

**NOTE: No guarantee for the correctness although I gave my best 😊. Where there were calculations but no values I tried to insert the slides where the calculation was shown/done. Have fun and good luck during the exam! Here you can find all the exam questions from VOWI exams posted from 2017 till 2024. The date when this document was lastly updated is: 24.06.2024.**

Is Gradient Boosting 0 Rule? – **True, first step is building zero rule model**

What is Data Augmentation? - **Transformation of data so the algorithm does not learn unnecessary features such as scaling, facing, rotation, color, contrast. The objective is for the algorithm to ignore such features for example in Image Classification . Through data augmentation it is also possible to create larger dataset.**

Adaboost weights initialization uniform – **True, initialised uniformly**

Calculations for k armed bandits :

.....

- Estimating the values of actions

$$Q_t(a) \doteq \frac{\text{sum of rewards when } a \text{ taken prior to } t}{\text{number of times } a \text{ taken prior to } t} = \frac{\sum_{i=1}^{t-1} R_i \cdot \mathbb{1}_{A_i=a}}{\sum_{i=1}^{t-1} \mathbb{1}_{A_i=a}},$$

- If the denominator is zero,  $Q_t(a)$  takes a default value (e.g. 0)
- Sample-average method
- Greedy action selection

$$A_t \doteq \arg \max_a Q_t(a)$$

- Concentrate on a single action:  $R_i$  is the reward received after the  $i^{\text{th}}$  selection of this action
- $Q_n$  denote the estimate of its action value after it has been selected  $n - 1$  times,

$$Q_n \doteq \frac{R_1 + R_2 + \dots + R_{n-1}}{n - 1}$$

$$Q_{n+1} = \frac{1}{n} \sum_{i=1}^n R_i$$

$$= \frac{1}{n} \left( R_n + \sum_{i=1}^{n-1} R_i \right)$$

$$= \frac{1}{n} \left( R_n + (n - 1) \frac{1}{n - 1} \sum_{i=1}^{n-1} R_i \right)$$

$$= \frac{1}{n} \left( R_n + (n - 1) Q_n \right)$$

$$= \frac{1}{n} \left( R_n + n Q_n - Q_n \right)$$

$$= Q_n + \frac{1}{n} [R_n - Q_n],$$

$$\text{NewEstimate} \leftarrow \text{OldEstimate} + \text{StepSize} \left[ \text{Target} - \text{OldEstimate} \right] \dots$$

A decision tree can be converted into a rule set. – **True, although complex computation for complex rule set**

k-armed bandits choose the next action based on the expected future reward. – **True, maximise the expected reward over some period of time**

A bayesian network is a directed cyclic graph. – **False, directed graph but without cycles**

Which of these methods helps to prevent overfitting?

- Regularization – **True, penalise models with many coefficients**
- Dropout – **True, drop some number of nodes 20% - 50% so the model can also correctly classify instances where not all features are present**
- Batch Normalization – **True, also works as regularizer**
- Cross Validation – **False, only to compute performance metrics**

What can a validation set be used for?

- Early stopping – **True, if we want to determine the stopping criteria**
- Hyperparameter tuning - **True**
- Significance Testing – **True, as part of cross-validation for example**

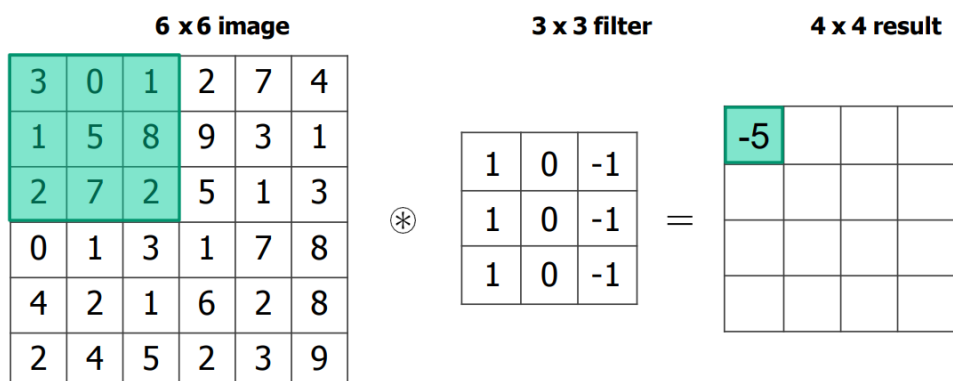
Which technique is similar to dropout in neural networks

- Bagging - **False**
- Boosting - **False**

Calculation of output size of convolution. - Convolutional layer: performs 2D convolution of 2D input with multiple learned 2D kernels:



## What is a Convolution?



$$\begin{aligned} & 3 \times 1 + 0 \times 0 + 1 \times -1 + \\ & 1 \times 1 + 5 \times 0 + 8 \times -1 + \\ & 2 \times 1 + 7 \times 0 + 2 \times -1 = \end{aligned}$$

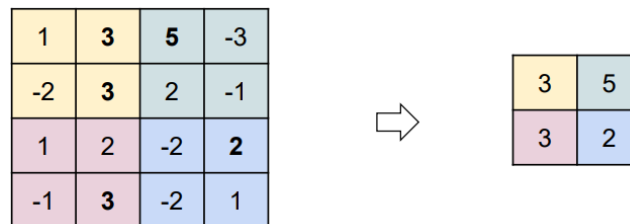
-5

Calculation of max pooling operation.



## Pooling Step

- Second very important aspect of a CNN
  - (also called subsampling or downsampling)
  - A **pooling layer** reduces the size of feature maps (i.e. output of a CNN layer and thus the input to the next layer)



**Max pooling:** take the max. activation across small regions (e.g. 2x2, as in the example above)

### Single Choice

Classification is a machine learning task where the target attribute is nominal - **True**

Decision trees can handle only binary classification problems - **False**

The error of a 1-NN classifier on the training set is 0 – **True, if you also compute the distance to itself**

A softmax function in MLPs transforms the activation to a range of  $-1 \dots 1$  – **False,  $0 \dots 1$**

Macro-averaging for classifier evaluation first calculates accuracy/precision/recall/... per class, before averaging across classes – **True**

The paired t-test is used when testing for statistical significance of results obtained with holdout validation - **True**

The paired t-test is used when testing for statistical significance of results obtained with cross validation - **True**

In a dataset the entropy is lowest when all classes have the same amount of samples – **False, does not depend on the amount of samples, much more whether all samples are of the same class in dataset**

In a dataset the entropy is highest when all classes have the same amount of samples - **False, does not depend on the amount of samples, much more whether all samples are of the same class in dataset**

In AdaBoost, the weights are randomly initialised – **False, Initially equal weights  $1/n$**

Support Vector Machines always finds a more optimal decision boundary (hyperplane) than Perceptrons – **False, not necessarily it aims on finding the more robust one by maximising the margin**

Support Vector Machines with a linear kernel are particularly suitable for classification of very high dimensional, sparse data – **True**

