Pandas Cheat Sheet

By Eugenia Anello

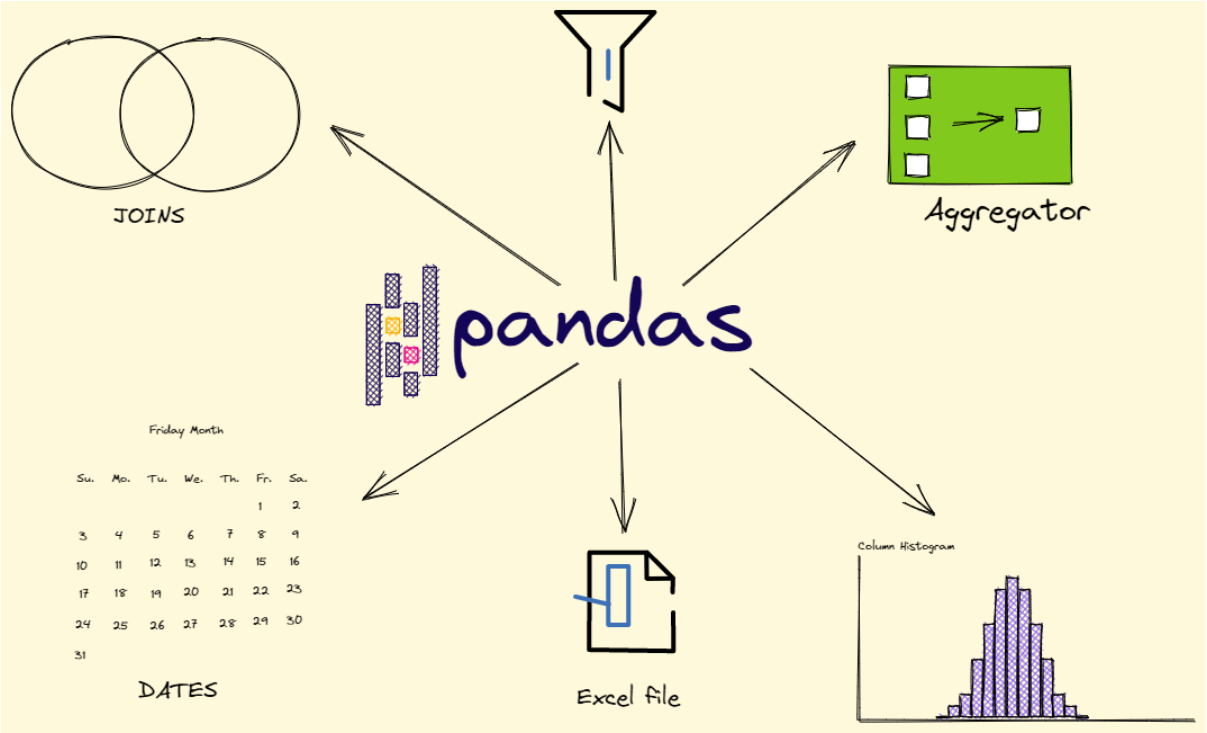


Table of Contents:

1. Import and Export files

Import Excel file

```
df = pd.read_excel('titanic.xlsx',sheet_name='survivals')
```

Export Excel file

```
df.to_excel('titanic.xlsx',sheet_name='survivals', index=False)
```

Import CSV file

```
df = pd.read_csv('titanic.csv',sep='|',dtype={'Region':'str'})
```

Export CSV file

```
df.to_csv('titanic.csv',index=False)
```

Import parquet files

```
df = pd.read_parquet('titanic.parquet')
```

Export parquet file

```
df.to_parquet('titanic.parquet')
```

2. Exploratory Data Analysis

Visualize the first five rows of the dataframe

```
df.head()
```

Visualize the last five rows of the dataframe

```
df.tail()
```

Calculate statistics of each column

```
df.describe()
```

Visualize quantiles from 0% to 100% with increments of 10%

```
df.describe([x*0.1 for x in range(10)])
```

