Lecture 1 Topic

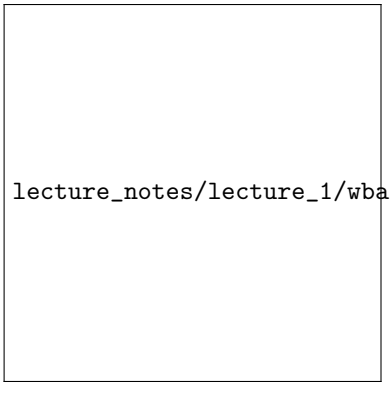
wbg231

December 2022

1 into to ml

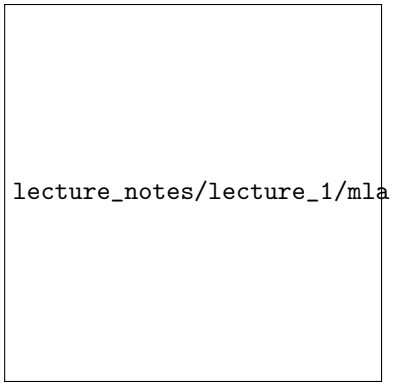
- lecture slides
- in machine learning our goal is typically to solve a prediction problem of the format given an input x predict an output y
- example problem types are binary classification (predict one of 2 classes), multi class classification (predict one of a number of classes), regression (predict an output that is continuous)

1.1 rule based approach

- machine learning should be understood in comparison to a rule based approach with the following work flow (consider for example medical diagnosis)
  - talk to experts (in this case doctors)
  - understand how the experts come up with a diagnosis.
  - implement this process as an algorithm or rule based system (eg given a certain set of symptoms output a diagnosis)
  - try to infer new rules from the rules you are explicitly told
  - 

- this type of approach has a number of pros
  - it leverages preexisting knowledge and expertise
  - generally the rules are interpretable
  - they produce good answers for questions or scenarios included within the knowledge base
- they also have some cons
  - they are labor intensive, and the time of experts is expensive
  - rules don't generalize to novel or not yet seen problems
  - rules do not handle uncertainty well

1.2 machine learning approach

- in contrast to the rule based approach is the ml approach
- instead of explicitly engineering the process that a human expert would use to make a decision we have a machine learn on its own form the inputs and output decision
- in supervised learning we provide training data with input output pairs (x,y) for which the model to learn from
- A machine learning algorithm learns form the training data. takes as input training data and outputs something that we want
- the goal of ml is to find the best (to be defined) prediction function automatically based on the training data
- our ability too do this well depends on two things
  - the availability of large amounts of data for training
  - how well our model will generalize to unseen cases.
- here is a diagram of the machine learning approach
- 

1.3 key concepts

- classification (binary, multi class)
- regression
- prediction function a function that predicts outputs (y) given input (x)
- training data a set of inputs x and output y pairs to train the model with
- supervised learning algorithm takes training data and produces a prediction function
- beyond prediction there are concepts such as unsupervised learning, reinforcement learning and representation learning

1.4 core questions

- the core questions to ask given any machine learning task are
  - model: what class of prediction function are we considering
  - learning how do we learn the best prediction function in this class from our training data
  - inference how do we compute the output of the prediction function for a new input?

2 statistical learning theory

- lecture notes
- in data science problems we generally need to:
  - make a decision (ie move an email to the spam folder)
  - take an action (have a self driving car turn, reject a hypothesis)
  - produce some output (translate a sentence from English to french, whose face is in an image)
  - predict where a storm will be in an hour (output x,y coordinates)
  - an action is the generic term for what is produced ie the output of our system
- what we base our decision off of are either called inputs (in machine learning) or Covariates (in stats) ex: pictures, a search query, stock price history
- inputs are often paired with outputs or labels. (does that picture have an animal in it, suggested URLs, should we sell a stock)
- Decision theory is all about finding optimal actions under carious definitions of optimally (ie evaluation criteria) ex (is the classification correct, how far off was our prediction) .

