Name: A.Keerthan

Reg no: 20BEC0620

Applied Data Science

Assignment-2

1. Load the dataset.

Ans: import pandas as pd

data=pd.read_csv('titanic.csv')

data.head()

```
In [4]: import pandas as pd
        import numpy as np
        import seaborn as sns
        data=pd.read_csv('titanic.csv')
        data.head()
```

Out[4]:

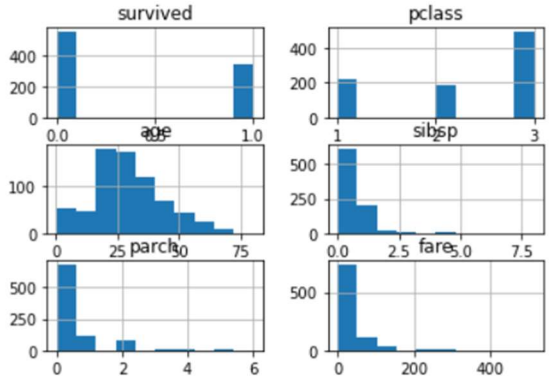
	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	NaN	Southampton	no	False
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	C	Cherbourg	yes	False
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	NaN	Southampton	yes	True
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	C	Southampton	yes	False
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	NaN	Southampton	no	True

2. Perform Below Visualizations.
   - Univariate Analysis

   Ans: data.hist()

```
In [5]: data.hist()
```

```
Out[5]: array([[<AxesSubplot:title={'center':'survived'}>,
                <AxesSubplot:title={'center':'pclass'}>],
               [<AxesSubplot:title={'center':'age'}>,
                <AxesSubplot:title={'center':'sibsp'}>],
               [<AxesSubplot:title={'center':'parch'}>,
                <AxesSubplot:title={'center':'fare'}>]], dtype=object)
```



● Bi - Variate Analysis

```
sns.boxplot(data, x = "age", y = "sex")
sns.boxplot(data, x = "age", y = "embark_town")
sns.boxplot(data, x = "fare", y = "sex")
copy = data
copy.head()
copy["pclass"] = copy["pclass"].replace(to_replace = [1,2,3], value =
["first","second","third"])
copy.head()
sns.boxplot(copy, x = "fare", y = "pclass")
x = data["age"]
y = data["fare"]
plt.scatter(x, y)
sns.displot(data, x = "age", hue = "alive", kind = "kde")
```

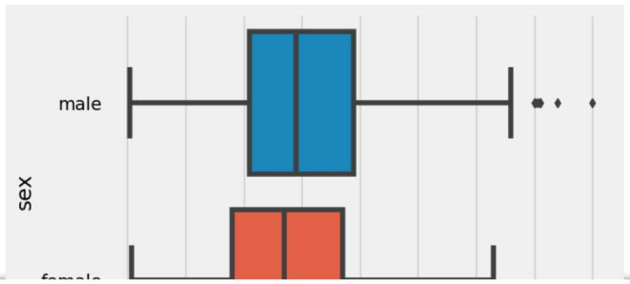
File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Run Code

Bi-variate analysis

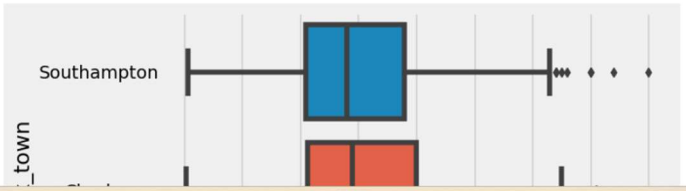
In [16]:
```
sns.boxplot(data, x = "age", y = "sex")
```

Out[16]: <Axes: xlabel='age', ylabel='sex'>



In [17]:
```
sns.boxplot(data, x = "age", y = "embark_town")
```

Out[17]: <Axes: xlabel='age', ylabel='embark_town'>



Search ENG IN

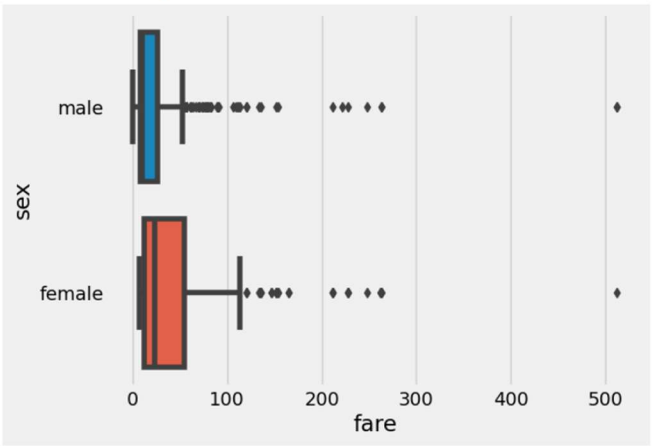
jupyter smartBridge assignment 2 (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Run Code

In [18]:
```
#analysing fares with respect to sex
sns.boxplot(data, x = "fare", y = "sex")
```

Out[18]: <Axes: xlabel='fare', ylabel='sex'>



In [19]:
```
# analysing passenger class and their respective fares
copy = data
copy.head()
```

Out[19]: survived pclass sex age sibsp parch fare embarked class who adult_male deck embark_town alive alone

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Run Code

In [19]:
```
# analysing passenger class and their respective fares
copy = data
copy.head()
```

Out[19]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	NaN	Southampton	no	False
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	C	Cherbourg	yes	False
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	NaN	Southampton	yes	True
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	C	Southampton	yes	False
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	NaN	Southampton	no	True

In [21]:
```
copy["pclass"] = copy["pclass"].replace(to_replace = [1,2,3], value = ["first","second","third"])
copy.head()
```

Out[21]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
0	0	third	male	22.0	1	0	7.2500	S	Third	man	True	NaN	Southampton	no	False
1	1	first	female	38.0	1	0	71.2833	C	First	woman	False	C	Cherbourg	yes	False
2	1	third	female	26.0	0	0	7.9250	S	Third	woman	False	NaN	Southampton	yes	True
3	1	first	female	35.0	1	0	53.1000	S	First	woman	False	C	Southampton	yes	False
4	0	third	male	35.0	0	0	8.0500	S	Third	man	True	NaN	Southampton	no	True

