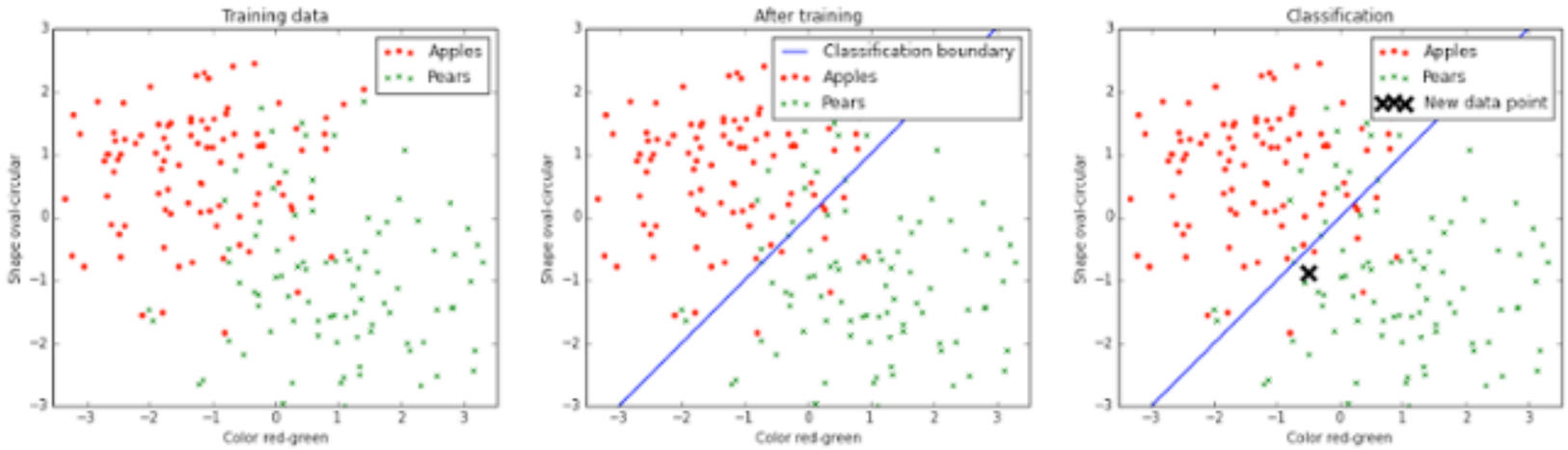
Machine Learning

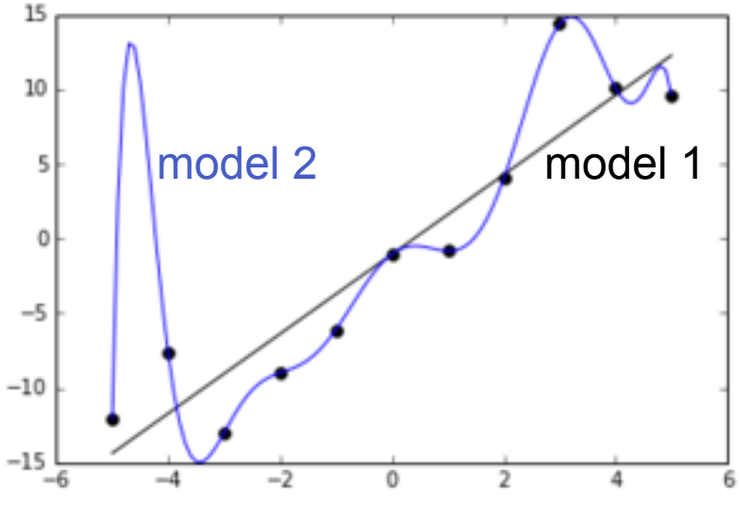
Overfitting, Model Selection & Validation

Classification Approaches



- Want to find boundaries in feature space that separate different classes of labelled examples
  - look for simple surface (best line or plane) that separates classes
  - look for more complex surfaces (e.g. several lines, curves)
  - use voting schemes (k-nearest neighbours, majority vote)
- Issues
  - How do we avoid overfitting?
  - How do we measure performance?
  - How do we select best features?

