Lec 20 Math 390.3 4/25/18

most common ML task btw...

Previously in binary classification, we measured error by

$$MAE = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}_{\hat{y}_i \neq y_i} \quad \text{AKA misclassification error}$$

Does this work for both in-sample & oos? YES

"negative"

"positive"

But this hides a lot of what's going on! There are two types of errors. (1) Saying a true 0 is a 1 (false positive) and (2) " " " " " true 1 is a 0 (false negative).

How can we visualize this? In a 2x2 table:

		prediction ($\hat{y}$) 0	1	
truth (y)	0	TN	FP	#N
	1	FN	TP	#P
		#PN predicted negative	#PP predicted positive	n

$$\text{misclassification error} = \frac{FP + FN}{n}$$

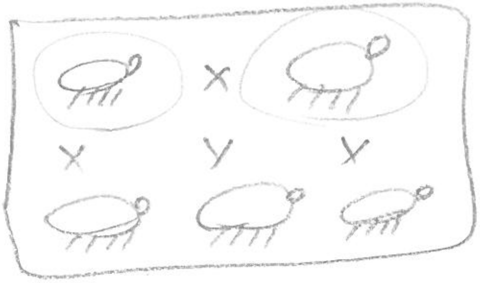
$$\text{Accuracy} = 1 - \text{misclassification error} = \frac{TP + TN}{n}$$

there are many metrics!

$\text{Precision} = \frac{TP}{\#PP}$ the prop. of those you predicted positive are truly positive?

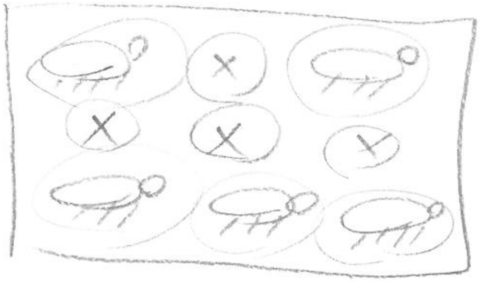
$\text{Recall / Sensitivity} = \frac{TP}{\#P}$ " " " " " " " those truly positive were predicted positive?

Example:



$$\text{Precision} = \frac{2}{2} = 100\%$$

$$\text{Recall} = \frac{2}{5} = 40\%$$



$$\text{precision} = \frac{5}{9} = 56\%$$

$$\text{recall} = \frac{5}{5} = 100\%$$

Both situations are bad! How to combine both together?

$$F_1 \text{ score} := \frac{2}{\frac{1}{\text{Recall}} + \frac{1}{\text{precision}}} \overset{e.g.}{=} \frac{2}{\frac{1}{1} + \frac{1}{.56}} = 72\%$$ (popular, acc. also popular)

More:

false discovery rate

$$FDR = 1 - \text{precision} = \frac{FP}{\#PP}$$

false omission rate

$$FOR = \frac{FP}{\#PN}$$

In my experience more important. They operate on the columns of the confusion table. This means FDR tells you if $\hat{y} = 1$ what's your error rate & FOR " " " " $\hat{y} = 1$, " " " " " "

$\boxed{\text{DEMO}}$

$\boxed{\text{In-sample and OOS}}$

Are there confusion tables for classification with $K > 2$ levels?

Yes!! $\boxed{\text{DEMO}}$

Credit Loan questions

1. How much will person pay back?
2. Will they pay back fully?
3. What is the chance they pay back?

Type of Supervised Learning Task

Regression

Classification

probabilistic classification

Probabilistic Classification : the response is still the label. Let's focus on binary $\mathcal{Y} = \{0, 1\}$

Previously...

$$y = \underbrace{t(z_1, \ldots, z_t)}_{\in \{0,1\}} = \underbrace{f(x_1, \ldots, x_p)}_{\in \{0,1\}} + \underbrace{\delta}_{\in \{+1,-1\}} = \underbrace{h^*(x_1, \ldots, x_p)}_{\in \{0,1\}} + \underbrace{\mathcal{E}}_{\in \{+1,-1\}} = \underbrace{g(x_1, \ldots, x_p)}_{\in \{0,1\}} + \underbrace{e}_{\in \{+1,-1\}}$$

f, h^*, g all have $\mathcal{Y} = \{0, 1\}$ as their output. They do <u>not</u> learn probs.

