Citi Bike Users

Exploring Patterns and Preferences in Urban Bikesharing

Daniel Craig, John Cruz, Shaya Engelman, Noori Selina, Gavriel Steinmetz-Silber

2024-05-07

Contents

Abstract

Achieving sustainable and equitable transportation systems is a pressing goal for urban areas, given the challenges posed by increasing motor vehicle use and limitations of traditional public transit. Bikesharing has emerged as a promising solution, offering flexibility, affordability, and sustainability benefits. This study focuses on predicting and classifying trip types by members versus casual users in the Citi Bike program, the largest bikeshare provider in the US. Using logistic regression and machine learning techniques, the study analyzes ride data alongside weather variables to identify factors influencing bikesharing usage patterns. Insights from the analysis reveal that electric bike usage and longer ride durations are significant predictors of membership behavior. Further investigation through Generalized Additive Models and Linear Discriminant Analysis elucidates non-linear relationships between weather conditions and bike type usage. Despite moderate model performance, the findings offer valuable insights for bikeshare operators and policymakers to optimize system design and promote membership growth through targeted interventions.

Keywords

Bikesharing, Physical Activity, Classification, Boosted Logistic Regression,

1. Introduction

Achieving more sustainable and equitable modes of transportation is an important objective of the US Transportation system. Cities are overburdened with motor vehicles, with diminishing returns due to high traffic and lack of available parking. Public transit systems, while crucial for many, can suffer from overcrowding, delays, limited accessibility, and the notorious "first and last mile" problem, where accessing transit stops from home or work can be challenging.

Bikesharing has emerged as a promising alternative mode of transportation in urban environments, offering flexibility, affordability, and accessibility to commuters and tourists alike. Its potential to alleviate the strain on traditional transportation infrastructure while promoting sustainability and healthier lifestyles has garnered significant attention from researchers and policymakers worldwide. As bikesharing continues to expand, understanding the factors influencing its usage patterns and membership dynamics becomes imperative for optimizing system design and promoting urban mobility. Citi Bike, owned by Lyft, is a privately owned public bicycle sharing system serving the New York City boroughs of the Bronx, Brooklyn, Manhattan, and Queens, as well as Jersey City and Hoboken, New Jersey. It is the largest bikeshare provider in the US with [over 30 million rides taken in 2022 alone][1] and a fleet of greater than 33,000 bicycles as of September 2023.[2]

This study aims to predict and classify the types of trips done by members versus casual users. If this can be accurately predicted, bikeshare programs can utilize this information to promote an upgrade to a membership tier in cases where a route that is expected to be taken by a member is taken by a casual user. This can be very useful for both the consumer and the program. The program benefits from increased subscriptions and membership fees. Additionally, regression techniques reveal an exponential "spillover" relationship in membership growth rates (Schoner et al. (2016)[3]), leading to even further growth. Consumers benefit from being targeted and made aware that they are likely to benefit from being a member. Furthermore, many consumers end up paying more in Single-Trip Fares (STF) than they would by just purchasing a membership. These interventions would minimize that.

2. Literature Review

Fischman et al. (2015)[4] investigated the factors influencing bikeshare membership in Melbourne and Brisbane, revealing insights into the demographic and behavioral differences between members and casual users. They utilized a logistic regression model to reveal several significant predictors of membership including reactions to mandatory helmet legislation, riding activity over the previous month, and the degree to which convenience motivated private bike riding. In addition, respondents aged 18–34 and having docking stations within 250.m of their workplace were found to be statistically significant predictors of bikeshare membership. Finally, those with relatively high incomes increased the odds of membership. Their study was done completely from the users' perspective and how they responded to a set of survey questions.

Kaviti et al. (2019)[5] conducted a comprehensive study on travel behavior and price preferences among bikesharing members and casual users, highlighting differences in trip purposes, demographic profiles, and pricing preferences. They created a custom survey and used logistic regression methods to generate insights. Their analysis revealed significant associations between membership status, ethnicity, income, and trip behaviors, offering valuable insights for bikesharing operators and policymakers.

On the other hand, Reilly et al. (2020)[6] found that low-income people are more likely to become members of bikesharing programs due to having a smaller rate of car ownership. This challenges the findings of other studies that show that higher income earners were more likely to be members.

Chen et al. (2024)[7] delved into the causes of transportation inequality in bikesharing usage, highlighting affordability as a primary barrier for lower-income earners. Their study challenged conventional assumptions regarding infrastructure availability, emphasizing the need for targeted interventions to address socioeconomic disparities in bikesharing access.

By synthesizing findings from these studies, it becomes evident that bikesharing usage patterns are shaped by a complex interplay of demographic, behavioral, and socioeconomic factors. While previous studies have focused on survey responses and user behaviors, there remains a gap in understanding how immutable ride characteristics can predict trip types. By shifting the focus to ride data analysis, we aim to provide a complementary perspective that enhances our understanding of bikesharing usage patterns and can then be used in conjunction with the user focused studies to enhance models and predictions.

3. Methodology

3.1 Citi Bike Data

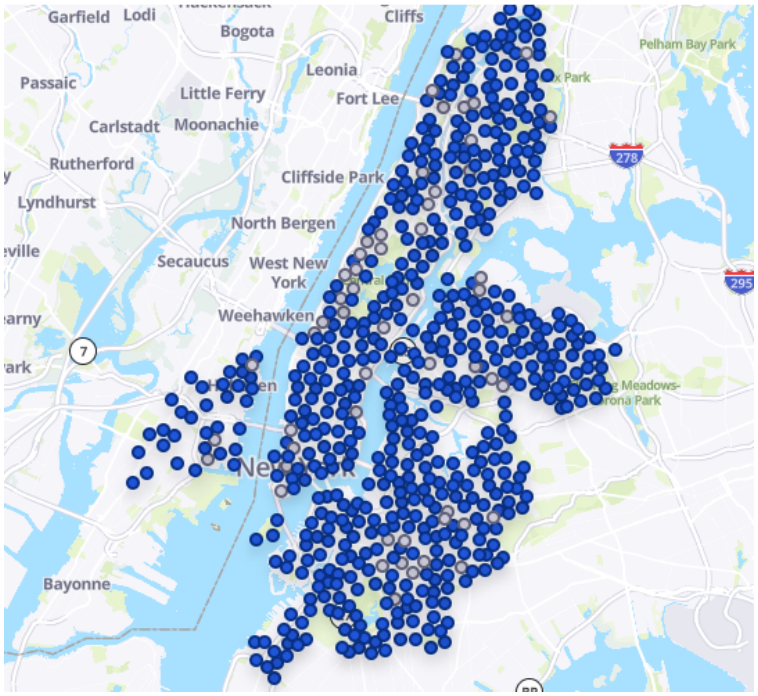


Figure 1: Distribution of Citi Bike Bike-Docks

While the aim of this study is to provide insight into all bikeshare networks and apply findings on a large scale, the study's focus is on the Citi Bike program in NYC. Citi Bike services the five boroughs of NYC as well as Jersey City and Hoboken, NJ. They provide an open data that gives people access to some of their system data of how users use their services. This includes station information, latitude and longitude, and ride types. For this analysis, data for March 2024 is used. While including more months might lead to more robust results, these files are around 1 GB and computational space needs to be considered.

The available data contains 2,737,881 records of the following variables:

Trips Predictor Variables

- `rideable_type`: type of bike rented (electric or classic)
- `started_at`: datetime rental was taken from the station
- `ended_at`: datetime rental was returned to a station
- `start_station_name`: bike taken from station
- `start_station_id`:
- `end_station_name`: bike returned to station

- `end_station_id:`

- `start_lat:` starting station latitude
- `start_lng:` starting station longitude
- `end_lat:` ending station latitude
- `end_lng:` ending station longitude

Response Variable

- `member_casual:` whether the rental was used by a member or casual (one-time rental) user.

The response variable was encoded as 1 for member and 0 for casual user. In addition to the above columns, the data also contains an index column, `ride_id`, this was dropped since it contains no value for prediction.

The data has some missing values in important columns, for example, some records are missing station names and IDs. More importantly, the data is also missing ending latitude and longitude information for some records. These records will likely have to be removed, assuming it cannot be determined where the rider ended their ride.[Table of Missing Values]

Developing Predictor Variables

An important predictor for the target variable is likely distance traveled and usage time. Ideally, these would be calculated precisely using the latitude and longitude values already contained in the dataset. This can be done using a query to `gmapsdistance` that would calculate the distance traveled based on the mode of transportation. This method would likely provide the most accurate results to determine traveled distance on these bikes. However, due to monetary constraints this is not a feasible option for this study. Instead the values for these variables are estimated based on the data where classic bikes travel around 8.3 miles per hour and electric bikes travel up to 20 miles per hour[8]. To account for the crowded nature of NYC, the maximum speeds on the electric bikes are arbitrarily limited to 15 miles per hours for this calculation. The results are the following two new predictor variables in our dataset:

Trips Predictor Variables

- `usage_time:` how long the ride was for (in seconds)
- `est_distance:` estimated traveled distance based on bike usage (in miles)

3.2 Weather Data

To supplement the Citi Bike Data and improve model fit and accuracy, hourly weather data was obtained and included for each ride. This was downloaded via Oikolab's API. It includes temperature, precipitation, humidity and wind speed. The weather data was in metric system and converted to imperial (US) standards. This includes the temperature as Fahrenheit, and total precipitation as inches.

Weather Predictor Variables

- `temp_deg_f:` temperature (Fahrenheit)
- `rel_humidity:` relative humidity
- `total_precip:` total precipitation (inches)
- `wind_speed:` wind speed (miles per hour)
- `day_of_week`: day of the week (Monday, Tuesday...)

