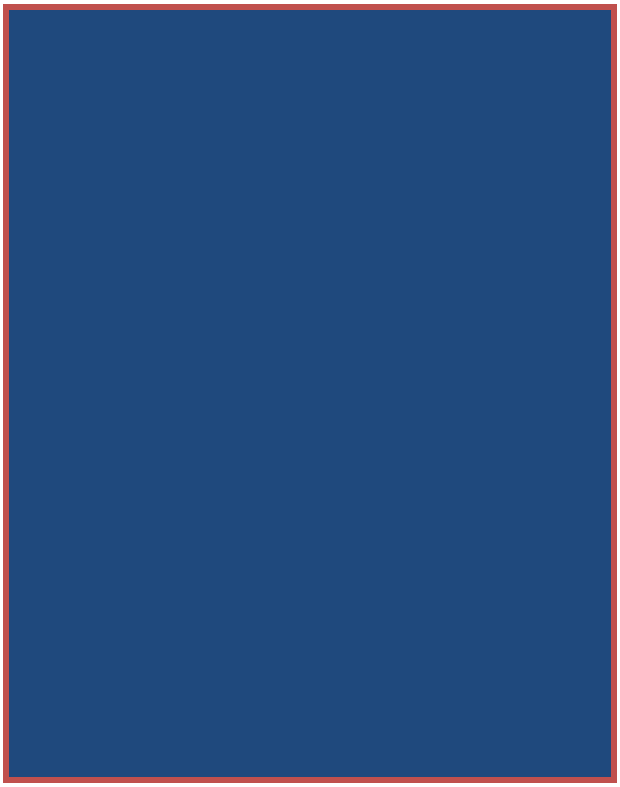


April / May 2015

Walmart Recruiting - Sales in stormy weather

Notes

Isaac

Authored by: Gino Tesei

Isaac

Executive summary

First section focuses on explanatory analysis. In particular, the following aspects are studied

- correlation among sold products,
- correlation among sold products in a given store,
- correlation among products in the stores associated to the same weather station,
- correlation among same sold products sold in all stores.

The next section compares two models for imputing missing values in weather data assessing the related performances.

On the basis of explanatory analysis's findings, the remaining sections experiment target models, assess related performances and the rationale behind them. In particular,

- **model #1** uses bootstrap and k-folds as resampling techniques and for each pair <store, item> takes the best performing model among Average / Mode / Linear Regression / Robust Linear Regression / PLS Regression / Ridge Regression / Enet Regression / KNN Regression / Bagged Tree Regression;
- **model #2 is a simple interpolation of sold products** for each pair <store, item> **without using weather data**;
- **model #3** improves the model #1 after analyzing what king of data is missing in the dataset that could boost its performances and using a proxy to estimate it;
- **model #4 is based on extreme gradient boosting.**

In conclusion, **best performances are obtained with the simplest model here evaluated**, i.e. **simple interpolation of sold units (model #2) without using weather data**. This model performed in public leaderboard as **0.10119** and it suggests the idea that weather data is not strong correlated to sold units. Besides performances, such a model has the following advantages:

- **very fast**: on my laptop (Mac Pro 2.4 GHZ 8GB RAM) required just a few minutes to fit all train data and to make predictions;

Isaac

- **very simple**: it doesn't require wheatear data and, hence, it doesn't require missing values imputation of such data.

Exploratory analysis

Weather

Exploratory analysis on weather data has been performed and related results are shown in the attached Excel document weather_elab.xlsx

Stores and products

In training data there's only 2.5% of sold <units> with a value > 0, i.e. 97.5% of <date,store,item> in the training are 0s (=unsold).

Key highlights:

- **20 weather stations**
  - that are associated to **2.25 stores** on average (standard dev. = 1.51)
- **45 stores**
- **111 items**
- **4995 different combinations** of < store_nbr, item_nbr> (45 x 111 = 4995), whose
  - **255 (5.1%)** combinations of < store_nbr, item_nbr> has at least one unit sold one day in the training set
    - where the average units sold are 19.35 (standard dev. is 33.02)
    - corresponding to 236038 observations, i.e. 5.1% of total training observations
  - **4740 (94.9%)** combinations of < store_nbr, item_nbr> has 0 units sold (=unsold) every day in the training set
    - corresponding to 4381562 observations, i.e. 94.9% of total training observations

Isaac

Predictors' distribution in stations/products sold (5.1%)

```
> train.sold.mean
       units         sold         tmax         tmin         tavg       depart     dewpoint      wetbulb         heat         cool      sunrise       sunset     snowfall
1.930620e+01 1.000000e+00 7.108756e+01 4.891950e+01 6.025441e+01 1.424580e+00 4.494122e+01 5.224110e+01 1.023583e+01 5.458313e+00 5.884067e+02 1.822394e+03 6.260930e-02
 preciptotal  stnpressure     sealevel  resultspeed    resultdir     avgspeed           TS           GR           RA           DZ           SN           SG           GS
7.183830e-02 2.863450e+01 3.001360e+01 6.286939e+00 1.858778e+01 7.950250e+00 9.383659e-02 6.778570e-05 2.455410e-01 1.582372e-02 5.216109e-02 1.694642e-05 2.880892e-04
          PL      FG_PLUS           FG           BR           UP           HZ           FU           DU           SS           SQ           FZ           MI           PR
6.820936e-04 3.697710e-02 6.088003e-02 2.852041e-01 9.176489e-03 6.116388e-02 2.529254e-03 1.287928e-03 2.880892e-04 5.380490e-04 1.842500e-02 2.690245e-03 1.694642e-05
          BC           BL           VC
5.634686e-03 3.215584e-03 2.345809e-02
> train.sold.sd
       units         sold         tmax         tmin         tavg       depart     dewpoint      wetbulb         heat         cool      sunrise       sunset     snowfall
39.425009919  0.000000000 19.137417762 18.876151515 18.583128631  4.436218223 19.391412663 16.603492721 13.493790837  7.350948641 61.243436539 64.014095071  0.386871736
 preciptotal  stnpressure     sealevel  resultspeed    resultdir     avgspeed           TS           GR           RA           DZ           SN           SG           GS
 0.270400573  1.932152225  0.188687828  4.135204279  9.606737576  3.818474905  0.291601854  0.008232945  0.430408397  0.124793426  0.222352249  0.004116577  0.016970782
          PL      FG_PLUS           FG           BR           UP           HZ           FU           DU           SS           SQ           FZ           MI           PR
 0.026108068  0.188705967  0.239110628  0.451512542  0.095353653  0.239631182  0.050228153  0.035864675  0.016970782  0.023189690  0.134482695  0.051797864  0.004116577
          BC           BL           VC
 0.074852922  0.056614994  0.151353568
```

