CS 6784 taught by Sarah Dean

Machine Learning in Feedback Systems

FALL 2023

Cornell University Computer Science Department

Contents

Introduction and Overview

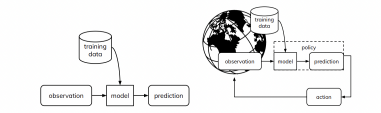


Figure 1: Though machine learning models are often trained with a static supervised learning framing in mind (left), when deployed, they become part of a feedback loop (right).

Machine Learning has a lot of predictive power to offer and thus constitutes an amazing potential tool for decision-making. In a traditional setting, an algorithm will output the insights its understood from the data it received allowing a decision-maker to react accordingly. Removing the decision-maker from the loop by building a system that both predicts and decides in a closed loop brings a whole new set of issues to consider. As Machine Learning models are often deployed as part of large applications, the study of these issues is increasingly consequential.

In feedback systems, a policy is used to produce an action that influences the environment. The policy may contain a model which is learned using training data and/or observations from the environment. The policy can also be rolled out (inference) using observations from the environment.

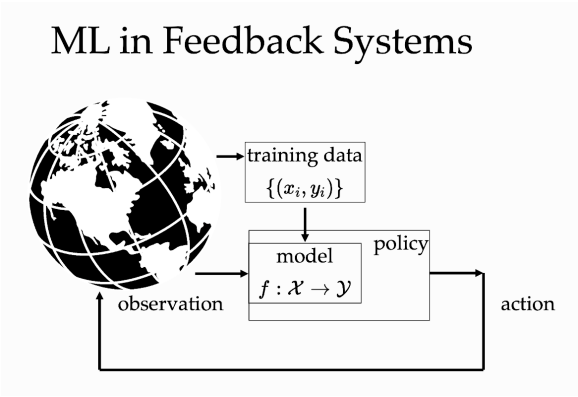


Figure 2: ML in Feedback Systems, from the Lecture 1 slides.

Overview of Machine Learning Frameworks

There are four major ML frameworks which we will deal with in this class: Supervised Learning, Online Learning, Contextual Bandits, and Online Control/RL

Supervised Learning

In a traditional supervised learning setting, a model is trained by sampling data points i.i.d from the distribution we are hoping to make predictions on. We train a predictive model with this dataset. We are aiming to be able to predict as accurately as possible the unknown label corresponding to the known features with a new sample from this distribution.

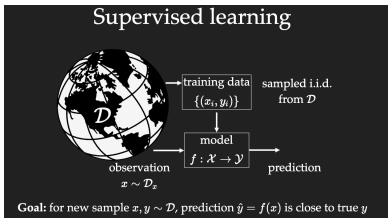


Figure 3: The Supervised Learning framework

Online Learning

We are now dealing with a framework for which we do not have a fixed set of data points drawn i.i.d from a fixed distribution. Our data points are collected progressively over time and we want to constantly be able to make predictions based on the information that we have gathered so far. At each time step t, we get a feature vector x_t, we make a prediction of its label $\hat{y}_t$, we then receive the true label y_t, which we use to calculate how erroneous our prediction function is and use that information to update our model p_t.

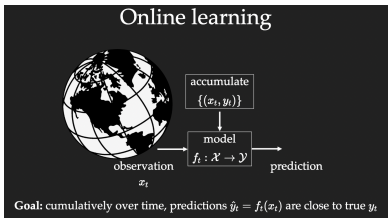


Figure 4: The Online Learning framework

Contextual bandits

In the setting of contextual bandits, similarly to the Online Learning setting we are not dealing with a fixed set of data points coming from a fixed distribution. To this we add another difference: we no longer wish to accurately predict the label of an input vector using a predictive model; instead we wish to learn a policy - a function that will map a context (an input vector) to an action (or an arm). We are no longer trying to minimize a loss function but we want to maximize a cumulative reward. Just like accuracy is the metric in supervised learning, here the metric is the reward - how good or bad this action was for our specific overall objective. Just like the online learning setting these updates to our policy are made progressively as time passes and we accumulate more and more data points. In a given context x_t we take an action a_t, observe a reward r_t, and update our policy π_t.

