Recall from last time ...

Long Short Term Memory (LSTM)

Vanilla RNN

$$h_t = \tanh\left(W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}\right)$$

LSTM

$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

$$c_t = f \odot c_{t-1} + i \odot g$$

$$h_t = o \odot \tanh(c_t)$$

Hochreiter and Schmidhuber, "Long Short Term Memory", Neural Computation, 1997

Long Short Term Memory (LSTM)

LSTM

$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

$$c_t = f \odot c_{t-1} + i \odot g$$

$$h_t = o \odot \tanh(c_t)$$

i : input gate, whether to write to cell

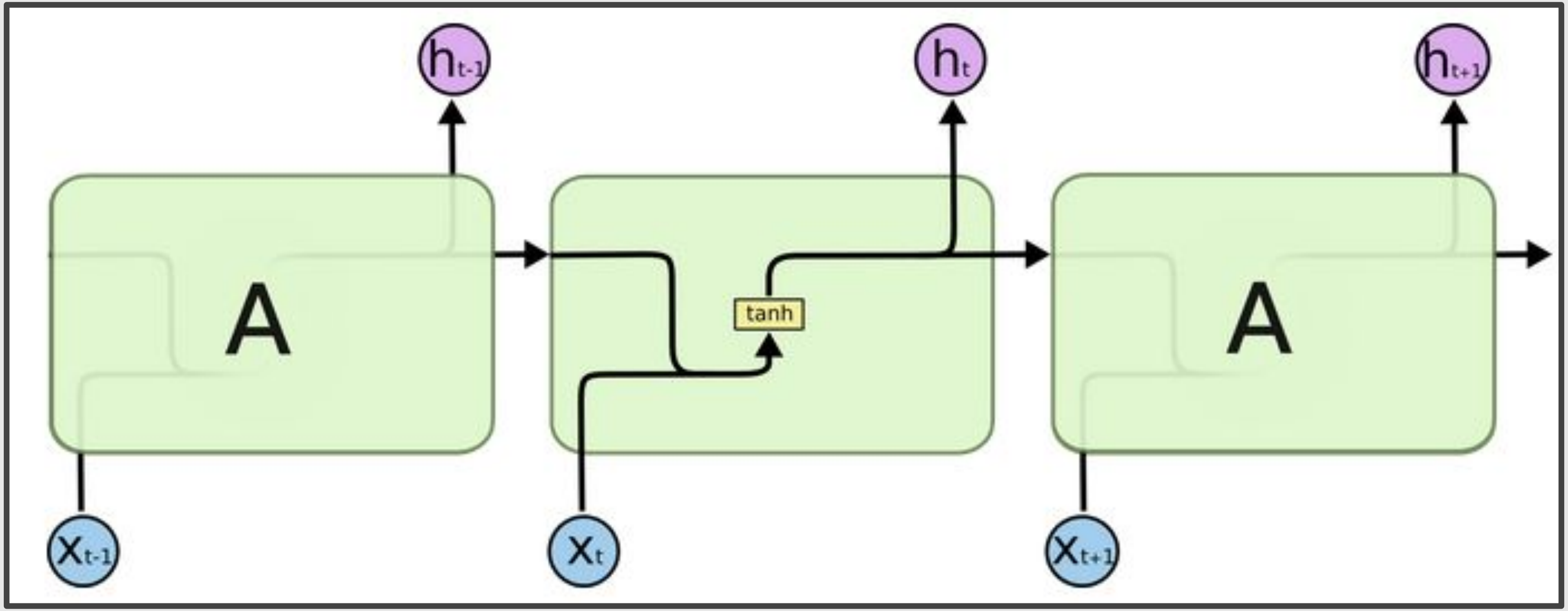
f : forget gate, whether to erase cell

o : output gate, how much to reveal cell

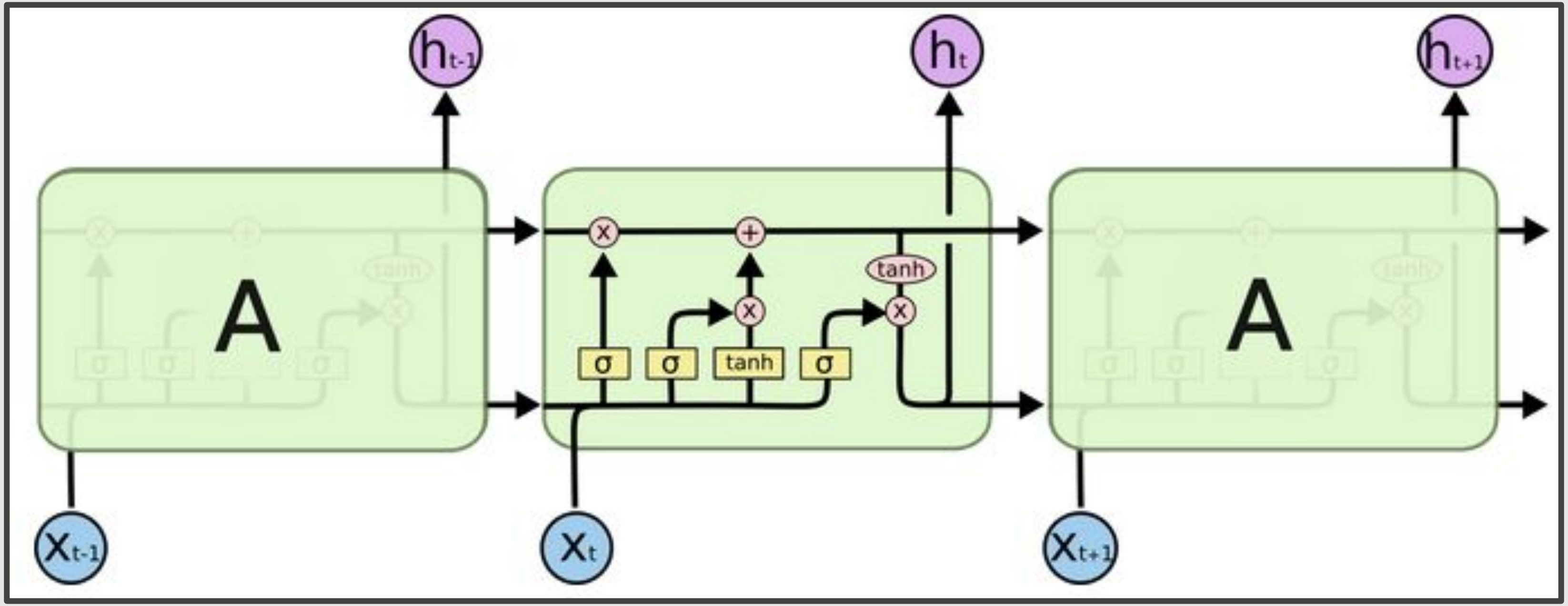
g : gate gate, how much to write to cell