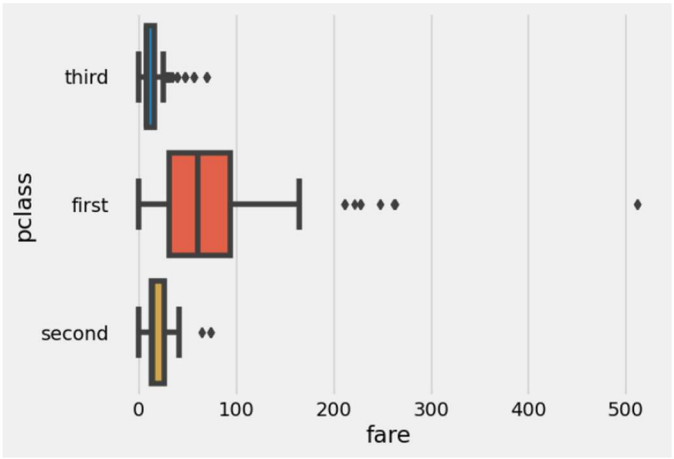
In [22]:
```
sns.boxplot(copy, x = "fare", y = "pclass")
```

Out[22]: `<Axes: xlabel='fare', ylabel='pclass'>`

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

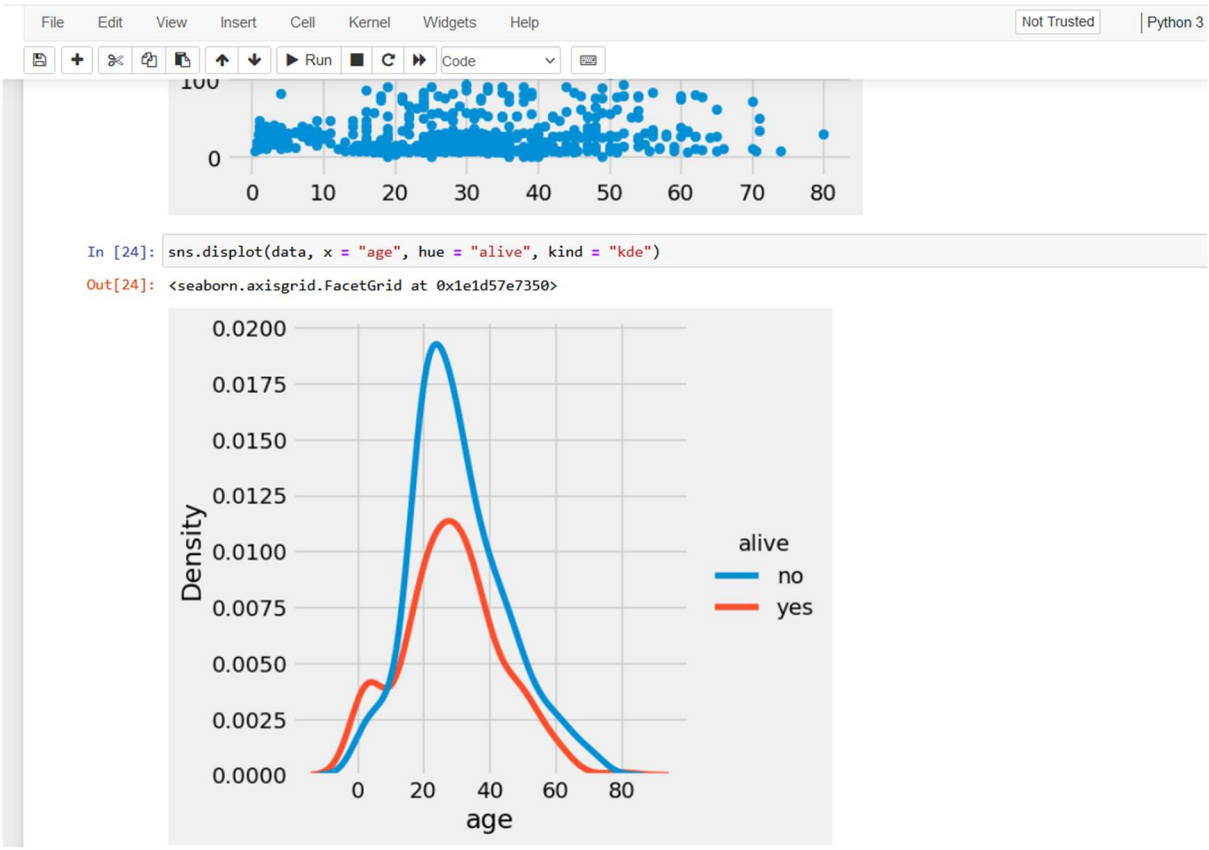
Run Code

Out[22]: <Axes: xlabel='fare', ylabel='pclass'>



In [23]:
```
x = data["age"]
y = data["fare"]
plt.scatter(x, y)
```

Out[23]: `<matplotlib.collections.PathCollection at 0x1e1d0e8ffd0>`



● Multi - Variate Analysis

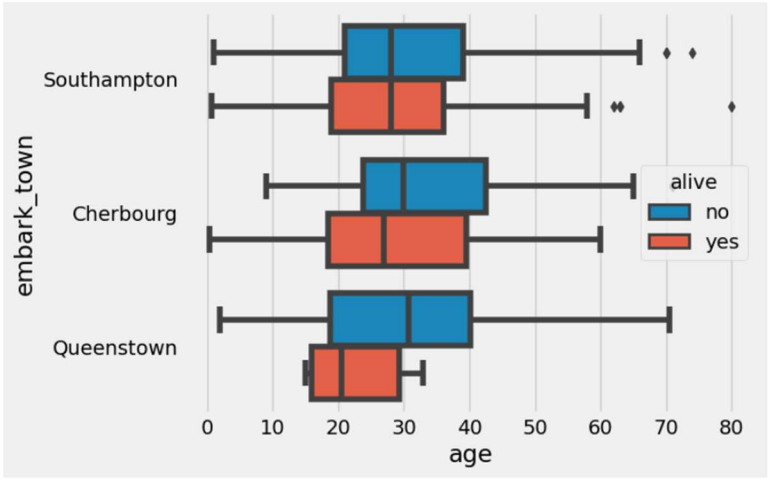
sns.boxplot(data, x = "age", y = "embark_town", hue = "alive")
sns.boxplot(data, x = "age", y = "pclass", hue = "alive")
sns.barplot(data, x = "pclass", y = "age", hue = "alive")

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Run Markdown

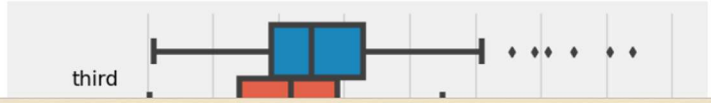
In [25]:
```
sns.boxplot(data, x = "age", y = "embark_town", hue = "alive")
```

