SEE CLICK PREDICT FIX

Name: José A. Guerrero

Location: Spain

Email: blindape@ono.com

1. Summary

The model for See Click Predict Fix is a blending of 10 models trained with boosted regression trees (gbm), random forest (randomForest), and general linear model (glmnet).

For feature creation I define cities based in latitude and longitude, for stratified training, and grouped similar tag_types. For the text features I used the length and a binary bag of word based in steams of summary terms. For the each issue I compute a leave one out (LOO) geographical radial basis average with three parameters that adjust the shape of influence to city, district and neighbor and a LOO time radial basis average with three parameters that adjust to short, middle and long term.

The models were trained in log scale and the predictions transformed to original scale. Some of the models were trained fitting the three responses all together, stacking three times the data and using binary features to label which responses each data belongs (‘big column approach’).

For model blending I used the ‘BigChaos’ team approach in the Netflix Grand Prize, a ridge regression using several naïve models for calibrating the predictions.

2. Features Selection / Extraction

The first dataset (basic_data) were crude or trivial features:

- day (from 0 to 625)

- latitude

- longitude

- hour (beginning at 6:00 am)

- week_day (beginning on Monday)

- summary_len

- description_len

- city (based in latitude and longitude)

- tag_type (onehot of categories with more than 20 cases)

- source (onehot of categories with more than 20 cases)

Hour and day where adjusted to begin at 6:00 am and Monday for facilitate the splitting of the feature with tree based models.

After the preliminary results, and see how models trained in last 60 days were better than other fitted with more data, I decided focus in feature creation based in time and geographical interaction. The goal was no need use an absolute time reference (a rule of gold for time series), so the model could be trained for a longer period.

The hypothesis was:

- The response to an issue depends (directly or inversely) of number of recent issues and similar issues (time dimension).

- The response to an issue depends (directly or inversely) of number of issues and similar issues reported close (geographic dimension).

- There are geographic zones more sensitive to some issues (geographic dimension).

With that in mind I defined three time windows, (short, middle and long) of 3, 14 and 60 days and three epsilon parameters (0.1, 0.4 and 1.2) for use them in a radial basis distance weighted average for each issue.

The selection of this values were for adjust the decay shape in a way the weights represent city, district and neighbor ambits.

The tag_type were grouped in: crime_n_social, rain_snow, traffic, lights, trees, trash, hydrant, graffiti, pothole, NA, and Other.

For each issue (row) I computed 3 (short, middle, long) x 3 (city, district and neighbor) features for each of 11 tag groups.

Each feature uses radial basis weights respects the distance in kms between issues.

I computed 3 x 3 features for the total of issues and for the issues of the same group, so in total I had 3 x 3 x 13 of such features, all computed with a LOO (Leave One Out) criteria for avoid overfitting. This 117 feature set I named LOO features. LOO features include a FORWARD_WINDOW parameter that controls the days ahead included in the average.

For a real time prediction this parameter must be set to 0. In the current models was set to 7 days, except for short time window that had an effective value of 3 days.

For a period of last 150 days and for each issue I computed the LOO weighted radial basis average for comments, votes and views for (city, district and neighbor) params (9 features) and the same but filtered to the issues in the same group (other 9 features). These 18 features I named BAYES features.

LOO and BAYES featured were normalized to (0,1) range for use them with linear models too.

For summary I computed a binary bag of more frequents words (> 50), naming it BoW.

Several models used data set with only a response and a binary feature for determine which of the three original responses it belongs. This multiplied the row by three, so I named such datasets with ‘big column’ label.

3. Modeling Techniques and Training

I trained 10 models:

basic_3_models_train_60_hold_0: Basic boosted regression trees (gbm) model for each response, trained in last 60 days before the test period.

basic_3_models_train_120_hold_60: Basic gbm model for each response trained in last 120 days before a hold period of 60 days.

gbm_BC_BAYES_train_60_hold_0: Gbm model trained for the last 60 days before test period with ‘big column’ approach and using LOO and BAYES features.

gbm_BC_LOO_train_60_hold_0: Gbm model trained for the last 60 days before test period with ‘big column’ approach and using LOO features.

gbm_BC_train_1000_hold_0: Gbm model trained for all training period with ‘big column’ approach.

gbm_BC_train_60_hold_0: Gbm model trained in last 60 days before test period with 'big column' approach.

gbm_BC_train_60_hold_60: Gbm model trained in last 60 days before a hold period of 60 days with 'big column' approach.

gbm_by_city_BC_BAYES_n_BoW: Gbm model trained by city with 'big column' approach and using BAYES and LOO features and binary Bag of Words based in summary.

glm_BAYES_n_BoW: General linear model trained for each response using BAYES and LOO features and binary Bag of Words based in summary.

