Bike Sharing in Washington D.C

Sidhant Thakur, DePaul University, School of Computing, sthakur5@depaul.edu

Parth Babubhai Patel, DePaul University, School of Computing, ppate270@depaul.edu

Saransh Thakur, DePaul University, School of Computing, sthakur6@depaul.edu

ABSTRACT

This project specifically focuses on using machine learning techniques to predict the demand, for bikes in Washington. The main goal is to address the challenges related to optimizing bike placement. By examining factors such as time, temperature and windspeed the study aims to improve the accuracy of predictions. The project utilizes the Washington bike demand dataset along with parameters in order to increase profits for companies like Zagster and City Bikes. Through visualization and assessments of machine learning models the objective is to optimize bike availability leading to revenue growth, for bike lending companies and encouraging efficient transportation alternatives.

Keywords; Bike demand prediction, machine learning, data science, feature engineering, optimization, Washington dataset, Zagster, City Bikes, profitability, user service.

INTRODUCTION

We have initiated a project that aims to tackle the challenges faced by bike sharing companies, in the United States such as Divvy, Zagster and Lime Bikes. Our collective objective is to optimize the distribution of bikes ensuring they are strategically positioned to meet varying demands at the time and place. This approach aims to strike a balance between customer satisfaction and profitability.

We are delving into this dataset because it holds the key to uncovering patterns of bike demand across regions and timeframes. The insights we gather will empower us to develop solutions that can enhance the user experience and contribute to the long-term success and sustainability of bike sharing programs.

In essence our mission is centered around determining the number of bikes required at locations and times streamlining operations for these companies. By doing we aim not to provide convenience for users but also help these businesses thrive in an ever-evolving transportation landscape. This collaborative effort strives towards creating a scenario, for riders and bike sharing companies.

Literature review:

In the paper titled "A Hybrid Machine Learning Model, for Predicting Demand of Bike Sharing System Using Internet of Things" by Tiantian Xu, Guangjie Han, Xingyue Qi Jiaxin Du, Chuan Lin and Lei Shu regression trees and self organizing maps (SOM) are utilized to estimate bicycle

demand. The authors achieve this by grouping the sample set into clusters and then constructing regression trees based on those clusters. This approach effectively incorporates algorithms and machine learning techniques [1].

In their study titled "Machine Learning Based Prediction of Bike Share Demand; Exploring the Role of Vehicle Accidents as a New Feature " Tae You Kim, Min Jae Park, Jiho Shin and Sungwon Oh highlight the use of machine learning and innovative features to improve bike sharing demand forecasts. They specifically consider factors such as data on traffic accidents. The study emphasizes the importance of feature engineering and exploratory data analysis in identifying patterns within the dataset. These findings underscore how predicting bike sharing demand contributes to urban transportation [2].

In their work titled "Predicting Demand, for Bike Sharing Using Machine Learning Methods " Chang Gao and Young Chen employ machine learning techniques to forecast bike sharing demand.

The research paper combines weather conditions, pollution levels, COVID 19 cases and socioeconomic characteristics to estimate the demand, for bike rentals in Seoul using machine learning models like support vector machines, random forests, k nearest neighbors and linear regression. The findings emphasize the influence of weather and COVID 19 related factors on bike rental demand. Although individual socioeconomic factors had impact incorporating them into the models significantly improved overall prediction accuracy and provided valuable insights into the dynamics of urban bike sharing demand.

In a study titled "Using data mining techniques for predicting bike sharing demand in a city " authors Sathishkumar V E, Jangwoo Park and Yongyun Cho employ regression algorithms to analyze the number of bikes rented in Seoul and examine how various factors affect forecasting. They demonstrate the vital role that time and weather information play in enhancing predictions. However it is important to note that since this study focuses on one region while disregarding relevant variables its effectiveness may vary depending on the specific context.

For an understanding of learning methods and their application to data analysis techniques, "The Elements of Statistical Learning”, by Hastie et al. (2009) serves as a foundational text providing invaluable insights into these principles.

The paper titled "Machine Learning Approaches, to Bike Sharing Systems" written by Albuquerque, Dias and Bacao discusses the findings of a research study conducted using the PRISMA method. By examining 35 papers published between 2015 and 2019 the authors shed light on the significance of classification techniques and prediction models in understanding user behaviors and identifying trends in bike sharing systems. Additionally the inclusion of analysis provides insights into the landscape that guides future research efforts in this evolving field.

In another study titled "Substitution effect or complementation effect for bicycle travel choice preference and other transportation availability; Evidence from US scale shared bicycle travel behavior data" authored by Zhaohua Wang, Yefei Sun, Yimeng Zeng Bo Wang, a different approach is taken. Of relying on a single choice model this study delves into travel patterns, within the Pronto! bicycle sharing system to reveal a complex multi choice travel model. Through an analysis of transportation modes interactions both replacement and supplementary impacts are identified. This offers perspectives on how commuters employ travel tactics and challenges conventional transportation models by highlighting the diversity of travel habits.

The paper titled "Bike Sharing Demand Prediction based on Knowledge Sharing, across Modes; A Graph based Deep Learning Approach" by Yuebing Liang, Guan Huang and Zhan Zhao

explores the use of graph-based learning and multi relational graph neural networks in the B MRGNN technique. This technique aims to integrate types of transportation data to predict bike sharing demand accurately. The study conducted tests using data from New York City. Found that the B MRGNN technique outperforms previous approaches emphasizing the importance of multimodal data for precise urban transportation forecasting.

In another paper titled " prescriptive performance of bike sharing demand forecasts for inventory management" by Daniele Gammelli, Yihua Wang, Dennis Prak, Filipe Rodrigues, Stefan Minner and Francisco Camara Pereira it is suggested that further research should be conducted to examine how inventory decisions in bike sharing systems relate to demand prediction. The study identifies a discrepancy between forecast accuracy and decision quality.

Furthermore, a study by Soheil Sohrabi, Rajesh Paleti, Lacramioara Balan and Mecit Cetin titled "Real time prediction of public bike sharing system demand using value count model" addresses technological challenges in predicting real time demand, in bike sharing systems. The researchers developed a Generalized Extreme Value count model which aids in trip planning optimization of bike redistribution efforts and hourly demand forecasting.

Furthermore, it explores the impact of policies by investigating the interplay, between the transportation system and bike sharing, as options to enhance mobility.