Consider an alternate construction. Let Y represent the r.v. whose realization is y. Instead of the above

prob that $\vec{x}$ will be 1

$$Y \sim \text{Bernoulli}\left(f_{pr}(x_1, \ldots, x_p)\right)$$

for a given $\vec{x}_i = [x_{i1}, \ldots x_{ip}]$, the above r.v. is realized to $y_i = 1$ or $y_i = 0$.

Same for

$$Y \sim \text{Bernoulli}\left(h^*_{pr}(x_1, \ldots, x_p)\right) \quad \& \quad Y \sim \text{Bernoulli}\left(g_{pr}(x_1, \ldots, x_p)\right)$$

Where are the error terms: $\delta, \mathcal{E}, e$? Not needed in this construction. The goal is to model the prob's. So errors now are differences with the true prob. function $t(z_1, \ldots z_t) = t_{pr}(z_1, \ldots z_t) \leftarrow$ why?

$$Y \sim \text{Bern}\big(\underbrace{t_{pr}(z_1,\ldots,z_t)}_{\text{always 0 or 1}}\big) \iff y = t(z_1,\ldots z_t)$$

No error means prob's are 0 or 1!

What is the difference between $t_{pr}(z_1,\ldots,z_t)$ & $f_{pr}(x_1,\ldots,x_p)$?

$f_{pr} \approx t$ (hopefully) but f_{pr} will return some value $\in (0,1)$ since the error due to ignorance will be explained as a non 0 or 1 prob!

e.g. $t(\vec{z}) = 1$ and $g_{pr}(\vec{x}) = 0.8$

these correspond

this means $g(\vec{x})$ will be correct most of the time. Only $\frac{1}{5}$ on avg, will $\hat{y} = 0$ when $y = 1$.

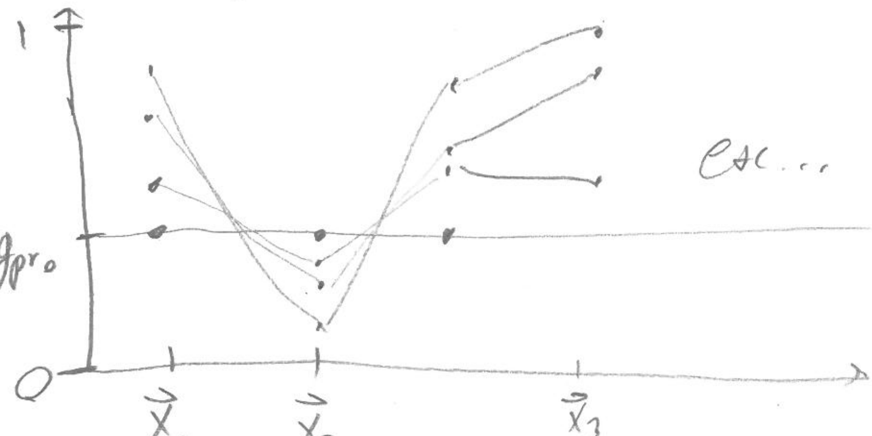
What is the difference between f_{pr}, h^*_{pr}, g_{pr}?

the values of these functions are further away from perfect 0's & 1's.

Null model

$$g_{pr,0} = \frac{1}{n}\sum y_i = \hat{p}$$

e.g let's say there are even # of 0's and 1's in $\mathbb{D}$ with $n = 200$.



f_{pr} is closer to 0's, 1's than h^*_{pr} which is closer to 0's, 1's than g_{pr}.

As your model gets worse and worse the prob. estimates move further from 0's & 1's to values closer to $g_{pr,0}$, the overall avg.

