MAJOR PROJECT

MACHINE LEARNING

TOPIC: - TAKE ANY DATASET OF YOUR CHOICE, PERFORM EDA (EXPLORATORY DATA ANALYSIS) AND APPLY A SUITABLE CLASSIFIER, REGRESSOR OR CLUSTERER AND CALCULATE THE ACCURACY OF THE MODEL.

In this project we have asked to take a dataset to perform exploratory data analysis and apply a suitable classifier regressor or clusterer and to calculate the accuracy of the model.

For this project, I took a database containing information about passengers who were aboard the Titanic ship. The data set provides information about each passenger's, age, gender, class, fare, cabin, and whether or not they survived the disaster. Then I performed the necessary Exploratory Data Analysis, like Data Summary, missing values analysis, etc. The last step was model building, which will use the data and evaluate the performance of the model.

Explanation of the source code

The first two lines import the required libraries, **pandas** and **numpy**. Source is a string variable containing the URL of the dataset, which is then used in the **pd.read_csv()** function to read the dataset into a pandas DataFrame called Data.

Then '**print(Data.head())**' prints the first few rows of the dataset, '**print(Data.dtypes)**' prints the data types of the columns in the dataset, '**print("The shape of the data: ",Data.shape)**' prints the shape of the dataset, which is the number of rows and columns in the DataFrame, '**print("The size of the data: ",Data.size)**' prints the size of the dataset, which is the total number of elements in the DataFrame, '**print(Data.info)**' prints a summary of the DataFrame, including the number of non-null values, the data types of the columns, and the memory usage. and then '**print(Data[32:46])**' slices the DataFrame from row 32 to row 46 and prints the resulting subset of the data, '**print(Data.iloc[32:46,0:4])**' slices the DataFrame from row 32 to row 46 also slices from column 0 to column 3 and prints the resulting subset of the data, '**print(Data.Sex.nunique())**' prints the number of unique values in the 'Sex' column of the DataFrame. '**print(Data.Sex.unique())**' prints the unique values in the 'Sex' column of the DataFrame,which are male and female, '**print(Data.groupby('Survived').size())**' groups the DataFrame by the 'Survived' column and prints the count of each group, '**print(Data.isnull().sum())**' prints the count of missing values in each column of the DataFrame. '**print(Data.isnull().mean() * 100)**' prints the percentage of missing values in each column of the DataFrame. '**print(Data.describe())**' prints the summary statistics of the DataFrame, including the count, mean, standard deviation, minimum, maximum, and quartile values of the numerical columns.

After that '**Data.drop(['PassengerId', 'Name', 'Ticket', 'Cabin'], axis=1, inplace=True)**' drops the columns 'PassengerId', 'Name', 'Ticket', and 'Cabin' from the DataFrame, since they are not relevant for the analysis. Then '**Data['Sex'] = np.where(Data['Sex'] == 'male', 0, 1)**' converts the 'Sex' column to binary values, where 0 represents male and 1 represents female. Now '**Data['Age'].fillna(Data['Age'].median(), inplace=True)**' fills the missing values in the 'Age' column with the median age of the dataset and '**Data['Embarked'].fillna(Data['Embarked'].mode()[0], inplace=True)**' fills the missing values in the 'Embarked' column with the mode of the column.'**Data['Embarked'] = Data['Embarked'].map({'S': 0, 'C': 1, 'Q': 2})**' it converts the 'Embarked' column to numerical values, where 0 represents 'S', 1 represents 'C', and 2 represents 'Q'

The P variable contains the features of the dataset, which are all the columns except the Survived column. The Q variable contains the target variable, which is the Survived column. This code uses the '**train_test_split**' function from Scikit-Learn to split the dataset into training and testing sets. The '**test_size**' parameter is set to 0.2, which means that 20% of the data will be used for testing, and the remaining 80% will be used for training. The '**random_state**' parameter is set to 42 to ensure that the random splitting is reproducible. '**clf = RandomForestClassifier(n_estimators=100, random_state=42)**'creates an instance of the Random Forest Classifier with 100 decision trees and fits the model on the training data. The '**random_state**' parameter is set to 42 to ensure that the random initialization of the model is reproducible.

At the end, the model is trained to predict the survival of passengers in the testing set '**(P_test)**'. The predicted values are stored in '**Q_pred**'. The accuracy of the model is then calculated by comparing the predicted values with the actual values '**(Q_test)**'. The '**accuracy_score**' function from Scikit-Learn is used to calculate the accuracy. Finally, the accuracy is printed on the console.

