Classification Challenge by KrediTech

Dmytro Dudenko

3. Mai 2016

Let's see and touch a bit the training data

```
##     v9            v17            v29          v20          v41
##  a   :1279   23,25  :  64   0      : 134   l   :  32   g   :3055
##  b   :2382   22,67  :  52   3e-04  : 121   u   :3055   gg  :  32
##  NA's:  39   19,17  :  49   0,00065:  98   y   : 549   p   : 549
##              20,42  :  49   0,00015:  90   NA's:  64   NA's:  64
##              25,17  :  49   5e-04  :  87
##              (Other):3398   5e-05  :  86
##              NA's   :  39   (Other):3084
##       v31            v36            v19        v2        v37
##  c      : 787   v      :2015   0      : 268   f: 529   f:1461
##  q      : 612   h      : 970   1,5    : 169   t:3171   t:2239
##  W      : 363   bb     : 339   0,04   : 127
##  cc     : 343   ff     : 154   2,5    : 115
##  x      : 340   z      :  49   1      : 111
##  (Other):1189   (Other): 107   0,25   :  97
##  NA's   :  66   NA's   :  66   (Other):2813
##       v12         v7        v27      v21           v39
##  Min.   : 0.00   f: 276   f:1924   g:3430   Min.   :   0.0
##  1st Qu.: 0.00   t:3424   t:1776   p:  81   1st Qu.:   0.0
##  Median : 2.00                     s: 189   Median : 120.0
##  Mean   : 4.16                              Mean   : 162.7
##  3rd Qu.: 6.00                              3rd Qu.: 280.0
##  Max.   :67.00                              Max.   :1160.0
##                                             NA's   :100
##       v34              v18             v35       classLabel
##  Min.   :     0   Min.   :       0   f   : 532   no. : 276
##  1st Qu.:     0   1st Qu.:       0   t   :1023   yes.:3424
##  Median :   113   Median : 1200000   NA's:2145
##  Mean   :  2247   Mean   : 1626950
##  3rd Qu.:  1060   3rd Qu.: 2800000
##  Max.   :100000   Max.   :11600000
##                   NA's   :100
```

```
## 'data.frame':    3700 obs. of  19 variables:
##  $ v9        : Factor w/ 2 levels "a","b": 1 2 2 1 2 1 1 2 2 2 ...
##  $ v17       : Factor w/ 288 levels "13,75","15,17",..: 17 10 142 240 153 171 98 48 123 20 ...
##  $ v29       : Factor w/ 179 levels "0","0,000104",..: 160 153 5 9 40 8 152 176 154 95 ...
##  $ v20       : Factor w/ 3 levels "l","u","y": 2 3 2 2 2 3 2 3 2 2 ...
##  $ v41       : Factor w/ 3 levels "g","gg","p": 1 3 1 1 1 3 1 3 1 1 ...
##  $ v31       : Factor w/ 14 levels "aa","c","cc",..: 2 9 6 7 9 7 8 2 2 2 ...
##  $ v36       : Factor w/ 9 levels "bb","dd","ff",..: 8 8 3 7 8 4 5 4 8 4 ...
##  $ v19       : Factor w/ 118 levels "0","0,04","0,085",..: 53 8 1 9 13 13 1 7 8 27 ...
##  $ v2        : Factor w/ 2 levels "f","t": 1 1 1 1 1 1 1 1 1 1 ...
##  $ v37       : Factor w/ 2 levels "f","t": 2 1 2 1 1 1 1 1 1 1 ...
##  $ v12       : int  1 0 1 0 0 0 0 0 0 0 ...
##  $ v7        : Factor w/ 2 levels "f","t": 1 1 1 1 1 1 1 1 1 1 ...
##  $ v27       : Factor w/ 2 levels "f","t": 2 1 1 1 2 2 2 1 1 1 ...
##  $ v21       : Factor w/ 3 levels "g","p","s": 1 3 1 1 1 1 1 1 1 1 ...
##  $ v39       : int  80 200 96 0 232 160 276 280 220 320 ...
##  $ v34       : int  5 0 19 120 0 0 1 204 140 13 ...
##  $ v18       : num  800000 2000000 960000 0 2320000 1600000 2760000 2800000 2200000 3200000 ...
##  $ v35       : Factor w/ 2 levels "f","t": 2 NA 2 NA 1 1 NA NA NA NA ...
##  $ classLabel: Factor w/ 2 levels "no.","yes.": 1 1 1 1 1 1 1 1 1 1 ...
```

Some of the features are factors (and some of them are skewed). Other features are numeric and intereger.

