Andrew Singh

Machine Learning I - Project 1

The Business Problem and Context

The Analytics Inc. is pleased to produce a report that will be used to summarize key findings and insights to the senior leadership team at Capital Bike Share informing their ability to predict bikeshare demand. For our analysis, we were supplied an internal dataset representing bikeshare demand in Washington D.C. between 2011 through 2012. We want to understand how bike share usage could be predicted based on the following independent variables: date in time (year, month, hour), whether it is a holiday, weekday, or weekend, and various weather conditions. We focused our prediction models on casual users due to the opportunity to increase the proportion of casual bikeshare usership.

Exploratory Data Analysis

Data-Cleaning

Prior to our data analysis of the Capital Bikeshare dataset, we performed data processing to re-code numerous variables that R thought to be integer datatypes to become categorical or factor variables. If we did not re-code the variables, our analysis would not produce reliable results as intended to find the true drivers of bikeshare usage in Washington D.C. We re-labeled numerical values into text-based values for several predictor variables to make interpreting and comprehending data visualizations easier with appropriate labels

Correlations and Multicollinearity

Based on our analysis, we found that there are strong correlated variables in the Capital Bike Share dataset, resulting in multi-collinearity, which may impact and may skew our correlation estimates. The variables that we intend to remove from our analytical model to reduce the likelihood of multi-collinearity is ‘atemp’ as the correlation coefficient between ‘atemp’ and ‘temp’ is 0.98. Additionally, we removed ‘registered’ and ‘casual’, as these

variables are subsets of the 'cnt' variable, similarly reproducing a multi-collinear effect on the correlation estimates.

Variables on Time

Our analysis shows there are seasonal differences in bikeshare usage, with demand spiking in Spring and Summer for causal users as shown in Appendix 1.6). When we classify for user type (registered and casual users) to find deeper insights into the overall population per time variable, we see that majority of trends remain consistent, except for casual users utilize bikeshares mostly on the weekends. Generally, weather is a factor that directly impacts usage for either user type, but casual users, usage is non-existent.

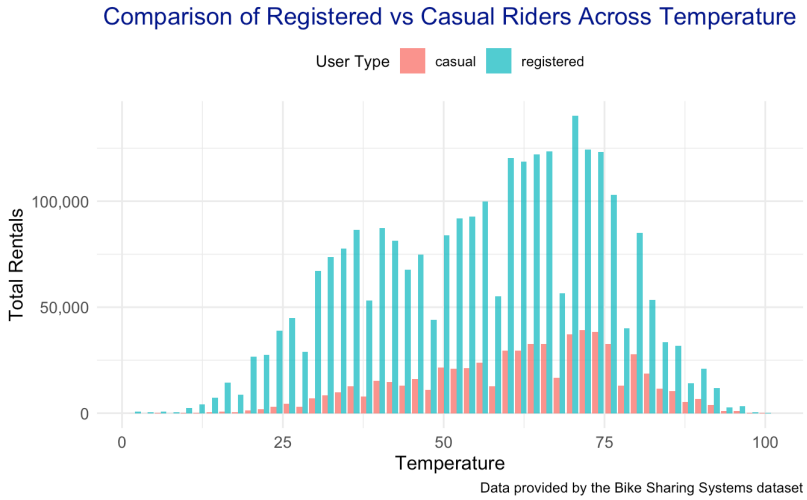
Findings from Exploratory Data Analysis

We believe that the key drivers for Capital Bike Share usage lies in understanding usership trends in terms of hours of usage, temperature, and weather, specifically we will focus on the general trends in registered versus casual users, and create our recommendations to enhance casual bikeshare usership.

1) Usage across time: Registered users leverage bikeshares to commute during rush hour on workdays users, while casual users may seek to use bikeshares for leisure or short errands (as shown in Appendix 1.2). This trend holds true for other cities implementing bikeshares, demonstrating demand spikes around the earlier and evening parts of the day for registered users as they may be commuting to work between 8am through 9am or their home between 5pm through 6pm, and remains consistent for casual users in the afternoon to early evening between 12pm through 6pm, according to the National Association of City Transportation Officials.[1]

[1] "Bike Share in the U.S.: 2017 | National Association of City Transportation Officials," National Association of City Transportation Officials, n.d., https://nacto.org/bike-share-statistics-2017/.

2) Temperature influences bikeshare usage: From the chart below, there is a positive relationship between bikeshare usage and temperature; as temperature increases, bikeshare directly increases.



3) Bikeshare usage is predominately utilized when the weather is clear and partially cloudy unlike wet weather conditions like snow or heavy rain, reducing bikeshare usage, due to wet conditions resulting in riding a bike less safe or preferable (as shown in Appendix 1.3).

Regression Model

To further predict the demand for bikeshares in Washington D.C., we created a machine learning model focused determining what are the driving factors for bikeshare usage and we refined our models removing unnecessary factors of time or similar factors to one another that did not add value. Our variable of interest is 'casual', as we are observing which factors are most likely to increase casual bikeshare user representation.

Within our initial model, we utilized all available factors in our dataset, except for registered and casual as those factors are subsets of the overall count of bikeshares. Afterwards, to improve the predictive ability of our model, we included polynomial terms on factors that held a moderately strong correlation between casual bikeshare users such as temperature ($r = 0.46$), or a somewhat apparent correlation such windspeed ($r = 0.09$). We found each polynomial term we included was useful due to the p-value being less than 5% indicating a

statistically significant relationship between temperature and windspeed with casual bikeshare users.

The following variables were utilized in our predictive model: ‘registered’, ‘dteday’, ‘yr’, ‘mnth’, ‘weekday’, ‘temp’, ‘hum’, ‘holidayrecode’, ‘workingdayrecode’, ‘weatherrecode’, and ‘seasonrecode’. The variables that were removed from the original model due to similarities to other variables in the model construing the true findings, the removed variables are ‘cnt’ and ‘atemp’. As we continued to refine the iterative model, we removed the polynomial term ‘windspeed ^2’ and ‘humidity’ which resulted in less predictability in casual bikeshare users.

Based on our best model, we found that our R-squared and adjusted R-squared values predicted 42.27% of the variation of ‘casual users which are explained by the variety of independent variables. When compared to our test dataset, our R-squared and adjusted R-squared decreased to 40.52%, which indicates our model still holds credibility.

Interpretation of Result

We found that there are opportunities to capitalize on the increases in temperature, whereby for every degree increase in temperature, there is an increase of approximately 95 casual bikeshare users. Similarly, in our model, we found that when we compare casual bikeshare usage from Sunday to other days of the week, Wednesday proves to be the best day of the week for casual bikeshare users, as we expect there to be an increase in approximately 97 bikeshare users than other days.

Based on our prediction model, we found that the weekday, humidity, weather, season, and holiday were mostly likely to contribute to our model fit. Each of the variables in our model were highly statistically significant with specific days and months being truly related with casual bikeshare usage.

The potential business implications are that during any of the weekdays, months of June and July, and when the temperature is increasing or the weather is clear with a few clouds

or a light mist, bikeshare usage will be higher in demand. This means that casual users of Capital Bike Share are more likely to rent bikes during these periods of time. Additionally, this presents an opportunity to create innovative strategies to focus strategically on converting casual bikeshare users into registered users. Capital Bike Share should focus on introducing new bikeshare user programs focused on enticing casual users to become a member, introducing newer version of rental bikes which may likely impact casual users to be more likely to rent more bikes when the weather becomes better or for weekend errands, and increase the amount of bikes available during the weekends and in the Spring and Summer.

Limitations and Assumptions

In our linear regression model, the regression assumption of heteroskedasticity was violated as the variance of the residuals demonstrate an increasing spread along the X-axis. We corrected for the non-constant variance by transforming variables and performing statistical tests to calculate the standard error, or the statistical accuracy of our predictions.

The other regression assumption that was violated was multi-collinearity as there were terms that influenced the outcomes of other independent variables in the model due to their similarity. We corrected multi-collinearity by removing variables that were like others in the regression model such as atemp, workday, and holiday.

Our team faced limitations in creating a robust regression model due to limited variables that may be of interest such as stations, neighborhoods, or the duration of time bikeshares were utilized in. If the variables on location and time were provided in our initial dataset, our regression insights could have yielded stronger predictive results, and the creation of specific actionable recommendations.

Recommendations and Conclusions

We believe that Capital Bike Share could focus their strategic direction on converting casual bikeshare users to become membership holders or increase the number of casual users renting a bike. While at the same time additional research on variables of interest such as the geographic utilization of bikeshare usage across Washington D.C., specifically where bikeshares are used the most in neighborhoods, docking stations, and duration of rides. To remain competitive in the future, Capital Bike Share could introduce a e-bike like New York City, new pricing models for their membership and single use options, and launch a targeted first-time user campaign.

Based on our findings, our team puts forth the following recommendations for Capital Bike Share to consider in the near-term to increase casual bikeshare usage. Additionally, our near-term recommendations could be applied for registered bikeshare users.