Predictors' distribution in stations/products unsold (94.9%)

```
> train.unsold.mean
       units         sold         tmax         tmin         tavg       depart     dewpoint      wetbulb         heat         cool      sunrise       sunset     snowfall
0.000000e+00 0.000000e+00 7.215166e+01 4.995986e+01 6.130508e+01 1.468940e+00 4.612701e+01 5.322650e+01 9.607355e+00 5.850438e+00 5.908603e+02 1.824785e+03 5.295791e-02
 preciptotal  stnpressure     sealevel  resultspeed    resultdir     avgspeed           TS           GR           RA           DZ           SN           SG           GS
7.237483e-02 2.878992e+01 3.001503e+01 6.431692e+00 1.833472e+01 8.041379e+00 9.602534e-02 7.234863e-05 2.440335e-01 1.900897e-02 4.532356e-02 2.442051e-05 2.884816e-04
          PL      FG_PLUS           FG           BR           UP           HZ           FU           DU           SS           SQ           FZ           MI           PR
8.499252e-04 3.464016e-02 5.686191e-02 2.838743e-01 8.499024e-03 6.604266e-02 2.447757e-03 1.551958e-03 3.138150e-04 6.296841e-04 1.628483e-02 2.743086e-03 2.442051e-05
          BC           BL           VC
5.928479e-03 2.993453e-03 2.518440e-02
> train.unsold.sd
       units         sold         tmax         tmin         tavg       depart     dewpoint      wetbulb         heat         cool      sunrise       sunset     snowfall
 0.000000000  0.000000000 18.939403873 18.793597247 18.439677991  4.642275712 19.146705091 16.457474840 13.076780217  7.619332597 60.496662201 62.676428600  0.345066760
 preciptotal  stnpressure     sealevel  resultspeed    resultdir     avgspeed           TS           GR           RA           DZ           SN           SG           GS
 0.273509611  1.742456336  0.185739044  4.198094056  9.626832363  3.913793690  0.294626030  0.008505493  0.429512751  0.136556353  0.208012854  0.004941652  0.016982298
          PL      FG_PLUS           FG           BR           UP           HZ           FU           DU           SS           SQ           FZ           MI           PR
 0.029141088  0.182866687  0.231578601  0.450876639  0.091797562  0.248356678  0.049414231  0.039364321  0.017712047  0.025085609  0.126568715  0.052302598  0.004941652
          BC           BL           VC
 0.076768052  0.054630513  0.156684873
```

Isaac

Z-score

Ordering by the difference of the predictors' means in the two sets and dividing for the standard deviation, we have the following situation

```
> z.score
    sealevel       sunset           GS      sunrise           BR  stnpressure           RA  preciptotal     avgspeed    resultdir         tmax         tavg       wetbul
4.750268e-05 1.310294e-03 1.360286e-03 4.153492e-03 4.683142e-03 5.399856e-03 6.175282e-03 7.416010e-03 1.133913e-02 1.379237e-02 1.475924e-02 1.715351e-02 1.853090e-0
          MI         tmin  resultspeed           TS     dewpoint       depart           FU           BC           GR         heat      FG_PLUS         cool            V
1.928229e-02 2.084609e-02 2.253224e-02 2.282014e-02 2.574097e-02 3.024536e-02 3.323781e-02 4.968214e-02 6.327267e-02 6.519843e-02 6.723144e-02 6.725542e-02 6.878785e-0
          FG           BL           HZ           UP           SS           FZ           SQ           SN           DZ           DU     snowfall           PL            S
7.041009e-02 7.392517e-02 7.415314e-02 7.938741e-02 8.232263e-02 1.305439e-01 1.466163e-01 1.497059e-01 1.690133e-01 1.716193e-01 1.805644e-01 1.994799e-01 3.109222e-0
          PR         sold        units
3.109222e-01 1.956295e+01 1.956295e+01
```

So, the most promising predictors seem

- PR (z-score 0.31)
- SG (z-score 0.31)
- PL (z-score 0.19)
- Snowfall (z-score 0.18)
- DU (z-score 0.17)
- SN (z-score 0.15)

On the other hand, **z-score is always minor than 1**.

Isaac

Best selling combinations of stores / products

```
> head(store.product.sold , n = 15)
     store_nbr item_nbr     units
3596        33       44 207.77133
1690        16       25 157.57993
3263        30       44 157.22696
1785        17        9 144.16081
155          2       44 133.85714
342          4        9 122.00312
3561        33        9 111.14442
2673        25        9  97.48764
3708        34       45  92.31151
4152        38       45  91.50629
1599        15       45  91.31553
3375        31       45  84.78036
1824        17       48  81.03514
1368        13       36  80.43844
2709        25       45  79.35707
```

Figure 1 units is the average of sold units

Isaac

Most sold products

```
> product.sold = product.sold[order(pro
> head(product.sold , n = 15)
   item_nbr      units
45       45 24.1613221
9         9 22.0340144
5         5 20.3524519
44       44 13.8748317
16       16  5.4512500
68       68  4.1404567
25       25  3.5637740
37       37  3.0085577
36       36  2.0159856
48       48  1.8291346
43       43  1.1191587
41       41  1.1147596
6         6  0.9951442
23       23  0.8266587
8         8  0.7294471
>
```

Figure 2 units is the average of sold units

Isaac

Best selling stores

```
> store.sold = ddply(train[,c(3,5)],
> store.sold = store.sold[order(store
> head(store.sold , n = 15)
   store_nbr    units
33        33 2.884223
17        17 2.604007
30        30 1.791464
2          2 1.787830
16        16 1.743469
25        25 1.648756
38        38 1.531532
15        15 1.430695
13        13 1.397097
34        34 1.346557
31        31 1.302425
3          3 1.256455
4          4 1.231963
21        21 1.130010
9          9 1.113797
>
```

Figure 3 units is the average of sold units

Correlation among sold products (cross selling?)

This analysis can be done at three levels, i.e. for each observed day in the training set

- among the products sold in each store (45 correlation matrices)
- among the products sold in the stores associates to the same weather station (20 correlation matrices)
- among the products sold in all stores (1 correlation matrix)

Isaac

Correlation among sold products in a given store

Let's focus only in a given store and let's consider the correlation among products sold in such a store in the same day.

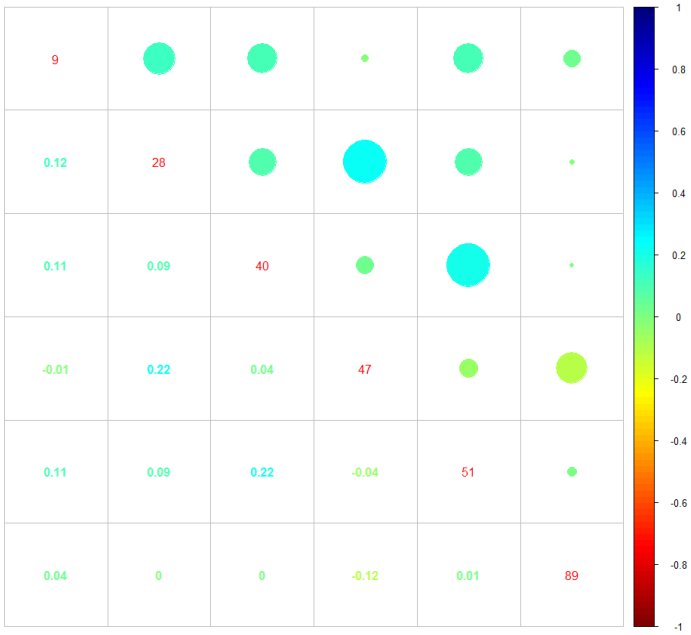


Figure 4 Correlation plot of sold products sold in store N.1 (unsold products are discarded)

For example, as shown by the previous correlation plot, in the store N.1 we can observe a direct correlation (0.22) among products (40,51) and (47,28) while an inverse correlation (-0.12) between products (47,89).

Let's focus now on the stores associated to the same weather station, i.e. assuming that selling happened with the same weather. For example, let's consider the **stores 14 and 45 associated to the same weather station N. 16**.

