CIVIL-557

Decision-aid methodologies in transportation

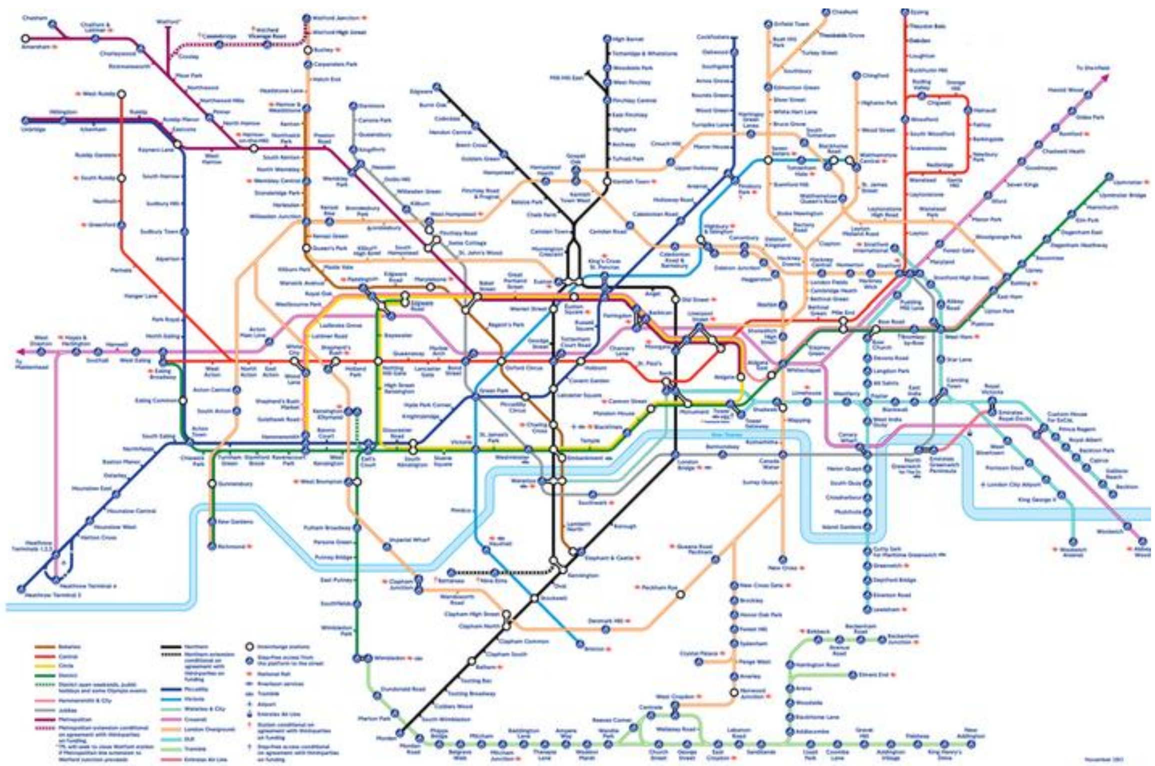
Lecture 4: Ensemble methods and hyperparameter search

Tim Hillel

Transport and Mobility Laboratory TRANSP-OR
École Polytechnique Fédérale de Lausanne EPFL