RF_BC: Random forest model using BAYES and LOO features and 'big column' approach.

Summary for the models

Model	Trees	Interac. depth	Learning rate	Min obs in node	public error	blending weight
basic_3_models_train_60_hold_0	1080, 11900, 2900	8	0.002	25	0.30792	0.24330
basic_3_models_train_120_hold_60	2700, 10900, 8000	8	0.001*	20	0.33199	-0.04979
gbm_BC_BAYES_train_60_hold_0	2200	8	0.0015	30	0.31381	0.21548
gbm_BC_LOO_train_60_hold_0	2200	8	0.0015	30	0.31803	-0.18694
gbm_BC_train_1000_hold_0	6000	15	0.001	30	0.31259	0.03036
gbm_BC_train_60_hold_0	1700	15	0.002	30	0.30497	0.01126
gbm_BC_train_60_hold_60	3600	15	0.001	20	0.33984	0.00717
gbm_by_city_BC_BAYES_n_BoW		10		25	0.31775	0.49740
glm_BAYES_n_BoW					0.33264	0.09906
RF_BC	2800			5	0.31707	0.19660

In the models with three responses the parameters refer to views, votes and comments
*** For votes 0.002**

In the models tried the 'big column' approach, multitask models trained in all responses all together, were better than train the three responses separately.

For the final blending I used the same method described by A. Toscher and M. Jahrer (BigChaos team) for the Netflix Grand Prize ensemble, introducing some dummies models for each response and city.

Refer to epigraph 7 and 8 for discussion.

4. Code Description

The code contains three R scripts:

run_models.R

It's the main source and do the data preprocess, model fitting and predictions. The parameters DATAPROCESS and ONLYPRED are used for specify a data set preprocessing and the only new predictions mode (case models are fitted previously).

For data preprocess, the new data need be placed in the DATA_PATH directory.

data.R

Do the data preprocessing. For that train.csv and test.csv are stacked and the process run over the alldata data table.

The computing of leave one out features is done for each issue so the CPU for this process is huge (several days).

models.R

Contain the functions for each model and the blending process. For each function the parameter ONLYPRED determines if the model is fitting (FALSE) or only predictions with new data (TRUE).

5. Dependencies

R version 3.0.0 (2013-04-03) -- "Masked Marvel"

Copyright (C) 2013 The R Foundation for Statistical Computing

Platform: x86_64-w64-mingw32/x64 (64-bit)

R packages:

gbm

Version: 2.1

Date: 2013-05-10

Depends: R (>= 2.9.0), survival, lattice, mgcv

License: GPL (version 2 or newer)

URL: http://code.google.com/p/gradientboostedmodels/

Greg Ridgeway <gregridgeway@gmail.com> with contributions by Daniel Edwards, Brian Kriegler, Stefan Schroedl and Harry Southworth.

randomForest

Version 4.6-7

Date 2012-10-16

Depends R (>= 2.5.0), stats

Suggests RColorBrewer, MASS

Author Fortran original by Leo Breiman and Adele Cutler, R port by Andy Liaw and Matthew Wiener.

Maintainer Andy Liaw <andy_liaw@merck.com>

License GPL (>= 2)

URL http://stat-www.berkeley.edu/users/breiman/RandomForests

glmnet

Version: 1.9-5

Depends: Matrix (= 1.0-6), utils

Suggests: survival, foreach

Published: 2013-08-04

Author: Jerome Friedman, Trevor Hastie, Rob Tibshirani

Maintainer: Trevor Hastie <hastie at stanford.edu>

License: GPL-2

URL: http://www.jstatsoft.org/v33/i01/

Matrix

Version: 1.1-1.1

Depends: R (= 2.15.0), methods

Imports: graphics, grid, stats, utils, lattice

Suggests: expm, MASS

Enhances: MatrixModels, graph, SparseM, sfsmisc

Published: 2013-12-30

Author: Douglas Bates and Martin Maechler

Maintainer: Martin Maechler <mmaechler+Matrix at gmail.com>

Contact: Doug and Martin <Matrix-authors@R-project.org>

License: GPL-2 | GPL-3 [expanded from: GPL (= 2)]

URL: http://Matrix.R-forge.R-project.org/

tm

Version: 0.5-9.1

Depends: R (= 2.14.0), methods

Imports: parallel, slam (= 0.1-22)

Suggests: filehash, proxy, Rgraphviz, SnowballC, XML

Published: 2013-07-10

Author: Ingo Feinerer [aut, cre], Kurt Hornik [aut]

Maintainer: Ingo Feinerer <feinerer at logic.at>

License: GPL-2 | GPL-3 [expanded from: GPL (= 2)]

URL: http://tm.r-forge.r-project.org/

data.table

Version: 1.8.10

Depends: R (= 2.12.0)

