Blue Yonder Data Science Consultant Challenge

Varun Balaji Srinivasan
July 7, 2021

1 TASK 1-EXPLORATORY DATA ANALYSIS AND PREDICTION MODEL

1.1 EXPLORATORY DATA ANALYSIS

The challenge is to analyse and get business insights on the given dataset "Bike Sharing". Following the CRISP-DM standard as a rule of thumb, the first step taken to implement this challenge was Business understanding followed by data understanding, pre-processing and modeling of the given data. The dataset "hour.csv" consists of 17,380 records and 17 columns. This small-to-medium dataset is read using pandas library. In the preprocessing stage, *instant* column which acts as an index for the dataset also does not have significant effect on the target variable '*cnt*', and thus it is dropped from the dataframe.
The "dataframe.info()" gives the column name and the corresponding data type for the values present in the dataframe. Similarly "dataframe.describe()" gives the count, minimum value,maximum value, mean, standard deviation of the dataset values.
The dataset does not contain any null or Nan values which is verified with ".isnull().sum()" that returns the number of null values present in each column.
The ´exploratory data analysis is experimented using Matplot library, Seaborn and Plotly Express in python.
Following are different plots where different columns are visualized in order to get some understanding of the behaviour of the dependent and independent variables.
The figure 1.1 shows the plot between date column on x axis and count column on y axis. It is observed that there is an increase in the bike share between June and October after which the demand for the bikes has reduced. Similar pattern is observed for the next year(2012) this can infer that there is a presence of seasonality. To verify the above mentioned statement, the count of bike shares are plotted against season.

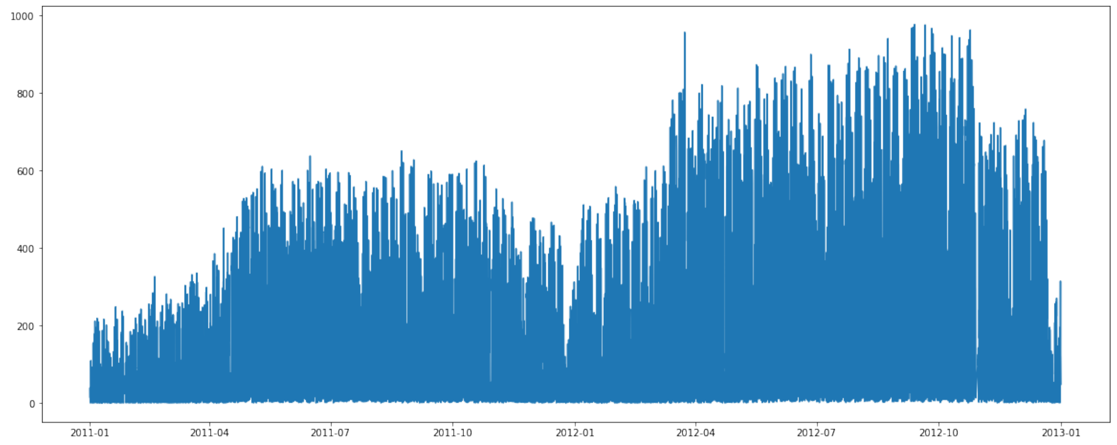


Figure 1.1: Bike share between Jan 2011 and Jan 2013

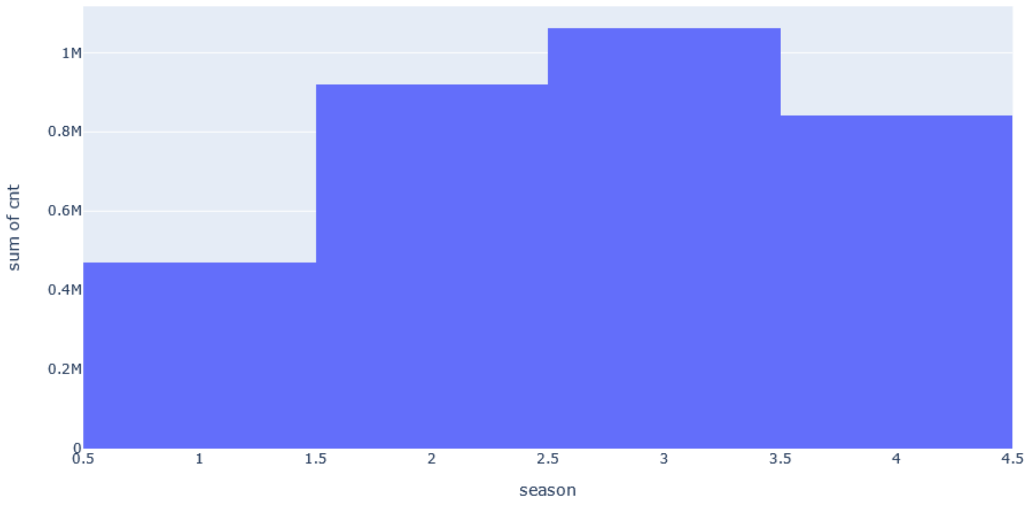


Figure 1.2: Bike share in different seasons

From the figure 1.2, it can be inferred that the demand for bikes are more in the summer compared to the other seasons followed by spring and fall. The lowest count is recorded in winter.To get an insight on the bike sharing for the year 2011 and 2012, the cumulative or the sum of count is plotted against the year.

