# Announcements

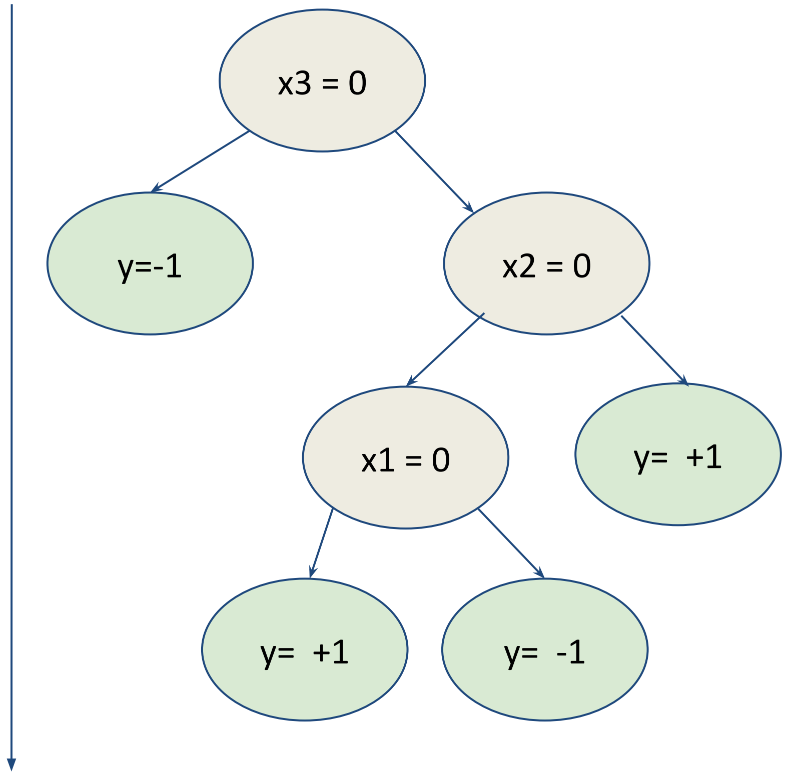
- Pset-2 due on March 6th.

# Last time

- Mixtures of Gaussians
  - Expectation Maximization
- Linear Regression
- Analyzing your model

# Quiz-2

depth = 3




| x1 | x2 | x3 | y |
|---|---|---|---|
| 1 | 1 | 1 | +1 |
| 0 | 1 | 0 | -1 |
| 1 | 0 | 1 | -1 |
| 0 | 0 | 1 | +1 |

# Quiz-2

What is one benefit cosine distance have over Euclidean distance?

1. Cosine distance normalizes the features being compared
2. Cosine distance is faster to compute
3. Cosine distance performs better

# Quiz-2

What is one benefit cosine distance have over Euclidean distance?

1. **Cosine distance normalizes the features being compared**
2. Cosine distance is faster to compute
3. Cosine distance performs better

# Diagnosing model's performance

- **Scenario:** Model performs well during training but performs poorly when deployed in production (test) environment.

- **Potential reasons:**
  - Training v/s test data mismatch.
  - Seasonal concepts (eg: political figures, covid masks)
  - Models trained on datasets curated in one part of the hemisphere but deployed globally.

# Examples of geographical mismatch

**Azure:** food, cheese
**Clarifai:** food, wood
**Google:** food, dish
**Amazon:** food, confectionary

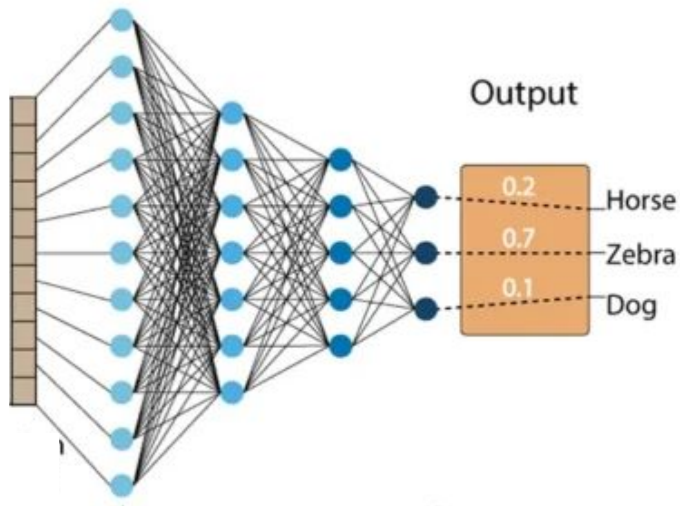
**Azure:** toilet, design
**Clarifai:** people, faucet
**Google:** product, liquid
**Amazon:** sink, indoors

Does Object Recognition Work for Everyone?, CVPR 19

# Geographically Diverse Evaluation Dataset for Object Recognition (GeoDE)


GeoDE: a Geographically Diverse Evaluation Dataset for Object Recognition, NeurIPS'23

# A practical scenario




- New “viral” concept: **masks** (during early 2020)
- How do you tweak existing classifier to work well on the new concept.

Do not edit
How to change the design

# What are some effective ways to successfully predict a new class: mask?

Presenting with animations, GIFs or speaker notes? Enable our Chrome extension

slido

**What are some effective ways to successfully predict a new class: mask?**

Train a stand-alone MLP on "mask" category and use two classifiers in practice: one just for masks, one for rest of the objects.

39%

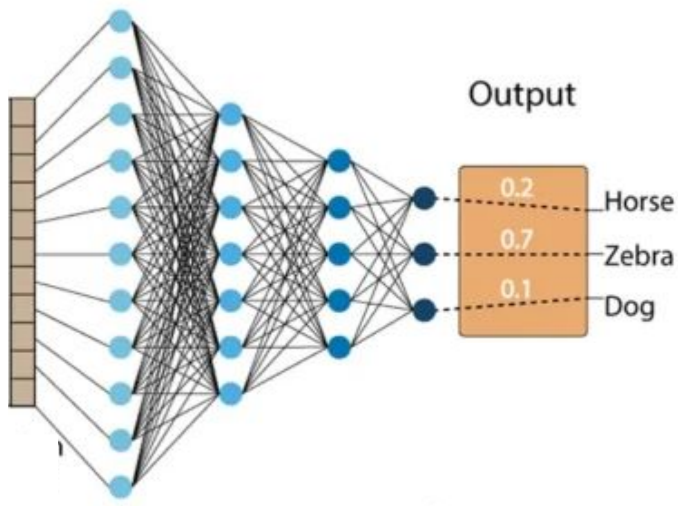
Retrain the large-scale classifier by adding a new class

16%

Keep the large-scale classifier intact, fine-tune the last layer to predict mask

46%

# A practical scenario




- New “viral” concept: **masks** (during early 2020)
- **Steps**:
  - Collect +ve and -ve images of the object class **mask**
  - Train just the classification head

# Diagnosing model's performance

- **Scenario:** Model performs well during training but performs poorly when deployed in production (test) environment.

- **Potential solutions:**

| | Size | Price |
|---|---|---|
| train | 2104 | 400 |
| train | 1600 | 330 |
| train | 2400 | 369 |
| train | 1416 | 232 |
| train | 3000 | 540 |
| val | 1985 | 300 |
| val | 1534 | 315 |
| val | 1427 | 199 |
| test | 1380 | 212 |
| test | 1494 | 243 |

→ **Offline** test split

**Online** test split

- Keep sampling from live production data.
- Keep evaluating model performance on both offline and online test splits.
- Refine pre-training, fine-tuning, offline test splits

# Today

- Model Selection using AIC/BIC
- Robust Learning
  - Different loss functions
  - Boosting
  - Weak learners
  - Regression Trees

# The Bias-Variance Trade-off

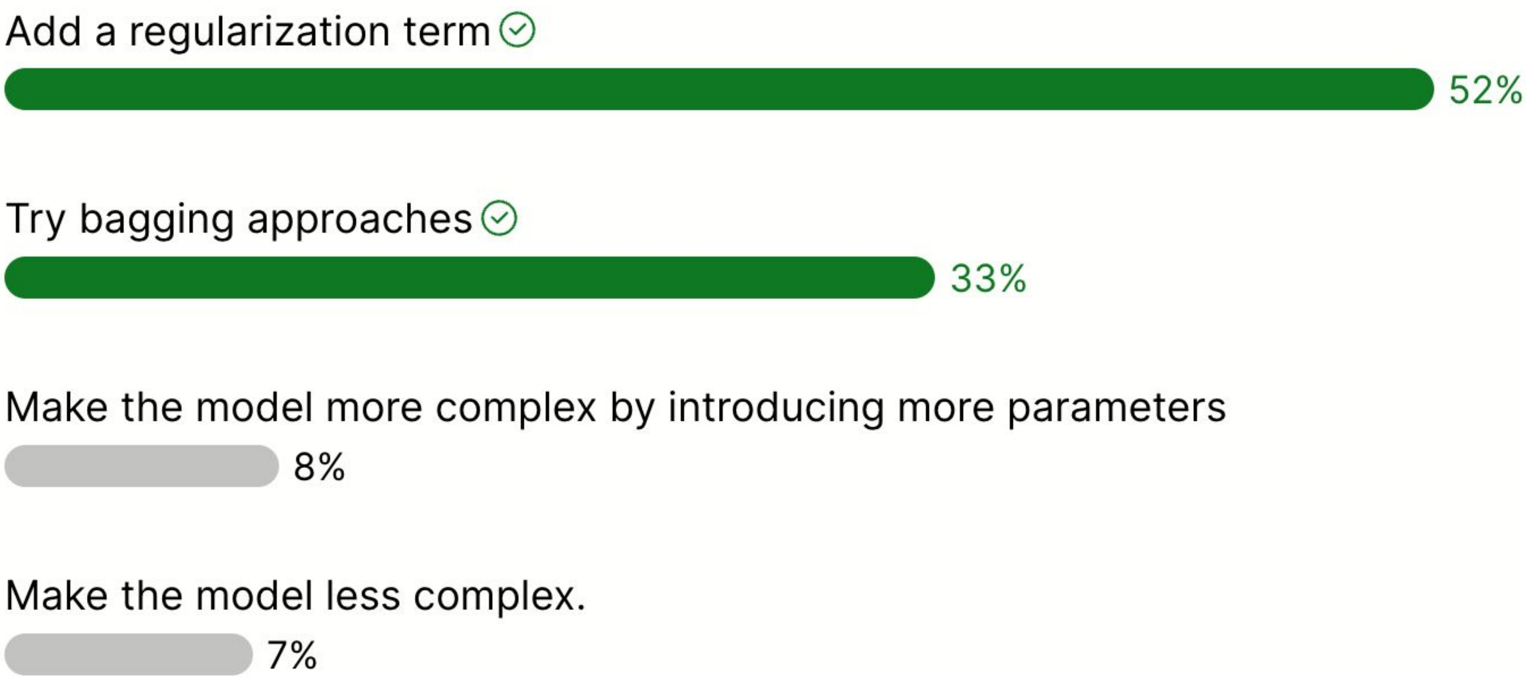
There is a trade-off between bias and variance:

- **Low bias:** Less overfitting.
- **Low variance:** Low variability in the loss on test data

- Less complex models (fewer parameters) have high bias and hence low variance
- More complex models (more parameters) have low bias and hence high variance
- Optimal model will have a balance

# How to reduce the variance of  model but also not increase bias by much? Select all that apply.

**How to reduce the variance of model but also not increase bias by much? Select all that apply.**



# Debugging a learning algorithm

- How do you fix high variance and high bias? 

**To fix high variance**

- Get more training examples
- Try smaller sets of features
- Try increasing $\lambda$
- Bagging, e.g. Random Forest

**To fix high bias**

- Try getting additional features
- Try adding polynomial features
- Try decreasing $\lambda$

# Today

- **Model Selection using AIC/BIC**
- Robust Learning
  - Different loss functions
  - Boosting
  - Weak learners
  - Regression Trees

# Model selection

- Akaike information criterion (AIC)

$$\mathrm{AIC} = \underset{\text{number of model parameters}}{2k} - \underbrace{2\ln(\hat{L})}_{\text{maximized value of the likelihood function}}$$

- Founded in information theory (derivation beyond the scope of this class)
- **Goal:** Try different values of k, pick the one with least AIC.

# Is there a difference between cross-validation and using AIC? Select all that apply

**Is there a difference between cross-validation and using AIC? Select all that apply**

No, both help pick the most fitting model.
3%

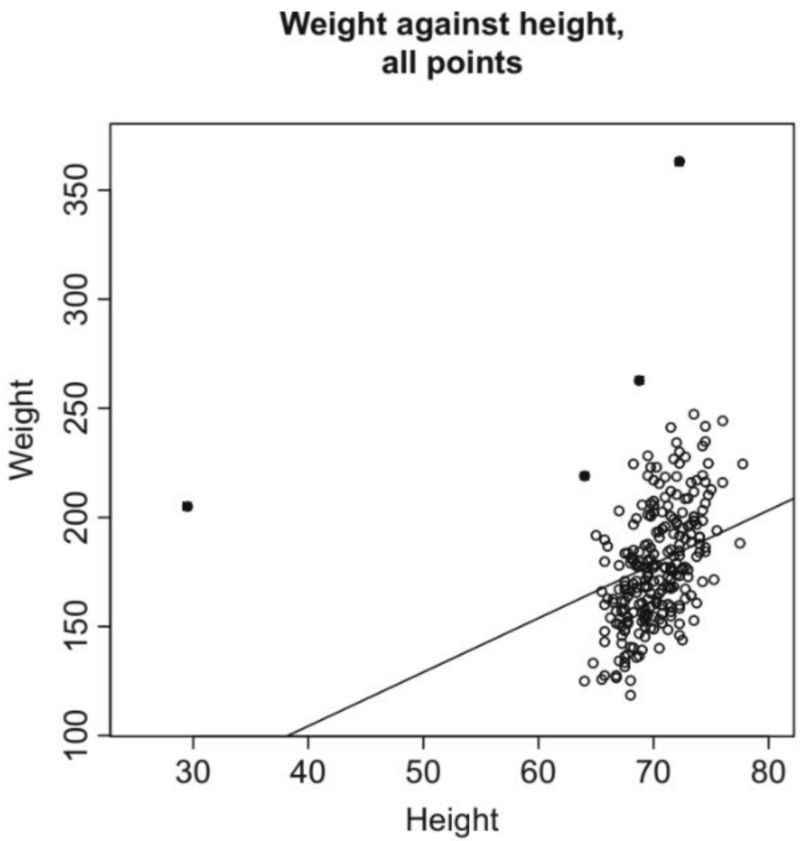
Yes, cross-validation does not directly help with model selection based on model complexity ✓
98%

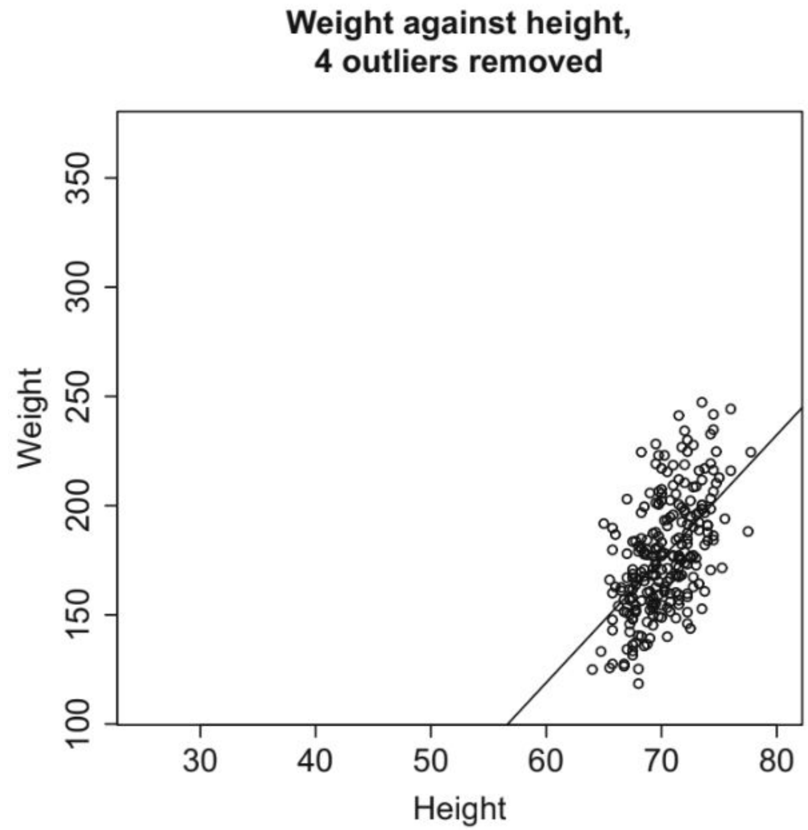
Yes, cross-validation helps with picking the most generalizable model ✓
60%

# Today

- Model Selection using AIC/BIC
- **Robust Learning**
  - **Different loss functions**
  - Boosting
  - Weak learners
  - Regression Trees

# **Recall:** Outliers are problematic



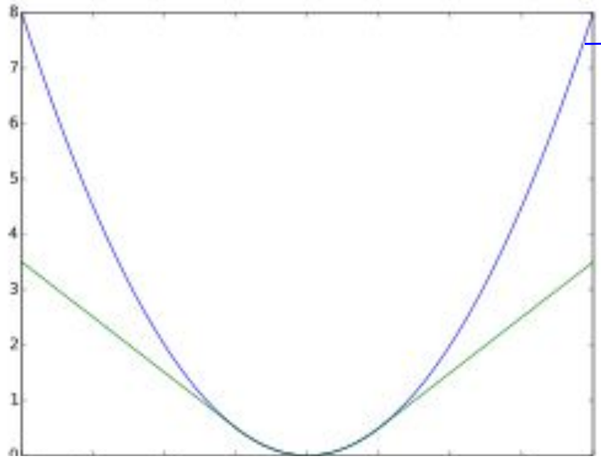



# Solution: Huber loss

**High level idea:** Reduce the influence of outliers on the overall loss.

$$L_\delta(a) = \begin{cases} \frac{1}{2}a^2 & \text{for } |a| \le \delta, \\ \delta \cdot \left(|a| - \frac{1}{2}\delta\right), & \text{otherwise.} \end{cases}$$

a = error




MAE

**huber loss**

# Logistic Regression

sigmoid function

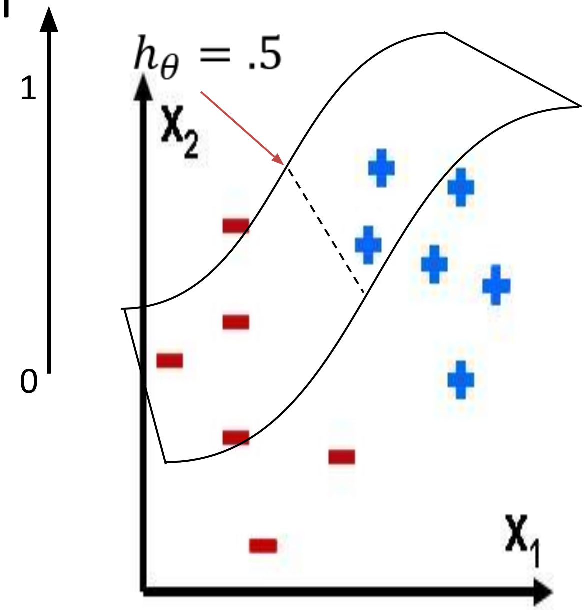
Hypothesis:

$$h_\theta(x) = P(y=0|x) = \frac{1}{1+e^{x^T\beta}}$$

predict "$y=1$" if $P(y=1\,|x) \geq 0.5$

predict "$y=0$" if $P(y=1\,|x) < 0.5$

"decision boundary"



