**Experiment 1**

Aim -

Data preparation using NumPy and Pandas.

Todo -

1. Load data in Pandas.
2. Description of the dataset.
3. Drop columns that aren’t useful.
4. Drop rows with maximum missing values.
5. Take care of missing data.
6. Create dummy variables.
7. Find out outliers (manually).
8. Standardization and normalization of columns.

Dataset -

The dataset contains data about customers' purchases during the Black Friday sale. This dataset was taken from Kaggle. The dataset has 550k rows and 12 columns. The various columns of the dataset are age, marital status, gender, total purchase amount, and many other features.

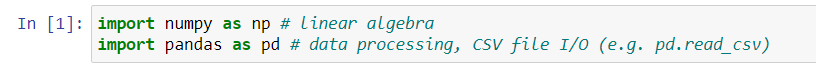
Theory -

pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is a fast, powerful, flexible, and easy-to-use open-source data analysis and manipulation tool.

NumPy is a library for the Python programming language, adds support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. It offers comprehensive mathematical functions, random number generators, linear algebra routines, Fourier transforms, and more.

Results -

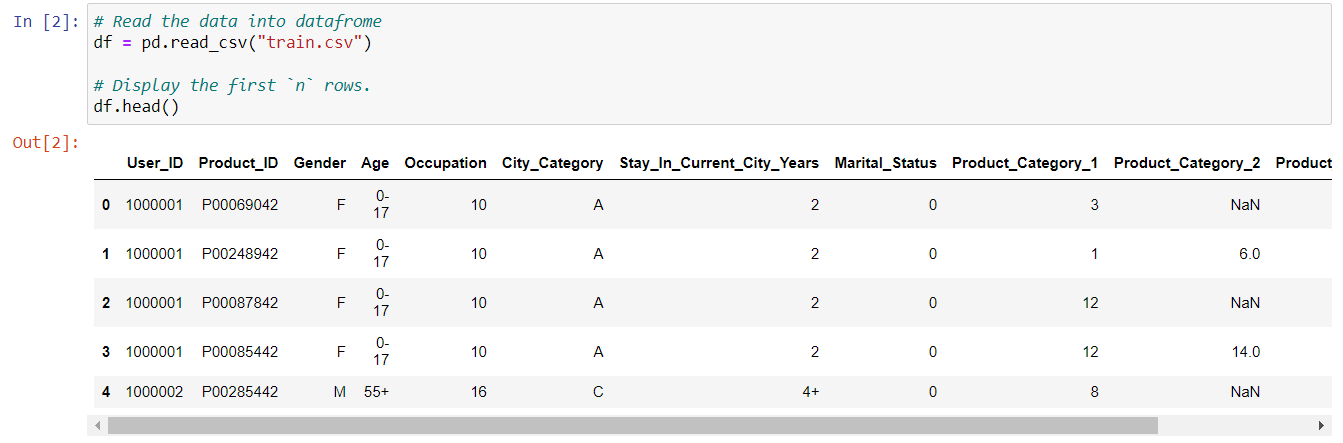
1. Load dataset.



read\_csv():

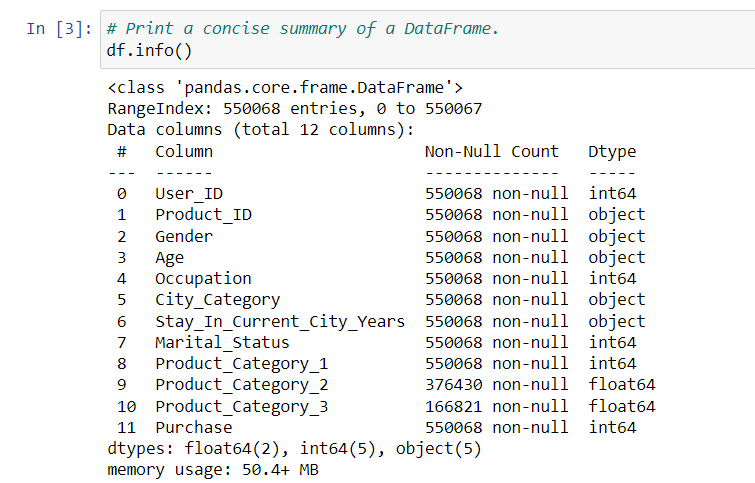
Read a comma-separated values (CSV) file into DataFrame.