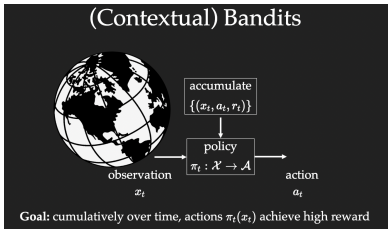


Figure 5: The Contextual Bandits framework

Online control / Reinforcement Learning

Finally, in the setting of Online Control/Reinforcement Learning, we are still in the context of an accumulation of data points that we use to progressively update a policy. However, the big difference with the previous setting is that the environment changes with respect to our actions. We observe a state x_t from a distribution χ_t, pick an action a_t based on our policy π_t, observe a reward r_t that we use to update our

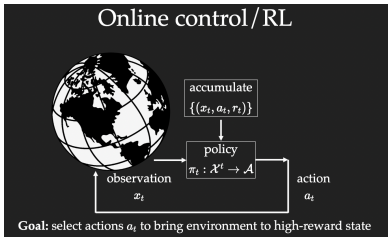


Figure 6: The Reinforcement Learning framework

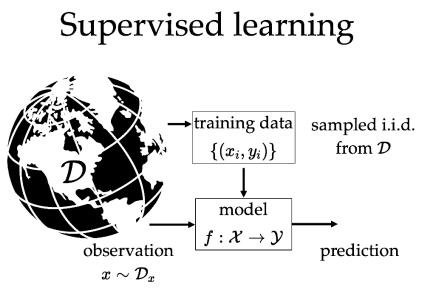
policy. At time $t+1$ however the new state of our environment x_{t+1} is coming from a distribution χ_{t+1}.

Originally scribed by Eliot Shekhtman & Yann Hicke on August 22nd, 2022

1
Supervised Learning

As stated earlier, in the supervised learning framework, the goal is to train a model to learn a relationship between datapoints and their corresponding labels from a sample of datapoints drawn i.i.d. from some fixed distribution $\mathcal{D}$. To evaluate the accuracy of our model, we define a loss function which evaluates how dissimilar a model's prediction is on a datapoint from the truth.

In supervised learning, training data is used to train a model, and a trained model can make predictions on observations. The goal is to learn predictions that are close to the true (often unobserved) y.



Goal: for new sample $x, y \sim \mathcal{D}$, prediction $\hat{y} = f(x)$ is close to true y

Figure 1.1: Supervised Learning, from the Lecture 1 slides.

The goal of supervised learning can be formalized as a risk minimization problem. Given some user defined loss function ℓ, the goal in risk minimization is

$$\min_{p \in \mathcal{P}} \mathcal{R}(p) = \mathbb{E}_{x,y \sim \mathcal{D}}[\ell(y, p(x))] \tag{1.1}$$

where $\mathcal{D}$ is a dataset and $\mathcal{P}$ is the set of candidate models. When we approximate $\mathbb{E}_{\mathcal{D}}$ with an actual dataset and evaluate R over real data points then this becomes empirical risk minimization. The process of supervised machine learning can thus be encapsulated in two things: defining a loss, and conducting risk minimization.

1.1 Predictions and Errors

A loss function $\ell(y, \hat{y})$ measures the loss when a model predicts $\hat{y}$ when the true label is y; this function varies on the setting and desired characteristics of the learning algorithm.

Classification Loss	Usage
Zero-One Loss $\mathbb{1}[y \neq \hat{y}]$	Classification accuracy; this loss is non-continuous and impractical to optimize because of its non convexity.
Hinge Loss $max(0, \hat{y} - y)$	Denotes the margin between the linear separator and its closest points on either class, it is convex but isn't differentiable at $\hat{y} = y$. This is used in the standard SVM.
Log Loss $log(1 + e^{-\hat{y}y})$	Outputs are well-calibrated probabilities for each class; this loss function is used for logistic regression.
Exponential Loss $e^{-\hat{y}y}$	Used in AdaBoost, misprediction loss increases exponentially: this can converge quickly or cause issues when the data is noisy.

Table 1.1: Different classification losses can be picked to handle different optimization schema.

Classification Loss	Usage
Mean Absolute Error $\lvert\hat{y} - y\rvert$	Estimates median label: this loss function is convex and less sensitive to noise but it isn't differentiable at 0.
Squared Loss $(\hat{y} - y)^2$	Estimates mean label: this loss function is convex and differentiable everywhere, but it's sensitive to outliers and noise.

Table 1.2: Different regression losses can also be picked to handle different optimization schema.

Risk can be seen as a natural lens to quantify a model's predictive power.

The loss may vary from sample to sample. The risk of a predictor p over a distribution $\mathcal{D}$ is the expected (average) loss. Risk is mathematically defined in the following way:

$$\mathcal{R}(p) = \mathbb{E}_{x,y\sim\mathcal{D}}[\ell(y, p(x))].$$

In supervised machine learning, we consider the best model to be the one with the lowest risk. The following claim describes when the best prediction for some feature x comes from the conditional expectation of the label y given x.