Support Vector Machines can by default only solve binary classification problems - **False**

If Naive Bayes is applied on a data set that contains also numeric attributes then a probability density function must always be used - **True**

Model based features used for metalearning are extracted directly from the data set - **False**

Majority voting is not used when k-nn is applied for linear regression - **True, average used instead**

For the Monte Carlo method in reinforcement learning value estimates and policies are changed only on the completion of an episode - **True**

Gradient descent is always more efficient than Normal Equation (analytical approach) for linear regression – **False, only if large number of dimensions**

Information gain is an unsupervised feature selection method – **False, supervised**

Feature selection is primarily useful to improve the effectiveness of machine learning – **False, efficiency so we reduce complexity, it can however lead to better prediction results**

Ordinal data does not allow distances to be computed between data points – **True**

The first model in gradient boosting is a zero rule model - **True**

PCA is a supervised feature selection method - **False**

**Open Choice** Can kernel methods also be applied to the perceptron classifier (also discuss why or why not!) – **2D Kernel Matrix can be applied to MLP, using kernel it is possible to transform data to higher dimensions and thus make it linearly separable**

What is polynomial regression? Which are advantages/disadvantages of polynomial regression compared to linear regression? – **Polynomial regression is regression using high order polynomials to compute regression values. Advantages – can approximate the data better than linear regression and is more flexible, disadvantages can overfit the data and introduce multicorrelality, where the predictor variable becomes highly correlated. Also bad for interpretability**

When are two nodes in a Bayesian Network d-separated? - **A set of variables E d-separates variables X and Y if E blocks every un-directed path between X and Y in the network. To determine whether X and Y are independent given the observed variables E, we can verify whether E d-separates X and Y.**

Which features are used in metalearning? What are landmarking features? - **Statistical and Information-theoretic features, Model-based features, Landmarking features, Learning Curve features.**

**Statistical and Information-theoretic: Number of attributes – Number of classes – Ratio of examples to attributes – Average class entropy – Degree of correlation between features and target concept.**

**Model based features - Properties of hypothesis induced on a particular problem are used as an indirect form of characterization, for example number of nodes, maximum depth, tree imbalance for Decision Tree.**

**Landmarking features - a learning mechanism whose performance is used to describe a task, because each learner has a set of tasks on which it performs very well and performance of an algorithm on tasks says sth. About the nature of a task.**

- Suppose that  $i_1, i_2, i_3$  are taken as landmarks
- One possible conclusion
  - Problems on which  $i_1$  and  $i_3$  perform well, and  $i_2$  poor are likely to be in the expertise of  $i_4$
- Landmarking is simple: learners are used to signpost learners
- We want landmarks to be efficient
- The cost of running landmarks should be smaller than the cost of running all learners in  $A \rightarrow$  otherwise we could run all learners and find which is the best
- Use landmarks that are more efficient  $\rightarrow$  e.g. use naive algorithms (naive bayes, OneR, ...)
- Landmarking has given good results

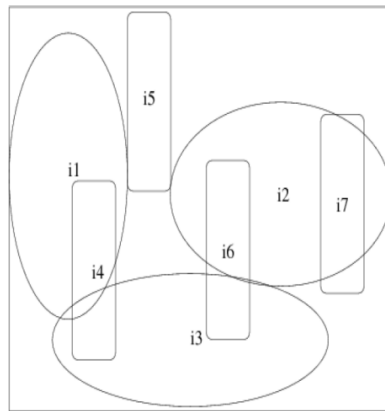


Figure taken from metalearning tutorial given to the end of these slides

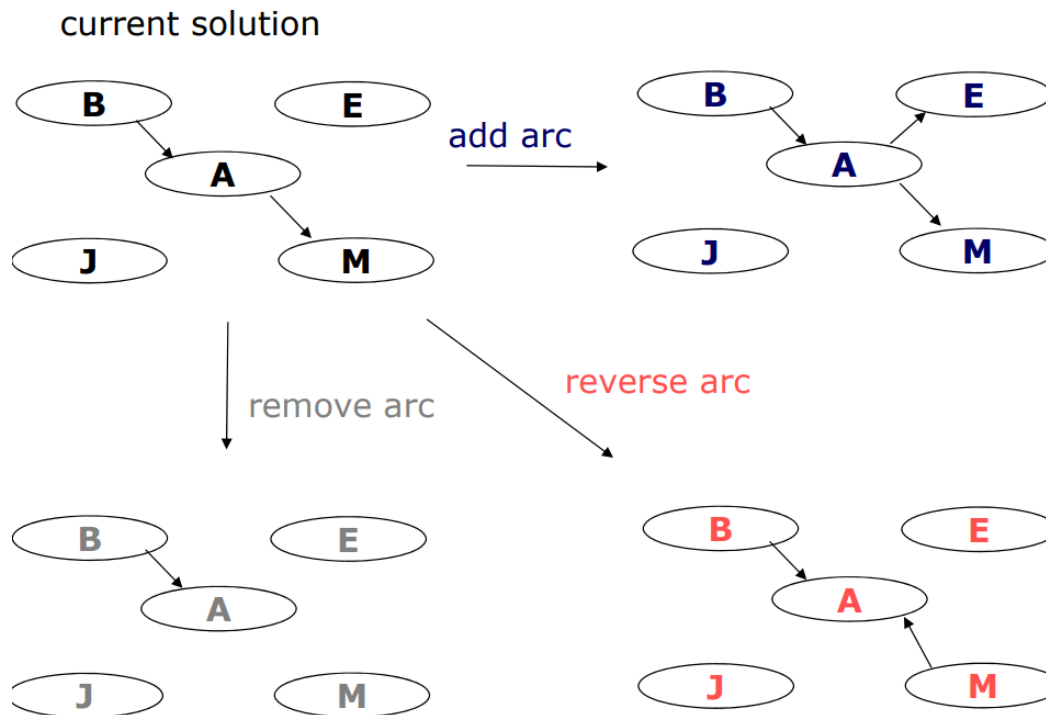
What are the difference between micro and macro averaged performance measures? - **Micro averaged – computing precision/recall etc. over all classes. Macro averaged – computing the performance measure per class and than average the class results.**

Given are 1000 observations, from which you want to train a decision tree. As pre-pruning the following parameters are set: - The minimum number of observations required to split a node is set to 200 - The minimum leaf size (number of observations) to 300 Then, what would be the maximum depth a decision tree can take (not counting the root node)? Explain you answer! – **Imagine a very unbalanced tree, on the left side from the root split we end up with a node including 300 observation and the node is pure so we stop, on the right side the node is not pure and we continue with remaining 700 observations. We repeat that and we end up with 400 observations and current length of 2 (not counting the root node). Because we cannot split any further so that each node has minimum 300 observations the maximal depth is 2.**

Describe a local search algorithm for Bayesian Network creation:

## Local search for Bayesian Networks

1. Construct an initial network
2. Calculate the score of the current BN network (with learned probabilities)
3. Generate the neighborhood by modifying the current network
4. Select one of networks in the neighborhood as a new current network for the next iteration
5. Go to Step 3 if termination criteria is not fulfilled



