CS771 Project
PREDICT DEMAND FOR RENTING BIKES

Guide: Prof. Harish Karnick
Rishav Raj Agarwal, Ritvik Srivastava, Anusha Chowdhury, Avikalp Kumar Gupta
Group: 35

Abstract. Bicycle sharing programs have emerged as a global trend as an affordable, convenient, and sustainable travel option with various benefits. In this project, we try different Machine Learning techniques on the Kaggle problem: Forecast use of a city bikeshare system, where we combine historical usage patterns with weather data in order to forecast bike rental demand in the Capital Bikeshare program in Washington,DC. After that we did an analysis of why some methods performed a lot better than others.

Keywords: SVM,Mixture model,Poisson Regression,Random Forest,Gradient Boosting.

1 Introduction

Bike sharing systems are a means of renting bicycles where the process of obtaining membership, rental, and bike return is automated via a network of kiosk locations throughout a city. Using these systems, people are able to rent a bike from a one location and return it to a different place on an as-needed basis. Currently, there are over 500 bike-sharing programs around the world. For detailed study on bikesharing we referred to the paper by Ting Ma et. al [3] where a regression analysis has also been done. For the results we were exploring the bike share usage and the variation of the data depending on various factors.

1.1 Problem

In the Kaggle problem [2] on forecasting use of a city bikeshare system, our aim is to predict the number of bikes that may get rented for the test data, after training a model on the train.csv file.
We have to first visualise the dependence of the total bike rental counts on individual members of the feature vector, to gain an insight into the provided data. Since the total bike rental counts is a sum of the casual bike rentals and registered bike rentals, initiated, we study the dependence of the various variables with respect to these individually. It is expected that the behavior of registered and casual bikers vary.

1.2 Data Source

The dataset used is an hourly bike rental data spanning two years from the Capital Bikeshare program in Washington, D.C. provided by Hadi Fanaee Tork and hosted on UCI machine learning repository. The training set comprises of the first 19 days of each month, while the task is to predict the count of rentals for the rest of the month.

2 Data pre-processing

A first glance at the dataset suggests that the date-time field of the feature vector cannot be efficiently used in the original form. The date-time field is split into weekday and hour fields, as they might be useful in further analysis.

3 Data Summary

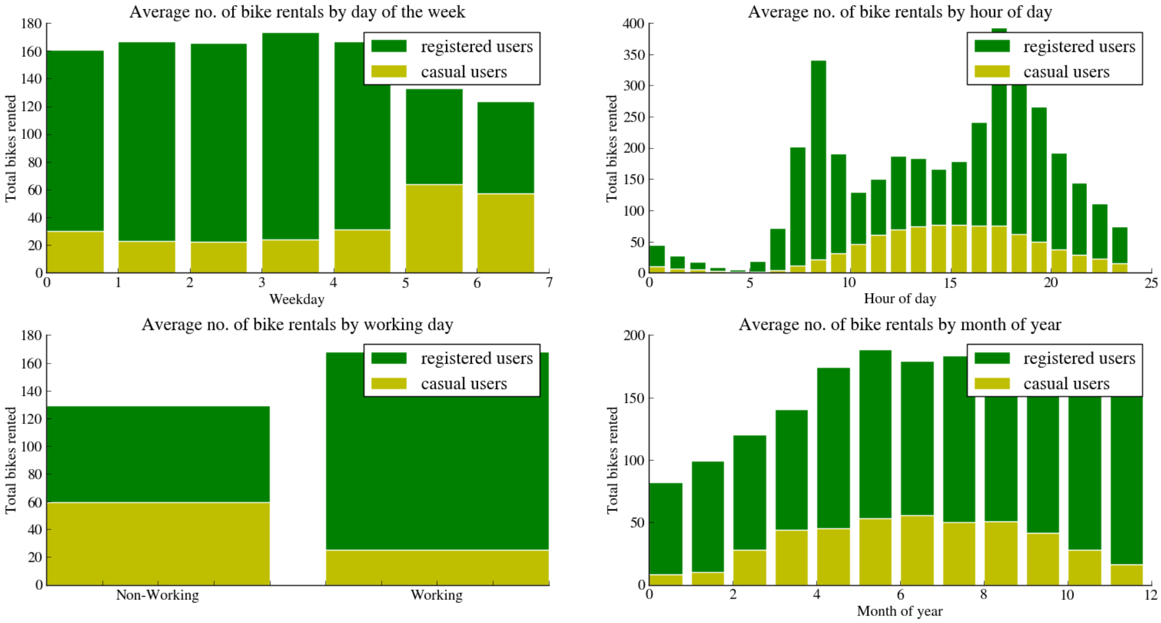
- datetime :- hourly date + time
- season :- 1=spring, 2=summer, 3=fall, 4=winter
- holiday :- whether the day is considered a holiday

