Programming for Artificial Intelligence

SUPERIOR UNIVERSITY

Name:

Mahak Farhan

Roll no.:

068

Class:

BSAI

Section:

4B

Subject:

Programming for Artifcial Intelligence

Submitted to:

Sir Rasikh Ali

Programming for Artificial Intelligence

Lab 2

Spaceship Titanic

Code:

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
```

Reading training data:

```
train_df = pd.read_csv(r"C:\Users\Hamza Computer\Desktop\spaceship-titanic\train.csv")
train_df
```

In [72]:
```
train_df = pd.read_csv(r"C:\Users\Hamza Computer\Desktop\spaceship-titanic\train.csv")
train_df
```

Out[72]:

	PassengerId	HomePlanet	CryoSleep	Cabin	Destination	Age	VIP	RoomService	FoodCourt	ShoppingMall	Spa	VRDeck	Name	Transporte
0	0001_01	Europa	False	B/0/P	TRAPPIST-1e	39.0	False	0.0	0.0	0.0	0.0	0.0	Maham Ofracculy	Fal
1	0002_01	Earth	False	F/0/S	TRAPPIST-1e	24.0	False	109.0	9.0	25.0	549.0	44.0	Juanna Vines	Tru
2	0003_01	Europa	False	A/0/S	TRAPPIST-1e	58.0	True	43.0	3576.0	0.0	6715.0	49.0	Altark Susent	Fal
3	0003_02	Europa	False	A/0/S	TRAPPIST-1e	33.0	False	0.0	1283.0	371.0	3329.0	193.0	Solam Susent	Fal
4	0004_01	Earth	False	F/1/S	TRAPPIST-1e	16.0	False	303.0	70.0	151.0	565.0	2.0	Willy Santantines	Tru
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
8688	9276_01	Europa	False	A/98/P	55 Cancri e	41.0	True	0.0	6819.0	0.0	1643.0	74.0	Gravior Noxnuther	Fal
8689	9278_01	Earth	True	G/1499/S	PSO J318.5-22	18.0	False	0.0	0.0	0.0	0.0	0.0	Kurta Mondalley	Fal

Checking null values of training data:

train_df.isnull().sum()

```
In [74]: train_df.isnull().sum()

Out[74]: PassengerId       0
         HomePlanet      201
         CryoSleep       217
         Cabin           199
         Destination     182
         Age             179
         VIP             203
         RoomService     181
         FoodCourt       183
         ShoppingMall    208
         Spa             183
         VRDeck          188
         Name            200
         Transported       0
         dtype: int64
```

Information of training data:

train_df.info()

```
In [75]: train_df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8693 entries, 0 to 8692
Data columns (total 14 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   PassengerId   8693 non-null   object
 1   HomePlanet    8492 non-null   object
 2   CryoSleep     8476 non-null   object
 3   Cabin         8494 non-null   object
 4   Destination   8511 non-null   object
 5   Age           8514 non-null   float64
 6   VIP           8490 non-null   object
 7   RoomService   8512 non-null   float64
 8   FoodCourt     8510 non-null   float64
 9   ShoppingMall  8485 non-null   float64
 10  Spa           8510 non-null   float64
 11  VRDeck        8505 non-null   float64
 12  Name          8493 non-null   object
 13  Transported   8693 non-null   bool
dtypes: bool(1), float64(6), object(7)
memory usage: 891.5+ KB
```

Filling string columns of training data:

def fillNaObjMode(cols):

    for i in cols:

        train_df[i] = train_df[i].fillna(train_df[i].mode()[0])

columns = ['HomePlanet','CryoSleep','Cabin','Destination','VIP','Name']

fillNaObjMode(columns)

In [76]:
```
def fillNaObjMode(cols):
    for i in cols:
        train_df[i] = train_df[i].fillna(train_df[i].mode()[0])

columns = ['HomePlanet','CryoSleep','Cabin','Destination','VIP','Name']
fillNaObjMode(columns)
```

Filling float columns of training data:

def fillNaFloat(cols):

    for i in cols:

        train_df[i] = train_df[i].fillna(train_df[i].mean())

columns = ['Age','RoomService','FoodCourt','ShoppingMall','Spa','VRDeck']

fillNaFloat(columns)

In [77]:
```
def fillNaFloat(cols):
    for i in cols:
        train_df[i] = train_df[i].fillna(train_df[i].mean())

columns = ['Age','RoomService','FoodCourt','ShoppingMall','Spa','VRDeck']
fillNaFloat(columns)
```

Converting float columns into int:

```
def convertFloatintoInt(cols):
    for i in cols:
        train_df[i] = train_df[i].astype('int64')
columns =  ['Age','RoomService','FoodCourt','ShoppingMall','Spa','VRDeck']
convertFloatintoInt(columns)
```

In [78]:
```
def convertFloatintoInt(cols):
    for i in cols:
        train_df[i] = train_df[i].astype('int64')

columns =  ['Age','RoomService','FoodCourt','ShoppingMall','Spa','VRDeck']
convertFloatintoInt(columns)
```

Converting object columns into int:

```
def dataEncoder(cols):
    for i in cols:
        dataLabelEncoder = LabelEncoder()
        train_df[i] = dataLabelEncoder.fit_transform(train_df[i])
columns =
['PassengerId','HomePlanet','CryoSleep','Cabin','Destination','VIP','Name']
dataEncoder(columns)
```

In [79]:
```
def dataEncoder(cols):
    for i in cols:
        dataLabelEncoder = LabelEncoder()
        train_df[i] = dataLabelEncoder.fit_transform(train_df[i])

columns = ['PassengerId','HomePlanet','CryoSleep','Cabin','Destination','VIP','Name']
dataEncoder(columns)
```

Splitting the training data into X,y:

X_train = train_df.drop(['Transported', 'PassengerId'], axis=1)

y_train = train_df['Transported']

```
In [80]: X_train = train_df.drop(['Transported', 'PassengerId'], axis=1)
         y_train = train_df['Transported']
```

Training the model:

model = RandomForestClassifier(random_state=42)

model.fit(X_train, y_train)

```
In [81]: model = RandomForestClassifier(random_state=42)
         model.fit(X_train, y_train)
```

```
Out[81]: RandomForestClassifier
         RandomForestClassifier(random_state=42)
```

Reading the testing data:

test_df = pd.read_csv(r"C:\Users\Hamza Computer\Desktop\spaceship-titanic\test.csv")

test_df

```
In [82]: test_df = pd.read_csv(r"C:\Users\Hamza Computer\Desktop\spaceship-titanic\test.csv")
         test_df
```

Out[82]:

	PassengerId	HomePlanet	CryoSleep	Cabin	Destination	Age	VIP	RoomService	FoodCourt	ShoppingMall	Spa	VRDeck	Name
0	0013_01	Earth	True	G/3/S	TRAPPIST-1e	27.0	False	0.0	0.0	0.0	0.0	0.0	Nelly Carsoning
1	0018_01	Earth	False	F/4/S	TRAPPIST-1e	19.0	False	0.0	9.0	0.0	2823.0	0.0	Lerome Peckers
2	0019_01	Europa	True	C/0/S	55 Cancri e	31.0	False	0.0	0.0	0.0	0.0	0.0	Sabih Unhearfus
3	0021_01	Europa	False	C/1/S	TRAPPIST-1e	38.0	False	0.0	6652.0	0.0	181.0	585.0	Meratz Caltilter
4	0023_01	Earth	False	F/5/S	TRAPPIST-1e	20.0	False	10.0	0.0	635.0	0.0	0.0	Brence Harperez
...	...	...	...	...	...	...	...	...	...	...	...	...	...
4272	9266_02	Earth	True	G/1496/S	TRAPPIST-1e	34.0	False	0.0	0.0	0.0	0.0	0.0	Jeron Peter
4273	9269_01	Earth	False	NaN	TRAPPIST-1e	42.0	False	0.0	847.0	17.0	10.0	144.0	Matty Scheron
4274	9271_01	Mars	True	D/296/P	55 Cancri e	NaN	False	0.0	0.0	0.0	0.0	0.0	Jayrin Pore
4275	9273_01	Europa	False	D/297/P	NaN	NaN	False	0.0	2680.0	0.0	0.0	523.0	Kitakan Conale
4276	9277_01	Earth	True	G/1498/S	PSO J318.5-22	43.0	False	0.0	0.0	0.0	0.0	0.0	Lilace Leonzaley

Information of testing data:

test_df.info()