Out[25]: <Axes: xlabel='age', ylabel='embark_town'>



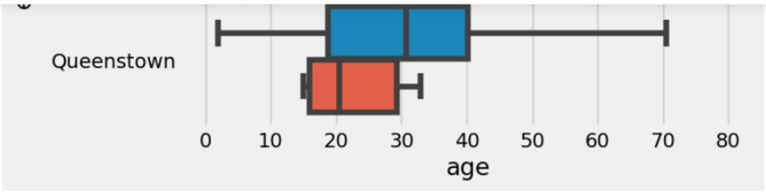
In [26]:
```
sns.boxplot(data, x = "age", y = "pclass", hue = "alive")
```

Out[26]: <Axes: xlabel='age', ylabel='pclass'>



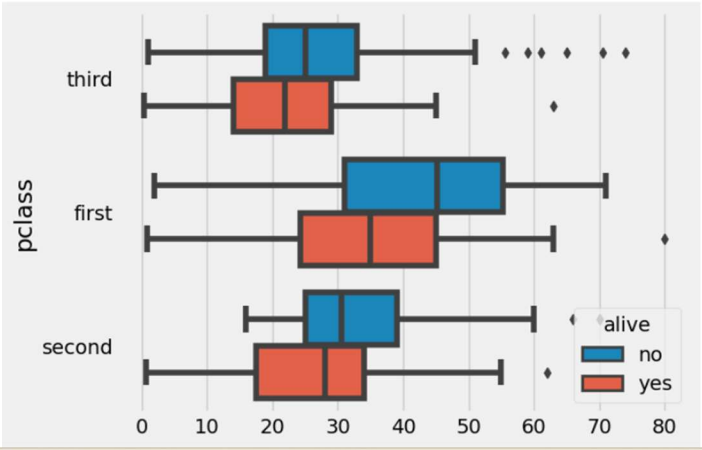
File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Run Markdown



In [26]:
```
sns.boxplot(data, x = "age", y = "pclass", hue = "alive")
```

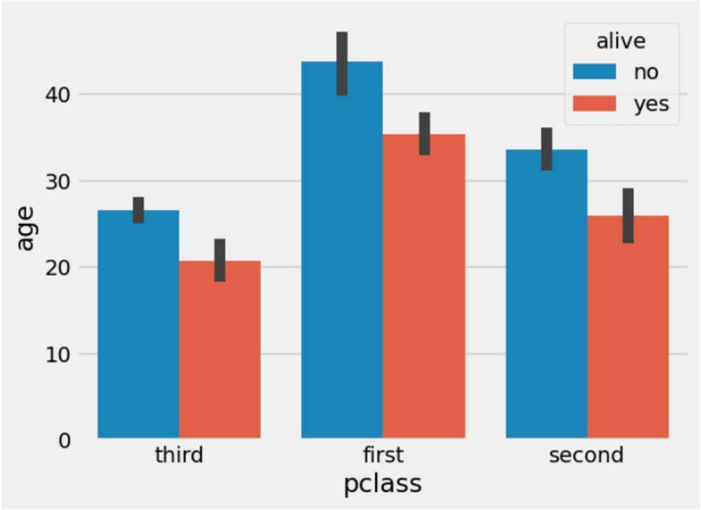
Out[26]: <Axes: xlabel='age', ylabel='pclass'>



File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipyl

Run Code

```
In [32]: sns.barplot(data, x = "pclass", y = "age", hue = "alive")
Out[32]: <Axes: xlabel='pclass', ylabel='age'>
```



3. Perform descriptive statistics on the dataset.

Ans: descriptive_stats = data.describe()

print(descriptive_stats)

```
In [9]: descriptive_stats = data.describe()
        print(descriptive_stats)
```

	survived	pclass	age	sibsp	parch	fare
count	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

```
In [10]: print(data.isnull().sum())
```

```
survived         0
pclass           0
sex              0
age            177
sibsp            0
parch            0
fare             0
embarked         2
class            0
who              0
adult_male       0
deck           688
embark_town      2
alive            0
alone            0
dtype: int64
```

4. Handle the Missing values.

Ans: data.dropna(inplace=True)

print(data.isnull().sum())

```
In [12]: data.dropna(inplace=True)
         print(data.isnull().sum())
```

```
survived       0
pclass         0
sex            0
age            0
sibsp          0
parch          0
fare           0
embarked       0
class          0
who            0
adult_male     0
deck           0
embark_town    0
alive          0
alone          0
dtype: int64
```

5. Find the outliers and replace the outliers

Ans: sns.boxplot(data, x = "age")

import numpy as np

q1 = np.percentile(data['age'],25)

q3 = np.percentile(data['age'],75)

lower_bound = q1 - 1.5*(q3-q1)

higher_bound = q3 + 1.5*(q3-q1)

median_age = data["age"].median()

median_age

q10 = np.percentile(data["age"], 10)

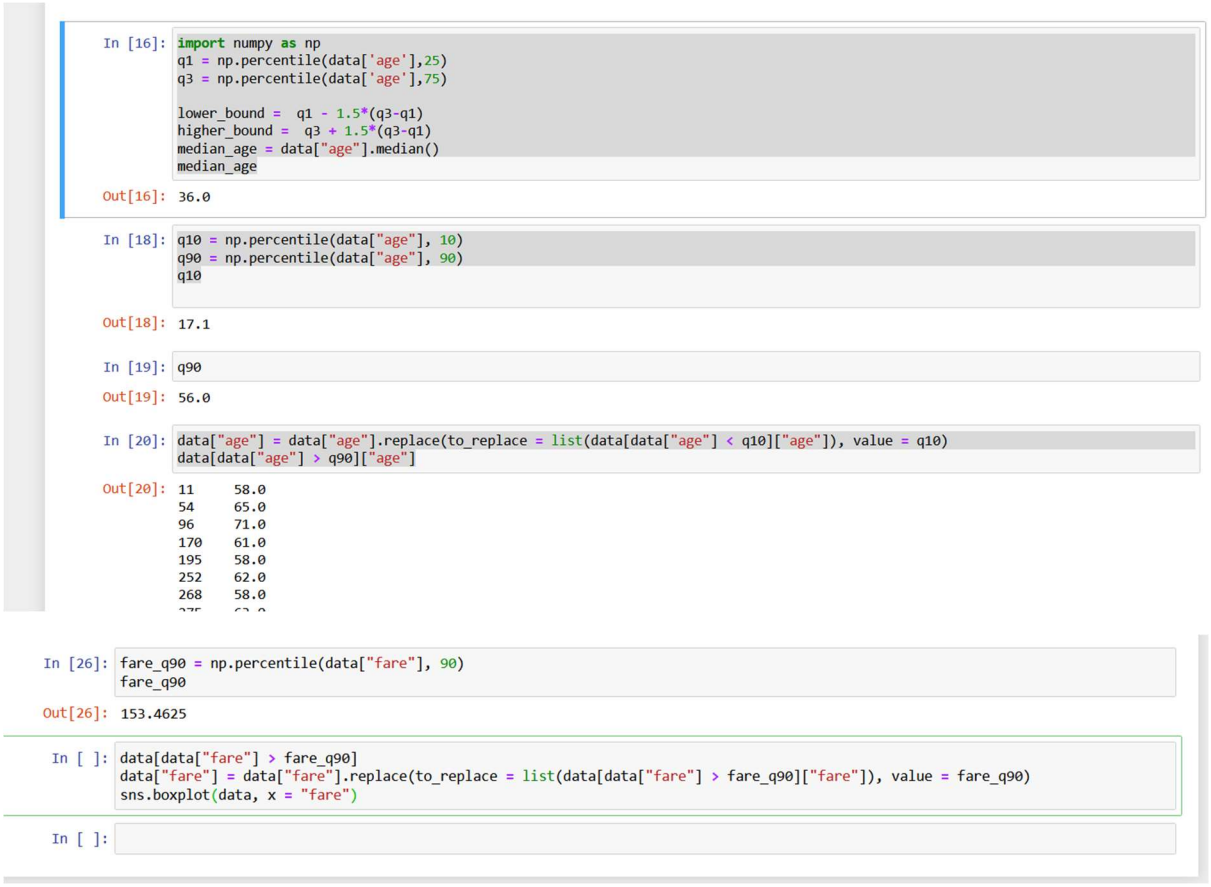
q90 = np.percentile(data["age"], 90)