1. Introduce e-bikes with pedal assist technology for bikeshare users to rent which are currently used in New York City, which may be a great alternative for casual users to move around the city either during their commutes or for leisure.[2]
2. Place additional bikes during the following times to increase casual bikeshare utilization during weekends, the Spring and Fall, and when the weather is clear and partially cloudy.
3. Create a data collection process to start aggregating user metrics based on geographic metrics of usage for all users, collecting data specifically on neighborhood, docking stations, and duration of rides, so current recommendations could be refined.

Based on our research on bikeshare usage in New York City, we suggest Capital Bike Share investigate the following recommendations that may support the proliferation of their casual userbase, primarily focusing on their pricing models and promotional campaigns. Similarly, these long-term recommendations could be applied to registered users.

[2] "Meet the Citi Bike Ebikes," Citi Bike NYC, n.d., https://citibikenyc.com/how-it-works/electric.

1. Expand their bikeshare plans to include affordable options for memberships and casual users by either charging an affordable low-month rate for membership users or a flat rate per unlimited usage for casual users per day. A competitive market rate for an annual membership is $220, while a flat rate day pass is only $19.[3] Affordable options for bikeshare usage may be an attractive option to convince casual users to convert to become a membership holder.
2. To attract new bikeshare users, Capital Bike Share should launch a promotional campaign targeting first-time users with a three-month special offering a competitive market rate of $0.24 per minute for a traditional bike or e-bikes for rides less than 30 minutes, with a maximum per ride cap for a 30-minute ride at $4.[4] The promotional campaign may attract new casual users to see how convenient a bikeshare could be as an alternative transportation method, with the opportunity to experience a traditional and electric bike.

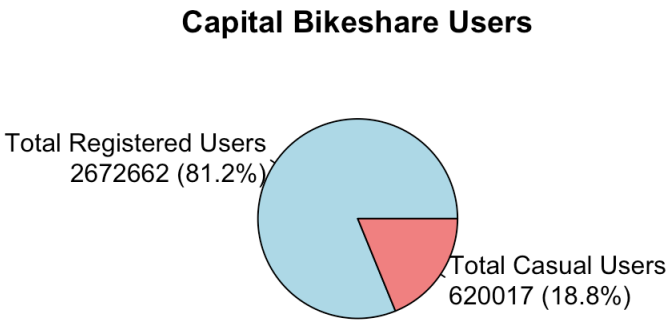
In terms of the ethical considerations and implications of pursuing implementation of our recommendations, Capital Bike Share should be vigilant about using user data to draw conclusions about bikeshare rental usage. One of the core issues numerous companies are under scrutiny today is using customer data to make business decisions, users will have to provide their informed consent as there user data is sensitive, especially when it comes to monetization of user data to innovate products and services.[5] Another implication is potential inherent bias within the dataset used to make decisions about attracting new registered users as 81% of the provided dataset is comprised of membership users. This may produce tactical strategies to target membership users, instead of focusing on casual users.

---

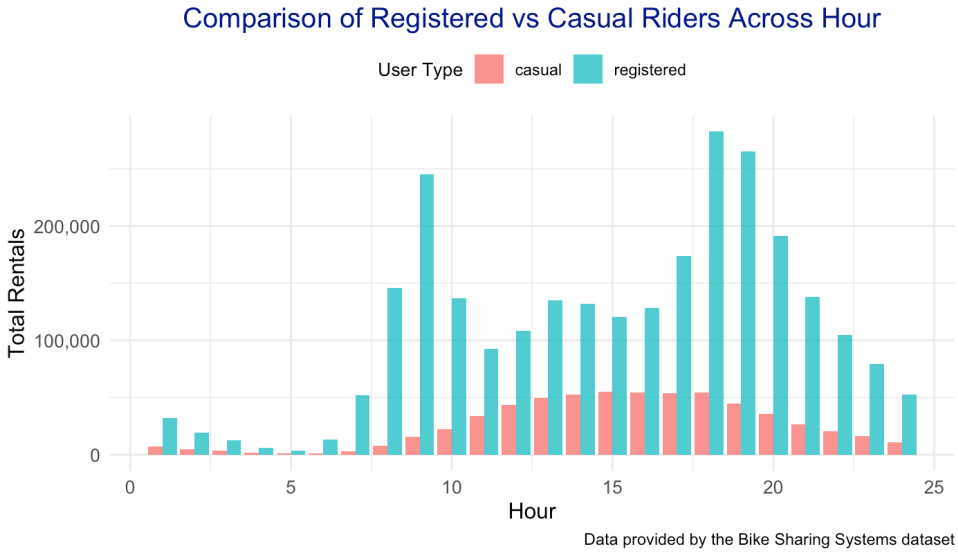
[3] "Citi Bike Membership & Pass Options | Citi Bike NYC," Citi Bike NYC, n.d., https://citibikenyc.com/pricing.
[4] "Meet the Citi Bike Ebikes," n.d.
[5] Winston & Strawn, "Competition Corner," Winston & Strawn - Consumer Data Monetization: The Antitrust Risks You Need to Know, n.d., https://www.winston.com/en/blogs-and-podcasts/competition-corner/consumer-data-monetization-the-antitrust-risks-you-need-to-know.

Technical Appendix

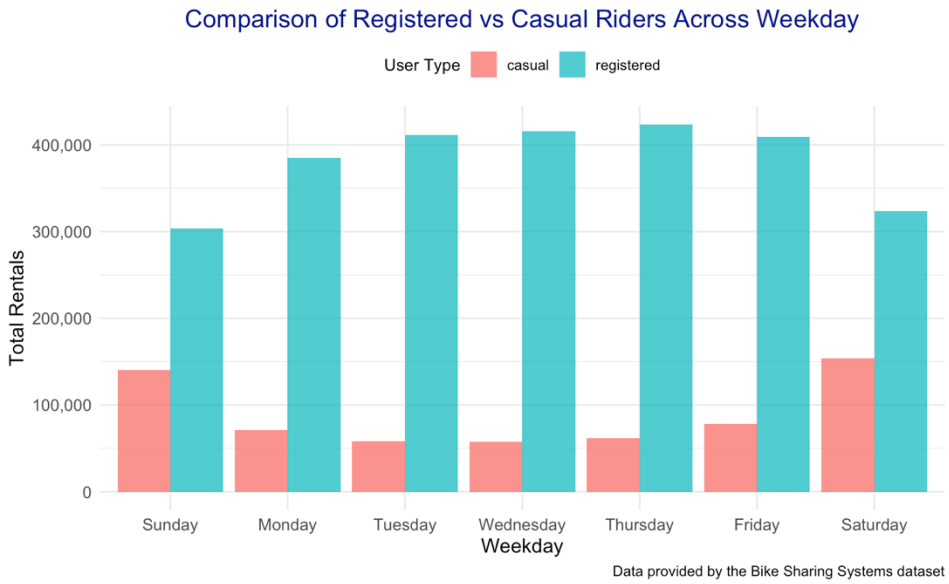
Appendix 1.1: Who are our users? Capital Bike Share users rented over 3.292,679 bikes within the two-year period.



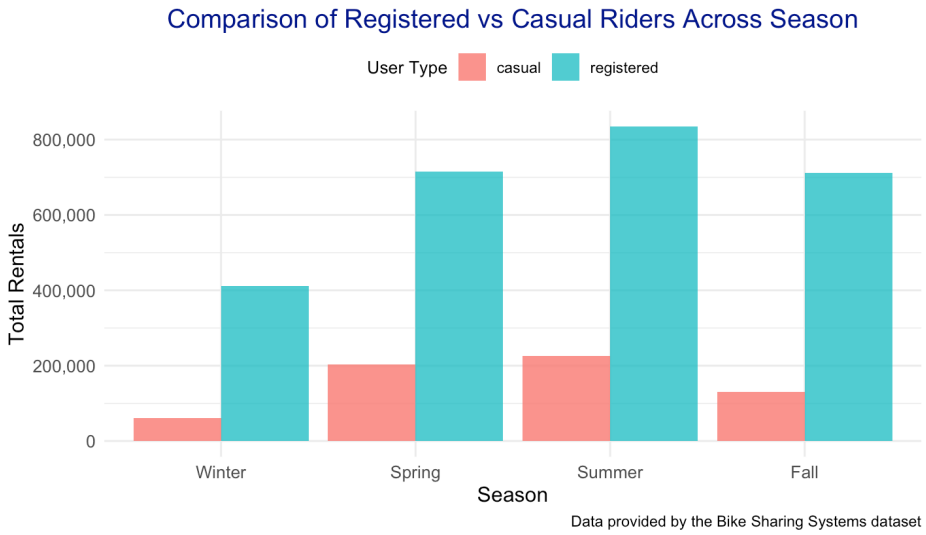
Appendix 1.2: How does bikeshare usage vary across the day? Membership users are more likely to use their bikes around 8am and between 5pm - 6pm. Relative to casual users bikeshare usage increases between 7am - 12pm and consistent usage between 12pm - 5pm.



Appendix 1.3: Which days are popular for bikeshare usage? Registered users are likely to use the bikeshare for their commute to work and from work to home during the weekdays, while casual users are likely to use bikeshares on the weekends for possibly errands or leisure.