Firstly, one shall load all libraries one might need during data play. Also, parallel run would be a good idea as training a neural network (avNNet or nnet) is simply too heavy for my laptop.

These features look like numbers, let's make them so.

Now one should get rid of NA's. There are two ways: either call complete.cases or do imputation. As the dataset is not that big, I would rather prefer to keep not clean records, we may desperately need them for nnet or rf training…Therefore, calling imputation

The training data is quite skewed, "no." is marginally present.

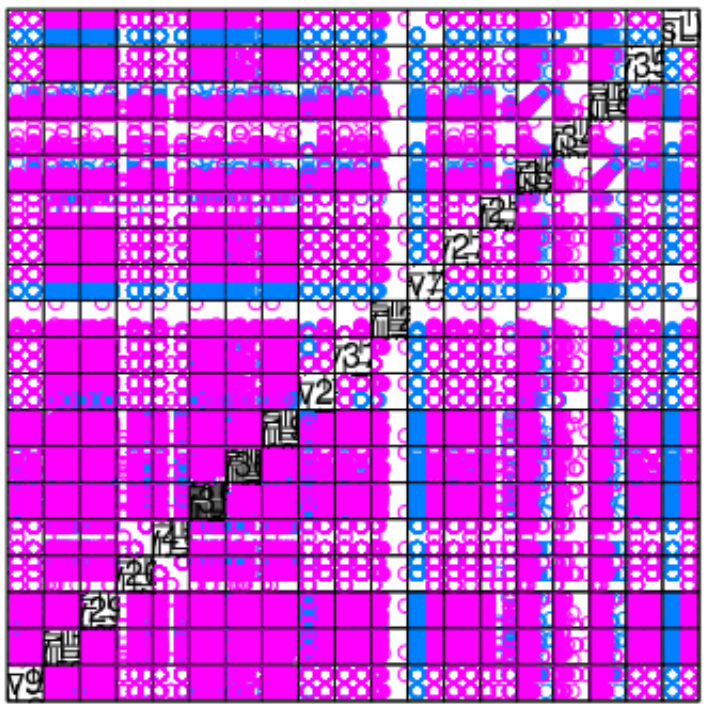
```
##
##  no. yes.
##  276 3424
```

```
##
##  no. yes.
##  107   93
```

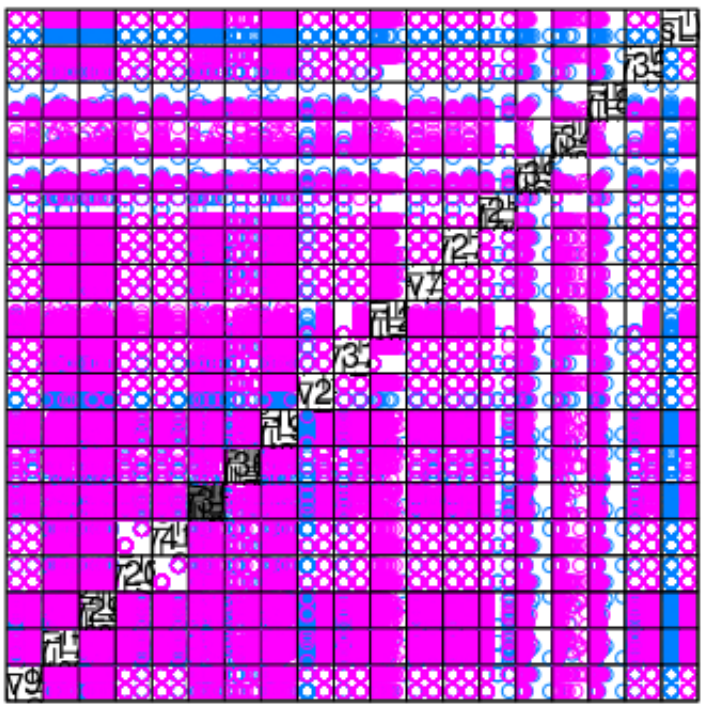
Here just trying to spot features with near to zero variation…None is detected.

```
##           freqRatio percentUnique zeroVar   nzv
## v9         2.118644     0.3623188   FALSE FALSE
## v17        1.125000    49.8188406   FALSE FALSE
## v29        1.000000    29.7101449   FALSE FALSE
## v20        3.459016     0.5434783   FALSE FALSE
## v41        3.508197     0.5434783   FALSE FALSE
## v31        1.646154     2.5362319   FALSE FALSE
## v36        2.495935     1.6304348   FALSE FALSE
## v19        2.103448    18.4782609   FALSE FALSE
## v2         1.253061     0.3623188   FALSE FALSE
## v37        1.369099     0.3623188   FALSE FALSE
## v12        5.696429     3.2608696   FALSE FALSE
## v7         1.000000     0.3623188   FALSE FALSE
## v27        1.290456     0.3623188   FALSE FALSE
## v21       11.386364     0.5434783   FALSE FALSE
## v39        3.516129    23.7318841   FALSE FALSE
## v34       10.173913    28.4420290   FALSE FALSE
## v18        3.516129    23.7318841   FALSE FALSE
## v35        1.555556     0.3623188   FALSE FALSE
## classLabel 1.000000     0.3623188   FALSE FALSE
```

