import libraries

In [3]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [9]:
```
data = sns.load_dataset("titanic")
data
```

Out[9]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True
...	...	...	...	...	...	...	...	...	...	...	...
886	0	2	male	27.0	0	0	13.0000	S	Second	man	True
887	1	1	female	19.0	0	0	30.0000	S	First	woman	False
888	0	3	female	NaN	1	2	23.4500	S	Third	woman	False
889	1	1	male	26.0	0	0	30.0000	C	First	man	True
890	0	3	male	32.0	0	0	7.7500	Q	Third	man	True

891 rows × 15 columns

pre_processing

handle categorical data

In [10]:
```
data.dtypes
```

Out[10]:
```
survived         int64
pclass           int64
sex             object
age            float64
sibsp            int64
parch            int64
fare           float64
embarked        object
class         category
who             object
adult_male        bool
deck          category
embark_town     object
alive           object
alone             bool
dtype: object
```

In [12]:
```
data1=data.drop(['adult_male','class','who','embark_town','alone'],axis=1)
data1
```

Out[12]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	deck	alive
0	0	3	male	22.0	1	0	7.2500	S	NaN	no
1	1	1	female	38.0	1	0	71.2833	C	C	yes
2	1	3	female	26.0	0	0	7.9250	S	NaN	yes
3	1	1	female	35.0	1	0	53.1000	S	C	yes
4	0	3	male	35.0	0	0	8.0500	S	NaN	no
...	...	...	...	...	...	...	...	...	...	...
886	0	2	male	27.0	0	0	13.0000	S	NaN	no
887	1	1	female	19.0	0	0	30.0000	S	B	yes
888	0	3	female	NaN	1	2	23.4500	S	NaN	no
889	1	1	male	26.0	0	0	30.0000	C	C	yes
890	0	3	male	32.0	0	0	7.7500	Q	NaN	no

891 rows × 10 columns

In [15]:
```
data1.isna().sum()
```

Out[15]:
```
survived      0
pclass        0
sex           0
age         177
sibsp         0
parch         0
fare          0
embarked      2
deck        688
alive         0
dtype: int64
```

In [24]:
```
data2=data1.drop('deck',axis=1)
data2
```

Out[24]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	alive
0	0	3	male	22.0	1	0	7.2500	S	no
1	1	1	female	38.0	1	0	71.2833	C	yes
2	1	3	female	26.0	0	0	7.9250	S	yes
3	1	1	female	35.0	1	0	53.1000	S	yes
4	0	3	male	35.0	0	0	8.0500	S	no
...	...	...	...	...	...	...	...	...	...
886	0	2	male	27.0	0	0	13.0000	S	no
887	1	1	female	19.0	0	0	30.0000	S	yes
888	0	3	female	NaN	1	2	23.4500	S	no
889	1	1	male	26.0	0	0	30.0000	C	yes
890	0	3	male	32.0	0	0	7.7500	Q	no

891 rows × 9 columns

In [26]:
```
data3=data2.replace({'alive':'no'},0).replace({'alive':'yes'},1)
data3
```

Out[26]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	alive
0	0	3	male	22.0	1	0	7.2500	S	0
1	1	1	female	38.0	1	0	71.2833	C	1
2	1	3	female	26.0	0	0	7.9250	S	1
3	1	1	female	35.0	1	0	53.1000	S	1
4	0	3	male	35.0	0	0	8.0500	S	0
...	...	...	...	...	...	...	...	...	...
886	0	2	male	27.0	0	0	13.0000	S	0
887	1	1	female	19.0	0	0	30.0000	S	1
888	0	3	female	NaN	1	2	23.4500	S	0
889	1	1	male	26.0	0	0	30.0000	C	1
890	0	3	male	32.0	0	0	7.7500	Q	0