Data

The dataset contains 21 variables and 17379 records such as instant: Record index ,dteday: Date, season: Season (1: springer, 2: summer, 3:fall, 4:winter) , yr: Year (0: 2011, 1:2012) , mnth: Month (1 to 12) , hr: Hour (0 to 23) , holiday: weather day is holiday or not (extracted from Holiday Schedule) , weekday: Day of the week , workingday: If day is neither weekend nor holiday is 1, otherwise is 0. , weathersit: (extracted from Freemeteo) , 1: Clear, Few clouds, Partly cloudy, Partly cloudy , 2: Mist + Cloudy, Mist + Broken clouds, Mist + Few clouds, Mist , 3: Light Snow, Light Rain + Thunderstorm + Scattered clouds, Light Rain + Scattered clouds , 4: Heavy Rain + Ice Pallets + Thunderstorm + Mist, Snow + Fog , temp: Normalized temperature in Celsius. The values are derived via (t-t_min)/(t_max-t_min), t_min=-8, t_max=+39 (only in hourly scale), atemp: Normalized feeling temperature in Celsius. The values are derived via (t-t_min)/(t_max-t_min), t_min=-16, t_max=+50 (only in hourly scale), hum: Normalized humidity. The values are divided to 100 (max), windspeed: Normalized wind speed. The values are divided to 67 (max) , casual: count of casual users , registered: count of registered users , cnt: count of total rental bikes including both casual and registered. The dataset covers the years 2011 and 2012.

	instant	dteday	season	yr	mnth	hr	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	casual	registered	cnt
0	1	2011-01-01	1	0	1	0	0	6	0	1	0.24	0.2879	0.81	0.0	3	13	16
1	2	2011-01-01	1	0	1	1	0	6	0	1	0.22	0.2727	0.80	0.0	8	32	40
2	3	2011-01-01	1	0	1	2	0	6	0	1	0.22	0.2727	0.80	0.0	5	27	32
3	4	2011-01-01	1	0	1	3	0	6	0	1	0.24	0.2879	0.75	0.0	3	10	13
4	5	2011-01-01	1	0	1	4	0	6	0	1	0.24	0.2879	0.75	0.0	0	1	1

Fig: First 5 row of the data.

```
hour.shape
(17379, 18)
```

Fig: Shape of the data.

```
hour.describe()
```

	instant	season	yr	mnth	hr	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	
count	17379.0000	17379.000000	17379.000000	17379.000000	17379.000000	17379.000000	17379.000000	17379.000000	17379.000000	17379.000000	17379.000000	17379.000000	17379.000000	173
mean	8690.0000	2.501640	0.502561	6.537775	11.546752	0.028770	3.003683	0.682721	1.425283	0.496987	0.475775	0.627229	0.190098	
std	5017.0295	1.106918	0.500008	3.438776	6.914405	0.167165	2.005771	0.465431	0.639357	0.192556	0.171850	0.192930	0.122340	
min	1.0000	1.000000	0.000000	1.000000	0.000000	0.000000	0.000000	0.000000	1.000000	0.020000	0.000000	0.000000	0.000000	
25%	4345.5000	2.000000	0.000000	4.000000	6.000000	0.000000	1.000000	0.000000	1.000000	0.340000	0.333300	0.480000	0.104500	
50%	8690.0000	3.000000	1.000000	7.000000	12.000000	0.000000	3.000000	1.000000	1.000000	0.500000	0.484800	0.630000	0.194000	
75%	13034.5000	3.000000	1.000000	10.000000	18.000000	0.000000	5.000000	1.000000	2.000000	0.660000	0.621200	0.780000	0.253700	
max	17379.0000	4.000000	1.000000	12.000000	23.000000	1.000000	6.000000	1.000000	4.000000	1.000000	1.000000	1.000000	0.850700	3

Fig: Describe the dataset.

Exploratory Data analysis:

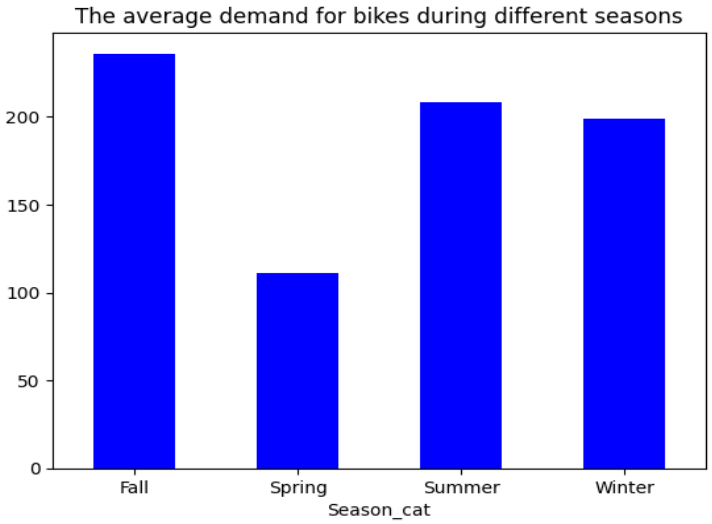


Fig: Average demand for bikes during different seasons

In comparison, to the seasons the fall season tends to witness a demand for motorcycles. Furthermore, research indicates that the demand for motorcycles is at its lowest, during the spring season.