Hochreiter and Schmidhuber, "Long Short Term Memory", Neural Computation, 1997

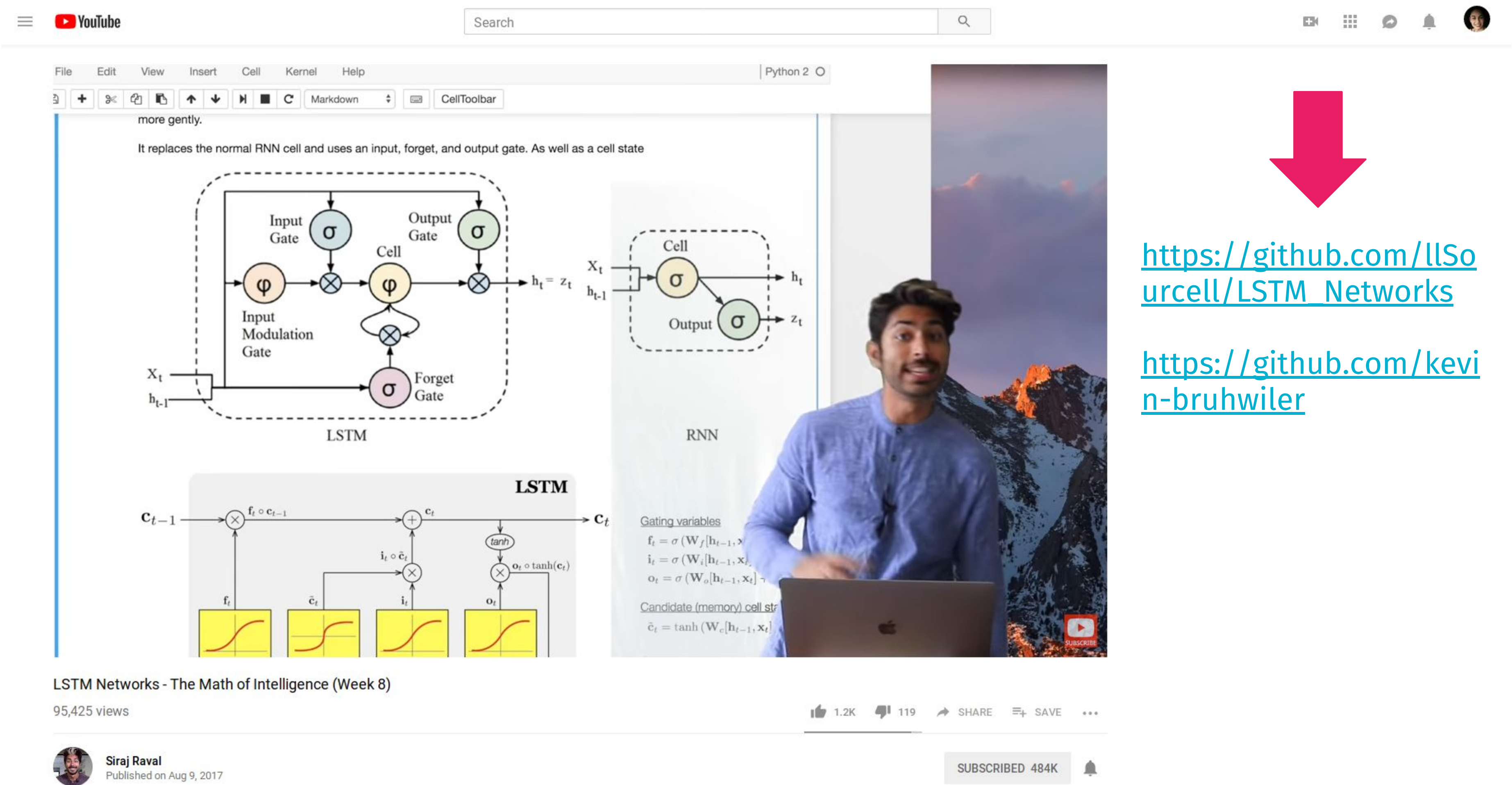
Long Short Term Memory (LSTM)



Long Short Term Memory (LSTM)



