Analytic Vidhya

Introduction to Machine learning

File Handling in Python.

r → read only r+ → read and write

w → write only w+ → write and read.

a → Append only a+ → Append & read.

Points [Reading .txt files].

* ~~In order to read~~ If we open a file (.txt) and we read it we have to close the file, ~~because~~ to read it again, we we read the file once the pointer points to the last location, and if we try to read the file again we will ~~not~~ be shown a blank because the pointer reads the last line.

* seek()
  ↳ It points the pointer to the given location. and it will start reading the file from that location.

  seek(0)
  ↳ pointer will start from 0th position.

* It is a good practice to close() the file after opening it.

* readlines() → Returns the lines of the file stored.

* with open('dataset/file_1.txt', 'r') as f:
      complete_text = f.read()

  with ↓
  using with statement, python will automatically close the file after reading it.

Difference b/w loc & iloc.

df.

index	id	gender	Marks
1	A103	M	34
2	A104	M	35
3	A105	M	36
4	A106	F	37
5	A107	F	38
6	A108	M	39

g-df.

index	Label	id	gender
0	3	A105	M
1	2	A104	M
2	1	A103	M
3	4	A106	F
4	6	A108	M
5		A107	F

loc.

df. loc [2:5]

2	A104	M	35
3	A105	M	36
4	A106	F	37.
5	A107	F	38.

→ In these function all the numbers will be considered b/w 2 & 5 (including)

→ Here, index acts as labels.

iloc

df. iloc [2:5]

2	A104	M	35.
3	A105	M	36
4	A106	F	38.

→ In these function it will take the index of the dataframe.

* If we jumble the dataframe and use the iloc ~~f~~ func.

~~i log iloc()~~

~~iloc g-df[2:5]~~

iloc g-df. iloc [2:6]

2 → 1	A103	M
3 → 4	A106	F

Explained further

~~i_df.iloc[2:4]~~

* iloc is manipulate the df. based on the index id.

i_df

id	gender
A105	M
A104	M
A103	M
A106	F
A108	F

→ i_df.iloc[2:4]

A103	M
A106	F

→ i_df.loc[2:4]

ERROR

→ i_df.loc['A104':'A106']

A104	M
A105	F
A106	M

some important code.

* # fill the null values of a column by the most frequent values.

df. col_name. fillna ("xyz", inplace = True)

↓ the most frequent value.

* How do find out the most frequent values in the categorical column.

→ df. col_name. mode. → gives the biggest number of counts of the particular col.

* Suppose you want to change the ~~data~~ categorical data in a particular column.

→ We can use mapping in these case —

~~Low Fat : LF~~

for eg — Low fat → LF.
Regular → ~~Reg~~ R.
LF → LF.
reg → R.

```
mapping = {
    'LowFat' : 'LF',
    'Regular' : 'R',
    'Reg' : 'R'
    'LF' : 'LF'
}.
```

df. col_name = df. col_name. map (mapping)

Python for Data Science

Reading data files

Subsetting, Modifying data.

Building ML Models.

Preprocessing, Sorting Aggregating data

Visualizing trends & patterns.

Preprocessing, Sorting & Aggregating Data.

id	name	Grade	Marks
1	AB	A	25
2	BC	A	26.
3	CD	B	21
4	DE	B	20.

These table is a sorted table, where sorting is done based on grades

But, we can see BC scored more marks than AB.

∴ To deal with such problems

∴ we sort the grades in ascending order and marks as descending order

= df.sort_values(by=['grades', 'marks'], ascending=[True, False], inplace=True)

Life cycle of a ML Project

1. Problem Definition.
2. Hypothesis Generation.
3. Data Extraction / Collection.
4. Data Exploration & Transformation.
5. Predictive Modeling
6. Model Deployment / Implementation

Problem Definition.

Bad problem statement

want to increase the sales of the company.

Good Problem Statement

want to increase the productivity of sales agent for the

Hypothesis Generation.

Once the problem statement is defined we have to list down all the possible solutions that might solve the problem.

* But, we have to be taken as these problem statement must be free from all the biased information