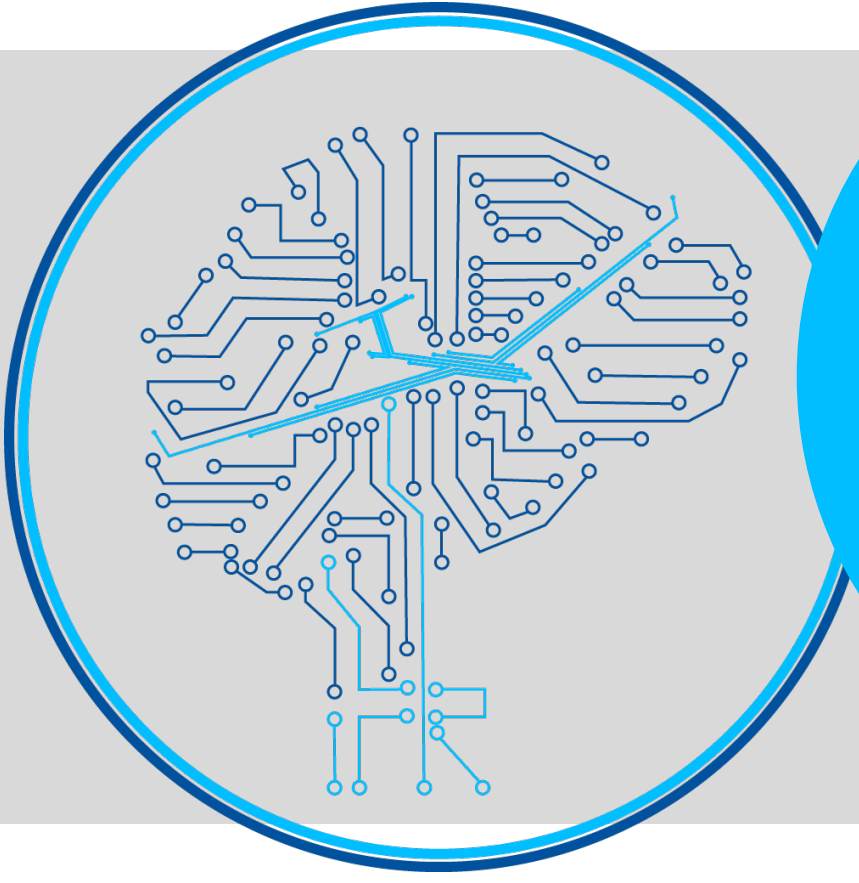
University of Stuttgart
Flight Mechanics and Controls Lab



Model Selection & Ensembles

Fabian Schimpf

IFR

Housekeeping

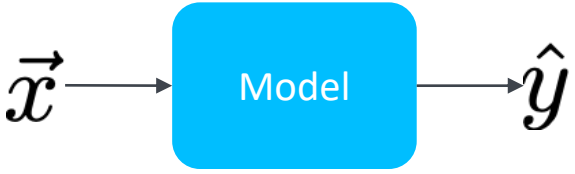
- Exam dates have been shared by e-mail and will be
  - March 29th and
  - April 26th

  →Please check out the e-mails and messages in ilias for details
- We received feedback from the survey in the forum and per e-mail that the course work is more time consuming than other comparable lectures. We have therefore decided to use next week's lecture slot to answer questions and to give you time to catch up with the coursework and exercises so far.
- We would appreciate it if you could participate in the official lecture online survey shared in ilias.

Recap

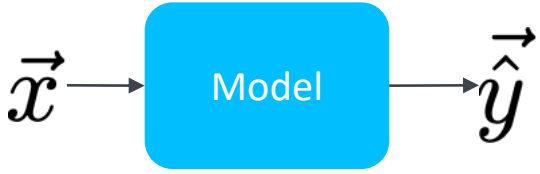
Types of supervised learning

REGRESSION



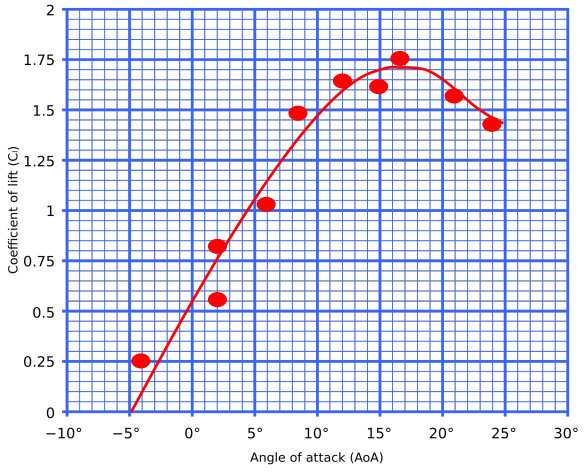
- Scalar, continuous output
- Example:
  - Given an airfoil predict the lift from angle of attack and air speed
  - Given the number of rooms, the area of the house and a postal code predict the price of the property (last one might also need encoding!)

CLASSIFICATION



- As many outputs as there are classes
- Pick class for prediction that the model assigns the highest value to
- Usually requires encoding of classes as a vector
- Example:
  - Given an image, does it show a cat [1,0] or a dog [0,1]

Intuitions



Consider the following problem:

- We want to find a function connecting an angle of attack to a lift coefficient $c_\alpha = f(\alpha)$
- There is a local maximum so we might want to consider a function with a quadratic term $c_\alpha = f(\alpha, \alpha^2)$
- We could also split the function into a linear part and a quadratic part and another linear part and tie them together with a continuous first derivative

Conclusions

- Different models can be used for a given problem
- Not everything that is mathematically possible makes sense (for example adding a sine above)
- We care about making predictions with a model which is fundamentally a hypothesis about the function which explains the observed data during training

Structure of this lecture

There is no unique model for a given prediction task.
Therefore, one has to navigate the multitude of potential models.