LSTM Networks: https://youtu.be/9zhrxE5PQgY



https://github.com/llSourcell/LSTM_Networks

https://github.com/kevin-bruhwiler

https://towardsdatascience.com/the-fall-of-rnn-lstm-2d1594c74ce0

Eugenio Culurciello Follow
I dream and build new technology
Apr 13 · 9 min read

The fall of RNN / LSTM

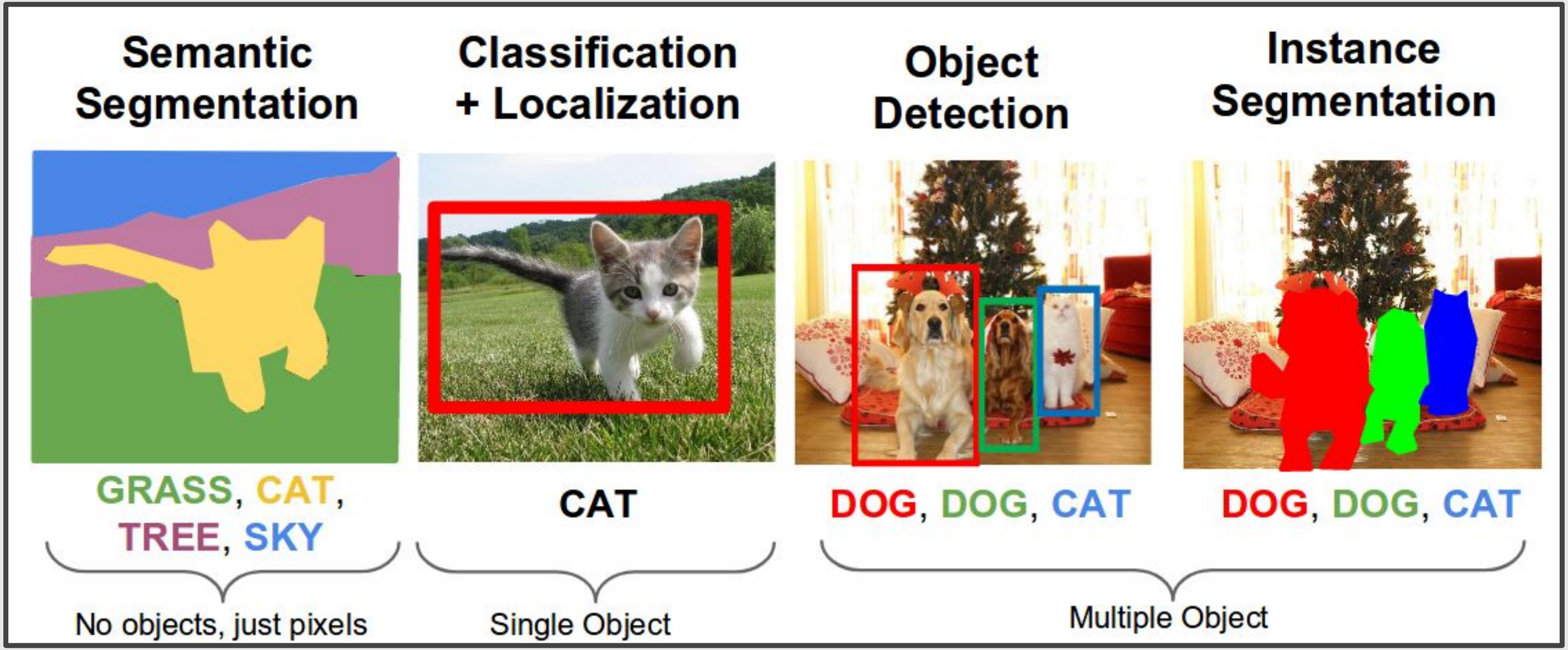
"Drop your RNN and LSTM, they are no good!"

See Notes!!!

We fell for Recurrent neural networks (RNN), Long-short term memory (LSTM), and all their variants. **Now it is time to drop them!**

Other Tasks …

Other Tasks …


Semantic Segmentation
Classification + Localization
Object Detection
Instance Segmentation
GRASS, CAT, TREE, SKY
CAT
DOG, DOG, CAT
DOG, DOG, CAT
No objects, just pixels
Single Object
Multiple Object

How to Intentionally Trick Neural Networks

https://medium.com/@ageitgey/machine-learning-is-fun-part-8-how-to-intentionally-trick-neural-networks-b55da32b7196

One Pixel Attack Defeats Neural Networks | Two Minute Papers #240

https://youtu.be/SA4YEAWVpbk

"One pixel attack for fooling deep neural networks"

https://arxiv.org/abs/1710.08864

This Fools Your Vision | Two Minute Papers #241

https://youtu.be/AbxPbfODGcs

"Adversarial examples that fool both human and computer vision"

https://arxiv.org/abs/1802.08195

References

Deep Learning Books

- Deep Learning, http://www.deeplearningbook.org/contents/rnn.html

Deep Learning Courses

- Recurrent Neural Networks - The Math of Intelligence (Week 5): https://youtu.be/BwmddtPFWtA
- LSTM Networks - The Math of Intelligence (Week 8): https://youtu.be/9zhrxE5PQgY
- Understanding LSTM Networks: http://colah.github.io/posts/2015-08-Understanding-LSTMs/
- https://www.coursera.org/learn/neural-network
- CS231n: Convolutional Neural Networks for Visual Recognition: http://cs231n.stanford.edu/
- "The 3 popular courses on Deep Learning": https://medium.com/towards-data-science/the-3-popular-courses-for-deeplearning-ai-ac37d4433bd

Ensemble Learning

Machine Learning and Pattern Recognition

Prof. Sandra Avila

Institute of Computing (IC/Unicamp)

