Tuning Hyperparameters

Alexander Brenning

Department of Geography, Friedrich Schiller University Jena

Geo 408B

Hyperparameters

- The classifier C_L is conditional on the learning sample, L!
- A classification technique is a procedure used for constructing C_L from a learning sample L:

$$C: L \rightarrow C_L$$

- **Parametric techniques**: C_L is derived by estimating coefficients $\beta_1, \dots, \beta_p$ from the learning sample
- **Non-parametric techniques**: Often algorithmic, cannot be written as a simple mathematical formula, or do not involve parameters
- Classifiers may also depend on **hyperparameters** that control their general behaviour:
  - E.g. in k-NN: $\theta = k$

Examples of Hyperparameters in Different Classification Techniques

- k-Nearest-Neighbour Classification: $k \in \{1,2,3,4,\dots\}$

- Which other hyperparameters are you aware of?

Examples of Hyperparameters in Different Classification Techniques

- k-Nearest-Neighbour Classification: $k \in \{1,2,3,4,\ldots\}$

- Which other hyperparameters are you aware of?

- What characteristics do these hyperparameters have?
- What might happen if you picked a hyperparameter outside its domain?

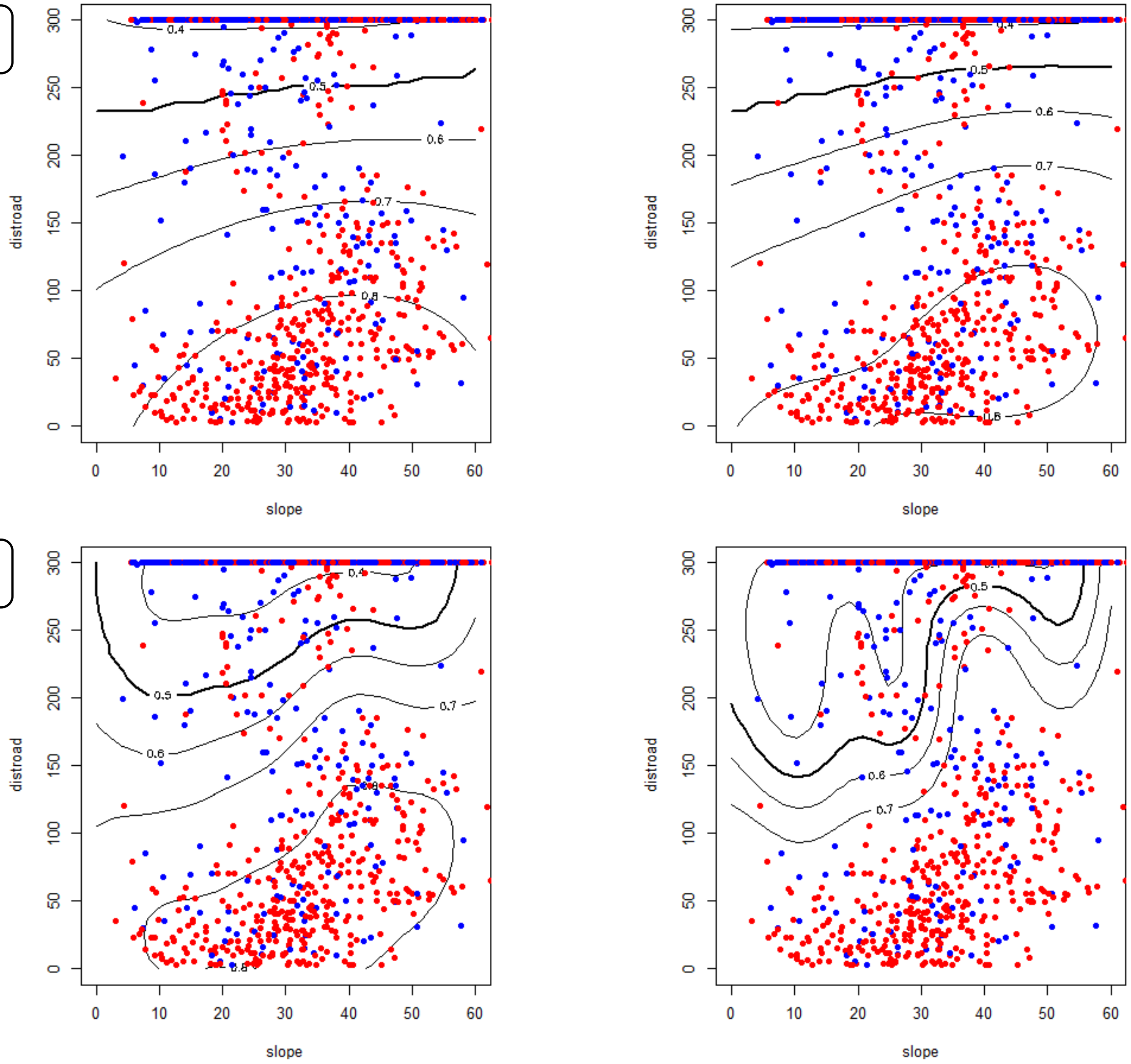
Dependence of SVM Predictions on Hyperparameters (1)

