Machine Learning 2: Optimisation and good practices

How do we improve a ML algorithm?

- More training examples.

How do we improve a ML algorithm?

- More training examples.
- Less features that fits training set equally well.

How do we improve a ML algorithm?

- More training examples.
- Less features that fits training set equally well.
- More features to fit the training set better.

How do we improve a ML algorithm?

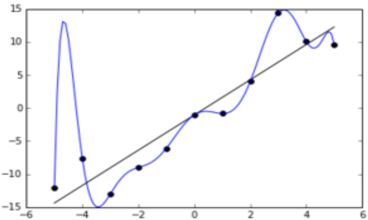
- More training examples.
- Less features that fits training set equally well.
- More features to fit the training set better.
- Choose features more carefully.

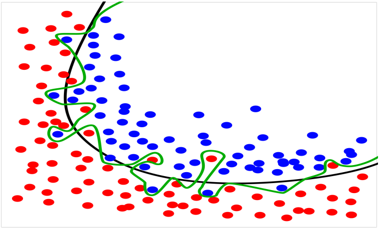
How do we improve a ML algorithm?

- More training examples.
- Less features that fits training set equally well.
- More features to fit the training set better.
- Choose features more carefully.
- Regularisation

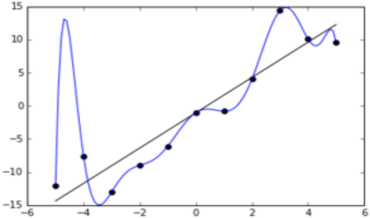
Regularisation

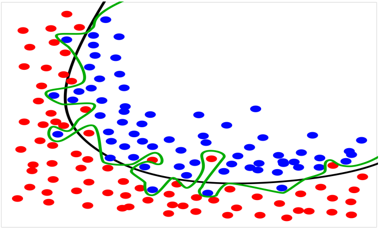
Regularisation





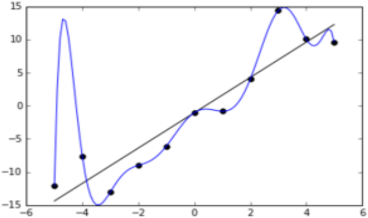
Regularisation

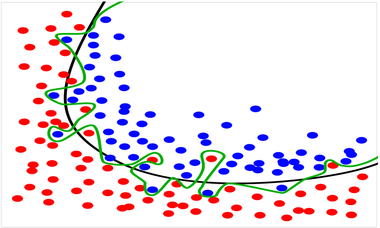




- Too many features lets $h(\theta^T x)$ fit the data very well, but would fail to generalise — **overfitting/ high variance**.

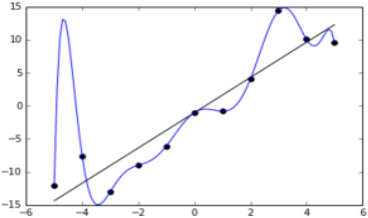
Regularisation

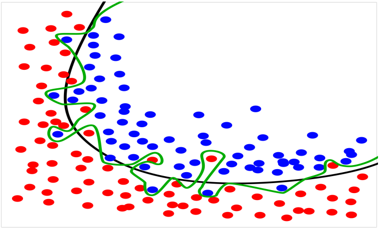




- Too many features lets $h(\theta^T x)$ fit the data very well, but would fail to generalise — **overfitting/ high variance**.
- Too few features and $h(\theta^T x)$ would fail to fit the data well and have a large error — **underfitting/ high bias**.

Regularisation



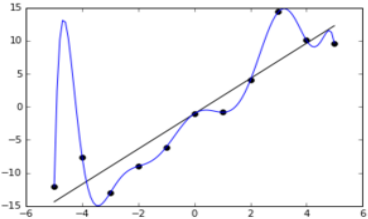


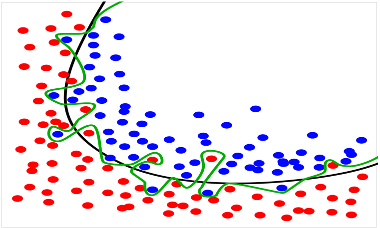
- Too many features lets $h(\theta^T x)$ fit the data very well, but would fail to generalise — **overfitting/ high variance**.
- Too few features and $h(\theta^T x)$ would fail to fit the data well and have a large error — **underfitting/ high bias**.