From figure 1.3, the bike sharing has been more in 2011 compared to 2012 with a total count of 2.04 Million.
To further understand the behaviour of the commuters in bike sharing, figure 1.4 shows that

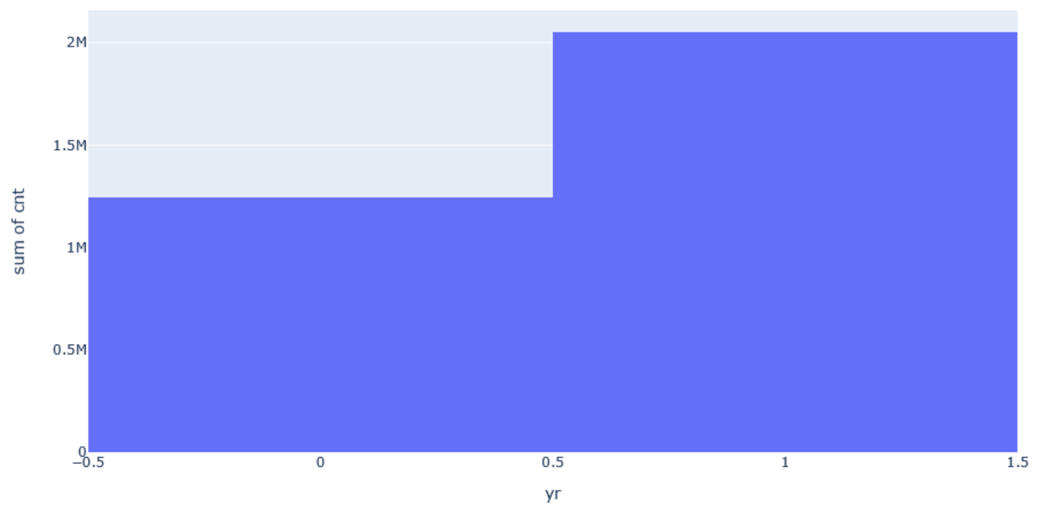


Figure 1.3: Bike share in year 2011(0) and 2012(1)

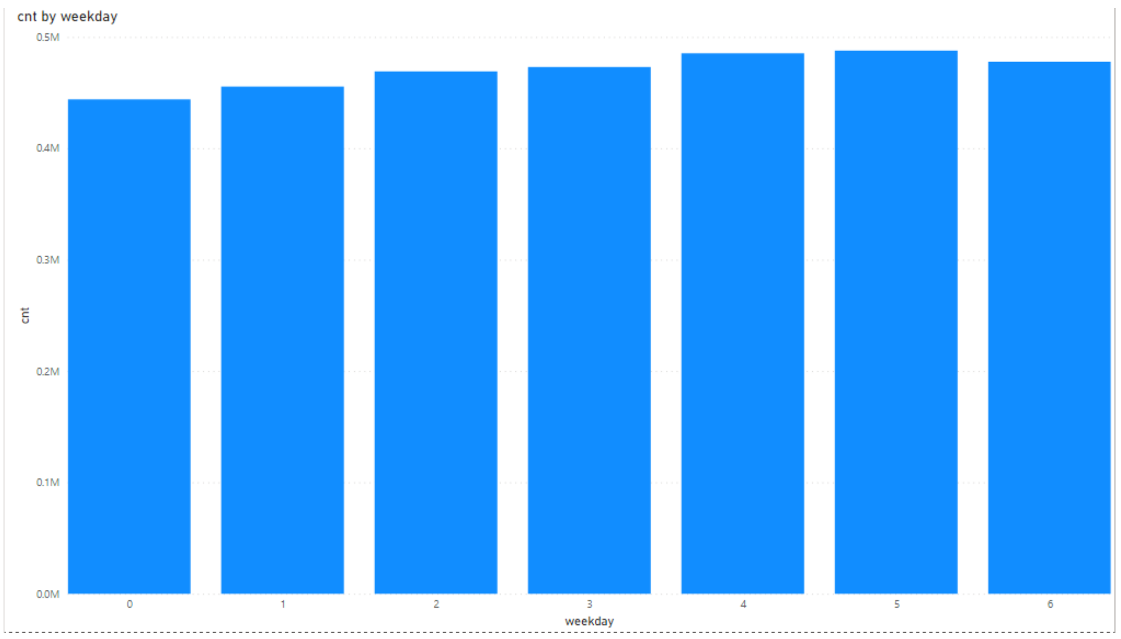


Figure 1.4: Bike share in different days of a week

more number of bikes have been mostly shared or used on Friday compared to other days of the week. The least number of shares are recorded on Sundays.
The figure 1.5 shows that the count was more during the working day(1) with 2.24million while