q10

q90

data["age"] = data["age"].replace(to_replace = list(data[data["age"] < q10]["age"]), value = q10)

```
data[data["age"] > q90]["age"]

data["age"] = data["age"].replace(to_replace = list(data[data["age"] > q90]["age"]), value = q90)

sns.boxplot(data, x = data["age"])

sns.boxplot(data, x = 'fare')

fare_q90 = np.percentile(data["fare"], 90)

fare_q90

data[data["fare"] > fare_q90]

data["fare"] = data["fare"].replace(to_replace = list(data[data["fare"] > fare_q90]["fare"]), value = fare_q90)

sns.boxplot(data, x = "fare")
```



```
In [16]: import numpy as np
         q1 = np.percentile(data['age'],25)
         q3 = np.percentile(data['age'],75)

         lower_bound =  q1 - 1.5*(q3-q1)
         higher_bound =  q3 + 1.5*(q3-q1)
         median_age = data["age"].median()
         median_age

Out[16]: 36.0

In [18]: q10 = np.percentile(data["age"], 10)
         q90 = np.percentile(data["age"], 90)
         q10

Out[18]: 17.1

In [19]: q90

Out[19]: 56.0

In [20]: data["age"] = data["age"].replace(to_replace = list(data[data["age"] < q10]["age"]), value = q10)
         data[data["age"] > q90]["age"]

Out[20]: 11     58.0
         54     65.0
         96     71.0
         170    61.0
         195    58.0
         252    62.0
         268    58.0
```



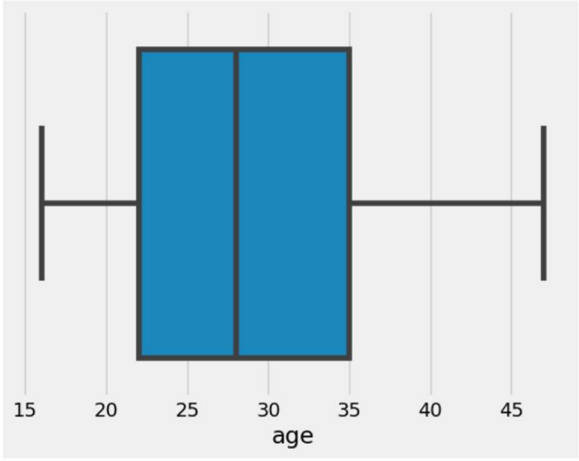
```
In [26]: fare_q90 = np.percentile(data["fare"], 90)
         fare_q90

Out[26]: 153.4625

In [ ]: data[data["fare"] > fare_q90]
        data["fare"] = data["fare"].replace(to_replace = list(data[data["fare"] > fare_q90]["fare"]), value = fare_q90)
        sns.boxplot(data, x = "fare")

In [ ]:
```

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Run Code

In [200]:
```
sns.boxplot(data, x = data["age"])
```

Out[200]: <Axes: xlabel='age'>

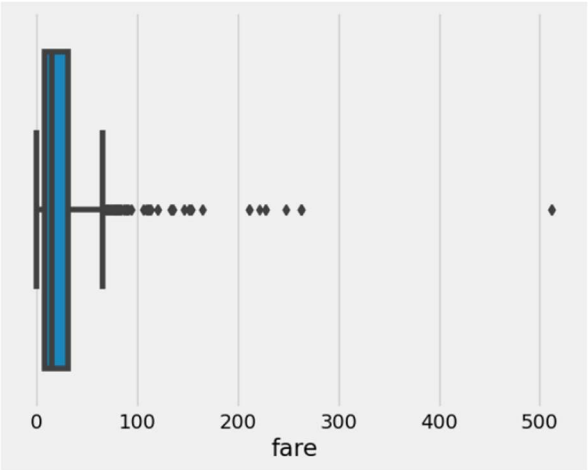


In [205]:
```
sns.boxplot(data, x = "fare")
```

Out[205]: <Axes: xlabel='fare'>

In [205]:
```
sns.boxplot(data, x = "fare")
```

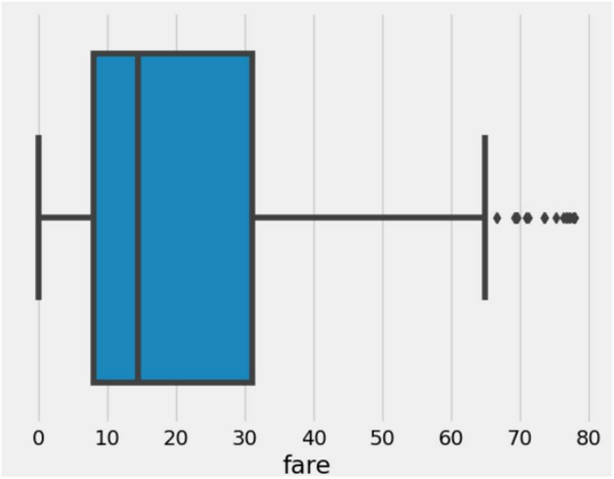
Out[205]: <Axes: xlabel='fare'>



In [206]:
```
fare_q90 = np.percentile(data["fare"], 90)
fare_q90
```

Out[206]: 77.9583

```
In [213]: data["fare"] = data["fare"].replace(to_replace = list(data[data["fare"] > fare_q90]["fare"]), value = fare_q90)

In [214]: sns.boxplot(data, x = "fare")

Out[214]: <Axes: xlabel='fare'>
```

