Machine learning for the prediction of railway fares

Bachelor Thesis Report

Fabian Hirschmann

Technische Universität Darmstadt
Knowledge Engineering Group
Algorithm Engineering Group

October 31th, 2013

Introduction

Goal: Predict fare cost based on attributes such as the distance travelled on various types of trains.

Multi-Objective Traffic Information System (MOTIS)

- Most work has focused on finding the fastest connections
- MOTIS allows to find train connections with respect to the ticket cost
- Uses a black-box pricing component provided by Deutsche Bahn
- MOTIS needs a *fast* fare prediction in order to optimize for the ticket cost

The German railway system

Trains can be divided into 3 classes:

- Class 0: Long-distance and high-speed trains (ICE)
- Class 1: Slower express trains (IC/EC)
- Class 3: Regional trains (RB/RE)

Class 2 trains can be neglected due to scarcity.

Sampling

Question: How to sample data instances?

Sampling

Question: How to sample data instances?
Idea: Weight stations according to the number of incoming and outgoing connections.

Probability Sampling

Rank	Name	Weight	Probability
1	Hannover Hbf	5691	0.010885
2	Köln Hbf	4890	0.009353
3	Frankfurt(Main)Hbf	4670	0.008932
4	Düsseldorf Hbf	4424	0.008462
5	Hamburg Hbf	4207	0.008047
6	Duisburg Hbf	4114	0.007869
7	Mannheim Hbf	3811	0.007289
8	Berlin-Spandau	3740	0.007153
9	Dortmund Hbf	3607	0.006899
10	Nürnberg Hbf	3605	0.006895
11	F-Flughafen Fernbf.	3509	0.006712
12	Würzburg Hbf	3463	0.006624
13	Göttingen	3424	0.006549
14	Kassel-Wilhelmshöhe	3392	0.006488
15	Fulda	3358	0.006423
16	Hamburg Dammtor	3248	0.006212

Data Sample

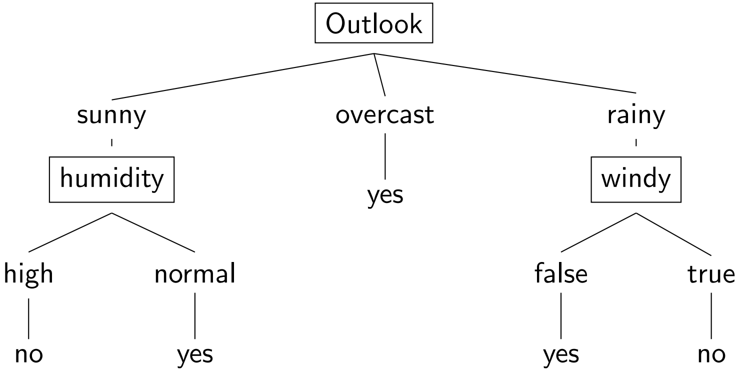
duration	transfers	stops	dist_0	dist_1	dist_3	dist	lindist	price
131	1	9	0	150	4	153	138	3400
373	3	32	0	226	191	417	305	7000
80	0	5	0	178	0	178	121	3300
379	3	25	413	0	185	598	393	10200
247	2	15	0	346	74	420	301	6700
864	5	79	0	0	731	731	294	7430
339	4	35	507	0	104	610	421	11800
104	0	4	229	0	0	229	172	4300
147	2	29	0	0	122	122	71	1870
480	3	20	265	309	70	643	446	9700
64	0	4	0	90	0	90	78	1950
398	2	29	0	0	446	446	332	5670
232	2	18	132	0	71	203	118	4400
207	1	16	0	140	57	196	154	4100

Algorithms

Methods for prediction utilized include:

- Decision Trees (M5, Cubist)
- Support Vector Machines (SVMs)
- Multivariate Adaptive Regression Splines (MARS)
- Artificial Neural Networks (Multilayer Perceptron)

Decision Trees: Example Tree



Decision Trees: ID3 (Quinlan 1986)

Recursively builds a tree and

- uses information theory to decide which attribute to split the data with
- creates a leaf when every instance belongs to the same class
- chooses the majority class when there are no more attributes to be selected

Decision Trees: C4.5 (Quinlan 1993a)