Imports: methods

Suggests: chron, ggplot2 (= 0.9.0), plyr, reshape, testthat (= 0.4), hexbin, fastmatch, nlme, xts, bit64

Published: 2013-09-03

Author: M Dowle, T Short, S Lianoglou with contributions from A Srinivasan, R Saporta

Maintainer: Matthew Dowle <mdowle at mdowle.plus.com>

BugReports: https://r-forge.r-project.org/tracker/?group_id=240

License: GPL-2 | GPL-3 [expanded from: GPL (= 2)]

URL: http://datatable.r-forge.r-project.org/

6. How To Generate the Solution

For train and predict:

- Download the data (train.csv and test.csv) from Kaggle web and put in DATA_PATH directory.

- Set parameters in run_models.R

    DATAPROCESS = TRUE

    ONLYPRED = FALSE

- Execute run_models.R

(Data process takes several days for LOO and BAYES features, the data.R file can easily run by steps)

- The models are saved in MODEL_PATH directory and the predictions in SUBMISSION_PATH directory

- The final solution is generated in MAIN_PATH \ blending.csv

For predict new data:

- Set parameters in run_models.R

    DATAPROCESS = TRUE

    ONLYPRED = TRUE

- Put the new data in DATA_PATH as test.csv

- Execute run_models.R

(Data process takes several days for LOO and BAYES features)

- The predictions are saved in SUBMISSION_PATH directory

(back up all previous prediction to other fold)

- The final solution is generated in MAIN_PATH \ blending.csv

7. Additional Comments and Observations

Both in the Hackathon and this competition most of solutions uses intensively post processing for calibrate the responses. Basic models trained using only the last months of data performs better than other more elaborates. The problem here is the RMSE metric is very sensible to the centralization and dispersion of estimations and the responses in this problem have a wide range of fluctuations across the time too.

Some of fluctuations could be very difficult to predict (council campaigns, communication media publicity or random events like bad weather), so we may think what is more important, predict the absolute value or views, votes or comments or predict the rank of most voted, viewed or commented issues.

Personally I prefer a solution tell me what of the issues are more important to citizens that other that predict accurately the mean of responses (so obtain better RMSE) with a sorting wrong.

In model benchmarking, the naïve model trained in last 60 days with 'big column', obtain the best individual RMSE score, but this is probably because the mean of responses in that period is the more like the test period.

After model blending and calibrating, we see this model has a low weight in final solution.

As an alternative, a Spearman correlation metric should avoid this problems letting more robust models.

8. Simple Features and Methods

The model with biggest weight in the blending was a boosted regression trees trained by city (four models) and the three responses all together over the data set with LOO, BAYES and BoW features.

Features like **total_ambit_time**, ambit in (neighbour, district, city) and time in (3, 14, 60) refer for each issue to the weighted radial basis average of number total of issues (exclude itself) in this ambit and time window.

Features like **self_ambit_time**, are like previous but restricted to that issues in the same tag group.

Features with the pattern **response_ambit** refer for each issue to the weighted radial basis average of the respective response (exclude itself) in this ambit and last 150 days window.

The words others than 'latitude', 'longitude', 'hour', 'week day'.., refer to its presence in summary feature.

The relative importance of the features in each model was:

New Haven			Oakland	
self_city_3	57.5103542		source_remote_api_created	70.5886549
source_city_initiated	14.4005889		total_city_14	10.8678981
description_len	5.41106037		total_city_60	4.74702205
votes_neighbour	2.41809542		self_city_60	1.95967715
hydrant	2.23298177		description_len	1.2006287
comments_city	2.0938478		votes_neighbour	1.1246164
views_neighbour	1.9499935		longitude	0.79582051
comments_neighbour	1.79910063		comments_neighbour	0.78862138
self_city_14	1.75121041		comments_city	0.72131141
hour	1.22040064		total_city_3	0.57625697
self_neighbour_3	1.18404546		total_district_60	0.5692561
self_city_60	1.17287636		votes_city	0.47212739
source_Map_Widget	0.91348554		total_neighbour_3	0.43800192
votes_city	0.83773061		self_city_3	0.39943562
comments_district	0.8340342		views_neighbour	0.39252176
tag_type_NA	0.77689002		hour	0.32189426
latitude	0.71440239		dump	0.29815804
summary_len	0.64507803		votes_district	0.26163132
total_neighbour_60	0.22743752		self_neighbour_3	0.24225192
views_district	0.22384777		self_city_14	0.24169892
self_district_3	0.19192012		total_district_14	0.19677926
views_city	0.17665966		views_city	0.18386343
total_city_3	0.16565318		summary_len	0.17951221
total_city_14	0.14694636		comments_district	0.17380717
votes_district	0.14177251		self_district_60	0.13314444
total_neighbour_14	0.10966402		views_district	0.13280931
longitude	0.09135665		self_neighbour_60	0.12295792
tag_type_signs	0.07770032		total_neighbour_60	0.1202092
total_neighbour_3	0.07210518		encamp	0.11498122
self_neighbour_14	0.06953183		self_district_3	0.10934491
total_district_3	0.05551709		light	0.10412886
total_district_14	0.0522375		burn	0.09946492
total_city_60	0.04714188		self_neighbour_14	0.09460444
fire	0.0441259		latitude	0.08559927
plow	0.03578485		self_district_14	0.08109303
trim	0.03239202		total_neighbour_14	0.07562828
tag_type_snow	0.02402253		week_day	0.0678867
self_district_14	0.02335956		blight	0.06644565
total_district_60	0.0217281		illeg	0.06128155
self_neighbour_60	0.02007762		tag_type_NA	0.05776785
week_day	0.02006662		total_district_3	0.04973292
self_district_60	0.01208355		tag_type_graffiti	0.04120876
source_iphone	0.01106482		tag_type_flood	0.03954926
lamp	0.00836172		source_web	0.03700854

