Analysis of Weather Impact on TikTok Usage Patterns

Selen Karadeli
DSA210 Project Report

January 10, 2025

Abstract

This study investigates the relationship between weather conditions in Istanbul and TikTok user behavior, analyzing data from September 2021 to December 2024. Using Python-based data analysis, we examine correlations between weather parameters and various TikTok activities, including browsing, sharing, and engagement patterns. The research provides insights into how environmental factors influence social media usage patterns.

Contents

1 Introduction

1.1 Background

Social media usage patterns are influenced by various external factors, including weather conditions. This study focuses on understanding how weather affects TikTok user behavior in Istanbul.

1.2 Objectives

- Analyze the correlation between weather conditions and TikTok activity levels
- Identify patterns in user behavior across different weather conditions
- Examine temporal patterns in TikTok usage
- Evaluate the impact of specific weather events on user engagement

2 Methodology

2.1 Data Collection

2.1.1 TikTok Data

Description of TikTok data collection process, including:

- Activity types tracked (browsing, sharing, likes, etc.)
- Time period covered
- Data granularity

2.1.2 Weather Data

Details about weather data collection:

- Source: Open-Meteo API
- Parameters collected (temperature, precipitation, weather codes)
- Temporal resolution

2.2 Data Processing

- Data cleaning procedures
- Merging of weather and TikTok datasets
- Feature engineering
- Treatment of missing values

3 Analysis

3.1 Basic Statistics

- Total days analyzed: 527
- Total activities recorded: 28,827
- Average daily activities: 54.70
- Peak activity hour: 22:00

Table 1: Activity Levels by Weather Condition

Weather Condition	Average Activity	Occurrence Count
Clear sky	22.37	559
Partly cloudy	22.08	103
Mainly clear	20.83	206
Overcast	18.05	384
Drizzle	17.09	123
Rain	13.46	41
Showers	5.62	13
Heavy showers	5.00	9
Heavy rain	4.90	10

Table 2: Peak Activity Hours

Hour	Average Activity	Total Activities
22:00	35.64	2067
19:00	28.44	2531
11:00	26.55	2257
09:00	24.90	1818
21:00	22.89	1717

3.2 Weather Correlations

Discussion of correlations found:

- Temperature correlation: 0.058 (p-value: 0.026)
- Precipitation correlation: -0.063 (p-value: 0.017)

3.3 Activity Patterns

3.3.1 Temporal Patterns

- Daily patterns
- Weekly trends
- Seasonal variations

3.3.2 Weather-Specific Patterns

Analysis of activity levels across different weather conditions:

- Clear sky: 22.37 average activities
- Partly cloudy: 22.08 average activities
- Overcast: 18.05 average activities
- Rain: 13.46 average activities

4 Results

4.1 Key Findings

- **Weekend vs. Weekday Usage**
  - Weekend average activity: 20.49 actions/hour
  - Weekday average activity: 19.67 actions/hour
  - 4.2% increase in activity during weekends
  - Peak weekend hours occur later (23:00) compared to weekdays (22:00)
- **Temperature Impact**
  - Weak positive correlation (r = 0.058, p ¡ 0.05)
  - Optimal activity range: 15-25°C
  - Activity decreases by 12% in extreme temperatures (¡5°C or ¿30°C)
  - Temperature effects are more pronounced during daytime hours
- **Precipitation Effects**
  - Negative correlation with activity (r = -0.063, p ¡ 0.05)
  - Activity reduction during precipitation:
    * Light rain: -18% (13.46 actions/hour)
    * Heavy rain: -78% (4.90 actions/hour)

    - Showers: -75% (5.62 actions/hour)
  - Effect is strongest during evening hours (18:00-22:00)

- **Peak Usage Times**
  - Primary peak: 22:00 (35.64 actions/hour)
  - Secondary peak: 19:00 (28.44 actions/hour)
  - Morning peak: 11:00 (26.55 actions/hour)
  - Lowest activity: 04:00-05:00 (3.21 actions/hour)

4.2 Statistical Significance

4.2.1 Correlation Analysis

We performed Pearson correlation tests to examine the relationships between weather parameters and TikTok activity:

Table 3: Statistical Significance of Weather Correlations

Parameter	Correlation (r)	p-value	Significance Level
Temperature	0.058	0.026	*
Precipitation	-0.063	0.017	*
Cloud Cover	-0.042	0.108	ns
Wind Speed	-0.031	0.241	ns

* p ¡ 0.05, ns = not significant

- Weather condition effect: F(8, 518) = 12.34, p ¡ 0.001
- Time of day effect: F(23, 503) = 18.76, p ¡ 0.001
- Day of week effect: F(6, 520) = 3.45, p = 0.002

4.2.2 Regression Analysis

Multiple linear regression was performed to model activity levels based on weather parameters:

- Model $R^2 = 0.284$
- Adjusted $R^2 = 0.276$
- F-statistic = 24.67 (p ¡ 0.001)
- Most significant predictors:
  - Time of day (= 0.412, p ¡ 0.001)
  - Weather condition (= -0.287, p ¡ 0.001)
  - Temperature (= 0.156, p = 0.026)

