Machine Learning 1

Review of week 1-3

Patrick Forré

1. Review of the most important topics from week 1-3 covered in lecture 08. This is an incomplete list! The midterm exam will cover more than these slides!!!

Different Types of Learning: Supervised Learning

What is? (Supervised Learning)

We talk about <u>*Supervised Learning*</u>*, if for the known data cases* $x_1, \ldots, x_n$ *we also know the target variables* $t_1, \ldots, t_n$*. The task now is to make a good prediction for the target variable* t *for new data* x*, where* t *is not known anymore.*

This boils down to estimating a function f *such that for all known and unknown(!)* (x, t) *we have* $f(x) \approx t$*.*

Definition (Classification and Regression)

- If t is a discrete variable (i.e. takes values in a countable or finite set like $\{0, 1\}$) this task is called <u>*Classification*</u>.
- If t is a continuous variable (i.e. takes values in $\mathbb{R}$ or $\mathbb{R}^d$) it is called <u>*Regression*</u>.

Different Types of Learning: Unsupervised Learning

What is? (Unsupervised Learning)

We talk about Unsupervised Learning if for the known data cases $x_1, \ldots, x_n$ no target variables are given.

The task now is to find an "inner representation" of the known data to make it more accessible and such that new data x can relate to it.

Typical approaches are Clustering, Dimensionality Reduction, Density Estimation.

Which approach to use depends on our application in mind.

Remark

Clustering can in some cases be seen as "unsupervised classification", and dimensionality reduction as a kind of "unsupervised regression".

The Rules of Probability Theory

Theorem (The Rules of Probability Theory)

For random variables $X \in \mathcal{X}$ and $Y \in \mathcal{Y}$ we have the following rules:

	discrete RV	*continuous RV*
σ-Additivity	$\mathbb{P}(X \in A) = \sum_{x \in A} p(x)$	$\mathbb{P}(X \in A) = \int_A p(x)dx$
Positivity	$p(x) \geqslant 0$	$p(x) \geqslant 0$
Normalization	$\sum_{x \in \mathcal{X}} p(x) = 1$	$\int_{\mathcal{X}} p(x)dx = 1$
Sum Rule	$p(x) = \sum_{y \in \mathcal{Y}} p(x, y)$	$p(x) = \int_{\mathcal{Y}} p(x, y)dy$
Product Rule	$p(x, y) = p(x\|y) \cdot p(y)$	$p(x, y) = p(x\|y) \cdot p(y)$

Bayes' Rule

Theorem (Bayes)

For random variables $X \in \mathcal{X}$ and $Y \in \mathcal{Y}$ we have the following rule:

$$p(y|x) = \frac{p(x|y) \cdot p(y)}{p(x)} = \begin{cases} \frac{p(x|y) \cdot p(y)}{\sum_{y' \in \mathcal{Y}} p(x|y') \cdot p(y')} & \text{for discrete RV} \\ \frac{p(x|y) \cdot p(y)}{\int_{\mathcal{Y}} p(x|y') \cdot p(y') dy'} & \text{for continuous RV} \end{cases}$$

I.a.w. the conditioning can be "exchanged" by this rule.

In the context of Bayesian inference (see later) we call:

- $p(y)$: the <u>prior probability</u> of Y (i.e. before observing x).
- $p(y|x)$: the <u>posterior probability</u> of Y (i.e. after observing x).
- $p(x|y)$: the <u>likelihood</u> of $X = x$ given $Y = y$.
- $p(x)$: the <u>evidence</u> for $X = x$.

Independent Random Variables

Definition (Independence)

Two random variables $X \in \mathcal{X}$ and $Y \in \mathcal{Y}$ are called <u>independent</u> if for all values x, y we have:

$$p(x, y) = p(x) \cdot p(y).$$

This is equivalent to saying that for all x and y (with $p(y) > 0$) we have:

$$p(x|y) = p(x).$$

In words: X and Y are independent iff measuring X gives no information about Y, and vice versa.

