DMML, 23 January 2020

Classifier evaluation

- Test set carved out of training data

Evaluation metric?

Accuracy – percentage of correct answers

Problem – unbalanced categories

Often, the interesting case is a minority

- Fraud, Junk Mail, Rare disease

Suppose "Yes" occurs 5% of time

Blind "No" classifier is 95% accurate

Want to force classifier to flag "Yes"

Categorize errors more finely

Prediction

		Y	N
Actual answer	Y	✓	✗
	N	✗	✓

5% Yes, Trivial "No" classifier

	Y	N
Y	0	50
N	0	950

goal

will cause this

1000 cases

950 N   50 Y

	Y	N
Y	True Positive	FN
N	FP	True Negative

$$\frac{TP}{TP+FN}$$

actually found

should have found

$\boxed{\text{RECALL}}$

$\boxed{\text{PRECISION}}$

$$\frac{TP}{TP+FP}$$

Precision-recall tradeoff

Screening test vs interview

Coronavirus vs pancreatic cancer

Single number?

F-score : Harmonic mean

Reciprocal of mean of reciprocals

$$\frac{1}{\left(\frac{\frac{1}{P}+\frac{1}{R}}{2}\right)} \qquad \frac{2PR}{P+R}$$

Regression — predicting a numeric value

Fit a function to the data

Attributes $A_1, \ldots, A_k$

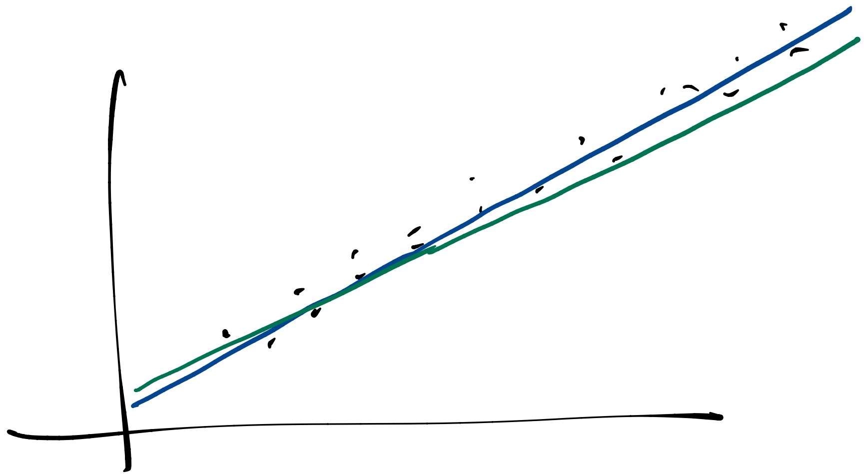
$$f(x_1, \ldots, x_k)$$

Simplest case: $f(x_1, \ldots x_k) = a_1 x_1 + a_2 x_2 + \ldots a_k x_k + b$

$$f(x) = mx + b$$

How to find "best" m, b

Define an error measure



Training data

(x_1, y_1)

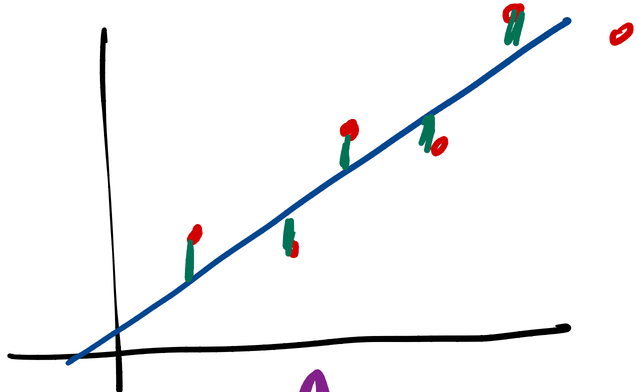
(x_2, y_2)

$\vdots$

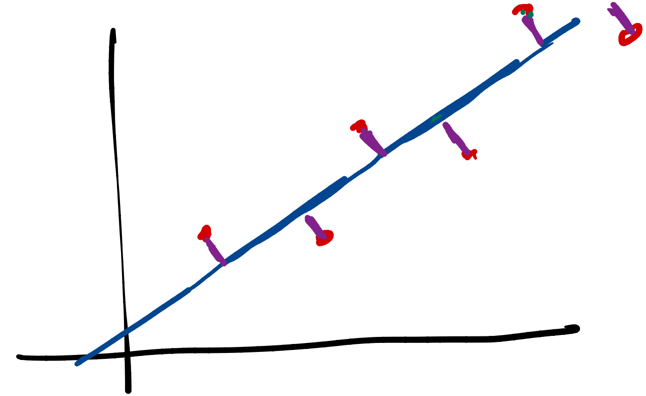
(x_n, y_n)

Prediction : $mx_i + b$

$$|(mx_i + b) - y_i|^2$$ – square error



vs



square error

Not this

$$\text{Mean Square Error} = \frac{1}{n}\sum_{i=1}^{n}\left((mx_i+b)-y_i\right)^2$$

