Multiple Linear Regression Analysis of Citi Bike Ridership Data

Javier Lopez

Western Governors University

Table of Contents

Research Question

This project aims to comprehensively analyze five years' worth of Citi Bike ridership data, in conjunction with weather patterns, a pandemic control variable, and US holidays data. The primary goal is to gain insights into the key factors that influence bike-sharing usage patterns in New York City and to predict future demand. The central research question that drives this endeavor is: " Can Citi Bike ridership demand be predicted using a multiple linear regression model based on market research data?"

The bike-sharing industry, much like scooter-sharing organizations, relies heavily on accurate demand predictions to efficiently redistribute inventory across the city and meet user needs. Notably, in 2019, landmark legislation paved the way for the implementation of the NYC Streets Plan, a transformative initiative. Recently, in 2022, Mayor Adams and the New York City Council dedicated a substantial investment of $904 million to bring the NYC Streets Plan to fruition (Transportation Alternatives, 2023). The impending transformation of New York City's landscape is projected to exert a significant impact on bike-sharing demand. To navigate these forthcoming shifts with precision, it is imperative to rigorously assess various prediction methods to ensure the highest attainable accuracy. Thus, the above-mentioned research question gains substantial justification due to the pressing need to comprehend the multifaceted influences that shape bike-sharing patterns within the urban context and to predict demand accurately.

This research embarks on a comprehensive analysis by studying Citi Bike ridership data spanning from June 1, 2018, to May 31, 2023. Additionally, it integrates weather data, a pandemic control variable, US holidays, and bicycle counts conducted around New York City at key locations at 15-minute intervals to gain deeper insights into the predictors that shape ridership trends. The analysis is grounded in a desire to understand how these variables impact

ridership demand and to formulate an effective predictive model. By understanding the relationship between the predictor variables and the response variable, it should be possible to predict the response variable given the predictor inputs (Faghih-Imani et al., 2014). The research hypothesis is anchored in the objective of achieving a minimum of 70% prediction accuracy through the utilization of a multiple linear regression model that leverages market research data.

The null hypothesis (H0) asserts that Citi Bike ridership demand cannot be accurately predicted through a multiple linear regression model based on market research data, with an accuracy surpassing 70%. On the contrary, the alternative hypothesis (H1) contends that such a model can indeed achieve a prediction accuracy exceeding 70%. To explore these hypotheses, the research employs multiple regression analysis to discern the significance of predictor variables and to identify which of these variables are most influential in predicting Citi Bike ridership trends. Furthermore, time series analysis is a powerful tool that can handle temporal dependencies and seasonal trends in the data, making it particularly suitable for this study (Hyndman & Athanasopoulos, 2018). This analytical approach not only aids in addressing the central research question but also provides a benchmark for algorithmic comparison.

Data Collection

The data collection process was initiated by sourcing Citi Bike ridership data from the Citi Bike System Data web portal, hourly weather data from Visual Crossing, the Covid-19 pandemic timeline from Investopedia, and US holidays data from Microsoft Azure.

Historical bike trip data was obtained from the official Citi Bike NYC website, a valuable source that provided a comprehensive snapshot of ridership patterns. This endeavor involved the acquisition of a substantial dataset consisting of 60 CSV files, collectively containing a staggering 122,523,911 rows of ridership data. This expansive dataset provided a rich and

detailed account of Citi Bike trips, spanning an extensive timeframe from June 2018 to May 2023. The dataset was organized into a structured format using the powerful data manipulation capabilities of the Pandas library in Python. By utilizing Pandas, the data was transformed into a DataFrame, a versatile data structure that facilitated efficient analysis and manipulation of the dataset. The resulting DataFrame featured a DatetimeIndex, allowing for temporal analysis and exploration of trends over time. All variables and their data types are listed in the table below:

Variable	Type	Variable	Type	Variable	Type
tripduration	continuous	start station latitude	continuous	end station longitude	continuous
starttime	continuous	start station longitude	continuous	bikeid	categorical
stoptime	continuous	end station id	categorical	usertype	categorical
start station id	categorical	end station name	categorical	birth year	continuous
start station name	categorical	end station latitude	continuous	gender	categorical

[1]

Hourly weather data was sourced from Visual Crossing, a comprehensive historical weather database. The dataset covered New York City's weather conditions from June 2018 to May 2023, providing variables such as temperature, wind speed, precipitation, and more. All variables and their data types are listed in the table below:

Variable	Type	Variable	Type	Variable	Type
name	categorical	snow	continuous	solarenergy	continuous
datetime	continuous	snowdepth	continuous	uvindex	continuous
temp	continuous	windgust	continuous	severerisk	categorical
feelslike	continuous	windspeed	continuous	conditions	categorical
dew	continuous	winddir	continuous	icon	categorical
humidity	continuous	sealevelpressure	continuous	stations	categorical
precip	continuous	cloudcover	continuous	preciptype	categorical
precipprob	continuous	visibility	continuous	solarradiation	continuous

[2]

Given the significant impact of the Covid-19 pandemic on ridership trends, pandemic control variables were incorporated. These variables were designed to account for variations in bikeshare trip volume during distinct phases of the pandemic. A timeline of the pandemic was

[1] Citi Bike Ridership Data, Variables
[2] Hourly Weather Data, Variables

extracted from Investopedia, enabling the categorization of data into pre-pandemic, during-pandemic, and post-pandemic periods. All variables and their data types are listed in the table below:

Variable	Type	Variable	Type	Variable	Type
pre-pandemic	continuous	lockdown	continuous	post-vaccine	continuous
early pandemic	continuous	reopening	continuous	post-pandemic	continuous

[3]

To assess the influence of holidays on ridership patterns, US holidays data was integrated from Microsoft Azure's open datasets API. This dataset facilitated the analysis of how holidays impact bike-sharing usage, adding a valuable contextual dimension. All variables and their data types are listed in the table below:

Variable	Type	Variable	Type	Variable	Type
countryOrRegion	categorical	normalizeHoliday Name	categorical	countryRegionCode	categorical
holidayName	categorical	isPaidTimeOff	categorical	date	continuous

[4]

The NYC Bicycle Counts dataset was sourced from the NYC Open Data portal, offering a comprehensive insight into bicycle usage around New York City. The dataset captures bicycle counts conducted at strategic locations within the city at 15-minute intervals. This rich temporal granularity enables the examination of detailed patterns and trends in bicycle ridership over time.

Variable	Type	Variable	Type	Variable	Type
countid	categorical	date	continuous	status	categorical
id	categorical	counts	continuous		

[5]

In summary, the data collection process involved meticulous sourcing from diverse channels, ensuring that the dataset encompassed influential factors in bike-sharing usage patterns. By combining this data with other sources such as weather conditions and holidays, it becomes possible to uncover correlations and trends that shape bicycle usage patterns. The data

[3] Pandemic Control Variables
[4] NYC Bicycle Counts, Variables
[5] NYC Bicycle Counts, Variables

can also be used to explore the determinants of cycling behavior, and the factors influencing the use of bike sharing systems (Ricci, 2019). Additionally, this data can serve as a foundation for predictive modeling, offering the potential to forecast future Citi Bike ridership based on various influencing factors.

Advantage of the Data-Gathering Methodology

An advantageous aspect of the data-gathering methodology lies in its comprehensive incorporation of a wide array of variables that hold potential influence over Citi Bike ridership. The inclusion of weather data, pandemic-related variables, and holidays data imparts a holistic perspective on the myriad factors that shape ridership patterns. As a result, this comprehensive approach contributes to the development of a predictive model characterized by enhanced accuracy and profound insights.

Disadvantage of the Data-Gathering Methodology

Conversely, a drawback of this data-gathering methodology lies in the potential introduction of noise and intricacy stemming from the amalgamation of diverse data sources. For instance, weather data could manifest variations that lack direct correlation with ridership trends. This introduces the possibility of injecting extraneous noise into the analysis, potentially culminating in overfitting within the predictive model.

Overcoming Challenges

Numerous challenges emerged during the data gathering phase, primarily rooted in data preprocessing and addressing missing values. The presence of incomplete or absent data points poses the risk of inducing bias within the subsequent analysis.When dealing with data that is

missing at random, related data can be deleted to reduce bias (Master's in Data Science, n.d.). In response, a range of data cleaning methodologies were implemented. These encompassed the removal of rows containing missing values and the transformation of categorical variables into binary counterparts via dummy variables. Moreover, meticulous attention was directed toward aligning data from disparate sources in a coherent manner. This intricate process demanded a judicious fusion of data manipulation and merging techniques, pivotal in attaining results that remained both precise and meaningful.

In summation, the data collection process entailed the comprehensive aggregation of diverse datasets for a comprehensive evaluation of Citi Bike ridership trends. While this methodology furnishes notable advantages by integrating multifaceted variables, it concurrently introduces intricacies necessitating meticulous management during preprocessing and analysis. Through meticulous diligence in data cleaning and manipulation, the complexities were surmounted, culminating in a dataset of exceptional quality poised for in-depth analysis.

Data Extraction and Preparation

Data Extraction

The initial stride within the data extraction process encompassed the transformation of the 60 resource-intensive CSV files containing Citi Bike data into more lightweight Parquet files. Subsequent to the conversion of CSV files to Parquet format, the ensuing progression involved the ingestion of each file. This phase entailed the establishment of uniformity in column names, the exclusion of superfluous columns, and the conversion of trip times to a datetime format, as illustrated below.

```
# Create an empty DataFrame to compile data
tripdata = pd.DataFrame()

# Initialize the start time
start_time = time.time()

# Iterate over each file
for i, file in enumerate(files):
    print(f"Processing file {i+1}/{len(files)}")
    df = pd.read_parquet(file)

    # Calculate the ETA
    calc_progress(i+1, len(files), start_time)

    # Standardize names and drop segmented columns
    if 'starttime' in df.columns:
        # Rename columns
        df.rename(columns={
            'start station id': 'start_station_id',
            'start station name': 'start_station_name',
            'start station latitude': 'start_lat',
            'start station longitude': 'start_lng',
            'end station id': 'end_station_id',
            'end station name': 'end_station_name',
            'end station latitude': 'end_lat',
            'end station longitude': 'end_lng'
        }, inplace=True)

        # Drop columns
        df.drop(['stoptime', 'tripduration', 'bikeid', 'birth year', 'gender'], axis=1, inplace=True)

        # Map usertype column values
        df['usertype'] = df['usertype'].map({'Subscriber': 'member', 'Customer': 'casual'})

    if 'started_at' in df.columns:
        # Rename columns
        df.rename(columns={
            'started_at': 'starttime',
            'member_casual': 'usertype'
        }, inplace=True)

        # Drop columns
        df.drop(['ride_id', 'rideable_type', 'ended_at'], axis=1, inplace=True)

    # Map station_id columns and add borough data
    df['start_station_id'] = df['start_station_name'].map(map_station_id)
    df['end_station_id'] = df['end_station_name'].map(map_station_id)
    df['start_borough'] = df['start_station_id'].map(map_borough)
    df['end_borough'] = df['end_station_id'].map(map_borough)

    # Drop station name and spatial columns
    df.drop(['start_station_id', 'end_station_id', 'start_station_name', 'end_station_name', 'start_lat', 'start_lng',
             'end_lat', 'end_lng'], axis=1, inplace=True)

    # Convert to datetime
    df['starttime'] = pd.to_datetime(df['starttime'])

    # Set a datetime index
    df.set_index('starttime', inplace=True)

    # Concatenate the clean data to the tripdata
    tripdata = pd.concat([tripdata, df])
```

[6]

Following these manipulations, each file underwent amalgamation into a principal DataFrame. To ensure a judicious management of this extensive procedure, the timing of the data collection endeavor was meticulously tracked. Each file's processing time was logged,

[6] Data Extraction, Step 1

concurrently displaying both the mean time per file and an estimate of the remaining duration. This iterative timing mechanism facilitated the maintenance of well-timed updates throughout the course of this protracted undertaking.

Subsequently, the following phase entailed the elimination of trips that had their inception or termination within the confines of the Bronx, New Jersey, or an undisclosed borough. This discerning filtration of data resulted in a reduction of dataset size by 22.91%. Nevertheless, this process left an abundant 94.45 million rows available for further analysis.

```
# **Exclude observations originating or ending in the Bronx, New Jersey, or Unknown**
tripdata = tripdata[(tripdata['start_borough'] != 'Bronx') & (tripdata['start_borough'] != 'New Jersey') &
        (tripdata['start_borough'] != 'Unknown') & (tripdata['end_borough'] != 'Bronx') &
        (tripdata['end_borough'] != 'New Jersey') & (tripdata['end_borough'] != 'Unknown')]
```

```
# Remove the Bronx, Unknown, and New Jersey categories
for col in ['start', 'end']:
    tripdata[f'{col}_borough'].cat.remove_categories(['Bronx', 'Unknown', 'New Jersey'], inplace=True)
```

[7]

Subsequently, the "usertype" column was divided into two separate columns representing user types, and a similar process was applied to the start and end borough columns. The original columns were eliminated, and distinct columns were generated for each pairing of trip origin or destination and user type. The final step involved resampling the datetime index at an hourly interval, leading to a reduction in data size from 3.2 gigabytes to a more manageable 7.4 megabytes.

```
# Get usertype dummy variables
df[['nyc_trips_casual', 'nyc_trips_member']] = pd.get_dummies(df['usertype'])
```

```
# Drop the user column
df.drop('usertype', axis=1, inplace=True)
```

```
# Create borough start and end count columns
for col in ['start', 'end']:
    df = pd.concat([df, pd.get_dummies(df[f'{col}_borough']).add_suffix(f"_{col}_all")], axis=1)
```

[7] Data Extraction, Step 2

```
# Drop the start and end borough categorical columns
df.drop(['start_borough', 'end_borough'], axis=1, inplace=True)
```

```
# Make column names lowercase
df.columns = [col.lower() for col in df.columns]
```

```
# Add a trip count column
df['nyc_trips_all'] = 1
```

```
# Break down trip counts by borough and rider type
for borough in ['brooklyn', 'manhattan', 'queens']:
    for rider in ['casual', 'member']:
        for n in ['start', 'end']:
            df[f"{borough}_{n}_{rider}"] = df[f"nyc_trips_{rider}"] * df[f"{borough}_{n}_all"]
```

[8]

The weather data contained extensive details, but excessive granularity could result in overfitting. To address this, the initial action involved removing irrelevant columns and designating the date as a datetime index.

```
weather.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 43824 entries, 0 to 43823
Data columns (total 6 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   datetime    43824 non-null  object
 1   temp        43824 non-null  float64
 2   humidity    43824 non-null  float64
 3   precip      43824 non-null  float64
 4   windspeed   43824 non-null  float64
 5   visibility  43824 non-null  float64
dtypes: float64(5), object(1)
memory usage: 2.0+ MB
```

```
# Convert data types
weather['datetime'] = pd.to_datetime(weather['datetime'])
```

```
# Set the datetime column as the index
weather.set_index('datetime', inplace=True)
```

[9]

The Azure holidays data encompassed global holidays. Initially, holidays not applicable to the United States were eliminated, and any country-related columns were discarded.

```
# Remove observations not from the US
holiday = holiday[holiday['countryRegionCode'] == 'US']
```

```
# Reset the index
holiday.reset_index(drop=True, inplace=True)
```

```
# Drop the country related columns and redundant columns
holiday.drop(['countryOrRegion', 'countryRegionCode', 'normalizeHolidayName'], axis=1, inplace=True)
```