Claim 1.1.1. *The predictor with the lowest possible risk is:*

- $\mathbb{E}[Y \mid X]$ *for squared loss*

- $1\{\mathbb{E}[Y \mid X] \geq t\}$ *for* $0-1$ *loss, with threshold* t *depending on* $\mathcal{D}$

If all our data were stored in a large table, we could compute the conditional expectation as follows: First, find all the entries with matching features $x_i = x$, and then average the corresponding labels y_i. In practice, we rarely have access to labelled data of the entire distribution—instead, we only have some samples. Furthermore, the feature description x may be so rich and high dimensional that nothing in our finite dataset exactly matches it. We will consider both of these issues later in this chapter.

Loss determines trade-offs between errors, as some variation might be truly unexplainable or our feature set might not be complete! An example of this might be attempting to predict whether a person in a picture is sitting or standing simply based on the position of their face in a frame. This feature set clearly won't give us a 100% accurate classifier.

When we classify, we might classify things correctly (predict stand for a standing person, sit for a sitting person), and here it would make sense for the loss to be 0; however, what if we predict standing for a sitting person versus sitting for a standing person? Would we want to assign both these situations the same loss? We must make this decision based on the motivation of building our model, and based on our priorities.

An example of where we might want to bias this is the proposed idea for skipping TV advertisements in the future: the idea proposed that people could stand up during an ad, and a camera would capture this movement and skip it. Perhaps we care about customer satisfaction, so we might want to give a higher loss to predicting sitting when the person stands, so customers wouldn't return TVs which force them to stand multiple times to skip an ad. Perhaps we care more about advertiser satisfaction, so we might want to give a higher loss to predicting standing when the person sits, so we can ensure that consumers won't accidentally skip ads which they might have acted upon while idly sitting. There are many things to consider for each decision while designing a loss function!

In many domains, decisions have moral and legal significance, and harms can occur at many levels. As machine learning is applied to a variety of settings, we must analyze several possible ways that machine learning algorithms might cause harm in application.

- Correctness: who is burdened by errors?
- Stereotyping: which correlations are permissible?
- Specification: who is left out?

An additional component of these issues is that we don't often have access to the entire population $\mathcal{D}$ and instead use a finite dataset; we revisit this issue towards the end of the chapter.

1.2 Fairness Metrics

Consider the problem of targeted job ads[1], where we want to tell targeted users that we are hiring a programmer. If we use demographic information and browsing history as our input features x_i and whether or not the user clicked (1) or not (-1) as our label y_i, we can use the following linear model to determine whether or not to serve ads to future users.

[1] https://www.theverge.com/2018/10/10/17958784/ai-recruiting-tool-bias-amazon-report

$$\hat{\theta} = \arg\min \sum_{i=1}^{N} (-\theta^T x_i \cdot y_i)_+ \qquad \hat{f}(x) = \mathbb{1}\{\hat{\theta}^T x \geq t\}$$

However, if we optimize this model, we may find that the index of $\hat{\theta}$ corresponding to "visited website for women's clothing store" is negative, which implies some sort of bias in the model. We can try to resolve this by removing the feature for women's clothing, but this will just result in other features being selected that may result in biased models. Clearly, removing features that are potentially problematic is not a solution, and this phenomena is known as "no fairness through unawareness."

1.2.1 Statistical Classification Criteria

A key component of measuring fairness is understanding what and when the model is predicting. Here we demonstrate some metrics that describe this along with an example:

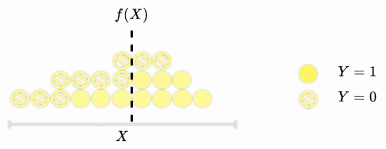


Figure 1.2: Examples of statistical classification criteria.

