Machine Learning Diploma

Session4: Pandas

AMIT

Agenda

- ➔ Pandas
- ➔ Data Types
- ➔ Mini project

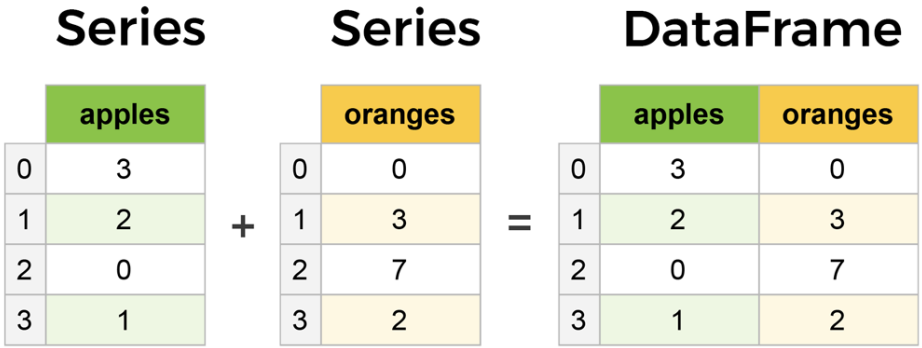
AMIT

1. Pandas

Pandas:

➔ Pandas provides data structures and functionality to quickly manipulate and analyze data.

➔ The key to understanding Pandas for machine learning is understanding the Series and DataFrame data structures.



Pandas Series:

➔ A series is a one dimensional array where the rows and columns can be labeled.

➔ You can access the data in a series like a NumPy array and like a dictionary.

```
import numpy
import pandas
myarray = numpy.array([1, 2, 3])
rownames = ['a', 'b', 'c']
myseries = pandas.Series(myarray, index=rownames)
print(myseries)
print(myseries[0])
print(myseries['a'])
```

```
a    1
b    2
c    3
dtype: int32
1
1
```


AMIT

Pandas DataFrame:

➔ A data frame is a multi-dimensional array where the rows and the columns can be labeled.

```
import numpy
import pandas
myarray = numpy.array([[1, 2, 3], [4, 5, 6]])
rownames = ['a', 'b']
colnames = ['one', 'two', 'three']
mydataframe = pandas.DataFrame(myarray, index=rownames, columns=colnames)
print(mydataframe)
print("method 1:")
print(mydataframe['one'])
print("method 2:")
print(mydataframe.one)
```

```
   one  two  three
a    1    2      3
b    4    5      6
method 1:
a    1
b    4
Name: one, dtype: int32
method 2:
a    1
b    4
Name: one, dtype: int32
```


AMIT

Data Loading:

- ➔ The most common format for machine learning data is CSV files.
- ➔ There are a number of considerations when loading your machine learning data from CSV files.
  - File Headers. Does your data have a file header?, you may need to name your attributes manually.
  - Delimiter. The standard delimiter that separates values in fields is the comma (,) Your file could use a different delimiter like tab or white space in which case you must specify it explicitly.
  - Quotes. Sometimes field values can have spaces. In these CSV files the values are often quoted.

Data Loading:

- ➔ We will use New York City Airbnb Open Data Database for practicing.
- ➔ Download the CSV file to your folder with your scripts.
- ➔ You can load the data into your script using:
  - Python Standard Library.
  - NumPy
  - Pandas

AMIT

Data Loading using pandas:

- ➔ You can load your CSV data using Pandas and the pandas.read csv() function.
- ➔ The function returns a pandas.DataFrame that you can immediately start summarizing and plotting.

```
import pandas as pd
labels = ['id', 'name', 'host_id', 'host_name', 'neighbourhood_group',
          'neighbourhood', 'latitude', 'longitude', 'room_type', 'price']

df                  = pd.read_csv('AB_NYC_2019.csv')
df_sep              = pd.read_csv('AB_NYC_2019.csv', sep=',')
df_head_2nd_row     = pd.read_csv('AB_NYC_2019.csv', header=2)
df_no_header        = pd.read_csv('AB_NYC_2019.csv', header=None)
df_assign_col_names = pd.read_csv('AB_NYC_2019.csv', names= labels)
df_assign_index     = pd.read_csv('AB_NYC_2019.csv', index_col = 'name')
```