We need to have a first glance at all-to-all features correlations. On the left the plot corresponds to the training set and on the right, to the validation one. First outstanding things to notice are that v18 is just a multiple of v39 and therefore is redundant. Second interesting thing is that v7 has 100% correlation with the most important thing - classLabel. And it can be seen from the validation plot, this feature is poisonous and totally misguiding. This v7 feature should be excluded, otherwise the power of the predictor will be similar to flipping a coin (50% chance) as f and t factors are equally populated in the validation set (in feature v7). Just a small remark, it becomes evident below that visualising 19 features is a tough job. However, in this report I plotted these matrices just for the purpose of first feeling. After this, one can plot specific regions for thorough understanding. Anyway, excluding feature by feature, the plots will be seen better.



Scatter Plot Matrix



Scatter Plot Matrix

Now I present my random forest classifier. Currently it has 90% accuracy. I played also with other methods, namely, svm, knn, glm, and nnet as well as its avNNet version. All of them result in somewhat lower accuracy, which is varying between 80-90%. Not bad after all. A small remark: creating new features (logarithmic or polynomial) didn’t help to achieve better accuracy. Sadly.

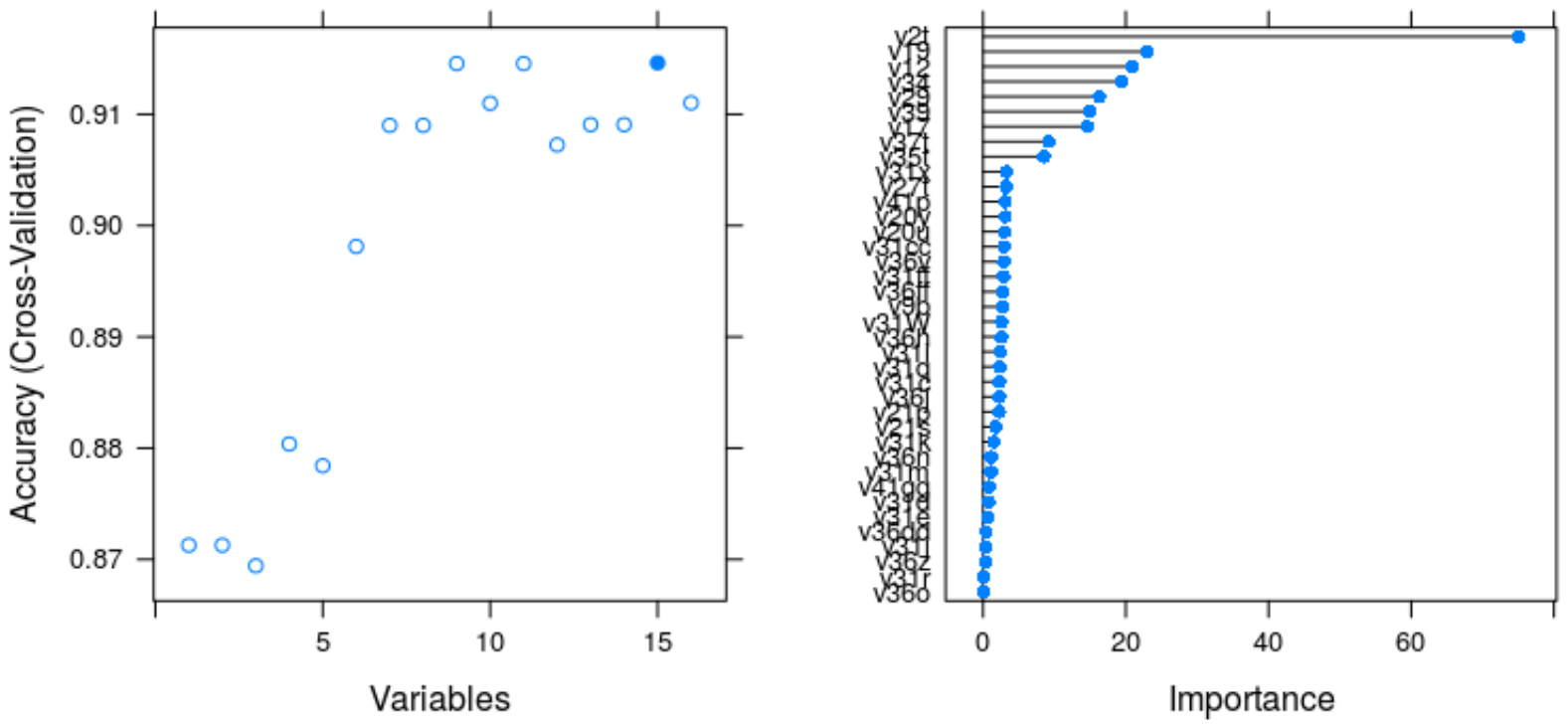
```
## Confusion Matrix and Statistics
##
##          Reference
## Prediction no. yes.
