NAME-MAYANK BAGAULI

ID - 20561038

ML END-SEMESTER

PRACTICAL

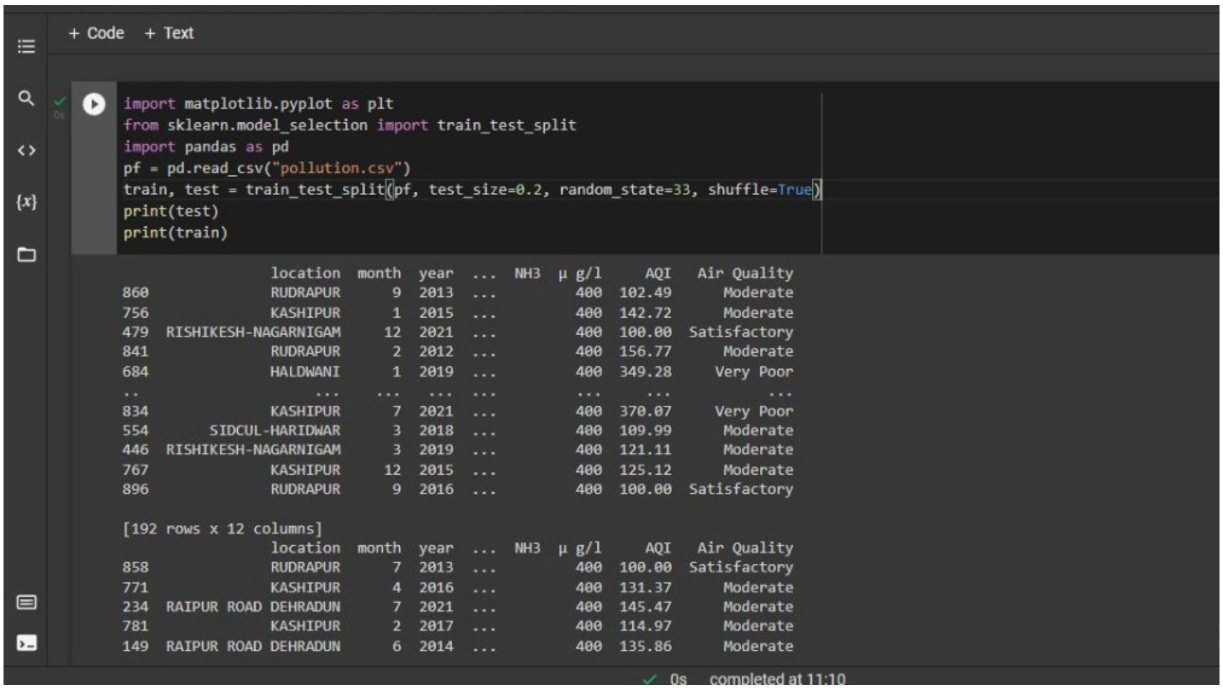
SOLUTION :

```
{
 "cells": [
  {
   "cell_type": "code",
   "execution_count": 25,
   "id": "c08d214d",
   "metadata": {},
   "outputs": [],
   "source": [
    "import numpy as np\n",
    "import matplotlib.pyplot as plt\n",
    "import pandas as pd\n",
    "from sklearn import preprocessing\n",
    " \n",
    "label_encoder = preprocessing.LabelEncoder()\n",
    " \n",
    "# Encode labels in column 'species'.\n",
    "\n",
    "\n",
    "\n",
    "# Importing the dataset\n",
    "df = pd.read_csv('Desktop/pollution.csv')\n",
    "df['Air Quality']= label_encoder.fit_transform(df['Air Quality'])\n",
    "X=df.iloc[:,:-1]\n",
    "y=df.iloc[:,-1]\n",
    "\n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 26,
   "id": "574f0f0a",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/html": [
       "<div>\n",
       "<style scoped>\n",
       "    .dataframe tbody tr th:only-of-type {\n",
       "        vertical-align: middle;\n",
       "    }\n",
"\n",
"    .dataframe tbody tr th {\n",
"        vertical-align: top;\n",
"    }\n",
"\n",
"    .dataframe thead th {\n",
"        text-align: right;\n",
"    }\n",
"</style>\n",
"<table border=\"1\" class=\"dataframe\">\n",
"  <thead>\n",
"    <tr style=\"text-align: right;\">\n",
"      <th></th>\n",
"      <th>location</th>\n",
"      <th>month</th>\n",
"      <th>year</th>\n",
"      <th>SO2 µg/l</th>\n",
"      <th>NO2µg/l</th>\n",
"      <th>PM10 µg/l</th>\n",
"      <th>PM2.5 µ g/l</th>\n",
"      <th>CO µg/l</th>\n",
"      <th>O3 µ g/l 8 HR</th>\n",
"      <th>NH3  µ g/l</th>\n",
"      <th>AQI</th>\n",
"    </tr>\n",
"  </thead>\n",
"  <tbody>\n",
"    <tr>\n",
"      <th>0</th>\n",
"      <td>CLOCK TOWER-DEHRADUN</td>\n",
"      <td>1</td>\n",
"      <td>2012</td>\n",
"      <td>27.33</td>\n",
"      <td>30.33</td>\n",
"      <td>193.28</td>\n",
"      <td>60.0</td>\n",
"      <td>2</td>\n",
"      <td>100</td>\n",
"      <td>400</td>\n",
"      <td>162.19</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>1</th>\n",
"      <td>CLOCK TOWER-DEHRADUN</td>\n",
"      <td>2</td>\n",
"      <td>2012</td>\n",
"      <td>25.68</td>\n",
"      <td>25.80</td>\n",
"      <td>173.77</td>\n",
"      <td>60.0</td>\n",
"      <td>2</td>\n",
"      <td>100</td>\n",
"      <td>400</td>\n",