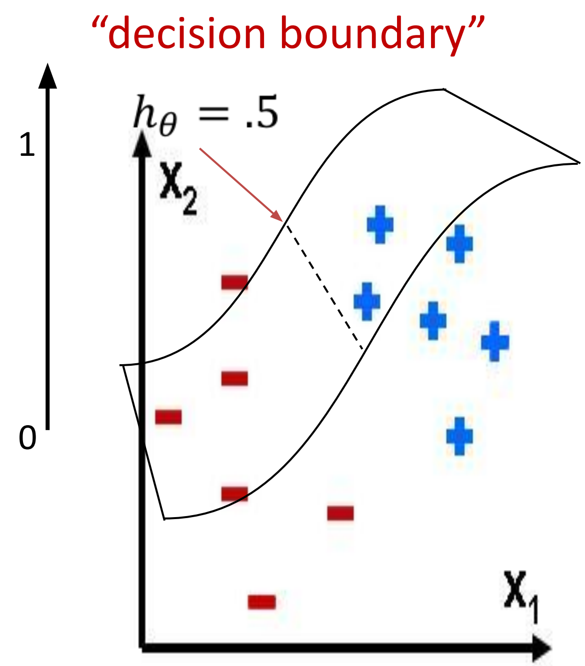
# Logistic Regression Cost

Hypothesis:

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

$\theta$: parameters

$D = \left(x^{(i)}, y^{(i)}\right)$ : data




Cost Function: cross entropy

$$J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)})$$

Goal: minimize cost $\quad \min_\theta J(\theta)$

# Logistic Regression: Cross-entropy loss

$$\text{Log Loss} = \sum_{(x,y)\in D} -y\log(y') - (1-y)\log(1-y')$$

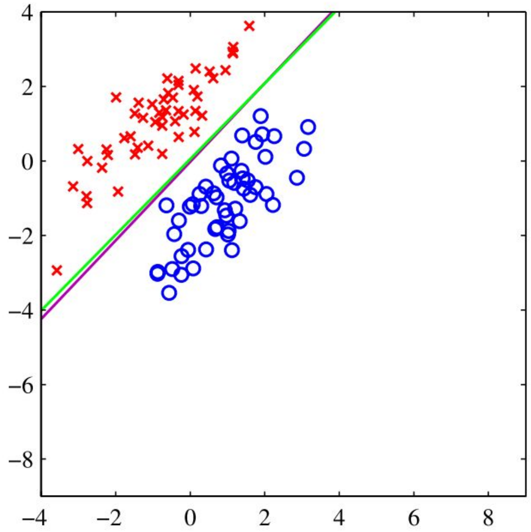
- $y$ is the label in a labeled example. Since this is logistic regression, every value of $y$ must either be 0 or 1.
- $y'$ is your model's prediction (somewhere between 0 and 1), given the set of features in $x$.

- when y = 1, cross-entropy loss reduces to -log(y')

# Least Squares vs. Logistic Regression for Classification

**Least squares**

**Cross-entropy**

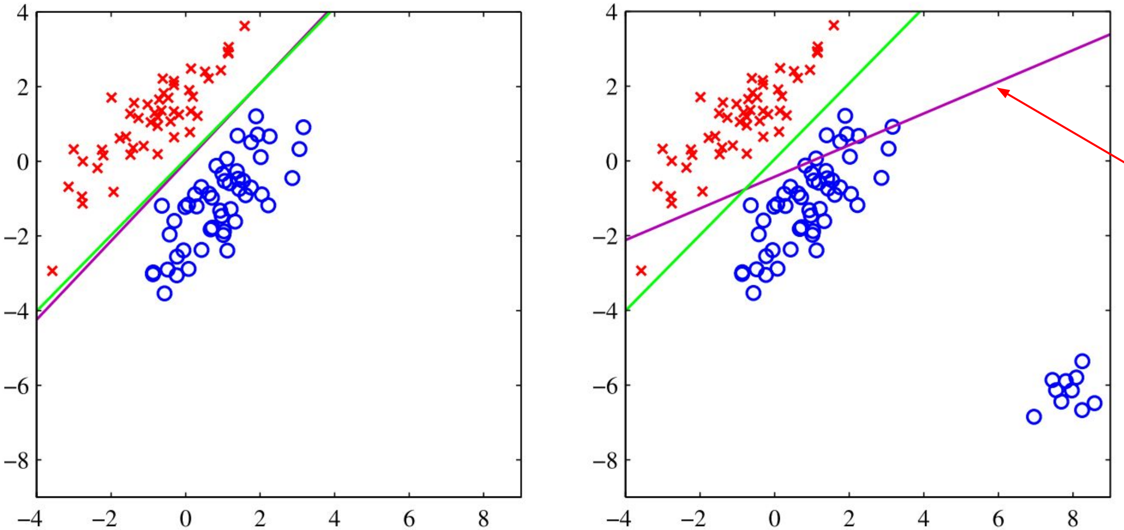



(see Bishop 4.1.3 for more details)

# Least Squares vs. Logistic Regression for Classification

Least squares

Cross-entropy

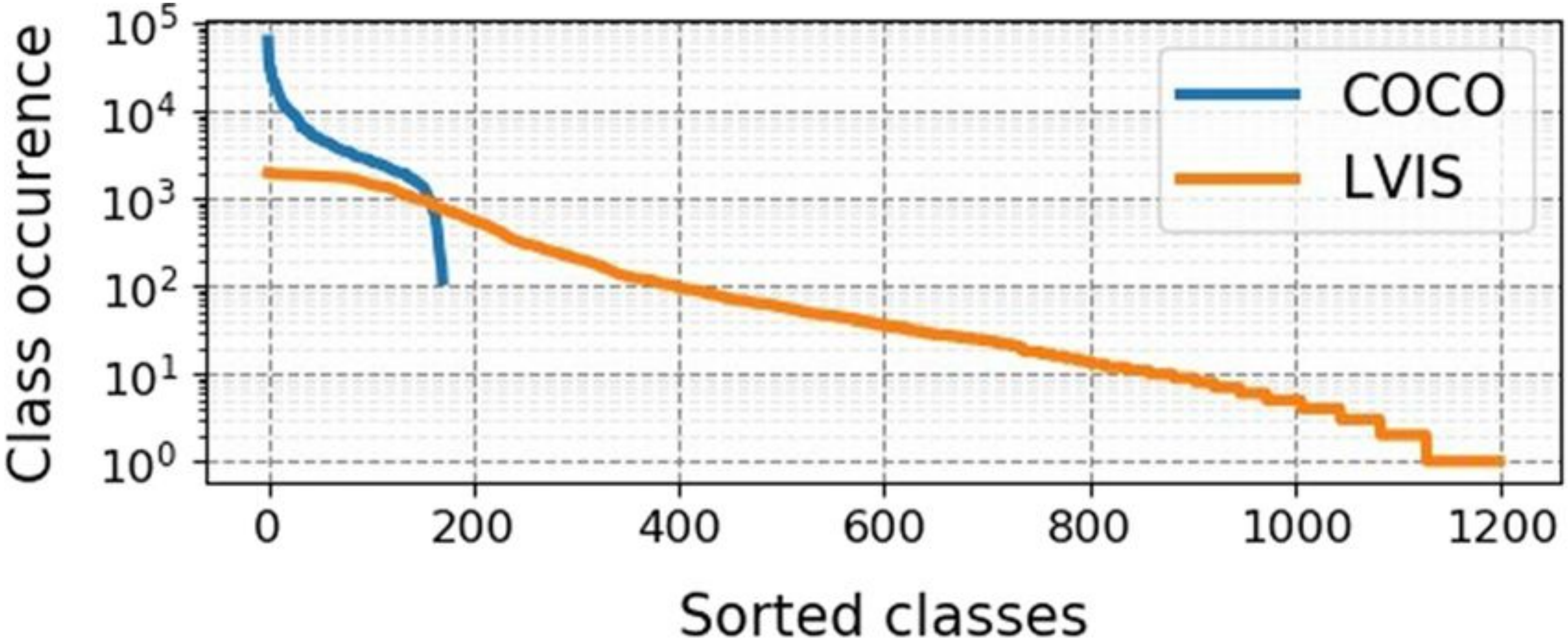



Least squares

- **Finding:** When extra data points are added at the bottom left of the diagram, showing that least squares is highly sensitive to outliers.