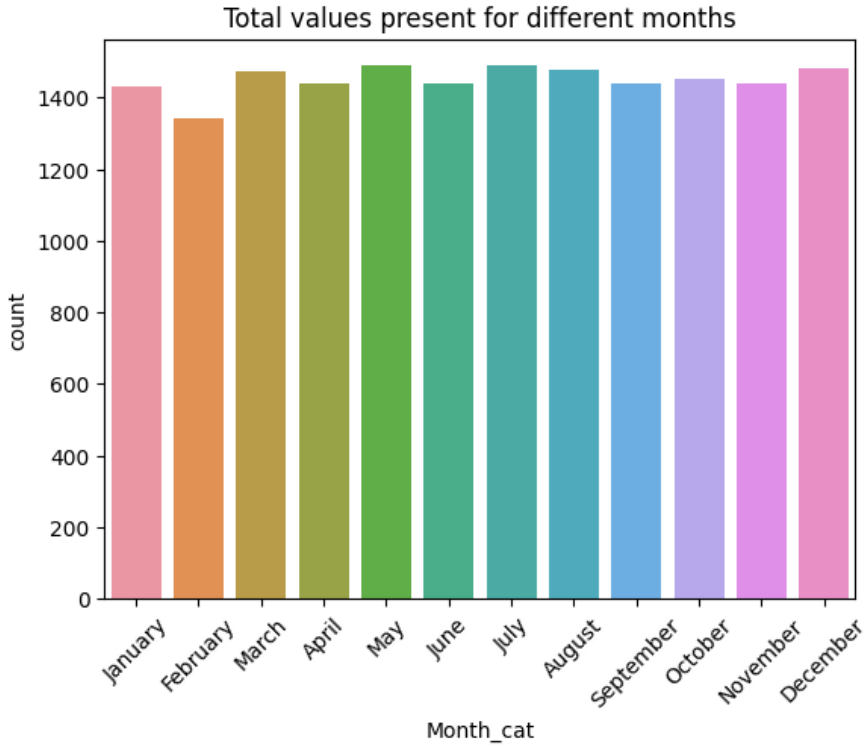


Fig: Count plot of different plot

A graph that represents the values observed over months. From this we can deduce that some months have days compared to others leading to lower values during those months as compared to the latter ones.

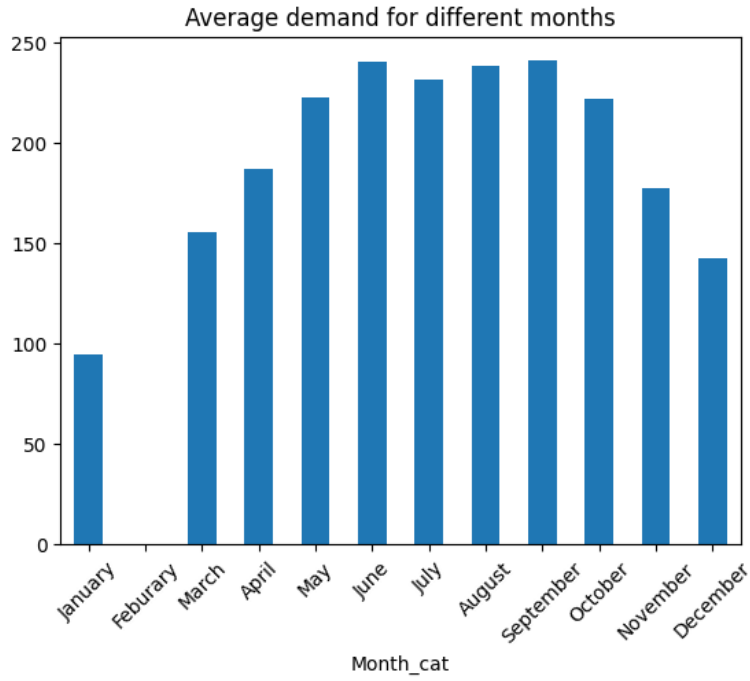


Fig: Avg demand of different months

September, June, and August are the months when bicycles in demand. It's noticeable that some months have demand compared to others. January has the demand, for motorcycles as well. Considering the expected surge in motorcycle demand in September it's important to take actions. However, we can observe that there isn't demand for bikes during January. To ensure access, to bicycles when needed they could be relocated to an area.

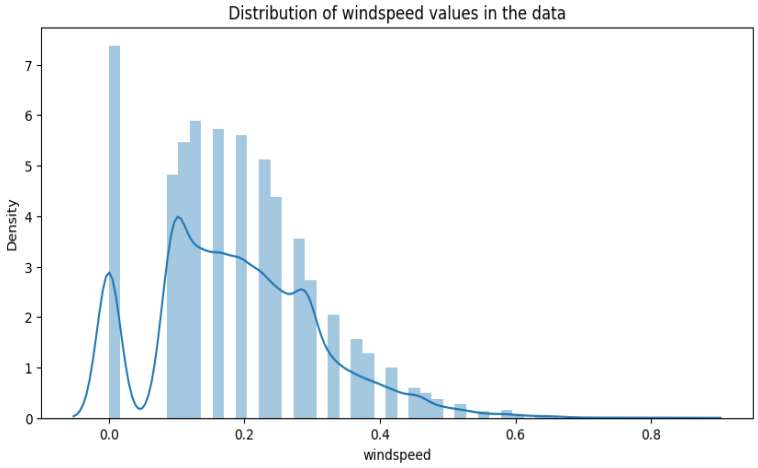


Fig: Distribution of windspeed

The majority of the wind speed figures we deal with are generally, on the end, which's evident, from the distribution of values. There are a few values. As a result, the dataset we are working with consists of wind speeds that're not particularly high.

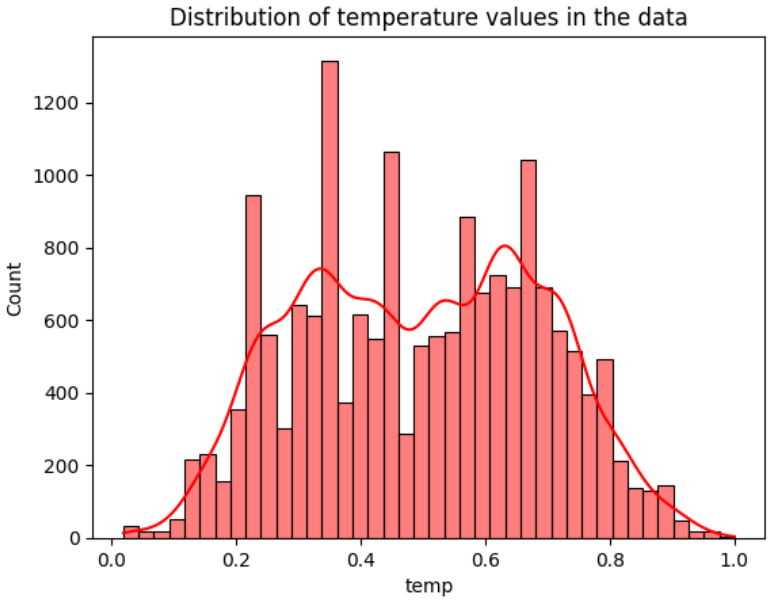


Fig: Distribution of temperature Values

The temperature data appears to be spread out as we can see. We are using these distributed temperature readings, which ensures that we cater to the demand, for temperature values.