"      <td>149.18</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>2</th>\n",
"      <td>CLOCK TOWER-DEHRADUN</td>\n",
"      <td>3</td>\n",
"      <td>2012</td>\n",
"      <td>29.64</td>\n",
"      <td>27.50</td>\n",
"      <td>211.35</td>\n",
"      <td>60.0</td>\n",
"      <td>2</td>\n",
"      <td>100</td>\n",
"      <td>400</td>\n",
"      <td>174.23</td>\n",
"    </tr>\n",
"  </tbody>\n",
"</table>\n",
"</div>"
```

```
   ],
   "text/plain": [
    "             location  month  year  SO2 µg/l  NO2µg/l  PM10 µg/l  \\\n",
    "0  CLOCK TOWER-DEHRADUN     1  2012     27.33    30.33     193.28   \n",
    "1  CLOCK TOWER-DEHRADUN     2  2012     25.68    25.80     173.77   \n",
    "2  CLOCK TOWER-DEHRADUN     3  2012     29.64    27.50     211.35   \n",
    "\n",
    "   PM2.5 µ g/l  CO µg/l  O3 µ g/l 8 HR  NH3  µ g/l     AQI  \n",
    "0        60.0        2           100        400  162.19  \n",
    "1        60.0        2           100        400  149.18  \n",
    "2        60.0        2           100        400  174.23  "
   ]
  },
  "execution_count": 26,
  "metadata": {},
  "output_type": "execute_result"
 }
],
"source": [
 "X.head(3)"
]
},
{
"cell_type": "code",
 "execution_count": 27,
 "id": "5cbd205f",
 "metadata": {},
 "outputs": [],
 "source": [
  "from sklearn.preprocessing import OneHotEncoder\n",
  "enc = OneHotEncoder()\n",
  "# transforming the column after fitting\n",
  "enc = enc.fit_transform(X[['location']]).toarray()\n",
  "# converting arrays to a dataframe\n",
  "encoded_colm = pd.DataFrame(enc)\n",
  "# concating dataframes \n",
  "X = pd.concat([X, encoded_colm], axis = 1) \n",
  "# removing the encoded column.\n",
  "X = X.drop(['location'], axis = 1)\n"
 ]
},
{
 "cell_type": "code",
 "execution_count": 28,
 "id": "4beb0363",
 "metadata": {},
 "outputs": [
  {
```

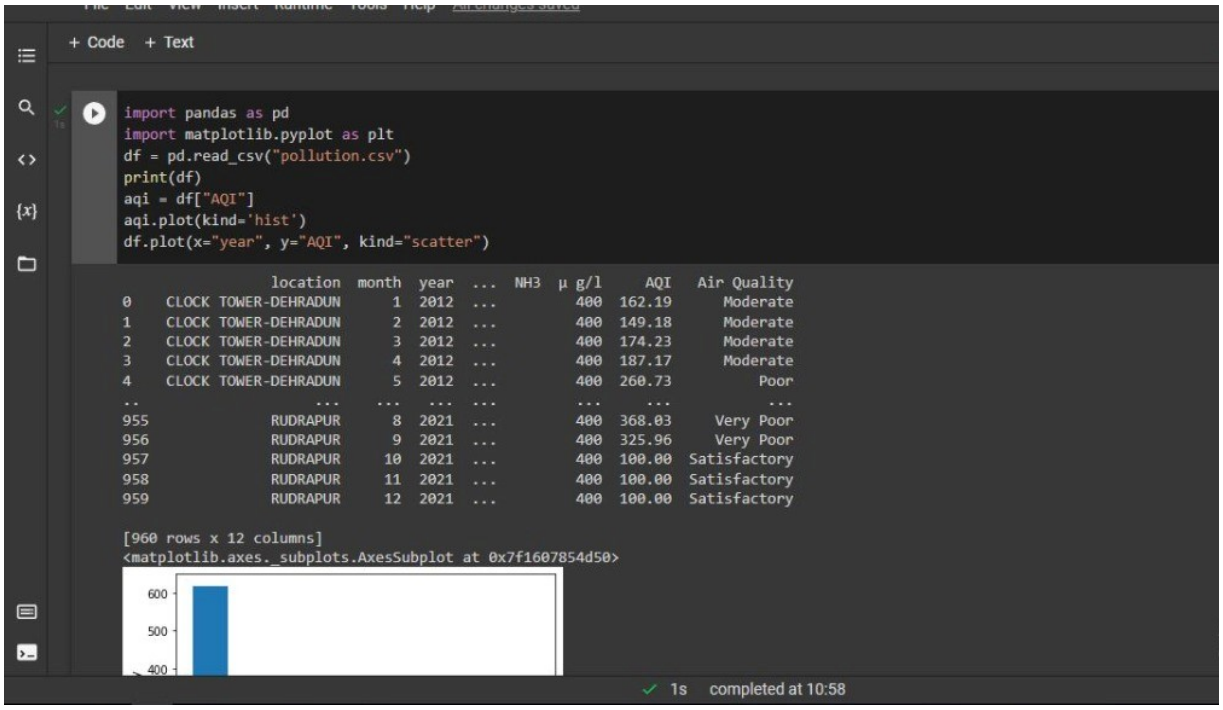
```
"data": {
 "text/html": [
  "<div>\n",
  "<style scoped>\n",
  "    .dataframe tbody tr th:only-of-type {\n",
  "        vertical-align: middle;\n",
  "    }\n",
  "\n",
  "    .dataframe tbody tr th {\n",
  "        vertical-align: top;\n",
  "    }\n",
  "\n",
  "    .dataframe thead th {\n",
  "        text-align: right;\n",
  "    }\n",
  "</style>\n",
  "<table border=\"1\" class=\"dataframe\">\n",
  "  <thead>\n",
  "    <tr style=\"text-align: right;\">\n",
  "      <th></th>\n",
  "      <th>month</th>\n",
  "      <th>year</th>\n",
  "      <th>SO2 µg/l</th>\n",
