Introduction to Machine Learning

A statistical perspective on supervised learning

Nils M. Kriege

WS 2023

Data Mining and Machine Learning
Faculty of Computer Science
University of Vienna

Credit: Slides based on the IML Lectures by Sebastian Tschiatschek and Andreas Krause

Organization

The remainder of the course

Next steps

- 3 more lectures (incl. today's lecture)
- 1 outlook and review session before the final exam (January 26, 2024)
  Tell me what I should review or recap until 23.1.2024
- Final exam: January 29, 2024
- 1 more pen & paper exercise (January 22, 2024)
  Bonus points
- 1 more programming assignment (January 31, 2024, without peer-review)
  Bonus points

Introduction

A statistical perspective on supervised learning

Motivation

- We have seen how we can fit prediction models (linear, non-linear) for regression and classification
- So far, these models do not have any statistical interpretation
- Often we would like to statistically model the data:
  - Quantify uncertainty
  - Express prior knowledge / assumptions about the data
- In the following, we will see how many of the approaches we have discussed can be interpreted as fitting probabilistic models
- This view will also allow us to derive new methods

Recall: Goal of supervised learning

- Given training data

$$\mathcal{D} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\} \subseteq \mathcal{X} \times \mathcal{Y}$$

- Want to identify a hypothesis $h \colon \mathcal{X} \to \mathcal{Y}$, e.g.,
    - Linear regression: $h(\mathbf{x}) = \mathbf{w}^T \mathbf{x}$
    - Kernel regression: $h(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i k(\mathbf{x}_i, \mathbf{x})$
    - Neural network (single hidden layer): $h(\mathbf{x}) = \sum_{i=1}^{k} w'_i \varphi(\mathbf{w}_i^T \mathbf{x})$
- **Goal:** Want to minimize prediction error (risk)

Minimizing generalization error

- **Fundamental assumption**: Our data set is generated independently and identically distributed (iid), i..e,

$$(\mathbf{x}_i, y_i) \sim P(\mathbf{X}, Y)$$

- Would like to identify a hypothesis $h \colon \mathcal{X} \to \mathcal{Y}$ that minimizes the prediction error (risk):

$$\begin{aligned} R(h) &= \int P(\mathbf{x}, y) \ell(y; h(\mathbf{x}))\, \mathrm{d}\mathbf{x}\, \mathrm{d}y \\ &= \mathbb{E}_{\mathbf{X},Y}[\ell(Y; h(\mathbf{X})] \end{aligned}$$

- Defined in terms of a loss function

Least-squares regression

- In least-squares regression, risk is

$$R(h) = \mathbb{E}_{\mathbf{X},Y}[(Y - h(\mathbf{X}))^2]$$

- Suppose (unrealistically) that we knew $P(\mathbf{X}, Y)$
- Which h minimizes the risk then?

Least-squares regression

- In least-squares regression, risk is

$$R(h) = \mathbb{E}_{\mathbf{X},Y}[(Y - h(\mathbf{X}))^2]$$

- Suppose (unrealistically) that we knew $P(\mathbf{X}, Y)$
- Which h minimizes the risk then?

$$\begin{aligned}
\min_{h\colon \mathbb{R}^d \to \mathbb{R}} R(h) &= \min_h \mathbb{E}_{(\mathbf{x},y)\sim P}\left[(y - h(\mathbf{x}))^2\right] \\
&= \min_h \mathbb{E}_{\mathbf{x}}\left[\mathbb{E}_y\left[(y - h(\mathbf{x}))^2 | \mathbf{X} = \mathbf{x}\right]\right] \\
&\overset{(*)}{=} \mathbb{E}_{\mathbf{x}}\left[\min_h \mathbb{E}_y\left[(y - h(\mathbf{x}))^2 | \mathbf{X} = \mathbf{x}\right]\right]
\end{aligned}$$

$(*)$ since we consider arbitrary h; choose $h(\mathbf{x})$ and $h(\mathbf{x}')$ arbitrarily for $\mathbf{x} \neq \mathbf{x}'$.

Least-squares regression

What is the optimal prediction for a given $\mathbf{x}$?

Least-squares regression

What is the optimal prediction for a given **x**?

$$y^*(\mathbf{x}) \in \operatorname*{argmin}_{\hat{y}} \underbrace{\mathbb{E}_y[(y - \hat{y})^2 | \mathbf{X} = \mathbf{x}]}_{=\ell(\hat{y})}$$

Least-squares regression

What is the optimal prediction for a given **x**?

$$y^*(\mathbf{x}) \in \underset{\hat{y}}{\operatorname{argmin}} \underbrace{\mathbb{E}_y[(y-\hat{y})^2 | \mathbf{X} = \mathbf{x}]}_{=\ell(\hat{y})}$$

$$\begin{aligned}
\ell(\hat{y}) &= \int (y-\hat{y})^2 P(y|\mathbf{x}) \, \mathrm{d}y \\
\frac{\mathrm{d}}{\mathrm{d}\hat{y}} \ell(\hat{y}) &= \int \frac{\mathrm{d}}{\mathrm{d}\hat{y}} (y-\hat{y})^2 P(y|\mathbf{x}) \, \mathrm{d}y = \int 2(y-\hat{y}) P(y|\mathbf{x}) \, \mathrm{d}y \overset{!}{=} 0 \\
&\Leftrightarrow \int \hat{y} P(y|\mathbf{x}) \, \mathrm{d}y \overset{!}{=} \int y P(y|\mathbf{x}) \, \mathrm{d}y \\
&\Leftrightarrow \hat{y} = \mathbb{E}[y | \mathbf{X} = \mathbf{x}]
\end{aligned}$$

Minimizing the least squares error

- Assuming data is generated iid according to

$$(\mathbf{x}_i, y_i) \sim P(\mathbf{X}, Y)$$

- The hypothesis h^* minimizing $R(h) = \mathbb{E}_{\mathbf{X},Y}[(Y - h(\mathbf{X}))^2]$ is given by the conditional mean

$$h^*(\mathbf{x}) = \mathbb{E}[Y|\mathbf{X} = \mathbf{x}]$$

- This (in practice unattainable) hypothesis is called the Bayes' optimal predictor for the squared loss

In practice we have finite data

- We know that

$$h^*(\mathbf{x}) = \mathbb{E}[Y|\mathbf{X} = \mathbf{x}]$$

- Thus, one strategy for estimating a predictor for training data is to estimate the conditional distribution

$$\hat{P}(Y|\mathbf{X})$$

and then, for test point **x**, predict label

$$\hat{y} = \hat{\mathbb{E}}[Y|\mathbf{X} = \mathbf{x}] = \int \hat{P}(y|\mathbf{X} = \mathbf{x})y \, \mathrm{d}y$$

Estimating conditional distributions

- Common approach: Parametric estimation
  - Choose a particular parametric form $\hat{P}(Y|\mathbf{X};\theta)$
  - Then optimize the parameters. How?

