Why do EDA

- Model building
- Analysis and reporting
- Validate assumptions
- Handling missing values
- feature engineering
- detecting outliers

Remember it is an iterative process

Column Types

- **Numerical** - Age,Fare,PassengerId
- **Categorical** - Survived, Pclass, Sex, SibSp, Parch,Embarked
- **Mixed** - Name, Ticket, Cabin

Univariate Analysis

Univariate analysis focuses on analyzing each feature in the dataset independently.

- **Distribution analysis**: The distribution of each feature is examined to identify its shape, central tendency, and dispersion.
- **Identifying potential issues**: Univariate analysis helps in identifying potential problems with the data such as outliers, skewness, and missing values

Dispersion is a statistical term used to describe the spread or variability of a set of data. It measures how far the values in a data set are spread out from the central tendency (mean, median, or mode) of the data.

There are several measures of dispersion, including:

- **Range**: The difference between the largest and smallest values in a data set.
- **Variance**: The average of the squared deviations of each value from the mean of the data set.

- **Standard Deviation**: The square root of the variance. It provides a measure of the spread of the data that is in the same units as the original data.
- **Interquartile range (IQR)**: The range between the first quartile (25th percentile) and the third quartile (75th percentile) of the data.

Dispersion helps to describe the spread of the data, which can help to identify the presence of outliers and skewness in the data.

Steps of doing Univariate Analysis on Numerical columns

- **Descriptive Statistics**: Compute basic summary statistics for the column, such as mean, median, mode, standard deviation, range, and quartiles. These statistics give a general understanding of the distribution of the data and can help identify skewness or outliers.
- **Visualizations**: Create visualizations to explore the distribution of the data. Some common visualizations for numerical data include histograms, box plots, and density plots. These visualizations provide a visual representation of the distribution of the data and can help identify skewness an outliers.
- **Identifying Outliers**: Identify and examine any outliers in the data. Outliers can be identified using visualizations. It is important to determine whether the outliers are due to measurement errors, data entry errors, or legitimate differences in the data, and to decide whether to include or exclude them from the analysis.
- **Skewness**: Check for skewness in the data and consider transforming the data or using robust statistical methods that are less sensitive to skewness, if necessary.
- **Conclusion**: Summarize the findings of the EDA and make decisions about how to proceed with further analysis.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
df = pd.read_csv('/content/train.csv')
df.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500
1	2	1	1	Cumings, Mrs. John Bradley	female	38.0	1	0	PC 17599	71.2833
2	3	1	3	Miss.	female	26.0	0	0	3101282	7.9250

age column

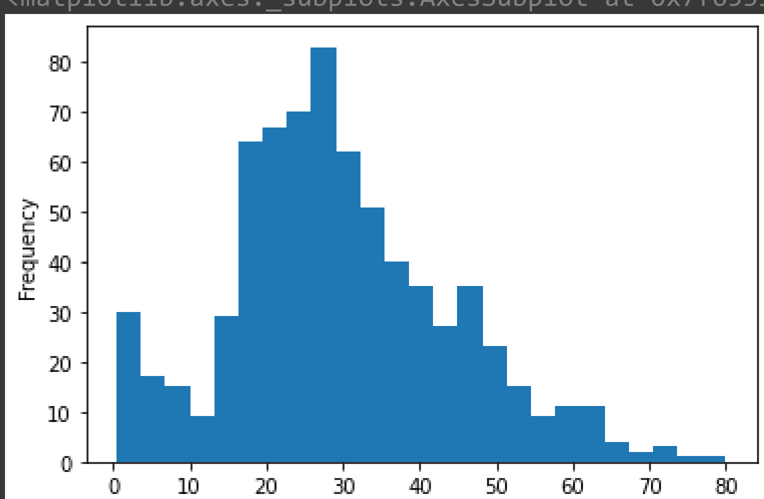
conclusion

```
# Descriptive Statistics
df['Age'].describe()
```

```
count    714.000000
mean      29.699118
std       14.526497
min        0.420000
25%       20.125000
50%       28.000000
75%       38.000000
max       80.000000
Name: Age, dtype: float64
```

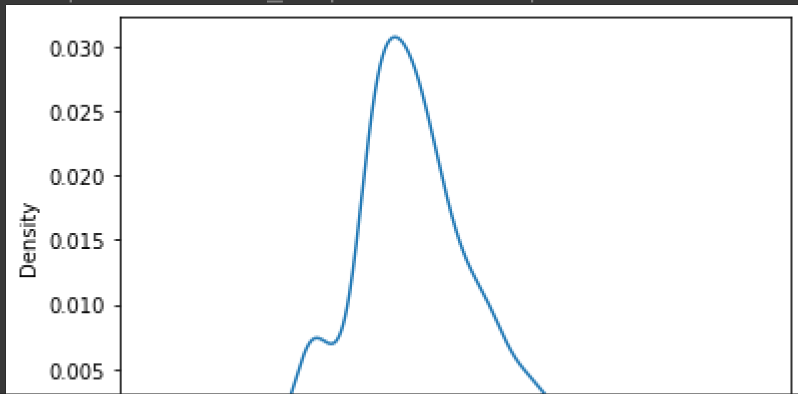
```
# Visualizations
df['Age'].plot(kind='hist', bins=25)
```

<matplotlib.axes._subplots.AxesSubplot at 0x7f6533d5ceb0>




```
df['Age'].plot(kind='kde')
```

<matplotlib.axes._subplots.AxesSubplot at 0x7f6533cc9160>

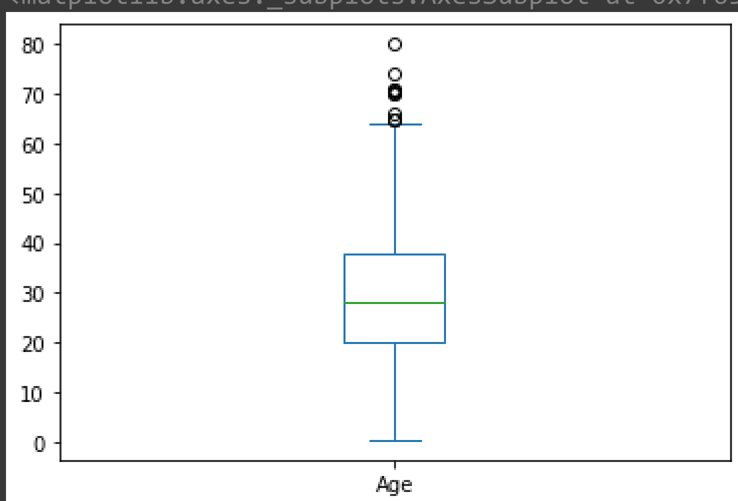



```
# skewness
df['Age'].skew()  #
```

0.38910778230082704

```
# Identifying Outliers
df['Age'].plot(kind='box')
```

<matplotlib.axes._subplots.AxesSubplot at 0x7f6531bfdfa0>




```
df[df['Age'] > 65]
```

PassengerId Survived Pclass Name Sex Age SibSp Parch Ticket Fare

```
# missing values
df['Age'].isnull().sum()
```

```
177
```

17754

```
df['Age'].isnull().sum()/len(df['Age'])
```

```
0.19865319865319866
```