891 rows × 9 columns

In [27]:
```
data3['survived'].equals(data3['alive'])
```

Out[27]:
```
True
```

In [28]:
```
data3.embarked.value_counts()
```

Out[28]:
```
S    644
C    168
Q     77
Name: embarked, dtype: int64
```

In [30]:
```
data4=data3.drop('alive',axis=1)
data5=data4.replace({'sex':'male'},1).replace({'sex':'female'},0)
data6=data5.replace({'embarked':'Q'},0).replace({'embarked':'C'},1).replace({'emb
sum_column=data6['sibsp']+data6['parch']
data6['total number of family']=sum_column
data7=data6.drop(['parch','sibsp'],axis=1)
data7
```

Out[30]:

	survived	pclass	sex	age	fare	embarked	total number of family
0	0	3	1	22.0	7.2500	2.0	1
1	1	1	0	38.0	71.2833	1.0	1
2	1	3	0	26.0	7.9250	2.0	0
3	1	1	0	35.0	53.1000	2.0	1
4	0	3	1	35.0	8.0500	2.0	0
...	...	...	...	...	...	...	...
886	0	2	1	27.0	13.0000	2.0	0
887	1	1	0	19.0	30.0000	2.0	0
888	0	3	0	NaN	23.4500	2.0	3
889	1	1	1	26.0	30.0000	1.0	0
890	0	3	1	32.0	7.7500	0.0	0

891 rows × 7 columns

handle missing value

In [31]:
```
data7.isna().sum()
```

Out[31]:
```
survived                    0
pclass                      0
sex                         0
age                       177
fare                        0
embarked                    2
total number of family      0
dtype: int64
```

In [33]:
```
data7.age.mode()
```

Out[33]:
```
0    24.0
dtype: float64
```

In [34]:
```
data7.age.mean()
```

Out[34]:
```
29.69911764705882
```

In [35]:
```
data7.describe()
```

Out[35]:

	survived	pclass	sex	age	fare	embarked	total number of family
count	891.000000	891.000000	891.000000	714.000000	891.000000	889.000000	891.000000
mean	0.383838	2.308642	0.647587	29.699118	32.204208	1.637795	0.904602
std	0.486592	0.836071	0.477990	14.526497	49.693429	0.636157	1.613459
min	0.000000	1.000000	0.000000	0.420000	0.000000	0.000000	0.000000
25%	0.000000	2.000000	0.000000	20.125000	7.910400	1.000000	0.000000
50%	0.000000	3.000000	1.000000	28.000000	14.454200	2.000000	0.000000
75%	1.000000	3.000000	1.000000	38.000000	31.000000	2.000000	1.000000
max	1.000000	3.000000	1.000000	80.000000	512.329200	2.000000	10.000000

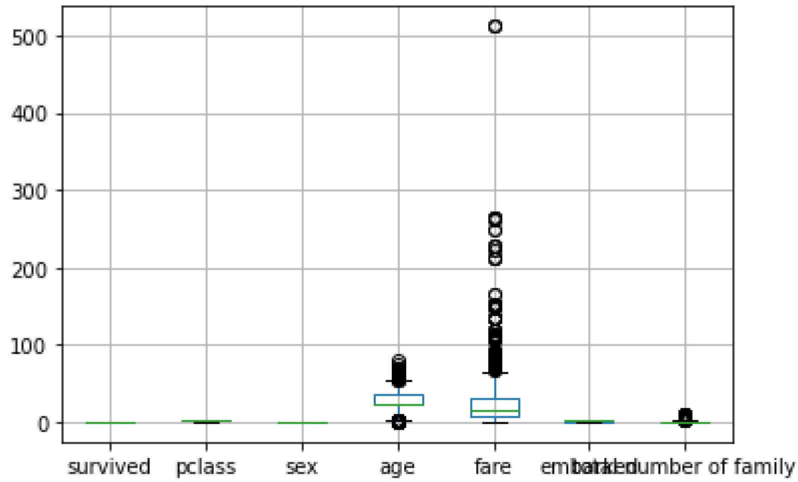
In [41]:
```
data8=data7.fillna({'age':24})
data9=data8.dropna()
data9.isna().sum()
```

