Bike Sharing Demand

Yunzhi BAI, Jie ZHENG

January 31, 2017

Abstract

Keywords: Time Series Forecasting, ARIMA, Random Forest, Extreme Gradient Boosting, Bike Sharing System Data, Air Quality Data, AXA Call Center Data.

Remark: We provide 3 ipython notebook (ML-PROJECT-P1,ML-PROJECT-P2,ML-PROJECT-P3), each corresponds to one dataset. For more details, please refer to README.

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.In this competition, we are asked to combine historical usage patterns with weather data in order to forecast bike rental demand in the Capital Bikeshare program in Washington, D.C. We are provided hourly rental data spanning two years. For this competition, the training set is comprised of the first 19 days of each month, while the test set is the 20th to the end of the month. Submissions are evaluated one the Root Mean Squared Logarithmic Error (RMSLE).Besides the *Bikeshare Data Set*, we use two more datasets (*Air Quality Data Set* and *AXA Call Center Data Set*) to analyze furthermore the performance of the proposed methods.

The common points of these three datasets are the following:

- Data fields contain datetime (date(YYYY/MM/DD)+time(HH.MM.SS));
- Data fields contain (or not) several variables related to their specific datetime (e.g. season,weather,temperature,humidity);
- The objective is to, based on knowledge of historical data, predict a certain value for a specific datetime covered by the test set.

2 Model building on toy dataset - bikeshare

- *Independent Variables*: datetime, season, holiday, workingday, weather, atemp, humidity, windspeed.
- *Dependent Variables*: casual (number of non-registered user rentals initiated),registered (number of registered user rentals initiated), count (number of total rentals)
- *Models*: time series analysis, random forest regression, extreme gradient boosting

2.1 Time series analysis

In this section, we want to see if it's possible to get something of time series types of models. Using Pandas, we load the bike sharing dataset and set the index such that the data has time object as index. The dtype='datetime[ns]' confirms that it is a datetime object. In plotting, the blue part consists of hourly rental data of the first 19 days of each month, the white part is the 20th to the end of the month, which we need to predict.

- **Check Stationary of a Time Series**

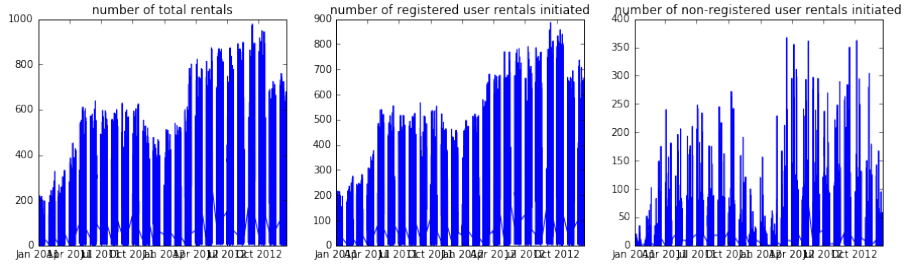

Figure 1: The evolution trend of number of rentals (total, registered, non-registered.

Given the fact that most of the TS models work on the assumption that the TS is stationary, we hereby plot rolling statistics in order to check whether the statistical properties such as mean, variance of bike sharing dataset remain constant over time. More formally, we plot the moving average and moving variance and see if it varies with time. At any instant 't', we'll take the average and variance of the last month, i.e. last 1 months. For each month, we dispose 24(hours)*18(days)=432 records.

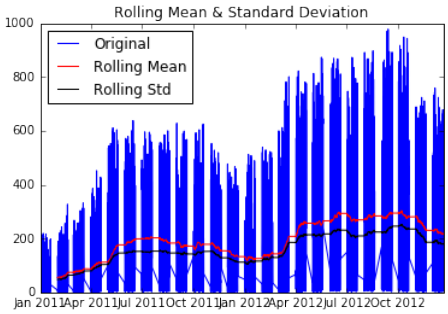

Figure 2: Before eliminating trend (entire data)