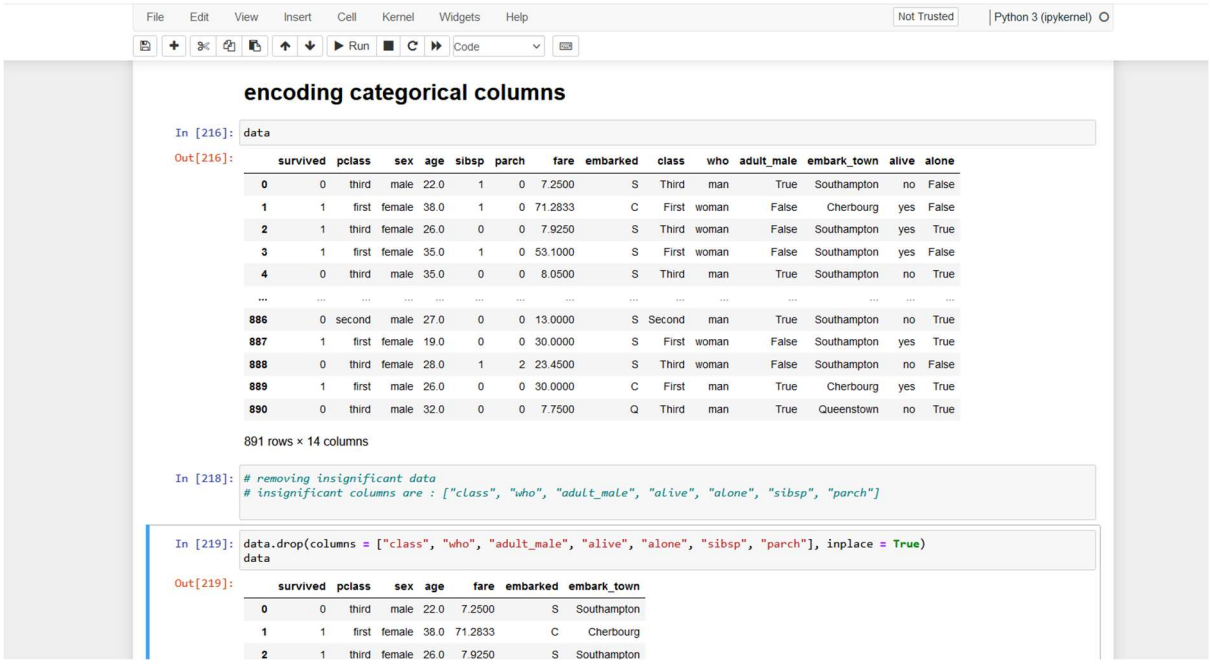


6. Check for Categorical columns and perform encoding.

Ans: data.drop(columns = ["class", "who", "adult_male", "alive", "alone", "sibsp", "parch"], inplace = True)

data = pd.get_dummies(data, columns = ["sex", "pclass","embarked","embark_town"], drop_first = True)

data



File Edit View Insert Cell Kernel Widgets Help — Not Trusted — Python 3 (ipykernel)

Run Code

encoding categorical columns

In [216]: `data`

Out[216]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	embark_town	alive	alone
0	0	third	male	22.0	1	0	7.2500	S	Third	man	True	Southampton	no	False
1	1	first	female	38.0	1	0	71.2833	C	First	woman	False	Cherbourg	yes	False
2	1	third	female	26.0	0	0	7.9250	S	Third	woman	False	Southampton	yes	True
3	1	first	female	35.0	1	0	53.1000	S	First	woman	False	Southampton	yes	False
4	0	third	male	35.0	0	0	8.0500	S	Third	man	True	Southampton	no	True
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
886	0	second	male	27.0	0	0	13.0000	S	Second	man	True	Southampton	no	True
887	1	first	female	19.0	0	0	30.0000	S	First	woman	False	Southampton	yes	True
888	0	third	female	28.0	1	2	23.4500	S	Third	woman	False	Southampton	no	False
889	1	first	male	26.0	0	0	30.0000	C	First	man	True	Cherbourg	yes	True
890	0	third	male	32.0	0	0	7.7500	Q	Third	man	True	Queenstown	no	True

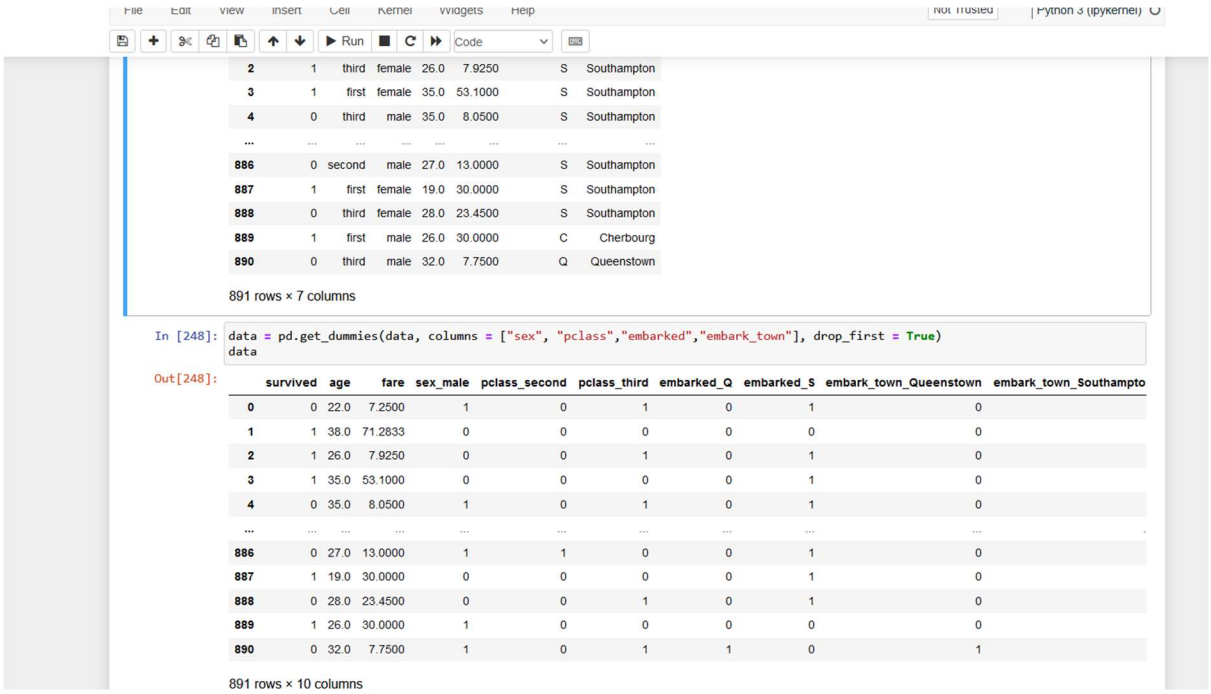
891 rows × 14 columns

In [218]:
```
# removing insignificant data
# insignificant columns are : ["class", "who", "adult_male", "alive", "alone", "sibsp", "parch"]
```

