Dr. Manasa Manjunatha

Goal: To predict the hourly utilization of rental bikes

EDA and Predictive model for Bike Sharing Dataset:

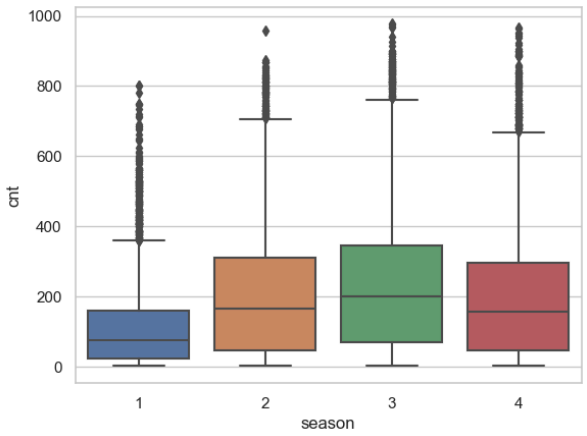
As the first step we present the exploratory data analysis that we conducted on the Bike Sharing Dataset. The data set contains hourly information over a period of two years regarding the following attributes: date, season, year, month, hour, whether it is a holiday or not, weekday or not, working day or not, what the weather situation is like, the temperature ('temp'), felt temperature ('atemp'), humidity ('hum'), wind speed and the count of total rental bikes('cnt').

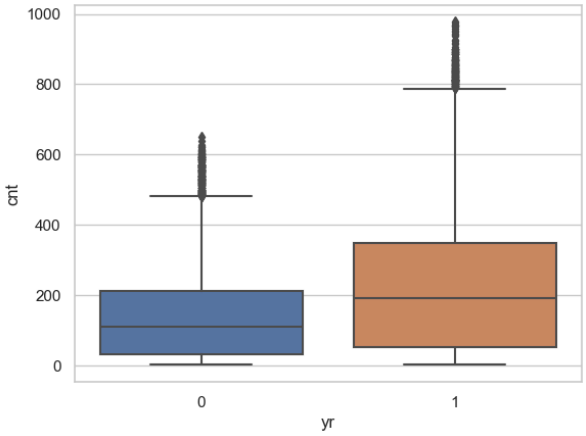
1. The data set has 17379 entries and no missing values. The categorical variables 'season', 'yr', 'mnth', 'hr', 'holiday', 'weekday', 'workingday', 'weathersit' are all encoded as integers and the numerical variables 'temp', 'atemp', 'hum', 'windspeed' are normalized.
2. Since we are to predict the count of total rental bikes, we plot the dependence of 'cnt' on the numerical variables and categorical variables using scatter plots and box plots respectively. The plots for this are attached.
3. Main observations are that, at low temperatures and at high wind speeds the utilization is less. Furthermore, in winter the utilization is the least of all the seasons. Utilization is most during working weekdays and during peak commuting hours 8hr and 17hr. We can also see that the utilization drops with the worsening weather situation.
4. Before building a predictive model, we would like to look into the correlation of the numerical features among themselves. The heatmap of the correlations shows that 'temp' and 'atemp' are highly correlated with each other, therefore we drop 'temp' from our considered features.

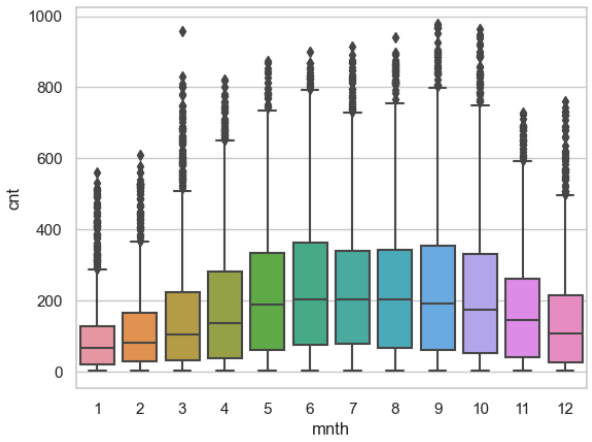
To predict the utilization of rental bikes we train a regression model on our dataset. We train and evaluate 5 different models using cross-validation namely, Linear Regression, Lasso Regression, Ridge Regression, Support Vector Regression and XGBoost Regression. We evaluated these models using the Average Mean Absolute Deviation (MAD) of the cross-validated models and found the highest accuracy in **XGBoost (MAD = 40.95)**. Therefore, we find the XGBoost model most suitable for this business case.