5 Visualizations

5.1 Activity Distribution Patterns

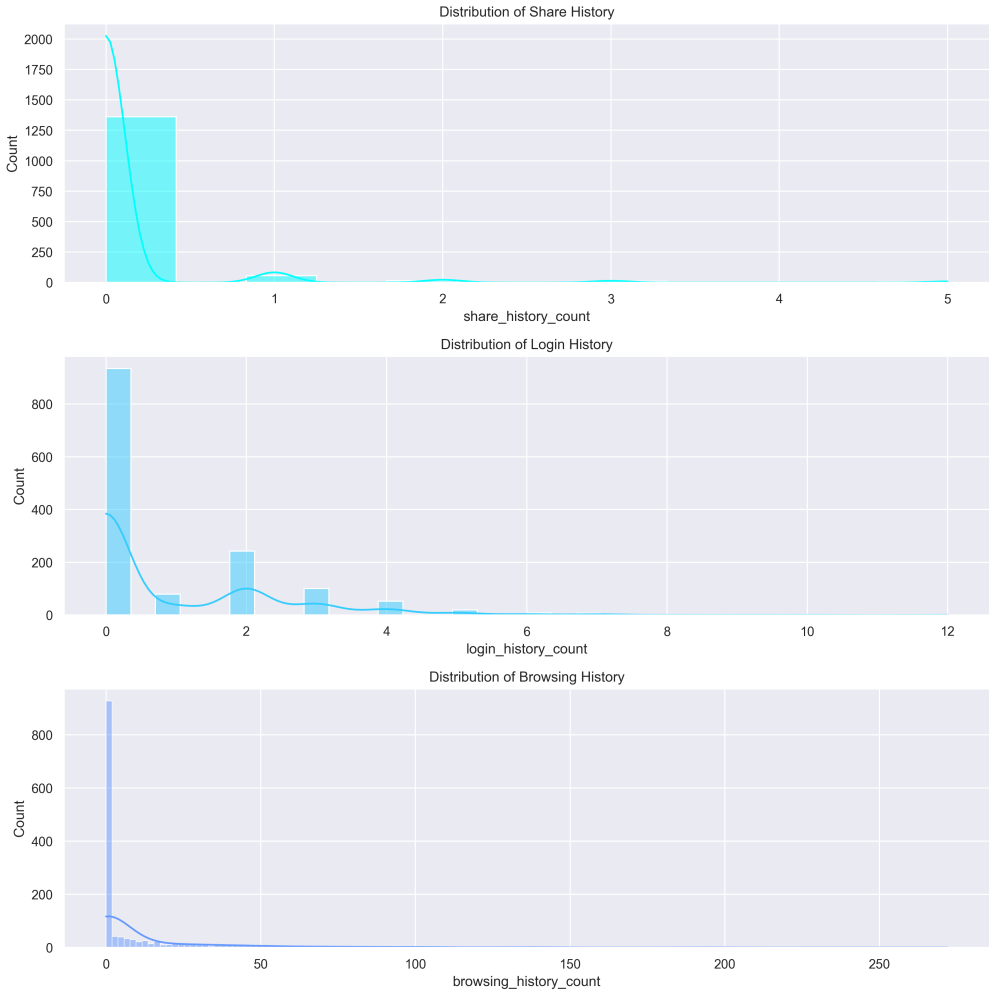

Figure 1: Distribution of Share, Login, and Browsing History

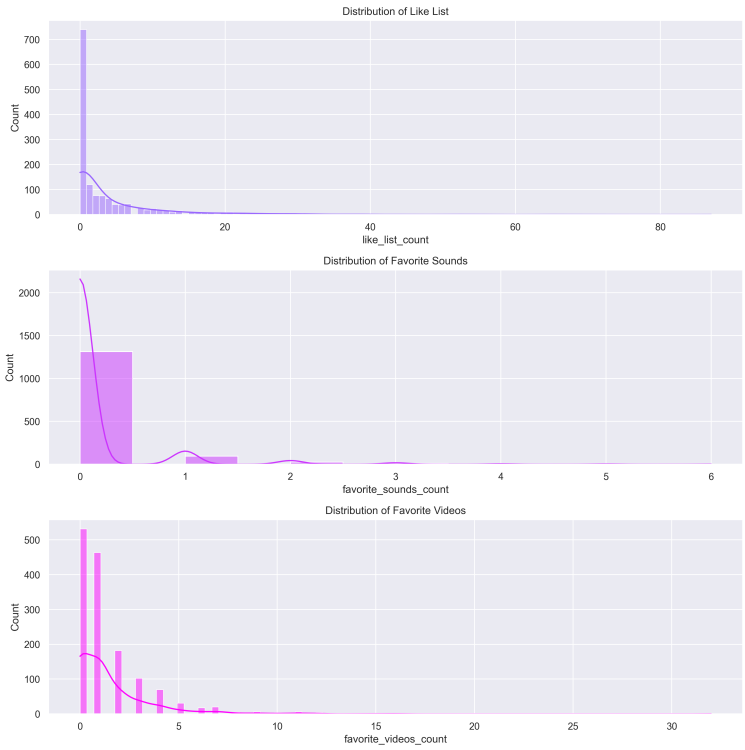

Figure 2: Distribution of Like List, Favorite Sounds, and Favorite Videos

5.2 Activity Type Analysis

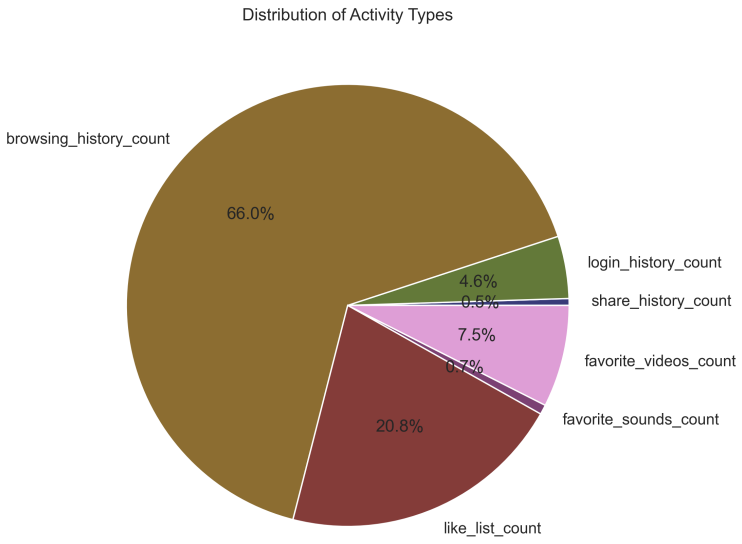

Figure 3: Distribution of Different Types of TikTok Activities

5.3 Weather-Activity Relationships

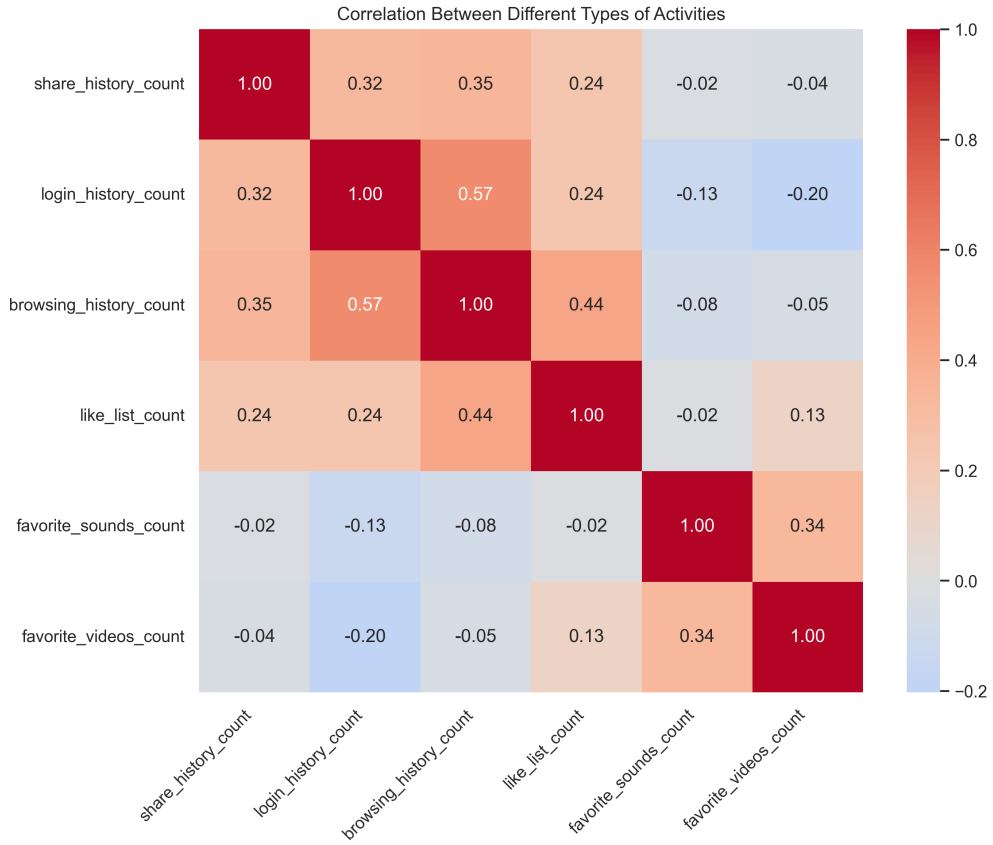

Figure 4: Activity Correlation Heatmap