Accuracy: $\mathbb{P}(\hat{Y} = Y) = 0.75$	Positive Rate: $\mathbb{P}(\hat{Y} = 1) = 0.45$
False Positive Rate: $\mathbb{P}(\hat{Y} = 1 \mid Y = 0) = 0.2$	False Negative Rate: $\mathbb{P}(\hat{Y} = 0 \mid Y = 1) = 0.3$
True Positive Rate: $\mathbb{P}(\hat{Y} = 1 \mid Y = 1) = 0.7$	True Negative Rate: $\mathbb{P}(\hat{Y} = 0 \mid Y = 0) = 0.8$
Positive Predictive Value: $\mathbb{P}(Y = 1 \mid \hat{Y} = 1) = \frac{7}{9}$	Negative Predictive Value: $\mathbb{P}(Y = 0 \mid \hat{Y} = 0) = \frac{8}{11}$

1.2.2 Fairness Frameworks

Here we discuss some frameworks for assessing fairness, a rough idea of the methods to integrate fairness into our model, and the limitations of these methods. Further we mention other instances of discrimination in non classifier models:

Due to the biases that risk minimization models develop, we need some additional criteria besides the loss function to achieve fairness. e.g. in the targeted ad example above, the goal is to treat individuals

roughly the same across groups. To formalize that, we present 3 criteria that measure this goal:

- **independence:**
  equalizes positive rate across groups; prediction does not depend on attribute. $\hat{y} \perp a$
  In the context of the example above, we look at individuals from different racial categories and want to see that predictions look the same, i.e show the ad at equal rate across gender. However, there might be scenarios where this criteria doesn't seem appropriate. As an example, if the predictor is whether somebody is currently pregnant and we would like to show pregnancy-related ad and baby ads. Since there is some underlying difference in pregnancy across genders, this is not the right criteria.

- **separation:**
  equalizes error rate across groups; given outcome, prediction does not depend on attribute. $\hat{y} \perp a \,|y$
  in the example, the ad in this case should be displayed to interested users at equal rates across gender. Here by conditioning on the actual outcome, we allow ourselves to account for the fact that certain properties of interest might differ across these protective attributes.
  This is relying on what the true qualification level in a population is, or what the underlying distribution of your labels is.

- **sufficiency:**
  equalizes predictive value across groups; given prediction, outcome does not depend on attribute. $y \perp a \,|\hat{y}$
  This is saying your predictions are equally useful across all groups by race, or gender or disability, status, or whatever the particular attribute encodes. In the ad example, the users who end up having the ad display to them are interested at equal rates across gender. This is focused less on the truth of the population, and more on the truth in the model, which is what we're designing. We just want to improve that.

1.2.3 Achieving non-Discrimination Criteria

to achieve these fairness criteria we need to process data, the methods are:

- **preprocessing:**
  The benefit of this is diagnostic to the downstream tasks, and it gets rid of any correlations that could otherwise be used, yet it might end up making accuracy hard to achieve. And it requires knowledge of attributes during data pruning.

in the example illustrated in the picture this corresponds to shifting the data itself.

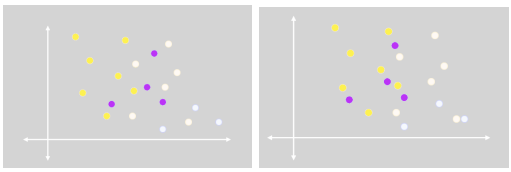


Figure 1.3: data befor and after pre-processing

- **inprocessing:**
  Where you change the learning algorithm itself with respect to these criteria, with respect to these criteria, this one will require that you know the protected attribute during training time.
  so in the example instead of drawing a linear boundary, we would draw a more complicated looking boundary that will actually satisfy the independence criteria, i.e. equal acceptance rate across groups.

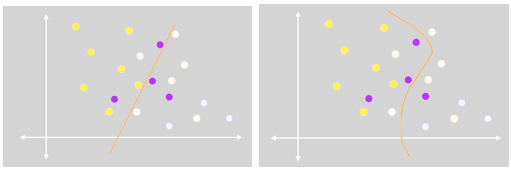


Figure 1.4: the linear boundary and the boundary after in-processing

- **post processing:**
  Where we train a model normally and adjust the thresholds(for binary classification) in a group dependent manner after that. This requires incorporating protected attributes at decision time. example.[2]

[2] See `https://research.google.com/bigpicture/attacking-discrimination-in-ml/` for an interactive post processing example.

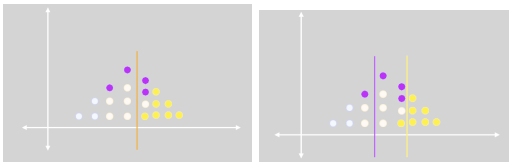


Figure 1.5: linear classifier without and with post-processing

These non-discriminative criteria have some limitations, to mention a few:

- Tradeoffs: It is impossible to simultaneously satisfy separation and sufficiency if populations have different base rates, so we need to decide which ones is of more value to achieve in a specific case. An example of different base rates would be the example

of pregnancy that we discussed. For a controversial example of how sufficiency and separation could be interpreted, refer to the slides.[3]

- Observational: Statistical criteria can measure only correlation; whereas intuitive notions of discrimination involve causation, careful modeling is required that distinguishes between the two.[4]
- Unclear legal grounding: While algorithmic decisions may have disparate impact, achieving criteria involves disparate treatment. And you could take either stance in legal battles.
- Limited view: focusing on risk prediction might miss the bigger picture of how these tools are used by larger systems to make decisions

Fairness related issues happen in non-classifier models as well, there has been such instances in face recognition, image cropping, generative models and search models.

