Decision trees and ways on removing noisy labels
Identify costumers in unsound service models

Yannick Misteli

Julius Bär

yannick.misteli@juliusbaer.com

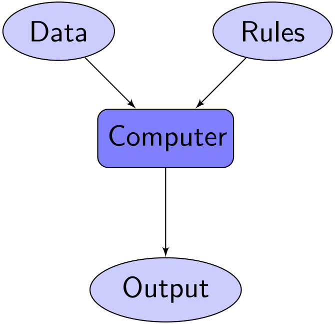
September 6, 2018

Overview

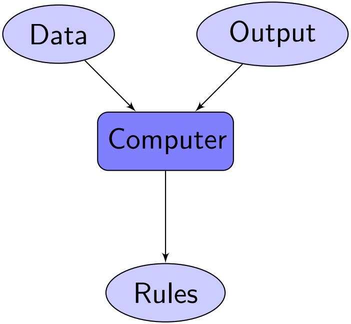
Interpretable Models and general aspects of ML

Introduction

Traditional Programming



Supervised Machine Learning



Interpretability[1]

Interpretability

Interpretability is the degree to which a human can understand the cause of a decision[2]

- The importance of interpretability (Regulator) or **what vs why and finding meaning in the world**
- Criteria for interpretability methods or **intrinsic vs post hoc**
- Human-friendly explanations or **what is a good explanation?**

[1] Molnar, Christoph. 2018., "Interpretable Machine Learning", Leanpub

[2] Miller, Tim. 2017. "Explanation in Artificial Intelligence: Insights from the Social Sciences." arXiv Preprint arXiv:1706.07269

Interpretable Models

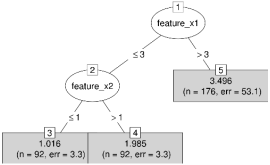
Models

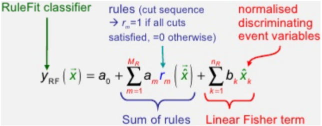
- Linear models
- Logistic regression
- Naive Bayes
- **Decision trees**
- RuleFit[3]
- k-Nearest Neighbours

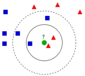
$$y_i = \beta_0 + \beta_1 x_{i,1} + \ldots + \beta_p x_{i,p} + \epsilon_i$$

$$P(y_i = 1) = \frac{1}{1 + exp(-(\beta_0 + \beta_1 x_{i,1} + \ldots + \beta_p x_{i,p}))}$$

$$P(C_k \mid x) = \frac{1}{Z} P(C_k) \prod_{i=1}^{n} P(x_i \mid C_k)$$







[3] Friedman-Popescu, Tech Rep, Stat. Dpt, Standford U., 2003

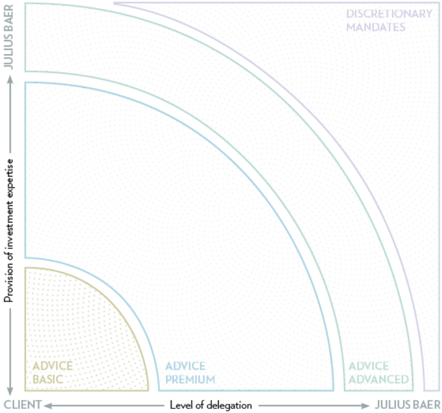
Use Case

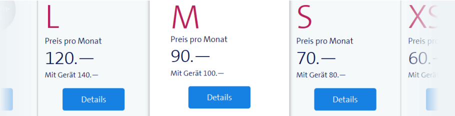
Service Models

Advisory Service Models

1. Basic
2. Premium
3. Advanced

Every advised client signs a service model agreement. Hence, according to preferences and service needs either a basic, premium or advanced service contract is put in place.





Use case

Problem

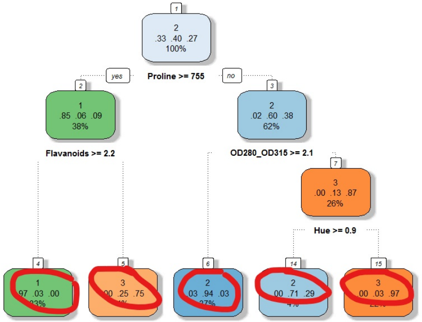
How to identify clients that should be in a different Service Model?

