ds_lab1

August 7, 2025

[76]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

[2]:
```
df=sns.load_dataset('titanic')
```

[3]:
```
df.head()
```

[3]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	\
0	0	3	male	22.0	1	0	7.2500	S	Third	
1	1	1	female	38.0	1	0	71.2833	C	First	
2	1	3	female	26.0	0	0	7.9250	S	Third	
3	1	1	female	35.0	1	0	53.1000	S	First	
4	0	3	male	35.0	0	0	8.0500	S	Third	

	who	adult_male	deck	embark_town	alive	alone
0	man	True	NaN	Southampton	no	False
1	woman	False	C	Cherbourg	yes	False
2	woman	False	NaN	Southampton	yes	True
3	woman	False	C	Southampton	yes	False
4	man	True	NaN	Southampton	no	True

[4]:
```
df.shape
```

[4]:
```
(891, 15)
```

[5]:
```
df.nunique()
```

[5]:
```
survived      2
pclass        3
sex           2
age          88
sibsp         7
parch         7
fare        248
embarked      3
class         3
```

```
who              3
adult_male       2
deck             7
embark_town      3
alive            2
alone            2
dtype: int64
```

[6]:
```
df.isnull().sum()
```

[6]:
```
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
```

- **There are 688 null values of deck out of 891.**
- **'deck' column does not effect dataset in any way.**

0.1 1.Frequency table

[10]:
```
freq_table=df['pclass'].value_counts().sort_index()
print(freq_table)
```

```
pclass
1    216
2    184
3    491
Name: count, dtype: int64
```

[12]:
```
rel_freq_table=df['pclass'].value_counts(normalize=True).sort_index()*100
print(rel_freq_table)
```

```
pclass
1    24.242424
2    20.650954
3    55.106622
```

```
Name: proportion, dtype: float64
```

24.24More than half population are travelling in class 3.

[14]:
```
cum_freq_table=df['pclass'].value_counts().sort_index().cumsum()
print(cum_freq_table)
```

```
pclass
1    216
2    400
3    891
Name: count, dtype: int64
```

[16]:
```
freq = pd.DataFrame({
    'Absolute Frequency': freq_table,
    'Relative Frequency': rel_freq_table.round(2),
    'Cumulative Frequency': cum_freq_table
})
freq
```

[16]:

pclass	Absolute Frequency	Relative Frequency	Cumulative Count
1	216	24.24	216
2	184	20.65	400
3	491	55.11	891

0.2 2.Joint, Marginal, and Conditional Probabilities

[39]:
```
two_way_table = pd.crosstab(df['sex'],␣
 ↪df['survived'],margins=True,margins_name='Total')
total=two_way_table.loc['Total','Total']
print(two_way_table)
```

```
survived    0    1  Total
sex
female     81  233    314
male      468  109    577
Total     549  342    891
```

- **81 females and 468 males have not survived.**

[89]:
```
print('total passengers: ',two_way_table.loc['Total','Total'])
print('total female who survived: ',two_way_table.loc['female',1])
joint_prob=(two_way_table.loc['female',1]/two_way_table.loc['Total','Total'])
joint_prob.round(2)
```

```
total passengers:  891
total female who survived:  233
```

[89]:
```
np.float64(0.26)
```

[41]:
```
p_female = two_way_table.loc['female', 'Total'] / total
p_survived = two_way_table.loc['Total', 1] / total
print(p_female.round(2))
print(p_survived.round(2))
```

```
0.35
0.38
```

[44]:
```
#another way without total

two_way_table1 = pd.crosstab(df['sex'], df['survived'])
two_way_table1
```

[44]:

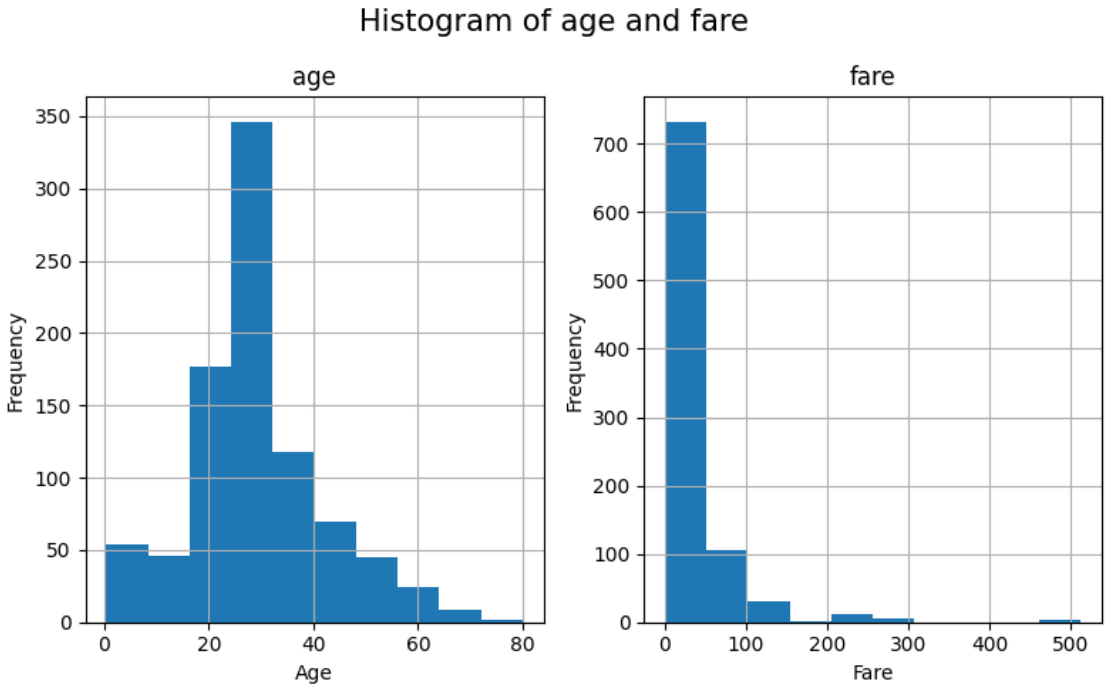
survived	0	1
sex		
female	81	233
male	468	109

[47]:
```
#for rows,use loc
#for cols,use index


p_survived_given_female=two_way_table1.loc['female',1]/two_way_table1.
 ↪loc['female'].sum()
p_female_given_survived=two_way_table1.loc['female',1]/two_way_table1[1].sum()

print(p_survived_given_female)
print(p_female_given_survived)
```

```
0.7420382165605095
0.6812865497076024
```

0.3 3.Correlation Analysis

[96]:
```
axes=df[['age','fare']].hist(figsize=(8,5))
plt.suptitle('Histogram of age and fare',fontsize=15)

axes[0][0].set_xlabel('Age')
axes[0][0].set_ylabel('Frequency')

axes[0][1].set_xlabel('Fare')
axes[0][1].set_ylabel('Frequency')

plt.tight_layout()
plt.show()
```



- **Most passengers are between age 20 and 40.**

0.3.1 Handle missing values

[61]:
```
mean_age=df['age'].mean()
mean_fare=df['fare'].mean()
mean_age,mean_fare
```

[61]:
```
(np.float64(29.69911764705882), np.float64(32.204207968574636))
```

[97]:
```
df['age'].fillna(mean_age, inplace=True)
df['fare'].fillna(mean_fare, inplace=True)
df[['age','fare']].isnull().sum()
```