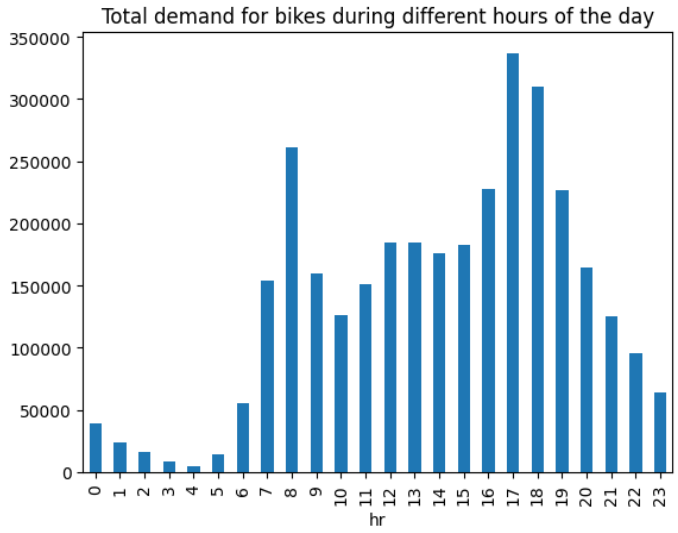


Fig: Total demand for bikes during different hours of the day

The busiest time, for bike usage is typically 5 p.m. In Washington. It's clear that there isn't demand for motorcycles during the morning hours specifically from 1 am to 7 am. However, we can see an increase, in bike demand starting at 8 a.m. This trend continues to rise throughout the day.

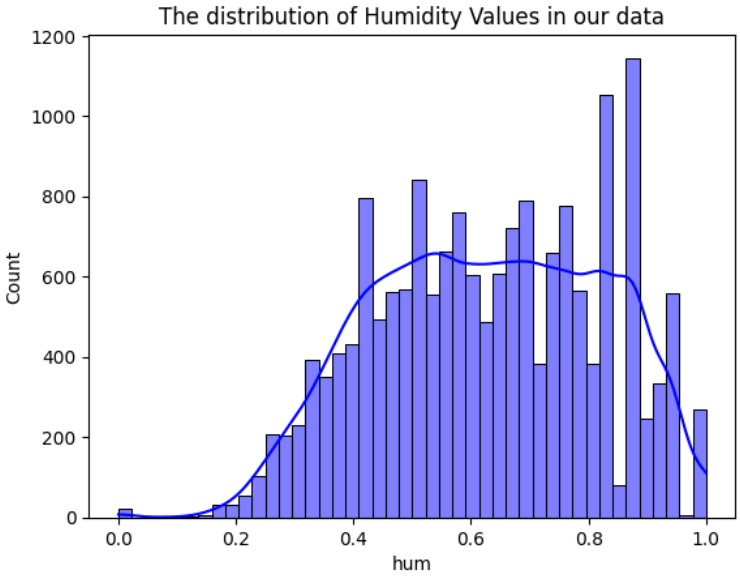


Fig: The distribution of humidity values in our data

The plot shows that the humidity levels are distributed quite evenly. The figures reflect a clear shift to the left.

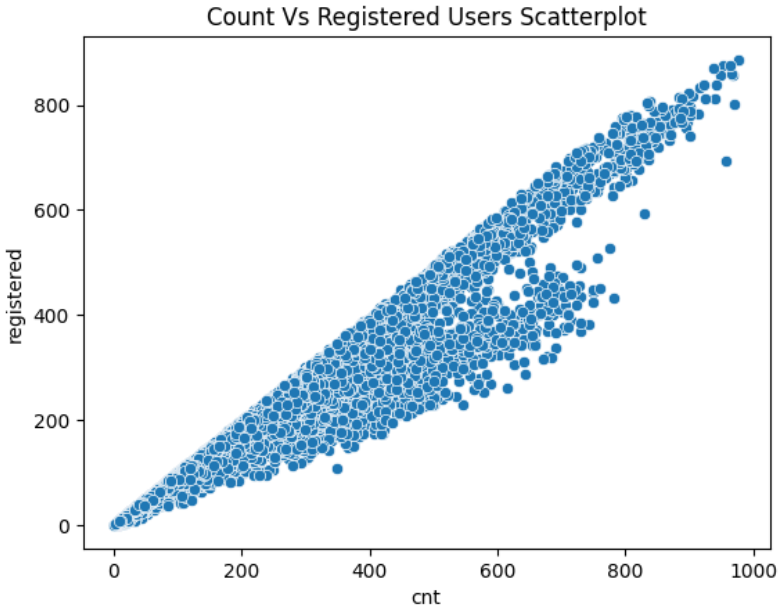


Fig: count Vs Registered Users Scatterplot

It is highly likely that as more people register the demand, for bikes will increase. The demand for motorcycles is mainly driven by riders when there are a few registered users. However, with an increase in registered users and bikes the statistics suggest that registered users play a role, in creating the demand. Conversely if the number of registered users is minimal it becomes evident that both registered users and casual riders contribute to the demand.

