WEATHER OR TIME?

Understanding What Drives Capital Bikeshare Usage

ABSTRACT

We analyze hourly Capital Bikeshare demand (2011–2012) to identify which variables most influence usage. Using a linear regression (for per-unit effect sizes) and a tuned Random Forest classifier (for non-linear importance).

Thanusha Balasubramanian

Data 601 – L02 – Fall 2025

Table of Contents

Ask a Question

Q1: What is the Question?

Which variables have the greatest impact on hourly Capital Bikeshare usage (cnt)? We wish to find the top 5 features with greatest magnitude of positive or negative impact on hourly bikeshare usage. All features included in the dataset are described in the "Q2: Data & Quality Discussion" section of the report.

The initial assumption, without reviewing the data, is that weather condition (in our dataset the categories are "Clear, Mist, Light Snow, Heavy Rain") or season (Winter, Spring, Summer, Fall), or Temperature (in degrees Celsius) may have the greatest impact on bike share usage as users typically do not want to bike in poor weather condition, winter-time, or cold weather.

If we can identify which factors have the largest impact on bike share usage, it be incorporated into business decision making, so bike share companies may be able to optimize pricing to reflect user demand or optimize operations by having more bikes being charged during low usage conditions.

Q2: Data & Quality Discussion

Data Source & Scope

This project uses the Capital Bikeshare dataset from the UC Irvine Machine Learning Repository. The dataset was donated to the repository on Dec 19, 2013. The data was collected from the Capital Bikeshare system from 2011 to 2012. The data is collected on an hourly basis. As the dataset is for the Capital Bikeshare system, which is in the Washington DC greater metropolitan area, the data can not necessarily be representative of other Bike share companies operating in other regions. For example, Washington DC has a much milder winter than most cities in Canada.

Data Quality

The data comes from the UC Irvine Machine Learning Repository and has already been pre-cleaned and transformed for usage in regression modeling. Upon checking the data no N/A or null values were found. The data is provided as 2 csv files, a file which contains the information broken down on an hourly basis (and includes temperature data) and a file which is summarised daily. For the purpose of this project the hour.csv file will be used as it provides more information, and we wish to analyze if time of day has any impact on the number of bike shares rides.

The data came with categorical datapoints pre-cleaned via label encoding. Binary data from the dataset was in the format of 1 or 0 so when it was imported into a Colab notebook it has a python datatype of Integer. Lastly, Temp (temperature), Atemp (feel temperature), and Windspeed were pre-scaled or normalized. Review Appendix A: Table of "Raw" Features/Columns in Hour.csv dataset to view details of which columns had come pre-cleaned and how they were pre-cleaned by UC Irvine.

Data "Cleaning" for Initial EDA

Although having, the "raw" data from the repository be pre-cleaned is useful for some applications, we did not necessarily agree with the choices for label encoding used. Additionally, we wished to view use the unnormalized data for the initial exploratory data analysis, so we transformed the data, by undoing the cleaning/transformation already done by UC Irvine. Review "Section 2: Initial Data Cleaning for EDA" in the associated Google Colab notebook or Appendix B: Code for Initial Pre-Cleaning for EDA in this report to see the coding steps to undo the labeling and transformation done in the "Raw" data. The result is our "Cleaned" data which is held in the df_bike_share_clean dataframe. Review Appendix C for a table of the columns, cleaning performed, range of values and data types.

Q3: ML Problem Type and Model Choice

The goal of the analysis is to find which explanatory variables have the greatest impact on a target variable (count of total users in a given hour). As we have a target variable, and explanatory variables the question is a supervised machine learning problem. Because the target variable (count of users in a given hour) is a numeric variable the most ideal model is a linear regression. With a linead regression we will be able to get the magnitude of each variables impact (slope coefficients for the x variables), and we will also be able to tell whether each explanatory variable has a positive or negative relationship with the target variable (ie. Sign/direction of relationship).

As we identified in the initial EDA that there are hours of the day where usage peaked. And hours with very low usage (ie. Night time), we will additionally use a classification model. We will split the target variable of Count of users in a given hour into 3 categories of usage, Low, Medium and High and perform a random forest model. We are choosing to use a random forest model as it will allow us to identify which features have the greatest impact, with a higher degree of accuracy than a decision tree, as the random forest incorporate multiple permutations of decision trees.

Data Understanding Exploration & Visualization

Summary Statistics & Interpretation

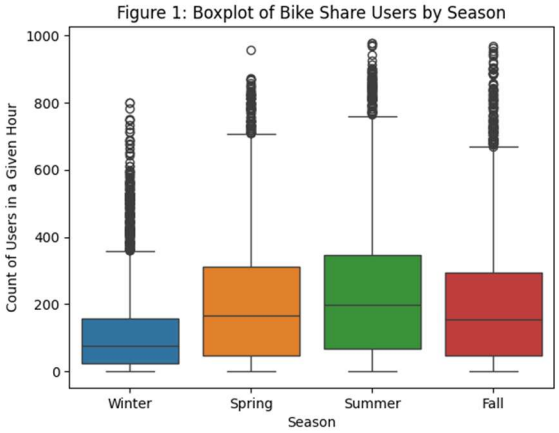
Firstly, to explore we reviewed the summary statistics for the df_bike_share_clean dataframe. Review Appendix D for the table of summary statistics. Some key insights are as follows:

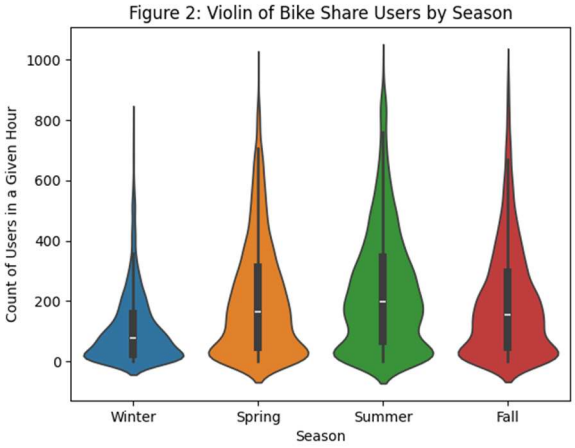
- Date_Cleaned - The dates in the dataset range from the minimum date of Jan 1, 2011 to the maximum date of Dec 31, 2012.
- Temp_Celcius - The minimum temperature was -16 degrees Celsius, and the maximum temperature was 39 degrees Celsius. The mean temperature is 15.36 degrees Celsius and the median temp is 15.5 degrees Celsius. This is inline with expectations as the dataset is from Capital Bikeshare which is based in Washington DC. The standard deviation is 9.05 degrees Celsius.
- The Feel_Temperature shows a similar mean, median and minimum but the max feel temperature is larger (likely due to the humidity).
- The Windspeed had a min value of 0 (indicating no wind) and a max of 56.996 km/h. The median and the mean are close together indicating the data is roughly symmetric
- The max humidity is 1.0 (100%) and the minimum humidity is 0. The mean humidity is close to the median humidity indicating the data is likely symmetric
- The median and mean bike share users who are registered is higher than the casual users. The standard deviation is also larger this indicates that a larger proportion of total users are registered users as opposed to casual users. The user counts are also provided on an hourly basis which could mean that the users with registered accounts are more likely to use the bike share more frequently.