Out[41]:
```
survived                  0
pclass                    0
sex                       0
age                       0
fare                      0
embarked                  0
total number of family    0
dtype: int64
```

handle outlier data

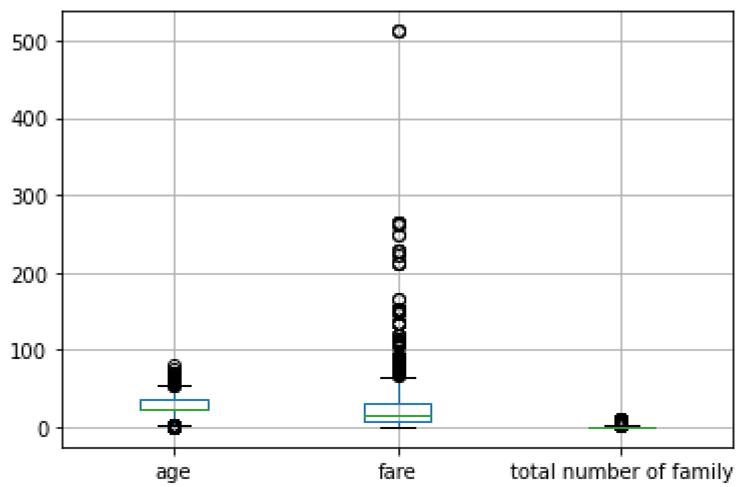
In [42]:
```
data9.boxplot()
```

Out[42]: <matplotlib.axes._subplots.AxesSubplot at 0x1d868603910>



In [43]:
```
data9.iloc[:,[3,4,6]].boxplot()
```

Out[43]: <matplotlib.axes._subplots.AxesSubplot at 0x1d868ddc130>



In [44]:
```
#age
Q1 = data9.iloc[:,3].quantile(0.25)
Q3 = data9.iloc[:,3].quantile(0.75)
LB = Q1-1.5*(Q3-Q1)
UB = Q3+1.5*(Q3-Q1)
print(LB,UB)
```

2.5 54.5

In [46]:
```
data9[data9['age']<2.5].shape
```

Out[46]: (24, 7)

In [47]:
```
data9[data9['age']>54.5].shape
```

Out[47]: (41, 7)

In [48]:
```
#fare
Q1 = data9.iloc[:,4].quantile(0.25)
Q3 = data9.iloc[:,4].quantile(0.75)
LB = Q1-1.5*(Q3-Q1)
UB = Q3+1.5*(Q3-Q1)
print(LB,UB)
```

```
-26.7605 65.6563
```

In [49]:
```
data9[data9['fare']>65.6563].shape
```

Out[49]: (114, 7)

In [50]:
```
#total number of family
Q1 = data9.iloc[:,6].quantile(0.25)
Q3 = data9.iloc[:,6].quantile(0.75)
LB = Q1-1.5*(Q3-Q1)
UB = Q3+1.5*(Q3-Q1)
print(LB,UB)
```

```
-1.5 2.5
```

In [51]:
```
data9[data9['total number of family']>2.5].shape
```

Out[51]: (91, 7)

In [52]:
```
data9[data9['total number of family']<-1.5].shape
```

Out[52]: (0, 7)

handle duplicated value

In [75]:
```
data.duplicated().sum()
```

Out[75]: 107

In [72]:
```
data1=data.drop_duplicates()
data1
```

Out[72]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True
...	...	...	...	...	...	...	...	...	...	...	...
885	0	3	female	39.0	0	5	29.1250	Q	Third	woman	False
887	1	1	female	19.0	0	0	30.0000	S	First	woman	False
888	0	3	female	NaN	1	2	23.4500	S	Third	woman	False
889	1	1	male	26.0	0	0	30.0000	C	First	man	True
890	0	3	male	32.0	0	0	7.7500	Q	Third	man	True

