Online Machine Learning

Ensemble Methods

Antonio Carta

antonio.carta@unipi.it

Outline

- **Basics**: what is an ensemble and why we need them in OML
- **Offline Ensembling** Methods
- **Online Ensembling** Methods

General Concepts

Wisdom of the Crowd

In 1907, [Sir Francis Galton] asked 787 villagers to guess the weight of an ox. None of them got the right answer, but when Galton averaged their guesses, he arrived at a near perfect estimate

Quote from: https://www.nationalgeographic.com/science/article/the-real-wisdom-of-the-crowds

What is an Ensemble?

- A set of models
- Combined together
- With a mechanisms to compute the output

Questions:

- How to train each model (e.g. what data to use)
- How to compute the output

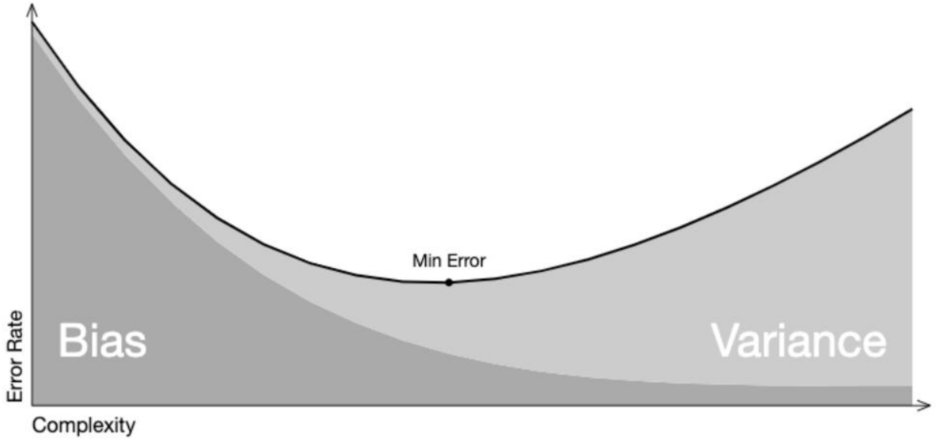
Ensemble in OML – Motivations

Many techniques we see today are offline methods. Why are we studying them?

- Ensembles are a popular technique in OML
- In the presence of **drifts**, models will be retrained, forgetting old concepts
- In the presence of **recurrent concepts**, this is a problem
- We can fix it by having a **separate model for each concept**
- Then, we can **select the correct model** at each step (easier said than done)

Bias-Variance Tradeoff

- **Bias**:
  - **Simple model** → ignores relevant information, error due to **bias is high**
  - **Complex models** → error due to bias decreases (**low bias**)
- **Variance**:
  - Simple model → less error due to variance
  - Complex model → error due to variance increases.
- **Tradeoff**: The ideal model finds the optimal bias-variance tradeoff



http://www.r2d3.us/visual-intro-to-machine-learning-part-2/

Ensemble Definition

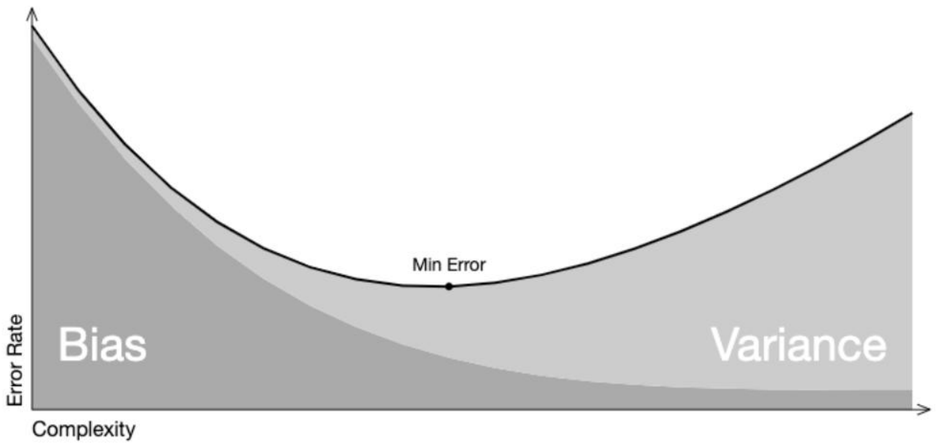
"An ensemble can be described as a ***composition*** *of multiple* ***weak learners*** *to form one with (expected) higher predictive performance (strong learner), such that a weak learner is loosely defined as a learner that performs slightly better than random guessing"*

Freund and Schapire, 1997

Weak Learners and Ensembling

- **Weak learners** have high-bias/low-variance
- An **ensemble** of weak learners improves the performance

- How do we choose and train the weak learner?
- **KEY IDEA**: The bias is decreased if the errors are decorrelated. We have to ensure the weak learners are different.