head():  
This function returns the first n rows for the object based on position. It is useful for quickly testing if your object has the right type of data in it.



info():

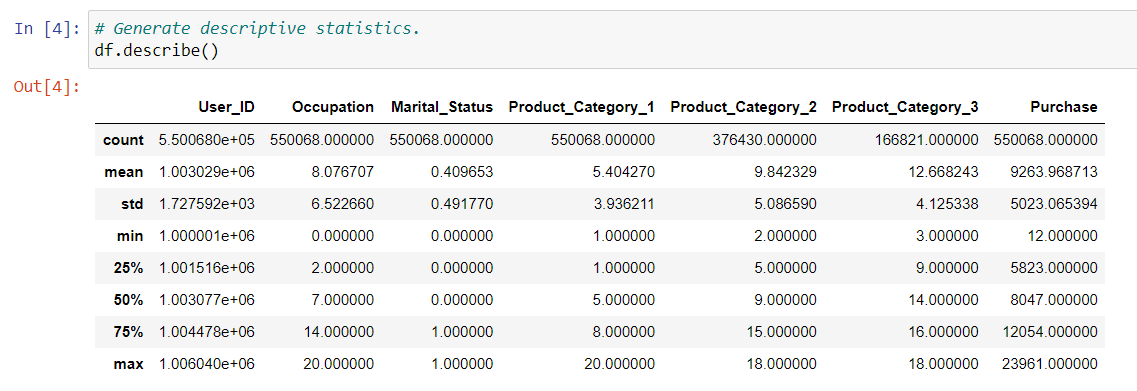
This method prints information about a DataFrame including the index dtype and columns, non-null values and memory usage.



1. Describe dataset.

describe():

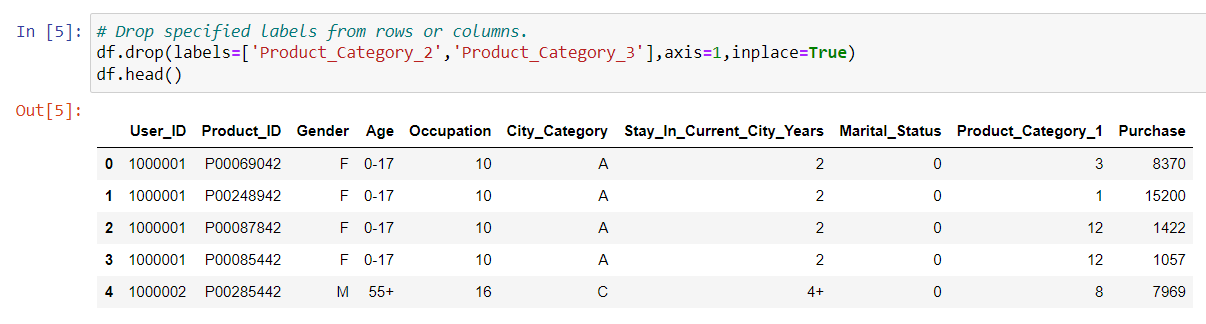
Descriptive statistics include those that summarize the central tendency, dispersion and shape of a dataset’s distribution, excluding NaN values.



1. Drop columns that aren’t useful.

drop():

Remove rows or columns by specifying label names and corresponding axis, or by specifying direct index or column names. When using a multi-index, labels on different levels can be removed by specifying the level.



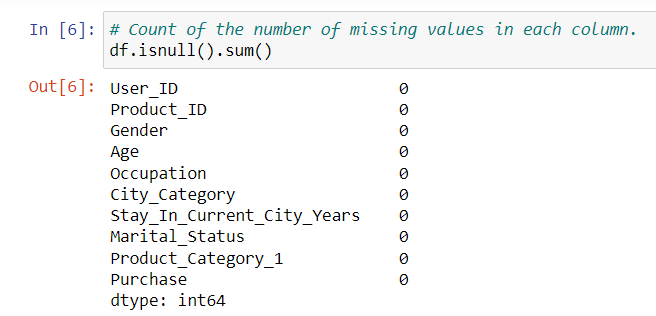
isnull():

Detect missing values.