Trying Too Hard



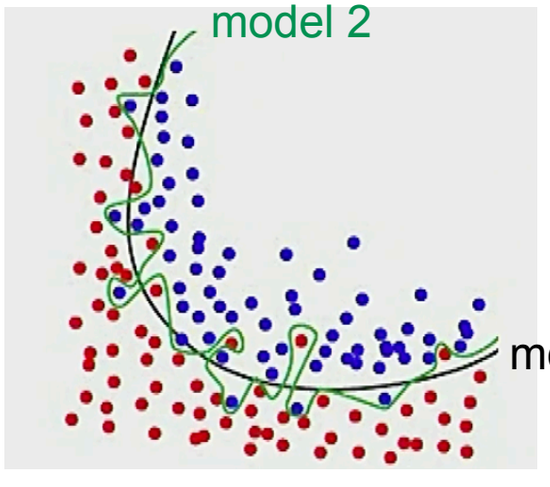
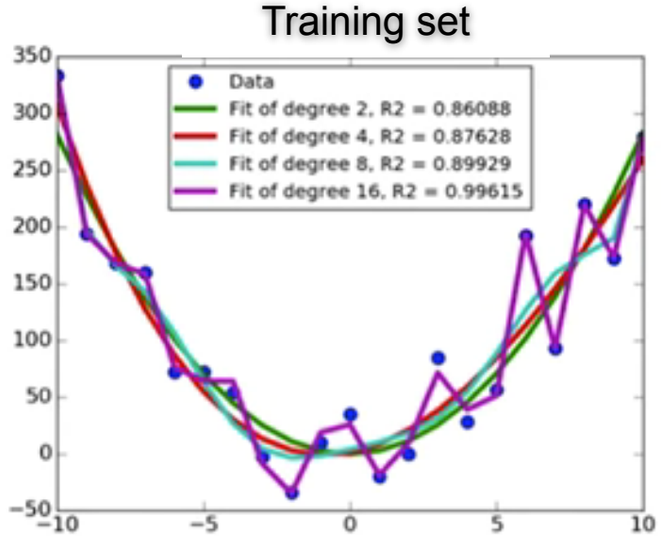


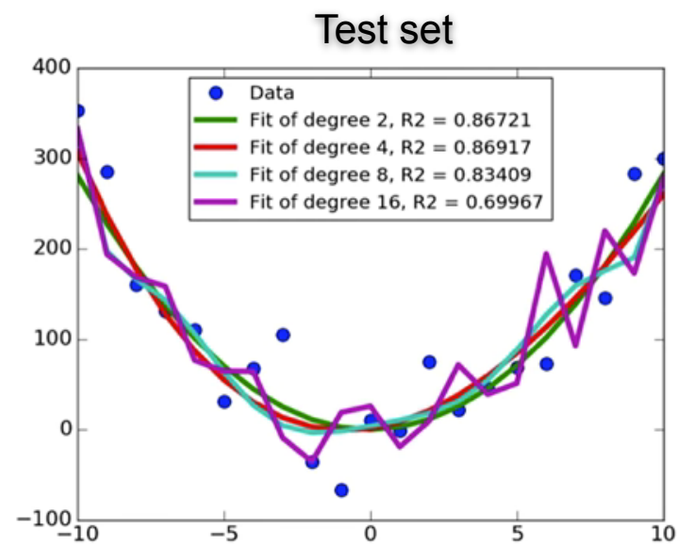
Figure: Two different models built to fit training data for regression (left) and classification (right). Model 1 is simpler than model 2

- Apart from fitting straight lines, one could also fit curves and other more complex models (involving $x^2, x^3,$ etc.)
- The aim of machine learning is to **generalise** well, i.e. to minimise the error of the model on new unseen data e.g. a voice recognition system should be able to work on different voice tones and accents
- For regression (left) and classification (right) the jagged boundaries are too fine-grained or specialised on the training data
- This would indicate that there is a high probability that it will not work well on new unseen data

Overfitting

- Overfitting is the phenomenon where a model is too complex that it also fits the noise (errors in dataset)
- A model that tries hard to fit all the training dataset might overfit the training set
- Training performance will be high but test set performance will be low – it performs poorly on unseen data
- Model should be simple but still captures the data well
- How do we go about finding the right model?





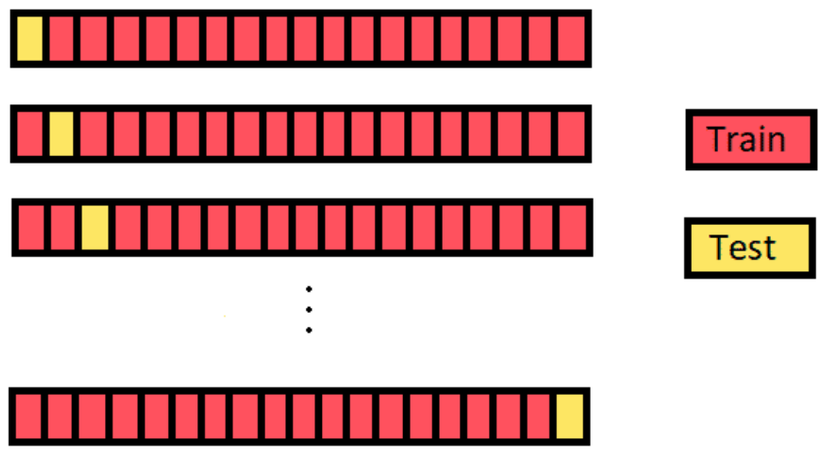
How Do We Overcome Overfitting?

- Model should have good pre-defined values, e.g. number of times to loop during training, good k for k-nearest neighbour (hyper-parameters)
- Every time we train a model on training set, check performance (e.g. accuracy) on test set
- If test performance is poor, go back and tune the hyper-parameters (e.g. add more training iterations, use a different k)
- There are several ways to overcome overfitting, so that the model generalises better to new unseen data
- Something called cross validation, works well when the dataset is limited

Leave-one-out Cross Validation

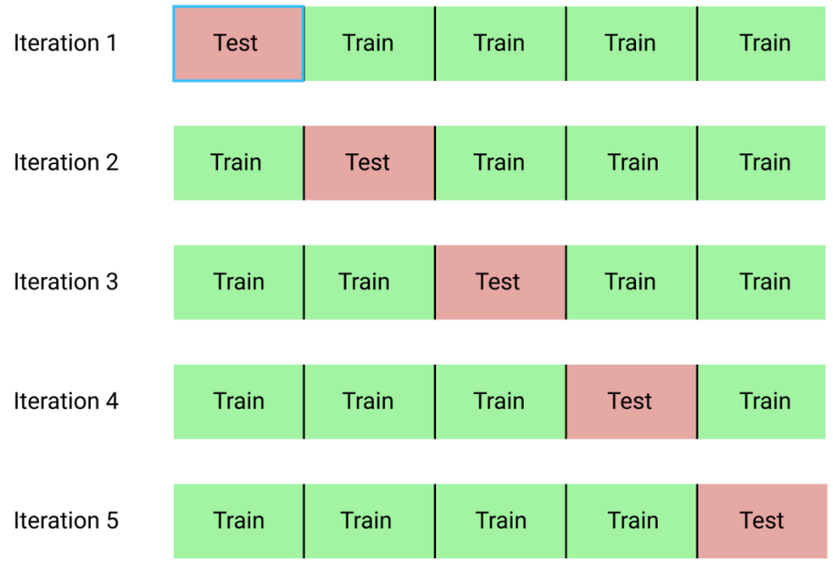
- Good for small datasets (size N)
  - remove one data point at a time, starting with the first, then the second and so on
  - at each iteration, build model (e.g. using linear regression) on the remaining dataset and test on the data point that was left out
  - repeat N times and take the average of the result

```
Let D be the original data set

testResults = []
for i in range(len(D)):
    training = D[:].pop(i)
    model = buildModel(training)
    testResults.append(test(model, D[i]))

Average testResults
```