Goal and settings of classification. To what tasks does it relate and from which it differs in machine learning ? – **Classification is supervised learning task, it relates to regression task, but instead of predicting continuous variable we predict a categorical variable from the given classes. It differs from unsupervised (clustering) and reinforcement learning tasks.**

What methods are there for combatting overfitting in Neural Networks? – **Dropout, Data Augmentation, Regularization (add a penalty term to the loss function that penalizes the model for having large weights. This encourages the model to learn smaller weights, leading to a simpler and less overfitted model), Batch Normalisation (can also be seen as a regularizer), smaller model architectures**

Describe 3 methods to compute the error in regression. (approximate question) – **Mean Squared error ( $\sum (y_i - \hat{y}_i)^2 / n$ ), mean absolute error ( $\sum |y_i - \hat{y}_i| / n$ ), root mean squared error ( $\sqrt{\sum (y_i - \hat{y}_i)^2 / n}$ ), R-Squared - This metric represents the proportion of variance in the dependent variable (y) that can be explained by the independent variable(s) (x)**

Something about regularisations z-score and min-max. What are they, when they are useful and on which type of features. – **Useful for numerical features with different value ranges between the features if distance metrics between instances are computed, so one feature does not dominate the computation and thus does not dominate the model prediction. Min max normalisation =  $(x_i - \min(x)) / (\max(x) - \min(x))$  so the result is between 0 and 1. Z-score =  $(x - \mu) / \sigma$ , where generally  $\mu$  is 0 and  $\sigma$  (standard deviation) is 1.**

Something about explaining the epsilon-greedy selection for the k-armed Bandit Problem. – **Selection of greedy action is exploitation. Exploitation and exploration (selection of a non greedy action) should be balanced. With probability  $1 - \epsilon$  select one of the actions with the highest estimated value**

True/False:

1. In AdaBoost weights are uniformly initialized: **True, because they are initialised with 1/n**
2. Categorical data should be normalized before training a k-NN: **True**
3. The error of a 1-NN classifier on the training set is 0: **True**
4. One-vs-all is an approach to solve multi-class problems for DTs: **False, if we want to split categorical variable**
5. Boosting ensembles can be easily parallelized – **False, because build on top of each other, bagging can be paralised**
6. 1-hot encoding is used to transform numerical into categorical attributes: - **False, the other way around**
7. The Pearson coefficient has a value range from -1 to 1: **True, measure linear correlation between two variables**
8. Off-the-shelf is a transfer learning technique that uses the output of layers from a deep-learning architecture as input for a shallow model - **False, it is not the output but the features/feature detection/layres are used**
9. SVMs search for a decision boundary with the maximum margin - **True**
10. SVMs always find a more optimal decision boundary (hyperplane) than Perceptrons – **False, they aim to find more robust one but not necessarily find a better one**
11. In Bayesian Networks we assume that attributes are statistically independent given the class – **False, we assume it only in Naïve Bayes**
12. Majority voting is not used when k-NN is applied for linear regression - **True**
13. Chain Rule does not simplify calculation of probabilities for BNs - **False**
14. Naive Bayes is a lazy learner – **False, eager learner, during the training it estimates the probabilities**
15. Normal Equation (analytical approach) is always more efficient than gradient descent for linear regression – **False, works slow if there are many features/dimension**
16. knn is based on supervised paradigm - **True**
17. knn is based on unsupervised paradigm – **False, unsupervised alternative to KNN is K based Clustering, where we define number of clusters**
18. one vs all is approach used by Naive Bayes – **False, used in Decision Trees**

Free Text:

Compare Perceptron with SVM algorithms. Common characteristics and differences. – **Both are for binary classification, both supervised and try to separate the data linearly. Perceptron just tries to find the split, SVM tries to find the optimal (most robust) split. Difference – perceptron (normally) does not use kernel functions, but SVM does. Perceptron separates the data points with minimal error, SVM tries to maximise the margin between support vectors. Perceptron simpler model, easier to interpret, SVM difficult to interpret. Perceptron is faster in training due to simpler updates, SVM slower due to optimisation.**

How can you learn the structure of a Bayesian Network – **Possibilities for building a network are human experts, learning from data and combination of both approaches. Human experts are better on finding the structure, computers better at calculating probabilities. Objective of finding a structure is finding a network which is a good fit to data and has low complexity. This is a search problem and is solved by local search or other searches**

such as hill climbing, simulated annealing or tabu search. Generate neighborhood solutions by applying neighborhood relations.

Explain how to deal with missing values and zero frequency problem in NB – **Zero frequency problem – if probability for will be zero, than a posteriori probability will also be zero. Fix - add 1 to every attribute value-class combination, thus probabilities will never be zero and it additionally stabilizes the probabilities computed from small samples of data. Missing values – instance is not include in the frequency count for value-class combination in training, continuous missing values during training are not included in calculation of mean and standard deviation. In classification - attribute will be omitted from calculation**

What is the difference between Lasso and Ridge regression? - **Both Lasso and Ridge regression add a penalty term to the cost function of linear regression. This penalty term discourages the model from having overly complex coefficients (weights associated with features). As the penalty term Lasso uses absolute value of the coefficients (L1 Norm), and Ridge uses Square of the coefficients (L2 Norm). Lasso can set some coefficients to exactly zero (sparsity), Ridge shrinks all coefficients towards zero, but rarely sets them to zero.**

Which data preparation/preprocessing steps are mentioned during the lecture? - **Encoding (Label encoding transform category to integer, 1-n encoding encode each value as a feature and set 1 or 0), missing data removal/imputation, standardisation, maybe definition of custom distance function, feature selection, binning train-validate-test split (without leakage from the test set).**

No free lunch theorem. Explain. - **...for any algorithm, any elevated performance over one class of problems is offset by performance over another class. Any two algorithms are equivalent when their performance is averaged across all possible problems. Meaning: we can not have (or still did not find) a single algorithm which performs the best on every problem**

Multiple Choice (Answer: True/False):

1. Is ensemble boosting easily parallelizable – **False, Bagging yes, Boosting no due to building on top of previous models**
2. Usually state of the art AutoML systems use grid search to find best hyperparameters – **False, because exhaustive search, can be outperformed by Randomized Search**
3. In AdaBoost, the weights are uniformly initialised - **True**

Free Text:

4. What is the Chain rule and how is it used in Bayesian Networks? - **The chain rule is a fundamental concept in probability theory that allows you to calculate the joint probability of a set of events from the conditional probabilities of each event given the previous ones. It essentially breaks down the joint probability of multiple variables into a product of conditional probabilities. By leveraging these conditional independence relationships, the chain rule formula can be simplified in Bayesian networks. You only need to consider the conditional probabilities between variables that are directly connected in the network. This significantly reduces the number of terms needed to calculate the joint probability**



compared to the general chain rule formula.