SOURCE CODE: -

```
import pandas as pd
import numpy as np

Source = "https://raw.githubusercontent.com/subhajyoti-prusty/publicSubhajyoti/main/Dataset.csv"

Data = pd.read_csv(Source)

# Show the entire dataset
print(Data)

# Check the first few rows of the dataset
print(Data.head())

# Check the data types of the columns
print(Data.dtypes)

# Check the shape of the dataset
print("The shape of the data: ",Data.shape)

# Check the size of the dataset
print("The size of the data: ",Data.size)

# Check the info of the dataset
print(Data.info())

# Slicing the dataset form row 32 to 46
print(Data[32:46])

# Slicing the dataset form row 32 to 46 and column index 0 to 3
print(Data.iloc[32:46,0:4])

#Check the number of unique value of the dataset
print("The number of unique values the sex column has is",Data.Sex.nunique())

#Check the unique value of the dataset
print("The unique values the sex column has is",Data.Sex.unique())

#Group by survived or not (survived=1 and Died=0)
print(Data.groupby('Survived').size())

# Check for missing values
print(Data.isnull().sum())

# Compute the percentage of missing values in each column
print(Data.isnull().mean() * 100)

# Check the summary statistics of the dataset
print(Data.describe())

# Remove the unnecessary columns
Data.drop(['PassengerId', 'Name', 'Ticket', 'Cabin'], axis=1, inplace=True)

# Convert the Sex column to binary values (0 = male, 1 = female)
Data['Sex'] = np.where(Data['Sex'] == 'male', 0, 1)

# Fill missing Age values with the median
Data['Age'].fillna(Data['Age'].median(), inplace=True)
```

```
# Fill missing Embarked values with the mode
Data['Embarked'].fillna(Data['Embarked'].mode()[0], inplace=True)

# Convert the Embarked column to numerical values (0 = S, 1 = C, 2 = Q)
Data['Embarked'] = Data['Embarked'].map({'S': 0, 'C': 1, 'Q': 2})

P = Data.drop('Survived', axis=1)
Q = Data['Survived']

from sklearn.model_selection import train_test_split

P_train, P_test, Q_train, Q_test = train_test_split(P, Q, test_size=0.2, random_state=42)

from sklearn.ensemble import RandomForestClassifier

clf = RandomForestClassifier(n_estimators=100, random_state=42)
clf.fit(P_train, Q_train)
from sklearn.metrics import accuracy_score

Q_pred = clf.predict(P_test)
accuracy = accuracy_score(Q_test, Q_pred)

print("Accuracy: {:.2f}%".format(accuracy*100))
```

OUTPUT

	Passenger Id	Survived	Pclass	Name	Sex
0	1	0	3	Braund, Mr. Owen Harris	male
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female
2	3	1	3	Heikkinen, Miss. Laina	female
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female
4	5	0	3	Allen, Mr. William Henry	male
..	...	...	...	…….	…….
886	887	0	2	Montvila, Rev. Juozas	male
887	888	1	1	Graham, Miss. Margaret Edith	female
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female
889	890	1	1	Behr, Mr. Karl Howell	male
890	891	0	3	Dooley, Mr. Patrick	male

	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	22.0	1	0	A/5 21171	7.2500	NaN	S
1	38.0	1	0	PC 17599	71.2833	C85	C
2	26.0	0	0	STON/O2.3101282	7.9250	NaN	S
3	35.0	1	0	113803	53.1000	C123	S
4	35.0	0	0	373450	8.0500	NaN	S
..	...	...	...	...	...	...	...
886	27.0	0	0	211536	13.0000	NaN	S
887	19.0	0	0	112053	30.0000	B42	S
888	NaN	1	2	W./C. 6607	23.4500	NaN	S
889	26.0	0	0	111369	30.0000	C148	C
890	32.0	0	0	370376	7.7500	NaN	Q

[891 rows x 12 columns]

	Passenger Id	Survived	Pclass	Name	Sex	Age	SibSp
0	1	0	3	Braund,Mr. Owen Harris	male	22.0	1
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1
4	5	0	3	Allen, Mr. William Henry	male	35.0	0

	Parch	Ticket	Fare	Cabin	Embarked
0	0	A/5 21171	7.2500	NaN	S
1	0	PC 17599	71.2833	C85	C
2	0	STON/O2. 3101282	7.9250	NaN	S
3	0	113803	53.1000	C123	S
4	0	373450	8.0500	NaN	S

```
Passenger Id   int64
Survived       int64
Pclass         int64
Name           object
Sex            object
Age            float64
SibSp          int64
Parch          int64
Ticket         object
Fare           float64
Cabin          object
Embarked       object
dtype: object
```

