**MAJOR PROJECT**

**WINE QUALITY ANALYSIS**

**Aim:**

To use machine learning to predict the quality of wine based on its chemical properties, such as acidity, pH, and alcohol content. By analysing these features, the goal is to develop models that can classify wine quality into categories (like poor, average, or excellent) and also predict its quality score on a scale.

**Code:**

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

import warnings

warnings.filterwarnings('ignore')

df = pd.read\_csv(r'C:\Users\yogesh\Downloads\upload\_b17b4ebe-b5ef-4168-b23e-759428717a8b.csv')

df.head()

from sklearn.impute import KNNImputer

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import MinMaxScaler

from sklearn.linear\_model import LogisticRegression

from sklearn.compose import ColumnTransformer

from sklearn.pipeline import Pipeline

from sklearn.metrics import classification\_report

from xgboost import XGBClassifier

from sklearn.ensemble import RandomForestClassifier

df.info()

df.describe().T

print(df.isnull().sum())

imputer = KNNImputer(n\_neighbors=3)

df\_imputed = pd.DataFrame(imputer.fit\_transform(df), columns=df.columns)

for col in df\_imputed.columns:

plt.figure(figsize=(14, 4))

plt.title(f'Distribution and Boxplot for {col}')

plt.subplot(1, 2, 1)

sns.distplot(df\_imputed[col])

plt.subplot(1, 2, 2)

sns.boxplot(df\_imputed[col])

plt.show()

for col in df\_imputed.columns:

percentile25 = df\_imputed[col].quantile(0.25)

percentile75 = df\_imputed[col].quantile(0.75)

IQR = percentile75 - percentile25

upper\_limit = percentile75 + 1.5 \* IQR

lower\_limit = percentile25 - 1.5 \* IQR

df\_imputed[col] = np.where(df\_imputed[col] > upper\_limit, upper\_limit,

np.where(df\_imputed[col] < lower\_limit, lower\_limit, df\_imputed[col]))

df\_imputed['best\_quality'] = [1 if x > 5 else 0 for x in df\_imputed['quality']]

df\_imputed.drop(columns=['quality'], inplace=True)

plt.figure(figsize=(20, 10))

sns.heatmap(df\_imputed.corr(), annot=True, cmap='coolwarm', fmt=".2f", linewidths=0.5)

plt.show()

X = df\_imputed.drop(columns=['best\_quality'])

y = df\_imputed['best\_quality']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

trf1 = ColumnTransformer([('scaler', MinMaxScaler(), slice(0, 12))], remainder='passthrough')

pipe = Pipeline([('trf1', trf1)])

X\_train\_transform = pipe.fit\_transform(X\_train)

X\_test\_transform = pipe.traQQnsform(X\_test)

models = {

'Logistic Regression': LogisticRegression(),

'XGBoost': XGBClassifier(use\_label\_encoder=False, eval\_metric='mlogloss'), # Fixed typo

'Random Forest': RandomForestClassifier()

}

for name, model in models.items():