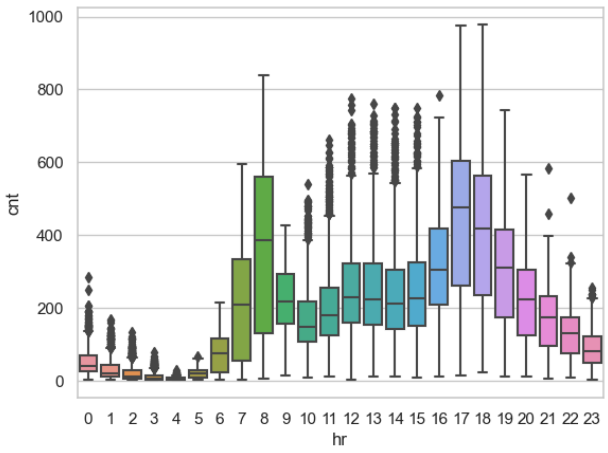
In our model we also performed a simple unit test to ensure the sanity of the model. We test that the MAD of the model is always less than the MAD of the data with its mean.

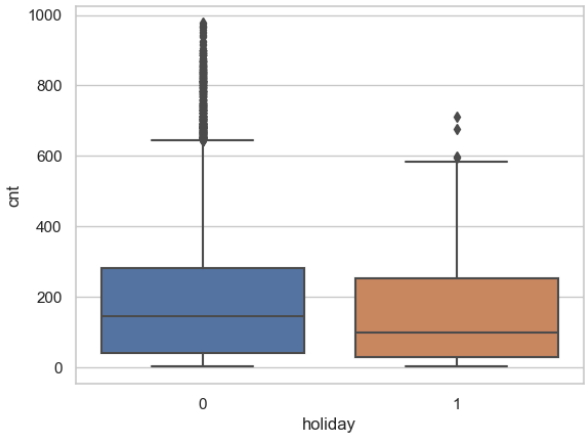
2

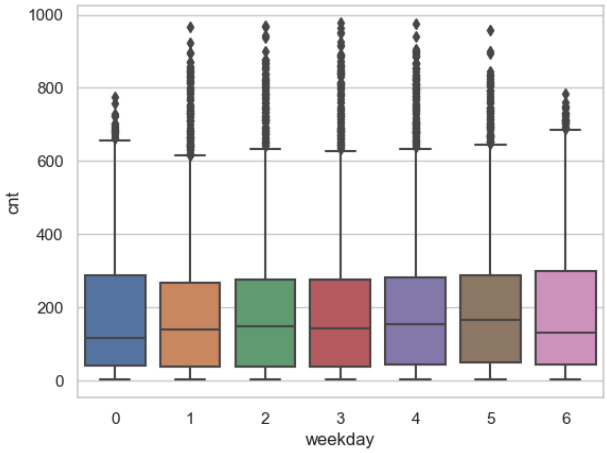

cnt
1000
800
600
400
200
0
1
2
3
4
season


cnt
1000
800
600
400
200
0
0
1
yr

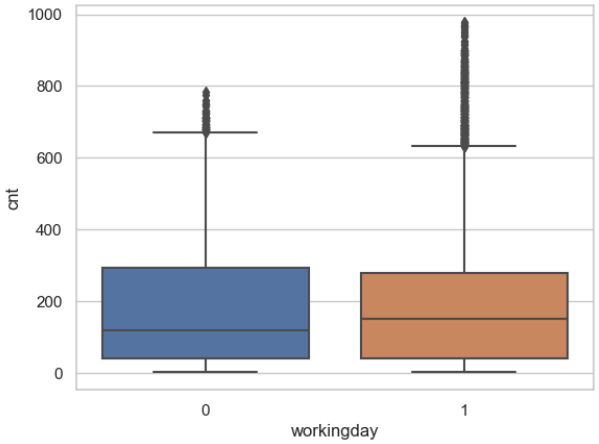

cnt
1000
800
600
400
200
0
1
2
3
4
5
6
7
8
9
10
11
12
mnth

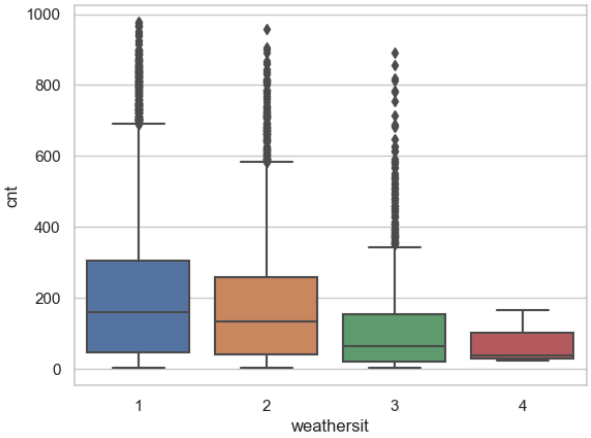

cnt
1000
800
600
400
200
0
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
hr

