Lecture 03: MATH 342W: Introduction to Data Science and Machine Learning

Sergio E. Garcia Tapia[*]

February 4, 2025 (last updated March 8, 2025)

Recap

Last class, we mentioned that one of the assumptions we make in this course is that all phenomena can be described mathematically:

$$y = t(z_1, z_2, \dots, z_t)$$

This has the following components:

- y: the output, phenomenon, response, outcome, or dependent variable.
- t: exact functional relationship
- $z_1, z_2, \dots, z_t$: true, causal input information.

Example of a Phenomenon

We'll consider an example. Let $y \in \{\text{creditworthy}, \text{uncreditworthy}\}$. By creditworthy we mean whether we can expect a person to pay back their loan. We'll encode the as a **binary response**, assigning 1 to creditworthy and 0 to uncreditworthy. The set $\mathcal{Y} = \{0, 1\}$ is called the **response space** or **output space**. We'll make up three z's and their t function:

- z_1: A person has sufficient funds at time of loan is due. (Yes/No), so $z_1 \in \{0, 1\}$.
- z_2: There is an unforeseen emergency (Yes/No), so $z_2 \in \{0, 1\}$.
- z_3: Criminal intentions (do they intend to pay back?) Also a yes/no, so $z_3 \in \{0, 1\}$.

Here's one possible way to specify a functional relationship:

$$y = t(z_1, z_2, z_3) = z_1(1 - z_2)(1 - z_3)$$

This is a product of 0's and 1's, so the output will be 0 or 1 as desired. Note $1 - z_2$ if there is no unforeseen emergency ($z_2 = 0$), then we expect the person to be able to pay back (so that $1 - 0 = 1$).

[*]Based on lectures of Dr. Adam Kapelner at Queens College. See also the course GitHub page.

Put aside for a moment whether this is the exact functional relationship. A foremost issue is that it's impossible for a bank to assess any of this information at the time they give out a loan, or years down the line. To put simply, we don't have access to the z's or the t function.

Modeling the Phenomenon

If we cannot use the z's, then what is the next best thing? *We can get other information about the person that approximates the z's*:

- *Salary* (x_1): If they are making enough money, they might be able to pay back. We'll say x_1 is the annal income now ("today").
- *Previous loan payments* (x_2): Credit score, or did they pay back a previous loan? (Yes/No).
- *Historical criminal record* (x_3): Do they have any misdemeanors, felonies, etc?

It's important to recognize that x_1, x_2, and x_3 do not contain the same information as z_1, z_2, and z_3, but they do approximate the information. Let

$$\mathbf{x} = \begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix} \in \mathcal{X}$$

be a row vector of dimension $\dim[\mathbf{x}] = p = 3$, where p is the number of **predictors**. Here is some nomenclature associated with this vector:

- $\mathbf{x}$: An observation, record, object, or input.
- x_i: A feature, attribute, characteristic, regression, covariate, independent variable, explanatory variable, or predictor.
- $\mathcal{X}$: Input space, or covariate space.

Let's say for this example that:

$$\begin{aligned} &x_1 \in \mathbb{R}, \\ &x_2 \in \{\text{missed a previous loan}, \text{did not miss a previous loan}\}, \\ &x_3 \in \{\text{none, infraction, misdemeanor, felony}\}. \end{aligned}$$

We will encode "missed a previous loan" as 0, and "did not miss a previous loan" as 1.

Categorical Variables

Though x_1 and x_2 can be used directly in a numerical computation, that is not so for x_3; we need to numerically encode it. x_3 is an example of a **categorical variable** (or factor) with four **levels** ($L = 4$). Note we've chosen the levels in x_3 to be mutually exclusive.