- workingday :- whether the day is neither a weekend nor a holiday
- weather :-
  - 1: Clear, few clouds, 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
- temp :- temperature in Celsius
- atemp :- ”feels like” temperature in Celsius
- humidity L:- relative humidity
- windspeed :- wind speed
- casual :- number of non-registered user rentals initiated
- casual :- number of registered user rentals initiated
- count :- number of total rentals

Table 1. The different fields in the data set

registered	season	hrs	workingday	wk	holiday
Min. : 0.0	1:2686	12 : 456	0:3474	Friday :1529	0:10575
1st Qu.: 36.0	2:2733	13 : 456	1:7412	Monday :1551	1: 311
Median :118.0	3:2733	14 : 456	-	Saturday :1584	-
Mean :155.6	4:2734	15 : 456	-	Sunday :1579	-
3rd Qu.:222.0	-	16 : 456	-	Thursday :1553	-
Max. :886.0	-	17 : 456	-	Tuesday :1539	-
-	-	(Other):8150	-	Wednesday:1551	-

Fig. 1. Data Exploration



4 RMSLE

Submissions in Kaggle are evaluated on the basis of Root Mean Squared Logarithmic Error (RMSLE). The RMSLE is calculated as:

$$\sqrt{\frac{1}{n}\sum_{i=1}^{n}(log(p_i+1)-log(a_i+1))^2}$$

Where:
• n is the number of hours in the test set
• p_i is your predicted count
• a_i is the actual count
• log(x) is the natural logarithm

5 Our Approach

We try different techniques to predict the renting of bikes:

- 3-component Mixture Model
- SVM
- Poisson Regression
- Nearest neighbour
- Random Forest
- Gradient Boosted Regression and Classification Trees

We find that Gradient Boosting gives the best results.

6 Feature Selection

⋆ **Hour** has the largest effect on the prediction, followed by **temperature**. The effect of the other predictors appears to be low in comparison and may offer some room for feature selection or transformation for model improvement.
⋆ We found that **holiday** feature was of the least importance, and when we did not take it into account, our accuracy results improved.
⋆ **Weather data** does not have that much importance because even though 4 categories are possible, only 2 are present in train.csv
⋆ If we take the days of the week instead of working day (binary value) then results improve a bit.
⋆ Also we observed that the working day parameter when not taken, does not give much change in result. So, to improve time, we can ignore it.

7 SVM

- Simple SVM (Regression) gives a RMSE of 70.
- Converted the problem into a classification by dividing the count data into High(=3), medium(=2) and low(=1) on the basis of quantiles.
- Quantiles calculated for the train data
  1.Without feature selection, accuracy = 59.91%
  2.With feature selection, accuracy = 80.96 (taking cost = 100)%
- SVM does not work well for multi-class classification.

Observation: We found that the results significantly improve when feature selection is used (the predicted no. of bike counts nearly matches the actual no. of bike counts).