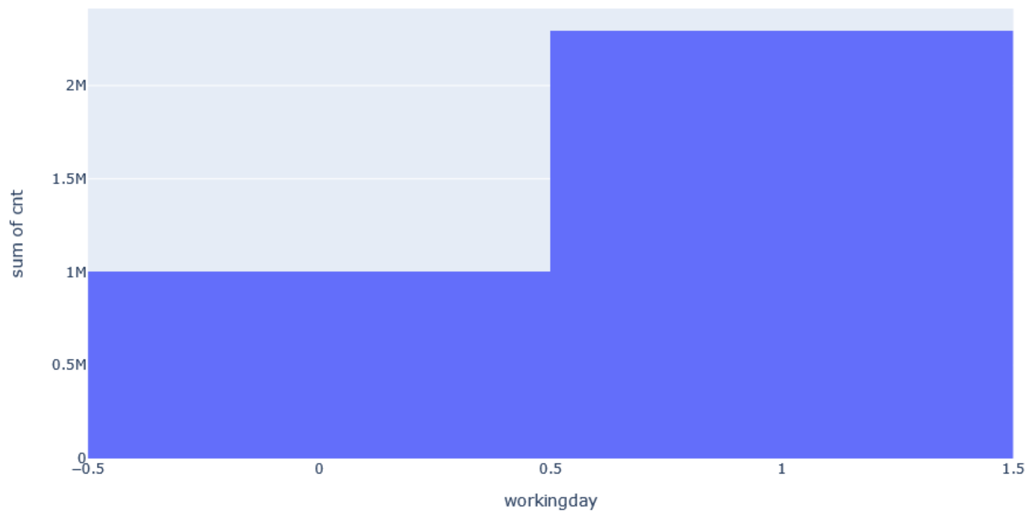


Figure 1.5: Bike share in working day or not

the rest of the count is found during holiday or weekend.

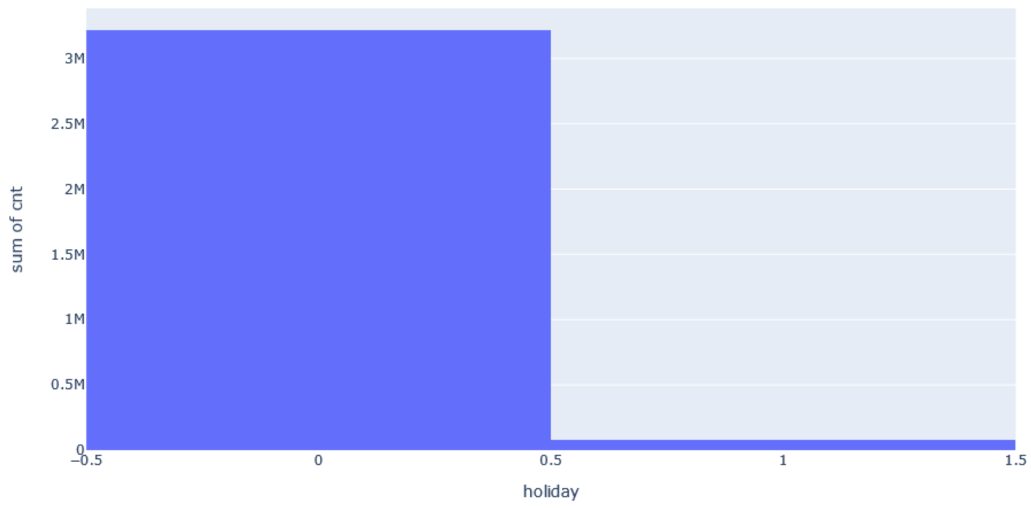


Figure 1.6: Count of Bike share in Holidays

The figure 1.6 infers that the number of bike share was very less with 78,435 during the holidays while it is high during the rest of the days with count of 3.21 Million.