Appendix 1.4: How does bikeshare usage vary by season? Registered users are likely to use the bikeshare during the Spring, Summer, and Fall, while casual users likely to use bikeshares in Spring and Summer.



Appendix 1.5: Summary Statistics for Dataset: *This table represents univariate descriptive analysis based on the entire dataset. There are many factors that are categorical or factor variables within the dataset.*

Table: Summary Statistics of Variables in bikesharenew

	Min	Mean	Median	Standard Deviation	Max
instant	1	8690.0000000	8690	5017.0294996	17379
dteday	2011-01-01	NA	2012-01-02	NA	2012-12-31
season	1	2.5016399	3	1.1069181	4
yr	0	0.5025606	1	0.5000078	1
mnth	1	6.5377755	7	3.4387757	12
hr	0	11.5467518	12	6.9144051	23
holiday	0	0.0287704	0	0.1671653	1
weekday	0	3.0036826	3	2.0057715	6
workingday	0	0.6827205	1	0.4654306	1
weathersit	1	1.4252834	1	0.6393569	4
temp	0.02	0.4969872	0.50	0.1925561	1.00
atemp	0.0000	0.4757751	0.4848	0.1718502	1.0000
hum	0.00	0.6272288	0.63	0.1929298	1.00
windspeed	0.0000	0.1900976	0.1940	0.1223402	0.8507
casual	0	35.6762184	17	49.3050304	367
registered	0	153.7868692	115	151.3572859	886
cnt	1	189.4630876	142	181.3875991	977

Appendix 1.6 Correlation Coefficients for Numeric Variables: *This table represents the correlation correlations from our entire dataset, focusing only on numeric variables.*

Table: Correlation Coefficients of Numeric Variables

	instant	temp	atemp	hum	windspeed	casual	registered	cnt
instant	1.00	0.14	0.14	0.01	-0.07	0.16	0.28	0.28
temp	0.14	1.00	0.99	-0.07	-0.02	0.46	0.34	0.40
atemp	0.14	0.99	1.00	-0.05	-0.06	0.45	0.33	0.40
hum	0.01	-0.07	-0.05	1.00	-0.29	-0.35	-0.27	-0.32
windspeed	-0.07	-0.02	-0.06	-0.29	1.00	0.09	0.08	0.09
casual	0.16	0.46	0.45	-0.35	0.09	1.00	0.51	0.69
registered	0.28	0.34	0.33	-0.27	0.08	0.51	1.00	0.97
cnt	0.28	0.40	0.40	-0.32	0.09	0.69	0.97	1.00

Appendix 1.7 Coefficients and P-Values for our Mode and Robust Statistics for Heteroskedasticity: *This table represents the coefficients, robust standard errors, and p-values from our final model.*

Table: Coefficients and P-Value Outputs from the Model

term	estimate	p.value
(Intercept)	1633.3213325	0.0007721
yr1	51.0685174	0.0000169
dteday	-0.1081461	0.0008464
mnth2	0.2340884	0.8933669
mnth3	6.0922179	0.0156680
mnth4	-0.9175669	0.8066122
mnth5	-14.4772907	0.0016507
mnth6	-29.7286833	0.0000001
mnth7	-35.6097727	0.0000000
mnth8	-18.0082657	0.0145901
mnth9	7.9573864	0.3343011
mnth10	27.5406840	0.0027517
mnth11	36.3785273	0.0003424
mnth12	35.0717674	0.0014593
weekday1	-31.8623686	0.0000000
weekday2	-36.2921221	0.0000000
weekday3	-36.5281270	0.0000000
weekday4	-35.8487389	0.0000000
weekday5	-26.8504146	0.0000000
weekday6	5.2883721	0.0000006
temp	-13.0234005	0.1672416
holidayrecode1	16.3471379	0.0000000
workingdayrecode1	NA	NA
weatherrecode2	-5.4545093	0.0000000
weatherrecode3	-14.1536772	0.0000000
weatherrecode4	-3.6026842	0.8678301
seasonrecode2	10.5846307	0.0000000
seasonrecode3	-8.2355955	0.0001005
seasonrecode4	-0.2899987	0.8718212
I(temp^2)	224.5626211	0.0000000
I(windspeed^2)	39.0315633	0.0000000

t test of coefficients:

	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	-0.4359422	0.0316391	-13.7786	< 2.2e-16	***
yr1	0.3437410	0.0129985	26.4448	< 2.2e-16	***
mnth2	0.0315730	0.0252401	1.2509	0.2109928	
mnth3	-0.1706837	0.0296092	-5.7646	8.387e-09	***
mnth4	-0.1740726	0.0504889	-3.4477	0.0005672	***
mnth5	-0.1424118	0.0535844	-2.6577	0.0078777	**
mnth6	-0.0707782	0.0567413	-1.2474	0.2122808	
mnth7	-0.1777815	0.0660222	-2.6928	0.0070962	**
mnth8	-0.1356544	0.0649460	-2.0887	0.0367531	*
mnth9	-0.1017404	0.0576743	-1.7641	0.0777484	.
mnth10	-0.2348929	0.0501447	-4.6843	2.840e-06	***
mnth11	-0.2468114	0.0471744	-5.2319	1.706e-07	***
mnth12	-0.1141927	0.0307362	-3.7152	0.0002039	***
weekday1	0.6216038	0.0237674	26.1537	< 2.2e-16	***
weekday2	0.7037569	0.0235439	29.8913	< 2.2e-16	***
weekday3	0.7437131	0.0241093	30.8475	< 2.2e-16	***
weekday4	0.7123482	0.0229945	30.9791	< 2.2e-16	***
weekday5	0.5725216	0.0202427	28.2829	< 2.2e-16	***
weekday6	0.0545607	0.0122144	4.4669	8.008e-06	***
temp	-0.0547514	0.0159709	-3.4282	0.0006096	***
casual	1.0363094	0.0164430	63.0242	< 2.2e-16	***
holidayrecode1	-0.5413095	0.0329880	-16.4093	< 2.2e-16	***
weatherrecode2	-0.0224266	0.0154994	-1.4469	0.1479418	
weatherrecode3	-0.1082190	0.0197087	-5.4909	4.079e-08	***
weatherrecode4	-0.0614032	0.1126288	-0.5452	0.5856384	
seasonrecode2	0.0624305	0.0375772	1.6614	0.0966602	.
seasonrecode3	0.1332018	0.0490311	2.7167	0.0066034	**
seasonrecode4	0.3751720	0.0357889	10.4829	< 2.2e-16	***
I(temp^2)	-0.0313713	0.0085700	-3.6606	0.0002527	***
I(casual^2)	-0.1773367	0.0058575	-30.2751	< 2.2e-16	***
temp:casual	0.0147308	0.0111785	1.3178	0.1876045	

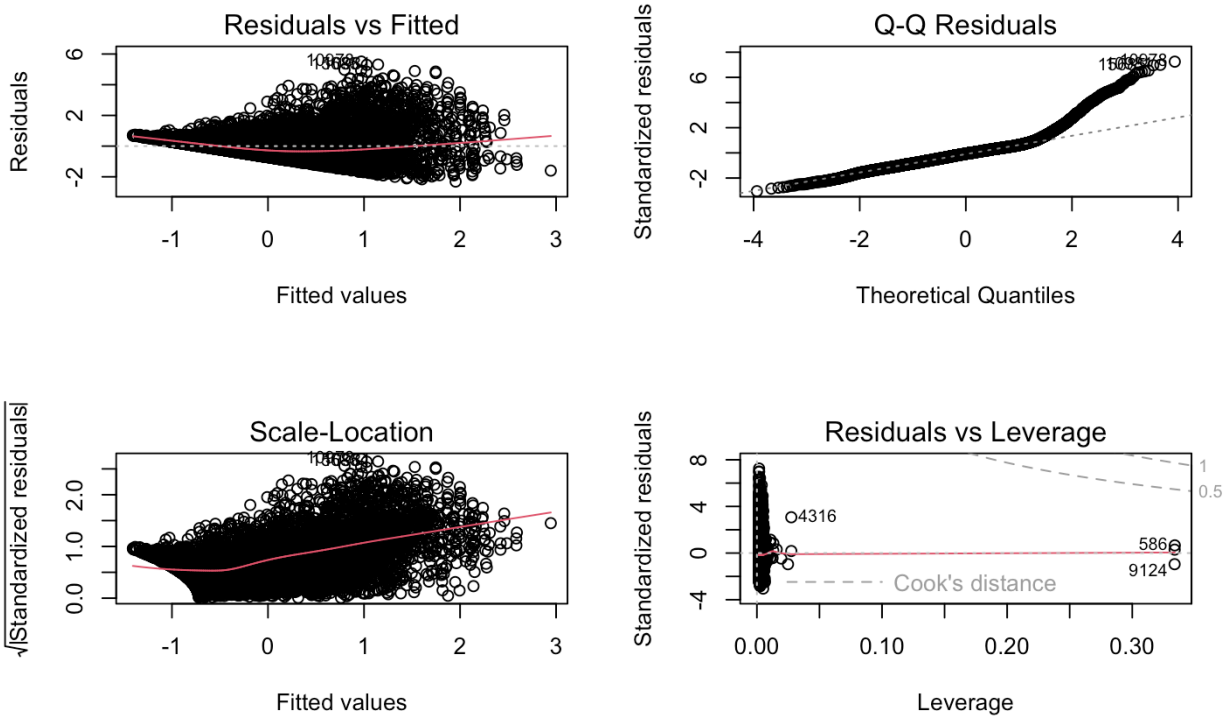
Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Appendix 1.8 Model Statistics: *This table represents the R-squared, adjusted R-squared, and RMSE values from our final model.*

Table: Model Statistics

r_squared	adj_r_squared	rmse
0.4241262	0.4231636	0.9247979

Appendix 1.9 Plot Diagnostics: *This table represents the series of plot diagnostics from our training model.*



Bibliography

"Bike Share in the U.S.: 2017 | National Association of City Transportation Officials," National Association of City Transportation Officials, n.d., https://nacto.org/bike-share-statistics-2017/.