2.1 formalize set up

- the problem can be formalized as follows
  - observe input x
  - take action a
  - observe outcome y
  - evaluate action in relation to the outcome
- there are three spaces to consider. the input space (X), the action space (A), the outcome space(Y)
- a prediction function (or decision function) takes an input $x \in X$ and produces an action $a \in A$
  $f : X \to A$
  $x \to f(x)$
- a loss function evaluates the action in the context of the outcome y.
  $l : A \times Y \to \mathbb{R}$
  $(a, y) \to l(a, y)$

2.2 evaluating a prediction function

- our goal is to find the optimal prediction function
- the loss function ℓ evaluates a single action
- but in order to fully evaluate a prediction function as a whole we need to use the statistical learning theory frame work

2.3 statistical learning theory framework setup

- our first step is to define a space where the prediction function is applicable
  - assume that there is a data generating process $P_{X \times Y}$
  - all input output pairs (x,y) are generated iid from $P_{X \times Y}$
- we want a prediction function that does well on average ie usually has a small loss.
- to do this we def fine the risk of a prediction function $f : X \to A$ as

$$R(f) = E_{(x,y)\ P_{X \times Y}}[\ell(f(x), y)]$$

  in words this is the expected loss of f over $P_{X \times Y}$

- to further clarify the risk is defined as the expected value of our loss function over the input and output space (X,Y) generated by the data generating process $P_{X\times Y}$
- we can not compute the risk in practice because we don't know the whole data generating process and thus can not compute expectation, but we can estimate it
- The Bayes prediction function $f^* : X \to A$ is a function that ac hives the minimal risk among all possible functions

  $$f^* \in argmin_f R(f)$$

  where the minimum is takes over all functions mapping the input space to the action space.
- the risk of the Bayes factor is called the Bayes risk, even the best function will usually have this as it is ineducable
- the Bayes prediction function is also called the Target function as it is the best prediction function we could get and thus what we are aiming to learn

2.4 example

- consider a multi-classification problem with action and output space A=Y=$\{1...k\}$
- further suppose we take 0-1 loss.

  $$\ell(a, y) = 1(a \neq y) = \begin{cases} 1 & \text{if } a \neq y \\ 0 & \text{otherwise} \end{cases}$$

- thus our risk function is $E[f] = E[1f(x) \neq y] = 0 * P(f(x) = y) + 1 * P(f(x) \neq y) = P(f(x) \neq y)$ ie just the likely would we misclassify something
- so the Bayes prediction function returns the most likely class ie $f^*(x) \in argmax_{1\leq c\leq k} P(y = c|x)$ this makes sense as this by definition would result in the minimum possible probability of making a mistake for each input and thus minimizes our risk function
- despite this we still can not compute risk because we do not know P_{XY} so we try to estimate it
- assume that we have some data of size n $\mathcal{D}_n = ((x_1, y_1)...(x_n, y_n))$ that is drawn iid from P_{XY}

- from the law of large numbers we know that if $z_1 ... z_n$ are drawn at random with expected value $E[z]$ then $lim_{n\to\infty} \frac{1}{n} \Sigma_{i=1}^{n} z_i = E[z]$ with probability 1. in other words if we draw idd samples as our sample size gets larger the average of the sample will approach the mean of the distribution it is drawn from
- the empirical risk of a prediction function $f : X \to A$ with respect to the dataset $\mathcal{D}_n$ is
$$\hat{R}_n(f) = \frac{1}{n} \Sigma_{i=1}^{n} \ell(f(x_i), y_i)$$
- and by the strong law of large numbers $lim_{n\to\infty} \hat{R}_n(f) = R(f)$
- in other words if we take a large enough iid sample from our data generating process our empirical risk will approximate the theoretical risk
- given this it is reasonable to try to minimize our empirical risk
- a function $\hat{f}$ is an empirical risk minimizer if
$$\hat{f} \in argmin_f R_n(f)$$
where the min is takes over all functions mapping from x to a.
- in an ideal world we would want the risk minimizer, but in practice due to finite data the empirical risk minimizer is the best we can hope for