Options to cure overfitting/ high variance:

1. Reduce number of features .... manually ??, use some model selection algorithm ??

Regularisation





- Too many features lets $h(\theta^T x)$ fit the data very well, but would fail to generalise — **overfitting/ high variance**.
- Too few features and $h(\theta^T x)$ would fail to fit the data well and have a large error — **underfitting/ high bias**.

Options to cure overfitting/ high variance:

1. Reduce number of features .... manually ??, use some model selection algorithm ??
2. Incorporate a method that 'weights' features in order of their importance — **regularisation**.

Regularisation can just be added as a term in the cost function ....

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left[y^i \log[h(\theta^{\mathsf{T}}\mathrm{x}^{\mathrm{i}})] + (1-y^i)\log[1-h(\theta^{\mathsf{T}}\mathrm{x}^{\mathrm{i}})]\right] + \lambda\theta^T\theta.$$

Regularisation can just be added as a term in the cost function ....

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left[y^i \log[h(\theta^{\mathsf{T}}\mathrm{x}^{\mathrm{i}})] + (1-y^i)\log[1-h(\theta^{\mathsf{T}}\mathrm{x}^{\mathrm{i}})]\right] + \lambda\theta^T\theta. \quad (1)$$

Regularisation parameter λ needs to be chosen carefully

λ too small and regularisation becomes useless.
λ too large and we get underfitting.

Regularisation can just be added as a term in the cost function ....

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left[y^i \log[h(\theta^\mathsf{T}\mathrm{x}^\mathrm{i})] + (1-y^i)\log[1-h(\theta^\mathsf{T}\mathrm{x}^\mathrm{i})]\right] + \lambda\theta^T\theta. \quad (1)$$

Regularisation parameter λ needs to be chosen carefully

λ too small and regularisation becomes useless.
λ too large and we get underfitting.

How do we choose value in practice?

Cross-validation and test sets

[1] done w/o regularisation!

Cross-validation and test sets

How do we test a given ML algorithm or choice of λ ?

[1] done w/o regularisation!

Cross-validation and test sets

How do we test a given ML algorithm or choice of λ ?

1. Split data roughly into three sets:
   - Training set $\approx 60\%$
   - Cross-validation set $\approx 20\%$
   - Test set $\approx 20\%$

[1] done w/o regularisation!

Cross-validation and test sets

How do we test a given ML algorithm or choice of λ ?

1. Split data roughly into three sets:
   - Training set $\approx 60\%$
   - Cross-validation set $\approx 20\%$
   - Test set $\approx 20\%$
2. Calculate hypothesis fits (θ) using training set.

[1] done w/o regularisation!

Cross-validation and test sets

How do we test a given ML algorithm or choice of λ ?

1. Split data roughly into three sets:
   - Training set $\approx 60\%$
   - Cross-validation set $\approx 20\%$
   - Test set $\approx 20\%$
2. Calculate hypothesis fits (θ) using training set.
3. Calculate error of each algorithm using the CV set. Choose best algorithm (λ value or model). [1]

[1] done w/o regularisation!

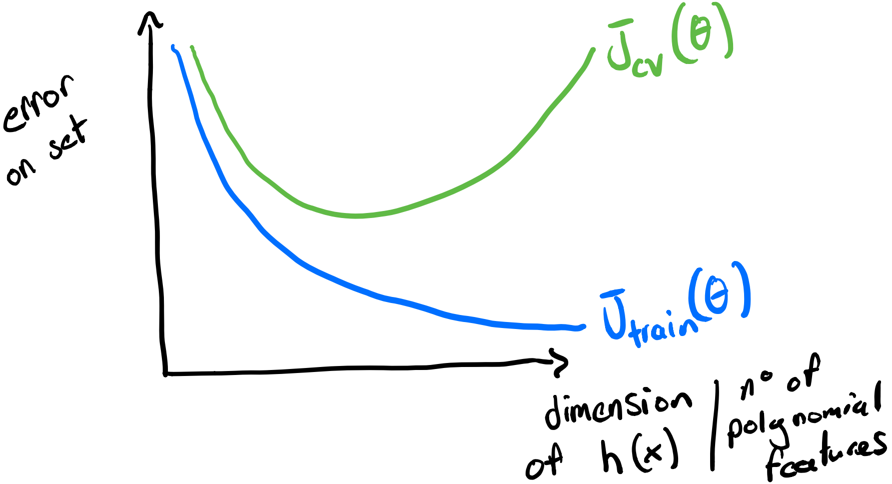
Cross-validation and test sets

How do we test a given ML algorithm or choice of λ ?