Isaac

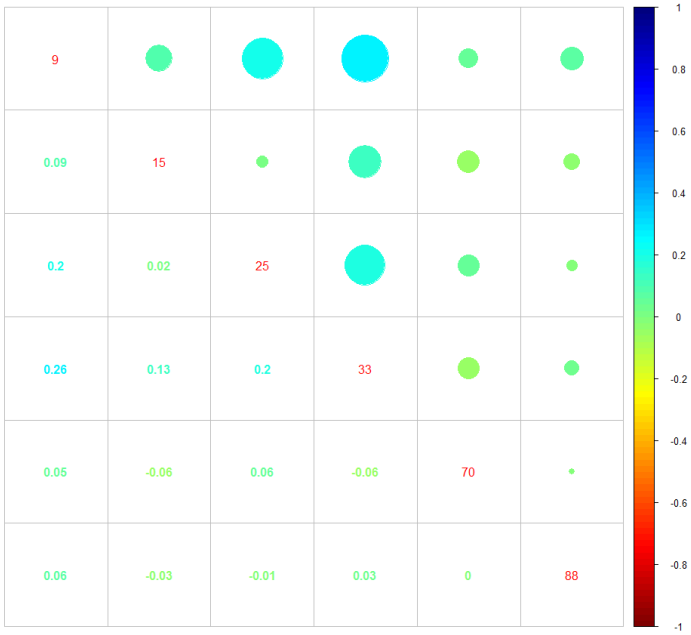


Figure 5 Correlation plot of sold products (unsold products are discarded) sold in store N.14 (station N. 16)

In the **store N. 14**, we can observe

- a **direct correlation (0.26)** between **products (33,9)**,
  - p-value: 4.441e-16
  - 95 percent confidence interval: 0.1998593 0.3183700
  - 99 percent confidence interval: 0.1805916 0.3362195
- a **direct correlation (0.2)** between **products (25,9)**,
  - p-value: 2.75e-10
  - 95 percent confidence interval: 0.1409988 0.2628716
  - 99 percent confidence interval: 0.1213516 0.2813823
- an inverse poor correlation (-0.06) between products (9,88)
  - **p-value: 0.05624**
  - **95 percent confidence interval: -0.001646013 0.124943908**
  - **99 percent confidence interval: -0.02163447 0.14457201**

Isaac

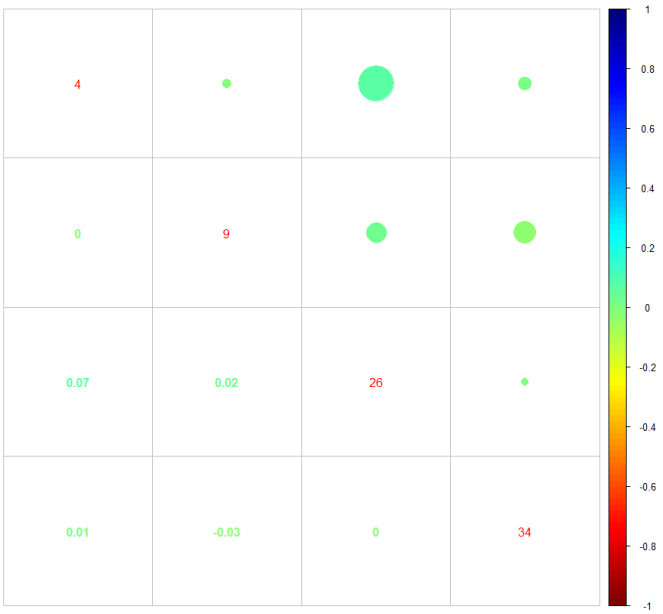


Figure 6 Correlation plot of sold products (unsold products are discarded) sold in store N.45 (station N. 16)

In the **store N. 45**, we can observe

- a direct **correlation (0.07)** between products **(26,4)**
  - p-value: 0.02467
  - 95 percent confidence interval: 0.00931338 0.13571730
  - **99 percent confidence interval: -0.01067778 0.15528520**
- an inverse poor correlation (-0.03) between products (34,4)
  - **p-value: 0.7548**
  - **95 percent confidence interval: -0.05343611 0.07362572**
  - **99 percent confidence interval: -0.07334693 0.09347732**

The question is why in store probably in the same area (i.e. associated to the same weather station) we have profiles so different? First of all, except for the product N.9, such stores has different sold products, i.e.

- Store N. 45: 9, 4, 26, 34
- Store N. 16: 9, 15, 25, 33, 70, 88

Moreover, the product correlation is different. We can make several hypotheses, e.g.

Isaac

- different customers and/or different customer's needs
- products out of stock in a store but not in the other
- etc.

In any case, we can conclude that **product correlation among the products sold in the stores associates to the same weather station is not very meaningful, in general**. For instance, referencing to the previous case, calculating the correlation between item N. 4 (sold in the store N. 45) and item N. 15 (sold in the store N. 16) is not very meaningful, as the previous item hasn't been sold in the second store, and vice-versa.

Correlation among sold products in stores associated to the same weather station

Referencing to the previous example (i.e. stores 14 and 45 associated the weather station 16), different considerations can be done for the **item N.9 sold in both stores**. On the other hand, we observe a poor **direct correlation of 0.05579844**

- **p-value: 0.0853**
- **95 percent confidence interval: -0.007766513 0.118914267**
- **99 percent confidence interval: -0.02775138 0.13857405**

Correlation among sold products sold all stores

Referencing to the previous example (i.e. stores 14 and 45 associated the weather station 16), as the correlation observed for the only item sold in both stores is not statistic significant, it seems not very meaningful to consider sold units of item N. 9 in stores associated to other weather stations.

Isaac

Weather missing values' imputation models

Discarded input variables

- station_nbr
- date

All predictors are assumed as numeric (no factors)

Models & Performance

- Performed **basic** imputation with **BlackGuido** on Mode/Average/LineraReg and observed mean imputing performance (RMSE) **17.9**
- Performed **full** imputation with **BlackGuido** on Mode/Average/LineraReg/KNN_Reg/PLS_Reg/Ridge_Reg/SVM_Reg/Cubist_Reg and observed mean imputing performance (RMSE) **17.8**

Predictive model #1 – basic

- For each date <d> and for each item <i> sold/predicted to be sold units in the store <s>, the related train/test set has been built with (imputed) weather data of the station <st> associated to the store <s> in the key.csv file, and the related output variable is the sold units for <d,s,i>
- Feature selection
    - Removing predictors that make ill-conditioned square matrix
    - Removing near zero variation predictors
    - Removing high correlated predictors
- Feature scaling
- Resampling: bootstrap + k-folds
- Models:
    - Average
    - Mode

Isaac

  - LinearReg
  - RobustLinearReg
  - PLS_Reg
  - Ridge_Reg
  - Enet_Reg
  - KNN_Reg
  - BaggedTree_Reg
- Leaderboard performance on dataset filled with basic imputation: **0.15193**

Isaac

Example (store n.1 / product n.9 / weather station n.1)

The output variable (units sold) has 929 observations with mean 29.48 and standard deviation 22.07. There are only 10 observations out of 929 (1%) with 0 units sold.