Fig. 2. Graphs in SVM(with and without feature selection

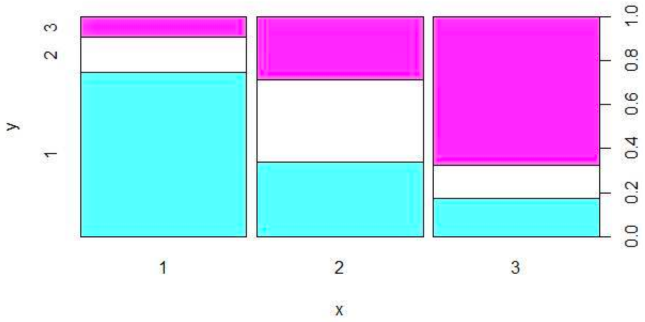
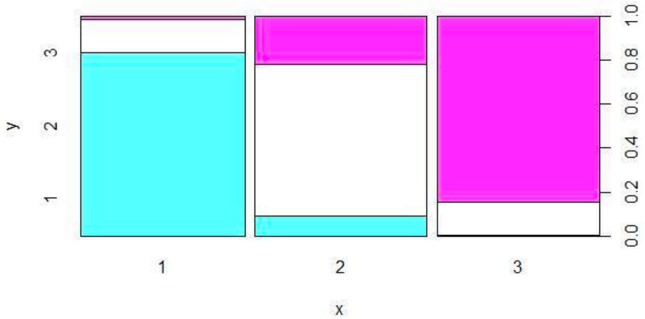


Fig. 3. Data Exploration



7.1 Mixture Model

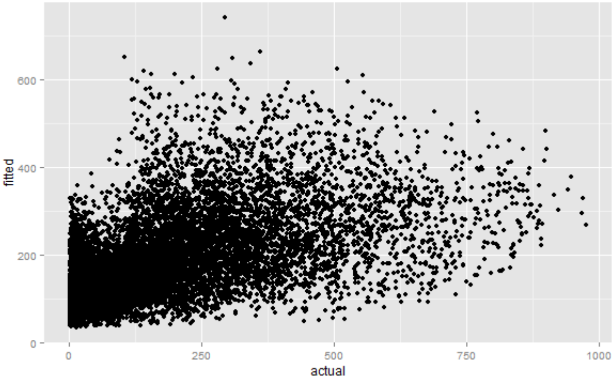
Mixture model is a probabilistic model that can be used in a discrimination problem by modelling each class conditional density as a mixture distribution:
p(θ)=$\sum_{i=1}^{k} \phi_i N(\mu_i, \sum_i)$

- We first constructed a **3-component mixture** model on the training data with 3 classes. For deciding the class boundaries, we used **quantiles** so that classes are of same sample size. We obtained an accuracy of nearly **99**%.
- Next we constructed the three component mixture with 10 classes. Here results with narrower range can be predicted but the accuracy decreases to around **56** %.

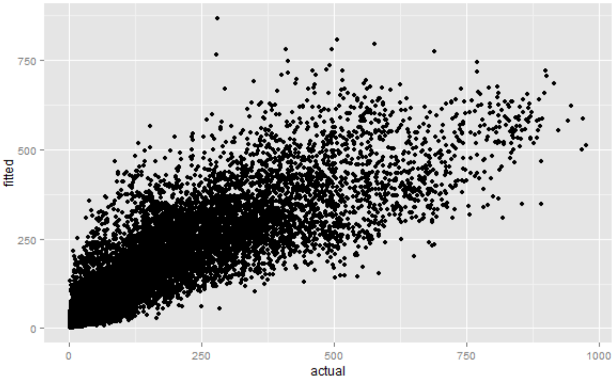
Reason for accuracies:
•When there are only 3 classes then the Bike Prediction is either low(0-99), medium(99-638) or high(above 638). As no. of classes is very low, predictions are not very good because a wide range of values fall within the same class. However, the prediction will be correct in terms of class, so we obtained high accuracy value. Moreover the feature holiday was removed to improve results.
•As we go on increasing no. of classes from 3 to 5 and 10, the accuracy falls off rapidly. However on increasing the no. of components, the accuracies increase as expected.
The training phase consists of estimating the parameters of each mixture model. This is done separately for each class. The code was written in Python using standard multivariate_normal.pdf() method. Conclusion: We should increase number of components along with no. of classes for better prediction capability.

