Westfälische
Wilhelms-Universität
Münster

Kaggle House

Applied Machine Learning

living.knowledge
WWU Münster

Frederik Elischberger,
Marco Niemann

January 16th, 2017

Agenda

1. Introduction and Background
2. Exploratory Analysis
3. Preprocessing
4. Supervised learning
5. Implementation
6. Results
7. Bibliography

Introduction and Background

Imagine ...

Imagine ...

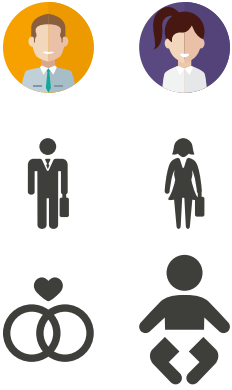



Imagine ...



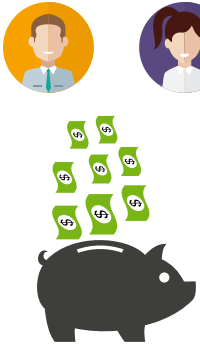


Imagine ...



Imagine ...

Imagine ...

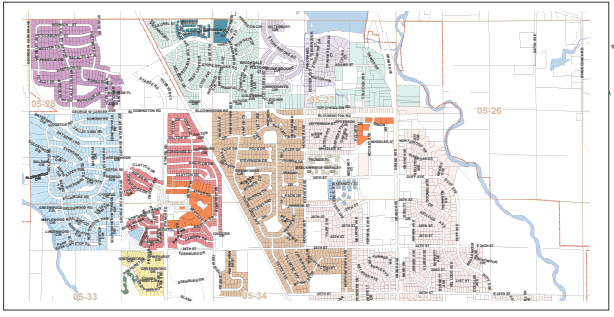


Imagine ...

What Might Determine the Price?

Zoning, Lot Area,
Lot Shape, Lot Frontage

What Might Determine the Price?



Zoning, Lot Area,
Lot Shape, Lot Frontage

Neighbourhood

What Might Determine the Price?

Zoning, Lot Area,
Lot Shape, Lot Frontage

Neighbourhood

House Age

What Might Determine the Price?

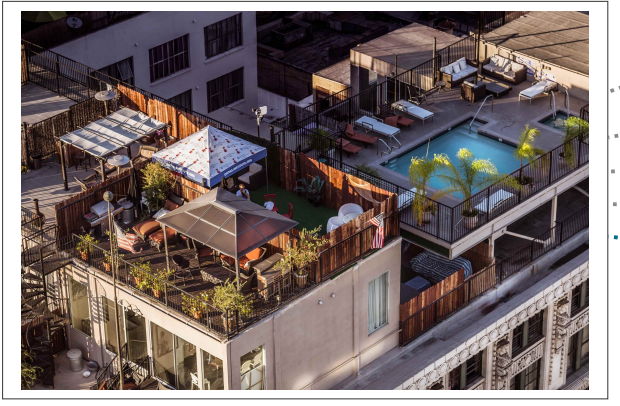
Zoning, Lot Area,
Lot Shape, Lot Frontage

Neighbourhood

House Age

Masonry Type

What Might Determine the Price?



Zoning, Lot Area,
Lot Shape, Lot Frontage

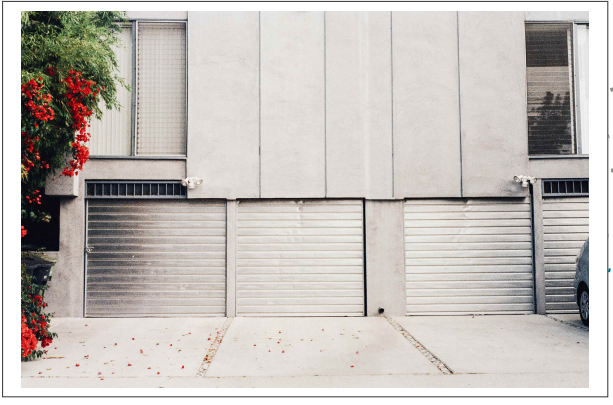
Neighbourhood

House Age

Masonry Type

Pool (Area/Quality)

What Might Determine the Price?



Zoning, Lot Area,
Lot Shape, Lot Frontage

Neighbourhood

House Age

Masonry Type

Pool (Area/Quality)

Garage (Cars, Area, ...)

What Might Determine the Price?

Zoning, Lot Area,
Lot Shape, Lot Frontage

Neighbourhood

House Age

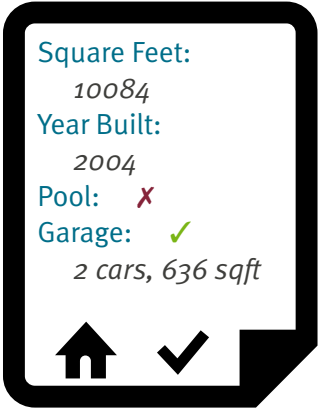
Masonry Type

Pool (Area/Quality)

Garage (Cars, Area, ...)

Fireplace (Count, Quality)

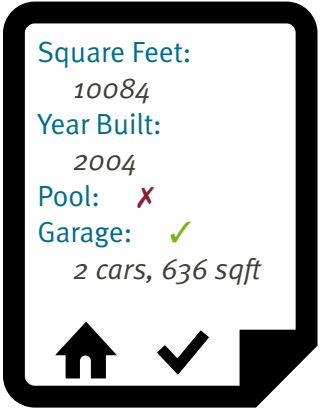
What would you pay?



Question

What would be the price you would pay for such a house?

What would you pay?



Question

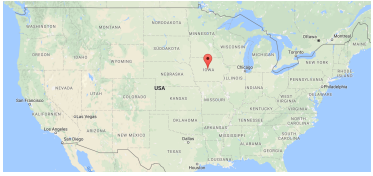
What would be the price you would pay for such a house?

Answer

The house was sold in 2007 for 307000$.

How does that relate to this Seminar?

- tasks like estimating house prices are classical regression tasks
  - for example the rather famous *Bouston Housing Data Set*
  - often used for educational purpose
- De Cock (2011) was convinced a new educational dataset was necessary
  - searched for a new, real-world dataset he could use for his statistics courses
  - by serendipitous discovery got access to a dataset of the city of Ames in Iowa



(Harrison and Rubinfeld 1978)

How does that relate to this Seminar?

- De Cock (2011) data set became the foundation for the Kaggle Challenge

  House Prices: Advanced Regression Techniques

- the challenge asks for multiple contributions:
  - sale price prediction with advanced regression methods (*Machine Learning*)
  - data exploration including insightful visualizations
  - feature engineering

Info - Kaggle

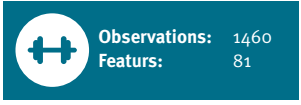
Kaggle is an online platform for predictive modelling and analytics competitions. Companies/Unis host competitions there to be solved by freelance data miners.