OK... how do we create g_{pr}. We need an algorithm $\mathcal{A}$.
First let's do some more prob. At the best we can know...

$$Y_1 \sim \text{Bern}(f_{pr}(\vec{x}_1)),\ Y_2 \sim \text{Bern}(f_{pr}(\vec{x}_2)), \ldots, Y_n \sim \text{Bern}(f_{pr}(\vec{x}_n))$$

$$\quad = \qquad\qquad\qquad\qquad = \qquad\qquad\qquad\qquad =$$

$$(f_{pr}(\vec{x}_1))^{y_1}(1-f_{pr}(\vec{x}_1))^{1-y_1} \qquad , , , , \qquad , , , , , ,$$

What is $P(Y_1, \ldots, Y_n)$ i.e. the "joint mass function"?

Unknown unless we know the <u>dependence</u> structure between $Y_1, \ldots, Y_n$.

Typically a <u>big assumption is made</u> ... <u>independence</u> which gives us:

1)  each indep. (X doesn't matter)

$$P(Y_1, \ldots, Y_n) = f_{pr}(\vec{x}_1)^{y_1}(1-f_{pr}(\vec{x}_1))^{1-y_1} \cdot \ldots \cdot f_{pr}(\vec{x}_n)^{y_n}(1-f_{pr}(\vec{x}_n))^{1-y_n}$$

$$= \prod_{i=1}^{n} f_{pr}(\vec{x}_i)^{y_i}(1-f_{pr}(\vec{x}_i))^{1-y_i}$$

Now... if we are trying to estimate f_{pr}, what are the "knowns"?
The $y_1, y_2, \ldots, y_n$. So our goal now is to come up with f_{pr} s.t. the prob. $P(Y_1, \ldots, Y_n)$ (AKA the "likelihood") is <u>MAXIMIZED</u>.

But of course, f_{pr} is arbitrarily complicated with interactions & nonlinearities. So let's make an assumption on candidate models. (a real-world)

$\mathcal{H}$ = { set of all candidate prob. functions } of which h^*_{pr} is the closest to f_{pr}. Then we use an alg. $\mathcal{A}$ to pick g_{pr} which is the best and hope it's close to h^*_{pr}.

What can we use for $\mathcal{H}$? How about $\mathcal{H} = \{\vec{w} \cdot \vec{x} : \vec{w} \in \mathbb{R}^{p+1}\}$?

Why doesn't this work? $\vec{w} \cdot \vec{x} \in \mathbb{R}$ and prob's $\in [0,1]$

What about $\mathcal{H} = \{\mathbb{1}_{\vec{w} \cdot \vec{x} \geq 0}\}$? Only 0 or 1... nothing in between!

What if we wanted to keep the linear model but wanted smooth output $\in (0,1)$?
↳ generalized linear model (GLM)

We need a "link function" $\phi(\vec{w} \cdot \vec{x})$ whose range is $(0,1)$ monotonically & smoothly.

(i.e. if $\vec{w} \cdot \vec{x} \uparrow \Rightarrow$ prob est. $\uparrow$) Also, $\phi(\vec{w} \cdot \vec{x}) \neq 0$ or 1 Why?

We can never be sure!! There is information we don't know!!

There are many possible ϕ functions! (These are also called activation functions in neural nets)

The most popular is the logistic function:

$$\phi(u) = \frac{e^u}{1+e^u} = \frac{1}{1+e^{-u}}$$

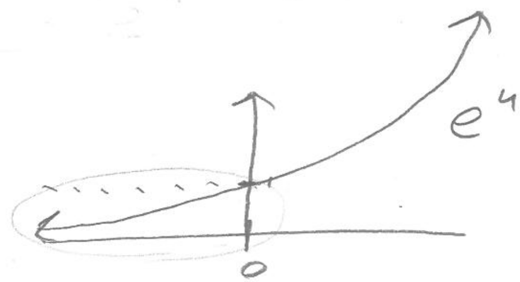
Also the probit function

$$\phi(u) = \Phi^{-1}(u)$$

inverse CDF of std. normal

or the "complementary log-log":

$$\phi(u) = 1 - e^{-e^u}$$



Why is this $\in (0,1)$?

