50.021 -AI

Alexander Binder

Tuesday, Week 01: Discriminative ML - quick intro

[The following notes are compiled from various sources such as textbooks, lecture materials, Web resources and are shared for academic purposes only, intended for use by students registered for a specific course. In the interest of brevity, every source is not cited. The compiler of these notes gratefully acknowledges all such sources.]

Supervised Discriminative Machine Learning

Learning goals for today

- data is modeled by a probability distribution
- Learning: finding a good predictive mapping of an input space to an output space.
- *good* mapping means: one that minizes an expectation of a loss function using that probability
- probability is unknown, but the expected loss can be approximated using data

  The Goal is to make predictions on Data.

- Classification of Images
  - input: image (how to represent it?)
  - output: concept label
- Classification of sounds
  - input: sound recording (how to represent it?)
  - output: concept label
- Regression of blood pressure based on amount of food, percentage of fibers, sugars,fats, proteins
  - input: a 5-dim vector
  - output: a real number in $[50, 300]$?
- Detection of a some dog races in images
  - input: image
  - output: ?

- Machine Translation
  - input: ?
  - output: ?
- Video captioning
  - input: ?
  - output: ?

Always the same structure:

- x – data sample, from an input space $\mathcal{X}$ get mapped onto an output space output space $\mathcal{Y}$
- Prediction by a mapping $f : \mathcal{X} \to \mathcal{Y}$, $f(x) = y \in \mathcal{Y}$.

Assumption 1: **Supervised Learning:** We know for every input sample what prediction is desired: (x, y) - an input sample x and its ground truth prediction (best possible) y.

Assumption 2: **Loss function:** can measure a loss $l(f(x), y)$ for how wrong our prediction $f(x)$ is relative to its ground truth y. [1]

[1] is groundtruth always unique?

Example for a loss function – classification with C classes. Output space $\mathcal{Y} = \{0, \ldots, C-1\}$. Classification is wrong on (x, y) if $f(x) \neq y$

$$L(f(x), y) = 1[f(x) \neq y] = \begin{cases} 0 & \text{if } f(x) == y \\ 1 & \text{if } f(x) \neq y \end{cases}$$

Example for a loss function – Regression

$$L(f(x), y) = (f(x) - y)^2$$

is 0 if prediction exactly matches the regression groundtruth, error grows quadratically.

What goal to achieve for the prediction? Part 1

What is the long term goal? When we deploy the mapping in application, it should perform well when deployed. Perform well means: We want the model to **have low loss** L **on new, unseen data, that will come in during the application phase**.

How to model **new, unseen data**? Do we need a model? Yes, for formalizing learning goals.

New unseen data can be variable, unpredictable.

Probabilistic Model: there exists a probability distribution P from which the new, unseen data, that will come in during the application

phase (x, y) is drawn. P is not known explicitly usually (imagine: incoming patient profiles, cat images, situations for a self-driving car).

The long term goal is: choose a mapping f such that

$$E_{(x,y)\sim P}[L(f(x), y)] \rightarrow \textbf{minimal}$$

the mapping f will have expected low loss for new, unseen data, that will come in during the application phase. The new data is modeled by $(x, y) \sim P$, so we look at the expected loss under the probability P.

What is if we would know the data generating distribution P in classification problems?

We assumed we do not know the distribution P that generates our new unseen data (x, y). Now suppose:

- we have a two class classification problem $y\ in \mathcal{Y} = \{-1, +1\}$
- we know for every x the distribution $P(y|x)$.

Does that help to achieve our goal – $E_{(x,y)\sim P}[L(f(x), y)] \rightarrow$ **minimal** ?

Lets fix one x. Our mapping $f(x)$ must choose one label $y \in \mathcal{Y} = \{-1, +1\}$ for x. What **expected loss** do we make in point x when we choose $f(x) = +1$ for the loss function $1[f(x) \neq y]$?

Recap what an expectation is:

$$E_{X\sim Q}[f(X)] = \sum_x f(x)Q(x) \text{ if } X \text{ is discrete variable}$$
$$E_{X\sim Q}[f(X)] = \int_x f(x)q(x)dx \text{ if } X \text{ has a density q(x)}$$

$P(y|x)$ is discrete: Y can take only $\{-1, +1\}$. We know

$$\begin{aligned} E_{(x,y)\sim P(x,y)}[1[1 \neq y]|x] &= 1[1 \neq 1]P(y = +1|x) + 1[1 \neq -1]P(y = -1|x) \\ &= 0 + P(y = -1|x) \end{aligned}$$

The loss will be $P(y = -1|x)$.

Same holds, if we choose $f(x) = -1$, the expected loss will be $E[1[-1 \neq y]|x] = P(y = +1|x)$.

Note: if $P(y = +1|x) \neq \{0, 1\}$, then we will always make a non-zero loss, no matter what we choose.

How to minimize our loss ? We must choose the prediction $f(x) \in \{+1, -1\}$ such that $P(f(x) \neq y|x)$ is minimal. So we chose the label y

that has the highest probability $p(y|x)$ in x. That can be expressed as

$$f_{opt}(x) = \text{argmax}_y P(y|x)$$

Takeaway is:

- in general the expected loss $E_{(x,y)\sim P}[L(f(x), y)]$ under the best possible mapping is **not zero**.
- If we know $P(y|x)$, then we can determine the optimal classifier. Its prediction is $f_{opt}(x) = \text{argmax}_y P(y|x)$ – we chose the label y that has the highest probability $P(y|x)$ in x.