Case study

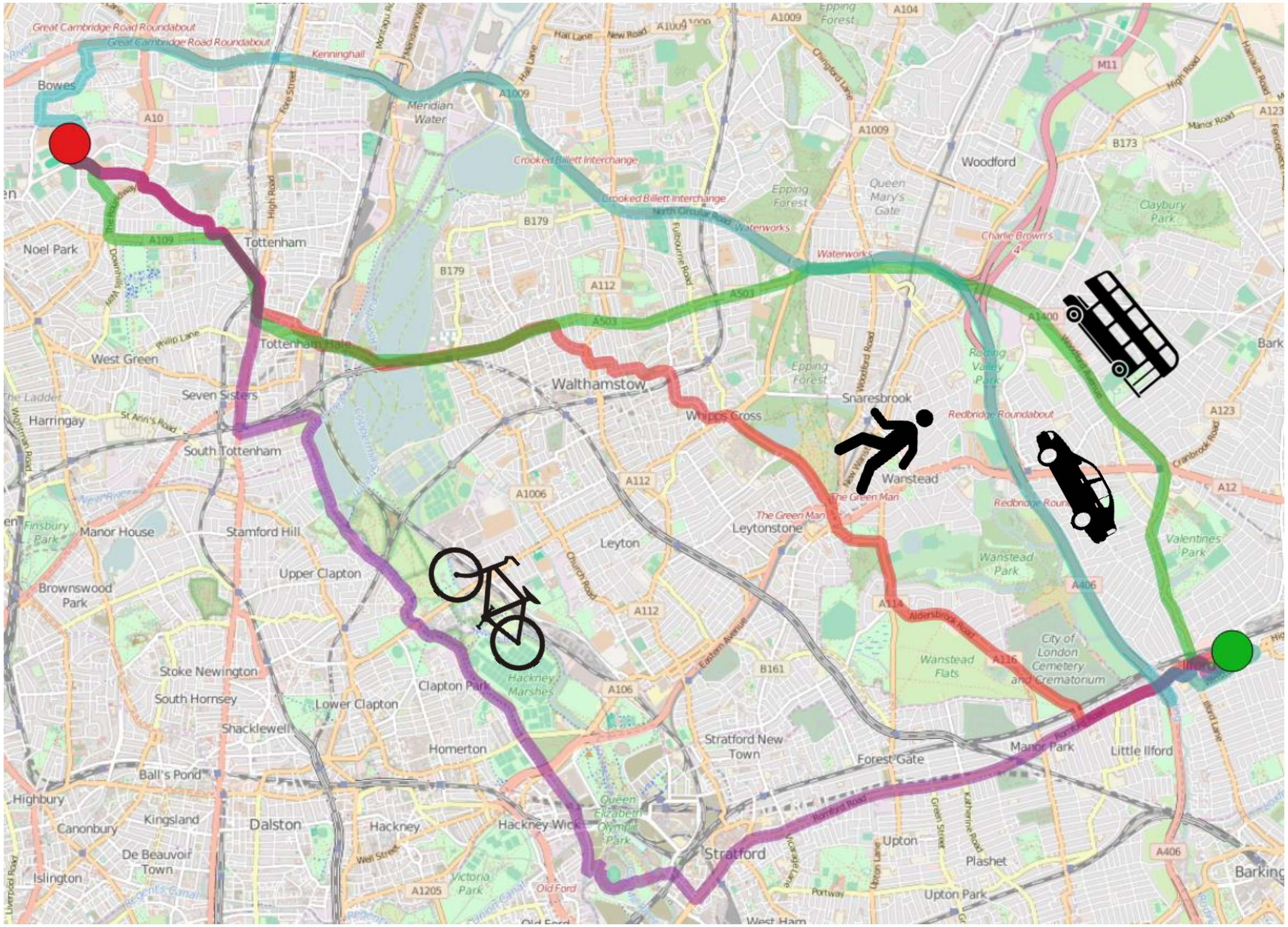
Transport for London



TRANSP-OR

EPFL

Mode choice



TRANSP-OR

EPFL

Last week

- Feature processing
  - Missing values
  - Categorical variables

- Theory of probabilistic classification

- Probabilistic metrics

- Probabilistic classifiers
  - Logistic regression

Today

1. Mid-term feedback
2. Logistic regression recap/feedback
3. Ensemble method theory
4. Hyperparameter selection theory
5. Practical class work

TRANSP-OR

EPFL

Mid-term feedback

- Overall, well done!

- 3 problem questions
  - Dictionary comprehension
  - k-Nearest Neighbours
  - Sampling bias

TRANSP-OR

EPFL

Logistic regression

- Probabilistic classifier:

$$f(x) = \sigma(\sum_{k=1}^{K} \beta_k x_k + \beta_o)$$

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{J} e^{z_i}}$$

- Evaluate with cross-entropy loss (CEL)

Logistic regression homework

Discussion of worked example

Ensemble methods

- Wisdom of crowds

Wisdom of crowds

- You go to see two doctors about some worrying symptoms
- Both doctors say the symptoms are nothing to worry about
  - Doctor A is right 60% of the time
  - Doctor B is right 75% of the time

How confident are you?