Multivariate Gaussian

Definition (Multivariate Gaussian distribution in D dimensions)

A vector valued random variable $X = (X_1, \ldots, X_D)^T$ is said to be <u>multivariate Gaussian distributed</u> with parameters $\mu = (\mu_1, \ldots, \mu_D)^T$ and $\Sigma = (\Sigma_{ij})_{i,j}$ if X has the density in $x = (x_1, \ldots, x_D)^T$:

$$\mathcal{N}(x|\mu, \Sigma) := \frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right),$$

where Σ is a $D \times D$ covariance matrix and $|\Sigma|$ its determinant.
We have: $\mathbb{E}[X] = \mu$ and $\mathrm{Cov}(X) = \Sigma$.
Note that $\mathcal{N}(x|\mu, \Sigma)$ has $D(D+3)/2$ parameters.

Maximum Likelihood Estimation

- Data set $D = (x_1, \ldots, x_N)$ of N independent observations given.
- We are presented with a class of probability distributions $\{p(x|w) | w \in \mathcal{W}\}$ for x, where $\mathcal{W}$ is an index set (in some $\mathbb{R}^d$).
- The <u>Maximum Likelihood Estimator</u> w_{ML} is determined by:

$$\begin{aligned} w_{\mathrm{ML}} &:= \operatorname{argmax}_{w \in \mathcal{W}} p(D|w) \\ &= \operatorname{argmax}_{w \in \mathcal{W}} \prod_{i=1}^{N} p(x_i|w) \\ &= \operatorname{argmax}_{w \in \mathcal{W}} \sum_{i=1}^{N} \log p(x_i|w) \\ &= \operatorname{argmin}_{w \in \mathcal{W}} \left\{ -\sum_{i=1}^{N} \log p(x_i|w) \right\} \end{aligned}$$

Bayesian Prediction

- Data set $D = (x_1, \ldots, x_N)$ of N independent observations given.
- We are presented with a class of probability distributions $\{p(x|w) | w \in \mathcal{W}\}$ for x, where $\mathcal{W}$ is an index set (in some $\mathbb{R}^d$).
- Goal: Estimate the distribution of a new data point x'.
- Bayesian Principle: Instead of searching for one optimal w consider all $w \in \mathcal{W}$ simultaneously and assign a probability distribution $p(w)$ over it, reflecting the plausability of each w.
- $p(w)$ is called the prior distribution of w.
- Then adjust/update $p(w)$ with the occurence of data D to $p(w|D)$, making some w more plausible and others less (in accordance with D).
- $p(w|D)$ is called the posterior distribution of w after observing D.

Bayesian Prediction (II)

- The posterior $p(w|D)$ can be computed with Bayes' Rule:

$$p(w|D) = \frac{p(D|w)}{p(D)} \cdot p(w).$$

- $p(D|w)$ is called the likelihood and $p(D)$ the evidence.
- Before learning D the predictive distribution is:

$$p(x') = \int_S p(x'|w)p(w)dw.$$

- After learning D the predictive distribution becomes:

$$p(x'|D) = \int_{\mathcal{W}} p(x'|D, w)p(w|D)dw = \int_{\mathcal{W}} p(x'|w)p(w|D)dw.$$

Maximum A Posteriori Probability Estimation

- Data $D = (x_1, \ldots, x_N)$ of N independent observations given.
- Probability distributions $\{p(x|w) | w \in \mathcal{W}\}$ for x given.
- Bayesian setting: probability distribution $p(w)$ given.
- <u>Maximum a Posteriori Principle:</u> The most likely "explanation" of D is not just given by the index w which maximizes $p(D|w)$, but which maximizes the a posteriori $p(w|D) = \frac{p(D|w)p(w)}{p(D)}$.
- The <u>Maximum A Posteriori Estimator</u> w_{MAP} is:

$$\begin{aligned} w_{\text{MAP}} &:= \operatorname{argmax}_{w \in \mathcal{W}} p(w|D) \\ &= \operatorname{argmax}_{w \in \mathcal{W}} p(D|w)p(w) \\ &= \operatorname{argmax}_{w \in \mathcal{W}} \log p(D|w) + \log p(w) \\ &= \operatorname{argmax}_{w \in \mathcal{W}} \textstyle\sum_{i=1}^{N} \log p(x_i|w) + \log p(w). \end{aligned}$$