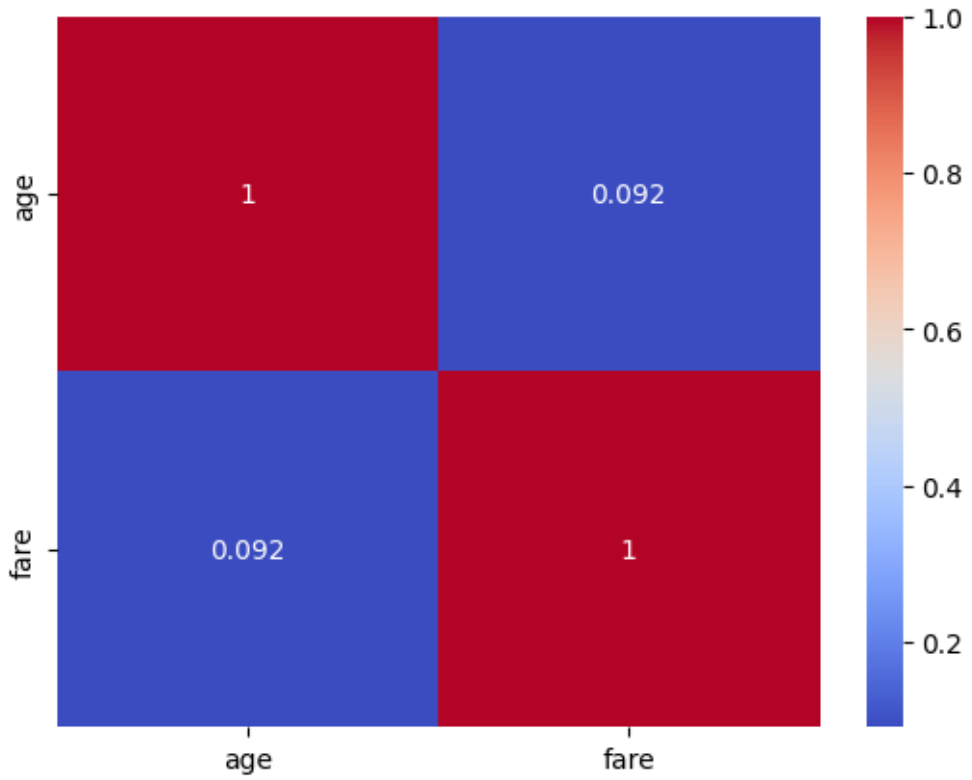
[97]:
```
age     0
fare    0
dtype: int64
```

[67]:
```
correlation=df[['age','fare']].corr(method='pearson')
correlation
```

[67]:
```
            age      fare
age    1.000000  0.091566
fare   0.091566  1.000000
```