784 rows × 15 columns

In [73]:
```
data2=data1.iloc[:,[0,1,2,3,4,5,6,7,11,13]]
data2
```

Out[73]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	deck	alive
0	0	3	male	22.0	1	0	7.2500	S	NaN	no
1	1	1	female	38.0	1	0	71.2833	C	C	yes
2	1	3	female	26.0	0	0	7.9250	S	NaN	yes
3	1	1	female	35.0	1	0	53.1000	S	C	yes
4	0	3	male	35.0	0	0	8.0500	S	NaN	no
...	...	...	...	...	...	...	...	...	...	...
885	0	3	female	39.0	0	5	29.1250	Q	NaN	no
887	1	1	female	19.0	0	0	30.0000	S	B	yes
888	0	3	female	NaN	1	2	23.4500	S	NaN	no
889	1	1	male	26.0	0	0	30.0000	C	C	yes
890	0	3	male	32.0	0	0	7.7500	Q	NaN	no

784 rows × 10 columns

In [78]:
```
data3=data2.drop('deck',axis=1)
data4=data3.replace({'alive':'no'},0).replace({'alive':'yes'},1)
data5=data4.drop('alive',axis=1)
data6=data5.replace({'sex':'male'},1).replace({'sex':'female'},0)
data7=data6.replace({'embarked':'Q'},0).replace({'embarked':'C'},1).replace({'emb
sum_column=data7['sibsp']+data7['parch']
data7['total number of family']=sum_column
data8=data7.drop(['parch','sibsp'],axis=1)
data9=data8.fillna({'age':24})
data10=data9.dropna()
data10.duplicated().sum()
data10
```

Out[78]:

	survived	pclass	sex	age	fare	embarked	total number of family
0	0	3	1	22.0	7.2500	2.0	1
1	1	1	0	38.0	71.2833	1.0	1
2	1	3	0	26.0	7.9250	2.0	0
3	1	1	0	35.0	53.1000	2.0	1
4	0	3	1	35.0	8.0500	2.0	0
...	...	...	...	...	...	...	...
885	0	3	0	39.0	29.1250	0.0	5
887	1	1	0	19.0	30.0000	2.0	0
888	0	3	0	24.0	23.4500	2.0	3
889	1	1	1	26.0	30.0000	1.0	0
890	0	3	1	32.0	7.7500	0.0	0

782 rows × 7 columns

feature scaling

z_score

In [104]:
```
x=data10.iloc[:,[1,3,4,5,6]].values
binary=data10.iloc[:,[0,2]].reset_index().drop('index',axis=1)
```

In [105]:
```
from sklearn.preprocessing import StandardScaler
ss=StandardScaler()
x_ss=ss.fit_transform(x)
x_ss
```

Out[105]:
```
array([[ 0.88270537, -0.50794209, -0.52443986,  0.56772464,  0.03875807],
       [-1.46118461,  0.64938541,  0.70359063, -1.05850491,  0.03875807],
       [ 0.88270537, -0.21861021, -0.51149471,  0.56772464, -0.62012912],
       ...,
       [ 0.88270537, -0.36327615, -0.21375629,  0.56772464,  1.35653244],
       [-1.46118461, -0.21861021, -0.08814041, -1.05850491, -0.62012912],
       [ 0.88270537,  0.2153876 , -0.51485086, -2.68473445, -0.62012912]])
```

In [106]:
```
df_ss=pd.DataFrame(x_ss,columns=data10.iloc[:,[1,3,4,5,6]].columns)
```

In [107]:
```
data11=pd.concat([df_ss,binary],axis=1)
data11
```

Out[107]:

	pclass	age	fare	embarked	total number of family	survived	sex
0	0.882705	-0.507942	-0.524440	0.567725	0.038758	0	1
1	-1.461185	0.649385	0.703591	-1.058505	0.038758	1	0
2	0.882705	-0.218610	-0.511495	0.567725	-0.620129	1	0
3	-1.461185	0.432387	0.354871	0.567725	0.038758	1	0
4	0.882705	0.432387	-0.509097	0.567725	-0.620129	0	1
...	...	...	...	...	...	...	...
777	0.882705	0.721718	-0.104921	-2.684734	2.674307	0	0
778	-1.461185	-0.724941	-0.088140	0.567725	-0.620129	1	0
779	0.882705	-0.363276	-0.213756	0.567725	1.356532	0	0
780	-1.461185	-0.218610	-0.088140	-1.058505	-0.620129	1	1
781	0.882705	0.215388	-0.514851	-2.684734	-0.620129	0	1