[10]

[8] Data Cleaning, Step 1
[9] Data Cleaning, Step 2
[10] Data Cleaning, Step 4

Following this, the date column was transformed into a datetime index. Subsequently, an "isHoliday" column was generated to indicate holiday dates. The data was further resampled on a daily basis to include non-holiday dates. In the "holidayName" column, entries were replaced with "Non-Holiday." The resultant DataFrame was resampled once more on an hourly basis to align with the Citi Bike and weather data.

```
# Convert the date column to datetime
holiday['date'] = pd.to_datetime(holiday['date'])

# Create an isHoliday column
holiday['isHoliday'] = 1

# Convert isPaidTimeOff and isHoliday to uint8 type
for col in ['isPaidTimeOff', 'isHoliday']:
    holiday[col] = holiday[col].astype('uint8')

holiday.set_index('date', inplace=True)

# Resample the data to daily frequency without forward filling
holiday = holiday.resample('D').apply(lambda x: x if x.values != np.nan else np.nan)

# Fill missing holidayName values
holiday['holidayName'].fillna('Non-Holiday', inplace=True)

# Fill remaining missing values with zero
holiday.fillna(0, inplace=True)

# Resample by hour
holiday = holiday.resample('H').ffill()
```

[11]

The NYC Bicycle Counts dataset's first cleaning step was to begin by removing unnecessary variables ('countid', 'id', 'status'). Subsequently, counts from various NYC locations are aggregated by grouping the data based on the 'date' column. This consolidation results in a summation of bicycle counts for each unique date. The datetime index of the aggregated DataFrame is converted for efficient temporal manipulation. Sorting the data chronologically by date ensures a coherent sequence. The analysis then narrows its focus to the period from June 1, 2018, to May 31, 2023, through the selection of relevant dates using the 'loc' function. The aggregated daily counts are further refined by resampling at an hourly frequency, enhancing the temporal granularity. Finally, to enhance clarity, the counts column is renamed as

[11] Data Cleaning, Step 5

'nyc_bike_counts'. These combined steps yield a well-preprocessed DataFrame containing hourly bicycle counts, setting the stage for comprehensive analysis of bicycle ridership patterns across New York City.

```
# Drop unnecessary variables
df.drop(['countid', 'id', 'status'], axis=1, inplace=True)
```

```
# Aggregate the counts from multiple locations in NYC
df_agg = df.groupby('date').sum()
```

```
# Convert the index to datetime
df_agg.index = pd.to_datetime(df_agg.index)
```

```
# Sort the data by date
df_agg.sort_index(inplace=True)
```

```
# Keep only relevant dates
df_agg = df_agg.loc[dt.datetime(2018, 6, 1): dt.datetime(2023, 5, 31, 23, 45)]
```

```
# Resample the data by hour
df = df_agg.resample('H').sum()
```

```
# Rename the counts column to be more descriptive
df.columns = ['nyc_bike_counts']
```

12

Justification and advantages/disadvantages.

The central tool utilized for efficient data extraction revolved around the Pandas library. By making use of the capabilities offered by the Glob2 and OS libraries, the process of loading Parquet and CSV files into DataFrames was executed smoothly. Additionally, the DateTime and Numpy libraries played a key role in enabling efficient access to data within substantial files. The decision to prioritize the Pandas library for data extraction was driven by its prevalent use in data manipulation tasks within the industry. Additionally, the integration of Azure ML's PublicHolidays class simplified the retrieval of public holiday data. This class was selected for its ability to streamline the data collection process, contributing to improved efficiency and accuracy throughout the endeavor.

Pandas, Glob2, OS, DateTime, and Numpy collectively provided a robust suite of tools that greatly facilitated the management and manipulation of the data. Pandas, being the central

[12] Data Cleaning, Step 6

library, played a pivotal role in handling various data formats like Parquet and CSV, which were effortlessly loaded into structured DataFrames. The inclusion of Glob2 and OS libraries further enhanced the data extraction process, streamlining the handling of files and directories.

DateTime and Numpy libraries seamlessly integrated into the workflow, ensuring efficient access to data within large files and enabling seamless datetime operations. These libraries were particularly beneficial in handling timestamps and time-based calculations, which were crucial for aligning data from multiple sources accurately.

The decision to rely on Pandas as the primary data extraction tool was well-founded due to its proven effectiveness in data manipulation tasks. Additionally, the utilization of Azure ML's dedicated class for acquiring holiday data demonstrated a strategic choice, simplifying the retrieval of essential information and enhancing overall efficiency in the process. This amalgamation of tools and techniques streamlined the data extraction phase and laid a strong foundation for subsequent analysis and modeling.

While Pandas is highly proficient in managing datasets of moderate sizes, it could encounter challenges when confronted with significantly larger datasets due to memory constraints. However, a strategic approach to mitigate these limitations involves converting CSV files to Parquet format, which substantially alleviates the memory burden. Furthermore, it's important to acknowledge that the process of extracting data from external sources may introduce additional intricacies, such as dealing with API authentication procedures and potential network connectivity issues. These factors underscore the importance of considering the scalability of data extraction methods and ensuring smooth interactions with external sources for comprehensive data collection.

Data Preparation

The data underwent a meticulous cleaning and preparation procedure, encompassing actions like renaming columns, discarding irrelevant attributes, handling missing values, converting data types, and generating novel features. The initial step in readying the data for analysis involved consolidating information from diverse sources into a unified DataFrame. Specifically, the weather data was initially integrated with the Citi Bike dataset utilizing the datetime index, followed by the incorporation of holiday data.

```
df_hourly = pd.read_parquet('../clean_data/hourly_tripdata.parquet')
```

```
# Merge weather with df
df = pd.merge(df_hourly.reset_index(), weather.reset_index(), left_on='starttime', right_on='datetime', how='left')
```

```
df.set_index('starttime', inplace=True)
```

```
# Merge holidays with weather and rides
df = pd.merge(df, holiday.reset_index(), left_on='datetime', right_on='date', how='left').drop(
    ['date', 'datetime'], axis=1).set_index(df.index)
```

[13]

Five instances of missing values were detected within the weather dataset, and to rectify this, they were imputed using the mean value of the preceding and subsequent observations.

```
for col in ['temp', 'humidity', 'precip', 'windspeed', 'visibility']:
    for i, row in df[df[col].isna()].iterrows():
        before = weather.loc[i - dt.timedelta(hours=1)]
        after = weather.loc[i + dt.timedelta(hours=1)]
        df.at[i, col] = (before[col] + after[col]) / 2
```

[14]

After detecting several outliers, the logical next step involved implementing data winsorization to address this issue. The primary goal of this process is to alleviate the influence of extreme values, which have the potential to disrupt the integrity of the models and their subsequent analyses. By strategically capping extreme values, data winsorization contributes to a more stable and reliable modeling outcome.

[13] Data Aggregation, Step 1
[14] Data Aggregation, Step 2

```
def winsorize_feature(feature, lower_percentile, upper_percentile):
    """
    Apply winsorization to a feature array.

    Args:
        feature (numpy.ndarray): The array containing the feature values.
        lower_percentile (float): The lower percentile value for winsorization.
        upper_percentile (float): The upper percentile value for winsorization.

    Returns:
        numpy.ndarray: The winsorized feature array with replaced outliers.
    """
    # Calculate the winsorizing values
    lower_value = np.percentile(feature, lower_percentile)
    upper_value = np.percentile(feature, upper_percentile)

    # Replace outliers with winsorizing values
    winsorized_feature = np.where(feature < lower_value, lower_value, feature)
    winsorized_feature = np.where(winsorized_feature > upper_value, upper_value, winsorized_feature)

    return winsorized_feature
```

```
# Winsorize outliers
for col in weather.columns:
    weather[col] = winsorize_feature(weather[col], 5, 95)
```

15

Following the consolidation of data into a unified file, the subsequent pivotal phase involved feature engineering. This strategic process imparts the dataset with enhanced insights, capable of bolstering the efficacy of predictive models. Key temporal attributes including the hour, weekday, month, year, and seasons were meticulously extracted from the datetime index. Leveraging the weekday information, a binary weekend variable was ingeniously crafted, assigning 0 to weekdays (Monday to Friday) and 1 to weekends (Saturday and Sunday). This nuanced transformation adds valuable context to the dataset and lays the foundation for more robust analyses and predictions.

```
# Extract the hour, weekday, month, and year from the start datetime
df['hour'] = (df.index.hour)
df['weekday'] = df.index.weekday
df['month'] = df.index.month
df['year'] = (df.index.year)
```

```
# Extract the season from the month
seasons = {1: 'winter', 2: 'winter', 3: 'spring', 4: 'spring', 5: 'spring', 6: 'summer',
           7: 'summer', 8: 'summer', 9: 'autumn', 10: 'autumn', 11: 'autumn', 12: 'winter'}
df['season'] = df['month'].map(seasons)
```

```
df['weekend'] = [0 if day < 5 else 1 for day in df.index.weekday]
```

16

[15] Data Aggregation, Step 3
[16] Feature Engineering, Step 1

According to The New York Times, the morning rush hours span from 7:30 AM to 9:30 AM, while the evening rush hour is from 5:00 PM to 7:00 PM. Given the dataset's hourly frequency, adjustments were made. The morning rush hour was extended from 7:00 AM to 10:00 AM, maintaining the evening rush hour as originally defined. These refined time intervals, in combination with the extracted weekend and hour variables, were harnessed to construct the informative "hour_type" variable. This new variable effectively encapsulates the temporal dynamics of rush hours and weekends, offering a valuable dimension for further analysis and interpretation.

```
def categorize_hour(day_type, hour):
    """
    Categorize an hour as rush hour or not based on the day type and hour value.

    Args:
        day_type (int): Indicator for the day type (1 for rush hour, 0 for non-rush hour).
        hour (int): The hour value (24-hour format) to be categorized.

    Returns:
        str: A string indicating whether the hour is categorized as rush hour or not.
    """
    if day_type == 1:
        return 'not rush hour'
    elif hour >= 7 and hour <= 10:
        return 'rush hour'
    elif hour >= 17 and hour <=19:
        return 'rush hour'
    else:
        return 'not rush hour'
```

```
df['hour_type'] = [categorize_hour(row['weekend'], row['hour']) for _, row in df.iterrows()]
```

17

The last phase of feature engineering culminated in the formulation of the "pandemic_period" variable, designed to encapsulate the variable ridership patterns witnessed during the Covid-19 pandemic. Drawing insights from the Investopedia timeline, pivotal pandemic events were identified and leveraged to segment the data into distinct periods: pre-pandemic, lockdown, reopening, post-vaccine, and post-pandemic. This comprehensive division provides a nuanced understanding of how different pandemic stages influenced Citi Bike ridership demand. By incorporating this variable into the analysis, the model gains the

[17] Feature Engineering, Step 2

ability to capture and differentiate ridership trends across these pivotal phases, contributing to the depth and accuracy of the predictions.

```
def categorize_date(date):
    if date < pd.to_datetime('2020-03-07'):
        return 'pre-pandemic'
    elif date < pd.to_datetime('2020-06-08'):
        return 'lockdown'
    elif date < pd.to_datetime('2021-04-06'):
        return 'reopening'
    elif date < pd.to_datetime('2022-01-01'):
        return 'post-vaccine'
    else:
        return 'post-pandemic'
```

```
df['pandemic_period'] = pd.Series(df.index).apply(categorize_date).values
```

[18]

In the final leg of data preparation, a pivotal step was taken to convert the data into a machine-readable format. Specifically, the ordinal variables "pandemic_period" and "season" were subjected to label encoding. This method preserves the inherent sequential order within these variables, ensuring that their underlying patterns and relationships are maintained in a manner that machine learning algorithms can readily comprehend. This transformation enhances the model's capacity to discern the nuanced distinctions in pandemic periods and seasons, fostering more accurate predictive outcomes.

```
# Label encode the pandemic_period column
pp = {'pre-pandemic': 0, 'lockdown': 1, 'reopening': 2, 'post-vaccine': 3, 'post-pandemic': 4}
df['pandemic_period'] = df['pandemic_period'].map(pp)
```

```
# Label encode the season column
seasons = {'spring': 0, 'summer': 1, 'autumn': 2, 'winter': 3,}
df['season'] = df['season'].map(seasons)
```

[19]

By applying trigonometric functions, specifically sine and cosine, to the “hour” and “season” variables, their inherent cyclical tendencies were safeguarded. This intricate process, known as cyclical encoding, works as a mechanism to retain the cyclic patterns that inherently reside within time-based features. The result is a dataset enriched with new variables that

[18] Feature Engineering, Step 3
[19] Data Transformation, Step 1

harmoniously capture the rhythmic ebbs and flows of hours and seasons, elevating the model's discernment of cyclic trends and bolstering the potency of predictive analyses.

```
df['hour_sin'] = np.sin(2 * np.pi * df['hour'] / 24)
df['hour_cos'] = np.cos(2 * np.pi * df['hour'] / 24)
```

```
df['season_sin'] = np.sin(2 * np.pi * df['season'] / 4)
df['season_cos'] = np.cos(2 * np.pi * df['season'] / 4)
```

[20]

The data transformation journey culminated with the application of one-hot encoding to the residual categorical variables. This task was deftly executed by harnessing the power of the Pandas library's "get_dummies" function. The purpose of this encoding method is to translate categorical variables into a format that machine learning algorithms can readily comprehend. Each categorical attribute metamorphosed into a set of binary variables, each representing a distinct category. This transformation empowers the predictive model by eliminating any hierarchical or ordinal assumptions while enhancing its ability to glean insights from categorical attributes.

```
# One-hot encode the hour type column
df['rush_hour'] = pd.get_dummies(df['hour_type'])['rush hour']
```

```
# Drop the hour type column
df.drop('hour_type', axis=1, inplace=True)
```

```
# Combine the isPaidTimeOff and isHoliday columns
df['is_holiday'] = (df['isPaidTimeOff'] + df['isHoliday']).replace(2, 1)
df.drop(['isPaidTimeOff', 'isHoliday'], axis=1, inplace=True)
```

[21]

Justification and advantages/disadvantages.