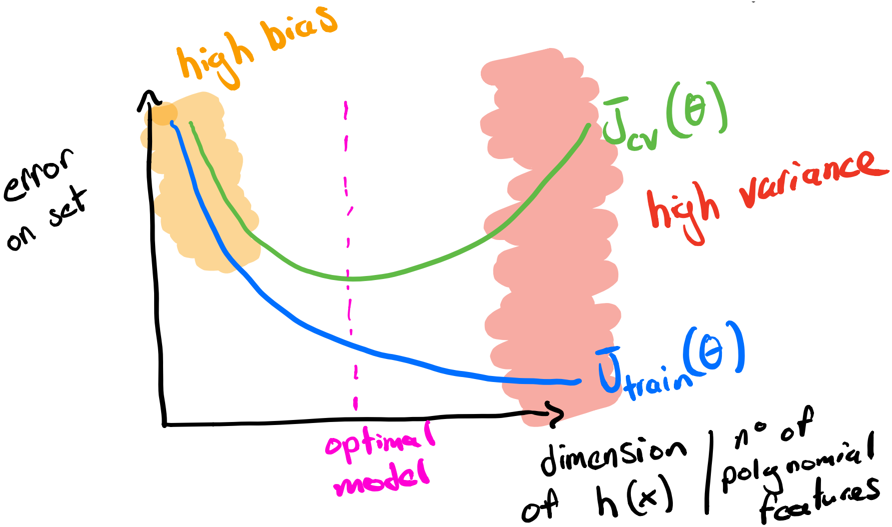
1. Split data roughly into three sets:
   - Training set $\approx 60\%$
   - Cross-validation set $\approx 20\%$
   - Test set $\approx 20\%$
2. Calculate hypothesis fits (θ) using training set.
3. Calculate error of each algorithm using the CV set. Choose best algorithm (λ value or model). [1]
4. Calculate the error on the test set to confirm choice.

[1] done w/o regularisation!

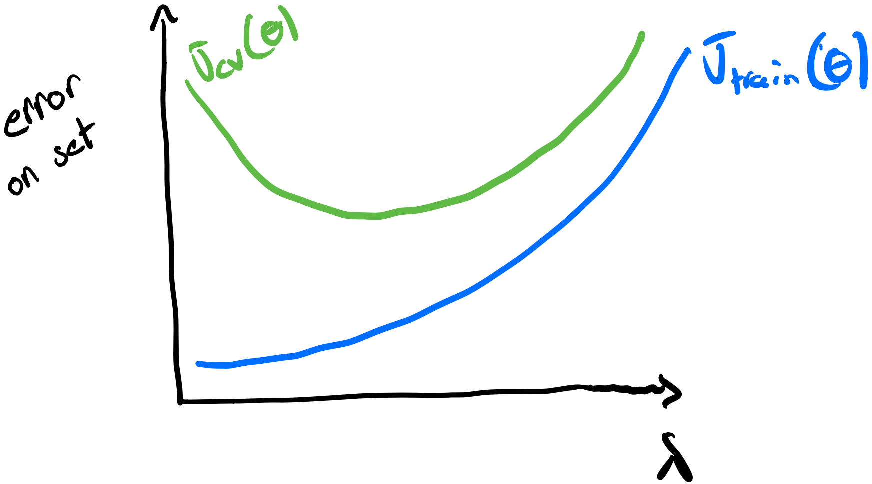
Diagnosing bias and variance - dimension



