COMS30035, Machine learning: Combining Models 1, Selecting and Combining

Edwin Simpson
edwin.simpson@bristol.ac.uk

Department of Computer Science, SCEEM
University of Bristol

November 8, 2023

Agenda

- Model Selection
- Model Averaging
- Ensembles: Bagging
- Ensembles: Boosting and Stacking
- Tree-based Models
- Conditional Mixture Models
- Ensembles of Humans

Textbook

We will follow Chapter 14 of the Bishop book: Bishop, C. M., Pattern recognition and machine learning (2006). Available for free here.

The Model Selection Problem

- Select a model h from a set of possible models in a set H,

The Model Selection Problem

- Select a model h from a set of possible models in a set H,
- The models in H can differ in various ways, such as:

The Model Selection Problem

- Select a model h from a set of possible models in a set H,
- The models in H can differ in various ways, such as:
  - The structure of the model, e.g., differences in graphical models;

The Model Selection Problem

- Select a model h from a set of possible models in a set H,
- The models in H can differ in various ways, such as:
  - The structure of the model, e.g., differences in graphical models;
  - The choice of prior distribution $p(\theta)$ over model parameters;

The Model Selection Problem

- Select a model h from a set of possible models in a set H,
- The models in H can differ in various ways, such as:
  - The structure of the model, e.g., differences in graphical models;
  - The choice of prior distribution $p(\theta)$ over model parameters;
  - The learning algorithm used, e.g., EM, MCMC, backpropagation,;
  - Parameters of the learning algorithm, like learning rate for stochastic gradient descent (SGD);

The Model Selection Problem

- Select a model h from a set of possible models in a set H,
- The models in H can differ in various ways, such as:
  - The structure of the model, e.g., differences in graphical models;
  - The choice of prior distribution $p(\theta)$ over model parameters;
  - The learning algorithm used, e.g., EM, MCMC, backpropagation,;
  - Parameters of the learning algorithm, like learning rate for stochastic gradient descent (SGD);
  - The features of the data used as inputs to the model;

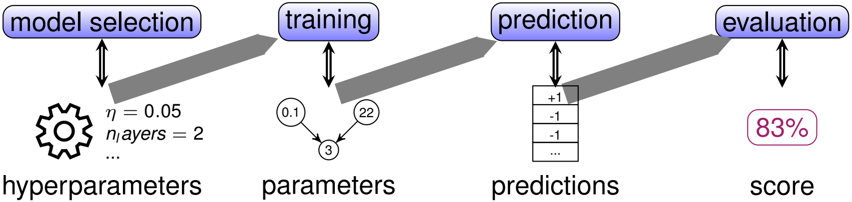
The Model Selection Problem

- Select a model h from a set of possible models in a set H,
- The models in H can differ in various ways, such as:
  - The structure of the model, e.g., differences in graphical models;
  - The choice of prior distribution $p(\theta)$ over model parameters;
  - The learning algorithm used, e.g., EM, MCMC, backpropagation,;
  - Parameters of the learning algorithm, like learning rate for stochastic gradient descent (SGD);
  - The features of the data used as inputs to the model;
  - The examples in the dataset the model is trained on;

The Model Selection Problem

- Select a model h from a set of possible models in a set H,
- The models in H can differ in various ways, such as:
  - The structure of the model, e.g., differences in graphical models;
  - The choice of prior distribution $p(\theta)$ over model parameters;
  - The learning algorithm used, e.g., EM, MCMC, backpropagation,;
  - Parameters of the learning algorithm, like learning rate for stochastic gradient descent (SGD);
  - The features of the data used as inputs to the model;
  - The examples in the dataset the model is trained on;
  - Random initialisation of parameters for EM or SGD

Hyperparameters



- It's useful to characterise all of these modelling decisions as *hyperparameters*
- Hyperparameters = all modelling choices that are fixed before training

Model Selection on a Validation Set

- Train a set of models H with different hyperparameters
- Choose the model h that maximises performance on a validation set.
  - Can't tune on the training set as it would lead to overfitting

Model Selection on a Validation Set

Dataset:	Train	Validation /Development	Test

- Train a set of models H with different hyperparameters
- Choose the model h that maximises performance on a validation set.
  - Can't tune on the training set as it would lead to overfitting
- Strength: optimises a performance metric we really care about.

