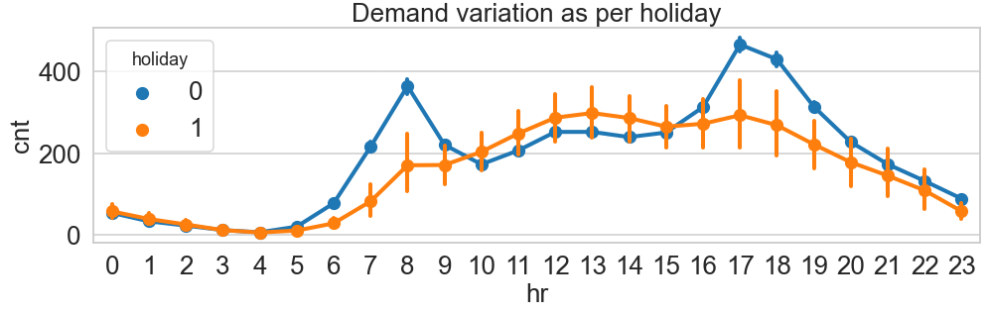
# Part-1: Bike Rental Dataset Data Exploration for

|  |  |
| --- | --- |
| Data URL | <https://archive.ics.uci.edu/ml/machine-learning-databases/00275/Bike-Sharing-Dataset.zip> |
| Problem Type | Time series, forecasting, data analysis |
| GitHub Repository | <https://github.com/ikespand/pyexamples> (sub folder **\data-science**) |
| Author | Sandeep Pandey |
| Files in repository | * **data\_parser.py** has data loader, parser and cleaner class * **exploration\_model.ipynb** contains a notebook to perform data analysis and then training an ML model |

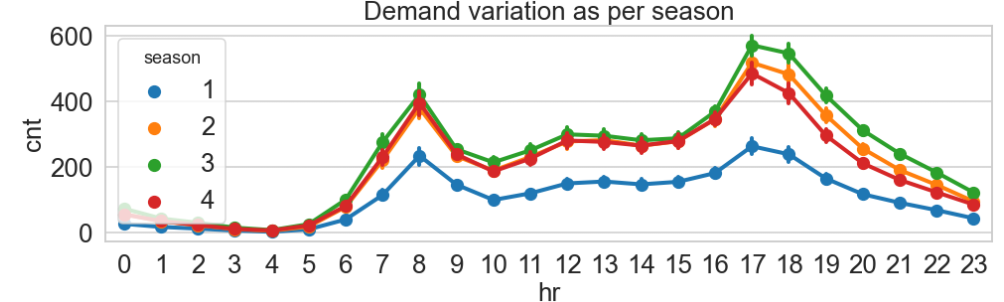
Problem Statement: The given problem is related to a time-series, where the end goal is to predict the future demand of bicycle rental. Here, we target to predict the total bicycle count (i.e. the summation of casual and registered in given dataset).

Data Analysis:

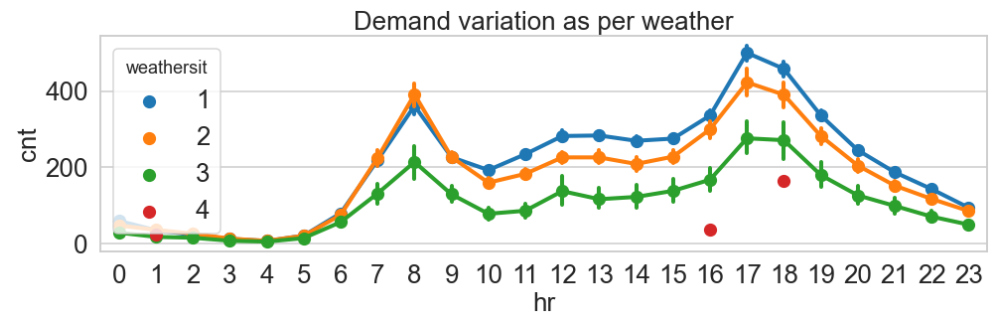
**Effect of holidays** on rental can be seen from Figure 1, where one can observe that demand shows 2 peak during the weekday around 8 AM and 5 PM, which is expected due to the fact that it is most common time to reach/leave offices. While, demand shows a peak around the afternoon during the holiday which is again expected due to leisure activities are often planned during this time.