8 Poisson Regression

In statistics, Poisson regression is a form of regression analysis used to model count data and contingency tables. The code was written in R language. Since we are building a regression model with count data, hence using Poisson regression is appropiate.
Results:
1. On normal regression using all the variables:-



Rmsle on train : 1.34566
2. Regression with feature selection:-



Rmsle on train : 0.6559582

Table 2. Data Properties

Min	1 Quarter	Median	3 Quarters	Max
-15.8951	-2.2259	-0.6809	1.2278	19.6127

Signif. codes: 0 *** 0.001 ** 0.01 * 0.05 . 0.1 1 (Dispersion parameter for poisson family taken to be 1)
Null deviance: 564064 on 10885 degrees of freedom
Residual deviance: 110649 on 10848 degrees of freedom
AIC: 157431
Number of Fisher Scoring iterations: 5
Best rank achieved 1929 **Criticism of the model:**
The model assumes that the dependant variable is distributed with a Poisson distribution which is true when:

- The probability of at least one occurrence of the event in a given time interval is proportional to the length of the interval.
- The probability of two or more occurrences of the event in a very small time interval is negligible.

Table 3. The coefficients in the Poisson Regression

Feature	Estimate	Std. Error	z value	Pr(¿—z—)
(Intercept)	1.0592229	0.0185966	56.958	¡ 2e-16 ***
season2	0.5677430	0.0068633	82.721	¡ 2e-16 ***
season3	0.4011676	0.0080671	49.729	¡ 2e-16 ***
season4	0.4510483	0.0063406	71.136	¡ 2e-16 ***
hrs1	-0.4230151	0.0234811	-18.015	¡ 2e-16 ***
hrs2	-0.7134324	0.0260091	-27.430	¡ 2e-16 ***
hrs3	-1.2970998	0.0327842	-39.565	¡ 2e-16 ***
hrs4	-2.0090536	0.0447860	-44.859	¡ 2e-16 ***
hrs5	-1.8412053	0.0416371	-44.220	¡ 2e-16 ***
hrs6	-0.7736095	0.0272702	-28.368	¡ 2e-16 ***
hrs7	0.1604198	0.0203691	7.876	3.3 9e-15 ***
hrs8	0.7660964	0.0177555	43.147	¡ 2e-16 ***
hrs9	1.0474447	0.0168722	62.081	¡ 2e-16 ***
hrs10	1.3657924	0.0162023	84.296	¡ 2e-16 ***
hrs11	1.5551715	0.0159080	97.760	¡ 2e-16 ***
hrs12	1.6375176	0.0158283	103.455	¡ 2e-16 ***
hrs13	1.6651622	0.0158118	105.311	¡ 2e-16 ***
hrs14	1.6757919	0.0158341	105.835	¡ 2e-16 ***
hrs15	1.6722996	0.0158505	105.504	¡ 2e-16 ***
hrs16	1.6718772	0.0158420	105.535	¡ 2e-16 ***
hrs17	1.7258673	0.0157973	109.251	¡ 2e-16 ***
hrs18	1.5497206	0.0159485	97.170	¡ 2e-16 ***
hrs19	1.3755921	0.0161510	85.170	¡ 2e-16 ***
hrs20	1.1342224	0.0165685	68.457	¡ 2e-16 ***
hrs21	0.9228048	0.0170465	54.134	¡ 2e-16 ***
hrs22	0.7283091	0.0176203	41.334	¡ 2e-16 ***
hrs23	0.3956666	0.0188440	20.997	¡ 2e-16 ***
wkMonday	-0.0941944	0.0065651	-14.348	¡ 2e-16 ***
wkSaturday	0.7157612	0.0055742	128.406	¡ 2e-16 ***
wkSunday	0.5922995	0.0057104	103.724	¡ 2e-16 ***
wkThursday	-0.2925350	0.0069280	-42.225	¡ 2e-16 ***
wkTuesday	-0.3332496	0.0070530	-47.249	¡ 2e-16 ***
wkWednesday	-0.3214116	0.0070825	-45.381	¡ 2e-16 ***
weather2	-0.0524808	0.0042039	-12.484	¡ 2e-16 ***
weather3	-0.5022357	0.0090549	-55.465	¡ 2e-16 ***
atemp	0.0492639	0.0003444	143.039	¡ 2e-16 ***
humidity	-0.0055156	0.0001171	-47.120	¡ 2e-16 ***
windspeed	-0.0027449	0.0002046	-13.413	¡ 2e-16 ***