TRANSP-OR

EPFL

Wisdom of crowds

- Case 1
  - Doctors went to same medical school
  - Used same tests/information/questions
  - Doctors guesses are correlated (identical mistakes)
- Case 2
  - Doctors went to different medical schools
  - Use different tests/info/questions
  - Doctors guesses are independent

Ensemble learning

- Feature vector $\boldsymbol{x}$
- Individual model (classifier/regressor):

$$\hat{y} = h(\boldsymbol{x})$$

- Set of *weak learners:*

$$D = \{h_1, \dots, h_T\}$$

- Prediction of ensemble:

$$\hat{y} = H(h_1, \dots, h_T)$$

- H is *aggregation function*

Weak learners

- To benefit from wisdom of crowds, ensemble must contain **weak** and **diverse learners** $h_i(x)$
  - Weak learners - need to be better than random guessing (more right than wrong!)
  - Diverse learners – make mistakes independently on different data points

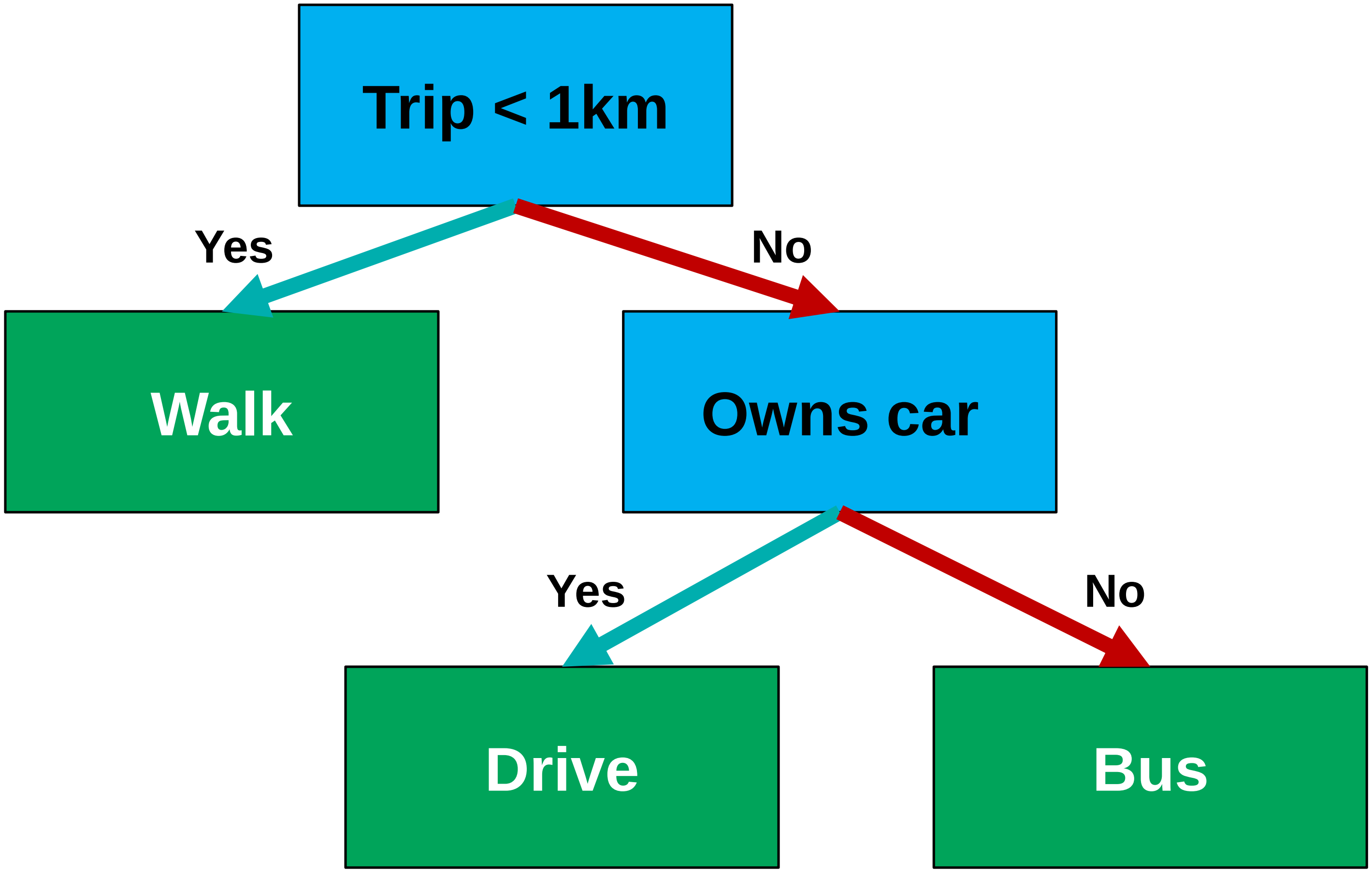
Ensemble learning classifiers

- Three questions:
  - Which classifiers are suitable for weak learners?
  - How to make them diverse?
  - How to aggregate classifiers?