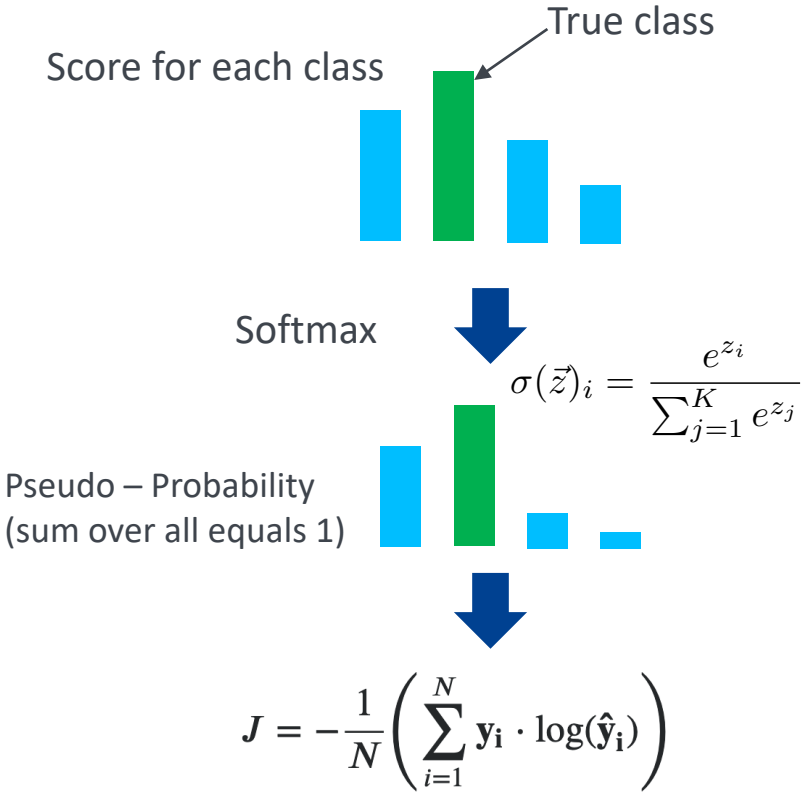
- **Loss functions**
  - What does it mean if a model fits data “well”
- **Model selection**
  - How to choose the “best” model for the job
- **Combination of models**
  - How to combine different models for even greater performance

Loss functions

Loss functions for classification

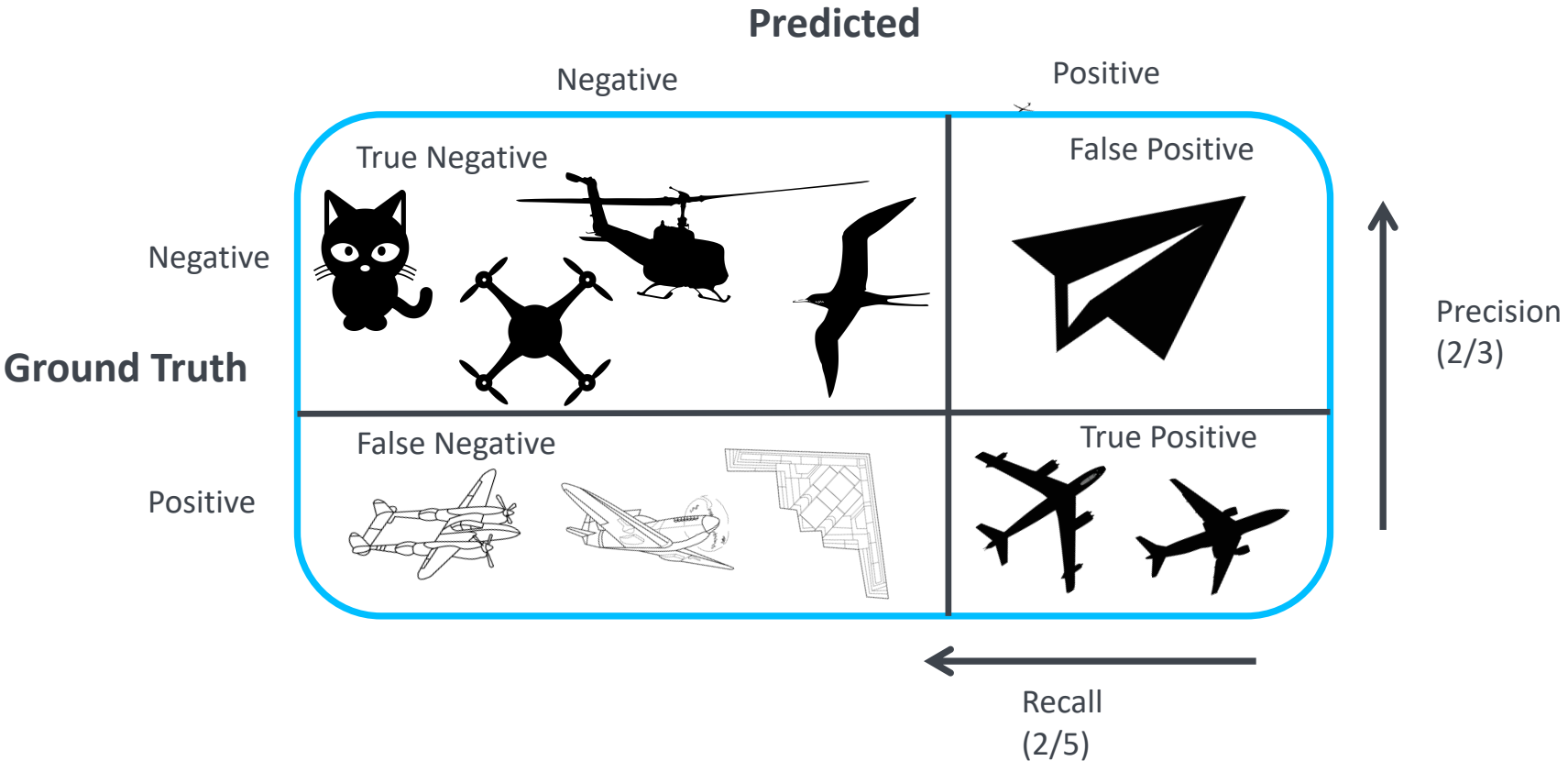
- Choice of function depends on problem at hand
  - Goodhart's law
- Performance indicators binary classification
  - Precision
  - Recall
  - Accuracy
- Loss functions for (multi class) classification
  - Cross Entropy (log loss on pseudo probabilities)
  - …