Idea

Fit decision tree and investigate terminal nodes for misclassified clients

Intention

Up- and downselling of misclassified clients



Generating multivariate tri-modal mixed distributions

Multivariate data with count and continuous variable with a pre-specified correlation matrix is generated. The count and continouous variable are assumed to have Poisson and normal marginals, respectively. The resulting mixture is

$$F(x) = \sum_{i=1}^{n} w_i P_i(x),$$

where $n = 3$; $w_1 = 0.6$, $w_2 = 0.3$, $w_3 = 0.1$ and P_i is the corresponding multivariate Poisson-Normal distribution.

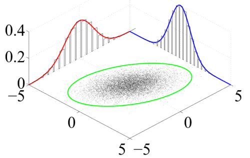


Figure: Example of sample points from a multivariate normal distribution with $\sigma = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ and $\Sigma = \begin{bmatrix} 1 & \frac{3}{5} \\ \frac{3}{5} & 2 \end{bmatrix}$, shown along with the 3-sigma ellipse.[4]

[4] en.wikipedia.org/wiki/Multivariate_normal_distribution

Data Summary

Synthetic Data

All data contained in these slides have been generated synthetically and not by Julius Bär. In no event shall the author or Julius Bär be liable for any direct, indirect, special or incidental damages resulting from, arising out of or in connection with, the use of the data contained herein.

```
head(data)
```

```
##   Service.Model Nr.Trades Nr.CCY Nr.Positions
## 1         basic         1      2            1
## 2         basic         2      4            4
## 3       premium         2      6           15
## 4       premium         9      6           18
## 5         basic         2      3            5
## 6       premium         0      3            7
##         AuM       Cash      Random
## 1 1198618.9  133102.70  0.46951209
## 2 1001045.8 -223904.30 -1.18726746
## 3 1629286.8  180392.06  0.54675242
## 4 3947500.7  234791.35  0.01287336
## 5  361907.7  -74267.38 -1.21565020
## 6 2760734.3  340989.71 -2.05284115
```

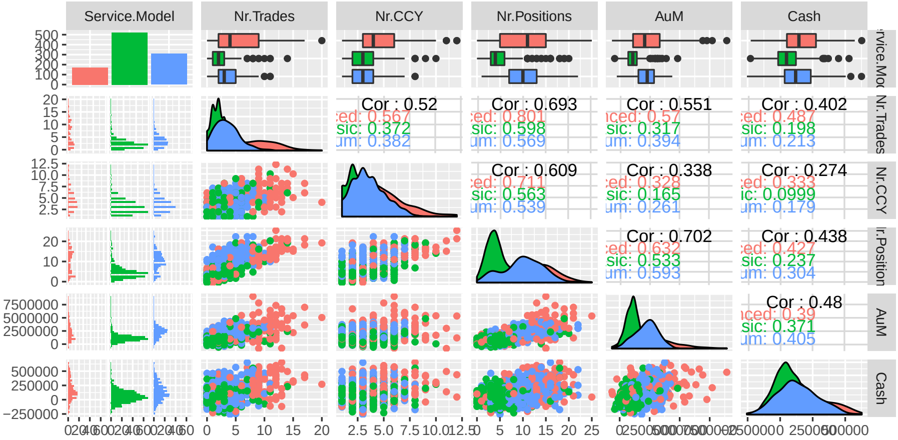
```
summary(data[,1:6])
```

```
##   Service.Model     Nr.Trades
##  advanced:17025   Min.   : 0.000
##  basic   :52136   1st Qu.: 1.000
##  premium :30839   Median : 3.000
##                   Mean   : 3.394
##                   3rd Qu.: 4.000
##                   Max.   :23.000
##      Nr.CCY         Nr.Positions
##  Min.   : 1.000   Min.   : 0.000
##  1st Qu.: 2.000   1st Qu.: 3.000
##  Median : 3.000   Median : 6.000
##  Mean   : 3.587   Mean   : 7.478
##  3rd Qu.: 5.000   3rd Qu.:11.000
##  Max.   :18.000   Max.   :30.000
##       AuM                Cash
##  Min.   :-4851052   Min.   :-464873
##  1st Qu.:  868542   1st Qu.:   3929
##  Median : 1385314   Median :  85995
##  Mean   : 1701572   Mean   : 105146
##  3rd Qu.: 2346875   3rd Qu.: 183797
##  Max.   :12001587   Max.   :1066861
```

Exploratory Data Analysis

Synthetic Data

All data contained in these slides have been generated synthetically and not by Julius Bär. In no event shall the author or Julius Bär be liable for any direct, indirect, special or incidental damages resulting from, arising out of or in connection with, the use of the data contained herein.