	PassengerId	Survived	Age
count	891.000.000	891.000.000	714.000.000
mean	446.000.000	0.383838	29.699.118
std	257.353.842	0.486592	14.526.497
min	1.000.000	0.000000	0.420000
0%	1.000.000	0.000000	0.420000
10%	90.000.000	0.000000	14.000.000
20%	179.000.000	0.000000	19.000.000
...	...	...	...
80%	713.000.000	1.000.000	41.000.000
90%	802.000.000	1.000.000	50.000.000
max	891.000.000	1.000.000	80.000.000

Find unique values of a categorical column

```
df.Survived.unique()
```

Find the number of unique values of a categorical column

```
df.Survived.nunique()
```

Visualize the frequency of each modality of the categorical column

```
df.Survived.value_counts()
```

Visualize the percentages of each modality of the categorical column

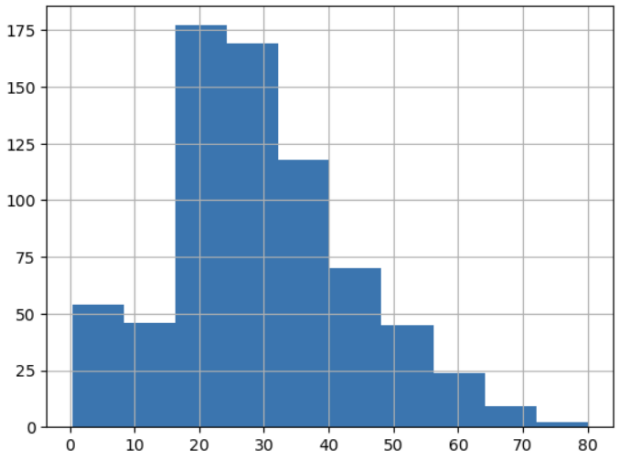
```
df.Survived.value_counts(normalize=True)*100
```

Count missing values for each column

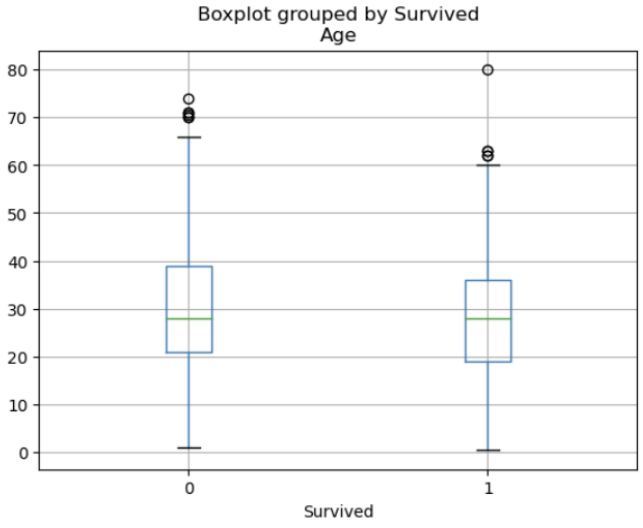
```
df.isnull().sum()
```

Plot the histogram to visualize the distribution of a column

```
df.Age.hist()
```




```
df.boxplot(column='Age',by='Survived')
```



3. Data Manipulation

Create a new feature using the `apply` function:

```
df['Surname'] = df['Name'].apply(lambda x: x.split(',')[0])
```

Create a new feature using the `map` function:

```
diz_country = {'E':'England','S':'Scotland','W':'Wales','N':'Northen
Ireland'}
df['Country'] = df['Country_ID'].map(lambda x: diz_country[x])
```

Create a categorical variables based on a numerical variable by giving in input bin values:

```
df['Age_levels'] = pd.cut(df['Age'],bins=[0,30,60,100])
```

Age	Age_levels
22.0	(0.0, 30.0]
38.0	(30.0, 60.0]
26.0	(0.0, 30.0]
35.0	(30.0, 60.0]