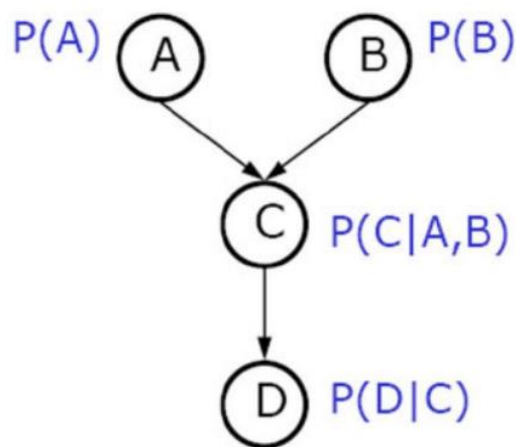
## Chain Rule (Example)

---

$$P(ABCD) = P(D|ABC) * P(ABC) =$$

$$P(D|C) * P(C|AB) * P(AB) =$$

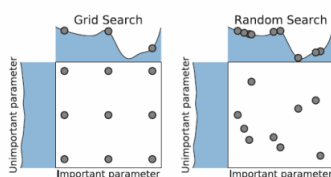
$$P(D|C) * P(C|AB) * P(A) * P(B)$$



5. What approaches can be chosen for linear regression? Describe them. – **Gradient Descent (iterative, good with many features, learning rate should be determined) or Normal Equations (no iterations, no learning rate, but slow if many features)**

8. Describe at least three methods for hyperparameter optimization. – **Grid Search, Random Search, Bayesian Optimisation.**

- Grid search
  - Exhaustive search
  - Continuous parameters have to be discretized
  - Cartesian product of sets (values of parameters)
  - Cross-validation is usually used for evaluation
- Randomized search
  - Random selection of configurations in the search space
  - Can outperform grid search
  - Specifying the distribution from which to sample



- General framework for minimizing blackbox functions  $f$
- A probabilistic model  $M$  is used to model  $f$  based
  - on point evaluation of  $f$
  - available prior information
- Applies  $M$  to select promising inputs to evaluate  $f$  next
- $M$  is a regression model fitted on data tuples  $(\lambda, L(A_\lambda))$
- Selection of next **hyperparameter** configuration  $\lambda$  :
  - Acquisition function:  $a_M: \Lambda \rightarrow \mathbb{R}$
  - Most useful configuration  $\lambda$  is evaluated next
  - Most popular acquisition function is *expected improvement* over the best previously-observed function value  $f_{\min}$
  - Probabilistic model  $M$  is used to predict a posterior distribution over function values  $p_M(f | \lambda)$

10. XOR dataset, which of perceptron, decision tree, SVM 1-NN can achieve 0 error rate? – **Not linear problem, so perceptron and SVM cannot make linear separation, so error rate would not be 0. SVM with non linear kernel can. 1 NN can theoretically classify perfectly. For decision trees we would need a very complex decision boundary. Depending on the splitting criteria used, the decision tree might not be able to find a perfect separation, because we would need to combine multiple splits.**

11. Describe in detail the random forest algorithm.

- **make n different bootstrap samples from the training data**
- **Create n decision trees based on these samples**
- **At each individual tree node only make a random subset of features available for splits**

⇒ **Accentuate difference between trees**

- **For Predictions: Let trees “vote” on the class of a given sample (majority wins)**

The coefficients  $w_0$ ,  $w_1$  and  $w_2$  had to be calculated when using the RSS as metric. Further, a learning rate was given ( $\alpha = 0.5$ ). When in the first step all  $w$  are 0, what will  $w_1$  be in the second step:

- GD problem; there were 2 features and 1 target (5 samples).

f1 f2 t

1 3 12

2 5 9 (these are example values - I dont remeber the exact values...)



## Gradient descent

Simple case:  $y = w_1 * x$

Initial value  $w_1 = 10$ ,  $\alpha = 4$

**Iteration 1:**

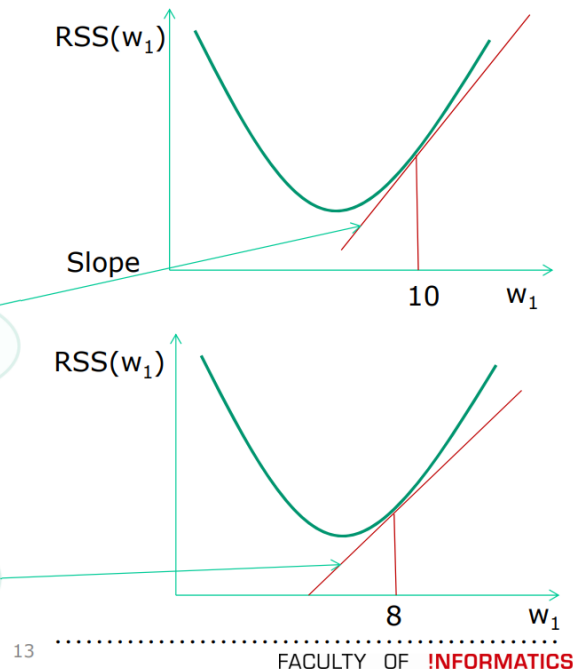
$$w_1 = 10 - 4 * \frac{\partial}{\partial w_1} RSS(w_1)$$

Suppose that this value is 0.5

**Iteration 2**

$$w_1 = 8 - 4 * \frac{\partial}{\partial w_1} RSS(w_1)$$

...

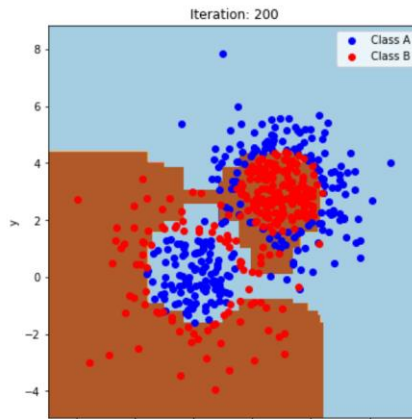


- SVM with gradient descent always finds the optimal hyperplane - **False**
- Gradient descent always finds the global optimum - **False**
- A RNN can be unrolled into an infinite fully connected network - **True**
- Pooling and convolution are operations related to RNNs - **False**
- Learning the structure of a bayesian network is less complex than learning the probabilities - **False**
- SVMs with a linear kernel are especially good for high-dimensional data - **True**
- Random forest is a homogeneous ensemble method - **True**
- If you use several weak learners  $h$  with boosting to get a classifier  $H$  and all  $h$  are linear classifiers, then  $H$  is also a linear classifier - **False**

Decision tree stump example with boosting:

**TU WIEN ! Ensemble Learning – AdaBoost**

- Iterations:
  - Train a new decision tree on training set
  - Very simple model
    - maxheight=1
    - aka One-R (1R)
    - aka Decision Stump
  - Combine all existing trees, weighted, to get a final vote

