Department of Data Science - Data and Visual Analytics Lab

Lab3. Pandas Indexing and Selection

Objectives

In this lab

```
- you will learn how to create Series object and Dataframe object.
- Then, you will learn to access elements using index and select rows and columns u
sing positions and column labels.
- You will finally learn aggregate functions and math operators in Pandas
```

Simple Series and DataFrames

Import necessary modules

```
In [1]: import pandas as pd
```

Create a Series to store Temperature values for 1 week

```
In [2]: temperature_trichy = pd.Series([40.2, 39.8, 36.3, 39.1, 41.3, 32.9, 36.6])
```

show temperature values

```
In [3]: temperature_trichy

Out[3]: 0    40.2
        1    39.8
        2    36.3
        3    39.1
        4    41.3
        5    32.9
        6    36.6
        dtype: float64
```

What is the weather on 2nd day?

```
In [4]: temperature_trichy [1]
Out[4]: 39.8
```

Find all days and temperatures where temperature over 40.0 degree Celsius

```
In [5]: temperature_trichy [temperature_trichy >40.0]
Out[5]: 0    40.2
        4    41.3
        dtype: float64
```

Find only day, not temperature where temperature over 40.0 degree Celsius

```
In [6]: temperature_trichy [temperature_trichy >40.0]. index
Out[6]: Int64Index([0, 4], dtype='int64')
```

Create a Dataframe for student details from List

```
In [7]: students = [['DS01', 'Rex', '1msc'], ['DS02', 'peter', '2msc'], ['CS01', 'ann'
        , '3bsc']]

        df_stud = pd.DataFrame(students, columns=['rollno', 'name', 'class'])  # row i
        ndex automatically generated
```

show df_stud dataframe

```
In [8]: df_stud
Out[8]:
```

	rollno	name	class
0	DS01	Rex	1msc
1	DS02	peter	2msc
2	CS01	ann	3bsc

Display all column names of df_stud

```
In [9]: df-stud . columns

Out[9]: Index(['rollno', 'name', 'class'], dtype='object')
```

Add a new column "address" with values ['Delhi', 'Bangalore', 'Chennai'] to df_stud

```
In [10]: address = ['Delhi', 'Bangalore', 'Chennai'], df-stud ['address']=address

In [11]: df_stud

Out[11]:
            rollno   name   class     address
         0   DS01     Rex   1msc        Delhi
         1   DS02   peter   2msc    Bangalore
         2   CS01     ann   3bsc      Chennai
```

Create a Dataframe for Phone book from Dictionary

```
In [12]: phonebook = {'rex':[9942002764, 'rex@abc.com'], 'sam':[9932176542, 'sam@xyz.co
         m'], 'peter':[9865323645, 'ann@bhc.com']}
         df_phonebook = pd.DataFrame.from_dict(phonebook, orient='index', columns=['mob
         ile', 'email'])
```

Display df_phonebook

```
In [ ]: df-phonebook
```

Exploratory Data Analysis on Video Game Review Dataset

Import ign.csv dataset

```
In [13]: reviews = pd.read_csv("ign.csv")
```

Show top-5 rows

In [14]:
```
reviews.head()
```

Out[14]:

	Unnamed: 0	score_phrase	title	url	platform	score	genre	ed
0	0	Amazing	LittleBigPlanet PS Vita	/games/littlebigplanet-vita/vita-98907	PlayStation Vita	9.0	Platformer	
1	1	Amazing	LittleBigPlanet PS Vita -- Marvel Super Hero E...	/games/littlebigplanet-ps-vita-marvel-super-he...	PlayStation Vita	9.0	Platformer	
2	2	Great	Splice: Tree of Life	/games/splice/ipad-141070	iPad	8.5	Puzzle	
3	3	Great	NHL 13	/games/nhl-13/xbox-360-128182	Xbox 360	8.5	Sports	
4	4	Great	NHL 13	/games/nhl-13/ps3-128181	PlayStation 3	8.5	Sports	

Show bottom 3 rows

In [15]:
```
reviews.tail(3)
```

Out[15]:

	Unnamed: 0	score_phrase	title	url	platform	score	genre	editc
18622	18622	Mediocre	Star Ocean: Integrity and Faithlessness	/games/star-ocean-5/ps4-20035681	PlayStation 4	5.8	RPG	
18623	18623	Masterpiece	Inside	/games/inside-playdead/xbox-one-121435	Xbox One	10.0	Adventure	
18624	18624	Masterpiece	Inside	/games/inside-playdead/pc-20055740	PC	10.0	Adventure	

How many rows and columns here?

