Data Cleaning Using Pandas in Python

BEGINNER DATA CLEANING PROGRAMMING PYTHON PYTHON STRUCTURED DATA

Introduction

As we know, Data Science is the discipline of study that involves extracting insights from huge amounts of data using various scientific methods, algorithms, and processes. To extract useful knowledge from data, Data Scientists need raw data. This Raw data is a collection of information from various outlined sources and an essential raw material for Data Scientists. It is also known as primary or source data, which is messy and needs cleaning. This beginner's guide will tell you all about Python data cleaning using pandas.

Primary data consists of irregular and inconsistent values, which leads to many difficulties. When using data, the insights and analysis extracted are only as good as the data we use. Essentially, when irregular data is in, then irregular analysis comes out. Here's where Python data cleaning comes into play. Data cleansing is an essential part of the data analytics process. Data cleaning using Pandas removes incorrect, corrupted, garbage, incorrectly formatted, duplicate, or incomplete data within a dataset.

Learning Objectives

- Define data cleaning and its importance in the data analytics process.
- Recognize the importance of accurate, consistent data for effective analysis and decision-making.
- Learn the various techniques and tools in the Python Pandas library for data cleaning.

Table of contents

What Is Data Cleaning?

When working with multiple data sources, there are many chances for data to be incorrect, duplicated, or mislabeled. If data is wrong, outcomes and algorithms are unreliable, even though they may look correct. Data cleaning in data science using Python is changing or eliminating garbage, incorrect, duplicate, corrupted, or incomplete data in a dataset. There's no absolute way to describe the precise steps in data

cleaning because the processes may vary from dataset to dataset. The general data preparation process initiative is data cleansing, data cleansing, or scrubbing.

Data Cleaning using Pandas in Python is important in developing reliable answers within the analytical process. It is observed to be a basic feature of the info science basics. The motive of Python data cleaning services is to construct uniform and standardized data sets that enable easy access to data analytics tools and business intelligence and perceive accurate data for each problem.

Why Is Data Cleaning Essential?

Data Cleaning using Pandas in Python is the most important task that a data science professional should do. Wrong or bad-quality data can be detrimental to processes and analysis. Clean data will ultimately increase overall productivity and permit the very best quality information in decision-making.

Following are some reasons why Python data cleaning is essential:

1. Error-Free Data:

When combining multiple data sources, there may be a chance of so much error. Through data cleaning in data science using Python, errors can be removed from data. Having clean data free from wrong and garbage values can help perform analysis faster and more efficiently. By doing this task, we save a considerable amount of time. The results won't be accurate if we use data containing garbage values. When we don't use accurate data, we will surely make mistakes. Monitoring errors and good reporting help find where errors come from and make it easier to fix incorrect or corrupt data for future applications.

2. Data Quality:

The quality of the data is the degree to which it follows the rules of particular requirements. For example, if we have imported phone number data of different customers, and in some places, we have added customers' email addresses. However, because our needs were straightforward for phone numbers, the email addresses would be invalid data. Here, some pieces of data follow a specific format. Some types of numbers have to be in a specific range.

Some data cells might require selected quiet data like numeric, Boolean, etc. In every scenario, there are some mandatory constraints our data should follow. Certain conditions affect multiple fields of data in a particular form. Particular types of data have unique restrictions. Data will always be invalid if it isn't in the required format. Data cleaning in data science using Python will help us simplify this process and avoid useless data values.

3. Accurate and Efficient:

Ensuring the data is close to the correct values. We know that most data in a dataset are valid, and we should focus on establishing its accuracy. Even if the data is authentic and correct, it isn't accurate. Determining accuracy helps to figure out whether the data entered is accurate or not. For example, if a customer's address is stored in the specified format, it may not be in the right one. The email has an additional character or value that makes it incorrect or invalid. Another example for data cleaning in machine learning Python is the phone number of a customer. This means that we have to rely on data sources to cross-check the data to figure out if it's accurate or not. Depending on the kind of data we are using, we might be able to find various resources that could help us in this regard for cleaning.

4. Complete Data:

Completeness is the degree to which we should know all the required values. Completeness is a little more challenging to achieve than accuracy or quality. Because it's nearly impossible to have all the info we need, only known facts can be entered. We can try to complete data by redoing the data-gathering activities like approaching the clients again, re-interviewing people, etc. For example, we might need to enter every customer's contact information. However, a number of them might not have email addresses. In this case, we have to leave those columns empty. If a system requires us to fill all columns, we can try to enter missing or unknown ones. However, entering such values does not mean that the data is complete. It would still be referred to as incomplete.

