
Day 1 Lecture 2

# Classification

# Image Classification

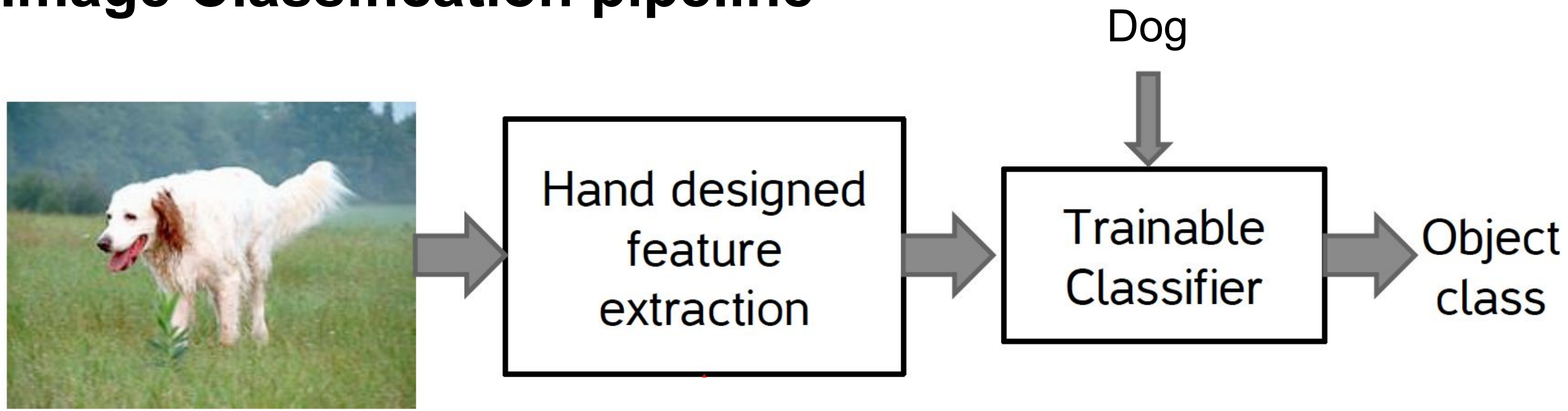
Set of predefined categories [eg: table, apple, dog, giraffe]
Binary classification [1, 0]

DOG

# Image Classification

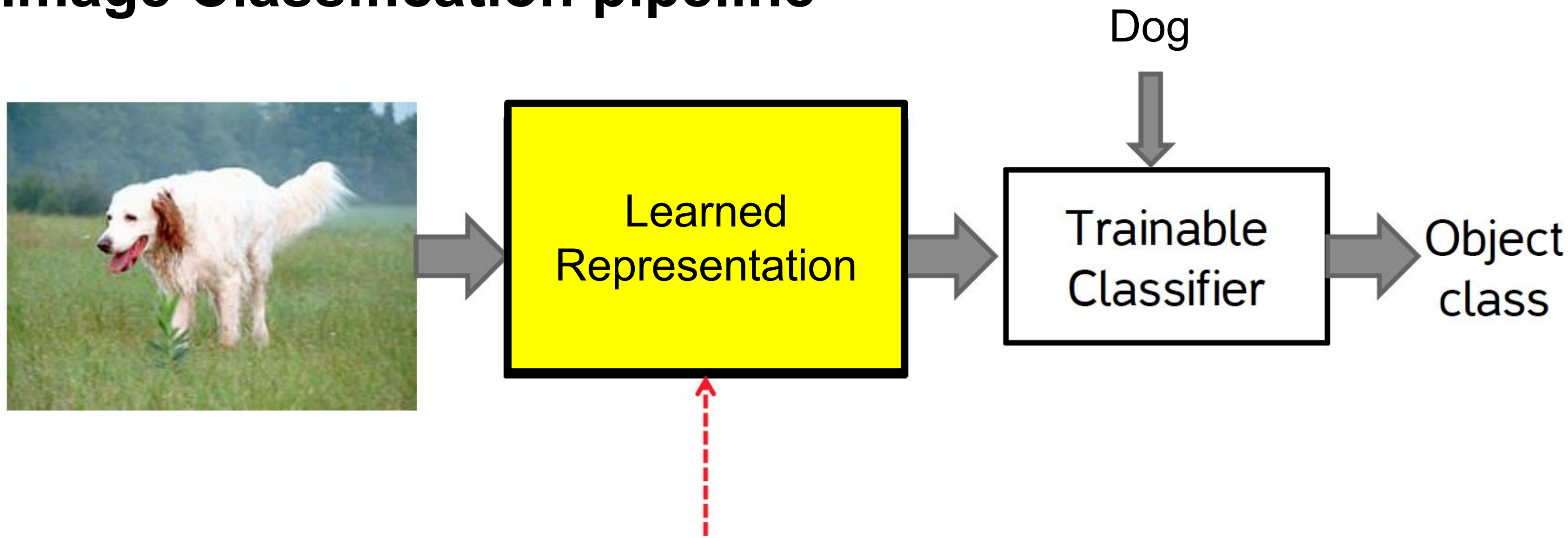
```
[[[132 131 118]
  [164 163 153]
  [209 208 200]
  ...,
  [247 249 251]
  [246 248 251]
  [246 248 251]]

 [[147 148 136]
  [187 186 178]
  [226 226 219]
  ...,
  [247 249 251]
  [246 248 251]
  [246 248 251]]

 [[158 159 149]
  [205 206 198]
  [237 238 232]
  ...,
  [247 249 251]
  [246 248 251]
  [246 248 251]]

 ...,
```