In [16]:
```
reviews.shape
```

Out[16]:
```
(18625, 11)
```

What are the datatypes?

```
In [17]: reviews.dtypes
```

```
Out[17]: Unnamed: 0           int64
         score_phrase        object
         title               object
         url                 object
         platform            object
         score              float64
         genre               object
         editors_choice      object
         release_year         int64
         release_month        int64
         release_day          int64
         dtype: object
```

Selecting Columns

Select a single column, say title and print head

```
In [18]: reviews['title'].tail()
```

```
Out[18]: 18620                 Tokyo Mirage Sessions #FE
         18621           LEGO Star Wars: The Force Awakens
         18622    Star Ocean: Integrity and Faithlessness
         18623                                      Inside
         18624                                      Inside
         Name: title, dtype: object
```

Select multiple columns, title and genre and print head

```
In [19]: reviews[['title','genre']].head(10)
```

Out[19]:

	title	genre
0	LittleBigPlanet PS Vita	Platformer
1	LittleBigPlanet PS Vita -- Marvel Super Hero E...	Platformer
2	Splice: Tree of Life	Puzzle
3	NHL 13	Sports
4	NHL 13	Sports
5	Total War Battles: Shogun	Strategy
6	Double Dragon: Neon	Fighting
7	Guild Wars 2	RPG
8	Double Dragon: Neon	Fighting
9	Total War Battles: Shogun	Strategy

Selection using Positions

Select top-5 rows and all columns, same as head() using iloc

```
In [20]: reviews.iloc[:5,:]
Out[20]:
```

	Unnamed: 0	score_phrase	title	url	platform	score	genre	ed
0	0	Amazing	LittleBigPlanet PS Vita	/games/littlebigplanet-vita/vita-98907	PlayStation Vita	9.0	Platformer	
1	1	Amazing	LittleBigPlanet PS Vita -- Marvel Super Hero E...	/games/littlebigplanet-ps-vita-marvel-super-he...	PlayStation Vita	9.0	Platformer	
2	2	Great	Splice: Tree of Life	/games/splice/ipad-141070	iPad	8.5	Puzzle	
3	3	Great	NHL 13	/games/nhl-13/xbox-360-128182	Xbox 360	8.5	Sports	
4	4	Great	NHL 13	/games/nhl-13/ps3-128181	PlayStation 3	8.5	Sports	

Select rows from position 5 onwards, and columns from position 5 onwards.

```
In [ ]: reviews.iloc[5:,5:]
```

Select the first column, and all of the rows for the column

```
In [ ]: reviews.iloc[:,:1]
```

the 10th row, and all of the columns for that row.

```
In [ ]: reviews.iloc[10:,:]
```

First column is not useful. So remove it

In [21]: reviews=reviews.iloc[:,1:], reviews.head()

Out[21]:

	score_phrase	title	url	platform	score	genre	editors_choice
0	Amazing	LittleBigPlanet PS Vita	/games/littlebigplanet-vita/vita-98907	PlayStation Vita	9.0	Platformer	Y
1	Amazing	LittleBigPlanet PS Vita -- Marvel Super Hero E...	/games/littlebigplanet-ps-vita-marvel-super-he...	PlayStation Vita	9.0	Platformer	Y
2	Great	Splice: Tree of Life	/games/splice/ipad-141070	iPad	8.5	Puzzle	N
3	Great	NHL 13	/games/nhl-13/xbox-360-128182	Xbox 360	8.5	Sports	N
4	Great	NHL 13	/games/nhl-13/ps3-128181	PlayStation 3	8.5	Sports	N

Selection using Row and Column Labels

We have already created students dataframe as below. Let us access name column with loc()

In [22]:

```
students = [['DS01', 'Rex', '1msc'], ['DS02', 'peter', '2msc'], ['CS01', 'ann'
, '3bsc']]
df_stud = pd.DataFrame(students, columns=['rollno', 'name', 'class'])  # row i
ndex automatically generated
```

In [23]:

```
df_stud
```

Out[23]:

	rollno	name	class
0	DS01	Rex	1msc
1	DS02	peter	2msc
2	CS01	ann	3bsc

Print all names using loc

In [24]:

```
df_stud.loc[:,"name"]
```

Out[24]:

```
0      Rex
1    peter
2      ann
Name: name, dtype: object
```

Let us come back to our reviews. Display the first five rows of reviews using the loc method

```
In [25]: reviews.loc[0:5, :].
```