Estimating conditional distributions

- Common approach: Parametric estimation
  - Choose a particular parametric form $\hat{P}(Y|\mathbf{X};\theta)$
  - Then optimize the parameters. How?
- 💡 Maximum (conditional) Likelihood Estimation

$$
\begin{aligned}
\theta^* &= \operatorname*{argmax}_\theta \hat{P}(y_1,\ldots,y_n|\mathbf{x}_1,\ldots,\mathbf{x}_n;\theta) \\
&= \operatorname*{argmax}_\theta \prod_{i=1}^n \hat{P}(y_i|\mathbf{x}_i,\theta) = \operatorname*{argmax}_\theta \log \prod_{i=1}^n \hat{P}(y_i|\mathbf{x}_i,\theta) \\
&= \operatorname*{argmax}_\theta \sum_{i=1}^n \log \hat{P}(y_i|\mathbf{x}_i,\theta) = \operatorname*{argmin}_\theta - \sum_{i=1}^n \log \hat{P}(y_i|\mathbf{x}_i,\theta)
\end{aligned}
$$

Example: Conditional linear Gaussian

Assumptions:

- $y = h(\mathbf{x}) + \varepsilon \qquad \varepsilon \sim \mathcal{N}(0, \sigma^2)$ Gaussian noise
- $h(\mathbf{x}) = \mathbf{w}^T \mathbf{x}$

$$\Rightarrow \hat{P}(y|\mathbf{x}, \mathbf{w}, \sigma^2) = \mathcal{N}(y; \mathbf{w}^T \mathbf{x}, \sigma^2)$$

$$\Rightarrow \hat{\mathbf{w}} = \operatorname*{argmax}_{\mathbf{w}} \hat{P}(y_{1:n}|\mathbf{x}_{1:n}, \mathbf{w}, \sigma^2)$$

$$= \operatorname*{argmin}_{\mathbf{w}} - \sum_{i=1}^{n} \log \hat{P}(y_i|x_i, \mathbf{w}, \sigma^2)$$

A probabilistic model for regression (1/2)

- Consider linear regression. Let's make the statistical assumption that the noise is Gaussian:

$$y_i \sim \mathcal{N}(\mathbf{w}^T \mathbf{x}_i, \sigma^2)$$

- Then we can compute the (conditional) likelihood of the data given any candidate model **w** as:

$$\begin{aligned} -\log \hat{P}(y|\mathbf{x}, \mathbf{w}, \sigma^2) &= -\log \mathcal{N}(y|\mathbf{w}^T\mathbf{x}, \sigma^2) \\ &= -\log \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y-\mathbf{w}^T\mathbf{x})^2}{2\sigma^2}\right) \\ &= \frac{1}{2}\log 2\pi\sigma^2 + \frac{(y-\mathbf{w}^T\mathbf{x})^2}{2\sigma^2} \end{aligned}$$

A probabilistic model for regression (2/2)

$$
\begin{aligned}
\Rightarrow \hat{\mathbf{w}} &= \underset{\mathbf{w}}{\operatorname{argmax}}\, \hat{P}(y_{1:n}|\mathbf{x}_{1:n}, \mathbf{w}, \sigma^2) \\
&= \underset{\mathbf{w}}{\operatorname{argmin}} \sum_{i=1}^{n} \left(\frac{1}{2}\log(2\pi\sigma^2) + \frac{(y_i - \mathbf{w}^T\mathbf{x}_i)^2}{2\sigma^2} \right) \\
&= \underset{\mathbf{w}}{\operatorname{argmin}}\, \frac{n}{2}\log(2\pi\sigma^2) + \frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - \mathbf{w}^T\mathbf{x}_i)^2 \\
&= \underset{\mathbf{w}}{\operatorname{argmin}} \sum_{i=1}^{n}(y_i - \mathbf{w}^T\mathbf{x}_i)^2
\end{aligned}
$$

MLE for conditional linear Gaussian

- The negative log likelihood is given by

$$L(\mathbf{w}) = -\log P(y_1, \ldots, y_n | \mathbf{x}_1, \ldots, \mathbf{x}_n; \mathbf{w}) = \frac{n}{2}\log(2\pi\sigma^2) + \sum_{i=1}^{n} \frac{(y_i - \mathbf{w}^T\mathbf{x}_i)^2}{2\sigma^2}$$

- Thus, under the "conditional linear Gaussian" assumption, maximizing the likelihood is equivalent to least squares estimation:

$$\operatorname*{argmax}_{\mathbf{w}} P(y_1, \ldots, y_n | \mathbf{x}_1, \ldots, \mathbf{x}_n; \mathbf{w}) = \operatorname*{argmin}_{\mathbf{w}} \sum_{i=1}^{n} (y_i - \mathbf{w}^T\mathbf{x}_i)^2$$

More generally: MLE for iid Gaussian noise

- Suppose $\mathcal{H} = \{h\colon \mathcal{X} \to \mathbb{R}\}$

- Assuming that $P(Y = y|\mathbf{X} = \mathbf{x}) = \mathcal{N}(y|h^*(\mathbf{x}), \sigma^2)$ for some function $h^*\colon \mathcal{X} \to \mathbb{R}$ and some $\sigma^2 > 0$ the MLE for data $\mathcal{D} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ in $\mathcal{H}$ is given by

$$\hat{h} = \operatorname*{argmin}_{h \in \mathcal{H}} \sum_{i=1}^{n} (y_i - h(\mathbf{x}_i))^2$$

Least-squares regression = Gaussian MLE

- The Maximum Likelihood Estimate (MLE) is given by the least squares solution, assuming that the noise is iid Gaussian with constant variance
- This is useful since MLE satisfies several nice statistical properties (not formally defined here):
  - Consistency (parameter estimate converges to true parameters in probability)
  - Asymptotic efficiency (smallest variance among all "well-behaved" estimators for large *n*
  - Asymptotic normality

Least-squares regression = Gaussian MLE

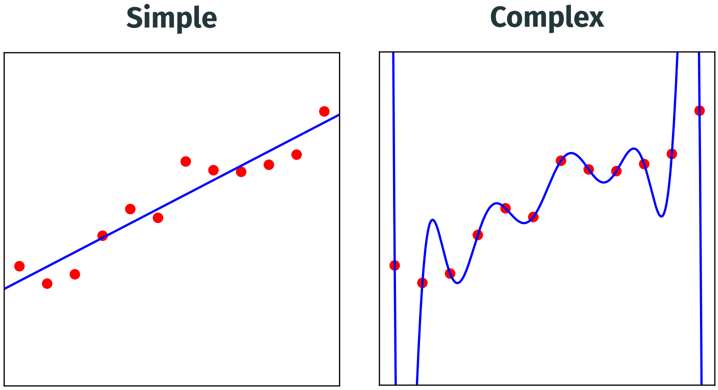
- The Maximum Likelihood Estimate (MLE) is given by the least squares solution, assuming that the noise is iid Gaussian with constant variance
- This is useful since MLE satisfies several nice statistical properties (not formally defined here):
  - Consistency (parameter estimate converges to true parameters in probability)
  - Asymptotic efficiency (smallest variance among all "well-behaved" estimators for large n
  - Asymptotic normality

⚠ However, all these properties are asymptotic (hold as $n \to \infty$). For finite n, we must avoid overfitting!