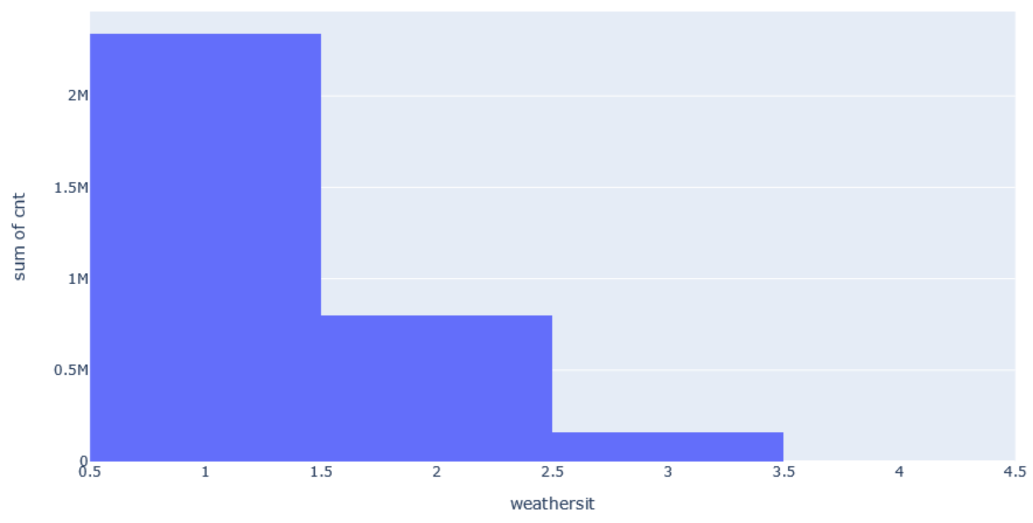


Figure 1.7: Count of Bike share in different weathers

The figure 1.7 shows that the users prefer to drive bikes during clear,few clouds or partly cloudy with the count of 2.33 Million while on the other hand, least number of bikes were taken during heavy rain, thunderstorm, mist snow or fog.

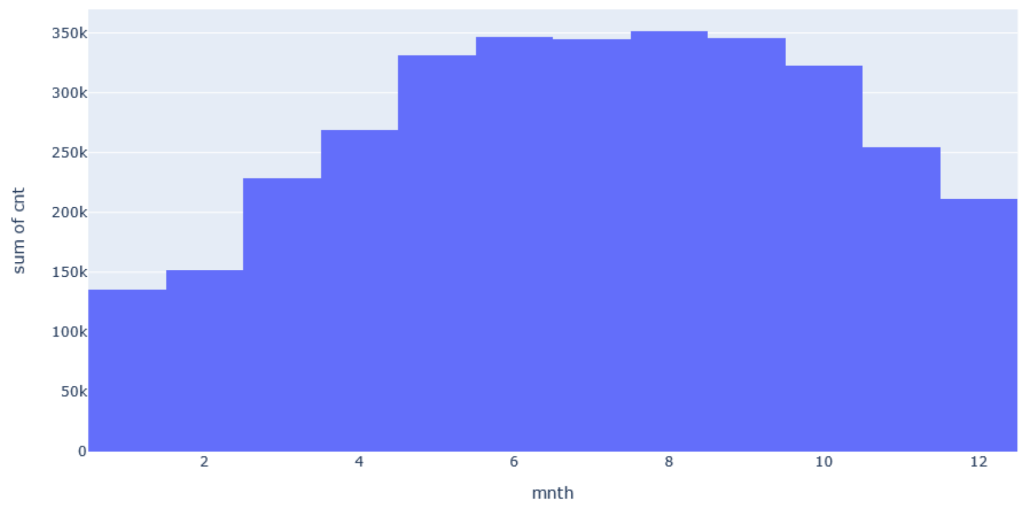


Figure 1.8: Count of Bike share in every month

The figure 1.8 depicts the highest count of bike sharing was recorded during the month of August and the least were recorded during January.