Exploratory Visualization & Interpretation

Firstly, as our initial assumption that perhaps, season and weather condition are important variables which impact the rideshare count of users we decided to visualize these relationships. The boxplot and violin plots of Count by Season were chosen as we wanted to visualize what the distribution of the bike share count while categorizing the count by season. This also has an esthetic component as there are less than 5 categories so the differentiation in color is easy to see. Additionally, the same scale (y-axis) is used for all seasons, and the seasons are plotted side by side to prevent exaggeration of small

differences. (ie. if the differences were marginal, they visual representation would show that). Figure 1 (below) is side-by-side boxplots of the count of hourly users which are grouped by season.

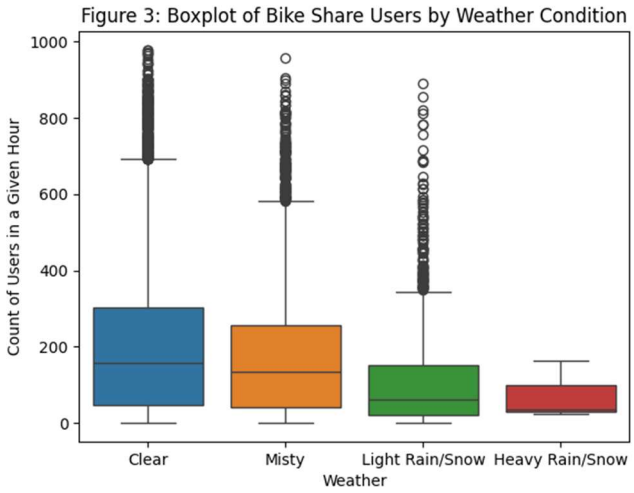




The median of the Winter box plot is lower than all the other seasons. Additionally, the interquartile range (middle 50%) of the data is smaller than all the IQR of all the other seasons. The IQR of Winter appears to be below the median of the other seasons. This indicates that there will be less bike share users in Winter than in other seasons. Summer has the largest median, IQR and upper whisker, which indicates that there is a greater variation in the count of users in summer as well as an indication that there are more users in summer. The draw back of the plot is that it does not indicate the detailed shape of the distribution, so we additionally created a violin plot (Figure 2) of the same data.

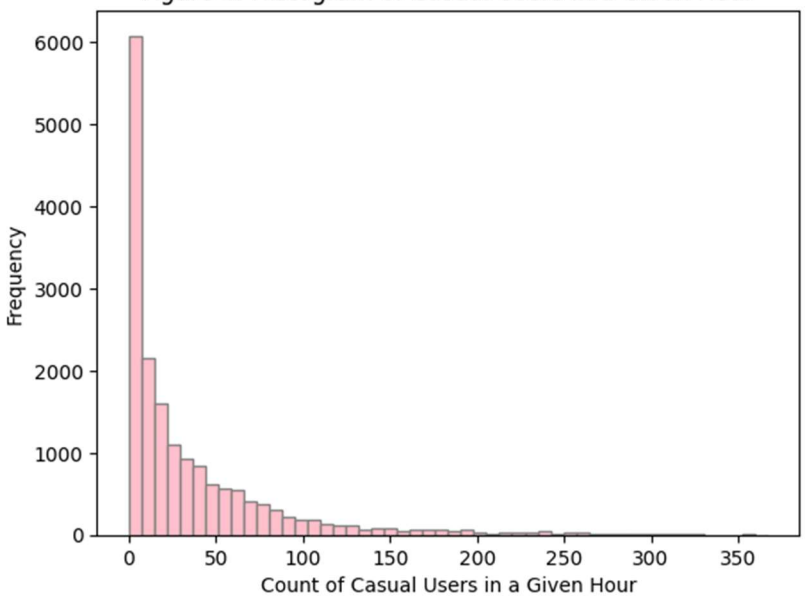
The violin plot of the bike share users by season indicates that the all of the distributions have a mode close to zero. However, summer seems to be bi-modal (ie. it has a secondary mode close to 200). This indicates that a large number of instances in all seasons have close to zero users (ie. perhaps people are unlikely to use bike shares at night-time so the counts for night-time hours are frequently low).

To assess whether there is a difference in user count based on weather condition we explored the relationship with a boxplot (Figure 3.) Each weather condition is plotted on the same y-axis as an ethical consideration to prevent the exaggeration of small differences. The box plot of users by Weather condition indicates that the user count has the highest median when the weather is clear. The IQR of Light Snow and Heavy Rain is below the median of clear weather.