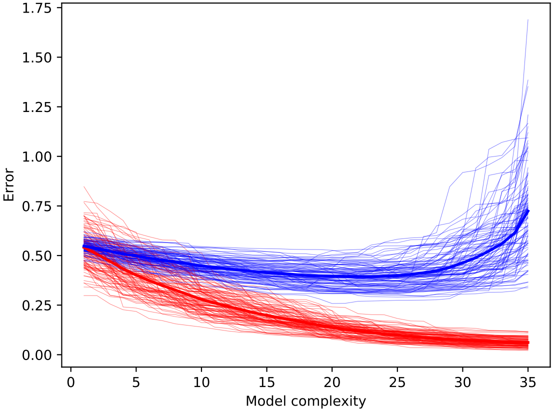
A statistical perspective on overfitting

Bias-variance tradeoff

Recall: Overfitting in regression



Recall: Overfitting in regression



— training — test

Bias-variance tradeoff

- Assume $Y = h^*(\mathbf{X}) + \epsilon$, with ϵ being zero mean noise
- Let $\mathcal{D}$ denote training data
- Then, for least-squares estimation the following holds:

$$
\begin{aligned}
\mathbb{E}_{\mathcal{D}}\mathbb{E}_{\mathbf{X},Y}[(Y - \hat{h}_{\mathcal{D}}(\mathbf{X}))^2] =& \mathbb{E}_{\mathbf{X}}[\mathbb{E}_{\mathcal{D}}\hat{h}_{\mathcal{D}}(\mathbf{X}) - h^*(\mathbf{X})]^2 \\
&+ \mathbb{E}_{\mathbf{X}}\mathbb{E}_{\mathcal{D}}[\hat{h}_{\mathcal{D}}(\mathbf{X}) - \mathbb{E}_{\mathcal{D}'}\hat{h}_{\mathcal{D}'}(\mathbf{X})]^2 \\
&+ \mathbb{E}_{\mathbf{X},Y}[Y - h^*(\mathbf{X})]^2
\end{aligned}
$$

Bias Variance Tradeoff

Bias Variance Tradeoff

$$\text{Prediction error} = \text{Bias}^2 + \text{Variance} + \text{Noise}$$

- **Bias** Excess risk of average prediction compared to minimal achievable risk knowing $P(\mathbf{X}, Y)$ (i.e., given infinite data)
- **Variance** Risk incurred due to estimating model from limited data
- **Noise** Risk incurred by optimal model (i.e., irreducible error)

Bias in estimation

- MLE solution depends on training data $\mathcal{D}$:

$$\hat{h} = \hat{h}_{\mathcal{D}} = \underset{h \in \mathcal{H}}{\text{argmin}} \sum_{(\mathbf{x}, y) \in \mathcal{D}} (y - h(\mathbf{x}))^2$$

- But training data $\mathcal{D}$ is itself random (drawn iid from P)
- We might want to choose h to have small bias (i.e., have small squared error on average):

$$\mathbb{E}_{\mathbf{X}}[\mathbb{E}_{\mathcal{D}} \hat{h}_{\mathcal{D}}(\mathbf{X}) - h^*(\mathbf{X})]^2$$

Variance in estimation

- MLE solution depends on training data $\mathcal{D}$:

$$\hat{h} = \hat{h}_{\mathcal{D}} = \underset{h \in \mathcal{H}}{\operatorname{argmin}} \sum_{(\mathbf{x}, y) \in \mathcal{D}} (y - h(\mathbf{x}))^2$$

- This estimator is itself random and has some variance:

$$\mathbb{E}_{\mathbf{X}} \mathrm{Var}_{\mathcal{D}}[\hat{h}_{\mathcal{D}}(\mathbf{X})]^2 = \mathbb{E}_{\mathbf{X}} \mathbb{E}_{\mathcal{D}}[\hat{h}_{\mathcal{D}}(\mathbf{X}) - \mathbb{E}_{\mathcal{D}'} \hat{h}_{\mathcal{D}'}(\mathbf{X})]^2$$

Noise in estimation

- Even if we know the Bayes' optimal hypothesis h^*, we would still incur some error due to noise:

$$\mathbb{E}_{\mathbf{X},Y}[(Y - h^*(\mathbf{X}))^2]$$

- This error is irreducible, i.e., independent of choice of the hypothesis class

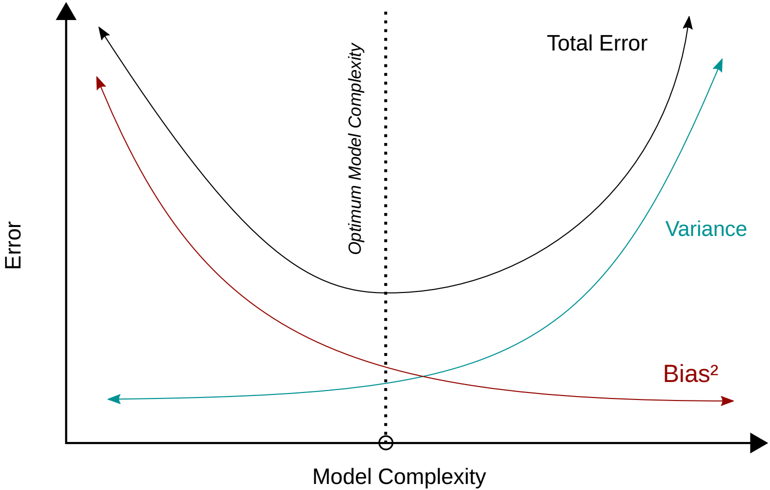
Bias-variance tradeoff

For least-squares estimation the following holds:

$$\begin{aligned}\mathbb{E}_{\mathcal{D}}\mathbb{E}_{\mathbf{X},Y}[(Y-\hat{h}_{\mathcal{D}}(\mathbf{X}))^2] =& \mathbb{E}_{\mathbf{X}}[\mathbb{E}_{\mathcal{D}}\hat{h}_{\mathcal{D}}(\mathbf{X}) - h^*(\mathbf{X})]^2 \\ &+ \mathbb{E}_{\mathbf{X}}\mathbb{E}_{\mathcal{D}}[\hat{h}_{\mathcal{D}}(\mathbf{X}) - \mathbb{E}_{\mathcal{D}'}\hat{h}_{\mathcal{D}'}(\mathbf{X})]^2 \\ &+ \mathbb{E}_{\mathbf{X},Y}[Y - h^*(\mathbf{X})]^2\end{aligned}$$

