Machine Learning 2

B. Ensembles / B.3. Mixtures of Experts

Lars Schmidt-Thieme

Information Systems and Machine Learning Lab (ISMLL)
Institute for Computer Science
University of Hildesheim, Germany

Syllabus

			A. Advanced Supervised Learning
Fri.	24.4.	(1)	A.1 Generalized Linear Models
Fri.	1.5.	—	— *Labour Day* —
Fri.	8.5.	(2)	A.2 Gaussian Processes
Fri.	15.5.	(3)	A.3 Advanced Support Vector Machines
			B. Ensembles
Fri.	22.5.	(4)	B.1 Stacking
			& B.2 Boosting
Fri.	29.5.	(5)	B.3 Mixtures of Experts
Fri.	5.6.	—	— *Pentecoste Break* —
			C. Sparse Models
Fri.	12.6.	(6)	C.1 Homotopy and Least Angle Regression
Fri.	19.6.	(7)	C.2 Proximal Gradients
Fri.	26.6.	(8)	C.3 Laplace Priors
Fri.	3.7.	(9)	C.4 Automatic Relevance Determination
			D. Complex Predictors
Fri.	10.7.	(10)	D.1 Latent Dirichlet Allocation (LDA)
Fri.	17.7.	(11)	Q & A

Outline

1. The Idea behind Mixtures of Experts
2. Learning Mixtures of Experts
3. Interpreting Ensemble Models

Outline

1. The Idea behind Mixtures of Experts

2. Learning Mixtures of Experts

3. Interpreting Ensemble Models

Underlying Idea

So far, we build ensemble models where the combination weights do not depend on the predictors:

$$\hat{y}(x) := \sum_{c=1}^{C} \alpha_c \, \hat{y}_c(x)$$

i.e., all instances x are reconstructed from their predictions $\hat{y}_c(x)$ by the component models in the same way α.

Underlying Idea

So far, we build ensemble models where the combination weights do not depend on the predictors:

$$\hat{y}(x) := \sum_{c=1}^{C} \alpha_c \, \hat{y}_c(x)$$

i.e., all instances x are reconstructed from their predictions $\hat{y}_c(x)$ by the component models in the same way α.

New idea: allow each instance to be reconstructed in an instance-specific way.

$$\hat{y}(x) := \sum_{c=1}^{C} \alpha_c(x) \, \hat{y}_c(x)$$

Mixtures of Experts

$x_n \in \mathbb{R}^M, y_n \in \mathbb{R}, c_n \in \{1, \ldots, C\}, \theta := (\beta, \sigma^2, \gamma), \beta, \gamma \in \mathbb{R}^{C \times M}$:

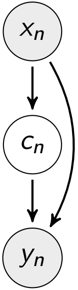
$$p(y_n \mid x_n, c_n; \theta) := \mathcal{N}(y \mid \beta_{c_n}^T x_n, \sigma_{c_n}^2)$$
$$p(c_n \mid x_n; \theta) := \mathrm{Cat}(c \mid \mathcal{S}(\gamma x))$$

with **softmax function**

$$\mathcal{S}(x)_m := \frac{e^{x_m}}{\sum_{m'=1}^M e^{x_{m'}}}, \quad x \in \mathbb{R}^M$$

- C component models (**experts**) $\mathcal{N}(y \mid \beta_c^T x, \sigma_c^2)$
- each model c is expert in some region of predictor space, defined by its component weight (**gating function**) $\mathcal{S}(\gamma x)_c$
- a mixture model with latent nominal variable $z_n := c_n$.

Mixtures of Experts



$x_n \in \mathbb{R}^M, y_n \in \mathbb{R}, c_n \in \{1, \ldots, C\}, \theta := (\beta, \sigma^2, \gamma), \beta, \gamma \in \mathbb{R}^{C \times M}$:

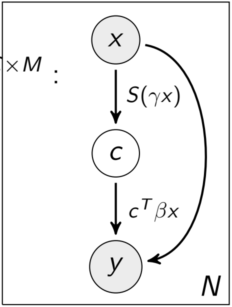
$$\begin{aligned} p(y_n \mid x_n, c_n; \theta) &:= \mathcal{N}(y \mid \beta_{c_n}^T x_n, \sigma_{c_n}^2) \\ p(c_n \mid x_n; \theta) &:= \text{Cat}(c \mid \mathcal{S}(\gamma x)) \end{aligned}$$

with **softmax function**

$$\mathcal{S}(x)_m := \frac{e^{x_m}}{\sum_{m'=1}^M e^{x_{m'}}}, \quad x \in \mathbb{R}^M$$

- C component models (**experts**) $\mathcal{N}(y \mid \beta_c^T x, \sigma_c^2)$
- each model c is expert in some region of predictor space, defined by its component weight (**gating function**) $\mathcal{S}(\gamma x)_c$
- a mixture model with latent nominal variable $z_n := c_n$.