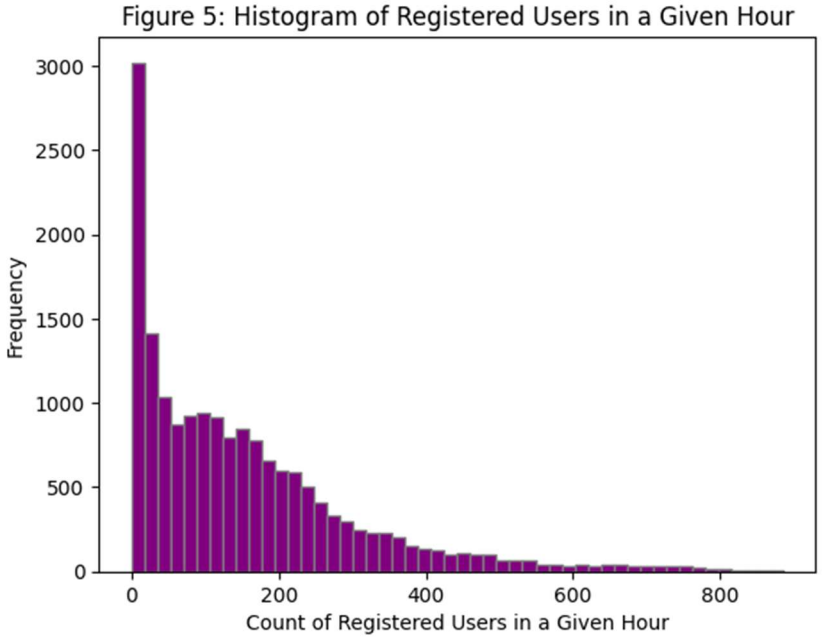
Next, in order to view the distribution of casual users a histogram was created (Figure 4). The histogram of casual users shows a right tail skew which indicates that lower values are more likely. Additionally the bin closest to zero has the highest frequency indicating in most hours the number of casual users is close to zero.

Figure 4: Histogram of Casual Users in a Given Hour



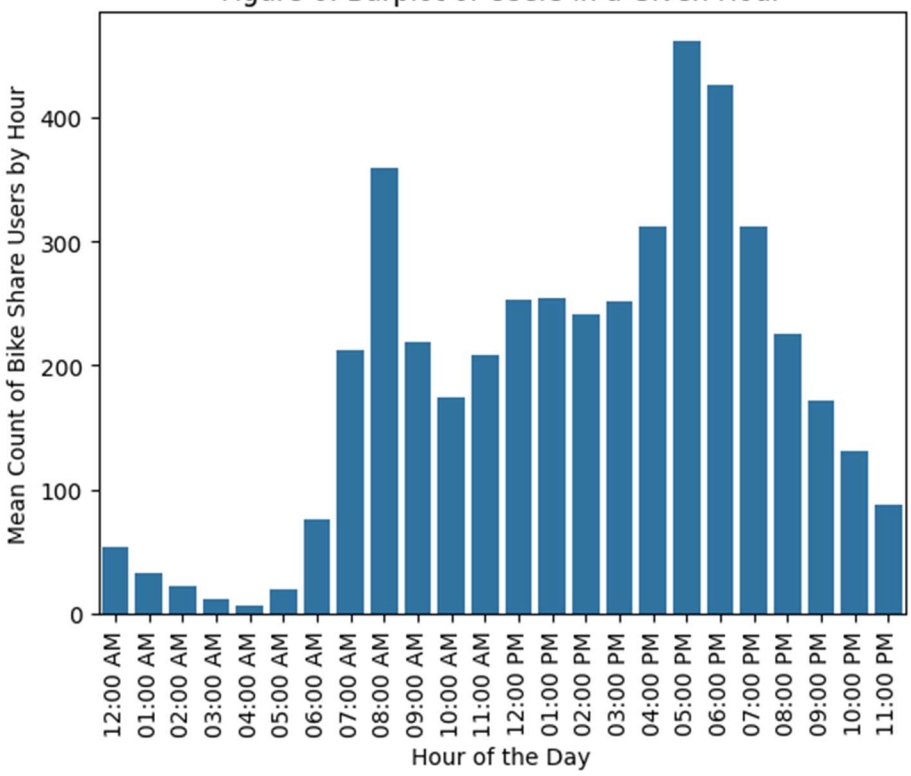
Additionally, the histogram of Register Users was created (Figure 5). The histogram of registered users is also skewed right; however, it appears to be bi-modal. This is likely since during most hours very few people are using the bike share, however there may be a spike in users during commute hours.

Figure 5: Histogram of Registered Users in a Given Hour



As the distributions of both Registered and Casual users (Figure 5, and Figure 4 respectively) showed a large frequency of having very few riders we chose to further explore whether time of day would affect usage which is why the bar plot of users by time of day was created (Figure 6). There is an ethical component to this decision as if one were to not use time of day, they may conclude that bike shares are largely unneeded and may curtail usage. The bar plot of users in a given hour is multi-modal. Additionally, the busiest time for bike share usage is at 4:00 PM and 5:00 PM. The second busiest time for bike hare usage is 8:00 AM. This indicates that people may use the bikeshare to commute to/from work and/or school.

Figure 6: Barplot of Users in a Given Hour



Correlation Analysis

As we intend to uncover which variables have the greatest impact on hourly bike share usage an important step in the initial data exploration is creating a correlation matrix. To better visualize the correlations, we are using a heatmap of the correlation matrix (Figure 7).

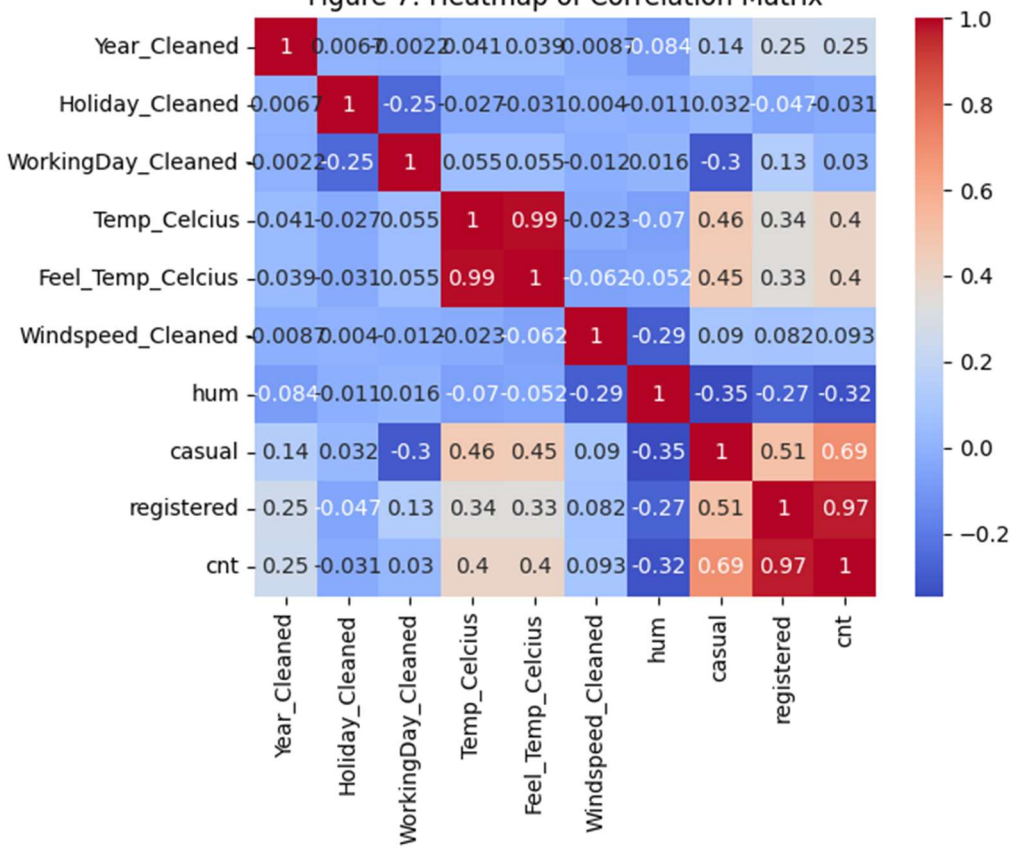
Based on the Coorelation matrix and heatmap there is only 4 variables which have a high correlation to each other. The Temp_Celsius and Feel_Temp have a correlation coefficient of 0.987672 which indicates they have a high degree of correlation.However, due to the nature of the data we would expect the two variables to be highly correlated as the temperature it feels like is dependent on the temperature in Celsius and the wind chill and humidity.

