Prepared by : DharmaTeja Chelamalasetty, id : DXC210022 Date : 05/10/2022

BUAN 6341 Applied Machine Learning

ASSIGNMENT NO 1

Seoul Rental Bike prediction

Executive Summary

- **Optimizing Hyperparameters like learning rate and convergence threshold improve accuracy and time taken to converge.**
- **Sensible feature selection and engineering improves prediction accuracy.**
- **Temperature, Functioning Day and Hour are the best parameters to predict the bike rental count .**
- **Prediction accuracy can be further improved by transforming the target variable with a function form to treat homoscedasticity of errors .**

Introduction

In this project, the objectives were to implement a gradient descent algorithm and utilise it to predict the number of bikes rented per hour. This report details the various experiments conducted using the dataset to understand the effect of tuning hyperparameters of the gradient descent algorithm. Also, the effect of the use of various independent variables/ features in prediction is evaluated and the best model was selected through experimentation.

About the Data

The dataset consists of 14 features and 8760 records. The dataset contains weather information (Temperature, Humidity, Windspeed, Visibility, Dewpoint, Solar radiation, Snowfall, Rainfall), the number of bikes rented per hour and date information. In order to remove the serial correlation with time, hours variable is hot encoded and converted to dummies and ran the model .

Project Outline

The Project is outlined to have 3 parts:

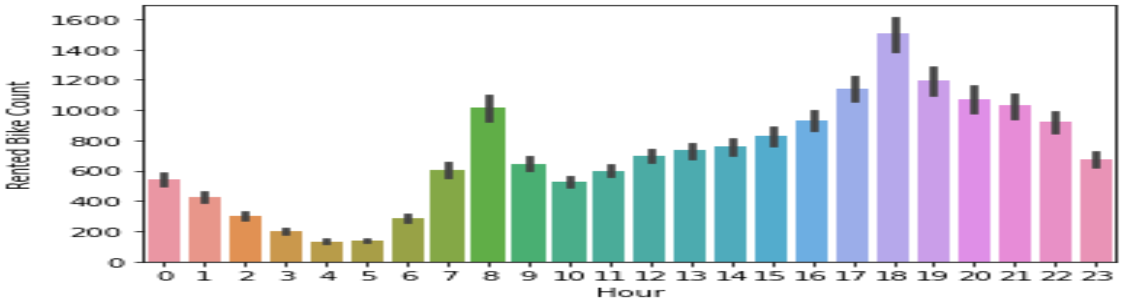
Part 1 : Data Preparation and Exploratory Data Analysis
Part 2 : Use Linear Regression Model for Rental Bike count Prediction and Report equation
Part 3 : Implement the gradient descent algorithm with batch update rule and carry out experiments with different values of i) learning rate, ii) Threshold and report the best fit values, iii) Pick random 8 features and train model with best hyper parameters , iv)

Hand pick best 8 features based on our assumption and train model with best hyper parameters .

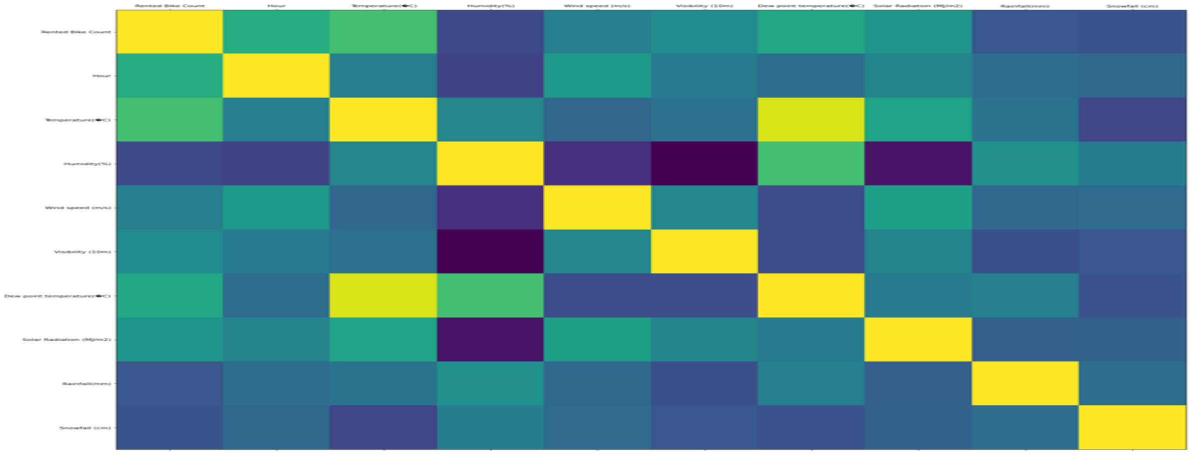
Part 1:

Data Preparation and Exploratory Data Analysis

The objective of the linear regression model is to accurately predict the predict the number of bikes rented per hour.