Ideally wish to find estimator that simultaneously minimizes bias and variance

Bias variance tradeoff illustration



Bias-variance demo

Bias and variance in regression

- The maximum likelihood estimate (= least-squares fit) for linear regression is unbiased (if h^* in class $\mathcal{H}$)
- Furthermore, it is the minimum variance estimator among all unbiased estimators
  (Gauss-Markov Theorem, not explained further here)
- However, we have already seen that the least-squares solution can overfit
- 💡 Thus, trade (a little bit of) bias for a (potentially dramatic) reduction in variance
- $\Rightarrow$ Regularization (e.g., ridge regression, Lasso, ...)

Summary: Bias Variance Tradeoff

Bias Variance Tradeoff

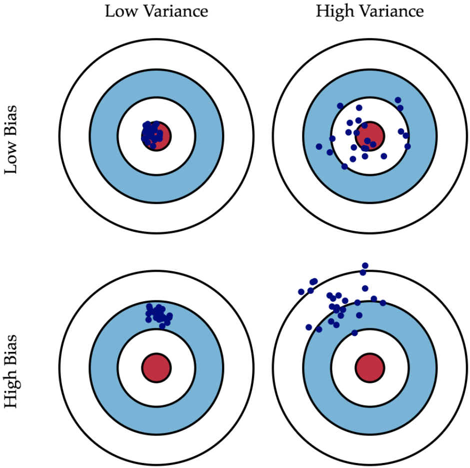
$$\text{Prediction error} = \text{Bias}^2 + \text{Variance} + \text{Noise}$$

- **Bias** Excess risk of average prediction compared to minimal achievable risk knowing $P(\mathbf{X}, Y)$ (i.e., given infinite data)
- **Variance** Risk incurred due to estimating model from limited data
- **Noise** Risk incurred by optimal model (i.e., irreducible error)

Trade bias and variance via model selection / regulariation

Summary: Bias Variance Tradeoff

[Scott Fortmann-Roe]



A Bayesian perspective

Introducing bias

Introducing bias through Bayesian modeling

- Can introduce bias by expressing assumptions on parameters through a Bayesian prior
- For example, let's assume $\mathbf{w} \sim \mathcal{N}(0, \beta^2 \mathbf{I})$
- Then, the posterior distribution of $\mathbf{w}$ is given using Bayes' rule by

$$
\begin{aligned}
P(\mathbf{w}|\mathbf{x}_{1:n}, y_{1:n}) &= \frac{P(\mathbf{x}_{1:n}, y_{1:n}|\mathbf{w})P(\mathbf{w})}{P(\mathbf{x}_{1:n}, y_{1:n})} \\
&= \frac{P(\mathbf{w})P(y_{1:n}|\mathbf{x}_{1:n}, \mathbf{w})P(\mathbf{x}_{1:n})}{P(y_{1:n}|\mathbf{x}_{1:n})P(\mathbf{x}_{1:n})} = \frac{P(\mathbf{w})P(y_{1:n}|\mathbf{x}_{1:n}, \mathbf{w})}{P(y_{1:n}|\mathbf{x}_{1:n})}
\end{aligned}
$$

Introducing bias through Bayesian modeling

- Can introduce bias by expressing assumptions on parameters through a Bayesian prior
- For example, let's assume $\mathbf{w} \sim \mathcal{N}(0, \beta^2 \mathbf{I})$
- Then, the posterior distribution of $\mathbf{w}$ is given using Bayes' rule by

$$P(\mathbf{w}|\mathbf{x}_{1:n}, y_{1:n}) = \frac{P(\mathbf{x}_{1:n}, y_{1:n}|\mathbf{w})P(\mathbf{w})}{P(\mathbf{x}_{1:n}, y_{1:n})}$$
$$= \frac{P(\mathbf{w})P(y_{1:n}|\mathbf{x}_{1:n}, \mathbf{w})P(\mathbf{x}_{1:n})}{P(y_{1:n}|\mathbf{x}_{1:n})P(\mathbf{x}_{1:n})} = \frac{P(\mathbf{w})P(y_{1:n}|\mathbf{x}_{1:n}, \mathbf{w})}{P(y_{1:n}|\mathbf{x}_{1:n})}$$

- ? Which parameters $\mathbf{w}$ are most likely a posteriori?

Maximum a posteriori estimate

$$\operatorname*{argmax}_{\mathbf{w}} P(\mathbf{w}|\mathbf{x}_{1:n}, y_{1:n}) = \operatorname*{argmin}_{\mathbf{w}} -\log P(\mathbf{w}) - \log P(y_{1:n}|\mathbf{x}_{1:n}, \mathbf{w})$$

$$\cancel{+\log P(y_{1:n}|\mathbf{x}_{1:n})}$$

$$-\log P(\mathbf{w}) = -\log \prod_{i=1}^{d} P(w_i) = -\sum_{i=1}^{d} \log \mathcal{N}(w_i; 0, \beta^2)$$

$$= -\sum_{i=1}^{d} \log \frac{1}{\sqrt{2\pi\beta^2}} \exp\left(\frac{-w_i^2}{2\beta^2}\right) = \frac{d}{2}\log 2\pi\beta^2 + \frac{1}{2\beta^2}\sum_{i=1}^{d} w_i^2$$

$$\operatorname*{argmin}_{\mathbf{w}} \cancel{\frac{d}{2}\log 2\pi\beta^2} + \frac{1}{2\beta^2}\|\mathbf{w}\|_2^2 + \frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - \mathbf{w}^T\mathbf{x}_i)^2$$

$$\operatorname*{argmin}_{\mathbf{w}} \frac{\sigma^2}{\beta^2}\|\mathbf{w}\|_2^2 + \sum_{i=1}^{n}(y_i - \mathbf{w}^T\mathbf{x}_i)^2 \Leftrightarrow \text{Ride regression with } \lambda = \frac{\sigma^2}{\beta^2}$$