Artagaveytia, PC

Age

conclusions

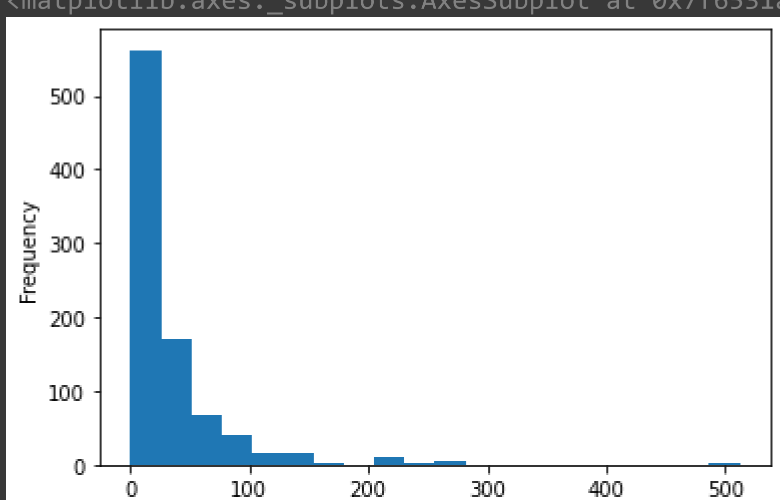
- Age is normally(almost) distributed
- 20% of the values are missing
- There are some outliers

```
df['Fare'].describe()
```

```
count    891.000000
mean      32.204208
std       49.693429
min        0.000000
25%        7.910400
50%       14.454200
75%       31.000000
max      512.329200
Name: Fare, dtype: float64
```

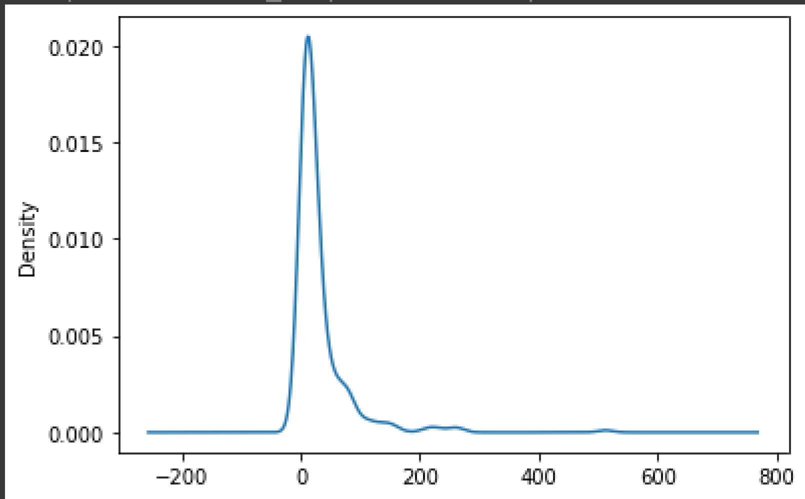
```
df['Fare'].plot(kind='hist', bins=20)  # right skewed
```

<matplotlib.axes._subplots.AxesSubplot at 0x7f6531a85a60>



```
# skew checking
df['Fare'].plot(kind='kde')
```

<matplotlib.axes._subplots.AxesSubplot at 0x7f65319ffbb0>

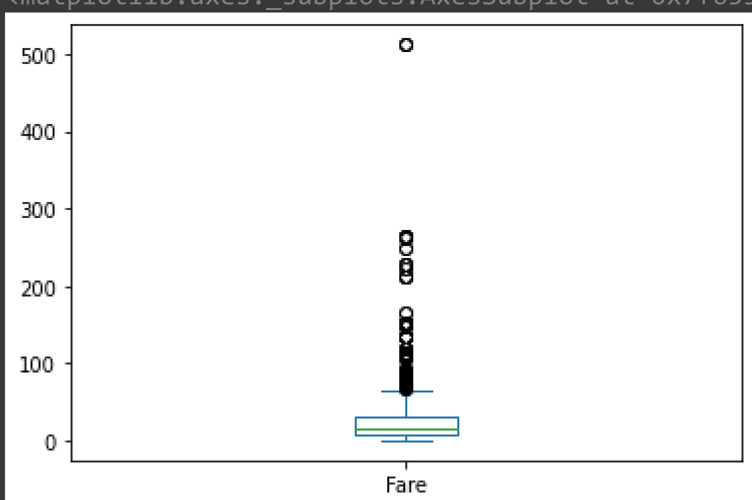



```
# skew checking
df['Fare'].skew()  # highly positively skewed
```

4.787316519674893

```
# outlier
df['Fare'].plot(kind='box') # got a lot of outliers
```

<matplotlib.axes._subplots.AxesSubplot at 0x7f65319ced90>




```
df[df['Fare'] > 250]
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Far
27	28	0	1	Fortune, Mr. Charles Alexander	male	19.0	3	2	19950	263.000
88	89	1	1	Fortune, Miss. Mabel Helen	female	23.0	3	2	19950	263.000
258	259	1	1	Ward, Miss. Anna	female	35.0	0	0	PC 17755	512.329

```
df['Fare'].isnull().sum()
```

```
0
```

Borie

Fair column

conclusions

- The data is highly (positively) skewed.
- Fare column actually contains the group fare and not the individual fare(can be a issue)
- we need to create a new column called `individual fare`
- no missing value found.

Steps of doing Univariate Analysis on Categorical columns

Descriptive Statistics: Compute the frequency distribution of the categories in the column. This will give a general understanding of the distribution of the categories and their relative frequencies.

Visualizations: Create visualizations to explore the distribution of the categories. Some common visualizations for categorical data include count plots and pie charts. These visualizations provide a visual representation of the distribution of the categories and can help identify any patterns or anomalies in the data.

Missing Values: Check for missing values in the data and decide how to handle them. Missing values can be imputed or excluded from the analysis, depending on the research question and the data set.

Conclusion: Summarize the findings of the EDA and make decisions about how to proceed with further analysis.

Survived

conclusions

- Parch and SibSp cols can be merged to form a new col call family_size
- Create a new col called is_alone

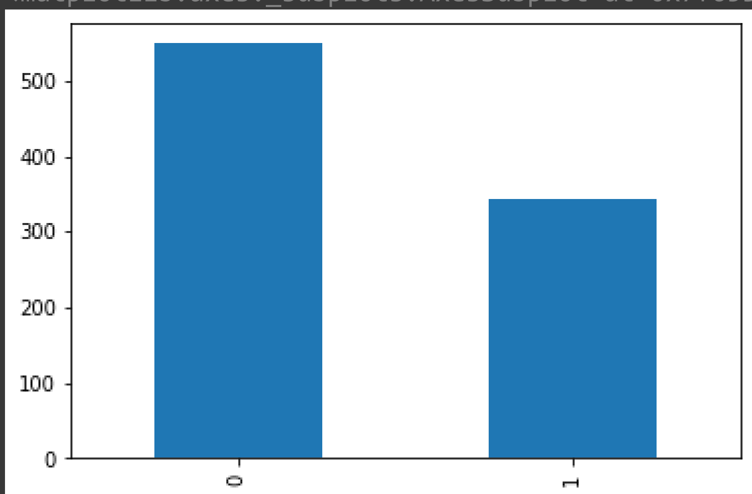
```
# survived column
```

```
df['Survived'].value_counts()
```

```
0    549
1    342
Name: Survived, dtype: int64
```

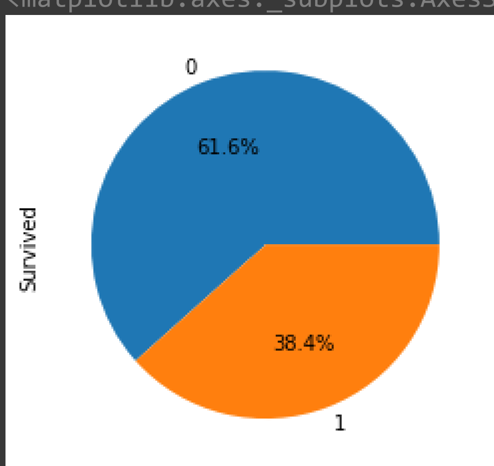
```
df['Survived'].value_counts().plot(kind='bar')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f6531936460>
```




```
df['Survived'].value_counts().plot(kind='pie', autopct='%0.1f%%')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f65318abfd0>
```



```
# missing values
df['Survived'].isnull().sum()
```

```
0
```

```
df['Survived'].describe()
```

```
count    891.000000
mean       0.383838
std        0.486592
min        0.000000
25%        0.000000
50%        0.000000
75%        1.000000
max        1.000000
Name: Survived, dtype: float64
```

Pclass column

conclusion

- surprisingly class 1 is more travelling than class 2. Why?