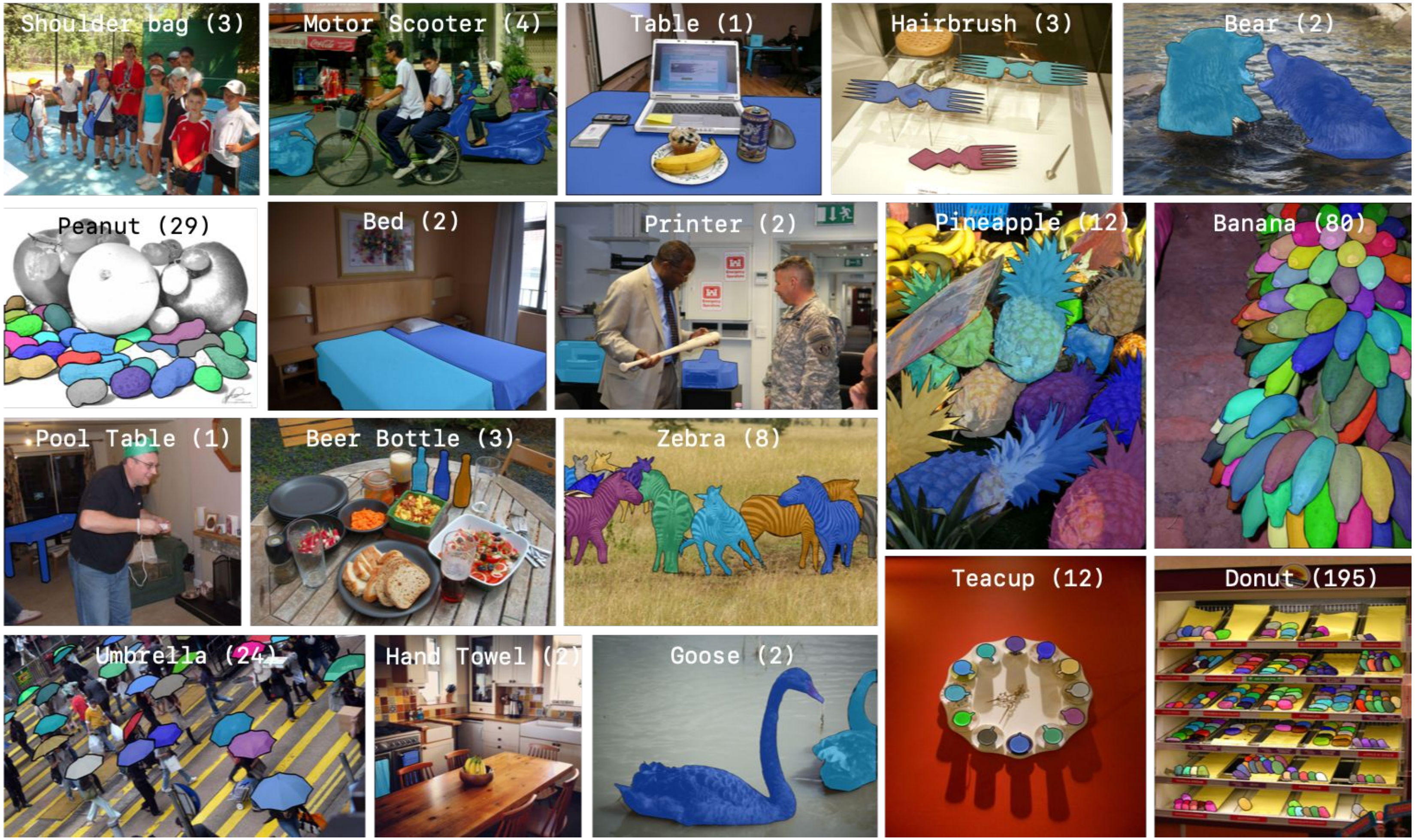
(see Bishop 4.1.3 for more details)

# Recall: long tailed distribution




https://www.lvisdataset.org/

# Large Vocabulary Instance Segmentation (LVIS)




https://www.lvisdataset.org/

# Focal loss

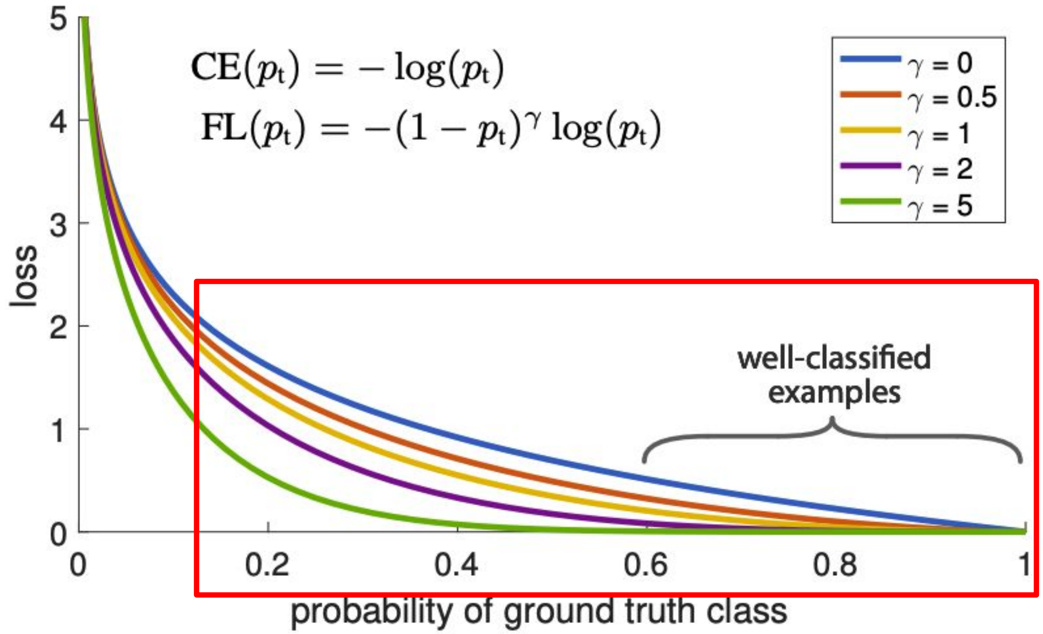




Figure 1. We propose a novel loss we term the *Focal Loss* that adds a factor $(1 - p_t)^\gamma$ to the standard cross entropy criterion. Setting $\gamma > 0$ reduces the relative loss for well-classified examples ($p_t > .5$), putting more focus on hard, misclassified examples. As our experiments will demonstrate, the proposed focal loss enables training highly accurate dense object detectors in the presence of vast numbers of easy background examples.

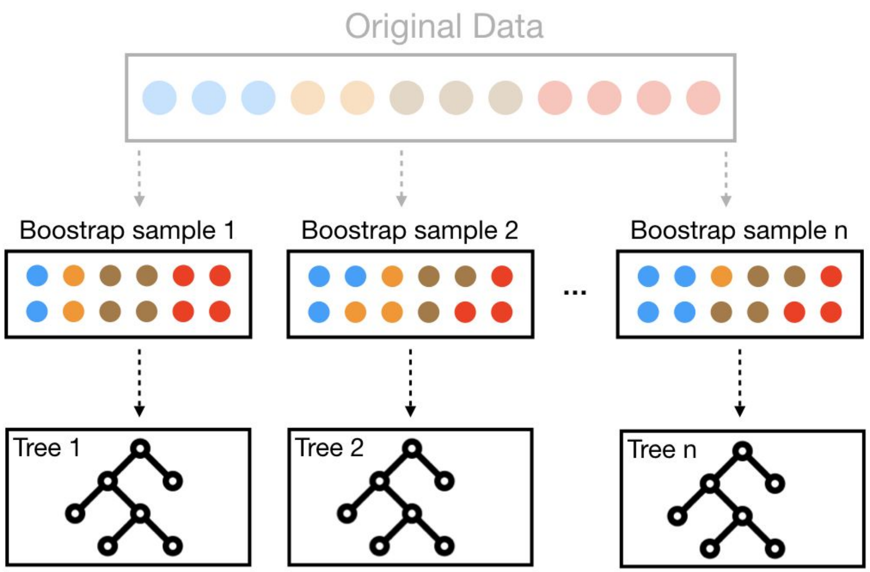
# Today

- Model Selection using AIC/BIC
- **Robust Learning**
  - Different loss functions
  - **Boosting**
  - Weak learners
  - Regression Trees