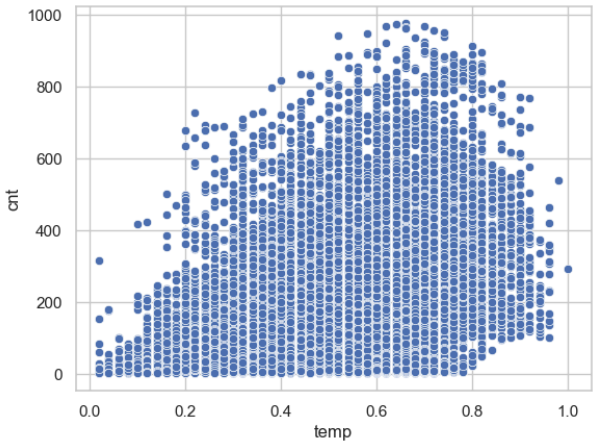

cnt
1000
800
600
400
200
0
0
1
holiday

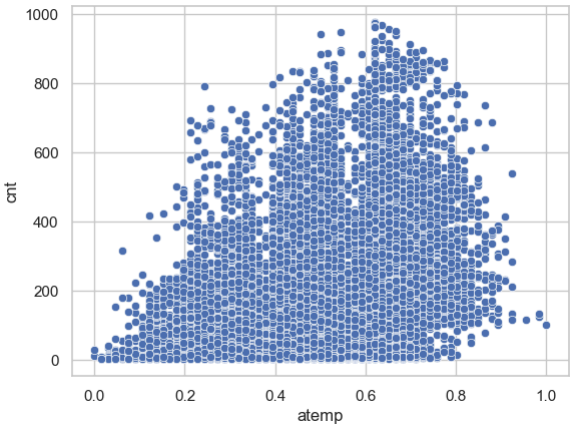

cnt
1000
800
600
400
200
0
0
1
2
3
4
5
6
weekday

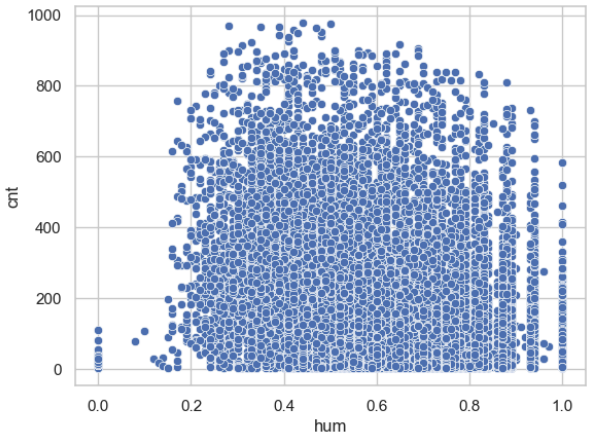
3

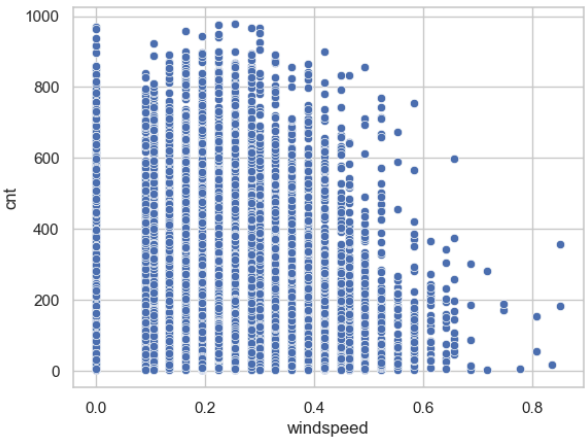

cnt
1000
800
600
400
200
0
0
1
workingday

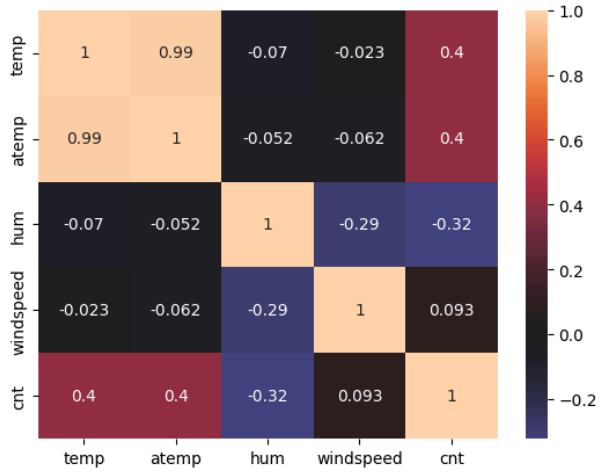

cnt
1000
800
600
400
200
0
1
2
3
4
weathersit

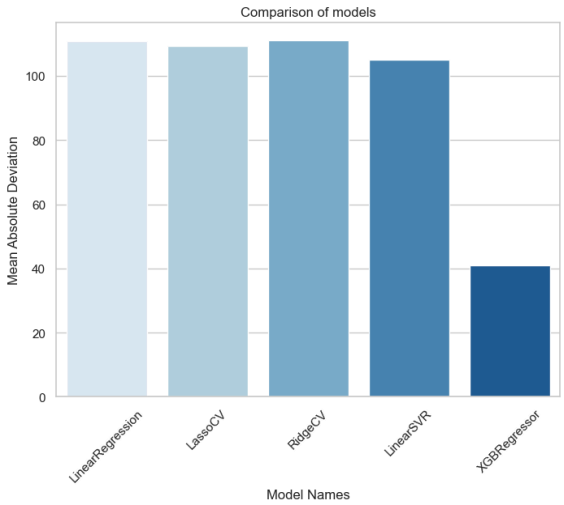

cnt
1000
800
600
400
200
0
0.0
0.2
0.4
0.6
0.8
1.0
temp


cnt
1000
800
600
400
200
0
0.0
0.2
0.4
0.6
0.8
1.0
atemp


cnt
1000
800
600
400
200
0
0.0
0.2
0.4
0.6
0.8
1.0
hum


cnt
1000
800
600
400
200
0
0.0
0.2
0.4
0.6
0.8
windspeed





Scaling the model:

The main challenges that one faces when scaling this model to handle several terabytes of data are the increase in computational time, insufficient memory for processing and storage space. These challenges can be handled by parallel processing of partitioned data and distributed storage systems. For this we can use distributed computing frameworks (e.g., Apache Spark, Dask), data storage solutions (e.g., Hadoop Distributed File System), and cloud-based services.

The model I present here, an XGBoost model, is designed for efficient parallel processing and distributed computing. I can adapt my model to do the same by using the 'Dask' library. An overview of how I would do this is by first setting up a Dask cluster on Kubernetes and distributing my data set across it. I would then train the Dask DataFrames by passing them to the 'dask_xgboost.train()' function. Once the model is trained in this way, it can also be used to perform distributed predictions.

While XGBoost is efficient with moderately large datasets, it may face problems when handling extremely large data sets. In the model I have presented, hyperparameter tuning has not been carried out, although it was a possibility. In such cases, despite the strong performance potential of gradient boosting in various scenarios, dealing with the model's many hyperparameters may lead to increased computational demands during the tuning process.