Cross entropy loss

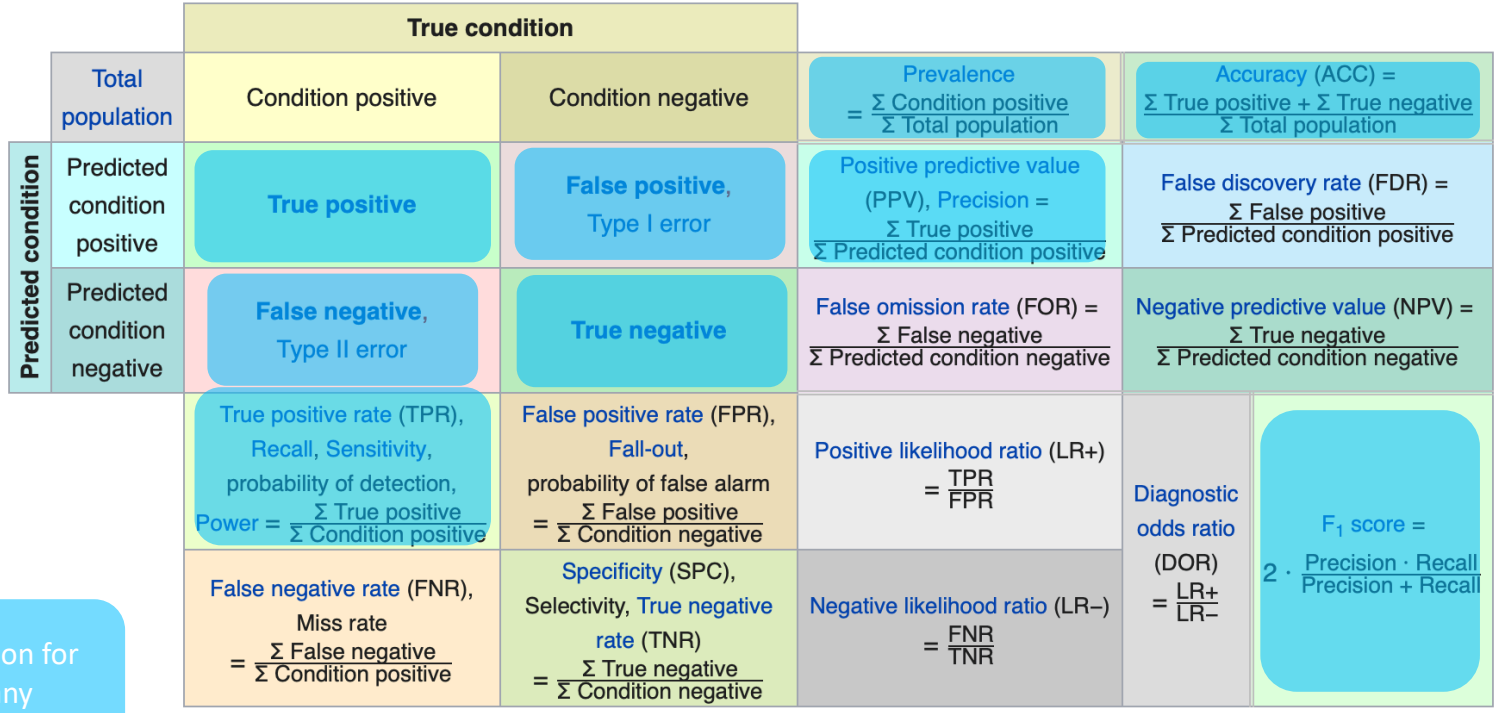


Confusion Matrix

Classifier for planes in an image



Confusion Matrix



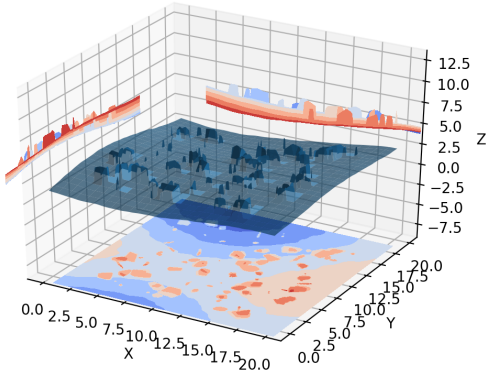
Common for many applications

Image credit: https://en.wikipedia.org/wiki/Confusion_matrix /

Example

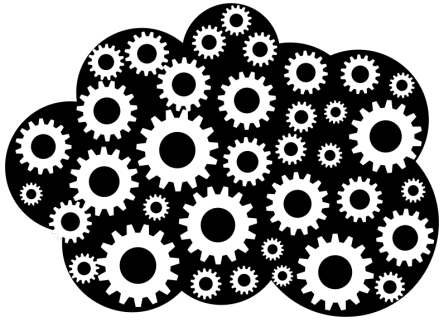
Parameter tuning for landing site selection