In [219]:
```
data.drop(columns = ["class", "who", "adult_male", "alive", "alone", "sibsp", "parch"], inplace = True)
data
```

Out[219]:

	survived	pclass	sex	age	fare	embarked	embark_town
0	0	third	male	22.0	7.2500	S	Southampton
1	1	first	female	38.0	71.2833	C	Cherbourg
2	1	third	female	26.0	7.9250	S	Southampton



2	1	third	female	26.0	7.9250	S	Southampton
3	1	first	female	35.0	53.1000	S	Southampton
4	0	third	male	35.0	8.0500	S	Southampton
...	...	...	...	...	...	...	...
886	0	second	male	27.0	13.0000	S	Southampton
887	1	first	female	19.0	30.0000	S	Southampton
888	0	third	female	28.0	23.4500	S	Southampton
889	1	first	male	26.0	30.0000	C	Cherbourg
890	0	third	male	32.0	7.7500	Q	Queenstown

891 rows × 7 columns

```
In [248]: data = pd.get_dummies(data, columns = ["sex", "pclass","embarked","embark_town"], drop_first = True)
          data
```

Out[248]:

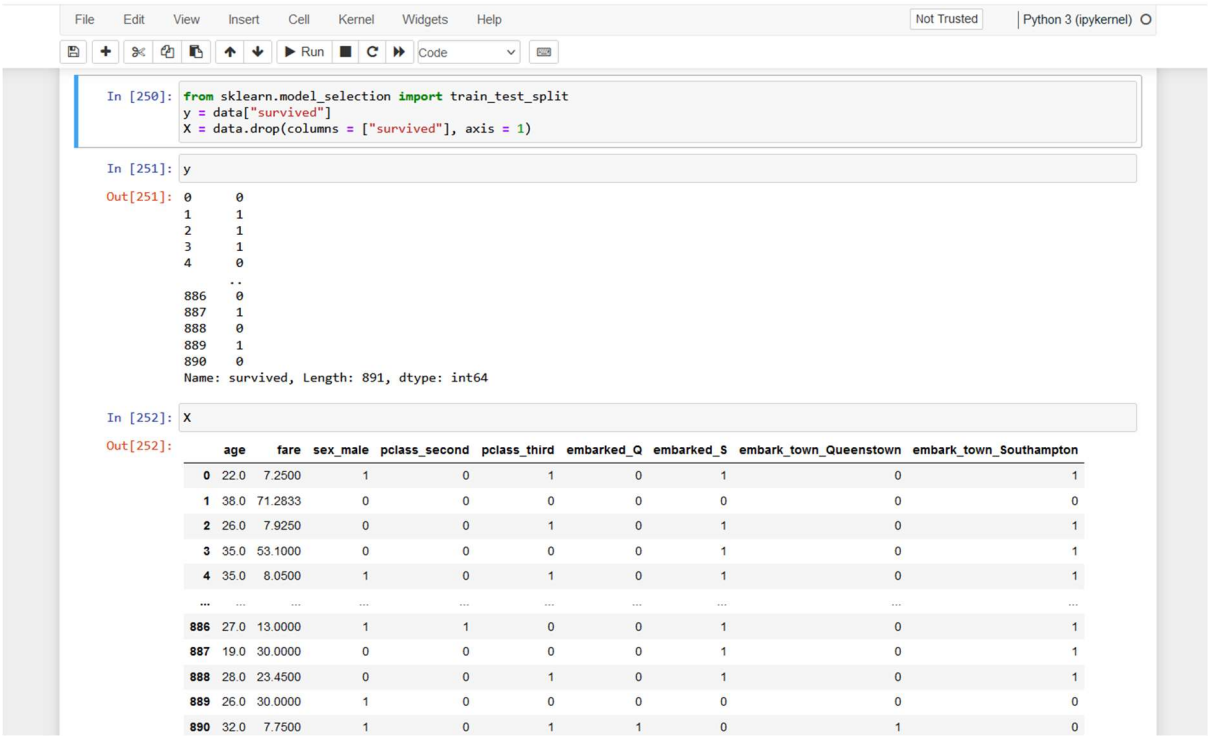
	survived	age	fare	sex_male	pclass_second	pclass_third	embarked_Q	embarked_S	embark_town_Queenstown	embark_town_Southampto
0	0	22.0	7.2500	1	0	1	0	1	0	
1	1	38.0	71.2833	0	0	0	0	0	0	
2	1	26.0	7.9250	0	0	1	0	1	0	
3	1	35.0	53.1000	0	0	0	0	1	0	
4	0	35.0	8.0500	1	0	1	0	1	0	
...	...	...	...	...	...	...	...	...	...	
886	0	27.0	13.0000	1	1	0	0	1	0	
887	1	19.0	30.0000	0	0	0	0	1	0	
888	0	28.0	23.4500	0	0	1	0	1	0	
889	1	26.0	30.0000	1	0	0	0	0	0	
890	0	32.0	7.7500	1	0	1	1	0	1	

891 rows × 10 columns

7. Split the data into dependent and independent variables.

Ans: from sklearn.model_selection import train_test_split

```
y = data["survived"]
X = data.drop(columns = ["survived"], axis = 1)
y
X
```