Return a boolean same-sized object indicating if the values are NA. NA values, such as None or numpy. NaN gets mapped to True values. Everything else gets mapped to False values. Characters such as empty strings '' or numpy.inf are not considered NA values (unless you set pandas.options.mode.use\_inf\_as\_na = True).

sum():

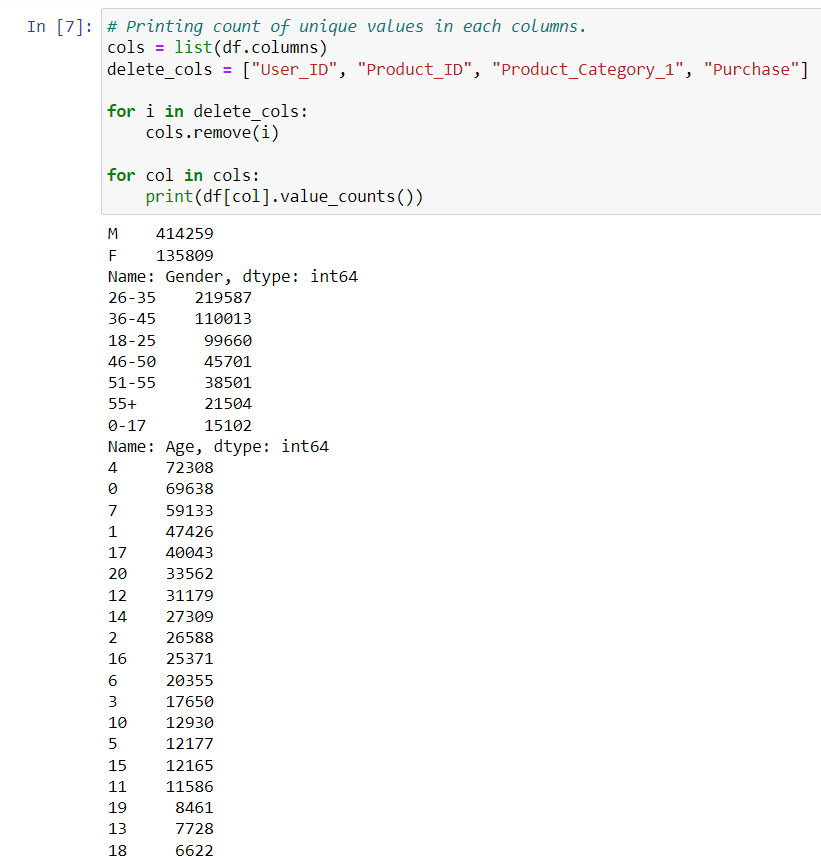
Returns a data frame or Series of the same size containing the cumulative sum.



1. Create dummy variables.

value\_counts():

Return a Series containing counts of unique rows in the DataFrame.



concat():

Concatenate pandas objects along a particular axis.

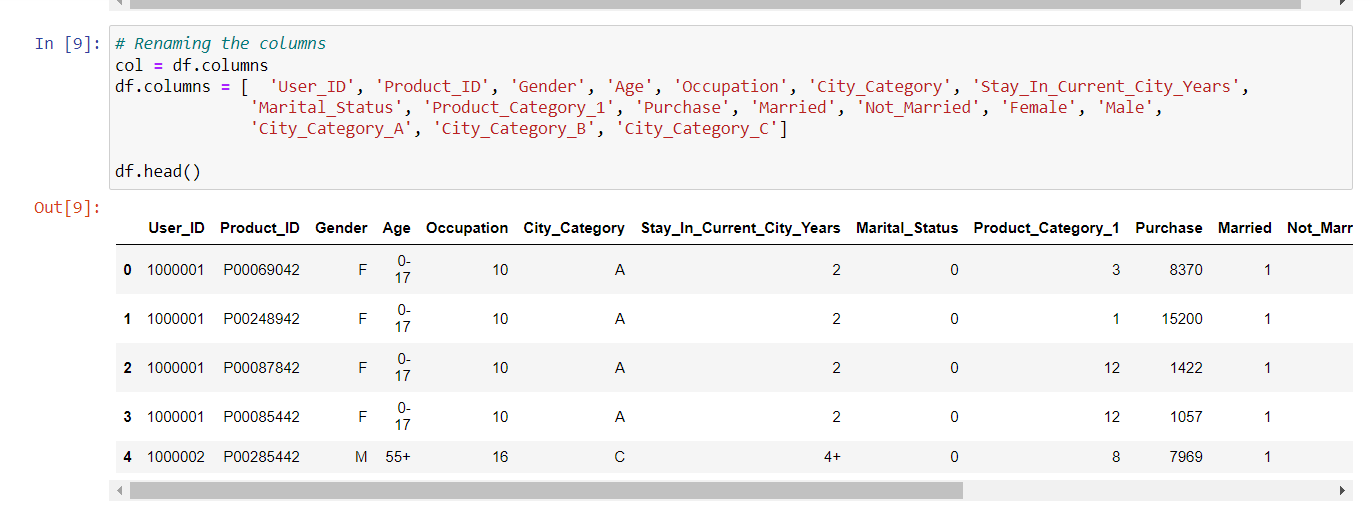
Allows optional set logic along the other axes.