- Can deal with continuous `predictors` by creating a threshold value that splits the data set into two sets
- Can prune trees if the expected error is greater than the error in a single leaf

Decision Trees: M5 (Quinlan 1992)

- Can deal with numeric predicted values
- Builds a piecewise linear model, i.e. terminal leaves contain linear regression models
- Similar model (`M5P`) invented by Wang and Witten (1997); part of `Weka` (Hall et al. 2009)

Decision Trees: Cubist

- Supports an ensemble method called committees, where iterative model trees are created in sequence
- Applies the nearest-neighbor algorithm (Quinlan 1993b)
- Deduces if-then-else rules (Quinlan 1987)

Multivariate Adaptive Regression Splines (Friedman 1991)

Multivariate Adaptive Regression Splines (MARS)

- are an extension of linear models
- model nonlinearities and the interaction between predictors
- use *hinge* functions to take into account nonlinearities

Hinge functions

Hinge functions can be written as

$$h(x) := \max(0, x) = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{if } x \leq 0 \end{cases}$$

where $\max(a, b)$ is a if $a > b$ else b.

MARS models

A MARS model has the form

$$y(\mathbf{x}) = \sum_{i=1}^{k} w_i \phi_i(\mathbf{x})$$

where w_i are constant coefficients and ϕ_i is a basis function which can take any of the following forms:

- a constant 1
- a hinge function h
- a product of two or more hinge functions

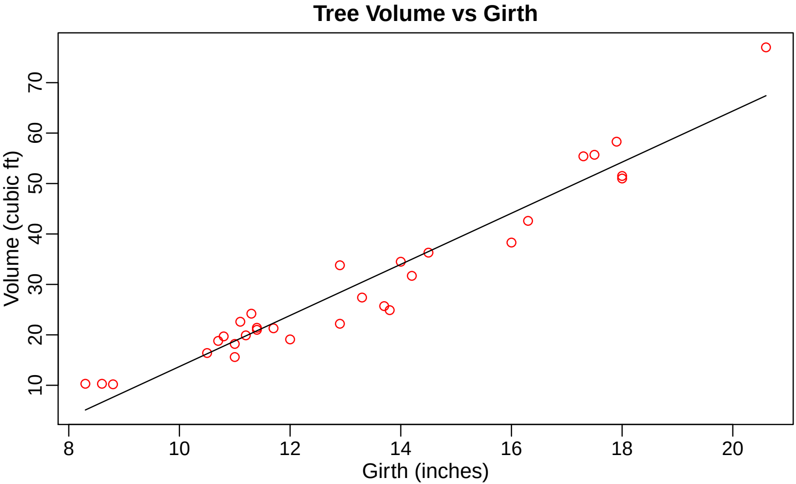


Figure: Scatter plot with fitted linear regression line.

$$\text{Volume} = -36.943 + 5.066 \cdot \text{Girth}$$

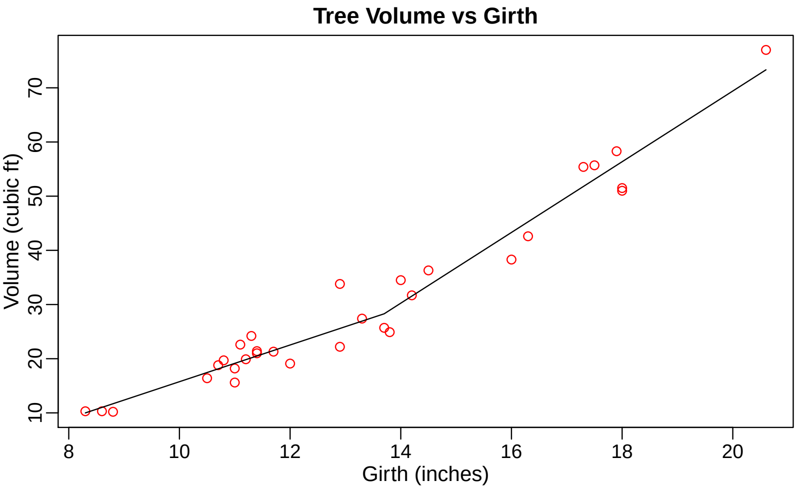


Figure: Scatter plot with fitted MARS model.