Out[25]:

	score_phrase	title	url	platform	score	genre	editors_choice
0	Amazing	LittleBigPlanet PS Vita	/games/littlebigplanet-vita/vita-98907	PlayStation Vita	9.0	Platformer	Y
1	Amazing	LittleBigPlanet PS Vita -- Marvel Super Hero E...	/games/littlebigplanet-ps-vita-marvel-super-he...	PlayStation Vita	9.0	Platformer	Y
2	Great	Splice: Tree of Life	/games/splice/ipad-141070	iPad	8.5	Puzzle	N
3	Great	NHL 13	/games/nhl-13/xbox-360-128182	Xbox 360	8.5	Sports	N
4	Great	NHL 13	/games/nhl-13/ps3-128181	PlayStation 3	8.5	Sports	N
5	Good	Total War Battles: Shogun	/games/total-war-battles-shogun/mac-142565	Macintosh	7.0	Strategy	N

Select score_phrase column using loc and print head

```
In [26]: reviews.loc[:, 'score-phrase'].head()

Out[26]: 0     Amazing
         1     Amazing
         2       Great
         3       Great
         4       Great
         Name: score_phrase, dtype: object
```

Print top 10 values of column label "score_phrase"

```
In [27]: reviews['score_phrase'].head(10)

Out[27]: 0    Amazing
         1    Amazing
         2      Great
         3      Great
         4      Great
         5       Good
         6      Awful
         7    Amazing
         8      Awful
         9       Good
         Name: score_phrase, dtype: object
```

Select from reviews of rows from 5 to 15

```
In [28]: some_reviews = reviews.iloc[5:15,]. some_reviews.head()
```

print top 5 rows from some_reviews

```
In [29]: some_reviews.head()

Out[29]:
```

	score_phrase	title	url	platform	score	genre	editors_choice	release_yea
5	Good	Total War Battles: Shogun	/games/total-war-battles-shogun/mac-142565	Macintosh	7.0	Strategy	N	201
6	Awful	Double Dragon: Neon	/games/double-dragon-neon/xbox-360-131320	Xbox 360	3.0	Fighting	N	201
7	Amazing	Guild Wars 2	/games/guild-wars-2/pc-896298	PC	9.0	RPG	Y	201
8	Awful	Double Dragon: Neon	/games/double-dragon-neon/ps3-131321	PlayStation 3	3.0	Fighting	N	201
9	Good	Total War Battles: Shogun	/games/total-war-battles-shogun/pc-142564	PC	7.0	Strategy	N	201

Select scores of first 3 rows some_reviews

```
In [30]: some_reviews.loc[5:7,'score']
Out[30]: 5    7.0
         6    3.0
         7    9.0
         Name: score, dtype: float64
```

Select "score", "genre", and "release_year" columns from reviews dataframe and print head

```
In [31]: reviews[["score", "genre", "release_year"]].head()
Out[31]:
```

	score	genre	release_year
0	9.0	Platformer	2012
1	9.0	Platformer	2012
2	8.5	Puzzle	2012
3	8.5	Sports	2012
4	8.5	Sports	2012

What is the datatype of "score" column?

```
In [32]: type(reviews["score"])
Out[32]: pandas.core.series.Series
```

Aggregate Columns

Find average value of score column in reviews dataframe

```
In [33]: reviews['score'].mean()
Out[33]: 6.950459060402666
```

Find average value of all numeric columns

```
In [34]: reviews.mean()
Out[34]: score               6.950459
         release_year     2006.515329
         release_month       7.138470
         release_day        15.603866
         dtype: float64
```

Find average value for each numeric column

```
In [35]: reviews.mean(axis=0)

Out[35]: score                6.950459
         release_year      2006.515329
         release_month        7.138470
         release_day         15.603866
         dtype: float64
```

Find average value for each row containing numeric values and print head

```
In [36]: reviews.mean(axis=1).head()

Out[36]: 0    510.500
         1    510.500
         2    510.375
         3    510.125
         4    510.125
         dtype: float64
```

Find lowest, highest, median, standard deviation of score column of reviews dataframe

show median of "score" column of reviews dataframe

```
In [37]: reviews['Score'].median()

Out[37]: 7.3
```

show minimum of "score" column of reviews dataframe

```
In [38]: reviews['Score'].min()

Out[38]: 0.5
```

show maximum of "score" column of reviews dataframe

```
In [39]: reviews['Score'].max()

Out[39]: 10.0
```

show standard deviation of "score" column of reviews dataframe

```
In [40]: reviews['score'].std()
Out[40]: 1.7117358608045874
```

