Hands-on Lab: Generative AI for Data Repository Maintenance and Administration

Learner effort: 30 mins

Introduction

Data repository management and administration is a crucial aspect of data engineering. It consists of all aspects that ensure the maintenance of the quality of the data, integrity of the data, and completeness of the data. In this lab, you will explore the different aspects of data repository administration and management in the context of a real-world data set.

Objectives

In this lab, you will learn how to use generative AI to generate codes that will do the following:

1. Handle missing entries in the data
2. Remove duplicate entries from the data
3. Outlier detection in the data

Testing Interface

The testing interface to be used for this lab is available in course page at the end of this lesson. Please keep that open in a browser on the side and follow the steps mentioned in it to set it up.

Data set

The data set being used for this lab is a modified subset that maps the price of laptops with different attributes. The data set is a filtered and modified version of the "Laptop Price Prediction using specifications" data set, available under the Database Contents License (DbCL) v1.0 on the Kaggle website.

The data set has been made available at the following URL.

```
URL = "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMSkillsNetwork-AI0273EN-SkillsNetwork/labs/v1/m2/data/laptop_pricing_dataset_base.csv"
```

Handle missing values

The first thing to do with any data would be to handle any missing data. Missing data can be addressed in multiple ways.

1. If the attribute for which the value is missing is categorical or discrete, replace it with most frequent value.

2. If the attribute for which the value is missing is continuous, replace it with either the mean or the median value.

3. If the attribute for which the value is missing is the target attribute, its best to drop that entire row of data.

In this case, the data points missing are in continuous variables, and missing data is represented by the character `?`. You can use the Generative AI model to write a code that handles these missing values for you.

```
Write a Python code that reads a csv record from a URL and does the following.
1. Identify the columns which have values as "?" in them.
2. Replace these with the mean value of the respective attribute.
2. Modify the data type of the attribute to float after replacement.
```

You can expect a code, similar to the one shown below, as a response.

```
import pandas as pd
# Read CSV record from URL
url = 'https://example.com/data.csv'
data = pd.read_csv(url)
print(data.dtypes)
# Identify columns with "?" values
cols_with_question_mark = data.columns[data.isin(['?']).any()]
print(cols_with_question_mark)
# Replace "?" values with the mean value of the respective attribute
for col in cols_with_question_mark:
    mean_val = pd.to_numeric(data[col], errors='coerce').mean()
    data[col] = pd.to_numeric(data[col].replace('?', mean_val))
# Modify the data type of the attribute to float after replacement
data[cols_with_question_mark] = data[cols_with_question_mark].astype(float)
# Print the modified data
print(data.dtypes)
```

You can update this code with the URL for the dataset as shared before. The output of the execution of this code is expected to be as shown in the image below.
Note that `Screen Size` and `Weight` attributes are the ones identified to have values missing and hence, their data type earlier was `object`. Upon modification and filling with attribute mean values, the data type has been modified to `float64`.

Remove duplicates

Next, you need to remove the duplicate entries from the data set. You can create the required Python code for the same using generative AI with the following prompt.

```
For a data in a python dataframe, write a code that removes all duplicate entries from the data. Print the total number of rows in the data before and after removal, to prove the working of the code.
```

You can expect to see the following code in response.

```
import pandas as pd
# Assuming 'data' is the pandas DataFrame containing the data
# Print the total number of rows before removal
print("Total number of rows before removal:", len(data))
# Remove duplicate entries
data.drop_duplicates(inplace=True)
# Print the total number of rows after removal
print("Total number of rows after removal:", len(data))
# Print the modified data
print(data)
```

This code snippet will remove all duplicate entries from the DataFrame 'data' and print the total number of rows before and after removal to demonstrate the effectiveness of the code.

You can append the relevant part of this code to your existing file on the testing interface. Make sure to comment the print statements of the previous step in order to have clear visibility of the output for this one.

The contents of your testing code should now be somewhat as shown below.

```
import pandas as pd
# Read CSV record from URL
url = "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMSkillsNetwork-AI0273EN-SkillsNetwork/labs/v1/m2/data/laptop_pricing_dataset_base.csv"
data = pd.read_csv(url)
#print(data.dtypes)
# Identify columns with "?" values
cols_with_question_mark = data.columns[data.isin(['?']).any()]
#print(cols_with_question_mark)
# Replace "?" values with the mean value of the respective attribute
for col in cols_with_question_mark:
    mean_val = pd.to_numeric(data[col], errors='coerce').mean()
    data[col] = pd.to_numeric(data[col].replace('?', mean_val))
# Modify the data type of the attribute to float after replacement
data[cols_with_question_mark] = data[cols_with_question_mark].astype(float)
# Print the modified data
#print(data.dtypes)
print("Total number of rows before removal:", len(data))
# Remove duplicate entries
data.drop_duplicates(inplace=True)
# Print the total number of rows after removal
print("Total number of rows after removal:", len(data))
```

Note that the previous print statements have been commented.
The output expected is

```
theia@theia-abhishekg1:/home/project$ python3 test_file.py
Total number of rows before removal: 243
Total number of rows after removal: 238
```

This shows that there were 5 duplicate entries in the data set, which have now been removed.

