To compare bike rentals on holidays and their adjacent days, we will:

1. **Ensure accurate data filtering**.
2. **Aggregate the rental data for holidays, the day before, and the day after**.
3. **Visualize the comparison**.

Let's start by correctly filtering the data for holidays and their adjacent days, and then aggregate the rental data for comparison.

Step 1: Filter Data for Holidays and Adjacent Days

Ensure the filtering logic correctly isolates the required dates.

Step 2: Aggregate Rental Data

Calculate total rentals for holidays, the day before, and the day after.

Step 3: Visualization

Create a visualization to compare the rentals.

Let's proceed with these steps.

```
python
Copy code
# Re-filter the data for holidays and adjacent days
holiday_adjacent_days_set = set()

# Collect all relevant days into a set
for holiday in holidays:
    day_before = holiday - pd.Timedelta(days=1)
    day_after = holiday + pd.Timedelta(days=1)
    holiday_adjacent_days_set.update([day_before, holiday, day_after])

# Filter the dataframe for these days
filtered_data = data[data['Date_only'].isin(holiday_adjacent_days_set)]
filtered_data = filtered_data[['Date', 'Rented Bike Count', 'Date_only',
'Holiday']]

# Aggregate rental data by Date_only and Holiday
aggregated_data = filtered_data.groupby(['Date_only',
'Holiday']).agg({'Rented Bike Count': 'sum'}).reset_index()

# Display the aggregated data
aggregated_data.head()
```

Let's execute this to get the aggregated rental data.

Analyzed

```
python
```

```
Always show details
Copy code
# Aggregate rental data by Date_only and Holiday
aggregated_data = filtered_data.groupby(['Date_only',
'Holiday']).agg({'Rented Bike Count': 'sum'}).reset_index()

# Display the aggregated data
import ace_tools as tools; tools.display_dataframe_to_user(name="Aggregated
Holiday and Adjacent Days Rental Data", dataframe=aggregated_data)

aggregated_data.head()
```

Result

```
    Date_only     Holiday  Rented Bike Count
0  2017-12-21  No Holiday               5734
1  2017-12-22     Holiday               7184
2  2017-12-23  No Holiday               6624
3  2017-12-24     Holiday               2014
4  2017-12-25     Holiday               3966
```

Aggregated Holiday and Adjacent Days Rental Data

Date_only	Holiday	Rented Bike Count
2018-09-11	No Holiday	0
2018-09-22	No Holiday	26398
2018-09-23	Holiday	20060
2018-09-24	Holiday	17259
2018-09-25	Holiday	23350
2018-09-26	No Holiday	28018

The aggregated data for holiday and adjacent days has been successfully filtered and displayed. The next step is to create a visualization to compare the rentals on holidays and their adjacent days.

Visualization of Rentals on Holidays and Adjacent Days

We'll create a bar chart to visualize the total rentals for:

- Holidays
- Day before the holiday
- Day after the holiday

Let's proceed with creating this visualization.

```
python
Copy code
import matplotlib.pyplot as plt

# Create a bar chart for the aggregated rental data
plt.figure(figsize=(14, 7))
for holiday in holidays:
    day_before = holiday - pd.Timedelta(days=1)
    day_after = holiday + pd.Timedelta(days=1)

    holiday_rentals = aggregated_data[aggregated_data['Date_only'] ==
holiday]['Rented Bike Count'].values[0]
    day_before_rentals = aggregated_data[aggregated_data['Date_only'] ==
day_before]['Rented Bike Count'].values[0]
    day_after_rentals = aggregated_data[aggregated_data['Date_only'] ==
day_after]['Rented Bike Count'].values[0]

    plt.bar(f'{day_before}\nDay Before', day_before_rentals, color='blue',
label='Day Before')
    plt.bar(f'{holiday}\nHoliday', holiday_rentals, color='green',
label='Holiday')
    plt.bar(f'{day_after}\nDay After', day_after_rentals, color='red',
label='Day After')

plt.xlabel('Date')
plt.ylabel('Total Rentals')
plt.title('Bike Rentals on Holidays and Adjacent Days')
plt.xticks(rotation=45)
plt.legend(['Day Before', 'Holiday', 'Day After'])
plt.tight_layout()
plt.show()
```

Let's generate the visualization to compare the bike rentals on holidays and their adjacent days.