The robust capabilities of Pandas were instrumental in the data extraction and preparation phases. The utilization of strategies such as column renaming, elimination, and data type conversion was pivotal in orchestrating the harmonious metamorphosis of the dataset. These transformative maneuvers were executed employing an array of Pandas functions, including the

[20] Data Transformation, Step 2
[21] Data Transformation, Step 3

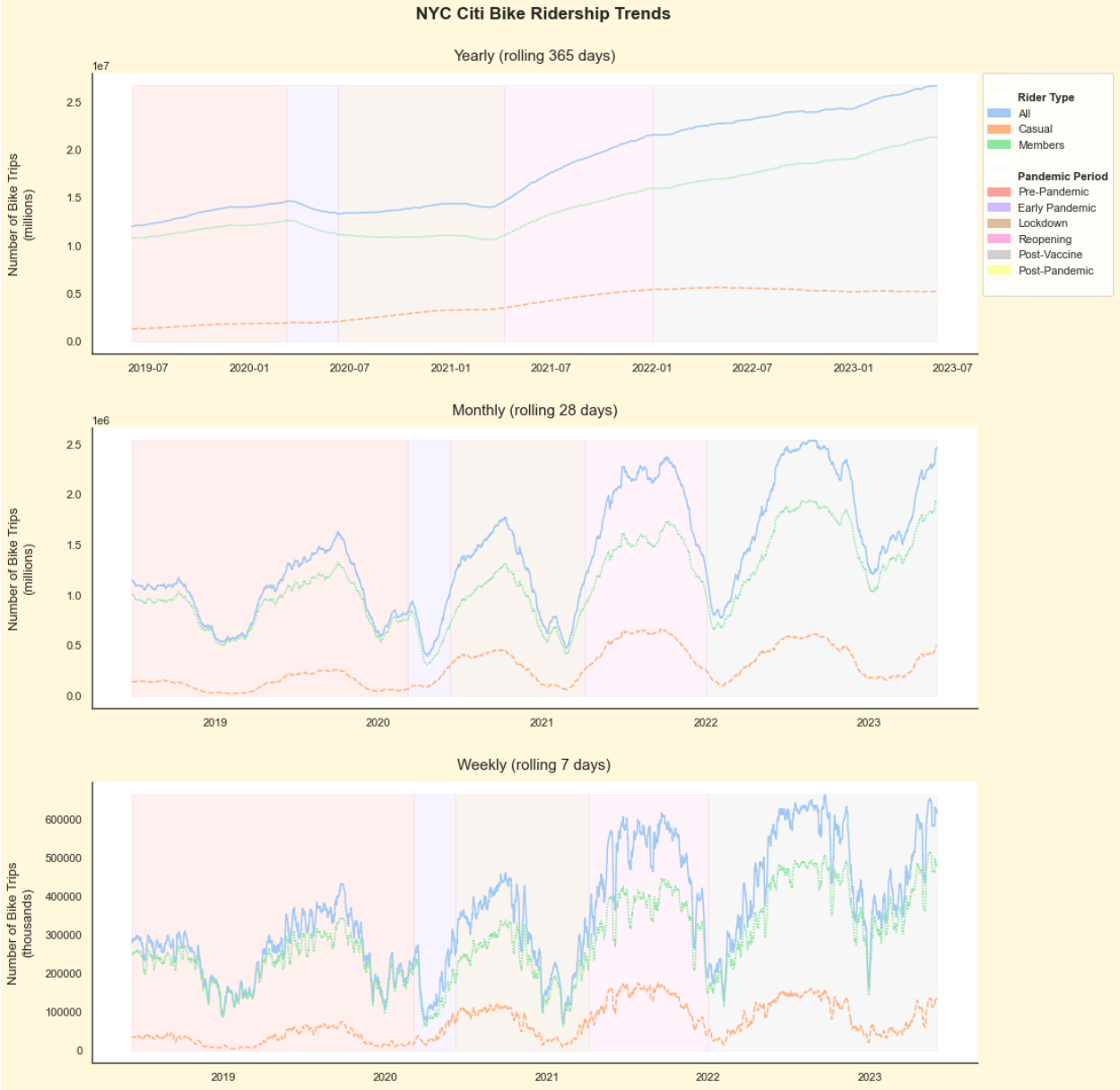
versatile "map" and "get_dummies" functionalities. The selection of Pandas as the linchpin of this operation was steered by its remarkable adaptability to diverse data manipulation tasks. This adaptability, in turn, fostered a seamless orchestration of data transformations while preserving data integrity.

The inherent flexibility of Pandas served as a cornerstone, empowering the adept manipulation and transformation of the data to align with the project's demands. This dynamic library proved to be an invaluable asset, effortlessly shouldering tasks such as the rectification of missing values and the conversion of data types.

Much like in the data extraction phase, Pandas' efficiency could diminish when handling exceptionally large datasets. Additionally, manual data preparation processes hold the potential for errors or inconsistencies if not carefully supervised. The utilization of one-hot encoding may result in expanded dataset dimensions, potentially elevating memory usage and computational intricacies. Moreover, excessive or inappropriate feature engineering could introduce unwanted noise or multicollinearity within the dataset.

In summary, the execution of the data extraction, transformation, and preparation process was skillfully conducted through the utilization of tools and techniques offered by Pandas, NumPy, and Azure ML's PublicHolidays class. These tools provided the necessary adaptability, functionality, and efficiency for navigating the intricate realm of data cleaning, transformation, and aggregation. Nonetheless, it remains vital to exercise prudence and account for resource management, especially when managing substantial datasets or intricate merging tasks.

Analysis



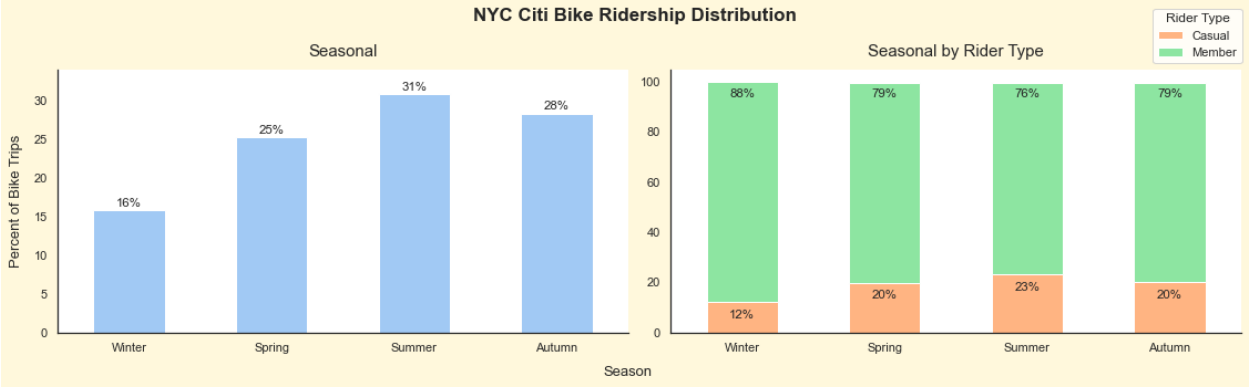
22

Data Exploration

Initiating the data exploration process involved generating plots to discern the weekly, monthly, and yearly trends through rolling mean calculations. The graphical representations in

[22] NYC Citi Bike Ridership Trends

Figure 22 revealed an accumulative trend across successive years, experiencing a temporary downturn during the initial pandemic period. Further visualization was carried out using a bar plot, as illustrated in Figure 23, showcasing the ridership distribution across different seasons. It was evident from this plot that Summer held the majority share at 31% of total trips, while Winter contributed a comparatively lower percentage of 16%. This observation lent substantial support to the hypothesis that temperature could wield a substantial impact on ridership patterns.

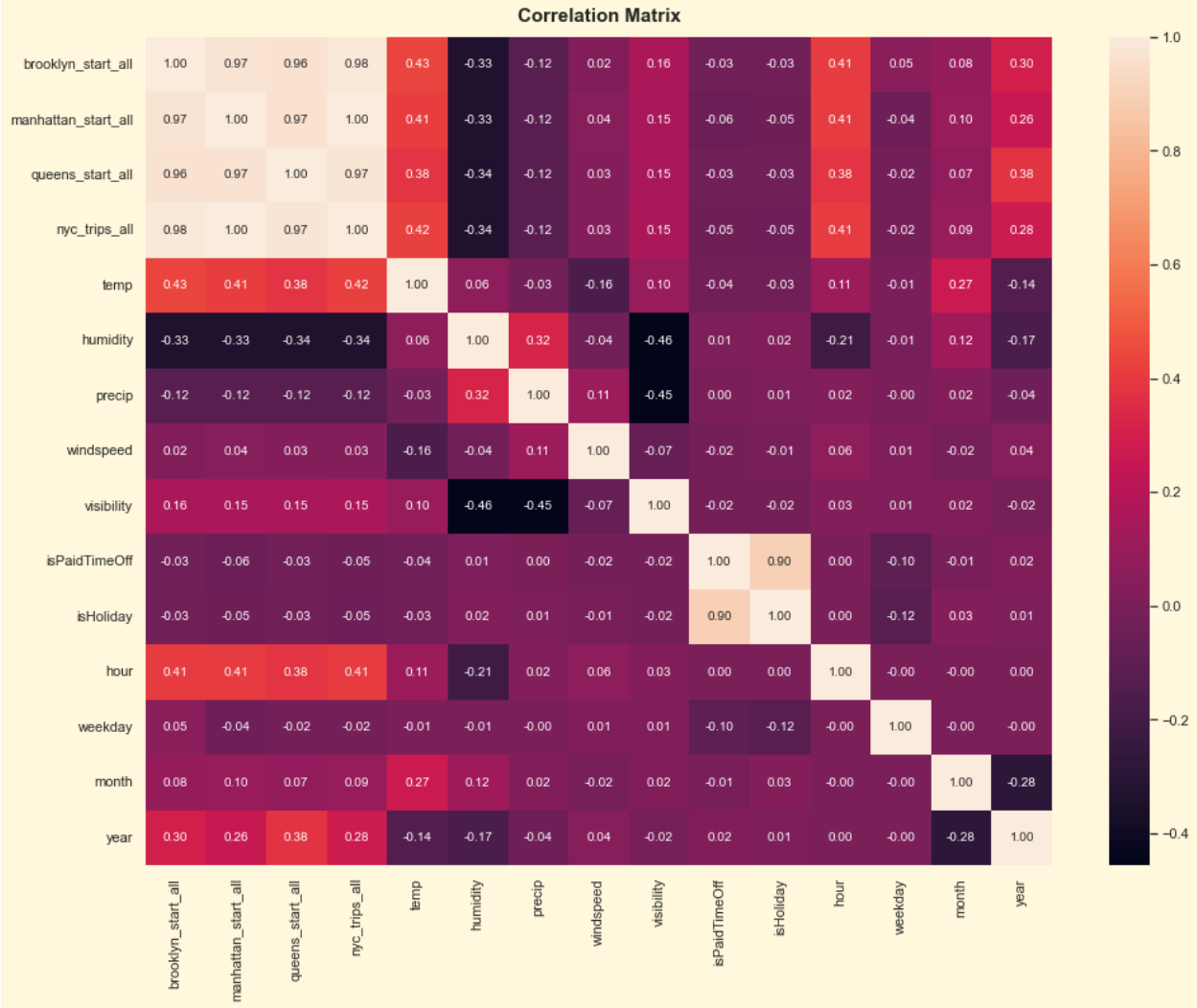


23

The correlation matrix, illustrated in Figure 24, provides additional evidence supporting the connection between temperature and ridership. The matrix analysis revealed that temperature and hour exhibit the most substantial positive correlation with ridership, indicating that higher temperatures and specific hours of the day are associated with increased ridership. Conversely, humidity emerges as the variable with the highest negative correlation, implying that higher humidity levels might deter ridership.

Exploring feature importance goes beyond mere correlation and offers a deeper understanding of how each variable contributes to predictive models. By quantifying the impact of temperature, hour, and other factors through feature importance analysis, their relative significance in predicting ridership patterns accurately can be ascertained. This quantitative assessment allows for influential variables to be prioritized when constructing predictive models.

[23] NYC Citi Bike Ridership Distribution by Season



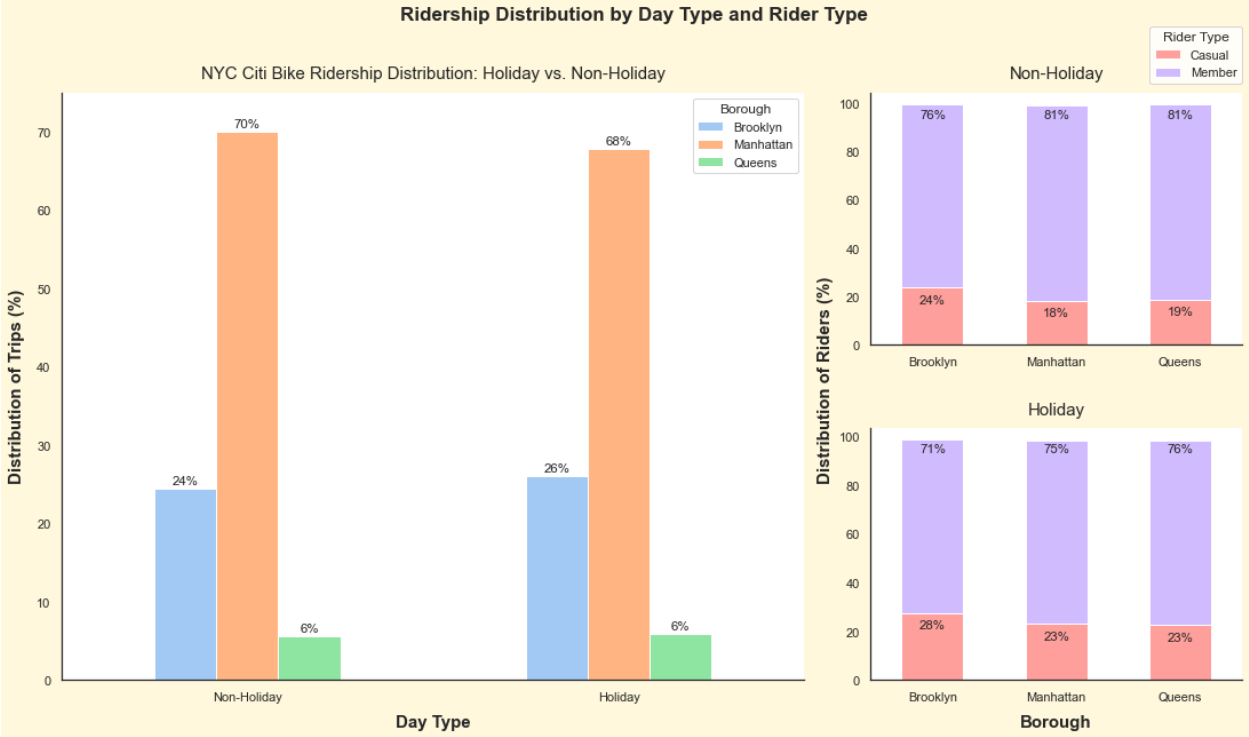
[24]

The examination of the influence of holidays on ridership unveiled interesting patterns. Specifically, it was observed that holidays trigger a decline in ridership within Manhattan, whereas Brooklyn experiences an uptick in ridership. On the other hand, Queens appears to remain relatively unaffected by holidays in terms of ridership trends. This differentiated response across boroughs underscores the complex interplay between holidays and ridership habits.