Properties

- **Diversity**: induce diversity among learners
- **Combination**: combine the predictions
- **Adaptation (for OML)**: adapt to evolving data

Advantages:

- High Predictive performance
- Flexibility

Disadvantages:

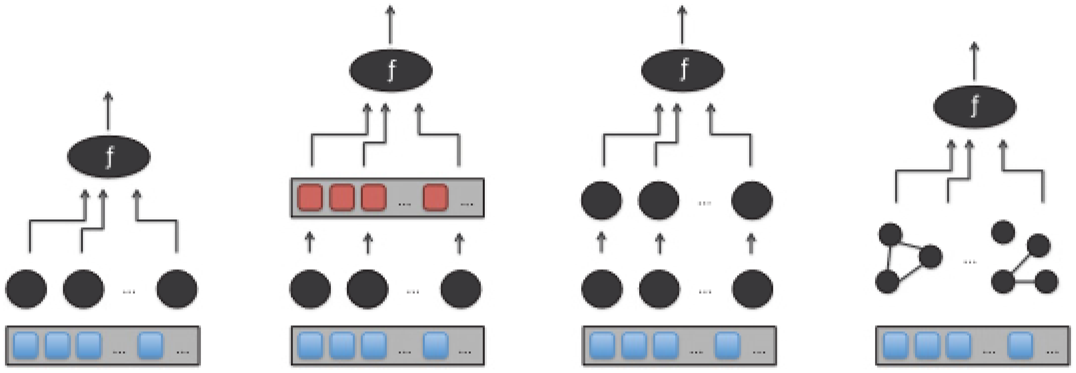
- Computational resources

Ensemble – Diversity

- Ensembles work best when the base models are different and have decorrelated errors
- **horizontal partitioning**: training classifiers in different chunks of data
- **vertical partitioning**: training with different subsets of features

Ensemble – Combination

- **Flat**: base models trained on input data, decision fusion via simple combination
- **Meta-learner**: the combination function is a model itself (meta-learned)
- **Hierarchical**: structured organization of the base learners
- **Network**: base models are nodes in a graph. The graph structure informs the combination scheme

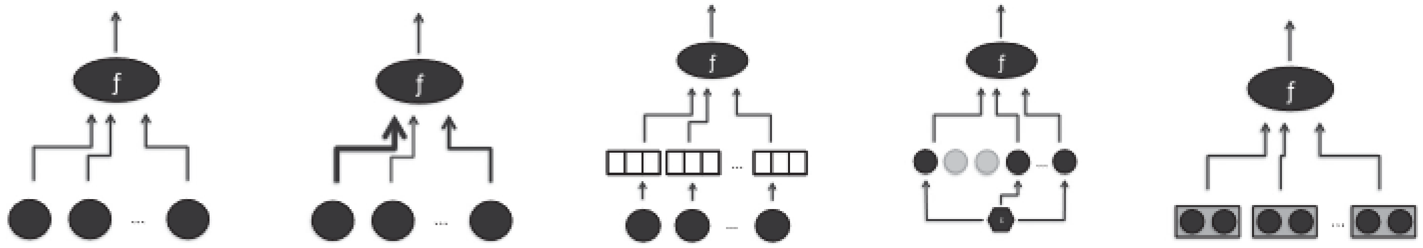


(a) Flat (b) Meta-Learner (c) Hierarchical (d) Network

Gomes, H. M., Barddal, J. P., Enembreck, F., & Bifet, A. (2017). A survey on ensemble learning for data stream classification. ACM, 50(2), 1-36.

Ensemble – Voting Schemes

- **Majority vote**: every classifier has the same weight
- **Weighted majority**: gives a different weight to each classifier
- **Classifier selection**: uses a dynamic criterion to select the best classifier for the current sample



(a) Majority (b) Weighted Majority (c) Rank (d) Classifier Selection (e) Relational

Gomes, H. M., Barddal, J. P., Enembreck, F., & Bifet, A. (2017). A survey on ensemble learning for data stream classification. ACM, 50(2), 1-36.

Ensemble – Adaptation

- How many base models?
- **Fixed**: the number of base learners cannot grow
- **Dynamic**: add classifiers on the fly
  - We still need to limit the memory growth somehow

Offline Ensembles

Bagging – Bootstrap Aggregating

IDEA: Increase diversity by training on different subsets of the data

Bootstrap: sample with replacement

- An example of horizontal partitioning

Training:

- Sample m iid datasets of dimension n from the original data (with replacement)
- Train m models independently, one for each bootstrapped set

Inference: Combine outputs (average or majority voting)

Properties:

- Bagging reduces the variance
- Base models should have high variance to encourage diversity