  The error in x will be (l.h.s. is general case, r.h.s. is for 2 classes):

$$\sum_{y \neq f_{opt}(x)} P(y|x) = \min(P(y = -1|x), P(y = +1|x))$$

Its error over the whole data space will be the integral of that error in x over the whole data space, where $p(x)$ is the marginal density of $P(y, x)$

$$\int_x \sum_{y \neq f_{opt}(x)} P(y|x)p(x)dx$$

This also holds in similar form if we have $c > 2$ classes.

Back to reality – we do not know $P(x, y)$ – but we can approximate the loss $E_{(x,y)\sim P}[L(f(x), y)]$.

What goal to achieve for the prediction? Part 2

We want to select an f which minimizes this loss, but we cannot compute it. So how to approximate it ?

$$E_{(x,y)\sim P}[L(f(x), y)]$$

If we collect some data $T_k = ((x_1, y_1), \ldots, (x_k, y_k))$ from the same source as the new unseen data from the application phase (and under same conditions), then by central limit theorem

$$E_{(x,y)\sim P}[L(f(x), y)] \approx \frac{1}{k} \sum_{(x_i,y_i)\in T_k} L(f(x_i), y_i)$$

Why ? because if it comes from the same source, and is collected under the same conditions, then T_k are samples drawn from our unknown distribution P used to model data incoming in application phase. Note: for above to hold you need an additional assumption:

For example a statistical independence of the collected data samples (x_i, y_i). **Approximation becomes better as k gets larger**.

Lets define for further use:

$$\hat{E}(L, f, T_k) \stackrel{def}{:=} \frac{1}{k} \sum_{(x_i,y_i)\in T_k} L(f(x_i), y_i)$$

How to learn? - the first, unregularized try

Assumption 3: Have a **training data** set $D_n = ((x_1, y_1), \ldots, (x_n, y_n))$ that is distributed according to P above, such that the approximation $E_{(x,y)\sim P}[L(f(x), y)] \approx \hat{E}(L, f, D_n)$ holds (for example independent data samples). Have a space of mappings: $\mathcal{F}$.

Learning means here: select a mapping f such that the loss is minimized on the average of the training data;

$$\min_f \hat{E}(f, L, D_n) = \min_f \frac{1}{n} \sum_{(x_i,y_i)\in D_n}^{n} L(f(x_i), y_i)$$

This is called **empirical risk minimization** (replace the word loss by risk of making a wrong prediction). Empirical: we minimize on data that we collected ... empirically obtained data. We will see a better way later on (adding regularization).

Learning means here select a mapping from a space (boring?), but "select" does not imply that we have a discrete set of mappings.

Example for a mapping space: thresholded Linear Classifiers with bias (Affine [Linear] classifiers)

$$f(x) = f_{w,b}(x) = sign(g_{w,b}(x))$$
$$g_{w,b}(x) = w \cdot x + b = \sum_{d=1}^{D} w_d x_d + b)$$

Takes a **vector** x, outputs $-1, +1$. $w \cdot x + b$ is a real number, $sign(\cdot)$ does the thresholding to get a prediction in $\{-1, +1\}$. For every set (w, b) of vector $w \in \mathbb{R}^d$ and real number b we have one mapping $f_{w,b}$. [2]

[2] The space of mappings is an image of $\mathbb{R}^d \times \mathbb{R}^1$ (but some values map on the same classifier ... e.g. divide (w, b) by a constant).

Dataset collection for D_n?

Should be done in a way such that

- $(x_i, y_i) \sim P$ holds, that is: the collected data is distributed same as the incoming samples during application phase.

- $E_{(x,y)\sim P}[L(f(x), y)] \approx \hat{E}(L, f, D_n)$ holds, that is: the loss on the collected data approximates the loss under the unknown distribution (independence is sufficient, or weaker conditions that allow for this convergence cf. conditions for convergence of markov chains)

How to measure the performance of what mapping was selected?

Use a evaluation/testing dataset T_n **disjoint (!!!) from the training data** D_n. Measure empirical error $\hat{E}(f, L, T_n)$. It is important: one cannot reliably evaluate the performance on the data D_n used for training. T_n must be disjoint from D_n for reliable performance estimates.

Why we have to evaluate the performance on a testset which is disjoint from the training set?

The reason is: We choose a mapping f based on minimizing loss on the training set. It is very easy to select f such that the loss on the training set becomes 0 or very low, but the loss on data not used for training is very high. The effect of choosing a mapping with very low training error but high error on new unseen data is called **overfitting** (to the training data). Other mappings may have lower loss on evaluation data or unseen data – and thus be better choices, even if their loss on training data is much higher. Lets illustrate this in an example:

What are the key components?

- Supervised Learning: We know for every input sample what prediction is desired: (x, y) - an input sample x and its ground truth prediction y.
- Loss function l: Can measure a loss $l(f(x), y)$ for how wrong our prediction $f(x)$ is relative to its ground truth y.
- a space of mappings: $\mathcal{F}$ to select from
- theoretical assumption: data in application phase is generated from some unknown distribution P - from which we are able to sample data
- Training Data set, sampled from P
- Evaluation/Test Data Set, sampled from P
- an algorithm for selection of $f \in \mathcal{F}$ based on the training data
- performance evaluation on Eval/Test Data

Steps to be done for machine learning

- preprocess your data (training, evaluation, application phase)
- Select a mapping f such that

$$\min_f \hat{E}(f, L, D_n) = \min_f \frac{1}{n} \sum_{(x_i, y_i) \in D_n}^{n} L(f(x_i), y_i)$$

  the loss on the training data is minimized.[3]

- Measure performance of the selected f on evaluation data.

Challenges:

- what space $f \in \mathcal{F}$ to choose?
- which selection algorithm?
- how to preprocess your data before training and for application phase?

[3] the definition of the loss L here does not exclude the possibility of regularization