Furthermore, when exploring the distribution of rider types, as illustrated in Figure 25, a noteworthy trend emerged. Despite the overall decrease in ridership within Manhattan, there is a notable increase in casual riders across all boroughs during the same time frame. This

[24] Correlation Matrix

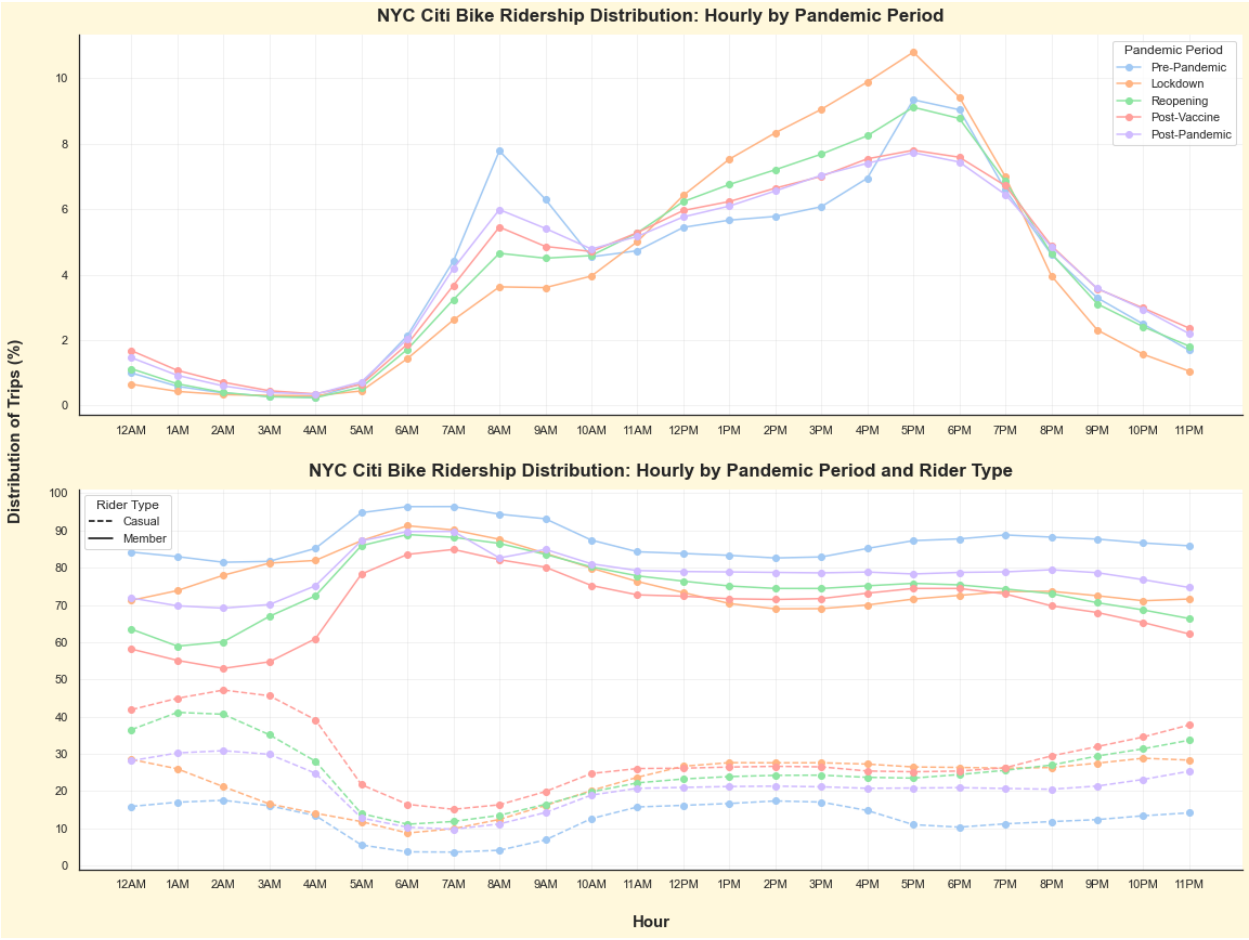
observation suggests that casual ridership might have a distinctive behavior pattern compared to members, which is important to consider when formulating predictive models.



[25]

The analysis of the Covid-19 pandemic's effects on ridership revealed that the most significant shifts in ridership trends occurred during the morning and evening rush hour periods. As illustrated in Figure 26, the hours between 6:00 AM and 10:00 AM accounted for no more than 4% of bike trips each hour during the lockdown period, a stark contrast to the pre-pandemic scenario where these same hours contributed to up to 8% of bike trips during the peak morning rush hour at 8:00 AM. Conversely, an inverse relationship was observed for the evening rush hour period. Before the pandemic, the afternoon and evening hours exhibited the smallest share of ridership compared to the other pandemic periods, while the lockdown phase witnessed the highest percentage of bike trips during these hours.

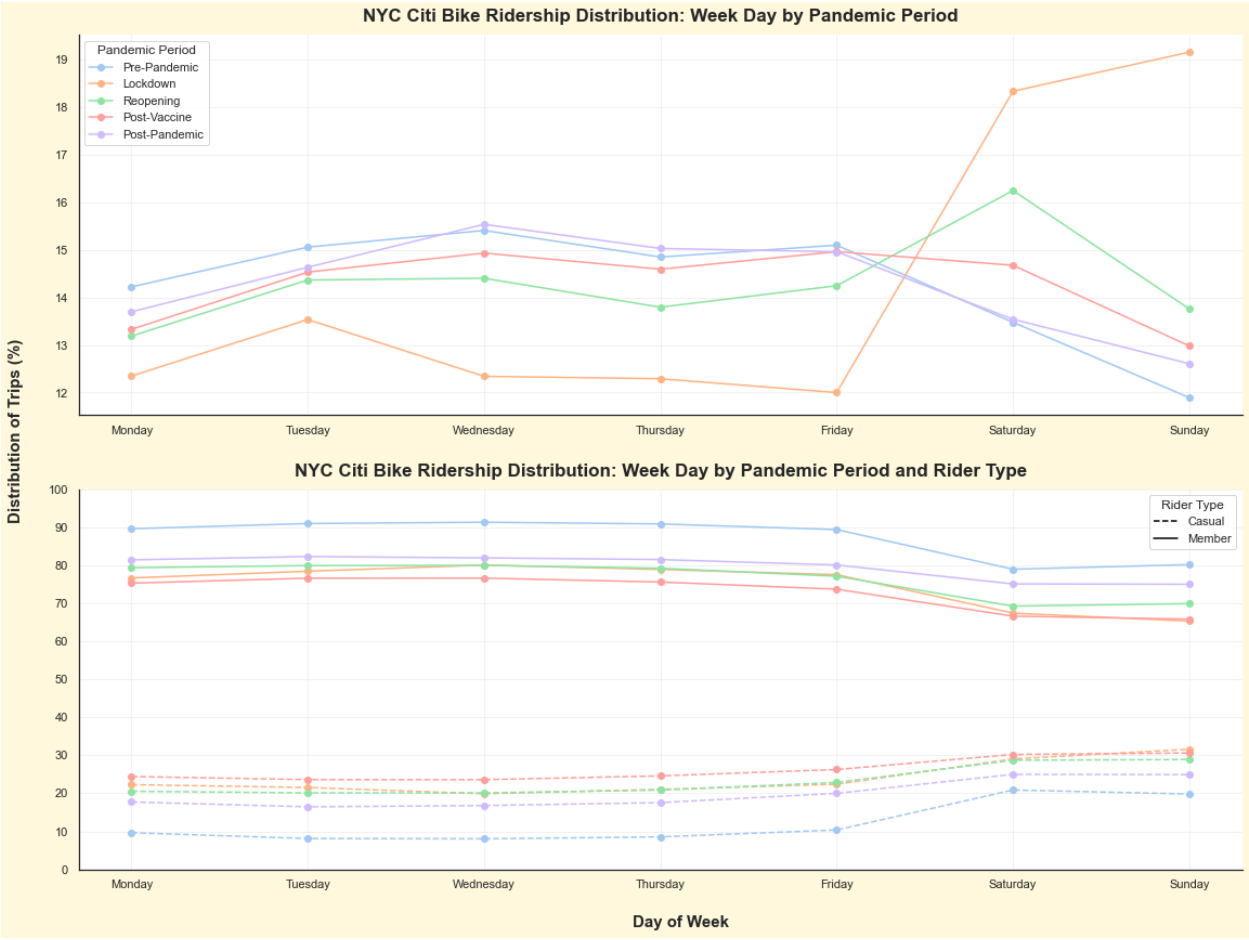
[25] Ridership Distribution by Day Type and Rider Type



26

The examination of ridership distribution by weekday and pandemic period revealed a consistent shift in trends from the pre-pandemic era to the lockdown phase and subsequently the reopening period. As depicted in Figure 27, the line plot demonstrates that ridership reached its zenith on Wednesdays during the pre-pandemic period and gradually declined for each subsequent day. However, during the lockdown period, ridership increased from Friday through Sunday and exhibited fluctuating patterns on weekdays. This behavior was further mirrored in the reopening phase, wherein ridership peaked on Saturdays and more closely adhered to the pre-pandemic trends during the weekdays.

[26] NYC Citi Bike Ridership Distribution: Hourly by Pandemic Period



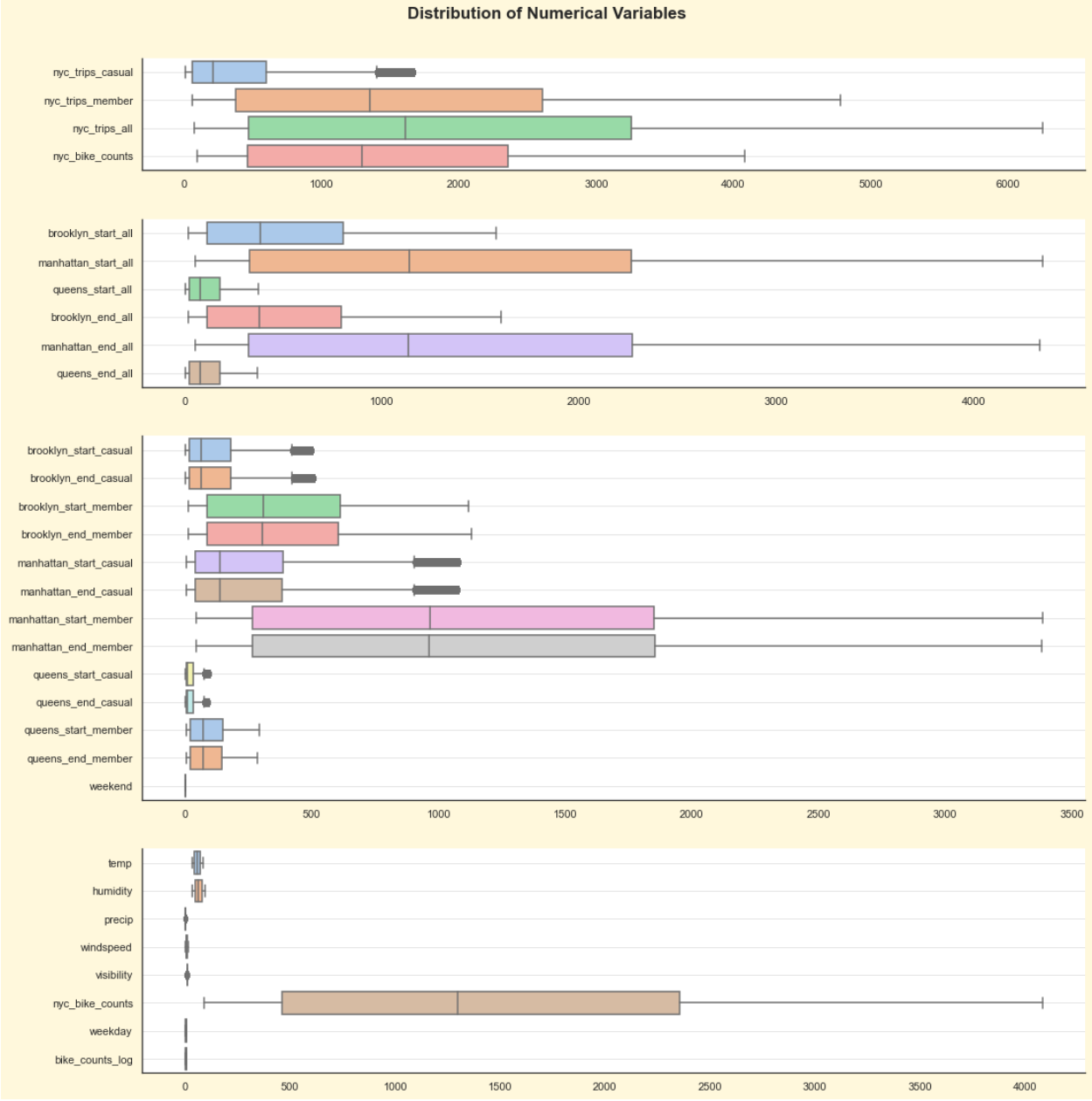
27

Upon delving into an extensive examination of ridership trends through the lenses of weather conditions, holiday occurrences, and pandemic periods, a subsequent phase of analysis encompassed the exploration of individual variable distributions. The insights drawn from this exploration contribute valuable context to the overall understanding of the data's characteristics.

The boxplots depicted in Figure 28 stand as informative visual representations of the data's distribution characteristics. These boxplots provide a comprehensive overview of how variables behave in terms of their central tendencies, spread, and presence of potential outliers. Specifically, the efficacy of the winsorization technique, which was applied during the preliminary data preparation stage, becomes readily apparent. This technique successfully

[27] NYC Citi Bike Ridership Distribution: Week Day by Pandemic Period

ensures that the total number of trips within New York City—the target variable for subsequent predictive modeling—displays a right-leaning distribution while maintaining a notable absence of significant outliers.



28

Furthermore, a detailed examination of other essential variables showcases intriguing patterns. The temperature, humidity, windspeed, and weekday variables, for instance, follow

[28] Distribution of Numerical Variables

distribution patterns consistent with a normal distribution. This conformity to a bell-shaped curve suggests a characteristic distribution that is symmetric and centered around the mean value. The attributes of these variables appear to align with expected trends and provide insights into their potential influence on ridership.

However, the visibility variable diverges from this norm. Its distribution exhibits a pronounced right-skew, indicating that the majority of data points are clustered toward the lower values of visibility. This unique distribution feature could potentially imply that instances of low visibility might be more frequent compared to higher visibility conditions, thereby suggesting a distinctive impact on ridership patterns under such circumstances. In essence, the exploration of individual variable distributions supplements the comprehensive understanding of the dataset, further guiding subsequent analyses and modeling endeavors. By uncovering the distinct distribution characteristics of various variables, this analysis adds depth to the insights derived from the overall exploratory process.

Justification and advantages/disadvantages.