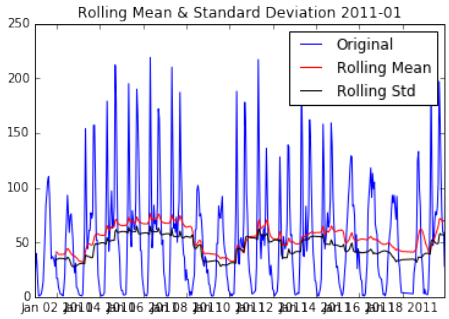

Figure 3: Before eliminating trend (partial data)

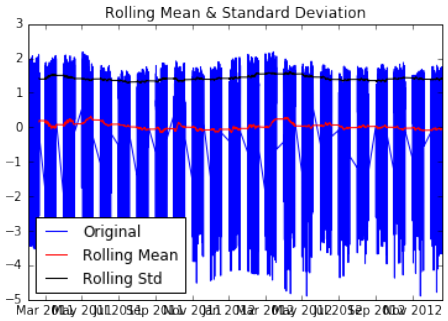

Figure 4: After eliminating trend (entire data)

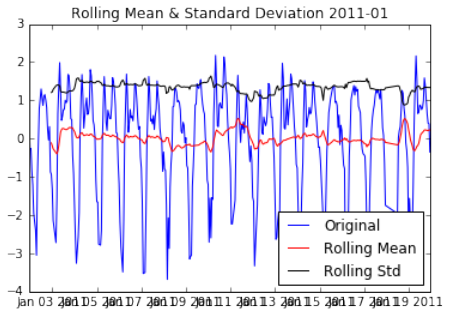

Figure 5: After eliminating trend (partial data)

- **Eliminating Trend**

Several methods can used to eliminate the trend. Here we choose a simple way, which is using the log transformation followed by removing the rolling mean. After this step, we obtain a more stationary series, whose mean and variance values remain constant over time. We can furthermore decompose the TS into additive time series (observed, trend, seasonal, random). We will apply these techniques in our real datasets.

- **Forecasting a Time Series**

Different techniques work for TS analysis,such as ARIMA (Auto-Regressive Integrated Moving Averages). In our case (bike sharing dataset),we observed that it is a bad idea to use only datetime and drop other features (season, holiday, workingday, weather, atemp, humidity, windspeed)

who have great effects on total number of rentals. Thus, it's preferable to use machine learning techniques.

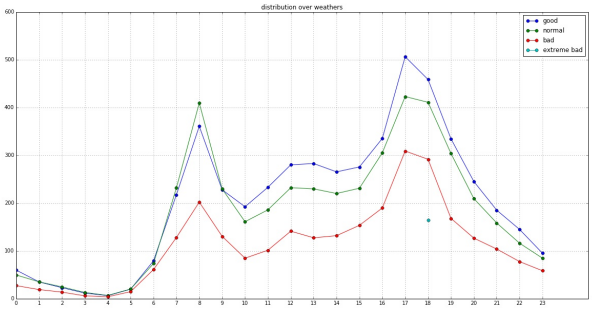

Figure 6: Distribution of number of rentals over hour,weather

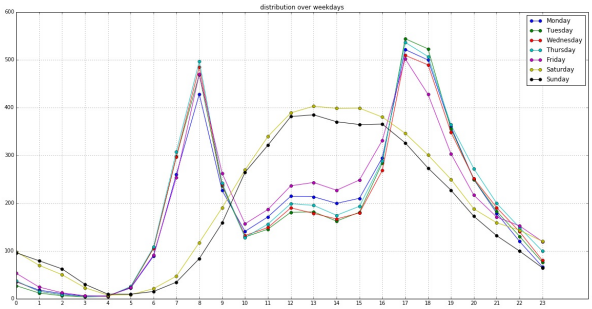

Figure 7: Distribution of number of rentals over hour,weekday

For instance, we can observed from Figure 4 and Figure 5 that 1) for working days, there are much more rentals between 6h-8h and 16h-19h, 2) unlike working days, during weekends, there are more rentals during the mid of the day, 3) Better the weather is, more rentals there will be. And when the weather is extremely bad, the rental will only occur during the period when people come back home from work.

