Our milestone 3 solution of the Heritage Health Prize

September 26th, 2012

Edward de Grijs Willem Mestrom

1 Introduction

In this paper we describe our milestone 3 solution to the Heritage Health. The solution is a blend of 29 models, 27 of them were already used in our milestone 2 solution. The other 2 models consists of one model described in chapter 2, and one additional 'optimized constant value' models described in chapter 3. Finally the final blend is given in chapter 4.

2 Stochastic Gradient Descent Model

One additional file in the blend is the model combination as described in our milestone 2 document. It is combination 4, of the table presented in paragraph 3.1.

As described in paragraph 3.11 of the milestone 2 document these four combinations were blended, but it showed that simply using this model in the final blend separately improved the overall score significantly. This model alone gave a public leaderboard score of 0.4586, just 0.0001 above the blend leaderboard score. Notice in chapter 4 that the weight of the c279 blend reduced from 0.413 (see our milestone 2 document) to 0.270, while this c279_comb4 model alone gets a weight of 0.220

3 Optimizing constant value

3.1 Observations

In the round 1 milestone prize paper of the team “Market Makers” it was noticed in figure 4 of this document that Y3 contains a large number of paydelay=0, while this was not present in Y2 and Y1. We now try to investigate this further, and create and additional constant value.

If we take another look at the distribution of paydelay in relation to the “Days since first claim”, the DSFS column of the Claims data file, we can observe the following distribution (see table 1):

DSFS: month: paydelay:	0-1 Jan	1-2 Feb	2-3 Mar	3-4 Apr	4-5 May	5-6 Jun	6-7 Jul	7-8 Aug	8-9 Sep	9-10 Oct	10-11 Nov	11-12 Dec
0....0	72	40	33	40	50	82	86	96	147	279	682	**16044**
1...10	56	41	56	36	38	12	3	5	11	12	30	152
11...20	494	330	273	240	142	145	120	111	373	451	338	1258
21...30	759	498	518	580	572	606	552	591	594	480	441	327
31...40	366	244	180	247	170	225	207	194	170	214	148	**0**
41...50	280	171	128	146	125	152	129	155	128	139	54	0
51...60	214	77	83	81	58	123	114	103	69	74	9	0
61...70	91	46	31	33	65	70	89	64	47	66	**0**	0
71...80	45	55	49	32	74	46	59	34	54	35	0	0
81...90	27	23	35	36	36	28	22	22	37	3	0	0
91..100	35	22	28	17	30	17	15	10	18	0	0	0
101..110	35	17	23	20	15	11	10	6	1	0	0	0
111..120	29	10	13	14	11	13	5	1	0	0	0	0
121..130	16	4	11	6	4	11	5	4	0	0	0	0
131..140	6	6	8	2	6	7	1	5	0	0	0	0
141..150	9	3	6	2	3	7	2	0	0	0	0	0
151..161	8	6	1	1	4	4	2	0	0	0	0	0
162..162	59	25	16	20	19	2	**0**	0	0	0	0	0

Tabel 1: Claims distribution of paydelay versus DSFS.

In table 1 the number of claims is counted for each DSFS month, and paydelay range.
The paydelay range for the 0….0 row only counts paydelay=0 claims, while the 162..162 row only counts the paydelay=162+ row.
Important is that only claims are used that belong to members of Year3, that have a maximum DSFS value of 11-12month, because we then know to which real month all the claims belong.

From table 1 we can observe that:

1) There are many claims where paydelay=0, but in the last month (11-12) almost all the claims have a paydelay=0

2) A paydelay in-between 1..10 happens sporadic. It is plausible to assume that almost every payment takes more than 10 days.

3) In the month (11-12) there are no paydelay values larger then 30.
   In the month (10-11) there are no paydelay values larger than 60.

4) From month (6-7) there are no paydelay values of 162+ anymore.

It looks like every payment in Year4 led to a paydelay of zero: In fact for December (month11-12) the paydelay can be 30 at maximum if the treatment took place on the first day of December. For November (month10-11) the paydelay can be 60 at maximum, which is also consistent with table 1. A treatment in July (6-7months) probably resulted in paydelay=0 if the paydelay originally was 162+, because the payment took place in Year4.

