RFM Sınıfı Dokümantasyonu - TR

Bu depo, müşteri segmentasyonu için RFM (Recency, Frequency, Monetary) analizini uygulayan bir Python `Rfm` sınıfı içerir.

Kullanım

1. **Sınıfı Örnekleyin**: `Rfm` sınıfının bir örneğini oluşturmak için, müşteri verilerini içeren CSV dosyasının yolunu, müşteri kodu, sipariş tarihi ve miktar için parametreler ile sağlayın.
2. **RFM Skorlarını Hesapla**: RFM skorlarını hesaplamak için `setScore()` metodunu kullanın.
3. **Müşterileri Kategorize Et**: İsteğe bağlı olarak, `setCategory()` metodu kullanarak RFM skorlarına dayanarak müşterileri kategorilere ayırın.
4. **Segmentleri Görselleştir**: `plotPie()` metodu kullanarak müşteri segmentlerinin dağılımını gösteren bir pasta grafiği oluşturun.
5. **Sonuçları Kaydet**: Sonuçları bir CSV dosyasına kaydetmek için `save()` metodunu kullanın.

Örnek

```
# Rfm sınıfını dosya yolu ve ek parametreler ile başlat
# (Not: Sütun adları, tarih formatı)
rfm = Rfm("dummy.csv",
        "Customer ID",
        "created_at",
        "grand_total",
        '%m/%d/%Y')

# RFM skorlarını hesapla
rfm.setScore(1, 1, 1)

# RFM skorlarına göre müşterileri kategorilere ayır
rfm.setCategory(4, 3, 2, 1, 0)

# Müşteri segmentlerinin pasta grafiğini çiz
rfm.plotPie()

# Sonuçları bir CSV dosyasına kaydet
rfm.save("rfmresults.csv")
```

Metodlar

- `setScore(r=0.15, f=0.28, m=0.57)`: Recency, frequency ve monetary değerlerine göre RFM skorlarını hesaplar.

- `setCategory(a=4.5, b=4, c=3, d=1.5, r=30)`: RFM skorlarına göre müşterileri segmentlere ayırır.
- `plotPie()`: Müşteri segmentlerinin pasta grafiğini oluşturur.
- `save(path)`: Sonuçları bir CSV dosyasına kaydeder.

Parametreler

- `filepath`: Müşteri verilerini içeren CSV dosyasının yolu.
- `cc`: Müşteri kodu için sütun adı.
- `od`: Sipariş tarihi için sütun adı.
- `a`: Miktar için sütun adı.
- `format`: Sipariş tarihi sütununun formatı. (Örnek: 30/12/22 13:40:23 = '%m/%d/%Y %H:%M:%S')
- `r`, `f`, `m`: Sırasıyla recency, frequency ve monetary değerler için ağırlık faktörleri. RFM skorlarını hesaplarken kullanılır. Varsayılan değerler sırasıyla 0.15, 0.28 ve 0.57'dir.
- `a`, `b`, `c`, `d`, `r`: Müşterileri segmentlere ayırmak için eşik değerler. Bu eşikler, farklı müşteri segmentleri arasındaki sınırları tanımlar. Varsayılan olarak müşteriler şu şekilde kategorilendirilir:
  - `a+`: Üst Müşteriler
  - `b+`: Yüksek Değerli Müşteriler
  - `c+`: Orta Değerli Müşteriler
  - `d+`: Düşük Değerli Müşteriler
  - `default`: a=4.5, b=4, c=3, d=1.5, r=30
  - `r`: Kayıp Müşteriler ve Yeni Müşterileri kategor

ize etmek için recency eşik değeri. Varsayılan olarak 30 gün olarak ayarlanmıştır.

Öznitelikler

- `df`: Müşteri verilerini içeren DataFrame. Veri temizleme işlemleri sonrasında, RFM skorları ve müşteri segmentasyonu bilgileri bu DataFrame içinde saklanır.

Detaylı Kullanım

1. **Dosya Yolu ve Parametreler**: `Rfm` sınıfını başlatırken, müşteri verilerini içeren CSV dosyasının yolunu, müşteri kodu, sipariş tarihi ve miktar sütunlarının adlarını ve tarih formatını sağlayın.

   ```
   rfm = Rfm("musteri_verileri.csv",
             "Musteri_ID",
             "Siparis_Tarihi",
             "Tutar",
             '%d/%m/%Y')
   ```

2. **RFM Skorlarını Hesaplama**: `setScore()` metodu, recency, frequency

ve monetary değerlerine dayanarak RFM skorlarını hesaplar. Bu metot, her müşteri için ağırlıklı bir ortalama kullanarak bir RFM skoru hesaplar.

```
rfm.setScore(0.15, 0.28, 0.57)
```

3. **Müşteri Segmentasyonu**: `setCategory()` metodu, RFM skorlarına göre müşterileri farklı segmentlere ayırır. Bu metot, belirli eşik değerlere göre müşterileri kategorilendirir.