How many non-null values in "score" column of reviews dataframe?

```
In [41]: reviews['score'].count()
Out[41]: 18625
```

Show the summary of reviews dataframe

```
In [42]: reviews.describe()
Out[42]:
```

```
              score  release_year  release_month   release_day
count  18625.000000  18625.000000    18625.00000  18625.000000
mean       6.950459   2006.515329        7.13847     15.603866
std        1.711736      4.587529        3.47671      8.690128
min        0.500000   1970.000000        1.00000      1.000000
25%        6.000000   2003.000000        4.00000      8.000000
50%        7.300000   2007.000000        8.00000     16.000000
75%        8.200000   2010.000000       10.00000     23.000000
max       10.000000   2016.000000       12.00000     31.000000
```

Check if review score has any correlation with other columns of reviews

```
In [43]: reviews.corr()
Out[43]:
```

```
                  score  release_year  release_month  release_day
        score  1.000000      0.062716       0.007632     0.020079
 release_year  0.062716      1.000000      -0.115515     0.016867
release_month  0.007632     -0.115515       1.000000    -0.067964
  release_day  0.020079      0.016867      -0.067964     1.000000
```

Review score has no correlation with other features. So, release timing doesn't linearly relate to review score

Math Operations on DF columns

Divide the values of "score" column in reviews dataframe by 2. There will be too many values, so just print head

```
In [44]: (reviews['score']/2).head()

Out[44]: 0    4.50
         1    4.50
         2    4.25
         3    4.25
         4    4.25
         Name: score, dtype: float64
```

Boolean Indexing in Pandas

Select all video games whose review score > 7, call it score_filter

```
In [45]: score_filter = reviews["score"] > 7
```

Print head of score_filter

```
In [46]: score-filter-head()

Out[46]: 0    True
         1    True
         2    True
         3    True
         4    True
         Name: score, dtype: bool
```

Select all rows for score_filter column and print its head

```
In [47]: filtered_reviews = reviews [[illegible] ["] & A [score_filter].
Out[47]: filtered_reviews.head()
```

	score_phrase	title	url	platform	score	genre	editors_choice
0	Amazing	LittleBigPlanet PS Vita	/games/littlebigplanet-vita/vita-98907	PlayStation Vita	9.0	Platformer	Y
1	Amazing	LittleBigPlanet PS Vita -- Marvel Super Hero E...	/games/littlebigplanet-ps-vita-marvel-super-he...	PlayStation Vita	9.0	Platformer	Y
2	Great	Splice: Tree of Life	/games/splice/ipad-141070	iPad	8.5	Puzzle	N
3	Great	NHL 13	/games/nhl-13/xbox-360-128182	Xbox 360	8.5	Sports	N
4	Great	NHL 13	/games/nhl-13/ps3-128181	PlayStation 3	8.5	Sports	N

Show the size of filtered_reviews

```
In [48]: filtered_reviews.shape
Out[48]: (9800, 10)
```

Show top 10 "title" from filtered_reviews

```
In [49]: filtered_reviews["title"].head(10)
Out[49]: 0                                       LittleBigPlanet PS Vita
         1     LittleBigPlanet PS Vita -- Marvel Super Hero E...
         2                                          Splice: Tree of Life
         3                                                        NHL 13
         4                                                        NHL 13
         7                                                  Guild Wars 2
         10                                     Tekken Tag Tournament 2
         11                                     Tekken Tag Tournament 2
         13                                            Mark of the Ninja
         14                                            Mark of the Ninja
         Name: title, dtype: object
```

Find games released for the Xbox One platform that have a score of more than 7

First create a filter, called xbox_one_filter for the conditions

```
In [50]: xbox_one_filter = (reviews["Score"]>7) & (reviews["platform"]=="Xbox One")
```

Select those rows from reviews of xbox_one_filter and print head

```
In [51]: filtered_reviews2 = reviews[xbox_one_filter]

         filtered_reviews2.head()

         #show top 5 rows of filtered_reviews2
```

Out[51]:

	score_phrase	title	url	platform	score	genre	editors_choice	releas
17137	Amazing	Gone Home	/games/gone-home/xbox-one-20014361	Xbox One	9.5	Simulation	Y	
17197	Amazing	Rayman Legends	/games/rayman-legends/xbox-one-20008449	Xbox One	9.5	Platformer	Y	
17295	Amazing	LEGO Marvel Super Heroes	/games/lego-marvel-super-heroes/xbox-one-20000826	Xbox One	9.0	Action	Y	
17313	Great	Dead Rising 3	/games/dead-rising-3/xbox-one-124306	Xbox One	8.3	Action	N	
17317	Great	Killer Instinct	/games/killer-instinct-2013/xbox-one-20000538	Xbox One	8.4	Fighting	N	

```
In [52]: # What is the size of filtered_reviews2 filtered_reviews2.shape

Out[52]: (140, 10)
```