$$e^u \in (0,\infty)$$

$$-e^u \in (-\infty, 0)$$

$$e^{-e^u} \in (0,1)$$

$$1 - e^{-e^u} \in (0,1)$$

or the hyperbolic tangent:

$$\phi(u) = \tanh(u) := \frac{e^u - e^{-u}}{e^u + e^{-u}}$$

So let's use logistic ϕ:

$$\mathcal{H} = \left\{ \frac{e^{\vec{w}\cdot\vec{x}}}{1+e^{\vec{w}\cdot\vec{x}}} : \vec{w} \in \mathbb{R}^{p+1} \right\}$$

What does $\mathcal{A}$ do? It maximizes the likelihood:

$$\vec{b} := \underset{\vec{w}}{\operatorname{argmax}} \left\{ P(Y_1, \ldots, Y_n) \right\} = \underset{\vec{w}}{\operatorname{argmax}} \left\{ \prod_{i=1}^{n} \left(\frac{e^{\vec{w}\cdot\vec{x}_i}}{1+e^{\vec{w}\cdot\vec{x}_i}} \right)^{y_i} \left(1 - \frac{e^{\vec{w}\cdot\vec{x}_i}}{1+e^{\vec{w}\cdot\vec{x}_i}} \right)^{1-y_i} \right\}$$

Since we are seeking the argmax of some function $\omega(\vec{w})$ we can easily just find the argmax of $\ln(\omega(\vec{w}))$ as well:

$$= \underset{\vec{w}}{\mathrm{argmax}} \left\{ \prod_{i=1}^{n} \left(\frac{1}{1+e^{-\vec{w}\cdot\vec{x}_i}} \right)^{y_i} \left(\frac{1}{1+e^{\vec{w}\cdot\vec{x}_i}} \right)^{1-y_i} \right\} \cancel{= \mathrm{argmax} \left\{ \prod_{i=1}^{n} \frac{1}{1+e^{\vec{w}\cdot\vec{x}_i}} \prod_{i=1}^{n} \left(\frac{1+e^{\vec{w}\cdot\vec{x}_i}}{1+e^{-\vec{w}\cdot\vec{x}_i}} \right)^{y_i} \right\}}$$

$$\cancel{\left(\frac{1}{1+e^{\vec{w}\cdot\vec{x}_i}}\right)\left(1+e^{\vec{w}\cdot\vec{x}_i}\right)^{y_i}}$$

$$= \begin{cases} \left(1+e^{-\vec{w}\cdot\vec{x}_i}\right)^{-1} & \text{if } y_i = 1 \\ \left(1+e^{\vec{w}\cdot\vec{x}_i}\right)^{-1} & \text{if } y_i = 0 \end{cases}$$

Consequence of using logistic link!

p91 LFD

$$= \left(1 + e^{(1-2y_i)\vec{w}\cdot\vec{x}}\right)^{-1}$$

let $z_i = 2y_i - 1 = -(1-2y_i)$

$y_i = 0 \Rightarrow z_i = -1$

$y_i = 1 \Rightarrow z_i = +1$

$$= \underset{\vec{w}}{\mathrm{argmax}} \left\{ \prod_{i=1}^{n} \left(1 + e^{-z_i \vec{w}\cdot\vec{x}_i}\right)^{-1} \right\}$$

Note: if we are taking $\mathrm{argmax}\{v(t)\} = \mathrm{argmax}\{\ln(v(t))\}$ since $\ln$ is a monotonic increasing transformation

$$= \underset{\vec{w}}{\mathrm{argmax}} \left\{ \ln\left(\quad \right) \right\} = \underset{\vec{w}}{\mathrm{argmax}} \left\{ -\sum_{i=1}^{n} \ln\left(1 + e^{-z_i \vec{w}\cdot\vec{x}_i}\right) \right\}$$

$$= \underset{\vec{w}}{\underline{\mathrm{argmin}}} \left\{ \sum_{i=1}^{n} \ln\left(1 + e^{-z_i \vec{w}\cdot\vec{x}_i}\right) \right.$$

Now we can take $\frac{d}{d\vec{w}}\left[\quad \right] \overset{set}{=} 0$ to solve for $\vec{b}$.