# Image Classification pipeline




Slide credit: Jose M Àlvarez

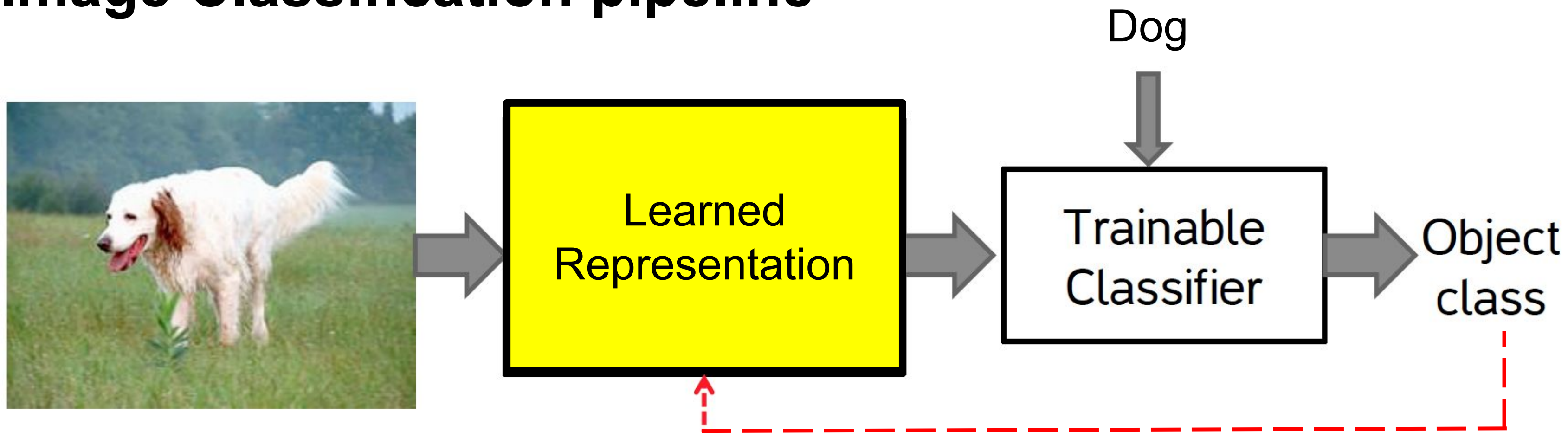
# Image Classification pipeline




Instead of design features,
let’s design feature learners;
**Deep learning provides a way of doing it**