Ridge regression = MAP estimation

- Ridge regression can be understood as finding the Maximum A Posteriori (MAP) parameter estimate for a linear regression problem, assuming that
  - The noise $P(y|\mathbf{x}, \mathbf{w})$ is iid Gaussian and
  - The prior $P(\mathbf{w})$ on the model parameters $\mathbf{w}$ is Gaussian

$$\underset{\mathbf{w}}{\operatorname{argmin}} \sum_{i=1}^{n} (y_i - \mathbf{w}^T \mathbf{x}_i)^2 + \lambda \|\mathbf{w}\|_2^2 = \underset{\mathbf{w}}{\operatorname{argmax}} \, P(\mathbf{w}) \prod_{i=1}^{n} P(y_i | \mathbf{x}_i, \mathbf{w})$$

Regularization vs. MAP inference

- More generally, regularized estimation can often be understood as MAP inference:

$$\operatorname*{argmin}_{\mathbf{w}} \sum_{i=1}^{n} \ell(\mathbf{w}^T\mathbf{x}_i; \mathbf{x}_i, y_i) + C(\mathbf{w}) = \operatorname*{argmax}_{\mathbf{w}} \prod_{i=1}^{n} P(y_i|\mathbf{x}_i, \mathbf{w})P(\mathbf{w})$$

$$= \operatorname*{argmax}_{\mathbf{w}} P(\mathbf{w}|\mathcal{D})$$

where $C(\mathbf{w}) = -\log P(\mathbf{w})$
and $\ell(\mathbf{w}^T\mathbf{x}_i; \mathbf{x}_i, y_i) = -\log P(y_i|\mathbf{x}_i, \mathbf{w})$

- 💡 This perspective allows changing priors (=regularizers) and likelihoods (=loss functions)

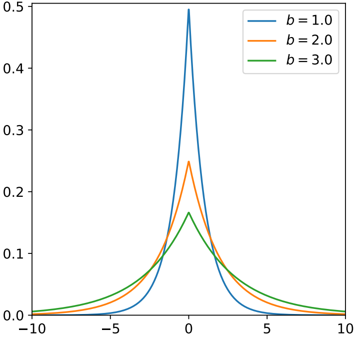
Example: l1-regularization

- ? Is there a prior that corresponds to l1-regularization?

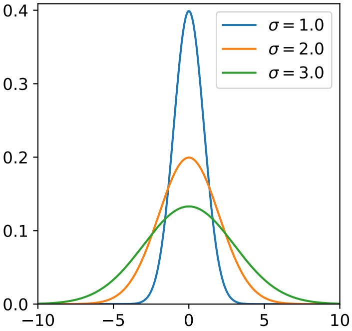
Example: l1-regularization

- ? Is there a prior that corresponds to l1-regularization?
- Answer: The Laplace prior

Laplace



Compare with **Gaussian**

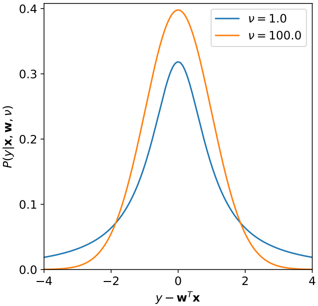


$$p(x; \mu, b) = \frac{1}{2b} \exp\left(-\frac{|x-\mu|}{b}\right)$$

[probabilistic-modeling/distributions.py]

Example: Student-t likelihood

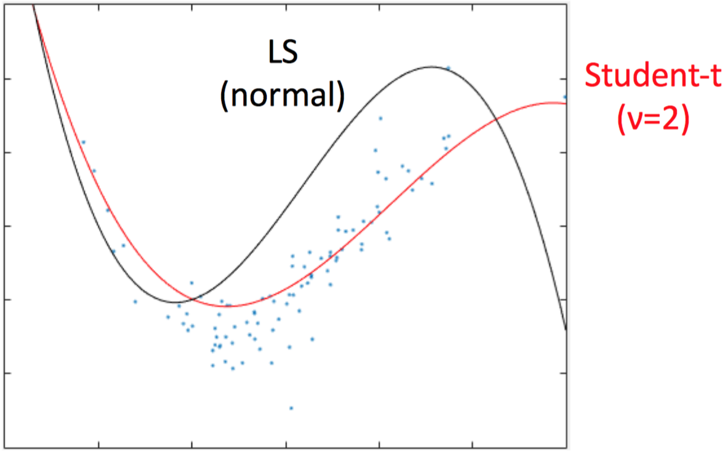
- Can introduce robustness by changing the likelihood (=loss) function
- **Example:** (non-standardized) Student's-t likelihood



$$p(y|\mathbf{x}, \mathbf{w}, \nu, \sigma^2) = \frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\sqrt{\pi\nu\sigma^2}\,\Gamma\left(\frac{\nu}{2}\right)}\left(1 + \frac{(y - \mathbf{w}^T\mathbf{x})^2}{\nu\sigma^2}\right)^{-\frac{\nu+1}{2}}$$

[probabilistic-modeling/distributions.py]

Example fits



Statistical models for classification

Statistical models for classification

- So far, we have focused on regression
- Are there natural statistical models for classification?

Risk in classification

- In classification, risk is

$$R(h) = \mathbb{E}_{\mathbf{x},y}[[Y \neq h(\mathbf{X})]]$$

- Suppose (unrealistically) we knew $P(\mathbf{X}, Y)$
- Which h minimizes the risk then?

Risk in classification

- In classification, risk is

$$R(h) = \mathbb{E}_{\mathbf{x},y}[[Y \neq h(\mathbf{X})]]$$

- Suppose (unrealistically) we knew $P(\mathbf{X}, Y)$
- Which h minimizes the risk then?

$$h^*(\mathbf{x}) = \operatorname*{argmin}_{\hat{y}} \underbrace{\mathbb{E}_y[[y \neq \hat{y}]|\mathbf{X} = \mathbf{x}]}_{=\ell(\hat{y})} = (*)$$

$$\ell(\hat{y}) = \sum_{y=1}^{c} P(Y = y|\mathbf{X} = \mathbf{x})[y \neq \hat{y}] = \sum_{y:y\neq\hat{y}} P(Y = y|\mathbf{X} = \mathbf{x})$$

$$= 1 - P(Y = \hat{y}|\mathbf{X} = \mathbf{x})$$

$$(*) = \operatorname*{argmax}_{\hat{y}} P(Y = \hat{y}|\mathbf{X} = \mathbf{x}) \Rightarrow$$ Predict most probable label under $P(Y|\mathbf{X} = \mathbf{x})$