##       no.   97    8
##       yes.  10   85
##
##               Accuracy : 0.91
##                 95% CI : (0.8615, 0.9458)
##     No Information Rate : 0.535
##     P-Value [Acc > NIR] : <2e-16
##
##                  Kappa : 0.8194
##  Mcnemar's Test P-Value : 0.8137
##
##             Sensitivity : 0.9065
##             Specificity : 0.9140
##          Pos Pred Value : 0.9238
##          Neg Pred Value : 0.8947
##              Prevalence : 0.5350
##          Detection Rate : 0.4850
##    Detection Prevalence : 0.5250
##       Balanced Accuracy : 0.9103
##
##        'Positive' Class : no.
##
```

So, after getting roughly 90% of accuracy, one could focus on features, understand them individual and collaborative role and importance.

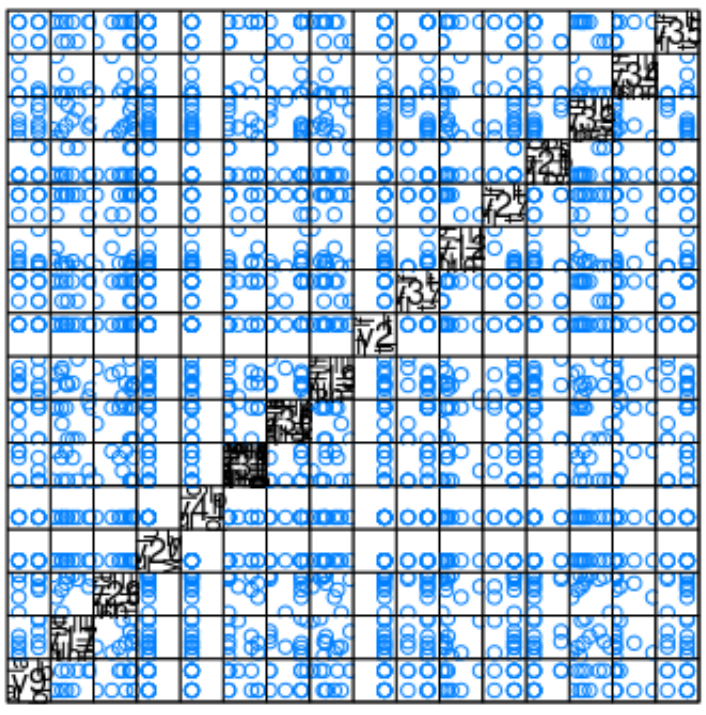
```
##
## Recursive feature selection
##
## Outer resampling method: Cross-Validated (10 fold)
##
## Resampling performance over subset size:
##
##  Variables Accuracy  Kappa AccuracySD KappaSD Selected
##          1   0.8713 0.7425    0.03893 0.07783
##          2   0.8713 0.7425    0.03893 0.07783
##          3   0.8694 0.7388    0.04131 0.08258
##          4   0.8804 0.7606    0.05103 0.10220
##          5   0.8784 0.7567    0.05550 0.11119
##          6   0.8981 0.7962    0.05830 0.11669
##          7   0.9090 0.8179    0.05092 0.10193
##          8   0.9090 0.8179    0.05757 0.11519
##          9   0.9146 0.8291    0.04183 0.08369
##         10   0.9110 0.8219    0.04286 0.08574
##         11   0.9146 0.8291    0.04254 0.08505
##         12   0.9073 0.8145    0.04877 0.09756
##         13   0.9091 0.8181    0.04668 0.09339
##         14   0.9091 0.8181    0.04829 0.09659
##         15   0.9146 0.8293    0.04133 0.08260        *
##         16   0.9110 0.8221    0.03900 0.07795
##
## The top 5 variables (out of 15):
##    v2, v31, v34, v19, v12
```

```
##  [1] "v2"  "v31" "v34" "v19" "v12" "v29" "v39" "v36" "v17" "v20" "v35"
## [12] "v37" "v41" "v27" "v21"
```

