Project 2

Least squares regression and nearest neighbor classifiers

Lukas Drexler, Leif Van Holland, Reza Jahangiri, Mark Springer, Maximilian Thiessen

Rheinische Friedrich-Wilhelms-Universität

December 19, 2017

Least squares regression for missing value prediction

Height and weight data:

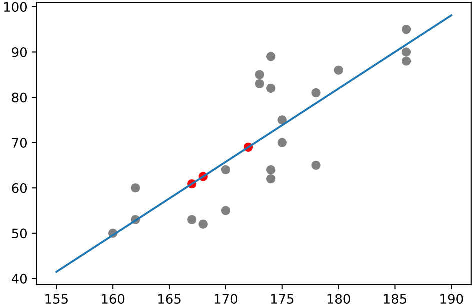
$$\mathbf{x} = [x_1, ..., x_n]^T \text{ and } \mathbf{y} = [y_1, ..., y_n]^T$$

Fit polynomials:

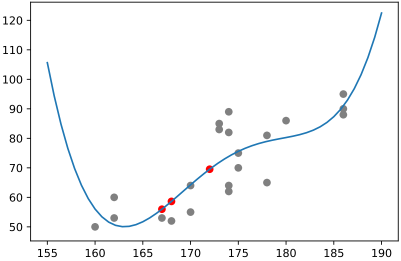
$$y(x) = \sum_{j=0}^{d} w_j x^j$$

Use least squares method with Vandermonde matrix:

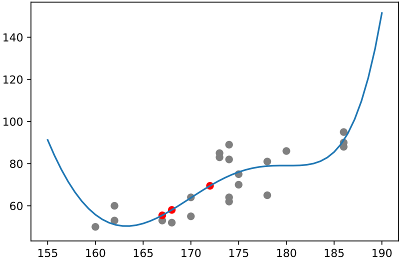
$$\begin{pmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^n \\ 1 & x_1 & x_1^2 & \cdots & x_1^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^n \end{pmatrix}$$



d=1, error = 38.8113253569



d=5, error = 37.2460869099



d=10, error = 36.9649147995

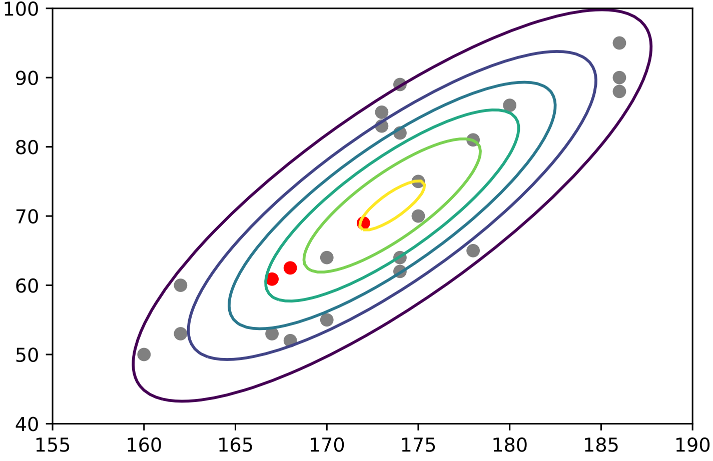
Conditional expectation for missing value prediction

Fit bi-variate Gaussian and use conditional Expectation for missing value prediction:

$$\begin{aligned}\mathbb{E}[w|h_0] &= \int wp(w|h_0)dw \\ &= \mu_w + \rho\frac{\sigma_w}{\sigma_h}(h_0 - \mu_h)\end{aligned}$$

Numeric Results

The red points have the predicted weight.