Bayes' optimal classifier

- Assuming the data is generated iid according to

$$(\mathbf{x}_i, y_i) \sim P(\mathbf{X}, Y)$$

- The hypothesis h^* minimizing $R(h) = \mathbb{E}_{\mathbf{x},y}[[Y \neq h(\mathbf{X})]]$ is given by the most probable class

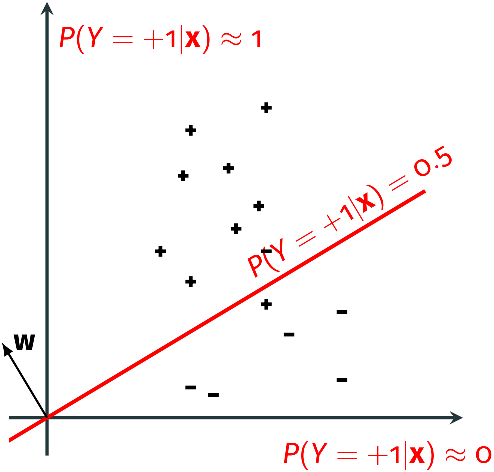
$$h^*(\mathbf{x}) = \operatorname*{argmax}_y P(Y = y | \mathbf{X} = \mathbf{x})$$

- This (in practice unattainable) hypothesis is called the Bayes' optimal predictor for the 0-1-loss
- Thus, natural approach is again to estimate $P(Y|\mathbf{X})$

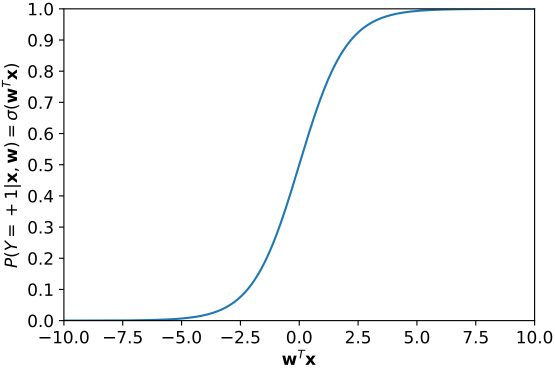
Logistic regression

Use (generalized) linear model for the class probability

$$P(Y = +1|\mathbf{X} = \mathbf{x}) = \sigma(\mathbf{w}^T\mathbf{x})$$



Link function for logistic regression



- Link function:

$$\sigma(\mathbf{w}^T\mathbf{x}) = \frac{1}{1 + \exp(-\mathbf{w}^T\mathbf{x})}$$

[probabilistic-modeling/link-function.py]

Link function for logistic regression

$$
\begin{aligned}
P(Y=+1|\mathbf{x}) &= \frac{1}{1+\exp(-\mathbf{w}^T\mathbf{x})} \\
P(Y=-1|\mathbf{x}) &= 1 - P(Y=+1|\mathbf{x}) = 1 - \frac{1}{1+\exp(-\mathbf{w}^T\mathbf{x})} \\
&= \frac{\exp(-\mathbf{w}^T\mathbf{x})}{1+\exp(-\mathbf{w}^T\mathbf{x})} = \frac{\exp(-\mathbf{w}^T\mathbf{x})\exp(\mathbf{w}^T\mathbf{x})}{\exp(\mathbf{w}^T\mathbf{x}) + \exp(-\mathbf{w}^T\mathbf{x})\exp(\mathbf{w}^T\mathbf{x})} \\
&= \frac{1}{1+\exp(\mathbf{w}^T\mathbf{x})} \\
P(Y=y|\mathbf{x}) &= \frac{1}{1+\exp(-y\mathbf{w}^T\mathbf{x})}
\end{aligned}
$$

- Logistic regression (a classification method) replaces the assumption of Gaussian noise (squared loss) by iid Bernoulli noise:

$$P(y|\mathbf{w}, \mathbf{x}) = \mathrm{Ber}(\sigma(\mathbf{w}^T \mathbf{x}))$$

Logistic Regression

- Logistic regression (a classification method) replaces the assumption of Gaussian noise (squared loss) by iid Bernoulli noise:

$$P(y|\mathbf{w}, \mathbf{x}) = \text{Ber}(\sigma(\mathbf{w}^T \mathbf{x}))$$

- How can we estimate the parameters $\mathbf{w}$?
- $\Rightarrow$ Maximum Likelihood Estimation / MAP estimation

MLE for logistic regression

$$\hat{\mathbf{w}} = \underset{\mathbf{w}}{\operatorname{argmax}} P(y_{1:n}|\mathbf{x}_{1:n}, \mathbf{w}) = \underset{\mathbf{w}}{\operatorname{argmax}} \prod_{i=1}^{n} P(y_i|\mathbf{x}_i, \mathbf{w})$$

$$= \underset{\mathbf{w}}{\operatorname{argmax}} - \sum_{i=1}^{n} \log P(y_i|\mathbf{x}_i, \mathbf{w}) = (*)$$

$$-\log P(y_i|\mathbf{x}_i, \mathbf{w}) = -\log \frac{1}{1 + \exp(-y_i \mathbf{w}^T \mathbf{x}_i)} = \log(1 + \exp(-y_i \mathbf{w}^T \mathbf{x}_i))$$

$$(*) = \underset{\mathbf{w}}{\operatorname{argmin}} \underbrace{\sum_{i=1}^{n} \underbrace{\log(1 + \exp(-y_i \mathbf{w}^T \mathbf{x}_i))}_{\ell_{\text{logistic}}(\mathbf{w}, \mathbf{x}_i, y_i)}}_{\hat{R}(\mathbf{w})}$$

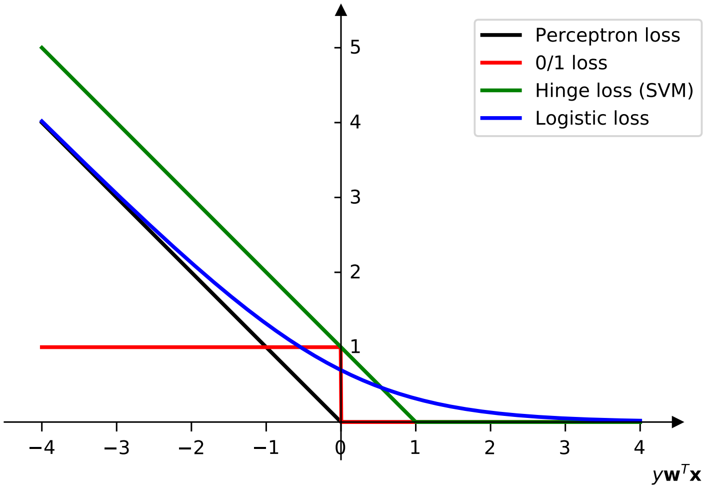
MLE for logistic regression

- Negative log likelihood (=objective) function given by

$$\hat{R}(\mathbf{w}) = \sum_{i=1}^{n} \log\left(1 + \exp(-y_i \mathbf{w}^T \mathbf{x}_i)\right)$$