$$\text{Volume} = 28.3 + 6.5 \cdot h(\text{Girth} - 13.7) - 3.4 \cdot h(13.7 - \text{Girth})$$

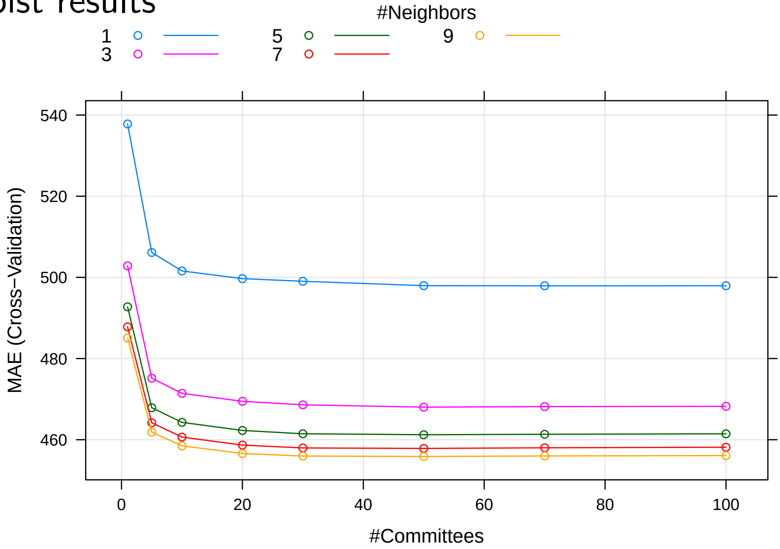
Implementation

- Experiments implemented in GNU R (R Core Team 2012)
- The `caret` package (Kuhn 2008; Kuhn 2013) is a framework for predictive modelling that integrates several other packages
- Packages used include `cubist` (Kuhn et al. 2013), `kernlab` (Karatzoglou et al. 2004) for SVMs, `RWeka` (Hornik, Buchta, and Zeileis 2009) for `M5`, and `RSNNS` (Bergmeir and Benítez 2012) for neural networks

Evaluation and Validation

- Validated using 10-fold cross-validation
- Tuned using a tuning grid, e.g.
  $G := N \times C = \{(n, c) | n \in N \wedge c \in C\}$ for cubist

Cubist results



Cubist model

```
Model 1:
  Rule 1/1: [568 cases, mean 1292.3, range 360 to 2960, est err 75.7]
    if
    duration <= 106
    dist_0 <= 1.67477
    dist_1 <= 9.4963
    dist_3 > 21.0408
    then
    outcome = 146 + 266.6 dist_1 + 128.8 dist_0 + 4.7 dist
              + 8.1 dist_3 + 1.3 duration + 1.2 lindist - 3 stops

  Rule 1/2: [41 cases, mean 1861.5, range 130 to 6800, est err 294.0]

    if
    dist_0 <= 1.67477
    dist_1 <= 9.4963
    dist_3 <= 21.0408
    then
    outcome = 3.6 + 70.9 dist_0 + 26 lindist + 16.7 dist_1
              - 16.9 dist_3 + 78 stops - 2.7 dist
```