Model Selection on a Validation Set

Dataset:	Train	Validation /Development	Test

- Train a set of models *H* with different hyperparameters
- Choose the model *h* that maximises performance on a validation set.
  - Can't tune on the training set as it would lead to overfitting
- Strength: optimises a performance metric we really care about.
- Weakness: we didn't use all the data in training;

Model Selection on a Validation Set

- Train a set of models H with different hyperparameters
- Choose the model h that maximises performance on a validation set.
  - Can't tune on the training set as it would lead to overfitting
- Strength: optimises a performance metric we really care about.
- Weakness: we didn't use all the data in training;
- Weakness: if the validation set is small, we might choose the wrong model!

Model Selection on a Validation Set

Dataset:

- Train a set of models *H* with different hyperparameters
- Choose the model *h* that maximises performance on a validation set.

Model Selection on a Validation Set

Dataset:	Train	Validation /Development	Test

- Train a set of models H with different hyperparameters
- Choose the model h that maximises performance on a validation set.
- How do we come up with H?

Model Selection on a Validation Set

Dataset: Train | Validation /Development | Test

- Train a set of models H with different hyperparameters
- Choose the model h that maximises performance on a validation set.
- How do we come up with H?
- Random search: test random combinations of hyperparameters

Model Selection on a Validation Set

- Train a set of models *H* with different hyperparameters
- Choose the model *h* that maximises performance on a validation set.
- How do we come up with *H*?
- Random search: test random combinations of hyperparameters
- Grid search: For each hyperparameter, define a set of values to test
  - Use your knowledge of the problem to test only reasonable values
  - For numerical hyperparameters, e.g., learning rate, choose a set of evenly-spaced values within a sensible range
  - *H* contains all combinations of the chosen hyperparameter values

Edwin Simpson
edwin.simpson@bristol.ac.uk

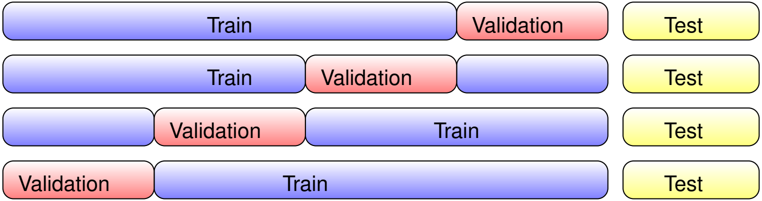
Model Selection on a Validation Set

Dataset:	Train	Validation /Development	Test

- Train a set of models H with different hyperparameters
- Choose the model h that maximises performance on a validation set.
- How do we come up with H?
- Random search: test random combinations of hyperparameters
- Grid search: For each hyperparameter, define a set of values to test
  - Use your knowledge of the problem to test only reasonable values
  - For numerical hyperparameters, e.g., learning rate, choose a set of evenly-spaced values within a sensible range
  - H contains all combinations of the chosen hyperparameter values
- Reduce the number of tests needed to find a good combination using a more intelligent strategy such as Bayesian Optimisation

Edwin Simpson
edwin.simpson@bristol.ac.uk

Cross-validation



- Split the training data into k random, equally-sized subsets;
- For each of the k folds: leave out the kth subset from training, train on the rest and test on the kth subset;
- Compute the average performance across all k folds;
- Avoids overfitting by tuning on training set performance...
- And avoids tuning on a single small validation set.

A Probabilistic View of Model Selection

- Suppose we have the following machine learning task:
  - Latent variables to predict (e.g., class labels in the test set): $\boldsymbol{z}$;
  - Observed data: $\boldsymbol{X}$;
  - Model: h.
- We obtain the prediction of $\boldsymbol{z}$ from a chosen model h as follows:

$$p(\boldsymbol{z}|\boldsymbol{X}) \approx p(\boldsymbol{z}|\boldsymbol{X}, h) \tag{1}$$

Bayesian Model Selection

- How can we choose h in $p(\boldsymbol{z}|\boldsymbol{X}, h)$?
- Choose $h = h^*$ to *maximise* the marginal likelihood of the data:

$$h^* = \underset{h}{\operatorname{argmax}} p(\boldsymbol{X}|h) = \underset{h}{\operatorname{argmax}} \int p(\boldsymbol{X}|\boldsymbol{\theta}, h) p(\boldsymbol{\theta}|h) d\boldsymbol{\theta} \tag{2}$$

- Similar to maximum likelihood estimation, which we used before to optimise parameters $\boldsymbol{\theta}$.
  - Here, we use a Bayesian approach and integrate out (marginalise) $\boldsymbol{\theta}$.
  - Relies on finding a single, good model given our training set.