- The logistic loss is convex
  $\Rightarrow$ Can use convex optimization techniques (e.g., SGD)

Logistic loss vs other losses



Gradient for logistic regression

- Loss for data point $(\mathbf{x}_i, y_i)$

$$\ell(\mathbf{w}) = \log\left(1 + \exp(-y_i \mathbf{w}^T \mathbf{x}_i)\right)$$

- Gradient for data point $(\mathbf{x}_i, y_i)$

$$\begin{aligned}
\nabla_{\mathbf{w}} \ell(\mathbf{w}) &= \frac{1}{1 + \exp(-y_i \mathbf{w}^T \mathbf{x}_i)} \exp(-y_i \mathbf{w}^T \mathbf{x}_i)(-y_i \mathbf{x}_i) \\
&= \frac{\exp(-y_i \mathbf{w}^T \mathbf{x}_i)}{1 + \exp(-y_i \mathbf{w}^T \mathbf{x}_i)}(-y_i \mathbf{x}_i) \\
&= \underbrace{\frac{1}{1 + \exp(y_i \mathbf{w}^T \mathbf{x}_i)}}_{=P(Y \neq y_i | x_i)}(-y_i \mathbf{x}_i)
\end{aligned}$$

SGD for logistic regression

SGD for logistic regression

- Initialize $\mathbf{w}$
- For $t = 1, 2, \ldots$
  - Pick data point $(\mathbf{x}, y)$ uniformly at random from data $\mathcal{D}$
  - Compute probability of misclassification with current model:

$$\hat{P}(Y = -y|\mathbf{w}, \mathbf{x}) = \frac{1}{1 + \exp(y\mathbf{w}^T\mathbf{x})}$$

  - Take gradient step:

$$\mathbf{w} \leftarrow \mathbf{w} + \eta_t \, y \, \mathbf{x} \, \hat{P}(Y = -y|\mathbf{w}, \mathbf{x})$$

Logistic regression and regularization

- Similar to SVMs and linear regression, want to use regularizer to control model complexity
- Thus, instead of solving MLE

$$\min_{\mathbf{w}} \sum_{i=1}^{n} \log\left(1 + \exp(-y_i \mathbf{w}^T \mathbf{x}_i)\right),$$

estimate MAP/solve regularized problem

- L2 (Gaussian prior):

$$\min_{\mathbf{w}} \sum_{i=1}^{n} \log\left(1 + \exp(-y_i \mathbf{w}^T \mathbf{x}_i)\right) + \lambda \|\mathbf{w}\|_2^2$$

- L1 (Laplace):

$$\min_{\mathbf{w}} \sum_{i=1}^{n} \log\left(1 + \exp(-y_i \mathbf{w}^T \mathbf{x}_i)\right) + \lambda \|\mathbf{w}\|_1$$

SGD for L2-regularized logistic regression

SGD for L2-regularized logistic regression

- Initialize $\mathbf{w}$
- For $t = 1, 2, \ldots$
  - Pick data point $(\mathbf{x}, y)$ uniformly at random from data $\mathcal{D}$
  - Compute probability of misclassification with current model:

$$\hat{P}(Y = -y|\mathbf{w}, \mathbf{x}) = \frac{1}{1 + \exp(y\mathbf{w}^T\mathbf{x})}$$

  - Take gradient step:

$$\mathbf{w} \leftarrow \mathbf{w}(1 - 2\lambda\eta_t) + \eta_t \, y \, \mathbf{x} \, \hat{P}(Y = -y|\mathbf{w}, \mathbf{x})$$

Regularized logistic regression

- Learning:
  - Find optimal weights by minimizing logistic loss + regularizer:

$$\hat{\mathbf{w}} = \underset{\mathbf{w}}{\operatorname{argmin}} \sum_{i=1}^{n} \log\left(1 + \exp(-y_i \mathbf{w}^T \mathbf{x}_i)\right) + \lambda \|\mathbf{w}\|_2^2$$

$$= \underset{\mathbf{w}}{\operatorname{argmax}}\, P(\mathbf{w} | \mathbf{x}_1, \ldots, \mathbf{x}_n, y_1, \ldots, y_n)$$

- Classification:
  - Use conditional distribution:

$$P(y|\mathbf{x}, \hat{\mathbf{w}}) = \frac{1}{1 + \exp(-y \hat{\mathbf{w}}^T \mathbf{x})}$$

  - E.g., predict more likely class label

$$\underset{y}{\operatorname{argmax}}\, P(y|\mathbf{x}, \hat{\mathbf{w}}) = \operatorname{sign}(\hat{\mathbf{w}}^T \mathbf{x})$$

Logistic regression demo

[probabilistic-modeling/lr-demo.ipynb]

More remarks on logistic regression

- Can kernelize (kernelized logistic regression)
- Can apply logistic loss function to neural networks, in order to have them output probabilities
- Natural multi-class variants
- ...

Kernelized logistic regression

- Learning:
  - Find optimal weights by minimizing logistic loss + regularizer:

$$\hat{\alpha} = \underset{\alpha}{\operatorname{argmin}} \sum_{i=1}^{n} \log\left(1 + \exp(-y_i \alpha^T \mathbf{k}_i)\right) + \lambda \alpha^T \mathbf{K} \alpha$$

- Classification:
  - Use conditional distribution:

$$P(y|\mathbf{x}, \hat{\alpha}) = \frac{1}{1 + \exp(-y \sum_{j=1}^{n} \hat{\alpha}_j k(\mathbf{x}_j, \mathbf{x}))}$$

  - E.g., predict more likely class label

Multi-class logistic regression

- Can extend logistic regression to multi-class setting
- Maintain one weight vector per class and model:

$$P(Y = i | \mathbf{x}, \mathbf{w}_i, \ldots, \mathbf{w}_c) = \frac{\exp(\mathbf{w}_i^T \mathbf{x})}{\sum_{j=1}^{c} \exp(\mathbf{w}_j^T \mathbf{x})}$$

Multi-class logistic regression

- Can extend logistic regression to multi-class setting
- Maintain one weight vector per class and model:

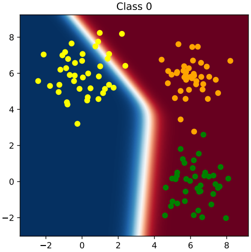
$$P(Y = i | \mathbf{x}, \mathbf{w}_i, \ldots, \mathbf{w}_c) = \frac{\exp(\mathbf{w}_i^T \mathbf{x})}{\sum_{j=1}^{c} \exp(\mathbf{w}_j^T \mathbf{x})}$$