In order to remove the serial correlation of errors w.r.t time , hour variable is converting to dummies using hot encoding .



From the above correlation matrix we can see that the features **Temperature(�C)** and **Dew point temperature(�C)** are highly correlated , so to avoid the multi collinearity issue we will dropping **Dew point temperature(�C)** feature from our data set to effectively predict the rental bike count .

Seasons feature is also converted to dummies using hot encoding .

Holiday and Function Day are ordered data converted to numeric values by replacing yes and no with 1 and 0 respectively .

Data is converted to extended Feature as Separated with Date, Month, Week etc .

Part 2:

Linear Regression Equation :

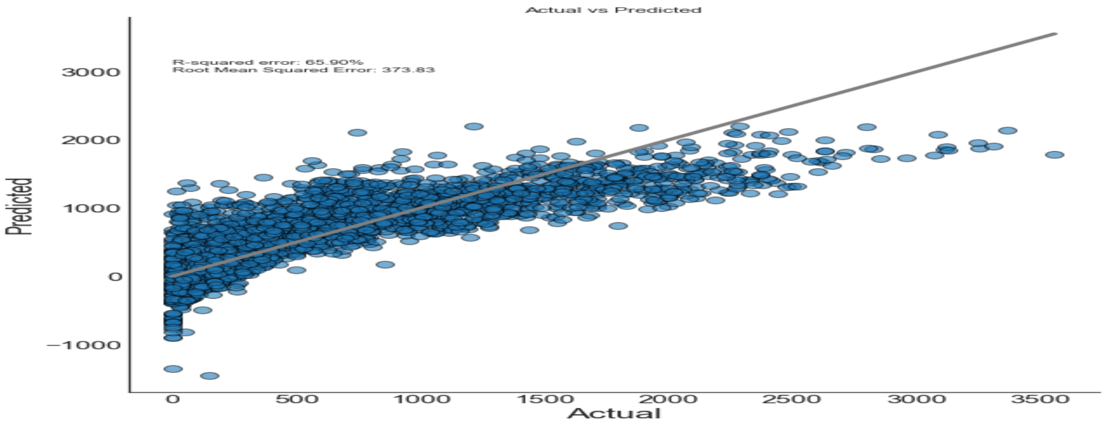
RentedBikeCount = B0+B1*Temperature(�C)+B2*Humidity(%)+B3*Wind speed (m/s)+B4* Visibility (10m)+B5*Solar Radiation (MJ/m2)+B6*Rainfall+B7*Snowfall+B8*Holiday+B9* FunctioningDay+B10*Year+B11*Month+B12*Day+B13*WeekDayEncoding+B14*Seasons_

Spring+B15*Seasons_Summer+B16*Seasons_Winter+B17*Hour_1+ B18*Hour_2+ B19*Hour_3+ B20*Hour_4+ B21*Hour_5+ B22*Hour_6+ B23*Hour_7+ B24*Hour_8+ B25*Hour_9+ B26*Hour_10+ B27*Hour_11+ B28*Hour_12+ B29*Hour_13+ B30*Hour_14+ B31*Hour_15+ B32*Hour_16+ B33*Hour_17+ B34*Hour_18+ B35*Hour_19+ B36*Hour_20+ B37*Hour_21+ B38*Hour_22+ B39*Hour_23