Classifiers

- Computationally simple to fit/predict with
- High variance – enables diversity in classifiers

Decision trees



Ensuring diversity

- Two approaches (meta-algorithms):

  1. Bagging (Bootstrap Aggregating)

     Random Forest

  2. Boosting

     Gradient Boosting

The bootstrap

Sample:

0, **1**, 2, 3, 4, 5, 6, 7, 8, 9

Bootstrap:

0, 7, 4, 6, 1, 7, 9, 5, 1, **1**

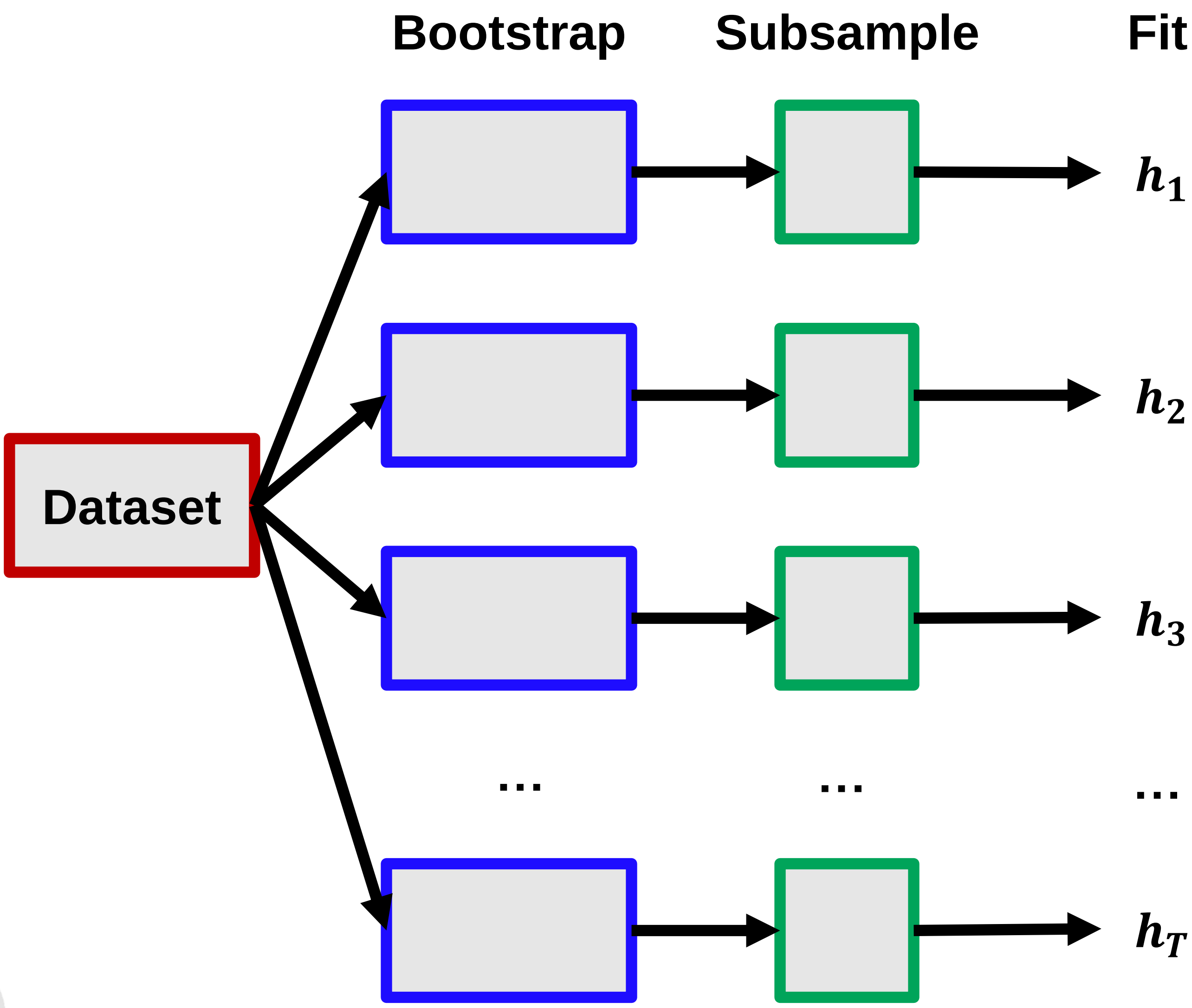
Random forest

- **Bootstrapping:** Create statistically similar versions of the dataset by sampling observations **(rows)** with replacement