AMIT

Data Exploratory:

➔ First step is to see how your data is formulated. You can view the first few rows of a dataframe using df.head() method. It can take the number of rows you want to see. df.head(7) will retrieve the first 7 rows.

➔ df.tail() is the same as df.head() instead it movies back the last few rows.

In [5]:
```
df = pd.read_csv('AB_NYC_2019.csv')
df.head(3)
```

Out[5]:

	id	name	host_id	host_name	neighbourhood_group	neighbourhood	latitude	longitude	room_type	price	minimum_nights	number_of_r
0	2539	Clean & quiet apt home by the park	2787	John	Brooklyn	Kensington	40.64749	-73.97237	Private room	149	1	
1	2595	Skylit Midtown Castle	2845	Jennifer	Manhattan	Midtown	40.75362	-73.98377	Entire home/apt	225	1	
2	3647	THE VILLAGE OF HARLEM....NEW YORK !	4632	Elisabeth	Manhattan	Harlem	40.80902	-73.94190	Private room	150	3	

Data Exploratory:

➔ df.info() gives a summary info of the data, like number if non-null values. You can find that some columns have lower number of non-null count; meaning they have null values.

```
In [7]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 48895 entries, 0 to 48894
Data columns (total 16 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   id                              48895 non-null  int64
 1   name                            48879 non-null  object
 2   host_id                         48895 non-null  int64
 3   host_name                       48874 non-null  object
 4   neighbourhood_group             48895 non-null  object
 5   neighbourhood                   48895 non-null  object
 6   latitude                        48895 non-null  float64
 7   longitude                       48895 non-null  float64
 8   room_type                       48895 non-null  object
 9   price                           48895 non-null  int64
 10  minimum_nights                  48895 non-null  int64
 11  number_of_reviews               48895 non-null  int64
 12  last_review                     38843 non-null  object
 13  reviews_per_month               38843 non-null  float64
 14  calculated_host_listings_count  48895 non-null  int64
 15  availability_365                48895 non-null  int64
dtypes: float64(3), int64(7), object(6)
memory usage: 6.0+ MB
```


AMIT

Data Exploratory:

➔ df.describe() give back summary statistics.

```
In [8]: df.describe()
```

Out[8]:

	id	host_id	latitude	longitude	price	minimum_nights	number_of_reviews	reviews_per_month	calculated_l
count	4.889500e+04	4.889500e+04	48895.000000	48895.000000	48895.000000	48895.000000	48895.000000	38843.000000	
mean	1.901714e+07	6.762001e+07	40.728949	-73.952170	152.720687	7.029962	23.274466	1.373221	
std	1.098311e+07	7.861097e+07	0.054530	0.046157	240.154170	20.510550	44.550582	1.680442	
min	2.539000e+03	2.438000e+03	40.499790	-74.244420	0.000000	1.000000	0.000000	0.010000	
25%	9.471945e+06	7.822033e+06	40.690100	-73.983070	69.000000	1.000000	1.000000	0.190000	
50%	1.967728e+07	3.079382e+07	40.723070	-73.955680	106.000000	3.000000	5.000000	0.720000	
75%	2.915218e+07	1.074344e+08	40.763115	-73.936275	175.000000	5.000000	24.000000	2.020000	
max	3.648724e+07	2.743213e+08	40.913060	-73.712990	10000.000000	1250.000000	629.000000	58.500000	

AMIT

Data Exploratory:

➔ df.columns give back column names/features.

```
In [12]: df.columns

Out[12]: Index(['id', 'name', 'host_id', 'host_name', 'neighbourhood_group',
                'neighbourhood', 'latitude', 'longitude', 'room_type', 'price',
                'minimum_nights', 'number_of_reviews', 'last_review',
                'reviews_per_month', 'calculated_host_listings_count',
                'availability_365'],
               dtype='object')
```


AMIT

Data Exploratory:

➔ df.dtypes give back column data types.

