Boosting

Gradient Boosting

@cambridgespark

CAMBRIDGE SPARK

Boosting

CAMBRIDGE SPARK

Boosting - Intuition

A boosting classifier is an ensemble classifier built **iteratively**.

Start with an ensemble **H** of a single base classifier $\mathbf{h}_1$

Boosting - Intuition

A boosting classifier is an ensemble classifier built **iteratively**.

Start with an ensemble **H** of a single base classifier **h** $_1$

1. Create a **new dataset** emphasising items **misclassified** by **H**

Boosting - Intuition

A boosting classifier is an ensemble classifier built **iteratively**.

Start with an ensemble **H** of a single base classifier $\mathbf{h}_{1}$

1. Create a **new dataset** emphasising items **misclassified** by **H**

2. Fit a classifier $\mathbf{h}_{k+1}$ and **integrate** it into **H**

CAMBRIDGE SPARK

Boosting - Intuition

A boosting classifier is an ensemble classifier built **iteratively**.

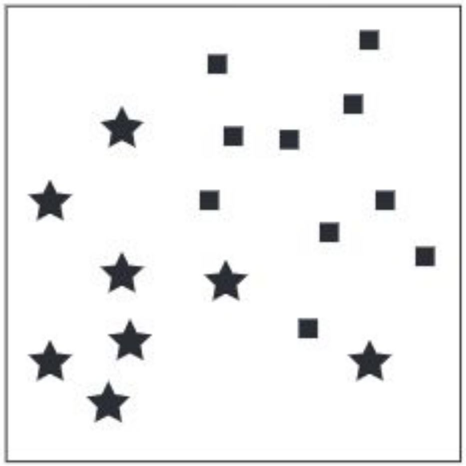
Start with an ensemble **H** of a single base classifier $\mathbf{h}_{1}$

1. Create a **new dataset** emphasising items **misclassified** by **H**
2. Fit a classifier $\mathbf{h}_{k+1}$ and **integrate** it into **H**
3. Repeat

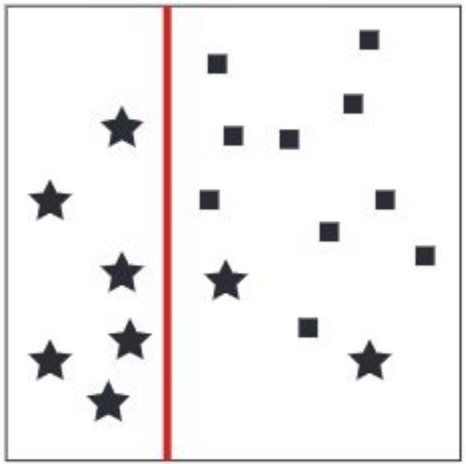
We fit classifiers that increasingly and actively focus on "hard points"