Slide credit: Jose M Àlvarez

# Image Classification pipeline




Part I: End-to-end learning (E2E)

<u>Slide credit</u>: Jose M Àlvarez

# Image Classification: Example Datasets

training set of 60,000 examples

test set of 10,000 examples

**THE MNIST DATABASE**

of handwritten digits

Yann LeCun, Courant Institute, NYU
Corinna Cortes, Google Labs, New York
Christopher J.C. Burges, Microsoft Research, Redmond

# Image Classification: Example Datasets

20 classes

PASCAL2

Pattern Analysis, Statistical Modelling and Computational Learning

# Training set

$$\{(\mathbf{x}_i, y_i) : 1 \leq i \leq N\}$$

$$X = \begin{bmatrix} 2.1 & 3.2 & 4.8 & 0.1 & 0.0 & 2.6 \\ 3.1 & 1.4 & 2.5 & 0.2 & 1.0 & 2.0 \\ 1.0 & 2.3 & 3.2 & 9.3 & 6.4 & 0.3 \\ 2.0 & 5.0 & 3.2 & 1.0 & 6.9 & 9.1 \\ 9.0 & 3.5 & 5.4 & 5.5 & 3.2 & 1.0 \end{bmatrix}$$

*N* training examples (rows)

*D* features (columns)