782 rows × 7 columns

min-max

In [135]:
```
from sklearn.preprocessing import MinMaxScaler
x1=data10.iloc[:,[1,3,4,5,6]].values
y1=data10.iloc[:,0].reset_index().drop('index',axis=1)
binary=data10.iloc[:,2].reset_index().drop('index',axis=1)
mm = MinMaxScaler()
x_mm = mm.fit_transform(x1)
x_mm
```

Out[135]:
```
array([[1.        , 0.27117366, 0.01415106, 1.        , 0.1       ],
       [0.        , 0.4722292 , 0.13913574, 0.5       , 0.1       ],
       [1.        , 0.32143755, 0.01546857, 1.        , 0.        ],
       ...,
       [1.        , 0.2963056 , 0.04577135, 1.        , 0.3       ],
       [0.        , 0.32143755, 0.0585561 , 0.5       , 0.        ],
       [1.        , 0.39683338, 0.01512699, 0.        , 0.        ]])
```

In [136]:
```
df_mm=pd.DataFrame(x_mm,columns=data10.iloc[:,[1,3,4,5,6]].columns)
df_mm
```

Out[136]:

	pclass	age	fare	embarked	total number of family
0	1.0	0.271174	0.014151	1.0	0.1
1	0.0	0.472229	0.139136	0.5	0.1
2	1.0	0.321438	0.015469	1.0	0.0
3	0.0	0.434531	0.103644	1.0	0.1
4	1.0	0.434531	0.015713	1.0	0.0
...	...	...	...	...	...
777	1.0	0.484795	0.056848	0.0	0.5
778	0.0	0.233476	0.058556	1.0	0.0
779	1.0	0.296306	0.045771	1.0	0.3
780	0.0	0.321438	0.058556	0.5	0.0
781	1.0	0.396833	0.015127	0.0	0.0

782 rows × 5 columns

In [137]:
```
data12=pd.concat([df_mm,binary,y1],axis=1)
data12
```

Out[137]:

	pclass	age	fare	embarked	total number of family	sex	survived
0	1.0	0.271174	0.014151	1.0	0.1	1	0
1	0.0	0.472229	0.139136	0.5	0.1	0	1
2	1.0	0.321438	0.015469	1.0	0.0	0	1
3	0.0	0.434531	0.103644	1.0	0.1	0	1
4	1.0	0.434531	0.015713	1.0	0.0	1	0
...	...	...	...	...	...	...	...
777	1.0	0.484795	0.056848	0.0	0.5	0	0
778	0.0	0.233476	0.058556	1.0	0.0	0	1
779	1.0	0.296306	0.045771	1.0	0.3	0	0
780	0.0	0.321438	0.058556	0.5	0.0	1	1
781	1.0	0.396833	0.015127	0.0	0.0	1	0