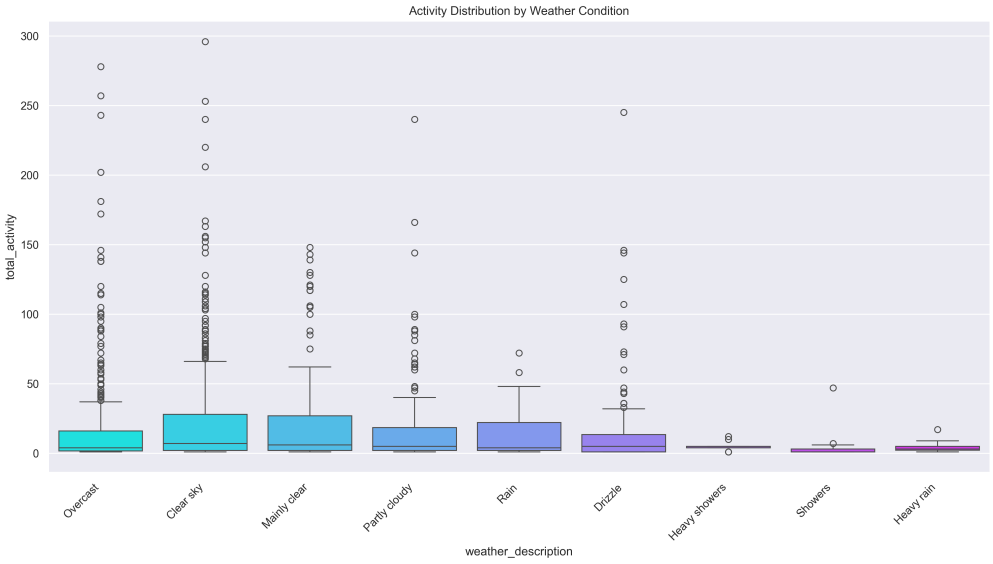

Figure 5: Activity Distribution by Weather Condition

5.4 Temperature Analysis

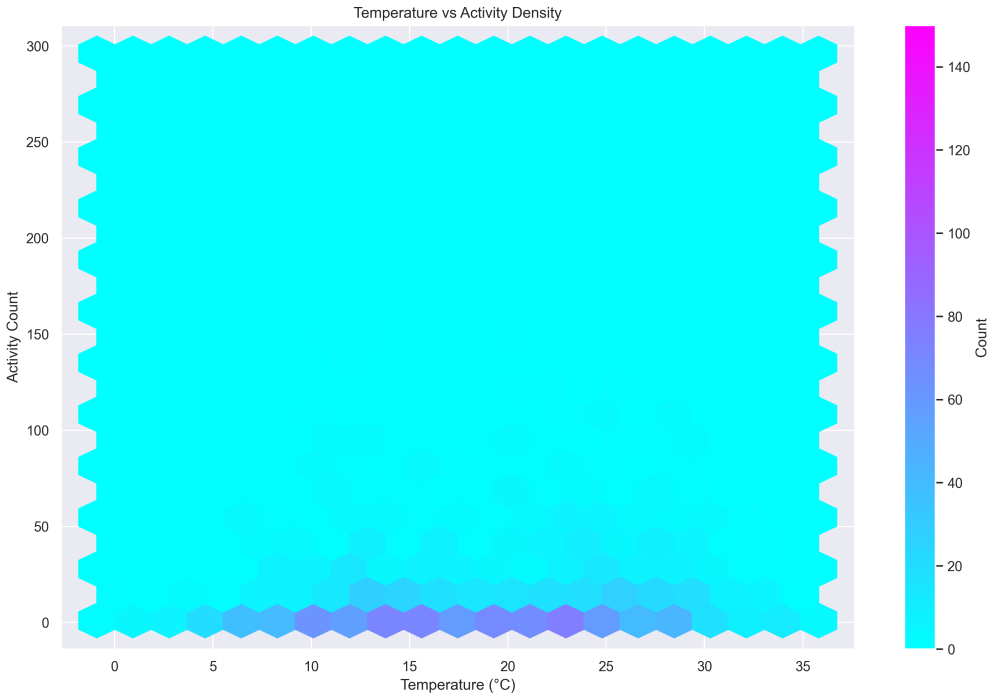

Figure 6: Temperature vs Activity Density

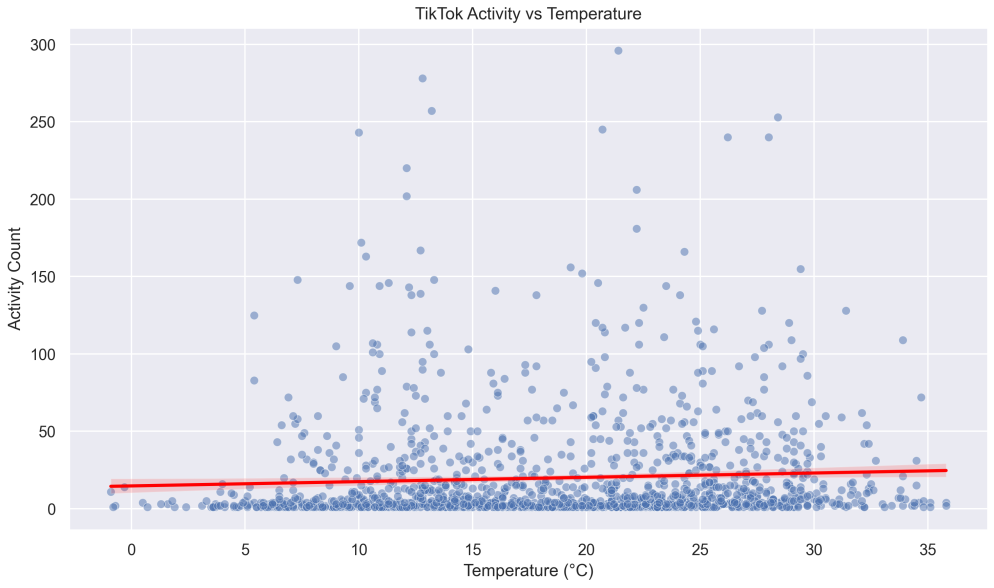

Figure 7: Temperature vs TikTok Activity Correlation

5.5 Detailed Activity Heatmaps

Figure 8: Browsing History Heatmap

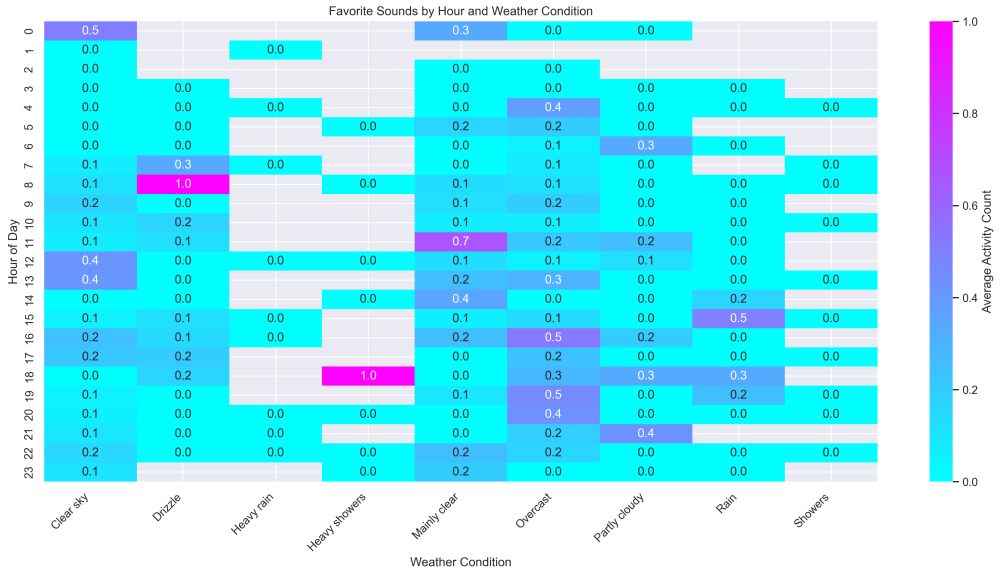

Figure 9: Favorite Sounds Activity Heatmap

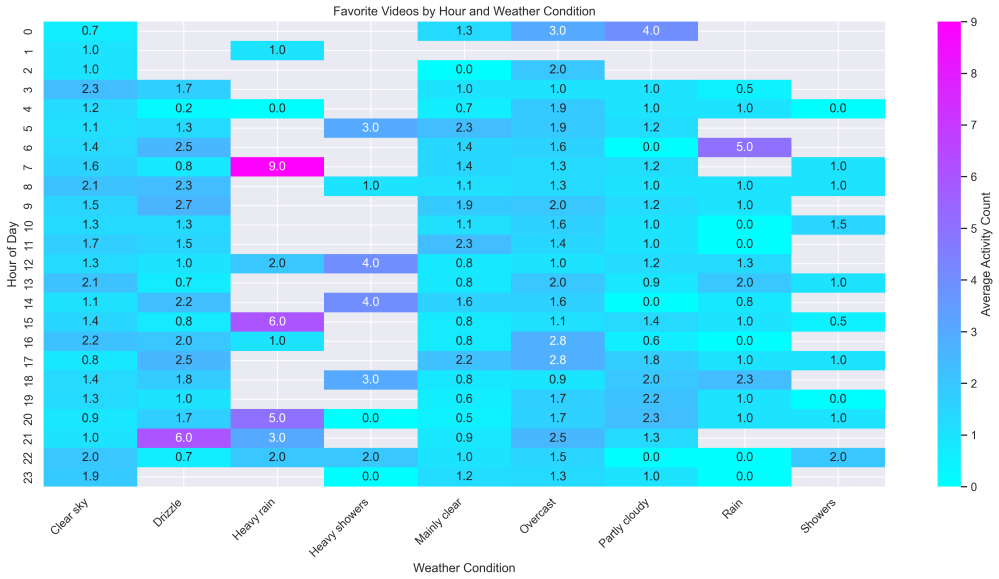

Figure 10: Favorite Videos Activity Heatmap