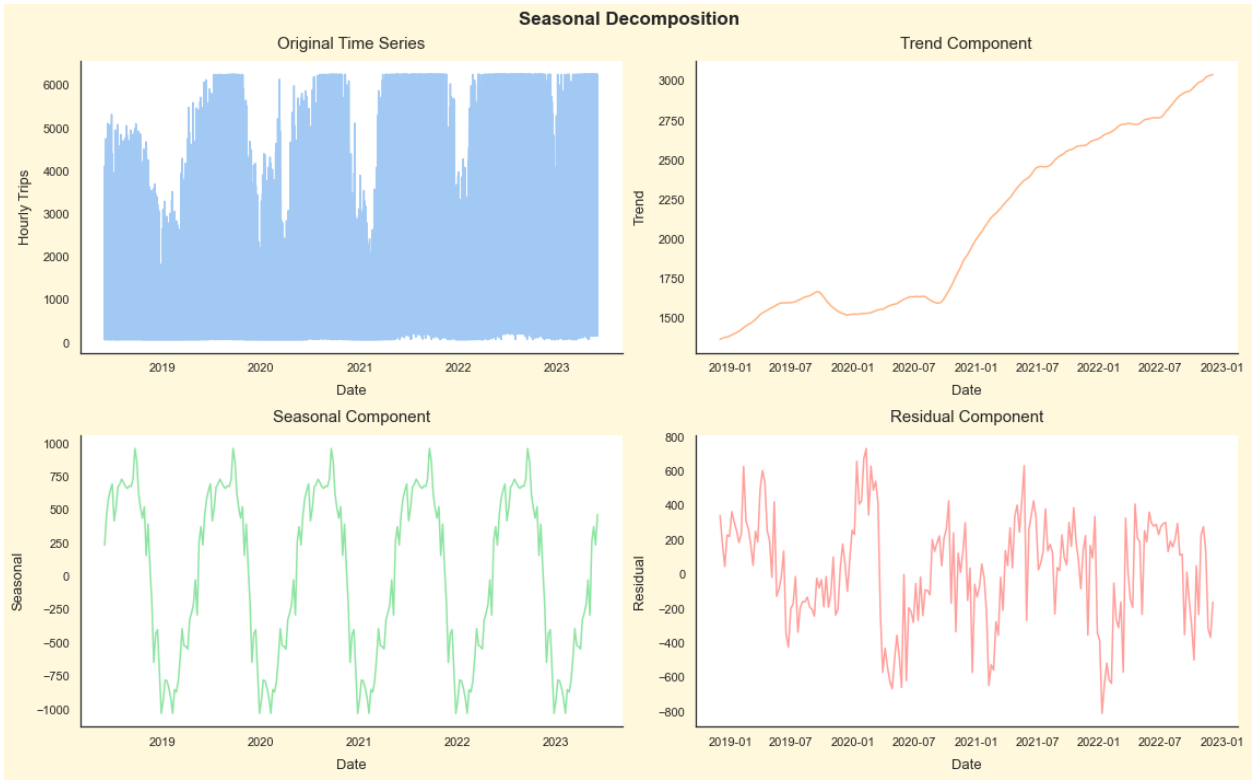
The selection of analysis techniques was driven by the specific objectives of uncovering insights into ridership trends and their underlying factors. Each technique was chosen to contribute to a comprehensive understanding of the data, aligning with the research question at hand. Rolling mean calculations and trend visualization were employed to identify long-term patterns over time, such as weekly, monthly, and yearly trends. While this technique smooths out noise, it might overlook short-term fluctuations. Bar plots were utilized for seasonal analysis, effectively portraying the distribution of categorical variables like seasons. This approach simplifies categorical relationships, but may miss complex interactions.

The correlation matrix and feature importance analysis offered a quantitative perspective on relationships and influential features. While the correlation matrix revealed potential drivers, it did not establish causation. Similarly, feature importance quantified variable impact but might not capture collective interactions. Examining the impact of external factors like the Covid-19 pandemic and holidays provided insights into unique ridership trends and disruptions. However, this analysis might not encompass all underlying causal factors.

Exploring variable distributions shed light on characteristics like skewness, normality, and outliers. This guided subsequent data preparation steps and modeling choices. However, focusing solely on individual variable distributions might not capture complex interactions between variables. It's important to acknowledge that these techniques have limitations. For instance, correlation does not imply causation, and individual variable analysis might not uncover all interactions. Despite these limitations, combining multiple techniques mitigates their drawbacks, providing a robust and comprehensive perspective on ridership trends.

Time-Series Analysis

In embarking upon the phase of time-series analysis, the initial stride involved an intricate exploration through seasonal decomposition—an insightful technique poised to unveil the latent constituents residing within the temporal data. This technique effectively disassembles the time series, unraveling it into three distinct and discernible elements: trend, seasonal, and residual components. By unraveling these components, a comprehensive panorama emerges, shedding light on the intricate interplay of temporal patterns and trends that characterize the dataset.



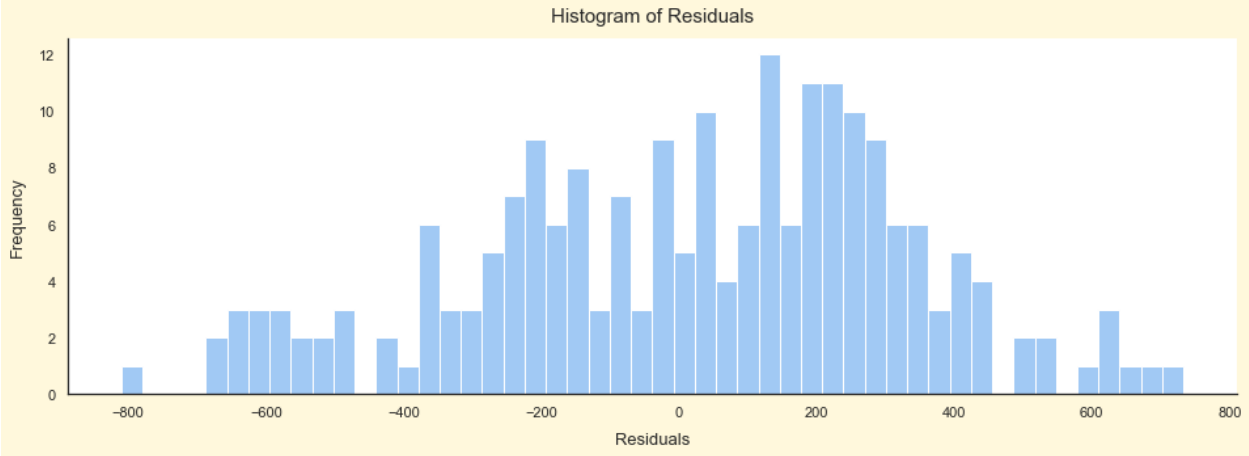
29

The seasonal decomposition plots showcased in Figure 29 serve as a visual aid in illustrating the results of this analysis. Notably, these plots bring to light the presence of an annual cyclic trend inherent within the data. This recurring pattern, unfolding across the span of a year, bespeaks a regularity that encapsulates the dataset's yearly rhythms. However, an equally significant revelation emerges in the trend component. In this aspect, the data diverges from the anticipated narrative, displaying a lack of discernible and conspicuous patterns within the trend component. The juxtaposition of these two components—the cyclical rhythm encompassed by the seasonal element and the relatively unstructured nature of the trend—forges a nuanced understanding of the temporal dynamics encapsulated within the data.

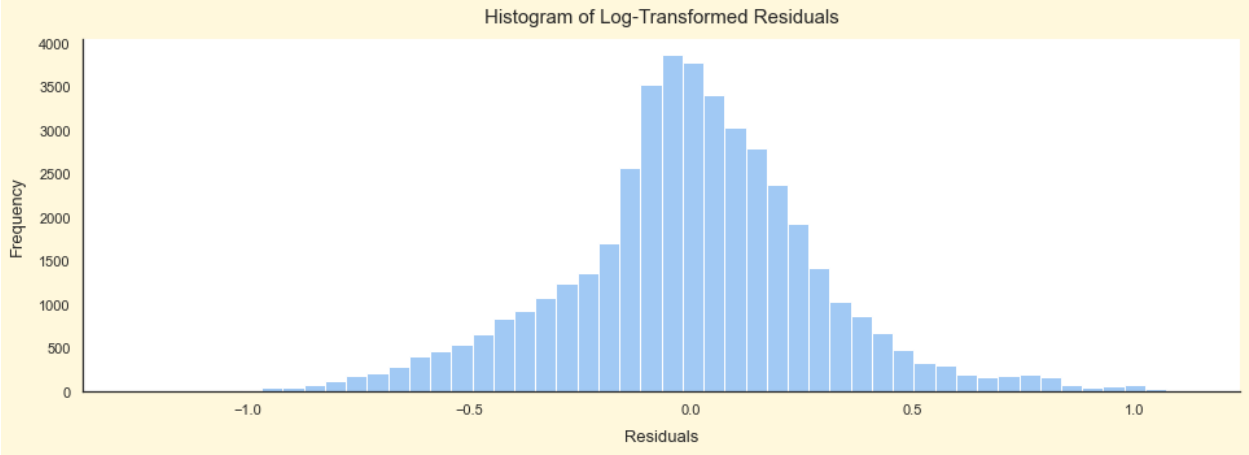
The subsequent step involved the depiction of residuals, captured within Figure 30. The resultant histogram plot provided a tangible representation of these residuals, revealing their

[29] Seasonal Decomposition

distributional tendencies. However, it became apparent that this distribution did not uniformly adhere to the contours of a typical normal distribution—warranting further consideration.



30

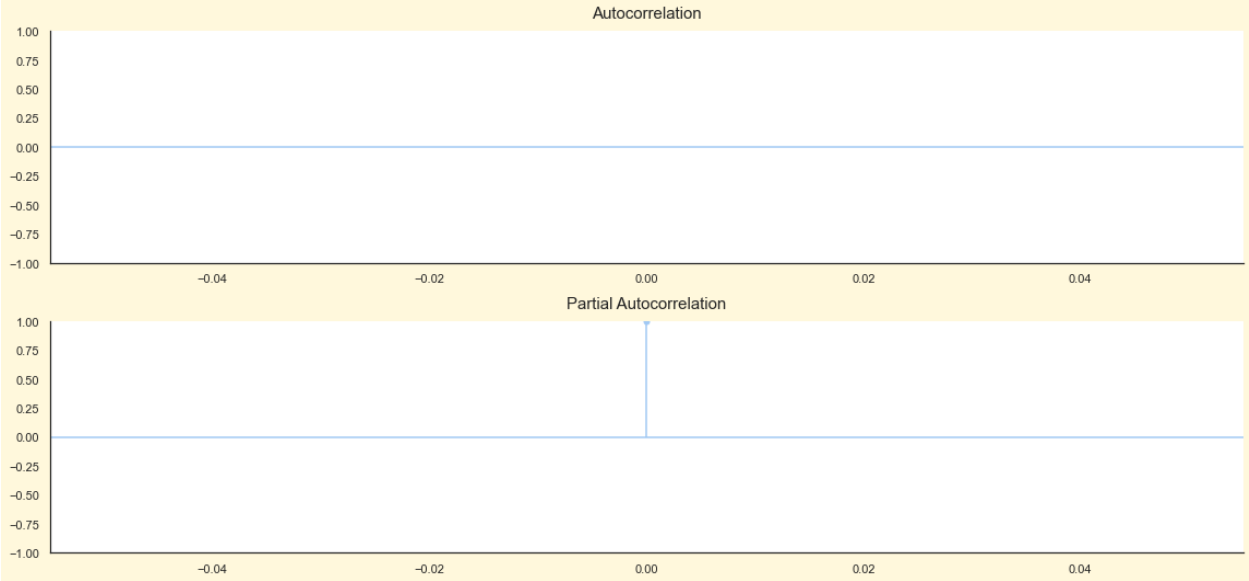


31

The target variable underwent a log-transformation, a method chosen to induce a more balanced and normative distribution within the residual plot. As this transformation took root, the histogram plot observed within Figure 31 reflected a shift, evidencing the normalization imparted by the log-transformation. Yet, as with many endeavors seeking equilibrium, a nuanced residue persisted—a slight leftward skew, reminiscent of the intricacies often present in complex real-world data.

[30] Histogram of Residuals
[31] Histogram of Log-Transformed Residuals

The subsequent phase involved generating autocorrelation and partial autocorrelation plots. These graphical tools are essential components of time series analysis, enabling the identification of underlying patterns and dependencies within data over time. The autocorrelation plot (ACF), shown in Figure 32, is a visualization that illustrates the correlation between a series and its lagged versions. When scrutinizing the autocorrelation plot, it was evident that there were no significant lags beyond which the correlation between the current observation and its past observations became negligible. This suggests that there isn't a pronounced repeating pattern in ridership demand that extends beyond the immediate past.



32

Conversely, the partial autocorrelation plot (PACF) delves deeper into the direct relationship between a data point and its lagged versions while accounting for the influence of intermediate observations. It allows for the identification of correlations that can't be explained by preceding lags. In the analysis, the partial autocorrelation plot revealed a significant correlation at a lag of zero, indicating that the current ridership demand is directly related to its

[32] Autocorrelation/Partial Autocorrelation Plots

immediate past value. This suggests a strong auto-regressive component in the time series, where the current demand is influenced by its previous demand value.

These plots serve as critical diagnostic tools to discern the temporal dependencies within the data, aiding in the informed decision-making about modeling techniques and parameter selections. In this case, the absence of significant autocorrelation lags and the presence of a notable partial autocorrelation at lag zero provided insights into the nature of the time series and guided the subsequent modeling choices.

In the subsequent phase, the Dickey-Fuller test was performed on the training dataset—an integral step in scrutinizing time series data. This test serves a crucial role in determining whether the data exhibits a consistent pattern or displays more unpredictable behavior over time. The Dickey-Fuller test yielded specific results that offer significant insights into the nature of the dataset. The Test Statistic stands as a pivotal indicator. The value of -9.20 indicates that the dataset reveals a consistent pattern, suggesting its stationary nature rather than susceptibility to random fluctuations. The exceedingly small P-value of 1.98e-15 underscores that the likelihood of encountering such a result by random chance is extremely low. This strengthens the argument that the dataset demonstrates stationary behavior.

Upon evaluating the Critical Values, it is evident that thresholds exist at the 1%, 5%, and 10% significance levels. Importantly, the Test Statistic surpasses all of these critical values. This observation is noteworthy as it reaffirms the dataset's consistent patterns, thus enhancing its reliability for subsequent analysis. In this context, the term "stationary" conveys that the statistical characteristics of the data remain stable over time. This outcome is highly favorable and confirms the dataset's steady and stationary patterns, as it plays an integral role in ensuring the credibility of subsequent analysis and predictions.

The Variance Inflation Factor (VIF) measures the multicollinearity between predictor variables in a regression model. In general, a VIF value of 1 indicates no multicollinearity, while higher values indicate increasing multicollinearity. A common guideline is that VIF values above 10 can be considered concerning and might warrant further investigation or potential remediation. Following the initial calculation of the VIF for each variable, the obtained results shed light on potential multicollinearity concerns. The VIF values for the variables were as follows:

Variable	VIF	Variable	VIF	Variable	VIF
year	36.81	season_sin	5.35	humidity	1.77
season	14.82	season_cos	4.32	rush_hour	1.76
hour	12.48	temp	4.15	visibility	1.49
weekday	8.78	hour_sin	3.25	precip	1.30
bike_counts_log	8.31	weekend	2.82	windspeed	1.09
month	8.23	hour_cos	2.12	is_holiday	1.04
nyc_bike_counts	6.52	pandemic_period	2.10		

[33]

These initial VIF results provided insights into the degree of collinearity among the predictor variables. Notably, variables such as "year" and "season" exhibited relatively high VIF values, indicating the potential presence of multicollinearity. In response to these findings, a refinement was conducted by excluding the "year" and "season" variables from the dataset. This adjustment aimed to alleviate the multicollinearity issue and ensure that the remaining variables contribute independently to the predictive model. The subsequent VIF results, were as follows:

Variable	VIF	Variable	VIF	Variable	VIF
bike_counts_log	7.50	season_sin	3.16	rush_hour	1.74
hour	7.43	hour_sin	2.88	season_cos	1.64
weekday	6.50	weekend	2.39	visibility	1.49
nyc_bike_counts	6.42	hour_cos	2.07	precip	1.29
month	5.75	pandemic_period	1.97	windspeed	1.08
temp	4.11	humidity	1.77	is_holiday	1.04

[33] Original Variance Inflation Factor Results

[34]

The refined VIF values indicated a reduction in multicollinearity concerns after the removal of "year" and "season." This iterative process of assessing and refining the VIF values contributes to the selection of independent and meaningful predictor variables, ensuring the robustness of the subsequent predictive model.