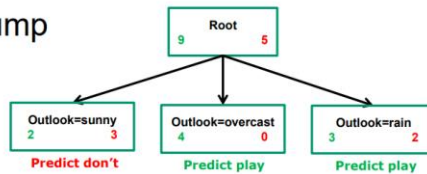


**TU WIEN ! AdaBoost: step by step**

- Weights in beginning: e.g. 1/n

Outlook	Temp	Hum	Windy	Play?	Weight
sunny	85	85	false	Don't	1/14
sunny	80	90	true	Don't	1/14
overcast	83	78	false	Play	1/14
rain	70	96	false	Play	1/14
rain	68	80	false	Play	1/14
rain	65	70	true	Don't	1/14
overcast	64	65	true	Play	1/14
sunny	72	95	false	Don't	1/14
sunny	69	70	false	Play	1/14
rain	75	80	false	Play	1/14
sunny	75	70	true	Play	1/14
overcast	72	90	true	Play	1/14
overcast	81	75	false	Play	1/14
rain	71	80	true	Don't	1/14

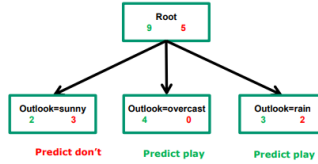
- Train first decision stump



- Compute error
  - 5 errors, each weights 1/14
  - Total weighted error: 5/14 (1/14 + 1/14 + 1/14 + 1/14 + 1/14)
  - Base for classifier weight!



## AdaBoost: step by step



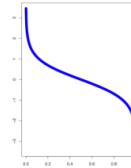
- Weighted error:

$$- 1/14 + 1/14 + 1/14 + 1/14 + 1/14 = 5/14$$

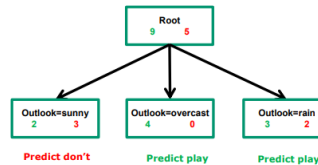
- Classifier weight: e.g.  $\alpha(t) = \frac{1}{2} \times \log \frac{1 - TotalError}{TotalError}$

$$= \frac{1}{2} \times \log \frac{1 - 5/14}{5/14}$$

→ 0.127636253



## AdaBoost: step by step



- Adapt data weights D

$$D_{t+1}(i) = D_{t(i)} \times e^{\pm \alpha t}$$

Decreased if correct (-)  
Increased if wrong (+)

- Normalise weights to sum = 1

Outlook	Temp	Hum	Windy	Play?	Weight	D(t+1)	D(t+1) (norm)
sunny	85	85	false	Don't	0,0714	0,0629	0,0647
sunny	80	90	true	Don't	0,0714	0,0629	0,0647
overcast	83	78	false	Play	0,0714	0,0629	0,0647
rain	70	96	false	Play	0,0714	0,0629	0,0647
rain	68	80	false	Play	0,0714	0,0629	0,0647
rain	65	70	true	Don't	0,0714	0,0812	0,0835
overcast	64	65	true	Play	0,0714	0,0629	0,0647
sunny	72	95	false	Don't	0,0714	0,0629	0,0647
sunny	69	70	false	Play	0,0714	0,0812	0,0835
rain	75	80	false	Play	0,0714	0,0812	0,0835
sunny	75	70	true	Play	0,0714	0,0812	0,0835
overcast	72	90	true	Play	0,0714	0,0629	0,0647
overcast	81	75	false	Play	0,0714	0,0629	0,0647
rain	71	80	true	Don't	0,0714	0,0812	0,0835

- Softmax as activation function is used to scale output to a range of 0..1 - **True**
- Kernels can only be used with SVMs – **False, kernel matrix in MLP**
- Boosting is easy to parallize - **False**
- A good machine learning model has low bias and low variance. - **True**
- Freezing layers means that their weights are only updated during fine-tuning. – **False, it means they are prevented to update their weights during fine-tuning process.**
- Leave-p-out cross validation is computationally expensive on large data sets. - **True**
- There exists no Bayesian network where all instatiated nodes are not d-separated after one node is instatiated. – **False, for example A-> B-> C and B gets instatiated with b1 making A and C d-separated**

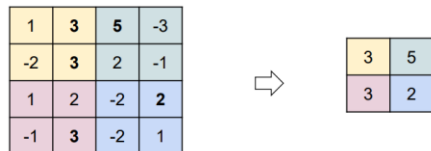
- Lasso can not be used for feature selection. – **False, can be used if coefficients are set to zero.**
- Something with "Off-the-shelf" model – **transfer Learning**
- F1 used with regression – **false, F1 is used with classification as a harmonic mean between precision and recall.**
- Naive Bayes - probability density function used when only nominal attributes used – **False, only numerical attributes**
- Convolutions and max-pooling layers are important for Recurrent Neural Networks – **False, important for CNNs**

Given was an 7x7 input tensor and a 3x3 layer. Had to tick the right output after max-pooling operation with stride 2:



## Pooling Step

- Second very important aspect of a CNN
  - (also called subsampling or downsampling)
  - A **pooling layer** reduces the size of feature maps (i.e. output of a CNN layer and thus the input to the next layer)



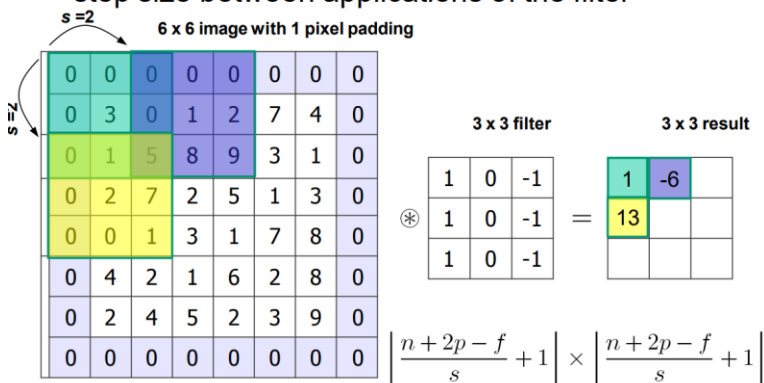
**Max pooling:** take the **max**. activation across small regions (e.g. 2x2, as in the example above)

Given was an 7x7 input tensor and a 3x3 filter. Had to tick the right output after convolution with stride 2.



## Convolutions: Stride

- We can choose a **stride** parameter  $s$  to define the step size between applications of the filter



Given was some training and test data. We should calculate the Recall of NB classifier with Laplace Correction.