[73]:
```
sns.heatmap(correlation,annot=True,cmap='coolwarm')
```

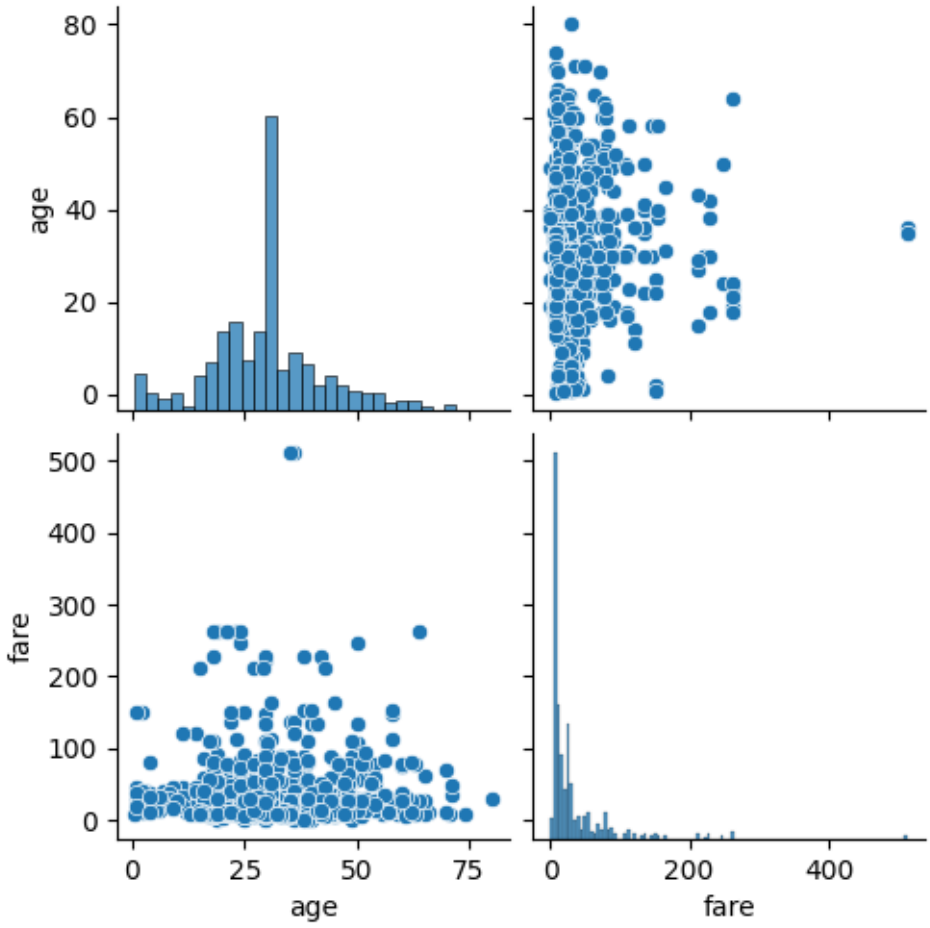
[73]: <Axes: >



- **the correlation between age and fare is approx 0.1.**
- **It means it is poorly related.**
- **Difference in age is not affecting fare.**

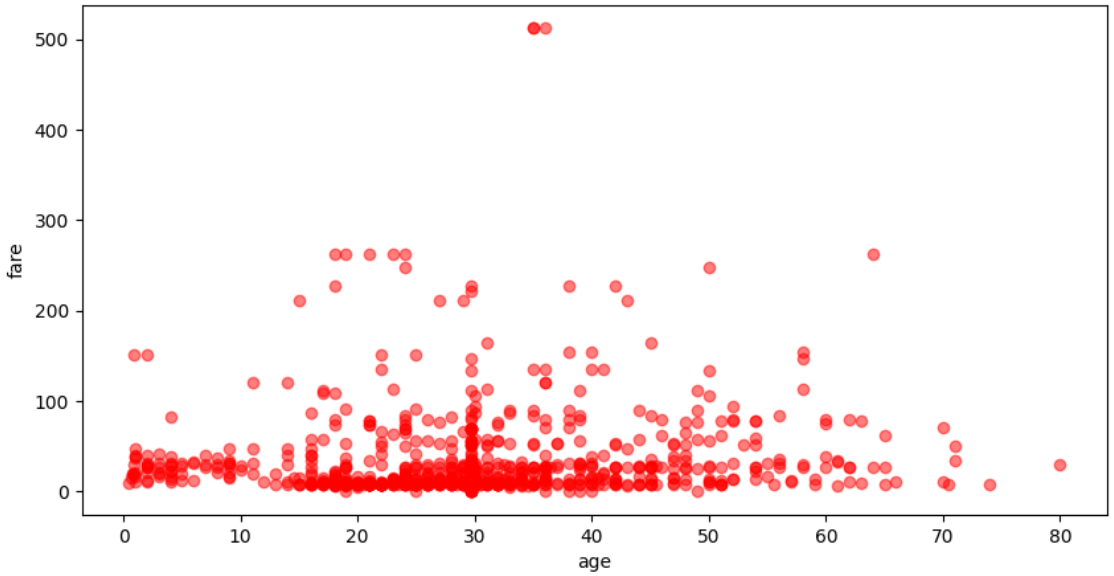
[75]:
```
sns.pairplot(df[['age','fare']])
```