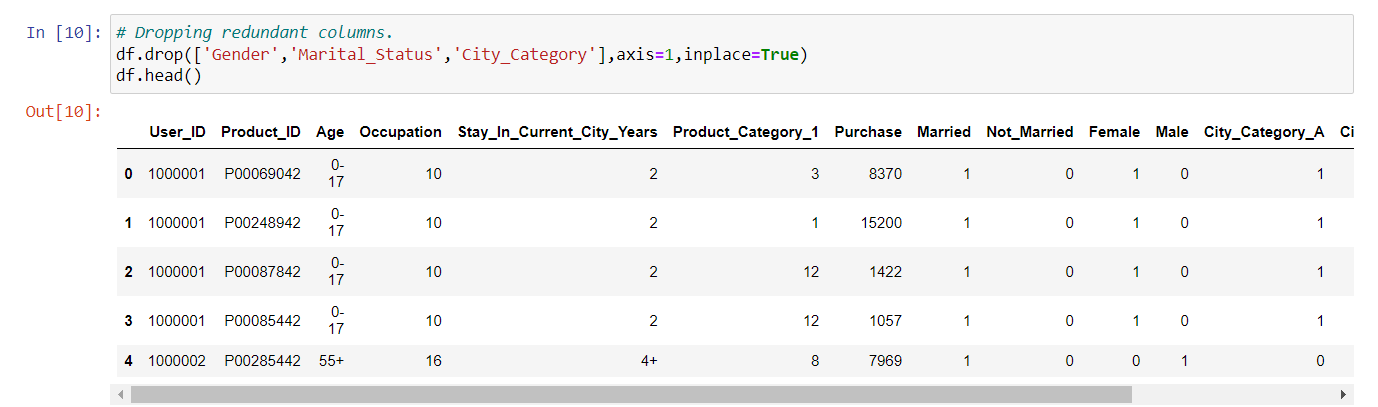
Can also add a layer of hierarchical indexing on the concatenation axis, which may be useful if the labels are the same (or overlapping) on the passed axis number.

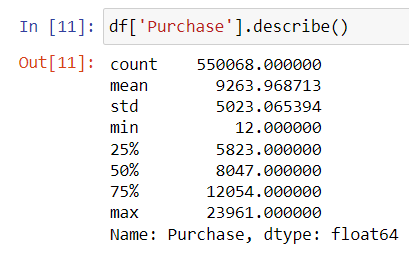
get\_dummies():

Convert categorical variable into dummy/indicator variables.