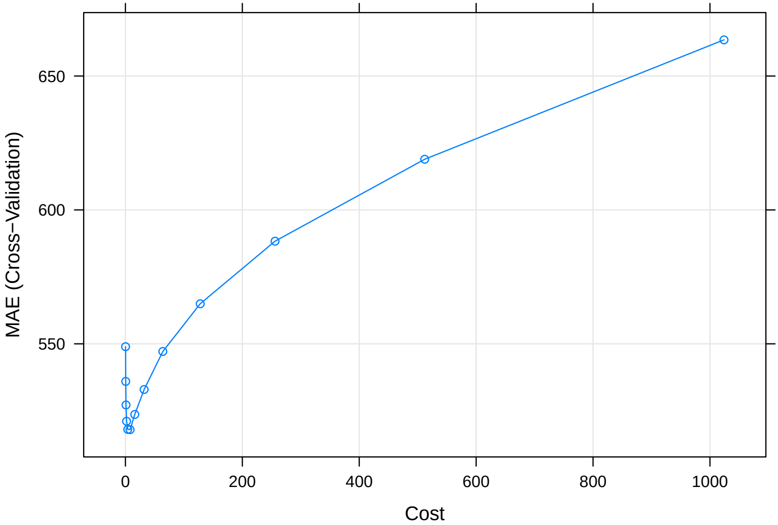
MARS model

$$
\begin{aligned}
\text{price} = \\
&+ 9507.948 \\
&+ 0.978 \cdot h(\text{dist} - 636.22) \\
&- 1.513 \cdot h(636.22 - \text{dist}) \\
&+ 1.085 \cdot h(\text{dist}_0 - 182.81) \\
&- 7.376 \cdot h(182.81 - \text{dist}_0) \\
&+ 0.003 \cdot h(182.81 - \text{dist}_0) \cdot h(\text{dist}_1 - 155.667) \\
&- 0.01 \cdot h(182.81 - \text{dist}_0) \cdot h(155.667 - \text{dist}_1) \\
&- 8.163 \cdot h(\text{lindist} - 558.684) \\
&- 7.045 \cdot h(558.684 - \text{lindist}) \\
&\ldots
\end{aligned}
$$

MARS model (continued)

$$
\begin{aligned}
\text{price} = {} & \\
& \ldots \\
& + 0.037 \cdot h(\text{dist} - 340.687) \cdot h(558.684 - \text{lindist}) \\
& - 0.017 \cdot h(340.687 - \text{dist}) \cdot h(558.684 - \text{lindist}) \\
& - 0.046 \cdot h(\text{dist}_3 - 279.84) \cdot h(636.22 - \text{dist}) \\
& + 0.007 \cdot h(279.84 - \text{dist}_3) \cdot h(636.22 - \text{dist}) \\
& - 0.596 \cdot h(3 - \text{transfers}) \cdot h(558.684 - \text{lindist}) \\
& + 10.503 \cdot h(\text{lindist} - 378.519)
\end{aligned}
$$

SVM results (radial kernel)



SVM results

Number of support vectors:

- polynomial kernel: 7083
- radial kernel: 7013

when trained using a data consisting of 14000 instances

The current method

The current method (Harnisch and Nuhn 2010) implemented in MOTIS is given by

$$\begin{aligned}\text{price} = &\\ &+ \min(12200, \max(700, 23.917 \cdot \text{dist} - 0.0122 \cdot \text{dist}_0^2 + 622.29))\\ &+ \min(11700, \max(600, 18.433 \cdot \text{dist} - 0.0073 \cdot \text{dist}_1^2 + 334.79))\\ &+ 14 \cdot \text{dist}_3\end{aligned}$$

and is made up of three separate linear regression models.

The old method

The original method implemented in MOTIS is given by

$$\text{price} = \begin{cases} 14 \cdot \text{dist} + 1200 & \text{if } \text{dist}_0 > 0 \\ 14 \cdot \text{dist} + 700 & \text{if } \text{dist}_0 = 0 \text{ and } \text{dist}_1 > 0 \\ 14 \cdot \text{dist} & \text{otherwise} \end{cases}$$

and adds a surcharge according to the highest train class involved.

Results

	Rank	MAE	RMSE	RRSE	RAE
Cubist Trees	1	456	673	0.206	0.163
M5	2	487	723	0.223	0.177
SVM (Radial)	3	518	760	0.235	0.188
SVM (Poly)	4	531	781	0.241	0.193
MARS	5	593	826	0.255	0.215
Current Method	6	597	797	0.246	0.217
Linear Regression	7	810	1090	0.335	0.294
Old Method	8	817	1250	0.387	0.297
Baseline (Mean)	9	2790	3270	1.000	1.000
Neural Net (MLP)	10	2810	3310	1.020	1.020

Time evaluation

	Prediction Time (s)
Linear Regression	0.07
M5	0.13
MARS	0.46
Cubist Trees	3.07
SVM (Radial)	5.22
SVM (Poly)	10.66
Neural Net (MLP)	13.70

Table: Time spent for the prediction of 3000 new data instances.

Conclusion

- Probability-based sampling method proposed
- Current prediction model can be beat (but is good already)
- Decision tree learner cubist provided the best results

Thanks. Questions?