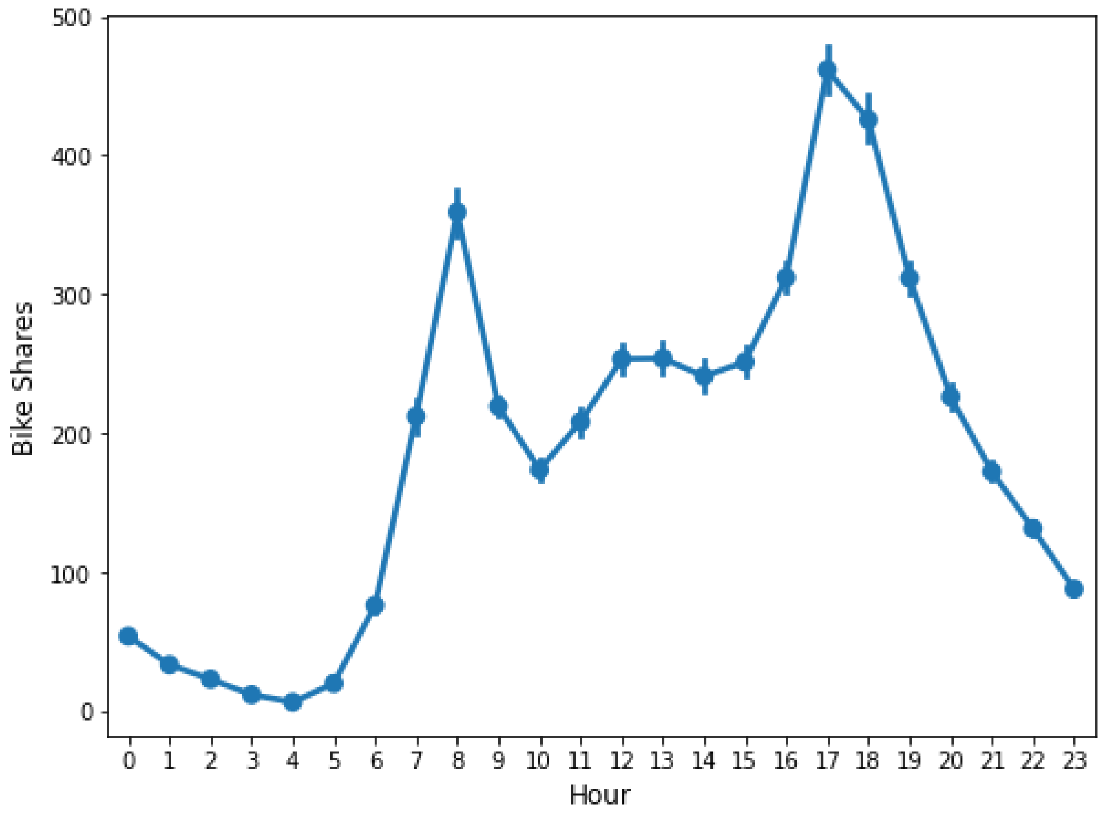


Figure 1.9: Count of Bike share in every hour

The figure 1.9(line plot) shows that the count of bike shares have been observed more during morning peak hours at 8 hrs and hits peak during the evening peak hours at 18 hrs of the day. Therefore, leveraging the CRISP-DM standards the data is analysed in python 3.8 using visualization libraries such as matplotlib, seaborn and plotly express before the modeling part which is the backbone of any data science or machine learning approaches.

1.2 Prediction model for predicting bike sharing

To predict the hourly utilization of the bike, three models were compared to conclude with the best suitable model**(Refer Section 1.5)** for the business use case.The business use case considered here is the time series analysis of Bike sharing.

1.3 LSTM for Prediction

As a baseline model for neural networks, initially LSTM was preferred. The dataset was preprocessed, split into train,test and also the required layers were added. LSTM(Long short term) model was used to fit on the train and evaluated on the test dataset with the choice of drop-out layer and with 100 epochs. The model resulted with R^2 score with 69% and Root mean square Error value of 122.88 and Mean Absolute Error value 97.05 for test while the Root mean square Error and Mean Absolute Error was computed for training set as 0.151 and 0.097 and R^2 value with 95%. It is inferred from this that the model is overfitting because it performed well with training but significantly bad on test data. This could be due to the reason that the model uses a small-to-medium dataset. In scenarios, where the data is scaled up with large number of samples LSTM will perform better.

1.4 SARIMA for Prediction

To further search for the most appropriate model for the dataset, Seasonal Auto Regression Integrated Moving Average(SARIMA) model which is a traditional time series technique was leveraged for the prediction. The reason to choose SARIMA is because the dataset consist of seasonality hence not a stationary dataset. To make the dataset stationary, there is a need for differencing which further leads to loss of data. To avoid the loss of data, the SARIMA is used because the model considers the fact that the dataset consist of seasonality and still results in good accuracy.

To check for stationarity, Augmented dicky fuller test was used that resulted with p-values 0.34 which is greater than 0.05, so it can be concluded that series is non-stationary. To check for seasonality in data, decomposition of the time series data in to trend, seasonal and residual was done using seasonal decompose of statsmodel.

The decompose plots in figure 1.10 shows no strong upward or downward trend which infers that there is no trend but there is a seasonality that is repeating every year. There are few parameters which are really important to SARIMA, they are order(p,d,q) and seasonal order(P,D,Q). To find the optimal order and seasonal order, Grid Search CV was leveraged and decided on the value of order,seasonal order for which the Bayesian Information Criteria value is minimum after running for predefined set of iterations.

Figure 1.11 shows the the SARIMA predictions for the target column 'cnt' for the date between 2012-12-29 00:00:00 and 2012-12-31 23:00:00. The model resulted with 34.929668 RMSE, 25.998036 MAE value and 73% for R^2.