# Bootstrap Aggregating: wisdom of the crowd

1. Sample with replacement (aka "bootstrap" the training data)

2. Fit an overgrown tree to each resampled data set

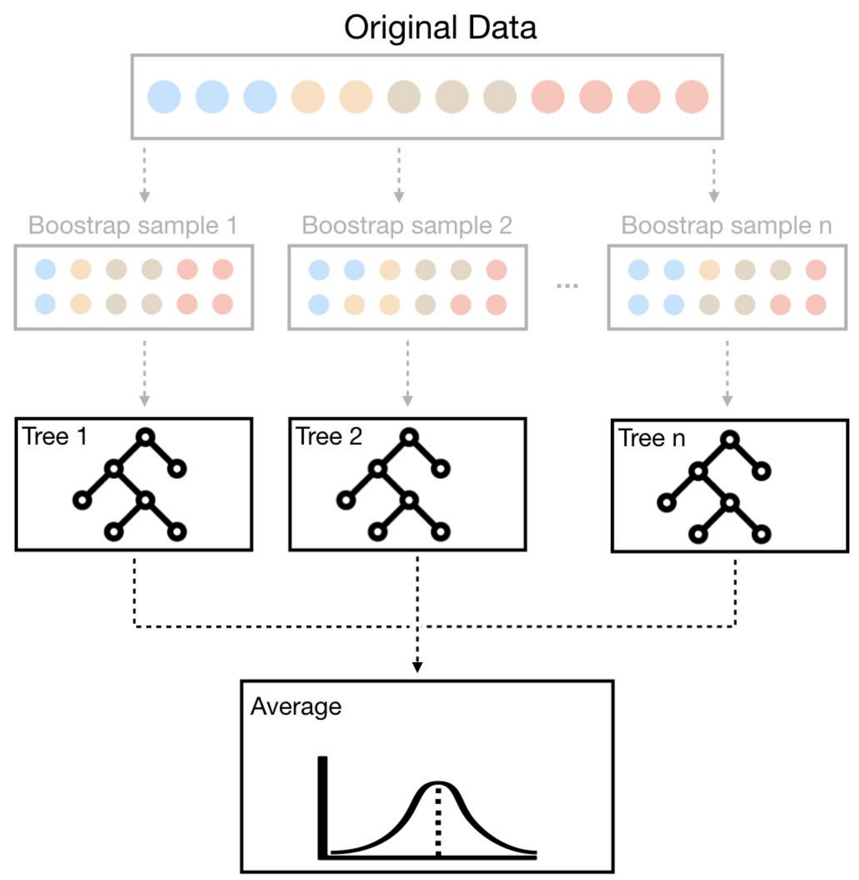


# Bootstrap Aggregating: wisdom of the crowd

1. Sample with replacement (aka "bootstrap" the training data)

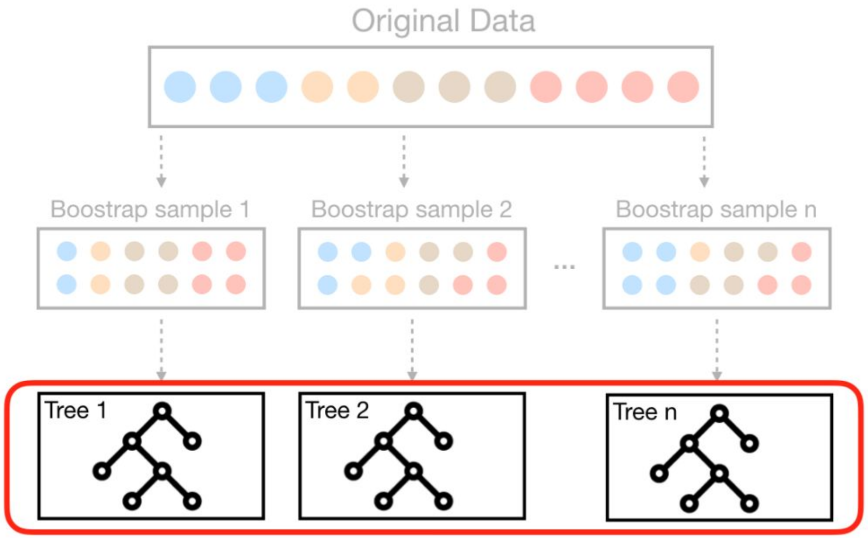
2. Fit an overgrown tree to each resampled data set

3. Average predictions



# Cons of Bagging

- Follow a similar bagging process but…

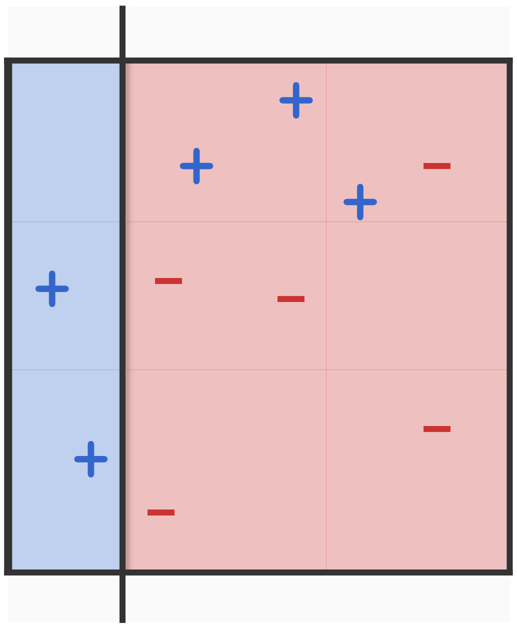



Bagging may produce many correlated trees

# Today

- Model Selection using AIC/BIC
- **Robust Learning**
  - Different loss functions
  - Boosting
  - **Weak learners**
  - Regression Trees

# A toy example




$f^{(1)}$

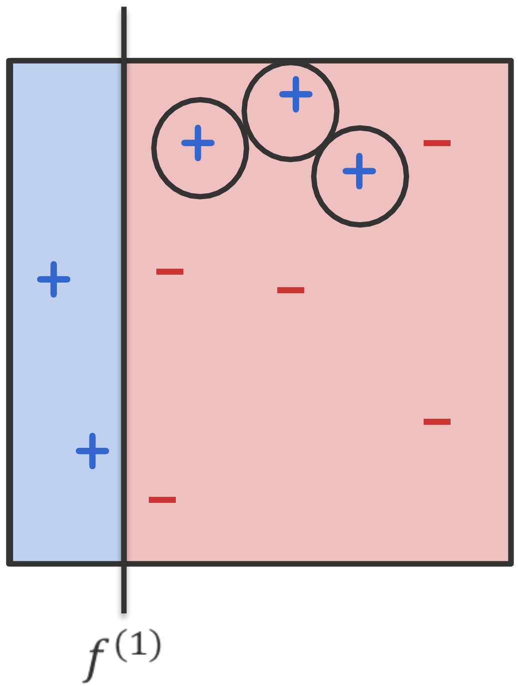
Our weak learner: An axis parallel line

Or

Initially all examples are equally important

$f^{(1)}$ = The best classifier on this data

# A toy example



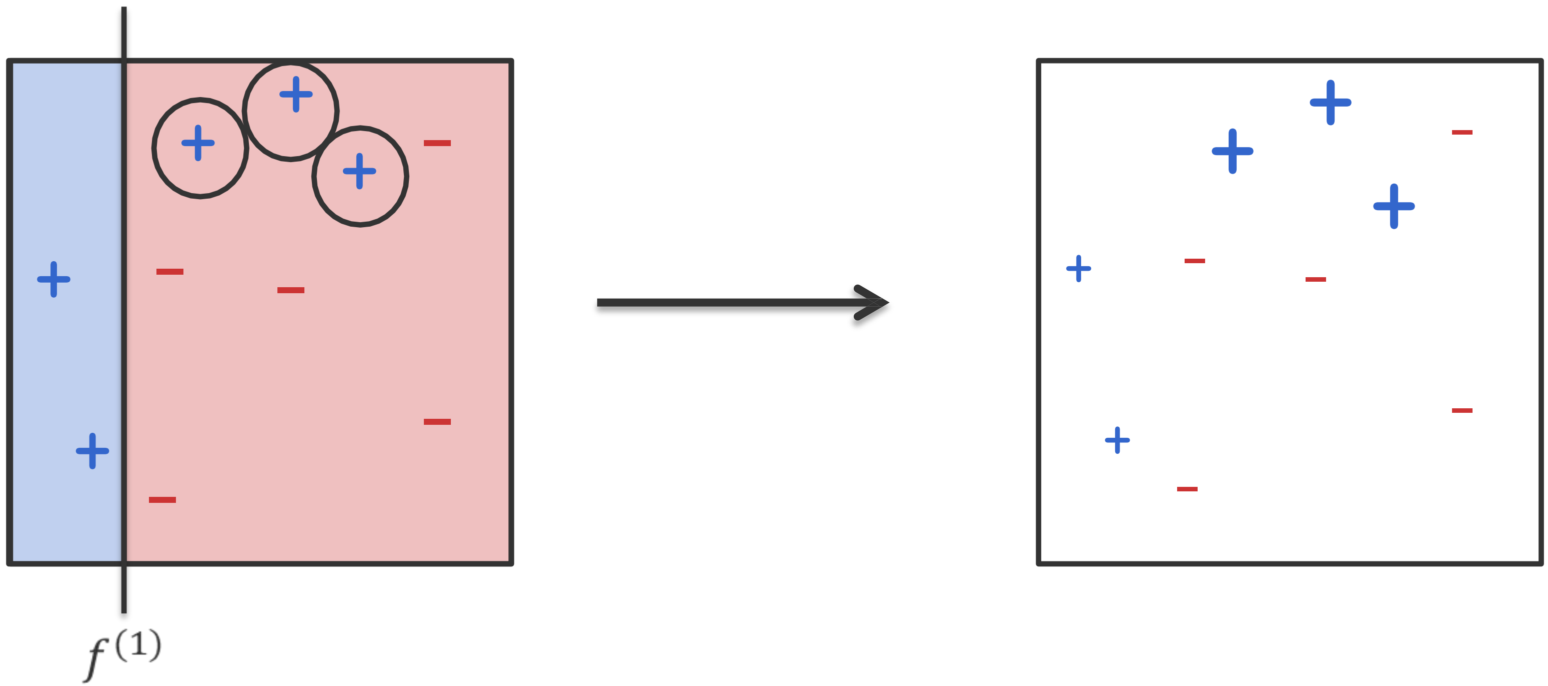

Our weak learner: An axis parallel line


Initially all examples are equally important

$f^{(1)}$ = The best classifier on this data

Clearly there are mistakes. Error=0.3

# A toy example



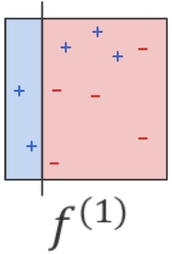

Initially all examples are equally important

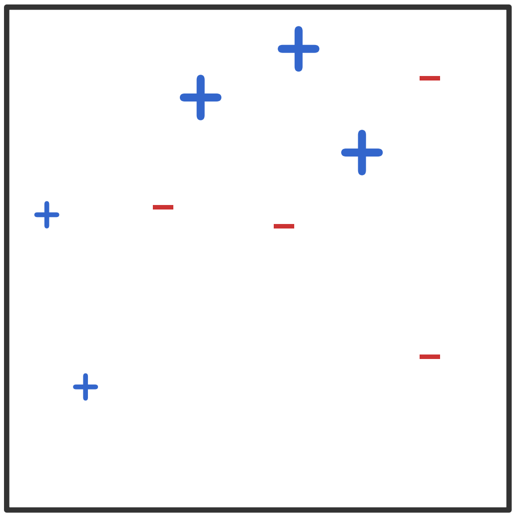
$f^{(1)}$ = The best classifier on this data

Clearly there are mistakes. Error=0.3

For the next round, increase the importance of the examples with mistakes and down-weight the examples that $f^{(1)}$ got correctly