The weather data was reviewed for missing values and none were found. It was then merged with the Citi Bike data to create one single file with all the data used for this study.[Table of Missing Values]

3.3 Data Summary

Reviewing a summary of the numeric variables in the dataset revealed some data integrity issues. In particular, `est_distance` and `usage_time` appeared to have severe negative values, neither of those should have been possible. These would need to either be replaced with imputed values or those records would need to be dropped from the dataset. Additionally, a summary table reveals interesting distributions in some of the, such as a slight right skewness in the wind speed value, indicating that higher speed winds are less frequent but within a reasonable range, and a fascinating distribution to the total precipitation column, with a majority of values concentrated at 0 and a few extreme values. While this is logical, as most days there is no rain, it is an interesting distribution nonetheless.[Summary Table for Variables]

Density

To better analyze the data, the distributions and skewness of the variables were plotted to help with visualization. The visualizations showed right skew in `wind_speed_mph`, `usage_time` and `est_distance`. Meanwhile, a Poisson distribution was observed in `total_precip`, as mentioned above, this is a zero inflated Poisson distribution. These skewed variables were deemed as likely suitable candidates for transformation. Additionally, the plots indicated that rel_humidity exhibited a multi-modal distribution, with peaks occurring roughly every 20% increase in humidity. Furthermore, there were apparent outliers present in the data, which necessitated further investigation.[Density Plots]

Boxplot

Boxplots provided additional insights into the distributions of the variables, further confirming the skewness and variability discussed earlier. Additionally, they revealed the presence of numerous outliers across many columns, particularly in `est_distance`, `total_precip`, and `usage_time`.[Boxplots]

Correlation Matrix A correlation matrix was employed to assess the relationships between variables in the dataset.

- **Negative Correlations:** Predictors `wind_speed_mph` and `rel_humidity` exhibited negative correlations with each other, indicating that as the relative humidity increases, the likelihood of the wind speed exceeding the median decreases. This is interesting, as one might have assumed more humidity brings a higher chance of rain and also wind speeds.
- **Positive Correlations:** Conversely, predictors such as `usage_time` and `est_distance` exhibited strong positive correlations with each other, which is logical given that distance traveled is derived from the duration of bike rides. `rel_humidity` and `total_precip` also had a positive relationship, which intuitively makes sense.[Correlation Matrix]

Class Imbalance

Finally, an assessment was conducted to determine whether the classes of the `member_casual` variable were balanced to prevent misleading models. Imbalanced classes, such as a ratio of 95 to 5 between success and fail rates, can lead to inaccurate model predictions. For instance, if a model predicts success 100 of the time, it may achieve a 95 success rate without providing any meaningful insights. It was observed that the majority of users were members rather than casual non-members. Therefore, careful consideration was required regarding which evaluation metrics to prioritize when assessing model performance. In addition, the ratio of electric bikes versus classic bikes was imbalanced. Electric bikes were utilized roughly twice as frequently as classic bikes were. This difference in usage rates may be attributed to factors such as inventory availability or user preferences regarding the type of bike they select for their trips.[Class Balance Plots]

4. Data Processing

4.1 Data Cleaning

Prior to conducting our analyses, data cleaning procedures were implemented to ensure the integrity and reliability of our dataset. Missing values were systematically removed, adhering to best practices in data

preprocessing. Subsequently, the dataset was partitioned into distinct training and testing sets, employing a conventional 70/30 ratio. This methodological approach was adopted to mitigate the risk of data leakage and to facilitate robust model evaluation.

4.2 Transformations

To address potential skewness and non-normality within certain variables, particularly 'usage time' and 'estimated distance', the bestNormalize function was employed. This algorithmic tool identifies optimal transformations to achieve normal distribution assumptions. By fixing how the data is distributed, we can mitigate the impact of outliers and improve the performance of statistical analyses that rely on assumptions of normality. This ensures more reliable inference and enhances the robustness of our findings. Moreover, normalizing the data simplifies interpretation and comparison across variables, facilitating clearer insights into the underlying patterns and relationships within the dataset.

Upon the determination of suitable transformations through the bestNormalize function, prescribed methods such as logarithmic and square root transformations were systematically applied. These mathematical operations were instrumental in normalizing variable distributions, a pivotal step towards enhancing model accuracy and reliability. Through systematic transformation procedures, the underlying assumptions of statistical models were upheld, thereby fortifying the validity of ensuing analyses.

4.3 Outliers

In contemplating the treatment of outliers within the dataset, a deliberate decision was made to retain these anomalous data points. This strategic choice was predicated on the recognition of the potential insights and information encapsulated within outlier observations and on deeming them to be true values. By preserving these data extremes, our analytical framework aimed to capture the entirety of data variability, thereby enriching the interpretative depth and robustness of subsequent analyses.

In sum, the meticulous execution of data processing procedures encompassing cleaning, normalization, transformation, and outlier handling served to ensure the integrity and reliability of the dataset. This comprehensive approach enabled us to produce a solid foundation for subsequent analyses and conclusions.

5. Model Building - Part #1

Given the nature of this problem, this was essentially a classification task. Therefore classification models were deemed suitable. A variety of classification models were employed to attempt to achieve the best results.

5.2 Binary Logistic Regression

The first method attempted was a classic logistic regression model. Feature selection was performed to identify relevant predictors. In addition, since a key goal of this model was to identify features for use in later iterations, interpretability was an important metric. For this reason, due to station name being a categorical feature with greater than 2000 unique values it was not included in the first attempt. The station name made the dataset too bulky and to work with. The variables included in this logistic regression model were rideable type -electric or classic bicycle-, temperature, humidity, precipitation, wind speed, day of week, usage time, estimated distance and time of day.

The model was validated using 5 fold cross validation and the sensitivity and specificity were calculated. The model achieved a sensitivity of 0.63 and specificity of 0.65. Since we wanted to identify the different outcomes across various thresholds, the ROC was selected as a key matric. The model achieved an ROC of 0.7. While this is not a perfect model, it is important to note that a model built on all the various variables in the original dataset achieved a sensitivity of 0.03 and a specificity of 0.997. This model was clearly overfitting and the model with the selected variables was a significant improvement.

Before evaluating the model, the model was checked for multicollinearity. It was hypothesized that electric bike users were more likely to ride longer distances and at night, resulting in multicollinearity. The VIF was calculated and it was found that there was indeed extremely significant multicollinearity between electric bikes and distance riden. This was expected and the model was rerun without the distance variable. The model achieved extremely similar metrics to the previous model, with a sensitivity of 0.65, specificity of 0.64 and an ROC of 0.7. The VIF values were again calculated and it was found that there was no multicollinearity in the model.

While not a perfect model, it did clearly indicate some important factors in determining whether a user was a member or a casual user. The model indicated that electric bike users were more likely to be members, increasing the log odds by 0.424. Additionally, the model indicated that for every 1 degree increase in temperature, the log odds of a user being a member increased by approximately 0.103. It is important to note that this does not indicate a causal relationship, merely a correlation. For example, it is possible that electric bike users are more likely to be members beacuse they already paid the membership fee while for casual users, the higher costs of electric bikes steered them towards classic bikes. Also, possibly on nicer days, members are quicker to think of biking as an activity. Another interesting finding was that weekend days were more likely to have member users. Members were also found to have a higher usage time, and only had a positive relationship with afternoon rides.

This model used 0.5 as the threshold for determining whether a user was a member or a casual user. However, this threshold can be adjusted to better fit the needs of the bikeshare program. For example, if the program is looking to increase membership, the threshold could be lowered to increase the number of users identified as members. This would increase the number of users that the program could target for membership promotions. An ROC curve visualization is particularly useful in determining the tradeoffs between sensitivity and specificity at different thresholds. This can be used to determine the best threshold for the program's needs.[Sensitivity vs Specificity Plot]

The model was evaluated by predicting the test set and generating the confusion matrix. The model had mixed results. It achieved an accuracy of 0.64, however, the no-information rate was 0.84, indicating that better results would be achieved by merely predicting the majority class every time. Even so, the model was not completely useless. It achieved a negative predictive value of 0.91, meaning, if the model predicted a user was a member, it was pretty reliable. The model also achieved an AUC of 0.64, indicating that the model **was** better than random chance.[ROC Curve]

5.2 Boosted Logistic Regression

To try and improve upon the original model's results, another method was implemented. A boosted logistic regression model was used. This model was chosen because it is a powerful tool for classification problems and can handle large datasets with many predictors. The simple logistic regression models created thus far exclude the inclusion of station names due to their size and difficulty in management. The boosted logistic regression model was trained on many more variables in the dataset, including station names.

Due to the size of the dataset and computational restraints, a stratified sample was taken from the original dataset. This sample preserved the ratio of casual to member records. To deal with the imbalance in the dataset, class weights were assigned. A higher weight was assigned to casual users, 0.9, and a lower weight was assigned to members, 0.1. The data was centered and near-zero variance predictors were removed. The boosted model was trained using 5-fold cross validation, optimizing for accuracy and robustness.

Upon training the model, its predictive performance was evaluated using various metrics, including accuracy, sensitivity, specificity, and the confusion matrix. Despite achieving an 84% accuracy rate, deeper analysis revealed significant shortcomings in Sensitivity and Balanced Accuracy, with values of 0.05 and 0.52, respectively. Despite incorporating station names and class weights, the model demonstrates limited improvement over simplified logistic models. Further investigation via ROC curve analysis highlights challenges in achieving balanced performance metrics, with marginal gains in Sensitivity necessitating significant sacrifices in Specificity. Overal the AUC was a disappointing 0.384.[ROC Curve]

The decision was made to continue with the original logistic regression model as the model to classify whether a given ride was more likely taken by a casual user or member. While not an ideal model, the logistic regression model was both, more accurate and more interpretable than the boosted model. The boosted model was more accurate, but the specificity and sensitivity were both very low.