- Remedy: add 1 to the count for every attribute value-class combination (*Laplace estimator*)
- Result: probabilities will never be zero
- Additional advantage: stabilizes probability estimates computed from small samples of data

25

## Modified probability estimates

- In some cases adding a constant different from 1 might be more appropriate
- Example: attribute *outlook* for class *yes*

$\frac{2 + \mu/3}{9 + \mu}$	$\frac{4 + \mu/3}{9 + \mu}$	$\frac{3 + \mu/3}{9 + \mu}$
<i>Sunny</i>	<i>Overcast</i>	<i>Rainy</i>

- Weights don't need to be equal (but they must sum to 1)

$\frac{2 + \mu p_1}{9 + \mu}$	$\frac{4 + \mu p_2}{9 + \mu}$	$\frac{3 + \mu p_3}{9 + \mu}$
-------------------------------	-------------------------------	-------------------------------

- MAE is less sensitive to outliers – **True, mean absolute error is less sensitive to outliers than the mean-squared error**
  - Freezing layers means these layers will be fine-tuned in the fine-tuning phase **False, it means they are prevented to update their weights during fine-tuning process.**
  - Overfit is more likely on a smaller test set - **True**
  - Boosting is easily parallelizeable - **False**
  - Paired t-tests used for folds verification in holdout method (train /test split) - **False**
  - Random Forests use Bootstrapping - **True**

A classifier called "stump" classifier is given, i.e. a 1-level decision tree (can make only one split). Given was an x-axis with 3 points  $x_1 = 1$ ,  $x_2 = 3$ ,  $x_3 = 5$  (or something like that), with associated class labels -1, 1, -1.

- what is the weight of each of the data points before classification? – **1/3 (1/n, n = number of samples)**
- let the first stump classifier split into two regions, i.e. draw a decision boundary – **possibly the decision boundary on  $x = 3$ , so either the left or the right point is misclassified**
- circle that data point that will get a higher weight for the second classification stage – **either left or right, depending on where to include the label -1**

Name two methods to compute coefficients in linear regression – **Gradient Descent and Normal Equation**

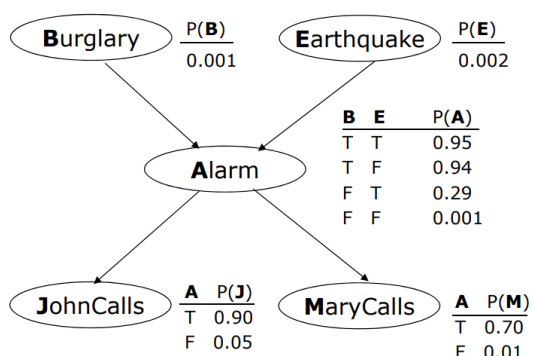
Given a dataset with 13 samples, 4 categorical variables (age, education, income, marital status) and target (purchase) Yes/No

a) Compute 1R on samples 1-8, and compute Precision and Accuracy for samples 9-13. Interestingly, the 1R had 1.0 accuracy and precision (split on age)

b) Propose a Bayesian Net for the full dataset and briefly argue why you chose this net. Compute some conditional probabilities in the net.

**Example:**

## Bayesian Networks (Example)



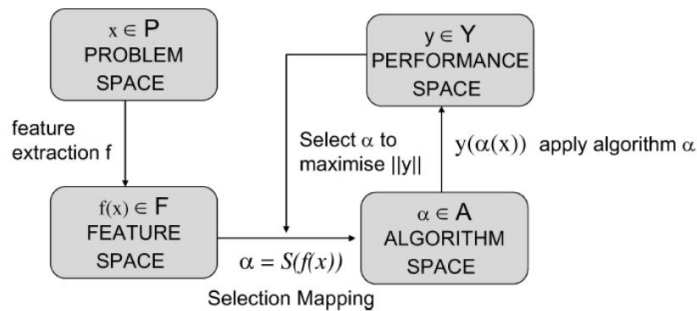
- Random forests heterogenous ensemble learner? – **False, homogenous because the same learners are used**
- Perceptron with Softmax solves XOR-Problem – **False, softmax layer does not solve the non linearity**
- D-Separation used in neighbourhood searches in a Bayes net – **False, in neighbourhood we add a connection, remove it or change its direction**
- Paired t-tests used for folds verification in holdout method (train /test split) – **False, used for significance testing**
- If information gain a method for unsupervised feature selection? – **False, supervised method for feature ranking**
- Is Bayesian optimization used for construction Bayesian networks? - **False**
- SVM: Does the SVM method/algorithm guarantee convergence to globally optimal solution? – **No, does not guarantee the convergence to globally optimal solution (more on that: <https://stackoverflow.com/questions/12606934/does-svm-classification-always-produces-unique-solution>)**



2) Explain Rice's framework for algorithm selection. Can it be used for (Bayesian?) optimization of hyperparameters?



## Algorithm selection with Rice's framework



Input (see [8] and [9]):

- Problem space  $P$  that represents the set of instances of a problem class
- A feature space  $F$  that contains measurable characteristics of the instances generated by a computational feature extraction process applied to  $P$
- Set  $A$  of all considered algorithms for tackling the problem
- The performance space  $Y$  represents the mapping of each algorithm to a set of performance metrics

Problem:

For a given problem instance  $x \in P$ , with features  $f(x) \in F$ , find the selection mapping  $S(f(x))$  into algorithm space, such that the selected algorithm  $a \in A$  maximizes the performance mapping  $y(a(x)) \in Y$

**It is not directly applicable for hyperparameter optimisation because it tries to choose the best suiting algorithm and not to tune algorithms hyperparameters. However, if we consider automated ML and choosing an algorithm as a hyperparameter itself than Rice's framework can help.**

7) Example data set, with some calculations to do. 9 given data points, 3 new data points given. It was explicitly asked to show the calculations necessary for arriving at the results.

for the 3 new data points, use 3-NN to predict the class labels. Define a suitable distance function, and explain why you chose it. (5pts) – **Use Euclidean per default if nothing is against it**

one specific variable was to be removed, now predict again, changes? (this was worded differently, but that's basically what was asked) (2pts)


Use Naive Bayes classifier to classify the 3 new observations, then calculate precision and recall of this classifier. Laplace correction should not be used. (6pts) – **Precision is True Positives / (True Positives + False Positives), Recall is True Positives / (True Positives + False Negatives)**

- Lazy learners good for big data – **False, bad because computationally very expensive (have to compute distances for the instances each time for example kNN)**
- Naive Bayes, Laplace correction must be used – **False, must not be used, should be used if we have zero frequency problem**
- Paired T-tests used at folds verification?? – **False, used for statistical significance**

Decision Tree:  $k$  features  $\gg N$  samples. A) If all  $k$  are binary, how many leaves are possible, what is max depth? -  $2^k$  and maximal depth is  $k$ , if each feature is used for splitting and  $2^k$  is the number of possible combinations of splits. However, if we consider that  $k > N$ , than we can have at max  $N$  leaf nodes and probably still the depth of  $k$ .

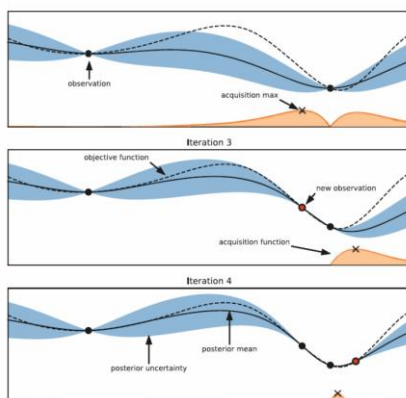
B) If  $k$  are continuous, how many max leaves and max depth? – Max of  $N$  leaf nodes, depth is still maximally  $k$ , because  $k > N$ . Otherwise the depth could converge to infinity because the features are continuous but will be more limited through the  $N$  samples, so no infinite depth.