CAMBRIDGE SPARK

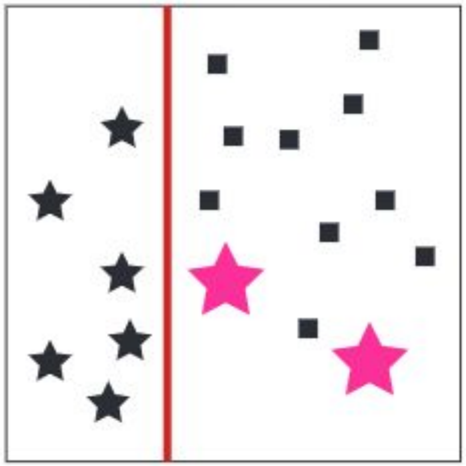
Visually



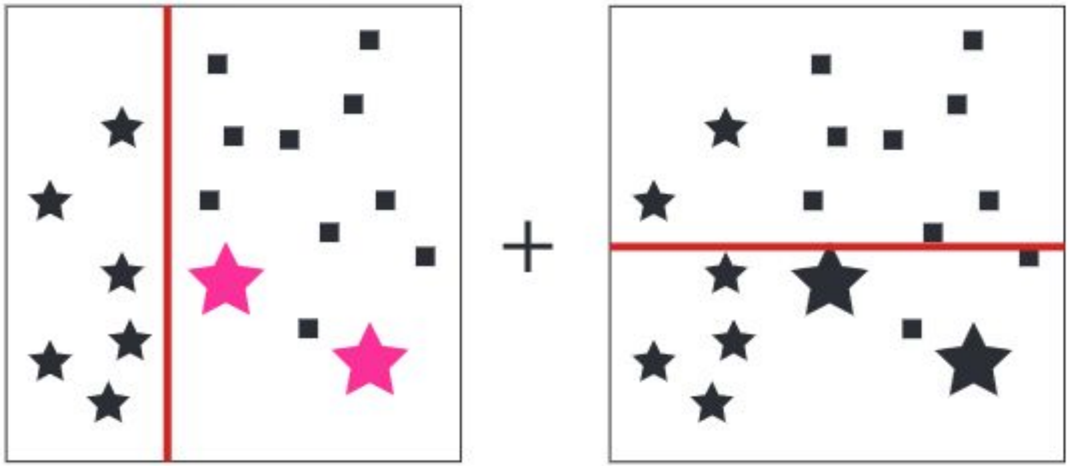
Visually



Visually



Visually



Comments on boosting

- Sequential algorithm

Comments on boosting

- Sequential algorithm
- Later classifiers have decreasing weight in the aggregation
  - weight of $\mathbf{h}_{\mathbf{k}}$ > weight of $\mathbf{h}_{\mathbf{k+1}}$

CAMBRIDGE SPARK

Comments on boosting

- Sequential algorithm
- Later classifiers have decreasing weight in the aggregation
  - weight of $\mathbf{h}_{\mathbf{k}}$ > weight of $\mathbf{h}_{\mathbf{k+1}}$
- The same principles work for regression

Boosting methods

- What "weights" to use for misclassified points?

Boosting methods

- What "weights" to use for misclassified points?
- What "weights" to use for the sequence of classifiers?

Boosting methods

- What "weights" to use for misclassified points?
- What "weights" to use for the sequence of classifiers?

Many different approaches based on answers.

AdaBoost ('96), LogitBoost ('98), BrownBoost ('01), . . .

Boosting methods

- What "weights" to use for misclassified points?
- What "weights" to use for the sequence of classifiers?

Many different approaches based on answers.

AdaBoost ('96), LogitBoost ('98), BrownBoost ('01), . . .

Boosting methods can be interpreted as a **gradient descent** which leads to a more generic framework (" AnyBoost ") and to extremely popular libraries:

XGBoost ('14), LightGBM ('16), CatBoost ('17)

CAMBRIDGE SPARK

Gradient Boosting

Gradient Boosting - Intuition

We want to build weak learners that **actively compensate** each others' errors.

Gradient Boosting - Intuition

We want to build weak learners that **actively compensate** each others’ errors.

Let’s focus on one sample: **(X, y)** with $y = 100$

Gradient Boosting - Intuition

We want to build weak learners that **actively compensate** each others' errors.

Let's focus on one sample: **(X, y)** with $y = 100$

1. Train DT1 on **(X, y)** $DT1(X) = 95$

Gradient Boosting - Intuition

We want to build weak learners that **actively compensate** each others' errors.

Let's focus on one sample: **(X, y)** with $y = 100$

1. Train DT1 on **(X**, **y)** — $DT1(X) = 95$
2. Compute residuals — $r = 100 - 95 = 5$

Gradient Boosting - Intuition

We want to build weak learners that **actively compensate** each others' errors.

Let's focus on one sample: **(X, y)** with $y = 100$

1. Train DT1 on **(X, y)** — $DT1(X) = 95$
2. Compute residuals — $r = 100 - 95 = 5$
3. Train DT2 on **(X, 5)** — $DT2(X) = 4$

CAMBRIDGE SPARK

Gradient Boosting - Intuition

We want to build weak learners that **actively compensate** each others' errors.

Let's focus on one sample: **(X, y)** with $y = 100$

1. Train DT1 on **(X, y)** — $DT1(X) = 95$
2. Compute residuals — $r = 100 - 95 = 5$
3. Train DT2 on **(X, 5)** — $DT2(X) = 4$
4. Aggregate DT1 and DT2 — $DT1(X) + DT2(X) = 95 + 4 = 99$

Gradient Boosting - Intuition

We want to build weak learners that **actively compensate** each others' errors.