6 Model Building - Part #2

In the previous sections, logistic regression models were used to classify whether a given ride was more likely taken by a casual user or a member. While not perfect models, the logisitic regression model did reveal some valuable insights. In particular, two variables were found to be significant predictors of whether a user was a member or a casual user. 1. rideable type. Using an electric bike increased the log odds of being a member compared to using a regular bike. As discussed, this is likely due to the pricing around electric bikes–they are especially expensive for casual users. 2. usagetime_transformed. Longer usage times likewise increased the log odds of being a member. This may be because members pay for a subscription, and so they more readily consider bikes as a mode of transportation.

Both of these results are quite intuitive, and they also present a business opportunity. This is to say, the model has identified "member behavior", or at least two aspects of it. Therefore, if it can be predicted *when* casual members are more likely to exhibit this "member behavior," then businesses can focus their marketing efforts on those very times. For example, suppose people are more likely to use e-bikes when it's pleasant weather. Well then, businesses could send out special deals right as the temperature is getting to be pleasant. In short, by figuring out when casual users act like members, they can more readily be converted to actual members.

6.1 Generalized Additive Model

The first step in this process is to identify when casual users are more likely to act like members. To test the above findings, the data was filtered to only include casual users. Then the distributions of the temperature by bike type and precipitation by bike type were analyzed using visualizations and summary tables.[Distributions of Temperature and Precipitation] The results confirmed the above hypothesis. Casual users were more likely to use electric bikes st higher temeratures and greater precipitation, likely to reach their destination faster when it is uncomfortable to be out. To further test this hypothsis, the probability of choosing an electric bike was plotted against the temperature.[Plot of Temperature] As expected there was a non-linear relationship. It appears that the likelihood of using an e-bike drops as the temperature rises at first, then plateaus, and at the very end rises as the temperatures increase. This is consistent with our hypothesis that people are more prone to using e-bikes in uncomfortable. The findings of a non-linear relationship suggest splines should be utilized in subsequent models.

A Generalized Additive Model (GAM) was then employed to further investigate these relationships. The GAM model is particularly suited for this task as it can handle non-linear relationships through the use of smooth functions, such as splines. This allows us to capture the observed non-linear relationship between temperature and the likelihood of using an electric bike. To address the class imbalance in bike type usage, undersampling was again employed. A model was fitted with rideable type as the response variable and temperature, humidity, precipitation, wind speed, day of week, estimated distance and time of day as predictor variables.

The results of the model were surprising. The only variables with a positive relationship with electric bike use were weekend bike rides. Friday, Saturday and Sunday were all more likely to be electric bike rides. This suggests that bikesharing companies should attempt to market memberships to casual users more frequently before weekends and emphasize members' e-bike savings.

However, the model was not great as a whole. It explained only 5.33% of the deviance–the key metric for binary logistic models. While not a great predictor, the model nevertheless serves a purpose in identifying some key variables in predicting bike type.

6.2 Linear Discriminant Analysis

To attempt to achieve a better result, a Linear Discriminant Analysis (LDA) model was employed. LDA is a classification method that is particularly useful when the response variable is binary. Since LDA becomes hard to interpret with too many features, only a select few were used. The GAM model results were used to help identify likely important features. Those features were; estimated distance, time of day, day of week, temperature and total precipitation.

The model was trained using 10-fold cross validation and the results were evaluated. The model achieved an accuracy of 0.601 and an AUC of 0.646.[LDA ROC Curve] While also not a great model, it was an improvement over the GAM model. However, visualizing the distributions of the LDA prediction scores was disappointing. The model did not seem to separate the classes too well[LDA Predictions].

7 Findings and Limitations

This study aimed to predict and classify the types of trips done by members versus casual users in the Citi Bike bikesharing program. Through a comprehensive analysis of bikesharing data from March 2024, coupled with weather data, various models were built and evaluated to identify significant predictors of membership and casual usage.

7.1 Findings

The logistic regression model revealed that electric bike usage and longer ride times were significant predictors of membership. This suggests that members are more likely to use electric bikes and ride for longer durations compared to casual users. These findings can inform targeted marketing strategies to promote membership upgrades and enhance user engagement. The boosted logistic regression model, while more accurate, demonstrated limited improvements in sensitivity and specificity, necessitating further refinement to achieve balanced performance metrics. The GAM and LDA models provided additional insights into the relationships between temperature, precipitation, and bike type usage, highlighting the potential for non-linear relationships and the importance of feature selection in classification tasks.

7.2 Limitations

Despite the promising results, several limitations were encountered during the analysis. The imbalanced class distribution of member and casual users posed challenges in model evaluation, necessitating the use of class weights and undersampling techniques. The limited sample size and computational constraints restricted the scope of the analysis, potentially affecting the generalizability of the findings. Furthermore, the absence of additional demographic and behavioral data may have limited the predictive power of the models, warranting further investigation into user preferences and trip behaviors. Future research should explore additional predictors and feature engineering techniques to enhance model performance and interpretability.

7.3 Future Directions

Moving forward, future research should focus on incorporating additional data sources, such as user surveys and trip histories, to enrich the predictive models and enhance user segmentation. This study was always supposed to be a supplementary analysis to the traditional user-focused studies. Combining the immutable ride characteristics with insight gained from user-focused studies would greatly enhance the predictive power of the models. Furthermore, the integration of real-time data streams and advanced machine learning algorithms, such as neural networks and ensemble methods, could further improve model accuracy and robustness. By leveraging the latest advancements in data science and predictive analytics, researchers and

policymakers can gain deeper insights into bikesharing usage patterns and membership dynamics, fostering sustainable urban mobility and equitable transportation systems.

8 Conclusion

In conclusion, this study aimed to predict and classify the types of trips done by members versus casual users in the Citi Bike bikesharing program. Through a comprehensive analysis of bikesharing and weather data, logistic regression, boosted logistic regression, GAM, and LDA models were built and evaluated to identify significant predictors of membership and casual usage. While the models demonstrated varying degrees of predictive power, the logistic regression model revealed that electric bike usage and longer ride times were significant predictors of membership. These findings can inform targeted marketing strategies to promote membership upgrades and enhance user engagement. Despite the limitations encountered, this study provides valuable insights into bikesharing usage patterns and membership dynamics, laying the groundwork for future research and policy interventions in urban mobility and sustainable transportation systems.

Citations and References

1. A. Ley. Citi Bike Service Is Worse in Low-Income Neighborhoods, Study Finds

2. Citi Bike September 2023 Monthly Report

3. J. Schoner et al. Is Bikesharing Contagious? Modeling Its Effects on System Membership and General Population Cycling

4. E. Fishman et al. Factors influencing bike share membership: An analysis of Melbourne and Brisbane Transportation Research Part a: Policy and Practice (2015)

5. S. Kaviti et al. Travel behavior and price preferences of bikesharing members and casual users: A Capital Bikeshare perspective

6. K. H. Reilly et al. From non-cyclists to frequent cyclists: Factors associated with frequent bike share use in New York City

7. J. Chen et al. Causes of transportation inequality: The case of bike sharing in the U.S.

https://en.wikipedia.org/wiki/Citi_Bike

Appendix 1: Tables and Plots

Table of Missing Values for Citi Bike Data

Field Name	Missing Count
rideable_type	0
started_at	0
ended_at	0
start_station_name	2869
start_station_id	2948
end_station_name	5910
end_station_id	6411
start_lat	0
start_lng	0
end_lat	751
end_lng	751
member_casual	0
usage_time	0
est_distance	0

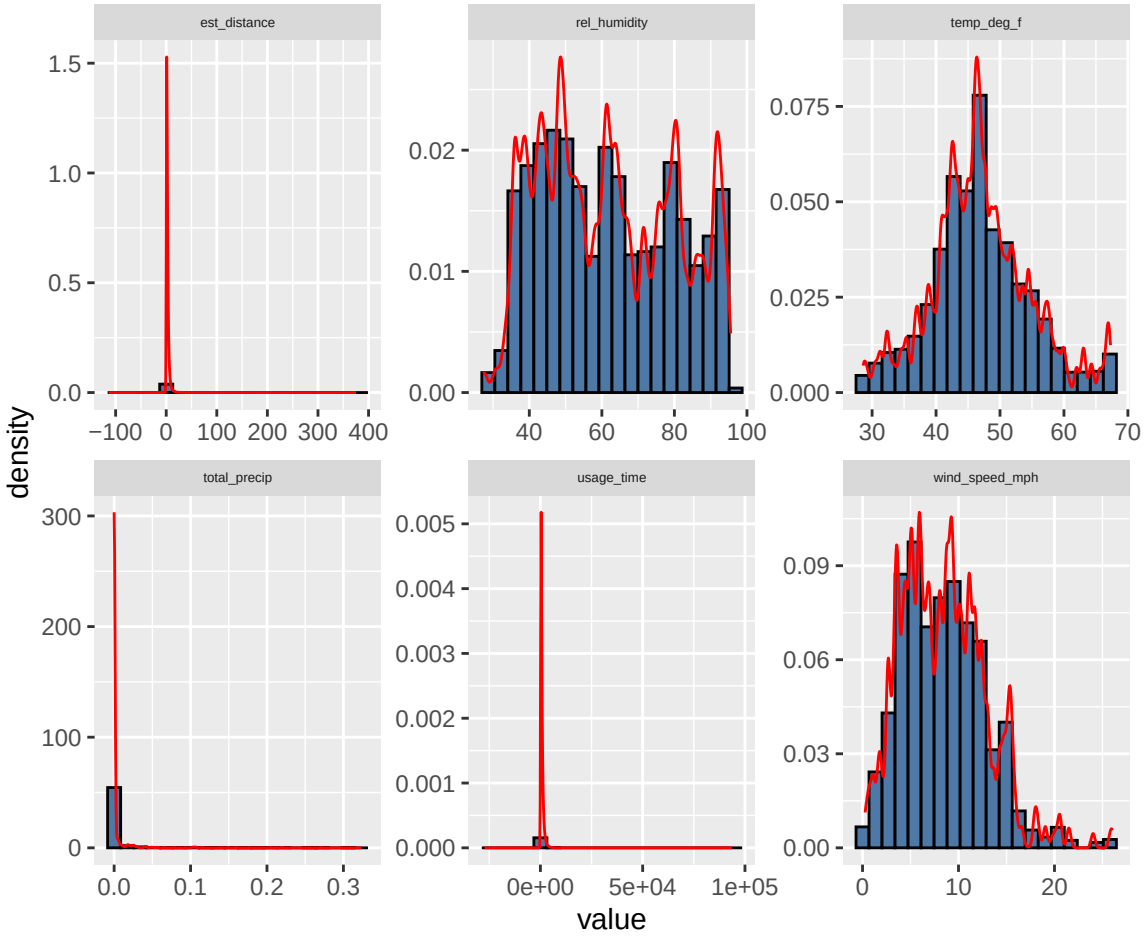
Table of Missing Values for Weather Data