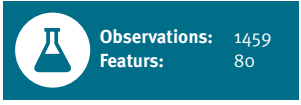
(Athanasopoulos and Hyndman 2011; Kaggle Inc 2016a; Kaggle Inc 2016b; Metz 2013; Taieb and Hyndman 2013; Wikipedia 2016)


Exploratory Analysis

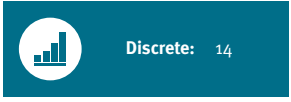
Summary

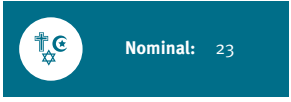
- the `train` and `test` dataset have the following characteristics:

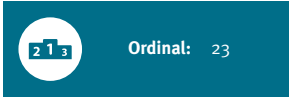




- among the features the following scales were discovered:





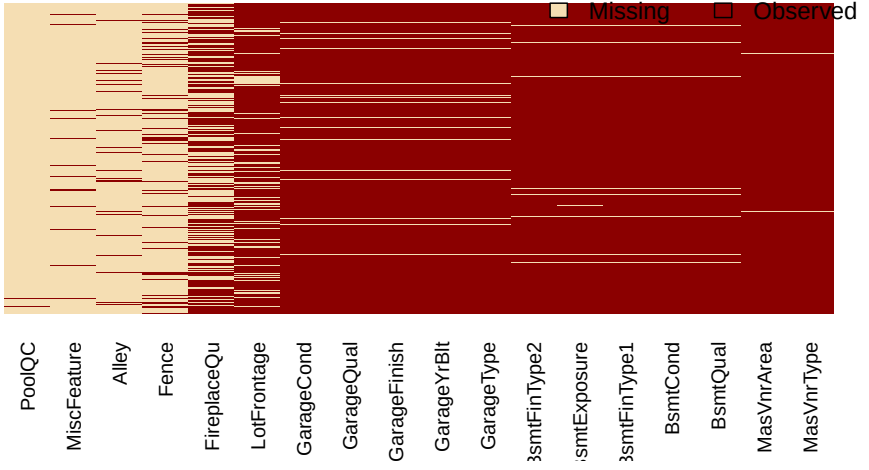


Summary

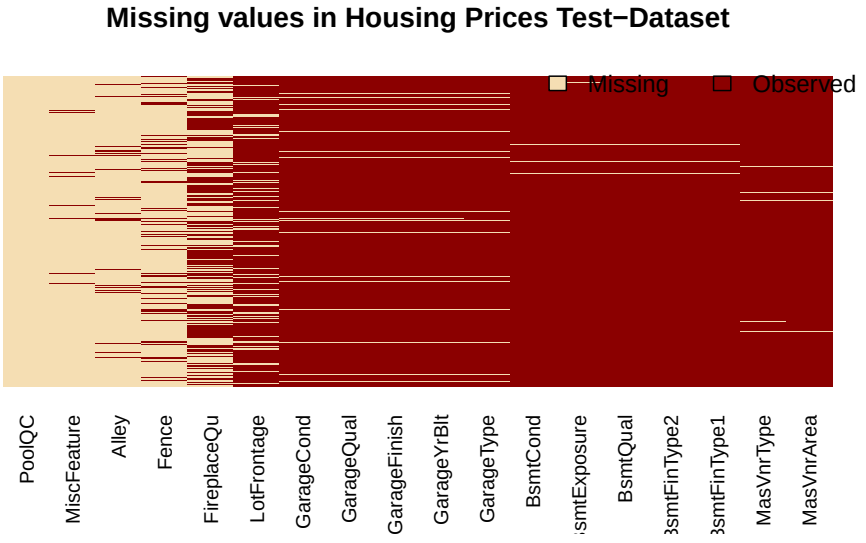
- going through the data reveals very different distributions and value ranges:

```
##   OverallQual      PoolQC       SalePrice
##  Min.   : 1.000   Ex  :   2   Min.   : 34900
##  1st Qu.: 5.000   Fa  :   2   1st Qu.:129975
##  Median : 6.000   Gd  :   3   Median :163000
##  Mean   : 6.099   NA's:1453   Mean   :180921
##  3rd Qu.: 7.000               3rd Qu.:214000
##  Max.   :10.000               Max.   :755000
```

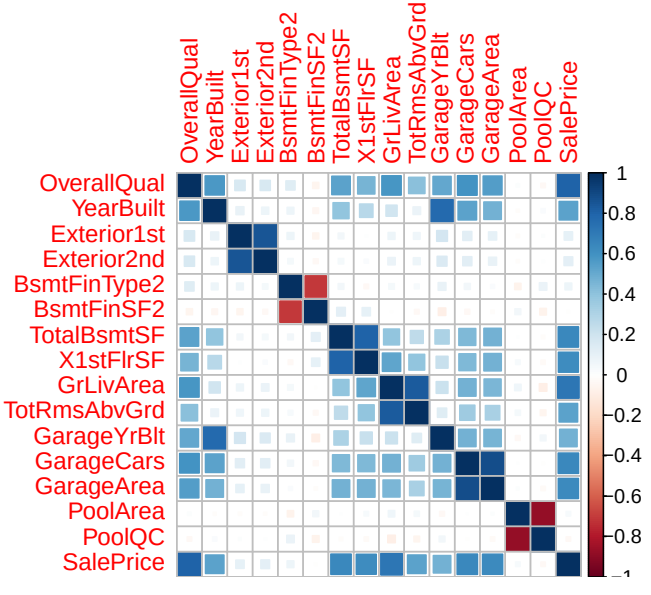
Missing values - Train



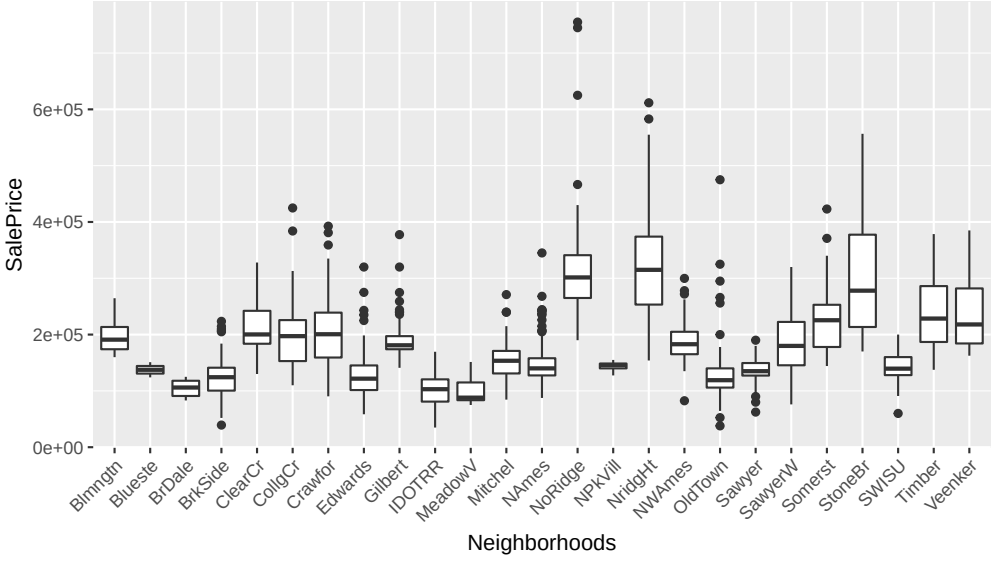
Missing values - Test



Correlation



Boxplot of Location to Saleprice




Preprocessing

Imputation

- many features contain `NA` values although there is a semantical meaning
  - substitute missing values in specific features with `'None'`
  - dataset documentation describes which features
  - transformation into numerical factors that do not contain NA values
- for some features a manual substitution was possible:
  $\Rightarrow$ `LotFrontage`, `Electrical` and `GarageYrBlt`
- remaining features are imputed using `mice`