5. Maintains Data Consistency:

To ensure the data is consistent within the same dataset or across multiple datasets, we can measure consistency by comparing two similar systems. We can also check the data values within the same dataset to see if they are consistent. Consistency can be relational. For example, a customer's age might be 25, which is a valid value and also accurate, but it is also stated as a senior citizen in the same system. In such cases, we must cross-check the data, similar to measuring accuracy, and see which value is true. Is the client a 25-year-old? Or is the client a senior citizen? Only one of these values can be true. There are multiple ways to make your data consistent.

- By checking in different systems.
- By checking the source.
- By checking the latest data.

Data Cleaning Cycle

It is the method of analyzing, distinguishing, and correcting untidy, raw data. Python Pandas Data Cleaning involves filling in missing values, handling outliers, and distinguishing and fixing errors in the dataset. Meanwhile, the techniques used for data cleaning in data science using Python might vary in step with

different types of datasets. In this tutorial, we will learn how to clean data using pandas. The following are standard steps to map out Python Pandas data cleaning:

Data Cleaning With Pandas

Data scientists spend a lot of time cleaning datasets and getting them in the form they can work. It is an essential skill of Data Scientists to work with messy data, missing values, and inconsistent, noisy, or nonsensical data. Python provides a built-in module called Pandas that works smoothly. Pandas is a popular Python library for data processing, cleaning, manipulation, and analysis. Pandas stand for "Python Data Analysis Library." It consists of classes on reading, processing, and writing CSV files. Numerous Data cleaning tools are present, but the Pandas library provides a fast and efficient way to manage and explore data. It does that by providing us with Series and DataFrames, which help us represent data efficiently and manipulate it in various ways.

This article will use the Pandas module to clean our dataset.

We are using a simple dataset for data cleaning, i.e., the iris species dataset. You can download this dataset from kaggle.com.

Let's get started with data cleaning using Pandas.

To start working with Pandas, we need first to import it. We are using Google Colab as IDE to import Pandas in Google Colab.

```
#importing module import pandas as pd
```

Step 1: Import Dataset

To import the dataset, we use Pandas' read_csv() function and store it in Pandas DataFrame named Data. As the dataset is in tabular format, it will be automatically converted into a DataFrame when working with tabular data in Pandas. A DataFrame is a two-dimensional, mutable data structure in Python. It is a combination of rows and columns like an Excel sheet.

Python Code:

```
#importing the dataset by reading the csv file data = pd.read_csv('Iris.csv') #displaying the first five rows of dataset print(data.head())
```

The head() function is a built-in function in pandas for the dataframe used to display the rows of the dataset. We can specify the number of rows by giving the number within the parenthesis. By default, it displays the first five rows of the dataset. If we want to see the last five rows of the dataset, we use the tail()function of the dataframe like this:

```
#displayinf last five rows of dataset data.tail()
```

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
145	146	6.7	3.0	5.2	2.3	Iris-virginica
146	147	6.3	2.5	5.0	1.9	Iris-virginica
147	148	6.5	3.0	5.2	2.0	Iris-virginica
148	149	6.2	3.4	5.4	2.3	Iris-virginica
149	150	5.9	3.0	5.1	1.8	Iris-virginica

Step 2: Merge Dataset

Merging the dataset combines two datasets and lining up rows based on some particular or common property for data analysis. We can do this by using the merge() function of the dataframe. Following is the syntax of the merge function:

```
DataFrame_name.merge(right, how='inner', on=None, left_on=None, right_on=None, left_index=False, right_index=False, sort=False, suffixes=('_x', '_y'), copy=True, indicator=False, validate=None)
```

However, we don’t need to merge two datasets in this case so that we will skip this step.

Step 3: Rebuild Missing Data

We will use another function to find and fill in the missing data in the dataset. There are 4 ways to find the null values if present in the dataset. Let’s see them one by one:

Using isnull() function:

```
data.isnull()
```

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	False	False	False	False	False	False
1	False	False	False	False	False	False
2	False	False	False	False	False	False
3	False	False	False	False	False	False
4	False	False	False	False	False	False
...	...	...	...	...	...	...
145	False	False	False	False	False	False
146	False	False	False	False	False	False
147	False	False	False	False	False	False
148	False	False	False	False	False	False
149	False	False	False	False	False	False

150 rows × 6 columns

This function in data cleaning in machine learning Python provides a boolean value for the complete dataset to determine whether any null value is present.

Using isna() function:

```
data.isna()
```

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	False	False	False	False	False	False
1	False	False	False	False	False	False
2	False	False	False	False	False	False
3	False	False	False	False	False	False
4	False	False	False	False	False	False
...	...	...	...	...	...	...
145	False	False	False	False	False	False
146	False	False	False	False	False	False
147	False	False	False	False	False	False
148	False	False	False	False	False	False
149	False	False	False	False	False	False

150 rows × 6 columns

This is the same as the isnull() function. And provides the same output.