Field Name	Missing Count
datetime_ny	0
datetime_utc	0
coordinates_lat_lon	0
model_name	0
model_elevation_surface	0
utc_offset_hrs	0
temperature_deg_c	0
dewpoint_temperature_deg_c	0
wind_speed_m_s	0
total_cloud_cover_0_1	0
total_precipitation_mm_of_water_equivalent	0
temp_deg_f	0
rel_humidity	0
total_precip	0
wind_speed_mph	0
day_of_week	0

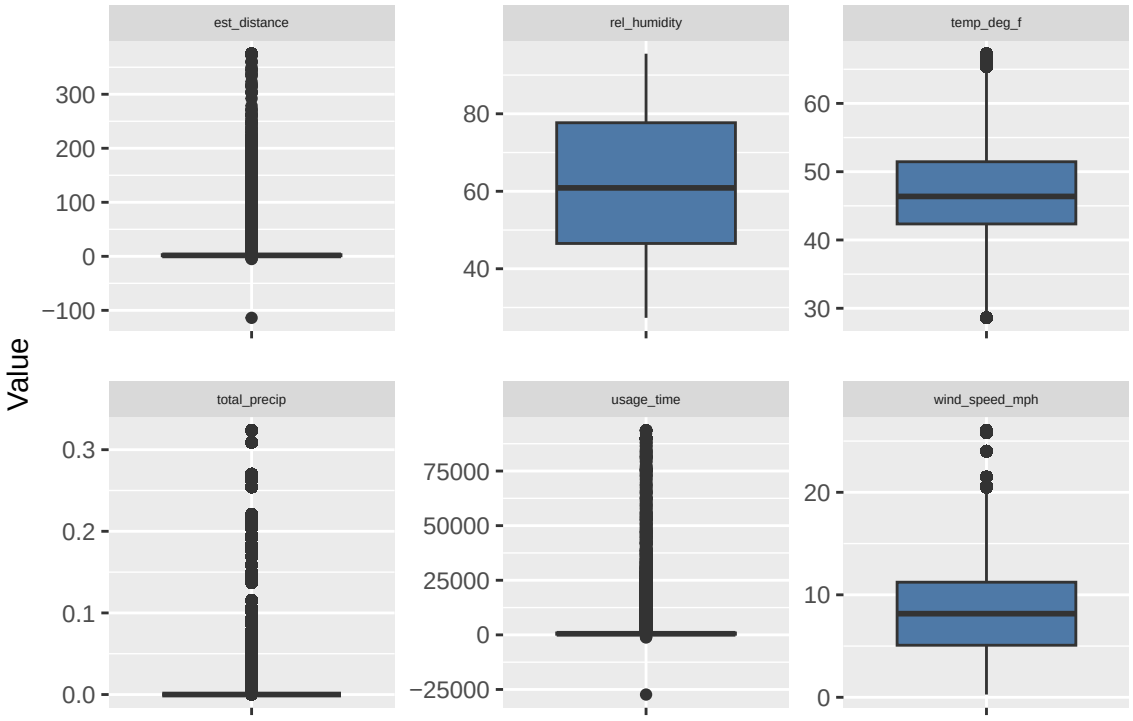
Summary Table of Variables in Dataset

	est_distance	rel_humidity	temp_deg_f	total_precip	usage_time	wind_speed_mph
Mean	2.63	61.87	46.95	0.00	757.83	8.44
Std.Dev	5.79	18.02	7.62	0.02	1970.00	4.35
Min	-113.83	27.33	28.60	0.00	-27320.00	0.27
Q1	0.93	46.53	42.35	0.00	283.00	5.08
Median	1.72	60.88	46.38	0.00	498.00	8.16
Q3	3.10	77.69	51.46	0.00	877.00	11.23
Max	375.06	95.51	67.28	0.32	93596.00	26.04
MAD	1.41	23.15	6.58	0.00	385.48	4.55
IQR	2.17	31.17	9.11	0.00	594.00	6.15
CV	2.20	0.29	0.16	5.21	2.60	0.51
Skewness	30.80	0.20	0.28	8.00	34.59	0.67
SE.Skewness	0.00	0.00	0.00	0.00	0.00	0.00
Kurtosis	1406.33	-1.15	0.26	73.62	1468.19	0.69
N.Valid	2737881.00	2737881.00	2737881.00	2737881.00	2737881.00	2737881.00
Pct.Valid	100.00	100.00	100.00	100.00	100.00	100.00