These observations enable us to predict more accurately the last DSFS month. We do not know the last DSFS month, because DSFS is a relative value in relation the the first claim made in that year. For instance a MemberID with only claims in DSFS=0-1months could be January, but also December of that year (or every month in-between).

If a MemberID has only claims in for instance DSFS=0-1months and 1-2months, but there are paydelay values of zero, than there is a large chance that the DSFS=1-2months is in reality December. This is important to know, because claims made in January and February of a certain year are less likely to cause DaysInHospital the next year, then claim made in November and December.

We can now select the Members that in Year3 have a maximum DSFS value of 9-10month (or lower), while having a paydelay=0 in the last DSFS month. By offsetting the DaysInHospital for these members with a constant value, a slightly better prediction is reached.
This is performed with the file “m13” as presented in the final blend of chapter 4.

3.2 Modeling

A somewhat better result can be reached by modeling the knowledge from the previous paragraph. We are now able to predict the real last DSFS month, by means of the following calculations:

Definitions:
$Claims_m$: Claims of member m in year Y3
$Cmon_{m,c}$: Month of claim c of member m. Claim months are numbered from 1 to 12, so 1 = month0-1, 2 = month1-2, etc.
$paydelay_{m,c}$: Paydelay for claim c of member m.
$Rmon$: Real months are numbered from 1 to 12, where 1=January, 2=February, etc.

First we calculate $CmonMin_m$ and $CmonMax_m$ for all members who have claims in Year3, only using the claims in Year3:

$$CmonMin_m = \min(Cmon_{m,c})$$
$$CmonMax_m = \max(Cmon_{m,c})$$
$$NrClaims_m = \#Claims_m$$

To calculate $RmonMin_m$, the supposed minimum month (Jan, Feb, etc.), it is assumed that the pay cannot fall in Year4, because in that case the paydelay would be zero.
This also depends on the $CmonMax_m - Cmon_m$ value, because this adds to the amount of available months, and so to a longer possible paydelay.
The used formula is used only for those claims where $paydelay_{m,c} > 0$:

$$RmonMin_m = \min(12.37 + CmonMax_m - Cmon_{m,c} - paydelay_{m,c}/22.50)$$

Note that in the formula above, the paydelay is divided by 22.50, and not 30, which was to be expected, because a month has about 30 days. These values were derived from optimizations, as we will explain in the next paragraph. $RmonMin_m$ is initialized on 12 (=December)

Then for all the claims in the last month for each member ($CmonMax_m$) we calculate:

$$NrPay0_m = \#\{c \in Claims_m \mid Cmon_{m,c} = CmonMax_m \wedge paydelay_{m,c} = 0\}$$
$$NrPayNot0_m = \#\{c \in Claims_m \mid Cmon_{m,c} = CmonMax_m \wedge paydelay_{m,c} > 0\}$$

Now we have obtained the following variables:

$NrClaims_m$, $CmonMin_m$, $CmonMax_m$, $RmonMin_m$, $NrPay0_m$ and $NrPayNot0_m$

With the values of these variables for each member in Y3 we will calculate our first estimate of the real month of the last claim

$$RmonMax1_m = 6.15 \ + \ 0.5 \cdot (CmonMax_m - \ CmonMin_m)$$

We now want to calculate how much the $RmonMax1$ variable can be trusted, because with fewer $NrClaims$ it is more uncertain.

$$interpolate_m = \begin{cases} 0.237 & \text{if } NrPayNot0_m > 0 \\ 0 & \text{if } NrPayNot0_m = 0 \end{cases} + \begin{cases} 0.296 & \text{if } NrClaims_m = 1 \\ 0.201 & \text{if } NrClaims_m = 2 \\ 0.143 & \text{if } NrClaims_m = 3 \\ 0.119 & \text{if } NrClaims_m = 4 \\ 0.000 & \text{if } NrClaims_m > 4 \end{cases}$$

The first term could be described as: if there are claims in the last month with $paydelay_{m,c} > 0$ then the real month is before December, and so the $RmonMax1$ value can be trusted more.