# A toy example

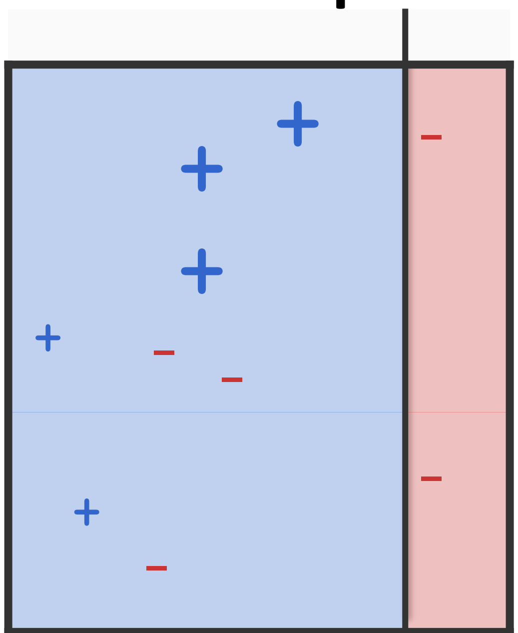


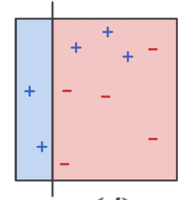



$\mathbf{w_i^{(j)}}$ = Set of weights at round j, one for each example **i**

**Motivation:** "How much should the weak learner care about this example in its choice of the classifier?"

# A toy example



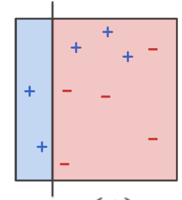


$f^{(j)}$

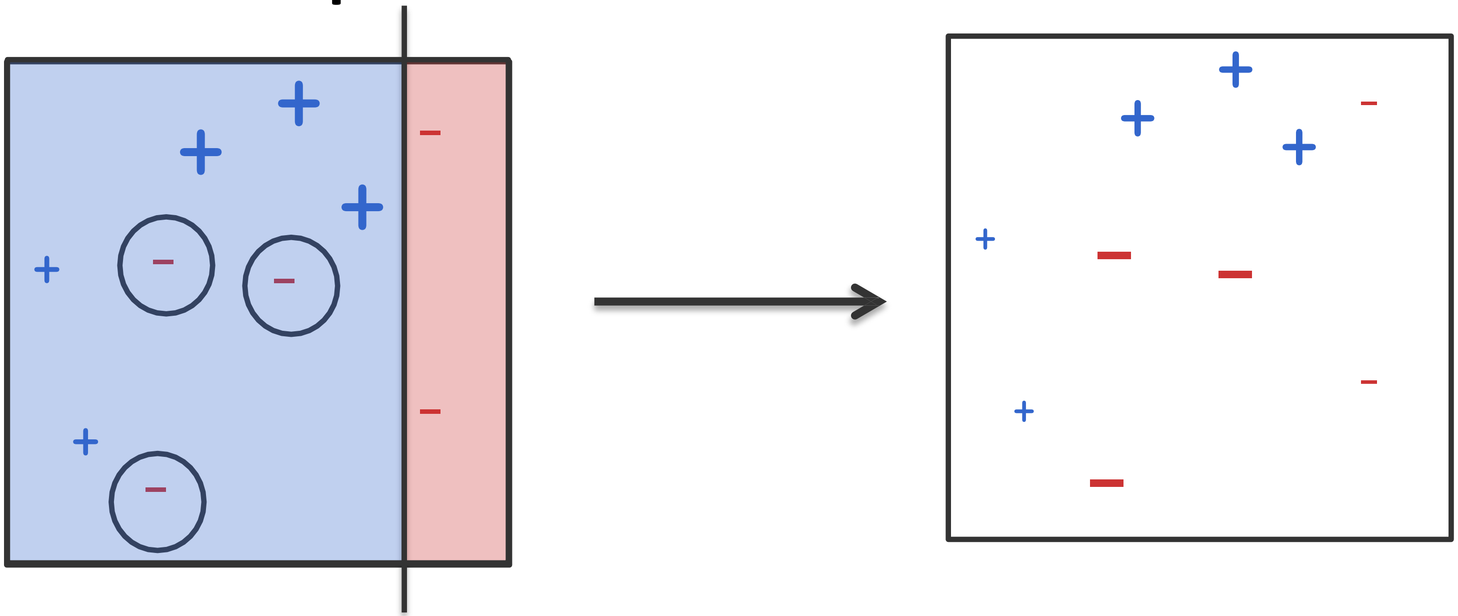
$\mathbf{w}_i^{(j)}$ = Set of weights at round j, one for each example **i**

**Motivation:** “How much should the weak learner care about this example in its choice of the classifier?”

# A toy example



$f^{(1)}$



$\mathbf{w}_{\mathbf{i}}^{\mathbf{(j)}}$ = Set of weights at round j, one for each example **i**

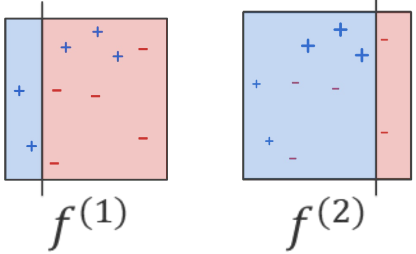
**Motivation:** "How much should the weak learner care about this example in its choice of the classifier?"

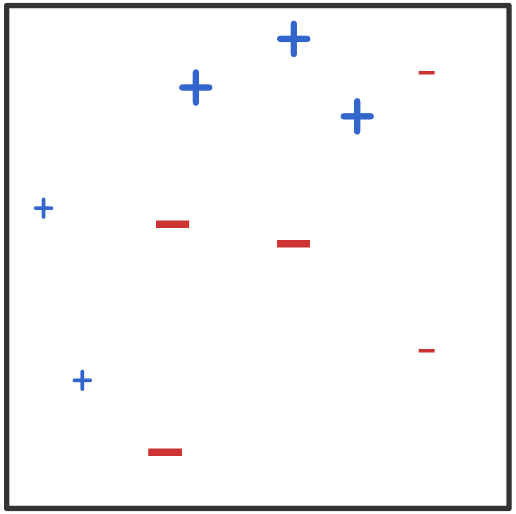
$f^{(2)}$ = A classifier learned on this data. *Has an error = 0.21*

Why not 0.3? Because while computing error, we will weight each example $x_i$ by its $w_i^{(j)}$

For the next round, increase the importance of the mistakes and down-weight the examples that $f^{(2)}$ got correctly

# A toy example

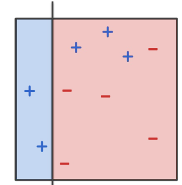


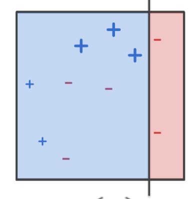
$\mathbf{w_i^{(j)}}$ = Set of weights at round j, one for each example **i**

**Motivation:** "How much should the weak learner care about this example in its choice of the classifier?"

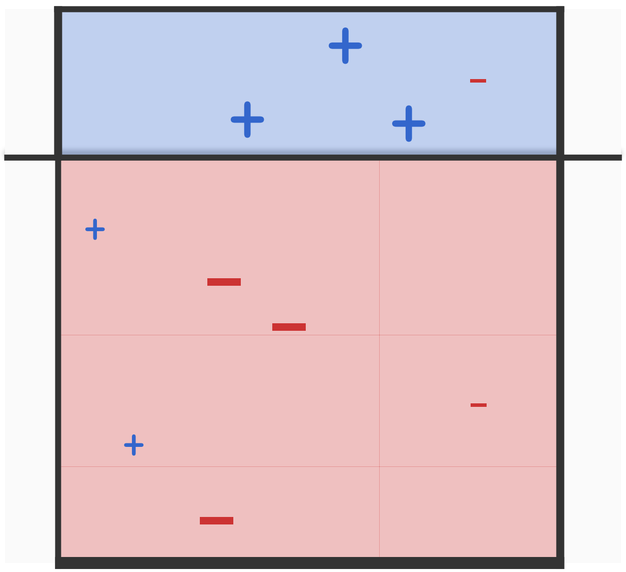
# A toy example



$f^{(1)}$



$f^{(2)}$



$\mathbf{w}_i^{(j)}$ = Set of weights at round j, one for each example **i**

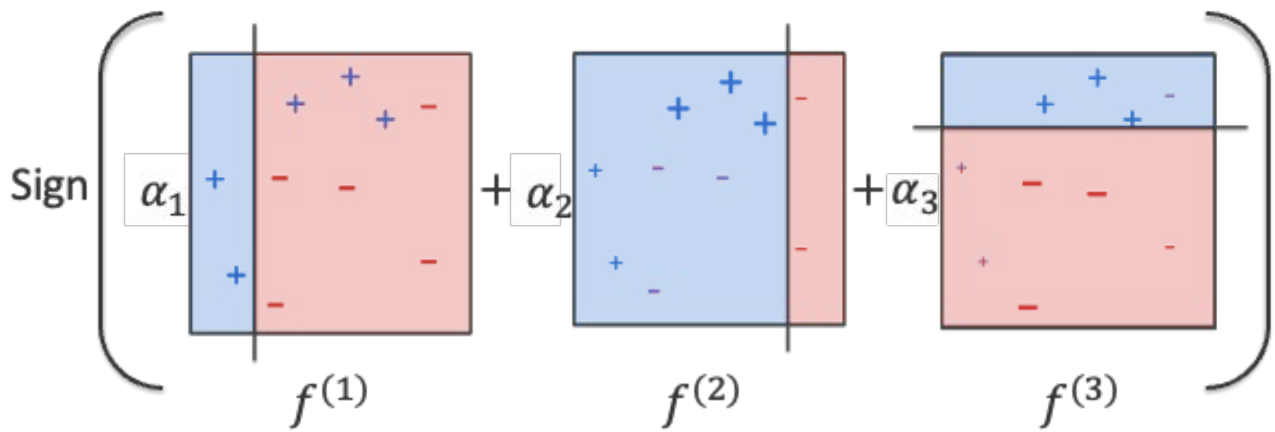
**Motivation:** "How much should the weak learner care about this example in its choice of the classifier?"