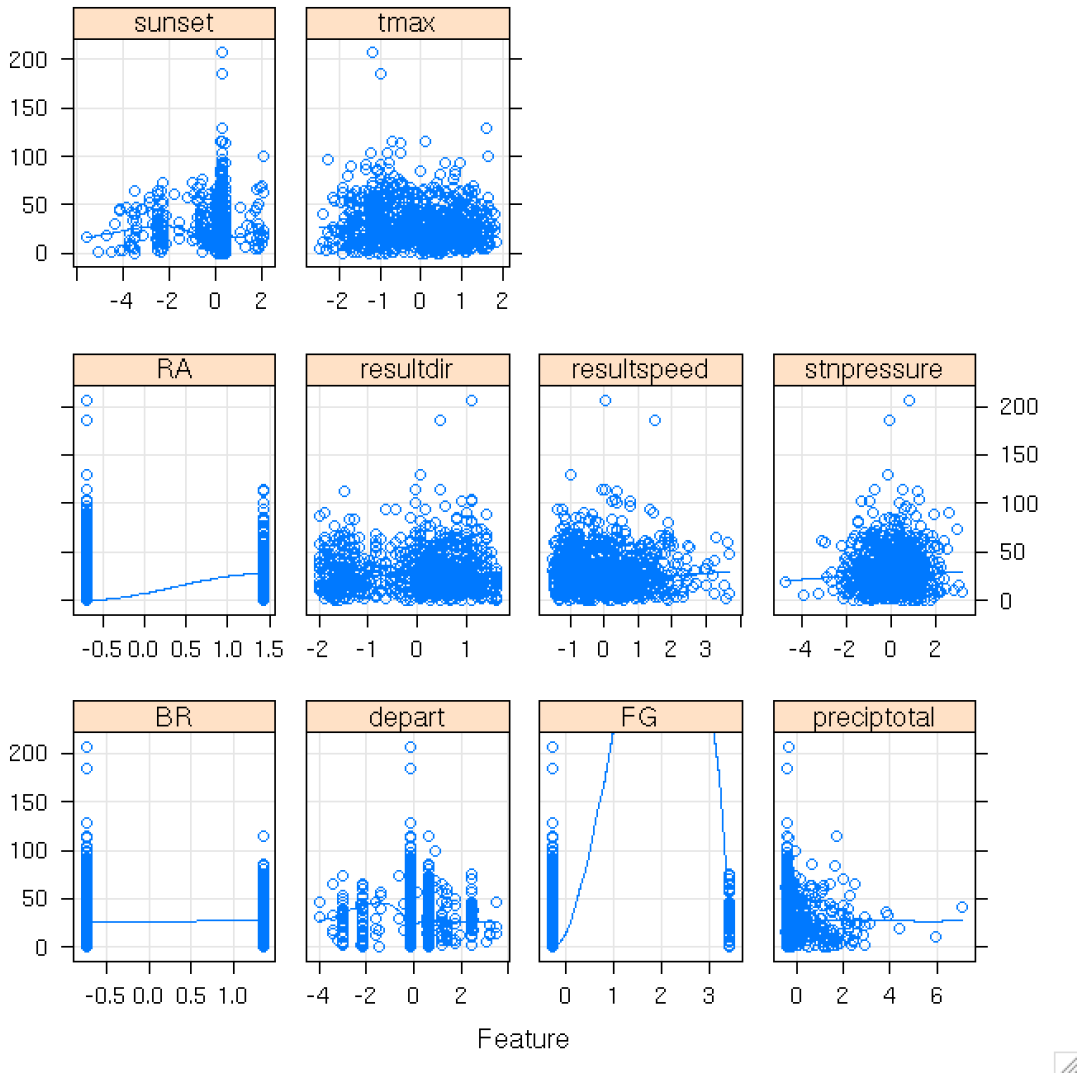


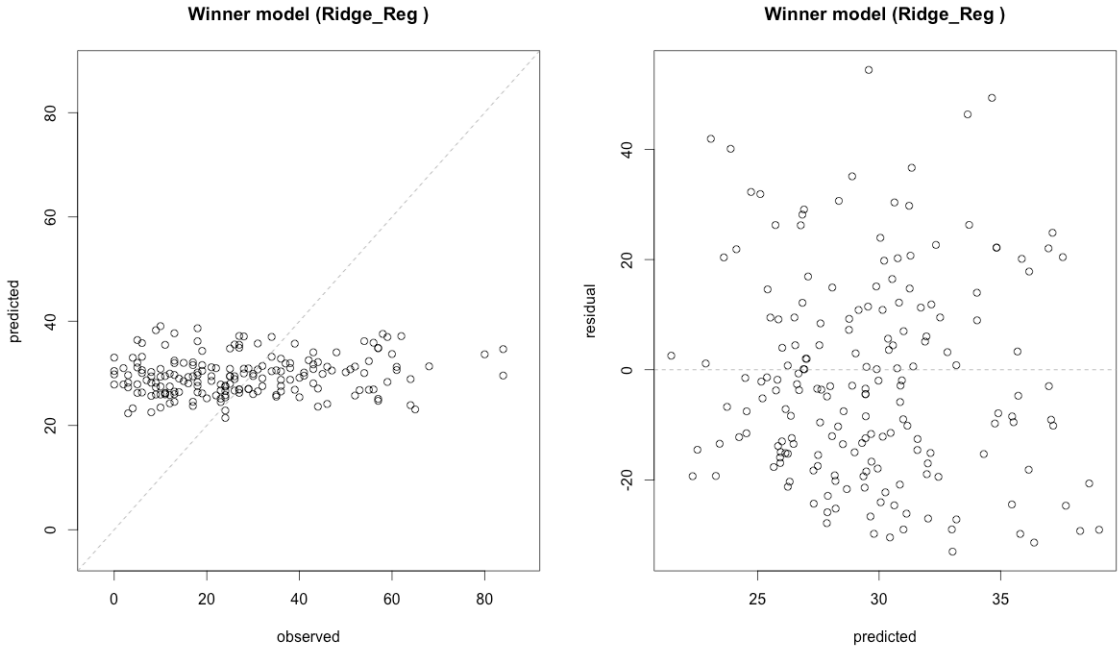
Figure 7 Scatter plots of predictors for store n.1 / product n.9 / weather station n.1 versus units sold

In the scatter plots, features are discarded according to the described feature selection process and scaled.

Isaac

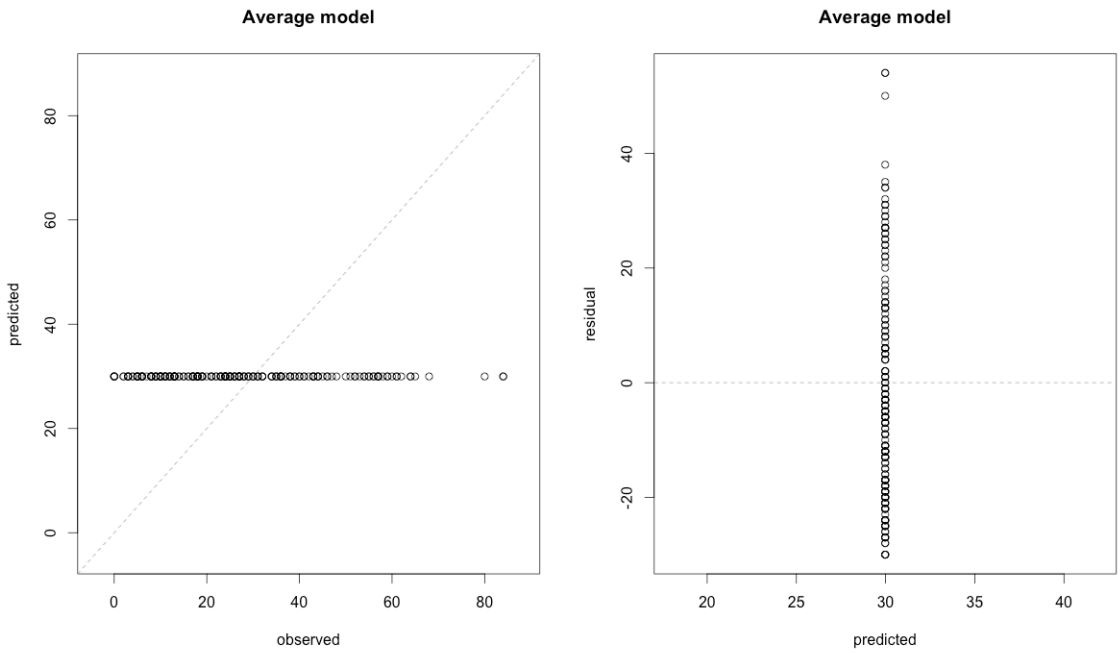
Why do performances look so bad?

