard deviation of 1.
Handling missing values	Handling missing values is crucial for ensuring the quality and reliability of your dataset. Missing values can skew results and impact the performance of machine learning models. The common strategies to handle missing values include:

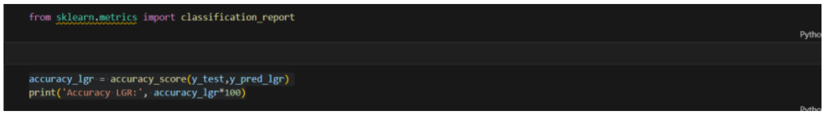
Splitting Dataset	Splitting the dataset into training and testing subsets allows you to evaluate the performance of your machine learning model on unseen data. Typically, the dataset is divided into a training set (used to train the model) and a test set (used to evaluate the model’s performance).
Calculating Accuracy	Accuracy measures the proportion of correctly classified instances out of the total instances. It’s a common metric for evaluating classification models. High accuracy indicates that the model performs well on the given dataset.
Data Preprocessing Code Screenshots	

Loading Data	
	`import pandas as pd`
	`import numpy as np`
	`import matplotlib.pyplot as plt`
	`import seaborn as sns`
	`%matplotlib inline`
	`data=pd.read_csv("Autism_Data.arff")`
	`data.head(10)`

Normalization	
	`from sklearn.model_selection import train_test_split`
	`from sklearn.linear_model import LogisticRegression`
	`from sklearn.metrics import classification_report`
	`from sklearn.svm import SVC`
	`from sklearn.tree import DecisionTreeClassifier`
	`from sklearn.neighbors import KNeighborsClassifier`
	`from sklearn.ensemble import RandomForestClassifier`
	`from sklearn.metrics import accuracy_score`
	`from sklearn import metrics`

Handling missing values	 `data.replace("?",np.nan,inplace=True)`

Splitting Dataset	
	`from sklearn.model_selection import train_test_split`
	`X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=101)`

Calculating Accuracy	
	`from sklearn.metrics import classification_report`
	
	`accuracy_lgr = accuracy_score(y_test,y_pred_lgr)`
	`print('Accuracy LGR:', accuracy_lgr*100)`