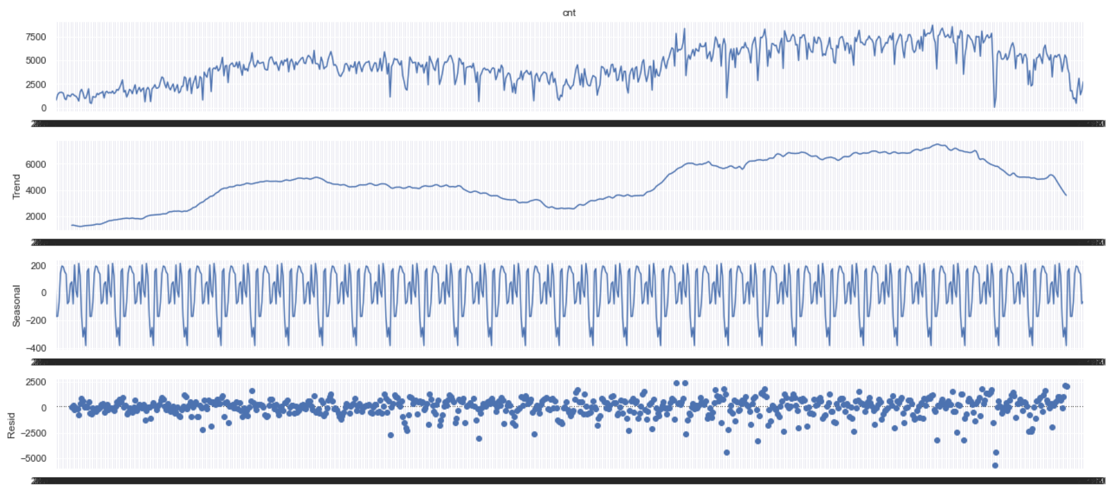


Figure 1.10: Seasonal decompose plot

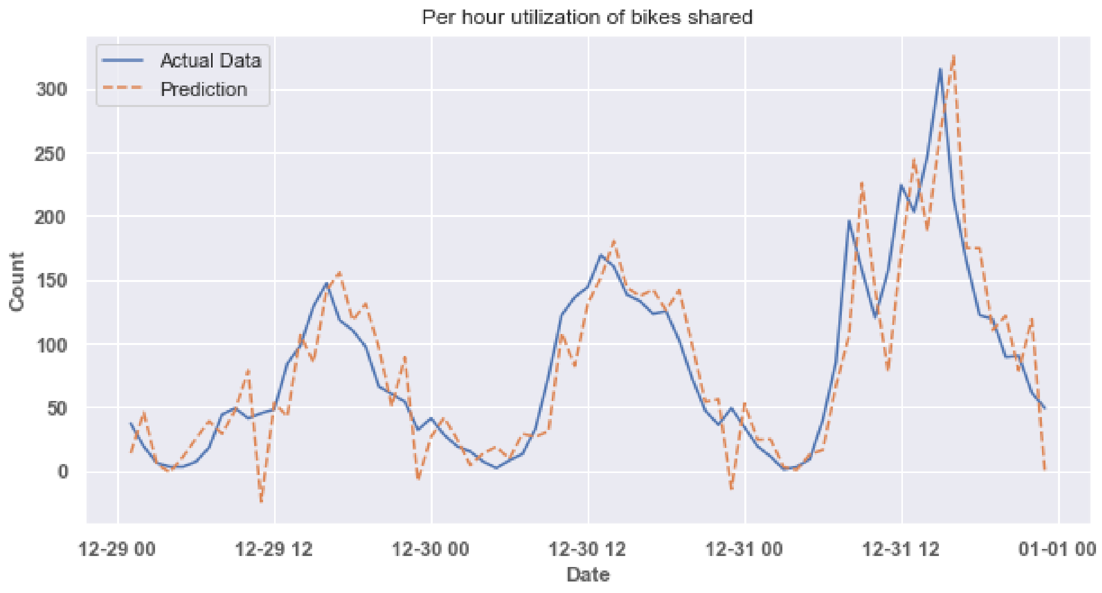


Figure 1.11: Prediction using SARIMAX

The disadvantage of using SARIMAX-Grid Search CV in this use case was that it takes a lot of time to converge and thus consumes a lot of resources and time to find the optimal values for order and seasonal order. Additionally, in this case SARIMAX is used only as an univariate model. Also less data points in the training set risks that the model will not adequately capture the appropriate trend and fluctuations influenced by seasonality in cases of predictions or forecasting.

1.5 XGBoost for Prediction

To predict and compare the performance among the other models for the given business use case, Extreme Gradient Boosting(XGBoost) is experimented, The reason for choosing this model is because it is proves to be a highly efficient algorithm called Stochastic Gradient Boosting. It is an ensemble of decision trees, where the trees fix the error of the previous tree. Using scikit learn's train_test_split the dataset is divided in to train test split.
Table 1.1 shows the most common evaluation metrics used to evaluate the model in test data whose Mean Absolute Error was 36.16, Root Mean Square Error was 56.88 and R^2 value was 89.75% while the RMSE for train prediction was 58.32, MAE was 36.90 and R^2 was 81.85%.
Figure 1.12 shows the prediction plot using XGB algorithm for hourly utilization of the bike sharing dataset.