Previous scatter plots show a bad correlation between input and output variables. Moreover, looking at the console, the performances of winner models are not so better of the performance of the Average model. How is it possible? When the Average model is so good means that its residuals are very close to white noise. For example, in the previous case the **winner model is Ridge_Reg** and in an ensemble its **RMSE is 18.45031** while **R2 is 0.024** (a very small piece of variability of output variable has been explained by predictors). Simple **correlation between** predicted and observed is **0.156** while **rank correlation 0.154**.



On the other hand, the **Average model has RMSE is 18.70546**.

Isaac



Residuals of the Average model is not exactly white noise but its performance is pretty similar to the performance of the winner model and the predicted vs. observed plot shows undesired similarities to winner model's plot. On the other hand, the residuals plot of winner model does not show any systematic pattern in the model predictions.

Why such performances? The simplest explanation is that such predictors alone cannot explain sold units variability. In fact, **the problem has been modeled so that the number of sold units is a strict function of weather condition indicators and this is a clear limit of the model**. **For example, if one day it's raining and a customer bought a rain jacket, the same jacket can be used for the following raining days for a while**. **So, there is, at least, one important missing information in training data, i.e. potential/actual customers for each pair store/product**. Possible proxies to estimate it are

- Potential customer of the store estimated as the different customer served in a given period (very raw estimation);
- Potential customer of each pair store/product estimated as the different customers who bought such product in a given period (estimation requiring some CRM capabilities);

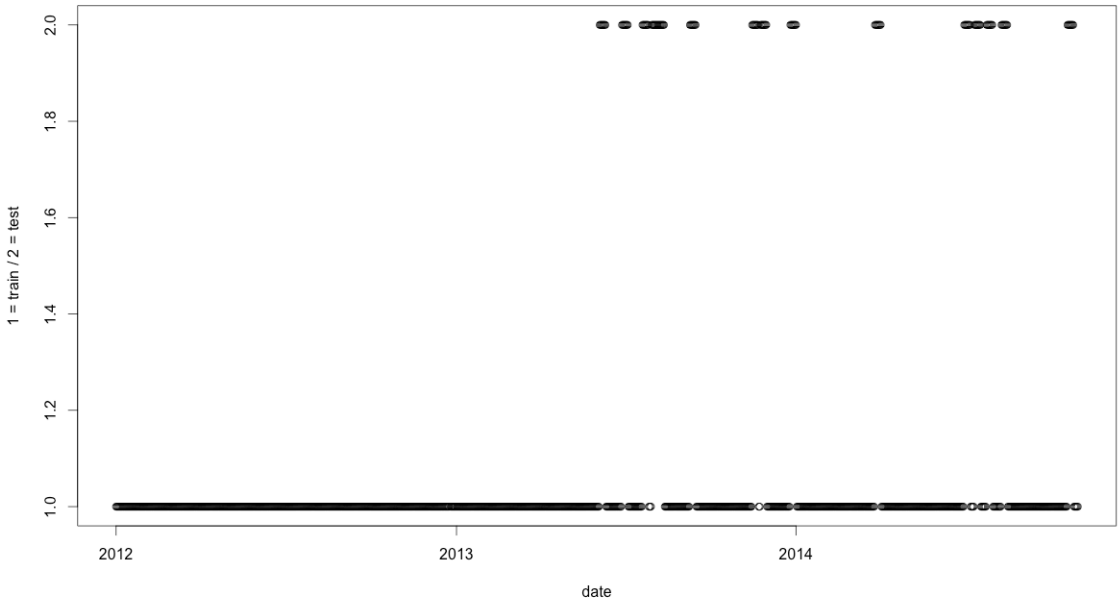
Isaac

- Potential customer of each pair store/product estimated as the different customers who bought such product and the products highly correlated[1] in a given period (estimation requiring some kind of basket analysis).

We can try to estimate this information inductively, but we have study better time periods in training and test data before.

Training dates vs. test dates

How are distributed training dates vs. test dates? For example, referencing at the previous example (store 1, product 9), the following plot shows it.



Moreover, for that pair store/product we find that test dates occur in train dates of some else store but don't occur in train dates of the same store.

[1] In this case bananas can be easily excluded from the analysis, but what about "bananas products" for this product category (e.g. item n. 9 in previous chapters)?