Bayesian regression for missing value prediction

Compare fifth degree polynomial

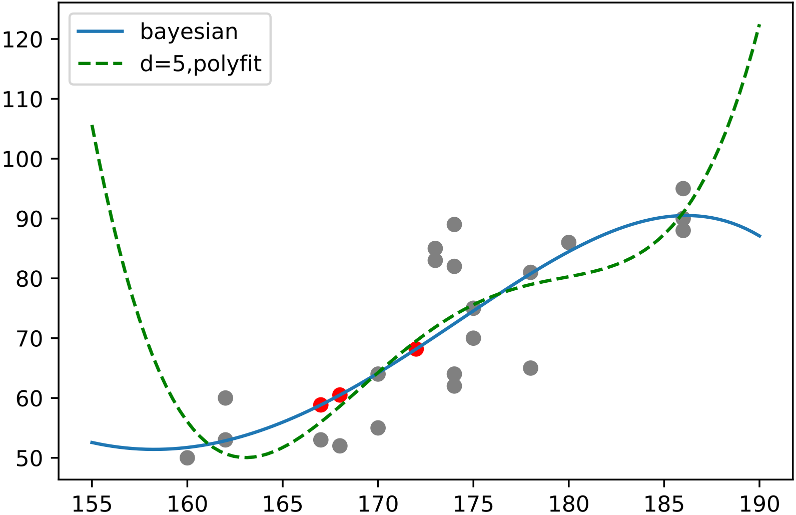
$$y(x) = \sum_{j=0}^{5} w_j x^j$$

to a Bayesian regression assuming a Gaussian prior

$$p(\mathbf{w}) \sim \mathcal{N}(\mathbf{w}|\mu_0, \sigma_0^2 \mathbf{I})$$

with $\mu_0 = \mathbf{0}$ and $\sigma_0^2 = 3$

Results


bayesian
d=5,polyfit
120
110
100
90
80
70
60
50
155
160
165
170
175
180
185
190

Boolean functions and the Boolean Fourier transform

(Naive) k nearest neighbor

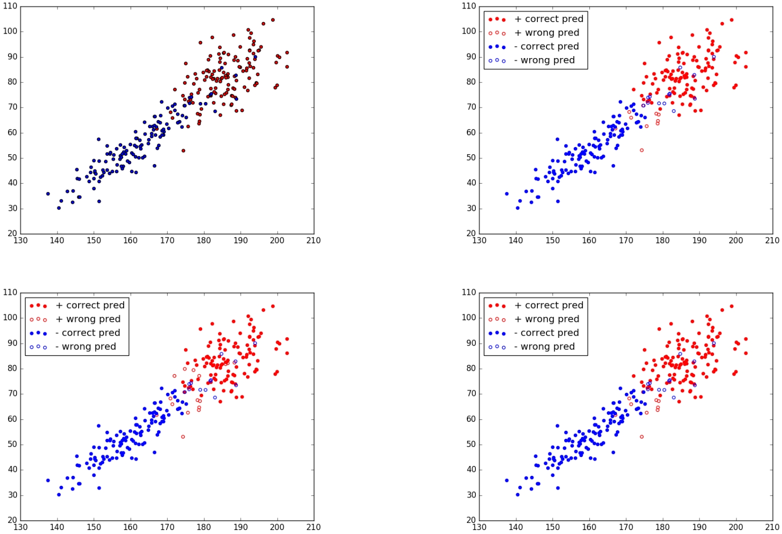
Use the train data set as a prediction for the test data set

Compute the k nearest neighbors for each data point in the data set

And use the majorant of those k labels as a prediction

Experiment on `data2.dat` with $k \in 1, 3, 5$

Results prediction



Accuracy and running time

Accuracies of kNN predictions:

- 0.77 for $k = 1$
- 0.82 for $k = 3$
- 0.83 for $k = 5$

Running times of our distance calculation vs `sklearn.metrics.pairwise`:

k	our implementation	`sklearn`
1	0.02s	0.003s
3	0.02s	0.003s
5	0.02s	0.003s

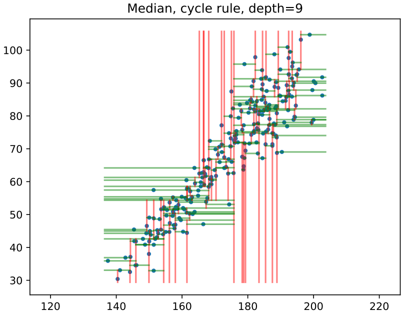
KDTrees

Plot four different KDTrees for combinations of axis-cycling rules:

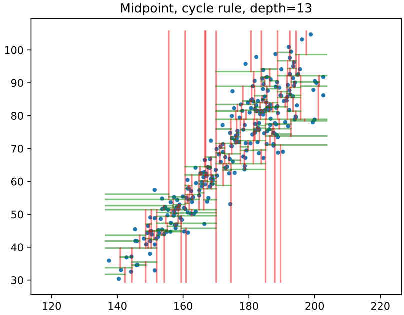
- cycle through axes
- select axis with highest variance

and the split point rules w.r.t. the splitting axis:

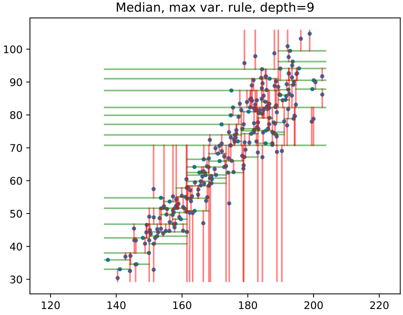
- select the median point
- select the midpoint



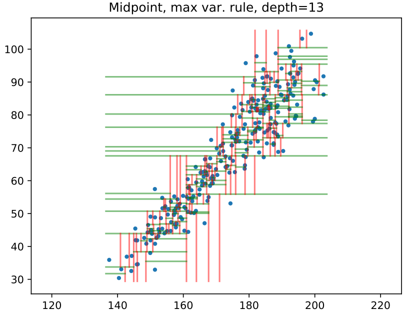
Median, cycle rule, depth=9



Midpoint, cycle rule, depth=13



Median, max var. rule, depth=9



Midpoint, max var. rule, depth=13

Timings for 1-NN per combination

	Median	Midpoint
Cycle	0.0142476272583s	0.0099373626709s
Max. Var	0.0136028242111s	0.0105255293846s

Table: Mean running time in seconds, 100 runs