SVM (C=.1, γ=.1)

SVM (C=1, γ=.1)

SVM (C=.1, γ=1)

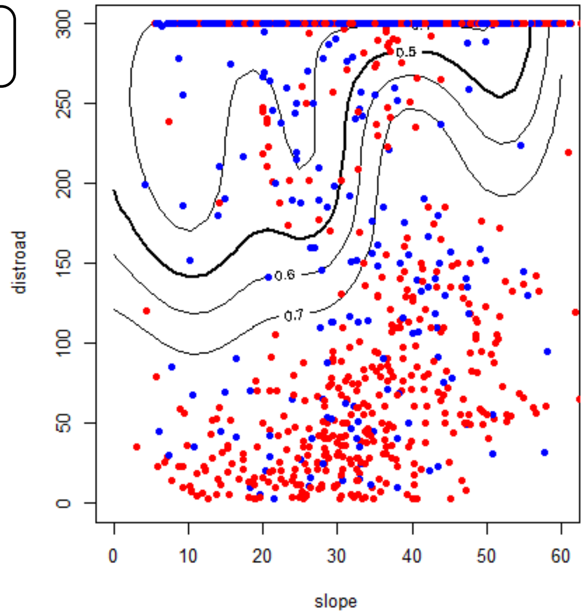
SVM (C=1, γ=1)



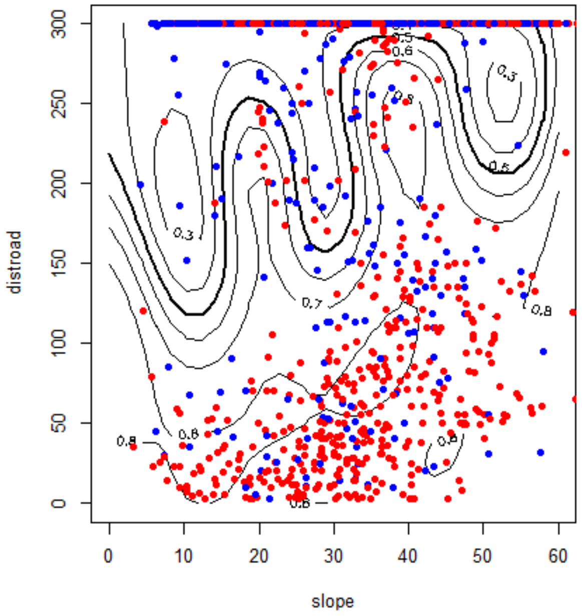
C-classification,
radial basis function kernel

Dependence of SVM Predictions on Hyperparameters (2)

