Overfitting, regularization, and model validation

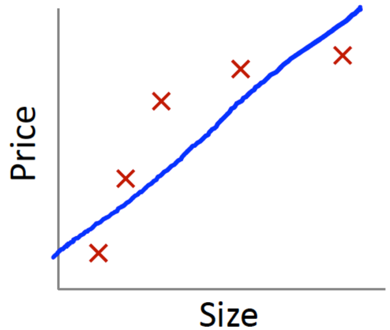
Ngô Minh Nhựt

2021

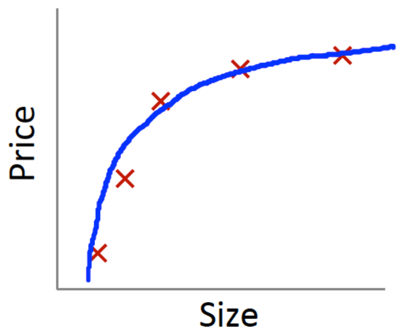
Content

- Overfitting
- Regularization
- Model validation

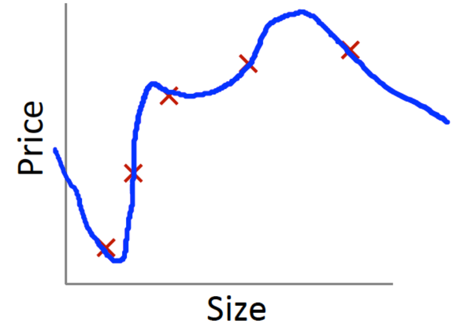
Overfitting



$\theta_0 + \theta_1 x$



$\theta_0 + \theta_1 x + \theta_2 x^2$



$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

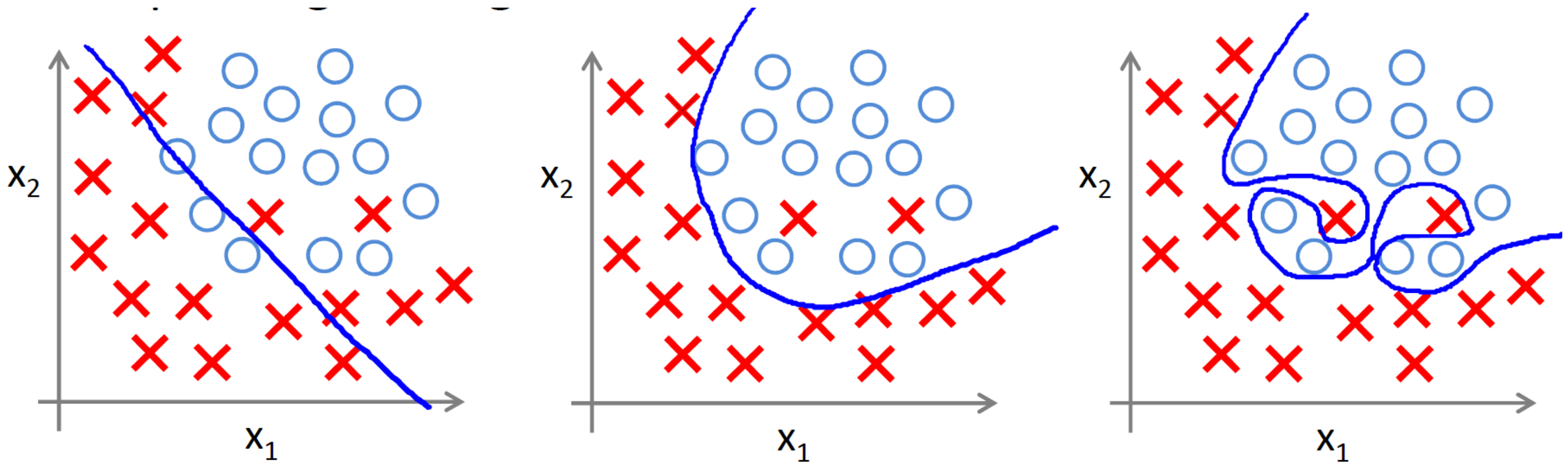
When there are many features:

- Model may overfit with training data (training cost ~ 0)
- Could not generalize for new samples (testing cost >> 0)

Source: Andrew Ng

Overfitting

- Logistic regression

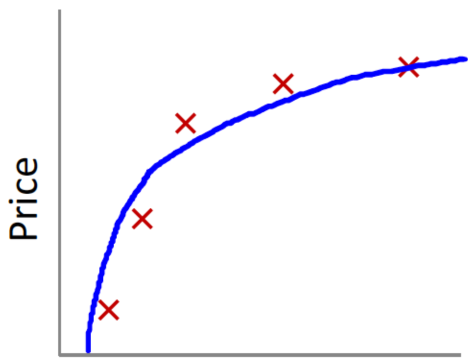


Source: Andrew Ng

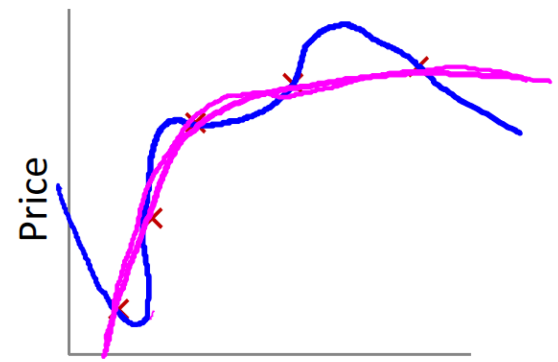
Overfitting

- Solutions:
  - Reduce number of features
    - Select features through data analysis
    - Select features through model validation
  - Regularization
    - Reduce effect of parameters to output
    - Good in case features contributes slightly to output

Regularization



$\theta_0 + \theta_1 x + \theta_2 x^2$



$\underline{\theta_0 + \theta_1 x + \theta_2 x^2} + \cancel{\theta_3 x^3} + \cancel{\theta_4 x^4}$

$$\min_{\theta} \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2 + 1000\theta_3^2 + 1000\theta_4^2$$

Source: Andrew Ng

Regularization

- When values of parameters become small
  - Produce simpler models
  - Reduce overfitting

Regularization

- Cost function

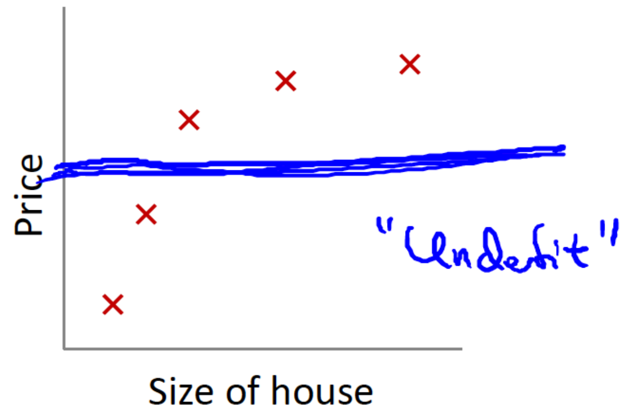
$$J(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2 + \lambda\sum_{j=1}^{n}\theta_j^2\right]$$

- Lambda: weight decay
- When lambda is bigger, model becomes simpler

Regularization

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

If lambda too big?



$h_\theta(x)$

$$\theta_0 + \cancel{\theta_1 x} + \cancel{\theta_2 x^2} + \cancel{\theta_3 x^3} + \cancel{\theta_4 x^4}$$

Source: Andrew Ng

Regularization

- Vector gradient

$$\frac{dJ}{d\theta_0} = \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)x_0^{(i)}$$

$$\frac{dJ}{d\theta_j} = \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)x_j^{(i)} + \frac{\lambda}{m}\theta_j$$

Gradient descent algorithm works as usual

Regularization

- Logistic regression – cost function

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

$$+ \frac{\lambda}{2m}\sum_{j=1}^{n} \theta_j^2$$

Regularization

- Logistic regression – vector gradient

$$\frac{dJ}{d\theta_0} = \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)x_0^{(i)}$$

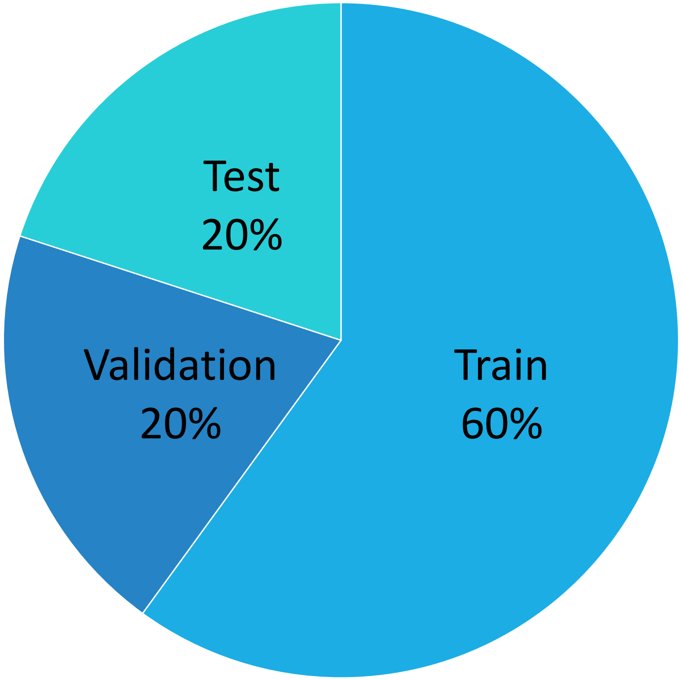
$$\frac{dJ}{d\theta_j} = \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)x_j^{(i)} + \frac{\lambda}{m}\theta_j$$

Model validation

- Learning process makes model fit with training data
- Can learnt model generalize for new samples?
  - Validate the model with unseen data
- Solution:
  - Learn, validate, and select the best model
  - Test if the selected model works well with new data

Dataset

- Split dataset into 3 subsets: train, validation, and test
- With large dataset, the ratio is: 60% : 20% : 20%



Cost function

Train error

$$J_{\text{train}}(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

Cross-validation error

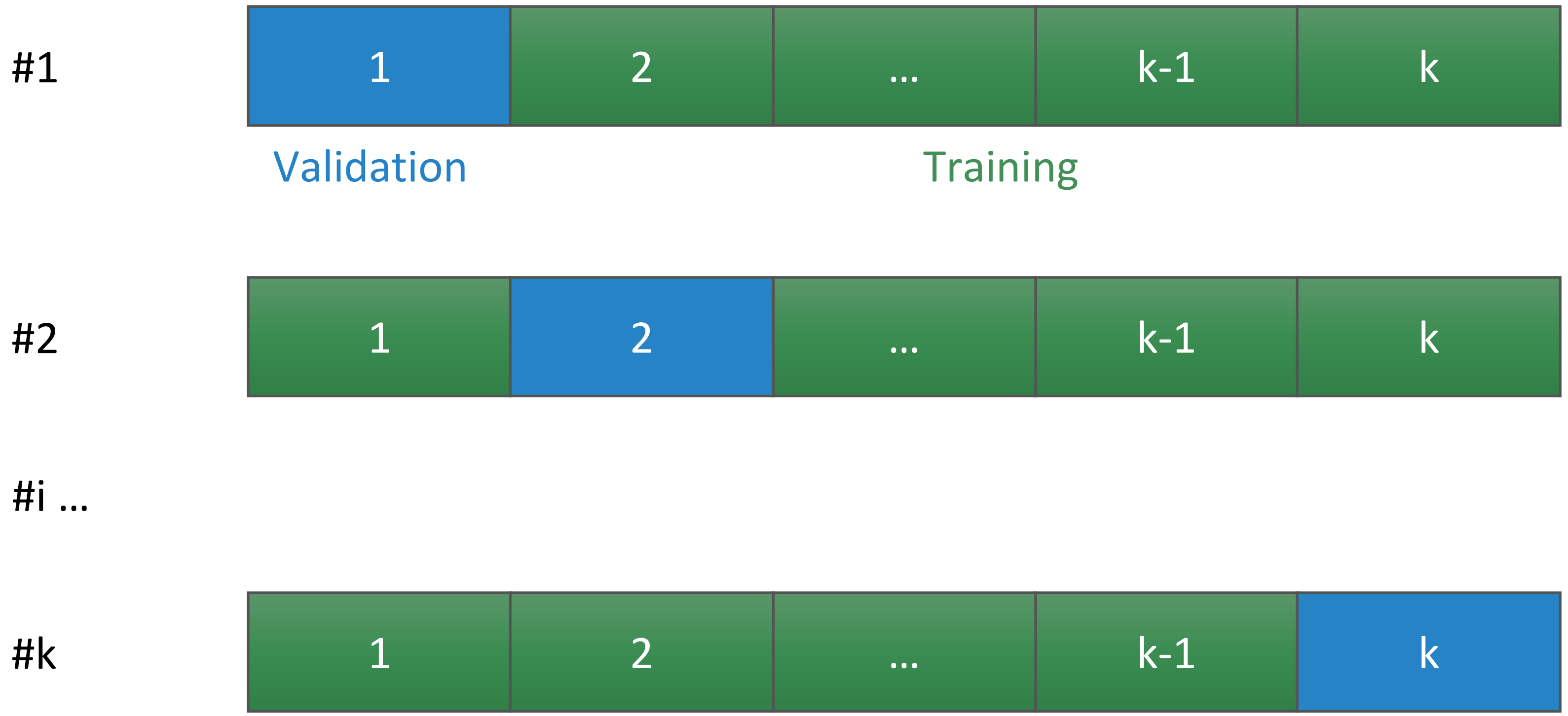
$$J_{\text{cv}}(\theta) = \frac{1}{2m_{\text{cv}}}\sum_{i=1}^{\text{cv}}\left(h_\theta(x_{\text{cv}}^{(i)}) - y_{\text{cv}}^{(i)}\right)^2$$

Test error

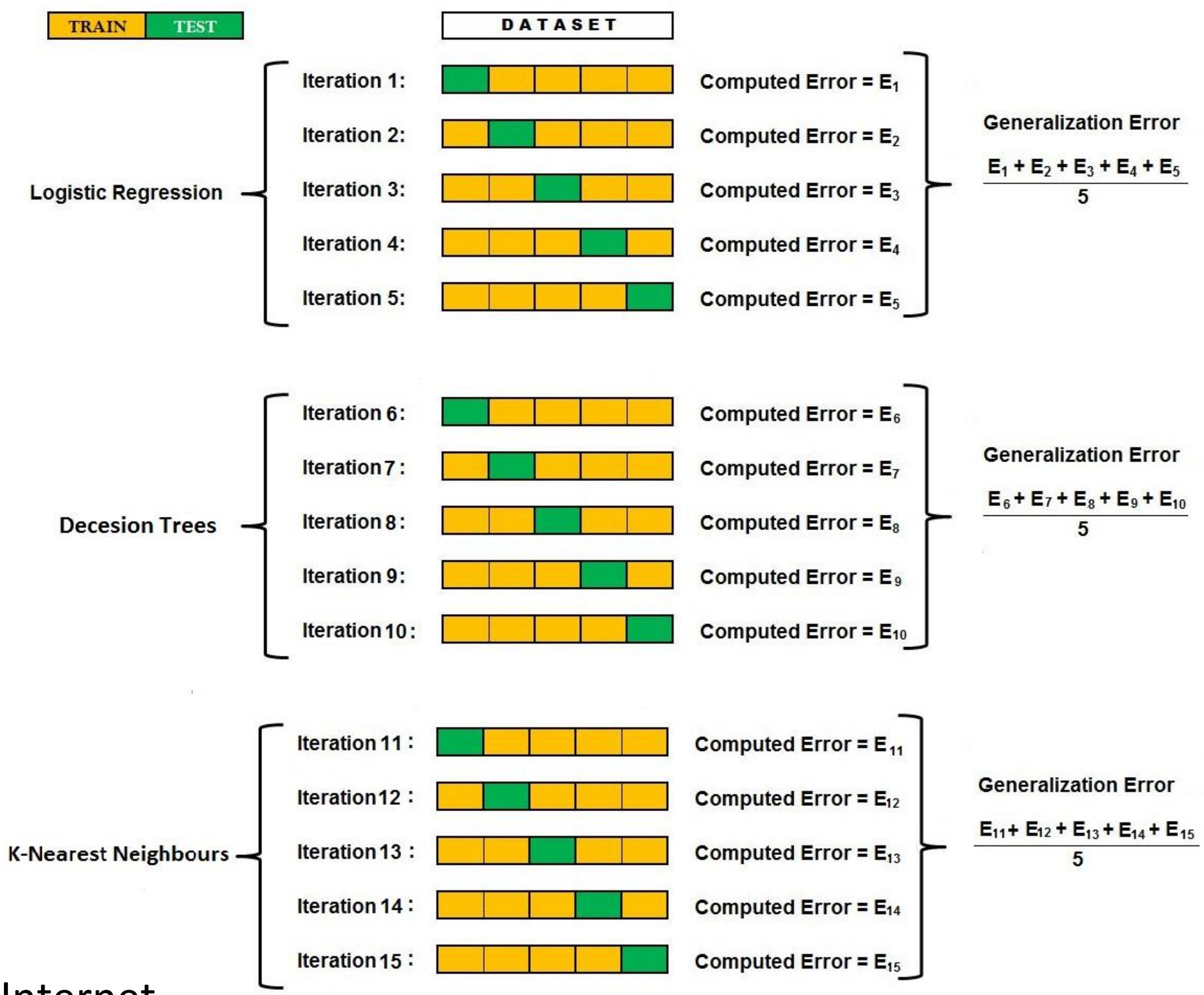
$$J_{\text{test}}(\theta) = \frac{1}{2m_{\text{test}}}\sum_{i=1}^{\text{test}}\left(h_\theta(x_{\text{test}}^{(i)}) - y_{\text{test}}^{(i)}\right)^2$$

k-fold cross validation

- Randomly split dataset into k parts
- Learn and validate k time. k is usually 10



k-fold cross validation

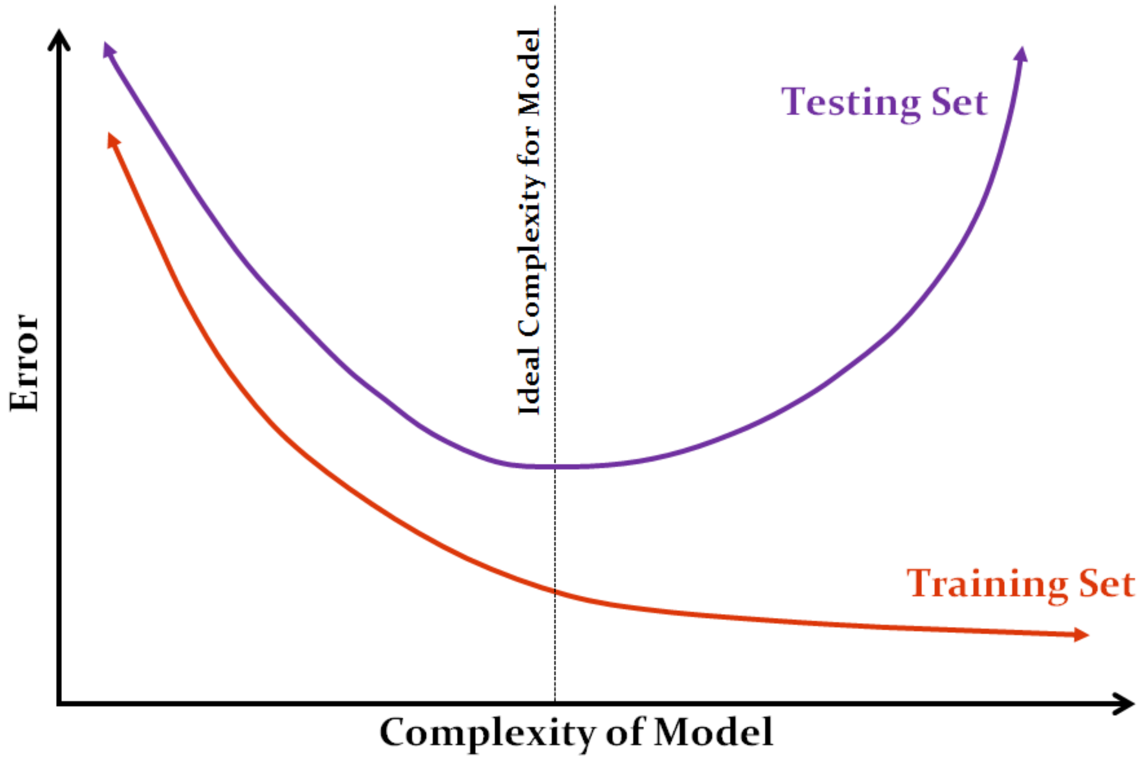


Source: Internet

Model finetuning

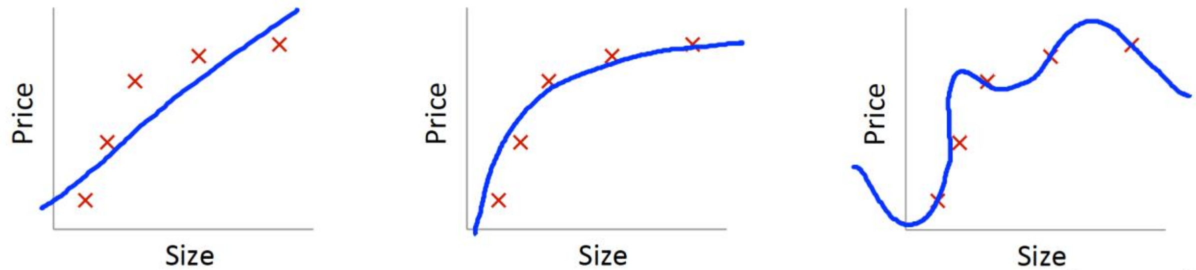
- Model has various parameters
- Learning algorithm has various hyper parameters
- What should we do when model does not work well?
  - Collect more data
  - Reduce number of features
  - Try with new features
  - Switch to other models or hypotheses
  - Reduce weight decay
  - Increase weight decay

Model finetuning



Bias and variance

- Bias: model error on training set
- Variance: model error on validation set
- Adjust bias and variance until they reach minimal



$\theta_0 + \theta_1 x$

High bias (underfit)

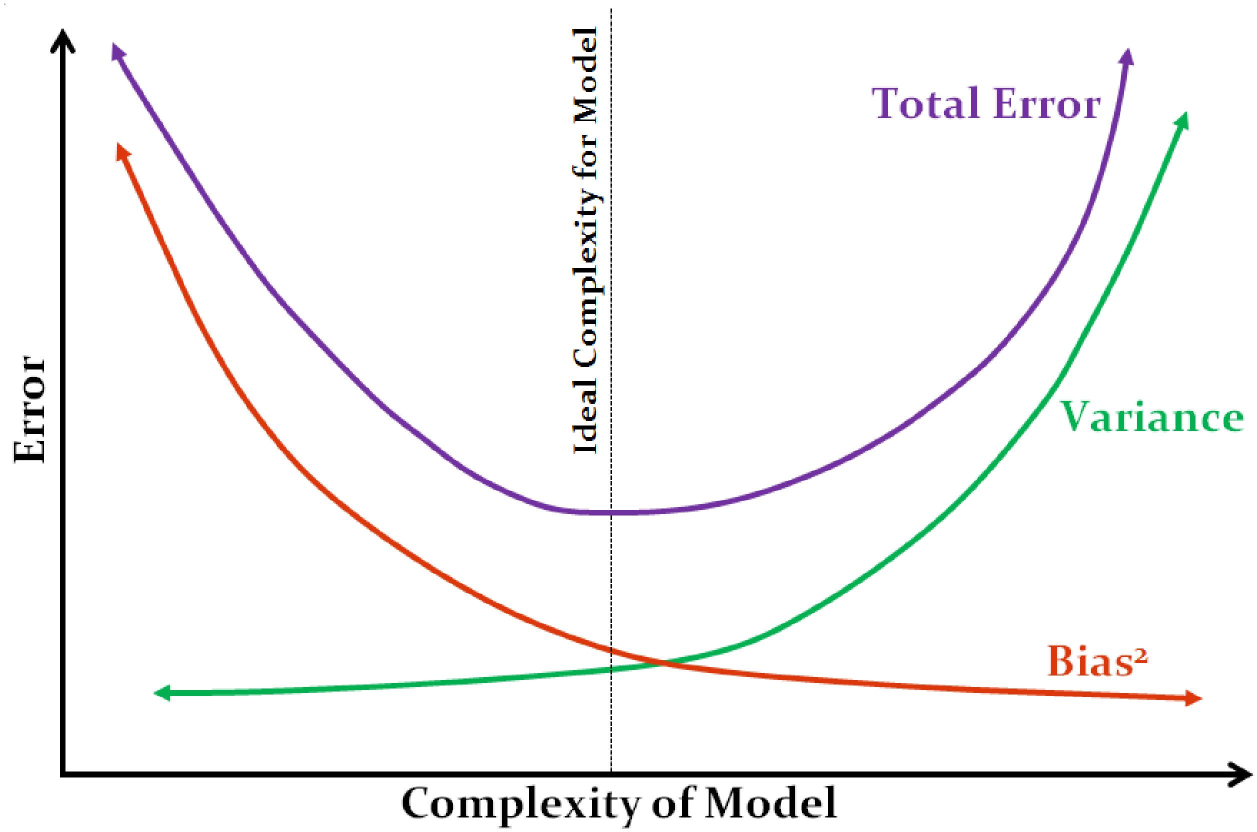
$\theta_0 + \theta_1 x + \theta_2 x^2$

"Just right"

$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

High variance (overfit)

Bias and variance



Model evaluation

- Measurements
  - Precision, recall
  - Accuracy
  - F1-score
- Depending on problems, we need some suitable measures

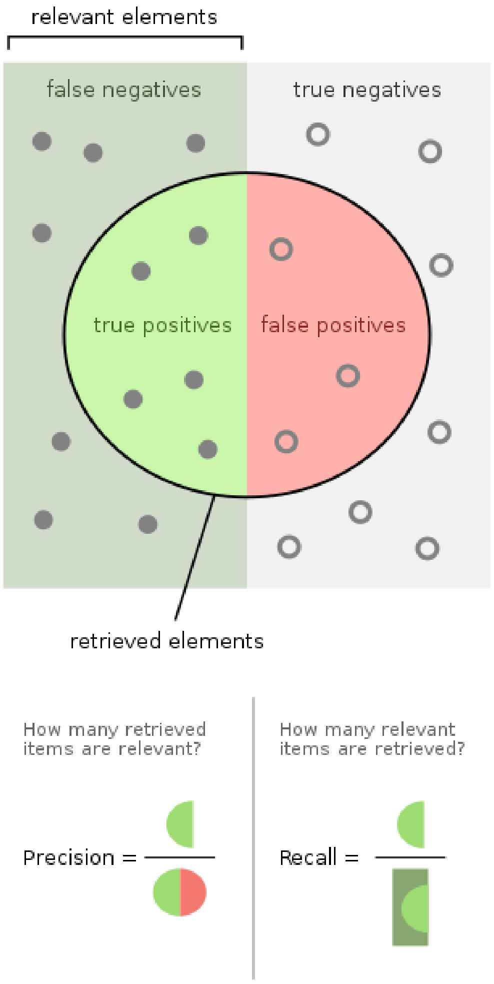
Model evaluation

- $\text{Precision} = \dfrac{\text{True positive}}{\text{Predicted positive}}$

- $\text{Recall} = \dfrac{\text{True positive}}{\text{Positive}}$

- $\text{F1} - \text{score} = \dfrac{2 \text{ x Precision x Recall}}{\text{Precision} + \text{Recall}}$

- $\text{Accuracy} = \dfrac{\text{True positive} + \text{True negative}}{\text{Positive} + \text{Negative}}$



Source: Wikipedia

Confusion matrix

		Predicted				Total
		Cat	Dog	Tiger	Wolf	
Actual	Cat	**6**	0	3	1	10
	Dog	2	**4**	0	4	10
	Tiger	3	3	**3**	0	9
	Wolf	1	4	1	**2**	8
Total		12	11	7	7	

>>> sklearn.metrics.classification_report

```
>>> y_true =  [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, …]
>>> y_pred = [0, 0, 0, 0, 0, 0, 2, 2, 2, 3, …]
>>> target_names = ['Cat', 'Dog', 'Tiger', 'Wolf']
```

```
              precision    recall  f1-score   support

         Cat      0.500     0.600     0.545        10
         Dog      0.364     0.400     0.381        10
       Tiger      0.429     0.333     0.375         9
        Wolf      0.286     0.250     0.267         8

    accuracy                          0.405        37
   macro avg      0.394     0.396     0.392        37
weighted avg      0.399     0.405     0.399        37
```