2.2 Random forest regression

The first classical regression model we apply on the bike sharing dataset is random forest regression. We implement two methods to forecast hourly bike rental demand: 1) direct prediction of the total number of rentals, 2) indirect prediction by summing registered rentals and casual rentals (total=registered+casual).

- **Direct Forecast**

Evaluation metrics : Root Mean Squared Logarithmic Error (RMSLE)

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

where n is the number of hours in the test set, p_i is your predicted count $a_i$is the actual count.

Feature engineering:
(1) extract "year", "month", "day" and "hour" from "datetime"
(2) compute average rentals over "season","holiday", "workingday", "hour", "month", "year" and

"weather.

Tune parameters: GridSearchCV

- **Indirect Forecast**

Evaluation metrics : Root Mean Squared Error (RMSE). Logarithm in initial evaluation metrics is realized in feature engineering step.

We have followed following steps to train the model:

1. Splitting training set and validation set: train-test-split
2. Logarithmic transformation of dependent cols: y1=log(casual+1) and y2=log(registered+1)
3. Tuning parameters of RandomForestRegressor with $GridSearchCV$, where scoring=mse : $regr = RandomForestRegressor(n_estimators = 250, maxfeatures = 0.5, criterion = 'mse')$
4. Re-transforming the predicted variables: casual=exp(y1)-1 and registered=exp(y2)-1
5. Setting an offset (ypred[ypred<0]=0) adding casual and registered predictions together

2.3 Extreme Gradient Boosting

Evaluation metrics: Root Mean Squared Logarithmic Error (RMSLE)

Xgboost allows us to train the model by customizing objective function and evaluation metrics. Knowing that RMSLE is not contained in its default metrics, hence here we defined our own customized evaluation metrics.

```
def rmsleErr(preds, dtrain):
    labels = dtrain.get_label()
    assert len(preds) == len(labels)
    return 'error', np.sqrt(np.mean(np.power(np.log1p(preds)-np.log1p(labels), 2)))
```

Figure 8: Customized evaluation metric RMSLE.

- **Direct Forecast**

Feature engineering:
(1) extract "year", "month", "day" and "hour" from "datetime"
(2) compute average rentals over "season","holiday", "workingday", "hour", "month", "year" and "weather.

Tune parameters:
(1) enable early stopping by setting "watchlist" with "train-test-split"
(2) use small eta (learning rate) to tune the best parameters

- **Indirect Forecast**

We predict separately for registered users and casual users with original features, by using the same method to tune hyper-parameters.

2.4 Analysis of performance

In this dataset, we realized basic analysis using Time Series techniques (check stationary, eliminate trend), . However, assumptions for the model are not satisfied, so it's preferable to use machine learning techniques.

Random forest comined with indirect forcast gave us the best results. In the meanwhile, its training and predicting time are very fast compared to other methods.

Method	Error(Direct)	Error(Indirect)
Time Series	N.A	N.A
Random Forest	0.36	0.16
XGBoost	0.34	0.23

Table 1: table of performance.

3 Model building on real dataset

3.1 AXA Call Center France

- *Independent Variables*: datetime.
- *Dependent Variables*: prediction (total number of incoming calls)
- *Models*: time series analysis, random forest regression

3.1.1 Random forest regression

Feature engineering:
(1) extract "year", "month", "day" and "hour" from "datetime"
(2) compute average rentals over "weekday", "hour","day", "month", "year".

Prediction:

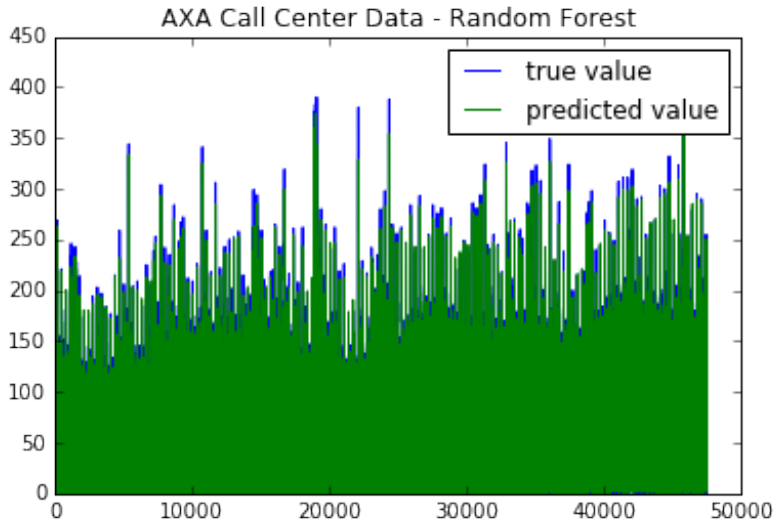

Figure 9: Random Forest prediction, RMSE=4.984

3.1.2 Time series analysis

Steps:
(1) Check stationary of a Time Series
(2) Eliminating trend
(3) Forecasting a Time Series(ARIMA)

We won't go into the technical details about the first two steps. Since we have shown its concepts in the previous section (*Bikeshare Data Set*)
As for the third step, forcasting a TS, we choose ARIMA (Auto-Regressive Integrated Moving Averages).We use two plots, **Autocorrelation Function** (ACF) and **Partial Autocorrelation Function** (PACF) to determine p, q, d. Here we obtain:
- Number of AR (Auto-Regressive) terms (p) = 2
- Number of MA (Moving Average) terms (q) = 1

- Number of Differences (d) = 0

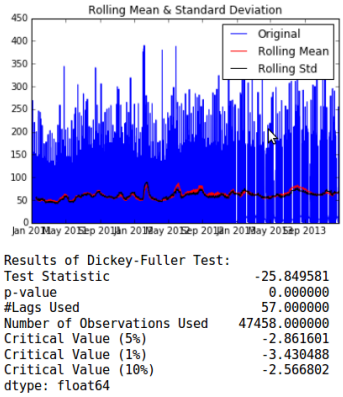

Figure 10: Plotting Rolling Statistics and Dickey-Fuller Test

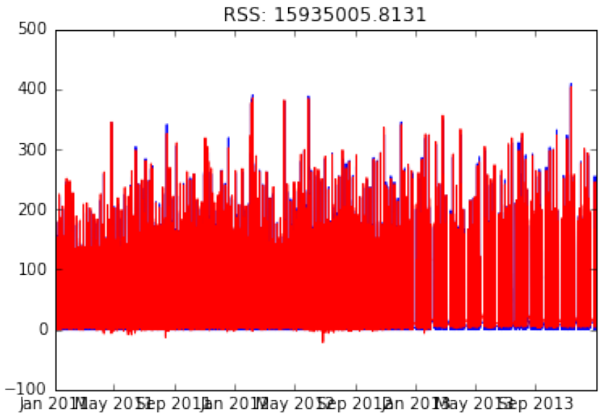

Figure 11: Time series prediction, RMSE=18.312

3.1.3 Analysis of performance

Method	RMSE
Time Series	18.312
Random Forest	4.987

Table 2: AXA Call Center Data - table of performance.

3.2 Air Quality Data

- *Independent Variables*:Date(DD/MM/YYYY), Time(HH.MM.SS), Temperature, Relative Humidity, Absolute Humidity.
- *Dependent Variables*: True hourly averaged concentration CO.

3.2.1 Random forest regression

Feature engineering:
(1) extract "year", "month", "day" and "hour" from "datetime"
(2) compute average rentals over "weekday", "hour","day", "month", "year".

Prediction:

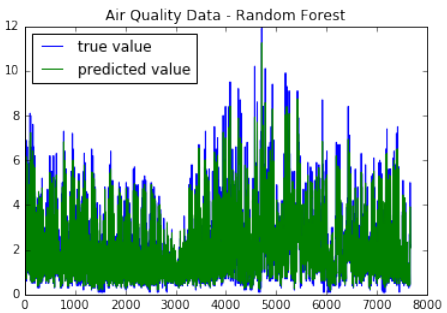

Figure 12: Random Forest prediction, RMSE=0.269.