$f^{(3)}$ = A classifier learned on this data. *Has an error = 0.14*

# A toy example

The final predictor is a combination of all the $f$'s we have seen so far

$$F(x_i) =$$




> Think of the $\alpha$ values as the vote for each weak classifier and the boosting algorithm has to somehow specify them

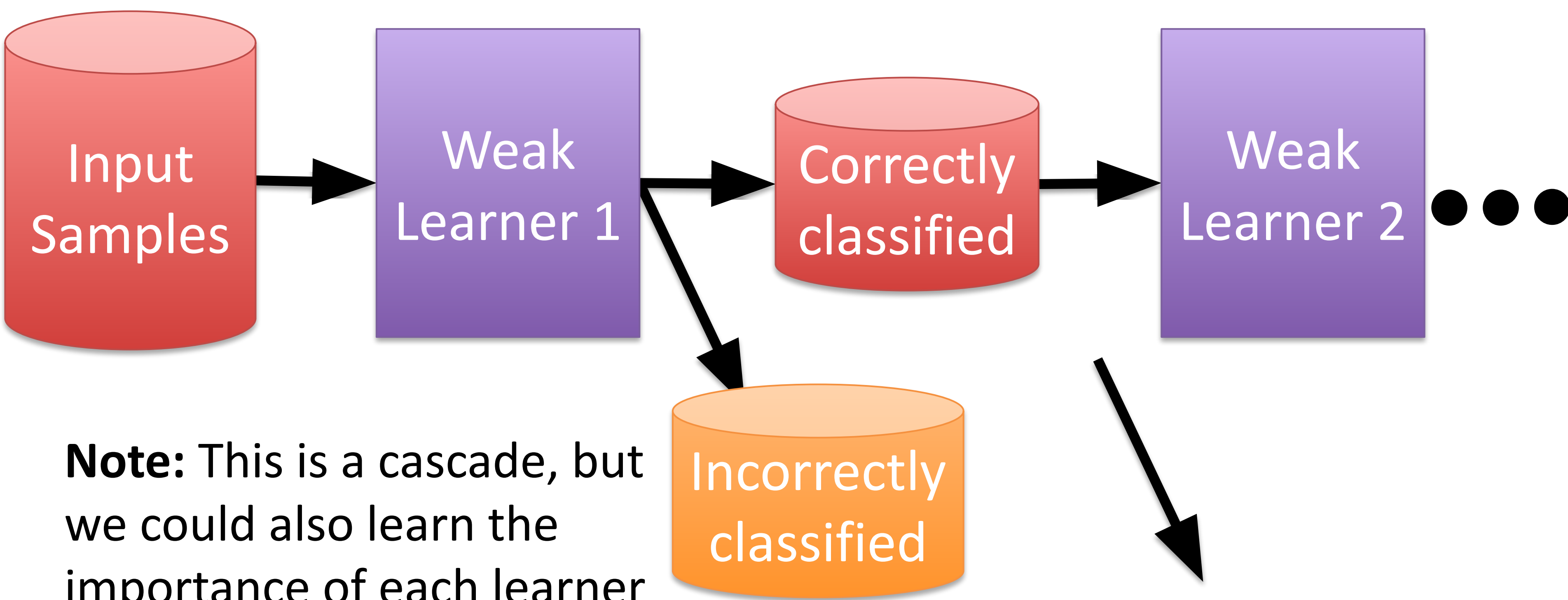
# An outline of Boosting

Given a training set $\mathcal{X}$

- For $j = 1, \ldots, J$
  - Construct a distribution $w_i^{(j)}$ on $\{1, 2, \cdots, N\}$
  - Find a weak hypothesis (rule of thumb) $f^{(j)}$. such that it has a small *weighted* error.
- Construct a final predictor $F(x_i)$ with weights $\alpha$ determined using ***line search*** (Forsyth Ch 12.2.2)

# Weak Learners

A set of simpler models that are combined to build a strong predictor




**Note:** This is a cascade, but we could also learn the importance of each learner

# Weak Learners

A set of simpler models that are combined to build a strong predictor

**Goal:** Each new weak learner improves on the sum of all previous weak learners

- Scales to large datasets
- Good performance
- Efficient

# Cascaded weak learners

- **Low individual performance:** A single weak learner on its own doesn't achieve high prediction accuracy.
- **Sequential combination:**
  - Iteratively train weak learners
  - Each new learner focuses on correcting the errors made by the previous ones.

# Gradient Boost

We choose a predictor $F$ that minimizes a loss

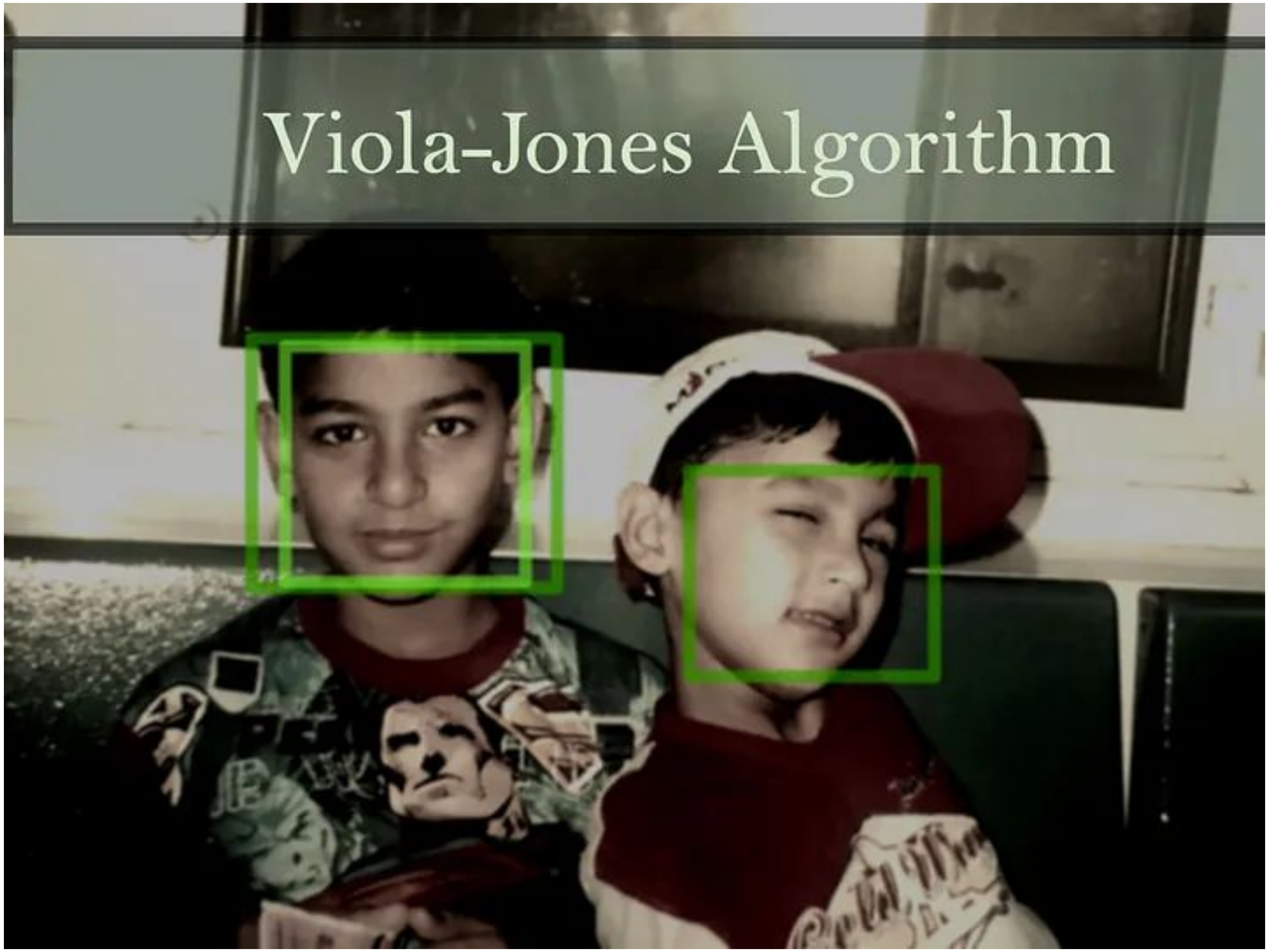
$$\mathcal{L}(F) = \frac{1}{N}\sum_{j} l(y_i, x_i, F)$$