- **Subsampling:** Randomly subsample features **(columns)**

Fit classifier on each subsampled bootstrap

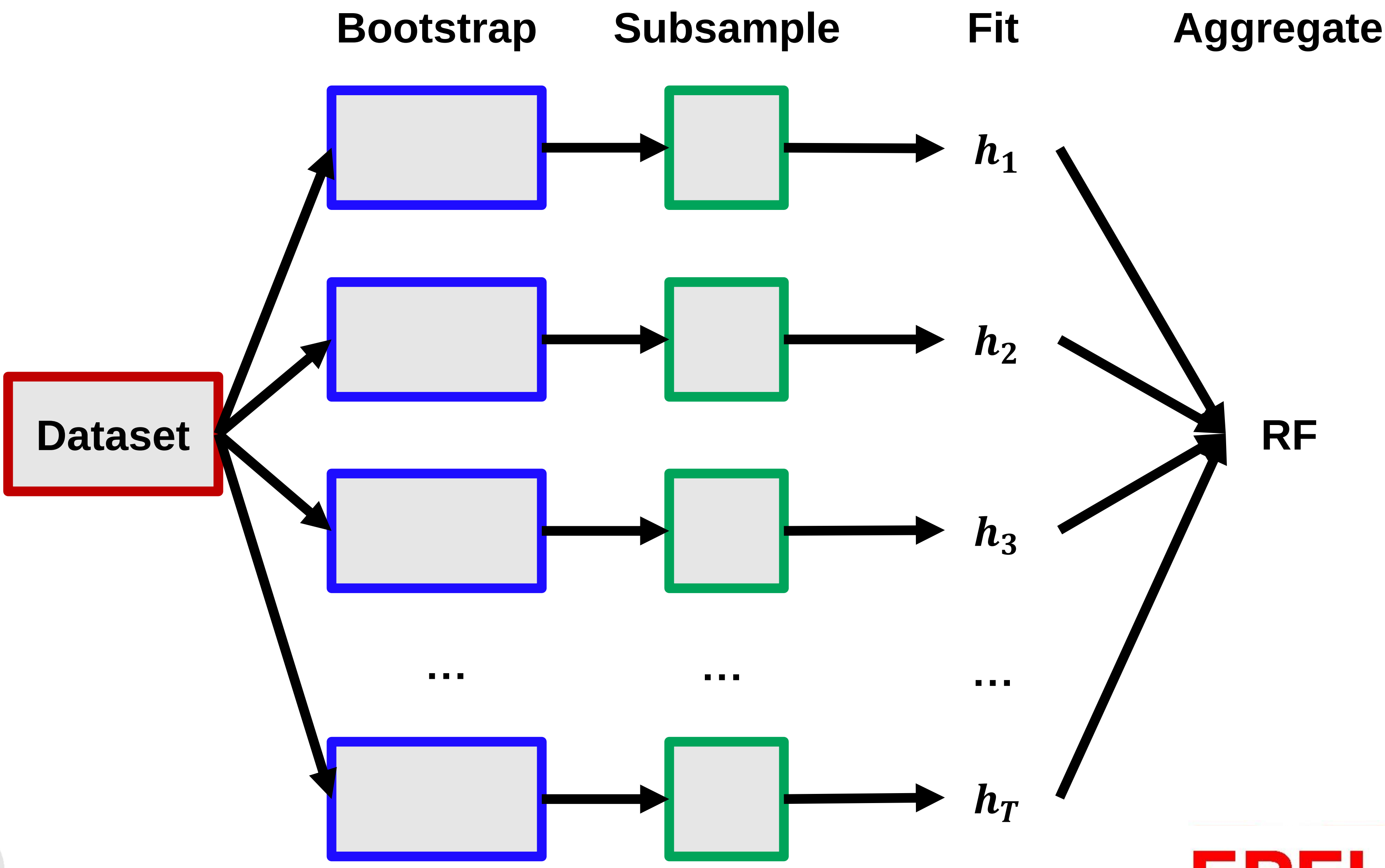
Random forest



Aggregation

- Discrete classification: majority vote
- Probabilistic classification: proportion of each class (need large ensemble for reliable values)

Random forest



Random forest

- Trivial to parallelise
- Can model complex non-linear relationships
- Can measure *feature importance*
  - Total gain of each feature over each split

- However, more difficult to interpret than single DT

When to stop splitting

- Maximum depth
- Minimum leaf node size
- Minimum split size
- Minimum gain from split
- Maximum number of leaf nodes
- Etc…

Also

- Number of trees (bootstrap samples)
- Subsample rate
  - Per tree
  - Per level in tree

Even more hyperparameters!