k-fold Cross Validation

- Good for bigger datasets



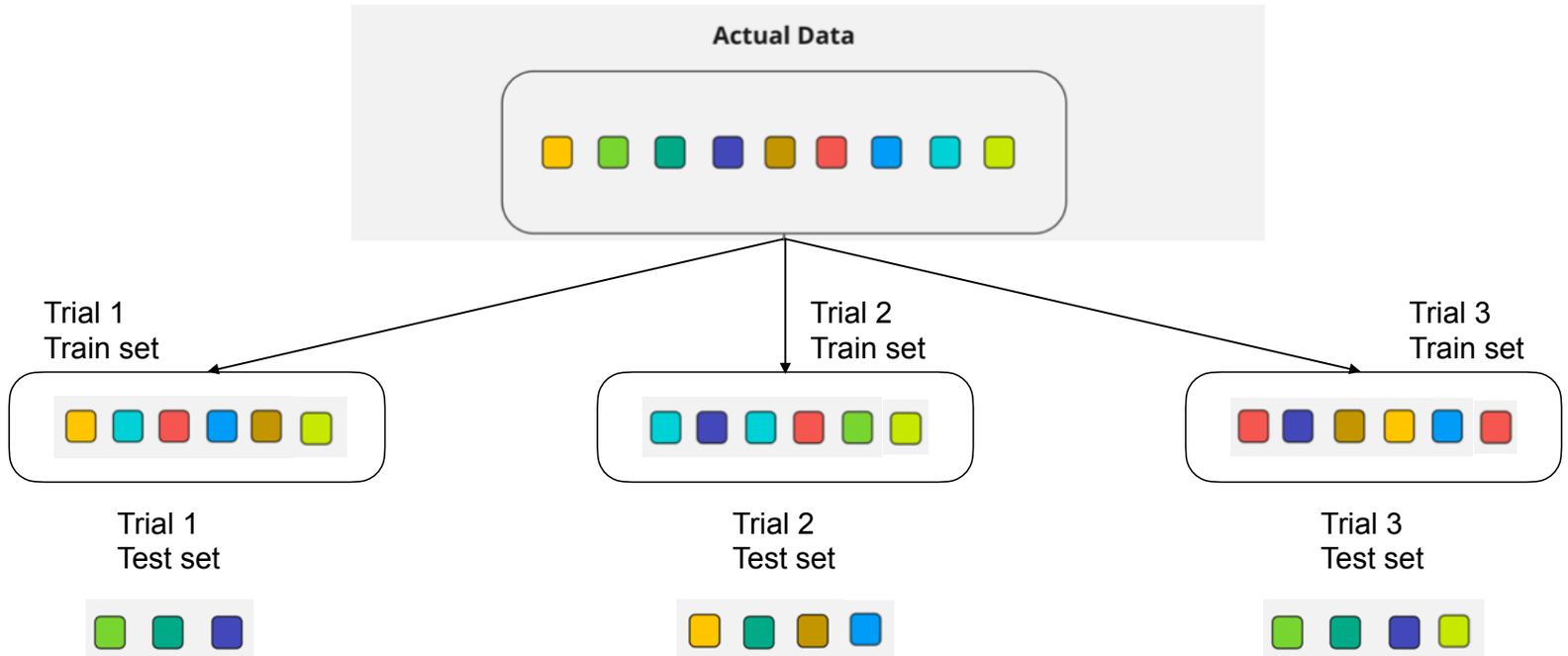
- Steps:

1. Divide dataset to k equal-sized chunks (k is decided manually)
2. Leave one of them out, use the rest ($k-1$ chunks) to build the model, then use that model to predict on the chunk you left out (e.g. test set accuracy)
3. Repeat by leaving out a different chunk

- Same idea but working on chunks rather than single data points

Bootstrapping – Sampling with Replacement

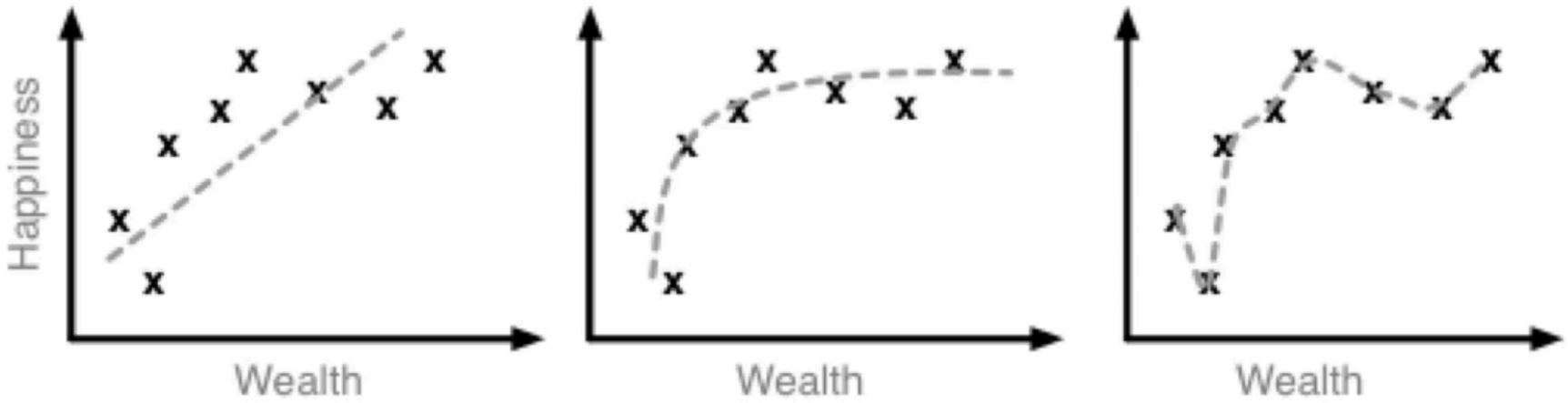
- Also called out-of-bootstrap
- Perform k trials
- At each trial, take out n elements (data points) randomly from the dataset, to generate the bootstrapped training set
  - every time an element is taken out, it is **returned** and can be resampled with equal chance as any other element
- Build model on training set and test on the rest of the dataset (test set)