Mixtures of Experts



$$x_n \in \mathbb{R}^M, y_n \in \mathbb{R}, c_n \in \{1, \ldots, C\}, \theta := (\beta, \sigma^2, \gamma), \beta, \gamma \in \mathbb{R}^{C \times M}:$$

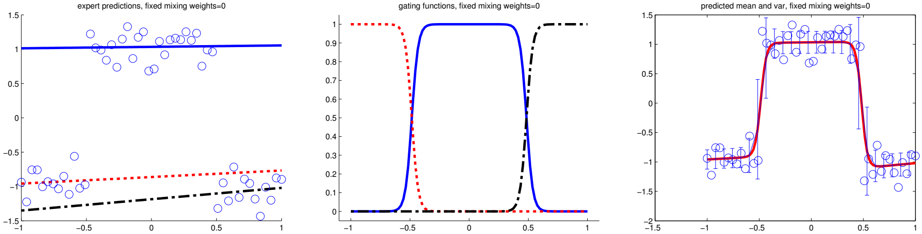
$$p(y_n \mid x_n, c_n; \theta) := \mathcal{N}(y \mid \beta_{c_n}^T x_n, \sigma_{c_n}^2)$$
$$p(c_n \mid x_n; \theta) := \text{Cat}(c \mid \mathcal{S}(\gamma x))$$

with **softmax function**

$$\mathcal{S}(x)_m := \frac{e^{x_m}}{\sum_{m'=1}^M e^{x_{m'}}}, \quad x \in \mathbb{R}^M$$

- C component models (**experts**) $\mathcal{N}(y \mid \beta_c^T x, \sigma_c^2)$
- each model c is expert in some region of predictor space, defined by its component weight (**gating function**) $\mathcal{S}(\gamma x)_c$
- a mixture model with latent nominal variable $z_n := c_n$.

Mixtures of Experts/ Example



component models component weight mixture of experts

[?, fig. 11.6]

Mixtures of Experts

Generic Mixtures of Experts model:

- variables: $x_n \in \mathcal{X}, y_n \in \mathcal{Y}$
- latent variables: $c_n \in \{1, \ldots, C\}$
- **component models**: $p(y_n \mid x_n, c_n; \theta^y)$
  - a separate model for each c: $p(y_n \mid x_n, c; \theta^y) = p(y_n \mid x_n; \theta^y_c)$, with θ^y_c and $\theta^y_{c'}$ being disjoint for $c \neq c'$.
- **combination model**: $p(c_n \mid x_n; \theta^c)$

Example Mixture of Experts model:

- variables: $\mathcal{X} := \mathbb{R}^M, \mathcal{Y} := \mathbb{R}$
- component models: linear regression models $\mathcal{N}(y \mid \beta_c^T x, \sigma_c^2)$
- combination model: logistic regression model $\text{Cat}(c \mid \mathcal{S}(\gamma x))$

For prediction:

$$p(y \mid x) = \sum_{c=1}^{C} p(y \mid x, c) p(c \mid x)$$

Mixtures of Experts

Generic Mixtures of Experts model:

- variables: $x_n \in \mathcal{X}, y_n \in \mathcal{Y}$
- latent variables: $c_n \in \{1, \ldots, C\}$
- **component models**: $p(y_n \mid x_n, c_n; \theta^y)$
  - a separate model for each c: $p(y_n \mid x_n, c; \theta^y) = p(y_n \mid x_n; \theta^y_c)$, with θ^y_c and $\theta^y_{c'}$ being disjoint for $c \neq c'$.
- **combination model**: $p(c_n \mid x_n; \theta^c)$

Example Mixture of Experts model:

- variables: $\mathcal{X} := \mathbb{R}^M, \mathcal{Y} := \mathbb{R}$
- component models: linear regression models $\mathcal{N}(y \mid \beta_c^T x, \sigma_c^2)$
- combination model: logistic regression model $\text{Cat}(c \mid \mathcal{S}(\gamma x))$

For prediction:

$$p(y \mid x) = \sum_{c=1}^{C} \underbrace{p(y \mid x, c)}_{=\hat{y}_c(x)} \underbrace{p(c \mid x)}_{=\alpha_c(x)}$$

Outline

1. The Idea behind Mixtures of Experts

2. Learning Mixtures of Experts

3. Interpreting Ensemble Models

Learning Mixtures of Experts

complete data likelihood:

$$L(\theta^y, \theta^c, c; \mathcal{D}^{\text{train}}) := \prod_{n=1}^{N} p(y_n | x_n, c_n; \theta^y) p(c_n | x_n; \theta^c), \quad c_n \in \{1, \ldots, C\}$$

Cannot be computed, as c_n is unknown.

