Project 02 : Data Wrangling

Data Wrangling

Data Wrangling is the process of gathering, collecting, and transforming Raw data into another format for better understanding, decision-making, accessing, and analysis in less time. Data Wrangling is also known as Data Munging.

Importance Of Data Wrangling

Data Wrangling is a very important step in a Data science project. The below example will explain its importance:

Books selling Website want to show top-selling books of different domains, according to user preference. For example, if a new user searches for motivational books, then they want to show those motivational books which sell the most or have a high rating, etc.

But on their website, there are plenty of raw data from different users. Here the concept of Data Munging or Data Wrangling is used. As we know Data wrangling is not by the System itself. This process is done by Data Scientists. So, the data Scientist will wrangle data in such a way that they will sort the motivational books that are sold more or have high ratings or user buy this book with these package of Books, etc. On the basis of that, the new user will make a choice. This will explain the importance of Data wrangling.

Data Wrangling in Python

Data Wrangling is a crucial topic for Data Science and Data Analysis. Pandas Framework of Python is used for Data Wrangling. Pandas is an open-source library in Python specifically developed for Data Analysis and Data Science. It is used for processes like data sorting or filtration, Data grouping, etc.
Data wrangling in Python deals with the below functionalities:

1. **Data exploration:** In this process, the data is studied, analyzed, and understood by visualizing representations of data.
2. **Dealing with missing values:** Most of the datasets having a vast amount of data contain missing values of *NaN*, *they are needed to be taken* care of by replacing them with mean, mode, the most frequent value of the column, or simply by dropping the row having a *NaN* value.
3. **Reshaping data:** In this process, data is manipulated according to the requirements, where new data can be added or pre-existing data can be modified.
4. **Filtering data:** Some times datasets are comprised of unwanted rows or columns which are required to be removed or filtered
5. **Other:** After dealing with the raw dataset with the above functionalities we get an efficient dataset as per our requirements and then it can be used for a required purpose like data analyzing, machine learning, data visualization, model training etc.

Data Wrangling Using Merge Operation

Merge operation is used to merge two raw data into the desired format.

Syntax: *pd.merge(data_frame1,data_frame2, on="field ")*
Here the field is the name of the column which is similar in both data-frame.

For example: Suppose that a Teacher has two types of Data, the first type of Data consists of Details of Students and the Second type of Data Consist of Pending Fees Status which is taken from the Account Office. So The Teacher will use the merge operation here in order to merge the data and provide it meaning. So that teacher will analyze it easily and it also reduces the time and effort of the Teacher from Manual Merging.

Data Wrangling Using Grouping Method

The grouping method in Data wrangling is used to provide results in terms of various groups taken out from Large Data. This method of pandas is used to group the outset of data from the large data set.

Example: There is a Car Selling company and this company have different Brands of various Car Manufacturing Company like Maruti, Toyota, Mahindra, Ford, etc., and have data on where different cars are sold in different years. So the Company wants to wrangle only that data where cars are sold during the year 2010. For this problem, we use another data Wrangling technique which is a pandas *groupby()* method

Data Wrangling by Removing Duplication

Pandas *duplicates()* method helps us to remove duplicate values from Large Data. An important part of Data Wrangling is removing Duplicate values from the large data set.
Syntax: *DataFrame.duplicated(subset=None, keep='first')*
Here subset is the column value where we want to remove the Duplicate value.

In keeping, we have 3 options :

- *if keep ='first' then the first value is marked as the original rest of all values if occur will be removed as it is considered duplicate.*
- *if keep='last' then the last value is marked as the original rest the above same values will be removed as it is considered duplicate values.*
- *if keep ='false' all the values which occur more than once will be removed as all are considered duplicate values.*

Libraries used:

Numpy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays. It is the fundamental package for scientific computing with Python.
Besides its obvious scientific uses, Numpy can also be used as an efficient multi-dimensional container of generic data.
Array in Numpy is a table of elements (usually numbers), all of the same type, indexed by a tuple of positive integers. In Numpy, number of dimensions of the array is called rank of the array.A tuple of integers giving the size of the array along each dimension is known as shape of the array. An array class in Numpy is called as **ndarray**. Elements in Numpy arrays are accessed by using square brackets and can be initialized by using nested Python Lists.

Pandas is an open-source library that is built on top of NumPy library. It is a Python package that offers various data structures and operations for manipulating numerical data and time series. It is mainly popular for importing and analyzing data much easier. Pandas is fast and it has high-performance & productivity for users.

Outliers

An Outlier is a data item/object that deviates significantly from the rest of the (so-called normal) objects. Identifying outliers is important in statistics and data analysis because they can have a significant impact on the results of statistical analyses. The analysis for outlier detection is referred to as outlier mining.
Outliers can skew the mean (average) and affect measures of central tendency, as well as influence the results of tests of statistical significance

How Outliers are Caused?
Outliers can be caused by a variety of factors, and they often result from genuine variability in the data or from errors in data collection, measurement, or recording. Some common causes of outliers are:

- **Measurement errors**: Errors in data collection or measurement processes can lead to outliers.
- **Sampling errors**: In some cases, outliers can arise due to issues with the sampling process.
- **Natural variability**: Inherent variability in certain phenomena can also lead to outliers. Some systems may exhibit extreme values due to the nature of the process being studied.
- **Data entry errors**: Human errors during data entry can introduce outliers.
- **Experimental errors**: In experimental settings, anomalies may occur due to uncontrolled factors, equipment malfunctions, or unexpected events.
- **Sampling from multiple populations:** Data is inadvertently combined from multiple populations with different characteristics.
- **Intentional outliers:** Outliers are introduced intentionally to test the robustness of statistical methods.

Skewness:

Python is a great language for doing data analysis, primarily because of the fantastic ecosystem of data-centric python packages. ***Pandas*** is one of those packages and makes importing and analyzing data much easier. Pandas `dataframe.skew()` function return unbiased skew over requested axis Normalized by N-1. Skewness is a measure of the asymmetry of the probability distribution of a real-valued random variable about its mean. For more information on skewness, refer this link.

Pandas: *DataFrame.skew(axis=None, skipna=None, level=None, numeric_only=None, **kwargs)*

Parameters :

axis : *{index (0), columns (1)}*

skipna : *Exclude NA/null values when computing the result.*

level : *If the axis is a MultiIndex (hierarchical), count along a particular level, collapsing into a Series*

numeric_only : *Include only float, int, boolean columns. If None, will*

attempt to use everything, then use only numeric data. Not implemented for Series.