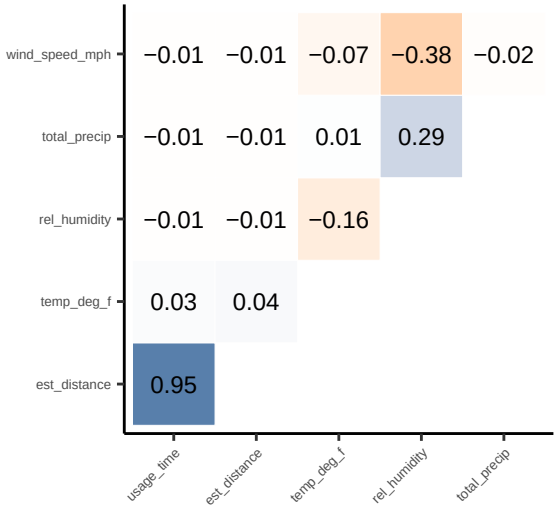
Density Plots of Variables



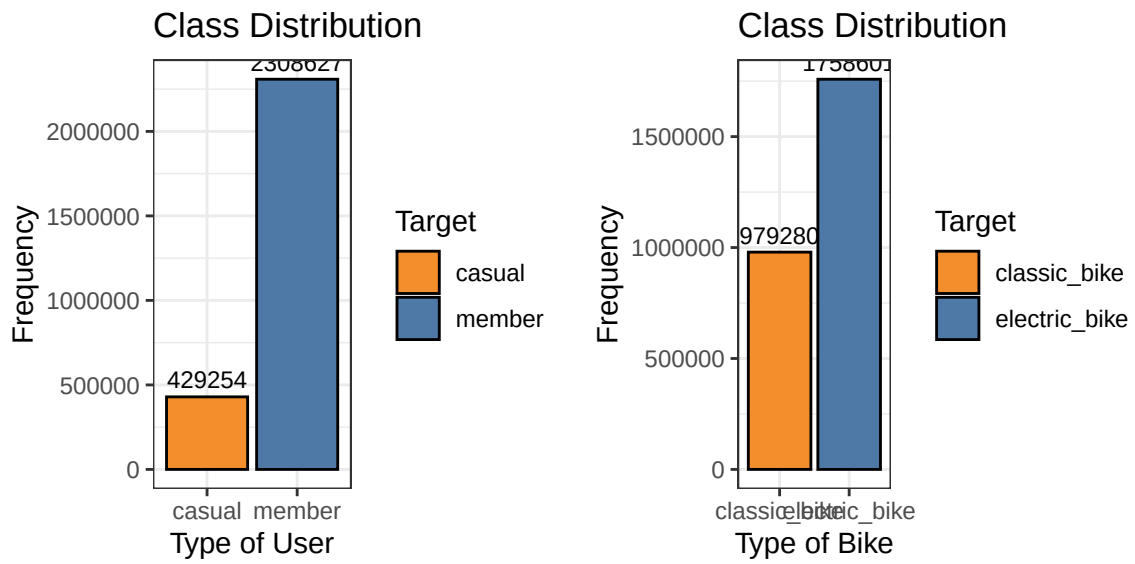
Boxplots of Variables



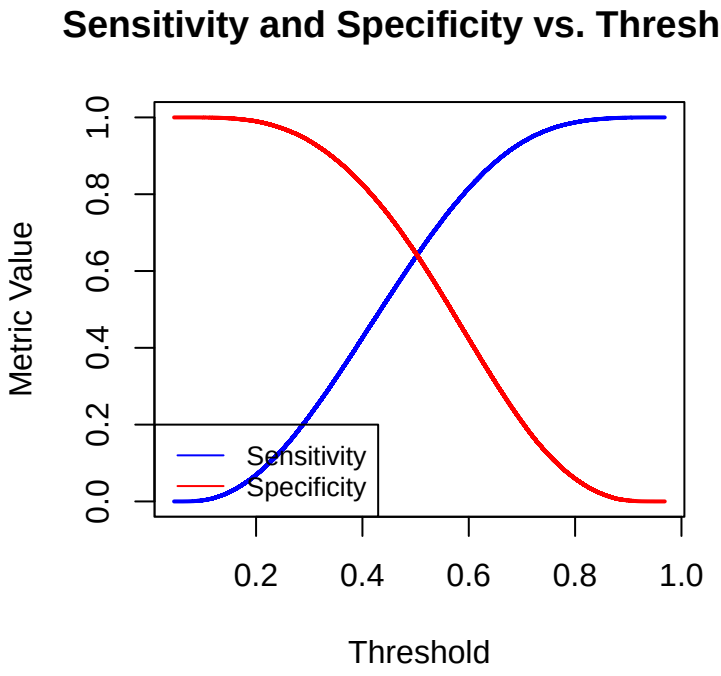
Correlation Matrix



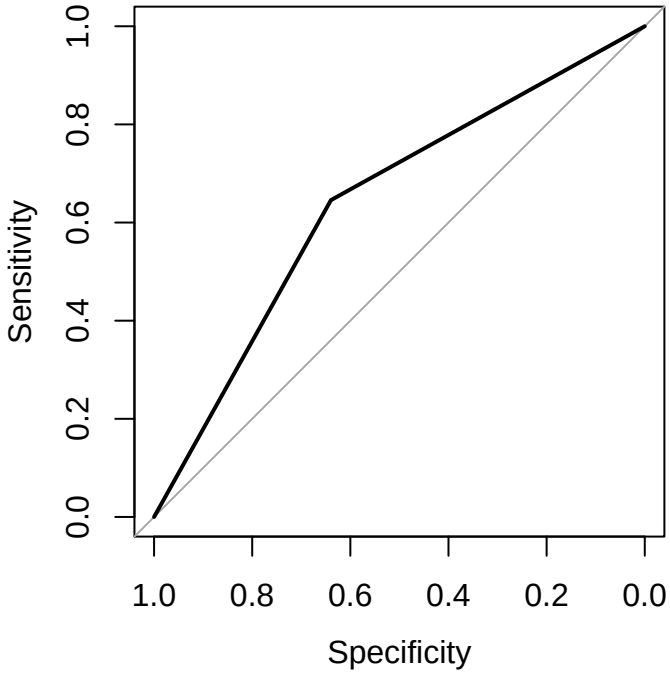
Barplots to Assess Class Imbalance in the Target Variable and Bike Type Variable



Sensitivity versus Specificity for Logistic Regression Model

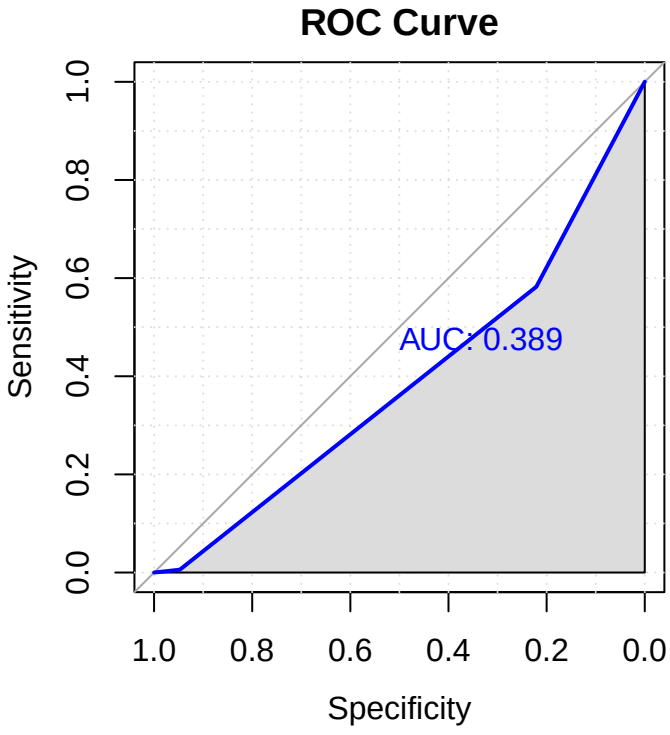


ROC Curve for Logistic Regression Model

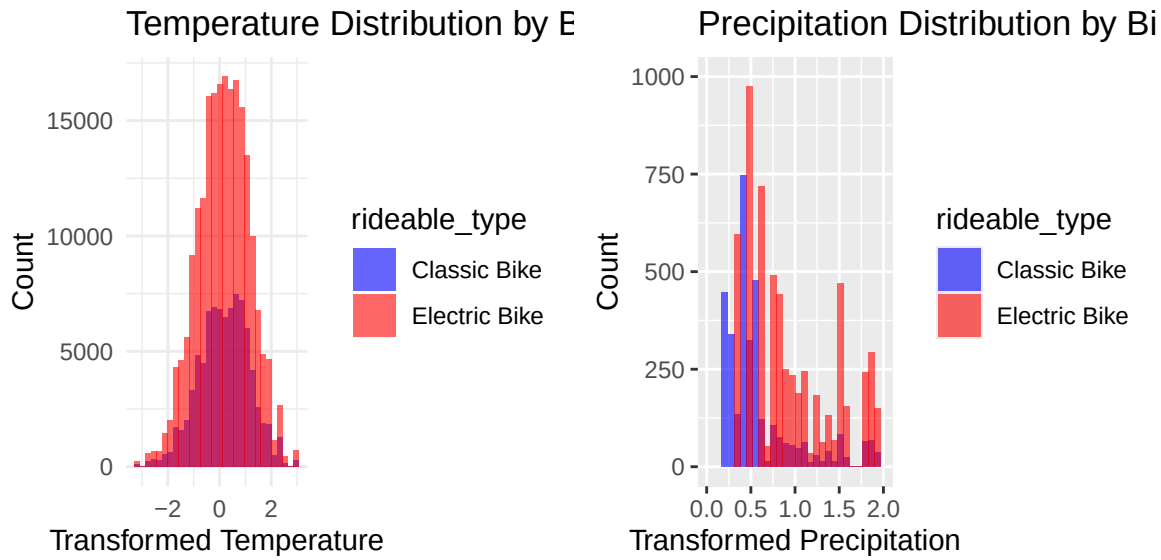



```
## Area under the curve: 0.6426
```

ROC Curve for Boosted Logistic Regression Model



Distributions of Temperature and Precipitation by Bike Type




```
## List of 136
##  $ line                            :List of 6
##   ..$ colour       : chr "black"
##   ..$ linewidth    : num 0.5
##   ..$ linetype     : num 1
##   ..$ lineend      : chr "butt"
##   ..$ arrow        : logi FALSE
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_line" "element"
##  $ rect                            :List of 5
##   ..$ fill         : chr "white"
##   ..$ colour       : chr "black"
##   ..$ linewidth    : num 0.5
##   ..$ linetype     : num 1
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_rect" "element"
##  $ text                            :List of 11
##   ..$ family       : chr ""
##   ..$ face         : chr "plain"
##   ..$ colour       : chr "black"
##   ..$ size         : num 11
##   ..$ hjust        : num 0.5
##   ..$ vjust        : num 0.5
##   ..$ angle        : num 0
##   ..$ lineheight   : num 0.9
##   ..$ margin       : 'margin' num [1:4] 0points 0points 0points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : logi FALSE
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ title                           : NULL
##  $ aspect.ratio                    : NULL
##  $ axis.title                      : NULL
```

```
##  $ axis.title.x                   :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 1
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 2.75points 0points 0points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.title.x.top               :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 0
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 0points 2.75points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.title.x.bottom            : NULL
##  $ axis.title.y                   :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 1
##   ..$ angle        : num 90
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 2.75points 0points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.title.y.left              : NULL
##  $ axis.title.y.right             :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 1
##   ..$ angle        : num -90
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 0points 0points 2.75points
```

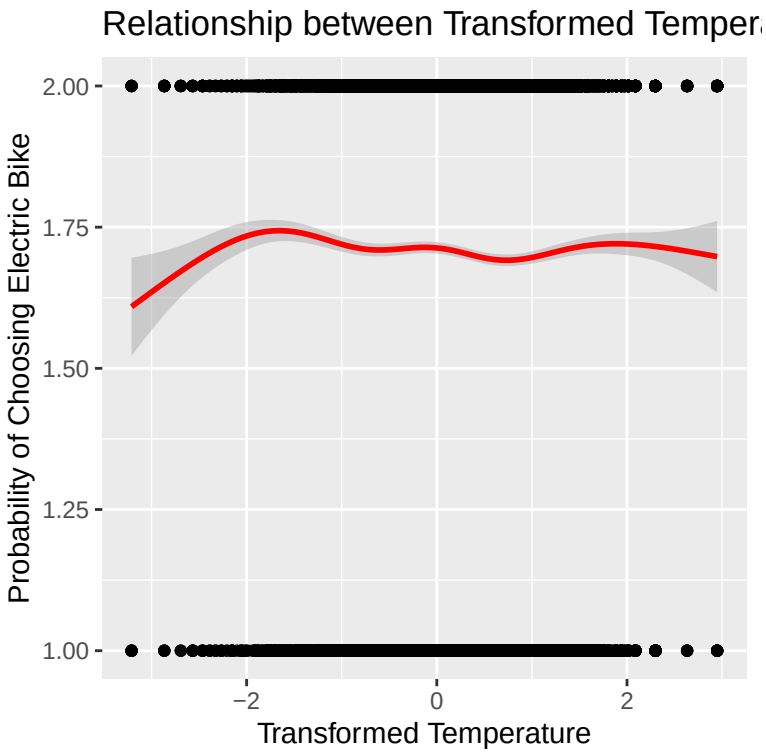
```
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text                      :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : chr "grey30"
##   ..$ size         : 'rel' num 0.8
##   ..$ hjust        : NULL
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text.x                    :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 1
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 2.2points 0points 0points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text.x.top                :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 0
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 0points 2.2points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text.x.bottom             : NULL
##  $ axis.text.y                    :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 1
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
```

```
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 2.2points 0points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text.y.left               : NULL
##  $ axis.text.y.right              :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 0
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 0points 0points 2.2points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text.theta                : NULL
##  $ axis.text.r                    :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 0.5
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 2.2points 0points 2.2points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.ticks                     : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ axis.ticks.x                   : NULL
##  $ axis.ticks.x.top               : NULL
##  $ axis.ticks.x.bottom            : NULL
##  $ axis.ticks.y                   : NULL
##  $ axis.ticks.y.left              : NULL
##  $ axis.ticks.y.right             : NULL
##  $ axis.ticks.theta               : NULL
##  $ axis.ticks.r                   : NULL
##  $ axis.minor.ticks.x.top         : NULL
##  $ axis.minor.ticks.x.bottom      : NULL
##  $ axis.minor.ticks.y.left        : NULL
##  $ axis.minor.ticks.y.right       : NULL
##  $ axis.minor.ticks.theta         : NULL
##  $ axis.minor.ticks.r             : NULL
##  $ axis.ticks.length              : 'simpleUnit' num 2.75points
##   ..- attr(*, "unit")= int 8
```