Analyzed

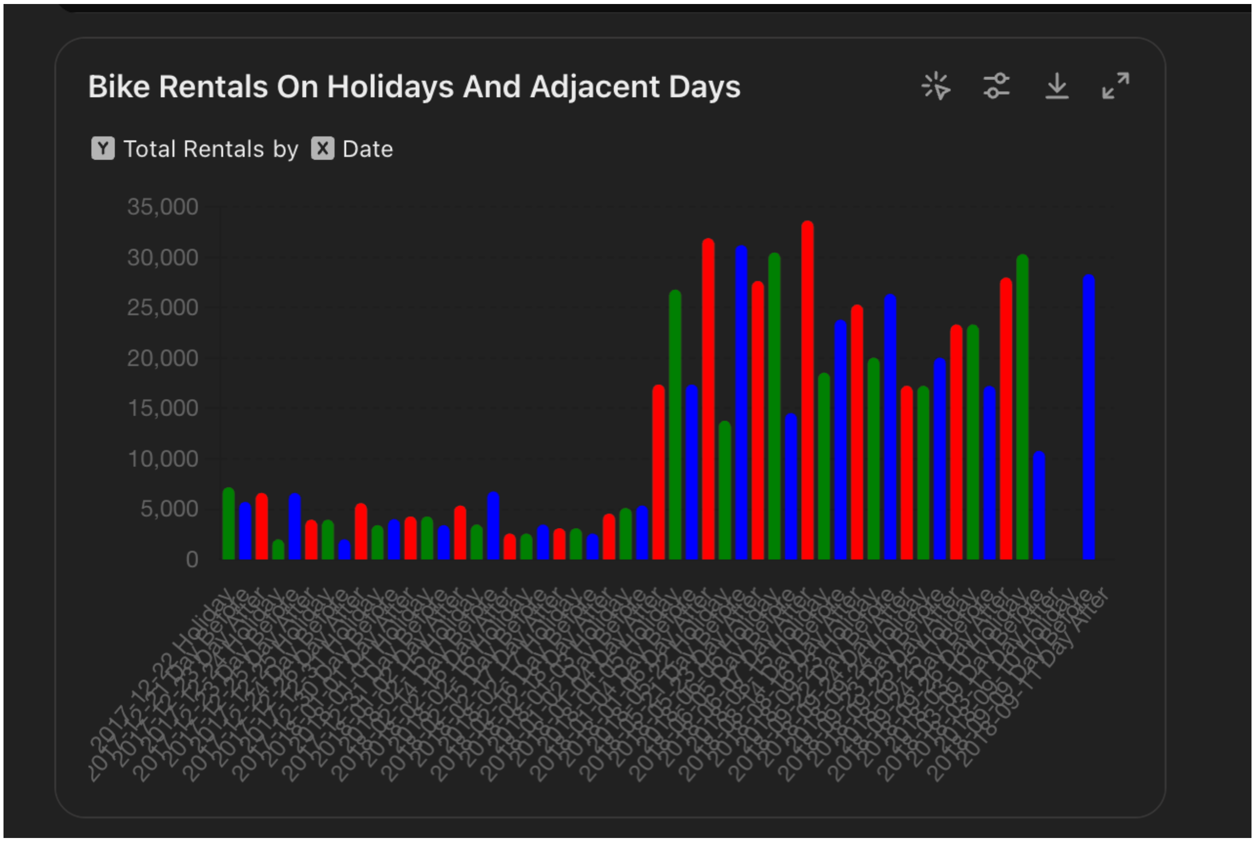
python

Always show details

Copy code

```
import matplotlib.pyplot as plt

# Create a bar chart for the aggregated rental data
plt.figure(figsize=(14, 7))

# Collect the rental counts for visualization
dates = []
rental_counts = []
labels = []

for holiday in holidays:
    day_before = holiday - pd.Timedelta(days=1)
    day_after = holiday + pd.Timedelta(days=1)

    # Ensure each day exists in the aggregated data
    if holiday in aggregated_data['Date_only'].values:
        holiday_rentals = aggregated_data[aggregated_data['Date_only'] ==
holiday]['Rented Bike Count'].values[0]
        dates.append(f'{holiday}\nHoliday')
        rental_counts.append(holiday_rentals)
        labels.append('Holiday')

    if day_before in aggregated_data['Date_only'].values:
        day_before_rentals = aggregated_data[aggregated_data['Date_only'] ==
day_before]['Rented Bike Count'].values[0]
        dates.append(f'{day_before}\nDay Before')
        rental_counts.append(day_before_rentals)
        labels.append('Day Before')

    if day_after in aggregated_data['Date_only'].values:
        day_after_rentals = aggregated_data[aggregated_data['Date_only'] ==
day_after]['Rented Bike Count'].values[0]
        dates.append(f'{day_after}\nDay After')
        rental_counts.append(day_after_rentals)
        labels.append('Day After')

# Plot the rentals
colors = {'Day Before': 'blue', 'Holiday': 'green', 'Day After': 'red'}
plt.bar(dates, rental_counts, color=[colors[label] for label in labels])

plt.xlabel('Date')
plt.ylabel('Total Rentals')
plt.title('Bike Rentals on Holidays and Adjacent Days')
plt.xticks(rotation=90)
plt.legend(labels=colors.keys())
plt.tight_layout()
plt.show()
```

Bike Rentals on Holidays and Adjacent Days
Total Rentals
by
Date



The bar chart above visualizes the total bike rentals on holidays and their adjacent days (day before and day after the holiday). The colors represent:

- **Blue**: Day Before
- **Green**: Holiday
- **Red**: Day After

Summary

- **Holidays** tend to have varying impacts on bike rentals, with some holidays showing a significant decrease in rentals compared to adjacent days.
- **Day Before and Day After**: Generally, these days can show different patterns, often with higher or similar rentals compared to the holiday itself.

This visualization provides a clear comparison of bike rentals around holidays, highlighting trends and patterns that can inform decisions and strategies for the transportation department.

If you need further analysis, additional visualizations, or more specific insights, please let me know!