  "      <th>NO2µg/l</th>\n",
  "      <th>PM10 µg/l</th>\n",
  "      <th>PM2.5 µ g/l</th>\n",
  "      <th>CO µg/l</th>\n",
  "      <th>O3 µ g/l 8 HR</th>\n",
  "      <th>NH3  µ g/l</th>\n",
  "      <th>AQI</th>\n",
  "      <th>0</th>\n",
  "      <th>1</th>\n",
  "      <th>2</th>\n",
  "      <th>3</th>\n",
  "      <th>4</th>\n",
  "      <th>5</th>\n",
  "      <th>6</th>\n",
  "      <th>7</th>\n",
  "    </tr>\n",
  "  </thead>\n",
  "  <tbody>\n",
  "    <tr>\n",
  "      <th>0</th>\n",
  "      <td>1</td>\n",
  "      <td>2012</td>\n",
  "      <td>27.33</td>\n",
  "      <td>30.33</td>\n",
  "      <td>193.28</td>\n",
  "     <td>60.0</td>\n",
  "     <td>2</td>\n",
  "     <td>100</td>\n",
  "     <td>400</td>\n",
  "     <td>162.19</td>\n",
  "     <td>1.0</td>\n",
  "     <td>0.0</td>\n",
  "     <td>0.0</td>\n",
  "     <td>0.0</td>\n",
  "     <td>0.0</td>\n",
  "     <td>0.0</td>\n",
  "     <td>0.0</td>\n",
  "     <td>0.0</td>\n",
  "   </tr>\n",
  "   <tr>\n",
  "     <th>1</th>\n",
  "     <td>2</td>\n",
  "     <td>2012</td>\n",
  "     <td>25.68</td>\n",
  "     <td>25.80</td>\n",
  "     <td>173.77</td>\n",
  "     <td>60.0</td>\n",
  "     <td>2</td>\n",
  "     <td>100</td>\n",
  "     <td>400</td>\n",
  "     <td>149.18</td>\n",
  "     <td>1.0</td>\n",
  "     <td>0.0</td>\n",
  "     <td>0.0</td>\n",
  "     <td>0.0</td>\n",
  "     <td>0.0</td>\n",
  "     <td>0.0</td>\n",
  "     <td>0.0</td>\n",
  "     <td>0.0</td>\n",
  "   </tr>\n",
  "   <tr>\n",
  "     <th>2</th>\n",
  "     <td>3</td>\n",
  "     <td>2012</td>\n",
  "     <td>29.64</td>\n",
  "     <td>27.50</td>\n",
  "     <td>211.35</td>\n",
  "     <td>60.0</td>\n",
  "     <td>2</td>\n",
  "     <td>100</td>\n",
  "     <td>400</td>\n",
  "     <td>174.23</td>\n",
  "     <td>1.0</td>\n",
  "     <td>0.0</td>\n",
  "     <td>0.0</td>\n",
  "     <td>0.0</td>\n",
  "     <td>0.0</td>\n",
  "     <td>0.0</td>\n",
  "     <td>0.0</td>\n",
  "     <td>0.0</td>\n",
  "   </tr>\n",
  "   <tr>\n",
  "     <th>3</th>\n",
  "     <td>4</td>\n",
  "     <td>2012</td>\n",
  "     <td>28.64</td>\n",
  "     <td>26.81</td>\n",
  "     <td>230.76</td>\n",
  "     <td>60.0</td>\n",
  "     <td>2</td>\n",
  "     <td>100</td>\n",
  "     <td>400</td>\n",
  "     <td>187.17</td>\n",
  "     <td>1.0</td>\n",
  "     <td>0.0</td>\n",
  "     <td>0.0</td>\n",
  "     <td>0.0</td>\n",