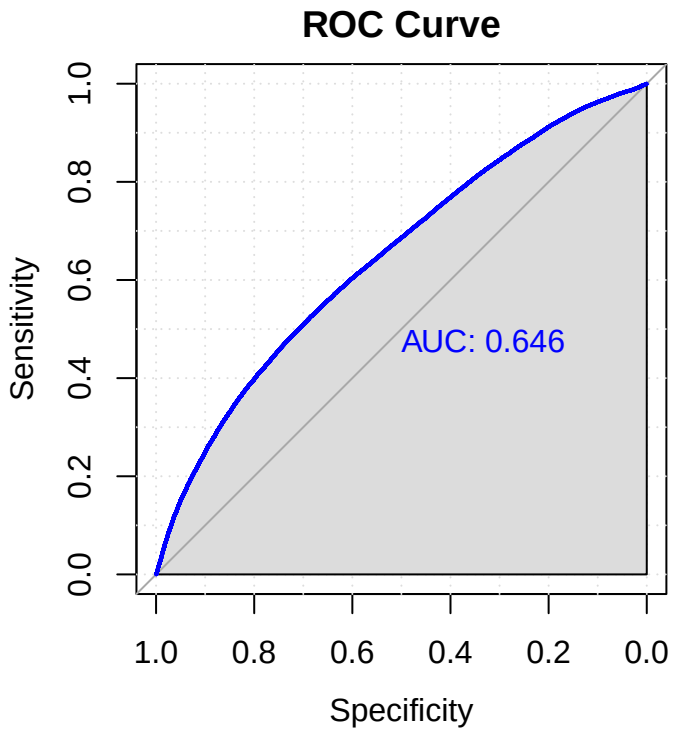
```
##  $ axis.ticks.length.x             : NULL
##  $ axis.ticks.length.x.top         : NULL
##  $ axis.ticks.length.x.bottom      : NULL
##  $ axis.ticks.length.y             : NULL
##  $ axis.ticks.length.y.left        : NULL
##  $ axis.ticks.length.y.right       : NULL
##  $ axis.ticks.length.theta         : NULL
##  $ axis.ticks.length.r             : NULL
##  $ axis.minor.ticks.length         : 'rel' num 0.75
##  $ axis.minor.ticks.length.x       : NULL
##  $ axis.minor.ticks.length.x.top   : NULL
##  $ axis.minor.ticks.length.x.bottom: NULL
##  $ axis.minor.ticks.length.y       : NULL
##  $ axis.minor.ticks.length.y.left  : NULL
##  $ axis.minor.ticks.length.y.right : NULL
##  $ axis.minor.ticks.length.theta   : NULL
##  $ axis.minor.ticks.length.r       : NULL
##  $ axis.line                       : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ axis.line.x                     : NULL
##  $ axis.line.x.top                 : NULL
##  $ axis.line.x.bottom              : NULL
##  $ axis.line.y                     : NULL
##  $ axis.line.y.left                : NULL
##  $ axis.line.y.right               : NULL
##  $ axis.line.theta                 : NULL
##  $ axis.line.r                     : NULL
##  $ legend.background               : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ legend.margin                   : 'margin' num [1:4] 5.5points 5.5points 5.5points 5.5points
##   ..- attr(*, "unit")= int 8
##  $ legend.spacing                  : 'simpleUnit' num 11points
##   ..- attr(*, "unit")= int 8
##  $ legend.spacing.x                : NULL
##  $ legend.spacing.y                : NULL
##  $ legend.key                      : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ legend.key.size                 : 'simpleUnit' num 1.2lines
##   ..- attr(*, "unit")= int 3
##  $ legend.key.height               : NULL
##  $ legend.key.width                : NULL
##  $ legend.key.spacing              : 'simpleUnit' num 5.5points
##   ..- attr(*, "unit")= int 8
##  $ legend.key.spacing.x            : NULL
##  $ legend.key.spacing.y            : NULL
##  $ legend.frame                    : NULL
##  $ legend.ticks                    : NULL
##  $ legend.ticks.length             : 'rel' num 0.2
##  $ legend.axis.line                : NULL
##  $ legend.text                     :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : 'rel' num 0.8
```

```
##   ..$ hjust        : NULL
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ legend.text.position            : NULL
##  $ legend.title                    :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 0
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ legend.title.position           : NULL
##  $ legend.position                 : chr "right"
##  $ legend.position.inside          : NULL
##  $ legend.direction                : NULL
##  $ legend.byrow                    : NULL
##  $ legend.justification            : chr "center"
##  $ legend.justification.top        : NULL
##  $ legend.justification.bottom     : NULL
##  $ legend.justification.left       : NULL
##  $ legend.justification.right      : NULL
##  $ legend.justification.inside     : NULL
##  $ legend.location                 : NULL
##  $ legend.box                      : NULL
##  $ legend.box.just                 : NULL
##  $ legend.box.margin               : 'margin' num [1:4] 0cm 0cm 0cm 0cm
##   ..- attr(*, "unit")= int 1
##  $ legend.box.background           : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ legend.box.spacing              : 'simpleUnit' num 11points
##   ..- attr(*, "unit")= int 8
##   [list output truncated]
##  - attr(*, "class")= chr [1:2] "theme" "gg"
##  - attr(*, "complete")= logi TRUE
##  - attr(*, "validate")= logi TRUE
```

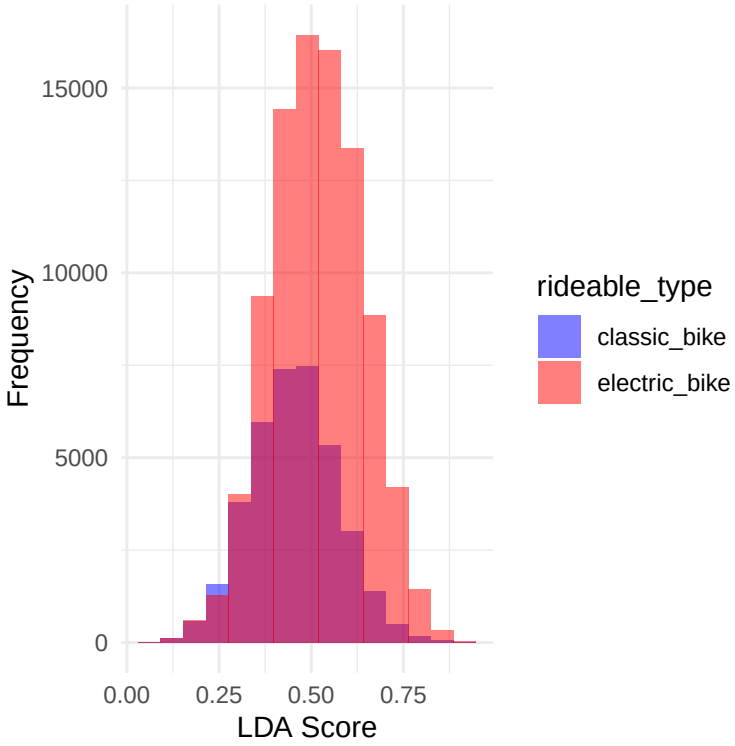
Plot of Probability of Choosing an Electric Bike by Temperature



LDA ROC Curve



Visualization of LDA scores



Appendix 2: R Code

Data Import

```
# Convert to parquet

# library(dplyr)
# library(readr)
#
# df <- list.files(path = "data/", full.names = TRUE, pattern = "\\.csv$") %>%
#   lapply(read_csv) %>%
#   lapply(\(x) mutate(x, across(end_station_id, as.character))) %>%
#   bind_rows
#
# write_parquet(df, "citi_bike_03_2024.parquet")
trips <- read_parquet(here("Final_Project", "Data", "citi_bike_03_2024.parquet"))
head(trips)
```

```
kbl(head(trips)) |>
  kable_classic(full_width = F, html_font = "Cambria") |>
  footnote(general_title = "Dimensions: ",
           TeX(paste0(nrow(trips), " x ", ncol(trips)))) %>%
  kable_styling(latex_options = "HOLD_position")
```