Supervised Learning: General Concept

- Goal: Predict target variable t from corresponding data x.
- Training Data: We have a data set $D = (x_1, \ldots, x_N)$ of N observations together with their target variables (estimates) $T = (t_1, \ldots, t_N)$ given ($\rightarrow$ supervision).
- Model: We choose a class of functions in x: $\{y(x, w) | w \in \mathcal{W}\}$ as possible prediction function (with the aim $y(x, w) \approx t$).
- Two Error functions:
  - Interested in minimal error on test data (like RMSE or misclassification rate).
  - But we minimize another (regularized) error function on training data (like Ridge/Lasso regularized sum-of-squares, cross-entropy, neg-log-likelihood, neg-log-a-posteriori etc.)
- Methods: Stochastic gradient descent (online), iteratively reweighted least squares (batch).

Supervised Learning: Evaluating the Error in Praxis

Question

How do we measure the error of a prediction for unknown data sets (x, t) (since we do not know the correct target variable t)?

Hold out known data! Given that our known data set $D = (x_1, \ldots, x_N)$ with targets $T = (t_1, \ldots, t_N)$ is big enough then we randomly divide the data set D into three groups:

1. training set ($D_{\mathrm{tr}} \approx 60\%$ of D): Only D_{tr} and T_{tr} will be used for training (i.e. finding the "right" w^* for $y(x, w^*)$).
2. validation set ($D_{\mathrm{val}} \approx 20\%$ of D): D_{val} and T_{val} will be used for monitoring the estimated test error: $E(y(D_{\mathrm{val}}, w^*), T_{\mathrm{val}})$.
3. test set ($D_{\mathrm{test}} \approx 20\%$ of D): D_{test} will only be allowed to be used once (!!!) for reporting the estimated test error: $E(y(D_{\mathrm{test}}, w^*), T_{\mathrm{test}})$.

If D is not big enough, we rely on weaker evaluation techniques.

Linear Basis Function Model with Ridge Regularization

- Training data: $D = (x_1, \ldots, x_N)^T$ with targets $T = (t_1, \ldots, t_N)^T$, where every $x_i \in \mathbb{R}^D$ is a D-dimensional vector $x_i = (x_{i,1}, \ldots, x_{i,D})^T$.
- Fix a number M and choose basis functions/"features" of x: $(\phi_0(x), \ldots, \phi_{M-1}(x))^T =: \phi(x)$, with $\phi_0 \equiv 1$.
- Model functions with parameters $w = (w_0, \ldots, w_{M-1}) \in \mathbb{R}^M$:

$$y(x, w) = \sum_{i=0}^{M-1} w_i \cdot \phi_i(x) = w^T \phi(x).$$

- Minimize the Ridge regularized sum-of-squares error function:

$$E_{\mathrm{RG}}(D, T, w) := \frac{1}{2} \sum_{i=1}^{N} (t_i - y(x_i, w))^2 + \frac{\lambda}{2} \sum_{k=0}^{M-1} |w_k|^2.$$

- Unique minimizer: $w_{\mathrm{RG}} = (\lambda \mathbb{1}_M + \Phi^T \Phi)^{-1} \Phi^T T$, with $N \times M$-matrix Φ with entries $\Phi_{ik} = \phi_k(x_i)$.

Problems: Underfitting and Overfitting

- Underfitting: model not flexible/complex enough (M too low) to capture variability of true function f.