Hands on

Notebook 1: Random forests

TRANSP-OR

EPFL

Gradient boosting

- Start with **regression**

- Train sequential trees on residuals from previous guesses

- Regularise using learning rate α

Gradient boosting

Fit | Fit | Fit

Dataset → h_1 → h_2 → h_3

$$r_1 = y - \alpha\hat{y}_1$$

$$r_2 = r_1 - \alpha\hat{y}_2$$

Residual | Residual

Gradient boosting

Fit

Fit

Dataset → h_1

h_2

h_{T-1}

h_T

$r_1 = y - \alpha\hat{y}_1$

$r_2 = r_1 - \alpha\hat{y}_2$

$r_{T-1} = r_{T-2} - \alpha\hat{y}_{T-1}$

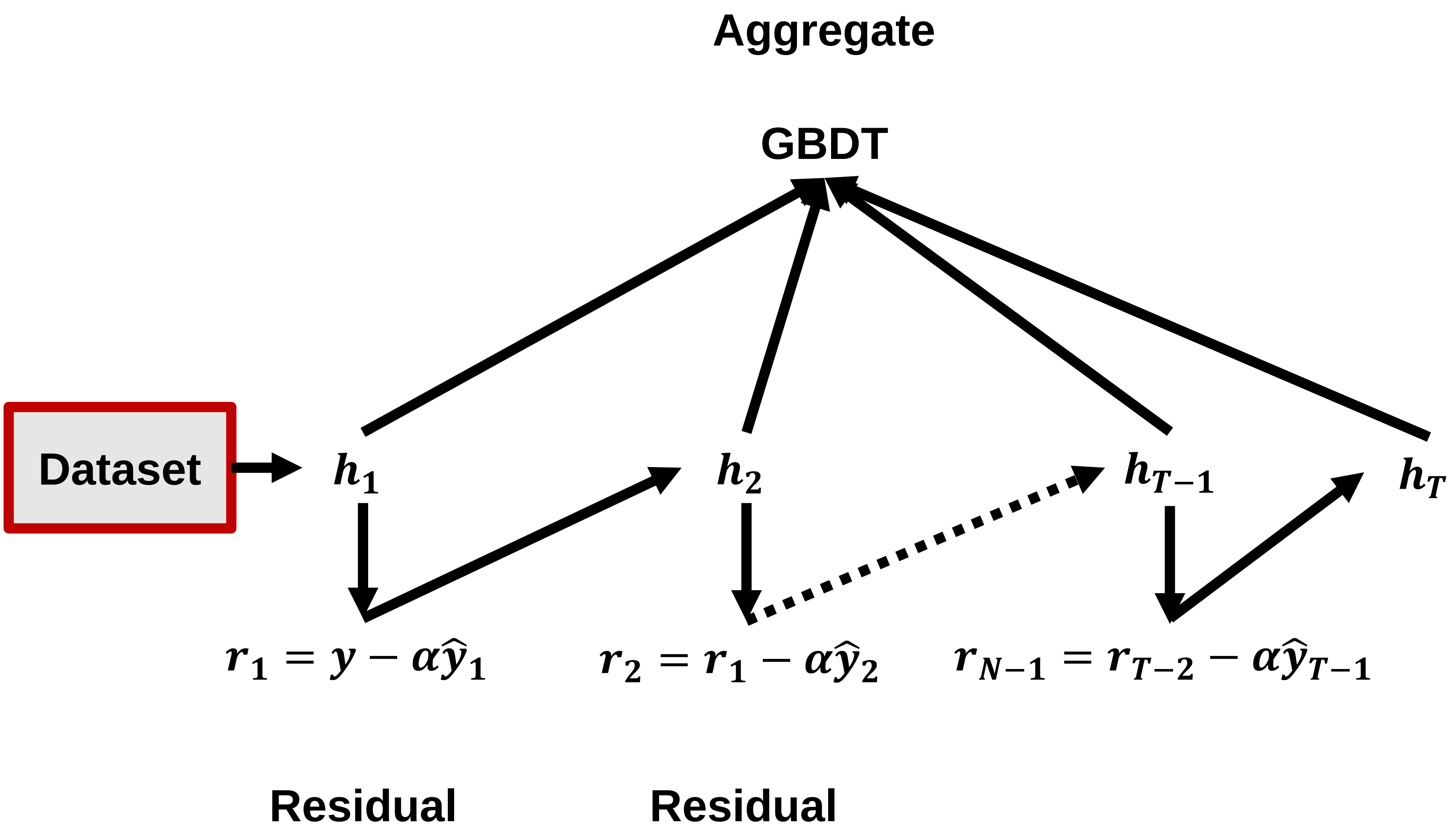
Residual

Residual

Aggregate

- Sum up the predictions of the decision trees (multiplied by learning rate!)

Gradient boosting



Classification

- So far only considered regression

- How to turn regression values into probabilities?

Softmax!

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{J} e^{z_i}}$$

Also

- Number of trees (boosting iterations)
- Learning rate
- Subsample rate
  - Per tree
  - Per level in tree

Even more hyperparameters!

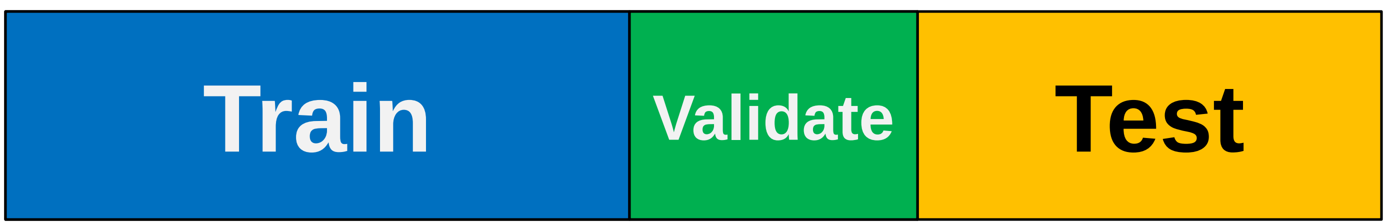
Hands on

Notebook 2: Gradient boosting

Hyperparameter selection

- How to select optimal hyperparameters for model algorithm?