Appendix

Probability Sampling

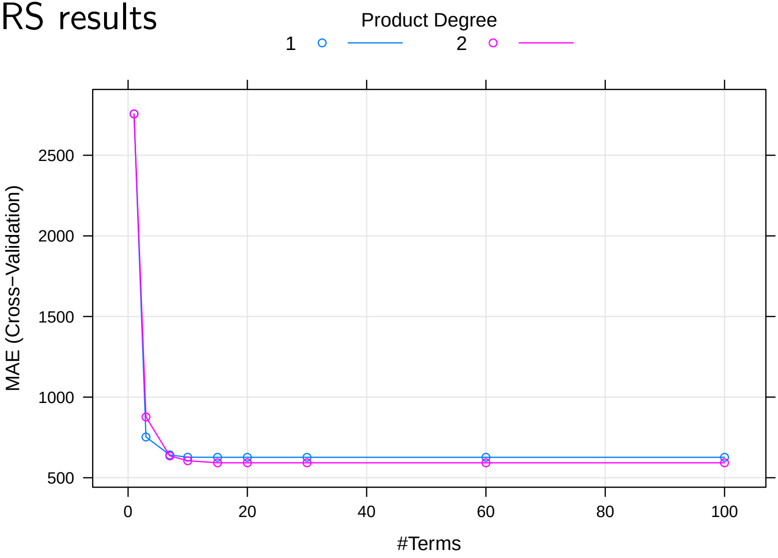
Each train station s_i, $i \in \{1, ..., n\}$ is assigned the value w_i of the application of a weight function:

$$\text{weight} : \mathbb{N} \times \mathbb{N} \times \mathbb{N} \times \mathbb{N} \to \mathbb{N}$$
$$\text{weight}(c_0, c_1, c_2, c_{\text{rbre}}) := 6c_0 + 5c_1 + 4c_2 + 1c_{\text{rbre}}$$

Evaluation: Metrics

$$
\begin{aligned}
\mathrm{mse}(p, a) &:= \frac{(p_1 - a_1)^2 + \ldots + (p_n - a_n)^2}{n} \\
\mathrm{rmse}(p, a) &:= \sqrt{\mathrm{mse}(p, a)} \\
\mathrm{mae}(p, a) &:= \frac{|p_1 - a_1| + \ldots + |p_n - a_n|}{n} \\
\mathrm{rse}(p, a) &:= \frac{(p_1 - a_1)^2 + \ldots + (p_n - a_n)^2}{(a_1 - \overline{a})^2 + \ldots + (a_n - \overline{a})^2} \\
\mathrm{rrse}(p, a) &:= \sqrt{\mathrm{rse}(p, a)} \\
\mathrm{rae}(p, a) &:= \frac{|p_1 - a_1| + \ldots + |p_n - a_n|}{|a_1 - \overline{a}| + \ldots + |a_n - \overline{a}|}
\end{aligned}
$$

MARS results



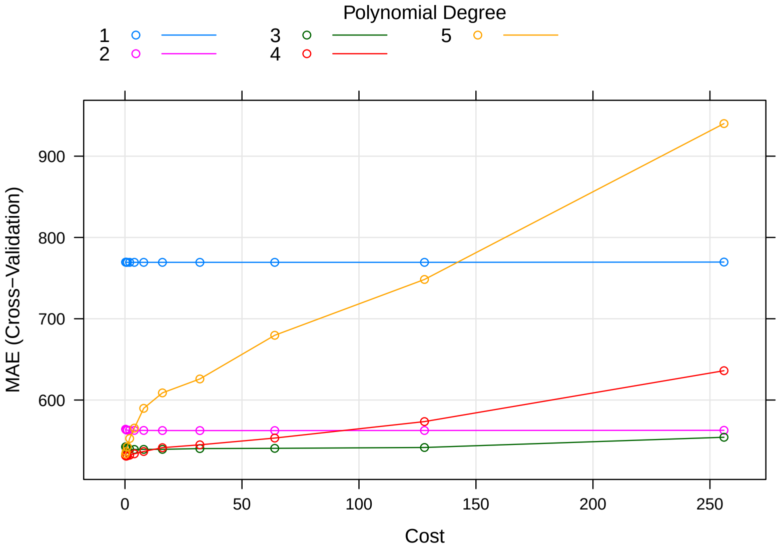
MARS results (continued)