"Citi Bike Membership & Pass Options | Citi Bike NYC," Citi Bike NYC, n.d., https://citibikenyc.com/pricing.

"Meet the Citi Bike Ebikes," Citi Bike NYC, n.d., https://citibikenyc.com/how-it-works/electric.

Winston & Strawn, "Competition Corner," Winston & Strawn - Consumer Data Monetization: The Antitrust Risks You Need to Know, n.d., https://www.winston.com/en/blogs-and-podcasts/competition-corner/consumer-data-monetization-the-antitrust-risks-you-need-to-know.

Project R-Code

```
# Andrew Singh

# PAN 6602 Machine Learning

# Project R Code for Project 1


# install the necessary packages


install.packages("tidyverse") # for data manipulation

install.packages("dplyr") # for data manipulation

install.packages("ggplot2") # data visualization

install.packages("GGally") # for pairs plots with ggplot

install.packages("caret") # our primary modeling package

install.packages("corrplot") # for correlation matrix plots

install.packages("scales") # for a continuous scaling axis

install.packages("broom") # support for tidying up the outputs for statistical models

install.packages("mvrsquared") # calculate the r2 and adjusted r2 for mutlivariate
regression plots

install.packages("ggrid") # package creating the grid-based visualizations for model
diagnostics and assessments

install.packages("lmtest") # performing statistical tests on linear models, including
diagnostic and specification tests

install.packages("sandwich") # methods for robust covariance

install.packages("knitr") # print out reports neatly
```

```
# use the library function


library(tidyverse) # for data manipulation

library(dplyr) # for data manipulation

library(ggplot2) # data visualization

library(GGally) # for pairs plots with ggplot

library(caret) # our primary modeling package

library(corrplot) # for correlation matrix plots

library(scales) # Make sure to load this for the comma and label formatting functions

library(broom) # support for tidying up the outputs for statistical models

library(mvrsquared) # calculate the r2 and adjusted r2 for mutlivariate regression plots

library(ggridges) # package creating the grid-based visualizations for model diagnostics and assessments

library(lmtest) # performing statistical tests on linear models, including diagnostic and specification tests

library(sandwich) # methods for robust covariance

library(car) # for variance inflation factor

library(knitr) # print out reports neatly



# import the dataset and set the working directory

setwd("~/Documents/Machine Learning I/Machine Learning OPAN 6602") # set the working directory for the project folder

bikeshare = read.csv("Data-Raw/Capital Bike Sharing data by hour.csv") # import the dataset for analysis
```

```
# explore the dataset from the original dataset

View(bikeshare) #view the entire the dataset
str(bikeshare) #explore the datatypes of the dataset
dim(bikeshare) #explore the dimensions of the dataset
head(bikeshare) #explore the first five rows of the dataset
tail(bikeshare) #explore the last five rows of the dataset
names(bikeshare) #get the name of the columns in the dataset

# perform basic data exploration of the dataset and check for null values

anyNA(bikeshare) #check if there are any NA values in the dataset
apply(bikeshare, 2, anyNA) #confirm that are no NA values in the dataset with a similar function
sum(is.na(bikeshare)) #there are no null values in the dataset

# explore the dataset for any insights
summary(bikeshare) #explore summary statistics for the dataset

#Convert the fields from integer to categorical and create a new dataframe to store results.

bikesharenew = bikeshare %>%
  mutate(dteday = as.Date(bikeshare$dteday),
         season = as.factor(bikeshare$season),
         yr = as.factor(bikeshare$yr),
```

```
      mnth = as.factor(bikeshare$mnth),
      hr = as.factor(bikeshare$hr),
      holiday = as.factor(bikeshare$holiday),
      weekday = as.factor(bikeshare$weekday),
      workingday = as.factor(bikeshare$workingday),
      weathersit = as.factor(bikeshare$weathersit),
      weatherrecode = as.factor(bikeshare$weathersit),
      holidayrecode = as.factor(bikeshare$holiday),
      workingdayrecode = as.factor(bikeshare$workingday),
      seasonrecode = as.factor(bikeshare$season))

# Recode the variables to name numerical values to categorical value names
# Renaming season
bikesharenew <- bikesharenew %>%
  mutate(seasonnew = case_when(
    season == 1 ~ "Winter",
    season == 2 ~ "Spring",
    season == 3 ~ "Summer",
    season == 4 ~ "Fall"
  ))

# Renaming month
bikesharenew <- bikesharenew %>%
  mutate(month = case_when(
    mnth == 1 ~ "January",
    mnth == 2 ~ "February",
```

```
    mnth == 3 ~ "March",
    mnth == 4 ~ "April",
    mnth == 5 ~ "May",
    mnth == 6 ~ "June",
    mnth == 7 ~ "July",
    mnth == 8 ~ "August",
    mnth == 9 ~ "September",
    mnth == 10 ~ "October",
    mnth == 11 ~ "November",
    mnth == 12 ~ "December"
  ))

# Renaming year
bikesharenew <- bikesharenew %>%
  mutate(year = case_when(
    yr == 0 ~ "2011",
    yr == 1 ~ "2012"
  ))

# Renaming weather
bikesharenew <- bikesharenew %>%
  mutate(weatherrecode = case_when(
    weathersit == 1 ~ "Clear",
    weathersit == 2 ~ "Mist",
    weathersit == 3 ~ "Light snow and rain",
    weathersit == 4 ~ "Heavy rain"
```

```
  ))

# Renaming holiday
bikesharenew <- bikesharenew %>%
  mutate(holidayrecode = case_when(
    holiday == 1 ~ "True",
    holiday == 0 ~ "False"
  ))

# Renaming workingday
bikesharenew <- bikesharenew %>%
  mutate(workingdayrecode = case_when(
    workingday == 1 ~ "if the day is neither a weekend nor holiday",
    workingday == 0 ~ "not"
  ))



# Refactoring variables to become factor categorical variables
bikesharenew = bikeshare %>%
  mutate(dteday = as.Date(bikeshare$dteday),
         season = as.factor(bikeshare$season),
         yr = as.factor(bikeshare$yr),
         mnth = as.factor(bikeshare$mnth),
         hr = as.factor(bikeshare$hr),
         holiday = as.factor(bikeshare$holiday),
```

```
    weekday = as.factor(bikeshare$weekday),
    workingday = as.factor(bikeshare$workingday),
    weathersit = as.factor(bikeshare$weathersit),
    weatherrecode = as.factor(bikeshare$weathersit),
    holidayrecode = as.factor(bikeshare$holiday),
    workingdayrecode = as.factor(bikeshare$workingday),
    seasonrecode = as.factor(bikeshare$season))

# explore the new dataset for bikesharenew

View(bikesharenew) #view the entire the dataset
str(bikesharenew) #explore the datatypes of the dataset
dim(bikesharenew) #explore the dimensions of the dataset
head(bikesharenew) #explore the first five rows of the dataset
tail(bikesharenew) #explore the last five rows of the dataset
names(bikesharenew) #get the name of the columns in the dataset

# perform basic data exploration of the dataset and check for null values

anyNA(bikesharenew) #check if there are any NA values in the dataset
apply(bikesharenew, 2, anyNA) #confirm that are no NA values in the dataset with a similar function
sum(is.na(bikesharenew)) #there are no null values in the dataset

# explore the dataset for any insights
summary(bikesharenew) #explore summary statistics for the dataset
```

```
summary(bikeshare)

# Double check the values in each recoded column is the same as the original dataframe

# Create a vector with the list of variables to analyze
variables <- c("instant", "dteday", "season", "yr", "mnth",
               "hr", "holiday", "weekday", "workingday", "weathersit",
               "temp", "atemp", "hum", "windspeed", "casual",
               "registered", "cnt", "weatherrecode", "holidayrecode",
               "workingdayrecode", "seasonrecode")

# Loop through variables and count subcategory values and the entire number records
for (var in variables) {
  cat("\nCounts for variable:", var, "\n")
  print(length(bikesharenew[[var]]))
  print(table(bikesharenew[[var]]))}



# List of variables to analyze for the original dataframe
variables <- c("instant", "dteday", "season", "yr", "mnth",
               "hr", "holiday", "weekday", "workingday", "weathersit",
               "temp", "atemp", "hum", "windspeed", "casual",
               "registered", "cnt")
```

```
# Loop through variables and count subcategory values and the entire records of the original dataframe
for (var in variables) {