[3] Jon M. Kleinberg, Sendhil Mullainathan, and Manish Raghavan. "Inherent Trade-Offs in the Fair Determination of Risk Scores". In: *8th Innovations in Theoretical Computer Science Conference, ITCS 2017, January 9-11, 2017, Berkeley, CA, USA*. ed. by Christos H. Papadimitriou. Vol. 67. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017, 43:1–43:23

[4] `https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2477899` big data's disparate impact

1.3 *Empirical Risk Minimization: Sample vs Population*

Rarely do we have access to the distribution decribing and entire population; instead we must learn from some dataset. The goal of empirical risk minimization is to find

$$\hat{p} = \min_{p \in \mathcal{P}} \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, p(x_i))$$

This generally states that we attempt to find the model in a model class which has the least loss over the dataset: the average loss over the training dataset is called the empirical risk and is denoted as $\mathcal{R}_N(p)$. Minimizing our risk over the training dataset is useful since we want a model which reduces the overall risk and our training set is all we have access to.

Theorem 1.3.1 (Fundamental Theorem of Supervised Learning)**.**

$$\underbrace{\mathcal{R}(p)}_{\text{risk}} \leq \underbrace{\mathcal{R}_N(p)}_{\text{empirical risk}} + \underbrace{|\mathcal{R}(p) - \mathcal{R}_N(p)|}_{\text{generalization error}}$$

where $\mathcal{R}_N$ is the empirical risk of p over some dataset $\mathcal{D}$. In other words, the true risk of p is bounded by the empirical risk of p plus the generalization error.

the risk associated with our model on the distribution from which its data is sampled from is bounded by the sum of the empirical risk of the model and the model's generalization error. The proof of this theorem merely relies on the fact that the absolute value of a quantity is always at least as big as the value itself.

Proof. Reordering terms,

$$\mathcal{R}(p) - \mathcal{R}_N(p) \leq |\mathcal{R}(p) - \mathcal{R}_N(p)|$$
$$a \leq |a|$$

which is true so thus the theorem holds. □

In general, the risk of the learned model depends on the *representation* of models available to the *optimization* algorithm and is bounded by how well the learned model *generalizes*. In the equation for $\hat{p}$, *representation* corresponds to "$p \in \mathcal{P}$" since $\mathcal{P}$ is the set of models we are optimizing over and hoping represents the data. *Optimization* corresponds to min since min optimizes the objective loss/risk function. Finally, *generalization* corresponds to $|\mathcal{R}(p) - \mathcal{R}_N(p)|$ since this term computes the difference between the true performance of p (including on out-of-sample data points) and the empirical risk over $\mathcal{D}$.

1.4 Least Squares Regression (LSR)

As a case study, let us examine LSR. Linear models might seem limiting at first, but with sufficiently complex kernels[5] linear models can actually be relatively powerful models. In LSR, the objective is to find $\hat{\theta}$ where

[5] e.g. the RBF kernel or the neural tangent kernel (NTK). For more, see Ch 4 of Hardt & Recht, "Patterns, Predictions, and Actions" `mlstory.org`.

$$\hat{\theta} = \arg\min_{\theta \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} (\theta^T x_i - y_i)^2$$

Since the risk (loss) in LSR is the L_2 norm and thus is differentiable and strictly convex, finding the optimal solution to LSR is relatively straightforward.