- i) **Using Linear Regression of SCIKIT Learn Package:**

RentedBikeCount = 706.63+ 268.75⋆Temperature(C)+-133.00⋆Humidity(%)+1.42*Wind speed (m/s)+3.36* Visibility (10m)+67.79*Solar Radiation (MJ/m2)-72.27*Rainfall+7.14*Snowfall-29.28*Holiday+174.88* FunctioningDay-28.98*Year-0.67*Month-12.76*Day-11.91*WeekDayEncoding-77.56⋆Seasons_Spring-65.48*Seasons_Summer-177.93*Seasons_Winter-18.75*Hour_1-42.83*Hour_2-56.10*Hour_3-68.88*Hour_4-67.11*Hour_5-36.78*Hour_6+23.26*Hour_7+87.08*Hour_8+ 6.73*Hour_9-41.57*Hour_10-43.89*Hour_11-39.30*Hour_12-36.7*Hour_13-37.7*Hour_14-20.56*Hour_15+10.41*Hour_16+63.55*Hour_17+151.95*Hour_18+ 108.42*Hour_19+90.44*Hour_20+88.59*Hour_21+67.04*Hour_22+22.74*Hour_23

Error :- 139747.364



The Above Curve indicates that errors are not linear and indicates heteroskedasticity to some extent .

- ii) **Using Gradient Descent Implementation batch rule:** At alpha = 0.1 and threshold : 1e-3

  RentedBikeCount = 706.63+ 268.58⋆Temperature(C)+-132.97⋆Humidity(%)+1.41*Wind speed (m/s)+3.33* Visibility (10m)+68.16*Solar Radiation (MJ/m2)-72.25*Rainfall+7.14*Snowfall-29.28*Holiday+174.89* FunctioningDay-29.03*Year-0.70*Month-12.77*Day-11.92*WeekDayEncoding-77.63⋆Seasons_Spring-65.47*Seasons_Summer-178.04*Seasons_Winter-19.22*Hour_1-43.30*Hour_2-56.57*Hour_3-69.37*Hour_4-67.59*Hour_5-37.27*Hour_6+22.27*Hour_7+86.57*Hour_8+ 6.18*Hour_9-42.14*Hour_10-44.49*Hour_11-39.93*Hour_12-37.33*Hour_13-38.32*Hour_14-21.15*Hour_15+9.85*Hour_16+63.05*Hour_17+151.46*Hour_18+ 107.94*Hour_19+89.98*Hour_20+88.12*Hour_21+66.58*Hour_22+22.27*Hour_23

Train Error :- 70280.99

Test Error :- 68060.76

From above we can see that the coefficients are almost same and using gradient descent batch rule we get low Mean Square Error when compared with Scikit Learn Package

Part 3:

Algorithm Implementation :

The gradient descent algorithm is implemented using python numerical computation package “numPy”. The numPy package provides new homogenous array and matrix data structures to python which is immensely beneficial to implement vectorized implementation of the gradient descent. The algorithm is implemented with options to change the hyperparameters: Learning rate, convergence threshold, and max epoch.

Experimentation and Results

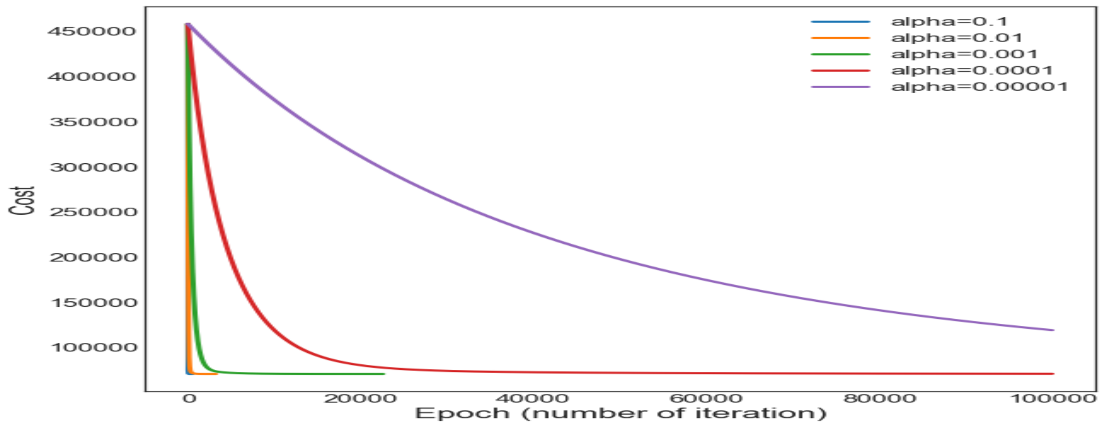
Mainly 4 experimentation was undertaken using the dataset in which we explored the effect of changing the hyperparameters of the algorithm and discuss the effectiveness of feature selection in predictive accuracy. Finally, the results of the experiments were discussed in the results section and further improvement opportunities were discussed.