  Detection: both training and test error comparatively high.

  Possible solutions:

  - Increase parameter space $\mathcal{W}$, i.e. complexity M,
  - create additional basis functions / "features" ϕ_j of the data x,
  - measure new meaningful properties of the samples.

- Overfitting: model too flexible (M too big in comparison to number of observations N). It will start to model variance and noise instead of true underlying function.

  Detection: training error low, test error high.

  Possible solutions:

  - get more data (increase N).
  - decrease parameter space $\mathcal{W}$, i.e. lower complexity M,
  - penalize big parameters / coefficients w_i ("Shrinkage", "Weight Decay", "Regularization", "Bayesian Approach").

Model Comparison and Model Selection

Question

If we have different models (e.g. different M, λ etc.) to describe the data which should we choose?

- If we have enough data then we split the data into training, validation and test data and evaluate every model (fully trained on the training set) on the validation set. Choose the one with lowest validation test error.
- If data is scarce one can use S-fold cross validation.
- One could use information criteria, which penalize complexity:
  - Akaike IC (AIC): Choose model with minimal:

$$M - \ln p(D|w_{\mathrm{ML}}).$$

  - Bayesian IC (BIC): Choose model with minimal:

$$\frac{1}{2} M \ln N - \ln p(D|w_{\mathrm{MAP}}).$$

  - Full Bayesian.

Expected Test Error: Bias - Variance - Decomposition

- Let X, ϵ be independent random variables with $\mathbb{E}[\epsilon] = 0$ and $T = h(X) + \epsilon$ and $D = (X_1, \ldots, X_N)$ i.i.d. instances of X and W a noisy parameter "learned" from D and y the predictive function. Then the <u>expected (quadratic) test error</u> is:

$$\begin{aligned} & \mathbb{E}[(T - y(X,W))^2] & \\ = \; & \mathbb{E}[(T - h(X))^2] & (\text{noise})^2 \\ + \; & \mathbb{E}[(h(X) - \mathbb{E}_D[y(X,W)])^2] & (\text{bias})^2 \\ + \; & \mathbb{E}[(\mathbb{E}_D[y(X,W)] - y(X,W))^2] & (\text{variance}) \end{aligned}$$

- Expected Test Error = Bias^2 + Variance + Noise^2,
- Bias: measures the "difference" between desired regression function h and the avarage prediction over all data sets.
- Variance: measures sensitivity of y to particular choice of data set around the average over all data sets.
- Noise: just a constant coming from the variance of ϵ.

Bayesian Model Comparison for Linear Basis Function Model

- Linear Basis Function Models:

$$\mathcal{M}_M = (M; \mathcal{W}_M = \mathbb{R}^M; \phi_0, \ldots, \phi_{M-1}; y(x, w) = w^T \phi(x); \alpha, \beta)$$

- Training data: $D = (x_1, \ldots, x_N)^T$ with targets $T = (t_1, \ldots, t_N)^T$.
- Likelihood: $p(T|w, D, \beta, M) = \prod_{i=1}^{N} \mathcal{N}(t_i | w^T \phi(x_i), \beta^{-1})$.
- Prior: $p(w|\alpha, M) = \mathcal{N}(w|0, \alpha^{-1} \mathbb{1}_M)$.
- Posterior: $p(w|T, D, \alpha, \beta, M) = \mathcal{N}(w|\mu_N, \Sigma_N)$ with:

$$\begin{aligned} \Sigma_N &= (\alpha \mathbb{1}_M + \beta \Phi^T \Phi)^{-1} \\ \mu_N &= \beta \Sigma_N \Phi^T T. \end{aligned}$$

- We get the log Model Evidence (by Bayes' rule):

$$\ln p(T|D, \alpha, \beta, M) = \frac{M}{2} \ln \alpha + \frac{N}{2} \ln \beta - E_{\mathrm{RG}}(\mu_N) + \frac{1}{2} \ln |\Sigma_N| - \ln(2\pi)$$

with $E_{\mathrm{RG}}(\mu_N) = \frac{\beta}{2} ||T - \Phi \mu_N||_2^2 + \frac{\alpha}{2} ||\mu_N||_2^2$.