We accomplish this by iteratively searching for a predictor of the form:

$$F(x;\theta) = \sum_{j} \alpha_j f\left(x;\theta^{(u)}\right)$$

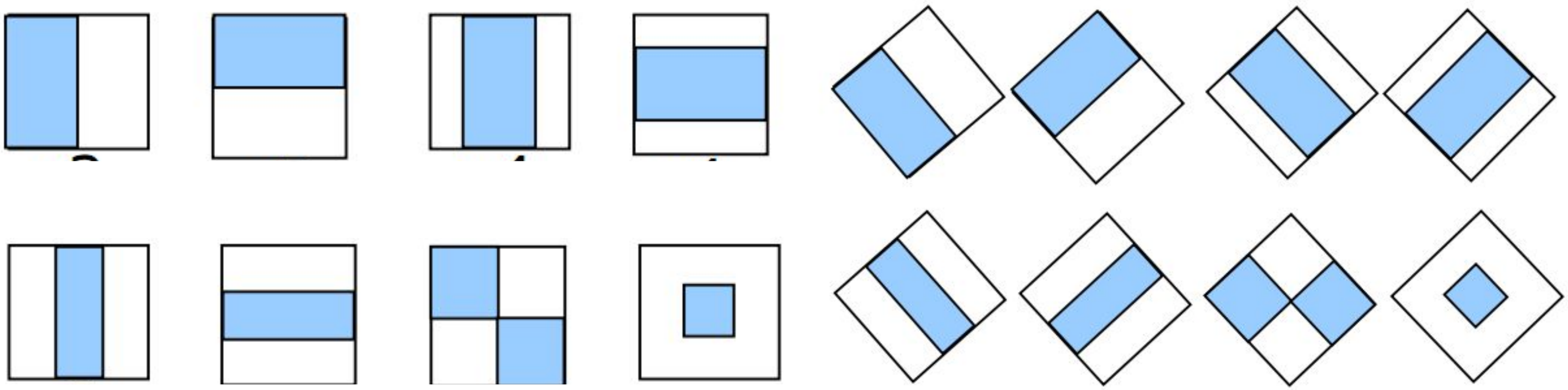
where $\theta$ is model parameters and $\alpha$ is a scaling factor for each weak leaner

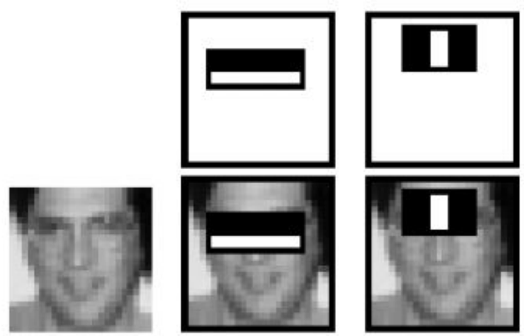
# Most famous example




P. Viola, M. Jones, ["Rapid object detection using a boosted cascade of simple features"](). CVPR, 2001

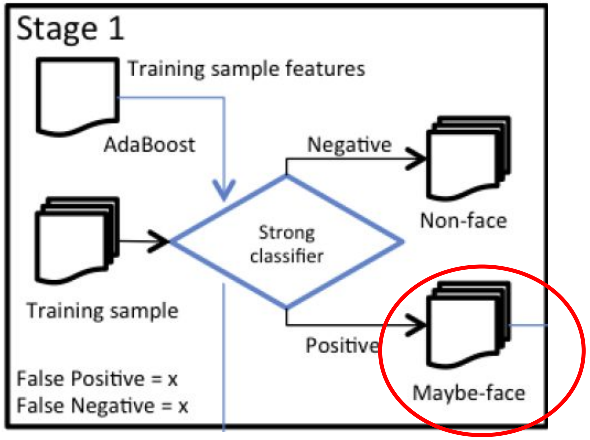
# Convolutional filters per classifier



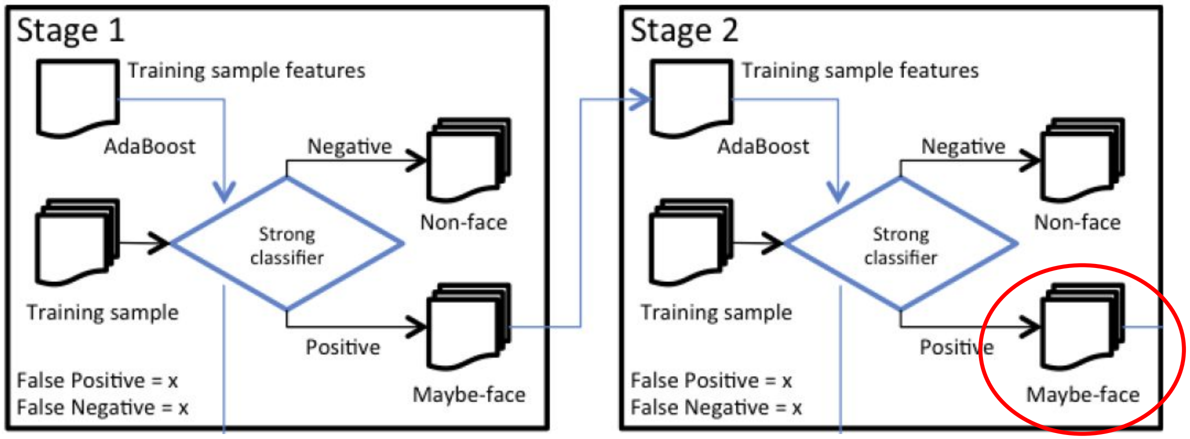


$$f(x, y) = \sum_i p_b(i) - \sum_i p_w(i)$$

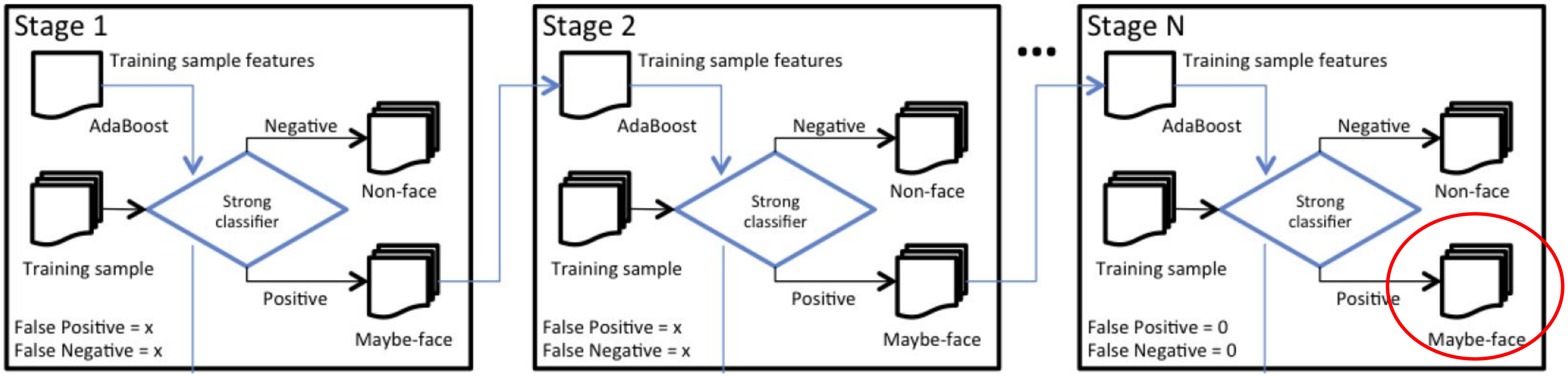
# Viola-Jones Face Detector



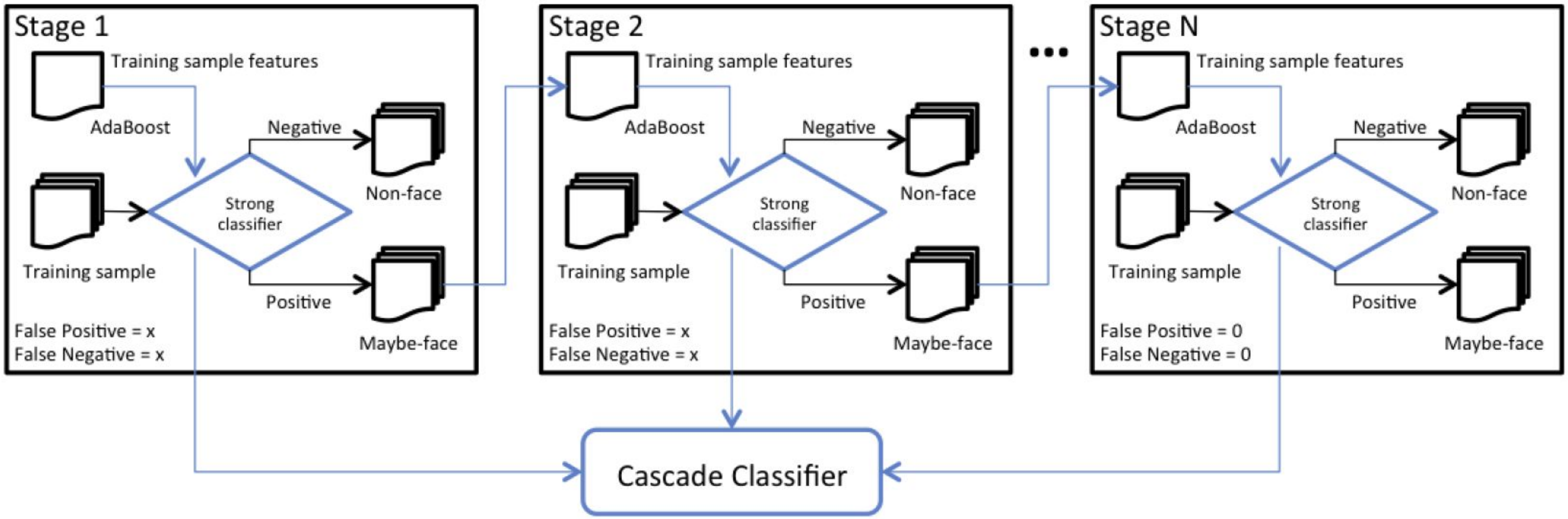
# Viola-Jones Face Detector



# Viola-Jones Face Detector



# Viola-Jones Face Detector




P. Viola, M. Jones, "Rapid object detection using a boosted cascade of simple features". CVPR, 2001

# Which of the following hold true for cascaded classifiers? Select all that apply

**Which of the following hold true for cascaded classifiers? Select all that apply**

Helps identify and reject negative samples quickly
94%

During inference, cascade learners offer more interpretability.
55%

Runtime complexity is significantly high because of the cascaded structure
28%

It is very important to have a highly accurate learner in each stage
53%