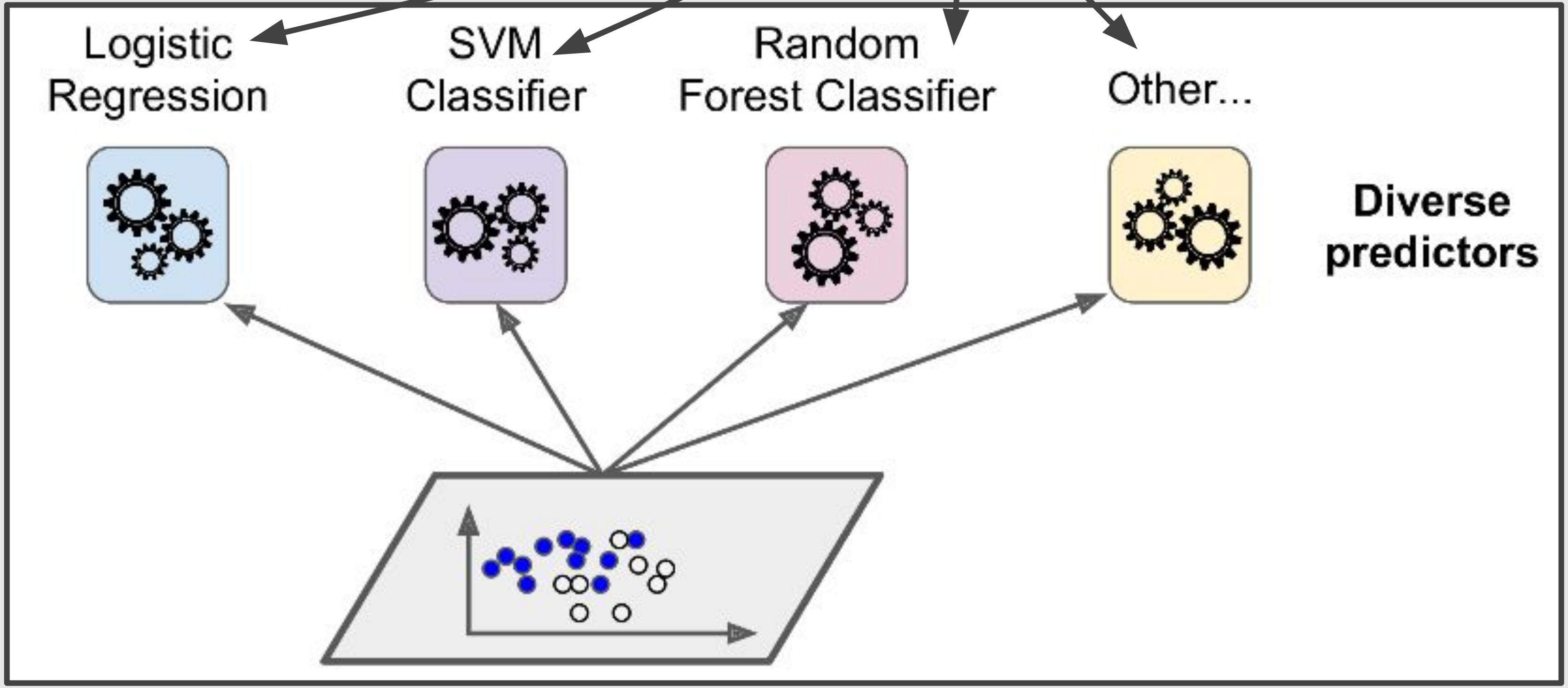
MC886/MO444, November 1, 2018

Ensemble Learning

- Multiple learning algorithms **to obtain better predictive performance** than could be obtained from any learning algorithms individually.