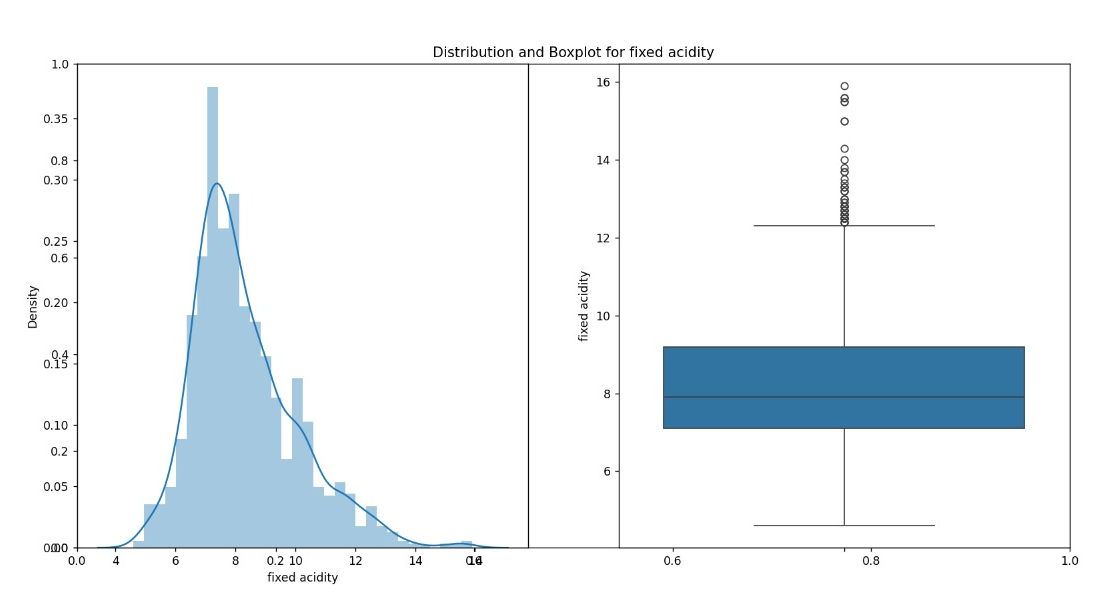
model.fit(X\_train\_transform, y\_train)

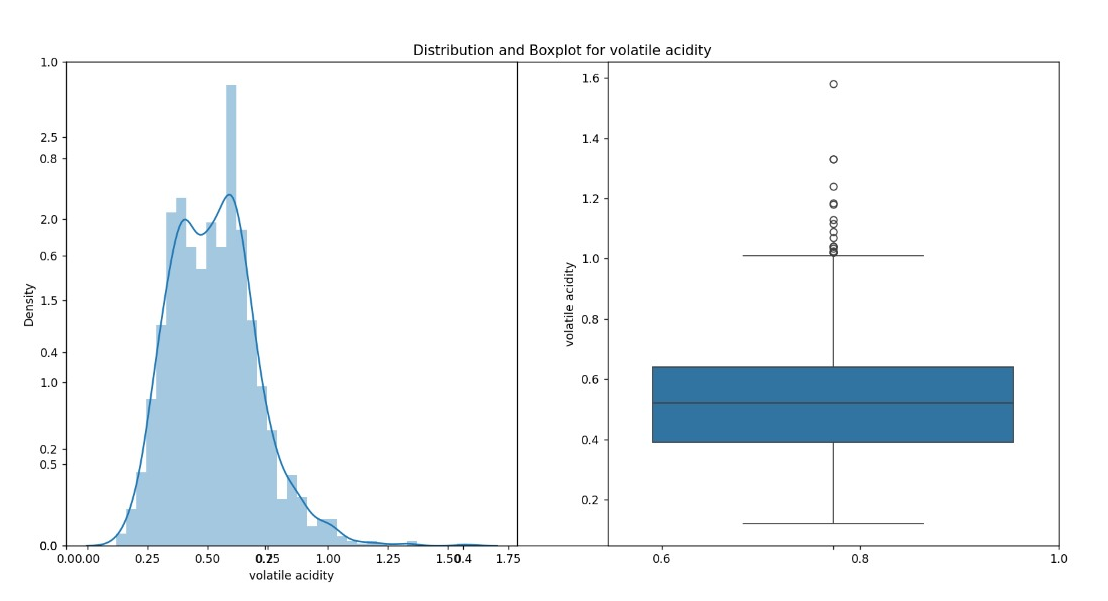
y\_pred = model.predict(X\_test\_transform)

cl\_report = classification\_report(y\_test, y\_pred)