```
df['Pclass'].describe()
```

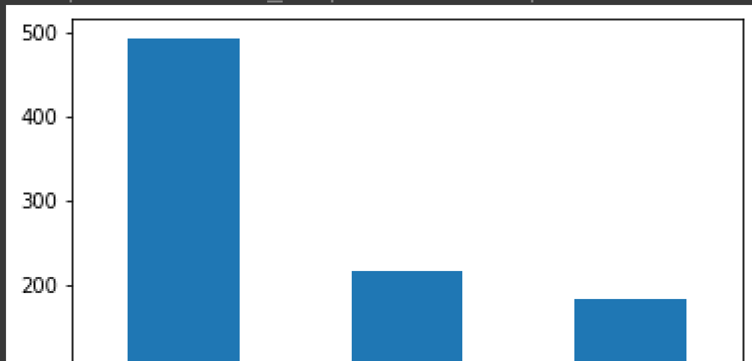
```
count    891.000000
mean       2.308642
std        0.836071
min        1.000000
25%        2.000000
50%        3.000000
75%        3.000000
max        3.000000
Name: Pclass, dtype: float64
```

```
df['Pclass'].value_counts()
```

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

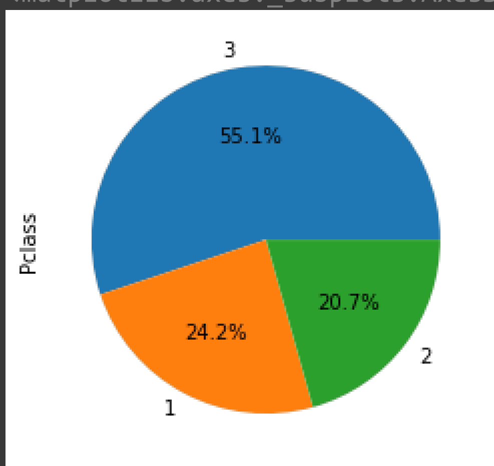
```
df['Pclass'].value_counts().plot(kind='bar')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f65317253d0>
```




```
df['Pclass'].value_counts().plot(kind='pie', autopct='%0.1f%%')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f653174a7f0>
```



Sex column

conclusion

```
df['Sex'].describe()
```

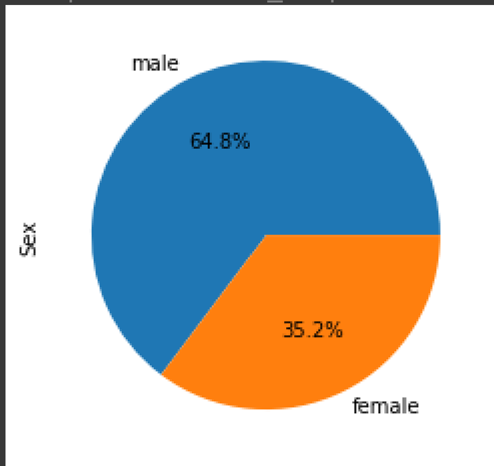
```
count      891
unique       2
top       male
freq       577
Name: Sex, dtype: object
```

```
df['Sex'].value_counts()
```

```
male      577
female    314
Name: Sex, dtype: int64
```

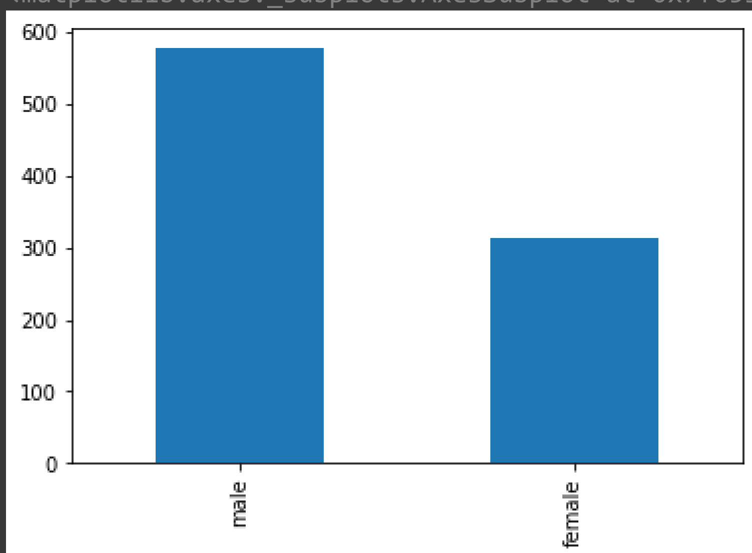
```
df['Sex'].value_counts().plot(kind='pie', autopct='%0.1f%%')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f653156f2b0>
```




```
df['Sex'].value_counts().plot(kind='bar')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f653159dc40>
```




```
# missing values
df['Sex'].isnull().sum()
```

```
0
```

SibSp column

```
df['SibSp'].describe()
```

```
count    891.000000
mean       0.523008
std        1.102743
min        0.000000
25%        0.000000
50%        0.000000
```

```
75%          1.000000
max          8.000000
Name: SibSp, dtype: float64
```

```
df['SibSp'].isnull().sum()
```

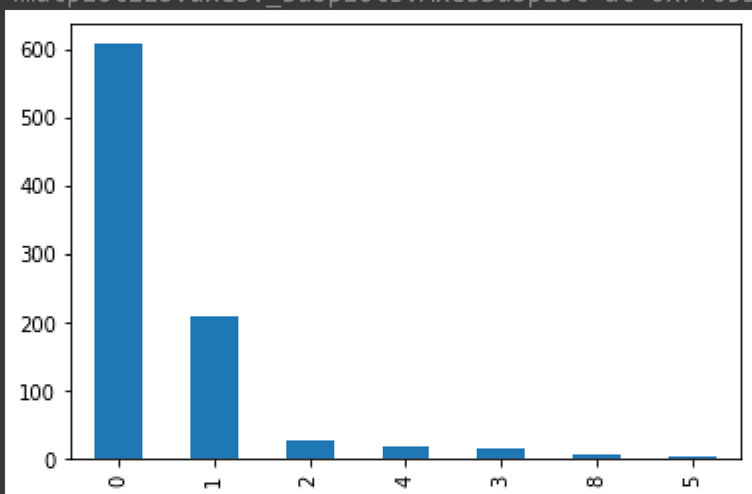
```
0
```

```
df['SibSp'].value_counts()
```

```
0    608
1    209
2     28
4     18
3     16
8      7
5      5
Name: SibSp, dtype: int64
```

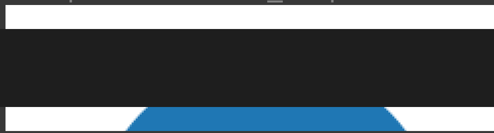
```
df['SibSp'].value_counts().plot(kind='bar')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f6531502700>
```




```
df['SibSp'].value_counts().plot(kind='pie')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f653143fa60>
```



Parch column

conclusion

- Parch col and SibSp cols can be merge to form a new col called `family_size`
- Create a new col called `is_alone`