Experimentation

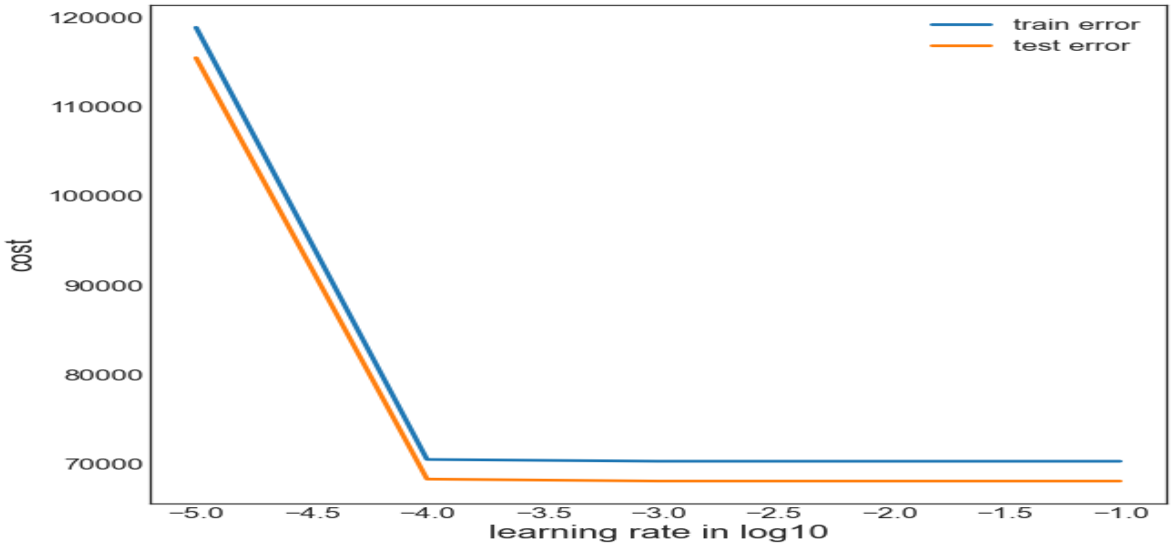
1. Hyperparameter tuning

In linear regression implemented using gradient descent algorithm, the hyperparameter we can tune is learning rate (∝). The value of learning rate has to be tuned for optimal convergence to estimates (β values).

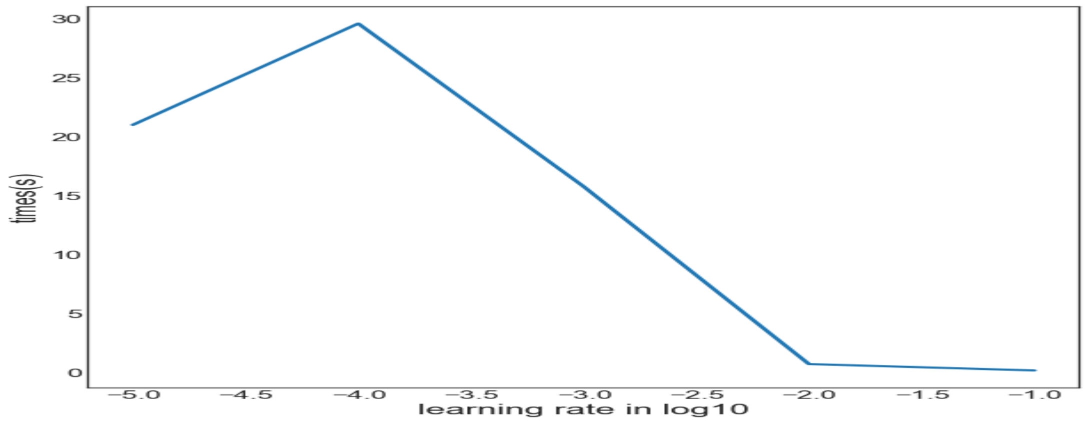
RentedBikeCount = B0+B1*Temperature(C)+B2*Humidity(%)+B3*Wind speed (m/s)+B4* Visibility (10m)+B5*Solar Radiation (MJ/m2)+B6*Rainfall+B7*Snowfall+B8*Holiday+B9* FunctioningDay+B10*Year+B11*Month+B12*Day+B13*WeekDayEncoding+B14*Seasons_ Spring+B15*Seasons_Summer+B16*Seasons_Winter+B17*Hour_1+ B18*Hour_2+ B19*Hour_3+ B20*Hour_4+ B21*Hour_5+ B22*Hour_6+ B23*Hour_7+ B24*Hour_8+ B25*Hour_9+ B26*Hour_10+ B27*Hour_11+ B28*Hour_12+ B29*Hour_13+ B30*Hour_14+ B31*Hour_15+ B32*Hour_16+ B33*Hour_17+ B34*Hour_18+ B35*Hour_19+ B36*Hour_20+ B37*Hour_21+ B38*Hour_22+ B39*Hour_23



