Project 2 - STAT 28, Spring 2017

April 20, 2017

1 Dataset

The dataset comes from the *Bike Sharing Demand* competition in Kaggle. You need to download the dataset from `https://www.kaggle.com/c/bike-sharing-demand`. Kaggle will give you two datasets: **train.csv** and **test.csv**. The goal of the project is to fit a linear regression equation to the total count of bikes rented during a particular hour in terms of the available explanatory variables for the training data (train.csv) and then to use this regression equation to predict the counts for the test dataset.

The training and test datasets that are downloaded from kaggle are actually generated from the larger bike sharing dataset that we used in class. So you can actually know the counts in the test dataset by using the dataset used in class. However, the goal of this project is to see how well you can predict the response values for the test data using the training data alone. Consequently, **you are prohibited from using the bike sharing dataset from class for this project and must only use the test and training data accompanying this project (if you use the full dataset, it will be obvious to us because you will do "too well"!).**

2 Specific Tasks

1. **Download the data:** Download the datasets and read the training data into R (**5 points**).
2. **Basic Exploratory Data Analysis:** Perform a basic exploratory data analysis of the training data. Look at histograms, summaries and pairs plots of all the variables. Note any interesting features of the data. (**10 points**).
3. **Choice of response in regression:** Decide whether to do a regression analysis with the total count as the response or to do separate analyses for the casual and registered counts. Ultimately, the goal is to predict the total count for the observations in the test dataset but this can be done either by getting a prediction directly for the total count by regression or by getting separate predictions for the casual and registered counts and then adding them. Also investigate whether it is appropriate to take logarithms of the counts for use in regression. (**10 points**).
4. **Regression Analysis:** Perform a regression analysis of the response on the explanatory variables. Note that an important role is played by the variable time. Indeed, even if all other explanatory variables are held constant, it is reasonable to give a different prediction for the number of bike rentals at 9 am compared to 11 pm. So the time needs to be used in your regression analysis. But the hourly time is a factor variable with 24 levels. It might therefore be reasonable to create a new variable for time by dividing into various groups (such as morning, afternoon, evening, night, midnight etc.). Most of the variables involved are categorical. You will need to investigate interactions between the variables. (**40 points**).

5. **Variable selection:**Perform variable selection to select a suitable model involving a subset of your explanatory variables. You can use either stepwise methods or regression subsets in conjunction with cross validation. (**30 points**).

6. **Regression Diagnostics:** Look at some simple diagnostic plots and comment on whether any of the regression assumptions are obviously violated for this dataset. (**10 points**).

7. **Predictions:** Read the test dataset into R and predict the counts for the test data. Submit your predictions to Kaggle and get a score for your prediction. For this competition, Kaggle is computing the Root Mean Squared Logarithmic Error (RMSLE) as the accuracy of prediction. In other words, if your predicted counts are $p_1, \ldots, p_m$ and the actual counts are $a_1, \ldots, a_m$, then Kaggle is computing your accuracy via:

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

   The plus one is to avoid taking a logarithm of zero. Note that this means that you cannot submit negative predictions (this might also motivate working with logarithms). Report the score given by Kaggle. (**15 points**).

8. **Comparison**: How does your kaggle score compare with the top scores in the leaderboard? Can you suggest some ways in which one can give improved predictions? (**5 points**).

3 Format for Submission

You are expected to create a *.Rmd* le for this project from scratch. RStudio will setup a new *.Rmd* le for you if you go to *File > New File* in the menu bar. The text from this pdf should not be part of your *.Rmd* le. Like the homeworks, you will turn in only a compiled *.pdf* to gradescope. An actual analysis would blend these components together into a single narrative. However, for grading purposes, we have divided the project into the specific tasks described above and each of the tasks should be addressed in a separate section and appropriately labeled so that you can tell gradescope which pages correspond to which task.

This project is intentionally more open-ended than the homework so as to be more reflective of an actual analysis of the data. For each of the specific tasks, provide commentary on what you are doing, provide R code and ouptut and also provide commentary on what are deducing from the output. The commentary should be just regular text typed in your .Rmd file (it does not need a > in front of it like the homework).

4 Kaggle Submissions

Note that you will not be graded on the score that you will get in Kaggle. The grade will be based on your analysis methods. You can find instructions below on how to make a submission in kaggle for this project.

Kaggle submission can, for example, done in the following way. Suppose I fit the following simple-minded regression equation to the training data:

```
md = lm(count ~ atemp + as.factor(workingday) + as.factor(workingday)
: atemp+ as.factor(weather), data = train.dt)
```

I can then use this model to generate predictions for the test data in the following way:

```
test.dt = read.csv(file.path(dataDir, "test.csv"))
pred = predict(md, test.dt[1:nrow(test.dt),])
```

My dumb regression equation can give negative predictions so I will take absolute values of the predictions:

```
pred = abs(pred)
```

I can now create a .csv file for submission to kaggle as:

```
sbm = data.frame(datetime = test.dt$datetime, count = pred)
write.csv(sbm, file.path(dataDir, "Subm.csv"), row.names = FALSE)
```

I will then simply upload the file *Subm.csv* to Kaggle to submit my prediction. Kaggle gave me an accuracy score of 1.43714 for these naive predictions which is well-off the top-scores in the leaderboard.