ARIMA (AutoRegressive Integrated Moving Average) is a popular time series forecasting method that combines autoregressive, integration, and moving average components. It models time-dependent data by considering the past values, differencing to achieve stationarity, and past prediction errors. ARIMA is specified by parameters 'p' (autoregressive order), 'd' (integration order), and 'q' (moving average order). It's used for predicting trends, seasonality, and cycles in time series data. While versatile, ARIMA assumes linearity and constant parameters over time.

In the subsequent step, the construction of an initial Autoregressive Integrated Moving Average (ARIMA) model revealed valuable insights into its performance. The Akaike Information Criterion (AIC) is a statistical measure used for comparing the quality of different statistical models. It balances the trade-off between model fit and complexity by penalizing models with more parameters. AIC quantifies how well a model fits the data relative to the number of parameters it uses, aiming to prevent overfitting. Lower AIC values indicate better model suitability, making it a useful tool in selecting the most appropriate model within a dataset by considering both goodness of fit and model simplicity.

The model's AIC value was computed at -76,138.82, offering a measure of the model's goodness of fit relative to its complexity. Furthermore, the mean-absolute-percentage error (MAPE) was computed at 26.69%. MAPE is a metric used to evaluate the accuracy of a

[34] Refined Variance Inflation Factor Results

forecasting model. It measures the average percentage difference between predicted and actual values, providing insight into the magnitude of prediction errors relative to the actual values.

The Root Mean Square Error (RMSE) is another widely used metric for assessing the accuracy of predictions or forecasts. It measures the square root of the average of the squared differences between predicted and actual values. RMSE represents the typical size of the errors in the model's predictions, giving an indication of how well the model fits the data. It is sensitive to larger errors due to the squaring process and is often used to compare the performance of different models. A lower RMSE indicates better predictive accuracy, with values closer to zero reflecting a closer match between predictions and actual data.

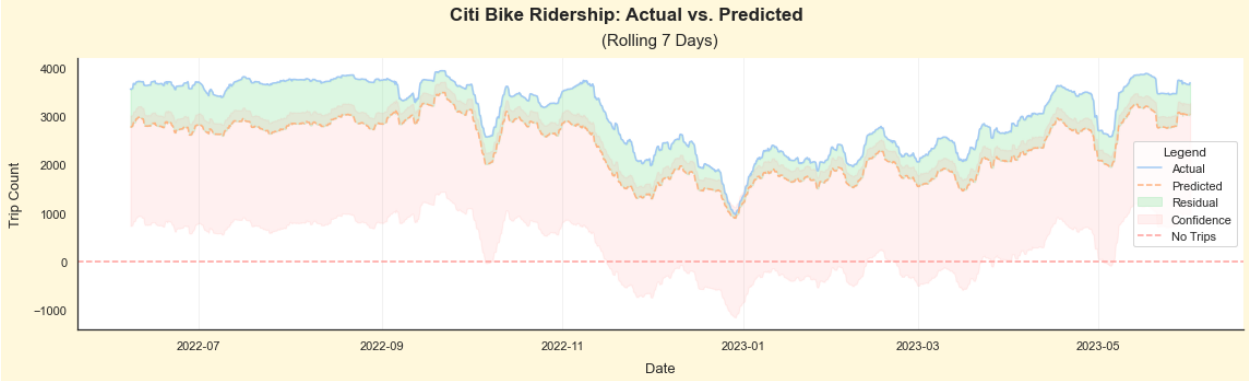
The root-mean-square error (RMSE) was assessed at 905.44, providing a representation of the average magnitude of the model's prediction errors, and finally the Accuracy Rate was determined to be 73.3%. These metrics collectively yielded a comprehensive understanding of the model's effectiveness in capturing inherent data patterns. The above metrics are calculated as follows:

Metric	Equation	Variables
AIC	$2k \ - \ 2ln(L)$	k = Number of parameters in the model L = Likelihood value of the model given the data
MAPE	$\frac{1}{n}\sum_{i=1}^{n}\left\|\frac{A_i - F_i}{A_i}\right\| \ \bullet \ 100$	A_i = Actual value at time i F_i = Forecasted value at time i n = Number of data points
RMSE	$\sqrt{\frac{1}{n}\sum_{i=1}^{n}(A_i - F_i)^2}$	A_i = Actual value at time i F_i = Forecasted value at time i n = Number of data points
Accuracy Rate	$100 \ - \ MAPE$	$MAPE$ = Mean-absolute-percentage error

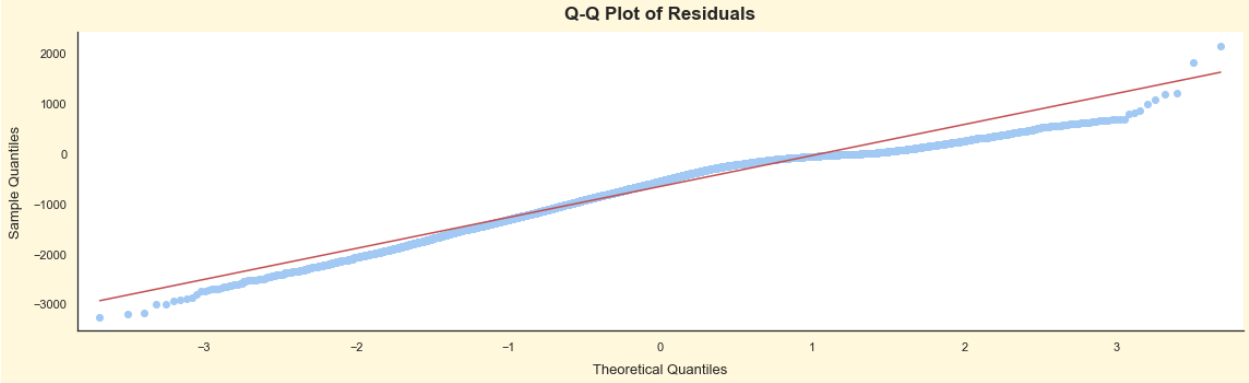
[35]

[35] Model Metric Calculations

To quantify the uncertainty within the model's predictions, confidence intervals were established through bootstrapping, employing 1000 data points. The resultant visualization (Figure 36), portraying actual versus predicted values along with their associated confidence intervals, offered a clear depiction of the model's predictive capabilities. This visual representation facilitated an assessment of how well our model's predictions aligned with actual observations.



36

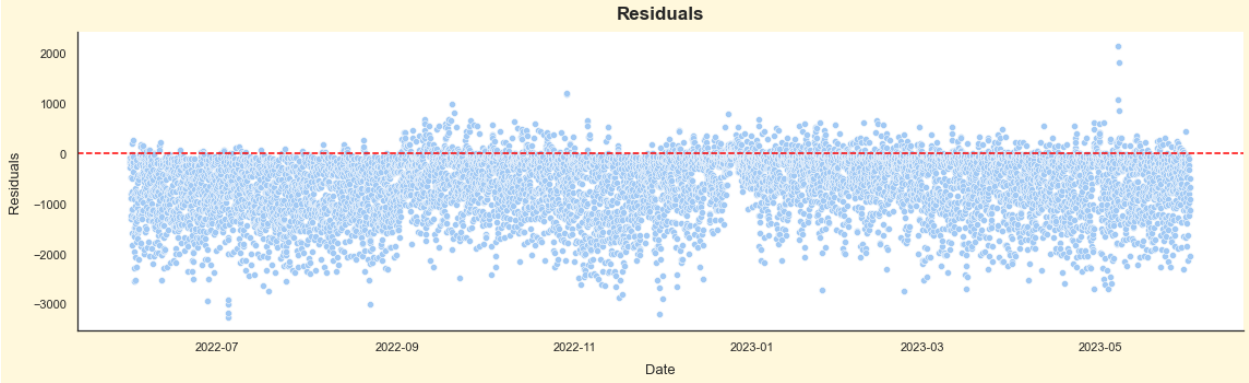


37

Subsequent to this, a thorough analysis of the residuals was conducted to assess the model's assumptions and performance. The Quantile-Quantile (Q-Q) plot (Figure 37) exhibited a robust positive correlation with a pronounced heavier tail towards the third theoretical quantile.

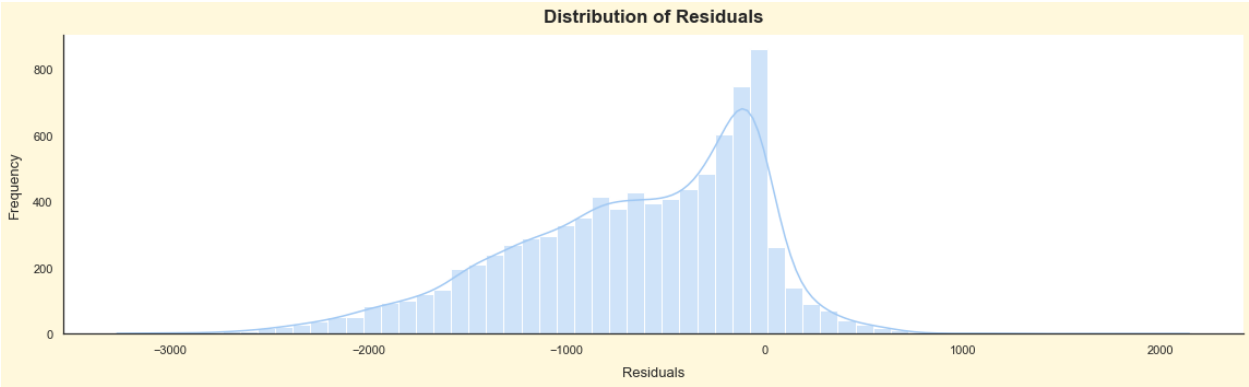
[36] Citi Bike Ridership: Actual vs. Predicted (Rolling 7 Days)
[37] Q-Q Plot of Residuals

This pattern suggested that the model performed well in capturing central tendencies but displayed deviations in extreme observations.



38

The residual plot (Figure 38) displayed residuals tightly clustered around the zero-line, affirming the model's competence in accurately predicting the majority of observations. Furthermore, examination of the residual distribution (Figure 39) unveiled a prolonged tail towards lower values, alongside a noticeable leftward skew. This indicated that the model occasionally encountered challenges when predicting extreme values.

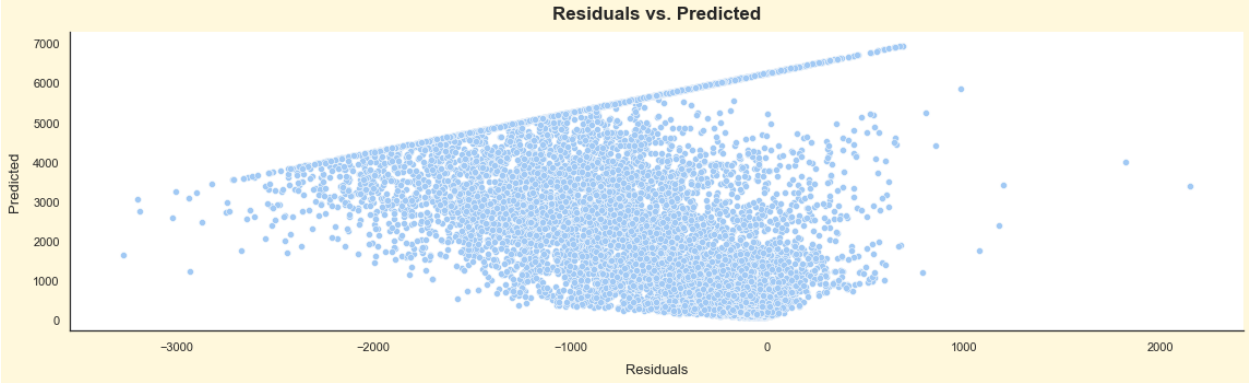


39

Concluding the analysis, the residuals versus predicted plot (Figure 40) showcased a significant positive correlation, with the majority of data points concentrated within the lower half. This trend underscored the model's tendency to underestimate higher values. Collectively,

[38] Residual Plot
[39] Distribution Plot of Residuals

these diagnostic plots offered invaluable insights into the model's performance and residual behavior, enabling a comprehensive evaluation of its accuracy, assumptions, and fit.



[40]

Additionally, the Anderson-Darling test, a vital diagnostic tool, was employed to assess the normality of the residuals. The calculated Anderson-Darling Test Statistic of 121.69 was compared against critical values at various significance levels indicating that the residuals deviate from normality at all significance levels. The outcomes of this test are summarized in the table below:

Significance Level	Critical Value
15%	0.576
10%	0.656
5%	0.787
2%	0.918
1%	1.092

[41]

The subsequent step involved determining feature importance using the random forest algorithm, a robust technique for assessing the relative significance of each feature in predictive modeling. These importance scores offer insights into the extent to which each feature contributes to the model's predictive performance. Subsequently, a stepwise regression process was undertaken, guided by the feature importance ranking. At each iteration, the feature with the

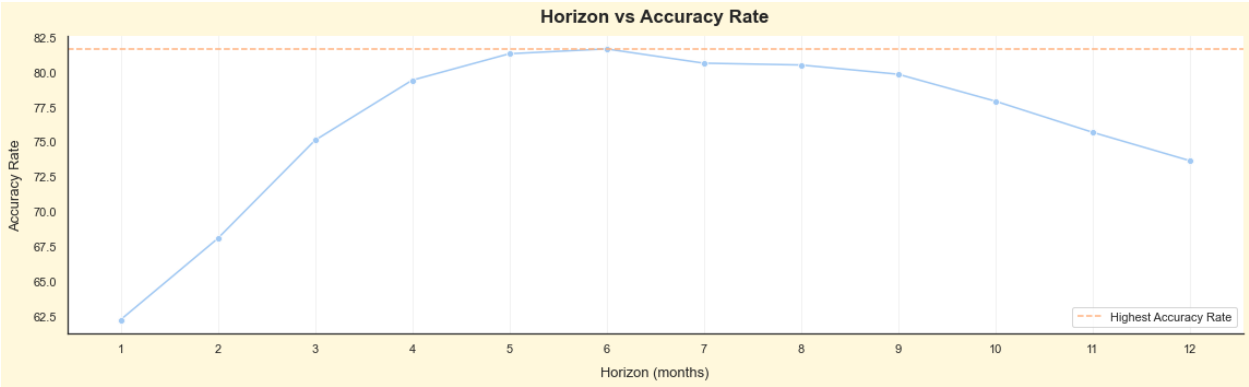
[40] Residuals vs. Predicted Plot
[41] Anderson-Darling Test Results

least significance was removed. However, intriguingly, the results underscored that the most effective model encompassed all but the 'is_holiday' feature. This outcome emphasized the collective significance of these features in accurate prediction. The importance scores obtained for various features are as follows:

Feature	Importance	Feature	Importance	Feature	Importance
nyc_bike_counts	0.474	hour_sin	0.007	visibility	0.002
bike_counts_log	0.430	weekday	0.004	season_sin	0.002
pandemic_period	0.033	humidity	0.004	precip	0.001
hour	0.014	windspeed	0.003	season_cos	0.001
hour_cos	0.012	month	0.002	rush_hour	0.001
temp	0.009	weekend	0.002	is_holiday	0.000

[42]

Subsequently, a systematic assessment was conducted to determine the optimal forecast horizon, a critical factor in refining predictive accuracy. An iterative search was performed, ranging from one to twelve months. These results were visualized in Figure 43, allowing for a comprehensive understanding of the relationship between forecast horizon and performance metrics.