Richmond			Chicago	
hour	22.6714263		source_remote_api_created	53.0848639
description_len	17.1163241		week_day	18.7691476
views_neighbour	11.6636911		hour	8.98960319
week_day	7.18206647		description_len	3.68877314
votes_neighbour	6.52605075		views_neighbour	2.96078096
total_city_60	6.23506373		self_city_3	2.69925693
self_city_60	5.30892201		self_city_60	2.38527818
tag_type_pothole	5.12479466		self_city_14	2.21196364
self_city_14	4.03809253		total_city_60	1.32345663
summary_len	1.91001589		views_city	0.90483946
views_district	1.87938608		pothol	0.76971471
bulk	1.49221946		views_district	0.54946498
total_city_14	1.35691351		total_district_60	0.44034048
tag_type_NA	1.24166363		total_city_3	0.38411344
total_city_3	1.14290231		total_district_3	0.29076209
self_city_3	0.8413175		self_neighbour_3	0.16393428
total_district_60	0.55348304		self_district_14	0.09553862
self_district_60	0.51232034		total_neighbour_3	0.09090213
brush	0.4858022		total_city_14	0.08225538
source_android	0.41038278		votes_city	0.04333453
source_iphone	0.39084173		self_district_3	0.02729601
light	0.38871829		total_district_14	0.01684477
street	0.19486881		total_neighbour_14	0.01588375
self_neighbour_3	0.18699767		self_neighbour_14	0.01165126
votes_district	0.16753478			
comments_city	0.16308104			
comments_neighbour	0.10939179			
views_city	0.07932902			
pickup	0.07567421			
longitude	0.06583112			
latitude	0.0624807			
votes_city	0.06130808			
total_neighbour_60	0.06087401			
total_district_14	0.05978843			
source_city_initiated	0.05171079			
comments_district	0.04081752			
nonfunct	0.03866569			
total_neighbour_3	0.03203524			
tag_type_street_light	0.01822534			
tag_type_tree	0.01596874			
alley	0.00966552			
total_neighbour_14	0.0089688			
total_district_3	0.00541582			
self_district_14	0.00524614			

9. References

A. Toscher and M. Jahrer. The BigChaos Solution to the Netflix Grand Prize. 2009.

Software used:

R version 3.0.0 (2013-04-03) -- "Masked Marvel"

R Core Team (2013). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria.

http://www.R-project.org/

Gbm

Greg Ridgeway with contributions from others (2013). gbm: Generalized Boosted Regression Models.

R package version 2.1.

http://CRAN.R-project.org/package=gbm

randomForest

A. Liaw and M. Wiener (2002). Classification and Regression by randomForest. R News 2(3), 18--22.

Glmnet

Jerome Friedman, Trevor Hastie, Robert Tibshirani (2010). Regularization Paths for Generalized Linear Models via Coordinate Descent. Journal of Statistical Software, 33(1), 1-22.

http://www.jstatsoft.org/v33/i01/

Matrix

Douglas Bates and Martin Maechler (2013). Matrix: Sparse and Dense Matrix Classes and Methods.

R package version 1.0-12.

http://CRAN.R-project.org/package=Matrix

tm

Ingo Feinerer, Kurt Hornik, and David Meyer (2008). Text Mining Infrastructure in R. Journal of Statistical Software 25(5): 1-54.

http://www.jstatsoft.org/v25/i05/

data.table

M Dowle, T Short and S Lianoglou (2013). data.table: Extension of data.frame for fast indexing, fast ordered joins, fast assignment, fast grouping and list columns..

R package version 1.8.8.

http://CRAN.R-project.org/package=data.table