فرهاد امان ۹۹۳۱۰۰۶

در این تحلیل داده، ما یک مجموعه داده از شهرهای مختلف جهان را مورد بررسی قرار می‌دهیم. هدف اصلی ما ارائه یک تحلیل جامع از ویژگی‌ها و ویژگی‌های جمعیتی این شهرهاست. در این گزارش، مراحلی که برای پیش‌پردازش و تحلیل داده‌ها انجام دادیم توضیح داده خواهد شد.

ابتدا کتابخانه‌های مورد نیاز را وارد می‌کنیم و dataframe را از فایل اکسل می‌خوانیم و داخل df می‌ریزیم

```
In 31  1  import pandas as pd
       2  import numpy as np
       3  import matplotlib.pyplot as plt
       4  import math
          Executed at 2023.10.31 15:39:18 in 5s 52ms

In 32  1  df = pd.read_excel('worldcities.xlsx')
       2  df.head()
```

پیش‌پردازش داده

حذف ستون‌ها: در این مرحله، ابتدا ستون‌های "admin_nom", "id", "capital", "ville_ascii" که اطلاعات مورد نظر ما نیستند را از داده‌ها حذف می‌کنیم.

```
In 33  1  df.drop(columns=['capital', 'id', 'admin_nom', 'ville_ascii'], inplace=True)
       2  df = df.rename(columns={'ville': 'city', 'pays': 'country'})
       3  df.head()
          Executed at 2023.10.31 15:40:11 in 9ms
```

حذف شهرهای با جمعیت کمتر از یک میلیون نفر: در این مرحله، شهرهایی که جمعیت آنها کمتر از یک میلیون نفر است را حذف می‌کنیم تا تنها شهرهای پرجمعیت در داده‌های ما باقی بمانند.

```
In 34  1  df = df.query('population >= 1000000')
       2  df.head()
Executed at 2023.10.31 15:40:15 in 59ms
```

تبدیل نوع داده‌ها: جمعیت شهرها به عدد صحیح تبدیل می‌شود و سپس میانگین مختصات عددی برای شهرهایی که مختصات دقیق ندارند محاسبه می‌شود.

```
In 35  1  df['population'] = df['population'].astype('int32')
       2  df.head()
Executed at 2023.10.31 15:40:17 in 5ms
```

```
In 38  1  df.drop_duplicates(inplace=True)
       2  df = df[df.isnull().sum(axis=1) < 2]
       3  print("Number of Rows: ", len(df))
       4  df.head()
Executed at 2023.10.31 15:46:21 in 45ms
```

```
In 39  1  mean_lat_df = pd.pivot_table(df, index=['country'], values=['lat'], aggfunc='mean')
       2  mean_lng_df = pd.pivot_table(df, index=['country'], values=['lng'], aggfunc='mean')
       3
       4  for c in mean_lat_df.index:
       5      df[df.country == c].fillna(mean_lat_df.loc[c, 'lat'])
       6  for c in mean_lng_df.index:
       7      df[df.country == c].fillna(mean_lng_df.loc[c, 'lng'])
       8  print(len(df))
       9  df.head()
Executed at 2023.10.31 15:46:23 in 94ms
```

محاسبه فاصله تا تهران: با استفاده از تابع Haversine، فاصله هر شهر از مرکز شهر تهران محاسبه می‌شود و به یک ستون جدید به نام "tehran_distance" اضافه می‌شود.

In 9

```python
def degree_to_radian(degree):
    return degree * (math.pi / 180)

def haversine_distance(lat1, lng1, lat2, lng2):
    lat1_radian = degree_to_radian(lat1)
    lng1_radian = degree_to_radian(lng1)
    lat2_radian = degree_to_radian(lat2)
    lng2_radian = degree_to_radian(lng2)

    d_lng = lng2_radian - lng1_radian
    d_lat = lat2_radian - lat1_radian
    a = np.sin(d_lat / 2) ** 2 + np.cos(lat1_radian) * np.cos(lat2_radian) *
     np.sin(d_lng / 2) ** 2

    d = 6371 * 2 * np.arctan2(np.sqrt(a), np.sqrt(1 - a))
    return d