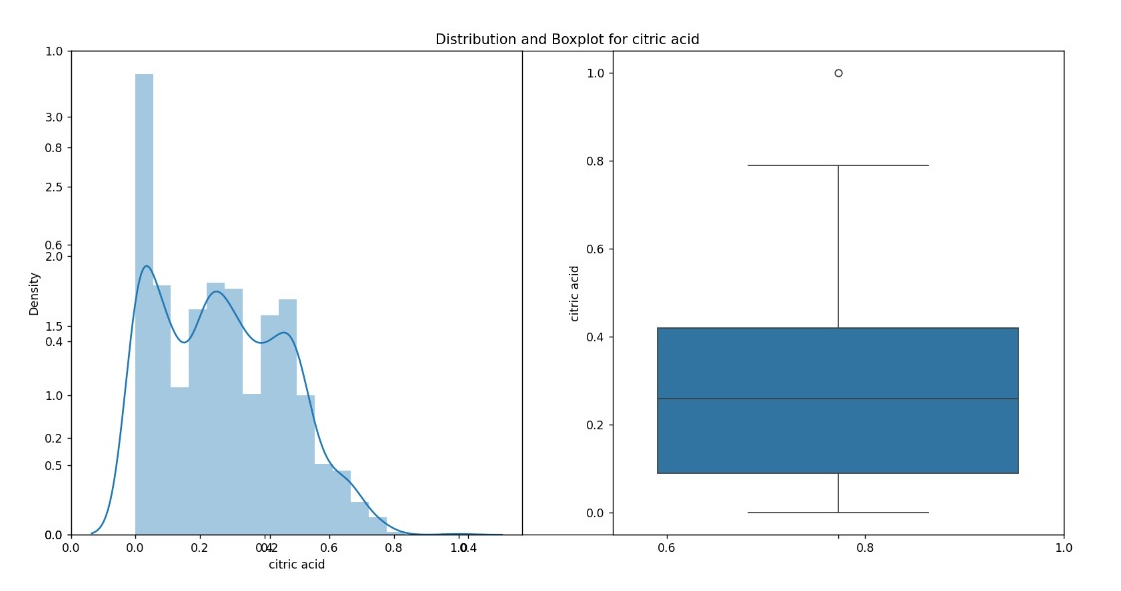
print(f'Classification Report for {name}:\n\n{cl\_report}')

**Output:**

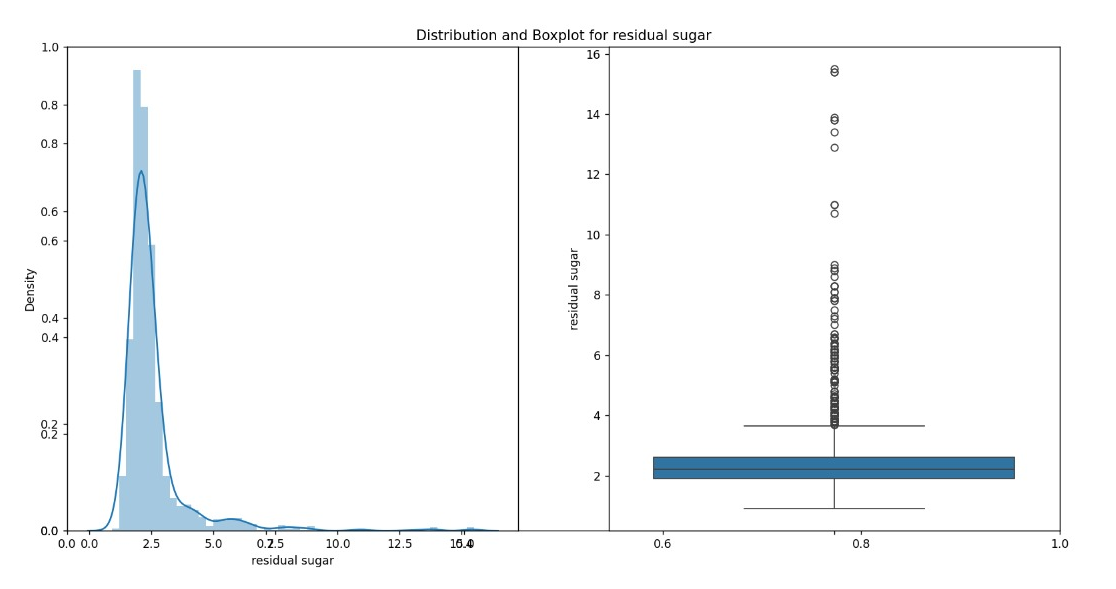
1. Fixed Acidity:
2. Volatile Acidity:



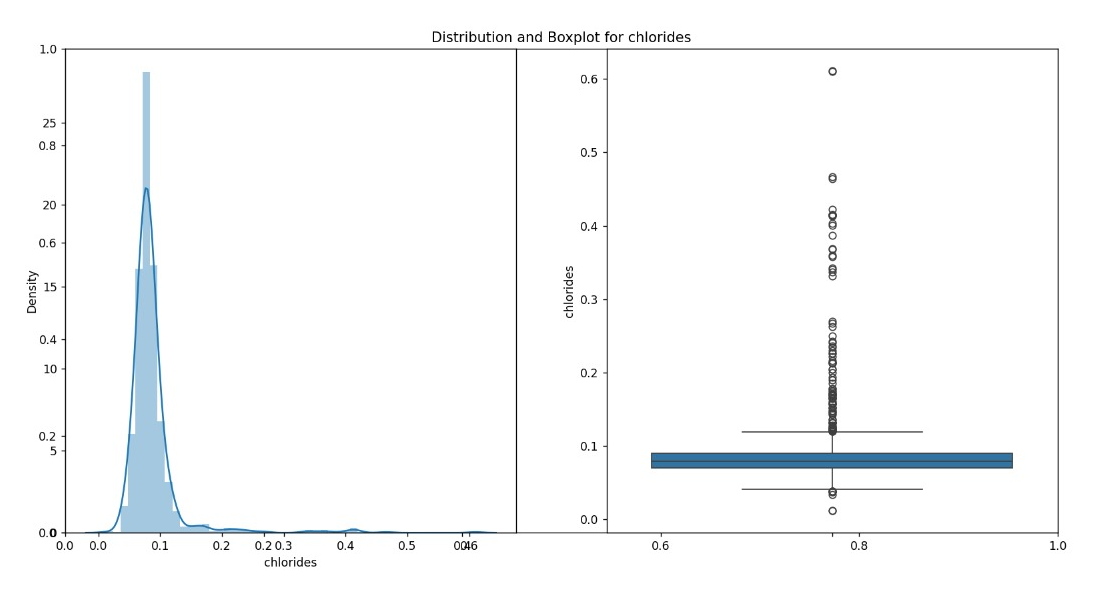
1. Citric Acid:



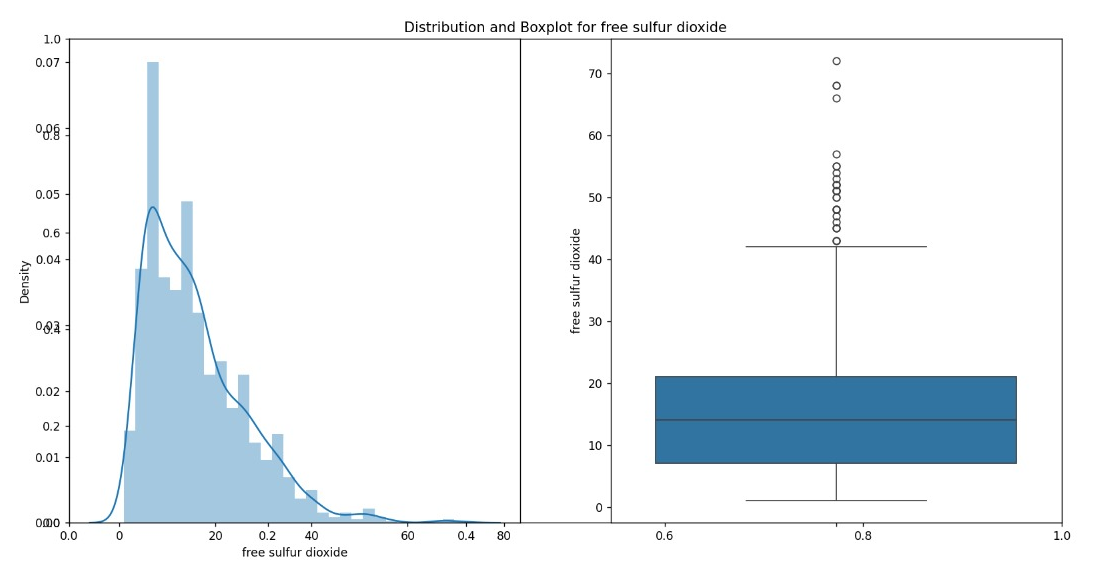
1. Residual Sugar:



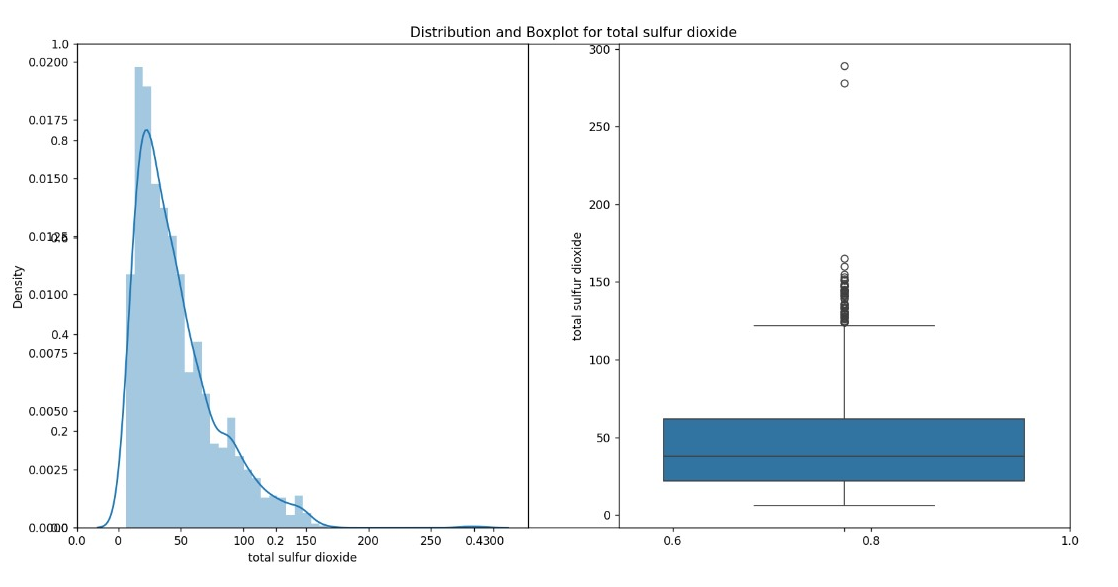
1. Chlorides:



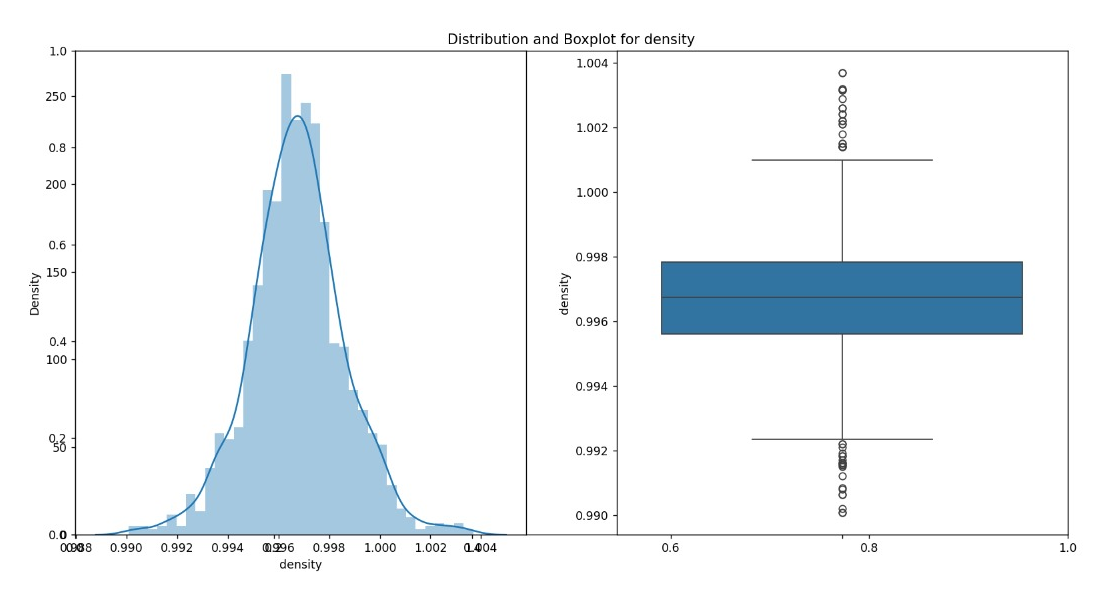
1. Free Sulfur Dioxide:



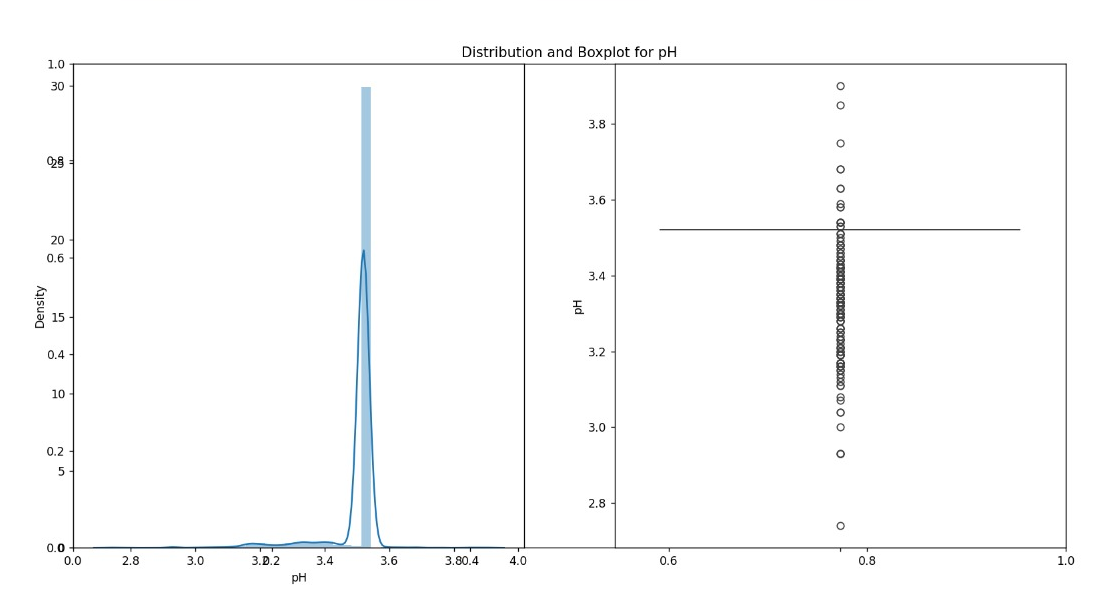
1. Total Sulfur Dioxide:

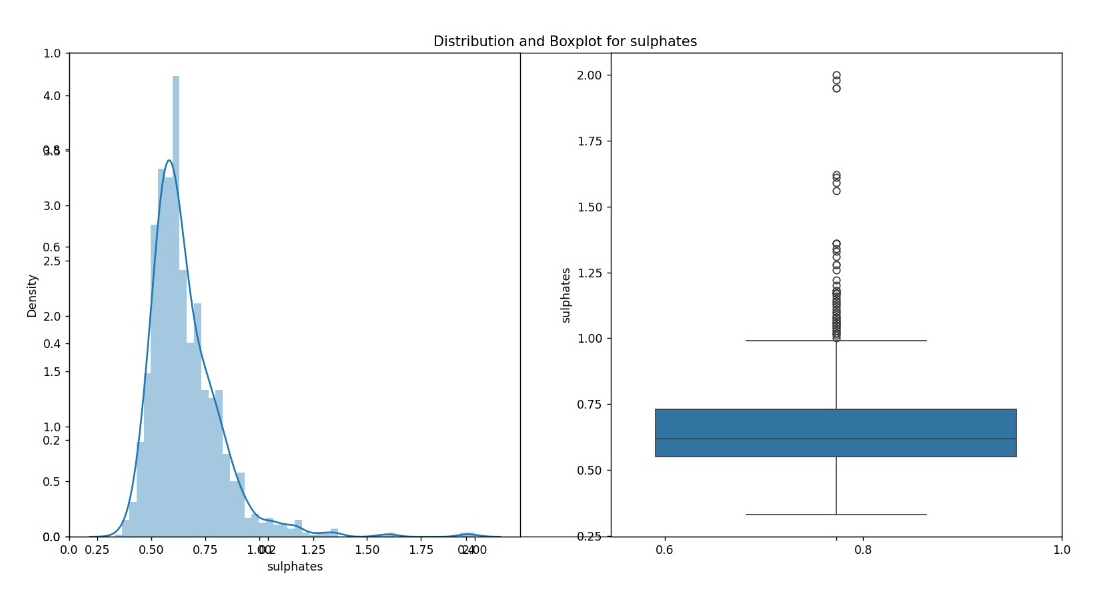


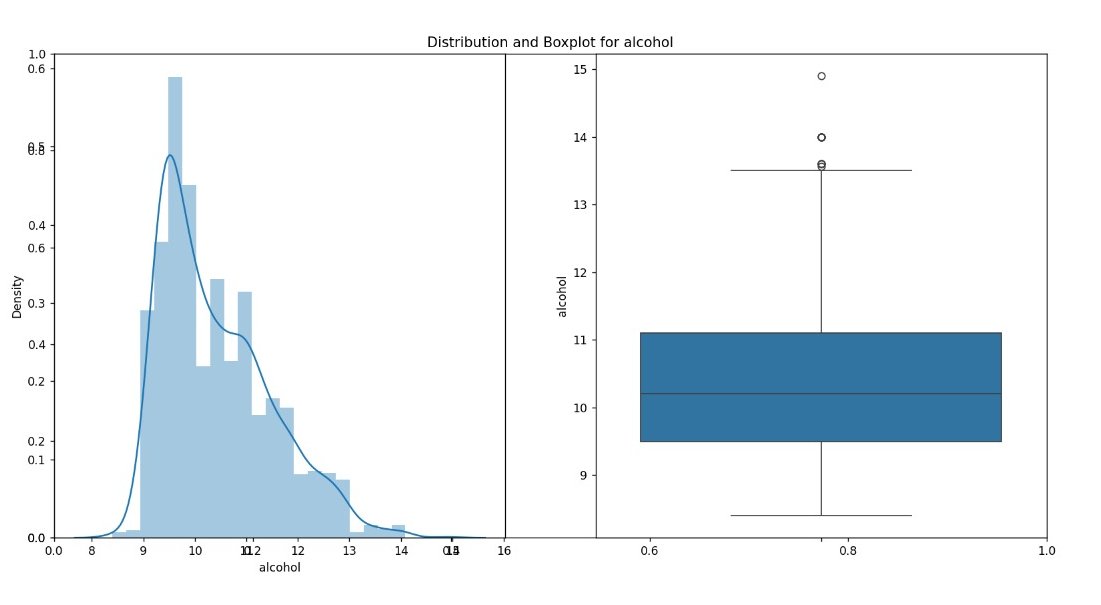
1. Density:



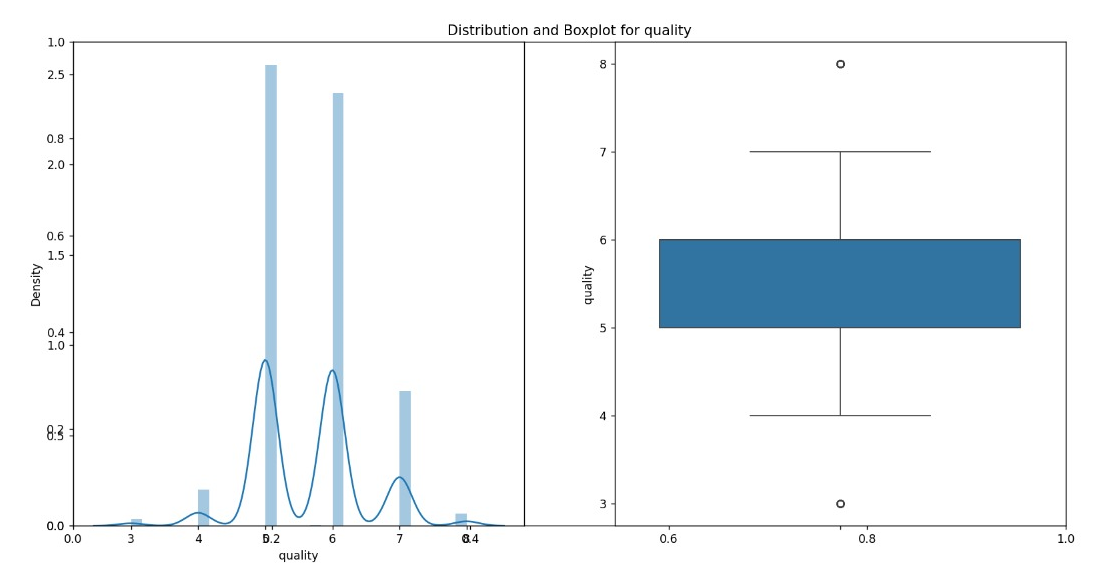
1. pH:

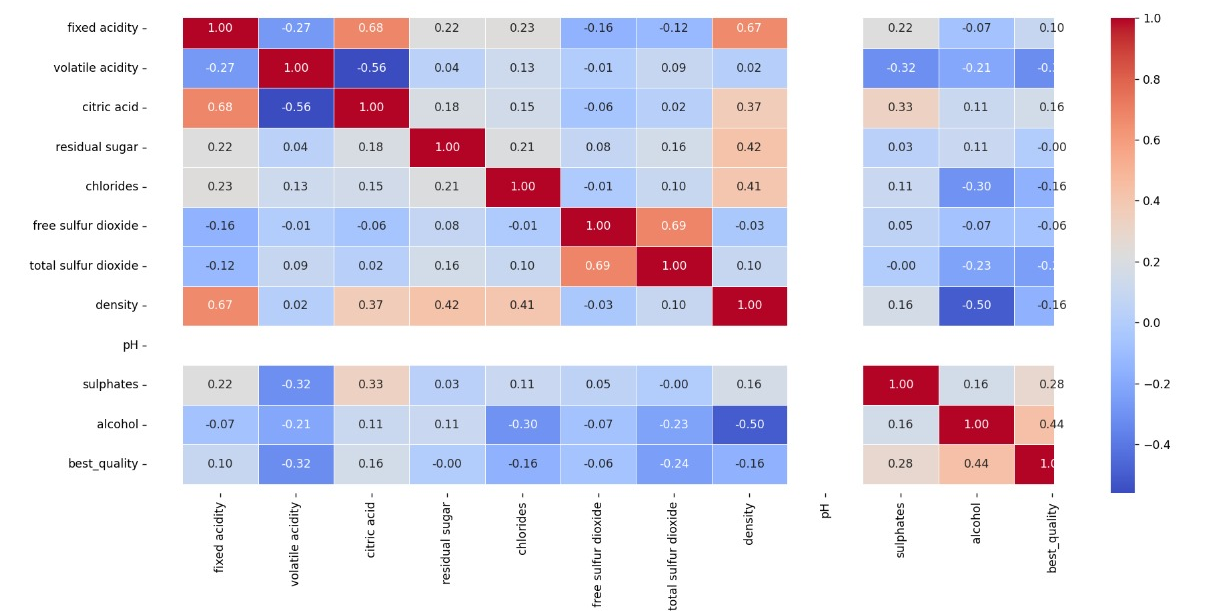


10.Sulphates: 

11.Alcohol:

12. Quality (score between o and10):





**TEAM NUMBER: 8**

**TEAM MEMBERS:**

S. Bhuwaneshwari

Yogesh S

Likitha.M

Bhoomika MS