  "     <td>0.0</td>\n",
  "     <td>0.0</td>\n",
  "     <td>0.0</td>\n",
  "     <td>0.0</td>\n",
  "   </tr>\n",
  "   <tr>\n",
  "     <th>4</th>\n",
  "     <td>5</td>\n",
  "     <td>2012</td>\n",
  "     <td>31.09</td>\n",
  "     <td>29.30</td>\n",
  "     <td>310.73</td>\n",
  "     <td>60.0</td>\n",
  "     <td>2</td>\n",
  "     <td>100</td>\n",
  "     <td>400</td>\n",
  "     <td>260.73</td>\n",
  "     <td>1.0</td>\n",
  "     <td>0.0</td>\n",
  "     <td>0.0</td>\n",
  "     <td>0.0</td>\n",
  "     <td>0.0</td>\n",
  "     <td>0.0</td>\n",
  "     <td>0.0</td>\n",
  "     <td>0.0</td>\n",
  "    </tr>\n",
  "  </tbody>\n",
  "</table>\n",
  "</div>"
  ],
  "text/plain": [
  "   month  year  SO2 µg/l  NO2µg/l  PM10 µg/l  PM2.5 µ g/l  CO µg/l  \\\n",
  "0      1  2012     27.33    30.33     193.28         60.0        2   \n",
  "1      2  2012     25.68    25.80     173.77         60.0        2   \n",
  "2      3  2012     29.64    27.50     211.35         60.0        2   \n",
  "3      4  2012     28.64    26.81     230.76         60.0        2   \n",
  "4      5  2012     31.09    29.30     310.73         60.0        2   \n",
  "\n",
  "   O3 µ g/l 8 HR  NH3  µ g/l     AQI    0    1    2    3    4    5    6    7  \n",
  "0            100          400  162.19  1.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  \n",
  "1            100          400  149.18  1.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  \n",
  "2            100          400  174.23  1.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  \n",
  "3            100          400  187.17  1.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  \n",
  "4            100          400  260.73  1.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  "
  ]
  },
  "execution_count": 28,
  "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "X.head(5)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 29,
   "id": "9c964fd5",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "0    0\n",
       "1    0\n",
       "2    0\n",
       "3    0\n",
       "4    1\n",
       "Name: Air Quality, dtype: int32"
      ]
     },
     "execution_count": 29,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "y.head(5)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 30,
   "id": "21c2d2a0",
   "metadata": {},
   "outputs": [],
   "source": [
    "# Splitting the dataset into the Training set and Test set\n",