- Previously used train-validate-test split with trial and error



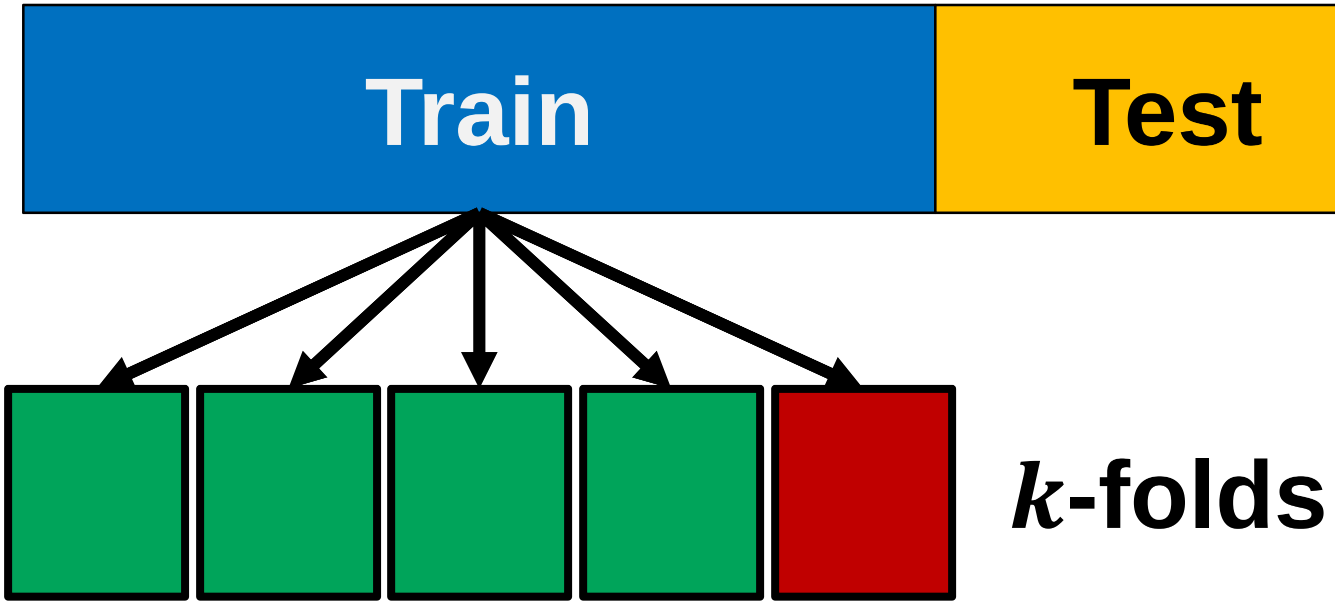
Hyperparameter selection

- Test set must be kept separate, leaving finite data for train-validation
  - Should represent external validation where possible

- Is there a better way of evaluating model performance on finite data?

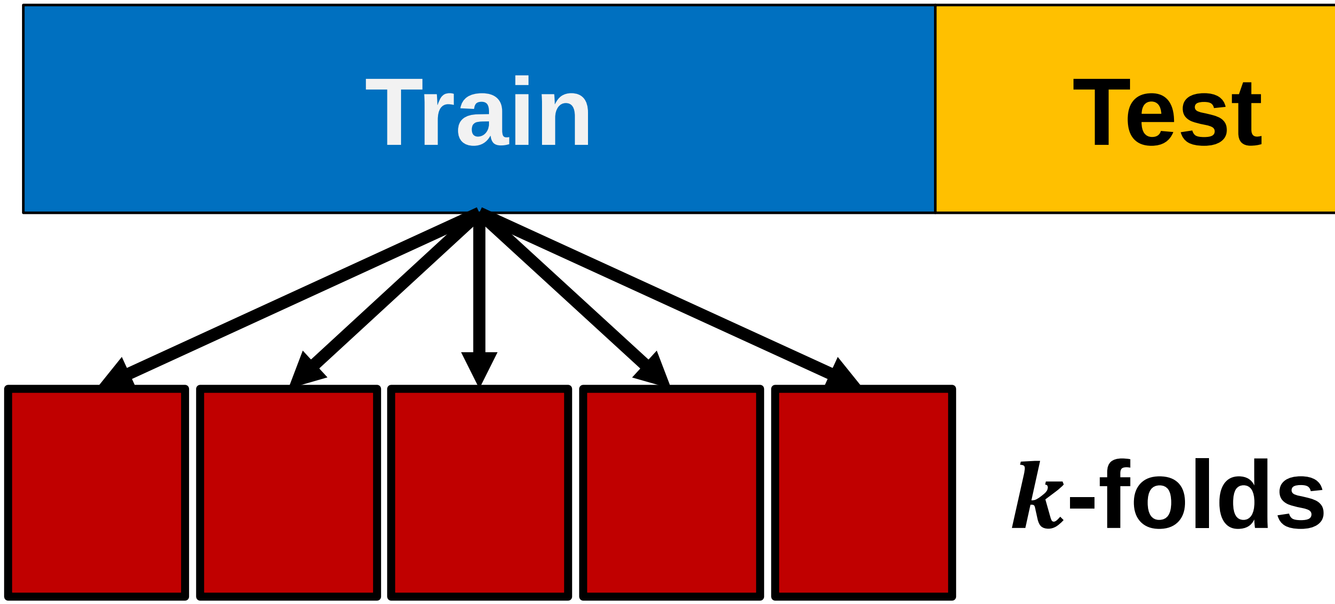
Yes!

Cross-validation



Train on $k-1$ folds **Test on remaining fold**

Cross-validation



Average k scores

Grid-search

- Predefine search values for each hyperparameter
- Search all combinations in exhaustive grid-search

- Simple to understand, implement, and parallelise

- Inefficient:
  - Lots of time evaluating options which are likely to be low performing
  - Few unique values for each hyperparameter tested

Early-stopping

- Gradient boosting
  - Most important hyperparameters are *learning rate* and *number of boosting rounds*
  - These hyperparameters are linked
  - Sequential model – number of boosting rounds can be set heuristically
- Early stopping
  - Fix learning rate < 0.1 (e.g. 0.01)
  - Perform boosting until performance does not increase for n iterations

Homework

Notebook 3: Hyperparameter search