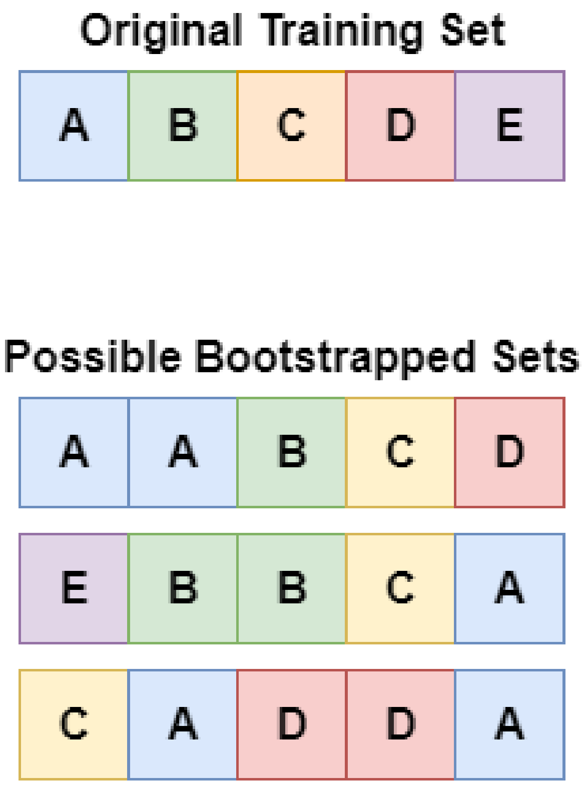


Image: wikimedia

Bagging

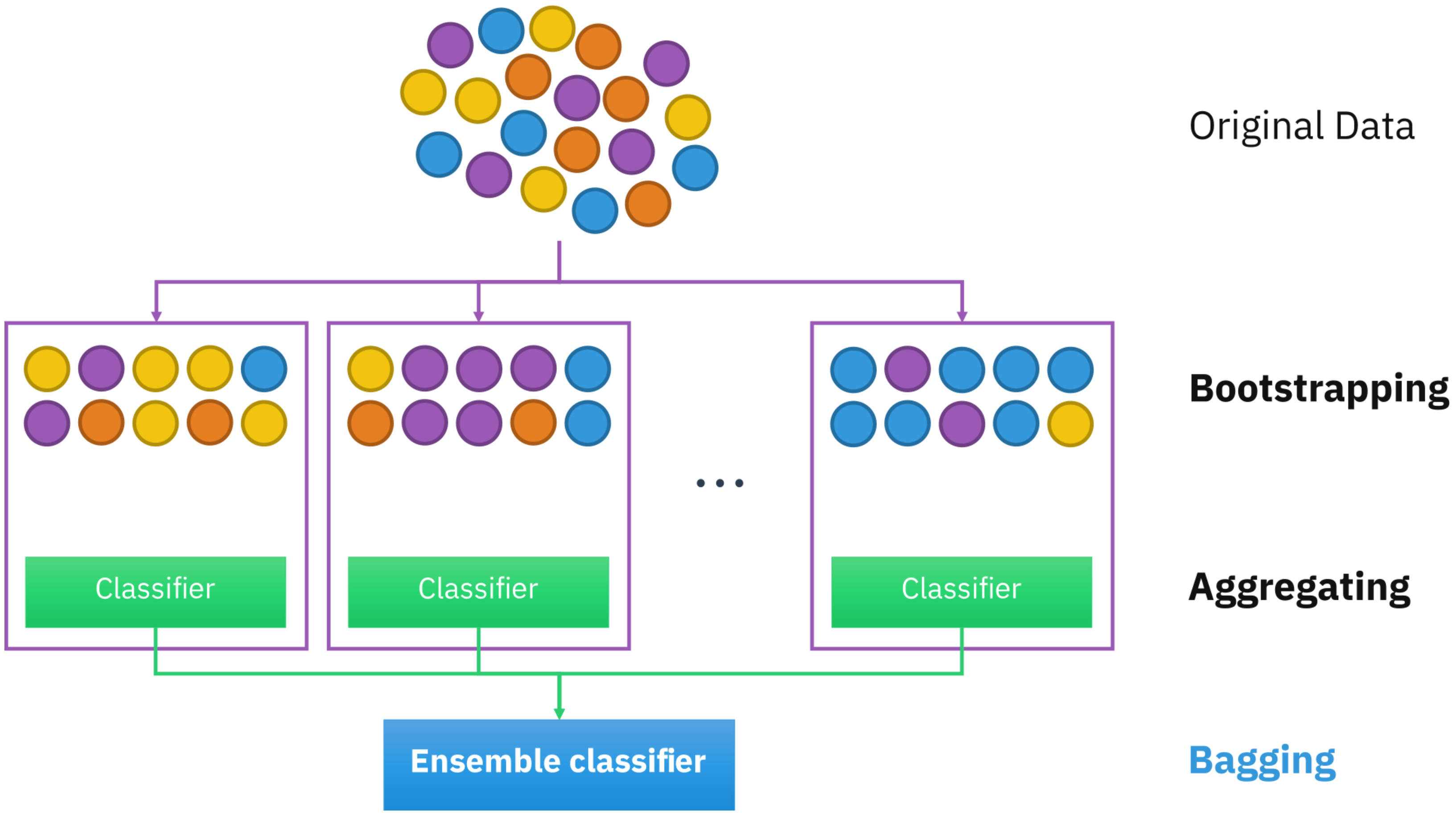


Image source: By Sirakorn - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=85888768

Random Forests

- Popular example of bagging
- **Train** m decision trees, each one on their own bootstrapped dataset
- **Weak learners:**
  - Limit the tree depth
  - Vertical partitioning: train each decision tree on a subset of features