[43]

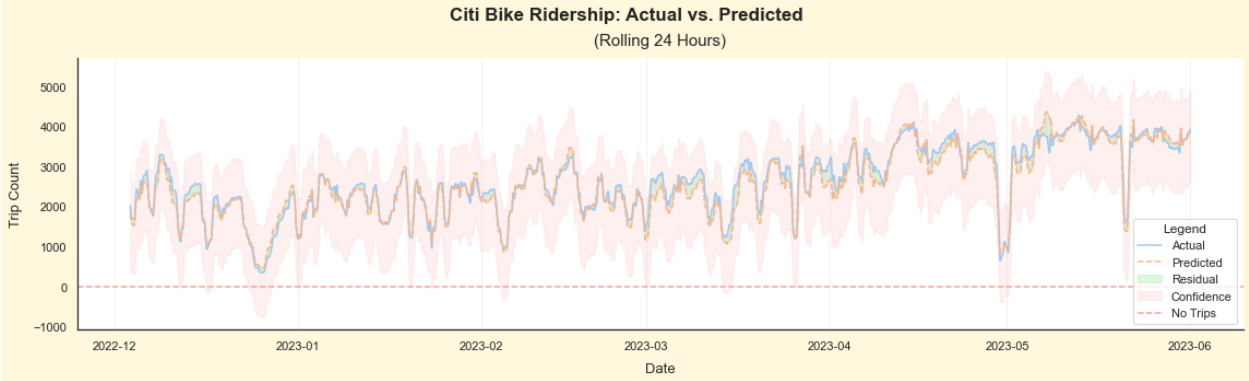
Notably, the evaluation identified the best forecast horizon as six months based on an Accuracy Rate of 81.66%. This new forecast horizon yielded a significant 11.39% improvement in the Accuracy Rate and a noteworthy 15.08% reduction in AIC compared to the initial forecast

[42] Feature Importance
[43] Horizon vs. Accuracy Rate Plot

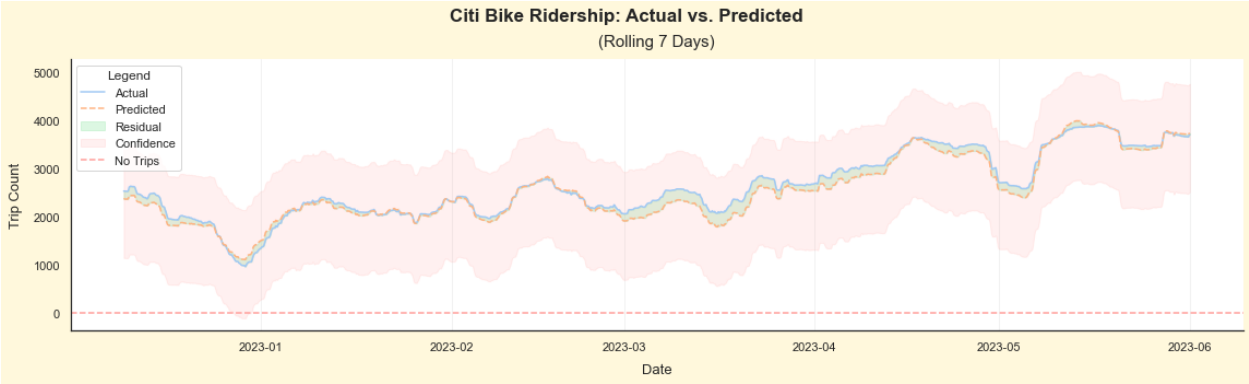
horizon. The Accuracy Rate and AIC improvements were calculated using the following equation where y is the percentage of improvement, x_0 is the original value, and x_1 is the new value:

$$y = \frac{x_1 - x_0}{x_0} \bullet 100$$

This assessment offers valuable guidance in choosing the appropriate forecast horizon, a key aspect in optimizing the predictive modeling process and ensuring robust predictions.



44

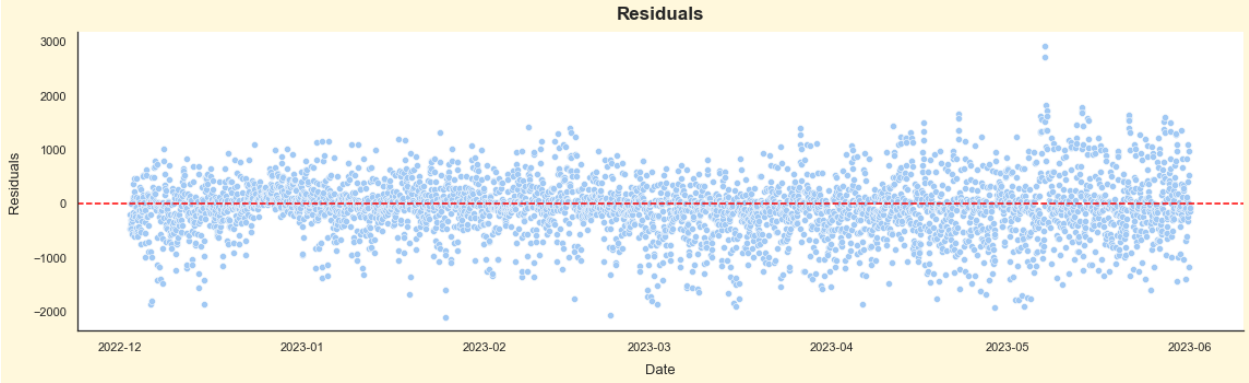


45

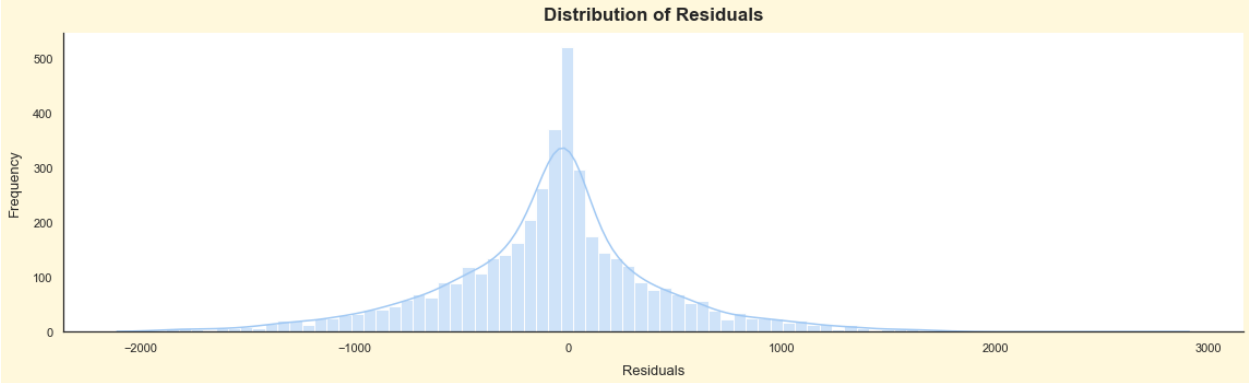
Upon determining the optimal 1-month forecast horizon, a new forecast was generated, incorporating recalculated confidence intervals through bootstrapping. This updated forecast was visually represented in Figure 44, providing an illustrative overview of the forecasted values and

[44] Citi Bike Ridership: Actual vs. Predicted (6-Month Horizon, Rolling 24 Hours)
[45] Citi Bike Ridership: Actual vs. Predicted (6-Month Horizon, Rolling 7 Days)

their associated uncertainties. A broader view can be observed in Figure 45, which depicts a rolling 7-day mean.



46



47

Upon analyzing the residuals, it became evident that they did not resemble those of the original model. The residual plot illustrated in Figure 46 depicts residuals that are much more tightly packed around the zero-line than those in Figure 38. The distribution displayed a significantly more normally distributed bell-curve. This distributional insight was visually depicted in Figure 47, providing a tangible representation of the residual behavior.

The calculated Anderson-Darling Test Statistic of 59.45 was contrasted with critical values at various significance levels, revealing that the residuals exhibit non-normal behavior across all significance levels. However, a smaller Anderson-Darling test statistic in comparison

[46] Residual Plot
[47] Distribution Plot of Residuals

to the prior test statistic suggests that the newly forecasted data conforms more closely to a normal distribution. This could imply an enhanced level of goodness of fit, indicating that the disparities from the specified distribution are relatively minimized, a desirable characteristic for statistical modeling and analysis. The results of this assessment are summarized in Figure 48. These evaluations and examinations significantly contributed to a deeper understanding of the forecast's precision, the underlying presumptions, and any deviations from the model's assumptions.

Significance Level	Critical Value
15%	0.575
10%	0.655
5%	0.786
2%	0.917
1%	1.091

[48]

Justification and advantages/disadvantages.

The selection of analysis techniques was meticulously guided by the research objectives and the nature of the temporal data. Seasonal decomposition was chosen as the initial technique to uncover underlying patterns within the time series, as it effectively breaks down the series into its trend, seasonal, and residual components. This technique is advantageous as it provides a holistic view of temporal patterns and trends. However, one limitation is that it might not capture interactions between components that contribute to the observed patterns.

The decision to employ log-transformation to achieve a more balanced residual distribution was based on the desire to normalize the residuals for accurate modeling. This technique's advantage lies in its ability to mitigate the impact of extreme values, leading to a

[48] Anderson-Darling Test Results

more robust modeling process. However, the disadvantage is that log-transformations might not always fully rectify skewed distributions and might not be suitable for all datasets.

The choice of autocorrelation and partial autocorrelation plots for time series analysis was driven by the need to identify temporal dependencies. These plots offer a visual representation of lagged correlations, aiding in understanding repeating patterns and informing modeling decisions. The advantage of these plots is their ability to highlight lag relationships. However, they might not be effective in capturing complex interactions involving multiple lags.

The application of the Dickey-Fuller test was crucial to assess the stationarity of the time series data. This test's advantage is its capability to determine if a series is stationary or not, a fundamental aspect of time series analysis. However, it has limitations in identifying the specific causes of non-stationarity or the presence of structural breaks.

Utilizing the Variance Inflation Factor (VIF) for feature selection was motivated by the need to identify and address multicollinearity among predictor variables. According to Phudinawala and Jha (2022) they were able to “predict the demand for shared bikes using multiple linear regression. To understand which attributes need to be dropped we used RFE (Recursive Feature Elimination) and VIF (Variance Inflation Factor) to drop columns with high p-values and to check the multicollinearity respectively. Now the model is able to predict the future demand for shared bikes in a city” (p. 481). The advantage of this technique is that it quantifies the degree of collinearity, aiding in the selection of independent predictors. Yet, the disadvantage is that it might not fully capture complex relationships among variables beyond linear correlations.

The evaluation of the best forecast horizon through an iterative search was a strategy to optimize predictive accuracy. The advantage of this approach is that it identifies the horizon that

minimizes prediction errors. However, the limitation is that it might not account for sudden shifts or changes in trends that could impact longer-term predictions.

In summary, the selected analysis techniques were driven by the objectives to uncover patterns, assess model assumptions, and optimize predictive accuracy. Each technique brings unique advantages while carrying specific limitations, underscoring the importance of a comprehensive and multi-faceted approach to address the complexities of time series data.

Model Development

Multiple regression analysis allows the expression of relationships between predictor variables and the response variable (Field, 2013). Ridge regression, also referred to as L2 regularization, is a linear regression approach aimed at addressing multicollinearity and potential overfitting in predictive modeling. It achieves this by incorporating a penalty term that is proportional to the square of the regression coefficients' magnitudes into the linear regression cost function. This introduces a regularization parameter, lambda, which governs the extent of regularization applied.

The primary goal of ridge regression is to find a middle ground between effectively fitting the model to training data and restraining the coefficients from becoming overly large, thus minimizing the model's susceptibility to variations in input features. By integrating this regularization component, ridge regression enhances the model's performance on new data while ensuring a more stable solution in the presence of correlated or influential predictor variables.

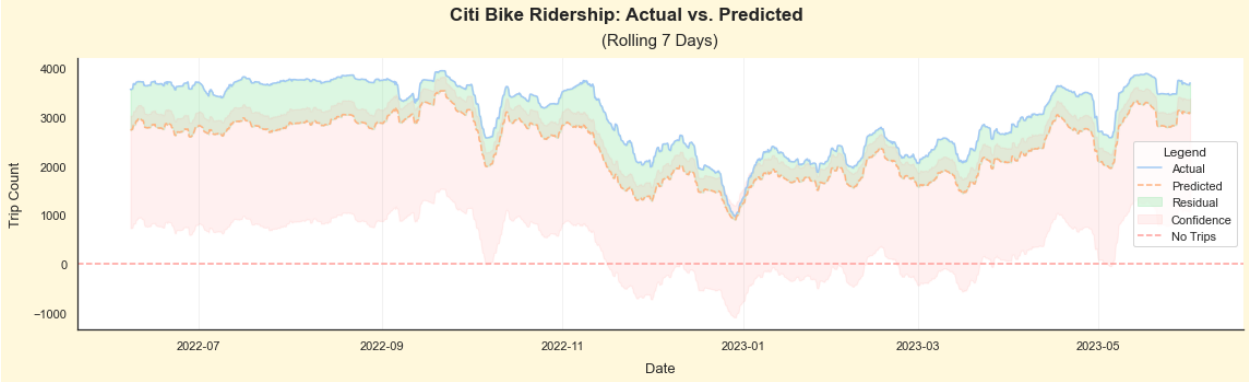
R-squared, also recognized as the coefficient of determination, serves as a statistical measure employed to evaluate the goodness of fit of a regression model. It quantifies the proportion of variance in the dependent variable (the predicted outcome) that can be explained

by the independent variables (predictors) incorporated within the model. Ranging from 0 to 1, a value of 0 implies that the model does not account for any variability, while a value of 1 signifies that the model completely accounts for all variability. Transitioning to model development, the initial phase involved the utilization of Ridge regression along with a grid search to determine the optimal regularization parameter, 'alpha.' The resultant model achieved an Accuracy Rate of 73.23%, accompanied by an R-squared value indicating that the model elucidates 82.60% of the data's variability. Additionally, the MAPE and RMSE were measured at 26.77% and 892.82, respectively.

Continuing the analysis, among the coefficients derived from the model, one of notable significance is the 'bike_counts_log' coefficient, which stands at 0.501. This coefficient represents the transformed log of bicycle counts, and its positive value suggests a substantial positive relationship between the log-transformed bicycle counts and the number of Citi Bike trips. Essentially, as the log-transformed bicycle counts increase, there is a corresponding increase in the predicted number of Citi Bike trips. This insight highlights the influence of higher bicycle counts on the demand for Citi Bikes.

Additionally, the 'pandemic_period' coefficient holds importance with a value of 0.06. This positive coefficient indicates that during specific pandemic periods, there were increases in the predicted number of Citi Bike trips. This aligns with the understanding that the pandemic-induced changes influenced ridership patterns. By analyzing these coefficients, a deeper understanding of the factors impacting the model's predictions is gained, contributing to a comprehensive grasp of their respective contributions.