Select all video games which are 'Action' genre

```
In [53]: action_reviews = reviews[reviews['genre']=='Action']
In [54]: action_reviews.head()
Out[54]:
```

	score_phrase	title	url	platform	score	genre	editors_choice	release_ye
17	Great	Avengers Initiative	/games/avengers-initiative/iphone-141579	iPhone	8.0	Action	N	20
34	Good	War of the Roses	/games/war-of-the-roses-140577/pc-115849	PC	7.3	Action	N	20
45	Amazing	Bad Piggies	/games/bad-piggies/iphone-141455	iPhone	9.2	Action	Y	20
49	Okay	Demon's Score	/games/demons-score/iphone-118050	iPhone	6.9	Action	N	20
69	Great	Hotline Miami	/games/hotline-miami/pc-139657	PC	8.8	Action	Y	20

```
In [55]: What is the size of action_reviews? action.reviews. shape
Out[55]: (3797, 10)
```

Plot Review Ratings of two Play Stations and Compare Which one has more ratings?

Now that we know how to filter, we can create plots to observe the review distribution for the Xbox One vs the review distribution for the PlayStation 4. This will help us figure out which console has better games.

We can do this via a histogram, which will plot the frequencies for different score ranges.

Plot Histogram for the frequencies of different score ranges of Xbox One platform

```
In [56]: # Import plotting libraries
```

Import matplotlib.pyplot as plt.

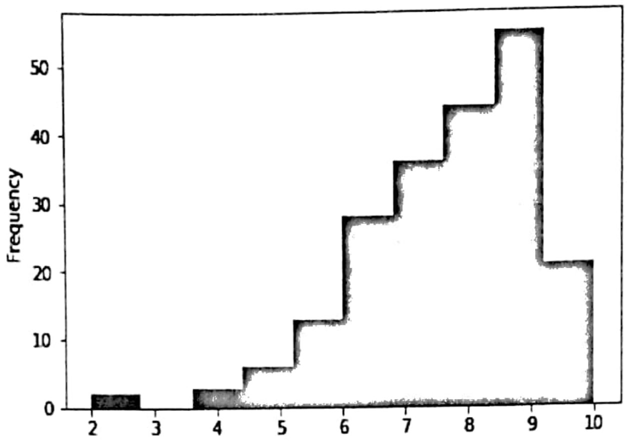
%. mat plot Inline

reviews[reviews["platform"]=="Xbox one"]["score"].
plot(kind="hist")

reviews[reviews["platform"]=="xbox one"]["score"].plot(kind="hist")

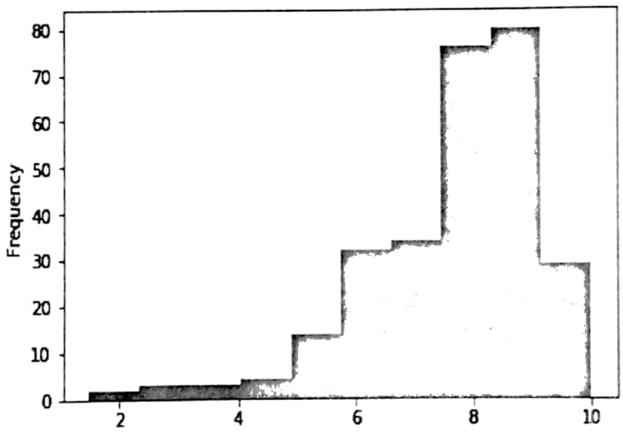
```
In [57]:  # Plot the following histogram of score values for Xbox One platform

Out[57]:  <matplotlib.axes._subplots.AxesSubplot at 0x25161f78c88>
```



Plot Histogram for Frequencies of the scores of Play Station4 platform

```
In [58]:  reviews[reviews["platform"]=="play station 4"]["score"].plot
          (kind="hist")
Out[58]:  <matplotlib.axes._subplots.AxesSubplot at 0x2516407e7b8>
```



Therefore, it appears from our histograms that the PlayStation4 has many more highly rated games than the Xbox One.

```
In [ ]:
```