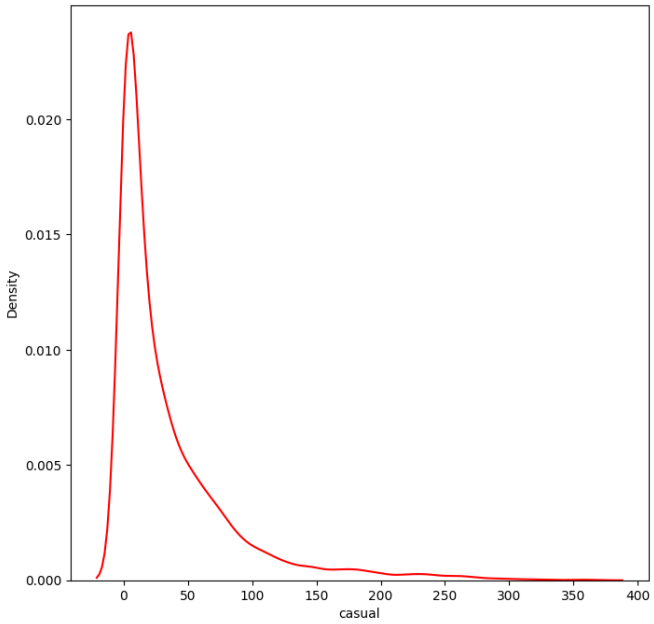


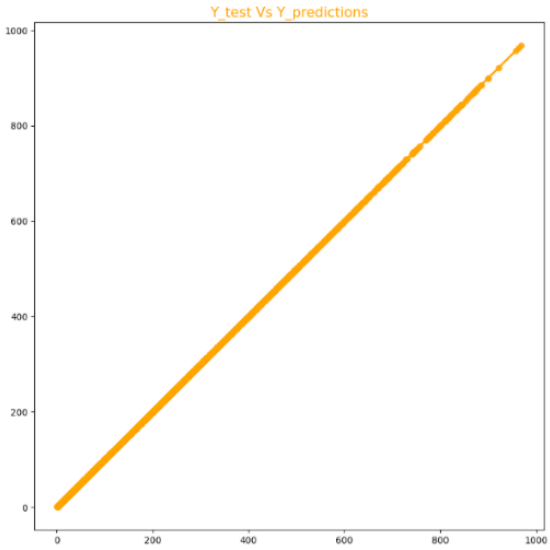
Fig: Plot for casual riders.

After analyzing the distribution plot, it becomes evident that there are instances where the number of riders exceeds 400. However, there is an occurrence of riders with a maximum count of, around 50. The graph below illustrates a distribution that's skewed to the right indicating that a small portion of values are concentrated on the side while the majority are spread out towards the left. Based on this graph we can infer that casual riders below the age of 50 are more common compared to those, above 50 who represent a proportion of the population.

MODELS:

We used models for the next step. First, we divided the dataset into training and test sets. For the prediction tasks we selected Linear Regression, K Nearest Neighbors, PLS Regression, Decision Tree Regressor and Gradient Boosting Regressor as our models. To assess their performance, we measured error and mean absolute error, for each model.

Linear Regression:



When we used Linear Regression on the test set, we discovered that the Mean squared error was low, at 0 and the mean absolute error was also 0. Upon examining the graph in comparison to y_prediction it's evident that there is a line indicating a resemblance, between the model's predictions and the actual values. This significant alignment implies that the model effectively captures the patterns and trends within the data.

K Nearest Neighbors:

```
[ ] model = KNeighborsRegressor()
    mean_squared_error_list = []
    mean_absolute_error_list = []
    roc_auc_score_list = []
    K_nearest_neighbors = [2, 3, 5, 8, 10, 11, 15, 20]
    for i in K_nearest_neighbors:
        model = KNeighborsRegressor(n_neighbors = i)
        model.fit(X_train, y_train)
        y_predict = model.predict(X_cv)
        mean_squared_error_list.append(mean_squared_error(y_predict, y_cv))
        mean_absolute_error_list.append(mean_absolute_error(y_predict, y_cv))
```

making a dictionary out of the dataframe can help us comprehend the various output faults that are there.

```
[ ] knn_dictionary = {'K Nearest Neighbors': K_nearest_neighbors, 'Mean Squared Error': mean_squared_error_list, 'Mean Absolute Error': mean_absolute_error_list}
```

```
[ ] knn_dataframe = pd.DataFrame(knn_dictionary)
```

We obtain various mean squared errors and mean absolute errors for different nearest neighbor values, accordingly.

```
[ ] knn_dataframe
```

	K Nearest Neighbors	Mean Squared Error	Mean Absolute Error
0	2	3630.991609	41.799482
1	3	3280.844713	40.439266
2	5	3057.204557	40.073878
3	8	3134.797444	41.297756
4	10	3245.559457	42.051151
5	11	3321.866870	42.586376
6	15	3521.381360	44.447168
7	20	3749.084734	46.243700

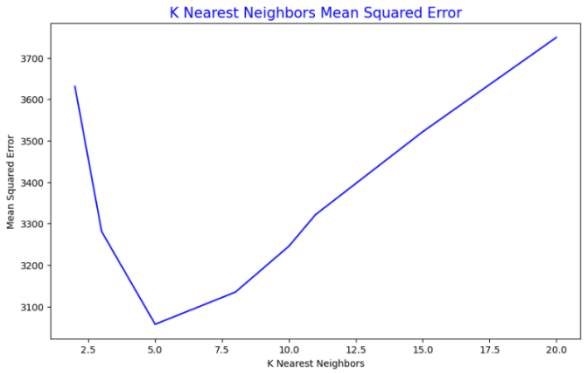


Fig: k Nearest Neighbors Mean Squared Error

The graph shows that with the increase in the value of K, there is an increase in the error. The best value of K as can be seen is 5.

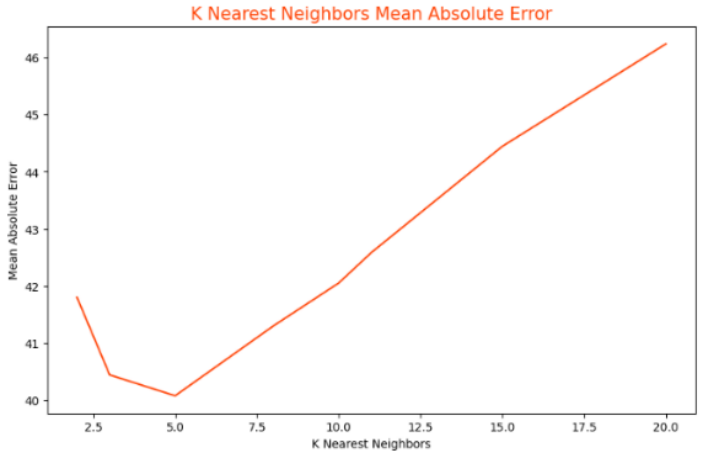


Fig: K Nearest Neighbors Mean Absolute Error

There seems to be a growing trend where the mean absolute error rises, in correlation with the value of K. Consequently, when the value of K is increased there is an increase in the mean absolute error. In this situation it appears that the optimal value, for K would be 5.

The plot depicts K Nearest Neighbors' performance on the test set.