With each epoch, the cost is seen continuously decreasing and the algorithm is said to have reached convergence when the decrease in cost is within a defined threshold. When the learning rate is low, the algorithm takes a large number of iterations to converge while the estimates can be more precise. When learning rate is very high, the algorithm might diverge and never reach the minimal point. At Alpha 0.0001 and at 0.00001 algorithm never reaches to global minimum even at 100000 iterations .



The figure shows train & test Cost as a function of learning rate(log). As we can observe, when learning rate is very low (∝ = 0.00001), the cost is higher. This is because of slow learning, even at 100000 iterations the algorithm did not converge within a tolerance value. The effect of changing tolerance value can be seen in a later experiment. At learning rate ((∝ = 0.0001) even though algorithm didn't converge to global minimum after 100000 iterations but it is close to global minimum . We can see that from 0.1 to 0.0001 the global minimum for cost function is almost same (very less value of change is observed) but nr of iterations are more . We can also say that algorithm for learning rate between range 0.7~0.0001 the cost function change is very minimal and after 0.7 the learning rate fails to reach global minimum .



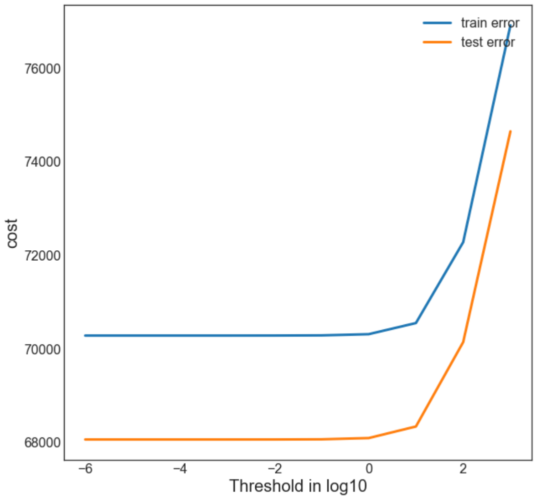
Above is the time complexity of the algorithm w.r.t learning rate(log10) . It is interesting to see that at learning rate = 0.0001 it took more time to complete the run when compared with 0.00001, But in general we can say that time complexity increases with slow learning rate values .

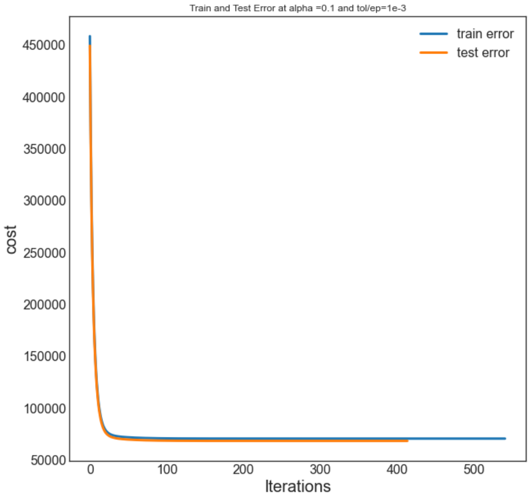
Conclusion :

- **For ideal learning rate, the algorithm converges fast to the minimal point.**
- **When learning rate is very low(~0.00001), the cost decreases nearly linear and takes more than 100000 iterations to converge. When learning is high (>0.7), the algorithm does not converge to a minimal point.**
- **The best value of alpha for linear regression is found to be in range 0.001 and 0.1 respectively.**