```
df['Parch'].describe()
```

```
count    891.000000
mean       0.381594
std        0.806057
min        0.000000
25%        0.000000
50%        0.000000
75%        0.000000
max        6.000000
Name: Parch, dtype: float64
```

```
df['Parch'].value_counts()
```

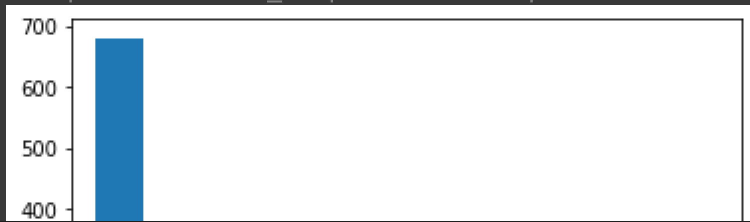
```
0    678
1    118
2     80
5      5
3      5
4      4
6      1
Name: Parch, dtype: int64
```

```
df['Parch'].isnull().sum()
```

```
0
```

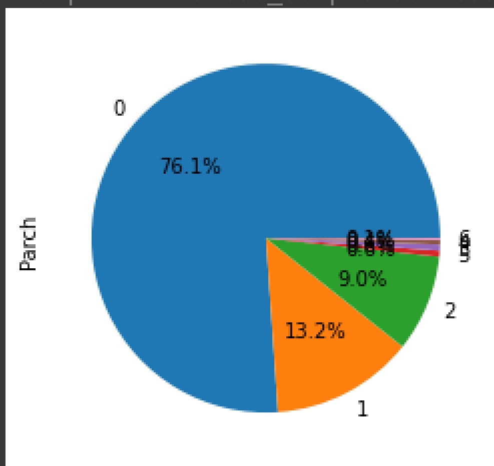
```
df['Parch'].value_counts().plot(kind='bar')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f65319fa430>
```




```
df['Parch'].value_counts().plot(kind='pie', autopct='%0.1f%%')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f65313f6b50>
```



Ebarked column

conclusion

- 2 missing values found

```
df['Embarked'].describe()
```

```
count     889
unique      3
top         S
freq      644
Name: Embarked, dtype: object
```

```
df['Embarked'].isnull().sum()
```

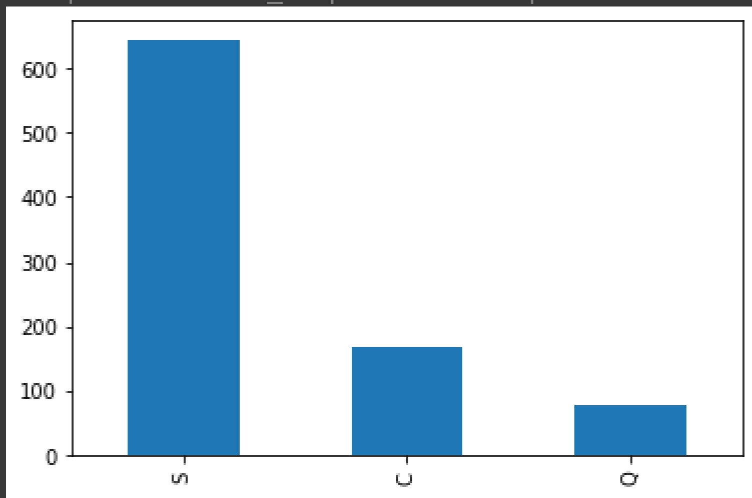
```
2
```

```
df['Embarked'].value_counts()
```

```
S    644
C    168
Q     77
Name: Embarked, dtype: int64
```

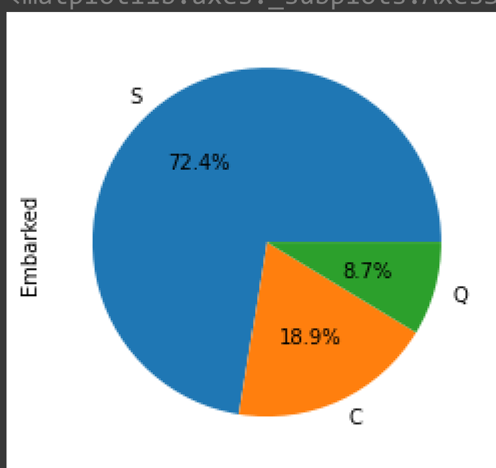
```
df['Embarked'].value_counts().plot(kind='bar')
```

<matplotlib.axes._subplots.AxesSubplot at 0x7f6531c0c460>




```
df['Embarked'].value_counts().plot(kind='pie', autopct='%0.1f%%')
```

<matplotlib.axes._subplots.AxesSubplot at 0x7f65313a3790>



mixed columns

- firstly have to do Feature Engineering for Analysis

Steps of doing Bivariate Analysis

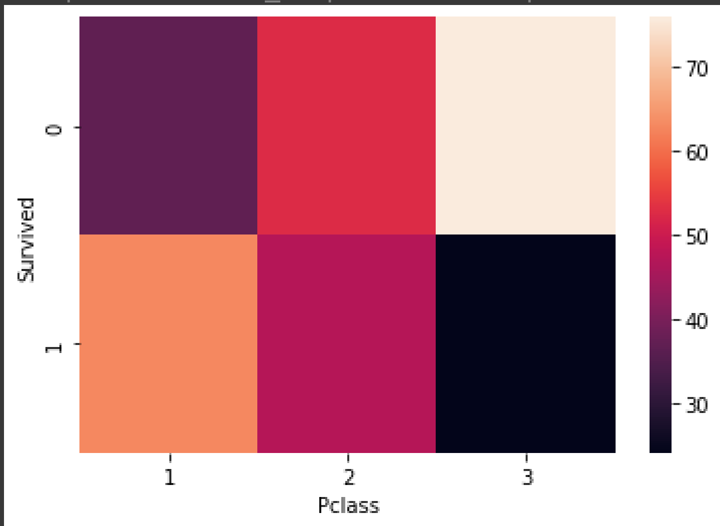
- Select 2 cols
- Understand type of relationship
  1. **Numerical - Numerical**
     a. You can plot graphs like scatterplot(regression plots), 2D histplot, 2D KDEplots
     b. Check correlation coefficent to check linear relationship
  2. **Numerical - Categorical** - create visualizations that compare the distribution of the numerical data across different categories of the categorical data.
     a. You can plot graphs like barplot, boxplot, kdeplot violinplot even scatterplots
  3. **Categorical - Categorical**
     a. You can create cross-tabulations or contingency tables that show the distribution of values in one categorical column, grouped by the values in the other categorical column.
     b. You can plots like heatmap, stacked barplots, treemaps
- Write your conclusions

```
df
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fa
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.25

```
sns.heatmap(pd.crosstab(df['Survived'], df['Pclass'], normalize='columns')*100)
```

<matplotlib.axes._subplots.AxesSubplot at 0x7f65312e3250>



4	5	0	3	William	male	35.0	0	0	373450	8.05

```
# categorical - categorical column (Bivariate Analysis)
# create cross-tabulations or contingency tables

pd.crosstab(df['Survived'], df['Pclass'])
```

Pclass	1	2	3
Survived			
0	80	97	372
1	136	87	119

```
pd.crosstab(df['Survived'], df['Pclass'], normalize='columns')*100
```

Pclass	1	2	3
Survived			
0	37.037037	52.717391	75.763747
1	62.962963	47.282609	24.236253

```
pd.crosstab(df['Survived'], df['Sex'], normalize='columns')*100
```

Sex	female	male
Survived		
0	25.796178	81.109185
1	74.203822	18.890815

```
pd.crosstab(df['Survived'], df['Embarked'], normalize='columns')*100
```