Data Exploration

Citi Bike Missing Values

```
trips <-
  trips |>
  dplyr::select(!ride_id)

missing_data <-
  trips %>%
  summarise(across(everything(), ~ sum(is.na(.x))))

long_missing_data <- pivot_longer(missing_data, cols = everything(),
   names_to = "Field Name", values_to = "Missing Count")

kbl(long_missing_data) |>
    kable_classic(full_width = F, html_font = "Cambria") %>%
  kable_styling(latex_options = "HOLD_position")
```

Developing Predictor Variables

```
trips <- trips |>
  mutate(usage_time = time_length(ended_at - started_at, "seconds"),
         est_distance = usage_time * case_when(
          rideable_type == "classic_bike" ~ 0.00230556,  ## converted to miles per second
            rideable_type == "electric_bike" ~ 0.00416667)) ## converted to miles per second

kbl(head(trips)) |>
  kable_classic(full_width = F, html_font = "Cambria") |>
  footnote(general_title = "Dimensions: ",
           TeX(paste0(nrow(trips), " x ", ncol(trips)))) %>%
  kable_styling(latex_options = "HOLD_position")
```

Weather Predictor Variables

```
weather <- read_csv(here("Final_Project", "Data", 'weather.csv')) |>
  janitor::clean_names()

weather <-
  weather |>
  mutate(temp_deg_f = celsius.to.fahrenheit(temperature_deg_c),
         rel_humidity = dewpoint.to.humidity(t = temperature_deg_c,
                                             dp = dewpoint_temperature_deg_c,
                                             temperature.metric = "celsius"),
         total_precip = total_precipitation_mm_of_water_equivalent / 25.4,
         wind_speed_mph = convert_wind_speed(wind_speed_m_s,
          old_metric="mps", new_metric="mph", round=2))

weather <-
```

```
  weather |>
  mutate(day_of_week = wday(datetime_utc, label = TRUE, week_start = 1, abbr = FALSE),
         day_of_week = as.factor(day_of_week),
         datetime_ny = with_tz(datetime_utc, "America/New_York")) |>
  relocate(datetime_ny)
```

Weather Missing Values

```
missing_data <-
  weather %>%
  summarise(across(everything(), ~ sum(is.na(.x))))

long_missing_data <- pivot_longer(missing_data, cols = everything(),
   names_to = "Field Name", values_to = "Missing Count")

kbl(long_missing_data) |>
  kable_classic(full_width = F, html_font = "Cambria") |>
  kable_styling(latex_options = "HOLD_position")
```

Merge Dataframes

```
raw_trips_weather <-
  trips |>
  mutate(datetime_ny = floor_date(started_at, "hour")) |>
  left_join(weather, by=join_by(datetime_ny))

write_parquet(raw_trips_weather, "raw_trips_weather.parquet")
```

Summary Statistics

```
num_var <- c("usage_time", "est_distance", "temp_deg_f",
             "rel_humidity", "total_precip", "wind_speed_mph")

num_raw_trips_weather <-
  raw_trips_weather |>
    dplyr::select(num_var)

summary <-
  round(descr(num_raw_trips_weather), 2)
kbl(summary, booktabs = TRUE) |>
  kable_classic(full_width = FALSE, html_font = "Cambria") |>
  kable_styling(latex_options = "HOLD_position") |>
  kableExtra::landscape()
```

Density

```
num_raw_trips_weather |>
  gather(key = "variable", value = "value") |>
  ggplot(aes(x = value)) +
  geom_histogram(aes(y = after_stat(density)), bins = 20, fill = '#4E79A7', color = 'black') +
  stat_density(geom = "line", color = "red") +
  facet_wrap(~ variable, scales = 'free') +
  theme(strip.text = element_text(size = 5))
```

Boxplot

```
num_raw_trips_weather |>
  gather(key = "Variable", value = "Value") |>
  ggplot(aes(x = "", y = Value)) +
  geom_boxplot(fill = "#4E79A7") +
  facet_wrap(~ Variable, scales = "free") +
  labs(x = NULL, y = "Value") +
  theme(strip.text = element_text(size = 5))
```

Correlation Matrix

```
q <- cor(num_raw_trips_weather)

ggcorrplot(q, type = "upper", outline.color = "white",
           ggtheme = theme_classic,
           colors = c("#F28E2B", "white", "#4E79A7"),
           lab = TRUE, show.legend = F, tl.cex = 5, lab_size = 3)
```

Class Imbalance

```
class_freq <- raw_trips_weather |>
  count(member_casual)

ggplot(raw_trips_weather, aes(x = member_casual, fill = as.factor(member_casual))) +
  geom_bar(color = "black") +
  geom_text(data = class_freq, aes(label = n, y = n), vjust = -0.5, size = 3, color = "black") +
  scale_fill_manual(values = c("#F28E2B", "#4E79A7")) +  # Customize fill colors
  labs(title = "Class Distribution",
       x = "Type of User",
       y = "Frequency",
       fill = "Target") +
  theme_bw()
```

```
class_freq <- raw_trips_weather |>
  count(rideable_type)
```

```
ggplot(raw_trips_weather, aes(x = rideable_type, fill = as.factor(rideable_type))) +
  geom_bar(color = "black") +
  geom_text(data = class_freq, aes(label = n, y = n), vjust = -0.5, size = 3, color = "black") +
  scale_fill_manual(values = c("#F28E2B", "#4E79A7")) +  # Customize fill colors
  labs(title = "Class Distribution",
       x = "Type of Bike",
       y = "Frequency",
       fill = "Target") +
  theme_bw()
```

Data Processing

Drop Missing Values

```
cleaned_raw_trips_weather <- na.omit(raw_trips_weather)
```

Splitting the Dataset

```
set.seed(1125)
trainIndex <- createDataPartition(y = cleaned_raw_trips_weather$member_casual,
                  p = 0.7, list = FALSE, times = 1)
train_data <- cleaned_raw_trips_weather[trainIndex, ]
test_data <- cleaned_raw_trips_weather[-trainIndex, ]
```

Model Building - Part #1

Binary Logistic Regression

```
train_data_transformed <- read_parquet(here("Final_Project", "Data", "train_data_transformed.parquet"))
test_data_transformed <- read_parquet(here("Final_Project", "Data", "test_data_transformed.parquet"))
names(train_data_transformed)
```

```
#considering splining
ggplot(train_data_transformed, aes(x = temp_deg_f_transformed, y = member_casual)) +
  geom_hex(bins = 30) +
  ggtitle("Hex Bin Plot of Predictor vs. Target") +
  xlab("Predictor") +
  ylab("Target") +
  scale_fill_gradient(low = "white", high = "blue")
```

```
ggplot(train_data_transformed, aes(x = total_precip_transformed, y = member_casual)) +
  geom_hex(bins = 30) +
  ggtitle("Hex Bin Plot of Predictor vs. Target") +
  xlab("Predictor") +
  ylab("Target") +
  scale_fill_gradient(low = "white", high = "blue")
```

Ensure Data types

```
train_data_transformed$rideable_type <- as.factor(train_data_transformed$rideable_type)
test_data_transformed$rideable_type <- as.factor(test_data_transformed$rideable_type)
train_data_transformed$member_casual <- as.factor(train_data_transformed$member_casual)
test_data_transformed$member_casual <- as.factor(test_data_transformed$member_casual)
train_data_transformed$day_of_week <- factor(train_data_transformed$day_of_week, ordered = FALSE)
test_data_transformed$day_of_week <- factor(test_data_transformed$day_of_week, ordered = FALSE)


train_data_transformed$hour <- hour(train_data_transformed$datetime_ny)
test_data_transformed$hour <- hour(test_data_transformed$datetime_ny)
```

Add Time of day

```
categorize_time_of_day <- function(hour) {
  if (hour >= 4 && hour < 8) {
    "Early Morning"
  } else if (hour >= 8 && hour < 12) {
    "Morning"
  } else if (hour >= 12 && hour < 16) {
    "Afternoon"
  } else if (hour >= 16 && hour < 20) {
    "Evening"
  } else {
    "Night"
  }
}

train_data_transformed$time_of_day <- sapply(train_data_transformed$hour, categorize_time_of_day)
train_data_transformed$time_of_day <- as.factor(train_data_transformed$time_of_day)

test_data_transformed$time_of_day <- sapply(test_data_transformed$hour, categorize_time_of_day)
test_data_transformed$time_of_day <- as.factor(test_data_transformed$time_of_day)
```

Imbalanced Data

```
print(table(train_data_transformed$member_casual))
```

Undersample

```
set.seed(7)
data_balanced <- ovun.sample(member_casual ~ ., data = train_data_transformed, method = "under",
                             N = 2 * table(train_data_transformed$member_casual)["casual"],
                             seed = 7)$data

print(table(data_balanced$member_casual))
```

Logistic Regression

```
set.seed(7)
train_control <- trainControl(
  method = "cv",
  number = 5,
  savePredictions = "final",
  classProbs = TRUE,  # store probabilities for calculations later on
  summaryFunction = twoClassSummary
)

simple_model <- train(
  member_casual ~ rideable_type + temp_deg_f_transformed +
    rel_humidity_transformed + total_precip_transformed +
    wind_speed_transformed + day_of_week + usage_time_transformed +
    est_distance_transformed + time_of_day,
  data = data_balanced,
  method = "glm",
  family = binomial(),
  trControl = train_control,
  metric = "ROC"
)

print(simple_model)
```

Summary Model 1

```
summary(simple_model)
```

Multicollinearity

```
extracted_simple_model = simple_model$finalModel
vif(extracted_simple_model)
```

Remove Distance

```
simple_model2 <- train(
  member_casual ~ rideable_type + temp_deg_f_transformed +
    rel_humidity_transformed + total_precip_transformed +
    wind_speed_transformed + day_of_week + usage_time_transformed +
    time_of_day,
  data = data_balanced,
  method = "glm",
  family = binomial(),
  trControl = train_control,
  metric = "ROC"
)
```

```r
print(simple_model2)
```