Input



Grid map of environment

Model with parameters



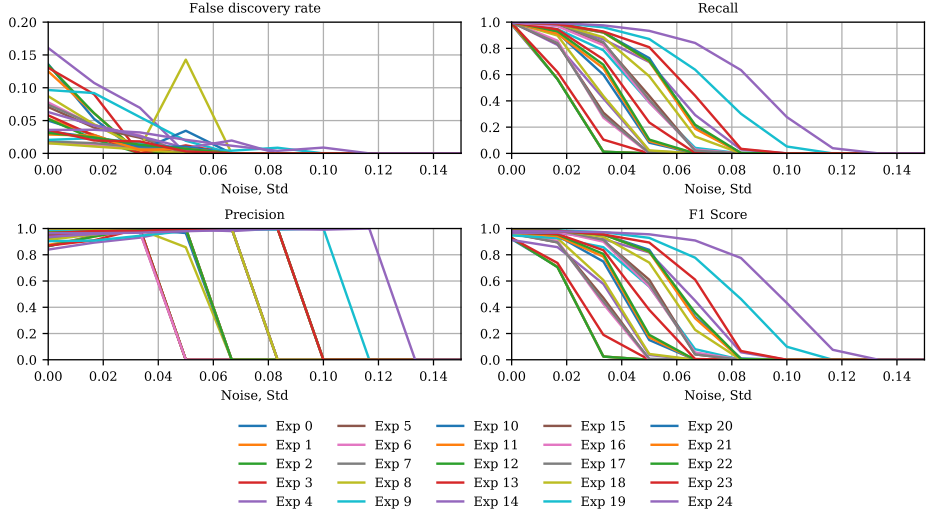
Prediction

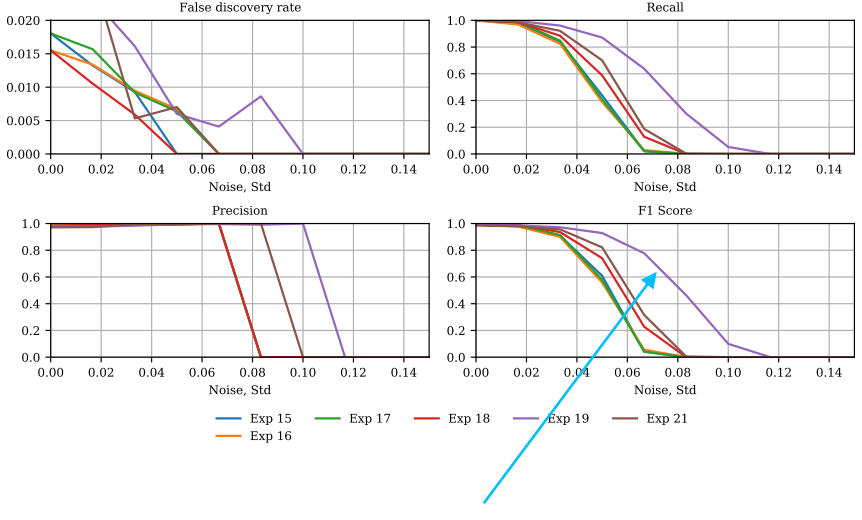
Safe / not Safe

Example

Parameter tuning for landing site selection

Select all parameter configurations
with a maximum false discovery rate of 2%



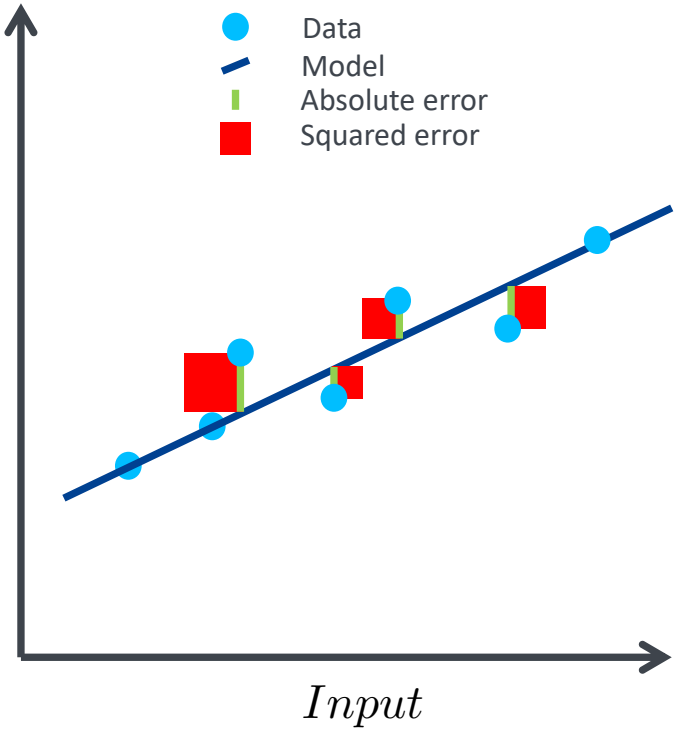


Choose parameter combination
with best F1 score

Loss functions for regression

- Common loss functions are:
  - Mean absolute error MAE $MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}|$
  - Mean squared error MSE $MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y})^2$
  - R squared $R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y})^2}{\sum_{i=1}^{N}(y_i - \overline{y})^2}$

Other loss functions are Root mean square error RMSE, Huber loss, Max error



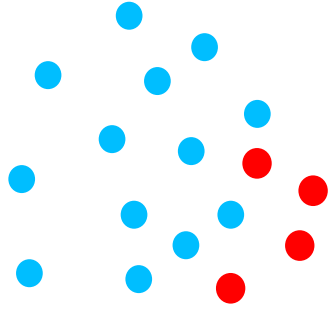
Imbalanced data

What does this mean?

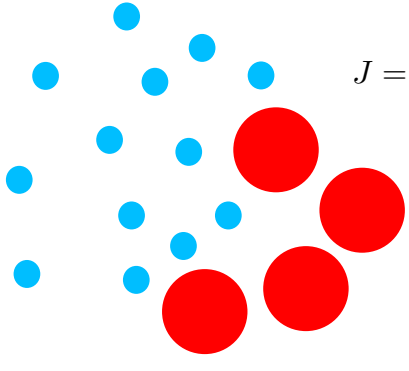
- Classes are not equally represented in a set, the data is skewed
- Related to distribution shift: the distribution of the data changes between training and test / production

Solution strategies

- Add weights according to prevalence of a certain class
- Resample neglected class until all classes are equally represented in loss function



$$J = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot log(\hat{y}_i)$$



$$J = -\frac{1}{N}\sum_{i=1}^{N} w(y_i) \cdot y_i \cdot log(\hat{y}_i)$$

Regularization

- Add loss term penalizing the weight of parameters
  - L1 Norm (Lasso) $J(\Theta) = MSE(\Theta) + \alpha \frac{1}{2} \sum_{i=1}^{N} |\Theta_i|$
  - L2 Norm (Ridge Regression): $J(\Theta) = MSE(\Theta) + \alpha \frac{1}{2} \sum_{i=1}^{N} \Theta_i^2$
  - L1 + L2 Norm (Elastic Net): $J(\Theta) = MSE(\Theta) + r\ \alpha \frac{1}{2} \sum_{i=1}^{N} |\Theta_i| + \frac{1-r}{2} \alpha \frac{1}{2} \sum_{i=1}^{N} \Theta_i^2$

Note, that these additional terms should only be applied during training to get a comparable loss with other models! It is not uncommon to train a model with a loss function that is not used in the end to evaluate its performance but that has nice properties for optimization.

Make sure to scale data before using the techniques above as the scale of the inputs will impact the scale of the penalty!

Model selection

Assumptions

The goal is to select a model that will optimally fit future examples

- Stationarity

$$P(E_j) = P(E_{j+1}) = P(E_{j+2}) = \cdots$$

  - The future is like the past
- Independence from previous samples

$$P(E_j) = P(E_j | E_{j-1}, E_{j-2}, \ldots)$$

Independet and identically distributed
i.i.d.

