Pandas

W2 - Practical 1

Learning Objectives

1. Reading CSV file into Data Frame
2. Working with columns
3. Working with rows
4. Slicing

Task 1: Read the "online_store_customer_data.csv" file into a data frame named as fulldata_df.

- Display the first 10 rows of data frame using function head().
- Display last 10 rows of data frame using function tail().

Task2: Select column "Age" from the data frame and as a data frame named as age_df.

- Select columns, "Age", "Gender", "Marital_status", into a new data frame as personal_df.
- Change the order of columns in personal_df to "Gender, Marital_status, age". Use rename function with inplace=True argument to rename the columns.
- Select columns 'Employees_status', 'Payment_method', 'Referal', 'Amount_spent' from fulldata_df and delete columns 'Employees_status', 'Payment_method' from it.

Task3: Working with rows.

- Show all the "Single" employees in the data frame.
- Show all the employees from State "Connecticut".
- Show all the employees where "Amount_spent" is greater than $1000.
- Show all the employees older than 30 years.
- Show only the "Age, Marital_status, and Gender" of the employees spending more than $1500.
- Show the States with the highest to lowest amount spent.

  Hint:
  - First select only state_name and amount_spent columns.
  - Sort the amount_spent columns using sort_values().
  - i.e. your df.sort_values("column name whose values you want to sort")

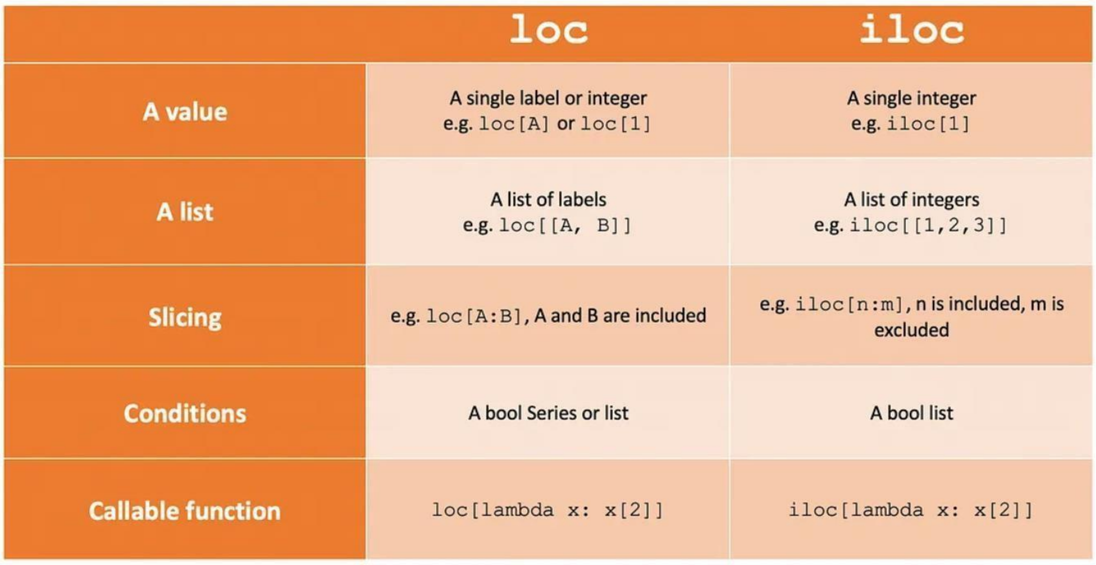
Slicing

When it comes to select data on a DataFrame, Pandas loc and iloc are two top favourites. They are quick, fast, easy to read, and sometimes interchangeable.

The main distinction between loc and iloc is:

loc is label-based, which means that you have to specify rows and columns based on their row and column labels.

iloc is integer position-based, so you have to specify rows and columns by their integer position values (0-based integer position).



	loc	iloc
A value	A single label or integer e.g. `loc[A]` or `loc[1]`	A single integer e.g. `iloc[1]`
A list	A list of labels e.g. `loc[[A, B]]`	A list of integers e.g. `iloc[[1,2,3]]`
Slicing	e.g. `loc[A:B]`, A and B are included	e.g. `iloc[n:m]`, n is included, m is excluded
Conditions	A bool Series or list	A bool list
Callable function	`loc[lambda x: x[2]]`	`iloc[lambda x: x[2]]`

Both loc and iloc are used for data slicing. Loc can use name labels as well as integer labels. Here is an example of selecting rows and column from a dataframe using loc and iloc.

Selecting first 10 rows and columns (“age, gender”)

Fulldata_df.loc[0:10,[‘Age’, ‘Gender’]] #First index selects rows, second index columns.

Fulldata_df.iloc[0:10 ,[3,2]] #You can use index of columns instead of their names.

- Selects rows from 10 to 30 and all columns.
- Select rows from 10 to 30 and only for the columns - Age, Gender, Amount_spent.
- Select all rows except first 5 rows and columns ['State_names', 'Segment', 'Employees_status', 'Payment_method']