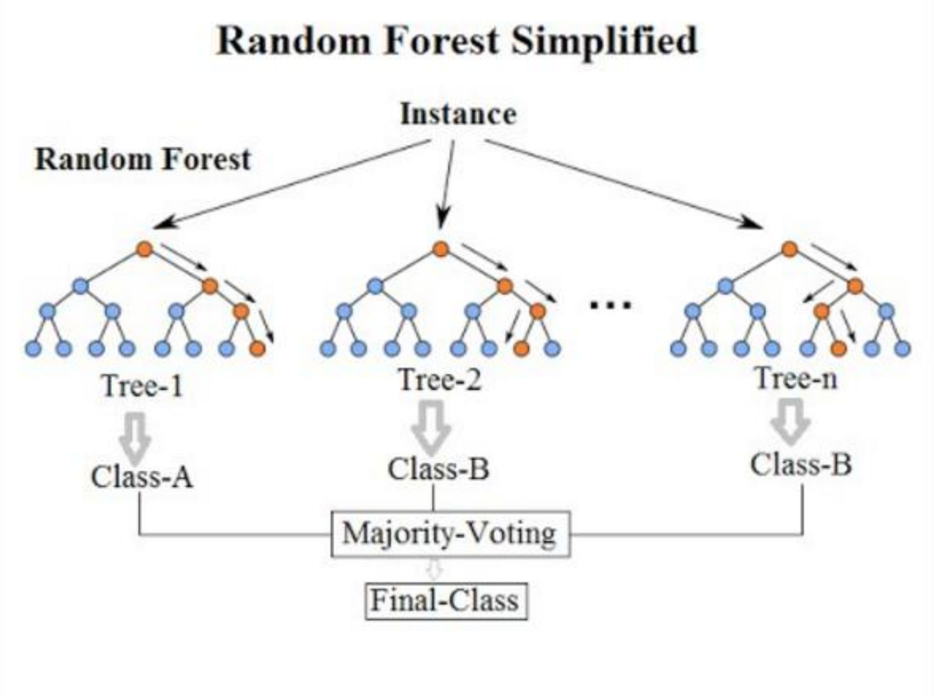
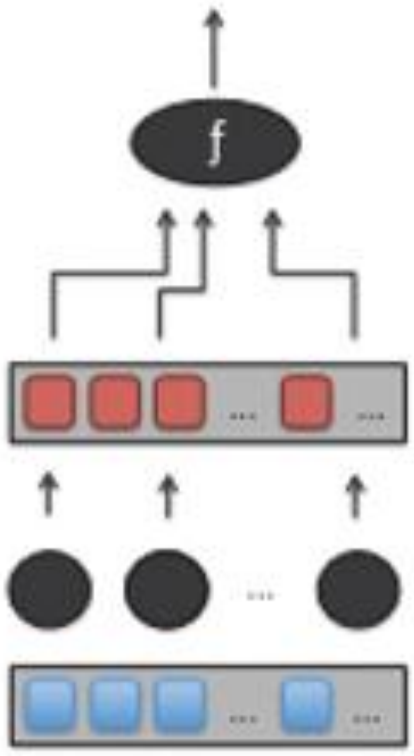


Image: wikimedia

Stacking

- aka meta-learner combination scheme
- Train base models
- Fit a meta-learner to combine the outputs
  - Often a very simple model, such as a linear combination of the outputs
  - Example: $y_{ensemble} = \sum_i \alpha_i y_{base}^i$
- Advantages: Can learn the combination scheme



Online Ensembles

Streaming Cross Validation

How do we split the stream if we want to train (and evaluate) an ensemble? We need to ensure diversity and robust training and evaluation.

K-fold distributed cross-validation:

each sample is used for testing in one classifier selected randomly, and used for training on all the others

- Adaptation of offline cross-validation
- Good use of the data (only one 1/k samples are unused)
- high redundancy

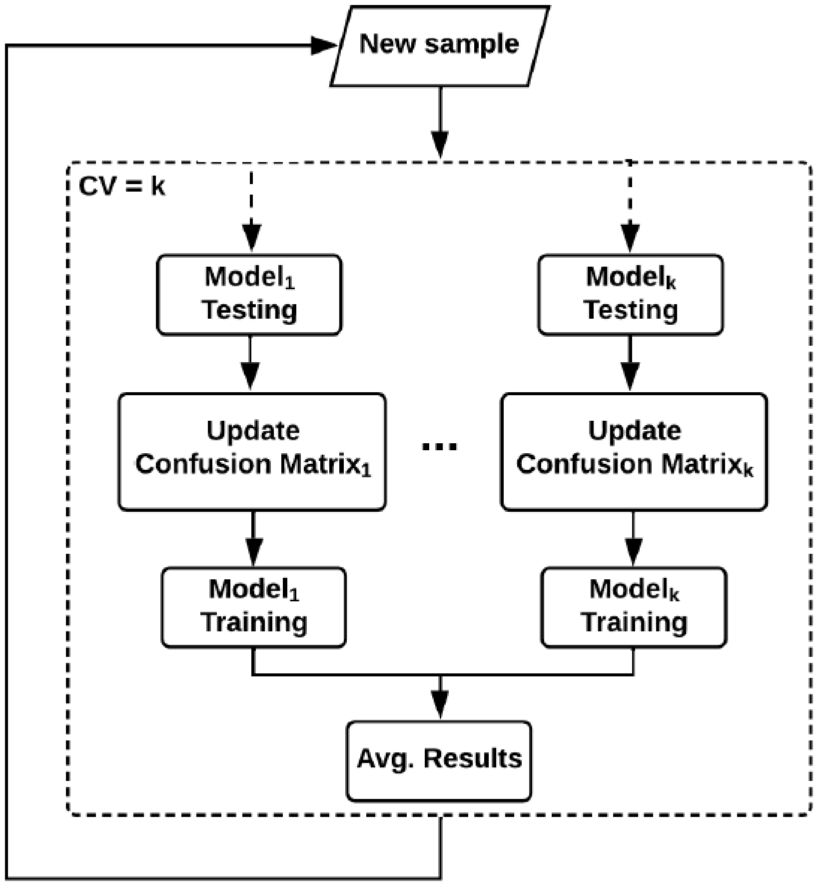
K-fold distributed split-validation:

each sample is used for training in one classifier selected randomly, and for testing in all the other classifiers