[75]: <seaborn.axisgrid.PairGrid at 0x1ab8a0623c0>



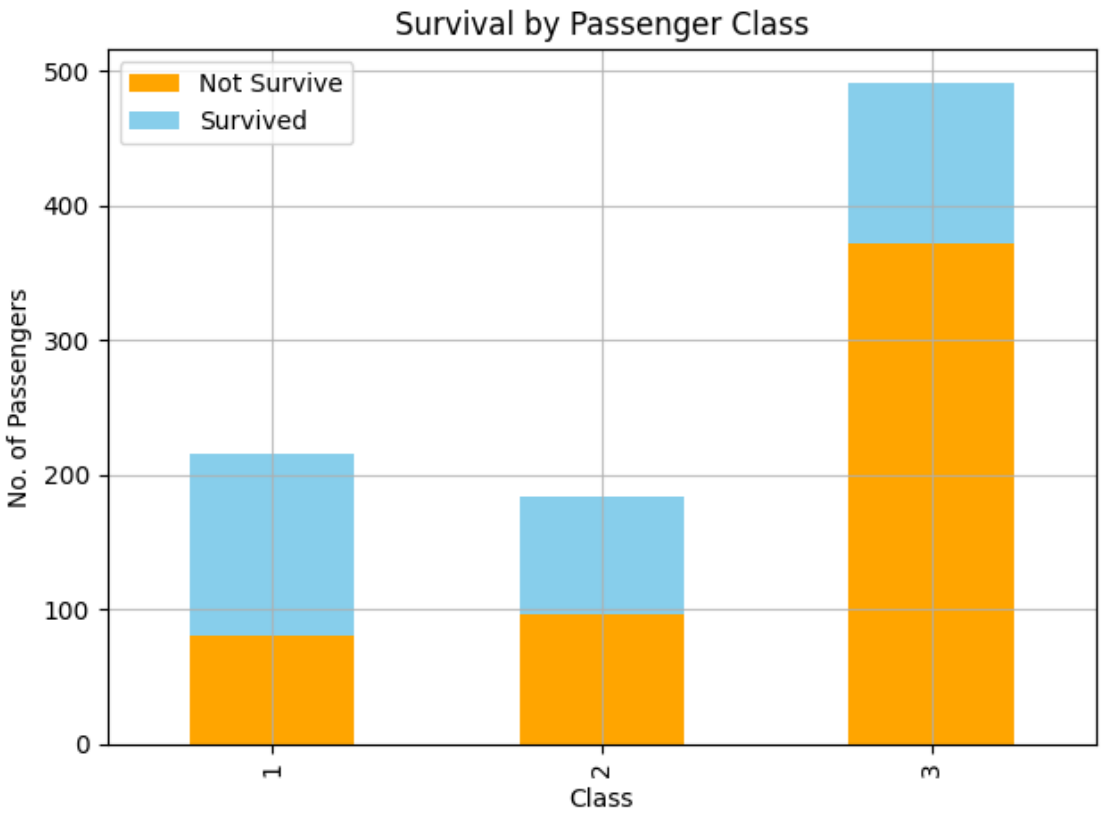
THere is no definite pattern or relation between age and fare. every age group is paying every range of fare.

[105]:
```
plt.figure(figsize=(10,5))
plt.scatter(df['age'],df['fare'],alpha=0.5,color=('red'))
plt.xlabel('age')
plt.ylabel('fare')

plt.show()
```



0.4 Bonus task

[88]:
```
survival_by_class = df.groupby(['pclass', 'survived']).size().unstack()
survival_by_class.plot(kind='bar', stacked=True, color=['orange', 'skyblue'])

# Labels and title
plt.title('Survival by Passenger Class')
plt.xlabel('Class')
plt.ylabel('No. of Passengers')
plt.legend(['Not Survive', 'Survived'])
plt.grid(axis='both',alpha=0.8)
plt.tight_layout()
plt.show()
```



Most of the class3 passengers did not survived. More than half passengers of class 1 survived. Highest survival rate is in class 1.

[ ]: