Mitglieder: Tobias Kurfess, Robby Baur, Nico Melzer

Data Warehousing Projekt

1 Vorbereitungen

Um alle Skripte des Projekts auszuführen, müssen die Pfad geändert werden und ein geeigneter Ablageort für die Dateien gewählt werden. Die stellen an welchen der Pfad verbessert werden muss, sind im Skript mit folgendem Kommentar markiert:

```
# PFAD ÄNDERN
```

Außerdem müssen einige Packages in der Miniconda-Umgebung installiert werden. Daher einfach den folgenden Quellcode kopieren und im Terminal bzw. der Shell einfügen:

```
conda install pandas
conda install numpy
conda install matplotlib
conda install xlsxwriter
```

2 Schritt 01: Business Understanding

Das Ziel des Projekts ist es Shopping Daten auf folgende Aspekte zu analysieren:

- Analyse über Verkäufe und Umsatz
- Analyse über die Altergruppen nach Shopping-Verhalten
- Analyse beliebter Produktgruppen

Daraus resultieren dann die folgenden Fragestellungen, die wir genauer analysieren wollen:

1. **Analyse über Sales und Revenue:** "Wie hat sich der Gesamtumsatz pro Monat im vergangenen Jahr verändert, und welche Produkte oder Produktkategorien trugen am meisten zum Umsatz bei?"
2. **Analyse der Altersgruppen nach Shopping-Verhalten:** "Gibt es signifikante Unterschiede im Kaufverhalten zwischen verschiedenen Altersgruppen, bezogen auf die Häufigkeit der Einkäufe, durchschnittliche Ausgaben pro Einkauf oder bevorzugte Produktkategorien?"
3. **Analyse beliebter Produktgruppen:** "Welche Produktgruppen sind am beliebtesten in Bezug auf Verkaufszahlen und Umsatz, und wie unterscheidet sich ihre Popularität nach Standort und Kundengeschlecht?"

3 Schritt 02: Data Understanding

Dieser Schritt wird mit dem Skript ”01_Data_Understanding.py“ durchgeführt. Die Shopping-Daten aus der CSV-Datei werden geladen und analysiert. Anschließend werden die Ergebnisse grafisch aufbereitet und in eine neue Excel-Datei mit dem Namen ”Analyse_DataUnderstanding“ gespeichert.

3.1 Code-Dokumentation (01_Data_Understanding.py)

Der Code in diesem Python-Skript macht folgendes:

1. **Daten Laden:**
   - Der Code lädt Daten aus einer CSV-Datei in einen **pandas** DataFrame. Ein DataFrame ist eine tabellarische Datenstruktur in **pandas**, ähnlich wie eine Tabelle in einer Datenbank oder ein Arbeitsblatt in Excel.
2. **Daten Analyse:**
   - Er berechnet den Gesamtumsatz für jede Transaktion, indem er die Spalten **Quantity** und **Avg_Price** multipliziert.
   - Dann extrahiert der Code den Monat aus dem Transaktionsdatum und summiert den Gesamtumsatz pro Monat.
   - Anschließend werden die Verkaufszahlen und der Gesamtumsatz nach Produktkategorien gruppiert.
   - Die Verteilung der Geschlechter wird berechnet, indem gezählt wird, wie oft jedes Geschlecht im Datensatz vorkommt.
3. **Erstellen von Grafiken:**
   - Der Code erzeugt mehrere Grafiken, darunter ein Balkendiagramm des monatlichen Gesamtumsatzes, Balkendiagramme der Verkaufszahlen und des Umsatzes pro Produktkategorie und ein Kreisdiagramm der Geschlechterverteilung.
4. **Erstellung einer Excel-Datei:**
   - Es wird eine Excel-Datei erstellt, in die die ersten fünf Zeilen des DataFrames, eine Beschreibung der Daten (wie z.B. Mittelwert, Standardabweichung, etc.), und die Anzahl der fehlenden Werte in jeder Spalte eingefügt werden.
5. **Einfügen der Grafiken in Excel:**
   - Die erstellten Grafiken werden als Bilder in die Excel-Datei eingefügt. Dafür wird jede Grafik in einem Binärdatenstrom (BytesIO-Objekt) gespeichert, der dann in ein Bild im Excel-Dokument eingefügt wird.
6. **Speichern der Excel-Datei:**
   - Zum Schluss wird die Excel-Datei geschlossen und gespeichert, was den Prozess abschließt.