- Models trained on disjoint data
- Under utilization of data. Each model uses 1/k data for training

K-fold distributed bootstrap-validation:

each sample is used for training in approximately 2/3 of the classifiers, with a separate weight in each classifier, and for testing in all the classifiers *(we will see online boostrap in the following slides)*

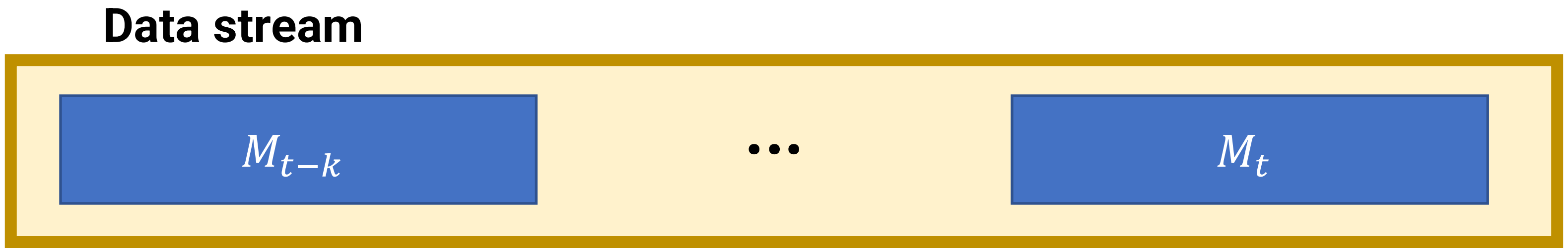


Bifet, A., et al: Efficient Online Evaluation of Big Data Stream Classifiers. In ACM SIGKDD, 2015.

Accuracy-Weighted Ensemble

IDEA: train an ensemble of offline methods. Adapt the ensemble by removing the oldest models.

- Process the stream in chunks
- Whenever a new chunk is ready, train a new model to add to the ensemble
  - Remove the oldest one to keep a fixed number of models in the ensemble
- LIMITATION: We have to fix the window size

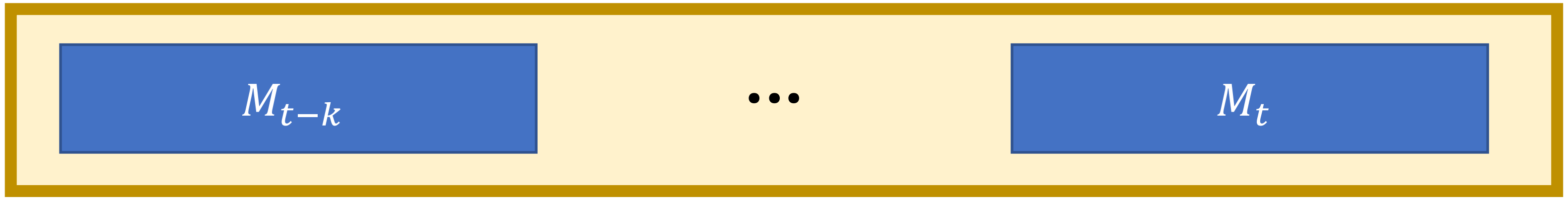


Reference: MOA book (Ch 7.1)

Accuracy-Weighted Ensemble

- **Weighting Scheme**: each classifier is weighted by the expected accuracy on the future data
  - Weight $w_i = err_i - err_r$
    - err_i error of classifier i
    - err_r error of random classifier
  - Ideally, estimated using a separate test set with recent data
  - Realistically estimated using the last chunk of data from the stream
- **Ensemble output**: $f(x) = sign(\sum_i w_i C_i(x))$

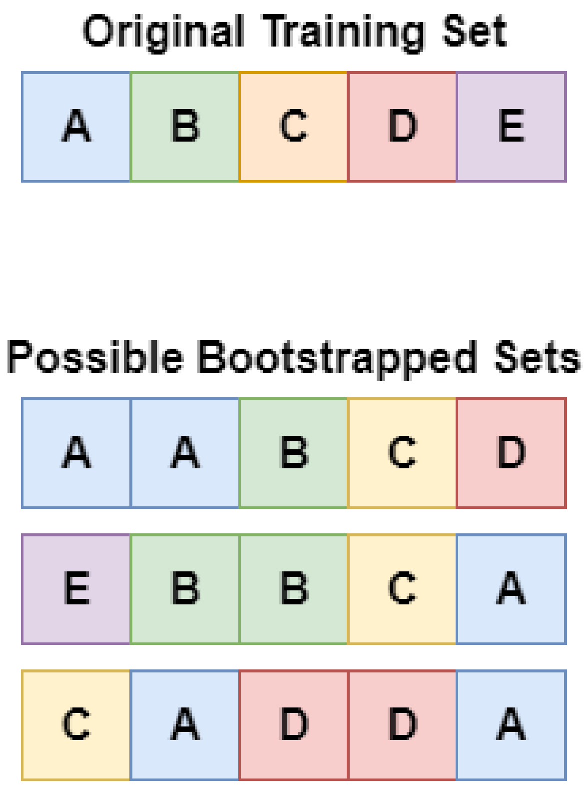
Data stream



Online Bagging

IDEA: We want to use bootstrap (remember: sampling with replacement). How can we do it online?