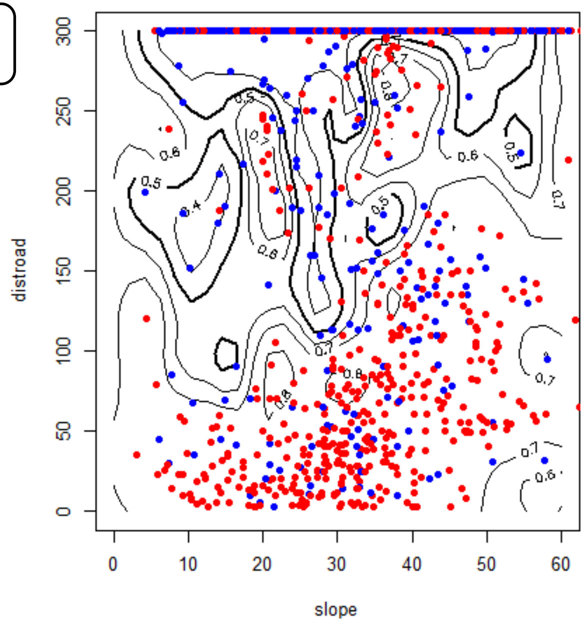
SVM (C=1, γ=1)



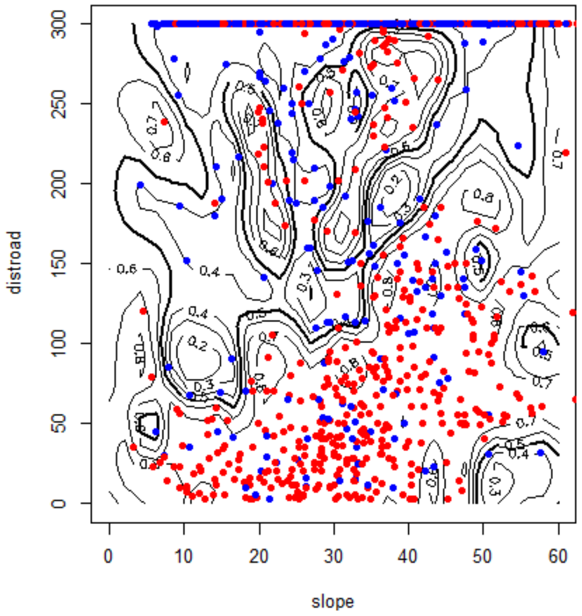
SVM (C=10, γ=1)



SVM (C=1, γ=10)



SVM (C=10, γ=10)



C-classification,
radial basis function kernel

Dealing with Hyperparameters

- What strategies for dealing with hyperparameters are you familiar with?

Hyperparameter Tuning (i.e. Optimization)

- We consider the performance measure to be a function of the hyperparameters, e.g.:

$$auroc(\theta)$$

- How can we find the/an optimal value of θ?

Hyperparameter Tuning (i.e. Optimization)

- **Grid search**
  - Simplest, perhaps most widely used
  - Discretize the hyperparameter domain using a grid
  - E.g. 10 grid points in each direction → 1000 estimates of performance required

- **Random search**
  - Specify a maximum number of iterations (e.g. 50) and/or a convergence criterion
  - More efficient when dealing with multiple / many hyperparameters, some of which are redundant
  - Inhomogeneous coverage of hyperparameter space, i.e. large gaps possible
  - Not recommended when using only one hyperparameter

Hyperparameter Tuning (i.e. Optimization)

- **Model-based optimization**
  - Uses e.g. kriging to interpolate the performance function in hyperparameter space
- **Simulated annealing**
- **Covariance Matrix Adaptation Evolution Strategy** (CMA-ES)

All the previously mentioned methods are implemented in R package mlr.

Hyperparameter Tuning (i.e. Optimization)

- **Gradient-descent methods**
  - Based on numerically calculating the first derivative of the objective function, and following the direction of the steepest gradient until reaching an optimum
  - Only works for numeric hyperparameters and a performance function that is differentiable with respect to the hyperparameters („smooth“)
  - May get stuck in local optima
  - Problem: Hyperparameter tuning is often a non-smooth, non-convex problem (i.e. non-differentiable, with local minima)

What Have We Learned

- Hyperparameters can be critical in determining a model‘s performance, especially in the case of flexible machine-learning techniques.
- Default values implemented in software may be OK, but they may also be completely inadequate for a particular classification task.
- Built-in optimizers (e.g. pruning in rpart) may be using performance measures that are irrelevant for our problem at hand.
- Hyperparameter tuning is computationally expensive (e.g. 1000-fold increase in computing time).
- If not done properly, hyperparameter tuning may lead to reporting over-optimistic performance estimates.
    - Always tune hyperparameters within an inner cross-validation.