Cross Validation vs Bootstrapping

- Cross validation resamples without replacement in each trial. Each data point is left out of training exactly once
- A bootstrapped training set may contain multiple instances of the same data point in each trial, and may completely omit training some data points after all trials
- There's often not much of a difference between the performance of k-fold cross validation and bootstrapping

Exercise



- Which regression model is the best one for the dataset given?
- Why are the other two models not suitable?

Evaluation Metrics

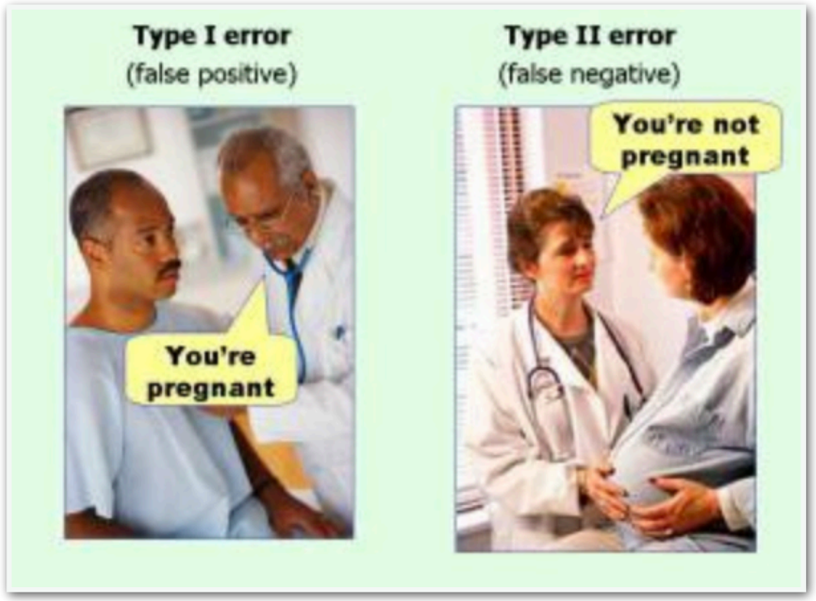
- Accuracy is one metric for evaluating classification models; the fraction of predictions our model got right
- Accuracy is not normally a great metric alone for ML tasks
- For classification, one might want to look at precision and recall
- AUC (area under the ROC curve) or F1 score are often more useful than accuracy
- For regression, metrics like R^2 and $RMSE$ (root mean squared error) are often more useful

Evaluation Metrics – Classification

		Predicted Class: Positive	Predicted Class: Negative	Recall
Actual Class	Positive	True Positive (TP)	False Negative (FN) **Type II Error**	**Sensitivity** $\frac{TP}{(TP+FN)}$
	Negative	False Positive (FP) **Type I Error**	True Negative (TN)	**Specificity** $\frac{TN}{(TN+FP)}$
		Precision $\frac{TP}{(TP+FP)}$	**Negative Predictive Value** $\frac{TN}{(TN+FN)}$	**Accuracy** $\frac{TP+TN}{(TP+TN+FP+FN)}$

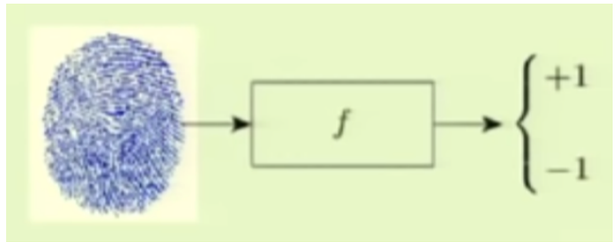
$$
\begin{aligned}
precision &= \frac{TP}{TP+FP} \\
recall &= \frac{TP}{TP+FN} \\
F1 &= \frac{2 \times precision \times recall}{precision + recall} \\
accuracy &= \frac{TP+TN}{TP+FN+TN+FP} \\
specificity &= \frac{TN}{TN+FP}
\end{aligned}
$$