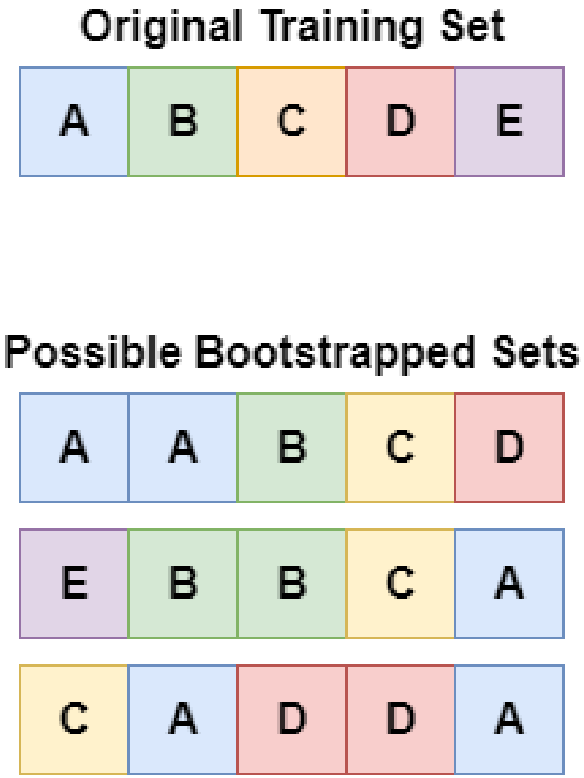
- **OFFLINE**: draw n random i.i.d. samples uniformly with replacement
- **ONLINE**: we give a discrete weight to the current sample in the stream
  - Equivalent to the number of copies of the current example that we would get in the offline setting
  - What distribution should we use?



Online Bagging – Binomial Distribution

- n = number of examples in a bootstrapped set
- k = number of replicas for a specific example
- The number of replicas in a bootstrapped set follows a binomial distribution:
  - We perform n independent experiments where:
    - With probability $p = \frac{1}{n}$ we pick the current example
    - With probability $1 - p$ we pick another example

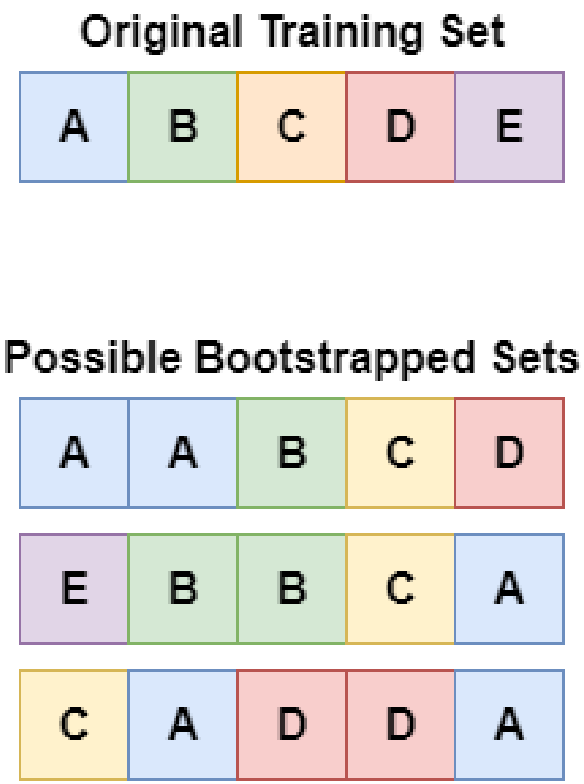
$$P(K = k) = \binom{n}{k} p^k (1-p)^{n-k} = \binom{n}{k} \frac{1}{n}^{k} \left(1 - \frac{1}{n}\right)^{n-k}$$



Online Bagging – Binomial vs Poisson

- For large n, **the binomial can be approximated with a Poisson distribution**
- **Online Bagging**: draw weights according to a Poisson with $\lambda = 1$

$$\mathrm{Poi}(x|\lambda) = e^{-\lambda}\,\frac{\lambda^x}{x!}$$



Ref: MOA Book (Ch. 7.4)

Online Bagging – Pseudocode

ONLINE BAGGING(*Stream*, M)

Input: a stream of pairs (x, y), parameter M = ensemble size
Output: a stream of predictions $\hat{y}$ for each x

initialize base models h_m for all $m \in \{1, 2, \ldots, M\}$
for each example (x, y) in *Stream*
    do predict $\hat{y} \leftarrow \arg\max_{y \in Y} \sum_{t=1}^{T} I(h_t(x) = y)$
        for $m = 1, 2, \ldots, M$
            do $w \leftarrow Poisson(1)$
                update h_m with example (x, y) and weight w

Figure 7.2

Online Bagging for M models. The indicator function I(condition) returns 1 if the condition is true, and 0 otherwise.