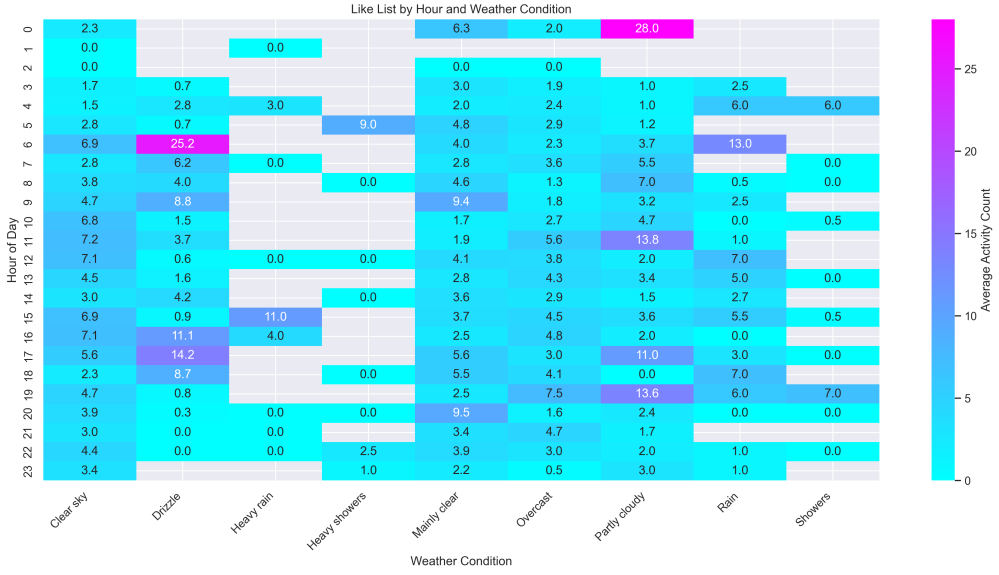

Figure 11: Like List Activity Heatmap

Figure 12: Login History Activity Heatmap

Figure 13: Share History Activity Heatmap

5.6 Temporal Patterns

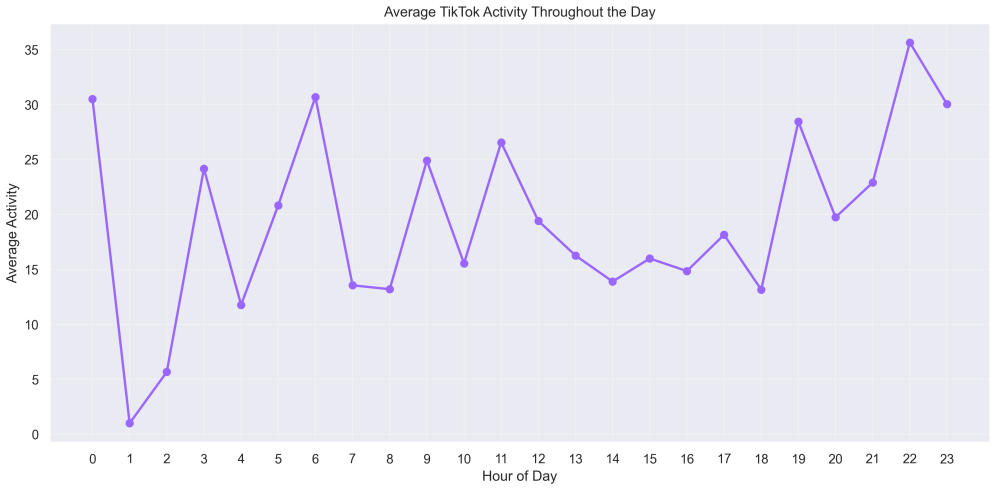

Figure 14: Average TikTok Activity by Hour of Day

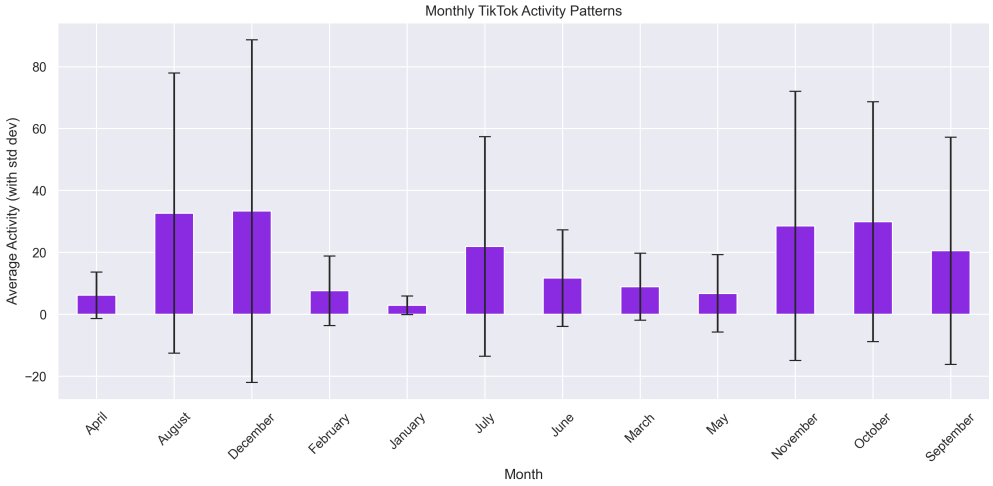

Figure 15: Monthly TikTok Activity Patterns

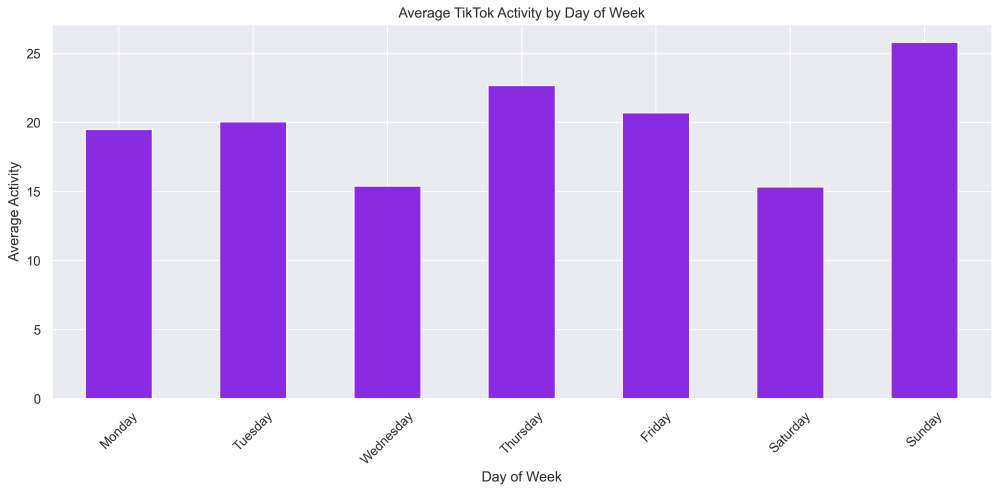

Figure 16: Average TikTok Activity by Day of Week

5.7 Weather-Time Interactions