Over- and underfitting

Overfitting:

Poor performance on unseen data (Model 2)

Underfitting:

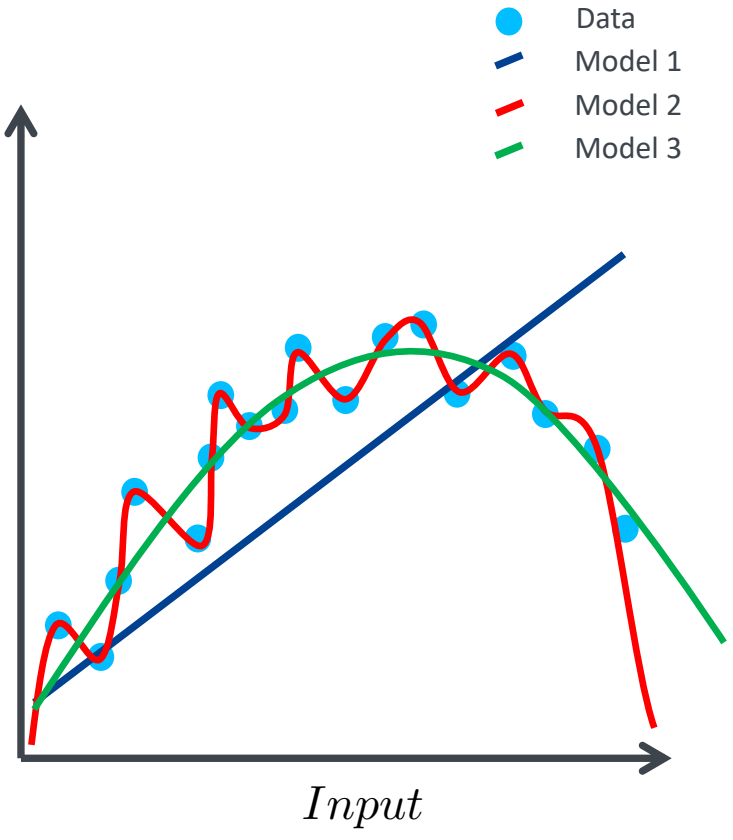
Poor performance on seen data (Model 1)

Generalization:

Doing well on unseen data (potentially Model 3)

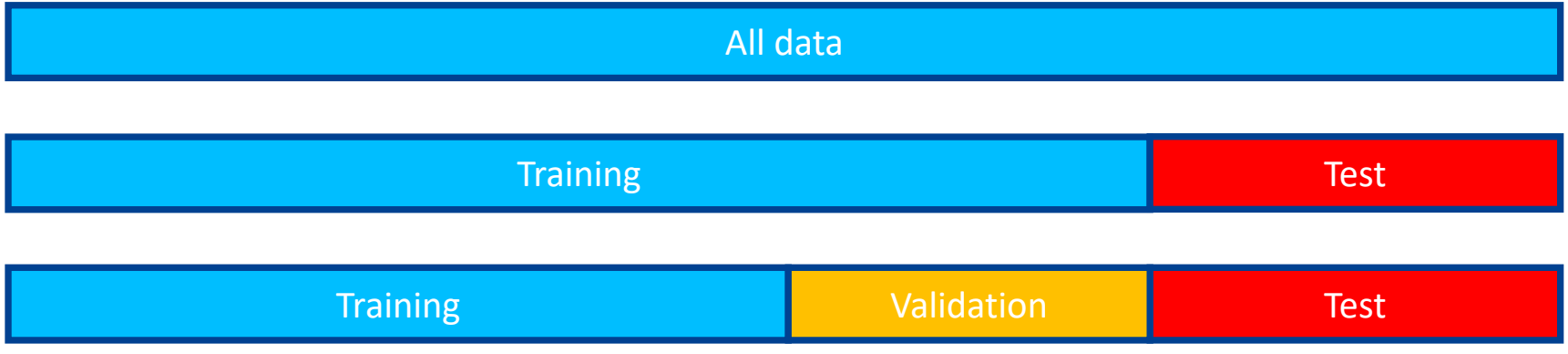
Ockham's Razor:

Plurality should not be posited without necessity

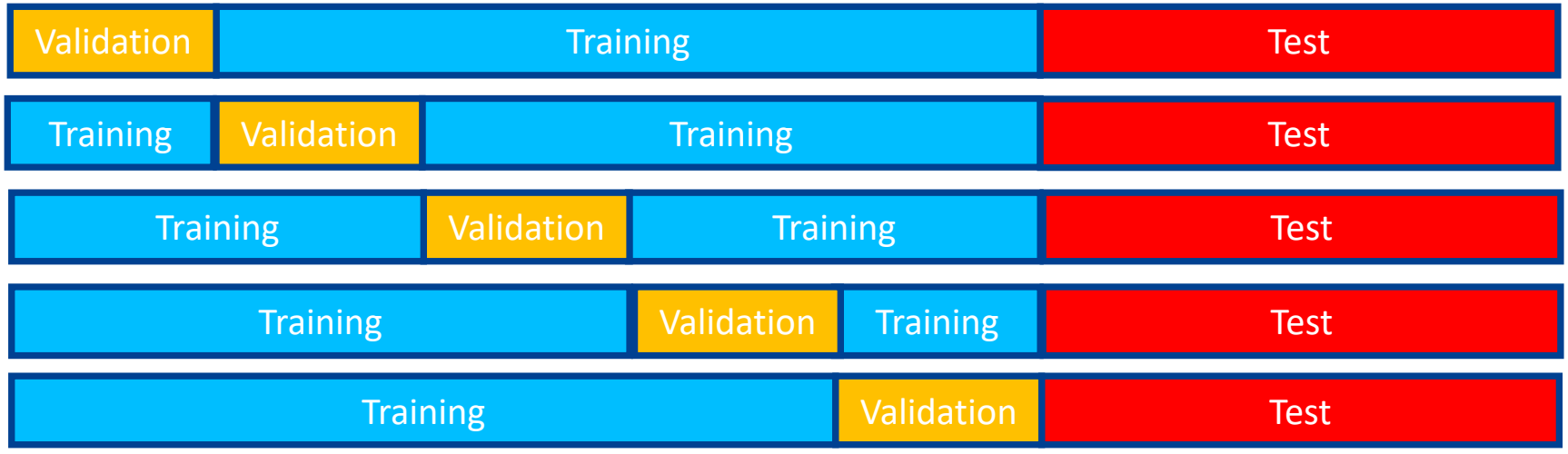


Data splitting

- To quantify how well models work with unknown data, the available data is split
  - One model: Training / Test ~ 80/20 split
    - Fails if multiple models are tested on the test set – overfitting to test set!
  - Multiple models: Training / Validation / Test (only used for final evaluation)
    - Problem: The amount of data available for training shrinks!