Following that, the established model was utilized to predict future demand, and a graphical representation was generated to showcase the alignment between actual and predicted values. This visualization also incorporated confidence intervals, as illustrated in Figure 49.



[49]

Much like the visualization depicted in Figure 43, the residuals exhibited a striking similarity to those of the initial ARIMA model. Interestingly, the Anderson-Darling test statistic for the residuals was notably smaller than that of the initial ARIMA model, registering at 106.29 in contrast to the previous value of 121.69. These comparisons are complemented by a summary of critical values at various levels of significance, as outlined in the table below:

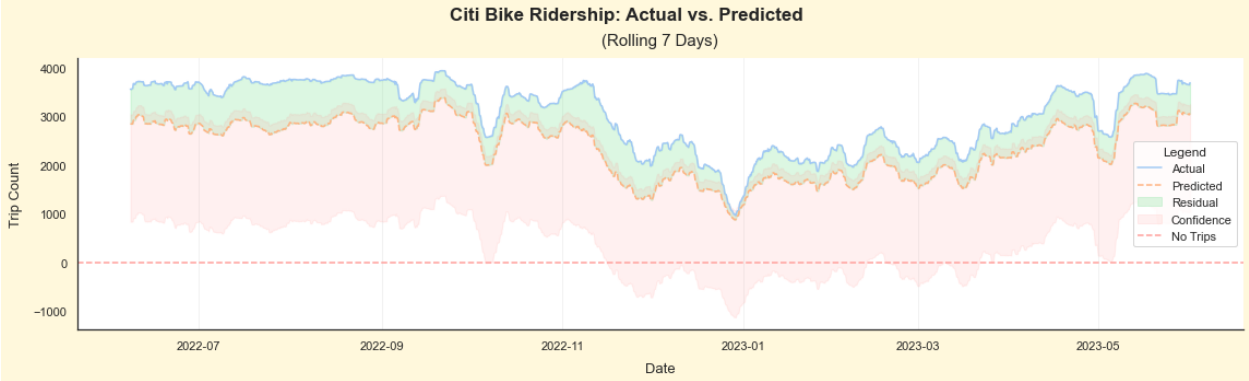
Significance Level	Critical Value
15%	0.576
10%	0.656
5%	0.787
2%	0.918
1%	1.092

[50]

In a manner reminiscent of the approach taken in the time series analysis, a step-back methodology was implemented for the purpose of feature selection. As a result, the resulting model achieved an accuracy rate of 73.28%, relying on twelve predictors—slightly trailing the performance of the ARIMA model, which achieved 73.31% accuracy with 17 predictors. A

[49] Citi Bike Ridership: Actual vs. Predicted (Rolling 7 Days)
[50] Anderson-Darling Test Results

noteworthy discrepancy emerged when considering the root-mean-square error (RMSE) values between the two models. The Ridge regression model displayed an RMSE of 893.52, distinct from the ARIMA model's RMSE of 904.97. This new model was then harnessed for forecasting future demand, presenting a marginal 0.07% enhancement in accuracy when compared to the initial model. A visual depiction of the actual versus predicted values, accompanied by confidence intervals, can be observed in Figure 51. Once more, the residual plots of the second Ridge regression model closely mirrored those of the second ARIMA model.

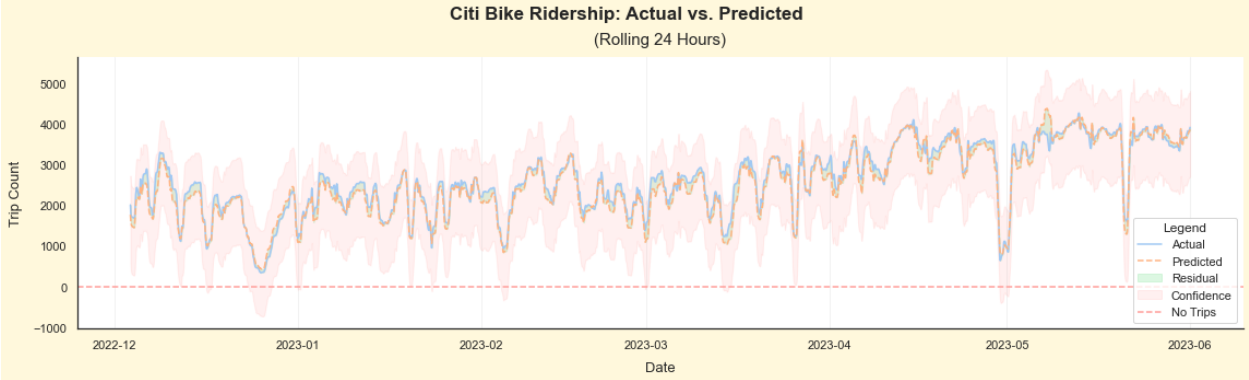


51

The next step delved into testing various forecast horizons to assess their impact on predictive performance. One particularly compelling result is observed with a horizon of 6 months, where the R-squared value soared to an impressive 93.97% from 82.58%, signifying the model's strong explanatory power. Furthermore, the MAPE of 17.66% and RMSE of 494.17 indicated a relatively low margin of prediction error, and the high Accuracy Rate of 82.34% underscored the model's proficiency in making accurate predictions. Comparing this performance to the model discussed earlier, it is evident that this particular forecast horizon demonstrates marked improvement in R-squared, 13.80%, indicating a better fit of the model to the data, as well as notable enhancements in prediction accuracy, as evidenced by the lower

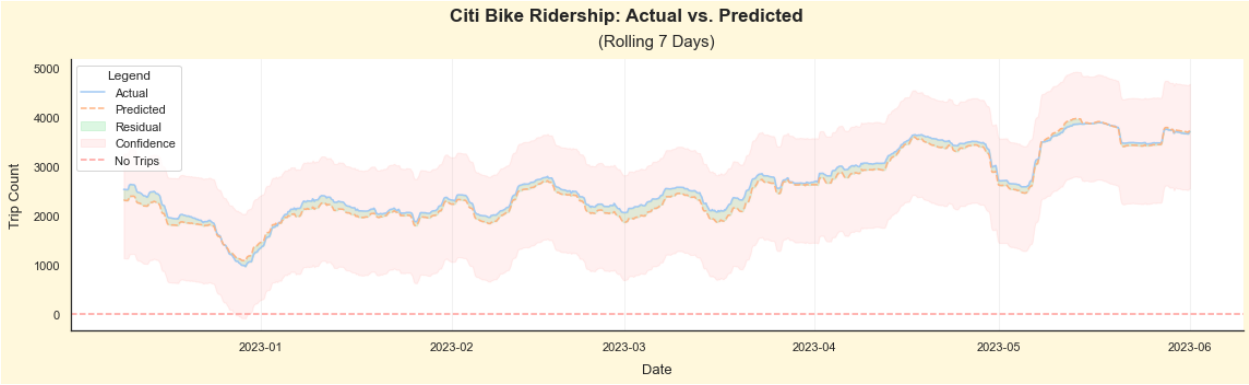
[51] Citi Bike Ridership: Actual vs. Predicted (Rolling 7 Days)

MAPE and RMSE values . This finding highlights the significance of selecting an appropriate forecast horizon to optimize the model's predictive capabilities.



52

The plot in Figure 52 depicts just how closely the model's predicted values match to the actual values. A broader view of the plot, illustrated in Figure 53, shows a rolling 7-day mean. Following the previous residual plot trends, the final model's residual plots also closely resemble those of the final ARIMA model.



53

Additionally, the Anderson-Darling test statistic, 60.02, was almost that of the final ARIMA model of 59.45. The table below shows the critical values at each significance level.

[52] Citi Bike Ridership Actual vs. Predicted (6-Month Horizon, Rolling 24 Hours)
[53] Citi Bike Ridership Actual vs. Predicted (6-Month Horizon, Rolling 7 Days)

Significance Level	Critical Value
15%	0.575
10%	0.655
5%	0.786
2%	0.917
1%	1.091

[54]

Justification and advantages/disadvantages.

The chosen tools for this analysis, including Ridge regression and time series modeling (ARIMA), were selected based on their suitability for addressing specific challenges in predictive modeling. Ridge regression, or L2 regularization, was employed to mitigate multicollinearity and overfitting, enhancing the model's generalization performance by balancing model complexity and variance. This technique introduces a regularization parameter to the linear regression cost function, which prevents coefficients from becoming too large and provides stability in the presence of correlated predictors. Ridge regression's advantage lies in its ability to improve the model's robustness and prevent overfitting by controlling the impact of individual predictors. However, a potential disadvantage is that it assumes that all predictors are relevant, which might not be the case in every scenario.

On the other hand, the use of time series analysis with ARIMA models was appropriate due to the temporal nature of the data, helping to capture patterns and dependencies over time. ARIMA models are adept at handling sequential data, making them valuable for forecasting time-dependent phenomena. One advantage of ARIMA models is their capability to capture complex temporal patterns and trends in data. However, a limitation lies in their sensitivity to data quality, as they assume stationarity, which might require preprocessing steps to ensure accurate modeling.

[54] Anderson-Darling Test Results

The combined use of Ridge regression and ARIMA models in this analysis allowed for a comprehensive exploration of predictive modeling techniques, leveraging the strengths of each to address the unique challenges posed by the dataset. Ridge regression addressed multicollinearity and overfitting, while ARIMA models captured temporal dependencies. This integration of methodologies contributed to a robust analysis and enriched understanding of the dataset's dynamics and predictive capabilities.

Data Summary and Implications

Implications of Data Analysis

The data analysis aimed to predict Citi Bike ridership demand using a multiple linear regression model based on market research data. The analysis involved a multi-faceted approach, utilizing techniques such as seasonal decomposition, log-transformation, autocorrelation and partial autocorrelation plots, Dickey-Fuller test, Ridge regression, and ARIMA modeling. These techniques collectively allowed for a comprehensive exploration of the dataset's patterns, dependencies, and predictive capabilities.

The Ridge regression model was employed to address multicollinearity and overfitting, enhancing the model's generalization performance by finding a balance between model complexity and variance. The results showed that the Ridge regression model achieved an accuracy rate of around 82.34%, and it provided valuable insights into the relationships between predictor variables and Citi Bike ridership demand. Notably, variables like transformed log of bicycle counts and pandemic periods exhibited significant positive relationships with the predicted number of Citi Bike trips.

The analysis also explored various forecast horizons, revealing that a forecast horizon of 6 months led to a substantial improvement in R-squared value, indicating strong explanatory power, as well as lower MAPE and RMSE values, signifying enhanced prediction accuracy.

Limitation of Analysis

One limitation of the analysis is that the Ridge regression model assumes that all predictors are relevant, which might not always hold true. It's possible that certain predictor variables may have limited impact on Citi Bike ridership demand, and the model might benefit from more rigorous feature selection techniques.

Recommendation

Based on the results of the analysis, I recommend leveraging the insights gained from the Ridge regression model to inform decision-making and strategies related to Citi Bike operations and ridership management. The model's ability to highlight significant predictor variables and their relationships with ridership demand can aid in optimizing resource allocation, marketing efforts, and infrastructure improvements.

Future Directions for Study

Advanced Feature Selection:

To address the limitation of assuming all predictors are relevant, future studies could employ more advanced feature selection techniques. Techniques like Recursive Feature Elimination (RFE) or LASSO regression could help identify the most influential predictors and improve the model's accuracy.

Dynamic Time Series Models:

While ARIMA models are effective for capturing temporal patterns, they might not account for sudden shifts or changes in trends. Future studies could explore more advanced time series models, such as state-space models or machine learning-based approaches like LSTM networks, which can better capture complex temporal dependencies and adapt to changes in the dataset.

In summary, the data analysis provides valuable insights into predicting Citi Bike ridership demand using a Ridge regression model and exploring forecast horizons. While the Ridge regression model's performance and insights are promising, addressing predictor relevance and exploring more advanced modeling techniques could further enhance the accuracy and robustness of predictions.

Sources

Bronx Times. (2020, May 5). *First Citi Bike stations installed in the Bronx as part of continued expansion*. Retrieved from https://www.bxtimes.com/first-citi-bike-stations-installed-in-the-bronx-as-part-ofcontinued-expansion/

Citi Bike NYC. (2023). *System Data*. Retrieved from https://www.citibikenyc.com/system-data

(DOT), D. of T. (2023, August 25). Bicycle counts: NYC open data. *Bicycle Counts | NYC Open Data*. Retrieved from https://data.cityofnewyork.us/Transportation/Bicycle-Counts/uczf-rk3c

Faghih-Imani, A., Hampshire, R., Marla, L., & Eluru, N. (2014). *An empirical analysis of bike sharing usage and rebalancing: Evidence from Barcelona and Seville*. *Transportation Research Part A: Policy and Practice*, 71, 326-344.

Field, A. (2013). *Discovering statistics using IBM SPSS statistics*. Sage.

Gebhart, K., & Noland, R. B. (2014). *The impact of weather conditions on bikeshare trips in Washington, DC*. Transportation, 41(6), 1205-1225.

Hyndman, R. J., & Athanasopoulos, G. (2018). *Forecasting: principles and practice*. OTexts.

Investopedia. (2020, October 5). *Timeline: How COVID-19 Took Over NYC*. Retrieved from https://www.investopedia.com/historical-timeline-of-covid-19-in-new-york-city-5071986

Master's in Data Science. (n.d.). *How to Deal with Missing Data*. Retrieved from https://www.mastersindatascience.org/learning/how-to-deal-with-missing-data/

National Constitution Center. (n.d.). *How Martin Luther King Jr.'s birthday became a holiday*. Retrieved from https://constitutioncenter.org/blog/how-martin-luther-king-jr-s-birthday-became-a-holiday-3

New York Times. (n.d.). *Fodor's Travel Guide to New York City*. Retrieved from https://archive.nytimes.com/www.nytimes.com/fodors/top/features/travel/destinations/unitedstates/newyork/newyorkcity/fdrs_tips_111_2.html

Phudinawala, H., & Jha, M. (2022). *Prediction of Demand for Shared Bikes using Multiple Linear Regression Model*. *International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)*, 2(1). Retrieved from https://ijarsct.co.in/Paper2577.pdf

Protected bike lane tracker. *Transportation Alternatives*. (2022, September 26). https://projects.transalt.org/bikelanes#:~:text=The%20NYC%20Streets%20Plan%20requires,30%20required%20miles%20in%202022

Ricci, M. (2019). *Bike Sharing: A Review of Evidence on Impacts and Processes of Implementation and Operation*. *Sustainability*, 11(12), 3314. Retrieved from https://doi.org/10.3390/su11123314

Visual Crossing. (2023). *Weather Data*. Retrieved from https://www.visualcrossing.com/weather-data

Yang, H., Xie, K., Ozbay, K., Ma, Y., & Wang, Z. (2018). *Use of Deep Learning to Predict Daily Usage of Bike Sharing Systems*. *Transportation Research Record*, 2672(36), 92–102. Retrieved from https://doi.org/10.1177/0361198118801354

Zamir, K. R., Bondarenko, I., Nasri, A., Brodie, S., & Lucas, K. (2019). *Comparative Analysis of User Behavior of Dock-Based vs. Dockless Bikeshare and Scootershare in Washington, D.C.* Retrieved from https://arxiv.org/pdf/1907.11526.pdf