**Bikes Rental Forecasting using Machine Learning models**

The objective of the project is - using historical usage patterns and weather data, forecast(predict) bike rental demand (number of bike users (‘cnt’)) on hourly basis.

Approach:

“Bikes Rental” dataset was used to predict the bike demand (bike users count - 'cnt') using various models (ML algorithms) and the values of the performance measures for different models were calculated. The model that performs best was finalised, and using Grid-search fine-tuning technique, best possible combination of hyperparameters for the model were selected. Lastly, the selected model was used to make final predictions and the values of various performance measures were calculated.

Following steps were performed:

1. Importing the libraries
2. Loading the data
3. Cleaning the data
4. Adding derived features
5. Dividing the dataset into training and test dataset
6. Training several models and analyzing their performance
7. Fine tuning the model
8. Evaluating final model using test dataset
9. Analyzing the residuals

**1. Importing the libraries**

Following libraries were initially imported into the environment

1. **numpy:** np
2. **pandas:** pd
3. **sklearn** - preprocessing, linear\_model, StandardScaler, mean\_squared\_error
4. **matplotplib.pyplot:** plt
5. **os**

### 2. Loading the data

The dataset was loaded from a csv which is present on the shared drive at the location: (Location: /cxldata/datasets/project/bikes.csv)

The dataset had the following parameters:

(total 17 columns):

1. instant: record index
2. dteday : date
3. season : season (1:springer, 2:summer, 3:fall, 4:winter)
4. yr : year (0: 2011, 1:2012)
5. mnth : month ( 1 to 12)
6. hr : hour (0 to 23)
7. holiday : weather day is holiday or not (extracted from [Web Link])
8. weekday : day of the week (0 to 6; 0 - Sunday, 6 - Saturday)
9. workingday : if day is neither weekend nor holiday is 1, otherwise is 0.
10. weathersit :

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

1. 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)
2. 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)
3. hum: Normalized humidity. The values are divided to 100 (max)
4. windspeed: Normalized wind speed. The values are divided to 67 (max)
5. casual: count of casual users
6. registered: count of registered users
7. cnt: count of total rental bikes including both casual and registered

The target data set ('y') had only one 'label' i.e. 'cnt'.

### 3. Data cleaning

Some of the attributes were not required for the project. They were: 'instant', 'casual', 'registered', 'atemp', 'dteday'. These columns were dropped.

Some of the numerical columns required scaling. They were: 'temp', 'hum', 'windspeed'. Standard scaler was used to scale the data.

Standard scales the data such that distribution is centered to 0 with standard deviation of 1.

**4. Adding derived features and transforming the data**

Following feature were derived from the raw set of features and added to the dataset:

1. **isWorking:** 1: Is a workingday and not a holiday, 0: Is not a workingday and is a holiday. This feature is added to check weather it is a working day or Holiday.
2. **dayCount:** count of the days from the beginning of the dataset which will help in splitting the data into train and test

### 5. Dividing the dataset into training and test dataset

After analyzing the dataset, the entire dataset was divided into training and test set using train\_test\_split in the ratio 70:30. It uses random sorting and hence the resulting train\_set and test\_set is sorted by daycount.

After splitting the data, we obtained 12165 train instances and 5214 testing instances.

### 6. Training and analyze models:

Models trained and analyzed were as followed:

1. DecisionTreeRegressor
2. LinearRegression
3. RandomForestRegressor

Using Cross-validation, following Metrics were calculated:

1. neg\_mean\_absolute\_error,
2. neg\_mean\_squared\_error

Predicted values were calculated from the classifier using cross\_val\_predict.

Function display\_scores(scores) was defined to print values of mean, standard deviation, and scores to analyze the models.

1. DecisionTreeRegressor(random\_state = 42):

1. for neg\_mean\_absolute\_error:

Mean: 54.6882

Standard deviation: 16.6583

2. for neg\_mean\_squared\_error:

Mean: 88.7806

Standard deviation: 24.6712

2. LinearRegression():

1. for neg\_mean\_absolute\_error:

Mean: 110.4324

Standard deviation: 31.4269

2. for neg\_mean\_squared\_error:

Mean: 141.1902

Standard deviation: 37.5556

3. RandomForestRegressor(n\_estimators=150, random\_state=42):

1. for neg\_mean\_absolute\_error:

Mean: 42.5128

Standard deviation: 15.8799

2. for neg\_mean\_squared\_error:

Mean: 67.1729

Standard deviation: 25.4255

The values of Mean absolute error and Mean squared error are minimum for Random Forest Regressor. Hence, RandomForestRegressor is selected as the model for fine tuning.

### 7. Fine-tuning the model:

Fine tuning the model involves improving the model by finding the best hyper-parameters and features

Different combinations of max\_dept and min\_samples\_leaf and min\_samples\_split were assigned to param\_grid.

The best parameter was calculated using GridSearchCV.

Importance score was calculated for each of the feature.

Following are the results of fine tuning: {'max\_depth': 15, 'max\_features': 10, 'n\_estimators': 150}

### 8. Evaluate Final model on test dataset:

Final model with fine-tuned values was evaluated on test data. Predicted values were calculated from the model and store it in 'predictedCounts\_test'

Mean squared error was calculated using mean\_squared\_error function which was 39.3346

### 9. Analyzing the residuals

The difference between the actual and the calculated values was analyzed by plotting histogram