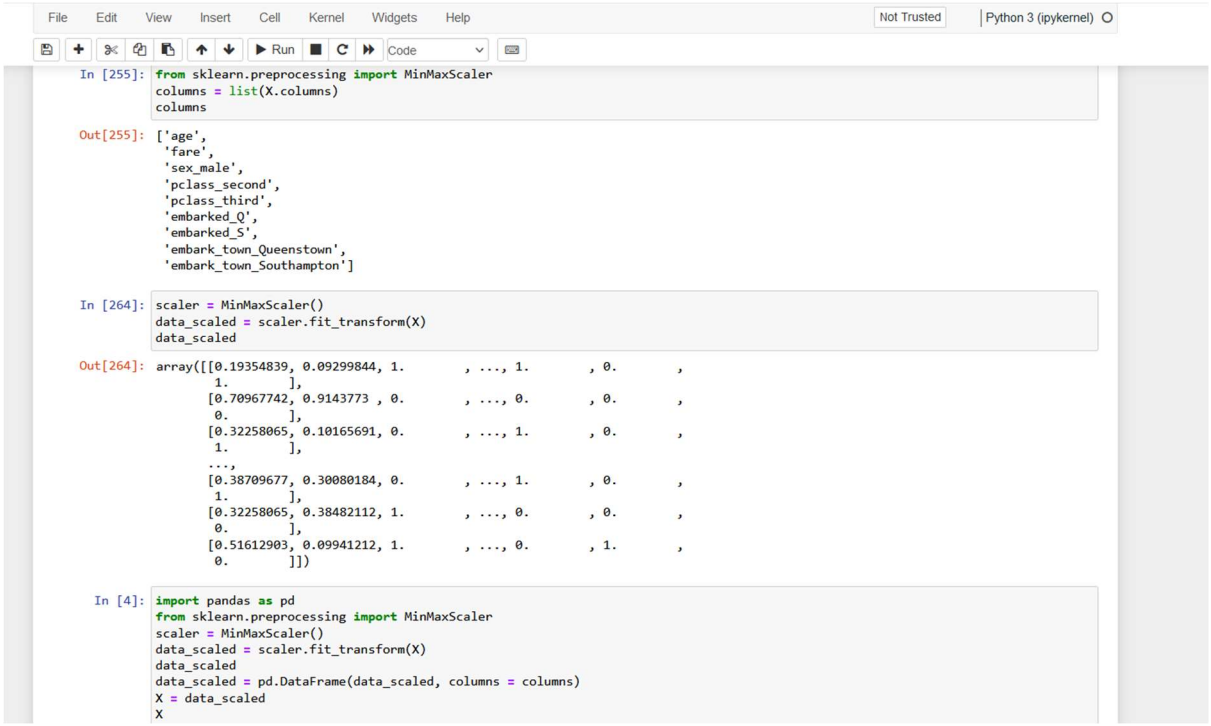
```
In [250]: from sklearn.model_selection import train_test_split
          y = data["survived"]
          X = data.drop(columns = ["survived"], axis = 1)

In [251]: y

Out[251]: 0      0
          1      1
          2      1
          3      1
          4      0
                ..
          886    0
          887    1
          888    0
          889    1
          890    0
          Name: survived, Length: 891, dtype: int64

In [252]: X
```

Out[252]:

	age	fare	sex_male	pclass_second	pclass_third	embarked_Q	embarked_S	embark_town_Queenstown	embark_town_Southampton
0	22.0	7.2500	1	0	1	0	1	0	1
1	38.0	71.2833	0	0	0	0	0	0	0
2	26.0	7.9250	0	0	1	0	1	0	1
3	35.0	53.1000	0	0	0	0	1	0	1
4	35.0	8.0500	1	0	1	0	1	0	1
...	...	...	...	...	...	...	...	...	...
886	27.0	13.0000	1	1	0	0	1	0	1
887	19.0	30.0000	0	0	0	0	1	0	1
888	28.0	23.4500	0	0	1	0	1	0	1
889	26.0	30.0000	1	0	0	0	0	0	0
890	32.0	7.7500	1	0	1	1	0	1	0

8. Scale the independent variables

Ans:
```
from sklearn.preprocessing import MinMaxScaler
columns = list(X.columns)
columns
scaler = MinMaxScaler()
data_scaled = scaler.fit_transform(X)
data_scaled
data_scaled = pd.DataFrame(data_scaled, columns = columns)
X = data_scaled
X
```



```
In [255]: from sklearn.preprocessing import MinMaxScaler
          columns = list(X.columns)
          columns

Out[255]: ['age',
           'fare',
           'sex_male',
           'pclass_second',
           'pclass_third',
           'embarked_Q',
           'embarked_S',
           'embark_town_Queenstown',
           'embark_town_Southampton']

In [264]: scaler = MinMaxScaler()
          data_scaled = scaler.fit_transform(X)
          data_scaled

Out[264]: array([[0.19354839, 0.09299844, 1.        , ..., 1.        , 0.        ,
                  1.        ],
                 [0.70967742, 0.9143773 , 0.        , ..., 0.        , 0.        ,
                  0.        ],
                 [0.32258065, 0.10165691, 0.        , ..., 1.        , 0.        ,
                  1.        ],
                 ...,
                 [0.38709677, 0.30080184, 0.        , ..., 1.        , 0.        ,
                  1.        ],
                 [0.32258065, 0.38482112, 1.        , ..., 0.        , 0.        ,
                  0.        ],
                 [0.51612903, 0.09941212, 1.        , ..., 0.        , 1.        ,
                  0.        ]])

In [4]:  import pandas as pd
         from sklearn.preprocessing import MinMaxScaler
         scaler = MinMaxScaler()
         data_scaled = scaler.fit_transform(X)
         data_scaled
         data_scaled = pd.DataFrame(data_scaled, columns = columns)
         X = data_scaled
         X
```

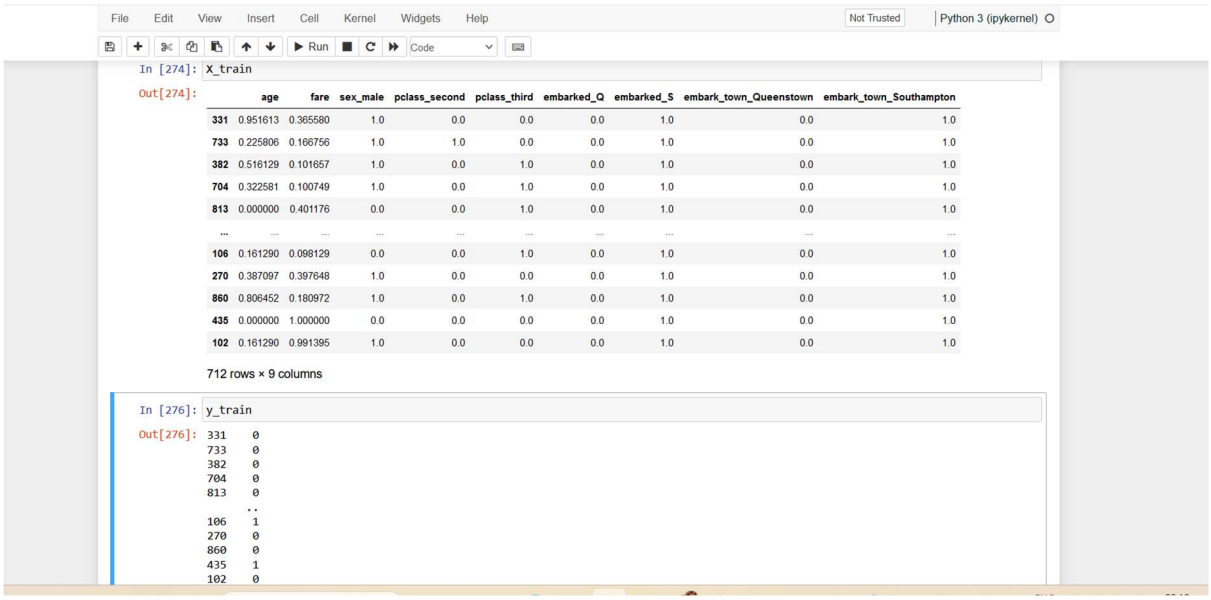
9. Split the data into training and testing

