Data Wrangling

In [ ]:
```
#install libraries
#pip install pandas
#pip install seaborn
#pip install numpy
```

In [ ]:
```
#import llibraries
import pandas as pd
import numpy as np
import seaborn as sns
```

In [ ]:
```
kashti = sns.load_dataset('titanic')
ks1 = kashti
ks2 = kashti
ks = sns.load_dataset('titanic')
```

In [ ]:
```
kashti.head()
```

Out[ ]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	decl
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	NaN
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	C
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	NaN
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	C
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	NaN

In [ ]:
```
# simple operations (Math operator)
(kashti['age']+12).head(10)
```

Out[ ]:
```
0    34.0
1    50.0
2    38.0
3    47.0
4    47.0
5     NaN
6    66.0
7    14.0
8    39.0
9    26.0
Name: age, dtype: float64
```

In [ ]:
```
#where exactly missing values are?
kashti.isnull().sum()
```

Out[ ]:
```
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
```

In [ ]:
```
#usd drop.na method
print(kashti.shape)
#kashti.dropna(subset=['deck'], axis=0, inplace=True) # this will remove specificall
#inplace = true modifies the data frame
```

```
(891, 15)
```

In [ ]:
```
kashti.isnull().sum()
```

Out[ ]:
```
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
```

In [ ]:
```
kashti = kashti.dropna()
kashti.isnull().sum() # remove na from whole dataframe
```

Out[ ]:
```
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
```

In [ ]:
```
kashti.shape
```

Out[ ]:
```
(182, 15)
```

In [ ]:
```
ks1.isnull().sum()
```

Out[ ]:
```
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
```

Replacing missing values with the average of that column

In [ ]:
```
#finding an average (mean)
mean = ks1['age'].mean()
mean
```

Out[ ]:
```
29.69911764705882
```

In [ ]:
```
# replacing nan with mean of the data (updating as well)
ks1['age'] = ks1['age'].replace(np.nan, mean)
```

In [ ]:
```
ks1.isnull().sum()
```

Out[ ]:
```
survived         0
pclass           0
sex              0
age              0
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
```

In [ ]:
```
ks1.dropna(subset=['deck'], axis=0, inplace=True)
ks1.dropna(subset=['embarked'], axis=0, inplace=True)
ks1.dropna(subset=['embark_town'], axis=0, inplace=True)
```

In [ ]:
```
ks1.isnull().sum()
```

Out[ ]:
```
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
```

Data Formatting

In [ ]:
```
# know the data type and convert it into the known one
kashti.dtypes
```

Out[ ]:
```
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 [ ]:
```
# use this method to convert datatype from one to another format
kashti['survived'] = kashti['survived'].astype("float64")
kashti.dtypes
```

```
C:\Users\Nasir\AppData\Local\Temp/ipykernel_14724/2526506595.py:2: SettingWithCopyWa
rning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  kashti['survived'] = kashti['survived'].astype("float64")
```

Out[ ]:
```
survived        float64
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 [ ]:
```
# here we will convert the age into days instead of years
ks1['age'] = ks1['age']*365
ks1.head(10)
```

Out[ ]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adu
1	1	1	female	13870.000000	1	0	71.2833	C	First	woman	
3	1	1	female	12775.000000	1	0	53.1000	S	First	woman	
6	0	1	male	19710.000000	0	0	51.8625	S	First	man	
10	1	3	female	1460.000000	1	1	16.7000	S	Third	child	
11	1	1	female	21170.000000	0	0	26.5500	S	First	woman	
21	1	2	male	12410.000000	0	0	13.0000	S	Second	man	
23	1	1	male	10220.000000	0	0	35.5000	S	First	man	
27	0	1	male	6935.000000	3	2	263.0000	S	First	man	
31	1	1	female	10840.177941	1	0	146.5208	C	First	woman	
52	1	1	female	17885.000000	1	0	76.7292	C	First	woman	

