Bike_sharing_Analysis_Poject_2

May 25, 2018

```
In [49]: import time
         import pandas as pd
         import numpy as np

         #CITY_DATA = { 'chicago': 'chicago.csv', 'new york city': 'new_york_city.csv', 'washi

         CITY_DATA = { 'chicago': 'C:/Users/Jisnu/Dataset/chicago.csv', 'new york city': 'C:/U
                       'washington': 'C:/Users/Jisnu/Dataset/washington.csv'}

In [50]: def checking_in_list(x,y):
             if x not in y:
                 print('\n Incorrect data. Please restart your program \n')

In [1]: def get_filters():
            """
            Asks user to specify a city, month, and day to analyze.

            Returns:
                (str) city - name of the city to analyze
                (str) month - name of the month to filter by, or "all" to apply no month filte
                (str) day - name of the day of week to filter by, or "all" to apply no day fil
            """
            print('Hello! Let\'s explore some US bikeshare data!')
            # user input for city (chicago, new york city, washington). HINT: Use a while loop
            city = input('\n Would you like to see data for Chicago, New York city or Washingt
            city_list = [ 'chicago', 'new york city', 'washington']
            checking_in_list(city,city_list)


            # user input for month (all, january, february, ... , june)
            month = input('\n Name of the month to filter by? January, February, March, April,
            month_list = ['january', 'february', 'march','april', 'may', 'june','all']
            checking_in_list(month,month_list)

            # user input for day of week (all, monday, tuesday, ... sunday)
            day = input('\n Name of the day of week to filter by? Monday, Tuesday, Wednesday, 
                        'Sunday or all.\n').lower()
            day_list = ['Monday','Tuesday','Wednesday','Thursday','Friday','Saturday','Sunday'
            checking_in_list(day,day_list)

            print('-'*40)
            return city, month, day
```

```
In [2]: def load_data(city, month, day):
            """
            Loads data for the specified city and filters by month and day if applicable.

            Args:
                (str) city - name of the city to analyze
                (str) month - name of the month to filter by, or "all" to apply no month filte
                (str) day - name of the day of week to filter by, or "all" to apply no day fil
            Returns:
                df - Pandas DataFrame containing city data filtered by month and day
            """

            df = pd.read_csv(CITY_DATA[city])

            # Start Time column to datetime
            df['Start Time'] = pd.to_datetime(df['Start Time'])

            # extracting month and day of week from Start Time
            df['month'] = df['Start Time'].dt.month
            df['day_of_week'] = df['Start Time'].dt.weekday_name

            # filtering by month
            if month != 'all':
                # using the index of the months list to get the corresponding int
                months = ['january', 'february', 'march', 'april', 'may', 'june']
                month = months.index(month) + 1

                # filtering by month
                df = df[df['month'] == month]
            else:
                df = df

            # filtering by day of week
            if day != 'all':
                # filtering by day of week
                df = df[df['day_of_week'] == day.title()]

            else:
                df = df
            return df
```

```
In [3]: def time_stats(df):
            """Displays statistics on the most frequent times of travel."""
            print('\nCalculating The Most Frequent Times of Travel...\n')
            start_time = time.time()

            # the most common month
            most_common_month = df.month.value_counts().idxmax()
            print('The most common month is {}'.format(most_common_month))

            # the most common day of week
            most_common_week = df.day_of_week.value_counts().idxmax()
            print('The most common day of the week is {}'.format(most_common_week))

            # the most common start hour

            most_common_hour = df['Start Time'].dt.hour.mode()[0]
            print('The most common day of the hour is {}'.format(most_common_hour))

            print("\nThis took %s seconds." % (time.time() - start_time))
            print('-'*40)

In [4]: def station_stats(df):
            """Displays statistics on the most popular stations and trip."""

            print('\nCalculating The Most Popular Stations and Trip...\n')
            start_time = time.time()

            # most commonly used start station
            print("The most frequent start station is:",df.groupby('Start Station')['Start Sta

            # most commonly used end station
            print("The most frequent end station is:",df.groupby('End Station')['End Station']

            # most frequent combination of start station and end station trip
            print("The most common trip is:",df.groupby(['Start Station','End Station']).size(

            print("\nThis took %s seconds." % (time.time() - start_time))
            print('-'*40)

In [5]: def trip_duration_stats(df):
            """Displays statistics on the total and average trip duration."""

            print('\nCalculating Trip Duration...\n')
            start_time = time.time()

            # display total travel time
            print("the sum of total trip duration is {} days".format(round(df['Trip Duration']

            # display mean travel time
            print("the average of total trip duration is {} mins".format(df['Trip Duration'].me

            print("\nThis took %s seconds." % (time.time() - start_time))
            print('-'*40)

In [6]: def user_stats(df):
            """Displays statistics on bikeshare users."""

            print('\nCalculating User Stats...\n')
            start_time = time.time()

            # counts of user types
            print('The ratio of user type is : \n',df['User Type'].value_counts())

            # counts of gender
            if 'Gender' in df:
                print('The gender balance is {}'.format(df.Gender.value_counts()))
            else:
                print("The Gender data is not available for this dataset!")

            # earliest, most recent, and most common year of birth
            if 'Birth Year' in df:
                print("The oldest year of birth is:",df['Birth Year'].min())
                print("The youngest year of birth is:",df['Birth Year'].max())
                print("The most common year of birth is:",df['Birth Year'].value_counts().idxma

            else:
                print("The year's data is not available for this dataset!")

            print("\nThis took %s seconds." % (time.time() - start_time))
            print('-'*40)

In [57]: def need_raw_data(df):
             raw_data = input("\n Would you like to see the raw data?Type Yes or No.\n").lower

             if(raw_data =='yes'):
                 print(df.head())

In [58]: def main():
             while True:
                 city, month, day = get_filters()
                 df = load_data(city, month, day)

                 time_stats(df)
                 station_stats(df)
                 trip_duration_stats(df)
                 user_stats(df)
                 need_raw_data(df)
                 restart = input('\n Would you like to restart? Enter yes or no.\n').lower()
                 if restart.lower() != 'yes':
                     break

         if __name__ == "__main__":
                 main()
```

```
Hello! Let's explore some US bikeshare data!

 Would you like to see data for Chicago, New York city or Washington?
WASHINGTON

 Name of the month to filter by? January, February, March, April, May, June or All?
June

 Name of the day of week to filter by? Monday, Tuesday, Wednesday, Thursday, Friday, Saturday,
all
----------------------------------------

Calculating The Most Frequent Times of Travel...

The most common month is 6
The most common day of the week is Thursday
The most common day of the hour is 8

This took 0.016043186187744140 seconds.
----------------------------------------

Calculating The Most Popular Stations and Trip...

The most frequent start station is: Lincoln Memorial
The most frequent end station is: Jefferson Dr & 14th St SW
The most common trip is: ('Jefferson Dr & 14th St SW', 'Jefferson Dr & 14th St SW')

This took 0.035092830657958984 seconds.
----------------------------------------

Calculating Trip Duration...

the sum of total trip duration is 1084 days
the average of total trip duration is 22.841529177092738 mins

This took 0.001003265380859375 seconds.
----------------------------------------

Calculating User Stats...
```

```
The ratio of user type is :
 Subscriber    47727
Customer      20612
Name: User Type, dtype: int64
The Gender data is not available for this dataset!
The year's data is not available for this dataset!

This took 0.009236574172973633 seconds.
----------------------------------------

 Would you like to see the raw data?Type Yes or No.
NO

 Would you like to restart? Enter yes or no.
NO
```