Degree	Nprune	MAE	RMSE	RRSE	RAE	Rsq
1	1	2757	3240	1.000	1.000	
1	3	753	1008	0.311	0.273	0.903
1	7	642	874	0.270	0.233	0.927
1	10	627	856	0.264	0.228	0.930
1	15	627	855	0.264	0.227	0.930
1	20	627	855	0.264	0.227	0.930
1	30	627	855	0.264	0.227	0.930
1	60	627	855	0.264	0.227	0.930
1	100	627	855	0.264	0.227	0.930
2	1	2757	3240	1.000	1.000	
2	3	877	1141	0.352	0.318	0.876
2	7	635	859	0.265	0.231	0.930
2	10	605	836	0.258	0.220	0.933
2	15	**593**	**826**	**0.255**	**0.215**	**0.935**
2	20	**593**	**826**	**0.255**	**0.215**	**0.935**
2	30	**593**	**826**	**0.255**	**0.215**	**0.935**
2	60	**593**	**826**	**0.255**	**0.215**	**0.935**
2	100	**593**	**826**	**0.255**	**0.215**	**0.935**

SVM results (radial kernel)

C	Sigma	MAE	RMSE	RRSE	RAE	Rsq
0.25	0.13	549	802	0.248	0.199	0.939
0.50	0.13	536	787	0.243	0.194	0.942
1.00	0.13	527	775	0.239	0.191	0.943
2.00	0.13	521	766	0.236	0.189	0.944
4.00	0.13	**518**	761	**0.235**	**0.188**	**0.945**
8.00	0.13	**518**	**760**	**0.235**	**0.188**	**0.945**
16.00	0.13	524	767	0.237	0.190	0.944
32.00	0.13	533	780	0.241	0.193	0.942
64.00	0.13	547	802	0.248	0.198	0.939
128.00	0.13	565	831	0.257	0.205	0.935
256.00	0.13	588	872	0.269	0.213	0.928
512.00	0.13	619	928	0.287	0.225	0.919
1024.00	0.13	664	1013	0.313	0.241	0.905

SVM results (polynomial kernel)



Neural Network results

Size	Decay	MAE	RMSE	RRSE	RAE	Rsq
1	0.000000	5732	6420	1.982	2.078	0.003
3	0.100000	4390	5089	1.572	1.594	**0.012**
5	0.053367	4299	5018	1.550	1.561	0.006
7	0.028480	3412	4024	1.241	1.237	0.008
9	0.015199	3850	4475	1.382	1.398	0.008
11	0.008111	3155	3773	1.165	1.145	0.006
13	0.004329	3757	4392	1.354	1.362	0.004
15	0.002310	3426	4035	1.245	1.241	0.004
17	0.001233	**2807**	**3314**	**1.023**	**1.018**	0.003
19	0.000658	3644	4396	1.358	1.323	0.006
21	0.000351	3728	4424	1.361	1.348	
23	0.000187	3367	3986	1.232	1.219	0.003
25	0.000100	3765	4596	1.433	1.380	

Residual Analysis

$$\text{residual} = \text{actual} - \text{predicted}$$

Residual Analysis (Cubist)

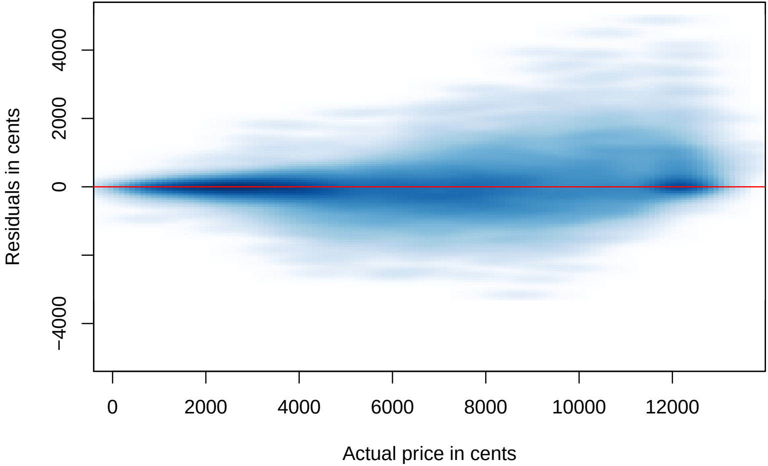


Figure: Points below the red line are overpredicted; points above are underpredicted. Darker regions represent a higher density of points.