Registered users feature also has a high correlation coefficient with Count of users feature, however it is not appropriate to classify that as a correlation as count is a function of adding together registered and casual users.

Additionally, there are week negative correlations between humidity and count of users (which is similar for casual users and registered users). This indicates that users may be unlikely to use a bike share when it is very humid. There is a moderate positive relationship (corr coefficient = 0.46) between temperature in Celsius and the number of casual users in each hour. This indicates users may be more likely to use a bike share in warmer temperatures.

The seasonality is not included in the correlation matrix as the season is a categorical variable. However, it may be a confounding factor with temperature as there is highest ridership in warm summer months.

Figure 7: Heatmap of Correlation Matrix



Data Wrangling and Preprocessing for Modeling

Label and One Hot Encoding

The data provided from the UC Irvine repository came pre-processed with label encoding for Season, Month, Day of Week and Weather. We undid the encoding to create the “df_bike_share_clean” data frame. We did this to explore our data without the processing.

As season, month and day of week have an inherent order, we will use the label encoding again. However, we will start ours at 1 instead of 0 (which was the pre-processing performed by UC Irvine). We are choosing to stat at 1 instead of zero as zero implies, there is an absence of season (or month or day of the week) and a null state which is not true in our dataset.

For Weather we will use one-hot encoding instead as there is not necessarily an ordinal relationship in weather conditions. The encoding is necessary as the model needs to have numerical or Boolean data. To see the code used for encoding please refer to Appendix E or the Colab notebook section 4.

Change Hour from a String to a Numerical Variable

The Hour variable was previously transformed from a numeric (integer) into the string format of hh:mm AM/PM. This string is not an appropriate format we will need numerical or Boolean data for our models, so we are transforming hour back into a numeric variable (Refer to Appendix F) or Colab notebook section 4 for the code.

Build & Test the Model(s)

Model Building 1: Linear Regression

The first model we are using is a linear regression model. The training vs test data split is 70 (training) / 30 (test). We are using a conventional split. The target variable is total users in each hour (cnt). For the X variables we are dropping Date variable as the variable is every date from Jan 1, 2011, to Dec 31, 2012 and in the context of the data set using specific date does not necessarily work with out model. Additionally, as we are only interest in the total count of user in a given hour (and not interested in registered vs casual users) we will be dropping the registered and casual users' columns. Lastly, in building the model, after we have split the model into test and training data, we will be scaling the data so the model is not adversely affected by the magnitude of different variables (refer to Colab notebook, or Appendix G to see model building code).

The resulting model has an intercept of 191.02 users. This indicates that absent the effects of any of the other variables (ie. All other variables were set to 0) we would expect on average the user count in a given hour to be 191 users. To review the coefficients for each x variable please refer to Appendix G or the Colab notebook. The model coefficients (Appendix G) show the change in the dependent variable (Count of Total Rideshare Users in an hour) for a one-unit increase in the independent variable, holding other variables constant. Larger coefficients indicate that a category has a higher relative impact on number of users (ie. Hour seems to have the largest impact on number of users). The sign on the coefficient corresponds to the relationship between the X variable and the count of users. For example, there is an inverse relationship between Holiday and count of users indicating that there are fewer users on holidays.

Q4: Linear Regression Performance Evaluation

The linear regression model had a training data RMSE of 142.30 and a test data RMSE of 139.78. The training data has a higher RMSE than the test data RMSE which indicates the model is not overfit to the training data. The test RMSE is still relatively high at 139.78 considering the mean ridership in given hour is 189.46 and the median ridership is 142. As the RMSE is it suggests that the model has a limited ability to capture non-linear weather–ridership relationships.

Model Building 2: Random Forest

Creating a Categorical (class) Variable for Hourly Ridership

As the linear regression model had a high RMSE and was unable to capture the non-linear relationship between weather and ridership effectively, we will also be employing a random forest model. To utilize the Random Forest Model, we will have to transform the ridership hourly count into a class variable. We have

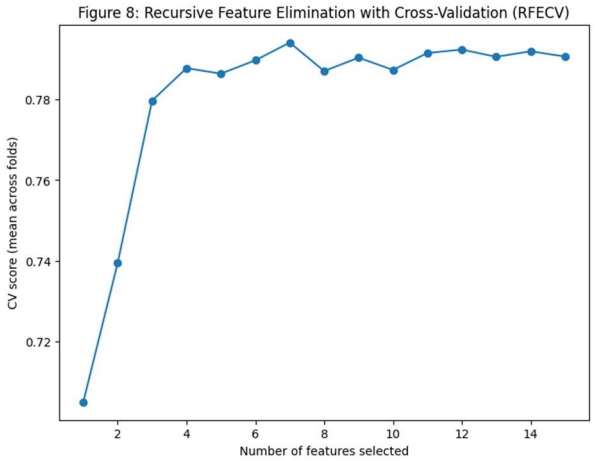
decided to classify the ridership as “Low,” “Medium” or “High” depending on the Interquartile range for the ridership count. Ridership has 25% quartile of 40 users and a 75% quartile of 281 users. Hourly user counts below 40 will be classified as “Low” usage. User counts between 40 and 281 (inclusive) will be classified as “Medium”. User counts above 281 users in a given hour will be classified as “High”. See Appendix I or the Colab Notebook to see the code for classifying usage.