(Buuren and Groothuis-Oudshoorn 2011)

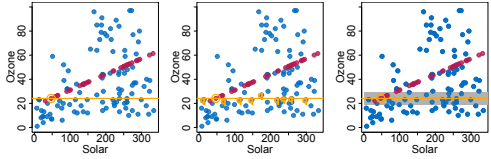
Imputation - Predictive Mean Matching and `mice`

1. generation of linear linear regression model
2. predict value for missing observations
3. include model and prediction uncertainty to create prediction candidates
4. sample one of the closest candidates to the regression model

(Vink et al. 2014; Buuren and Groothuis-Oudshoorn 2011; Horton and Lipsitz 2001)

Imputation - Predictive Mean Matching and `mice`

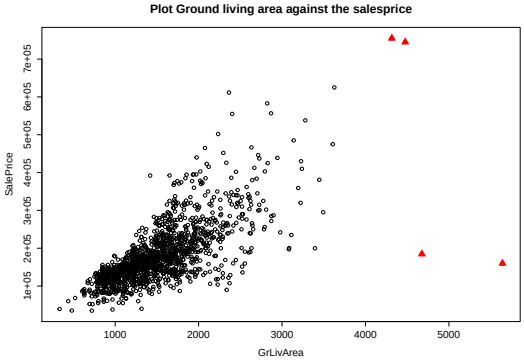
1. generation of linear linear regression model
2. predict value for missing observations
3. include model and prediction uncertainty to create prediction candidates
4. sample one of the closest candidates to the regression model
5. the `mice` package additionally conducts the following step:
   take the observations with the smallest distance to the predicted value



(Vink et al. 2014; Buuren and Groothuis-Oudshoorn 2011; Horton and Lipsitz 2001)

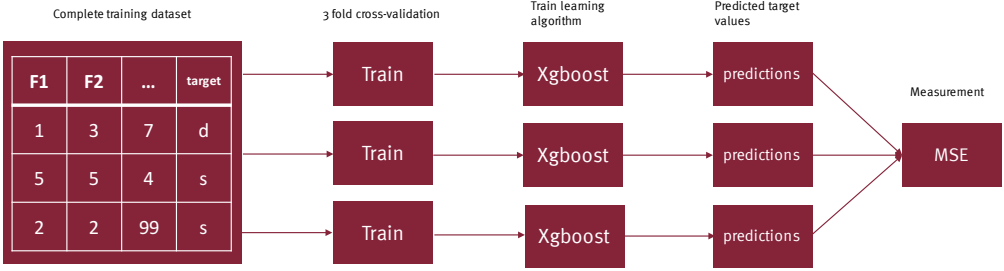
Outlier

- no reasonable results when applying Local-outlier-factor
- author suggests to remove rows with ground living area larger than 4000
- removed column `Utilities`



Supervised Learning

What is supervised learning

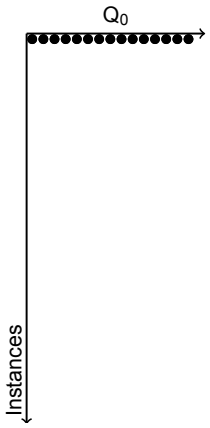


irace

- given a set of Instances $\mathcal{I}$
- given a set of Candidates Θ
- given a cost function c and an instance $i \in \mathcal{I}$ to calculate $c(\theta, i) \in \mathcal{R}$

(López-Ibáñez et al. 2016)

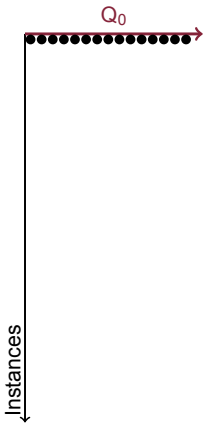
irace



- take the initial set of candidates θ
- and one problem instance *i*
- to calculate the set of costs $c(\Theta, i)$ for *i*
- to discard candidates that perform statistically worse item Iterate over all instances until one candidate solution **survives**

(López-Ibáñez 2012)

irace



- take the initial set of candidates θ
- and one problem instance *i*
- to calculate the set of costs $c(\Theta, i)$ for *i*
- to discard candidates that perform statistically worse item Iterate over all instances until one candidate solution **survives**

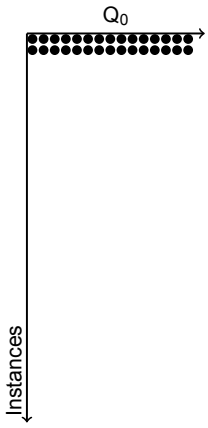
(López-Ibáñez 2012)

irace

- take the initial set of candidates θ
- and one problem instance *i*
- to calculate the set of costs $c(\Theta, i)$ for *i*
- to discard candidates that perform statistically worse item Iterate over all instances until one candidate solution **survives**

(López-Ibáñez 2012)

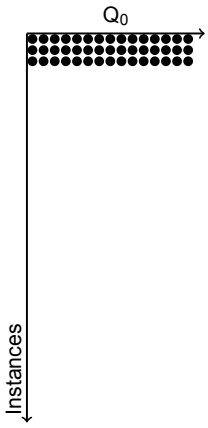
irace



- take the initial set of candidates θ
- and one problem instance *i*
- to calculate the set of costs $c(\Theta, i)$ for *i*
- to discard candidates that perform statistically worse item Iterate over all instances until one candidate solution **survives**

(López-Ibáñez 2012)

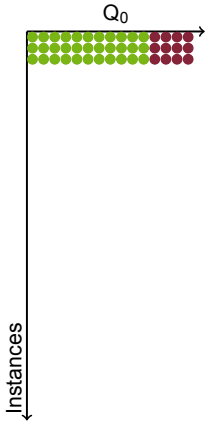
irace



- take the initial set of candidates θ
- and one problem instance *i*
- to calculate the set of costs $c(\Theta, i)$ for *i*
- to discard candidates that perform statistically worse item Iterate over all instances until one candidate solution **survives**

(López-Ibáñez 2012)

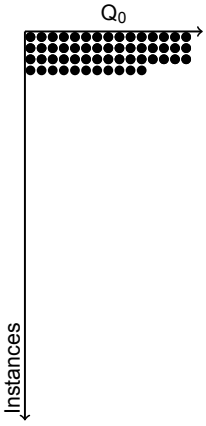
irace



- take the initial set of candidates θ
- and one problem instance *i*
- to calculate the set of costs $c(\Theta, i)$ for *i*
- to discard candidates that perform statistically worse item Iterate over all instances until one candidate solution **survives**

(López-Ibáñez 2012)

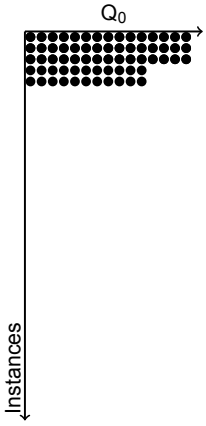
irace



- take the initial set of candidates θ
- and one problem instance *i*
- to calculate the set of costs $c(\Theta, i)$ for *i*
- to discard candidates that perform statistically worse item Iterate over all instances until one candidate solution **survives**