Residual Analysis (MARS)

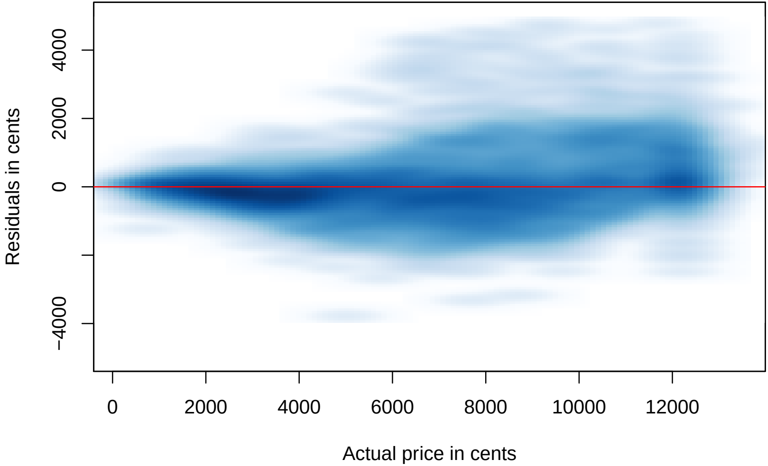


Figure: Points below the red line are overpredicted; points above are underpredicted. Darker regions represent a higher density of points.

Residual Analysis (svmRadial)

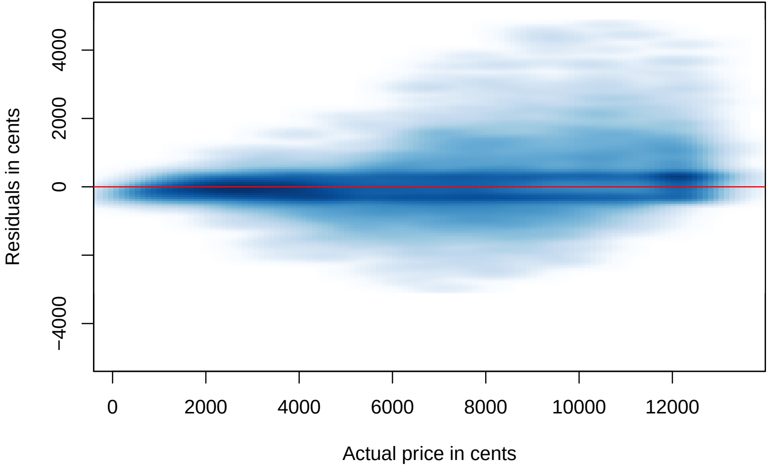


Figure: Points below the red line are overpredicted; points above are underpredicted. Darker regions represent a higher density of points.

Residual Analysis (Current Method)

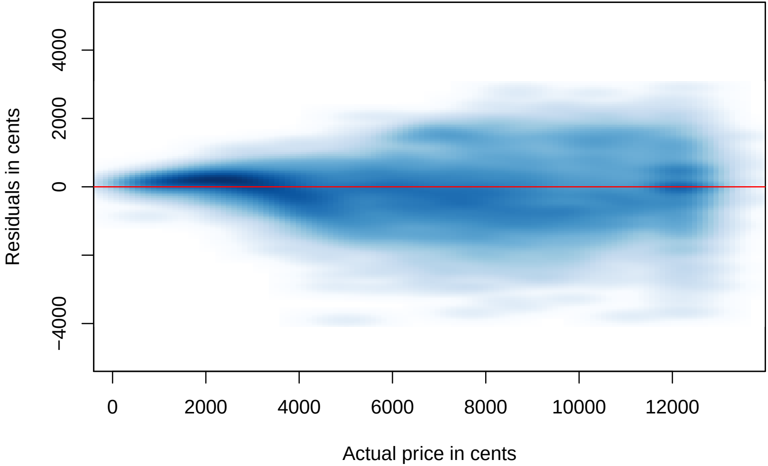


Figure: Points below the red line are overpredicted; points above are underpredicted. Darker regions represent a higher density of points.