Summary Model 2

```r
summary(extracted_simple_model2)
```

Evaluating Model

```r
predictions <- predict(simple_model2, data_balanced, type = "prob")[, "member"]
roc_obj <- roc(response = data_balanced$member_casual, predictor = predictions)

plot(roc_obj, main = "ROC Curve", col = "blue")

sens_spec <- coords(roc_obj, x = "all",
           ret = c("threshold", "sensitivity", "specificity"), transpose = FALSE)
plot(sens_spec$threshold, sens_spec$sensitivity, type = "l", col = "blue", lwd = 2,
       xlab = "Threshold", ylab = "Metric Value", ylim = c(0, 1),
        main = "Sensitivity and Specificity vs. Threshold")
lines(sens_spec$threshold, sens_spec$specificity, type = "l", col = "red", lwd = 2)
legend("bottomleft", legend = c("Sensitivity", "Specificity"),
         col = c("blue", "red"), lty = 1, cex = 0.8)
```

```r
predicted_probs <- predict(simple_model2, newdata = test_data_transformed, type = "prob")

threshold = 0.5 # we can change this
predicted_classes <- ifelse(predicted_probs[, "member"] > threshold, "member", "casual")

# just to make sure they are factors with same levels
predicted_classes <- factor(predicted_classes, levels = c("casual", "member"))
test_data_transformed$member_casual <- factor(test_data_transformed$member_casual,
     levels = c("casual", "member"))

confusionMatrix(data = predicted_classes, reference = test_data_transformed$member_casual)

roc_result <- roc(response = test_data_transformed$member_casual,
     predictor = as.numeric(predicted_classes))
plot(roc_result)
auc(roc_result)
```

Boosted Logistic Regression

```r
# The first model was trained on these columns.
train_sample_m1 <- train_sample %>% dplyr::select(
    rideable_type,start_station_name,end_station_name,start_lat,start_lng,
    end_lat,end_lng,member_casual,dewpoint_temperature_deg_c,total_cloud_cover_0_1,
    rel_humidity,day_of_week,usage_time_transformed,temp_deg_f_transformed,
     wind_speed_transformed,hour,time_of_day)
```

```r
test_sample_m1 <- test_sample %>% dplyr::select(rideable_type,start_station_name,
    end_station_name,start_lat,start_lng,end_lat,end_lng,member_casual,
    dewpoint_temperature_deg_c,total_cloud_cover_0_1,rel_humidity,day_of_week,
    usage_time_transformed,temp_deg_f_transformed, wind_speed_transformed,hour,time_of_day)

test_sample_m1 <- test_sample_m1 %>%
   filter(!(start_station_name == 'Baldwin at Montgomery' |
    start_station_name == 'Brooklyn Ave & Tilden Ave' |
     end_station_name == 'Clinton St & Newark St' | end_station_name == 'Dey St'))
      #removing these four observations since the train dataset did not contain
      # these levels and when predicting on these records, the model fails
```

```r
# DO NOT RUN THE BELOW CODE --- it takes 5 - 6 hours to train this model

preprocess_steps <- c("nzv", "corr")
preprocess_obj <- preProcess(train_sample_m1[, -which(names(train_sample_m1)
   == "member_casual")], method = preprocess_steps)
train_preprocessed <- predict(preprocess_obj, train_sample_m1)
test_preprocessed <- predict(preprocess_obj, test_sample_m1)
#
# model_weights <- train(
#   member_casual ~ .,
#   data = train_preprocessed,
#   method = "LogitBoost",
#   trControl = trainControl(method = "cv", number = 5, classProbs = TRUE),
#   weights = ifelse(train_preprocessed$member_casual == "casual", .9, .1),
#   importance = TRUE
# )

model_weights <- readRDS(here("Final_Project", "Data", "LogitBoost_weights.rds"))
summary(model_weights)
predictions <- predict(model_weights, newdata = test_preprocessed)
confusionMatrix(predictions, as.factor(test_preprocessed$member_casual))

#saveRDS(model_spec, "LogitBoost_weights.rds")
#model_weights <- readRDS("LogitBoost_weights.rds")
```

Evaluate Model

```r
# Make predictions on the testing dataset
predictions <- predict(model_weights, newdata = test_sample_m1, type = "prob")

# Extract the predicted probabilities for the positive class
predicted_probs <- predictions[, "casual"]  # Assuming "casual" is the positive class

# Create the ROC curve
roc_obj <- roc(test_sample_m1$member_casual, predicted_probs)

# Plot the ROC curve
plot(roc_obj, main = "ROC Curve", print.auc = TRUE,
   auc.polygon = TRUE, grid = TRUE, col = "blue", lwd = 2)
```

Model Building - Part #2

Filter Data

```
casual_train_data <- train_data_transformed %>%
  filter(member_casual == "casual")

casual_test_data <- test_data_transformed %>%
  filter(member_casual == "casual")
```

Temperature Distribution

```
ggplot(casual_train_data, aes(x = temp_deg_f_transformed, fill = rideable_type)) +
  geom_histogram(position = "identity", alpha = 0.6, bins = 30) +
  labs(x = "Transformed Temperature", y = "Count", title = "Temperature Distribution by Bike Type") +
  scale_fill_manual(values = c("blue", "red"), labels = c("Classic Bike", "Electric Bike")) +
  theme_minimal()
```

Precipitation Distribution

```
casual_train_data <- casual_train_data %>%
  mutate(total_precip_transformed =
   log1p(total_precip_transformed - min(total_precip_transformed) + 0.01))

min_precip_train <- min(casual_train_data$total_precip_transformed, na.rm = TRUE)
casual_test_data <- casual_test_data %>%
  mutate(total_precip_transformed = log1p(total_precip_transformed - min_precip_train + 0.01))
```

```
ggplot(casual_train_data, aes(x = total_precip_transformed, fill = rideable_type)) +
  geom_histogram(position = "identity", alpha = 0.6, bins = 30) +
  labs(x = "Transformed Precipitation", y = "Count", title = "Precipitation Distribution by Bike Type")
  scale_fill_manual(values = c("blue", "red"), labels = c("Classic Bike", "Electric Bike")) +
  xlim(0, 2) +
  ylim(0, 1000)
  theme_minimal()
```

```
summary_stats <- casual_train_data %>%
  group_by(rideable_type) %>%
  summarise(
    Count = n(),
    Mean_Temp = mean(temp_deg_f_transformed, na.rm = TRUE),
    Median_Temp = median(temp_deg_f_transformed, na.rm = TRUE),
    SD_Temp = sd(temp_deg_f_transformed, na.rm = TRUE),
    Mean_Precip = mean(total_precip_transformed, na.rm = TRUE),
    Median_Precip = median(total_precip_transformed, na.rm = TRUE),
    SD_Precip = sd(total_precip_transformed, na.rm = TRUE)
  )

print(summary_stats)
```

Electric Bike/Temp Relationship

```
# Sample 10% of the data
sampled_data <- casual_train_data %>%
  dplyr::sample_frac(size = 0.1)

# Now plot using the sampled data
ggplot(sampled_data, aes(x = temp_deg_f_transformed, y = as.numeric(rideable_type))) +
    geom_point(alpha = 0.3) +
    geom_smooth(method = "gam", formula = y ~ s(x), color = "red") +
    labs(title = "Relationship between Transformed Temperature and Rideable Type",
         x = "Transformed Temperature", y = "Probability of Choosing Electric Bike")
```

Generalized Additive Model

```
casual_balanced <- ovun.sample(rideable_type ~ ., data = casual_train_data, method = "under",
                               N = 2 * 87260,
                               seed = 7)$data
```

```
gam1 <- gam(rideable_type ~ s(temp_deg_f_transformed, bs = "cs") +
                                    rel_humidity_transformed + total_precip_transformed +
                                    wind_speed_transformed + day_of_week +
                                    est_distance_transformed + time_of_day,
                    family = binomial(), data = casual_balanced)

summary(gam1)
```

Linear Discriminant Analysis

```
train_control <- trainControl(
  method = "cv",
  number = 10,
  savePredictions = "final",
  classProbs = TRUE
)
```

```
lda_model <- train(
  rideable_type ~ est_distance_transformed + time_of_day
   + temp_deg_f_transformed + total_precip_transformed,
  data = casual_balanced,
  method = "lda",
  trControl = train_control
)
```

Summary LDA Model

```r
print(lda_model)
summary(lda_model)
```

```r
final_lda_model <- lda_model$finalModel

print(final_lda_model)
summary(final_lda_model)
```

Evaluate LDA Model

```r
# Make predictions on the testing dataset
lda_scores <- predict(lda_model, casual_test_data, type = "prob")

# Extract the predicted probabilities for the positive class
predicted_probs <- lda_scores$electric_bike  # Assuming "electric_bike" is the positive class

# Create the ROC curve
roc_obj <- roc(casual_test_data$rideable_type, predicted_probs)

# Plot the ROC curve
plot(roc_obj, main = "ROC Curve", print.auc = TRUE,
   auc.polygon = TRUE, grid = TRUE, col = "blue", lwd = 2)
```

```r
# calculate LDA scores
lda_scores <- predict(lda_model, casual_test_data, type = "prob")

casual_test_data$score <- lda_scores$electric_bike

ggplot(casual_test_data, aes(x = score, fill = rideable_type)) +
  geom_histogram(position = "identity", alpha = 0.5, bins = 15) +
  labs(x = "LDA Score", y = "Frequency") +
  scale_fill_manual(values = c("blue", "red")) +
  theme_minimal()
```