(López-Ibáñez 2012)

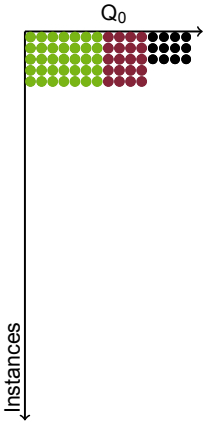
irace



- take the initial set of candidates θ
- and one problem instance *i*
- to calculate the set of costs $c(\Theta, i)$ for *i*
- to discard candidates that perform statistically worse item Iterate over all instances until one candidate solution **survives**

(López-Ibáñez 2012)

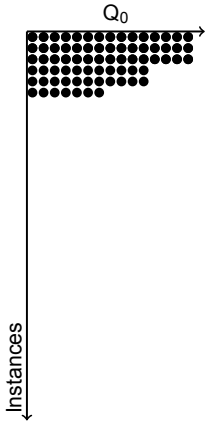
irace



- take the initial set of candidates θ
- and one problem instance *i*
- to calculate the set of costs $c(\Theta, i)$ for *i*
- to discard candidates that perform statistically worse item Iterate over all instances until one candidate solution **survives**

(López-Ibáñez 2012)

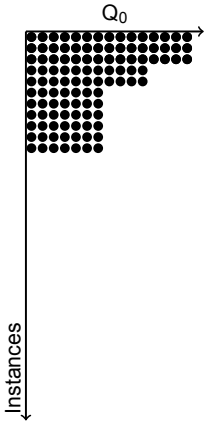
irace



- take the initial set of candidates θ
- and one problem instance *i*
- to calculate the set of costs $c(\Theta, i)$ for *i*
- to discard candidates that perform statistically worse item Iterate over all instances until one candidate solution **survives**

(López-Ibáñez 2012)

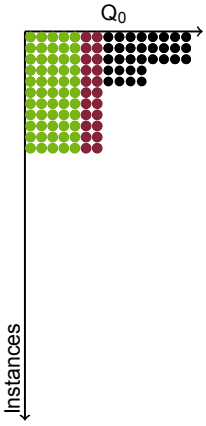
irace



- take the initial set of candidates θ
- and one problem instance i
- to calculate the set of costs $c(\Theta, i)$ for i
- to discard candidates that perform statistically worse item Iterate over all instances until one candidate solution **survives**

(López-Ibáñez 2012)

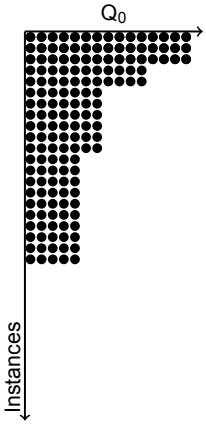
irace



- take the initial set of candidates θ
- and one problem instance i
- to calculate the set of costs $c(\Theta, i)$ for i
- to discard candidates that perform statistically worse item Iterate over all instances until one candidate solution **survives**

(López-Ibáñez 2012)

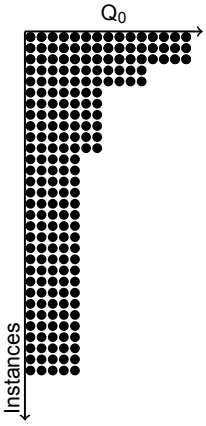
irace



- take the initial set of candidates θ
- and one problem instance i
- to calculate the set of costs $c(\Theta, i)$ for i
- to discard candidates that perform statistically worse item Iterate over all instances until one candidate solution **survives**

(López-Ibáñez 2012)

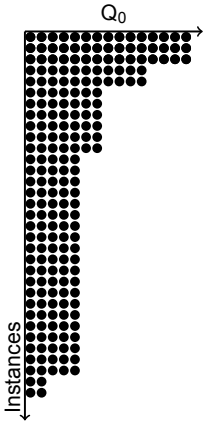
irace



- take the initial set of candidates θ
- and one problem instance *i*
- to calculate the set of costs $c(\Theta, i)$ for *i*
- to discard candidates that perform statistically worse item Iterate over all instances until one candidate solution **survives**

(López-Ibáñez 2012)

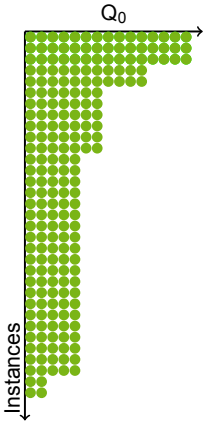
irace



- take the initial set of candidates θ
- and one problem instance i
- to calculate the set of costs $c(\Theta, i)$ for i
- to discard candidates that perform statistically worse item Iterate over all instances until one candidate solution **survives**

(López-Ibáñez 2012)

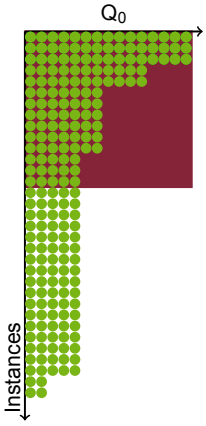
irace



- take the initial set of candidates θ
- and one problem instance *i*
- to calculate the set of costs $c(\Theta, i)$ for *i*
- to discard candidates that perform statistically worse item Iterate over all instances until one candidate solution **survives**

(López-Ibáñez 2012)

irace



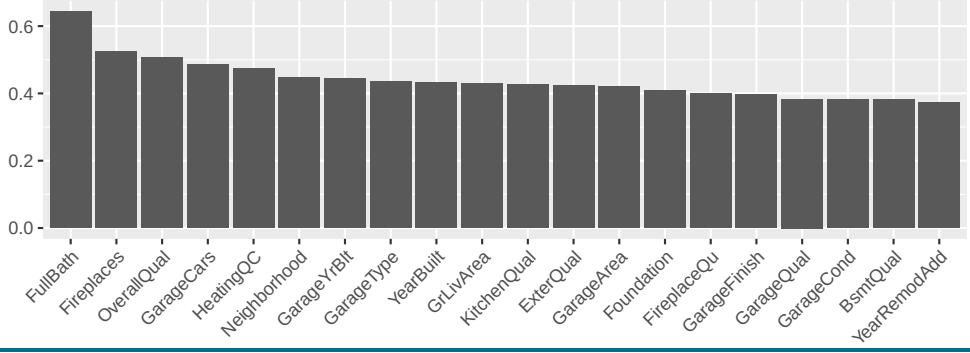
- take the initial set of candidates θ
- and one problem instance i
- to calculate the set of costs $c(\Theta, i)$ for i
- to discard candidates that perform statistically worse item Iterate over all instances until one candidate solution **survives**

(López-Ibáñez 2012)

Tuning of feature selection

- `mlr` provides a method combining a learner with a feature selection method
- both can be tuned at the same time

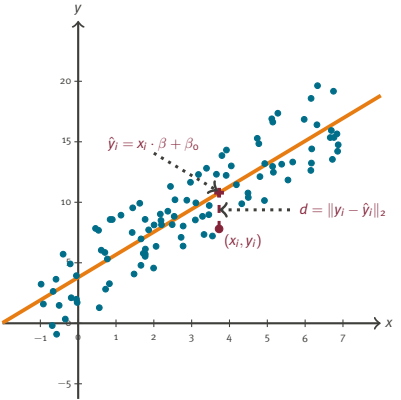
kaggleHouse (78 features), filter = chi.squared