$\mathbf{x}_3^T$

$$\mathbf{y} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

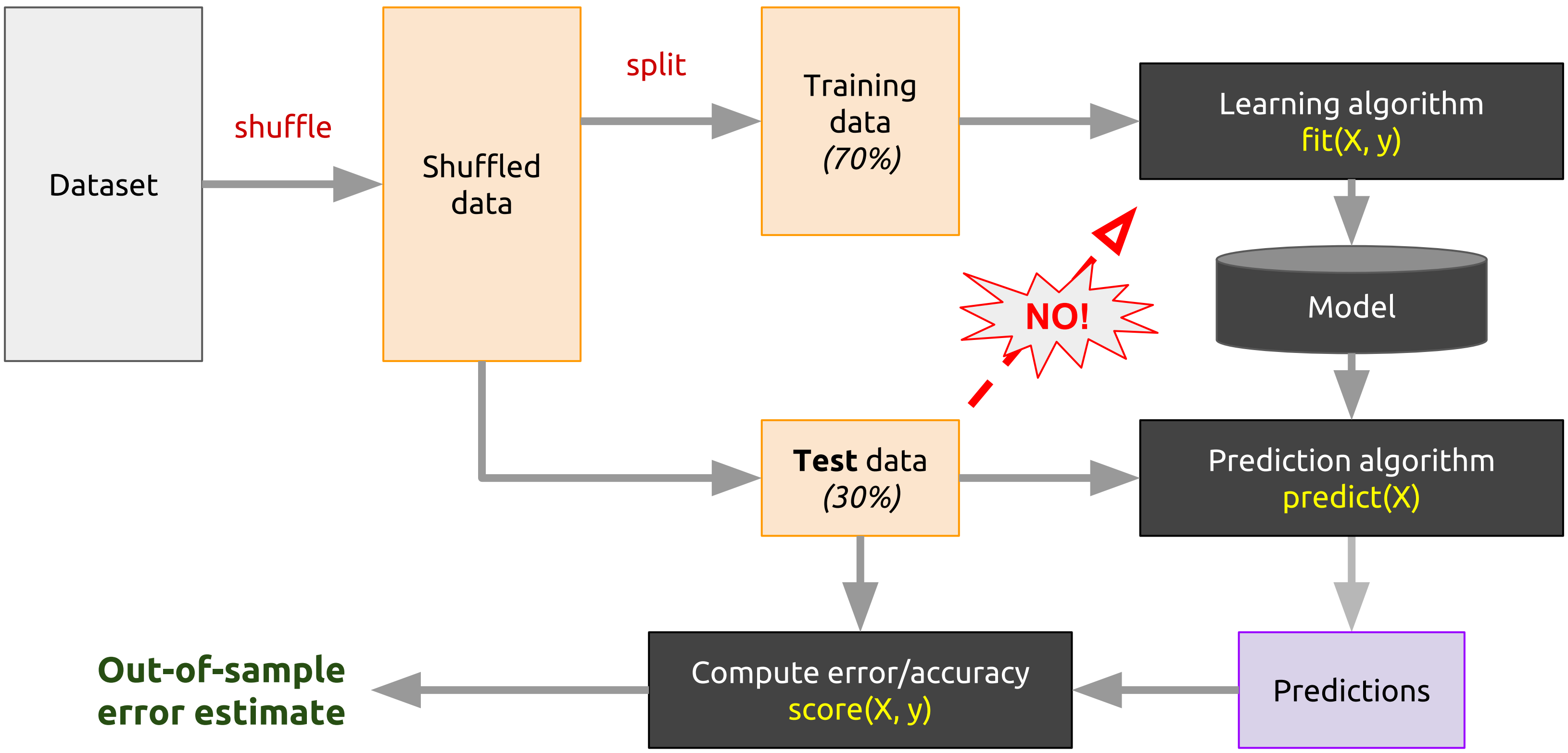
*N*

$y_3$

$$X \in \mathbb{R}^{N \times D}$$

$$\mathbf{y} \in \{0, 1\}^N$$

# Train/Test Splits



# Metrics

Confusion matrices provide a by-class comparison between the results of the automatic classifications with ground truth annotations.

| | | Automatic | | |
|---|---|---|---|---|
| | | class1 | class2 | class3 |
| Manual | class1 | 12 | 1 | 0 |
| | class2 | 3 | 13 | 0 |
| | class3 | 0 | 0 | 20 |

| | | Automatic | | |
|---|---|---|---|---|
| | | class1 | class2 | class3 |
| Manual | class1 | 100% | 0% | 0% |
| | class2 | 0% | 100% | 0% |
| | class3 | 0% | 0% | 100% |

# Metrics

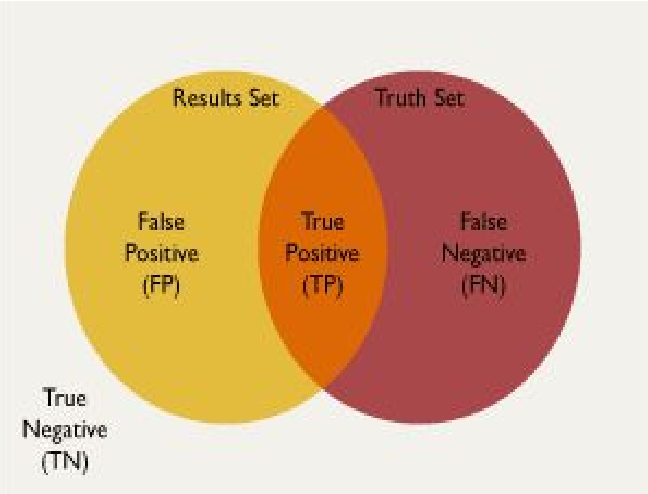
Correct classifications appear in the diagonal, while the rest of cells correspond to errors.

| | | Prediction | | |
|---|---|---|---|---|
| | | Class 1 | Class 2 | Class 3 |
| Ground Truth | Class 1 | x(1,1) | x(1,2) | x(1,3) |
| | Class 2 | x(2,1) | x(2,2) | x(2,3) |
| | Class 3 | x(3,1) | x(3,2) | x(3,3) |

# Metrics

Special case: Binary classifiers in terms of "Positive" vs "Negative".

| | | Prediction | |
|---|---|---|---|
| | | Positives | negative |
| Ground Truth | Positives | True positive (TP) | False negative (FN) |
| | negative | False positives (FP) | True negative (TN) |



# Metrics

The "accuracy" measures the proportion of correct classifications, not distinguishing between classes.