Decision Trees: Entropy (measure of uncertainty)

For the weather problem:

$$E(\mathcal{D}) := -p_{\oplus} log_2 p_{\oplus} - p_{\ominus} log_2 p_{\ominus}$$
$$E(\text{Outlook} = \text{sunny}) = -\frac{2}{5} log_2 \frac{2}{5} - \frac{3}{5} log_2 \frac{3}{5} = 0.971$$

For more than two classes:

$$E(\mathcal{D}) := -\sum_i p_i \log_2 p_i$$

where

- $\mathcal{D}$ is the set of instances
- p_i is the proportion of samples in class i

Decision Trees: Information Gain

The Information Gain is given by

$$\mathsf{IG}(\mathcal{D}, A) = E(\mathcal{D}) - \sum_i \frac{|\mathcal{D}_i|}{|\mathcal{D}|} \cdot E(\mathcal{D}_i)$$

Bibliography I

Bergmeir, Christoph and José M. Benítez (2012). Neural Networks in R Using the Stuttgart Neural Network Simulator: RSNNS. In: *Journal of Statistical Software*, 46.7, pp. 1–26 (cit. on p. 20).

Friedman, Jerome H. (1991). Multivariate adaptive regression splines. In: *The Annals of Statistics*, 19, pp. 1–67. ISSN: 00905364 (cit. on p. 15).

Hall, Mark, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten (2009). The WEKA data mining software: an update. In: *SIGKDD Exploration Newsletter*, 11.1, pp. 10–18. ISSN: 19310145 (cit. on p. 13).

Harnisch, Stefan and Helge Nuhn (2010). *Analyse und Verbesserung der Preisschätzung im MOTIS-System*. German. Tech. rep. Technische Universität Darmstadt (cit. on p. 28).

Hornik, Kurt, Christian Buchta, and Achim Zeileis (2009). Open-Source Machine Learning: R Meets Weka. In: *Computational Statistics*, 24.2, pp. 225–232. ISSN: 09434062 (cit. on p. 20).

Bibliography II

Karatzoglou, Alexandros, Alexander J. Smola, Kurt Hornik, and Achim Zeileis (2004). kernlab – An S4 Package for Kernel Methods in R. In: *Journal of Statistical Software*, 11.9, pp. 1–20 (cit. on p. 20).

Kuhn, Max (2008). Building Predictive Models in R Using the caret Package. In: *Journal of Statistical Software*, 28.5, pp. 1–26. ISSN: 15487660 (cit. on p. 20).

— (2013). *caret: Classification and Regression Training*. R package version 5.16-04 (cit. on p. 20).

Kuhn, Max, Steve Weston, Chris Keefer, and Nathan Coulter (2013). *Cubist: Rule- and Instance-Based Regression Modeling*. R package version 0.0.13 (cit. on p. 20).

Quinlan, John R. (1986). Induction of decision trees. In: *Machine Learning*, 1.1, pp. 81–106. ISSN: 08856125 (cit. on p. 11).

— (1987). Generating production rules from decision trees. In: *Proceedings of the 10^{th} International Joint Conference on Artificial Intelligence*. Vol. 1. Milan, Italy: Morgan Kaufmann, pp. 304–307 (cit. on p. 14).

Bibliography III

Quinlan, John R. (1992). Learning with continuous classes. In: *Proceedings of the 5th Australian Joint Conference on Artificial Intelligence*. Vol. 92, pp. 343–348. ISBN: 981021250X (cit. on p. 13).

— (1993a). *C4.5: programs for machine learning*. San Francisco, CA, USA: Morgan Kaufmann. ISBN: 1558602380 (cit. on p. 12).

— (1993b). Combining instance-based and model-based learning. In: *Proceedings of the 10th International Conference on Machine Learning*. Amherst, MA, USA, pp. 236–243 (cit. on p. 14).

R Core Team (2012). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria. ISBN: 3900051070 (cit. on p. 20).

Wang, Yong and Ian H. Witten (1997). Inducing model trees for continuous classes. In: *Proceedings of the 9th European Conference on Machine Learning*, pp. 128–137 (cit. on p. 13).