Definition 1 (Convexity)**.** *A function f is convex iff $\forall$ pairs of points a, b in the domain of f and $\forall 0 \leq t \leq 1$, $f(ta + (1-t)b) \leq tf(a) + (1-t)f(b)$.*

Definition 2 (Strict Convexity)**.** *A function is strictly convex iff $\forall$ pairs of points a, b in the domain of f and $\forall 0 \leq t \leq 1$, $f(ta + (1-t)b) < tf(a) + (1-t)f(b)$.*

Definition 3 (Strong Convexity)**.** *A function is strongly convex iff $\forall$ pairs of points a, b in the domain of f and for any inner product $\langle \cdot, \cdot \rangle$ and corresponding norm $\| \cdot \|$, $\langle \nabla f(x) - \nabla f(y), x - y \rangle \geq m\|x - y\|^2$. Intuitively, for the Euclidean inner product, this means that the growth of the function is lower bounded by some constant proportional to m.*

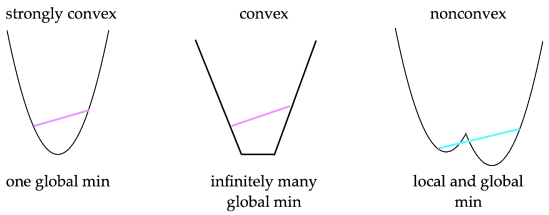


Figure 1.6: Various functions and their (non) convexity properties.

LSR has a closed form solution $\hat{\theta}$ which is pretty rare; most complex models do not have a nice closed form solution. To find the closed form solution of $\hat{\theta}$, consider when the gradient of $\frac{1}{n}\sum_{i=1}^{n}(\theta^T x_i - y_i)$ w/rt to θ is 0.

$$\nabla \frac{1}{n}\sum_{i=1}^{n}(\theta^T x_i - y_i) = \frac{1}{n}\sum_{i=1}^{n}\nabla(\theta^T x_i - y_i)^2 \tag{1.2}$$

$$= \frac{1}{n}\sum_{i=1}^{n} 2x_i(x_i^T\theta - y_i) \tag{1.3}$$

The gradient[6] is equal to 0 when

$$\sum_{i=1}^{n} x_i x_i^T \hat{\theta} = \sum_{i=1}^{n} y_i x_i.$$

[6] We can move the transposes around since $\theta^T x_i$ and y_i are scalars.

This is the *first order optimality condition* for LSR. To solve for $\theta = \hat{\theta}$, let X be a matrix where each row is a data point x_i and Y be a matrix where each row is a label y_i. Then, the first order optimality condition becomes

$$X^T X\hat{\theta} = X^T Y$$

If X is full rank (i.e. the problem is not underspecified), then $X^T X$ is full rank and thus invertible. If X is not full rank (i.e. the problem is underspecified), then we can use the psuedoinverse[7] to get $\hat{\theta}$. Thus,

$$\hat{\theta} = (X^T X)^\dagger X^T Y = (\sum_{i=1}^{n} x_i x_x^T)^\dagger \sum_{i=1}^{n} y_i x_i.$$

[7] Also called the Moore Penrose Inverse, see `https://en.wikipedia.org/wiki/Moore\%E2\%80\%93Penrose_inverse`. The pseudoinverse is denoted with $\dagger$.

The pseudoinverse has the nice property of producing the min-norm $\hat{\theta}$[8]. This can be shown using Lagrange multipliers or projections among other methods. In the case of a fully specified system (i.e. $X^T X$ is invertible) then the pseudoinverse is the inverse and $\hat{\theta}$ is still the min-norm solution since there is only one solution.

[8] Since the system is underspecified, there are infinitely many $\hat{\theta}$'s that achieve the minimum risk.

Thus, we have derived the empirical risk minimizer for this least squares problem. We now consider the population risk. To do so, we start with a *fixed design* generative model, the features describing the setup are fixed and the labels are determined by:

$$y_i = \theta_*^T x_i + v_i$$

where v_i is i.i.d with a mean of 0 and variance of σ^2. The population risk is given by

$$R(\theta) = \frac{1}{n}\sum_{i=1}^{n} \mathop{\mathbb{E}}_{yi}[(x_i)^T\theta - y_i)^2]$$

Due to the random noise on the labels y_i, even the optimal $\theta_\star$ does not have zero risk. We will consider the excess risk, defined as $R(\theta) - R(\theta_*)$ In the exercises, you will prove that the fixed design excess risk of the least squares estimate $\hat{\theta}$ is given by $\frac{\sigma^2 d}{n}$.

This intuitively makes sense as the higher variance the random the noise, the larger the risk. The more dimensions needed will also increase the excess risk displayed as it's more information to gather. Increasing the number of samples will also lower the excess risk to zero as you see a large number of more samples, your understanding of the true distribution will approach the optimal.

Originally scribed by Eliot Shekhtman & Yann Hicke on August 22nd, 2022 and Albert Tseng & Kimia Kazemian on 8/24/2022

1.5 Exercises

Exercise 1. *Prove Claim 1.1.1.*

Proof. This can be proven using the tower property of expectation, also known as the law of total expectation.

