May 21, 2021

e.g. Cancer classifier w/ 1% error on test set.

However, only 0.5% of patients have cancer.

→ You'd get a lower error just by predicting $y=0$ for everyone!

This is the problem of skewed classes.

↳ We need a different evaluation metric.

Precision and Recall:

		Actual Class: 1	Actual Class: 0
Predicted Class	1	True Positive	False Positive
	0	False Negative	True Negative

$$\text{Precision} = \frac{\text{True positives}}{\#\text{predicted positive}} = \frac{\text{True positives}}{\text{True positives} + \text{False positives}}$$

↳ i.e. Of all the patients we predicted $y=1$, what fraction actually have cancer?

$$\text{Recall} = \frac{\text{True positives}}{\#\text{actual positives}} = \frac{\text{True positives}}{\text{True positives} + \text{False negatives}}$$

↳ i.e. Of all the patients that actually have cancer, how many did we detect?

Trading off Precision and Recall:

Logistic regression: $0 \leq h_\theta(x) \leq 1$

Predict 1 if $h_\theta(x) \geq$ ~~0.5~~ 0.7 } higher confidence required

Predict 0 if $h_\theta(x) <$ ~~0.5~~ 0.7 }

- This results in higher precision and lower recall due to less false positives and more false negatives.
- We can also lower the threshold if we want to avoid missing too many cases.

F_1 Score (F Score):

How do we compare precision/recall numbers?

$$F_1 \text{ Score} = 2\frac{PR}{P+R}$$

	Precision (P)	Recall (R)	F_1 Score
Algo 1	0.5	0.4	0.444
Algo 2	0.7	0.1	0.175
Algo 3	0.02	1.0	0.0392

→ Avg of P+R wouldn't work cuz same problem as before

How to Train on Large Datasets:

1. Use a learning algo w/ many parameters
   (i.e. linear/logistic regression w/ many features
   or a neural network w/ many hidden units).
   ↳ results in low bias; small $J_{train}(\theta)$
2. Use a very large training set so algo doesn't overfit
   ↳ results in low variance; $J_{test}(\theta) \approx J_{train}(\theta)$

Finally, $J_{test}(\theta)$ will be very small, which is what we want.