Glmnet

$$\min_{(\beta_0,\beta)\in\mathcal{R}^{p+1}} \frac{1}{2N}\sum_{i=1}^{N}(y_i-\beta_0-\mathbf{x_i}^T\beta)^2+\lambda\left[\frac{(1-\alpha)\|\beta\|_2^2}{2}+\alpha\|\beta\|_1\right]$$

- Gaussian as default loss function
- halved quadratic deviation of predictions and observations
- minimize by adapting the weightings β_0 and β

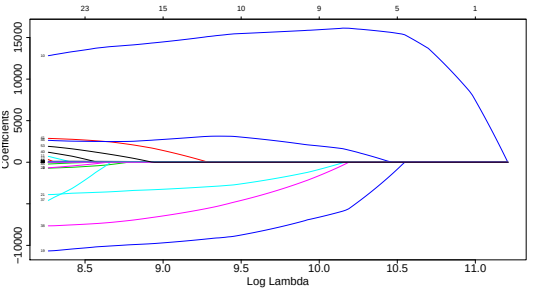


(Friedman et al. 2010; Hastie and Qian 2014; Tibshirani 1996)

Glmnet

$$\min_{(\beta_0,\beta)\in\mathcal{R}^{p+1}} \frac{1}{2N}\sum_{i=1}^{N}(y_i-\beta_0-\mathbf{x_i}^T\beta)^2+\lambda\left[\frac{(1-\alpha)\|\beta\|_2^2}{2}+\alpha\|\beta\|_1\right]$$

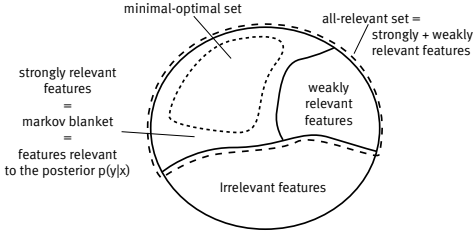
- penalty term where λ is a complexity parameter
- α as compromise between $\mathcal{L}^1$-norm(lasso) and $\mathcal{L}^2$-norm(ridge) of β
- use coordinate descent step for each β_j



(Friedman et al. 2010; Hastie and Qian 2014; Tibshirani 1996)

Boruta Feature Selection

- relatively new `randomForest`-based, all-relevant feature selection method
  $\rightarrow$ *most other algorithms work with the minimal-optimal algorithm*
- not only selects technically significant data but all 'interesting' data points
- BUT: computationally hard-problem



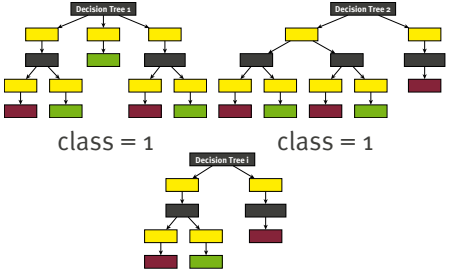
(Homola 2015; Kursa et al. 2010; Kursa and Rudnicki 2010; Nilsson et al. 2007; Rudnicki et al. 2015)

Boruta Feature Selection - Boruta Method

- the Boruta algorithm works as a wrapper around the `randomForest` package
- allows to eliminate influence of random fluctuations and correlations

$$S = \begin{bmatrix} f_{A1} & f_{B1} & f_{C1} & C_1 \\ \vdots & & \vdots & \\ f_{AN} & f_{BN} & f_{CN} & C_N \end{bmatrix} \quad S_1 = \begin{bmatrix} f_{A12} & f_{B12} & f_{C12} & C_{12} \\ f_{A15} & f_{B15} & f_{C15} & C_{15} \\ \vdots & & \vdots & \\ f_{A35} & f_{B35} & f_{C35} & C_{35} \end{bmatrix}$$

$$S_2 = \begin{bmatrix} f_{A2} & f_{B2} & f_{C2} & C_2 \\ f_{A6} & f_{B6} & f_{C6} & C_6 \\ \vdots & & \vdots & \\ f_{A20} & f_{B20} & f_{C20} & C_{20} \end{bmatrix} \quad S_3 = \begin{bmatrix} f_{A4} & f_{B4} & f_{C4} & C_4 \\ f_{A9} & f_{B9} & f_{C9} & C_9 \\ \vdots & & \vdots & \\ f_{A12} & f_{B12} & f_{C12} & C_{12} \end{bmatrix}$$



(Kursa et al. 2010; Kursa and Rudnicki 2010)

Boruta Feature Selection - Boruta Method

- as a first importance measure the Boruta algorithm makes use of the *z*-score
- additionally so called *shadow features* are introduced as an external reference

F_1	F_2	F_3	F_4
1	1	2	3
3	0	2	1
2	1	3	2

(a) Original Data

S_1	S_2	S_3	S_4
3	1	2	1
1	1	3	2
2	0	2	3

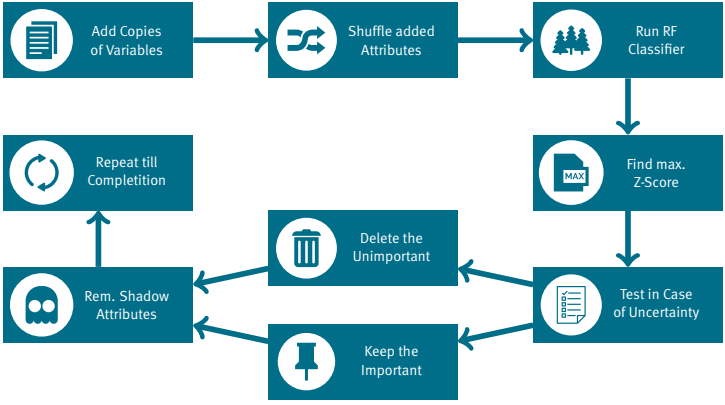
(b) Shadow Features

	F_1	F_2	F_3	F_4	S_1	S_2	S_3	S_4
MDI	.2	.05	.01	.15	.001	.02	.009	.09
Hit	+1	0	0	+1	-	-	-	max

(c) Selection of Relevant Features

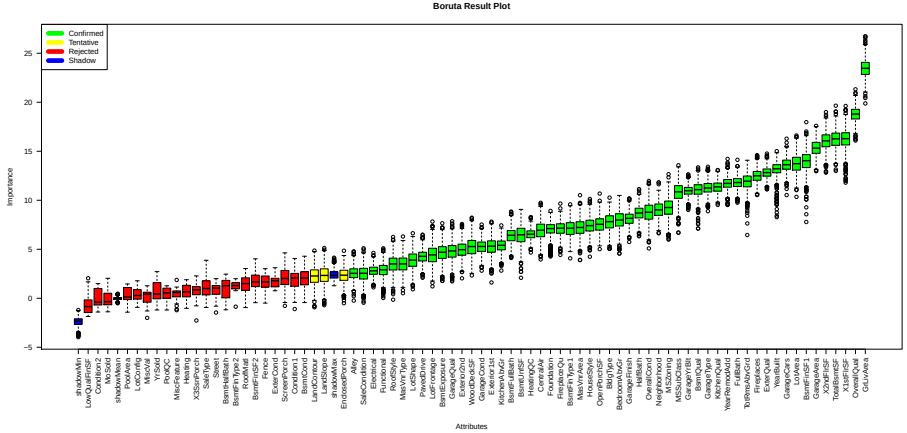
(Homola 2015; Kursa et al. 2010; Kursa and Rudnicki 2010)