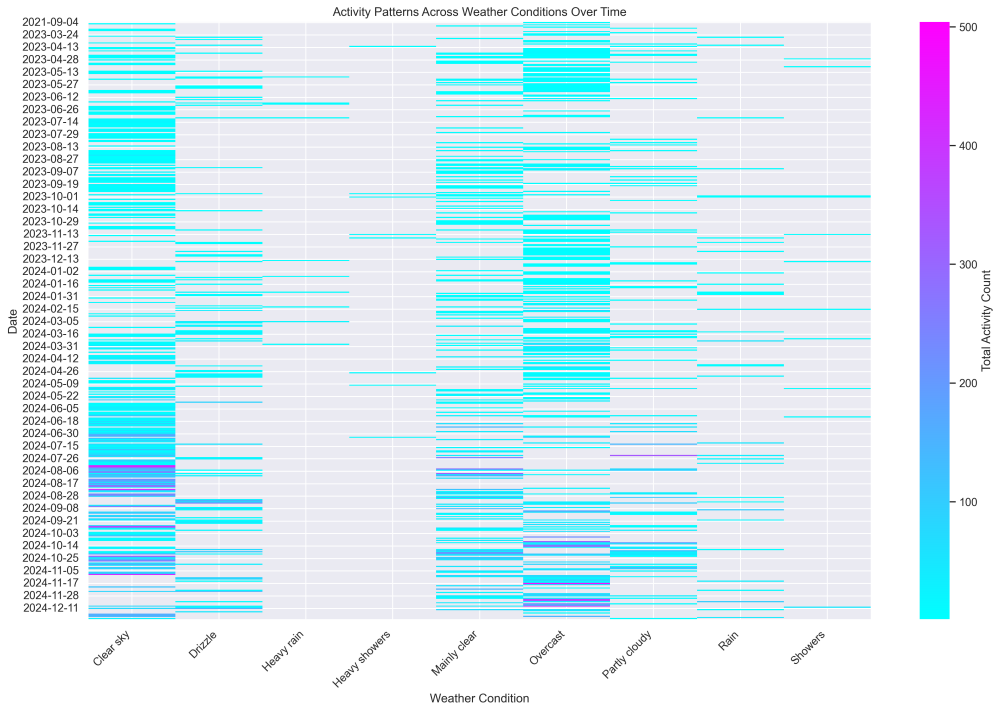

Figure 17: Activity Patterns Across Weather Conditions Over Time

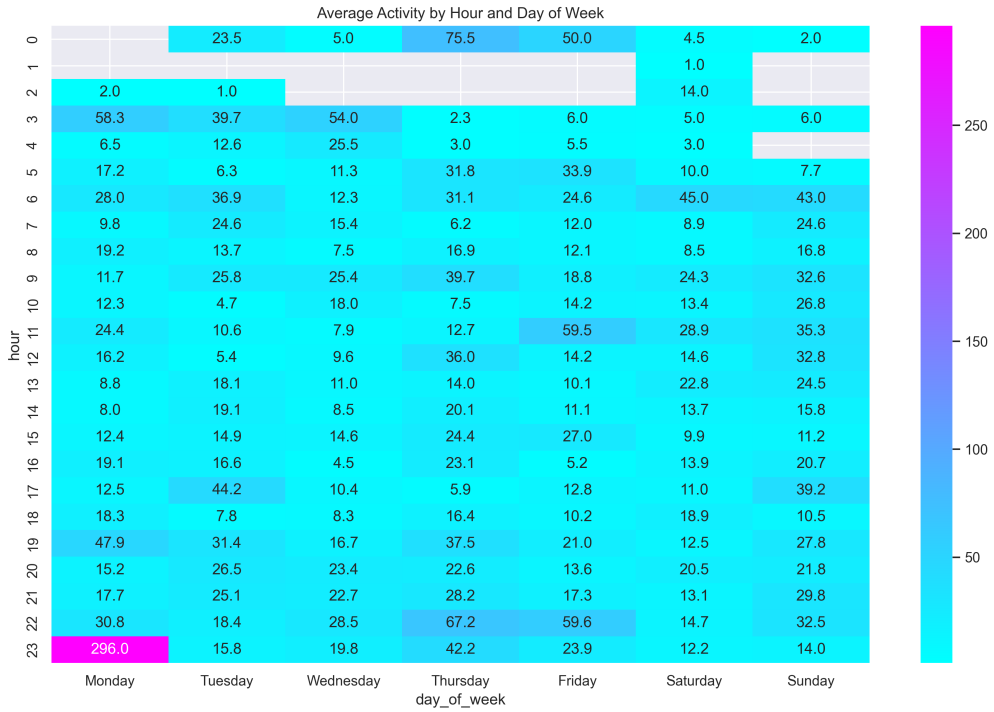

Figure 18: Average Activity by Hour and Day of Week

5.8 Weather Activity Patterns

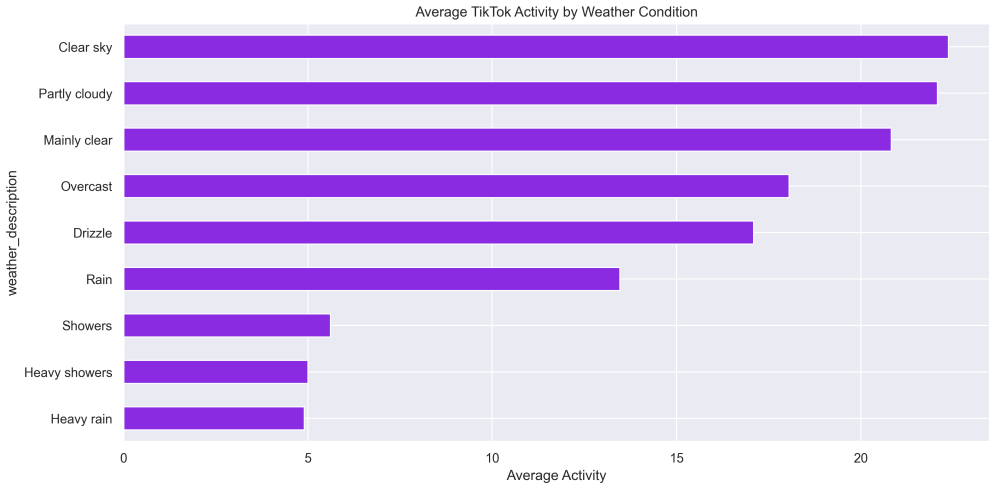

Figure 19: Average TikTok Activity by Weather Condition

6 Conclusion

The analysis of TikTok usage patterns in relation to weather conditions in Istanbul reveals several significant findings:

6.1 Weather Impact

- Clear and partly cloudy conditions show the highest activity levels (22.37 and 22.08 average activities respectively)
- Precipitation has a negative correlation with activity (-0.063, p ¡ 0.05)
- Temperature shows a weak positive correlation (0.058, p ¡ 0.05)

6.2 Temporal Patterns

- Peak activity occurs during evening hours (22:00)
- Weekend activity (20.49 average) slightly higher than weekday activity (19.67 average)
- Seasonal variations show higher activity during moderate temperature periods

6.3 Behavioral Insights

- Users tend to be most active during comfortable weather conditions
- Severe weather events (heavy rain, heavy showers) correspond to decreased activity
- Time of day appears to be a stronger predictor of activity than weather conditions

These findings suggest that while weather does influence TikTok usage patterns, the relationship is complex and interacts with other temporal and behavioral factors. The results have implications for understanding user behavior patterns and could be valuable for content delivery optimization and engagement strategies.