- Model Selection: Choose the one with highest model evidence.

Linear Discriminant Analysis (LDA) for Multiple Classes

- Given: Training set $D = (x_1, \ldots, x_N)^T$ with targets $T = (t_1, \ldots, t_N)^T$ of K classes $t_i \in \{c_1, \ldots, c_K\}$.
- Prior: $p(c_k) =: q_k,\ k = 1, \ldots, K$.
- LDA-assumption: $p(x|c_k) = \mathcal{N}(x|\mu_k, \Sigma)$ (same Σ for every k).
- (Unbiased) maximum likelihood estimates:

$$\begin{aligned} N_k &:= \#\{1 \leqslant n \leqslant N | t_n = c_k\}, \\ q_{k,\mathrm{ML}} &= \frac{N_k}{N}, \\ \mu_{k,\mathrm{ML}} &= \frac{1}{N_k} \sum_{n:t_n=c_k} x_n, \\ \tilde{\Sigma}_{\mathrm{ML}} &= \frac{1}{N-K} \sum_{k=1}^{K} \sum_{n:t_n=c_k} (x_n - \mu_{k,\mathrm{ML}})(x_n - \mu_{k,\mathrm{ML}})^T, \end{aligned}$$

- Posterior: $p(c_k|x) \approx \sigma_k(w_1^T x + w_{10}, \ldots, w_K^T x + w_{K0})$ with:

$$w_j = \tilde{\Sigma}_{\mathrm{ML}}^{-1} \mu_{j,\mathrm{ML}}, \qquad w_{j0} = -\frac{1}{2} \mu_{j,\mathrm{ML}}^T \tilde{\Sigma}_{\mathrm{ML}}^{-1} \mu_{j,\mathrm{ML}} + \ln q_{j,\mathrm{ML}}.$$

- We assign x to class c_k if $\sigma_k > \sigma_j$ for all $j \neq k$, i.e.:
- Decision regions: $\mathcal{R}_k = \{x | w_k^T x + w_{k0} > w_j^T x + w_{j0}, \forall j \neq k\}$.
- Decision boundaries: $\mathcal{B}_{jk} = \{x | w_j^T x + w_{j0} = w_k^T x + w_{k0}\}$.
- For use of basis functions ϕ_m replace x with $\phi(x)$ everywhere.

Quadratic Discriminant Analysis (QDA) for Multiple Classes

- Given: Training set $D = (x_1, \ldots, x_N)^T$ with targets $T = (t_1, \ldots, t_N)^T$ of K classes $t_i \in \{c_1, \ldots, c_K\}$.
- Prior: $p(c_k) =: q_k,\ k = 1, \ldots, K$.
- QDA-assumption: $p(x|c_k) = \mathcal{N}(x|\mu_k, \Sigma_k)$.
- (Unbiased) maximum likelihood estimates:

$$\begin{aligned} N_k &:= \#\{1 \leqslant n \leqslant N | t_n = c_k\}, \\ q_{k,\mathrm{ML}} &= \frac{N_k}{N}, \\ \mu_{k,\mathrm{ML}} &= \frac{1}{N_k} \sum_{n:t_n=c_k} x_n, \\ \tilde{\Sigma}_{k,\mathrm{ML}} &= \frac{1}{N_k - 1} \sum_{n:t_n=c_k} (x_n - \mu_{k,\mathrm{ML}})(x_n - \mu_{k,\mathrm{ML}})^T, \end{aligned}$$