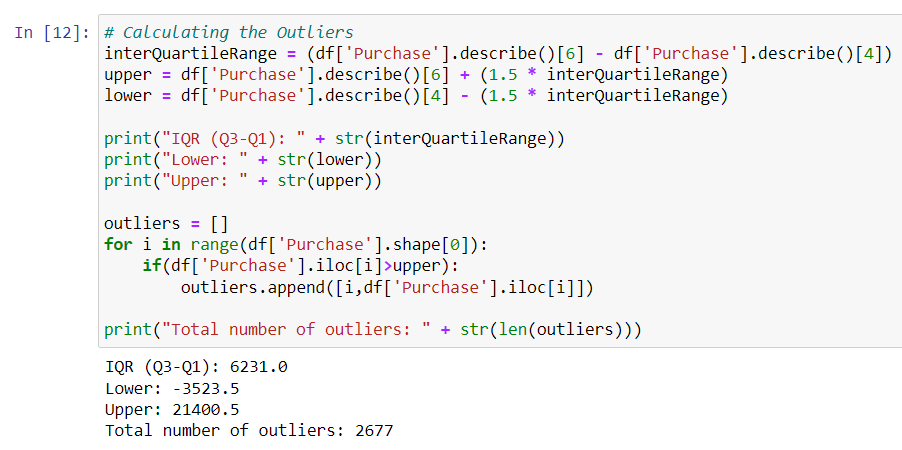




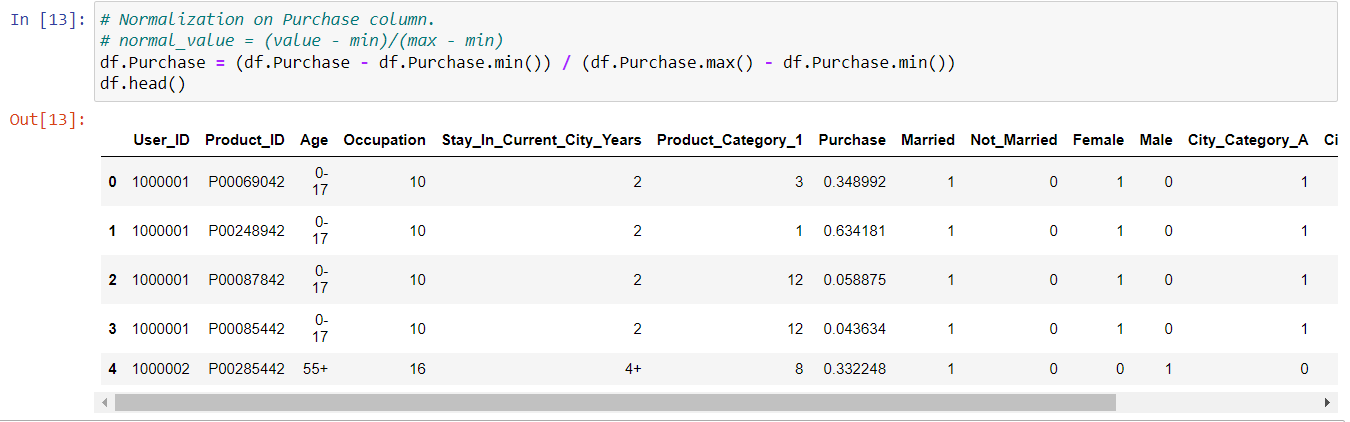


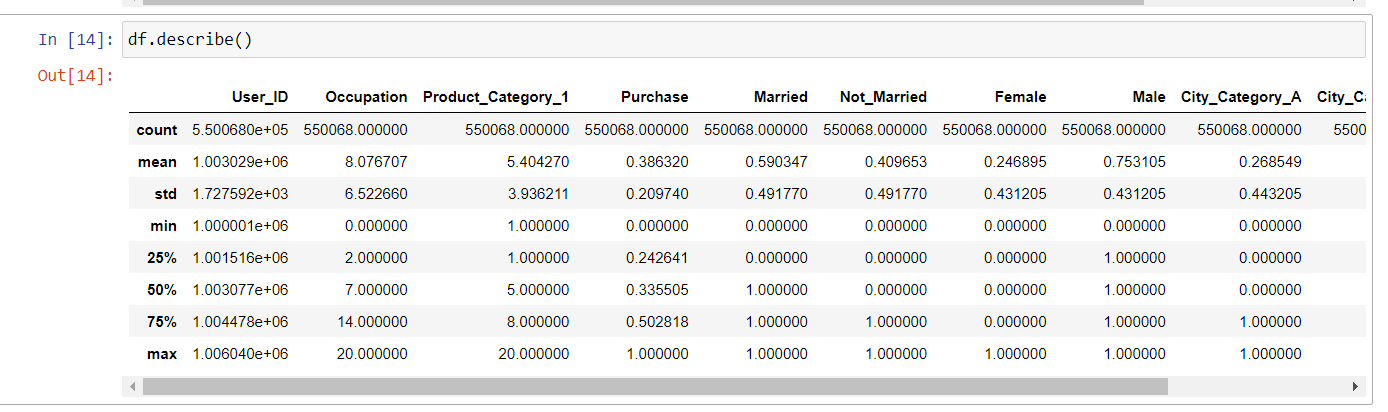


1. Find out outliers (manually).



1. Standardization and normalization of columns.





Conclusion-

We have successfully preprocessed the data. We cleaned the dataset by removing missing values and redundant columns. We also used dummy variables and then normalized the values.