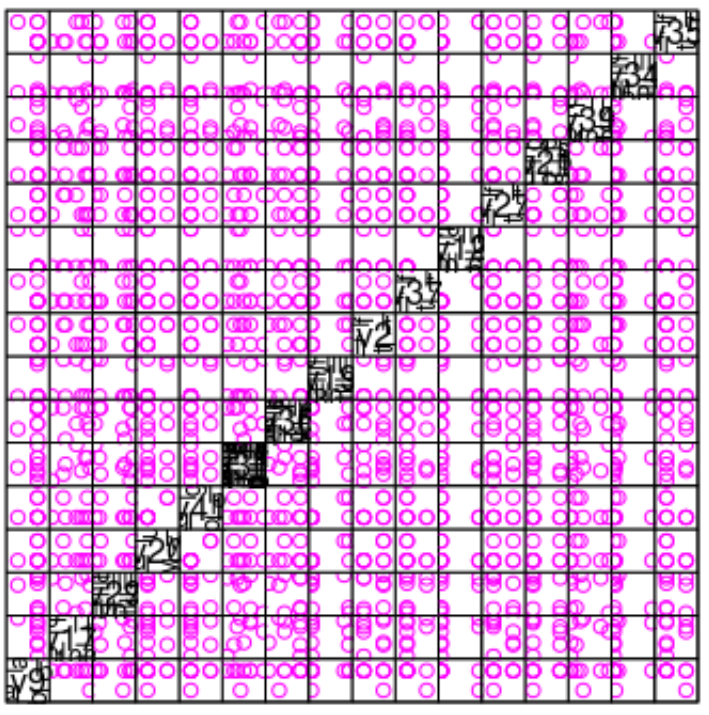
```
## rf variable importance
##
##   only 20 most important variables shown (out of 38)
##
##       Overall
## v2t    75.114
## v19    22.915
## v12    20.842
## v34    19.373
## v29    16.287
## v39    14.933
## v17    14.625
## v37t    9.209
## v35t    8.532
## v31x    3.300
## v27t    3.229
## v41p    3.093
## v20y    3.088
## v20u    3.031
## v31cc   2.959
## v36v    2.929
## v31ff   2.866
## v36ff   2.801
## v9b     2.753
## v31W    2.591
```



Above we found most important features: v2, v31, v34, v19, v12. These are most relevant for our predictor. We should keep it in mind. Here would be interesting to have a closer look at False Positives and False Negatives of our predictor. For instance, it can be seen that for features v2, v34, v21, v12 some regions are very narrowed. With a caution and care this observation can be exploited for creating extra features, which would differentiate better points corresponding to different classLabels.



Scatter Plot Matrix



Scatter Plot Matrix