We will use this interpolation with some extra trust requirements to get our second estimate of the real month of the last claim:

$$\alpha_m = \begin{cases} interpolate_m & \text{if } NrPay0_m > 3 \ \vee \ NrPay0_m > NrPayNot0_{\ m} \\ 1 & otherwise \end{cases}$$

$$RmonMax2_m = \alpha_m \cdot RmonMax1_m + (1 - \alpha_m) \cdot 12$$

Finally some boundaries are placed upon the $RmonMax2_m$ values, in case the calculations of $RmonMax2$ above are too far off, yielding our final estimate:

$$RmonMax_m = \min\left(RmonMin_m, \max(RmonMax2_m, \ CmonMax_m)\right)$$

Now all the members are selected that have a maximum DSFS value of 9-10month (or lower), while the predicted “real” last month is larger than 10.5:

$$MSelected \ = \{m \in members \mid CmonMax_m < 10 \ \vee \ \ RmonMax_m > 10.5\}$$

For all the selected members a constant offset value is added to the DaysInHospital value, by which a better prediction is reached.
This is done using the file “m13” as presented in the final blend of chapter 4

3.3 Model parameter optimalisation

The model of the previous paragraph is optimized using the Year3 data of all members which have a $CmonMax_m$ = 12 (this is month11-12), because then all the claim months are equal to the real months.

The same formulas are used as in paragraph 3.2, but now parameters are defined for each constant in the various formulas.

A fitness value is calculated as a RMSE between the (known) upper real month ($Rmon$), and the predicted upper claims months ($RmonMax$) of the previous paragraph, where a number of months are chosen from the available months: The start month and upper claims month ($CmonMax$) are randomly selected out of the available months, as if certain months are not available. (Naturally, the start month has to be lower or equal to the upper claims month).

By means of a simple stepwise optimization the parameters are optimized for a minimum RMSE.

4 Final blend

The final solution is a blend of 29 models, 27 of them were already used in our milestone 2 solution. On additional model is the submitted combination 4 (see paragraph 2.1), which was also part of the very strong model (c279) as described in chapter 2. The other additional file is a new 'optimized constant value' model as described in chapter 3. The weights were calculated using the same procedure as used for milestone 2, but due to the additional files the weights differ.

After the final blend, all the output values were capped on 0.04: Lower values were increased to 0.04, as described in the "MarketMakers milestone 1 document".

Model	Algorithm	RMSLE (Leaderboard)	Weight
CatVec1	SDG	0.4758	0.048
CatVec2	SDG	0.4666	-0.116
CatVec3	SDG	0.4666	-0.075
SigCatVec1	SDG	0.4644	0.142
SigCatVec2	SDG	0.4657	-0.082
PerClaim	SDG	0.4640	0.094
SigCatVec5	SDG	0.4625	0.188
SigCatVec3c-Y3	SDG	0.4750	0.256
SigCatVec3b	SDG	0.4656	-0.133
SigCatVec7	SDG	0.4645	0.143
SigCatVec6	SDG	0.4633	-0.069
SicClaimVec7	SDG	0.4606	0.216
GBM2	GBM	0.4626	0.064
c279	TreeEnsemble + SGD	0.4585	0.270
c279_comb4	TreeEnsemble + SGD	0.4586	0.220
all mean	average	0.4865	-0.271
m1	average	0.4913	0.082
m2	average	0.4904	-0.059
m3	average	0.4844	-0.382
m4	average	0.4811	0.128
m5	average	0.4854	-0.198
m6	average	0.4833	0.070
m7	average	0.4876	-0.089
m8	average	0.4855	0.092
m9	average	0.4840	0.091
m10	average	0.4886	0.093
m11	average	0.4844	-0.079
m12	average	0.4831	0.138
m13	average	0.4888	0.186

Tabel 2: Scores and weights of models in the final blend