   ```
   rfm.setCategory(4.5, 4, 3, 1.5, 30)
   ```

4. **Görselleştirme**: `plotPie()` metodu, müşteri segmentlerinin dağılımını gösteren bir pasta grafiği oluşturur. Bu grafik, müşteri segmentlerinin oransal dağılımını görsel olarak temsil eder.

   ```
   rfm.plotPie()
   ```

5. **Sonuçları Kaydetme**: `save()` metodu, RFM analiz sonuçlarını belirtilen bir dosya yolunda CSV dosyası olarak kaydeder. Bu, analiz sonuçlarını daha sonra kullanmak veya incelemek üzere saklamayı kolaylaştırır.

   ```
   rfm.save("rfm_sonuclari.csv")
   ```

Bu dokümantasyon, `Rfm` sınıfını kullanma konusunda genel bir rehber sağlar. Pratik uygulamalarda, veri setinin özelliklerine ve iş gereksinimlerine göre parametreler ve metotlar uygun şekilde ayarlanmalıdır.

RFM Class Documentation - EN

This repository contains a Python `Rfm` class that implements RFM (Recency, Frequency, Monetary) analysis for customer segmentation.

Usage

1. **Instantiate the Class**: Create an instance of the `Rfm` class by providing the file path to the CSV containing customer data, along with parameters for customer code, order date, and amount.
2. **Calculate RFM Scores**: Use the `setScore()` method to compute the RFM scores.
3. **Categorize Customers**: Optionally, categorize customers based on their RFM scores using the `setCategory()` method.
4. **Visualize Segments**: Generate a pie chart showing the distribution of customer segments using the `plotPie()` method.
5. **Save Results**: Save the results to a CSV file using the `save()` method.

Example

```
# Initialize the Rfm class with the file path and additional parameters
# (Note: Column names, date format)
rfm = Rfm("dummy.csv",
        "Customer ID",
        "created_at",
        "grand_total",
        '%m/%d/%Y')

# Calculate RFM scores
rfm.setScore(1, 1, 1)

# Categorize customers based on RFM scores
rfm.setCategory(4, 3, 2, 1, 0)

# Plot a pie chart of customer segments
rfm.plotPie()

# Save the results to a CSV file
rfm.save("rfmresults.csv")
```

Methods

- `setScore(r=0.15, f=0.28, m=0.57)`: Calculates the RFM scores based on recency, frequency, and monetary values.
- `setCategory(a=4.5, b=4, c=3, d=1.5, r=30)`: Categorizes customers into segments based on RFM scores.
- `plotPie()`: Generates a pie chart of customer segments.
- `save(path)`: Saves the results to a CSV file.

Parameters

- `filepath`: Path to the CSV file containing customer data.
- `cc`: Column name for customer code.
- `od`: Column name for order date.
- `a`: Column name for amount.
- `format`: Format of the order date column. (Sample: 12/30/22 13:40:23 = '%m/%d/%Y %H:%M:%S')
- `r`, `f`, `m`: Weighting factors for recency, frequency, and monetary values, respectively, used in calculating the RFM scores. Defaults are 0.15, 0.28, and 0.57, respectively.
- `a`, `b`, `c`, `d`, `r`: Threshold values for categorizing customers into segments. These thresholds define the boundaries between different customer segments. By default, customers are categorized as follows:
  - `a+`: Top Customers

  - `b+`: High Value Customers
  - `c+`: Medium Value Customers
  - `d+`: Low Value Customers
  - `default`: a=4.5, b=4, c=3, d=1.5, r=30
  - `r`: Recency threshold for categorizing Lost Customers and New Customers. Default is set to 30 days.

Attributes

- `df`: DataFrame containing the customer data. After data cleaning processes, RFM scores and customer segmentation information are stored in this DataFrame.

Detailed Usage

1. **File Path and Parameters**: When initializing the `Rfm` class, provide the path to the CSV file containing customer data, along with the names of the customer code, order date, and amount columns, and the date format.

   ```
   rfm = Rfm("customer_data.csv",
        "Customer_ID",
        "Order_Date",
        "Amount",
        '%d/%m/%Y')
   ```

2. **Calculating RFM Scores**: The `setScore()` method calculates RFM scores based on recency, frequency, and monetary values. This method computes an RFM score for each customer using a weighted average.

   ```
   rfm.setScore(0.15, 0.28, 0.57)
   ```

3. **Customer Segmentation**: The `setCategory()` method categorizes customers into different segments based on their RFM scores. This method categorizes customers according to specific threshold values.

   ```
   rfm.setCategory(4.5, 4, 3, 1.5, 30)
   ```

4. **Visualization**: The `plotPie()` method generates a pie chart that shows the distribution of customer segments. This chart visually represents the proportional distribution of customer segments.

   ```
   rfm.plotPie()
   ```

5. **Saving Results**: The `save()` method saves the results of the RFM analysis to a specified file path as a CSV file. This facilitates the storage of analysis results for later use or review.

   ```
   rfm.save("rfm_results.csv")
   ```

This documentation provides a general guide on how to use the `Rfm` class. In

practical applications, parameters and methods should be adjusted according to the characteristics of the data set and business requirements.