Learning Mixtures of Experts

complete data likelihood:

$$L(\theta^y, \theta^c, c; \mathcal{D}^{\text{train}}) := \prod_{n=1}^{N} p(y_n | x_n, c_n; \theta^y) p(c_n | x_n; \theta^c), \quad c_n \in \{1, \dots, C\}$$

Cannot be computed, as c_n is unknown.

marginalize out unknown c_n:

$$L(\theta^y, \theta^c; \mathcal{D}^{\text{train}}) := \prod_{n=1}^{N} \sum_{c=1}^{C} p(y_n | x_n, c; \theta^y)\, p(c | x_n; \theta^c)$$

$$\ell(\theta^y, \theta^c) := -\log L(\theta^y, \theta^c)$$

$$= -\sum_{n=1}^{N} \log \sum_{c=1}^{C} p(y_n | x_n, c; \theta^y) \log p(c | x_n; \theta^c)$$

log-sum is difficult to optimize (as it does not decompose in a big sum).

Optimizing log-sums (review)

Lemma

For $x_1, x_2, \ldots, x_N \in \mathbb{R}_0^+$*:*

$$\log \sum_{n=1}^{N} x_n = \max_{q \in \Delta_N} \sum_{n=1}^{N} q_n \log \frac{x_n}{q_n}$$

Proof: "$\geq$":

$$\log \sum_{n=1}^{N} x_n = \log \sum_{n=1}^{N} q_n \frac{x_n}{q_n} \underset{\text{Jensen's ineq.}}{\geq} \sum_{n=1}^{N} q_n \log \frac{x_n}{q_n}, \quad \forall q \in \Delta_N$$

$$\log \sum_{n=1}^{N} x_n \geq \max_{q \in \Delta_N} \sum_{n=1}^{N} q_n \log \frac{x_n}{q_n}$$

"$\leq$": Especially for $q_n := \frac{x_n}{\sum_{n'=1}^{N} x_{n'}}$:

$$\sum_{n=1}^{N} q_n \log \frac{x_n}{q_n} = \sum_{n=1}^{N} \frac{x_n}{\sum_{n'=1}^{N} x_{n'}} \log \sum_{n'=1}^{N} x_{n'} = \log \sum_{n'=1}^{N} x_{n'}$$

Joint Objective Function

$$
\begin{aligned}
\ell(\theta^y, \theta^c) &= -\sum_{n=1}^{N} \log \sum_{c=1}^{C} p(y_n | x_n, c; \theta^y) \log p(c | x_n; \theta^c) \\
&= -\sum_{n=1}^{N} \max_{q_n \in \Delta_C} \sum_{c=1}^{C} q_{n,c} \log \frac{p(y_n | x_n, c; \theta^y)\, p(c | x_n; \theta^c)}{q_{n,c}} \\
\ell(\theta^y, \theta^c, q) &:= -\sum_{n=1}^{N} \sum_{c=1}^{C} q_{n,c} \log \frac{p(y_n | x_n, c; \theta^y)\, p(c | x_n; \theta^c)}{q_{n,c}}
\end{aligned}
$$

Learning Mixtures of Experts

$$\ell(\theta^y, \theta^c, q) := -\sum_{n=1}^{N}\sum_{c=1}^{C} q_{n,c} \log \frac{p(y_n|x_n, c; \theta^y)p(c|x_n; \theta^c)}{q_{n,c}}$$

coordinate descent:

1. minimize w.r.t. θ^c: (maximization step)

$$\ell(\theta^c; \theta^y, q) \propto -\sum_{n=1}^{N}\sum_{c=1}^{C} q_{n,c} \log p(c|x_n; \theta^c)$$

$$\rightsquigarrow \mathcal{D}^{\text{train}}_{\theta^c} := \{(q_{n,c}, x_n, c) \mid n = 1:N, c = 1:C\}$$

alternatively, $\mathcal{D}^{\text{train}}_{\theta^c} := \{(1, x_n, (q_{n,c})_{c=1:C}) \mid n = 1:N\}$

- train combination model on all completed instances, each with case weight $q_{n,c}$ (alternatively: on all instances to predict $q_{n,c}$)

Note: $\mathcal{D}^{\text{train}}$ is given as triples (q, x, y) with instances (x, y) with case weights q.

