STRUCTURING ML PROJECTS

COURSE III

ORTHOGONALIZATION

* what to tune in order to achieve what effect?

CHAIN OF ASSUMPTIONS IN ML	IF NOT TRUE, TRY...
#1. Train set fits well on cost function	Bigger network or Adam
#2. Dev set fits well on cost function	Regularization, bigger train set
#3. Test set fits well on cost function	Bigger dev set
#4. Performs well in real world	Change dev set or cost function

* one could also try early stopping, but it affects both → network (assumption #1)
  → regularization (assumption #2)

SINGLE NUMBER EVALUATION METRIC

* your progress will be much faster if you have a single real number evaluation metric that lets you quickly tell if the new thing you just tried is working better/worse than your last idea.

$$\text{F1 SCORE} = \frac{2}{1/P + 1/R}$$

Harmonic mean → it combines both Precision + Recall (F1 Score)

* if you are computing errors, then you can take the average of the errors.

SATISFICING AND OPTIMIZING METRICS

* example:

CLASSIFIER	ACCURACY	RUNNING TIME
A	90%	80 ms
B	92%	95 ms
C	95%	1,500 ms

OPTIMIZING METRIC — SATISFICING METRIC

* you want to maximize the accuracy subject to running time.

* if you have (N) metrics, then you choose ONE optimizing and N-1 satisficing

SETS DISTRIBUTIONS

* dev and test set should come from the same distribution
  should reflect data you expect to get in the future

→ for 1,000,000 data points

* test set → set it to be big enough to give high confidence in the overall performance of the system.
  → for some systems you might not even need a test set. (UNUSUAL)

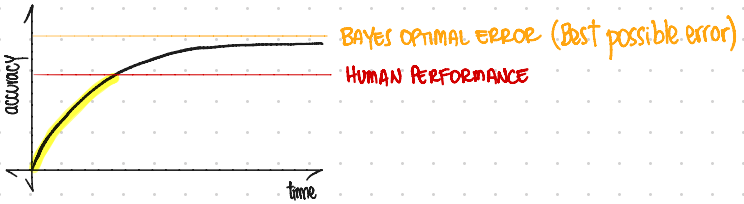
* let's say that you have two algorithms that were designed to classify cat images
  → algorithm A: 3% error + it misclassifies porn images
  → algorithm B: 5% error + it does not classify porn images
  ↓
  redifine the loss to penalize for the porn images

Error: $\frac{1}{m_{dev}} \sum_{i=1}^{m_{dev}} w^{(i)} \mathcal{L}\{y_{pred}^{(i)} \neq y^{(i)}\}$

$\sum w^{(i)}$

$$w^{(i)} = \begin{cases} 1 \text{ if } x^{(i)} \text{ is not porn} \\ 10 \text{ if } x^{(i)} \text{ is porn} \end{cases}$$

WHY HUMAN-LEVEL PERFORMANCE?



* progress is fast until you surpass human-level performance

↓

not actually very far from Bayes error

* so long as accuracy < human-level performance, you can:

(#1). get labeled data from humans

(#2). gain insight from manual error analysis (why did a person get this right?)

(#3). better analysis bias/variance

AVOIDABLE BIAS

* you don't want to do TOO WELL on the training set.

		PROXY FOR BAYES ERROR	
Human error	1%	7.5%	AVOIDABLE BIAS (0.5%)
Training error	8%	8%	AVOIDABLE VARIANCE (2%)
Dev error	10%	10%	
	↓	↓	
	FOCUS ON BIAS	FOCUS ON VARIANCE	

* ML significantly surpasses human-level performance:
  1. online advertising
  2. product recommendation
  3. logistics (predicting transit time)
  4. loan approvals

  } Structured data + not natural perception tasks (lots of data)

IMPROVING YOUR MODEL PERFORMANCE

FUNDAMENTAL ASSUMPTIONS IN SUPERVISED LEARNING	IF NOT TRUE, TRY...
1. Train set fits well (low avoidable bias)	Train bigger/longer model, momentum, Adam, RMS prop, try CNN or RNN
2. Train set performance generalizes pretty well to the dev/test set (low variance)	More data, regularization, dropout, data augmentation.

ERROR ANALYSIS

* should you try to make your classifier do better? (Let's say we achieved 90% accuracy + 10% error)
  1. get ~100 mislabeled dev set examples
  2. count how many are dogs (Let's say 5% ~ 5/100)

  If that's the case, then you might be able to decrease the error to 9.5% → NOT WORTH IT!

  But if 50% ~ 50/100 are dog pictures, then the error would go down to 5%. → WORTH IT!

* you can also evaluate multiple ideas in parallel:
  1. fix pictures of dogs recognized as cats
  2. fix great cats (lions, panthers, etc.)
  3. improve performance on blurry images.

  } calculate percentage of mislabels corresponding to each category and focus on that problem