$$Accuracy = \frac{\sum_{i=1}^{3} x(i,i)}{\sum_{i=1}^{3}\sum_{j=1}^{3} x(i,j)}$$

Binary

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

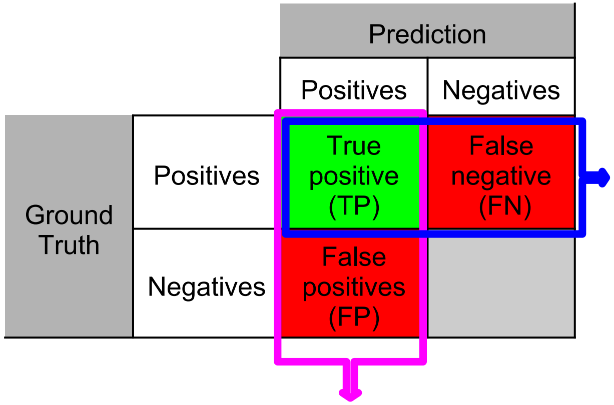
| | | Prediction | | |
|---|---|---|---|---|
| | | Class 1 | Class 2 | Class 3 |
| Ground Truth | Class 1 | x(1,1) | x(1,2) | x(1,3) |
| | Class 2 | x(2,1) | x(2,2) | x(2,3) |
| | Class 3 | x(3,1) | x(3,2) | x(3,3) |

| | | Prediction | |
|---|---|---|---|
| | | Positives | negative |
| Ground Truth | Positives | True positive (TP) | False negative (FN) |
| | Negative | False positives (FP) | True negative (TN) |

# Metrics

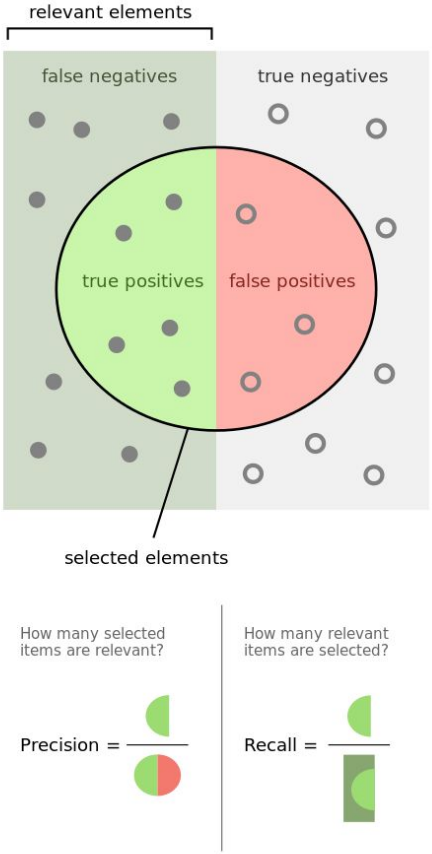
Given a reference class, its [Precision (P)] and [Recall (R)] are complementary measures of relevance.

Example: Relevant class is "Positive" in a binary classifier.




$$Recall = \frac{TP}{TP+FN}$$

$$Precision = \frac{TP}{TP+FP}$$



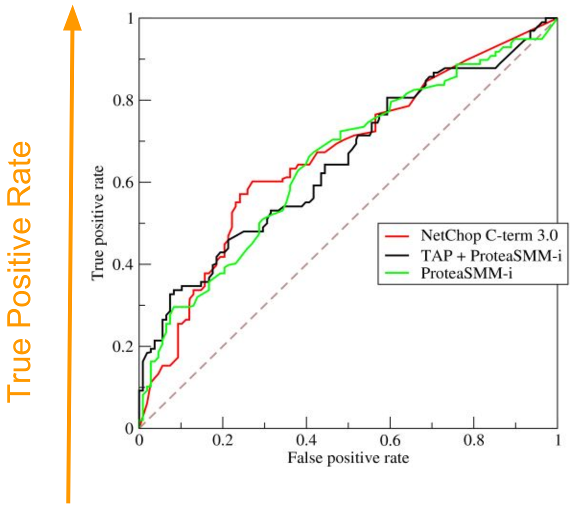
# Metrics

Binary classification results often [depend from a parameter]() (eg. decision threshold) whose value directly impacts precision and recall.