2.5 example

- suppose we are drawing data from $P_{X\times Y}$ such that x is uniformly distributed on [0,1] and y is always 1.
- such a distribution would look like this lecture_notes/lecture_1/plot_1.jpg
- suppose we got data $D_3 = \{.25, .5, .75\}$ and we deffined our prediction function as
$$f(x) = 1(x \in \{.25, .5, .75\}) = \begin{cases} 1 & \text{if } x \in \{.25, .5, .75\} \\ 0 & \text{otherwise} \end{cases}$$

- under a 0/1 loss function f would have an empirical risk of 0 and a risk of 1.
- so in this case empirical loss minimization let a function f that just memoir zed the data, but clearly this does not generalize well.

2.6 Constrained ERM

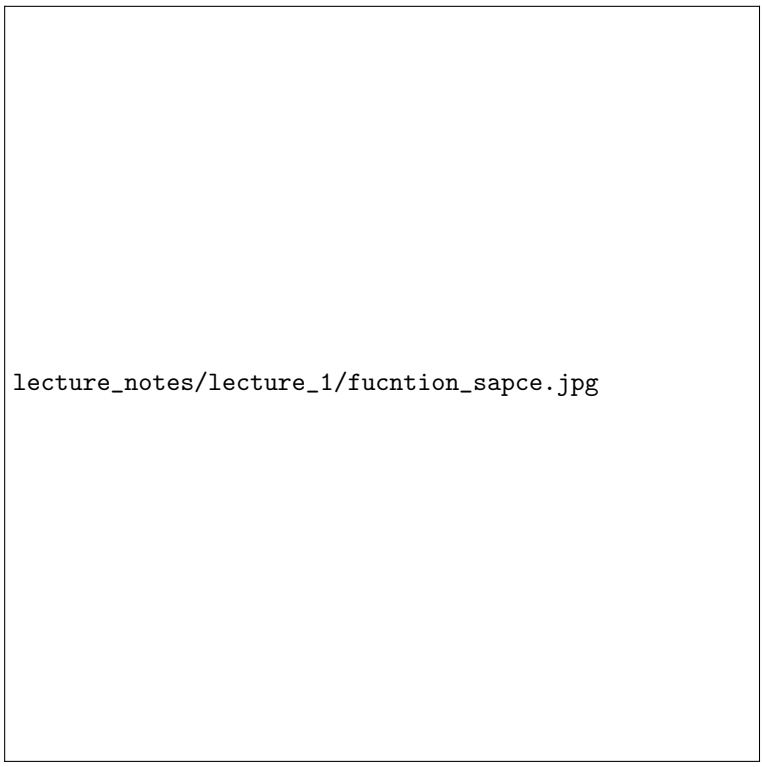
- so we want a way to try to smooth things out, that is to extrapolate information we have from out dataset to the unobserved parts of the input space over all
- one approach is called constrained ERM (empirical risk minimization) in which instead of minimizing empirical risk over all prediction functions we constraint our search to a particular subset of the space of functions called a hypothesis space.
- A hypothesis space $\mathcal{F}$ is a set of prediction functions $X \to A$ that we consider during ERM
- a good hypothesis space should have the following two properties
  - include only those functions that have the desired regularity that is properties we want such as smoothness of simplicity
  - the functions in this space should be easy to work with
- a lot of applies work is about finding a good hypothesis space.
- an empirical risk minimizer over a hypothesis space $\mathcal{F}$ is a function $\hat{f}_n$ such that

  $$\hat{f}_n \in argmin_{f \in \mathcal{F}} \frac{1}{n} \Sigma_{i=1}^{n} \ell(f(x_i), y_i)$$

  ie it is the function that minimizes empirical risk within the hypothesis space.
- a risk minimizer in $\mathcal{F}$ is a function $f_{\mathcal{F}}^*$ such that

  $$f_{\mathcal{F}}^* \in argmin_{f \in \mathcal{F}} E[\ell(f(x), y)]$$

  ie a function that minimizer risk in the hypothesis space.
- the set up for our problem can be visualized like this