**Figure 1**: Variation of bike rental vs. the daily hours. Legend- 0: working day, 1: holiday.

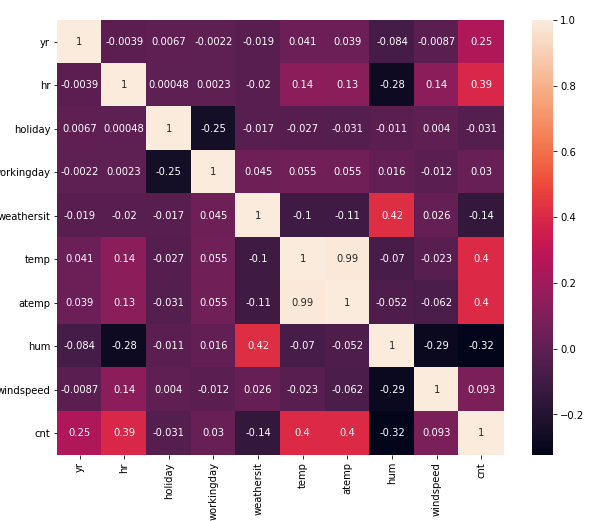


**Figure 2**: Variation of bike rental vs. the daily hours. Legend- 1: spring, 2: summer, 3: fall, 4: winter



**Figure 3**: Variation of bike rental vs. the daily hours. Legend- 1: clear, 2: slight rain, 3: moderate, 4: sever

Figure 2 and 3 shows how a season and weather can affect the bicycle rental. When weather becomes sever then demands deeps to almost zero while when weather is pleasant then demands shows their peak. From both the figure, we can also see the dominance of days when there are no holidays.



**Figure 4**: Cross-correlation of all the variables

Figure 4 depicts the cross-correlation among different variables available which can affect the bike demand. One expected outcome is that *temp* and *atemp* has very high correlation, therefore, we can remove either one of it to circumvent multi-collinearity problem. Another observation is that hr, temp affects the most.

## Machine learning model:

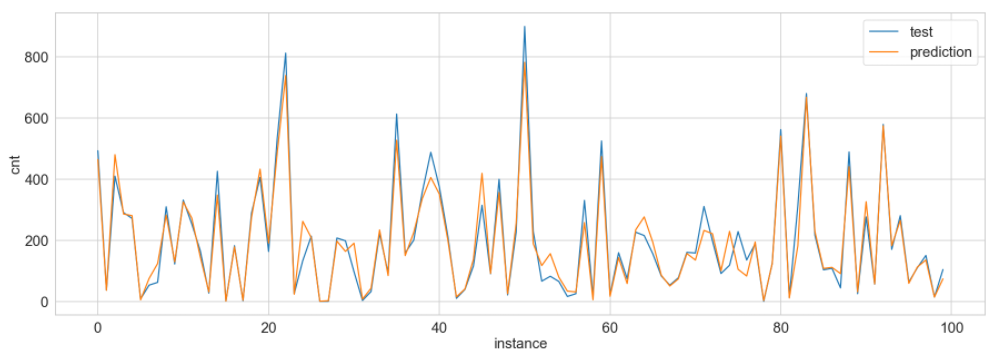
The problem can be solved with 2 different methods. One is by assuming it purely as a time-series and using **RNNs** to capture the trend to predict next hourly demand. However, for this (limited time) task, solving it as a regression seems to be most optimal and time efficient. In this case, it is safe to assume that all other variables are available to predict the demand at that instance. For instance, some variables are easy to determine e.g. holiday, day of the week and season. Some variables are limited but suitable forecast are available for them e.g. weather, temperature, humidity etc. Typically, these are available for 14-days forecast and have high accuracy for bigger cities. For the same, any weather api can be subscribed (world weather online, open weather api etc.).

As, tree based model generally yields a good outcome, therefore, we focused upon the random forest and used *scikit-learn* for the same. As a data-processing step, we already have dropped *atemp* and also converted categorical feature by using *OneHotEncoder* even they are not necessary in tree based method. But, it will allow us to try new model like deep learning algorithms without too much data processing. Data scaling and normalization were skipped, again because tree-based algorithm can handle the data. However, if we want to try out deep learning or other distance based algorithm then it will be necessary. The code to develop ML model is a part of notebook **exploration\_model.ipynb**.