Learning Mixtures of Experts

$$\ell(\theta^y, \theta^c, q) := -\sum_{n=1}^{N}\sum_{c=1}^{C} q_{n,c} \log \frac{p(y_n|x_n, c; \theta^y) p(c|x_n; \theta^c)}{q_{n,c}}$$

coordinate descent:

1. minimize w.r.t. θ^c: (maximization step)
2. minimize w.r.t. θ^y: (maximization step)

$$\ell(\theta^y; \theta^c, q) \propto -\sum_{n=1}^{N}\sum_{c=1}^{C} q_{n,c} \log p(y_n|x_n, c; \theta^y)$$

decomposes over c:

$$\ell(\theta^y_c; \theta^c, q) \propto \sum_{n=1}^{N} q_{n,c} \log p(y_n|x_n, c; \theta^y_c)$$

$$\rightsquigarrow \mathcal{D}^{\text{train}}_{\theta^y,c} := \{(q_{n,c}, x_n, y_n) \mid n = 1:N\}, \quad c = 1:C$$

- train each component model θ^y_c on all instances, each with case weight $q_{n,c}$

Learning Mixtures of Experts

$$\ell(\theta^y, \theta^c, q) := -\sum_{n=1}^{N}\sum_{c=1}^{C} q_{n,c} \log \frac{p(y_n|x_n, c; \theta^y)p(c|x_n; \theta^c)}{q_{n,c}}$$

coordinate descent:

1. minimize w.r.t. θ^c: (maximization step)
2. minimize w.r.t. θ^y: (maximization step)
3. minimize w.r.t. q: (expectation step)

$$q_{n,c} = \frac{p(y_n|x_n, c; \theta^y)\, p(c|x_n; \theta^c)}{\sum_{c'=1}^{C} p(y_n|x_n, c'; \theta^y)p(c'|x_n; \theta^c)}$$

- can be solved analytically.

Remarks

- Mixtures of experts can use **any model as component model**.
- Mixtures of experts can use **any classification model as combination model**.
  - both models need to be able to **deal with case weights**
  - both models need to be able to **output probabilities**

Remarks

- Mixtures of experts can use **any model as component model**.
- Mixtures of experts can use **any classification model as combination model**.
  - both models need to be able to **deal with case weights**
  - both models need to be able to **output probabilities**

- if data is **sparse**, sparsity can be naturally used in both, component and combination models.

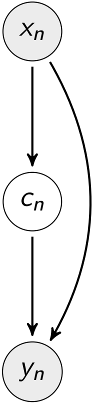
Remarks

- Mixtures of experts can use **any model as component model**.
- Mixtures of experts can use **any classification model as combination model**.
  - both models need to be able to **deal with case weights**
  - both models need to be able to **output probabilities**

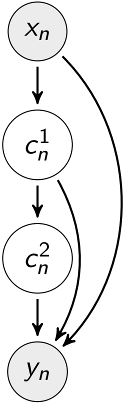
- if data is **sparse**, sparsity can be naturally used in both, component and combination models.

- Updating the three types of parameters can be **interleaved**.
  - this way, $q_{n,c}$ never has to be materialized
    (but for a mini batch, possibly a single n)

Outlook: Hierarchical Mixture of Experts



mixture of experts



hierarchical mixture of experts

Outline

1. The Idea behind Mixtures of Experts

2. Learning Mixtures of Experts

3. Interpreting Ensemble Models

Variable Importance

Some models allow to assess the importance of single variables (or more generally subsets of variables; **variable importance**), e.g.,

- linear models: the z-score
- decision trees: the number of times a variable occurs in its splits

Variable Importance

Some models allow to assess the importance of single variables (or more generally subsets of variables; **variable importance**), e.g.,

- linear models: the z-score
- decision trees: the number of times a variable occurs in its splits

Variable importance of ensembles of such models can be measured as **average variable importance in the component models**:

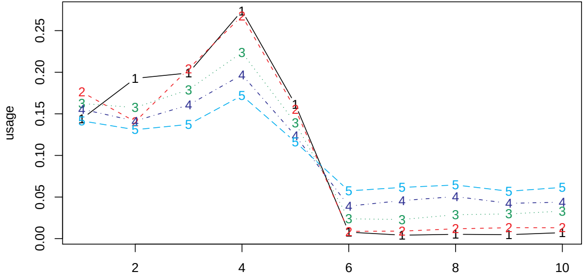
$$\text{importance}(X_m, \hat{y}) := \frac{1}{C} \sum_{c=1}^{C} \text{importance}(X_m, \hat{y}_c), \quad m \in \{1, \ldots, M\}$$

Variable Importance / Example

Synthetic data:

$$x \sim \text{uniform}([0,1]^{10})$$
$$y \sim \mathcal{N}(y \mid 10\sin(\pi x_1 x_2) + 20(x_3 - 0.5)^2 + 10x_4 + 5x_5, 1)$$

Model: Bayesian adaptive regression tree
(variant of a random forest; see [?, p. 551]).



[?, fig. 16.21]

Color denotes the number C of component models.

Variable Dependence: Partial Dependence Plot

For any model $\hat{y}$ (and thus any ensemble), the dependency of the model on a variable X_m can be visualized by a **partial dependence plot**:

plot $z \in \text{range}(X_m)$ vs.

$$\hat{y}_{\text{partial}}(z; X_m, \mathcal{D}^{\text{train}}) := \frac{1}{N} \sum_{n=1}^{N} \hat{y}((x_{n,1}, \ldots, x_{n,m-1}, z, x_{n,m+1}, \ldots, x_{n,M})),$$

or for a subset of variables

$$\hat{y}_{\text{partial}}(z; X_V, \mathcal{D}^{\text{train}}) := \frac{1}{N} \sum_{n=1}^{N} \hat{y}(\rho(x, V, z)), \quad V \subseteq \{1, \ldots, M\}$$

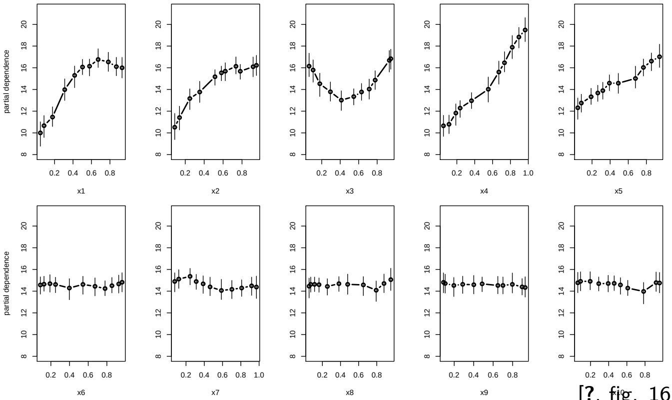
$$\text{with } \rho(x, V, z)_m := \begin{cases} z_m, & \text{if } m \in V \\ x_m, & \text{else} \end{cases}, \quad m \in \{1, \ldots, M\}$$

Variable Dependence / Example

Synthetic data:

$$x \sim \text{uniform}([0,1]^{10})$$

$$y \sim \mathcal{N}(y \mid 10\sin(\pi x_1 x_2) + 20(x_3 - 0.5)^2 + 10x_4 + 5x_5, 1)$$



[?, fig. 16.20]

Summary

- **Mixtures of Experts** additionally allow the combination weights to depend on x (**gating function**)
  - jointy model
    - a latent component each instance belongs to and
    - a model for y for each component
  - can be learned via block coordinate descent / EM.
    - requiring just learning algorithms for the component models
    - as well as for the combination model.

- Ensemble models can be diagnosed by **partial dependence plots** (as any model).

Further Readings

- Mixtures of Experts: [?, chapter 14.5]. [?, chapter 11.2.4, 11.4.3], [?, chapter 9.5].
- Optimizing log-sums and EM algorithm as coordinate descent:
  - lecture Machine Learning, C.1 Clustering, section 2 on Gaussian Mixture Models.

Acknowledgements: Thanks a lot to my PhD student Randolf Scholz for spotting a bad mistake on an earlier version of these slides!

References

Christopher M. Bishop.
Pattern recognition and machine learning, volume 1.
springer New York, 2006.

Trevor Hastie, Robert Tibshirani, Jerome Friedman, and James Franklin.
The elements of statistical learning: data mining, inference and prediction, volume 27.
Springer, 2005.

Kevin P. Murphy.
Machine learning: a probabilistic perspective.
The MIT Press, 2012.