There are two common strategies to encode categorical variables:

(a) We can use the mapping:

$$\begin{aligned} 0 &\mapsto \text{none} \\ 1 &\mapsto \text{infraction} \\ 2 &\mapsto \text{misdemeanor} \\ 3 &\mapsto \text{felony} \end{aligned}$$

Then $x_3 \in \{0, 1, 2, 3\}$. An implicit assumption here is that there is an *order* of severity. If a categorical variable's levels are ordered, then this makes sense. Another point which can be problematic is that the numerical values are arbitrary, and it may make a difference in your model. The conclusion is that **ordered factors** (i.e., ordered categorical variables) are difficult.

(b) Alternatively, we can "dummify" the levels. We say x_3 is made up of x_{3a}, x_{3b}, x_{3c}, and x_{3d}:

- $x_{3a} \in \{0, 1\}$, where 0 means *not none* and 1 means *none*.
- $x_{3b} \in \{0, 1\}$, where 0 means *no infraction* and 1 means *an infraction exists*.
- $x_{3c} \in \{0, 1\}$, where 0 means *no misdemeanor* and 1 means *a misdemeanor exists*.
- $x_{3d} \in \{0, 1\}$, where 0 means *no felony*, and 1 means *a felony exists*.

We've inflated x_3 into 4 other variables (one for each level). We do not need all of them, so we can arbitrarily drop any one of them to have a total of $L - 1$ remaining levels. The **dummy** for the variable that we drop is called the **reference variable**.

On a related note, a **nominal categorical variable** has levels that do not have an inherent order of severity. Therefore, it would be inappropriate to code them as $\{1, 2, 3, \ldots\}$. In this example, x_3 is actually an **ordered category variable**. If x_3 were a nominal categorical variable, then we would have no choice but to use (b).

Supervised Learning

Now let's go back to our starting point, and add some more information:

$$y = t(z_1, z_2, x_3) = f(x_1, \ldots, x_p) + \delta$$

Here, f is the "best" functional relationship we can get from the inputs. An important point is that we can never hope to have $t(z_1, z_2, z_3) = f(x_1, \ldots, x_p)$, because the partial information of the x's cannot possibly contain all of the real information of the z's. To account for this we introduce δ, which is what we will call **the error due to ignorance**.

Our next issue is that we need to find f. As it turns out, we cannot solve for f analytically. For example, in a calculus class, you might be asked to find the minimum of a function such as $y = (x - 3)^2$, and you might use techniques such as the first and second derivative test to obtain an exact solution. No such technique applies here to find f. Instead, we must find an **empirical solution**, i.e., through data. The concepts "empirical solution", "learning from historical data", and "supervised learning" are the main focus of this course.

There are three ingredients in supervised learning:

(1) **Training data/historical data**: We will use the notation $\mathbb{D} = \langle X, \vec{\mathbf{y}} \rangle$ for the **historical data**. Here, $\mathbf{y} \in \mathcal{Y}^n$ is an n-length vector, and X is a matrix of the form:

$$X = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_n \end{bmatrix}$$

where each $\mathbf{x}_j$ is a row vector with p features (an *observed input*).

(2) **Hypothesis set**: The next ingredient is some way to reduce the number of ways that the inputs can be combined to produce an output. Ordinarily, there are uncountably many functions that could conceivably be checked. To make it possible for us to move forward, we might hypothesize, for example, that the functional form is linear. We'll use the following notation for our **candidate set of functions**:

$$\mathcal{H} = \{ \text{ candidate functions } h \text{ to approximate optimal function } f. \ \}$$

These are the functions under consideration; this is a choice we make.

(3) **The algorithm**: $\mathcal{A}$ is an algorithm that takes $\mathbb{D}$ and $\mathcal{H}$, and selects $g \in \mathcal{H}$. That is,

$$g = \mathcal{A}(\mathbb{D}, \mathcal{H}) \in \mathcal{H}$$

Note that g will be one of the functions in $\mathcal{H}$.