2. Changing convergence threshold

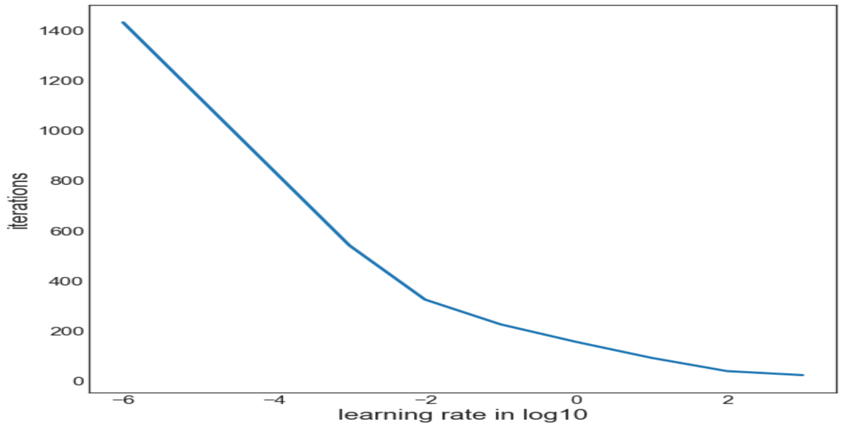
When the Change in Cost is within the Convergence threshold, the gradient descent algorithm is said to be converged. So, the algorithm converges faster at higher threshold value . We find the optimal value of threshold by plotting cost as a function of convergence threshold and find the point at which the test error is minimum.





Threshold	Train Error	Test Error
1.00E-06	70280.87143	68060.71953
1.00E-03	**70280.99707**	**68060.76926**
1.00E-02	70281.72163	68061.14238
1.00E-01	70285.13553	68064.16704
1.00E+00	70310.96398	68090.70301
1.00E+01	70549.20149	68338.30474
1.00E+02	72277.20655	70141.28844
1.00E+03	76911.5933	74646.56919

From above we can say that ideal threshold value is 1e-3 after that the error remains almost constant for both train and test .



We can see from above graph is that when we decrease the threshold it takes more iterations to reach for convergence .

Conclusion :

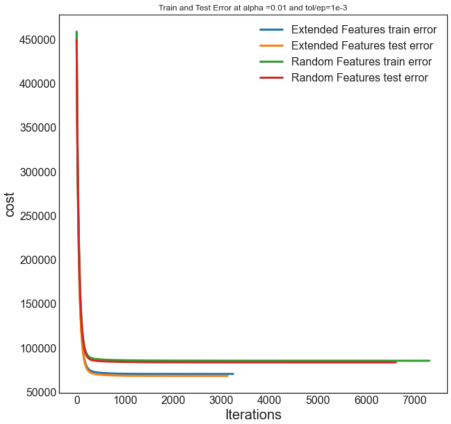
- **Tuning convergence threshold improves accuracy of prediction and convergence threshold of minimum error .**
- **The cost and learning of the algorithm exponentially decrease with iteration and therefore trade of time and accuracy can be an important consideration.**

3. Random Feature Selection

In this experiment, 8 features are selected at random using the DataFrame.sample function and the train and test errors are compared to the errors from the original set of features .

RentedBikeCount = B0+B1*Holiday+B2* Snowfall+B3* FunctioningDay+B4*RainFall+B5*Season_Spring+B7*Season_Summer+B8*Season_Winter +B9* Hour_1+ B10*Hour_2+ B11*Hour_3+ B12*Hour_4+ B13*Hour_5+ B14*Hour_6+ B15*Hour_7+ B16*Hour_8+ B17*Hour_9+ B18*Hour_10+ B19*Hour_11+ B20*Hour_12+ B21*Hour_13+ B22*Hour_14+ B23*Hour_15+ B24*Hour_16+ B25*Hour_17+ B26*Hour_18+ B27*Hour_19+ B28*Hour_20+ B29*Hour_21+ B30*Hour_22+ B31*Hour_23+B32*Year+B33*Month+B34*Day+B35*WeekDayEncoding

The train and test error have been increased by 17.5% and 18.3% respectively



	Original Features	Random Features	% Change
Train	70281.73173	85206.78469	17.51627293
Test	68061.15016	83391.27977	18.38337253