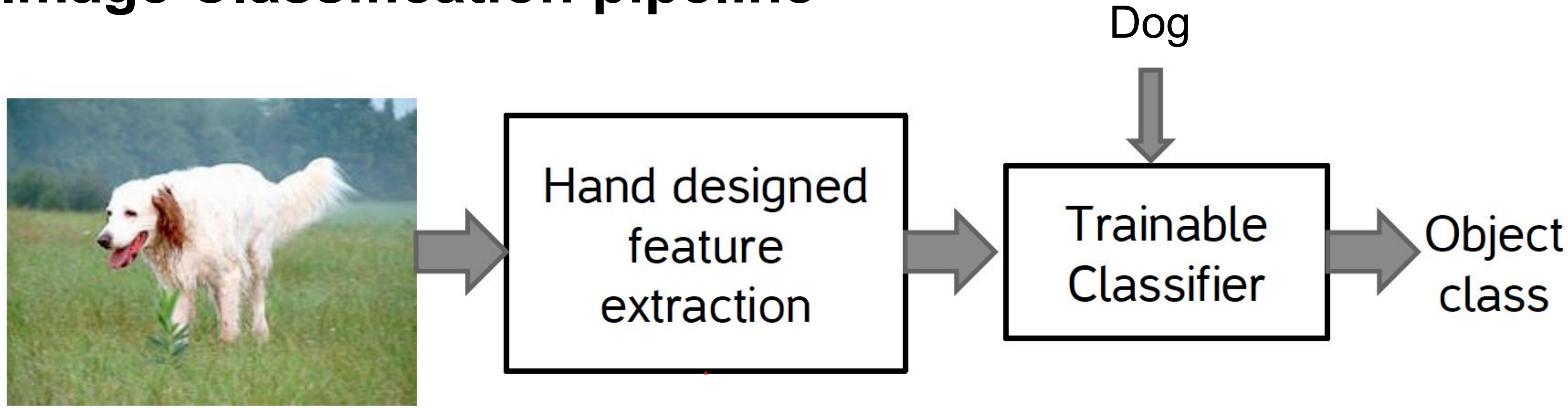
For this reason, in many cases a [Receiver Operating Curve]() (ROC curve) is provided as a result.