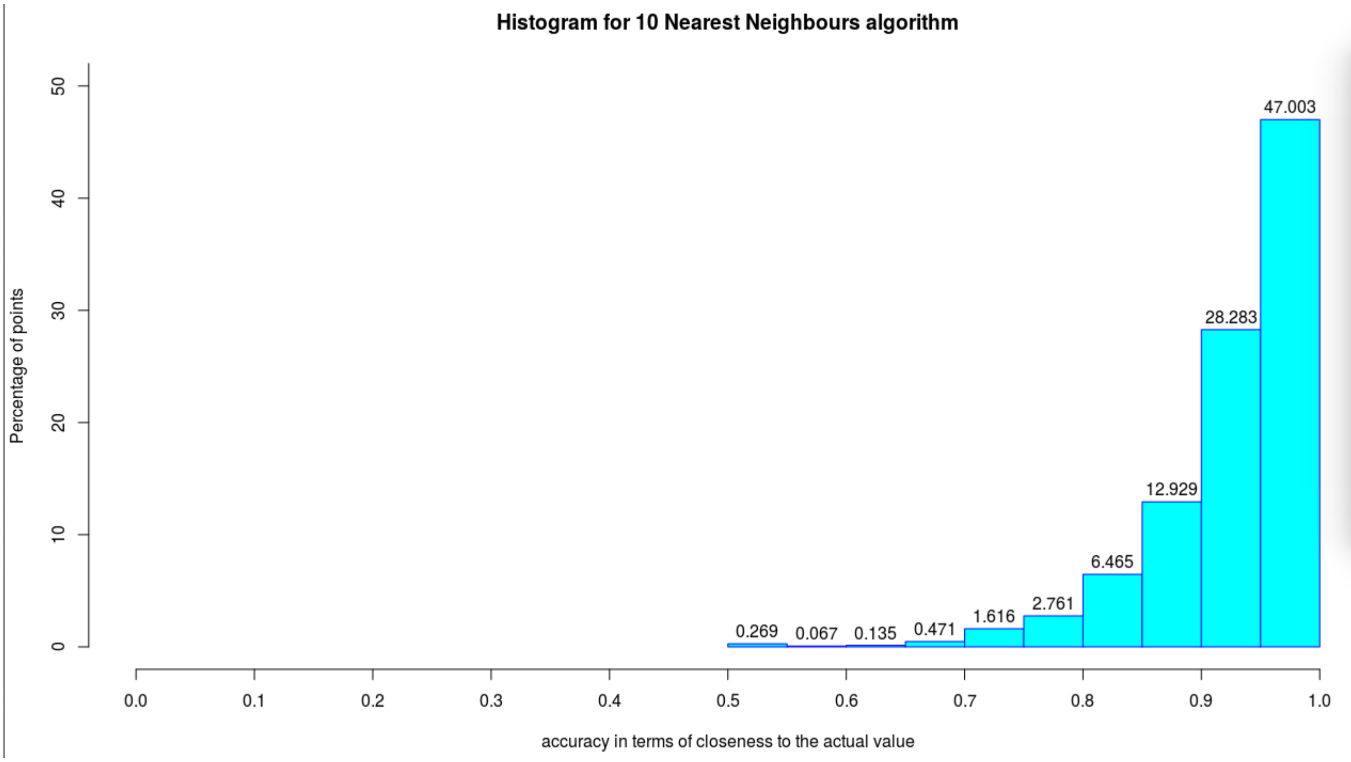
- The numbers of occurrences of the event in disjoint time intervals are mutually independent.