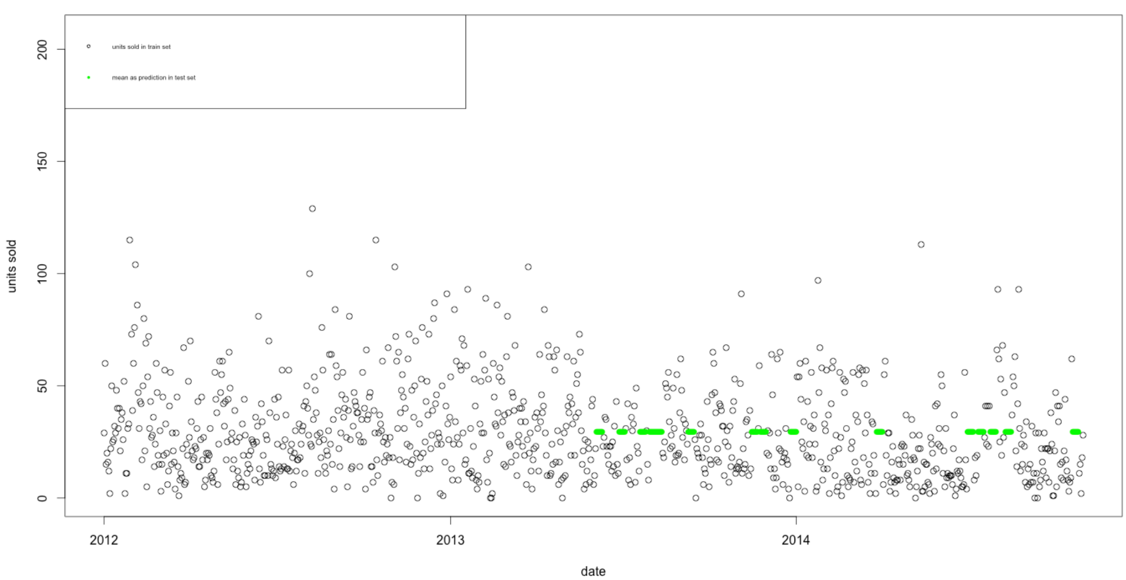
Isaac

```
> train.st = train[train$store_nbr == st , ]
> is_in_train_set = apply(testdata.header , 1 , function(x) ifelse(x[2] %in% train$date , 1 , 0) )
> is_in_train_set
  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 [75] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
> is_in_train_set_same_store = apply(testdata.header , 1 , function(x) ifelse(x[2] %in% train.st$date , 1 , 0) )
> is_in_train_set_same_store
  [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 [75] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
> is_in_traindata_set = apply(testdata.header , 1 , function(x) ifelse(x[2] %in% traindata.header$date , 1 , 0) )
> is_in_traindata_set
  [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 [75] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

This fact can be used in case there's a good correlation between product 9 and the pairs store/product were same test data occur. For instance, the first test data ("2013-06-04") occur in the following stores:

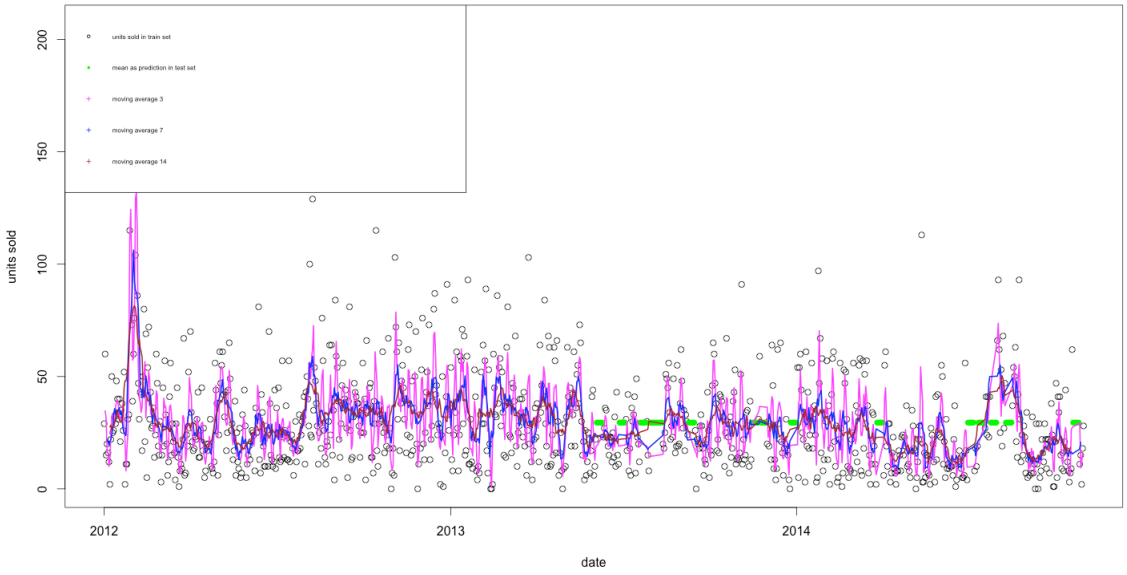
```
> unique(train[which(train$date == "2013-06-04") , ]$store_nbr)
 [1]  4  7  8  9 13 15 17 18 21 23 24 25 26 29 30 31 32 33 34 35 36 37 39 40
```

Coming back to train/test dates comparison, the following plot shows sold units in training set (black points) and its mean as prediction in the test set (green points).

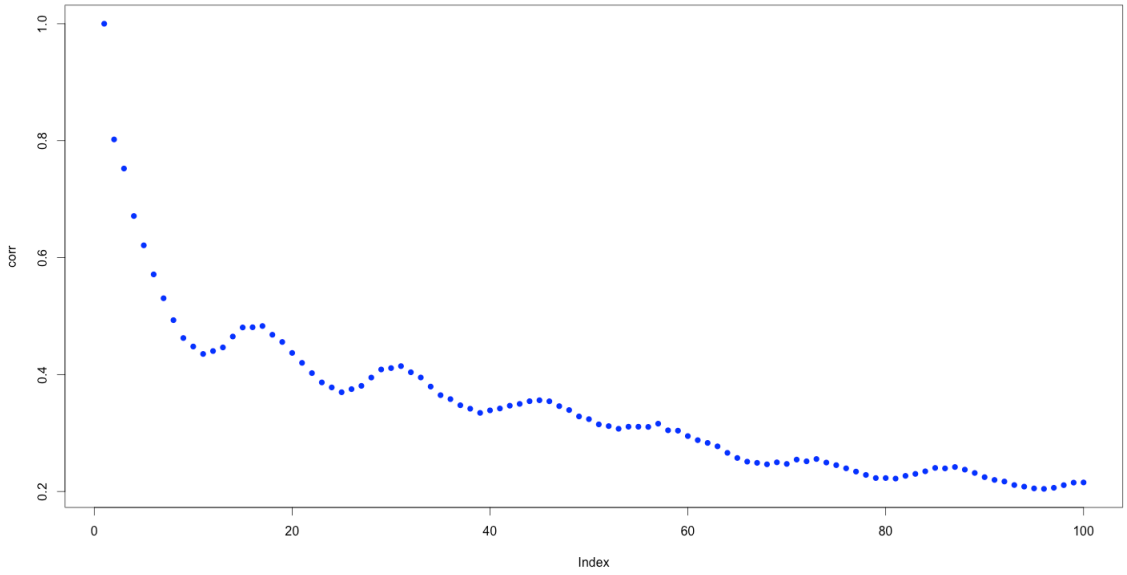


So, if this pattern holds also for other store/product pairs, there is plenty of training points around test points. Hence, in order to estimate potential/actual customer of such pair store/product, we can use for **moving average**.

Isaac



We can estimate the correlation between moving average x and sold units in training data. The following plot shows such a correlation versus time window.



Local minima can be calculated approximately in the following way.

Isaac