Initial Random Forrest

We initially ran a random forest model without optimization. The data was split into X and Y variables. The Y was the Usage class. The x variables were all the same as the ones used in the linear regression (i.e. all variables except for Date, Registered User, and Casual Users.) We initially ran a random forest model with 1000 bootstraps, max features of 0.8, max samples of 0.8, max depth of 5 branches and a random state (seed value) of 2005 (refer to Appendix J).

Random Forrest Optimization

In order to optimize the random forest model for the number of features to select we used Recursive feature elimination with cross-validation (refer to Appendix K). We found that the optimal number of features to select for the model was seven as indicated in Figure 8 below.



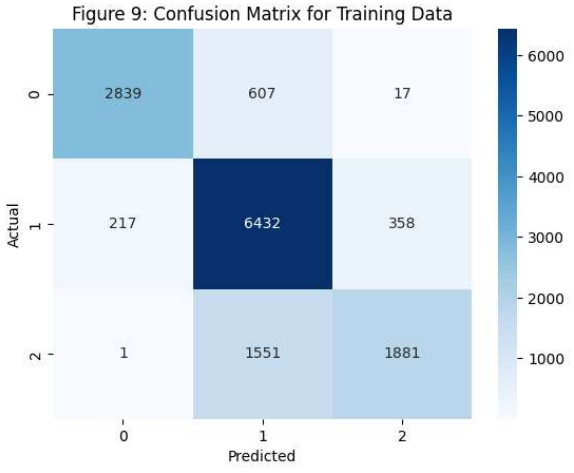
The optimal features are:

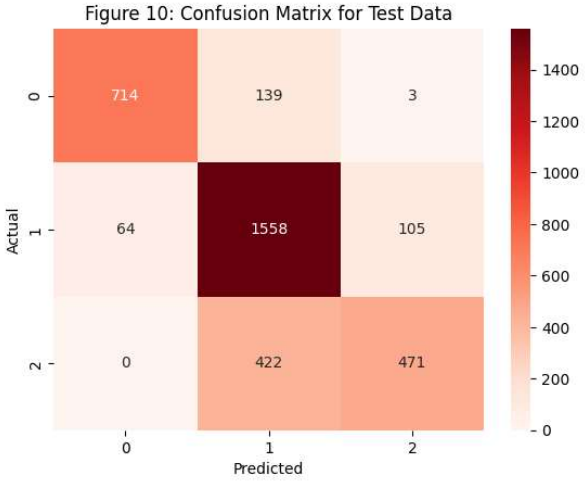
1. Year
2. Hour
3. Weekday
4. WorkingDay
5. Temp_Celcius
6. 'Feel_Temp_Celcius
7. Humidity

We re-ran the model on a subset of the data containing only the optimal features (refer to Appendix L).

Q4: Random Forest Performance Evaluation

In order to evaluate the random forest model the confusion matrix and accuracy, recall and precision scores were calculated on both the training dataset and the test dataset.





The measures of model performance for the training data set were:

- Accuracy: 0.80
- Precision: 0.81
- Recall: 0.80

The measures of model performance for the test data were:

- Accuracy: 0.79
- Precision: 0.80
- Recall: 0.79

As the accuracy on both the training and test data is very close, (within 1-2 percentage points) we say there appears to be minimal / none materially impactful overfitting on the model. The precision and recall are both lesser for the test data than the training data indicating the model isn't trading off on false positive vs false negatives unevenly.

Interpret Model Results

Model Interpretation 1: Linear Regression

Q5: Explanation of Model Feature Contributions

Based on the linear regression model utilized the features with the largest magnitude of impact are:

1. Hour (coefficient = 53.50) – the sign on the coefficient is positive indicating that for each one unit increase in Hour there is 53.50 average increase in riders.
2. Feel Temperature (coefficient = 44.54) – The feel temperature has a positive relationship with ridership in a given hour.
3. Year (coefficient = 40.71) – We expect an effect of an increase of 40.71 riders hourly for a one unit increase in the year.
4. Humidity (coefficient = - 37.60) – Humidity is the variable with the highest magnitude affect of ridership where the relationship between the variable and ridership is inverse (negative). This indicates that for a one unit increase in humidity hourly ridership decrease bs 37.60 users.
5. Season (coefficient = 21.89) – The season coefficient is positive. Within the context of our model the seasons were label encoded where Winter = 1, Spring = 2, Summer = 3, and Fall = 4.

To view a full listing of all the features ranked by magnitude of affect on ridership please view Appendix H. The table in Appendix H also notes whether the relationship between the feature and ridership is positive or negative.

Model Interpretation 2: Random Forrest

Q5: Explanation of Model Feature Contributions

In order to test which features had the greatest importance we used a permutation test, and graphed the results in the figure below:

Figure 11: Feature Importance

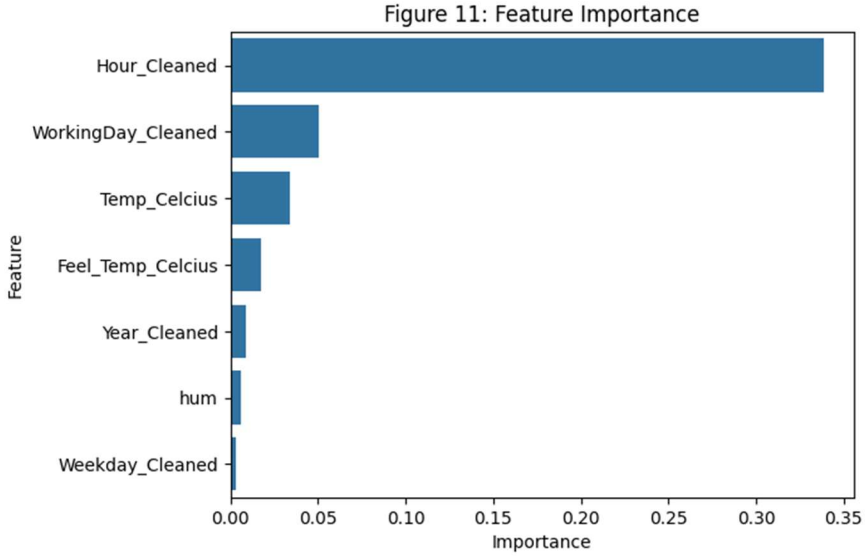


Figure 11 indicates that hour has the greatest effect on bike share usage. The next most important variable seems to be working day. And thirdly temperature in Celsius. This pattern aligns with our findings from the EDA the bike share usage seems to be at it's highest during morning and evening commute times. Additionally, the regression we performed also indicates a significant relationship between the hour of the day and the count of users. However, the regression performed is a linear regression which may not suit the data (especially when it comes to hour of day), as bike share usage appears to be highest during commute times and not at the hour of the day with the highest value (11:00 PM).

Answer the Question

Q6: Outcome and Next Steps

The Linear Regression model indicated that the most 5 most important feature variables that influence hourly bike share ridership are Hour, Feel Temperature, Year, Humidity and Season. The linear regression model had a test data RMSE of 139.78 which is relatively high (as discussed in Q4). Despite the model not being overfit, it was a poor predictor of hourly ridership. This indicates that there are non-linear relationships which can not be captured by the model. For example, weather and seasons are not necessarily linear.

The Random Forrest model indicated that the 5 most important feature for classifying hourly bike share usage are Hour, Working Day, Temperature, Feel Temperature and Year. The random forest had an accuracy of 0.79 indicting it is relatively good model. Additionally, the model was not overfit.

Both models identify hour and temperature as strong positive predictors of bike share usage, while humidity has a clear negative effect. In the regression model, a one-unit increase in ‘hour’ corresponds to roughly 53 additional riders, whereas humidity decreases ridership by about 38. The Random Forest confirms this pattern, with Hour having the highest importance (0.33), followed by Working Day and

Temperature. This indicates that ridership peaks during warm commuting hours and declines sharply with poor weather. The consistency across models reinforces that time-of-day and weather are the key drivers of demand.

As the random forest model had a better predictive capability, we will be using the results from the random forest model to answer the question:

Which variables have the greatest impact on hourly Capital Bikeshare usage (cnt)? We wish to find the top 5 features with greatest magnitude of positive or negative impact on hourly bikeshare usage.

The top 5 variables with greatest influence on hourly ridership are:

1. Hour
2. Working Day
3. Temperature
4. Feel Temperature
5. Year

Knowing that the variable of hour has the greatest influence on ridership is important in a business context. Capital Bikeshare could apply employ dynamic pricing (by applying a surcharge on the bike rentals during peal hours). The company could also improve operational efficiency by charging and re-distributing bikes during times of “Low” usage. The company could also market to users by incentivising off-peak rise with discounts or loyalty offers.

To improve on the random forest model with the limitations of the current dataset we can apply additional fine tuning of hyperparameters by using grid search or another type of search algorithm to find optimal hyperparameters. For the regression model a polynomial model or other transformation (beyond scaling data) may yield better results. A random forest model for regression may also produce higher accuracy in predicting bike share usage.

To improve on project by incorporating additional data, we might look to find bike share usage for different cities to see if the patterns found on the analysis of the Capital Bikeshare dataset holds in cities other than Washington DC where the data in the specific dataset was collected from. Additionally, if information on bike share routes was collected (ie. Location data, ride length etc…) we would be able to potentially find other insights and patterns on predicted usage.

References

Fanaee-T, H. (2013). Bike Sharing [Dataset]. UCI Machine Learning Repository. https://doi.org/10.24432/C5W894.

Appendices

Appendix A: Table of “Raw” Features/Columns in Hour.csv dataset

Column in Data	Description of Column	Type of Data from Repository	Python Data Type	Missing Values
instant	Record index (row id)	Integer	Integer	No
dteday	Date (YYYY-MM-DD)	Date	String	No
season	Season (transformed using label encoding): 1=winter, 2=spring, 3=summer, 4=fall	Categorical	Integer	No
yr	Year code: 0=2011, 1=2012	Categorical	Integer	No
mnth	Month number (1 to 12)	Categorical	Integer	No
hr	Hour of day (0 to 23)	Categorical	Integer	No
weekday	Day of week (0=Sunday, 6=Saturday)	Categorical	Integer	No
holiday	Holiday flag (1 if holiday)	Binary	Integer	No
workingday	1 if day is neither weekend nor holiday	Binary	Integer	No
weathersit	Weather situation: 1=Clear; 2=Mist; 3=Light Snow or Rain; 4=Heavy Rain or Snow	Categorical	Integer	No
temp	Normalized temperature in Celsius: Scaled: (t - (-8)) / (39 - (-8)); only hourly scale	Real	Float	No
atemp	Normalized 'feels like' temperature in Celsius: Scaled: (t - (-16)) / (50 - (-16)); only hourly scale	Real	Float	No
hum	Normalized humidity: Divided by 100 (max)	Real	Float	No
windspeed	Normalized wind speed: Divided by 67 (max)	Real	Float	No
casual	Count of casual users in that hour. A "Casual" user is a user who does not have an annual membership.	Integer	Integer	No
registered	Count of registered users in that hour. A "Registered" user is a user who has an annual membership.	Integer	Integer	No
cnt	Total rentals in the hour (casual users + registered users)	Integer	Integer	No

