# Cribsheet for Machine Learning (COMS30035)

James Cussens

November 13, 2024

## 1 Machine Learning Principles

You need to know what the following terms mean:

- Unsupervised learning
- Supervised learning
- Regression
- Classification
- Underfitting
- Overfitting
- Model selection
- Training dataset
- Validation dataset
- Test dataset
- Cross-validation
- No free lunch theorem
- Model parameters
- Parametric model
- Nonparametric model
- Likelihood function
- Maximum likelihood estimation (MLE)

## 2 Linear Regression

You need to know that the linear regression model is:

$$p(y|\mathbf{x}, \mathbf{w}) = \mathbf{w}^\top \mathbf{x} + \epsilon \tag{1}$$

where $\epsilon$ has a Gaussian distribution with mean 0: $\epsilon \sim \mathcal{N}(0, \sigma^2)$. You need to know what a *bias* parameter is and how in (1) it was included in the parameter vector $\mathbf{w}$ by the addition of a 'dummy' variable which always has the value 1.

You need to understand that we can apply linear regression to a *feature vector* $\phi(\mathbf{x})$ rather than the original data $\mathbf{x}$:

$$p(y|\phi(\mathbf{x}), \mathbf{w}) = \mathbf{w}^\top \phi(\mathbf{x}) + \epsilon \tag{2}$$

You need to know:

1. what the least-squares problem for linear regression is
2. that the solution to this problem has a closed-form (but you don't need to memorise this closed-form)
3. and that the least-squares solution is also the maximum likelihood solution (you do not need to be able to prove this).

## 3 Linear Discriminant

You need to know that when there are two classes Linear Discriminant computes $y = \mathbf{w}^\top \mathbf{x}$ for input $x$ and assigns $x$ to class $C_1$ if $y \geq 0$ and class $C_2$ otherwise. Parameters are 'learnt' by assuming that: (1) data for each class have a Gaussian distribution, (2) these 2 Gaussian distributions have the same covariance matrix. Parameters can then be found by applying MLE.

## 4 Logistic Regression

You need to know that the *logistic sigmoid function* (sometimes called just the *logistic function*) is:

$$\sigma(a) = \frac{1}{1 + \exp(-a)}$$

You need to know that the logistic regression model for two classes is:

$$p(C_1|\mathbf{x}) = \sigma(\mathbf{w}^\top \mathbf{x}) \qquad p(C_2|\mathbf{x}) = 1 - p(C_1|\mathbf{x}) \tag{3}$$

You need to know that the MLE parameters for logistic regression can be found by gradient descent.

# 5 Neural networks

You need to know what the following terms mean:

- Weights
- Activation function
- Input layer
- Hidden layer
- Output layer
- Cost function / Loss function
- Forward pass
- Backward pass
- Backpropagation
- Vanishing gradient problem
- Exploding gradient problem
- Gradient clipping
- Non-saturating activation functions
- Residual layer / network
- Parameter initialisation
- Early stopping
- Weight decay
- Dropout

You need to know that a unit $j$ in a neural network (but not in the input layer) computes a value $z_j$ by first computing $a_j$, a weighted sum of its inputs (from the previous layer), and then sending $a_j$ to some nonlinear *activation function* $h$:

$$a_j = \sum_i w_{ji} z_i \tag{4}$$

$$z_j = h(a_j) \tag{5}$$

$$\tag{6}$$

You need to know the *backpropagation formula*:

$$\delta_j = h'(a_j) \sum_k w_{kj} \delta_k \tag{7}$$

and be able to explain what each of the symbols in this formula represents.

## 6 Trees

You need to know...

- what a classification tree is
- what a regression tree is
- how trees partition the input space
- that trees are a nonparametric method
- how the standard greedy algorithm (CART) for learning trees works, including the final pruning stage

## 7 Kernels and SVMs

You need to know what the following terms mean

- kernel function
- the kernel trick
- dual parameter
- Gram matrix
- the margin
- support vectors
- a soft margin
- slack variables

You need to know...