	Train	**Test**
RMSE	58.32	56.88
MAE	36.90	36.16
R2	0.81	0.89

Table 1.1: Evaluation Metrics of XGB Algorithm

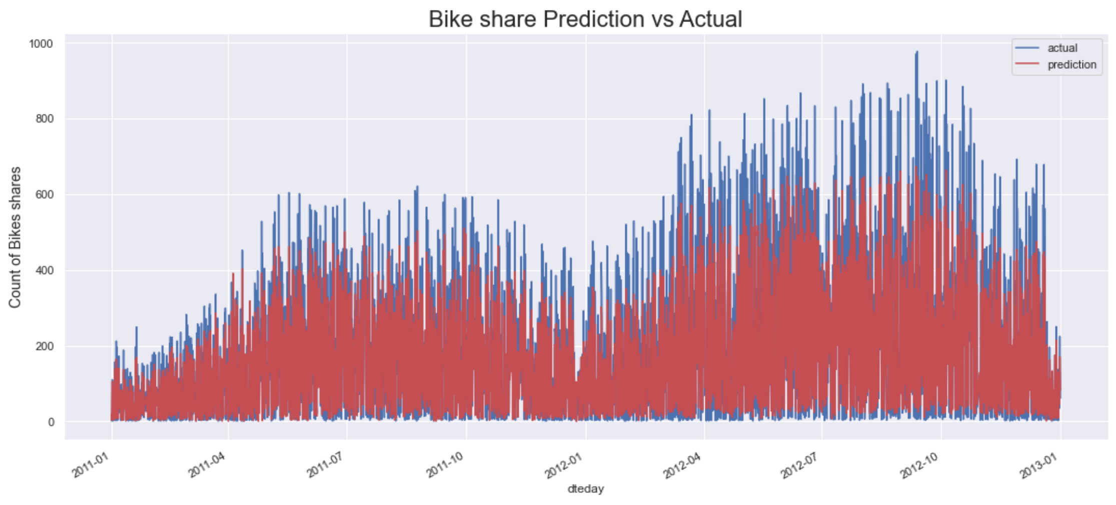


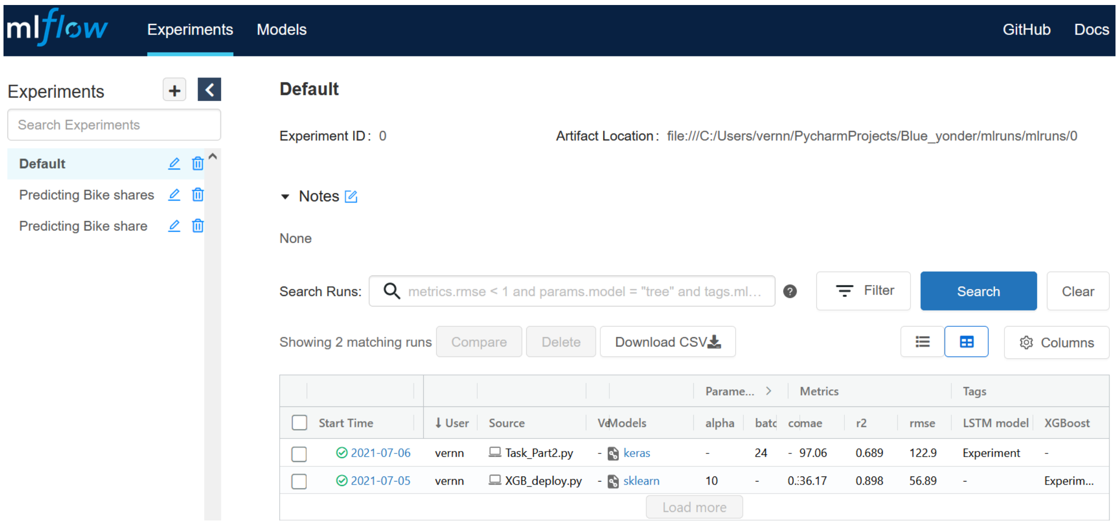
Figure 1.12: Prediction using XGB Algorithm

Analysing the results obtained from test set and train set(Table 1.1), it is evident that the model is able to generalise well and does not overfit. On comparison with all the 3 prediction model, it is clear that XGBoost outperforms the others for the provided data set. The advantage of using XGB boost in this use case is that, it fits well for a small-to-medium dataset. It is also flexible and supports regularization. It handles missing data with its built-in features.