Bayesian Model Averaging (BMA)

- Even after computing marginal likelihood, we may be uncertain about which model h is correct
- We can express this by assigning a probability to each model given the training data, $p(h|\boldsymbol{X})$.

Bayesian Model Averaging (BMA)

- Rather than choosing a single model, we can now take an expectation.
- Our predictions now come from a *weighted sum* over models, where $p(h|\boldsymbol{X})$ are weights :

$$p(\boldsymbol{z}|\boldsymbol{X}) = \sum_{h=1}^{H} p(\boldsymbol{z}|\boldsymbol{X}, h)p(h|\boldsymbol{X}) \tag{3}$$

Bayesian Model Averaging (BMA)

- Apply Bayes' rule to estimate the weights:

$$p(h|\boldsymbol{X}) = \frac{p(\boldsymbol{X}|h)p(h)}{\sum_{h'=1}^{H} p(\boldsymbol{X}|h)p(h')} \quad (4)$$

- What happens as we increase the amount of data in $\boldsymbol{X}$? $p(h|\boldsymbol{X})$ becomes more focussed on one model.
- So BMA is soft model selection, it does not *combine* models to make a more powerful model.

Ensemble Methods

Wisdom of the crowd

Guess the weight!

Wisdom of the crowd

Guess the weight!
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.

Wisdom of the crowd

Guess the weight!
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.
The combination was more effective than any one ‘model’.

Ensemble Methods

- Ensemble: a combination of different models.
- Often outperforms the average individual, and sometimes even the best individual.
- Different principle to BMA:
  - BMA weights try to identify a single, correct model
  - BMA weights do not provide the optimal combination

Expected Error of an Ensemble

- Given a set of models, $1, ..., M$,
- $y_m(\boldsymbol{x})$ is the prediction from model m.
- Simple ensemble: the mean of the individual predictions, $y_{COM} = \frac{1}{M}\sum_{m=1}^{M} y_m(\boldsymbol{x})$,

Expected Error of an Ensemble

- Given a set of models, $1, ..., M$,
- $y_m(\boldsymbol{x})$ is the prediction from model m.
- Simple ensemble: the mean of the individual predictions, $y_{COM} = \frac{1}{M}\sum_{m=1}^{M} y_m(\boldsymbol{x})$,
- Let's compare the sum-of-squares error of y_{COM} with that of the individual models...

Expected Error of an Ensemble

Firstly, the error of our combination for a particular input $\boldsymbol{x}$ is:

$$(y(\boldsymbol{x}) - y_{COM}(\boldsymbol{x}))^2 = \left(\frac{1}{M} \sum_{m=1}^{M} y(\boldsymbol{x}) - y_m(\boldsymbol{x}) \right)^2 . \tag{5}$$

Expected Error of an Ensemble

Firstly, the expected error of our combination is:

$$E_{COM} = \mathbb{E}_{\boldsymbol{x}}[(y(\boldsymbol{x}) - y_{COM}(\boldsymbol{x}))^2] = \mathbb{E}_{\boldsymbol{x}}\left[\left(\frac{1}{M}\sum_{m=1}^{M}(y(\boldsymbol{x}) - y_m(\boldsymbol{x}))\right)^2\right]. \quad (6)$$

Expected Error of an Ensemble

- The expected error of an individual model m is: $E_m = \mathbb{E}_{\boldsymbol{x}}[(y(\boldsymbol{x}) - y_m(\boldsymbol{x}))^2]$.

Expected Error of an Ensemble

- The **average** expected error of an individual model is: $E_{AV} = \frac{1}{M}\sum_{m=1}^{M} \mathbb{E}_{\boldsymbol{x}}\left[(y(\boldsymbol{x}) - y_m(\boldsymbol{x}))^2\right]$.
- If we make two assumptions...
  1. The errors of each model have zero mean;
  2. The errors of different models are not correlated;

Expected Error of an Ensemble

- The **average** expected error of an individual model is: $E_{AV} = \frac{1}{M}\sum_{m=1}^{M} \mathbb{E}_{\boldsymbol{x}}\left[(y(\boldsymbol{x}) - y_m(\boldsymbol{x}))^2\right]$.
- If we make two assumptions...
  1. The errors of each model have zero mean;
  2. The errors of different models are not correlated;
- Remember: $E_{COM} = \mathbb{E}_{\boldsymbol{x}}\left[\left(\frac{1}{M}\sum_{m=1}^{M}(y(\boldsymbol{x}) - y_m(\boldsymbol{x}))\right)^2\right]$.