    "from sklearn.model_selection import train_test_split\n",
    "X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.20, random_state = 0)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 31,
    "id": "7ffce8ff",
    "metadata": {},
    "outputs": [],
    "source": [
     "# Feature Scaling\n",
     "from sklearn.preprocessing import StandardScaler\n",
     "sc = StandardScaler()\n",
     "X_train = sc.fit_transform(X_train)\n",
     "X_test = sc.transform(X_test)\n",
     "\n"
    ]
   },
   {
    "cell_type": "code",
    "execution_count": null,
    "id": "7cbe33a9",
    "metadata": {},
    "outputs": [],
    "source": []
   },
   {
    "cell_type": "code",
    "execution_count": 36,
    "id": "e11e1826",
   "metadata": {},
   "outputs": [],
   "source": [
    "# Fitting Decision Tree Classification to the Training set\n",
    "from sklearn.tree import DecisionTreeClassifier\n",
    "classifier = DecisionTreeClassifier(criterion = 'entropy', random_state = 0)\n",
    "classifier.fit(X_train, y_train)\n",
    "\n",
    "# Predicting the Test set results\n",
    "y_pred = classifier.predict(X_test)\n",
    "\n",
    "from sklearn.metrics import confusion_matrix\n",
    "from sklearn.metrics import accuracy_score"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 37,
   "id": "eccfc2b7",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
      "0.9895833333333334"
     ]
    },
    "execution_count": 37,
    "metadata": {},
    "output_type": "execute_result"
   }
  ],
  "source": [
   "#printing the accuracy of Decision tree\n",
   "accuracy_score(y_test, y_pred)"
  ]
  },
  {
  "cell_type": "code",
  "execution_count": 38,
  "id": "4be837a6",
  "metadata": {},
  "outputs": [],
  "source": [
   "# Fitting SVM to the Training set\n",
   "from sklearn.svm import SVC\n",
   "classifier = SVC(kernel = 'linear', random_state = 0)\n",
   "classifier.fit(X_train, y_train)\n",
    "\n",
    "# Predicting the Test set results\n",
    "y_pred = classifier.predict(X_test)\n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 40,
   "id": "fc547a72",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "0.9739583333333334"
      ]
     },
     "execution_count": 40,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "#Accuracy of SVM\n",
    "accuracy_score(y_test, y_pred)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 41,
   "id": "eeb25178",
   "metadata": {},