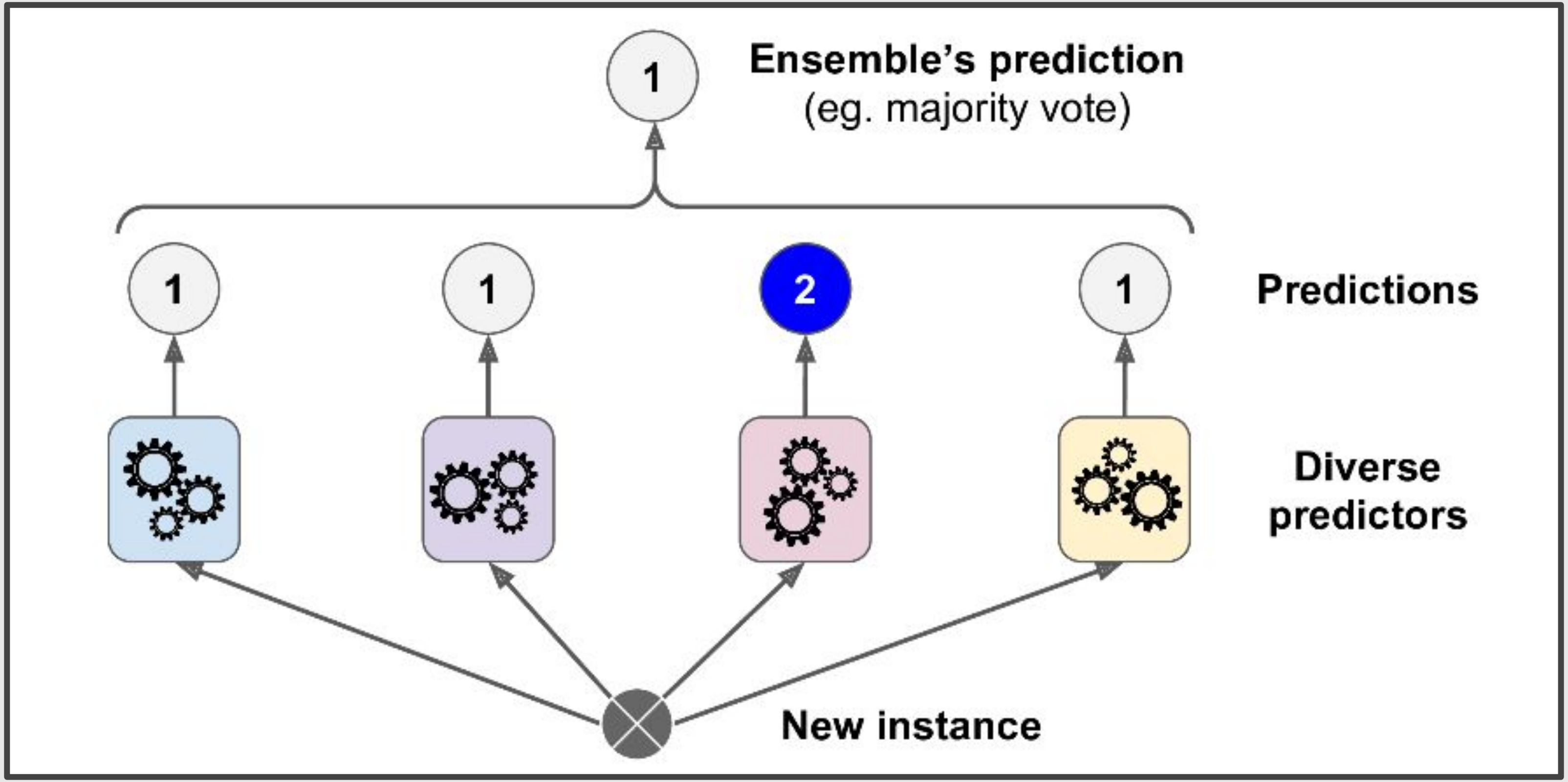
Voting Classifiers

~80%



Voting Classifiers

Hard/Soft voting classifier



Voting Classifiers

- Voting classifier **often achieves a higher accuracy** **than the best classifier** in the ensemble.

Voting Classifiers

- Voting classifier **often achieves a higher accuracy than the best classifier** in the ensemble.

- Even if each classifier is a **weak learner**, the ensemble can still be a **strong learner**, provided there are a sufficient number of weak learners and they are sufficiently diverse.

Voting Classifiers

- Ensemble methods work best when the predictors are as **independent** from one another as possible.

Voting Classifiers

- Ensemble methods work best when the predictors are as **independent** from one another as possible.

- One way to get diverse classifiers is to train them using **very different algorithms**: this increases the chance that they will make very different types of errors, improving the ensemble's accuracy.