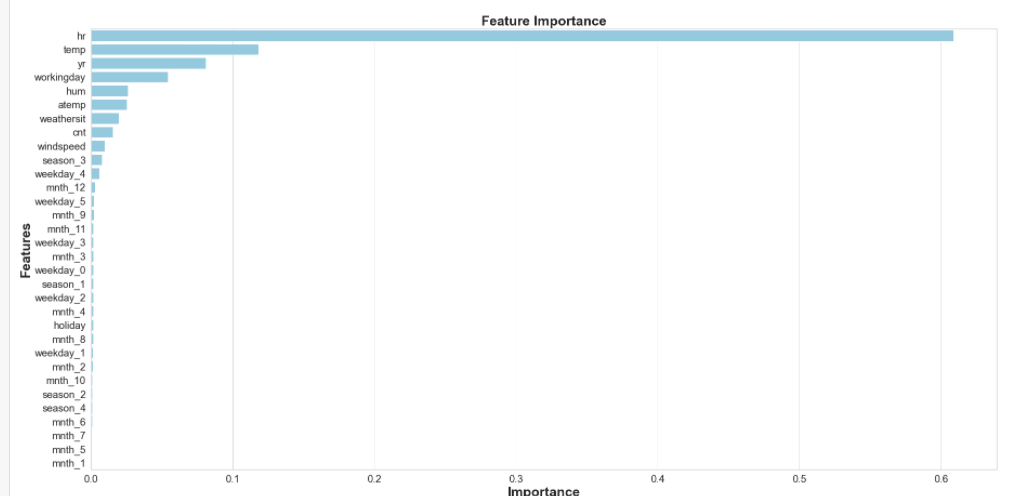
We splitted test dataset with 25% split ration. After a brief hyperparameter tuning, we came with the model parameter. Here remains, a great scope to improve model by using sophisticated tuning methods such as grid search, random search or Bayesian optimization. Table 1 shows the root mean squared error (RMSE) and mean absolute deviation (MAD). Figure 5 shows a sample visual comparison of test data vs prediction by our model. It can be seen that model able to predict the demand with a good accuracy along with that, it can predict the trend. This warrants that model is not completely overfitting. However, there remains a scope to improve it which is beyond the scope of this work. Figure 6 shows the feature importance.

|  |  |  |
| --- | --- | --- |
|  | RMSE | MAD |
| Train | 16.33 | 9.75 |
| Test | 43.16 | 25.97 |

**Table 1**: Model performance



**Figure 5**: A sample comparison of prediction from the model vs the actual values from the blind test

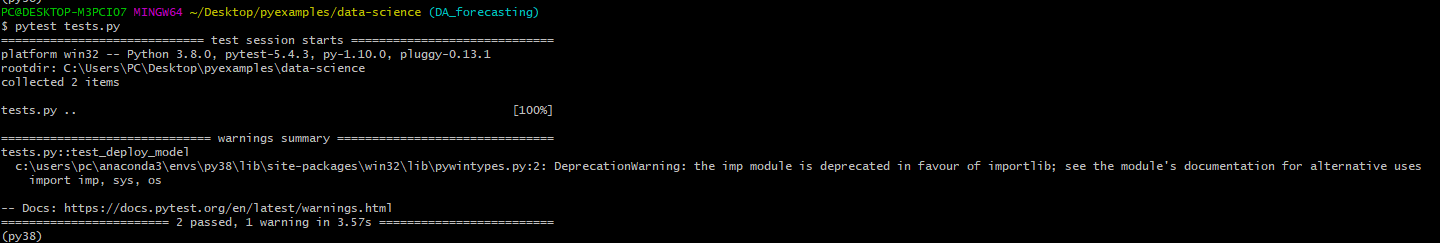


**Figure 6**: Feature importance extracted from RandomForest for given dataset

**Use of code in production in a daily prediction service**:

* Assuming model need not be retrained like an online training and the purpose is to come with a solution by which model can be deployed and work as black box. For this purpose, we have 2 solutions depending on the demand. If the demand is low and maybe we need to predict only few times in a day, then the most optimal solution will be batch job which can be auto-triggered or manual triggered.
* However, if demand is very high in which we need to do many predictions in a day and they are required very often. Then, using a dedicated server would be a better option. Here, a REST API can be build using flask or FASTAPI which will get all parameters as query parameters and our model will predict a provide the response in real-time. This service can be scale horizontally.
* A sample script is provided here which simply focused on model deployment in **deploy\_model.py**, which can be distributed to the deployment team.

**Testing:** For the simplicity of the task, few tests are implemented in tests.py file. Following is the output:



**Figure 7**: Outcome from the tests