- Not unique – can force uniqueness by setting $\mathbf{w}_c = 0$ (this recovers logistic regression as special case)
- Corresponding loss function (cross-entropy loss):

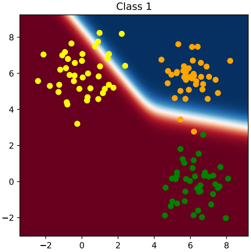
$$\ell(y; \mathbf{x}, \mathbf{w}_1, \ldots, \mathbf{w}_c) = -\log P(Y = y | \mathbf{x}, \mathbf{w}_1, \ldots, \mathbf{w}_c)$$

Illustration

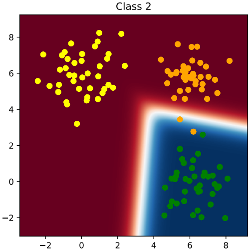
$P(Y = 0|\mathbf{x})$



$P(Y = 1|\mathbf{x})$



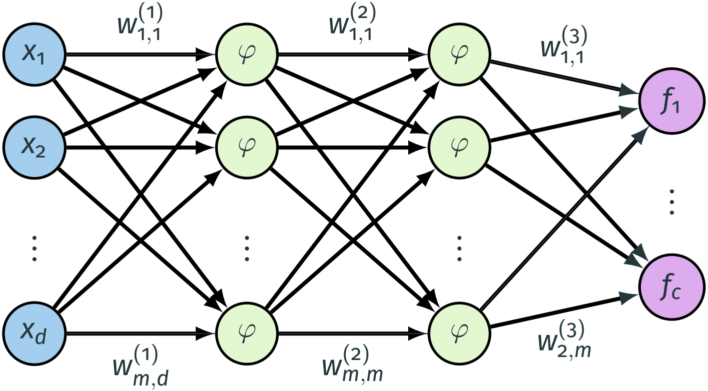
$P(Y = 2|\mathbf{x})$



high probability low probability

[probabilistic-modeling/lr.py]

Training neural nets for multi-class



Loss: $\ell(Y = i; f_1, \dots, f_c) = -\log \frac{\exp(f_i)}{\sum_{j=1}^{c} \exp(f_j)} =$
$-\log \exp(f_i) + \log \sum_{j=1}^{c} \exp(f_j)$

SVM vs. Logistic regression

Method	*SVM / Perceptron*	*Logistic regression*
Advantages	Sometimes higher classification accuracy; Sparse solutions	Can obtain class probabilities
Disadvantages	Can't (easily) get class probabilities	Dense solutions

Outlook: Bayesian learning and inference

Outlook: Bayesian learning

"Optimization" based learning (MAP, MLE, ...):

$$\hat{\mathbf{w}} = \operatorname*{argmax}_{\mathbf{w}} P(\mathbf{w}|\mathcal{D}) \qquad\qquad P(y|\mathbf{x}, \hat{\mathbf{w}})$$

Ignores uncertainty in model

Optimization typically efficient

Outlook: Bayesian learning

"Optimization" based learning (MAP, MLE, ...):

$$\hat{\mathbf{w}} = \operatorname*{argmax}_{\mathbf{w}} P(\mathbf{w}|\mathcal{D}) \qquad\qquad P(y|\mathbf{x}, \hat{\mathbf{w}})$$

Ignores uncertainty in model

Optimization typically efficient

"Integration" based learning / Bayesian model averaging:

$$P(y|\mathbf{x}, \mathcal{D}) = \int P(y|\mathbf{x}, \mathbf{w})P(\mathbf{w}|\mathcal{D})$$

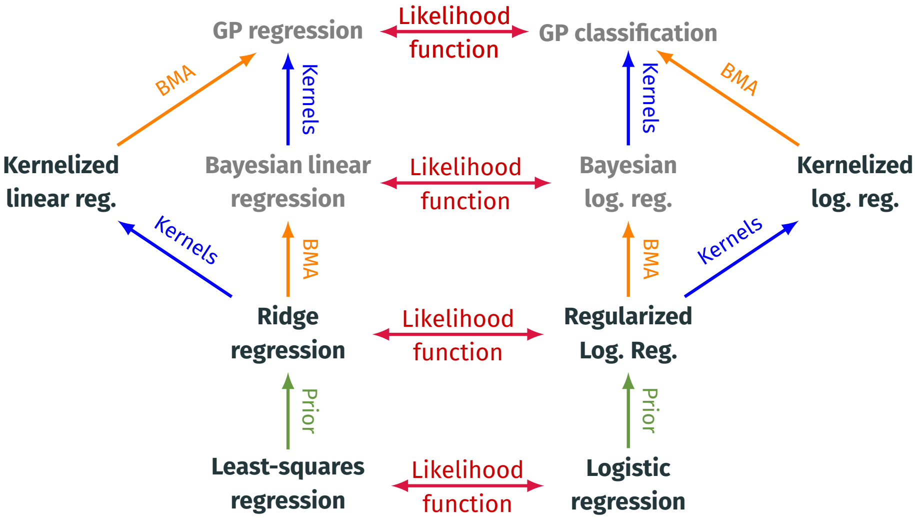
Quantifies uncertainty in model

Integration typically intractable

A statistical perspective on supervised learning

Wrap-up

Probabilistic modeling big picture so far



What we've seen so far

Representation/ features	Linear hypotheses, non-linear hypotheses through feature transformations, kernels, learn nonlinear features via neural nets
Probabilistic/ Optimization model	**Likelihood** / Loss-function: Squared loss=Gaussian lik., ℓ_p loss, 0/1 loss, Perceptron loss, Hinge loss, cost-sensitive loss, multi-class hinge loss, reconstruction error, logistic loss=Bernoulli lik., cross-entropy loss=Categorical lik. * **Prior** / + Regularization: ℓ_2 norm (=Gaussian prior), ℓ_1 norm (=Laplace prior), ℓ_0 penalty, early stopping, dropout
Method	Exact solution, Gradient Descent, (mini-batch) SGD, Greedy selection, reductions, Lloyd's heuristic, Bayesian model averaging
Evaluation metric	Empirical risk = (mean) squared error, Accuracy, F1 score, AUC, confusion matrices, compression performance, log-likelihood on validation set
Model selection	k-fold cross-validation, Monte Carlo cross-validation, Bayesian model selection

References

- “The Elements of Statistical Learning”, Chapters 7.1–7.3
- “Pattern Recognition and Machine Learning”, Chapters 3.2 and parts of 3.3