Expected Error of an Ensemble

- The **average** expected error of an individual model is: $E_{AV} = \frac{1}{M}\sum_{m=1}^{M} \mathbb{E}_{\boldsymbol{x}} \left[(y(\boldsymbol{x}) - y_m(\boldsymbol{x}))^2 \right]$.
- If we make two assumptions...
  1. The errors of each model have zero mean;
  2. The errors of different models are not correlated;
- Remember: $E_{COM} = \mathbb{E}_{\boldsymbol{x}} \left[\left(\frac{1}{M}\sum_{m=1}^{M} (y(\boldsymbol{x}) - y_m(\boldsymbol{x})) \right)^2 \right]$.
- ...so we have $E_{COM} = \frac{1}{M} E_{AV}$, since for E_{COM}, the $\frac{1}{M}$ is squared

Expected Error of an Ensemble

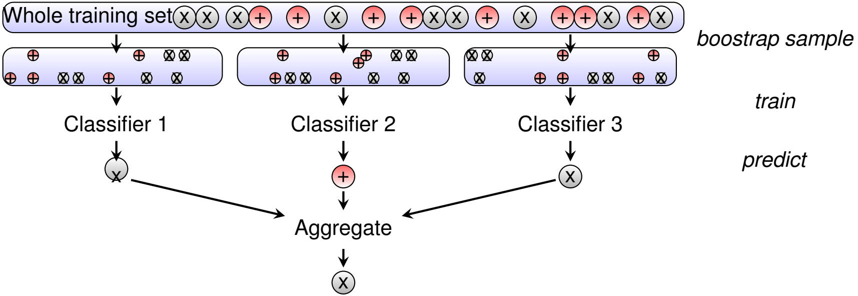
- The **average** expected error of an individual model is: $E_{AV} = \frac{1}{M}\sum_{m=1}^{M} \mathbb{E}_{\boldsymbol{x}}\left[(y(\boldsymbol{x}) - y_m(\boldsymbol{x}))^2\right]$.
- If we make two assumptions...
  1. The errors of each model have zero mean;
  2. The errors of different models are not correlated;
- Remember: $E_{COM} = \mathbb{E}_{\boldsymbol{x}}\left[\left(\frac{1}{M}\sum_{m=1}^{M}(y(\boldsymbol{x}) - y_m(\boldsymbol{x}))\right)^2\right]$.
- ...so we have $E_{COM} = \frac{1}{M}E_{AV}$, since for E_{COM}, the $\frac{1}{M}$ is squared
- Intuition: if models make different, random errors, they will tend to cancel out.

Expected Error of an Ensemble

- $E_{COM} = \frac{1}{M} E_{AV}$ is pretty amazing, but is it realistic?
- No, because we have made extreme assumptions about the models' errors – in practice, they are usually highly correlated and biased.
- However, the combined error cannot be worse than the average error:$E_{COM} \leq E_{AV}$[1]
- The results tells us that the models should be *diverse* to avoid repeating the same errors.

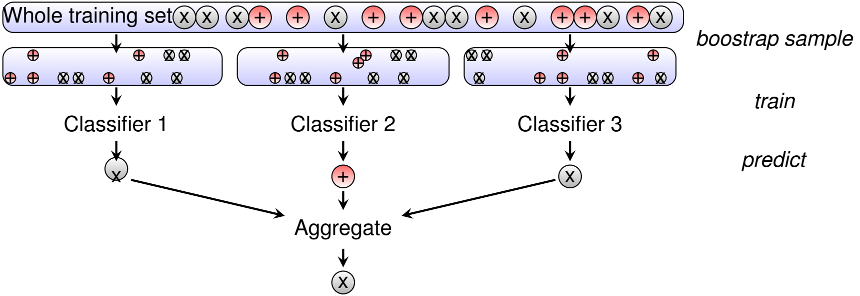
[1] This bound is due to *Jensen's inequality*.

Bootstrap Aggregation (Bagging)



- Create diversity by training models on different samples of the training set.

Bootstrap Aggregation (Bagging)



- Create diversity by training models on different samples of the training set.
- For each model *m*, randomly sample *N* data points with replacement from a training set with *N* data points and train *m* on the subsample.
- In each sample, some data points will be repeated and others will be omitted.
- Combine predictions by taking the mean or majority vote.

Now do the quiz!

Please do the quiz for this lecture on Blackboard.