Embarked	C	Q	S
Survived			
0	44.642857	61.038961	66.304348
1	55.357143	38.961039	33.695652

```
pd.crosstab(df['Sex'], df['Embarked'], normalize='columns')*100
```

Embarked	C	Q	S
Sex			
female	43.452381	46.753247	31.521739
male	56.547619	53.246753	68.478261

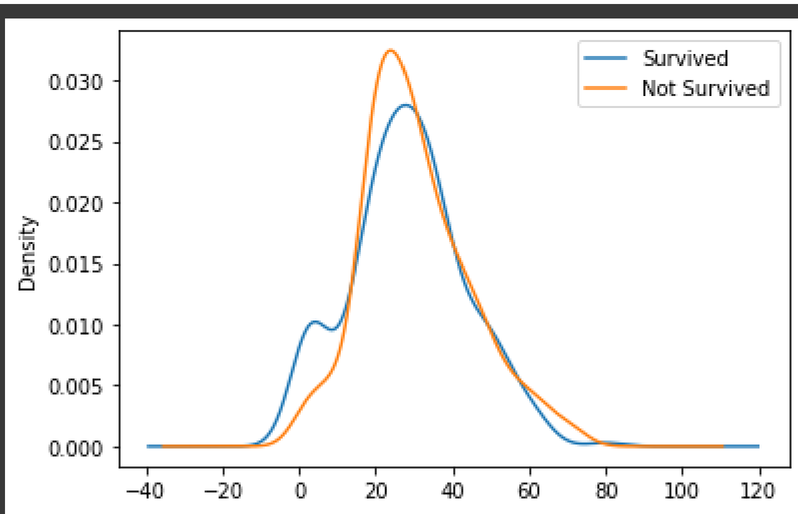
```
pd.crosstab(df['Pclass'], df['Embarked'], normalize='columns')*100
```

Embarked	C	Q	S
Pclass			
1	50.595238	2.597403	19.720497
2	10.119048	3.896104	25.465839
3	39.285714	93.506494	54.813665

Numerical - categorical (Bivariate Anlysis)

```
# Survived and age

df[df['Survived'] == 1]['Age'].plot(kind='kde', label='Survived')
df[df['Survived'] == 0]['Age'].plot(kind='kde', label='Not Survived')
plt.legend()
plt.show()
```

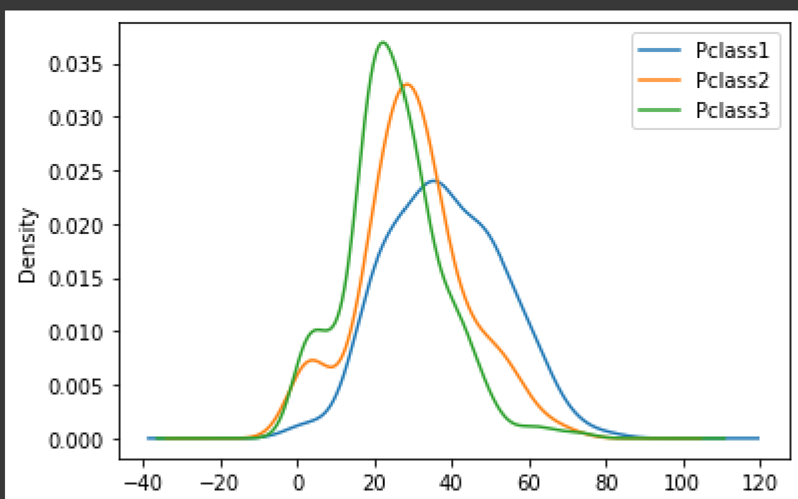



```
df[df['Pclass'] == 1]['Age'].mean()
```

```
38.233440860215055
```

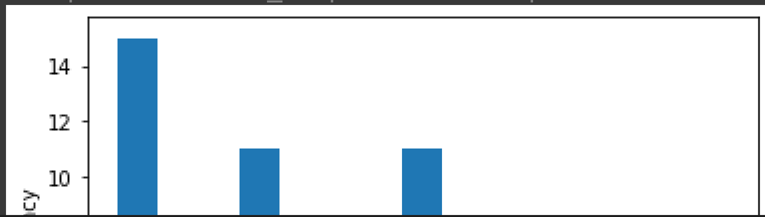
```
df[df['Pclass'] == 1]['Age'].plot(kind='kde', label='Pclass1')
df[df['Pclass'] == 2]['Age'].plot(kind='kde', label='Pclass2')
df[df['Pclass'] == 3]['Age'].plot(kind='kde', label='Pclass3')

plt.legend()
plt.show()
```



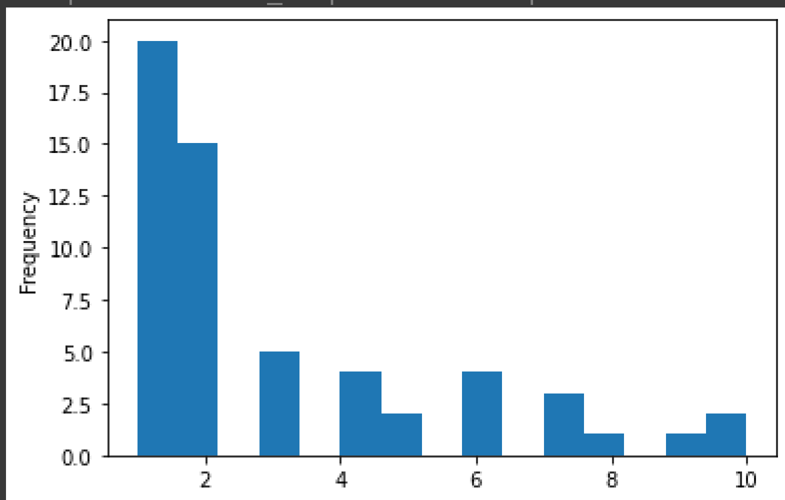

```
df[df['Pclass'] == 1]['Age'].value_counts().plot(kind='hist', bins=15)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f652df89e50>
```



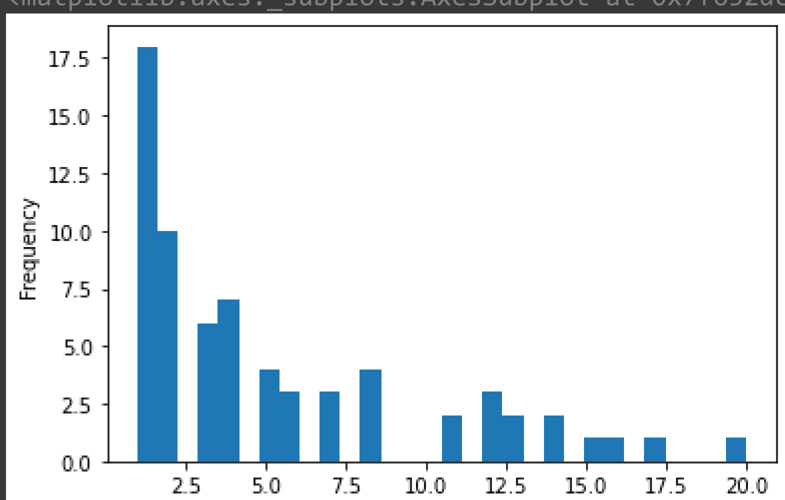

```
df[df['Pclass'] == 2]['Age'].value_counts().plot(kind='hist', bins=15)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f652e00a250>
```




```
df[df['Pclass'] == 3]['Age'].value_counts().plot(kind='hist', bins=30)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f652dc6d100>
```




```
# Feature Enginnering on Fear column
```

```
df['SibSp'].value_counts()
```

```
0    608
1    209
```

```
2     28
4     18
3     16
8      7
5      5
Name: SibSp, dtype: int64
```

```
df[df['SibSp']==8]
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare
159	160	0	3	Sage, Master. Thomas Henry	male	NaN	8	2	CA. 2343	69.55
180	181	0	3	Sage, Miss. Constance Gladys	female	NaN	8	2	CA. 2343	69.55
201	202	0	3	Sage, Mr. Frederick	male	NaN	8	2	CA. 2343	69.55