Cross-Validation



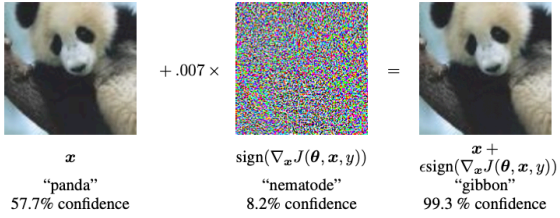
- Instead of having a fixed validation set the data is split into k (5 or 10 are often used) subsets and where k-1 are used for training and the left out one is used for validation, this is called k-fold cross validation.
- The test set is still needed
- Trade time for more reliable performance score

Model selection

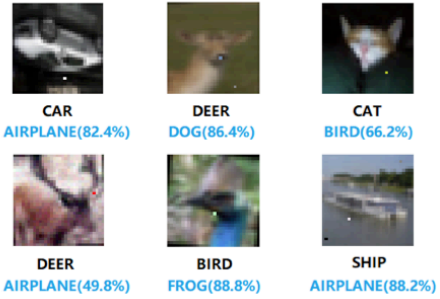
Criteria:

- Performance wrt. loss function
- Updates possible with new data (online learning)?
- Interpretable results – usually a downside of neural networks
- Runtime / training time
- Robustness to noise / adversarial attacks
- Memory consumption – neural networks and random forests can take up gigabytes of memory

→ **Very problem- and hardware specific**



Goodfellow et al., Explaining and Harnessing Adversarial Examples



Su et al., One Pixel Attack for Fooling Deep Neural Networks

Hyperparameter Search

- Hand tune
- Grid search
  - Test permutations of different choices for each parameter
- Coordinate descent
  - Pick a parameter, optimize it with respect to this parameter, repeat
- Random search
- Population based strategies
- Bayesian optimization
  - Used if evaluation is expensive / takes long
  - Build surrogate model with mean and standard deviation
  - Use acquisition function to select next sampling location (Thompson sampling, expected improvement, prob. improvement)

Picking the right parameters to tune is crucial!

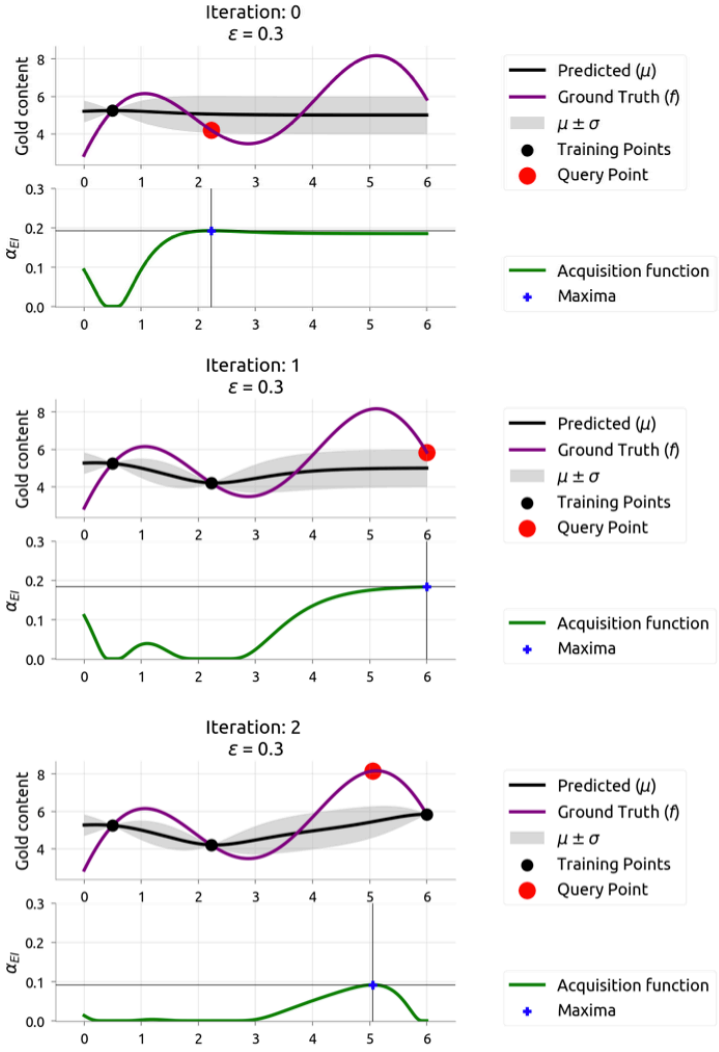
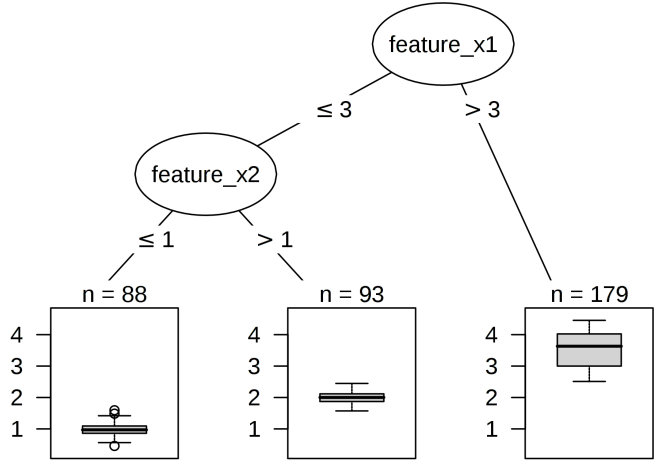