  cat("\nCounts for variable:", var, "\n")
  print(length(bikeshare[[var]]))
  print(table(bikeshare[[var]]))}

# Univariate analysis for holiday, weekday, workingday, weathersit, temp,
# humidity, windspeed, registered, casual, and count.

str(bikesharenew) # display the datatypes for each variable

# Filter numeric columns only

# Apply functions to numeric data
Min <- apply(bikeshare, 2, min, na.rm = TRUE) # create the min value vector
Mean <- sapply(bikeshare, function(col) { # create the mean value vector using a function
  if (is.numeric(col)) {
    mean(col, na.rm = TRUE)  #
  } else {
    NA  #
  }})
Median <- apply(bikeshare, 2, median, na.rm = TRUE) # create the median value vector
Standard_Deviation <- apply(bikeshare, 2, sd, na.rm = TRUE) # create the standard deviation value vector
Max <- apply(bikeshare, 2, max, na.rm = TRUE) # create the max value vector
```

```
# Combine results into a summary table
summary_table <- cbind(data.frame(Min, Mean, Median, Standard_Deviation, Max))
print(summary_table)


# Output the table neatly using knitr::kable
knitr::kable(
  summary_table,
  col.names = c("Min","Mean", "Median", "Standard Deviation", "Max"),
  caption = "Summary Statistics of Variables in bikesharenew",
  format = "markdown", # Change to "latex" or "markdown" depending on your report type
  align = "c"
)


# Data transformation: Creating a new dataframe 'bikeshare_long' - bikeshare_long is used for the ggplot visualizations
bikeshare_long <- bikesharenew %>%
  mutate(
    # Convert `dteday` to Date format
    Date = as.Date(dteday),


    # Recode and convert weather condition as a factor
    Weather = recode_factor(weathersit,
                            `1` = "Clear, cloudy",
                            `2` = "Mist + cloudy, mist",
                            `3` = "Light snow and rain",
                            `4` = "Heavy rain + ice pellets"),
```

```
# Recode and convert holiday variable as a factor
Holiday = recode_factor(holiday,
                        `1` = "Yes",
                        `0` = "No"),

# Recode and convert working day variable as a factor
Workday = recode_factor(workingday,
                        `1` = "Working day",
                        `0` = "Non-working day"),

# Recode and convert month as a factor
Month = recode_factor(mnth,
                      `1` = "January",
                      `2` = "February",
                      `3` = "March",
                      `4` = "April",
                      `5` = "May",
                      `6` = "June",
                      `7` = "July",
                      `8` = "August",
                      `9` = "September",
                      `10` = "October",
                      `11` = "November",
                      `12` = "December"),
```

```
# Recode and convert weekday as a factor
Weekday = recode_factor(weekday,
                        `0` = "Sunday",
                        `1` = "Monday",
                        `2` = "Tuesday",
                        `3` = "Wednesday",
                        `4` = "Thursday",
                        `5` = "Friday",
                        `6` = "Saturday"),

# Recode and convert year as a factor
Year = recode_factor(yr,
                     `0` = "2011",
                     `1` = "2012"),

# Recode feel-like temperature
Relative_Temperature = atemp * 100,

# Recode temperature
Temperature = temp * 100,

# Recode and convert season as a factor
Season = recode_factor(season,
                       `1` = "Winter",
                       `2` = "Spring",
                       `3` = "Summer",
```

```
                `4` = "Fall"),

  # Convert `hour` to integer
  Hour = as.integer(hr)
 )


# Verify the transformed data
head(bikeshare_long)

# Summarize the data for Date
summarized_data <- bikeshare_long %>%
 pivot_longer(cols = c(registered, casual), names_to = "user_type", values_to = "count")
%>%
 drop_na(count) %>%
 group_by(Date, user_type) %>%
 summarize(total_count = sum(count), .groups = 'drop')

# Plot the data
ggplot(summarized_data, aes(x = Date, y = total_count, fill = user_type)) +
 geom_col(position = "dodge", alpha = 0.8) +
 labs(
  x = "Date",
  y = "Total Ride Count",
  title = "Comparison of Registered vs Casual Riders Across Date",
  caption = "Data provided by the Bike Sharing Systems dataset",
```

```
  fill = "User Type"
 ) +
 theme_minimal() +
 theme(
  plot.title = element_text(hjust = 0.5, size = 15, color = "darkblue"),
  axis.title.x = element_text(size = 12),
  axis.title.y = element_text(size = 12),
  axis.text.x = element_text(size = 10),
  axis.text.y = element_text(size = 10),
  legend.title = element_text(size = 10),
  legend.position = "top"
 ) +
 scale_y_continuous(labels = scales::comma)



# Summarize the data for Hour
summarized_data <- bikeshare_long %>%
 pivot_longer(cols = c(registered, casual), names_to = "user_type", values_to = "count")
%>%
 drop_na(count) %>%
 group_by(Hour, user_type) %>%
 summarize(total_count = sum(count), .groups = 'drop')


# Plot the data for Hour
ggplot(summarized_data, aes(x = Hour, y = total_count, fill = user_type)) +
 geom_col(position = "dodge", alpha = 0.8) +
```

```
  labs(
    x = "Hour",
    y = "Total Rentals",
    title = "Comparison of Registered vs Casual Riders Across Hour",
    caption = "Data provided by the Bike Sharing Systems dataset",
    fill = "User Type"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(hjust = 0.5, size = 15, color = "darkblue"),
    axis.title.x = element_text(size = 12),
    axis.title.y = element_text(size = 12),
    axis.text.x = element_text(size = 10),
    axis.text.y = element_text(size = 10),
    legend.title = element_text(size = 10),
    legend.position = "top"
  ) +
  scale_y_continuous(labels = scales::comma)



# Summarize the data for Month
summarized_data <- bikeshare_long %>%
  pivot_longer(cols = c(registered, casual), names_to = "user_type", values_to = "count")
%>%
  drop_na(count) %>%
  group_by(Month, user_type) %>%
```

```
  summarize(total_count = sum(count), .groups = 'drop')


# Plot the data for Month
ggplot(summarized_data, aes(x = Month, y = total_count, fill = user_type)) +
  geom_col(position = "dodge", alpha = 0.8) +
  labs(
    x = "Month",
    y = "Total Rentals",
    title = "Comparison of Registered vs Casual Riders Across Month",
    caption = "Data provided by the Bike Sharing Systems dataset",
    fill = "User Type"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(hjust = 0.5, size = 15, color = "darkblue"),
    axis.title.x = element_text(size = 12),
    axis.title.y = element_text(size = 12),
    axis.text.x = element_text(size = 10),
    axis.text.y = element_text(size = 10),
    legend.title = element_text(size = 10),
    legend.position = "top"
  ) +
  scale_y_continuous(labels = scales::comma)


# Summarize the data for Relative Temperature
```

```
summarized_data <- bikeshare_long %>%
  pivot_longer(cols = c(registered, casual), names_to = "user_type", values_to = "count") %>%
  drop_na(count) %>%
  group_by(Relative_Temperature, user_type) %>%
  summarize(total_count = sum(count), .groups = 'drop')

# Plot the data for Relative Temperature
ggplot(summarized_data, aes(x = Relative_Temperature, y = total_count, fill = user_type)) +
  geom_col(position = "dodge", alpha = 0.8) +
  labs(
    x = "Relative Temperature",
    y = "Total Rentals",
    title = "Comparison of Registered vs Casual Riders Across Relative Temperature",
    caption = "Data provided by the Bike Sharing Systems dataset",
    fill = "User Type"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(hjust = 0.5, size = 15, color = "darkblue"),
    axis.title.x = element_text(size = 12),
    axis.title.y = element_text(size = 12),
    axis.text.x = element_text(size = 10),
    axis.text.y = element_text(size = 10),
    legend.title = element_text(size = 10),
    legend.position = "top"
```

```
  ) +
  scale_y_continuous(labels = scales::comma)



# Summarize the data for Temperature
summarized_data <- bikeshare_long %>%
  pivot_longer(cols = c(registered, casual), names_to = "user_type", values_to = "count")
%>%
  drop_na(count) %>%
  group_by(Temperature, user_type) %>%
  summarize(total_count = sum(count), .groups = 'drop')