Create a categorical variables based on a numerical variable by giving in input bin values and replace the intervals with labels:

```
df['Age_levels'] = pd.cut(df['Age'],bins=[0,30,60,100],
                   labels=['low','middle','high'])
```

Age	Age_levels
22.0	low
38.0	middle
26.0	low
35.0	middle

Create a categorical variables based on three quantiles of a numerical variable (minimum, median and maximum):

```
df['Age_levels'] = pd.qcut(df['Age'],q=3,duplicates='drop')
```

Age	Age_levels
22.0	(0.419, 23.0]
38.0	(34.0, 80.0]
26.0	(23.0, 34.0]
35.0	(34.0, 80.0]

Create a categorical variables based on specific quantiles of a numerical variable:

```
df['Age_levels'] = pd.qcut(df['Age'],
                   q=[0,0.25,0.5,0.75,1],duplicates='drop')
```

Drop columns in dataframe:

```
df.drop(['Pclass','Cabin', 'Embarked'],axis=1,inplace=True)
```

Sort dataframe in descending order of Age:

```
df.sort_values(by='Age',ascending=False)
```

PassengerId	Survived	Age
631	1	80.0
852	0	74.0
494	0	71.0
97	0	71.0

Add two rows to dataframe:

```
data_row = {'PassengerId':[892,893],'Survived':[0,1],'Age':[26,30]}
df2 = pd.DataFrame.from_dict(data_row)
df_new = pd.concat([df,df2], ignore_index=True)
```

PassengerId	Survived	Age
890	1	26.0
891	0	32.0
892	0	26.0
893	1	30.0

Merge two dataframes with PassengerId as common primary key:

```
df_new = df.merge(df2,how='left',on='PassengerId')
```

Group data based on a categorical column:

```
df.groupby('Age_levels').agg({'Survived':pd.Series.mode,
                              'Fare':['mean','max']})
```

	Survived	Fare	
Age_levels	mode	mean	max
(0, 30]	890	1	26.0
(30, 60]	891	0	32.0
(60, 100]	892	0	26.0

4. Deal with Time Series

Convert date field to pandas datetime object.

```
df['date'] = pd.to_datetime(df['date'])
```

Extract day, month and year from date field:

```
d['day'] = df.date.dt.day
df['month']= df.date.dt.month
df['year']= df.date.dt.year
```

date	day	month	year
01/01/2013	1	1	2013
02/01/2013	2	1	2013
03/01/2013	3	1	2013

Set the date field as index

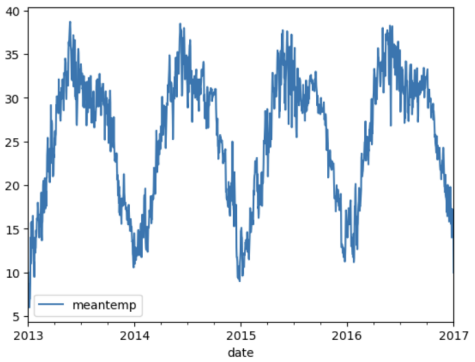
```
df.set_index('date',inplace=True)
```

Visualize the data between `'2014-07-07':'2014-07-14'`:

```
df.loc['2014-07-07':'2014-07-14']
```

Create the time series plot using the meantemp field:

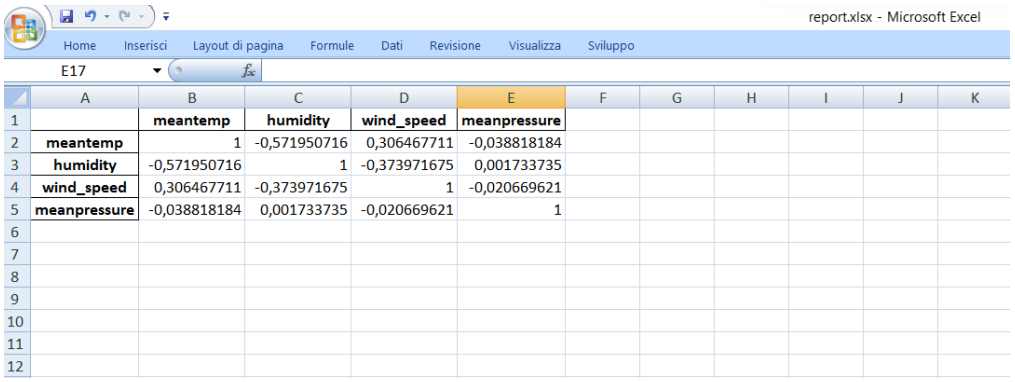
```
df.plot(x='date',y='meantemp')
```



5. ExcelWriter

Load dataframe to excel sheet

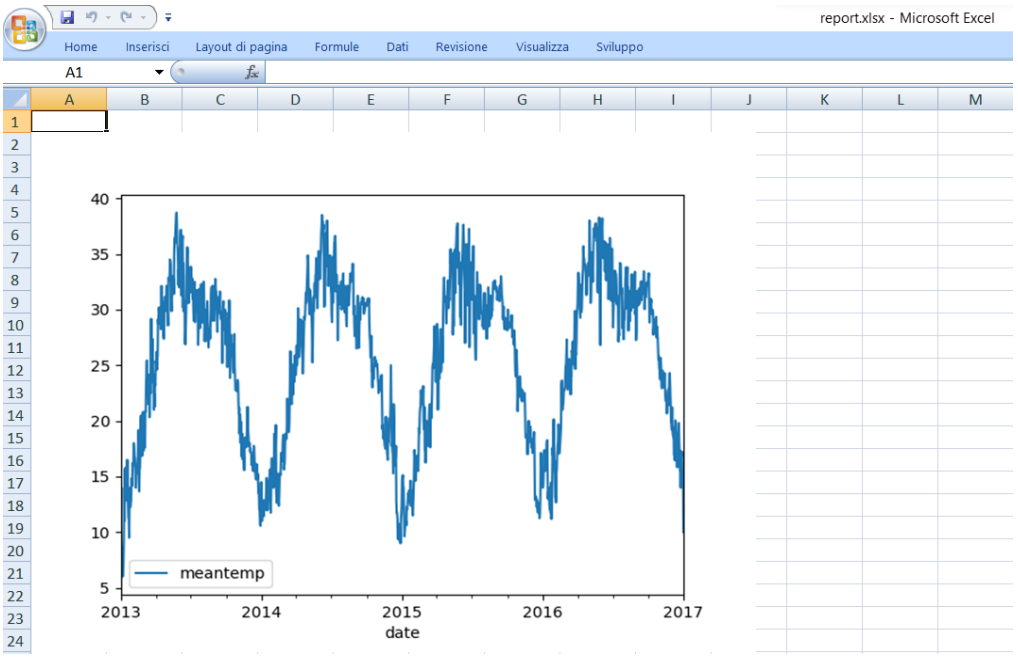
```
writer = pd.ExcelWriter('report.xlsx',engine='xlsxwriter')
df_cor = df.corr()
df_cor.to_excel(writer,sheet_name='eda',startrow=0,startcol=0,
                index=False)
```



	meantemp	humidity	wind_speed	meanpressure
meantemp	1	-0,571950716	0,306467711	-0,038818184
humidity	-0,571950716	1	-0,373971675	0,001733735
wind_speed	0,306467711	-0,373971675	1	-0,020669621
meanpressure	-0,038818184	0,001733735	-0,020669621	1

Load plot to excel sheet

```
writer = pd.ExcelWriter('report.xlsx')
workbook = writer.book
worksheet = workbook.add_worksheet('eda')
worksheet.insert_image('A2','timeseries.png')
writer.save()
```



Close and save excel file

```
writer.save()
```