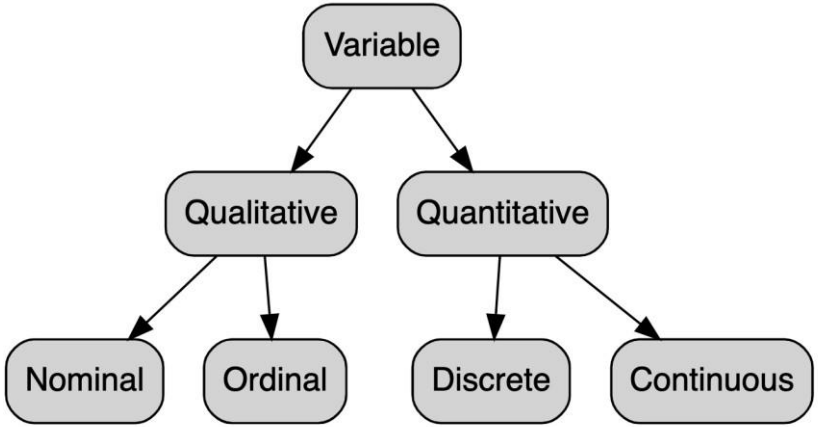
```
In [13]: df.dtypes

Out[13]: id                                  int64
         name                               object
         host_id                             int64
         host_name                          object
         neighbourhood_group                object
         neighbourhood                      object
         latitude                          float64
         longitude                         float64
         room_type                          object
         price                               int64
         minimum_nights                      int64
         number_of_reviews                   int64
         last_review                        object
         reviews_per_month                 float64
         calculated_host_listings_count      int64
         availability_365                    int64
         dtype: object
```

2. Data Types

Data Types:

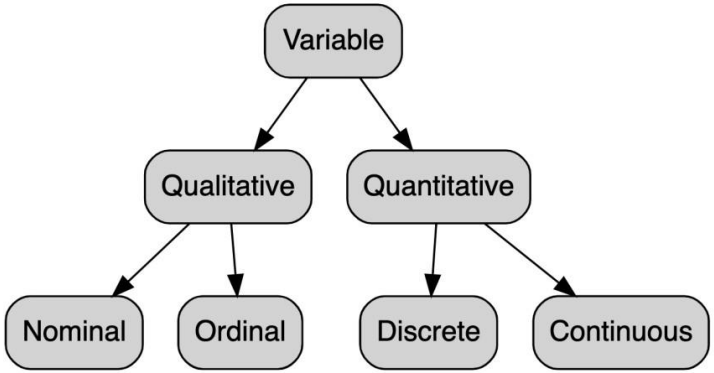
➔ Quantitative is numerical data like number of dogs
➔ Qualitative is text data like the bread of dogs.



Data Types:

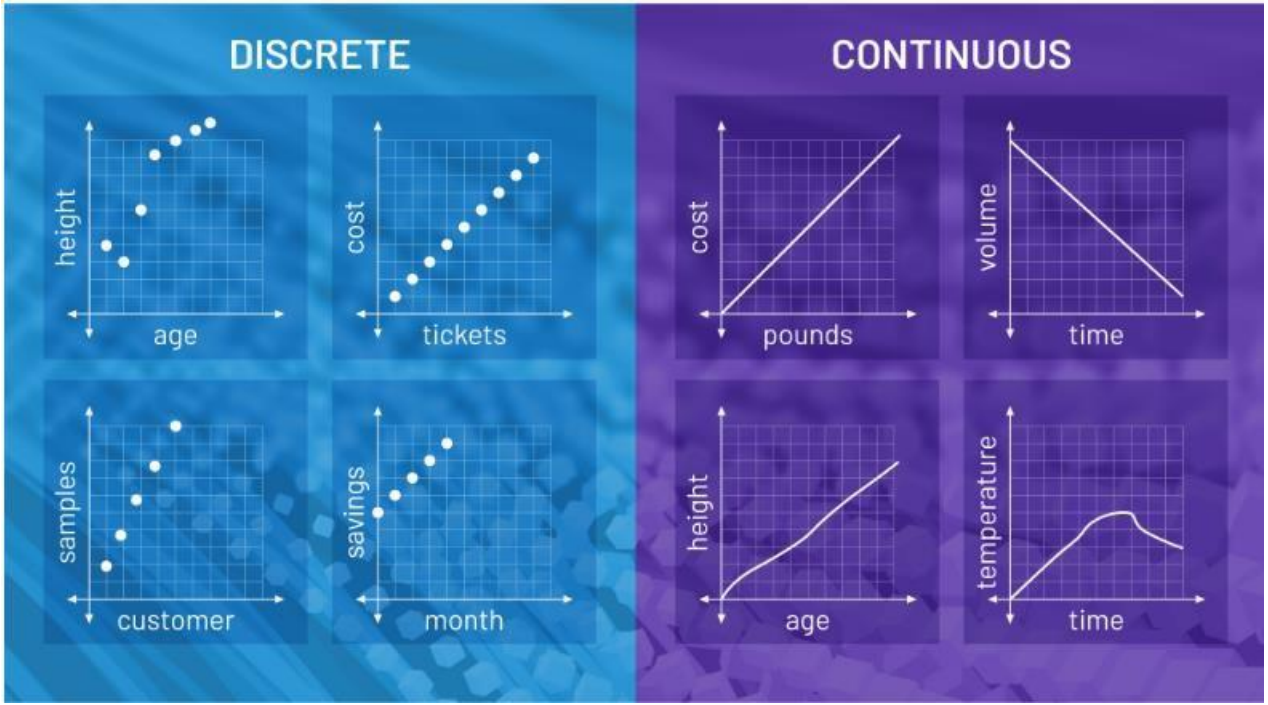
➔ Discrete data is a numerical type of data that includes whole, concrete numbers with specific and fixed data values determined by counting. Like number of dogs.

➔ Continuous data includes complex numbers and varying data values that are measured over a specific time interval. Like temperature readings.



AMIT

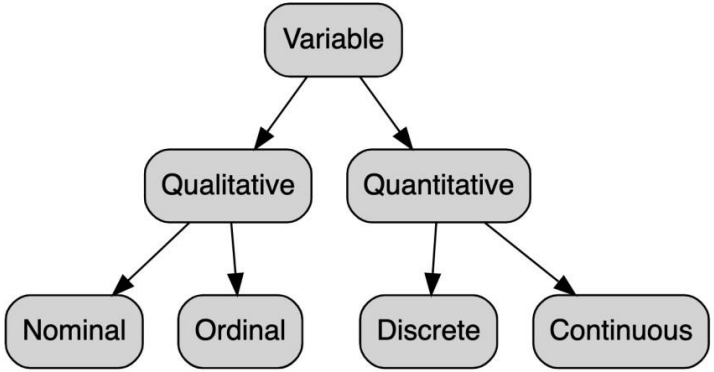
Data Types:


DISCRETE
CONTINUOUS
height
age
cost
tickets
cost
pounds
volume
time
samples
customer
savings
month
height
age
temperature
time

Data Types:

➔ Nominal data simply names something without assigning it to an order in relation to other numbered objects or pieces of data. Like colors.

➔ Ordinal data, unlike nominal data, involves some order. Like grades.



3. Mini project

Any Questions?

THANK YOU!