Source: MOA Book (Fig. 7.2)

ADWIN Bagging

IDEA: Bagging in the presence of Concept Drift

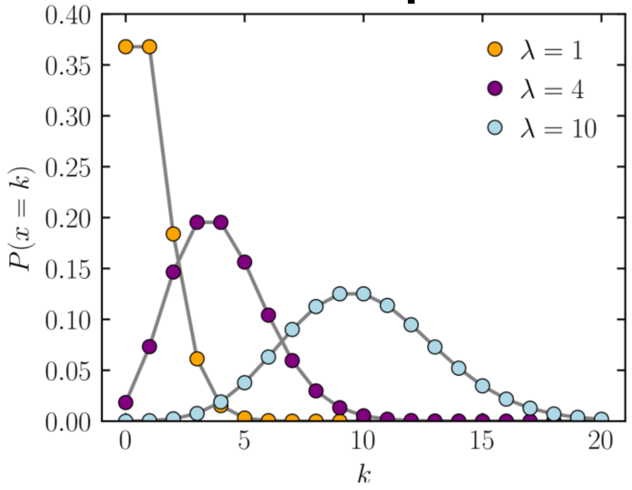
- **ADWIN Bagging**:
  - Bagging ensemble
  - ADWIN for concept drift detection
- **Keep m CD detectors (ADWIN), one for each base model**
  - REMEMBER: ADWIN looks at the stream of errors and consider a CD whenever the error decreases too much between the old and new window (according to a statistical test)
- **When a CD detector finds a CD**:
  - Remove the worst classifier («*replace the loser*» strategy)
  - Train a new one

Ref: MOA Book (Ch 7.4.2)

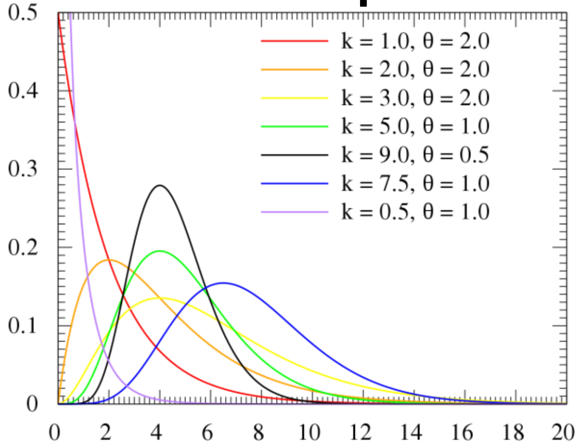
Leveraging Bagging

- Adding more randomness can improve the performance
- We can also change λ in a Poisson distribution to control the weights
  - Higher λ increases the mean and results in a flatter distribution
- Alternative distributions: Gamma $\Gamma(k, \theta)$
- Example: **Leveraging Bagging**
  - Samples from Poisson with $\lambda \geq 1$

Poisson pdf



Gamma pdf



Ref: MOA book (Ch. 7.4)

Leveraging Bagging

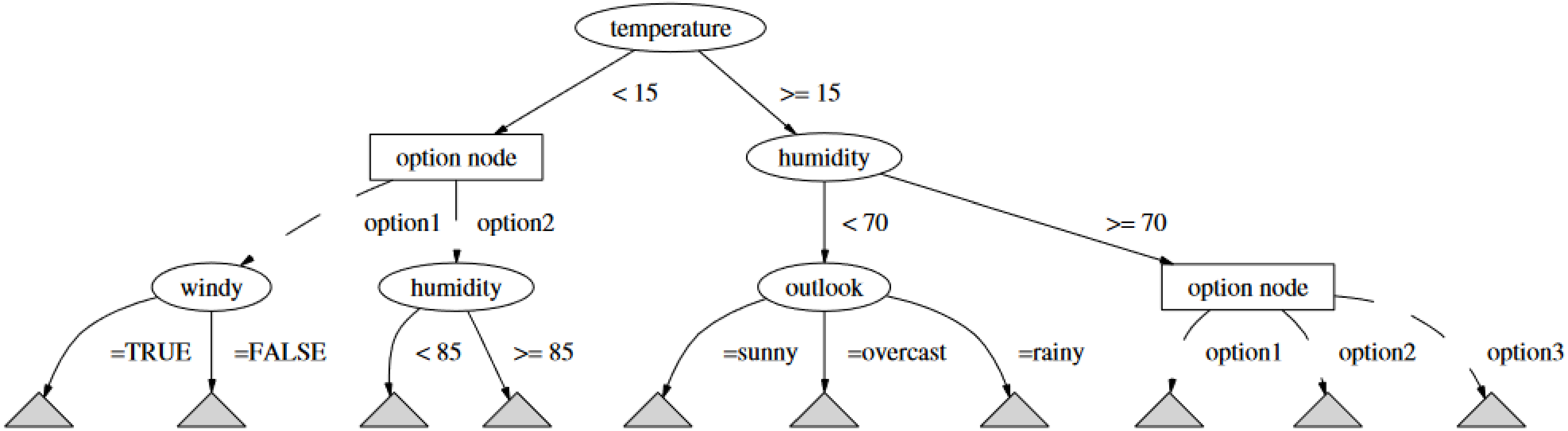
- Sample from **Poisson** with $\lambda \geq 1$
- **ADWIN Bagging** with replace the loser strategy
- **Error-correcting output codes**:
  - In multi-class problems map set of classes into two sets {0, 1}
  - Converts a multi-class problem into a binary classification problem
  - Each base model uses a different code
    - Increases diversity because each model is learning a different function

Hoeffding Option Tree (HOT)

- Ensemble of trees in a single tree structure
- **Option Nodes** represent an ensemble of subtrees
- **INFERENCE**:
  - Each subtree is evaluated in parallel
  - Output from the leaves are aggregated (e.g. weighted voting)
- **TRAINING**:
  - If the splitting criterion for different attributes is similar and we have enough data (according to the Hoeffding bound), we split the tree into **several options**
  - Less instability
  - We can split earlier
  - REMINDER: in the VFDT, if attributes are close enough, we pick only the best.