Let's focus on one sample: **(X, y)** with $y = 100$

1. Train DT1 on **(X**, **y)** — $DT1(X) = 95$
2. Compute residuals — $r = 100 - 95 = 5$
3. Train DT2 on **(X, 5)** — $DT2(X) = 4$
4. Aggregate DT1 and DT2 — $DT1(X) + DT2(X) = 95 + 4 = 99$
5. Repeat

CAMBRIDGE SPARK

Boosting - Gradient Boosting

too many stages **OR** too complex trees = overfit to noise

Boosting - Gradient Boosting

too many stages **OR** too complex trees = overfit to noise

- Getting the number of stages right is **extremely** important

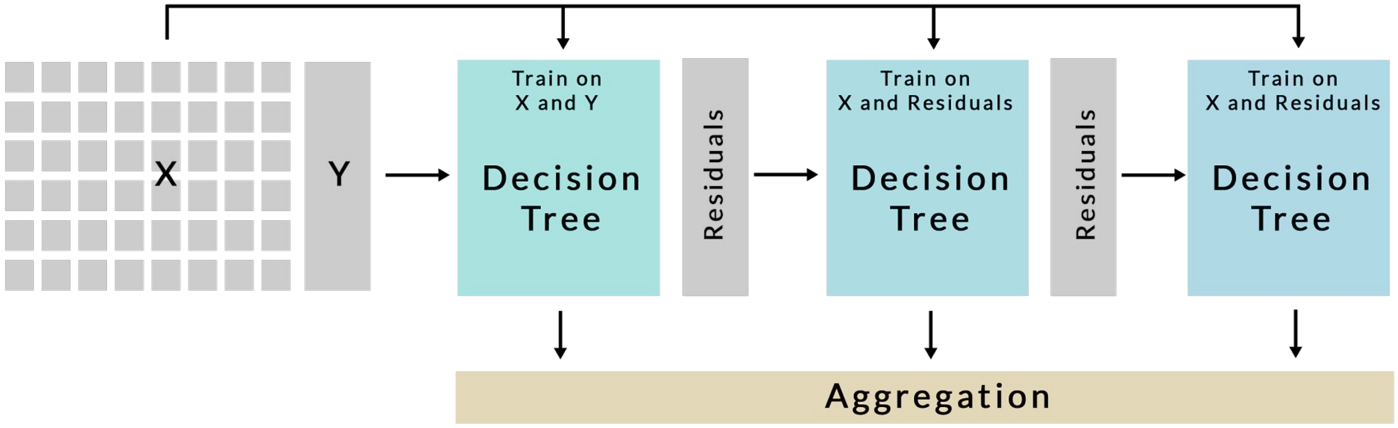
CAMBRIDGE SPARK

Boosting - Gradient Boosting

too many stages **OR** too complex trees = overfit to noise

- Getting the number of stages right is **extremely** important
- We need to build **small**, **constrained** trees

Boosting - Gradient Boosting



CAMBRIDGE SPARK

Some pros and cons

+ Great **performance** (usually)

Some pros and cons

+ Great **performance** (usually)

+ Decision Trees = we can get **feature importance**

Some pros and cons

+ Great **performance** (usually)
+ Decision Trees = we can get **feature importance**
− Hard to run in **parallel**

Some pros and cons

+ Great **performance** (usually)
+ Decision Trees = we can get **feature importance**
− Hard to run in **parallel**
− Hard to **interpret**

Some pros and cons

+ Great **performance** (usually)
+ Decision Trees = we can get **feature importance**
− Hard to run in **parallel**
− Hard to **interpret**
− Can easily **overfit**

Gradient Boosting

In Practice

CAMBRIDGE SPARK

Gradient Boosting in practice

A lot of things depend on the family of base models and can be optimised a lot.

XGBoost and **LightGBM** essentially consider decision trees of fixed depth . Then a lot of tricks are applied:

Gradient Boosting in practice

A lot of things depend on the family of base models and can be optimised a lot.

XGBoost and **LightGBM** essentially consider decision trees of fixed depth . Then a lot of tricks are applied:

- (approximate) computation of residuals (fast)
- Sophisticated computations of γ_k (“step size” or “learning rate”)
- Shrinkage methods to avoid "bad" trees (regularisation)
- Lots of software optimisation. . . (and open source!)