```
df[df['Ticket']=='CA. 2343']
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare
159	160	0	3	Sage, Master. Thomas Henry	male	NaN	8	2	CA. 2343	69.55
180	181	0	3	Sage, Miss. Constance Gladys	female	NaN	8	2	CA. 2343	69.55
201	202	0	3	Sage, Mr. Frederick	male	NaN	8	2	CA. 2343	69.55

```
df[df['Name'].str.contains('Sage')]
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare
159	160	0	3	Sage, Master. Thomas Henry	male	NaN	8	2	CA. 2343	69.55

Sage

```
df1 = pd.read_csv('/content/test.csv')
df = pd.concat([df,df1])
df
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	
0	1	0.0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	
1	2	1.0	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	7
2	3	1.0	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	
3	4	1.0	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	5
4	5	0.0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	
...	...	...	...	...	...	...	...	...	...	

```
df[df['Ticket']=='CA.·2343']
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare
159	160	0.0	3	Sage, Master. Thomas Henry	male	NaN	8	2	CA. 2343	69.55
180	181	0.0	3	Sage, Miss. Constance Gladys	female	NaN	8	2	CA. 2343	69.55

```
df[df['Ticket']=='CA 2144']
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Ca
59	60	0.0	3	Goodwin, Master. William Frederick	male	11.0	5	2	CA 2144	46.9	
71	72	0.0	3	Goodwin, Miss. Lillian Amy	female	16.0	5	2	CA 2144	46.9	
386	387	0.0	3	Goodwin, Master. Sidney Leonard	male	1.0	5	2	CA 2144	46.9	

```
# creating new column
df['individual_fare'] = df['Fare'] / (df['SibSp']+df['Parch']+1)
df['individual_fare']
```

```
0        3.625000
1       35.641650
2        7.925000
3       26.550000
4        8.050000
          ...
413      8.050000
414    108.900000
415      7.250000
416      8.050000
417      7.452767
Name: individual_fare, Length: 1309, dtype: float64
```

```
df['individual_fare'].describe()
```

```
count    1308.000000
mean       20.518215
```

```
std       35.774337
min        0.000000
25%        7.452767
50%        8.512483
75%       24.237500
max      512.329200
Name: individual_fare, dtype: float64
```

```
df[['Fare','individual_fare']].describe()
```

	Fare	individual_fare
count	1308.000000	1308.000000
mean	33.295479	20.518215
std	51.758668	35.774337
min	0.000000	0.000000
25%	7.895800	7.452767
50%	14.454200	8.512483
75%	31.275000	24.237500
max	512.329200	512.329200

```
df
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket
0	1	0.0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171
				Cumings, Mrs. John					

```
# featuring engineering
# new column called family_size

df['family_size'] = df['SibSp']+df['Parch']+1
df
```

PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket

```
# creating family_type column
# 1 -> alone
# 2-4 -> small
# >5  -> large

def transform_family_size(num):
  if num == 1:
    return 'alone'
  elif num>1 and num<5:
    return 'small'
  else:
    return 'large'
```

Mrs.

```
df['family_type'] = df['family_size'].apply(transform_family_size)
df
```

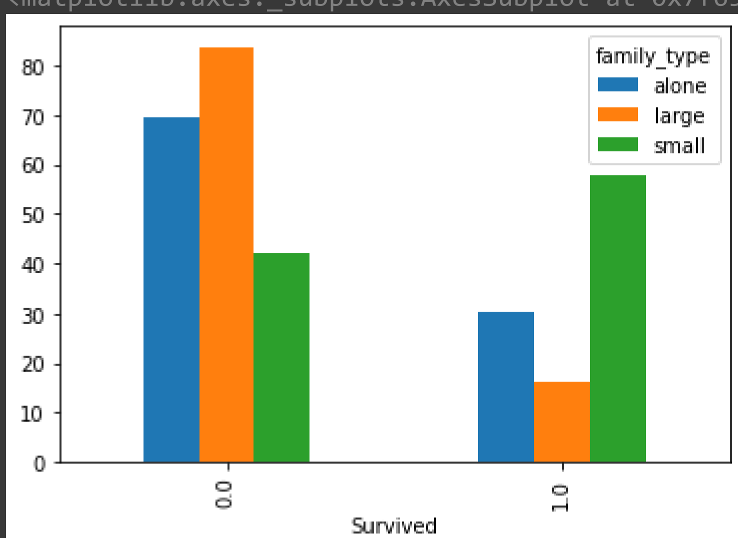
	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket
0	1	0.0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171
1	2	1.0	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599
2	3	1.0	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282
				Futrelle, Mrs. Jacques					
413	1305	NaN	3		male	NaN	0	0	A.5. 3236

```
pd.crosstab(df['Survived'], df['family_type'], normalize = 'columns')*100
```

family_type Survived	alone	large	small
0.0	69.646182	83.870968	42.123288
1.0	30.353818	16.129032	57.876712

```
a = pd.crosstab(df['Survived'], df['family_type'], normalize = 'columns')*100
a.plot(kind='bar')
```

<matplotlib.axes._subplots.AxesSubplot at 0x7f652da78ca0>




```
# Feature Engineering  ->  working with Name column
df['Name'].str.split(',')
```

```
0                              [Braund,  Mr. Owen Harris]
1      [Cumings,  Mrs. John Bradley (Florence Briggs ...
2                               [Heikkinen,  Miss. Laina]
3        [Futrelle,  Mrs. Jacques Heath (Lily May Peel)]
4                             [Allen,  Mr. William Henry]
                              ...
413                                 [Spector,  Mr. Woolf]
414                         [Oliva y Ocana,  Dona. Fermina]
415                         [Saether,  Mr. Simon Sivertsen]
416                                [Ware,  Mr. Frederick]
417                            [Peter,  Master. Michael J]
Name: Name, Length: 1309, dtype: object
```

```
df['surname'] = df['Name'].str.split(',').str.get(0)
df
```

Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Emba
0.0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	
1.0	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	
		Heikkinen,					STON/O2			

```
# Name title
df['title'] = df['Name'].str.split(',').str.get(1).str.strip().str.split(' ').str.get(0)
df['title'].value_counts()
```

```
Mr.          757
Miss.        260
Mrs.         197
Master.       61
Rev.           8
Dr.            8
Col.           4
Mlle.          2
Major.         2
Ms.            2
Lady.          1
Sir.           1
Mme.           1
Don.           1
Capt.          1
the            1
Jonkheer.      1
Dona.          1
Name: title, dtype: int64
```

3101262

```
# not worked
l = ['Dr.','Col.','Major.','Don.','Capt.','the','Jhonkheer.']
def transform_title(l):
  return df['title'].str.replace(l, 'other')
```

NaN	3	Master.	male	NaN	1	1	2668	22.3583	NaN

```
df['title'].apply(transform_title)
```

```
<ipython-input-193-a0285dc59117>:3: FutureWarning: The default value of regex will chang
  return df['title'].str.replace(l, 'other')
---------------------------------------------------------------------------
InvalidIndexError                         Traceback (most recent call last)
<ipython-input-194-ce6354f1617d> in <module>
----> 1 df['title'].apply(transform_title)