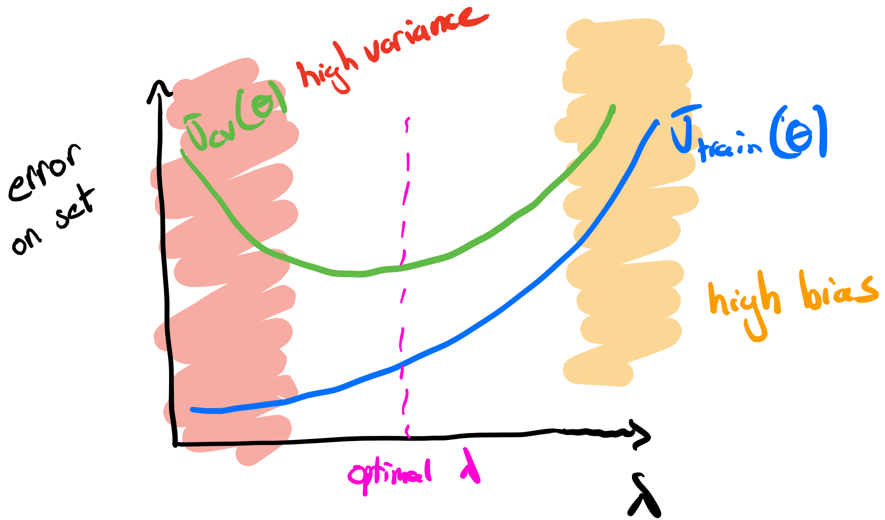
Diagnosing bias and variance - dimension



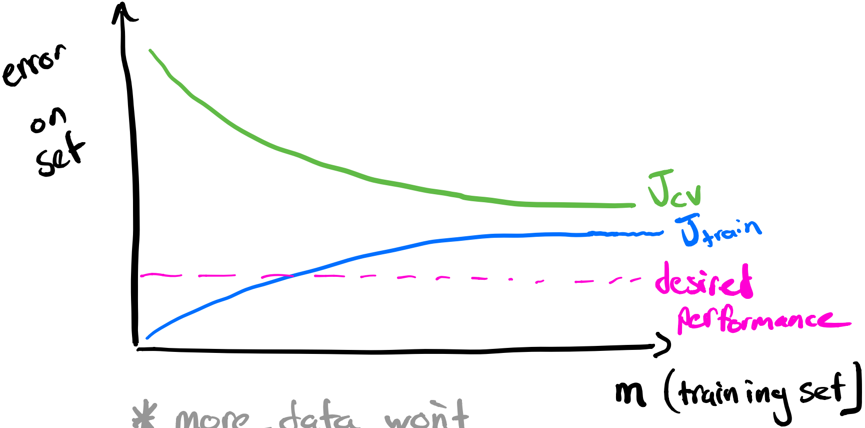
Diagnosing bias and variance - λ



Diagnosing bias and variance - λ

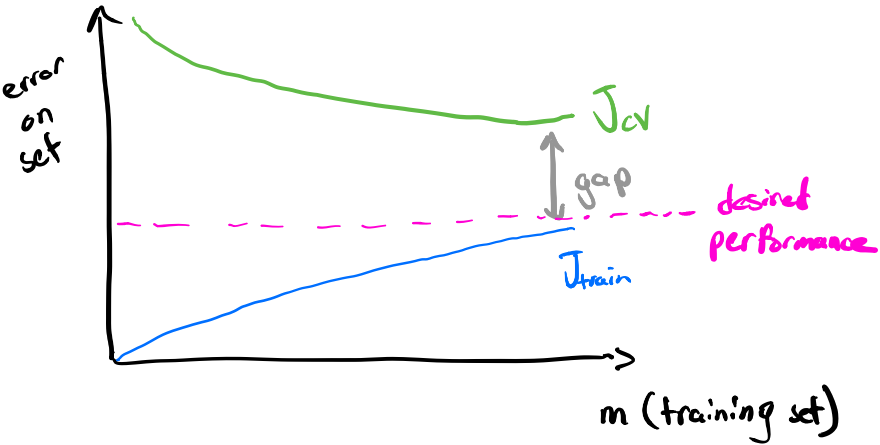


Diagnosing bias and variance - training set size

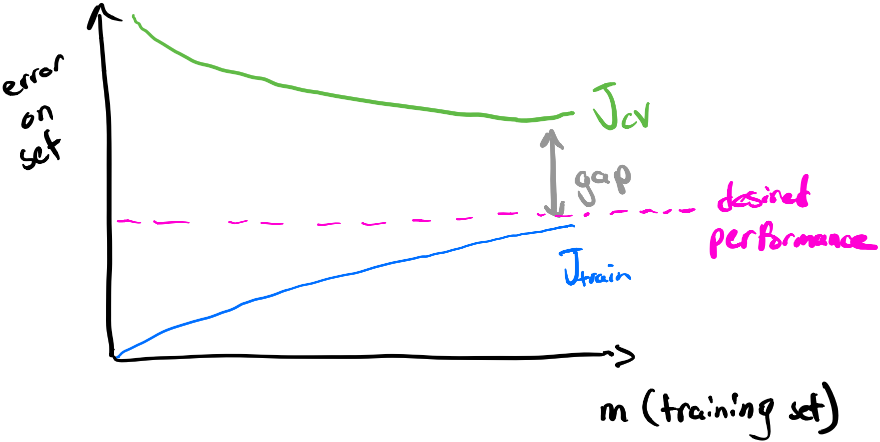


* more data won't generally help → asymptotic.