Using isna().any()

```
data.isna().any()
```

```
Id              False
SepalLengthCm   False
SepalWidthCm    False
PetalLengthCm   False
PetalWidthCm    False
Species         False
dtype: bool
```

This function in Python Pandas also gives a boolean value indicating whether a null value is present, but it gives results column-wise, not in tabular format.

Using isna(). sum()

```
data.isna().sum()
```

```
Id              0
SepalLengthCm   0
SepalWidthCm    0
PetalLengthCm   0
PetalWidthCm    0
Species         0
dtype: int64
```

This function gives the sum of the null values preset in the dataset column-wise.

Using isna().any().sum()

```
data.isna().any().sum()
```

```
data.isna().any().sum()
```

```
0
```

This function gives output in a single value, whether any null is present.

There are no null values present in our dataset. But if any null values are preset, we can fill those places with any other value using the fillna() function of DataFrame.Following is the syntax of fillna() function:

```
DataFrame_name.fillna(value=None, method=None, axis=None, inplace=False, limit=None, downcast=None)
```

This function will fill NA/NaN or 0 values instead of null spaces. You may also drop null values using the dropna method when the amount of missing data is relatively small and unlikely to affect the overall.

Step 4: Standardization and Normalization

Data standardization and normalization are common practices in machine learning.

Standardization is another scaling technique where the values are centred around the mean with a unit standard deviation. This means that the mean of the attribute becomes zero, and the resultant distribution has a unit standard deviation.

Normalization is a scaling technique in which values are shifted and rescaled to range between 0 and 1. It is also known as Min-Max scaling.

To know more about this, click here.

This step is not needed for the dataset we are using. So, we will skip this step.

Step 5: De-Duplicate Data

De-Duplicate means removing all duplicate values. There is no need for duplicate values in data analysis. These values only affect the accuracy and efficiency of the analysis result. To find duplicate values in the dataset, we will use a simple dataframe function, i.e., duplicated(). Let's see the example:

```
data.duplicated()
```

```
0      False
1      False
2      False
3      False
4      False
       ...
145    False
146    False
147    False
148    False
149    False
Length: 150, dtype: bool
```

This function also provides bool values for duplicate values in the dataset. As we can see, the dataset doesn't contain any duplicate values. A dataset containing duplicate values can be removed using the drop_duplicates() function. Following is the syntax of this function:

```
DataFrame_name.drop_duplicates(subset=None, keep='first', inplace=False, ignore_index=False)
```

Step 6: Verify and Enrich the Data

After removing null, duplicate, and incorrect values, we should verify the dataset and its accuracy. In this step, we must check that the data cleaned so far makes sense. If the data is incomplete, we have to enrich it again by data gathering activities like approaching the clients again, re-interviewing people, etc. Completeness is a little more challenging to achieve accuracy or quality in the dataset.

Step 7: Export Dataset

This is the last step of the data-cleaning process. After performing all the above operations, the data is transformed into a clean dataset and is ready to export for the next process in Data Science or Data Analysis.

Conclusion

Data cleaning in machine learning python is a critical task in data science that helps ensure the accuracy and reliability of analysis and decision-making. Through data cleaning using Pandas in Python, errors can be removed, data quality can be improved, and the data can be made more accurate and complete. By utilizing the various techniques and tools available for data cleaning in the Python Pandas library, data scientists can gain insights from the raw data and make better-informed decisions.

Key Takeaways:

- Importance of data cleaning using Pandas in Python for error-free, high-quality, and consistent data.
- Step-by-step data cleaning process using Pandas functions.
- Handling missing data, duplicates, standardization, and data enrichment.

Frequently Asked Questions

Q1. What is data cleaning in Python?

A. Data cleaning using Pandas in Python involves removing or correcting errors, inconsistencies, and inaccuracies in datasets using libraries like Pandas and NumPy.

Q2. What is the data cleaning process?

A. The data cleaning includes identifying missing or incorrect data, removing duplicates, correcting errors, and standardizing formats for consistency.

Q3. Is Python best for data cleaning?

A. Yes, Python is highly regarded for data cleaning due to its powerful libraries, such as Pandas and NumPy, which provide efficient tools for manipulating and cleaning data.

Q4. How do you clean data in CSV using Python?

A. To clean data in a CSV using Python, load the data with Pandas, identify and handle missing values, remove duplicates, correct inconsistencies, and save the cleaned data to a CSV file.

Article Url - https://www.analyticsvidhya.com/blog/2021/06/data-cleaning-using-pandas/

Neelu