                              7 frames
/usr/local/lib/python3.8/dist-packages/pandas/core/indexes/base.py in get_indexer(self,
target, method, limit, tolerance)
   3440
```

```
temp_df1 = df[df['title'].isin(['Dr.','Col.','Major.','Don.','Capt.','the','Jhonkheer.'])]
temp_df1
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fa
30	31	0.0	1	Uruchurtu, Don. Manuel E	male	40.0	0	0	PC 17601	27.72
245	246	0.0	1	Minahan, Dr. William Edward	male	44.0	2	0	19928	90.00
317	318	0.0	2	Moraweck, Dr. Ernest	male	54.0	0	0	29011	14.00
398	399	0.0	2	Pain, Dr. Alfred	male	23.0	0	0	244278	10.50
449	450	1.0	1	Peuchen, Major. Arthur Godfrey	male	52.0	0	0	113786	30.50
536	537	0.0	1	Butt, Major. Archibald Willingham	male	45.0	0	0	113050	26.55
632	633	1.0	1	Stahelin-Maeglin, Dr. Max	male	32.0	0	0	13214	30.50
647	648	1.0	1	Simonius-Blumer, Col. Oberst Alfons	male	56.0	0	0	13213	35.50
				Frauenthal,					PC	

```
df[df['other_title']]
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fa
30	31	0.0	1	Uruchurtu, Don. Manuel E	male	40.0	0	0	PC 17601	27.72
245	246	0.0	1	Minahan, Dr. William Edward	male	44.0	2	0	19928	90.00
317	318	0.0	2	Moraweck, Dr. Ernest	male	54.0	0	0	29011	14.00
398	399	0.0	2	Pain, Dr. Alfred	male	23.0	0	0	244278	10.50
449	450	1.0	1	Peuchen, Major. Arthur Godfrey	male	52.0	0	0	113786	30.50
536	537	0.0	1	Butt, Major. Archibald Willingham	male	45.0	0	0	113050	26.55
632	633	1.0	1	Stahelin-Maeglin, Dr. Max	male	32.0	0	0	13214	30.50
647	648	1.0	1	Simonius-Blumer, Col. Oberst Alfons	male	56.0	0	0	13213	35.50
660	661	1.0	1	Frauenthal, Dr. Henry William	male	50.0	2	0	PC 17611	133.65
694	695	0.0	1	Weir, Col. John	male	60.0	0	0	113800	26.55
745	746	0.0	1	Crosby, Capt. Edward Gifford	male	70.0	1	1	WE/P 5735	71.00
759	760	1.0	1	Rothes, the Countess. of (Lucy Noel Martha Dye...	female	33.0	0	0	110152	86.50
766	767	0.0	1	Brewe, Dr. Arthur Jackson	male	NaN	0	0	112379	39.60

```
temp_df = df[df['title'].isin(['Mr.','Miss.','Mrs.','Master.'])]
temp_df
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket
0	1	0.0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171
1	2	1.0	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599
2	3	1.0	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282
3	4	1.0	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803
4	5	0.0	3	Allen, Mr. William Henry	male	35.0	0	0	373450
...	...	...	...	...	...	...	...	...	...
412	1304	NaN	3	Henriksson, Miss. Jenny Lovisa	female	28.0	0	0	347086
413	1305	NaN	3	Spector, Mr. Woolf	male	NaN	0	0	A.5. 3236
415	1307	NaN	3	Saether, Mr. Simon Sivertsen	male	38.5	0	0	SOTON/O.Q. 3101262
416	1308	NaN	3	Ware, Mr. Frederick	male	NaN	0	0	359309
417	1309	NaN	3	Peter, Master. Michael J	male	NaN	1	1	2668

1275 rows × 18 columns

```
pd.crosstab(temp_df['Survived'], temp_df['title'], normalize='columns')*100  # percentage
```

title	Master.	Miss.	Mr.	Mrs.
Survived				
0.0	42.5	30.21978	84.332689	20.8
1.0	57.5	69.78022	15.667311	79.2

```
pd.crosstab(temp_df1['Survived'], temp_df1['title'],normalize='columns')*100  # percentage
```

title	Capt.	Col.	Don.	Dr.	Major.	the
Survived						
0.0	100.0	50.0	100.0	57.142857	50.0	0.0
1.0	0.0	50.0	0.0	42.857143	50.0	100.0

```
# df['title'] = df['title'].str.replace('Rev.','other')
# df['title'] = df['title'].str.replace('Dr.','other')
# df['title'] = df['title'].str.replace('Col.','other')
# df['title'] = df['title'].str.replace('Major.','other')
# df['title'] = df['title'].str.replace('Capt.','other')
# df['title'] = df['title'].str.replace('the','other')
# df['title'] = df['title'].str.replace('Jonkheer.','other')
# ,'Dr.','Col.','Major.','Don.','Capt.','the','Jonkheer.']
```

```
# cabin column
df['Cabin'].isnull().sum()
```

```
1014
```

```
df['Cabin'].isnull().sum()/len(df['Cabin'])
```

```
0.774637127578304
```

```
df['Cabin'].value_counts().head(20)
```

```
C23 C25 C27        6
G6                 5
B57 B59 B63 B66    5
C22 C26            4
F33                4
F2                 4
B96 B98            4
C78                4
F4                 4
D                  4
E34                3
B58 B60            3
A34                3
```

```
E101                3
C101                3
B51 B53 B55         3
C31                 2
C55 C57             2
D37                 2
C54                 2
Name: Cabin, dtype: int64
```

```
df['Cabin'].fillna('M', inplace=True)
```

```
df['Cabin'].value_counts()
```

```
M                   1014
C23 C25 C27            6
B57 B59 B63 B66        5
G6                     5
F33                    4
                    ...
A14                    1
E63                    1
E12                    1
E38                    1
C105                   1
Name: Cabin, Length: 187, dtype: int64
```

```
df['deck'] = df['Cabin'].str.get(0)
```

```
df['deck'].value_counts()
```

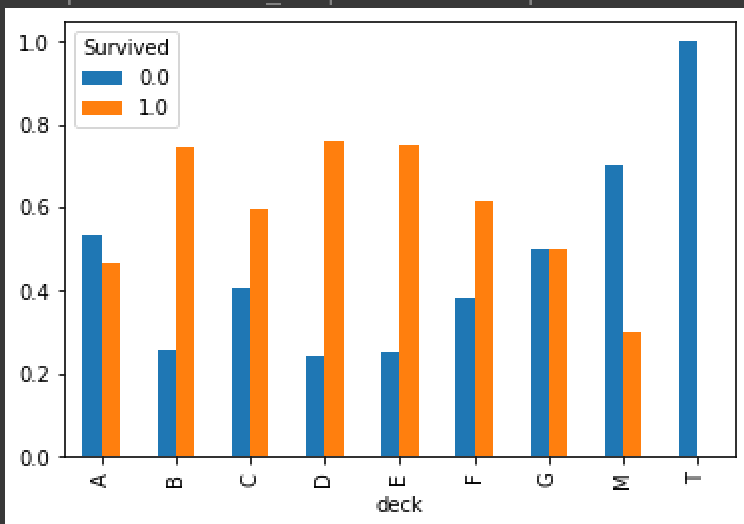
```
M    1014
C      94
B      65
D      46
E      41
A      22
F      21
G       5
T       1
Name: deck, dtype: int64
```

```
pd.crosstab(df['deck'],df['Pclass'])
```