1.6 Continuous Integration and Continuous Deployment

Once the best model is chosen for the business use case, it is pushed to a shared repository where the colleagues can commit their work for better collaboration and software quality, in such way the colleagues can contribute with Continuous Integration(CI) where any defects can be detected and any issues related to software quality.The committed code is immediately run for unit testing and various other test. These automated testing provides immediate feedback to the developer whether the current commit successfully passed the unit testing. The next step of the process is to pass the tested code to the delivery environment using Continuous Deployment(CD). The continuous Deployment helps in pushing the model to the appropriate services such as web servers, API services. The CD also executes continuous testing in the deployment environment and rollbacks when there is a failure in the deployment to the chosen environment.There are various CI/CD tools available that helps in continuous development, test and deployment, they are Git, Jenkins, Circle CI and cloud infrastructure such as AWS Codebuild and Azure DevOps.

In this use case, **"MLFlow" - a deployment API** is used in-order to make the code modular, reproducible and production ready. MLflow also has the benefits of UI Tracking where different machine learning models can be stored and tracked on the server as seperate runs or experiments. Finally, the model to be deployed can be selected and passed on to testing, staging environments. It can be coupled with different platforms like Sklearn, Keras, Tensorflow(**Refer implementation of the attached .py file**) which takes care of end-to-end ML projects[1] [2]. Figure 1.13 shows the snapshot of the MLFlow(deployment API) implementation where the models are logged along with parameters, metrics, artifacts, tags.



Experiment ID: 0

Artifact Location: file:///C:/Users/vernn/PycharmProjects/Blue_yonder/mlruns/mlruns/0

Notes

None

Search Runs: metrics.rmse < 1 and params.model = "tree" and tags.ml...

Showing 2 matching runs

					Parame...		Metrics			Tags	
Start Time	User	Source	Ve	Models	alpha	batc	comae	r2	rmse	LSTM model	XGBoost
2021-07-06	vernn	Task_Part2.py	-	keras	-	24	- 97.06	0.689	122.9	Experiment	-
2021-07-05	vernn	XGB_deploy.py	-	sklearn	10	-	0.:36.17	0.898	56.89	-	Experim...

Figure 1.13: Snapshot of MLflow UI Tracking Implementation

[1]https://www.mlflow.org/
[2]The same has been used for a couple of my research projects earlier

2 Task 2

A scaled up data means that the dataset contains many features with increased number of samples resulting in **high dimensionality** leading to a complexities with data. Some of the problems faced due to large dataset are: the training data does not fit on a single machine**(the data storage)**, the volume of the incoming data is too high**(data loading)** and handling of the missing data. Also, **memory usage**, **computation time** and **costs**/budget would be of major issue due to virtual machines.

In the context of **machine learning**, though it can take different directions, models like **LightGBM** can be beneficial as it supports **parallel computation**which may solve the problem of lags, time. It is also an advantage when there are categorical variables in the dataset, the usual approach is to do encoding of variables before feeding to the model whereas LightGBM handles encoding of categorical variables by itself directly. In some cases where no.of.features(p) » no.of.samples(n) the concept of **curse of dimensionality** rises. To address this shortcoming, techniques like **dimensionality reduction**(Principal Component Analysis(PCA), KPCA), **feature extraction and feature selection** can be done depending on the business goal.

In the context of **Big data**, to store the dataset that goes up to terabytes and for deployment of the ML models, many technologies in Cloud services could be useful such as Snowflake, AWS, GCP and other storage services provided by Microsoft are of great benefit. One other big data technology to store such massive amount of data is by using **Apache Hadoop** which can effectively store data in cluster, the data is stored in **Hadoop File System(HDFS)** and can be accessed using Hadoop map reduce. Also, distributed processing engine with map reduce jobs like **Spark** handles the problems of big data and processing of it for predictive analytics.

It is advantageous when the model is on the cloud and can be deployed in any system with cloud access. Monitoring parameters are also readily available. As mentioned above, **cloud solutions** for storing data and building models**(Jupyter notebook)** are flexible and scalable. There are large variety of computing and storing resources which enables deployment of the model on a virtual space shared among several users with the possiblity to grant specific rights for either a single user or group of users. All files and models can be easily retrieved and the modelling steps can be monitored. The **demerit** here would be that to **get an increased budget** due to **virtual machine** and **data storage**. The key could be to use less performant VM'S. One other **drawback** would be, cloud data storage **needs stable internet connection** with high upload and download speed. For modeling, some of the **demerits** when dimensionality reduction is performed is that the data is transformed to a new orthogonal space which may result in **loss of original features**. Also, LightGBM is **based on bias algorithm** and so it may be **sensitive to noise**. It may result in **overfitting** when grown into a deeper decision tree.

I have completed hands on courses on Spark, Hadoop and some experience with AWS. In particular, AWS Sagemaker, ML/Deep learning libraries, Neural Networks and Rest API, GIT are my daily go-to technologies. I have worked on some projects and used them in my academic research work for a period of about one year.