# Plot the data for Relative Temperature
ggplot(summarized_data, aes(x = Temperature, y = total_count, fill = user_type)) +
  geom_col(position = "dodge", alpha = 0.8) +
  labs(
    x = "Temperature",
    y = "Total Rentals",
    title = "Comparison of Registered vs Casual Riders Across Temperature",
    caption = "Data provided by the Bike Sharing Systems dataset",
    fill = "User Type"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(hjust = 0.5, size = 15, color = "darkblue"),
    axis.title.x = element_text(size = 12),
```

```
    axis.title.y = element_text(size = 12),
    axis.text.x = element_text(size = 10),
    axis.text.y = element_text(size = 10),
    legend.title = element_text(size = 10),
    legend.position = "top"
  ) +
  scale_y_continuous(labels = scales::comma)

# Summarize the data for Weekday
summarized_data <- bikeshare_long %>%
  pivot_longer(cols = c(registered, casual), names_to = "user_type", values_to = "count") %>%
  drop_na(count) %>%
  group_by(Weekday, user_type) %>%
  summarize(total_count = sum(count), .groups = 'drop')

# Plot the data for Weekday
ggplot(summarized_data, aes(x = Weekday, y = total_count, fill = user_type)) +
  geom_col(position = "dodge", alpha = 0.8) +
  labs(
    x = "Weekday",
    y = "Total Rentals",
    title = "Comparison of Registered vs Casual Riders Across Weekday",
    caption = "Data provided by the Bike Sharing Systems dataset",
    fill = "User Type"
  ) +
```

```
 theme_minimal() +
 theme(
  plot.title = element_text(hjust = 0.5, size = 15, color = "darkblue"),
  axis.title.x = element_text(size = 12),
  axis.title.y = element_text(size = 12),
  axis.text.x = element_text(size = 10),
  axis.text.y = element_text(size = 10),
  legend.title = element_text(size = 10),
  legend.position = "top"
 ) +
 scale_y_continuous(labels = scales::comma)



# Summarize the data for Workday
summarized_data <- bikeshare_long %>%
 pivot_longer(cols = c(registered, casual), names_to = "user_type", values_to = "count")
%>%
 drop_na(count) %>%
 group_by(Workday, user_type) %>%
 summarize(total_count = sum(count), .groups = 'drop')

# Plot the data for Workday
ggplot(summarized_data, aes(x = Workday, y = total_count, fill = user_type)) +
 geom_col(position = "dodge", alpha = 0.8) +
 labs(
  x = "Workday",
```

```
    y = "Total Rentals",
    title = "Comparison of Registered vs Casual Riders Across Workday",
    caption = "Data provided by the Bike Sharing Systems dataset",
    fill = "User Type"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(hjust = 0.5, size = 15, color = "darkblue"),
    axis.title.x = element_text(size = 12),
    axis.title.y = element_text(size = 12),
    axis.text.x = element_text(size = 10),
    axis.text.y = element_text(size = 10),
    legend.title = element_text(size = 10),
    legend.position = "top"
  ) +
  scale_y_continuous(labels = scales::comma)



# Summarize the data for Season
summarized_data <- bikeshare_long %>%
  pivot_longer(cols = c(registered, casual), names_to = "user_type", values_to = "count")
%>%
  drop_na(count) %>%
  group_by(Season, user_type) %>%
  summarize(total_count = sum(count), .groups = 'drop')
```

```
# Plot the data for Season
ggplot(summarized_data, aes(x = Season, y = total_count, fill = user_type)) +
  geom_col(position = "dodge", alpha = 0.8) +
  labs(
    x = "Season",
    y = "Total Rentals",
    title = "Comparison of Registered vs Casual Riders Across Season",
    caption = "Data provided by the Bike Sharing Systems dataset",
    fill = "User Type"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(hjust = 0.5, size = 15, color = "darkblue"),
    axis.title.x = element_text(size = 12),
    axis.title.y = element_text(size = 12),
    axis.text.x = element_text(size = 10),
    axis.text.y = element_text(size = 10),
    legend.title = element_text(size = 10),
    legend.position = "top"
  ) +
  scale_y_continuous(labels = scales::comma)

# Summarize the data for Holiday
summarized_data <- bikeshare_long %>%
  pivot_longer(cols = c(registered, casual), names_to = "user_type", values_to = "count")
%>%
```

```
  drop_na(count) %>%
  group_by(Holiday, user_type) %>%
  summarize(total_count = sum(count), .groups = 'drop')

# Plot the data for Holiday
ggplot(summarized_data, aes(x = Holiday, y = total_count, fill = user_type)) +
  geom_col(position = "dodge", alpha = 0.8) +
  labs(
    x = "Holiday",
    y = "Total Rentals",
    title = "Comparison of Registered vs Casual Riders Across Holiday",
    caption = "Data provided by the Bike Sharing Systems dataset",
    fill = "User Type"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(hjust = 0.5, size = 15, color = "darkblue"),
    axis.title.x = element_text(size = 12),
    axis.title.y = element_text(size = 12),
    axis.text.x = element_text(size = 10),
    axis.text.y = element_text(size = 10),
    legend.title = element_text(size = 10),
    legend.position = "top"
  ) +
  scale_y_continuous(labels = scales::comma)
```

```
# Summarize the data for Weather
summarized_data <- bikeshare_long %>%
  pivot_longer(cols = c(registered, casual), names_to = "user_type", values_to = "count") %>%
  drop_na(count) %>%
  group_by(Weather, user_type) %>%
  summarize(total_count = sum(count), .groups = 'drop')

# Plot the data for Weather
ggplot(summarized_data, aes(x = Weather, y = total_count, fill = user_type)) +
  geom_col(position = "dodge", alpha = 0.8) +
  labs(
    x = "Weather",
    y = "Total Rentals",
    title = "Comparison of Registered vs Casual Riders Across Holiday",
    caption = "Data provided by the Bike Sharing Systems dataset",
    fill = "User Type"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(hjust = 0.5, size = 15, color = "darkblue"),
    axis.title.x = element_text(size = 12),
    axis.title.y = element_text(size = 12),
    axis.text.x = element_text(size = 10),
    axis.text.y = element_text(size = 10),
    legend.title = element_text(size = 10),
```

```
  legend.position = "top"
 ) +
 scale_y_continuous(labels = scales::comma)

# Summarize the data for Temperature
summarized_data <- bikeshare_long %>%
 pivot_longer(cols = c(registered, casual), names_to = "user_type", values_to = "count")
%>%
 drop_na(count) %>%
 group_by(atemp, user_type) %>%
 summarize(total_count = sum(count), .groups = 'drop')

# Plot the data for Temperature
ggplot(summarized_data, aes(x = atemp, y = total_count, fill = user_type)) +
 geom_col(position = "dodge", alpha = 0.8) +
 labs(
  x = "Feel-Like Temperature",
  y = "Total Rentals",
  title = "Comparison of Registered vs Casual Riders Across Feel-Like Temperature",
  caption = "Data provided by the Bike Sharing Systems dataset",
  fill = "User Type"
 ) +
 theme_minimal() +
 theme(
  plot.title = element_text(hjust = 0.5, size = 15, color = "darkblue"),
  axis.title.x = element_text(size = 12),
```

```
  axis.title.y = element_text(size = 12),
  axis.text.x = element_text(size = 10),
  axis.text.y = element_text(size = 10),
  legend.title = element_text(size = 10),
  legend.position = "top"
 ) +
 scale_y_continuous(labels = scales::comma)


## Creating the bins for the feel like temperature
bikesharenew |>
 transmute(
  atempbinned = cut(atemp, 20)
 ) |>
 table() |>
 barplot()

body_mass_cut <- cut(bikesharenew$atempbinned, 10)



#How do the time variables relate to each other?
#Based on the output, identify any correlations between the independent variables and the dependent variable.

str(bikeshare) # look at the data types
```

```
bikesharecorr = bikesharenew %>% select_if(is.numeric) # create a vector for the data types to only hold numeric values


corrplot(cor_matrix, method = "circle", type = "upper", tl.col = "black", tl.srt = 45, title = "Correlation Matrix of Numerical Variables") # create an optional corrplot


cor_matrix = round(cor(bikesharecorr),2) # create a matrix with correlation coefficients



# Output the table neatly using knitr::kable


correlation_table <- data.frame(cor_matrix) # save the correlation coefficients into a dataframe

print(correlation_table) # print out the correlation coefficient table


knitr::kable(

  correlation_table, # The data frame to display

  caption = "Correlation Coefficients of Numeric Variables", # Table title

  format = "markdown", # Specify the output format ("markdown" for Markdown reports, "latex" for LaTeX reports, etc.)