Image credit: https://distill.pub/2020/bayesian-optimization/

Combination of models

Decision Trees

- Execution:
  - Start at root node at the top and for each sample follow the true condition
  - At leaf node return aggregate of samples in this i.e. most prevalent class, mean for regression,
- Benefits:
  - Don’t require any feature engineering like scaling and centering
  - The decision is clearly understandable for humans
  - Impurity at leaf node can be analyzed
- Downsides:
  - Not all data can be nicely split along one variable
  - Does not scale to large input spaces



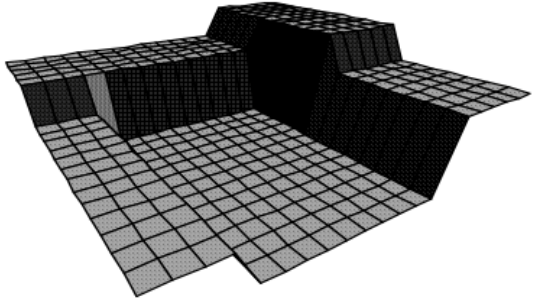


Image credit: Seni & Elder, KDD 2007

Decision Trees

CART

- At each node the trainings data is separated into two subsets
  - Using a feature k
  - At the threshold t_k
- The objective function is to minimize the impurity in each subset weighted by its size
- Measures for impurity are
  - Regression: MSE
  - Classification: Gini Coefficient, Entropy
- There are alternatives to CART that can produce more than 1 split per node like ID3

Classification

$$J(k, t_k) = \frac{m_{left}}{m} G_{left} + \frac{m_{right}}{m} G_{right}$$

$$G = 1 - \sum_{i=1}^{n} p_i^2$$

Regression

$$J(k, t_k) = \frac{m_{left}}{m} MSE_{left} + \frac{m_{right}}{m} MSE_{right}$$

$$MSE_{node} = \sum_{i \in node} (\overline{y}_{node} - y_i)^2$$

Ensembles

- Idea: Wisdom of the crowd
- Ensembles are the combination of different base models:
  - Hard voting for classifiers (5 vote cat, 3 dog, 1 airplane)
  - Averaging for regression or class probabilities
  - Stacking (Combination with separate model, more on this later)

Comments:

- The ensemble often performs better than the strongest base model it contains
- Therefore, ensembling is usually used in competions like the Netflix Price
- Ensembling works best if the errors from the submodels are uncorrelated, therefore it is usually beneficial to use a diverse set of models for ensembles

"Our final solution (RMSE=0.8712) consists of blending 107 individual results. "

--	No Progress Prize candidates yet	--	--
Progress Prize - RMSE <= 0.8625			
1	BellKor	0.8705	8.50
Progress Prize 2007 - RMSE = 0.8712 - Winning Team: KorBell			
2	KorBell	0.8712	8.43
3	When Gravity and Dinosaurs Unite	0.8717	8.38
4	Gravity	0.8743	8.10
5	basho	0.8746	8.07
6	Dinosaur Planet	0.8753	8.00
7	ML@UToronto A	0.8787	7.64
8	Arek Paterek	0.8789	7.62
9	NIPS Reject	0.8808	7.42
10	Just a guy in a garage	0.8834	7.15
11	Ensemble Experts	0.8841	7.07
12	mathematical capital	0.8844	7.04
13	HowLowCanHeGo2	0.8847	7.01
14	The Thought Gang	0.8849	6.99
15	Reel Ingenuity	0.8855	6.93
16	strudeltamale	0.8859	6.88
17	NIPS Submission	0.8861	6.86
18	Three Blind Mice	0.8869	6.78
19	TrainOnTest	0.8869	6.78
20	Geoff Dean	0.8869	6.78
21	Rookies	0.8872	6.75
22	Paul Harrison	0.8872	6.75
23	ATTEAM	0.8873	6.74
24	wxyzconsulting.com	0.8874	6.73
25	ICMLsubmission	0.8875	6.72
26	Efratko	0.8877	6.70
27	Kitty	0.8881	6.65
28	SecondaryResults	0.8884	6.62
29	Birgit Kraft	0.8885	6.61

Bagging / Pasting

- Instead of training a diverse array of models it is also possible to train the same model with different subsets from the training data!
  - Bagging: Sampling with replacement (a model can draw the same sample multiple times)
  - Pasting: Sampling without replacement
- Scales very well as data might not be in the same place, predictions can be made in parallel
- Reduces lower variance than a model that is trained on the whole data set

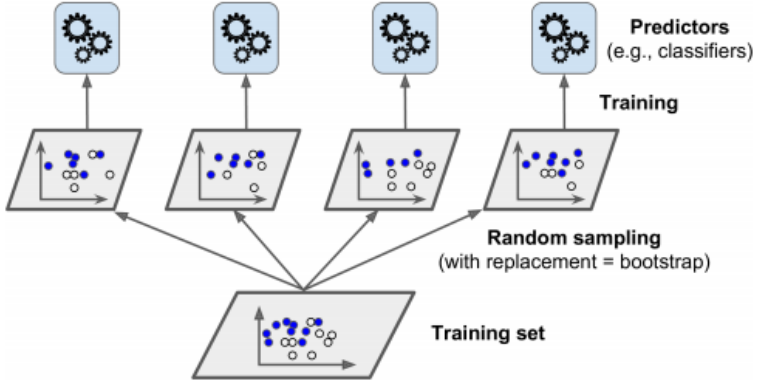
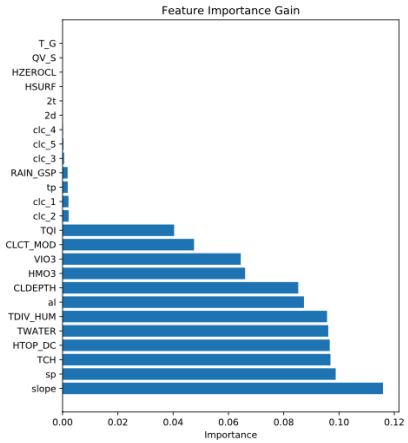