Approximations and Errors

Let's update our functional relationship equation:

$$\begin{aligned} y = t(z_1, z_2, z_3) &= f(x_1, \ldots, x_p) + \delta \\ &= h^*(x_1, \ldots, x_p) + \epsilon \\ &= g(x_1, \ldots, x_p) + e \end{aligned}$$

The function h^* is known as the **optimal approximation** to f within $\mathcal{H}$. The approximation h^* to f incurs even more error than what is captured by δ; here that is called ϵ. On the subject of errors, there are three main forms of error:

- **Ignorance error**: $y - f = \delta$. This error occurs because we simply do not know the actual drivers (the z's); the features we use as proxies (x's) cannot possibly contain the same information as the z's. Of course, we also do not know the functional relationship t itself. One way to mitigate ignorance error is by increasing the number of proxies $x_1, \ldots, x_p$ to the z's.

- **Misspecification error**: $f - h^*$. This occurs because we specified a set of candidate functions $\mathcal{H}$, which may be too small and thus may miss the correct functional form of f. One way to get around this is to pick a different functional form, i.e., a different $\mathcal{H}$.

- **Estimation error**: $h^* - g$. We can get around this by designing $\mathcal{A}$ better.

Note that the algorithm will not necessarily find h^* because the data $\mathbb{D}$ is insufficient; it will find g. The function g is our fitted model, and now we want to use it for predictions, which we now define:

$$\hat{y}_i = g(\mathbf{x}_i), \quad e_i = y_i - \hat{y}_i, \quad \hat{y}_* = g(\mathbf{x}_*)$$

Here $\hat{y}_i$ is a **observed response**, $\hat{y}_*$ is a **future prediction**, and $\hat{\mathbf{x}}_*$ is a **future observation**. The quantity e_i is called the **residual error**. Our main reason for doing this is the future predictions from the future observations.

Threshold Model

Back to our loan example. Assume $p = 1$; we only have $x_1 = \{\text{salary}\}$. Then the equation

$$\hat{y} = g(x)$$

is called a **univariate model**. Here, $x \in \mathbb{R}$ and $\mathcal{Y} = \{0, 1\}$. Univariate models tend to be high in ignorance error, but we are using it as a pedagogical tool. Here the inputs represent salaries. An output of 0 means they did not pay back the loan, and an output of 1 means they did pay back. One useful abstraction here is the **indicator function**:

$$\mathbb{I}_a(x) = \begin{cases} 1 & \text{if } x = a \\ 0 & \text{if not } x \neq a \end{cases}$$

We can start by picking a **threshold** and use it to make a decision. That is, we'll let our set of candidate functions be:

$$\mathcal{H} = \{\mathbb{I}_{x \geq \theta} : \theta \in \mathbb{R}\}$$

Here, θ is called a **model parameter**, and this model is sometimes known as a **threshold model**. The candidate function $\mathbf{I}_{x \geq \theta}$ means if the condition $x \geq \theta$ is satisfied, the output is 1; otherwise, the output is zero.

The algorithm $\mathcal{A}$ for the threshold model tries to pick a value for θ. One idea involves letting θ vary across all unique historical x's. Thus, we fit n different models (one for each x value in the input). Note that there are uncountably-many x's in reality, so we are approximating this with the n observations we have. After, we compute an "error" for each, measured in the output space. Here are some possibilities:

$$SAE := \sum_{i=1}^{n} |\hat{y}_i - y_i| = \sum_{i=1}^{n} \mathbb{I}_{y_i \neq \hat{y}_i}$$
$$SSE := \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

SAE stands for **sum of absolute error**, and SSE stands for **sum of squared errors**. To compute these values, we apply our model $g = \mathbb{I}_{x \geq \theta}$, for some model parameter θ that we have determined, to each input observation in $x_1, x_2, \ldots, x_n$ in $\mathbb{D}$. Then we compare it against the respective responses $y_1, y_2, \ldots, y_n$ in $\mathbb{D}$. Regardless of whether we choose to measure error with SAE or SSE, we want the error to be small as possible.