Label Noise

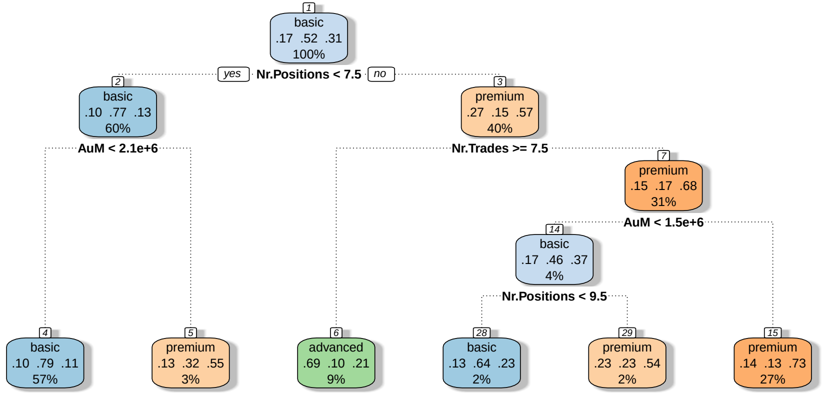
Synthetic Data

All data contained in these slides have been generated synthetically and not by Julius Bär. In no event shall the author or Julius Bär be liable for any direct, indirect, special or incidental damages resulting from, arising out of or in connection with, the use of the data contained herein.

```
noisy_data <- clean_data
leng <- nrow(noisy_data)
labelnoise <- 20
resample <- sample.int(leng, leng/100*labelnoise)
mylabels <- unique(clean_data$Service.Model)
for(k in resample){
  myset <- noisy_data[k,]
  noisy_data[k,1] <- sample(mylabels[(myset$Service.Model != mylabels)],1)
}
```

Decision Tree

Synthetic Data



Rattle 2018−Sep−06 09:28:31 yannick

After Math

Noisy lables - sources and effects[5]

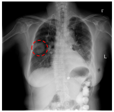
Sources of noise

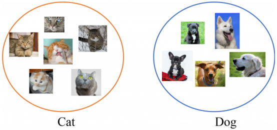
- insufficient information provided to the expert
- errors in the expert labelling itself
- subjectivity of the labelling task
- communication/enconding problems

Effects of noise

- decrease the classification performances
- increase/decrease the complexity of learned models
- pose a threat to tasks like e.g. feature selection

X	Y	Z	Label
0.01	-0.01	-0.02	Resting
4.04	-10.2	7.66	Running
1.23	1.73	0.02	Walking
0.03	-0.07	0.09	Resting
15.72	-25.76	12.23	Running
1.45	0.33	0.43	Walking





[5] https://labelnoise2017.loria.fr/wp-content/uploads/2017/11/présentation-LABELNOISE17-Frénay.pdf

Dealing with Noise

Dealing with Noise

- overfitting avoidance and robust losses
- data cleansing
- noise-tolerant algorithms

Package 'NoiseFiltersR'

August 29, 2016

Type Package

Title Label Noise Filters for Data Preprocessing in Classification

Version 0.1.0

Description An extensive implementation of state-of-the-art and classical algorithms to preprocess label noise in classification problems.

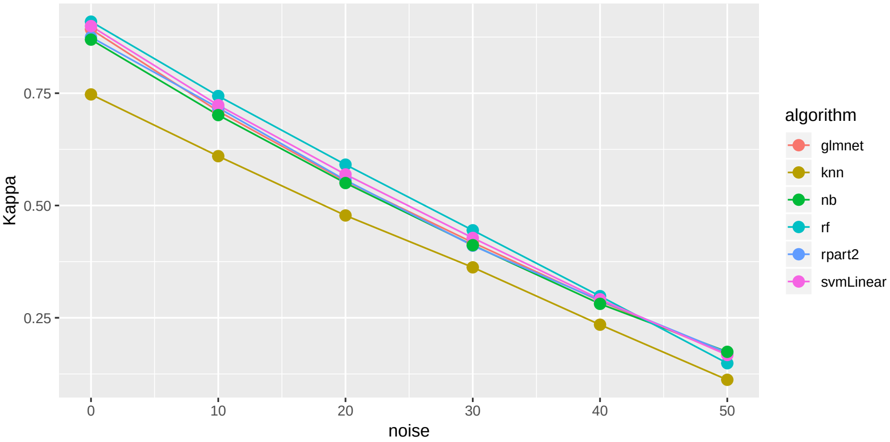
License GPL-3

LazyData TRUE

Imports RWeka, kknn, nnet, caret, e1071, rpart, randomForest, MASS, rJava, stats, utils