Also the AIC: 156728 becomes too large as the no of degrees of freedom increasing making the model poorer. For the bike data we can clearly see that these conditions are violated. Thus, we conclude that Poisson regression is not the best method.

9 Nearest Neighbour

- Used only nominal attributes (year, month, hour, holiday, workingday, season, weather)
- Did not use date, as train data is for dates 1-20 and the test data is for range 20-30

We applied K-nn for K=10 (i.e. for 10 nearest neighbours), giving more priority to nearer neighbours:



- For about half of the test data points, the difference between the predicted and the actual values was not greater than 50 bikes.
- For more than three quarters of test data points, the error was not greater than 100 bikes. Thus this method was very efficient in predicting the range (low, medium or high).
- Average accuracy in terms of closeness to the actual value = 92.8%, with a standard deviation of 6.7%
- RMSLE = 1.131

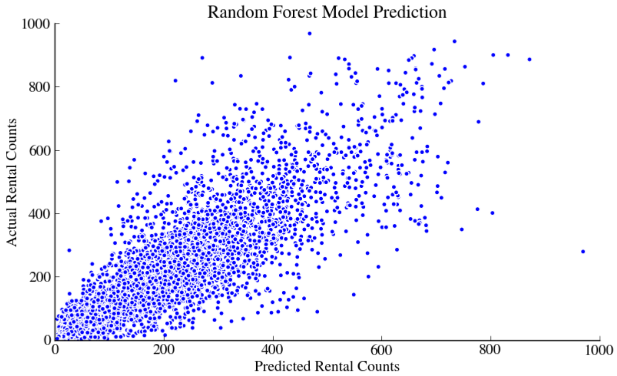
10 Random Forest

Random forests are an ensemble learning method for classification and regression. The algorithm for inducing a random forest was developed by Leo Breiman and Adele Cutler. We thought of applying PCA but it will not lead to any better result because we are already doing feature selection, and as it is that no. of useful features are limited, there is no point in reducing further. Dimensionality reduction is useful only when there are a huge no. of features.

The code for Random Forest was done in python using standard packages and is pretty straightforward, but takes some time to run. It uses a number of decision trees at training time and outputs the class that is the mode of classes (classification) or mean prediction (regression) of individual trees.

Result: We get an RMSE value of **0.49** when we submitted the output of this technique onto Kaggle.

Fig. 4. Scatterplot for Random Forest



11 Gradient Boosting

Gradient boosting is a machine learning technique for regression problems, which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees. The algorithm basically improves at every step of the iteration as:

$$F_{m+1}(x) = F_m(x) + h(x) \tag{1}$$

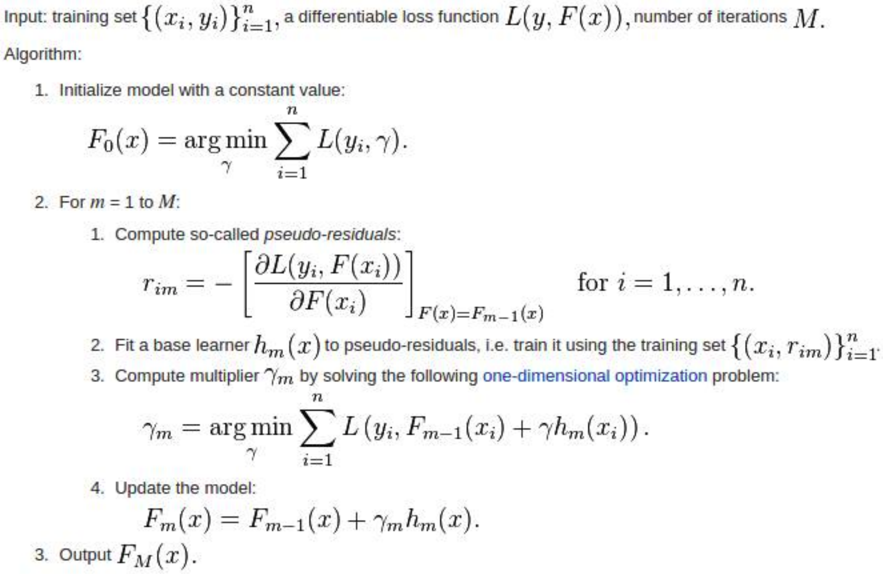
gradient boosting will fit h to the residual y-$F_m(x)$.

On doing with log-totalcount Gradient Boosting method gives **0.43** as RMSLE.

If we apply the model first on casual and then on registered, and then sum up to get the total count then we get an RMSLE of **0.429** which is the best result.

Reason for better results: The gradient boosting method performs better compared to the other methods because it uses an ensemble of classifiers (instead of a single classifier) and improves the model at each step of the iteration. So, it is quite logical that this method gave the best result. Moreover it is quite obvious that instead of predicting total count directly, as we apply models individually on casual and registered counts it would give better predictions for count.

Fig. 5. Gradient Boosting Pseudocode



```
Input: training set $\{(x_i, y_i)\}_{i=1}^{n}$, a differentiable loss function $L(y, F(x))$, number of iterations $M$.
Algorithm:
  1. Initialize model with a constant value:
       $F_0(x) = \arg\min_{\gamma} \sum_{i=1}^{n} L(y_i, \gamma).$
  2. For $m$ = 1 to $M$:
       1. Compute so-called pseudo-residuals:
            $r_{im} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F(x)=F_{m-1}(x)}$   for $i = 1, \ldots, n.$
       2. Fit a base learner $h_m(x)$ to pseudo-residuals, i.e. train it using the training set $\{(x_i, r_{im})\}_{i=1}^{n}$.
       3. Compute multiplier $\gamma_m$ by solving the following one-dimensional optimization problem:
            $\gamma_m = \arg\min_{\gamma} \sum_{i=1}^{n} L\left(y_i, F_{m-1}(x_i) + \gamma h_m(x_i)\right).$
       4. Update the model:
            $F_m(x) = F_{m-1}(x) + \gamma_m h_m(x).$
  3. Output $F_M(x)$.
```

12 Conclusion

So, let us now just summarise the steps we followed to predict the number of bikes that may get rented, for the test data. At first, relations and dependence of the total bike rental counts initiated versus other factors is visualised individually, to gain an insight into the provided data. After that we try out different MLT techniques and check the accuracy level of our prediction by submitting the files onto Kaggle, where we get idea from the RMSLE value generated and the rank. We found that in general regression models work better than trying to transform it into classification. For future work, one can try varying the different parameters and check the results.

13 Acknowledgement

We would like to thank Prof.Harish Karnick and the course tutors and TAs for helping us throughout the project.

References

1. Jay (Haijie) Gu. Using Gradient Boosted Trees to Predict Bike Sharing Demand
   Available at `http://blog.dato.com/using-gradient-boosted-trees-to-predict-bike-sharing-demand`
2. Forecast use of a city bikeshare system: Kaggle problem
   Available at `https://www.kaggle.com/c/bike-sharing-demand`
3. Ting Ma, Chao Liu, Sevgi Erdoan. Bicycle Sharing and Transit: Does Capital Bikeshare Affect Metrorail Ridership in Washington, D.C.?
   Available at `http://smartgrowth.umd.edu/assets/bikeshare_transit_for_parisws_v1.pdf`