Describe optimization of hyper parameters with Bayes optimization:

 Sequential Model-based Bayesian Optimization

- General framework for minimizing blackbox functions  $f$
- A probabilistic model  $M$  is used to model  $f$  based
  - on point evaluation of  $f$
  - available prior information
- Applies  $M$  to select promising inputs to evaluate  $f$  next
- $M$  is a regression model fitted on data tuples  $(\lambda, L(A_\lambda))$
- Selection of next hyperparameter configuration  $\lambda$  :
  - Acquisition function:  $a_M: \Lambda \rightarrow \mathbb{R}$
  - Most useful configuration  $\lambda$  is evaluated next
  - Most popular acquisition function is *expected improvement* over the best previously-observed function value  $f_{\min}$
  - Probabilistic model  $M$  is used to predict a posterior distribution over function values  $p_M(f | \lambda)$

 Bayesian Optimization



---

**Algorithm 1** Sequential Model-based Bayesian Optimization (SMBO)

---

```
1: Initialize model  $\mathcal{M}$ ;  $\mathcal{H} \leftarrow \emptyset$ 
2: while time budget for optimization has not been exhausted do
3:    $\lambda \leftarrow$  determine candidate configuration from  $\mathcal{M}$ 
4:    $i \leftarrow$  select cross-validation fold
5:   Compute  $c = \mathcal{L}(A_\lambda, \mathcal{D}_{\text{train}}^{(i)}, \mathcal{D}_{\text{valid}}^{(i)})$ 
6:    $\mathcal{H} \leftarrow \mathcal{H} \cup \{(\lambda, c)\}$ 
7:   Update  $\mathcal{M}$  given  $\mathcal{H}$ 
8: end while
9: return  $\lambda$  from  $\mathcal{H}$  with minimal  $c$  across cross-validation folds
```

---

- when k-nn is used with 1-n encoding, min-max scaling is needed to perform euclid. Distance – **True, if there are other attributes which have other values ranges.**
- backpropagation is a technique used with perceptrons (single layer) – **False, with multiple layers**
- pruning is needed for underfitting – **False, pruning is needed to avoid overfitting**
- k-nn is recommended for large datasets – **False, very slow because everything has to be calculated**
- paired t-tests are used when dealing with train/test splits – **False, when checking the significance of the difference in algorithm performance**
- for k-nn categorical features should be normalized – **False, encoding should be done**
- difference between z-score and min-max, when to use which on which feature types - 4 pts – **z-Score has mu (often 0) and sigma (often 1) which centers and scales data. Min -max scales data to a specific range – most often between 0 and 1. For continuous features both can be used, for discret min-max is recommended. Z-score assumes normal distribution, if data is skewed than min-max should be used. Z-score is more robust to outliers, min-max could be sensitive to them.**
- 3. difference naive bayesian network to normal network - 6 pts – **if we consider naive bayes network to Bayesian network then the key difference is independence assumption in naive bayes.**
- 5. name some model based attributes in meta learning - 3 pts - **Decision Trees: maximum depth, nodes per feature, tree imbalance. KNN – k and distance function, SVM kernel function and polynomial number**
- 6. example with 8 instances, perform 1R and naive bayes on 5 test examples (calculate accuracy, precision and recall), zero frequency was optional to handle - 7 pts 1R - 8 pts naive bayes

Entropy max when all elements from same class? - **False, then it is minimal**

RF heterogeneous ensemble learner? - **False, homogenous**

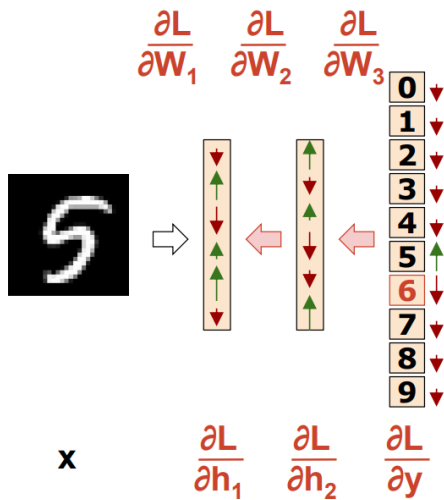
PCA supervised feature selection method? – **False, unsupervised**

McNemar's Test is used for significance testing? - **True**

4.) Gradient Descent pseudo code.



## Multi-Layer Perceptron: Training



1. Initialize learnable weights randomly
2. Repeat until satisfied:
  - a. pick training example
  - b. compute output
  - c. compute **gradient** of loss wrt. output
  - d. backpropagate loss, computing **gradients** wrt. learnable weights
  - e. update weights in direction of negative **gradient** (to reduce loss for this example)

5.) HMM structure:



## Markov Decision Processes

- We consider finite Markov decision processes (MDPs)
- Feedback, as in bandits, but also selection of different actions in different situations
- Actions influence also subsequent situations and future rewards
- In MDPs we estimate the value  $q(s, a)$  of each action  $a$  in each state  $s$
- The agent and environment interact at each of a sequence of discrete time steps,  $t = 0, 1, 2, 3,$
- At each time step  $t$   $S_t \in \mathcal{S}$



## MDPs

- Selects an action  $A_t \in \mathcal{A}(s)$
- As a consequence of its action, the agent receives a numerical reward  $R_{t+1} \in \mathcal{R} \subset \mathbb{R}$ .

$$S_0, A_0, R_1, S_1, A_1, R_2, S_2, A_2, R_3, \dots$$

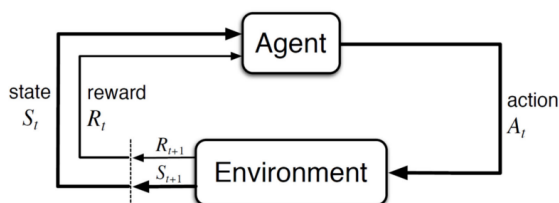


Figure 3.1: The agent–environment interaction in a Markov decision process.