Diagnosing bias and variance - training set size



Diagnosing bias and variance - training set size



See python example 2

How do we improve a ML algorithm?

- More training examples. -fixes high variance

How do we improve a ML algorithm?

- More training examples. -fixes high variance
- Less features that fits training set equally well. -fixes high variance

How do we improve a ML algorithm?

- More training examples. -fixes high variance
- Less features that fits training set equally well. -fixes high variance
- More features to fit the training set better. -fixes high bias

How do we improve a ML algorithm?

- More training examples. -fixes high variance
- Less features that fits training set equally well. -fixes high variance
- More features to fit the training set better. -fixes high bias
- Regularisation: increase λ -fixes high variance

How do we improve a ML algorithm?

- More training examples. -fixes high variance
- Less features that fits training set equally well. -fixes high variance
- More features to fit the training set better. -fixes high bias
- Regularisation: increase λ -fixes high variance
- Regularisation: decrease λ -fixes high bias

How do we improve a ML algorithm?

- More training examples. -fixes high variance
- Less features that fits training set equally well. -fixes high variance
- More features to fit the training set better. -fixes high bias
- Regularisation: increase λ -fixes high variance
- Regularisation: decrease λ -fixes high bias
- **Choose features more carefully!**

How to approach an ML problem?

1. Take a look at your data and pre-process them
2. Quick and dirty algorithm to get some predictions.
3. Plot learning curves to inspire accuracy improvement.
4. Error analysis — **requires a well defined error-metric!**

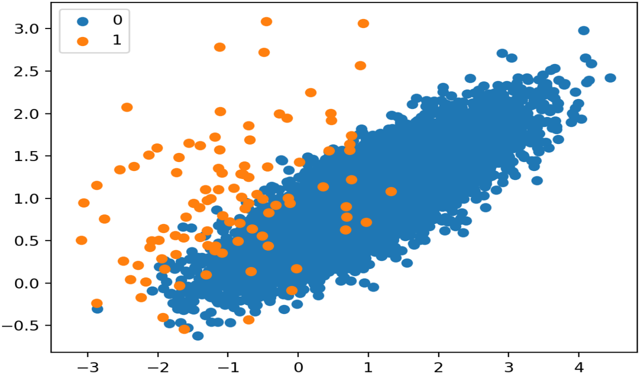
One can use the obvious metric :

$$Error = \frac{1}{m_{set}} \sum_{i=1}^{m_{set}} err(h(\theta^T x^i), y^i), \tag{2}$$

where

$$err(h(\theta^T x^i), y^i) = \begin{cases} 1 & h(\theta^T x^i)_{y=0} \ \geq 0.5 \quad OR \quad h(\theta^T x^i)_{y=1} \ < 0.5 \\ 0 & \text{otherwise} \end{cases},$$

Skewed data



Setting $h(\theta^T x) = 0$ always will give a very low error....

Precision and Recall

Actual

Predicted	1	0
1	True positive	False positive
0	False negative	True negative

$$Precision = \frac{\text{True Pos}}{\text{True Pos} + \text{False Pos}}, \qquad Recall = \frac{\text{True Pos}}{\text{True Pos} + \text{False Neg}} \tag{3}$$

Note: Set rarest class to be $y = 1$!

Precision and Recall

Some notes:

- We can change the threshold, $h(\theta^T x) \geq threshold$, such that we gain or lose precision or recall.

Precision and Recall

Some notes:

- We can change the threshold, $h(\theta^T x) \geq \mathit{threshold}$, such that we gain or lose precision or recall.
- The higher the threshold the higher the precision but the lower the recall.
- One can combine the two into the **F-score** which we try to maximise

$$0 \leq F_1 = \frac{2PR}{P+R} \leq 1 \tag{4}$$