Fact 1.5.1. *The Law of Total Expectation states that if X is a random variable whose expected value $\mathbb{E}[X]$ is defined, and Y is any random variable on the same probability space, then $\mathbb{E}[X] = \mathbb{E}[\mathbb{E}[X|Y]]$.*

Giving loss $\ell(f(X), Y)$ of a model on a dataset as a random variable with X as another random variable in the same probability space, we can use this fact to expand:

$$\mathbb{E}_{X,Y\sim\mathcal{D}}[\ell(f(X), Y)] = \mathbb{E}_X[\mathbb{E}_Y[\ell(f(X), Y) \mid X]]$$

For a squared loss we can plug in $\ell(f(X), Y) = (f(X) - Y)^2$, and we may thus say that we are looking for

$$\hat{f} = \arg\min_f \mathbb{E}_X[\mathbb{E}_Y[(f(X) - Y)^2 \mid X]]$$

Since squared loss is convex and the minima of a convex function can be found at the point where the derivative is zero, we may find that:

$$\frac{d}{df(X)}\mathbb{E}_X[\mathbb{E}_Y[(f(X) - Y)^2 \mid X]] = 0$$
$$\mathbb{E}_X[\mathbb{E}_Y[2(\hat{f}(X) - Y) \mid X]] = 0$$

By linearity of expectation, this gives us $\hat{f}(X) = \mathbb{E}_Y[Y|X]$

$$\begin{aligned}\mathbb{E}_X[\mathbb{E}_Y[2(\hat{f}(X) - Y) \mid X]] &= 2(\mathbb{E}_X[\mathbb{E}_Y[\hat{f}(X) \mid X]] - \mathbb{E}_X[\mathbb{E}_Y[Y \mid X]]) \\ &= 0 \\ \mathbb{E}_X[\mathbb{E}_Y[\hat{f}(X) \mid X]] &= \mathbb{E}_X[\mathbb{E}_Y[Y \mid X]] \\ \hat{f}(X) &= \mathbb{E}_X[\mathbb{E}_Y[Y \mid X]]\end{aligned}$$

For a 0/1 loss we can take advantage of the fact that $f(x) \in \{0, 1\}$ to expand the inner expectation:

$$\mathbb{E}_Y[\ell(0, Y) \mid X = x] = \ell(0,0)\mathbb{P}[Y = 0 \mid X = x] + \ell(0,1)\mathbb{P}[Y = 1 \mid X = x]$$
$$\mathbb{E}_Y[\ell(1, Y) \mid X = x] = \ell(1,0)\mathbb{P}[Y = 0 \mid X = x] + \ell(1,1)\mathbb{P}[Y = 1 \mid X = x]$$

Since it is a 0/1 loss, $\ell(1,0) = \ell(1,0) = 1$ and $\ell(0,0) = \ell(1,1) = 0$ so this expansion simplifies to:

$$\mathbb{E}_Y[\ell(0, Y) \mid X = x] = \mathbb{P}[Y = 1 \mid X = x]$$
$$\mathbb{E}_Y[\ell(1, Y) \mid X = x] = \mathbb{P}[Y = 0 \mid X = x]$$

We are looking for the predictor that will always predict the smaller of these two terms for a given x (since we are trying to minimize this inner expectation for our overall risk). We want to predict 0 if $\mathbb{P}[Y = 1 \mid X = x] \leq \mathbb{P}[Y = 0 \mid X = x]$ and 1 if $\mathbb{P}[Y = 1 \mid X = x] \leq \mathbb{P}[Y = 0 \mid X = x]$.
We can use this to rewrite:

$$\hat{f}(x) = 1\{\mathbb{P}[Y = 1 \mid X = x] \geq \mathbb{P}[Y = 0 \mid X = x]\}$$

From the property that $Y \in \{0, 1\}$, we can use $\mathbb{P}[Y = 1 \mid X = x] + \mathbb{P}[Y = 0 \mid X = x] = 1$ to simplify this expression:

$$\hat{f}(x) = 1\{\mathbb{P}[Y = 1 \mid X = x] \geq 1 - \mathbb{P}[Y = 0 \mid X = x]\}$$

By the definition of expectation, we simultaneously have:

$$\begin{aligned}\mathbb{E}[Y \mid X = x] &= 1 * \mathbb{P}[Y = 1 \mid X = x] + 0 * \mathbb{P}[Y = 0 \mid X = x] \\ &= \mathbb{P}[Y = 1 \mid X = x]\end{aligned}$$

Putting this all together and setting $t = 1 - \mathbb{P}[Y = 0 \mid X = x]\}$ we get:

$$\hat{f}(x) = 1\{\mathbb{E}[Y \mid X = x] \geq t\}$$

□

Exercise 2. *For the* fixed design *model, show that the excess risk* $R(\theta) = R(\hat{\theta}) - R(\theta_*) = \frac{\sigma^2 d}{n}$.