Conclusion :

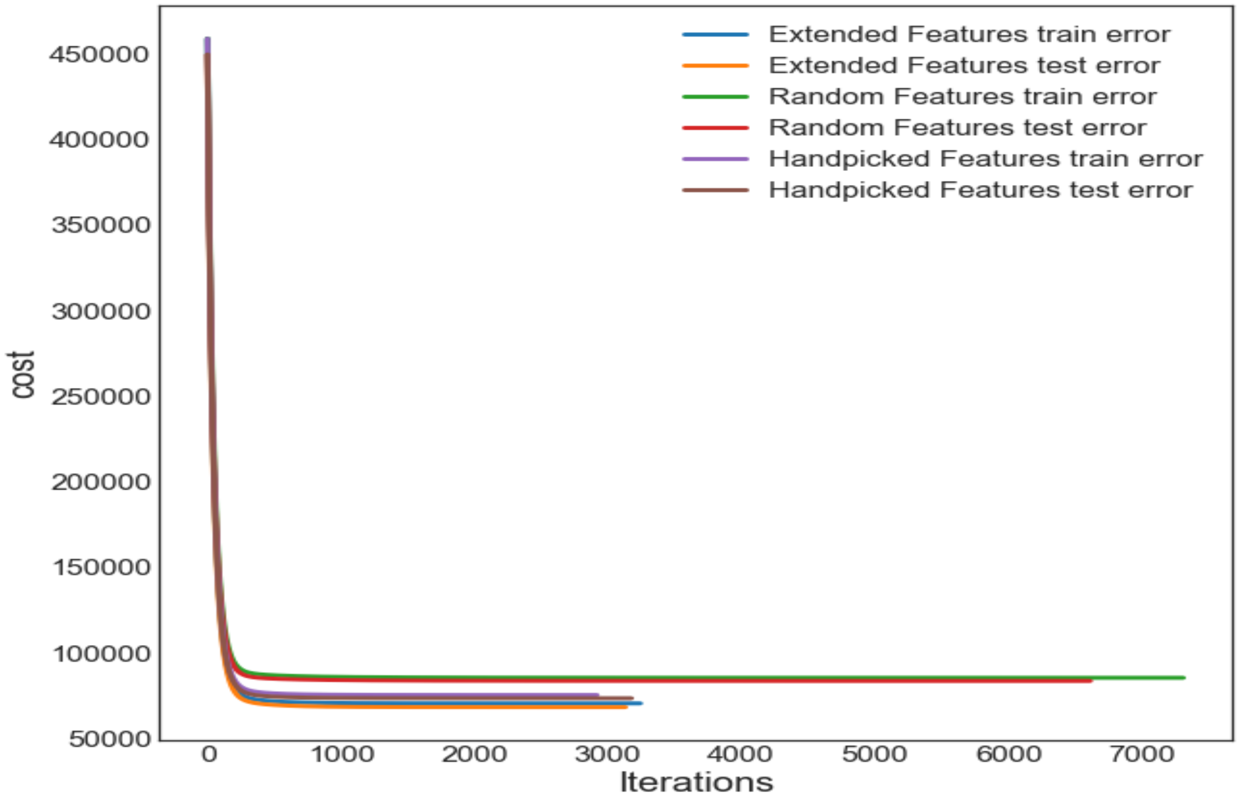
- **Random feature selection has increased the train and test errors by 17.5% and 18.3% respectively .**
- **It looks we need clear knowledge of business understanding and requires little bit of SME .**

4. Manual Feature Selection :

So Based on our knowledge like using some common understanding of nature we selected 8 features for algorithm ,and will compare the train and test errors with original and random feature method .

RentedBikeCount = B0+B1*Temperature(C)+B2*Solar Radiation (MJ/m2)+B3*Rainfall(mm)+B4*Snowfall (cm)+B5*Functioning Day+B6*Holiday+B7*Seasons_Spring+B8*Season_Summer+B9*Season_Winter+B10*Hour_1+B11*Hour_2+B12*Hour_3+B13*Hour_4+B14*Hour_5+B15*Hour_6+B16*Hour_7+B17*Hour_8+B18*Hour_9+B19*Hour_10+B20*Hour_11+B21*Hour_12+B22*Hour_23+B23*Hour_14+B24*Hour_15+B25*Hour_16+B26*Hour_17+B27*Hour_18+B28*Hour_19+B29*Hour_20+B30*Hour_21+B31*Hour_22+B32*Hour_23

The train and test error is found to increase by 6.5% and 7.11% respectively from the original set .