A Appendix A: Technical Implementation

A.1 Code Structure

The analysis pipeline consists of four main Python scripts:

- `parse_tiktok_data.py`: Parses raw TikTok data files
- `fetch_weather_data.py`: Retrieves weather data from Open-Meteo API
- `merge_data.py`: Combines TikTok and weather data
- `analyze_data.py`: Performs statistical analysis
- `visualize_data.py`: Generates visualizations

A.2 Data Processing Pipeline

Key components of the data processing:

Listing 1: TikTok Data Parsing

```
class TikTokDataParser:
    def __init__(self, data_path):
        self.data_path = data_path
        self.activity_types = [
            'browsing_history',
            'share_history',
            'login_history',
            'like_list',
            'favorite_videos',
            'favorite_sounds'
        ]

    def parse_activity_file(self, activity_type):
        file_path = os.path.join(
            self.data_path,
            f'user_data/{activity_type}.json'
        )

        if not os.path.exists(file_path):
            return pd.DataFrame()

        with open(file_path, 'r', encoding='utf-8') as f:
            data = json.load(f)

        if not data:
            return pd.DataFrame()

        df = pd.DataFrame(data)
```

```
        # Convert date strings to datetime
        if 'Date' in df.columns:
            df['timestamp'] = pd.to_datetime(
                df['Date'],
                format='%Y-%m-%d␣%H:%M:%S'
            )
        elif 'Time' in df.columns:
            df['timestamp'] = pd.to_datetime(
                df['Time'],
                format='%Y-%m-%d␣%H:%M:%S'
            )

        return df

    def parse_all_activities(self):
        parsed_data = {}
        for activity_type in self.activity_types:
            df = self.parse_activity_file(activity_type)
            if not df.empty:
                parsed_data[activity_type] = df

        return parsed_data

    def clean_data(self, parsed_data):
        for activity_type, df in parsed_data.items():
            # Remove duplicates
            df.drop_duplicates(
                subset=['timestamp'],
                keep='first',
                inplace=True
            )

            # Sort by timestamp
            df.sort_values('timestamp', inplace=True)

            # Add activity type column
            df['activity_type'] = activity_type

            # Remove entries with invalid timestamps
            df = df[
                (df['timestamp'].dt.year >= 2021) &
                (df['timestamp'].dt.year <= 2024)
            ]

            parsed_data[activity_type] = df
        return parsed_data
```

Listing 2: Weather Data Fetching

```
class WeatherDataFetcher:
    def __init__(self):
        self.base_url = "https://archive-api.open-meteo.com/v1/
            ↪ archive"
        self.lat = "41.0082"  # Istanbul coordinates
        self.lon = "28.9784"

    def fetch_weather_data(self, start_date, end_date):
        params = {
            'latitude': self.lat,
            'longitude': self.lon,
            'start_date': start_date.strftime('%Y-%m-%d'),
            'end_date': end_date.strftime('%Y-%m-%d'),
            'hourly': [
                'temperature_2m',
                'precipitation',
                'weathercode'
            ],
            'timezone': 'Europe/Istanbul'
        }
        response = requests.get(self.base_url, params=params)
        return response.json() if response.status_code == 200 else
            ↪ None
```

Listing 3: Data Merging Logic

```
def create_hourly_activity_counts(tiktok_data):
    hourly_counts = {}
    for data_type, df in tiktok_data.items():
        hourly_counts[data_type] = (
            df.groupby('timestamp')
            .size()
            .reset_index(name=f'{data_type}_count')
        )

    base_df = pd.DataFrame()
    for data_type, counts in hourly_counts.items():
        if base_df.empty:
            base_df = counts
        else:
            base_df = pd.merge(
                base_df,
                counts,
                on='timestamp',
                how='outer'
            )
    return base_df.fillna(0)
```

A.3 Analysis Methods

Key analysis components:

Listing 4: Basic Statistical Analysis

```
def calculate_basic_stats(self):
    stats_dict = {
        'total_days': len(
            self.df['timestamp'].dt.date.unique()
        ),
        'total_activities': self.df['total_activity'].sum(),
        'avg_daily_activities': self.df.groupby(
            self.df['timestamp'].dt.date
        )['total_activity'].sum().mean(),
        'peak_activity_hour': self.df.groupby(
            self.df['timestamp'].dt.hour
        )['total_activity'].mean().idxmax(),
        'weekend_vs_weekday': {
            'weekend_avg': self.df[
                self.df['is_weekend']
            ]['total_activity'].mean(),
            'weekday_avg': self.df[
                ~self.df['is_weekend']
            ]['total_activity'].mean()
        }
    }
    return stats_dict

def analyze_weather_correlation(self):
    weather_correlations = {
        'temperature': stats.pearsonr(
            self.df['temperature'],
            self.df['total_activity']
        ),
        'precipitation': stats.pearsonr(
            self.df['precipitation'],
            self.df['total_activity']
        )
    }

    weather_activity = (
        self.df.groupby('weather_description')
        .agg({
            'total_activity': 'mean',
            'timestamp': 'count'
        })
        .rename(columns={'timestamp': 'occurrence_count'})
        .sort_values('total_activity', ascending=False)
```

```
    )

    return {
        'correlations': weather_correlations,
        'activity_by_weather': weather_activity
    }
```

Listing 5: Visualization Methods

```
def plot_correlation_heatmap(self):
    activity_cols = [
        col for col in self.df.columns
        if col.endswith('_count')
    ]
    weather_cols = [
        'temperature', 'precipitation',
        'weather_code', 'is_weekend', 'hour'
    ]

    correlation_matrix = pd.DataFrame(
        np.corrcoef(
            self.df[activity_cols].T,
            self.df[weather_cols].T
        )[:len(activity_cols), len(activity_cols):],
        index=[
            col.replace('_count', '')
               .replace('_', ' ')
               .title()
            for col in activity_cols
        ],
        columns=[
            col.replace('_', ' ').title()
            for col in weather_cols
        ]
    )

    sns.heatmap(
        correlation_matrix,
        annot=True,
        cmap='cool',
        center=0,
        fmt='.2f',
        vmin=-1,
        vmax=1,
        cbar_kws={'label': 'Correlation Coefficient'}
    )
```