- True positive (TP): Predicted positive and actually positive (correct)
- False positive/accept (FP): Predicted positive but actually negative
- True negative (TN): Predicted negative and actually negative (correct)
- False negative/reject (FN): Predicted negative but actually positive
- In **imbalanced** data, accuracy is not a meaningful measure, e.g. rare diseases (0.1%): 0.99 accuracy for predicting that someone is disease-free is useless



Example Error Measure 1

- Supermarket fingerprint verification
- Some customers are eligible for discounts, verified by their fingerprints
- False negative is a problem – customer is eligible but system fails to recognise them
  - ➡ Customer gets annoyed



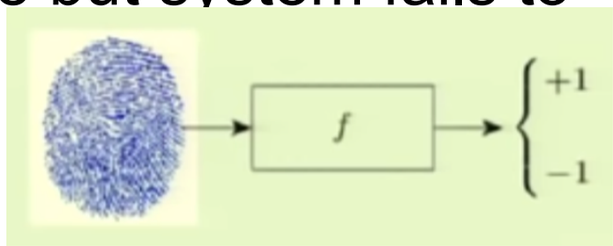
- False positive is less problematic – customer is not eligible but system incorrectly accepts their fingerprints
  - ➡ Gave away discount and fingerprint is in the system
- Use Recall to penalise false negatives

Y. Abu-Mostafa. Learning From Data. Lecture 4 (Caltech)

Example Error Measure 2

- CIA verifies fingerprint for security
- If you are an eligible employee, then the door opens to the building
- Otherwise, you are denied entry (intruder)
- False negative is ok – you are an employee but system fails to recognise you

  ➡ try again, get verified in another way



- False positive is disastrous – you are an intruder but system recognised you as an employee

  ➡ allowing an intruder into the building is unacceptable

- Use Precision to penalise false positives

Y. Abu-Mostafa. Learning From Data. Lecture 4 (Caltech)

FP and FN Check

- Legal: In a case where ML model is used to predict if someone is a criminal or not, what evaluation criteria will be important? `guilty/not_guilty`
- *"It's better that ten guilty persons escape (FN) than that one innocent suffer (FP)"*
- FN are far less problematic than FP (OK to have FNs but not FPs)
- Determine if these are FP or FN and if they are ok within their contexts
  - Airport security: keys or coins get mistaken for dangerous items (`dangerous/not_dangerous`)
  - Spam filter: a normal email is thought to be spam (`spam/ham`)
  - Airport security: aerosol deodorant accepted through scanner (`dangerous/not_dangerous`)
  - A truly innocent prisoner who has been found "guilty" (`guilty/not_guilty`)
  - Covid-19 test: An actual COVID 19 resident is predicted as healthy (`positive/negative`)

Other Supervised Learners

- Decision Trees
- Naive Bayes
- Neural Networks (covered in Deep Learning)
- Support Vector Machines
- Random Forest

Summary

- Overfitting
- Model selection using cross validation and repeated random sampling
- Model validation using evaluation metrics

Links

- Russell and Norvig Chapter 18
- A. Ng. Linear and Logistic Regression Cost Function (Videos) [LinearR1] [LinearR2] [LogisticR]
- J. Brownlee. Logistic Regression for Machine Learning [Link]
- A. Ng Deeplearning.ai Logistic Regression Working [Link] Loss & Cost function [Link], Training [Link]
- StatQuest with Josh Starmer (Videos) [Coefficients] [Max Likelihood] [Significance] [R-squared]
- DS StackExchange. How F1 score is good with unbalanced dataset [Link]

Tumour Classifier

- Let's suppose we have a classification model that is trying to predict if a tumour is harmful (positive class) or not (negative class)
- Below is the confusion matrix of the model's classification of 100 tumours

True Positives (TPs): 1	False Positives (FPs): 1
False Negatives (FNs): 8	True Negatives (TNs): 90

- Calculate the accuracy of the model
- Is the accuracy enough to determine the model's performance?
- What other metrics would you use?