Boruta Feature Selection - Boruta Method



(Kursa et al. 2010; Kursa and Rudnicki 2010)

Boruta Feature Selection - Results



Basic learner - Xgboost(Extreme Gradient Boosting)

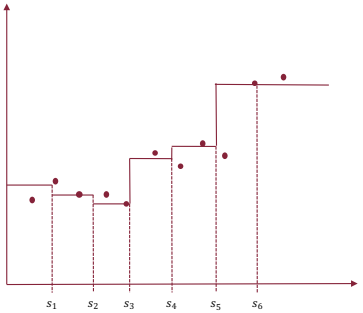
$$Obj(\theta) = \sum_{i}^{N} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k)$$

- Xgboost is a tree ensemble method
- Objective combines training loss and regularization
- Additive training

(Chen and Guestrin 2016; Chen et al. 2016; DMLC 2016)

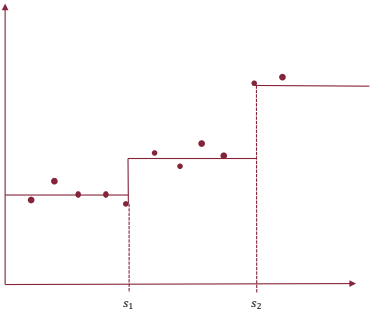
Xgboost - Ensemble method

Overfitting



$L(f)$ low but high $\Omega(f)$

Good balance



$L(f)$ and $\Omega(f)$ low

(Chen and Guestrin 2016; Chen et al. 2016; DMLC 2016)

Xgboost - Ensemble method

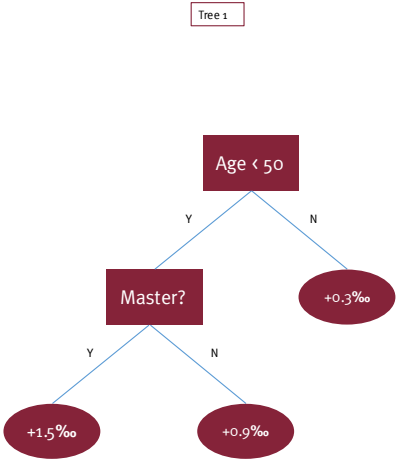
$$Obj(\theta) = \sum_{i}^{N} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k)$$

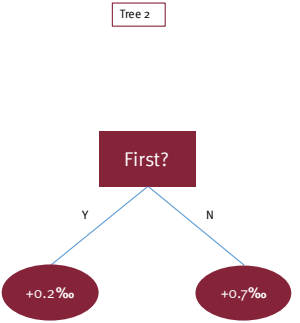
$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i), f_k \in \mathcal{F}$$

- How do we compute all K f_k that minimize the objective function?

(Chen and Guestrin 2016; Chen et al. 2016; DMLC 2016)

Xgboost - Ensemble method





$f($ 394261 $) = 0.9‰ + 0.2‰ = 1.1‰$

(Chen and Guestrin 2016; Chen et al. 2016; DMLC 2016)

Xgboost - Additive training

- Cannot globally find the best K trees that lead to the lowest objective
- Stepwise optimization by finding the best tree in step t

$$\hat{y}_i^{(t)} = \sum_{k=1}^{t} f_k(x_i) = \hat{y}_i^{(t-1)} + f_t(x_i)$$

- Testing all possible tree configuration is infeasible
- Build each tree recursively by calculating the best split point
- Score a split with a gain function
- Pruning by using the parameter γ defined in the regularization term

(Chen and Guestrin 2016; Chen et al. 2016; DMLC 2016)

Deep Learning

- receive lots of attention in recent years by outperforming traditional learners
- old concept (1960s) benefitting from improvements in computing power
- successesful competition in image recognition and games against humans
  - a deep learning network outperforming humans in image recognition
  - *Alpha Go* beating two Go champions 5:0 respectively 4:1

The Schloss dreamed by a Deep Learner

(Schmidhuber 2015; Silver et al. 2016; Ciresan et al. 2011; Ciresan et al. 2012)

Deep Learning - Artificial Neural Networks

- neural networks try to model the human brain
- performance via massive number of interconnected simple classifiers
- learning happens via weight adjustment
- ANNs are capable of modifying their own topology

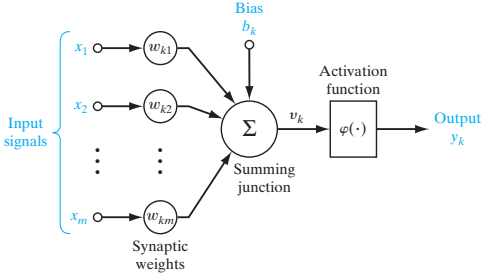


Figure: Schema of an ANN Neuron

(Haykin 2008)

Deep Learning - H2O Framework

- H2O is an open-source machine-learning library
- one of their features is the provision of deep learning algorithms
- widely used and active package

- provision of a fully scriptable `R` API via the `h2o` CRAN package
- BUT: rather targeted at computer clusters (*Hadoop*, *Spark*, ...)

(Candel et al. 2015)

Deep Learning - Encountered Issues

Initial Hope

Creation of a sufficiently deep neural net was expected to provide at least comparable results to other learners.

However, hopes were shattered fast:

- nearly infeasible tuning due to 86 different parameters
- insufficient computational resources (*library warnings*)
- generated models often of dubious quality (*single predicted value*)

Deep Learning - Attempts to Mitigate Issues

- manual tuning of well-understandable parameters like `hidden`
- using `h2o`s built-in grid search to approximate good parameter configurations