```

Executed at 2023.10.29 20:25:40 in 26ms

In 40

```python
tehran_lat = 35.6892
tehran_lng = 51.3889
df['tehran_distance'] = df.apply(
    lambda row: haversine_distance(tehran_lat, tehran_lng, row['lat'], row['lng']),
     axis=1)

df.head()
```

Executed at 2023.10.31 16:10:02 in 77ms

تحلیل داده

مرتب‌سازی بر اساس ویژگی‌ها: داده‌ها بر اساس ویژگی‌های انتخابی مانند جمعیت، مساحت یا فاصله تا تهران مرتب می‌شوند. این امکان به ما می‌دهد تا شهرها را بر اساس این ویژگی‌ها صعودی یا نزولی مرتب کنیم.

In 46

```python
df = df.sort_values(by=['city', 'lat'], ascending=[True, False])
df.head()
```

Executed at 2023.10.31 17:15:10 in 97ms

```
In 47  1  df.to_csv('9931006.csv', index=False)
          Executed at 2023.10.31 17:15:38 in 61ms
```

نتایج و نمودارها

نمودار میله‌ای از 10 شهر با بیشترین جمعیت: در این نمودار، 10 شهر با بیشترین جمعیت نمایش داده می‌شوند تا بتوانیم برترین شهرها از نظر جمعیت را مشاهده کنیم.

```
In 49  1  smallest_distances = df.nsmallest(11, 'tehran_distance')
       2  smallest_distances = smallest_distances[smallest_distances['city'] != 'Tehran']
       3
       4  plot_1 = smallest_distances.plot.bar(x='city', y='tehran_distance', rot=0,
           figsize=(10, 6))
       5  plt.xlabel('City')
       6  plt.ylabel('Distance from Tehran (Kilometer)')
       7  plt.title('Top 10 Cities with Smallest Tehran Distance')
       8  plt.savefig('plot_1.png', bbox_inches='tight')
          Executed at 2023.10.31 17:19:14 in 307ms
```

نمودار پراکندگی جمعیت در کشورهای مختلف: در این نمودار، توزیع جمعیت در کشورهای مختلف به شکل پراکندگی نمایش داده می‌شود تا ببینیم کدام کشورها دارای شهرهای پرجمعیت‌تری هستند.

```
In 50  1  plot_2 = smallest_distances.plot.bar(x='city', y='population', rot=0,  figsize=(10,
           6))
       2  plt.xlabel('City')
       3  plt.ylabel('Population')
       4  plt.title('Top 10 Cities with Smallest Tehran Distance Population')
       5  plt.savefig('plot_2.png', bbox_inches='tight')
          Executed at 2023.10.31 17:19:55 in 307ms
```

نمودار میله‌ای از 10 شهر با بیشترین مساحت: در این نمودار، 10 شهر با بیشترین مساحت نمایش داده می‌شوند تا ببینیم کدام شهرها دارای بزرگ‌ترین مساحت هستند.

```
In 51
plt.figure(figsize=(10, 6))
plt.scatter(df['lat'], df['lng'], marker='o', s=50, c='blue', alpha=0.5)
plt.xlabel('Latitude')
plt.ylabel('Longitude')
plt.title('Cities Scatter Plot')
plt.grid(True)
plt.savefig('plot_3.png', bbox_inches='tight')
Executed at 2023.10.31 17:21:03 in 273ms
```

نتیجه‌گیری: در این تحلیل داده، ما با استفاده از انجام مراحل پیش‌پردازش و تحلیل دقیق داده‌ها، به نتایج جالبی در مورد شهرهای جهان رسیدیم. این تحلیل می‌تواند به محققان، برنامه‌ریزان شهری و افراد مختلف دیگر کمک کند تا درک بهتری از ویژگی‌ها و ویژگی‌های جمعیتی شهرها داشته باشند و تصمیم‌گیری‌های بهتری انجام دهند.