Pclass	1	2	3
deck			
A	22	0	0
B	65	0	0
C	94	0	0
D	40	6	0
E	34	4	3

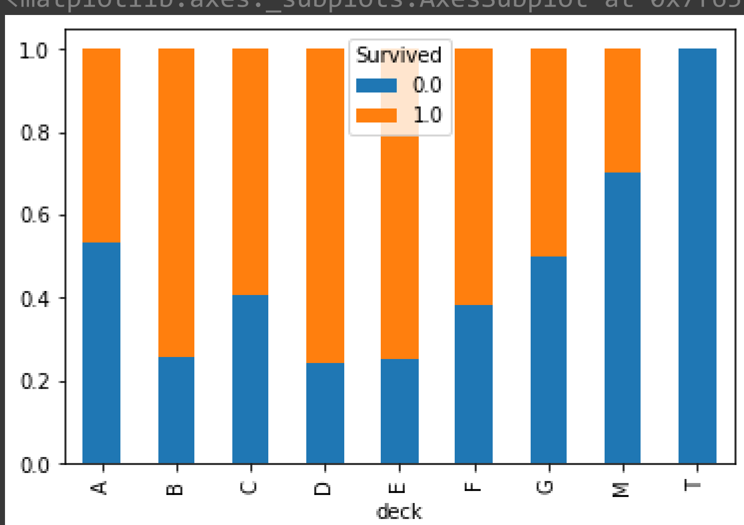
```
pd.crosstab(df['deck'], df['Survived'], normalize='index').plot(kind='bar')
```

<matplotlib.axes._subplots.AxesSubplot at 0x7f652e5f7310>



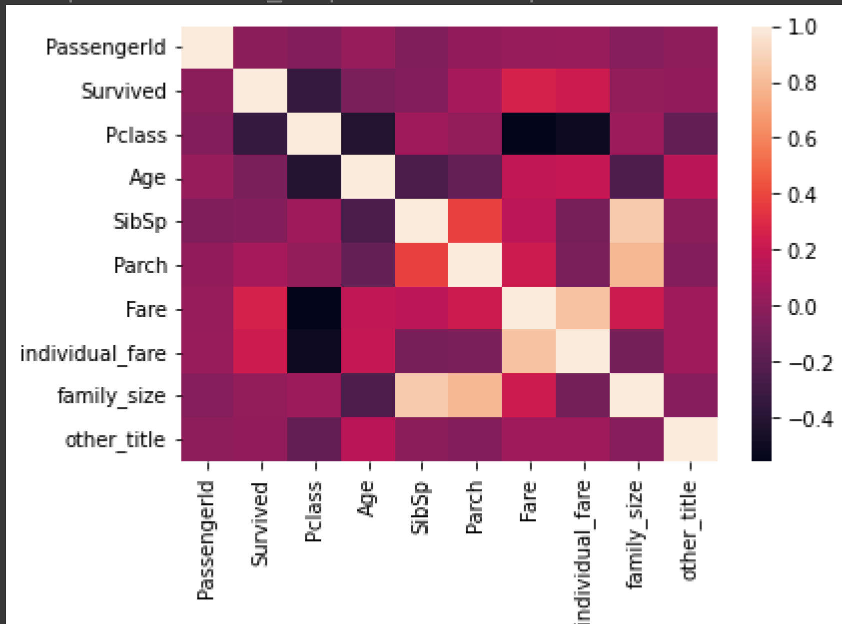

```
pd.crosstab(df['deck'], df['Survived'], normalize='index').plot(kind='bar', stacked=True)
```

<matplotlib.axes._subplots.AxesSubplot at 0x7f652b419dc0>



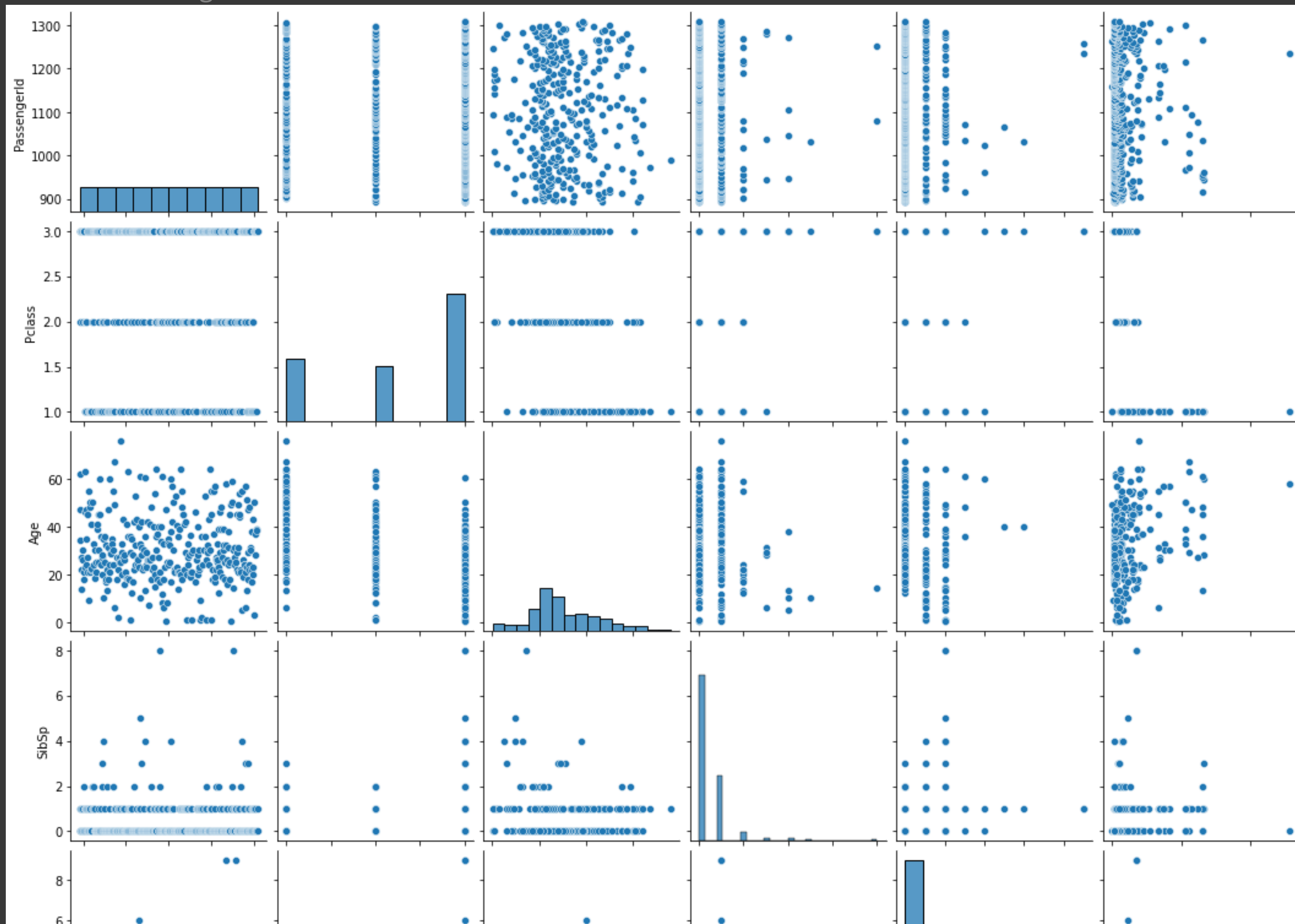

```
sns.heatmap(df.corr())
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f652b3f3fd0>
```




```
sns.pairplot(df1)
```

```
<seaborn.axisgrid.PairGrid at 0x7f652b0aa310>
```




```
df1
```

	PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cab
0	892	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	N
1	893	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	N
2	894	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	N
3	895	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	N
4	896	3	Hirvonen, Mrs. Alexander (Helga E	female	22.0	1	1	3101298	12.2875	N

0s completed at 9:01 PM