```
> candidates = NULL
> for (i in 1:(ww-2)) {
+   if ( (delta[i] > 0 & delta[i+1] < 0) |
+          (delta[i] < 0 & delta[i+1] > 0)) {
+     candidates = c(candidates,i)
+   }
+ }
> candidates
 [1] 10 16 24 30 38 44 52 53 55 56 67 68 69 70 71 72 80 84 85 86 95
> corr[candidates]
 [1] 0.4479530 0.4809073 0.3779937 0.4110607 0.3415908 0.3544222 0.3118441 0.3072924 0.3107852 0.3104560 0.2490121 0.2465134 0.2498823
[14] 0.2471552 0.2545579 0.2516503 0.2230234 0.2345236 0.2403354 0.2395247 0.2052800
>
```

So, moving average 10 is the first min (correlation = 0.44) and moving average 16 is the first local max (correlation = 0.48).

So, at a given date t, let be

- p_i the probability that a customer that does not have the product i can buy it
- C(t) the number of customers that does not have the product buy it;
- $us_i(t)$ the units of product i sold at the time t;
  - notice that $us_i(t) = C(t-1)p_i$
  - notice that $C(t) = C(t-1) - us_i(t-1) + refill(t-1)$
- C_i the number of potential customer that can buy it
- $T_i$the mean time product life cycle
- $refill(t)$ the number of new potential customers or recurrent customers entering at the time t (net old customers)
  - notice that $refill(t) = us_i(t - T_i)$ - *stationary hypothesis*

Hence,

$$us_i(t) + p_i ui_i(t-1) - p_i us_i(t - T_i - 1) = C(t-1)p_i$$

$$us_i(z) = \frac{p_i C(z) z^{-1}}{1 + p_i z^{-1} - p_i z^{T_i + 1}}$$

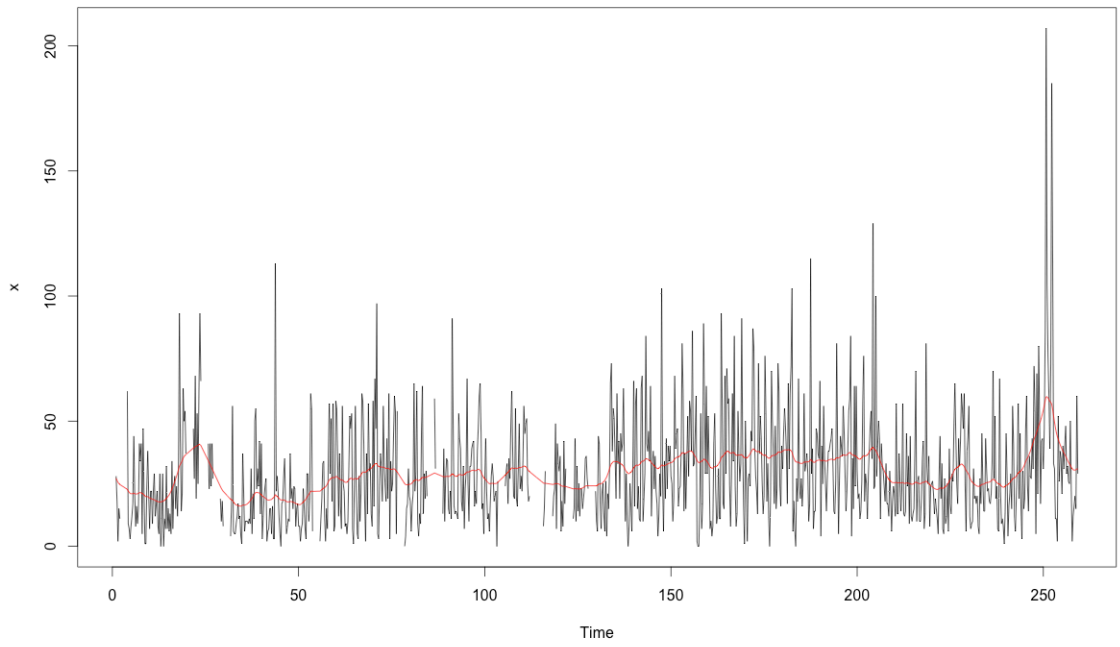
$$C(t-1) = \frac{su_i(t) + p_i[su_i(t-1) - su_i(t-T_i)]}{p_i}$$

Some observations:

- $C(t-1)$ depends, besides $us_i(t)$, on $p_i$and T_i that we don't know
- though we know $us_i(t)$, $us_i(t-1)$and $us_i(t - T_i - 1)$ could be undefined

Isaac

Predictive model #2 – simple interpolation of sold units



Performance on leaderboard after some basic adjustments (see next section): **0.10119**

Isaac

Performance analysis

Let's do a bit of performance analysis on above model.

What are store/item combinations with highest RMSE?

Model # 1

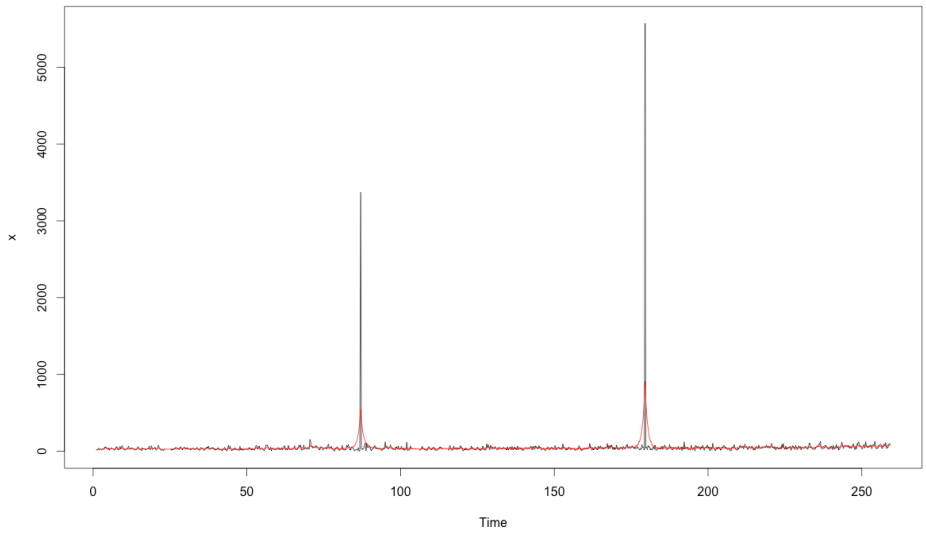
```
> head(model.grid[order(model.grid$best.perf,decreasing = T),])
     store item test.num all0s   Average      Mode LinearReg RobustLinearReg   PLS_Reg Ridge_Reg  Enet_Reg   KNN_Reg BaggedTree_Reg best.perf      best.model
3890    37    5       23 FALSE 139.47365 141.44961 146.78797       137.77080 143.68601 146.35384 139.80595 141.95783      169.36963 137.77080 RobustLinearReg
1824    17   48       95 FALSE  84.38119 116.86696  84.20791        84.02939  84.13210  84.02237  83.80321  86.27312       82.88861  82.88861  BaggedTree_Reg
1785    17    9       95 FALSE  70.81263  74.52034  71.58876        71.54281  71.32566  71.52339  70.95462  73.38761       71.60945  70.81263         Average
3263    30   44      166 FALSE  72.44819  91.35997  68.95346        69.99839  68.95219  68.93799  68.86089  70.44979       67.59354  67.59354  BaggedTree_Reg
3596    33   44      120 FALSE  63.37857  71.69133  63.54614        63.66928  63.65549  63.45739  63.47089  66.33095       63.90117  63.37857         Average
3561    33    9      120 FALSE  63.34929  66.18432  63.74024        63.83506  63.71916  63.61144  63.39921  64.82774       63.27945  63.27945  BaggedTree_Reg
```

Model # 2