  align = "c" # Center-align columns

)


cor(bikesharenew$temp, bikesharenew$casual) # there is a moderately weak correlation 0.45 between the temperature and casual users

cor(bikesharenew$temp, bikesharenew$cnt) #there is a moderately weak correlation 0.40 between temp and total users
```

```
cor(bikesharenew$workingday,bikesharenew$casual) #there is a negatively weak correlation between workingday and casual users


# Building the regression models


# set seed for reproducibility

set.seed(90210)


### Step 1: Create a test/train split


bikesharenew_partition =

 createDataPartition(

  1:nrow(bikesharenew),

  times = 1,

  p = .3

 )


bikesharenew_train =

 bikesharenew[-bikesharenew_partition[[1]], ]



bikesharenew_test =

 bikesharenew[bikesharenew_partition[[1]], ]



## Step 2: Data Exploration
```

```
str(bikesharenew) #numeric variables are temp, atemp, humidity, windspeed


bikesharenew_eda <- bikesharenew_train %>%
  dplyr::select(where(is.numeric),
                instant,
                dteday,
                windspeed,
                yr,
                -hr,
                mnth,
                weekday,
                temp,
                hum,
                atemp,
                casual,
                registered,
                holidayrecode,
                workingdayrecode,
                weatherrecode,
                seasonrecode)



bikesharenew_eda %>%
  ggpairs(aes(color = seasonrecode, alpha = 0.3))
```

```
bikesharenew_eda %>%
 ggpairs(aes(color = yr, alpha = 0.3))

bikesharenew_eda %>%
 ggpairs(aes(color = mnth, alpha = 0.3))

bikesharenew_eda %>%
 ggpairs(aes(color = holidayrecode, alpha = 0.3))

bikesharenew_eda %>%
 ggpairs(aes(color = weekday, alpha = 0.3))

bikesharenew_eda %>%
 ggpairs(aes(color = workingdayrecode, alpha = 0.3))

bikesharenew_eda %>%
 ggpairs(aes(color = weatherrecode, alpha = 0.3))



### Step 3: Data pre-processing ----
# There is no need to perform data pre-processing as the
# dataset has no missing values.
anyNA(bikesharenew_eda)
```

```
### Step 4: Feature Engineering ----
## Regularize the numeric variables using the caret package
## Add quadratic terms for each of the regularized variables

standardizer = # create a procedure to standardize data (store means and sds)
 bikesharenew_train %>%
 select(
  where(is.numeric)
 ) %>%
 preProcess(
  method = c("center","scale")
 )

bikesharenew_train_standardized =
 predict(standardizer, bikesharenew_train)

### Step 5: Feature & Model Selection ----
## build a kitchen sink model:
## include all regularized and quadratic terms and all categorical variables

#this model keeps humidity and windspeed (even the polynominal term for windspeed)
f1 = lm(
 casual ~
  yr +
  dteday +
  atemp +
```

```
  mnth +
  weekday +
  temp +
  hum +
  windspeed +
  holidayrecode +
  workingdayrecode +
  weatherrecode +
  seasonrecode +
  I(temp ^ 2) +
  I(hum ^2) +
  I(windspeed ^ 2),
 data = bikesharenew_train_standardized
)
summary(f1)

#model two - using registered as the response variable
f2 = lm(
 registered ~
  yr +
  dteday +
  mnth +
  weekday +
  temp +
  holidayrecode +
  workingdayrecode +
```

```
   weatherrecode +
   seasonrecode +
   I(temp ^ 2) +
   I(hum ^2) +
   I(windspeed ^ 2),
  data = bikesharenew_train_standardized
)



#model three - using casual as the response variable
f3 = lm(
  casual ~
   yr +
   dteday +
   mnth +
   weekday +
   temp +
   holidayrecode +
   workingdayrecode +
   weatherrecode +
   seasonrecode +
   I(temp ^ 2) +
   I(windspeed ^ 2),
  data = bikesharenew_train_standardized
)
```

```
summary(f3)

#model 4 removes workingdayrecode as there is no statisical impact to the R2 coefficients

f4 = lm(
  registered ~
    yr +
    mnth +
    weekday +
    temp +
    casual +
    holidayrecode +
    weatherrecode +
    seasonrecode +
    I(temp ^ 2) +
    I(casual ^2) +
    I(temp ^ 3) +
    I(casual * temp),
  data = bikesharenew_train_standardized
)


#model 5 - this model looks at cnt as the dependent variable

#this model keeps humidity and windspeed (even the polynominal term for windspeed)
f5 = lm(
```

```
  casual ~
    yr +
    dteday +
    atemp +
    mnth +
    weekday +
    temp +
    hum +
    cnt +
    windspeed +
    holidayrecode +
    workingdayrecode +
    weatherrecode +
    seasonrecode +
    I(temp ^ 2) +
    I(casual ^2) +
    I(windspeed ^2) +
    I(temp ^3),
  data = bikesharenew_train_standardized
)

## summarize and plot your kitchen sink model
## do you see evidence of model misspecification in the plots?

#test model 1
summary(f1)
```

```
# Plot the model 1
plot(f1)

#test model 2
summary(f2)

plot(f2)

#test model 3
summary(f3)

plot(f3)

# Set up a 2x2 plotting grid
par(mfrow = c(2, 2))

# Plot all model diagnostics
plot(f3)

# Reset the plotting grid to default
par(mfrow = c(1, 1))
```

```
# Do backwards selection on your kitchen sink model

## summarize and plot it for model 1


f_back1 = step(
  object = f1,
  direction = "back"
)


summary(f_back1)


plot(f_back1)


# backwards selection for model 2
f_back2 = step(
  object = f2,
  direction = "back"
)


summary(f_back2)


plot(f_back2)



# backwards selection for model 3
f_back3 = step(
  object = f3,
```

```
  direction = "back"
)

summary(f_back3)

plot(f_back3)



# forward selection for model 1
f_forward1 = step(
  object = f1,
  direction = "forward"
)

summary(f_forward1)

plot(f_forward1)

# forward selection for model 2
f_forward2 = step(
  object = f2,
  direction = "forward"
)

summary(f_forward2)
```

```
plot(f_forward2)

# forward selection for model 3
f_forward3 = step(
  object = f3,
  direction = "forward"
)

summary(f_forward3)

plot(f_forward3)



## Do stepwise selection starting with your kitchen sink model
## summarize and plot it for model 1

f_step1 = step(
  object = f1,
  direction = "both"
)

summary(f_step1)

plot(f_step1)

## Do stepwise selection for model 2
```

```
f_step2 = step(
  object = f2,
  direction = "both"
)

summary(f_step2)

plot(f_step2)



## Do stepwise selection for model 3

f_step3 = step(
  object = f3,
  direction = "both"
)

summary(f_step3)

plot(f_step3)



### Step 6: Model Evaluation ----
## Create a 10-fold partition of indices on your training set using caret
## Call that object k_fold_indices
```

```
k_fold_indices = createFolds(
  1:nrow(bikesharenew_train_standardized),
  k = 10
)

my_training_function = function(dat){
  f = lm(
    casual ~
      yr +
      dteday +
      mnth +
      weekday +
      temp +
      holidayrecode +
      workingdayrecode +
      weatherrecode +
      seasonrecode +
      I(temp ^ 2) +
      I(windspeed ^ 2),
    data = dat
  )
  f
}
```

```
# Declare an evaluation function
my_evaluation_function <- function(model, new_data) {

 # remove any NA values for simplicity
 new_data <-
  new_data |>
  na.omit()

 preds <- predict(model, new_data)

 oos_r2 <- calc_rsquared(
  y = new_data$cnt,
  yhat = preds
 )

 oos_rmse <- ((new_data$cnt - preds) ^ 2) |>
  mean() |>
  sqrt()

 tibble(
  r2 = oos_r2,
  rmse = oos_rmse
 )
}

## loop over your k-fold indices to get k-fold cross validation of your model
```

```
cv_results =
 k_fold_indices %>%
 map( # lapply() also works.
  function(fold){
    train <- bikesharenew_train_standardized[-fold, ]

    test <- bikesharenew_train_standardized[fold, ]

    f <- my_training_function(train)

    eval <- my_evaluation_function(f3, test)

    eval
  }
 ) %>%
 bind_rows()



# K-fold cross-validation
cv_results =
 k_fold_indices %>%
 map(
  function(fold) {
    train <- bikesharenew_train_standardized[-fold, ]
    test <- bikesharenew_train_standardized[fold, ]
```

```
    # Train the model
    f <- my_training_function(train)

    # Evaluate the model
    eval <- my_evaluation_function(f3, test)

    eval
  }
 ) %>%
 bind_rows()



summary(cv_results)
## Plot a histogram or density of your k-fold r-squared and RMSE
## Did your cross-validation results do better or worse than training on the whole set?
## Do you think this model is good enough to move forward, or do we need to do more tweaking?

cv_results %>%
  ggplot(aes(x = r2)) +
  geom_density(fill = "red", alpha = 0.5)

cv_results %>%
  ggplot(aes(x = rmse)) +
  geom_density(fill = "red", alpha = 0.5)
```