Outlier identification

You should now test the data for quality and identify if there are any anomalous outliers in target attribute 'Price'. You can simply use the following prompt on the generative AI platform to create a code that will extract the required entries.

```
Write a python code to extract entries of a dataframe where the attribute 'Price' has outliers that might be anomalies in comparison to the other data.
```

You can expect the following code in response.

```
import pandas as pd
import numpy as np
# Assuming 'data' is the pandas DataFrame containing the data
# Calculate the mean and standard deviation of the prices
mean_price = data['Price'].mean()
std_price = data['Price'].std()
# Define a threshold for outliers (e.g., 3 standard deviations away from the mean)
outlier_threshold = 3
# Identify outliers using the threshold
outliers = data[(np.abs(data['Price'] - mean_price) > outlier_threshold * std_price)]
# Extract entries with price outliers
entries_with_outliers = data[data['Price'].isin(outliers['Price'])]
# Print the entries with price outliers
print("Entries with price outliers:")
print(entries_with_outliers)
```

This code snippet calculates the mean and standard deviation of the prices in the data set, identifies outliers based on a specified threshold, and then extracts entries where the 'Price' attribute has outliers. This will help you isolate and analyze entries with anomalous price values in comparison to the rest of the data.

You can use the relevant part of this code and add it to the existing file on the testing interface. Make sure to comment out the previous print statements to effectively see the output of this code snippet properly.

Updated code on the testing interface should be as shown below.

```
import pandas as pd
# Read CSV record from URL
url = "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMSkillsNetwork-AI0273EN-SkillsNetwork/labs/v1/m2/data/laptop_pricing_dataset_base.csv"
data = pd.read_csv(url)
#print(data.dtypes)
# Identify columns with "?" values
cols_with_question_mark = data.columns[data.isin(['?']).any()]
#print(cols_with_question_mark)
# Replace "?" values with the mean value of the respective attribute
for col in cols_with_question_mark:
    mean_val = pd.to_numeric(data[col], errors='coerce').mean()
    data[col] = pd.to_numeric(data[col].replace('?', mean_val))
# Modify the data type of the attribute to float after replacement
data[cols_with_question_mark] = data[cols_with_question_mark].astype(float)
# Print the modified data
#print(data.dtypes)
#print("Total number of rows before removal:", len(data))
# Remove duplicate entries
data.drop_duplicates(inplace=True)
# Print the total number of rows after removal
#print("Total number of rows after removal:", len(data))
import numpy as np
# Assuming 'data' is the pandas DataFrame containing the data
# Calculate the mean and standard deviation of the prices
mean_price = data['Price'].mean()
std_price = data['Price'].std()
# Define a threshold for outliers (e.g., 3 standard deviations away from the mean)
outlier_threshold = 3
# Identify outliers using the threshold
outliers = data[(np.abs(data['Price'] - mean_price) > outlier_threshold * std_price)]
# Extract entries with price outliers
entries_with_outliers = data[data['Price'].isin(outliers['Price'])]
# Print the entries with price outliers
print("Entries with price outliers:")
print(entries_with_outliers)
```

Output of this code will be

```
theia@theia-abhishekg1:/home/project$ python3 test_file.py
Entries with price outliers:
    Manufacturer  Category     Screen  GPU  OS  ...  CPU_frequency  RAM_GB  Storage_GB_SSD  Weight_kg  Price
64          Asus         1    Full HD    3   1  ...            2.9      16             256       3.60   3810
77          Dell         5    Full HD    3   1  ...            2.9      16             256       3.42   3665
144       Lenovo         3  IPS Panel    3   1  ...            2.8       8             256       3.40   3810
159        Razer         1    Full HD    3   1  ...            2.8      16             256       1.95   3301

[4 rows x 12 columns]
```

As can be seen, this data is identified as having abnormal price values.

Conclusion

Congratulations on completing this lab!

You have now learned how to use generative AI to:

1. Handle missing values in the data
2. Remove duplicate values from the data
3. Identify outlier data

Author(s)

Abhishek Gagneja