Appendix B: Code for Initial Pre-Cleaning for EDA

```
## Clean the Date
from datetime import date
df_bike_share["Date_Cleaned"] = pd.to_datetime(df_bike_share["dteday"])

## Cleam the Season
df_bike_share["Season_Cleaned"] = df_bike_share["season"].replace([1,2,3,4],
["Winter","Spring","Summer","Fall"])

## Clean the Year
df_bike_share["Year_Cleaned"] = df_bike_share["yr"].replace([0,1],[2011,2012])

## Clean the Month
df_bike_share["Month_Cleaned"] = df_bike_share["mnth"].replace([1,2,3,4,5,6,7,8,
9,10,11,12],["January","February","March","April","May","June","July","August",
"September","October","November","December"])

## Clean the Hour
df_bike_share["Hour_Cleaned"] = pd.to_datetime(df_bike_share["hr"],
format='%H').dt.strftime('%I:%M %p')

 ## Clean the Holiday Column
 df_bike_share["Holiday_Cleaned"] = df_bike_share["holiday"].astype(bool)

 ## Clean the Weekday Column
 df_bike_share["Weekday_Cleaned"] = df_bike_share["weekday"].replace([0,1,2,3,4,
 5,6],["Sunday","Monday","Tuesday","Wednesday","Thursday","Friday","Saturday"])

 ## Clean the Working Day Column
 df_bike_share["WorkingDay_Cleaned"] = df_bike_share["workingday"].astype(bool)

 ## Clean the Weather Situation column
 df_bike_share["Weather_Cleaned"] = df_bike_share["weathersit"].replace([1,2,3,
 4],["Clear","Misty","Light Rain/Snow","Heavy Rain/Snow"])

 ## Transform the Temperature back to celcius
 t_min = -8
 t_max = 39
 df_bike_share["Temp_Celcius"] = (df_bike_share["temp"]*(t_max - t_min)) + t_min


## Transform the Feeling Temperature back to celcius
t_min = -16
t_max = 50
df_bike_share["Feel_Temp_Celcius"] = (df_bike_share["atemp"]*(t_max - t_min)) +
t_min

## Transform Windspeed back
df_bike_share["Windspeed_Cleaned"] = df_bike_share["windspeed"]*67

## select which columns we want to subset into a new data frame
selected_columns = ["Date_Cleaned", "Season_Cleaned", "Year_Cleaned",
"Month_Cleaned","Hour_Cleaned","Holiday_Cleaned","Weekday_Cleaned",
"WorkingDay_Cleaned","Weather_Cleaned","Temp_Celcius","Feel_Temp_Celcius",
"Windspeed_Cleaned","hum","casual","registered","cnt"]
## Create a new datafram with cleaned columns
df_bike_share_clean = df_bike_share[selected_columns]
```

Appendix C: Table of Cleaned Data (df_bike_share_clean dataframe)

Column	Cleaning Description	Variable Type	Python Data Type	Range
Date_Cleaned	Transformed into datetype for Python	String	datetime (string-friendly)	Jan 1, 2011 to Dec 31, 2012
Season_Cleaned	Undid Label Encoding to have String Categories	Categorical	categorical (string)	Winter/Spring/Summer/Fall
Year_Cleaned	Changed from 0 and 1 to 2011 and 2012	Categorical	integer	2011 or 2012
Month_Cleaned	Changed from Integer Category to Month names	Categorical	categorical	"January" …. "December"
Hour_Cleaned	Changed to a string formatted as HH:mm AM/PM	Categorical	categorical	12:00 AM to 11:59 PM
Holiday_Cleaned	Changed 0 or 1 to True or False (bool)	Boolean	Boolean	True or False
Weekday_Cleaned	Undid Label Encoding to have String Categories	Categorical	String	Sunday to Saturday
WorkingDay_Cleaned	Changed 0 or 1 to True or False (bool)	Boolean	Boolean	True or False
Weather_Cleaned	Undid Label Encoding to have String Categories	Categorical	String	"Clear","Misty or","Light Rain or Snow","Heavy Rain or Snow"
Temp_Celcius	Undid the normalization of the temperature.	Numeric	Float	-7.06 degrees celcius to 39 degrees celcius
Feel_Temp_Celcius	Undid the normalization of the a_temperature.	Numeric	Float	-16 degrees celcius to 50 degrees celcius
Windspeed_Cleaned	Undid the scaling. (multiplied by 67)	Numeric	Float	0 to 56.99 km/h
hum	Unchanged	Numeric	Float	0.00 to 1.00
casual	Unchanged	Numeric	Integer	0 to 367 users
registered	Unchanged	Numeric	Integer	0 to 886 users
cnt	Unchanged	Numeric	Integer	1 to 977 users

Appendix D: Summary Statistics (df_bike_share_clean dataframe)

index	Date_Cleaned	Year_ Cleaned	Temp_ Celcius	Feel_Temp _Celcius	Windspeed_ Cleaned	hum	casual	registered	cnt
count	17379	17379	17379	17379	17379	17379	17379	17379	17379
mean	2012-01-02 4:08:35 AM	2011.50256	15.3583969	15.401157	12.7365396	0.62722884	35.6762184	153.786869	189.463088
min	2011-01-01 12:00:00 AM	2011	-7.06	-16	0	0	0	0	1
25%	2011-07-04 12:00:00 AM	2011	7.98	5.9978	7.0015	0.48	4	34	40
50%	2012-01-02 12:00:00 AM	2012	15.5	15.9968	12.998	0.63	17	115	142
75%	2012-07-02 12:00:00 AM	2012	23.02	24.9992	16.9979	0.78	48	220	281
max	2012-12-31 12:00:00 AM	2012	39	50	56.9969	1	367	886	977
std	NaN	0.50000783	9.0501377	11.342114	8.19679531	0.19292983	49.3050304	151.357286	181.387599

Appendix E: One Hot Encoding and Label Encoding Pre-Processing Code

```
## label encoding for season where Winer, Spring, Fall, Summer Fall is 1, 2, 3,
4
df_bike_share_clean['Season_Cleaned'] = df_bike_share_clean['Season_Cleaned'].
replace(['Winter','Spring','Summer','Fall'],[1,2,3,4])

## Label encoding for day of week where 1 = Sunday
df_bike_share_clean['Weekday_Cleaned'] = df_bike_share_clean['Weekday_Cleaned'].
replace(['Sunday','Monday','Tuesday','Wednesday','Thursday','Friday',
'Saturday'],[1,2,3,4,5,6,7])

## Label encoding for Month
df_bike_share_clean['Month_Cleaned'] = df_bike_share_clean['Month_Cleaned'].
replace(['January','February','March','April','May','June','July','August',
'September','October','November','December'],[1,2,3,4,5,6,7,8,9,10,11,12])

## On Hot encoding for weather
df_bike_share_clean = pd.get_dummies(df_bike_share_clean, columns=
['Weather_Cleaned'])
```

Appendix F: Code to Change Hour Back to Numerical Variable

```
## We need to transform the hour back into a numerical variable
df_bike_share_clean['Hour_Cleaned'] = pd.to_datetime(df_bike_share_clean
['Hour_Cleaned'], format='%I:%M %p').dt.hour
df_bike_share_clean.head(5)
```