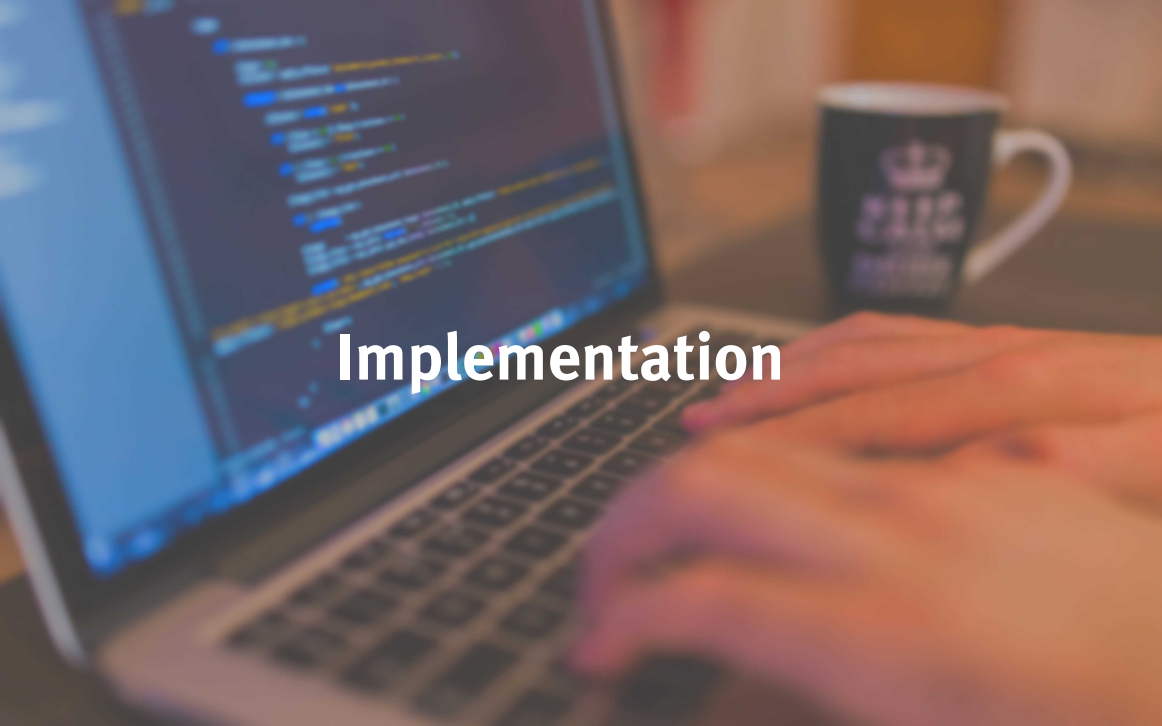
```
hyper_params <- list(hidden = list(c(2,2), c(4,4),...,
  c(1024,1024,1024)), input_dropout_ratio = c(0, ..., 0.2),
  rate = c(0.01, 0.02), rate_annealing = c(1e-8, ..., 1e-5),
  epochs = c(seq(from = 10, to = 90, by = 10), ...,
    seq(from = 10000, to = 90000, by = 10000), 100000),
  activation = c("Tanh", "Rectifier", "Maxout"))
```

Deep Learning - Results

- grid search resulted in non-deep networks as the ideal configuration
  - typically either a 32–32 or 64–64 network would provide best results
- even proposed networks performed rather bad on Kaggle

Submission	Files	Score	Selected?
Mon, 26 Dec 2016 00:49:32 Edit description	deeplearning.csv	0.29127	☐
Mon, 26 Dec 2016 00:22:09 Edit description	deeplearning.csv	0.32431	☐
Sun, 25 Dec 2016 23:36:28 Edit description	deeplearning.csv	0.35232	☐
Sun, 25 Dec 2016 21:13:30 Edit description	deeplearning.csv	0.40929	☐

Avg. Learner Score
typ. $\sim$ 0.13



Implementation

Implementation - The KaggleHouse Package

- R package that contains all relevant code and datasets
- includes documentation for all functions and approaches
- freely available via GitHub:

```
devtools::install_github("MarcoNiemann/kaggle_house")
```

Implementation - But why a Package?

And not a simple script file?

- easy provision (and use) of the competition data sets
- creation of reusable code that could be easily reapplied to different scenarios
- easy splitting of code into different logical components and files
- lots of helper tools to compile and combine code and documentation creation
- packages can be easily shared with others (*incl. dependencies etc.*)

(Wickham 2015)

Implementation - Performance Optimization

- use of Rcpp to improve execution speed of some customly written functions:

Benchmark of Dotplot Functions (Time in ms)

	min	lq	mean	median	uq	max
C++/Rcpp	3.38	3.49	3.68	3.56	3.69	9.11
R	8.06	8.30	9.16	8.45	8.73	134.85

- compilation of R functions to squeeze out some addtional performance

Info - Rcpp

Rcpp is used to integrate C++ code into R. Also provides semantic sugar like R-like datastructures for C++.

(Eddelbuettel and Francois 2011; Eddelbuettel 2013; R Core Team 2016)

Results

Comparison of the MSE

	whole	glmnet	boruta
xgboost	$5.50e+08$	$5.58e+08$	$7.61e+06$
glmnet	$7.74e+08$	$9.73e+08$	$7.39e+08$
deeplearning	$3.91e+08$	$3.79e+08$	$5.79e+06$

- smaller feature set (56 vs. 78) using Boruta feature selection $\Rightarrow$ smaller MSE
- linear model: $-5.829 + 0.9947 \cdot pred_1 - 0.0001 \cdot pred_2 + 0.0054 \cdot pred_3$
  $\Rightarrow$ Kaggle-score: 0.12553
- stacked learner MSE using linear regression
  $\Rightarrow$ 80756.74

What we learned

- overfitting is a serious issue
- in depth tuning takes time
- feature selection is mandatory if there are too many features
- not missing at random (MNAR) values are really challenging

Improvements

- feature engineering
- manual imputation of MNAR values
- longer tuning runs with limited parameter set
- prefer grid search over irace heuristic
- use of more than three basic learners


Any
Questions

Bibliography I


Athanasopoulos, George and Rob J. Hyndman (2011). "The value of feedback in forecasting competitions". In: *International Journal of Forecasting* 27.3, pp. 845–849. ISSN: 01692070. DOI: 10.1016/j.ijforecast.2011.03.002. URL: http://www.sciencedirect.com/science/article/pii/S0169207011000495.


Buuren, S. and K. Groothuis-Oudshoorn (2011). "mice: Multivariate Imputation by Chained Equations in R". In: *Journal of Statistical Software* 45.3. ISSN: 1548-7660.


Candel, Arno et al. (2015). *Deep Learning with H2O*. Third Edit. August. H2O.ai, Inc. URL: http://h2o.gitbooks.io/deep-learning/.


Chen, Tianqi and Carlos Guestrin (2016). "XGBoost: A Scalable Tree Boosting System". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '16*. New York, New York, USA: ACM Press, pp. 785–794. ISBN: 9781450342322. DOI: 10.1145/2939672.2939785. URL: http://dl.acm.org/citation.cfm?id=2939672.2939785http://dl.acm.org/citation.cfm?doid=2939672.2939785.


Chen, Tianqi, Tong He, and Michael Benesty (2016). *xgboost: Extreme Gradient Boosting*. URL: https://cran.r-project.org/package=xgboost.

Bibliography II


Ciresan, Dan et al. (2011). "A committee of neural networks for traffic sign classification". In: *The 2011 International Joint Conference on Neural Networks*. Vol. 1. 1. San Jose, California, USA: IEEE, pp. 1918–1921. ISBN: 978-1-4244-9635-8. DOI: 10.1109/IJCNN.2011.6033458. URL: http://ieeexplore.ieee.org/document/6033458/.


Ciresan, Dan et al. (2012). "Multi-column deep neural network for traffic sign classification". In: *Neural Networks* 32, pp. 333–338. ISSN: 08936080. DOI: 10.1016/j.neunet.2012.02.023. arXiv: arXiv:1202.2745v1. URL: http://dx.doi.org/10.1016/j.neunet.2012.02.023http://linkinghub.elsevier.com/retrieve/pii/S0893608012000524.


DMLC (2016). *Introduction to Boosted Trees*. URL: http://xgboost.readthedocs.io/en/latest/model.html (visited on 01/06/2017).