- K-d-Tree can be used as search space optimisation for k-NN - **True**
- Random Forests is a boosting ensemble technique – **False, bagging ensemble technique**
- Back propagation is a method for training Multi-Layer Perceptrons - **True**
- Ordinal data does not allow distances to be computed between data points - **True**
- In AdaBoost, the weights are uniformly initialised - **True**
- Suppose we have a neural network with ReLU activation function. Let's say, we replace ReLU activations by linear activations. Would this new neural network be able to approximate an XOR function? (Note: The neural network was able to approximate the XOR function with activation function ReLU) – **No, because XOR is non linear separable**
- The entropy of a data set is based solely on the relative frequencies of the data distribution, not on the absolute number of data points present - **True**
- k-nearest neighbors is based on a supervised learning paradigm - **True**
- Support Vector Machines with a linear kernel are particularly suitable for classification of very high dimensional, sparse data – **True**
- Support Vector Machine can by default only solve binary classification problems - **True**
- Naive Bayes gives usually good results for regression data sets - **False**
- Learning the structure of Bayesian networks is usually simpler than learning the probabilities - **False**
- Learning the structure of Bayesian networks is usually more complicated than learning the probabilities - **True**
- The mean absolute error (a performance metric used for regression) is less sensitive to outliers than MSE - **True**
- Chain Rule simplifies calculation of probabilities in Bayesian Networks - **True**
- "Number of attributes of data set" is not a model based features that is used for metalearning - **True**
- Kernel projections can only be used in conjunction with support vector machines – **False, kernel matrix MLP**
- Suppose a convolutional neural network is trained on ImageNet dataset. This trained model is then given a completely white image as an input. The output probabilities for this input would be equal for all classes. - **False**
- When learning an SVM with gradient descent, it is guaranteed to find the globally optimal hyper plane. - **False**
- Usually state of the art AutoML systems use grid search to find best hyperparameters – **False computationally exhaustive thus uses simulated annealing more**
- Linear regression converges when performed on linearly separable data - **False**
- Linear regression converges when performed on linearly not separable data – **True, however still has some error**
- Laplace Corrector must be used when using Naive Bayes – **False, can be used**
- Gradient boosting minimizes residual of previous classifiers - **True**
- Decision Trees using error rate vs entropy leads to different results - **True**
- Depth of decision tree can be larger than the number of training samples used to create a tree - **F (depth of tree not larger than number of samples)**

•

Consider the following 2D data set. Which classifier(s) will achieve zero training error on this data set?

+

0

- Perceptron – **No, data not linearly separable**
  - SVM with a linear kernel – **No, data not linearly separable**
  - Decision tree (T) - **Yes**
  - 1-NN classifier - **Yes**
- Describe at least three methods that are used for hyperparameter optimization - **Grid Search, Random Search, Bayesian Optimisation.**
- How can we select automatically the most promising machine learning algorithm for a particular data set? – **Using Rice's Framework**
- Why a general Bayesian Network can give better results than Naive Bayes? – **Because we do not have to assume independence and in real world data independence is rarely given**
- What is overfitting, and when & why is it a problem? Explain measures against overfitting on an algorithm discussed in the lecture – **Through overfitting model learns much noise and unnecessary aspects of learning data and thus cannot generalise well on the unknown test data. Measures against – Regularisation (penalising complex models), batch normalisation, dropout, feature selection**
- What is the difference between micro and macro averaged performance measures? – **Micro computes performance metrics across all classes, macro computes them per class and averages across classes**
- Which are the important issues to consider when you apply Rice's framework for automated selection of machine learning algorithms. – **Have to assume Closed Classification World Assumption, have to consider the computational costs of feature selection and selection mapping (choosing the algorithm). Ideally a small set of algorithms should be selected with a good coverage but in practice it is very difficult with minimal set of learning algorithms, No Free Lunch theorem (cannot have an algorithm which is best for every problem), base learners should have different biases and be representatives of different model classes.**
- How can we avoid overfitting for polynomial regression? – **Regularisation (Lasso Regression, Ridge Regression). For regularisation – large lambda -> high bias, low variance, small lambda -> low bias, high variance. Lambda controls the model complexity.**
- something like: how is 1-R related to Decision tree – **1 Rule or Decision Stump is decision tree with only a root node**

NOT EXACT, BUT SOMETHING LIKE THIS:

- In which order should the steps be when training neural network with gradient descent (and 5 options listed, should be place in correct order) :
  - - Initialize weights and bias - **1**
    - Let input through the NN to get output - **2**
    - Get error rate(compare what was expected to output or smthg like that) - **3**
    - Adjust weights - **4**
    - Reiterate until the best weights are in place - **5**
- Can kernel be used in perceptron ? – **Yes, kernel matrix in Multi Layer Perceptron, if single Layer than No**
- How can we automatically pick best algorithm for a specific dataset ? – **Using Rice's Framework**

- How can you learn the structure of Bayesian Networks? – **Through Experts doing it manually or computers. Experts are better for structure, computers for probability.**
- Explain how we can deal with missing values or zero frequency problem in Naive Bayes.
  - **Ignore the missing values / apply Laplace correction (more on that above)**

What is Deep Learning? Describe how it differs from "traditional" Machine Learning approaches? Name two application scenarios where Deep Learning has shown great advances over previous methods.

#### Traditional ML:

- **express problems through mathematical function**
- **Learn parameters from prepared data**  
⇒ **Feature extraction hard coded**

#### Deep Learning:

- **Use an ensemble of simple functions to create solutions**
- **Let algorithm operate on raw data (e.g. images) and learn its own feature extraction**
- **Learn parameters based on extracted features**

**Great advances – Image Classification, Audio classification/Extraction, Text Processing (LLMs).**

- What is the randomness in random forests? Describe where in the algorithm randomness plays a role – **Random Forest uses Bootstrapping, it randomly chooses subsets from a dataset and trains single decision trees on them. Additionally, for each node the random selection of features from the dataset is done.**
- Describe 2 AutoML systems :



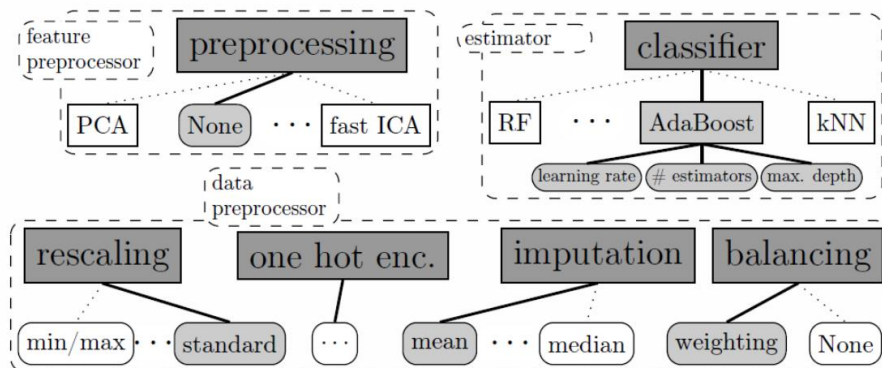
## AutoML Systems

---

- Auto-WEKA
  - Automatic model selection and hyperparameter optimization (see Chapter 4 in [H2018])
- Hyperopt-Sklearn (Chapter 5 in [H2018] )
- Auto-Sklearn (Chapter 6 in [H2018] )
- TPOT ([O2016])
- Google AutoML



## Auto-Sklearn ([F2017])



The only things from the slides. Possibly also SMAC can be seen as an AutoML system, although not sure:



## SMAC [H2011]

- Can be used for optimization of parameters for arbitrary algorithms
- Based on set of instances
- Optimization of hard combinatorial problem solvers
- Hyperparameter optimization of machine learning algorithms
- Uses random forest to model  $p_M(f | \lambda)$