TDI PYTHON FOR DATA ANALYTICS WEEK 5:

Data Cleaning, Transformation, and Merging DataFrames

Welcome to Week 5 of Python for Data Analytics!

This week, you will focus on more advanced data analysis tasks using the Pandas library. We will dive into cleaning messy data, transforming and normalizing it for analysis, and combining multiple datasets efficiently. These skills are essential for preparing data before performing more indepth analysis or feeding it into machine learning models.

Topics Covered:

Data cleaning

Data transformation and normalization

Merging and concatenating DataFrames

Dataset: Titanic Analysis

We will continue to use the Titanic dataset. This dataset contains information about passengers, such as age, class, fare, and survival status. You'll work with this data to explore data cleaning techniques, transformations, and the merging of multiple DataFrames.

Learning Resources:

Data Cleaning with Pandas

Data Cleaning, Transformations and Normalizations

Data Cleaning

Questions:

1. Data Cleaning

1. Explain why handling missing data is crucial for analysis.
2. Demonstrate how to identify missing values in the Titanic dataset using Pandas.
3. What are the different methods for handling missing data in Pandas (e.g., “dropna()”, “fillna()”)? When would you choose each method?
4. Clean the Titanic dataset by filling missing values in the 'Age' column with the median age of the passengers, and explain your choice.
5. Explain how duplicate records can impact analysis.
6. Demonstrate how to detect and remove duplicate rows using Pandas’ “drop_duplicates()” method.
7. Check the Titanic dataset for duplicates and remove any found, if applicable.
8. perform everyother data cleaning steps to prepare your data for analysis eg(convert the passanger class column, from 1 to 1st class)
9. Define what outliers are and discuss how they can impact data analysis.
10. Demonstrate how to identify outliers in a numerical column like 'Fare' in the Titanic dataset.
11. Choose a strategy (e.g., capping, removing) for handling outliers in the 'Fare' column, and explain why you made this choice.

2. Data Transformation and Normalization

1. Discuss the importance of ensuring the correct data types for columns in a DataFrame.
2. Examine the data types of each column in the Titanic dataset using the “dtypes” attribute and convert any columns that are incorrectly typed (e.g., convert 'PassengerId' to a string type).

3. Discuss how transforming existing data can lead to the creation of new insights.
4. Create a new feature called “FamilySize” by summing the 'SibSp' and 'Parch' columns, adding 1 (to include the passenger themselves).
5. Create another feature called “FarePerPerson” by dividing the 'Fare' by the new “FamilySize” feature, and explain how this new feature could provide more insights about passengers' economic backgrounds.

3. Merging and Concatenating DataFrames

1. Explain the difference between merging and concatenating DataFrames.
2. Split the Titanic dataset into two subsets: one containing passengers who survived and one containing passengers who did not survive. Concatenate these two DataFrames back together using the “concat()” function.
3. Describe the different types of joins (inner, outer, left, right) used in merging DataFrames.
4. Suppose you have an additional DataFrame containing information about passengers’ destinations. Demonstrate how to merge this DataFrame with the Titanic dataset using “merge()” based on a common key like 'PassengerId'.
5. Perform an inner join, and explain how the join type affects the final DataFrame.
6. Suppose you have a DataFrame containing information about passengers' cabin numbers and their amenities. Merge this DataFrame with the Titanic dataset to explore if access to amenities influenced survival rates. Perform an analysis using a “groupby()” function to calculate survival rates based on amenities.

4. Advanced Data Cleaning Techniques

1. String Manipulation:
2. Explain how Pandas can handle string manipulation for cleaning and transforming text data.
3. Use Pandas' string functions to extract the first letter from the 'Cabin' column to create a new column that represents the deck of each passenger.
4. Investigate whether passengers from different decks had different survival rates.

Submission Instructions:

Submit your Python scripts or Jupyter notebook containing your solutions to the tasks outlined above.

Ensure that your code is clean, welldocumented, and follows best practices for readability and performance.