Proof.

$$\begin{aligned}\frac{1}{n}\sum_{i=1}^{N}\mathop{\mathbb{E}}_{yi}[(\theta^T x_i - (\theta_*^T x_i + v_i))^2] &= \frac{1}{n}\sum_{i=1}^{N}\mathop{\mathbb{E}}_{yi}[((\theta^T - \theta_*^T)x_i + v_i)^2] \\ &= \frac{1}{n}\sum_{i=1}^{N}\mathop{\mathbb{E}}_{yi}[((\theta^T - \theta_*^T)x_i)^2 - 2(\theta - \theta_*)^T x_i v_i + v_i^2]\end{aligned}$$

The terms within the middle of the expectation are independent under a fixed setting and can be reduced to 0, simplifying it further to

$$\frac{1}{n}\sum_{i=1}^{N}\mathop{\mathbb{E}}_{yi}[(\theta^T x_i - (\theta_*^T x_i + v_i))^2] = \frac{1}{n}\sum_{i=1}^{N}\mathop{\mathbb{E}}_{yi}[((\theta^T - \theta_*^T)x_i)^2 + \sigma^2]$$

Knowing that

$$\theta = (x^T x)^{-1}x^T y = (x^T x)^{-1}x^T(x\theta_* + v)$$

The left side of the expectation can be reduced as follows:

$$\begin{aligned}\mathbb{E}\,||x(\theta - \theta_*)||_2^2 &= \mathbb{E}\,||x[(x^T x)^{-1}x^T(x\theta_* + v) - \theta_*]||_2^2 \\ &= \mathbb{E}\,||x(x^T x)^{-1}x^T v||_2^2\end{aligned}$$

Substituting this back we can further proceed

$$\begin{aligned}\frac{1}{n}\,\mathbb{E}[v^T x(x^T x)^{-1}x^T x(x^T x)^{-1}x^T v] &= \frac{1}{n}\,\mathbb{E}[v^T x(x^T x)^{-1}x^T v] \\ &= \frac{1}{n}\,\mathbb{E}[v^T P v] \\ &= \frac{1}{n}tr(p)\,\mathbb{E}[vv^T] \\ &= \frac{\sigma^2}{n}tr(p) \\ &= \frac{\sigma^2 d}{n}\end{aligned}$$

□

Exercise 3. *For the same fixed design generative model and a new fixed x_{n+1} what is the expected loss $\mathbb{E}_y[(\hat{\theta}^T x_{n+1} - y_{n+1})^2]$? Can you interpret the quantities?*

Proof.

$$\begin{aligned}
\mathbb{E}_y[(\hat{\theta}^T x_{n+1} - y_{n+1})] &= (\hat{\theta}^T x_{n+1})^2 - 2\hat{\theta}^T x_{n+1}\mathbb{E}_y[y_{n+1}] + \mathbb{E}[y_{n+1}^2] \\
&= (\hat{\theta}^T x_{n+1})^2 - 2\hat{\theta}^T x_{n+1}\theta_*^T x_{n+1} + (\theta_*^T x_{n+1})^2 \\
&= ([\hat{\theta}^T - \theta_*^T]x_{n+1})^2 \\
&= ([Y^T X(X^T X)^\dagger - \theta_*^T]x_{n+1})^2 \\
&= ([(\theta_*^T X^T + V^T)X(X^T X)^\dagger - \theta_*^T]x_{n+1})^2 \\
&= (V^T X(X^T X)^\dagger x_{n+1})^2
\end{aligned}$$

□

Exercise 4. *Now consider the random design setting which extends the fixed design model by taking each x_i to be drawn i.i.d from $\mathcal{N}(0, \Sigma)$. We further assume that v_i is also Gaussian (and independent of the features). The risk is then $R(\theta) = \mathbb{E}_{x,y}[(x^T\theta - y)^2]$. What is the excess risk of $\hat{\theta}$ in terms of $X^T X$ and Σ? What is the excess risk in terms of σ^2, n, d?*