Appendix G: Code to Create the Linear Regression Model and Test It

```
### Let's split the data into training and test
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

## we will drop Date_Cleaned (as it is a datetimeformat), registered and casual
(as we are only looking at the total users)
X = df_bike_share_clean.drop(['cnt','Date_Cleaned','registered', 'casual'],
axis=1)
y = df_bike_share_clean['cnt']

## Randomly split the data using a seed value of 1387
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=1387)

##
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

## Let's fit the linear regression model
from sklearn.linear_model import LinearRegression
reg = LinearRegression()
reg.fit(X_train_scaled, y_train)
```

```
LinearRegression
LinearRegression()
```

```
## Let's print the model intercept
print(reg.intercept_)
```

```
191.026387176332
```

```
## Let's print the model X variables and their respective linear regression
coeficients
pd.DataFrame(reg.coef_, X.columns, columns=['Coefficient'])
```

```
## Let's test the model
y_pred_test = reg.predict(X_test_scaled)
y_pred_test
y_pred_train = reg.predict(X_train_scaled)
y_pred_train
```

```
##Let's get the RMSE for Training Data
from sklearn.metrics import mean_squared_error
mse_train = mean_squared_error(y_train, y_pred_train)
rmse_train = np.sqrt(mse_train)
rmse_train
```

```
np.float64(142.30309815865888)
```

```
##Let's get the RMSE for Test Data
from sklearn.metrics import mean_squared_error
mse_test = mean_squared_error(y_test, y_pred_test)
rmse_test = np.sqrt(mse_test)
rmse_test
```

```
np.float64(139.78435870570055)
```

Appendix H: Linear Regression Model Coefficients Table

Variable	Coefficient	Rank of Importance (Magnitude)	Relationship (Postive = Direct Relationship, Negative = Inverse Relationship)
Hour_Cleaned	53.50257906	1	Positive
Feel_Temp_Celcius	44.53735421	2	Positive
Year_Cleaned	40.70890844	3	Positive
hum	-37.60209642	4	Negative
Season_Cleaned	21.89011925	5	Positive
Temp_Celcius	11.76487935	6	Positive
Windspeed_Cleaned	6.653328321	7	Positive
Weather_Cleaned_Light Rain/Snow	-6.361142682	8	Negative
Holiday_Cleaned	-5.523858469	9	Negative
Weather_Cleaned_Misty	4.400312531	10	Positive
Weekday_Cleaned	3.665947518	11	Positive
WorkingDay_Cleaned	0.78898782	12	Positive
Weather_Cleaned_Heavy Rain/Snow	0.45852799	13	Positive
Weather_Cleaned_Clear	-0.43286152	14	Negative
Month_Cleaned	-0.231909597	15	Negative

Appendix I: Usage Classification Calculation

Get the IQR for classification

```
# Get the quartiles for cnt
q1 = df_bike_share_clean['cnt'].quantile(0.25)
q3 = df_bike_share_clean['cnt'].quantile(0.75)
print(q1, q3)
```

```
40.0 281.0
```

Add the classifcation for usage.

```
### add the encoding for outcome based on cnt
## Class 1 = low useage
## Class 2 = medium usage
## Class 3 = high usage
df_bike_share_clean['usage_class'] = np.where(df_bike_share_clean['cnt'] < q1,
1, np.where(df_bike_share_clean['cnt'] > q3, 3, 2))
```

Appendix J: Code for Initial Random Forrest

```
## Let's split the data into x and y
x_rf = df_bike_share_clean.drop(['cnt','Date_Cleaned','registered', 'casual',
'usage_class'], axis=1)
y_rf = df_bike_share_clean['usage_class']

## Randomly split the data using a seed
x_train_rf, x_test_rf, y_train_rf, y_test_rf = train_test_split(x_rf, y_rf,
test_size=0.2, random_state=2005)

## Build the random forrest model
from sklearn.ensemble import RandomForestClassifier
forrest_classifier = RandomForestClassifier(n_estimators=100, bootstrap=True,
max_features=0.8, max_samples=0.8, max_depth=5,random_state=2005)

## Fit the model to our data
forrest_classifier.fit(x_train_rf, y_train_rf)
```

```
RandomForestClassifier(max_depth=5, max_features=0.8, max_samples=0.8,
                       random_state=2005)
```

Appendix K: Recursive Feature Elimination with Cross Validation Code

```
from sklearn.feature_selection import RFECV

#min number of variables/features
min_features_to_select = 1

#build the feature selection algorithm
rfecv = RFECV(estimator= forrest_classifier, step=1, cv=5,
min_features_to_select=min_features_to_select)

#fit the algorithm to the data
rfecv.fit(x_train_rf, y_train_rf)

print("Optimal number of features : %d" % rfecv.n_features_)
```

```
Optimal number of features : 7
```

```
selected_features = X_train.columns[rfecv.support_]
print("Selected features:", list(selected_features))
```

Selected features: ['Year_Cleaned', 'Hour_Cleaned', 'Weekday_Cleaned', 'WorkingDay_Cleaned', 'Temp_Celcius', 'Feel_Temp_Celcius', 'hum']

Appendix L: Optimized Model Code

```
## Let's subset our dat for the optimal features
x_rf_opt = df_bike_share_clean[selected_features]
y_rf_opt = df_bike_share_clean['usage_class']

## Split the data
x_train_rf_opt, x_test_rf_opt, y_train_rf_opt, y_test_rf_opt = train_test_split
(x_rf_opt, y_rf_opt, test_size=0.2, random_state=2005)

## Build the random forrest model
from sklearn.ensemble import RandomForestClassifier
forrest_class = RandomForestClassifier(n_estimators=100, bootstrap=True,
max_features=0.8, max_samples=0.8, max_depth=5,random_state=2005)

## Fit the model to our data
forrest_class.fit(x_train_rf_opt, y_train_rf_opt)
```

```
RandomForestClassifier
RandomForestClassifier(max_depth=5, max_features=0.8, max_samples=0.8,
                       random_state=2005)
```

```
## predict the results of the training data
y_pred_rf_opt_train = forrest_class.predict(x_train_rf_opt)
y_pred_rf_opt_train
```

```
array([2, 3, 2, ..., 1, 2, 2])
```