The shape of the data: (891, 12)
The size of the data: 10692

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):

#	Column	Non-NullCount	Dtype

---	----------	------------------	--------
0	Passenger Id	891non-null	int64
1	Survived	891 non-null	int64
2	Pclass	891 non-null	int64
3	Name	891 non-null	object
4	Sex	891 non-null	object
5	Age	714 non-null	float64
6	SibSp	891 non-null	int64
7	Parch	891 non-null	int64
8	Ticket	891 non-null	object
9	Fare	891 non-null	float64
10	Cabin	204 non-null	object
11	Embarked	889 non-null	object

dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
None

	Passenger Id	Survived	Pclass	Name	Sex	Age	SibSp
32	33	1	3	Glynn, Miss. Mary Agatha	female	NaN	0
33	34	0	2	Wheadon, Mr. Edward H	male	66.0	0
34	35	0	1	Meyer, Mr. Edgar Joseph	male	28.0	1
35	36	0	1	Holverson, Mr. Alexander Oskar	male	42.0	1
36	37	1	3	Mamee, Mr. Hanna	male	NaN	0
37	38	0	3	Cann, Mr. Ernest Charles	male	21.0	0
38	39	0	3	Vander Planke, Miss. Augusta Maria	female	18.0	2
39	40	1	3	Nicola-Yarred, Miss. Jamila	female	14.0	1
40	41	0	3	Ahlin, Mrs. Johan (Johanna Persdotter Larsson)	female	40.0	1
41	42	0	2	Turpin, Mrs. William John Robert (Dorothy Ann ...	female	27.0	1
42	43	0	3	Kraeff, Mr. Theodor	male	NaN	0
43	44	1	2	Laroche, Miss. Simonne Marie Anne Andree	female	3.0	1
44	45	1	3	Devaney, Miss. Margaret Delia	female	19.0	0
45	46	0	3	Rogers, Mr. William John	male	NaN	0

	Parch	Ticket	Fare	Cabin	Embarked
32	0	335677	7.7500	NaN	Q
33	0	C.A. 24579	10.5000	NaN	S
34	0	PC 17604	82.1708	NaN	C
35	0	113789	52.0000	NaN	S
36	0	2677	7.2292	NaN	C
37	0	A./5. 2152	8.0500	NaN	S
38	0	345764	18.0000	NaN	S
39	0	2651	11.2417	NaN	C
40	0	7546	9.4750	NaN	S
41	0	11668	21.0000	NaN	S
42	0	349253	7.8958	NaN	C
43	2	SC/Paris 2123	41.5792	NaN	C
44	0	330958	7.8792	NaN	Q
45	0	S.C./A.4. 23567	8.0500	NaN	S

	Passenger Id	Survived	Pclass	Name
32	33	1	3	Glynn, Miss. Mary Agatha
33	34	0	2	Wheadon, Mr. Edward H
34	35	0	1	Meyer, Mr. Edgar Joseph
35	36	0	1	Holverson, Mr. Alexander Oskar
36	37	1	3	Mamee, Mr. Hanna
37	38	0	3	Cann, Mr. Ernest Charles
38	39	0	3	Vander Planke, Miss. Augusta Maria
39	40	1	3	Nicola-Yarred, Miss. Jamila
40	41	0	3	Ahlin, Mrs. Johan (Johanna Persdotter Larsson)
41	42	0	2	Turpin, Mrs. William John Robert (Dorothy Ann ...
42	43	0	3	Kraeff, Mr. Theodor
43	44	1	2	Laroche, Miss. Simonne Marie Anne Andree
44	45	1	3	Devaney, Miss. Margaret Delia
45	46	0	3	Rogers, Mr. William John

The number of unique values the sex column has is 2
The unique values the sex column has is ['male' 'female']

	Survived
0	549
1	342

dtype: int64

Passenger Id	0
Survived	0
Pclass	0
Name	0
Sex	0
Age	177
SibSp	0
Parch	0
Ticket	0
Fare	0
Cabin	687
Embarked	2

dtype: int64

Passenger Id	0.000000
Survived	0.000000
Pclass	0.000000
Name	0.000000
Sex	0.000000
Age	19.865320
SibSp	0.000000
Parch	0.000000
Ticket	0.000000
Fare	0.000000
Cabin	77.104377
Embarked	0.224467

dtype: float64

	Passenger Id	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

Accuracy: 82.68%

Link of the dataset used in the project :-

https://raw.githubusercontent.com/subhajyoti-prusty/publicSubhajyoti/main/Dataset.csv

(The link is taken from https://github.com/)

SUBMITTED BY: - SUBHAJYOTI PRUSTY