CAMBRIDGE SPARK

Boosting in Python

With Scikit-Learn:

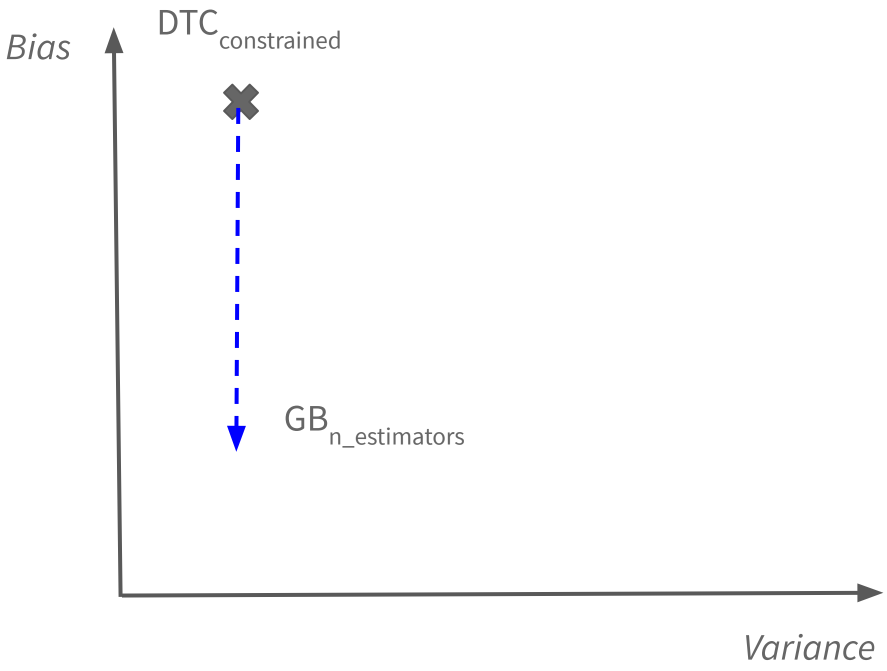
- *AdaBoostClassifier*, *AdaBoostRegressor*
- *GradientBoostingClassifier*, *GradientBoostingRegressor*
  - not as optimised as XGB or LGBM but work with any estimator, not just trees

Other gradient boosting libraries include:

- *XGBoost*
- *LightGBM*
- *CatBoost*

CAMBRIDGE SPARK

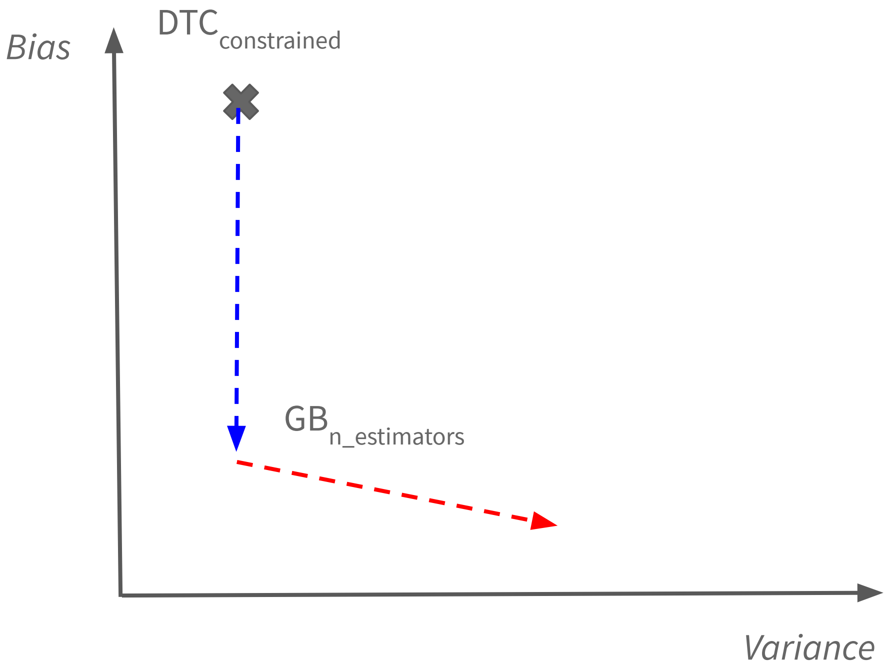
Bias vs Variance



- Gradient Boosting allows to reduce bias of shallow trees
- What happens if we keep adding trees?

CAMBRIDGE SPARK

Bias vs Variance



- Gradient Boosting allows to reduce bias of shallow trees
- What happens if we keep adding trees?
    - We'll start overfitting

CAMBRIDGE SPARK

Hands-on session

01-boosting.ipynb

CAMBRIDGE SPARK