3.2.2 Time series analysis

Steps:
(1) Check stationary of a Time Series
(2) Eliminating trend
(3) Forecasting a Time Series (ARIMA)
Here we set
- Number of AR (Auto-Regressive) terms (p) = 2
- Number of MA (Moving Average) terms (q) = 1
- Number of Differences (d) = 0

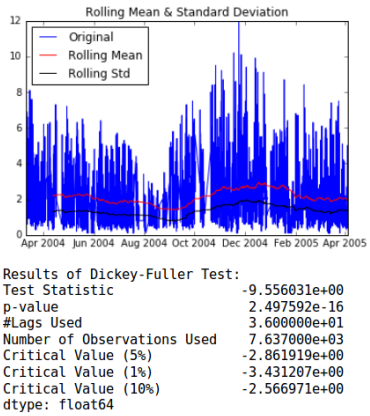

Figure 13: Plotting Rolling Statistics and Dickey-Fuller Test

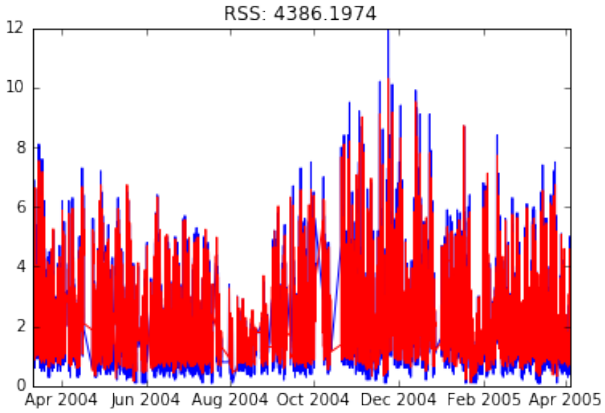

Figure 14: Time series prediction, RMSE=0.756

3.2.3 Analysis of performance

Method	RMSE
Time Series	0.756
Random Forest	0.269

Table 3: Air Quality Data - table of performance.

4 Conclusion

This project studies mainly three methods, time series analysis, random forest regression and extreme gradient boosting, on three datasets, bikeshare dataset, AXA call center dataset and air quality dataset. All these three datasets contain a field datetime spanning over 1-2 years and require us to predict a certain value for a specific datetime, based on knowledge of historical data.

In general, random forest performs much better than pure time series forecast (in terms of RMS error). The result is not surprising.Time Series only takes into account the datetime variable, and ignore the effect of other independent variables, such as weather, temperature, humidity, which also reflect properties of the datetime. On the contrary, random forest, more generally machine learning methods, not only take into account the datetime but also other relative information provided in the data set. XGBoost, compared to classical machine learning method, has one great advantage, which is allowing us to customize objective function and evaluation metrics. When the evaluation metrics is not a very common metrics (for instance, the linear exponential loss, which gives a relatively higher penalty to underestimation), this functionality can become very useful.

However, we also notice that time series is more useful in terms of predicting the trend in the data, since it is capable of identifying additive time series (observed, trend, seasonal, random).

For example, if an activity increases linearly over a period between year 1992 and 2012, you would expect it to continue increasing in the future. Time series analysis can catch this information perfectly and implement it into its model. Random forest, however, would not make that forecast.

In conclusion, machine learning methods, especially random forest, is a good choice to do short-term forecasts, like has been shown in our project.

5 Perspectives

Knowing that Time series forecast and machine learning methods both have their proper advantages on forecasting, we are sure there is a way to combine these two methods in order to get a better performance on time series type problems.

6 References

1. *Bikeshare Data Set*: Fanaee-T, Hadi, and Gama, Joao, Event labeling combining ensemble detectors and background knowledge, Progress in Artificial Intelligence (2013): pp. 1-15, Springer Berlin Heidelberg.
2. *AXA Call Center Data Set*: AXA-X DaScIS chair.
3. *Air Quality Data Set*: Saverio De Vito, ENEA - National Agency for New Technologies, Energy and Sustainable Economic Development.