```
>
> head (ts.grid[order(ts.grid$best.perf , decreasing = T) , ]  , 15)
     store item test.num all0s Average TS best.perf best.model
3890    37    5       23 FALSE       0  0 203.82617         TS
3561    33    9      120 FALSE       0  0  57.07363         TS
1785    17    9       95 FALSE       0  0  45.51529         TS
1690    16   25      177 FALSE       0  0  43.17491         TS
342      4    9       74 FALSE       0  0  43.07941         TS
3596    33   44      120 FALSE       0  0  42.81222         TS
155      2   44      159 FALSE       0  0  35.32643         TS
4005    38    9      159 FALSE       0  0  35.03315         TS
3263    30   44      166 FALSE       0  0  33.82963         TS
116      2    5      159 FALSE       0  0  33.26251         TS
2673    25    9       23 FALSE       0  0  33.11250         TS
1559    15    5       23 FALSE       0  0  31.91139         TS
227      3    5      138 FALSE       0  0  31.43917         TS
2559    24    6       74 FALSE       0  0  30.74072         TS
1337    13    5      108 FALSE       0  0  30.32725         TS
```

Store n. 37 / item n. 5

Isaac

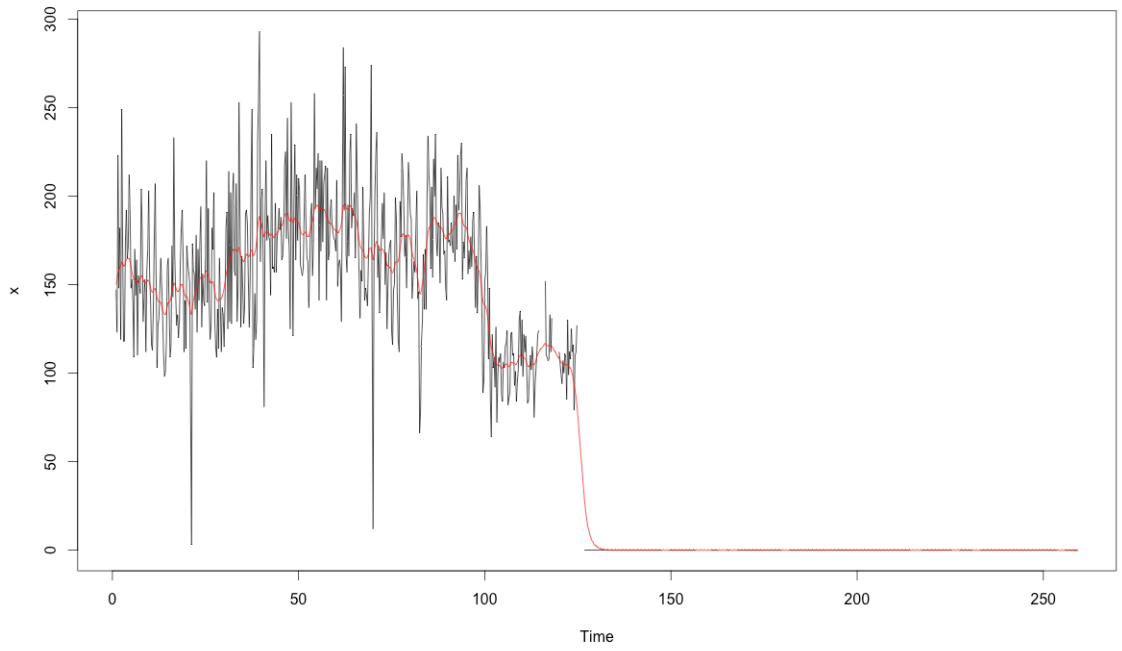


- Estimated RMSE of model # 1 is 137.77
- Estimated RMSE of model # 2 is 211.732

The 2 peeks are not related to weather conditions but they are 2 dates occurring exactly **1 week before thanksgiving days** of such years. Probably, Wal-Mart made some kind of promotions, so as 2014 test set last date is 2014-08-02, the safest choice is to replace such peeks in train set (e.g. with the mean). It's not a surprise that in model # 1 the winner is *Robust Linear regression*.

Isaac

Store n. 17 / item n. 48



Here, before 2013-05-19 sold units were something like 100, while after 2013-05-19 sold units were 0. Probably such an **item has been put out of shelves**. So, here the safer choice is setting up a prediction of 0 for test dates after 2013-05-19 and excluding such data from training set.

Isaac

Few more cases

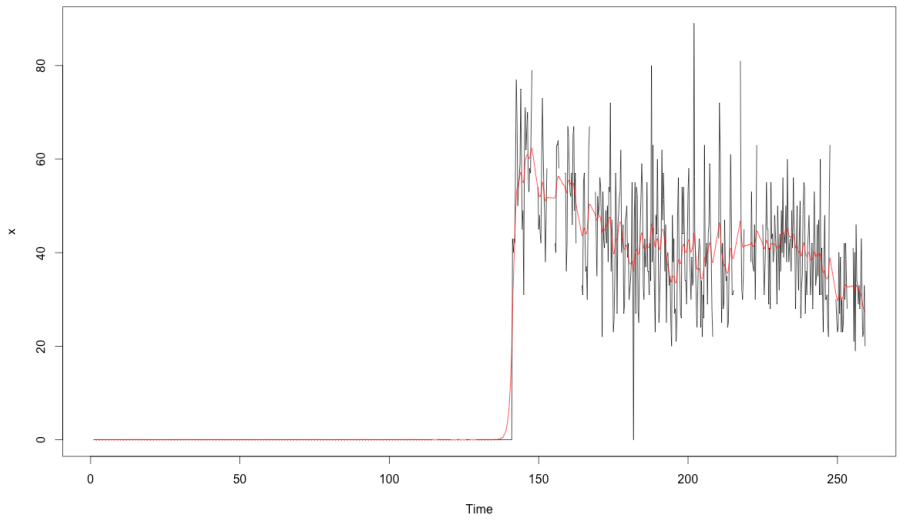
```
> head(data,20)
     store_nbr item_nbr units.mean.train units.mean.sub     delta delta.abs
1824        17       48         81.03514       15.44572  65.58942  65.58942
3263        30       44        157.22696      109.16770  48.05926  48.05926
761          7       95         17.92009       34.52310 -16.60301  16.60301
3708        34       45         92.31151       76.09293  16.21858  16.21858
4001        37        5         49.51830       33.91787  15.60043  15.60043
2780        26        5         45.87962       30.79370  15.08591  15.08591
671          7        5         67.20302       52.98552  14.21750  14.21750
1368        13       36         80.43844       66.82523  13.61321  13.61321
4508        41       68         34.46396       21.27710  13.18687  13.18687
1559        15        5         63.81998       50.79225  13.02773  13.02773
3929        36       44         66.49689       53.47088  13.02600  13.02600
1690        16       25        157.57993      144.90756  12.67237  12.67237
1337        13        5         72.61447       60.14509  12.46938  12.46938
3596        33       44        207.77133      219.27678 -11.50544  11.50544
3235        30       16         39.88825       28.46019  11.42806  11.42806
2114        20        5         79.27902       67.85287  11.42615  11.42615
2399        22       68         36.75167       25.68335  11.06832  11.06832
2954        27       68         32.69710       21.68455  11.01255  11.01255
933          9       45         69.72756       59.04124  10.68632  10.68632
4841        44       68         49.76914       39.51482  10.25433  10.25433
```

Figure 8 - Mean of submitted predictions vs. training data

We can note that in most cases the mean of submitted predictions is minor than the mean of training output variable. This comes from the fact the model, fitting a structural model for a time series by maximum likelihood, isn't able to proper predict its oscillations.

Isaac

Store n. 7 / item n. 95



Same dynamic as store n. 17 / item n. 48. Before 2013-07-15 sold units are 0.

All these adjustments give an **improvement of 0.00076 on the leaderboard**.

Predictive model #3 – model #1 improvement

Looking back at the equation

$$us_i(z) = \frac{p_i C(z) z^{-1}}{1 + p_i z^{-1} - p_i z^{T_i + 1}}$$

that implies

$$us_i(z)\ \alpha\ C(z)$$

As a proxy of $C(z)$ we use time series interpolation.

Isaac

Performed as **0.10887**on leaderboard. **This would suggest that time series interpolation alone gets better performance.**

Predictive model #4 – extreme gradient boosting

Gradient Boosting

1. Compute the average response using time series interpolation, y, and use this as the initial predicted value for each sample
2. **Compute the residual**, the difference between the observed value and the current predicted value, for each sample
3. **Fit a regression model using the winner model of model #1 using the residuals as the response**
4. Predict each sample using the regression model in the previous step
5. Update the predicted value of each sample by adding the previous iteration's predicted value to the predicted value generated in the previous step

Performed on leaderboard as **0.10204**