$$True\ Positive\ Rate = \frac{TP}{TP+FN} = Recall = Sensitivity$$

$$False\ Positive\ Rate = \frac{FP}{TP+FN} = 1 - specificity$$



# Image Classification pipeline

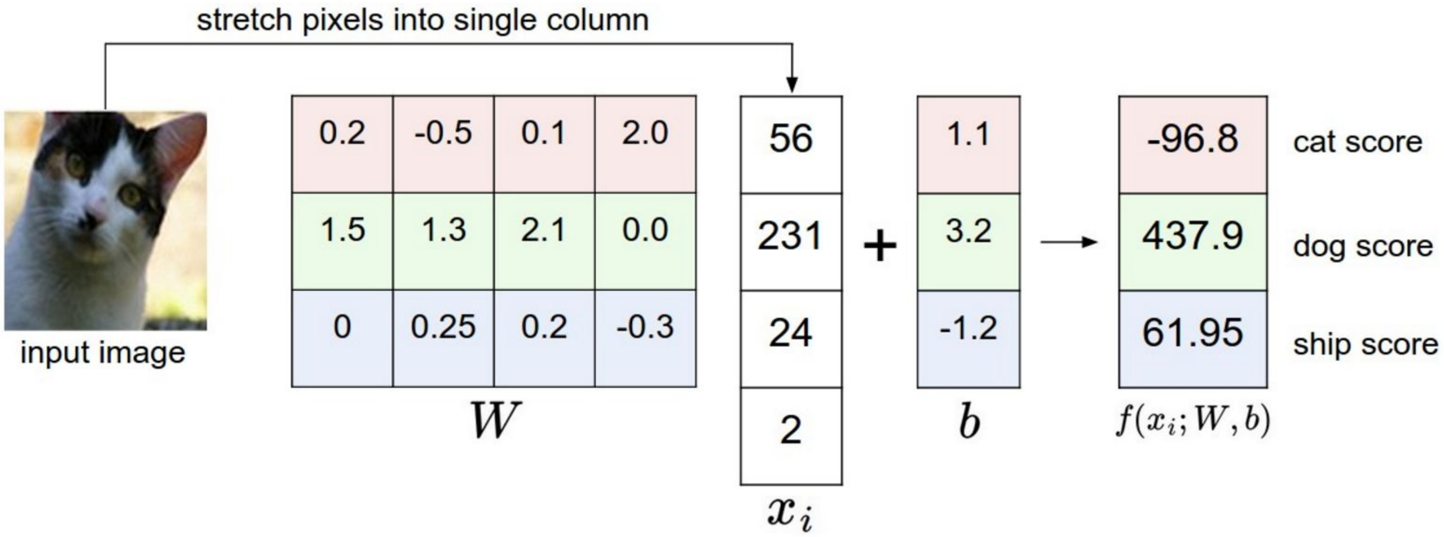



Slide credit: Jose M Àlvarez

# Linear Models

Mapping function to predict a score for the class label

$$f(x, \mathbf{w}) = (\mathbf{w}^T x + b)$$



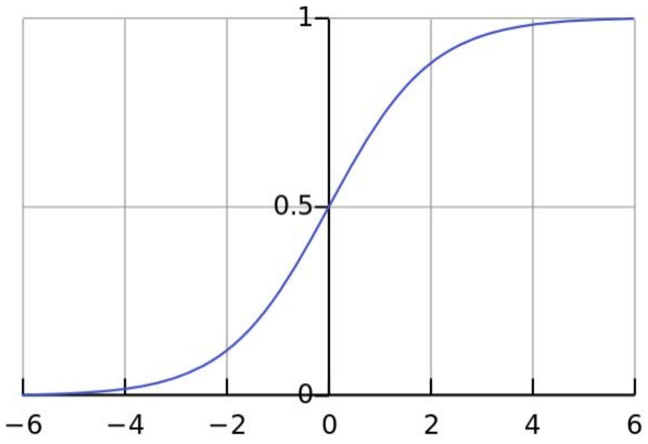

[CS231n: Convolutional Neural Networks for Visual Recognition]

# Sigmoid

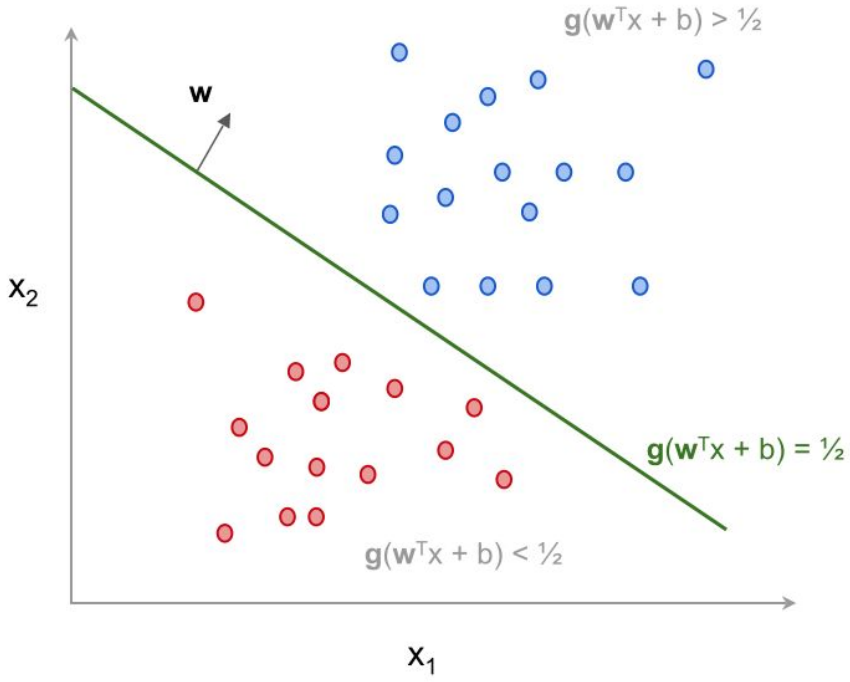
Activation function: Turn score into probabilities