Find m, b to minimize MSE

Statistics – direct formula for m, b based on mean, variance etc of training points

Instead – iteratively improve m & b

Adjust m & b so that MSE reduces

$\boxed{\text{MSE}}$ – Error, Loss, Cost $\Theta(m, b)$

$$\Theta(m,b) = \frac{1}{n}\sum_{i=1}^{n}\left((mx_i+b)-y_i\right)^2$$

Want

$\frac{\partial \theta}{\partial m}$, $\frac{\partial \theta}{\partial b}$

Remember that x_i's are fixed values, & y_i

$$\frac{1}{2n} \sum_{i=1}^{n} (\ \_\_ \ )^2$$

$$\frac{\partial \theta}{\partial m} \quad 2(mx_i + b - y_i) \cdot x_i$$

$$\frac{\partial \theta}{\partial b} \quad 2(mx_i + b - y_i) \cdot 1$$

Adjust m by $\alpha \cdot -\frac{\partial\theta}{\partial m}$

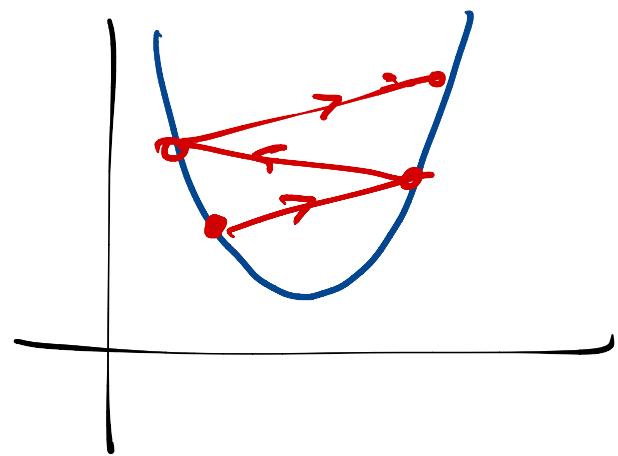
b by $\alpha \cdot -\frac{\partial\theta}{\partial b}$

α is a small value

"learning rate"

If α is too small - progress is slow

If α is too big?



Gradient descent — batch of "predictions"
update coefficient

Can also do smaller batches & update incrementally

Stochastic Gradient Descent (SGD)

Pick a random subset to update
Recompute gradient
Repeat

Suppose the function is not linear?

Classically – transform the data

Suppose $f(x_1) = a_1 x_1 + a_2 x_1^2 + a_3$

$a_1 x_1 + a_2 x_2 + a_3$

$\downarrow$

$x_2 = x_1^2$

$x_1, y_1 \longrightarrow x_1, x_1^2, y_1$

$x_2, y_2 \qquad x_2, x_2^2, y_2$

$\vdots \qquad \vdots$

$x_n, y_n \qquad x_n, x_n^2, y_n$

Python - sklearn

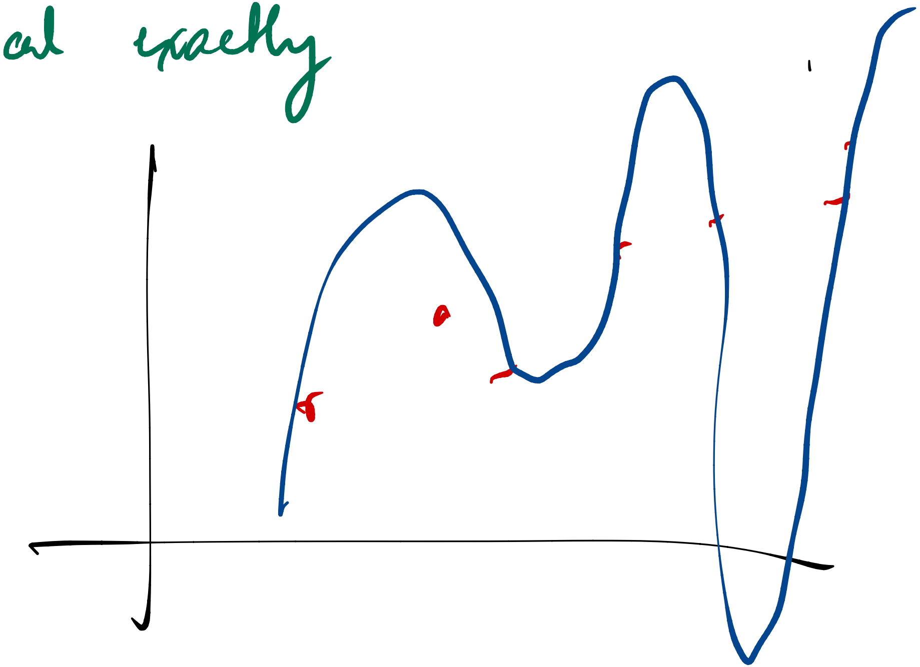
Specify degree of regression

Input is (x_1, x_2, y)

Degree is 2

$$(x_1, x_2, x_1^2, x_2^2, x_1 x_2, y)$$

We can always fit an arbitrarily high degree polynomial exactly



Overfitting