- the role of the regularisation parameter in soft margins
- how SVMs can be extended to deal with having more than two classes

## 8 Probabilistic Graphical Models

You need to know what the following terms mean

- Directed acyclic graph
- Conditional independence
- Bayesian network
- the structure of a Bayesian network

- the parameters of a Bayesian network
- child (in a Bayesian network)
- parent (in a Bayesian network)
- descendant (in a Bayesian network)
- path (in a Bayesian network)
- collider (in a Bayesian network)
- blocked path (in a Bayesian network)

You need to know...

- the factorisation of a joint probability distribution defined by the structure of a given Bayesian network
- how to use plate notation to compactly represent a Bayesian network
- how to translate a machine learning model (described in words) to a Bayesian network
- how to use d-separation to check for conditional independence relations in a Bayesian network.

# 9 Bayesian machine learning

You need to know what the following terms mean

- Prior distribution
- Likelihood
- Posterior distribution

You need to know that in the Bayesian approach: the parameters, the data and any unobserved (latent) variables are all represented as random variables in a joint probability distribution. Unknown quantities (parameters and latent variables) are unobserved random variables, know quantities (the data) are observed random variables.

# 10 Sampling and MCMC

You need to know what the following terms mean

- ancestral sampling
- rejection sampling

- Markov chain
- homogeneous Markov chain
- initial distribution (in a Markov chain)
- transition distribution (in a Markov chain)
- Markov chain Monte Carlo
- target probability distribution
- Metropolis-Hastings algorithm
- Metropolis algorithm
- proposal distribution
- acceptance probability
- burn-in
- convergence (in context of MCMC)

You need to know...

- the equations for the acceptance probability for both the Metropolis and Metropolis-Hastings algorithms.
- how a sample from a distribution can be used to approximate an expected value defined by that distribution
- that in MCMC we sample from a **sequence** of distributions and that the samples are not independent
- why we throw away samples in the burn-in
- why we typically run several chains when doing MCMC
- that $\hat{R}$ is a value computed from an MCMC run used to check for convergence; if the run has been successful (i.e. there's been convergence) it will be close to 1.

# 11 k-means and Gaussian mixtures

You need to know what the following terms mean

- clustering
- soft clustering
- Gaussian mixture model

- mixing coefficient
- responsibility (in context of a mixture model)

You need to know...

- how the k-means algorithm works
- that one can do soft clustering by applying MLE to a Gaussian mixture model

# 12 The EM algorithm

You need to know that

- the EM algorithm is an iterative algorithm that attempts to find a value of $\boldsymbol{\theta}$ that maximises the *log-likelihood*: $\ln p(\mathbf{X}|\boldsymbol{\theta})$, where $\mathbf{X}$ is observed data.
- there is no guarantee that the EM algorithm will succeed in maximising the log-likelihood. It may converge to a local maximum which is not a global maximum of the log-likelihood function.

If you are given any or all of the following three EM-related equations:

$$\ln p(\mathbf{X}|\boldsymbol{\theta}) = \mathcal{L}(q, \boldsymbol{\theta}) + \mathrm{KL}(q||p)$$

$$\begin{aligned} \mathcal{L}(q, \boldsymbol{\theta}) &= \sum_{\mathbf{Z}} q(\mathbf{Z}) \ln \left\{ \frac{p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})}{q(\mathbf{Z})} \right\} \\ \mathrm{KL}(q||p) &= -\sum_{\mathbf{Z}} q(\mathbf{Z}) \ln \left\{ \frac{p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta})}{q(\mathbf{Z})} \right\} \end{aligned}$$

you should be able to explain what each of the symbols in these equations represent. It can be helpful to you to simply memorise these three equations.