De Cock, Dean (2011). "Ames, Iowa: Alternative to the Boston Housing Data as an End of Semester Regression Project". In: *Journal of Statistics Education* 19.3, pp. 1–15. ISSN: 10691898. URL: www.amstat.org/publications/jse/v19n3/decock.pdf.


Eddelbuettel, Dirk (2013). *Seamless R and C++ Integration with Rcpp*. Springer. ISBN: 978-1-4614-6867-7.


Eddelbuettel, Dirk and Romain Francois (2011). "Rcpp: Seamless R and C++ Integration". In: *Journal of Statistical Software* 40.8, pp. 1–18. URL: http://www.jstatsoft.org/v40/i08/.

Bibliography III


Friedman, Jerome, Trevor Hastie, and Robert Tibshirani (2010). "Regularization Paths for Generalized Linear Models via Coordinate Descent". In: *Journal of Statistical Software* 33.1, pp. 1–22. ISSN: 1548-7660. DOI: 10.18637/jss.v033.i01. arXiv: arXiv:0908.3817v2. URL: http://www.jstatsoft.org/v33/i01/https://www.jstatsoft.org/index.php/jss/article/view/v033i01.


Harrison, David and Daniel L. Rubinfeld (1978). "Hedonic housing prices and the demand for clean air". In: *Journal of Environmental Economics and Management* 5.1, pp. 81–102. ISSN: 00950696. DOI: 10.1016/0095-0696(78)90006-2. arXiv: arXiv:1011.1669v3. URL: http://linkinghub.elsevier.com/retrieve/pii/0095069678900062.


Hastie, Trevor and Junyang Qian (2014). *Glmnet Vignette*. URL: https://web.stanford.edu/{~}hastie/glmnet/glmnet{_}alpha.html (visited on 12/30/2016).


Haykin, Simon (2008). *Neural Networks and Learning Machines*. Third Ed. Vol. 3. Pearson Education, Inc. ISBN: 9780131471399. DOI: 978-0131471399.


Homola, Daniel (2015). *BorutaPy âĂŞ an all relevant feature selection method*. URL: http://danielhomola.com/2015/05/08/borutapy-an-all-relevant-feature-selection-method/ (visited on 01/03/2017).


Horton, Nicholas J. and Stuart R. Lipsitz (2001). "Multiple Imputation in Practice". In: *The American Statistician* 55.3, pp. 244–254. ISSN: 0003-1305. DOI: 10.1198/000313001317098266. URL: http://www.tandfonline.com/doi/abs/10.1198/000313001317098266.

Bibliography IV

Kaggle Inc (2016b). *House Prices: Advanced Regression Techniques - Prizes*. URL: https://www.kaggle.com/c/house-prices-advanced-regression-techniques/details/prizes (visited on 12/28/2016).

— (2016a). *House Prices: Advanced Regression Techniques*. URL: https://www.kaggle.com/c/house-prices-advanced-regression-techniques (visited on 12/26/2016).

Kursa, Miron B. and Witold R. Rudnicki (2010). "Feature Selection with the Boruta Package". In: *Journal Of Statistical Software* 36.11, pp. 1–13. ISSN: 15487660. DOI: Vol.36, Issue11, Sep2010. URL: http://www.jstatsoft.org/v36/i11/paper.

Kursa, Miron B., Aleksander Jankowski, and Witold R. Rudnicki (2010). "Boruta - A system for feature selection". In: *Fundamenta Informaticae* 101.4, pp. 271–285. ISSN: 01692968. DOI: 10.3233/FI-2010-288. URL: http://content.iospress.com/articles/fundamenta-informaticae/fi101-4-02.

López-Ibáñez, Manuel (2012). *Automatic Design of Algorithms with iRace for Multi-objective Optimization and Anytime Optimization*. URL: http://ls11-www.cs.tu-dortmund.de/{_}media/staff/preuss/asta/ppsn2012/lopezibaneztalkppsn2012.pdf.

López-Ibáñez, Manuel et al. (2016). "The irace package: Iterated racing for automatic algorithm configuration". In: *Operations Research Perspectives* 3, pp. 43–58. ISSN: 22147160. DOI: 10.1016/j.orp.2016.09.002. URL: http://linkinghub.elsevier.com/retrieve/pii/S2214716015300270.

Metz, Rachel (2013). "A Startup Called Kaggle Tries to Bring Smart People to Knotty Problems". In: *MIT Technology Review*, pp. 1–2. URL: https://www.technologyreview.com/lists/innovators-under-35/2013/entrepreneur/anthony-goldbloom/.

Bibliography V


Nilsson, Roland et al. (2007). "Consistent Feature Selection for Pattern Recognition in Polynomial Time". In: *The Journal of Machine Learning Research* 8, pp. 589–612. ISSN: 15324435. URL: http://www.jmlr.org/papers/volume8/nilsson07a/nilsson07a.pdf.


R Core Team (2016). *Byte Code Compiler*. Vienna, Austria. URL: https://www.r-project.org/.


Rudnicki, Witold R., Mariusz Wrzesień, and Wiesław Paja (2015). "All Relevant Feature Selection Methods and Applications". In: *Studies in Computational Intelligence*. Ed. by Urszula Stańczyk and Lakhmi C. Jain. Vol. 584. Springer Berlin Heidelberg. Chap. Chapter 2, pp. 11–28. ISBN: 978-3-662-45619-4. DOI: 10.1007/978-3-662-45620-0_2. URL: http://www.scopus.com/inward/record.url?eid=2-s2.0-84921712130{\&}partnerID=tZOtx3y1http://link.springer.com/10.1007/978-3-662-45620-0{_}2.


Schmidhuber, Jürgen (2015). "Deep learning in neural networks: An overview". In: *Neural Networks* 61, pp. 85–117. ISSN: 08936080. DOI: 10.1016/j.neunet.2014.09.003. arXiv: 1404.7828. URL: http://linkinghub.elsevier.com/retrieve/pii/S0893608014002135.


Silver, David et al. (2016). "Mastering the game of Go with deep neural networks and tree search". In: *Nature* 529.7587, pp. 484–489. ISSN: 0028-0836. DOI: 10.1038/nature16961. URL: http://www.nature.com/doifinder/10.1038/nature16961.


Taieb, Souhaib Ben and Rob J. Hyndman (2013). "A gradient boosting approach to the Kaggle load forecasting competition". In: *International Journal of Forecasting* 30.2, pp. 382–394. ISSN: 01692070. DOI: 10.1016/j.ijforecast.2013.07.005. URL: http://www.sciencedirect.com/science/article/pii/S0169207013000812.

Bibliography VI

Tibshirani, Robert (1996). "Regression Shrinkage and Selection via the Lasso". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 58.1, pp. 267–288. URL: http://www.jstor.org/stable/2346178.

Vink, Gerko et al. (2014). "Predictive mean matching imputation of semicontinuous variables". In: *Statistica Neerlandica* 68.1, pp. 61–90. ISSN: 00390402. DOI: 10.1111/stan.12023. URL: http://doi.wiley.com/10.1111/stan.12023.

Wickham, Hadley (2015). *R Packages*. 1st Edit. O'Reilly Media. ISBN: 978-1491910597. URL: http://r-pkgs.had.co.nz/.

Wikipedia (2016). *Kaggle*. URL: https://en.wikipedia.org/wiki/Kaggle (visited on 11/27/2016).