In [ ]:
```
# always rename afterwards
ks1.rename(columns={"age": "age in days"}, inplace=True)
ks1.head()
```

Out[ ]:

	survived	pclass	sex	age in days	sibsp	parch	fare	embarked	class	who	adult_male
1	1	1	female	13870.0	1	0	71.2833	C	First	woman	False
3	1	1	female	12775.0	1	0	53.1000	S	First	woman	False
6	0	1	male	19710.0	0	0	51.8625	S	First	man	True
10	1	3	female	1460.0	1	1	16.7000	S	Third	child	False
11	1	1	female	21170.0	0	0	26.5500	S	First	woman	False

In [ ]:
```
ks1['age in days'] = ks1['age in days'].astype("int64")
ks1.head(10)
```

Out[ ]:

	survived	pclass	sex	age in days	sibsp	parch	fare	embarked	class	who	adult_male
1	1	1	female	13870	1	0	71.2833	C	First	woman	False

	survived	pclass	sex	age in days	sibsp	parch	fare	embarked	class	who	adult_male
3	1	1	female	12775	1	0	53.1000	S	First	woman	False
6	0	1	male	19710	0	0	51.8625	S	First	man	True
10	1	3	female	1460	1	1	16.7000	S	Third	child	False
11	1	1	female	21170	0	0	26.5500	S	First	woman	False
21	1	2	male	12410	0	0	13.0000	S	Second	man	True
23	1	1	male	10220	0	0	35.5000	S	First	man	True
27	0	1	male	6935	3	2	263.0000	S	First	man	True
31	1	1	female	10840	1	0	146.5208	C	First	woman	False
52	1	1	female	17885	1	0	76.7292	C	First	woman	False

Data Normalization

In [ ]:
```
kashti.head()
```

Out[ ]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	de
1	1.0	1	female	38.0	1	0	71.2833	C	First	woman	False	
3	1.0	1	female	35.0	1	0	53.1000	S	First	woman	False	
6	0.0	1	male	54.0	0	0	51.8625	S	First	man	True	
10	1.0	3	female	4.0	1	1	16.7000	S	Third	child	False	
11	1.0	1	female	58.0	0	0	26.5500	S	First	woman	False	

In [ ]:
```
ks4 = kashti[["age", "fare"]]
ks4.head()
```

Out[ ]:

	age	fare
1	38.0	71.2833
3	35.0	53.1000
6	54.0	51.8625
10	4.0	16.7000
11	58.0	26.5500

Method of Normalization

In [ ]:
```
# simle feature scaling
ks4['fare'] = ks4['fare']/ks4['fare'].max()
```

```
ks4['age'] = ks4['age']/ks4['age'].max()
ks4.head()
```

```
C:\Users\Nasir\AppData\Local\Temp/ipykernel_14724/1199163970.py:2: SettingWithCopyWa
rning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  ks4['fare'] = ks4['fare']/ks4['fare'].max()
C:\Users\Nasir\AppData\Local\Temp/ipykernel_14724/1199163970.py:3: SettingWithCopyWa
rning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  ks4['age'] = ks4['age']/ks4['age'].max()
```

Out[ ]:

	age	fare
1	0.4750	0.139136
3	0.4375	0.103644
6	0.6750	0.101229
10	0.0500	0.032596
11	0.7250	0.051822

In [ ]:

```
# Min - Max Method

ks4['fare'] = (ks4['fare']-ks4['fare'].min()) / (ks4['fare'].max()-ks4['fare'].min()
ks4.head()
```

```
C:\Users\Nasir\AppData\Local\Temp/ipykernel_14724/410330791.py:3: SettingWithCopyWar
ning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  ks4['fare'] = (ks4['fare']-ks4['fare'].min()) / (ks4['fare'].max()-ks4['fare'].min
())
```

Out[ ]:

	age	fare
1	0.4750	0.139136
3	0.4375	0.103644
6	0.6750	0.101229
10	0.0500	0.032596
11	0.7250	0.051822