```
In [84]: test_df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4277 entries, 0 to 4276
Data columns (total 13 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   PassengerId   4277 non-null   object 
 1   HomePlanet    4190 non-null   object 
 2   CryoSleep     4184 non-null   object 
 3   Cabin         4177 non-null   object 
 4   Destination   4185 non-null   object 
 5   Age           4186 non-null   float64
 6   VIP           4184 non-null   object 
 7   RoomService   4195 non-null   float64
 8   FoodCourt     4171 non-null   float64
 9   ShoppingMall  4179 non-null   float64
 10  Spa           4176 non-null   float64
 11  VRDeck        4197 non-null   float64
 12  Name          4183 non-null   object 
dtypes: float64(6), object(7)
memory usage: 434.5+ KB
```

Checking null values of testing data:

test_df.isnull().sum()

```
In [85]: test_df.isnull().sum()

Out[85]: PassengerId      0
         HomePlanet      87
         CryoSleep       93
         Cabin          100
         Destination     92
         Age             91
         VIP             93
         RoomService     82
         FoodCourt      106
         ShoppingMall    98
         Spa            101
         VRDeck          80
         Name            94
         dtype: int64
```

Filling object columns:

```
def fillNaObjMode(cols):
    for i in cols:
        test_df[i] = test_df[i].fillna(test_df[i].mode()[0])
columns =
['PassengerId','HomePlanet','CryoSleep','Cabin','Destination','VIP','Name']
fillNaObjMode(columns)
```

Filling float columns:

```
def fillNaFloat(cols):
    for i in cols:
        test_df[i] = test_df[i].fillna(test_df[i].mean())
columns = ['Age','RoomService','FoodCourt','ShoppingMall','Spa','VRDeck']
fillNaFloat(columns)
```

In [86]:
```
def fillNaObjMode(cols):
    for i in cols:
        test_df[i] = test_df[i].fillna(test_df[i].mode()[0])

columns = ['PassengerId','HomePlanet','CryoSleep','Cabin','Destination','VIP','Name']
fillNaObjMode(columns)
```

In [87]:
```
def fillNaFloat(cols):
    for i in cols:
        test_df[i] = test_df[i].fillna(test_df[i].mean())

columns = ['Age','RoomService','FoodCourt','ShoppingMall','Spa','VRDeck']
fillNaFloat(columns)
```

Converting float columns:

```
def convertFloatintoInt(cols):
    for i in cols:
        test_df[i] = test_df[i].astype('int64')
columns =  ['Age','RoomService','FoodCourt','ShoppingMall','Spa','VRDeck']
convertFloatintoInt(columns)
```

Converting object columns:

```
def dataEncoder(cols):
    for i in cols:
        dataLabelEncoder = LabelEncoder()
        test_df[i] = dataLabelEncoder.fit_transform(test_df[i])
columns =
['PassengerId','HomePlanet','CryoSleep','Cabin','Destination','VIP','Name']
dataEncoder(columns)
```

In [88]:
```
def convertFloatintoInt(cols):
    for i in cols:
        test_df[i] = test_df[i].astype('int64')

columns =  ['Age','RoomService','FoodCourt','ShoppingMall','Spa','VRDeck']
convertFloatintoInt(columns)
```

In [89]:
```
feature_test = test_df.drop(['PassengerId'],axis=1)
```

In [90]:
```
def dataEncoder(cols):
    for i in cols:
        dataLabelEncoder = LabelEncoder()
        test_df[i] = dataLabelEncoder.fit_transform(test_df[i])

columns = ['PassengerId','HomePlanet','CryoSleep','Cabin','Destination','VIP','Name']
dataEncoder(columns)
```

Now testing the model on a feature:

```
feature_test = test_df.drop(['PassengerId'],axis=1)
```

Predicting on a model:

```
model_predictions = model.predict(feature_test)
```

Submission file:

```
submission1=pd.DataFrame({'PassengerId':x['PassengerId'],'Transported':mo
del_predictions})
submission1.to_csv('submission1.csv',index=False)
print("submission successfully")
```