WEEK #6
Advice for Applying M.L.

1. EVALUATING A LEARNING ALGORITHM.

1.1. Deciding What to Try Next.

→ If you are developing ML. stms, that you know how to choose one of the most promising avenues to spend your time pursuing.

⇒ **Debugging a learning algorithm.**

$$J(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda\sum_{j=1}^{m}\theta_j^2\right]$$

→ Get more training examples

→ Try smaller sets of features (to prevent overfitting)

→ Try getting additional features (may be current features aren't informative enough & you want to collect more data in the sense of getting more features).

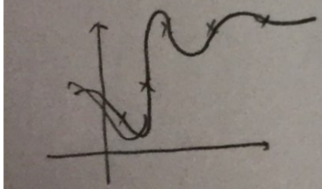
→ Try adding polynomial features (x_1^2, x_2^2, x_1x_2, etc.)

→ Try decreasing λ

→ Try increasing λ

⇒ **M.L. diagnostic:** test that you can run to gain insight what is/isn't working with a learning algorithm, and gain guidance as to how best to improve its performance.

1.2. Evaluating a Hypothesis:



Fails to generalize to new examples not in training set

How to be sure about overfitting?

Divide the training set into 2 groups: → training set (70%)
↘ test set (30%)

→ Learn paramtr θ from training data

→ Compute test error:

1.3. Model Selectn and Train / Validatn / Test Sets:

⇒ Model Selection:

d=degree of polynomial.

$d=1 \longrightarrow$ Training error $(\theta^{(1)}) \rightarrow$ Test error $(\theta^{(1)})$

⋮

$d=5 \longrightarrow$ Training error $(\theta^{(5)}) \longrightarrow$ Test error $(\theta^{(5)})$

$J_{test}(\theta^{(5)})$ ⟶

⋮

→ $J_{test}(\theta^{(5)})$ is like to be an optimistic estimate of generalizatn error, i.e. our extra paramtr (d) is fit to test set.

→ Divide the dataset into 3 groups: training set (60%) → $J_{train}(\theta)$
Cross-Validatn set (20%) → $J_{cv}(\theta)$
test set (20%) → $J_{test}(\theta)$

Use CVset to select the d-paramtr (rather than using test set).

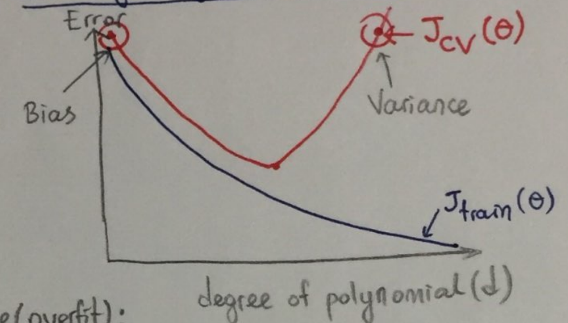
Pick the optimum hypothesis (ex/ $\theta_0+\theta_1 x+\theta_2 x^2+\theta_3 x^3+\theta_4 x^4$)

Estimate generalizatn error for test set $J_{test}(\theta^{(4)})$

2. BIAS Vs. VARIANCE

2.1. Diagnosing bias Vs. variance:

bias → underfitting (d=1)
variance → overfitting (d=10)
just right (d=5)



Bias (underfit):

$J_{train}(\theta)$ → high
$J_{cv}(\theta)$ → high

Variance (overfit):

$J_{train}(\theta)$ → low
$J_{cv}(\theta) \gg J_{train}(\theta)$

2.2. Regularizatn and Bias/Variance:

$$J(\theta)=\frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)})-y^{(i)})^2+\frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$$

Large λ:
High bias, underfit
$\lambda=10K$, $\theta_1\approx0$, $\theta_2\approx0$
$h_\theta(x)\approx0$

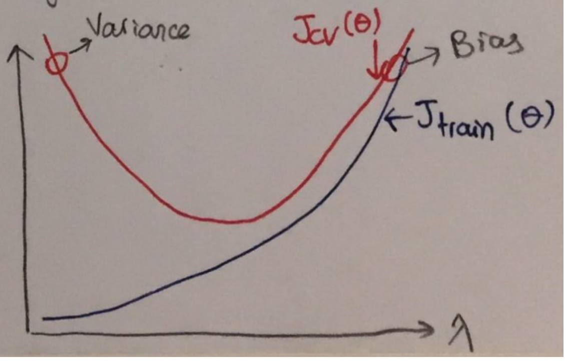
Intermediate λ:
Just Right

Small λ:
High Variance, Overfit
$\lambda=0$.

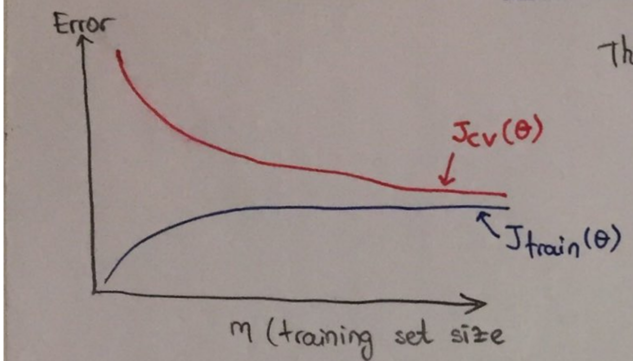
⇒ Choosing the regularizatn error:

1) $\lambda=0 \longrightarrow \min_\theta J(\theta) \longrightarrow \theta^{(1)} \rightarrow J_{cv}(\theta^{(1)})$
2) $\lambda=0.01$
3) $\lambda=0.02$
4) $\lambda=0.04 \longrightarrow \min_\theta J(\theta) \longrightarrow \theta^{(4)} \longrightarrow J_{cv}(\theta^{(4)})$
⋮
12) $\lambda=10 \longrightarrow \theta^{(12)} \longrightarrow J_{cv}(\theta^{(12)})$

Pick (say) $\theta^{(5)}$ → Test Error: $J_{test}(\theta^{(5)})$

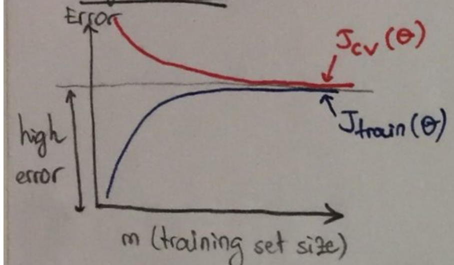


2.3. Learning Curves:



The more data you have ⇒ the better the hypothesis you fit.

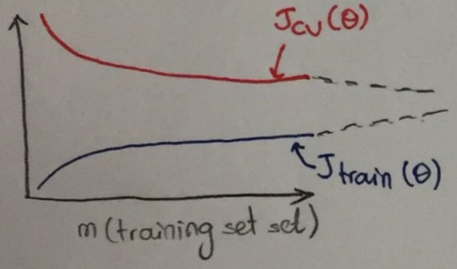
High Bias:



J_{cv} & J_{train} will be very similar

If a learning algorithm is suffering from high bias, getting more training data will not (by itself) help much.

High Variance:



→ Large gap btw training error & cv-erron

If a learning algorithm is suffering from high variance, getting more training is likely to help.

2.4. Deciding What to do Next (Revisited)

→ Get more training examples → fixes high variance

→ Try smaller sets of features → fixes high variance

→ Try getting additnl features → fixes high bias

→ Try adding polynomial features → fixes high bias

→ Try decreasing λ → fixes high bias

→ Try increasing λ → fixes high variance

Small NN	Large NN
• computatnly cheaper	• more expensive
• prone to underfitting	• prone to overfitting
	• Use regulztn ↗
	• Select # of hidden layers.

3. REVIEW: QUIZ + P.A.

4. BUILDING A SPAM CLASSIFIER.

4.1. Prioritizing What to Work On:

1) x=features of email , y= 1 (spam) / 0 (non-spam)

features x: choose 100 words indicative of spam/non-spam.

$$x = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ \vdots \end{bmatrix}$$

2) collect lots of data

3) develop sophisticated features based on email routing info.

4) " " " for message body ("discounts"...)

5) " " algorithm to detect mispellings (med1cine)

4.2. Error Analysis:

1) start with simple algorithm

2) plot learning curves

3) error analysis: manually examine the examples

ex/ 100 errors on cathegorizatn of spam classifier:

i) what type of mail it is

ii) what cues (features) you think would have helped the algorithm to classify.

4.3. Error metrics for skewed classes:

Find that you got 1% error on test set.

→ Only 0.50% of patients have cancer.
↳ skewed classes.

```
function y=predictCancer(x)
    y=0;
end
```

→ 0.5% error ? gives

recall=0

Precision /Recall. (y=1 in presence of rare class)

y=1 in presence of rare class that we want to detect.

Predicted Class		Actual class 1	Actual class 0
	1	True ~~posititle~~	False positive
	0	False negative	True negative

Precision: $\frac{\text{true positive}}{\#\text{ predicted positive}} = \frac{\text{True positive}}{\text{True pos} + \text{False pos}}$

↑ good

Recall: $\frac{\text{True pos}}{\#\text{actual pos}} = \frac{\text{True pos}}{\text{True pos} + \text{False neg.}}$

↑ good

5. HANDLING SKEWED DATA.

5.2. Trading off Precision and Recall.

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

Suppose we want to predict $y=1$ (cancer) only if very confident.

Predict 1 if $h_\theta(x) \geq 0.7$ → Higher precision.

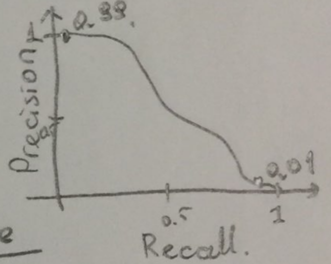
Predict 0 if $h_\theta(x) < 0.7$ → Lower recall.

Suppose we want to avoid missing too many cases of cancer (avoid false negatives).

1 if $h_\theta(x) \geq 0.3$ → Higher recall

0 if $h_\theta(x) < 0.3$ → Lower precision.

More generally: Predict 1 if $h_\theta(x) \geq$ threshold



F$_1$ score (F score)

	Precision (P)	Recall (R)	~~Average~~	F$_1$ score
Algor.1	0.5	0.4	~~0.45~~	0.444
Algor.2	0.7	0.1	~~0.4~~	0.175
Algor.3	0.02	1.0	~~0.51~~	0.0392

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

6. USING LARGE DATA SETS.

6.1. Data for M.L.

Designing a high accuracy learning stm:

Algorithms:

- Perceptron (Logistic Regression)
- Winnow
- Memory-based
- Naive Bayes.

Large data rationale: Assume feature $x \in \mathbb{R}^{n+1}$ has sufficient infn to predict y.

⇒ Use a learn. alg. with many paramtrs (hidden units)

→ low bias algorithms. $J_{train}(\theta)$ will be small.

⇒ Use a very large training set

→ low variance

$J_{train}(\theta) \approx J_{test}(\theta)$ will be small