	Original Features	Random Features	% Change	HandPicked Features	% Change
Train	70281.73173	85206.78469	17.51627293	75170.44567	6.50351
Test	68061.15016	83391.27977	18.38337253	73271.30008	7.11076

Conclusion :

- **Hand Picked feature selection has increased train and test errors by 6.5% and 7.11% respectively**
- **From the analysis we can infer that random selection and handpicked selection method has increased the train and test error when compared with original selection .**
- **Hence we can say that we need SME to conclude on which features to select in order to effectively predict the rental bike count per hour , in order to improve accuracy and to decrease the time complexity in execution .**

Results

Through experimentation, we found that **optimizing hyperparameters** like learning rate and convergence threshold are effective for better accuracy of the prediction. But some accuracy improving measures like low alpha settings or very low convergence threshold increases the time taken for fitting the ML algorithm with the training set. Therefore, machine learning involves a balancing between accuracy and computational limitations.

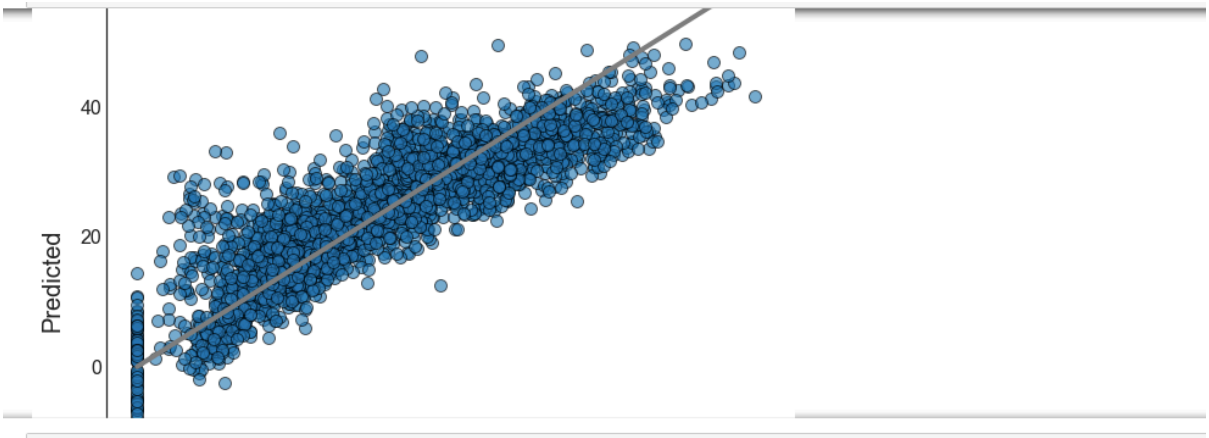
Also, we saw the importance of **feature selection and engineering** to improve the performance of the ML algorithm .

Temp	268.75
Humidity	-133
WindSpeed	1.42
visibiity	3.36
Solar Rad	67.79
Rainfall	-72.27
SnowFall	7.14
Holiday	-29.28
FunctioningDay	174.88
Year	-28.98
Month	-0.67
Day	-12.76
WeekDayEnc	-11.91
Season_spr	-77.56
Season_sum	-65.48
Season_win	-177.9
Hour_1	-18
Hour_2	-42
Hour_3	-56
Hour_4	-68
Hour_5	-67
Hour_6	-36
Hour_7	23
Hour_8	87
Hour_9	6
Hour_10	-41

Hour_11	-43
Hour_12	-39
Hour_13	-36
Hour_14	-37
Hour_15	-20
Hour_16	10
Hour_17	63
Hour_18	151
Hour_19	108
Hour_20	90
Hour_21	88
Hour_22	67
Hour_23	22

From above we can understand that Temp, Functioning Day are high importance features to predict the bike rental count .

Now after the errors we would like to try a functional form of Y to make errors linear so we applied sqrt to Y and ran the algorithm and below is the results .



It is observed that Errors are linear now and train and test errors have been reduced .

Train Error: 19.59629709101611

Test Error: 19.595079634283376