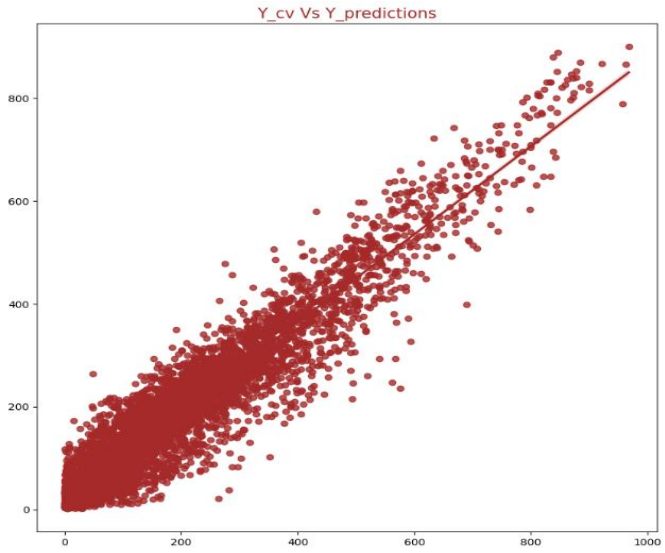


Fig: Scatterplot between Y_test and Y_Prediction

There appears to be a difference, in the predictions made by the model and the actual test data. Therefore, we can explore models that may enhance performance more. The graph illustrates how K Nearest Neighbors performs on the test set.

PLS Regressor

```
n_components_list = [2, 3, 5, 7, 10]
mean_squared_error_list = []
mean_absolute_error_list = []
for i in n_components_list:
    model = PLSRegression(n_components = i)
    model.fit(X_train, y_train)
    y_predict = model.predict(X_cv)
    mean_squared_error_list.append(mean_squared_error(y_predict, y_cv))
    mean_absolute_error_list.append(mean_absolute_error(y_predict, y_cv))
```

```
pls_regression_dict = {'Number of Components': n_components_list, 'Mean Absolute Error': mean_absolute_error_list,
                       'Mean Squared Error': mean_squared_error_list}
```

```
pls_regression_dataframe = pd.DataFrame(pls_regression_dict)
```

```
pls_regression_dataframe
```

	Number of Components	Mean Absolute Error	Mean Squared Error
0	2	39.913369	2631.306818
1	3	19.629708	590.610873
2	5	3.243457	18.599942
3	7	0.549953	0.545682
4	10	0.076876	0.010923

```
plot_function(pls_regression_dataframe)
```

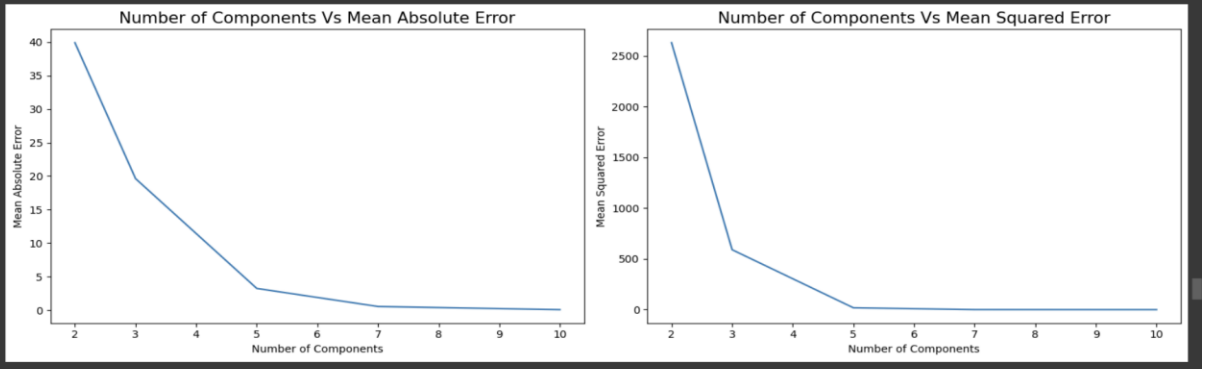


Fig: Number of components Vs Mean Absolute Error & Number of Components Vs Mean Squared Error

Based on the findings we can observe that the best number of components is 5. This value yields the absolute error and a relatively low mean squared error as well. Although there may be component values, with lower mean squared errors we opt for the optimal hyperparameter to guarantee that both the mean absolute and mean squared errors remain low.

Decision tree regressor

```
[ ]  max_depth_list = [10, 15, 16, 17, 18, 20, 25]
     mean_absolute_error_list = []
     mean_squared_error_list = []
     for i in max_depth_list:
         model = DecisionTreeRegressor(max_depth = i)
         model.fit(X_train, y_train)
         y_predict = model.predict(X_cv)
         mean_absolute_error_list.append(mean_absolute_error(y_predict, y_cv))
         mean_squared_error_list.append(mean_squared_error(y_predict, y_cv))
     decision_tree_dict = {'Max Depth': max_depth_list, 'Mean Absolute Error': mean_absolute_error_list,
                           'Mean Squared Error': mean_squared_error_list}
     decision_tree_dataframe = pd.DataFrame(decision_tree_dict)
```