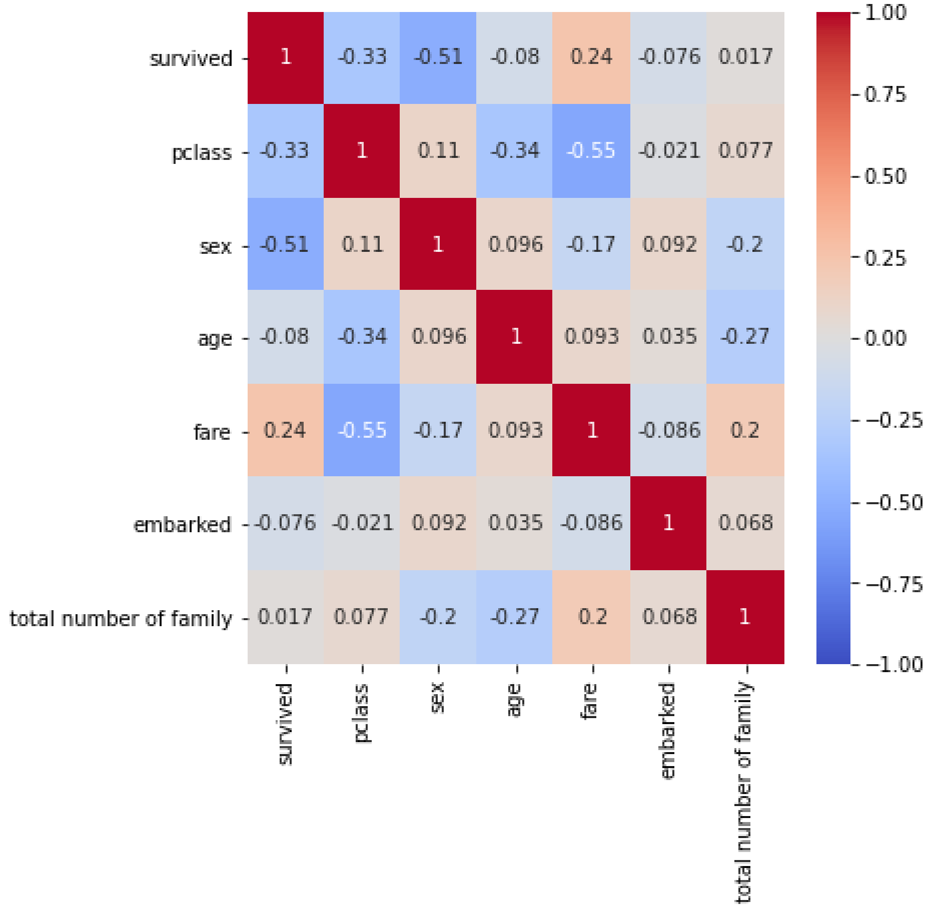
782 rows × 7 columns

feature selection

brute_force

In [138]:
```python
fig=plt.figure(figsize=(6,6))
sns.heatmap(data10.corr(),vmin=-1,vmax=1,annot=True,cmap='coolwarm')
```

Out[138]: <matplotlib.axes._subplots.AxesSubplot at 0x1d86c0e51f0>



filter method

In [139]:
```
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2
from sklearn.feature_selection import f_classif
```

In [145]:
```
x2=data12.iloc[:,0:6]
y2=data12.iloc[:,-1]
bestfeatures = SelectKBest(score_func = chi2, k = 'all')
fit = bestfeatures.fit(x1,binary)
dfscores = pd.DataFrame(fit.scores_)
dfscores
```

Out[145]:

	0
0	3.125439
1	47.757101
2	1716.284499
3	1.519728
4	74.408395

In [144]:
```
x2=data12.iloc[:,0:6]
y2=data12.iloc[:,-1]
bestfeatures = SelectKBest(score_func = f_classif, k = 'all')
fit = bestfeatures.fit(x1,binary)
dfscores = pd.DataFrame(fit.scores_)
dfscores
```

```
C:\Users\sun\anaconda3\lib\site-packages\sklearn\utils\validation.py:73: DataCo
nversionWarning: A column-vector y was passed when a 1d array was expected. Ple
ase change the shape of y to (n_samples, ), for example using ravel().
  return f(**kwargs)
```

Out[144]:

	0
0	9.740201
1	7.300893
2	22.408319
3	6.674788
4	31.551836

In [148]:
```
data
```

Out[148]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True
...	...	...	...	...	...	...	...	...	...	...	...
886	0	2	male	27.0	0	0	13.0000	S	Second	man	True
887	1	1	female	19.0	0	0	30.0000	S	First	woman	False
888	0	3	female	NaN	1	2	23.4500	S	Third	woman	False
889	1	1	male	26.0	0	0	30.0000	C	First	man	True
890	0	3	male	32.0	0	0	7.7500	Q	Third	man	True

891 rows × 15 columns

In [152]:
```
sns.relplot(x= 'embarked', y = "fare", hue = "alive", data = data)
```

Out[152]: <seaborn.axisgrid.FacetGrid at 0x1d86c095430>