Ans: X_train

y_train

X_test

y_test



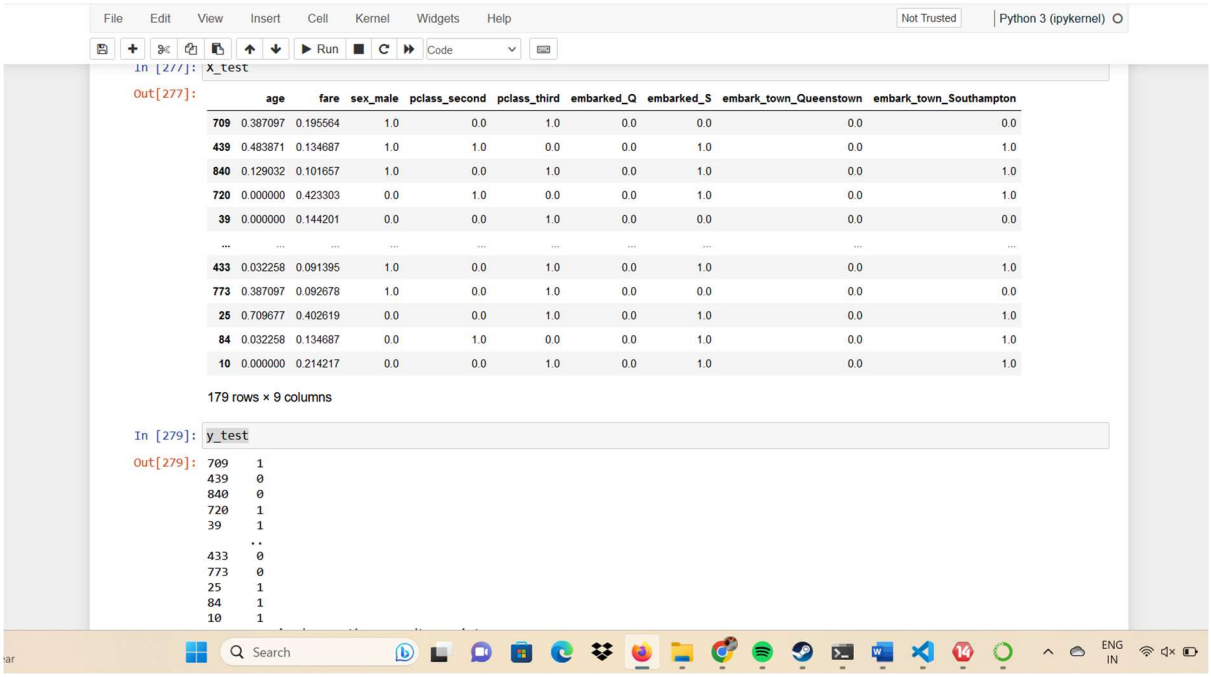
In [274]: X_train

Out[274]:

	age	fare	sex_male	pclass_second	pclass_third	embarked_Q	embarked_S	embark_town_Queenstown	embark_town_Southampton
331	0.951613	0.365580	1.0	0.0	0.0	0.0	1.0	0.0	1.0
733	0.225806	0.166756	1.0	1.0	0.0	0.0	1.0	0.0	1.0
382	0.516129	0.101657	1.0	0.0	1.0	0.0	1.0	0.0	1.0
704	0.322581	0.100749	1.0	0.0	1.0	0.0	1.0	0.0	1.0
813	0.000000	0.401176	0.0	0.0	1.0	0.0	1.0	0.0	1.0
...	...	...	...	...	...	...	...	...	...
106	0.161290	0.098129	0.0	0.0	1.0	0.0	1.0	0.0	1.0
270	0.387097	0.397648	1.0	0.0	0.0	0.0	1.0	0.0	1.0
860	0.806452	0.180972	1.0	0.0	1.0	0.0	1.0	0.0	1.0
435	0.000000	1.000000	0.0	0.0	0.0	0.0	1.0	0.0	1.0
102	0.161290	0.991395	1.0	0.0	0.0	0.0	1.0	0.0	1.0

712 rows × 9 columns

```
In [276]: y_train

Out[276]: 331    0
          733    0
          382    0
          704    0
          813    0
                ..
          106    1
          270    0
          860    0
          435    1
          102    0
```



	age	fare	sex_male	pclass_second	pclass_third	embarked_Q	embarked_S	embark_town_Queenstown	embark_town_Southampton
709	0.387097	0.195564	1.0	0.0	1.0	0.0	0.0	0.0	0.0
439	0.483871	0.134687	1.0	1.0	0.0	0.0	1.0	0.0	1.0
840	0.129032	0.101657	1.0	0.0	1.0	0.0	1.0	0.0	1.0
720	0.000000	0.423303	0.0	1.0	0.0	0.0	1.0	0.0	1.0
39	0.000000	0.144201	0.0	0.0	1.0	0.0	0.0	0.0	0.0
...	...	...	...	...	...	...	...	...	...
433	0.032258	0.091395	1.0	0.0	1.0	0.0	1.0	0.0	1.0
773	0.387097	0.092678	1.0	0.0	1.0	0.0	0.0	0.0	0.0
25	0.709677	0.402619	0.0	0.0	1.0	0.0	1.0	0.0	1.0
84	0.032258	0.134687	0.0	1.0	0.0	0.0	1.0	0.0	1.0
10	0.000000	0.214217	0.0	0.0	1.0	0.0	1.0	0.0	1.0

179 rows × 9 columns