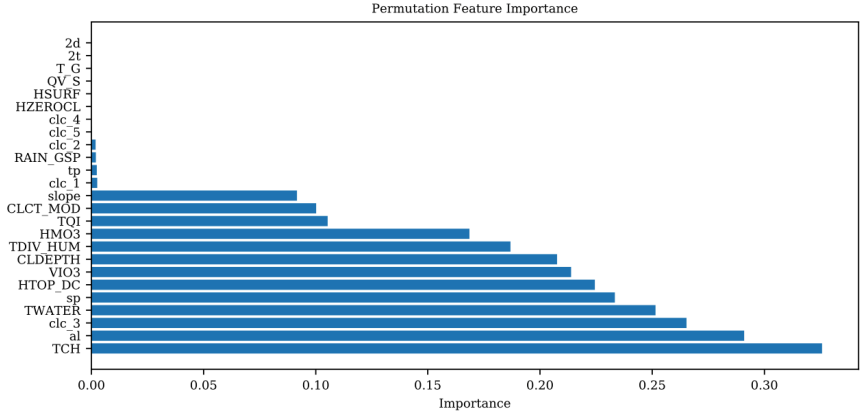
Image credit: Aurélien Géron – Hands on Machine Learning with Scikit-Learn & Tensorflow

Random Forest

- Another way to get different models from the same decision tree algorithm is to calculate the split from a random subset of features that can be checked at each node
- Extra (EXTremely RAndomized)Tree: Random threshold at each split
- Feature Importance
  - Prevalence of features in decision / reduction in impurity because of a given feature
  - Scores can be misleading if categorical (especially binary) and continuous features are present in the data
  - In these cases it is possible to use permutation importance



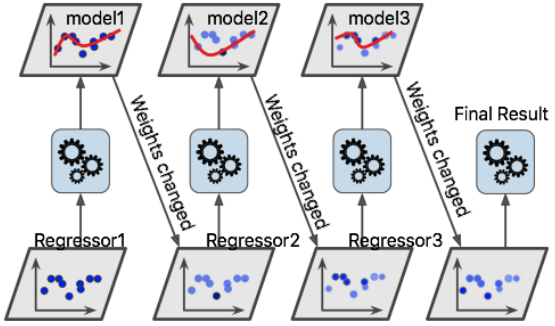
- clc_3 is a binary feature indicating being over forests, fields or gras
- As RFs (like all DTs) need to store all training data they get quite memory consuming for large data sets



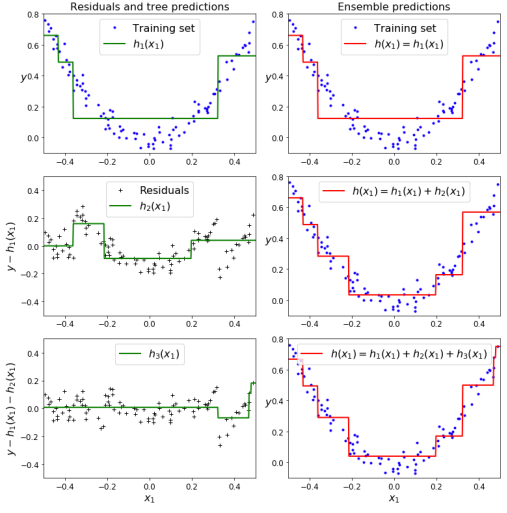
Boosting

- Base models for ensembles so far were trained independently. Boosting adds the capacity to learn from all preceding models
- Adaboost (binary classification)
  - Adapt weights of samples based on the systems performance
- Gradient boosting
  - Every new model fits the residuum of the combination of prior models
- Problem:
  - The training of the ensemble has to happen sequentially which does not scale as well as bagging or pasting based methods
  - These methods need tuning to avoid overfitting

Adaboost

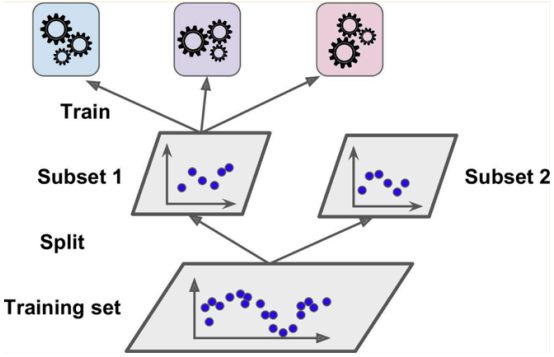


Gradient Boosting



Stacking

- Hierarchical structuring of the meta-model:
  - Base models make predictions
  - A separate model takes these predictions (and features) for a final output
  - Instead of voting, this approach uses a model to combine multiple prediction
- The training data is usually split into the number of “layers” that the combined model has and each layer is sequentially trained on one subset



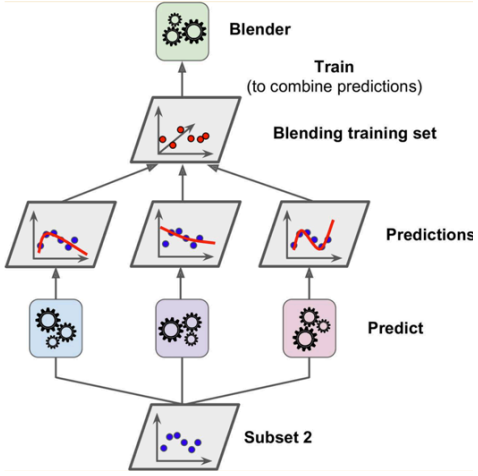


Image credit: Aurélien Géron – Hands on Machine Learning with Scikit-Learn & Tensorflow

Comments

Comments

- Model selection is not a one off thing
  - Companies deploy new models every hour
  - For ML in production (after deployment) a monitoring system is needed
- For commercial applications an additional selection criteria is the license under which code is available
- Early stopping is often a good idea – The best model is the model with the best validation loss, stop when it increases
- Tree based models can achieve SOTA performance on many tabular datasets but cannot easily incorporate new data
- Model selection (or optimization) is not going to save the day if the problem is not appropriate or there are bugs in the pipeline
- Reduce complexity by reducing the dimensionality of the problem also acts as regularization → feature selection

Further Reading

Books

- Aurélien Géron – Hands on Machine Learning with Scikit-Learn & Tensorflow
- Russell, Norvig – Artificial Intelligence a Modern Approach, 4th Edition
- Bishop – Pattern Recognition and Machine Learning

Thank you!

Fabian Schimpf

e-mail Fabian.schimpf@ifr.uni-stuttgart.de
phone +49 (0) 711 685-
fax +49 (0) 711 685-

University of Stuttgart
Flight Mechanics and Controls Lab
Pfaffenwaldring 27, 70569 Stuttgart