Step 7: Predictions and Conclusions ----

Get out-of sample predictions from your model using the test set

Compare the R-squared and RMSE. Are they worse or better than the CV above?

```
bikesharenew_test_standardized = predict(standardizer, bikesharenew_test)

# let's remove the missing value in bikesharenew_test_standardized

bikesharenew_test_standardized =
  bikesharenew_test_standardized %>%
  drop_na()

oos_predictions <- predict(f3, bikesharenew_test_standardized)

oos_results <- tibble(
  r2 = calc_rsquared(
    y = bikesharenew_test_standardized$cnt,
    yhat = oos_predictions |> na.omit()
  ),
  rmse = ((bikesharenew_test_standardized$cnt - oos_predictions) ^ 2) |> #– RMSE values.
    mean(na.rm = TRUE) |>
    sqrt()
)
```

```
print(oos_results)

## Train your model on the whole dataset, using the specification from above

f_final <- my_training_function(bikesharenew)

## Use tidy() from the broom package to extract your coefficients

f_final_coeffs <- tidy(f_final) # create the final coefficients from the model
summary(f_final)$coefficients # print only the coefficients
r_squared = summary(f_final)$r.squared # print only the R-squared
adj_r_squared = summary(f_final)$adj.r.squared # print only the adjusted R-squared
rmse = ((bikesharenew_test_standardized$cnt - oos_predictions) ^ 2) |> #– RMSE values.
  mean(na.rm = TRUE) |>
  sqrt()

other_metrics = cbind(data.frame(r_squared,adj_r_squared,rmse))
print(other_metrics)

knitr::kable( # format the dataframe into a clean table
  other_metrics, # The data frame to display
  caption = "Model Statistics", # Table title
  format = "markdown", # Specify the output format ("markdown" for Markdown reports, "latex" for LaTeX reports, etc.)
  align = "c" # Center-align columns
)
```

```
# Create a formatted coefficient table

final_coeffs <- data.frame(f_final_coeffs) # save the correlation coefficients into a dataframe

print(final_coeffs) # print out the correlation coefficient table

formatted_coeffs = final_coeffs %>%  # select only coefficients and p-values

  select(term, estimate, p.value)

format_coeffs = cbind(data.frame(formatted_coeffs)) # create a dataframe with the selected variables

print(format_coeffs) # print the dataframr

knitr::kable( # format the dataframe into a clean table

  format_coeffs, # The data frame to display

  caption = "Coefficients and P-Value Outputs from the Model", # Table title

  format = "markdown", # Specify the output format ("markdown" for Markdown reports, "latex" for LaTeX reports, etc.)

  align = "c" # Center-align columns

)

print(f_final_coeffs) # – Coefficients for each independent variable.
```

```
print(f_final_coeffs, n= 40)


## Create a barplot with 95% confidence intervals for all coefficients
## except the intercept
f_final_coeffs |>
 filter(term != "(Intercept)") |>
 ggplot(aes(x = term, y = estimate)) +
 geom_bar(fill = "skyblue", stat = "identity") +
 geom_errorbar(
  aes(x = term, ymin = estimate - 1.96 * std.error, ymax = estimate + 1.96 * std.error),
  width = 0.4,
  color = "orange",
  alpha = 0.9
 )



# Correcting for linear regression assummptions that were violated


# Marginal effects


## set up some data
### Fit a model ----
# Note: this example assumes you've already gone through our modeling steps


f3 = lm(
 registered ~
```

```
  yr +
  mnth +
  weekday +
  temp +
  casual +
  holidayrecode +
  workingdayrecode +
  weatherrecode +
  seasonrecode +
  I(temp ^ 2) +
  I(casual ^2) +
  casual * temp,
 data = bikesharenew_train_standardized
)

summary(f3)

plot(f3)



### Extract coefficients ----

coefs <- tidy(f3)

print(coefs)
```

```
#multicollinearity


f3 <- tibble(
  x1 = rnorm(n = 1000, mean = 7, sd = 3),


  x2 = rnorm(n = 1000, mean = 3, sd = 7),


  x3 = x2 + rnorm(n = 1000, mean = 0, sd = 0.5),


  y = 3 + 5 * x1 + 3 * x2 + rnorm(n = 1000, mean = 0, sd = 20)
)



# Discuss any limitations of the model, suhc as the potential for omitted variable or biases in the data.



Correcting for linear regression assummptions that were violated


# Marginal effects


# Fit your model
f3 <- lm(
  registered ~ yr + mnth + weekday + temp + casual + holidayrecode +
    workingdayrecode + weatherrecode + seasonrecode + I(temp^2) +
    I(casual^2) + casual * temp,
```

```
  data = bikesharenew_train_standardized
)

# View summary of the model
summary(f3)

# Extract coefficients
coefs <- tidy(f3)
print(coefs, n = 40)

# Heteroskedasticity
### Model plots 1 and 3 look heteroskedastic ----

# w know it's from x2
plot(f3$x2, f3$residuals)

# Breusch-Pagan test against heteroskedasticity
bptest(f3)

# Get robust standard errors, vcov function drives this
coeftest(f3, vcov = vcovHC(f3, type = 'HC0'))

#Multicollinearity
### Fit a model with all variables ----
 f1all = lm(
  cnt  ~
```

```
    yr +
    dteday +
    atemp +
    mnth +
    weekday +
    temp +
    hum +
    windspeed +
    holidayrecode +
    workingdayrecode +
    weatherrecode +
    seasonrecode,
  data = bikesharenew_train_standardized
)

summary(f1all)

plot(f1all)

vif(f1all)

vif(f2)

vif(f3)
```

```
### fit two models with x2 or x3, but not both ---
f_nmc_1 = lm(
  registered ~
    yr +
    mnth +
    weekday +
    temp +
    holidayrecode +
    weatherrecode +
    seasonrecode +
    I(temp ^ 2) +
    I(windspeed ^ 3),
  data = bikesharenew_train_standardized
)

summary(f_nmc_1)

plot(f_nmc_1)

vif(f_nmc_1)



f_nmc_2 = lm(
  registered ~
    yr +
    mnth +
```

```
    weekday +
    temp +
    holidayrecode +
    weatherrecode +
    seasonrecode +
    I(temp ^ 2) +
    I(cnt ^ 3),
  data = bikesharenew_train_standardized
)

summary(f_nmc_2)

plot(f_nmc_2)

vif(f_nmc_2)



# Non-Linearity Test
# Let's do diagnostic plots
# Notice the relationship between error and fitted values
# A pattern! That's bad. It means your errors aren't distributed N(0, sd)

plot(f3)

shapiro.test(f3$residuals)
```

```
# Let's do some feature engineering on the outcome to see if it helps
f_nonlinear = f3 = lm(
  casual ~
    yr +
    dteday +
    mnth +
    weekday +
    temp +
    holidayrecode +
    workingdayrecode +
    weatherrecode +
    seasonrecode +
    I(temp ^ 2) +
    I(hum ^2) +
    I(windspeed ^ 2) +
    I(atemp ^2),
  data = bikesharenew_train_standardized
)

summary(f_nonlinear)

plot(f_nonlinear)

## Influential Observation
```

```
# create an influential observation
bikesharenew$temp[110] <- 80

summary(bikesharewnew)

pairs(bikesharewnew)

### fit a model with the influential point ----
f_infl = lm(
  casual ~
    yr +
    dteday +
    mnth +
    weekday +
    temp +
    holidayrecode +
    workingdayrecode +
    weatherrecode +
    seasonrecode +
    I(temp ^ 2) +
    I(hum ^2) +
    I(windspeed ^ 2),
  data = bikesharenew_train_standardized
)

summary(f_infl)
```

```
plot(f_infl)



### fit a model with the influential point removed ----
f_non_infl = lm(
  casual ~
    yr +
    dteday +
    mnth +
    weekday +
    temp +
    holidayrecode +
    workingdayrecode +
    weatherrecode +
    seasonrecode +
    I(temp ^ 2) +
    I(hum ^2) +
    I(windspeed ^ 2),
  data = bikesharenew_train_standardized[-100, ]
)

summary(f_non_infl)

plot(f_non_infl)
```

```
### compare the difference in regression lines ----
bikesharenew_train_standardized %>%
  ggplot(aes(x = mnth, y = casual)) +
  geom_point(alpha = 0.5) +
  geom_line(aes(y = f_infl$fitted.values), color = "red") +
  geom_line(aes(y = c(f_non_infl$fitted.values, NA)), color = "blue", lty = 2)



## Autocorrelation


### Fit a model with autocorrelation ----
f_ac = lm(
  casual ~
    yr +
    dteday +
    mnth +
    weekday +
    temp +
    holidayrecode +
    workingdayrecode +
    weatherrecode +
    seasonrecode +
    I(temp ^ 2) +
    I(hum ^2) +
    I(windspeed ^ 2),
```

```
  data = bikesharenew_train_standardized)

summary(f_ac)

plot(f_ac)

### Test for autocorrelation ----
dwtest(f_ac)



### Correct for autocorrelation by adding lags of y ----
# Note: we don't add time as an explicit variable.
# Sometimes you can add time variables, but only for cyclic variables like
# week, month, season, etc.
# if you add a time variable for every day, for example, you'll get a perfect fit
# in training then fail in production.
# or you will run out of degrees of freedom
bikesharenew_train_standardized =
  bikesharenew_train_standardized |>
  arrange(mnth) |>
  mutate(
    lag_y = lag(casual)
  )



f_nac <- lm(
```

```
  casual ~
    yr +
    dteday +
    mnth +
    weekday +
    temp +
    holidayrecode +
    workingdayrecode +
    weatherrecode +
    seasonrecode +
    I(temp ^ 2) +
    I(hum ^2) +
    I(windspeed ^ 2) +
    lag_y,
  data = bikesharenew_train_standardized)

summary(f_nac)

plot(f_nac)

dwtest(f_nac)
```