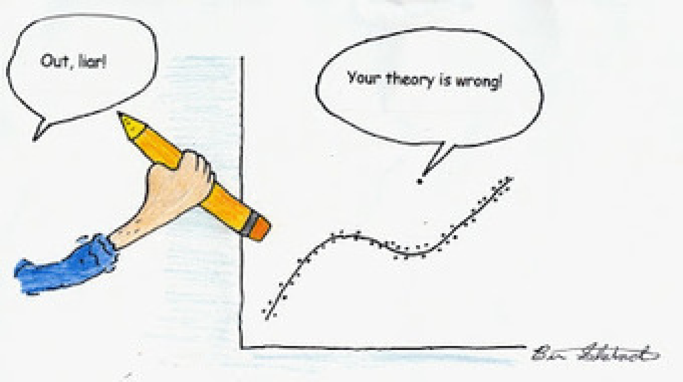
```r
library(NoiseFiltersR)
out <- C45robustFilter(Service.Model ~.,data = data)

## Iteration 1:  21897 instances removed
## Iteration 2:  572 instances removed
## Iteration 3:  192 instances removed
## Iteration 4:  36 instances removed
## Iteration 5:  26 instances removed
## Iteration 6:  20 instances removed
## Iteration 7:  7 instances removed
## Iteration 8:  1 instances removed
## Iteration 9:  0 instances removed
## Summary:  22751 instances removed in 9 iterations

cldata <- out$cleanData
print(out)

## Call:
## C45robustFilter(formula = Service.Model ~
##     ., data = data)
##
## Results:
## Number of removed instances: 22751 (22.751 %)
## Number of repaired instances: 0 (0 %)
```

Noise Sensitivity of ML algorithms (preliminary results)



Thanks

Thank you for your attention!



Noise Sensitivity of ML algorithms (preliminary results)

noise	algorithm	Accuracy	Kappa	Time	Description
0	glmnet	0.9424	0.8924	285.754	glmnet
0	rpart2	0.9326	0.8749	6.865	CART
0	rf	0.9513	0.9094	956.818	Random Forest
0	knn	0.8680	0.7473	130.746	k-Nearest Neighbors
0	svmLinear	0.9461	0.8995	90.185	Support Vector Machines with Linear Kernel
0	nb	0.9306	0.8693	47.250	Naive Bayes
10	glmnet	0.8407	0.7099	220.618	glmnet
10	rpart2	0.8423	0.7174	7.293	CART
10	rf	0.8574	0.7438	1453.731	Random Forest
10	knn	0.7880	0.6098	130.298	k-Nearest Neighbors
10	svmLinear	0.8473	0.7233	546.760	Support Vector Machines with Linear Kernel
10	nb	0.8340	0.7013	44.617	Naive Bayes
20	glmnet	0.7466	0.5544	192.642	glmnet
20	rpart2	0.7427	0.5565	7.708	CART
20	rf	0.7645	0.5910	1595.194	Random Forest
20	knn	0.7048	0.4778	130.875	k-Nearest Neighbors
20	svmLinear	0.7545	0.5692	926.669	Support Vector Machines with Linear Kernel
20	nb	0.7439	0.5500	46.129	Naive Bayes
30	glmnet	0.6551	0.4176	170.448	glmnet
30	rpart2	0.6468	0.4105	7.892	CART
30	rf	0.6676	0.4448	1620.355	Random Forest
30	knn	0.6231	0.3623	132.471	k-Nearest Neighbors
30	svmLinear	0.6612	0.4279	1181.009	Support Vector Machines with Linear Kernel
30	nb	0.6520	0.4117	45.797	Naive Bayes
40	glmnet	0.5655	0.2871	148.020	glmnet
40	rpart2	0.5620	0.2888	8.023	CART
40	rf	0.5669	0.2982	1686.300	Random Forest
40	knn	0.5307	0.2346	131.614	k-Nearest Neighbors
40	svmLinear	0.5681	0.2913	1385.847	Support Vector Machines with Linear Kernel
40	nb	0.5621	0.2810	45.725	Naive Bayes
50	glmnet	0.4767	0.1688	122.134	glmnet
50	rpart2	0.4753	0.1725	8.447	CART
50	rf	0.4561	0.1488	1739.850	Random Forest
50	knn	0.4314	0.1119	132.068	k-Nearest Neighbors
50	svmLinear	0.4765	0.1671	1461.812	Support Vector Machines with Linear Kernel
50	nb	0.4784	0.1742	46.320	Naive Bayes