$$f(x, \mathbf{w}) = g(\mathbf{w}^T x + b)$$

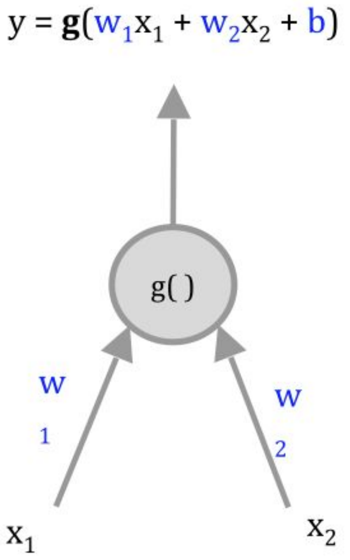
$$g(x) = \frac{1}{1 + e^{-x}}$$

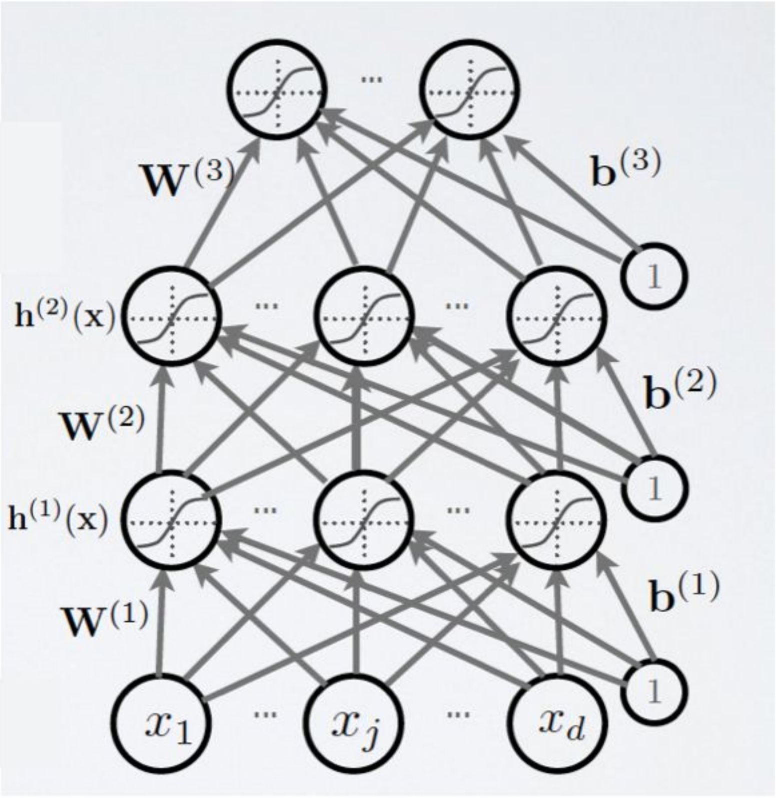



Logistic Regression



# Neuron




Slide Credit: Hugo Laroche NN course

# Data hygiene

Split your dataset into train and test at the very start

- Usually good practice to shuffle data (exception: time series)

Do not look at test data (data snooping)!

- Lock it away at the start to prevent contamination

**NB: Never** ever train on the test data!

- You have no way to estimate error if you do
- Your model could easily overfit the test data and have poor generalization, you have no way of knowing without test data
- Model may fail in production