You need to know that

- $\mathrm{KL}(q||p) \geq 0$ for any choice of $q$, so $\mathcal{L}(q, \boldsymbol{\theta}) \leq \ln p(\mathbf{X}|\boldsymbol{\theta})$.
- In the E-step we increase $\mathcal{L}(q, \boldsymbol{\theta})$ by updating $q$ (and leaving $\boldsymbol{\theta}$ fixed).
- In the M-step we increase $\mathcal{L}(q, \boldsymbol{\theta})$ by updating $\boldsymbol{\theta}$ (and leaving $q$ fixed).
- After the E-step we have $\mathcal{L}(q, \boldsymbol{\theta}) = \ln p(\mathbf{X}|\boldsymbol{\theta})$ (and so $\mathrm{KL}(q||p) = 0$), so that in the following M-step increasing $\mathcal{L}(q, \boldsymbol{\theta})$ will also increase $\ln p(\mathbf{X}|\boldsymbol{\theta})$.

# 13 Hidden Markov models

You need to know that

- A hidden Markov model (HMM) is a state space model where the states are discrete.
- An HMM is defined by an initial distribution over states $p(\boldsymbol{z}_1|\boldsymbol{\pi})$, emission distributions $p(\boldsymbol{x}_n|\boldsymbol{z}_n, \boldsymbol{\phi})$ and state transition distributions $p(\boldsymbol{z}_n|\boldsymbol{z}_{n-1}, \boldsymbol{A})$ where the joint distribution over states $\boldsymbol{Z}$ and observations $\boldsymbol{X}$ is:

  $$p(\boldsymbol{X}, \boldsymbol{Z}|\boldsymbol{A}, \boldsymbol{\pi}, \boldsymbol{\phi}) = p(\boldsymbol{z}_1|\boldsymbol{\pi}) \prod_{n=2}^{N} p(\boldsymbol{z}_n|\boldsymbol{z}_{n-1}, \boldsymbol{A}) \prod_{n=1}^{N} p(\boldsymbol{x}_n|\boldsymbol{z}_n, \boldsymbol{\phi})$$

  where $\boldsymbol{A}$, $\boldsymbol{\pi}$ and $\boldsymbol{\phi}$ are the parameters defining the initial, emission and transition distributions, respectively.
- The observations for an HMM can be discrete or continuous and there is no restriction on the emission distributions.
- If the sequence of states $\boldsymbol{Z}$ is not observed then the EM algorithm can be used to (attempt to) find values of the parameters $(\boldsymbol{\pi}, \boldsymbol{\phi}, \boldsymbol{A})$ which maximise the log-likelihood $\ln p(\boldsymbol{X}|\boldsymbol{A}, \boldsymbol{\pi}, \boldsymbol{\phi})$.
- When using the EM algorithm for HMMs it is not necessary (in the E step) to compute and fully represent the distribution $p(\boldsymbol{Z}|\boldsymbol{X}, \boldsymbol{\theta}^{old})$. Only certain expectations with respect to this distribution are required (for the following M step). These are computed using the forward-backward algorithm.
- The forward pass of the forward-backward algorithm computes for each time-step $n > 1$ and state value $k$: $\alpha(z_{nk}) = p(\boldsymbol{x}_1, ..., \boldsymbol{x}_n, z_n = k|\boldsymbol{\pi}, \boldsymbol{A}, \boldsymbol{\phi})$ $= p(\boldsymbol{x}_n|z_n = k, \boldsymbol{\phi}_k) \sum_{l=1}^{K} A_{lk} \alpha(z_{n-1,l})$
- The backward pass of the forward-backward algorithm computes for each time-step $n < N$ and state value $k$: $\beta(z_{nk}) = p(\boldsymbol{x}_{n+1}, ..., \boldsymbol{x}_N|z_n = k, \boldsymbol{A}, \boldsymbol{\phi}) =$ $\sum_{l=1}^{K} A_{kl} p(\boldsymbol{x}_{n+1}|z_{n+1} = l, \boldsymbol{\phi}_l) \beta(z_{n+1,l})$
- The forward and backward probabilities ($\alpha(z_{nk})$ and $\beta(z_{nk})$) can be used to compute the expectations for the E step (but you don't need to memorise the relevant equations, nor the equations for the M step).
- There is an algorithm called the Viterbi algorithm which can efficiently find, for a particular HMM, the most probable sequence of states given a sequence of observations. You do not need to be able to describe this algorithm.

