# Boosting

Christian Peters

January 29, 2021

# What you will know

$\rightarrow$ The idea behind boosting

# What you will know

$\rightarrow$ The idea behind boosting

$\rightarrow$ How to create a strong and efficient learning algorithm

# What you will know

$\rightarrow$ The idea behind boosting

$\rightarrow$ How to create a strong and efficient learning algorithm

$\rightarrow$ What is AdaBoost and why is it so successful?

# Let’s talk about training a model

# How to train a machine learning model

What we have learned so far...

- We have to pick a hypothesis class $\mathcal{H}$

# How to train a machine learning model

What we have learned so far...

- We have to pick a hypothesis class $\mathcal{H}$
- $\mathcal{H}$ can't be too complex (VC dim needs to be finite)

# How to train a machine learning model

What we have learned so far...

- We have to pick a hypothesis class $\mathcal{H}$
- $\mathcal{H}$ can't be too complex (VC dim needs to be finite)
- We need enough training data (more than some threshold $m_{\mathcal{H}}$)

# How to train a machine learning model

What we have learned so far...

- We have to pick a hypothesis class $\mathcal{H}$
- $\mathcal{H}$ can't be too complex (VC dim needs to be finite)
- We need enough training data (more than some threshold $m_{\mathcal{H}}$)
- Then we use ERM to pick the best $h \in \mathcal{H}$ that minimizes the empirical error

# How to train a machine learning model

What we have learned so far...

- We have to pick a hypothesis class $\mathcal{H}$
- $\mathcal{H}$ can't be too complex (VC dim needs to be finite)
- We need enough training data (more than some threshold $m_{\mathcal{H}}$)
- Then we use ERM to pick the best $h \in \mathcal{H}$ that minimizes the empirical error

But there is one problem...

# The problem with ERM

$\boxed{\text{ERM can be hard.}}$

# $\boxed{\text{ERM can be hard.}}$

- Depending on $\mathcal{H}$, the optimization problem can become arbitrarily complex

$$\boxed{\text{ERM can be hard.}}$$

- Depending on $\mathcal{H}$, the optimization problem can become arbitrarily complex
- e.g. implementing ERM for halfspaces in the non-separable case is computationally hard (chapter 9)

# ERM can be hard.

- Depending on $\mathcal{H}$, the optimization problem can become arbitrarily complex
- e.g. implementing ERM for halfspaces in the non-separable case is computationally hard (chapter 9)
- For many interesting classes, it is infeasible to implement ERM
    - Solving the optimization problem takes forever

# $\boxed{\text{ERM can be hard.}}$

- Depending on $\mathcal{H}$, the optimization problem can become arbitrarily complex
- e.g. implementing ERM for halfspaces in the non-separable case is computationally hard (chapter 9)
- For many interesting classes, it is infeasible to implement ERM
  - Solving the optimization problem takes forever

...so what can we do?

# A first idea...

**Idea:** Use simpler hypothesis classes where ERM isn't hard.

# A first idea...

Idea: Use simpler hypothesis classes where ERM isn't hard.

- Problem: Simple classes can be too ”weak” to estimate all relationships in the data
  - $\rightarrow$ Can lead to underfitting and poor performance

# A first idea...

**Idea:** Use simpler hypothesis classes where ERM isn't hard.

- Problem: Simple classes can be too ”weak” to estimate all relationships in the data
  - $\rightarrow$ Can lead to underfitting and poor performance
- Approximation error is high ($\rightarrow$ B/C tradeoff)

# A first idea...

Idea: Use simpler hypothesis classes where ERM isn't hard.

- Problem: Simple classes can be too ”weak” to estimate all relationships in the data
  - $\rightarrow$ Can lead to underfitting and poor performance
- Approximation error is high ($\rightarrow$ B/C tradeoff)
- Still, these classes can be useful for us
  - If the resulting hypothesis is at least better than random

# A first idea...

Idea: Use simpler hypothesis classes where ERM isn't hard.

- Problem: Simple classes can be too "weak" to estimate all relationships in the data
  - $\rightarrow$ Can lead to underfitting and poor performance
- Approximation error is high ($\rightarrow$ B/C tradeoff)
- Still, these classes can be useful for us
  - If the resulting hypothesis is at least better than random

Let's call ERM on a simple class a **weak learner**. We will formally define it later...

# The idea behind boosting

Why not combine many weak learners?
Can this give us an efficient strong learner?

# The idea behind boosting

Why not combine many weak learners?
Can this give us an efficient strong learner?

- This theoretical question is the origin of boosting

# The idea behind boosting

Why not combine many weak learners?
Can this give us an efficient strong learner?

- This theoretical question is the origin of boosting
- It was first raised in 1988 by Kearns and Valiant [2]

# The idea behind boosting

**Why not combine many weak learners?**
**Can this give us an efficient strong learner?**

- This theoretical question is the origin of boosting
- It was first raised in 1988 by Kearns and Valiant [2]
- The first (practical) answer was given in 1995 by Freund and Schapire [1]
  - $\rightarrow$ It is YES!

# The idea behind boosting

**Why not combine many weak learners?**
**Can this give us an efficient strong learner?**

- This theoretical question is the origin of boosting
- It was first raised in 1988 by Kearns and Valiant [2]
- The first (practical) answer was given in 1995 by Freund and Schapire [1]
  - $\rightarrow$ It is YES!
- The result is **AdaBoost**, a widely popular and award winning algorithm
  - We will take a look at this later...

# The idea behind boosting

Why not combine many weak learners?
Can this give us an efficient strong learner?

- This theoretical question is the origin of boosting
- It was first raised in 1988 by Kearns and Valiant [2]
- The first (practical) answer was given in 1995 by Freund and Schapire [1]
  - $\rightarrow$ It is YES!
- The result is **AdaBoost**, a widely popular and award winning algorithm
  - We will take a look at this later...

But first, let's get back to weak learning.

# Weak Learnability

# Weak Learnability

# AdaBoost

AdaBoost

# Conclusion

Conclusion

The End.

# References i


Y. Freund and R. E. Schapire.
**A decision-theoretic generalization of on-line learning and an application to boosting.**
*Journal of Computer and System Sciences*, 55(1):119 – 139, 1997.

M. Kearns and L. G. Valiant.
**Learning boolean formulae or finite automata is as hard as factoring.**
Technical Report TR 14-88, Harvard University Aiken Computation Laboratory, 1988.

S. Shalev-Shwartz and S. Ben-David.
*Understanding Machine Learning - From Theory to Algorithms.*
Cambridge University Press, 2014.