- Posterior: $p(c_k|x) \approx \sigma_k(a_1(x), \ldots, a_K(x))$ with:

$$a(x) = -\frac{1}{2}|\tilde{\Sigma}_{k,\mathrm{ML}}| - \frac{1}{2}(x - \mu_{k,\mathrm{ML}})^T \tilde{\Sigma}_{k,\mathrm{ML}}^{-1}(x - \mu_{k,\mathrm{ML}}) + \log q_{k,\mathrm{ML}}.$$

- We assign x to class c_k if $a_k(x) > a_j(x)$ for all $j \neq k$, i.e.:
- Decision regions: $\mathcal{R}_k = \{x | a_k(x) > a_j(x), \forall j \neq k\}$.
- Decision boundaries: $\mathcal{B}_{jk} = \{x | a_j(x) = a_k(x)\}$.
- For use of basis functions ϕ_m replace x with $\phi(x)$ everywhere.

Classification with Logistic Regression

- Given: Data set $D = (x_1, \ldots, x_N)^T$ with binary classes $T = (t_1, \ldots, t_N)^T$ with $t_i \in \{c_0, c_1\} = \{0, 1\}$.
- Basis functions: $\phi = \phi(x) = (\phi_0(x), \ldots, \phi_M(x))^T$.
- Model assumption of Logistic Regression: $p(c_1|\phi, w) = \sigma(w^T\phi)$.
- Minimizing the cross-entropy error:

$$\begin{aligned} E(w) &= -\ln p(T|\Phi, w) \\ &= -\sum_{n=1}^{N} \left[t_n \ln y_n + (1 - t_n) \ln(1 - y_n)\right]. \end{aligned}$$

- Using stochastic gradient descent or iterative reweighted least squares we end up with a approximate minimizer w^* of $E(w)$.
- We assign a new data point x to class c_1 if $\sigma((w^*)^T\phi(x)) > \frac{1}{2}$, i.e. if $(w^*)^T\phi(x) > 0$.
- Decision regions: $\mathcal{R}_1 = \{x|(w^*)^T\phi(x) > 0\}$ and $\mathcal{R}_0 = \{x|(w^*)^T\phi(x) < 0\}$.
- Decision boundaries: $\mathcal{B} = \{x|(w^*)^T\phi(x) = 0\}$.

Logistic Regression for multiple classes

- Data $D = (x_1, \ldots, x_N)^T$ with $T = (t_1, \ldots, t_N)^T$ of K-dim one-vs-the-rest vectors $t_i = (0, \ldots, 1, \ldots, 0)^T$.
- Model assumption of Logistic Regression:

$$p(c_k|\phi, w_1, \ldots, w_k) = \sigma_k(w_1^T\phi, \ldots, w_K^T\phi),$$

  with weight vectors $w_k = (w_{k,0}, \ldots, w_{k,M}) \in \mathbb{R}^{M+1}$.
- Put $y_{nk} := \sigma_k(w_1^T\phi(x_n), \ldots, w_K^T\phi(x_n))$.
- Minimize the cross-entropy error w.r.t. w:

$$E(W) = -\ln p(T|\Phi, W) = -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk} \ln y_{nk}.$$

- Gradient: $\nabla_{w_j} E(W) = \sum_{n=1}^{N}(y_{nj} - t_{nj})\phi(x_n)$
- Hessian: $\nabla_{w_k}\nabla_{w_j} E(W) = -\sum_{n=1}^{N} y_{nk}(\mathbb{1}_{nj} - y_{nj})\phi(x_n)\phi(x_n)^T$.
- We assign x to class c_k if $\sigma_k > \sigma_j$ for all $j \neq k$, i.e.:
- Decision regions: $\mathcal{R}_k = \{x|(w_k^*)^T\phi(x) > (w_j^*)^T\phi(x), \forall j \neq k\}$.
- Decision boundaries: $\mathcal{B}_{jk} = \{x|(w_j^*)^T\phi(x) = (w_k^*)^T\phi(x)\}$.