# 14 Linear Dynamical Systems

You need to know that

- A linear dynamical system (LDS) is a state space model where the states are continuous.
- The initial state, state transition and emission distributions for an LDS are, respectively:
  1. $p(\boldsymbol{z}_1) = \mathcal{N}(\boldsymbol{z}_1|\boldsymbol{\mu}_0, \boldsymbol{V}_0)$;
  2. $p(\boldsymbol{z}_n|\boldsymbol{z}_{n-1}) = \mathcal{N}(\boldsymbol{z}_n|\boldsymbol{A}\boldsymbol{z}_{n-1}, \boldsymbol{\Gamma})$;
  3. $p(\boldsymbol{x}_n|\boldsymbol{z}_n) = \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{C}\boldsymbol{z}_n, \boldsymbol{\Sigma})$.

  where $(\boldsymbol{\mu}_0, \boldsymbol{V}_0, \boldsymbol{A}, \boldsymbol{\Gamma}, \boldsymbol{C}, \boldsymbol{\Sigma})$ are the parameters of the LDS.
- The LDS analogue to computing HMM forward probabilities is called the Kalman filter. (You don't need to memorise the relevant equation.)
- The LDS analogue to computing HMM backward probabilities is called the Kalman smoother. (You don't need to memorise the relevant equation.)
- There is no need for an analogue to the Viterbi algorithm for LDS, since the most likely sequence of states is just the sequence of most probable states.

# 15 Ensemble methods

You need to know what the following terms mean:

- Model selection
- Bayesian model averaging
- Bagging
- Boosting
- Stacking
- Hyperparameters
- Random search for good hyperparameters
- Grid search for good hyperparameters
- Random forest
- Conditional mixture model
- Mixture of experts

You do not need to know about

- Bayesian Optimisation

You need to know

- How predictions are made using Bayesian model averaging, namely:

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

- If $E_{COM}$ is the expected error of a simple ensemble which makes predictions as follows: $y_{COM}(\boldsymbol{x}) = \frac{1}{M}\sum_{m=1}^{M} y_m(\boldsymbol{x})$ and $E_{AV}$ is the average expected error of an individual model then $E_{COM} = \frac{1}{M}E_{AV}$ as long as:
  1. The errors of each model have zero mean.
  2. The errors of different models are not correlated.
- Even if the above 2 assumptions do not hold we still have $E_{COM} \leq E_{AV}$.
- How bagging works, as described, for example on slide L15:23/46.
- How AdaBoost works, as described, for example on slide L15:30/46.
- How predictions are made from an AdaBoost ensemble, namely:

$$y_M(\boldsymbol{x}_n) = \sum_{m=1}^{M} \alpha_m y_m(\boldsymbol{x}_n)$$

  where $\alpha_m = \ln\left(\frac{1-\epsilon_m}{\epsilon_m}\right)$ and where $\epsilon_m$ is the weighted error rate of model $m$.
- A random forest is a collection of trees built where each individual tree is built as follows: a dataset of size $N$ is sampled with replacement from the original dataset which has $N$ datapoints; when determining a split only a randomly selected set of $d$ features are considered where $d \ll D$ where $D$ is the total number of features in the data.
- Predictions from random forests are typically made using the mean (for regression) or majority vote (for classification).
- In a Mixture of Experts model where the component models provide a distribution over the target variable, the distribution given by the mixture is:

$$p(t_n|\boldsymbol{x}_n, \boldsymbol{\phi}, \boldsymbol{\pi}) = \sum_{k=1}^{K} \pi_k(\boldsymbol{x}_n)p(t_n|\boldsymbol{x}_n, \boldsymbol{\phi}_k)$$

  Note that the weights $\pi_k(\boldsymbol{x}_n)$ depend on the input $x_n$.
- EM can be used to learn the parameters of a Mixture of Experts model.