Bemerkung: Die Funktion **save_fig_to_bytes** hilft dabei, die Grafiken zu speichern, ohne sie auf dem Dateisystem ablegen zu müssen, indem sie direkt in den Speicher (BytesIO-Objekt) geschrieben werden. Diese Bilder werden dann in der Excel-Datei verwendet, um visuelle Darstellungen der analysierten Daten bereitzustellen. Während des ganzen Prozesses gibt der Code Statusmeldungen in der Konsole aus, damit der Benutzer den Fortschritt des Skripts verfolgen kann.

4 Schritt 03: Data Preparation

Dieser Code führt eine Datenvorbereitung und -bereinigung für die CSV-Datei mit den Shopping-Daten durch. Dabei wird die CSV-Datei importiert und verschiedene Operationen zur Bereinigung der Datei durchgeführt. Am Ende gibt das Skript eine bereinigte Version der CSV-Datei aus. Eine genauere Erklärung für jeden Schritt ist im ersten Unterkapitel zu finden.

4.1 Code-Dokumentation (02_Data_Preparation.py)

Der Code in diesem Python-Skript macht folgendes:

1. **Daten Laden:**
   - Der Code lädt eine CSV-Datei in einen DataFrame, eine tabellarische Datenstruktur, die von der Bibliothek **pandas** zur Datenmanipulation verwendet wird.
2. **Berechnung des Gesamtumsatzes:**
   - Der Code prüft, ob die Spalten **Quantity** und **Avg_Price** im DataFrame vorhanden sind und berechnet dann den Gesamtumsatz jeder Transaktion, indem er diese beiden Spalten multipliziert.
3. **Behandlung Fehlender Werte:**
   - Für jede Spalte im DataFrame prüft der Code, ob es sich um kategoriale (textbasierte) oder numerische Daten handelt.
   - Fehlende Werte in kategorialen Spalten werden mit dem häufigsten Wert (Modus) ersetzt.
   - Fehlende Werte in numerischen Spalten werden mit dem Median der Spalte ersetzt.
4. **Identifizierung und Behandlung von Ausreißern:**
   - Der Code identifiziert Ausreißer in numerischen Spalten. Ein Wert gilt als Ausreißer, wenn er mehr als drei Standardabweichungen vom Mittelwert entfernt ist.
   - Ausreißer werden durch **NaN** (eine Repräsentation für fehlende Daten in pandas) ersetzt.
   - Diese **NaN**-Werte, die durch die Entfernung von Ausreißern entstanden sind, werden anschließend durch den Median der jeweiligen Spalte ersetzt.
5. **Kodierung Kategorialer Variablen:**
   - Der Code wandelt kategoriale Variablen in numerische um, indem er für jede Kategorie eine neue Spalte (sogenannte Dummy-Variable) erstellt. Die Option **drop_first=True** vermeidet Redundanzen, indem sie die erste Dummy-Spalte weglässt.

6. **Erstellung neuer Merkmale:**
    - Wenn die Spalte **CustomerID** vorhanden ist, summiert der Code den Gesamtumsatz pro Kunde und fügt diese Information als neue Spalte **Customer_Total_Sales** zum DataFrame hinzu.
7. **Speichern der Bereinigten Daten:**
    - Der bereinigte und vorbereitete DataFrame wird als neue CSV-Datei gespeichert.
8. **Abschluss:**
    - Schließlich gibt der Code eine Statusmeldung aus, die den Abschluss des Vorgangs anzeigt und den Speicherort der bereinigten Daten angibt.

Zusammenfassend führt dieser Code wichtige Schritte der Datenvorbereitung durch, um die Daten für die Analyse oder Modellierung vorzubereiten. Er behandelt fehlende Werte, Ausreißer und kodiert kategoriale Variablen in ein Format, das von den meisten Analyse- und Modellierungswerkzeugen verwendet werden kann.