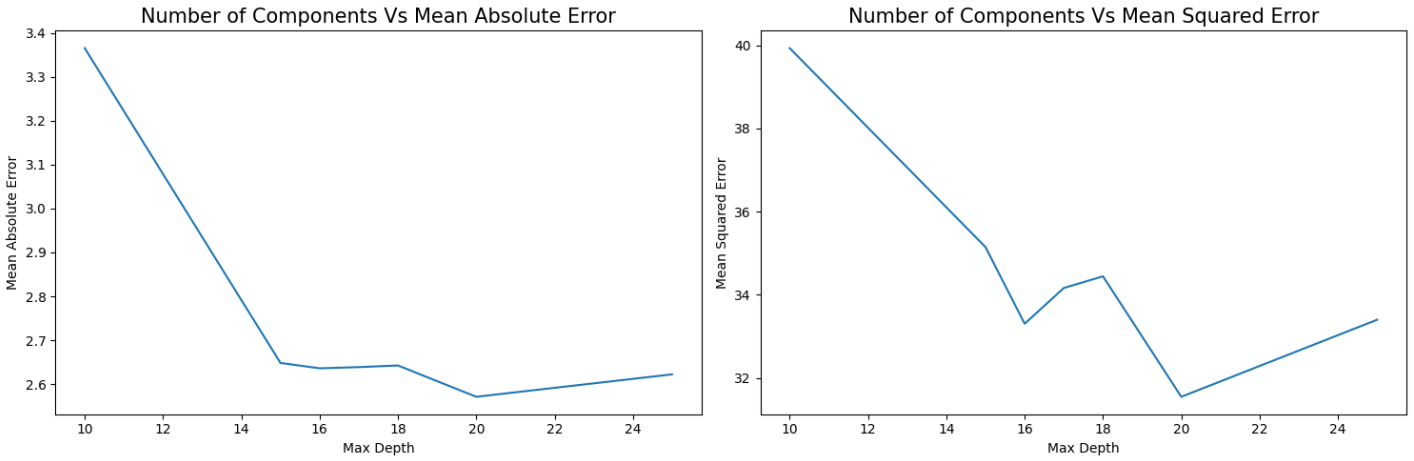
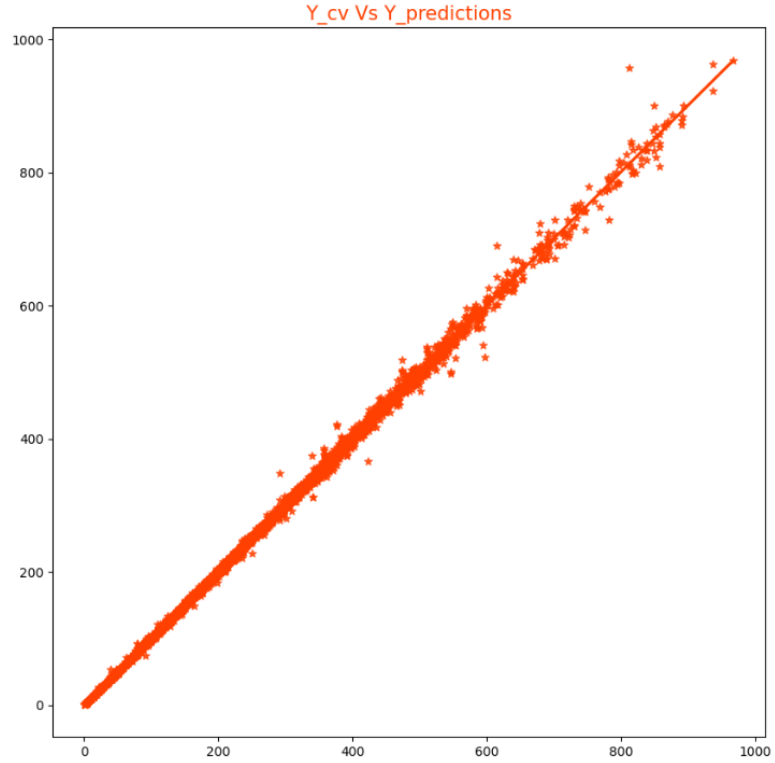


Fig: Number of components Vs Mean Absolute error & Number of components vs Mean squared error

Based on our observations we have determined that the ideal value, for the depth in terms of minimizing both mean error and mean absolute error is 20. Therefore 20 can be considered as the depth value as it yields the lowest values, for both metrics.

```
[ ]  model = DecisionTreeRegressor(max_depth = best_max_depth)
     model.fit(X_train, y_train)
     y_predict = model.predict(X_cv)
     plt.figure(figsize=(10, 10))
     sns.regplot(x=y_predict, y=y_cv, marker='*', color='#ff4000')
     plt.title('Y_cv Vs Y_predictions', fontsize=15, color='#ff4000')
     plt.show()
```



The scatter plot comparing the predicted values (y_prediction) to the values (y_test) of the Decision Tree Regressor reveals a linear relationship. This suggests that the models' predictions align closely with the values indicating its accuracy, in capturing underlying patterns and trends, within the data. Overall, the model performs well.

Gradient Boosting Regressor

```
from sklearn.ensemble import GradientBoostingRegressor
```

```
[ ] n_estimators_list = [25, 50, 100, 150, 200, 400, 1000]
    mean_squared_error_list = []
    mean_absolute_error_list = []
    for i in n_estimators_list:
        model = GradientBoostingRegressor(n_estimators = i, max_depth = 10)
        model.fit(X_train, y_train)
        y_predict = model.predict(X_cv)
        mean_squared_error_list.append(mean_squared_error(y_cv, y_predict))
        mean_absolute_error_list.append(mean_absolute_error(y_cv, y_predict))
    gradient_boosting_regressor_dict = {"Number of Estimators": n_estimators_list, "Mean Absolute Error": mean_absolute_error_list,
                                        "Mean Squared Error": mean_squared_error_list}
    gradient_boosting_regressor_dataframe = pd.DataFrame(gradient_boosting_regressor_dict)
```

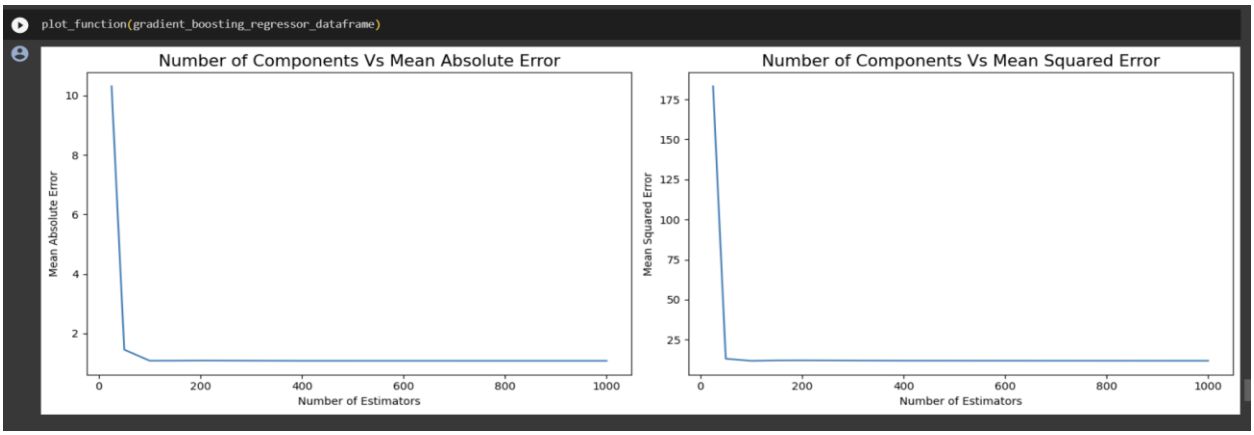
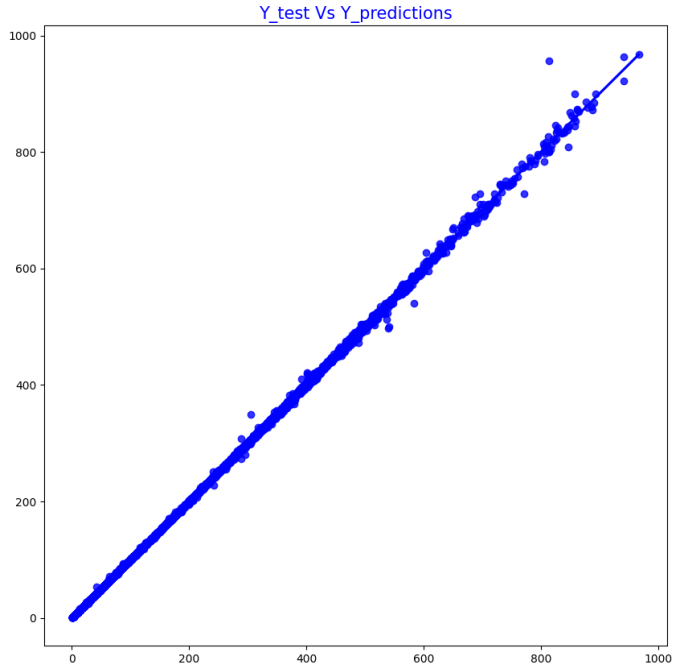