- 

- the approximation error (of $\mathcal{F}$) $= R(f_{\{}) - R(f^*)$ so that is the difference between the lowest risk function overall and the lowest risk function within the hypothesis space, think of this as what we lose by choosing a hypothesis space
- the estimation error (of $\hat{f}_n$ in $\mathcal{F}$)$= R(\hat{f}_n) - R(f_{\mathcal{F}})$ so that is the distance between the best function find empirically and the best possible function within your space. think of this as what you lose from your data
- the excess risk compares the risk of f to the Bayes optimal f^* excess risk(f)$= R(f) - R(f^*)$ thus tells us how much more risk a given function has than the overall optimal function
- the excess risk of the ERM $\hat{f}_n$ can be decomposed as excess risk$(\hat{f}_n) = R(\hat{f}_n) - R(f^*) = R(\hat{f}_n) - R(f_{\mathcal{F}}) + R(f_{\{}) - R(f^*)$ which is the estimator error plus the approximation error
- there is a trade off between estimation and approximation error

2.7 approximation error

- approximation error $R(f_{\mathcal{F}}) - R(f^*)$ is a property of the class of $\mathcal{F}$, it is the penalty for restricting to $\mathcal{F}$

- a bigger $\mathcal{F}$ means smaller approximation error,
- approximation error is a non-random variable

2.8 estimation error

- estimation error $R(\hat{f}_n) - R(f_{\mathcal{F}}$ is the performance hit for choosing f using finite training data, further it is what we lose for minimizing empirical rather than true risk.
- with a smaller $\mathcal{F}$ there are less functions to consider and thus we expect a lower estimation error
- under typical conditions with infinite training data estimation error goes to zero estimation error should be a random quantity.

2.9

ERM in practice

- in practice finding the exact optimum within a space is hard, and does not always work out
- so in practice we don't always find $\hat{f}_n \in \mathcal{F}$ that is the function with the lowest risk in our hypothesis space instead we find $\tilde{f}_n \in \mathcal{F}$ and hope it is good enough
- given $\tilde{f}_n$ is the function our optimization method returns and $\hat{f}_n$ is the empirical risk minimize then the optimization error=$R(\tilde{f}_n) - R(\hat{f}_n)$
- so in practice the excess risk decomposition of the function $\tilde{f}_n$ returned by an optimization algorithm is excess risk$(\tilde{f}_n) = R(\tilde{f}_n) + R(f^*) = R(\tilde{f}_n) - R(\hat{f}_n) + R(\hat{f}_n) - R(f_{\mathcal{F}}) + R(f_{\mathcal{F}}) - R(F^*)$ which is the sum of optimization error, estimation error and approximation error
- once again this is not something that we can observe in practice, but instead something to keep in mind during the process.

2.10 ERM overview

- given a loss function $\ell : a \times y \to \mathbb{R}$
- chose a hypothesis space $\mathcal{F}$
- use an optimization method to find an empirical risk minimizer in the hypothesis space $\hat{f}_n \in \mathcal{F}$ such that $\hat{f}_n = argmin_{f \in \mathcal{F}} \frac{1}{n} \Sigma_{i=1}^{n} \ell(f(x_i), y_i)$ or at the least fine an $\tilde{f}_n$ close to it

- the job of the designer is to chose $\mathcal{F}$ that balances approximation and estimation error
- as we get more training data we can pick a larger $\mathcal{F}$