Ref: MOA Book (Ch. 7.6)

Hoeffding Option Tree

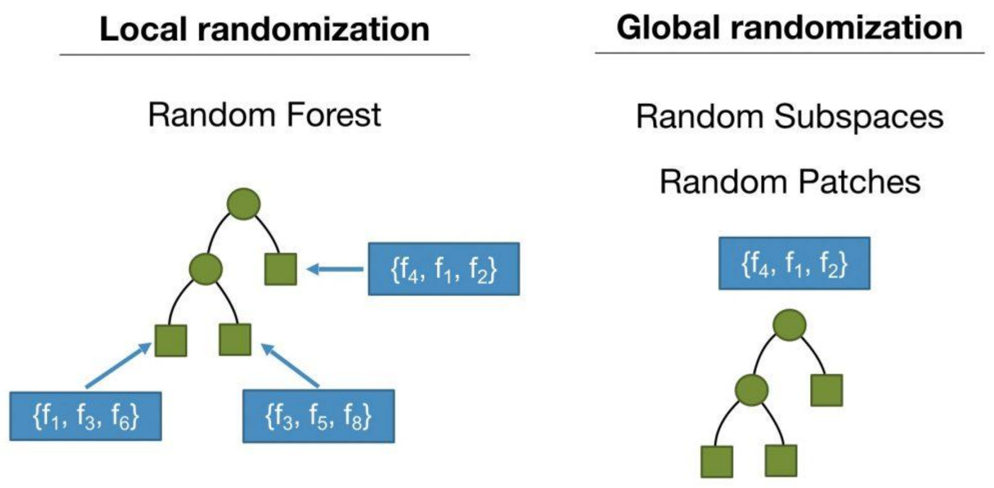


B. Pfahringer et al. "New Options for Hoeffding Trees." In AI 2007: Advances in Artificial Intelligence, LNCS Springer, 2007.

Streaming Random Patches (SRP)

Adaptation of Random Forest for online learning

- **Adaptive Random Forest (ARF)** uses **local subspace randomization**: random subsets of features are set for each leaf to be considered for future node splits
- **SRP uses global subspace randomization**: each base model is trained on a randomly selected subset of features
- ARP and SRP are available in river



https://moa.cms.waikato.ac.nz/streaming-random-patches/

Recurrent Concepts

- **Recurrent Concepts**: in presence of concept drifts, previous concepts can reoccur
  - Seasonal trends (daily/weekly/yearly)
  - We don’t want to learn from scratch every time
- **IDEA**: keep two sets of classifiers:
  - Library of available classifiers, currently inactive (past concepts)
  - Current ensemble of active models (current concepts)
  - **We need a policy to move the models from one set to the other**
    - in general, estimating the accuracy of a model on the current data requires much less samples than what we need for training

Recurrent Concept Drift (RCD)

IDEA: keep a model for each concept, identify drifts, reuse models

- **DDM for CD Detection**
  - REMINDER: DDM uses the errors to detect CD, assuming errors decrease over time for stationary data
- Train current model c_a and store training samples in buffer b_a
- **At WARNING level**
  - Start training a new model
  - Train new classifier c_n and c_a (old classifier) and store samples in b_n
- **IF DRIFT is confirmed**
  - Compare b_n with all the stored buffers to check if it is a new/old concept

Init current model and buffer

WARNING: prepare a new model

DRIFT: check if current concept is already known

Algorithm 1. RCD algorithm

Input: (m) Max buffer size, (α) Significance value, (e) Ensemble size

Data: (c_a) Actual classifier, (b_a) Actual buffer, (c_n) New classifier, (S) Data stream, (b_n) New buffer

Result: C: Classifiers list, B: Buffers list

```
begin
  C ← {Create(c_a)};
  B ← {Create(b_a)};
  for each s ∈ S do
    level ← DDM(c_a, s);
    switch level bf do
      case WARNING
        if c_n = null bf then
          Create(c_n);
          Create(b_n);
        end
        SaveFIFO (b_n, s, m);
        Train(c_n, s);
      case DRIFT
        if StatTest (C, B, b_n, α) then
          (c_a, b_a) ← Stored (C, B, b_n);
        end
      else
          SaveFIFO (C, c_n, e);
          SaveFIFO (B, b_n, m);
          c_a ← c_n;
          b_a ← b_n;
        c_n = b_n = ∅;
      otherwise
        ifc_n ≠ null then
          c_n = b_n = ∅;
        end
        SaveFIFO (b_a, s, m);
      end
    end
    Train (c_a,s);
  end
end
```

"RCD: A recurring concept drift framework» Pattern Recognition Letters

Conclusion

Take-Home Messages

- **Ensembles are one of the easiest and most reliable methods to improve the performance of an ML method**
  - As long as you can guarantee diversity
- **In the online world**:
  - Train ensembles on windows (AWE)
  - In nonstationary streams: each ensemble learns different concepts
  - Bagging -> online bagging

References

- Streaming Data Analytics Course - Emanuele Della Valle and Alessio Bernardo @ POLIMI
- MOA Book

Next Lecture

- Lab with river
  - Concept drift
  - Online classification
  - Ensembles