   "outputs": [],
   "source": [
    "#fitting knn model \n",
    "from sklearn.neighbors import KNeighborsClassifier\n",
    "classifier=KNeighborsClassifier(n_neighbors=5,metric='minkowski',p=2)\n",
    "classifier.fit(X_train,y_train)\n",
    "\n",
    "# Predicting the Test set results\n",
    "y_pred = classifier.predict(X_test)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 42,
   "id": "5aa5afdf",
   "metadata": {},
   "outputs": [],
   "source": [
    "# Predicting the Test set results\n",
    "y_pred = classifier.predict(X_test)\n",
    "\n",
    "# Making the Confusion Matrix\n",
    "from sklearn.metrics import confusion_matrix\n",
    "cm = confusion_matrix(y_test, y_pred)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 43,
   "id": "f3891fd2",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "0.875"
      ]
     },
     "execution_count": 43,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "#Accuracy of knn\n",
    "accuracy_score(y_test, y_pred)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "c1908c18",
   "metadata": {},
   "outputs": [],
   "source": []
  }
  ],
  "metadata": {
   "kernelspec": {
    "display_name": "Python 3",
    "language": "python",
    "name": "python3"
   },
   "language_info": {
     "codemirror_mode": {
      "name": "ipython",
      "version": 3
     },
     "file_extension": ".py",
     "mimetype": "text/x-python",
     "name": "python",
     "nbconvert_exporter": "python",
     "pygments_lexer": "ipython3",
     "version": "3.8.8"
    }
   },
   "nbformat": 4,
   "nbformat_minor": 5
  }
```

OUTPUT :



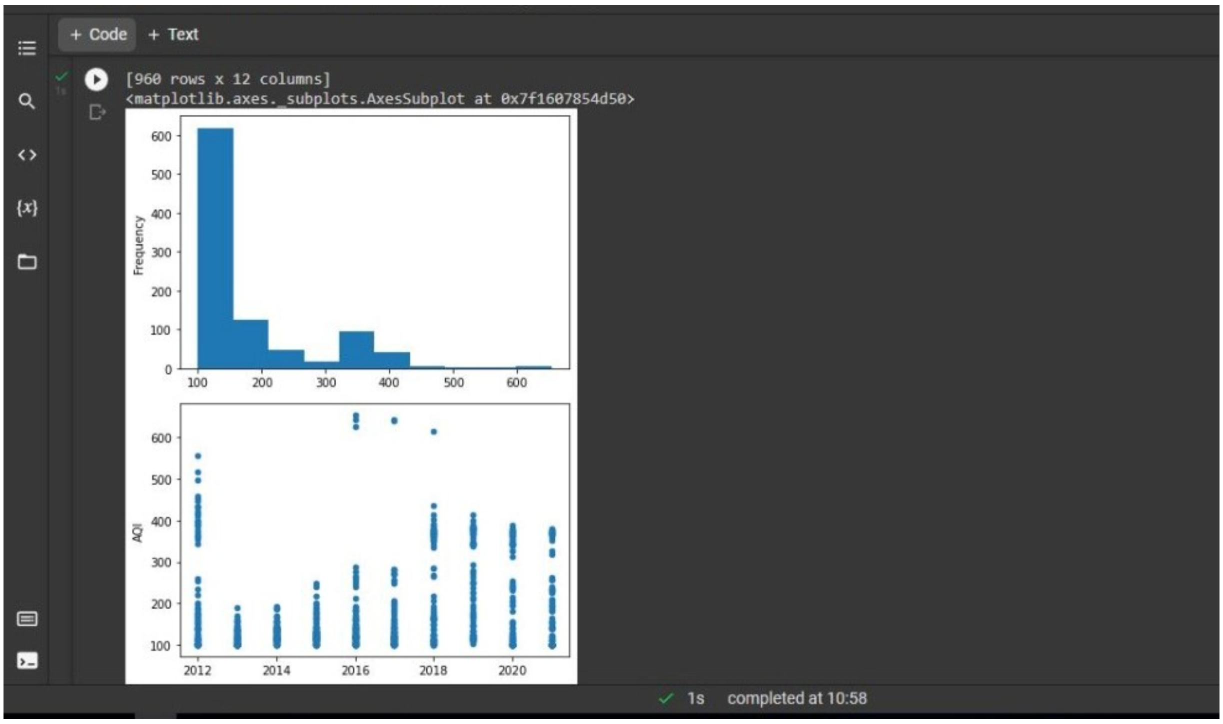
```
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
import pandas as pd
pf = pd.read_csv("pollution.csv")
train, test = train_test_split(pf, test_size=0.2, random_state=33, shuffle=True)
print(test)
print(train)
```

```
                location  month  year  ...  NH3  µ g/l     AQI   Air Quality
860             RUDRAPUR      9  2013  ...          400  102.49      Moderate
756             KASHIPUR      1  2015  ...          400  142.72      Moderate
479  RISHIKESH-NAGARNIGAM    12  2021  ...          400  100.00  Satisfactory
841             RUDRAPUR      2  2012  ...          400  156.77      Moderate
684             HALDWANI      1  2019  ...          400  349.28     Very Poor
..                   ...    ...   ...  ...          ...     ...           ...