Fig: Number of components vs Mean absolute error & Number of components Vs Mean squared error

gradient_boosting_regressor_dataframe

	Number of Estimators	Mean Absolute Error	Mean Squared Error
0	25	10.306138	183.158872
1	50	1.450300	13.184505
2	100	1.078649	11.857324
3	150	1.080747	12.092939
4	200	1.085139	12.136908
5	400	1.075596	11.955964
6	1000	1.075004	11.892425



Gradient boosted regression models are also quite effective, in predicting and forecasting the demand for bicycles at times throughout the day. Their performance is quite comparable to that of decision tree regression as depicted in the figure provided below.

Result & Conclusion

The models that showed the predictions, for bike demand were Gradient Boosting Regressor and Decision tree regressor.

We utilized Exploratory Data Analysis (EDA) to gain an understanding of the features and their impact, on the output variable.

The performing machine learning models achieved an absolute error (MAE) of approximately 20 which is impressive considering the significance of the problem we were addressing.

Contributions

Sidhant Thakur

In the update I began by exploring the data through visualizations, for Exploratory Data Analysis. I created plots like a Countplot of Different Seasons showing the demand for bikes during seasons as well as the total demand for bikes, in each season.

Moving on to the update I implemented regression and generated visualizations to analyze its results.

For the update I fine-tuned the Gradient Boosting Regression algorithms. Created several visualizations to assess their performance. Through this analysis I identified the model among them.

Saransh Thakur

In the update I started by exploring and cleaning the data. I also created visualizations to analyze the data further.

Moving on to the update I implemented K NN and created visualizations, like line plots that show how neighbors relate to mean error. I also used line plots to demonstrate the correlation between neighbors and mean error. Additionally, I utilized scatter plots to illustrate the relationship, between y_test and Y_prediction.

In Update 3 I focused on PLS Regression. Generated plots that depict how changing the number of components affects mean error and mean squared error. Furthermore, I created a plot to visualize the results of the PLS regression process.

Parth Patel

In the update I analyzed the data. Began by cleaning and exploring it. I also created scatterplots to visualize aspects of the data.

Moving on to the update I implemented the decision tree algorithm. Provided guidance to my team members on applying the KNN algorithm. After that we plotted scatterplots, for both the training and test datasets once we applied our model.

In Update 3 my focus shifted towards reviewing a paper and assisting Saransh and Sidhant, with PLS regression and Gradient Boosting regression. Our collective conclusion was that among all the algorithms we explored Gradient Boost Algorithm proved to be the effective.

Future Scope

To enhance the accuracy of bike demand prediction models it would be beneficial to incorporate data, such, as the connectivity of streets and public sentiment towards cycling in a particular area. This would enable the models to make forecasts. Moreover, utilizing machine learning models in time can help identify regions with a significant demand for bikes. This empowers administrators to take steps like redistributing bikes or increasing bike availability, in locations to meet the demands effectively.

Sources:

1] IEEE Xplore Full-Text PDF:

2] (PDF) Prediction of Bike Share Demand by Machine Learning: Role of Vehicle Accident as the New Feature (researchgate.net)

3]Using Machine Learning Methods to Predict Demand for Bike Sharing | SpringerLink

4]main.pdf (sciencedirectassets.com)

5] Machine Learning Approaches to Bike-Sharing Systems

6]Substitution effect or complementation effect for bicycle travel choice preference and other transportation availability: Evidence from US large-scale shared bicycle travel behaviour data - ScienceDirect

7] 2203.10961.pdf (arxiv.org)

8]Predictive and prescriptive performance of bike-sharing demand forecasts for inventory management - ScienceDirect

9]Real-time prediction of public bike sharing system demand using generalized extreme value count model - ScienceDirect

10]Using Machine Learning Methods to Predict Demand for Bike Sharing | SpringerLink

11]main.pdf (sciencedirectassets.com)

12] Machine Learning Approaches to Bike-Sharing Systems

13]Substitution effect or complementation effect for bicycle travel choice preference and other transportation availability: Evidence from US large-scale shared bicycle travel behavior data - ScienceDirect

14]Performance of bike-sharing demand forecasts for inventory management - ScienceDirect

15]Complementation effect for bicycle travel choice preference and other transportation availability: Evidence from US large-scale shared bicycle travel behaviour data - ScienceDirect

16] 2203.10961.pdf (arxiv.org)

17]bike-sharing demand forecasts for inventory management - ScienceDirect

18] IEEE Xplore Full-Text PDF:

19] (PDF) Prediction of Bike Share Demand by Machine Learning: Role of Vehicle Accident as the New Feature (researchgate.net)

20]Using Machine Learning Methods to Predict Demand for Bike Sharing | SpringerLink