In [ ]:

```
# Z-score (standard score)
ks4['fare'] = (ks4['fare']-ks4['fare'].mean()) / ks4['fare'].std()
ks4.head()
```

```
C:\Users\Nasir\AppData\Local\Temp/ipykernel_14724/1430778810.py:2: SettingWithCopyWa
rning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  ks4['fare'] = (ks4['fare']-ks4['fare'].mean()) / ks4['fare'].std()
```

Out[ ]:

	age	fare
1	0.4750	-0.099835
3	0.4375	-0.337554
6	0.6750	-0.353732
10	0.0500	-0.813428
11	0.7250	-0.684654

In [ ]:

```
# Log transformation
ks['fare'] = np.log(ks['fare'])
ks.head()
```

```
C:\Users\Nasir\AppData\Local\Programs\Python\Python310\lib\site-packages\pandas\core
\arraylike.py:364: RuntimeWarning: divide by zero encountered in log
  result = getattr(ufunc, method)(*inputs, **kwargs)
```

Out[ ]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	dec
0	0	3	male	22.0	1	0	1.981001	S	Third	man	True	Na
1	1	1	female	38.0	1	0	4.266662	C	First	woman	False	
2	1	3	female	26.0	0	0	2.070022	S	Third	woman	False	Na
3	1	1	female	35.0	1	0	3.972177	S	First	woman	False	
4	0	3	male	35.0	0	0	2.085672	S	Third	man	True	Na

Binning

In [ ]:

```
kashti = sns.load_dataset('titanic')
```

In [ ]:

```
bins = np.linspace(min(kashti['age']), max(kashti['age']), 4)
age_groups = ["Bachay", "Jawan", "Boorhay"]
kashti['age'] = pd.cut(kashti['age'], bins, labels=age_groups, include_lowest=True)
kashti['age']
# how this will change the anames in dataset based on grouping? (Assignment)
```

Out[ ]:

```
0      Bachay
1       Jawan
2      Bachay
3       Jawan
4       Jawan
        ...
886     Jawan
887    Bachay
888       NaN
```

```
889    Bachay
890     Jawan
Name: age, Length: 891, dtype: category
Categories (3, object): ['Bachay' < 'Jawan' < 'Boorhay']
```

In [ ]:
```
kashti.head(10)
```

Out[ ]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male
0	0	3	male	Bachay	1	0	7.2500	S	Third	man	True
1	1	1	female	Jawan	1	0	71.2833	C	First	woman	False
2	1	3	female	Bachay	0	0	7.9250	S	Third	woman	False
3	1	1	female	Jawan	1	0	53.1000	S	First	woman	False
4	0	3	male	Jawan	0	0	8.0500	S	Third	man	True
5	0	3	male	NaN	0	0	8.4583	Q	Third	man	True
6	0	1	male	Boorhay	0	0	51.8625	S	First	man	True
7	0	3	male	Bachay	3	1	21.0750	S	Third	child	False
8	1	3	female	Jawan	0	2	11.1333	S	Third	woman	False
9	1	2	female	Bachay	1	0	30.0708	C	Second	child	False

converting categories into dummies

- easy to use for computation
- Male Female (0,1)

In [ ]:
```
ks = sns.load_dataset('titanic')
ks.head()
```

Out[ ]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	decl
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	NaN
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	C
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	NaN
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	C
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	NaN

In [ ]:
```
ks['sex'] = ks['sex'].map({'female': 1, 'male': 0})
ks.head()
# how to use get dummies to change data inside a dataframe (Assignment)
```

Out[ ]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck
0	0	3	0	22.0	1	0	7.2500	S	Third	man	True	NaN
1	1	1	1	38.0	1	0	71.2833	C	First	woman	False	C

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck
2	1	3	1	26.0	0	0	7.9250	S	Third	woman	False	NaN
3	1	1	1	35.0	1	0	53.1000	S	First	woman	False	C
4	0	3	0	35.0	0	0	8.0500	S	Third	man	True	NaN