834             KASHIPUR      7  2021  ...          400  370.07     Very Poor
554       SIDCUL-HARIDWAR     3  2018  ...          400  109.99      Moderate
446  RISHIKESH-NAGARNIGAM     3  2019  ...          400  121.11      Moderate
767             KASHIPUR     12  2015  ...          400  125.12      Moderate
896             RUDRAPUR      9  2016  ...          400  100.00  Satisfactory

[192 rows x 12 columns]
                location  month  year  ...  NH3  µ g/l     AQI   Air Quality
858             RUDRAPUR      7  2013  ...          400  100.00  Satisfactory
771             KASHIPUR      4  2016  ...          400  131.37      Moderate
234  RAIPUR ROAD DEHRADUN     7  2021  ...          400  145.47      Moderate
781             KASHIPUR      2  2017  ...          400  114.97      Moderate
149  RAIPUR ROAD DEHRADUN     6  2014  ...          400  135.86      Moderate
```

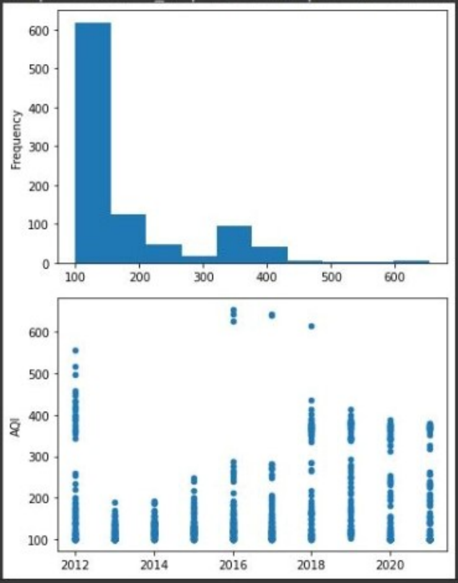


```
import pandas as pd
import matplotlib.pyplot as plt
df = pd.read_csv("pollution.csv")
print(df)
aqi = df["AQI"]
aqi.plot(kind='hist')
df.plot(x="year", y="AQI", kind="scatter")
```

```
                location  month  year  ...  NH3  µ g/l     AQI   Air Quality
0    CLOCK TOWER-DEHRADUN     1  2012  ...          400  162.19      Moderate
1    CLOCK TOWER-DEHRADUN     2  2012  ...          400  149.18      Moderate
2    CLOCK TOWER-DEHRADUN     3  2012  ...          400  174.23      Moderate
3    CLOCK TOWER-DEHRADUN     4  2012  ...          400  187.17      Moderate
4    CLOCK TOWER-DEHRADUN     5  2012  ...          400  260.73          Poor
..                   ...    ...   ...  ...          ...     ...           ...
955             RUDRAPUR      8  2021  ...          400  368.03     Very Poor
956             RUDRAPUR      9  2021  ...          400  325.96     Very Poor
957             RUDRAPUR     10  2021  ...          400  100.00  Satisfactory
958             RUDRAPUR     11  2021  ...          400  100.00  Satisfactory
959             RUDRAPUR     12  2021  ...          400  100.00  Satisfactory

[960 rows x 12 columns]
<matplotlib.axes._subplots.AxesSubplot at 0x7f1607854d50>
```



```
[960 rows x 12 columns]
<matplotlib.axes._subplots.AxesSubplot at 0x7f1607854d50>
```



✓ 1s completed at 10:58