Voting Classifiers

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import VotingClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
log_clf = LogisticRegression()
rnd_clf = RandomForestClassifier()
svm_clf = SVC()
voting_clf = VotingClassifier(
        estimators=[('lr', log_clf), ('rf', rnd_clf), ('svc', svm_clf)],
                    voting='hard'
    )
voting_clf.fit(X_train, y_train)
```

Ensemble Learning

- Types: Bagging (and Pasting), Boosting, and Stacking

Bagging & Pasting

Bagging and Pasting

- Use **the same training algorithm** for every predictor, but to train them on different random subsets of the training set.

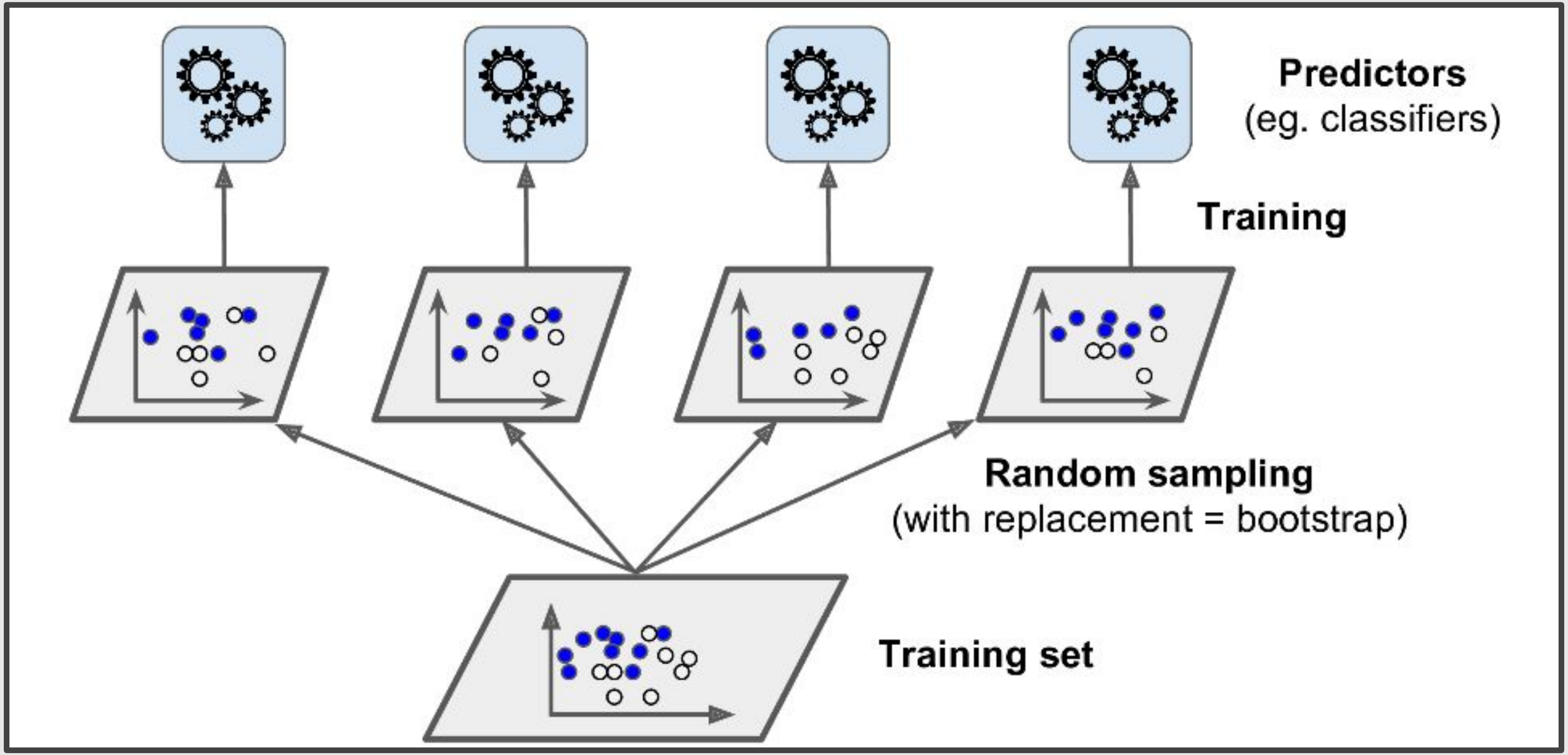
“Bagging Predictors”, L. Breiman (1996): http://goo.gl/o42tml

Bagging and Pasting

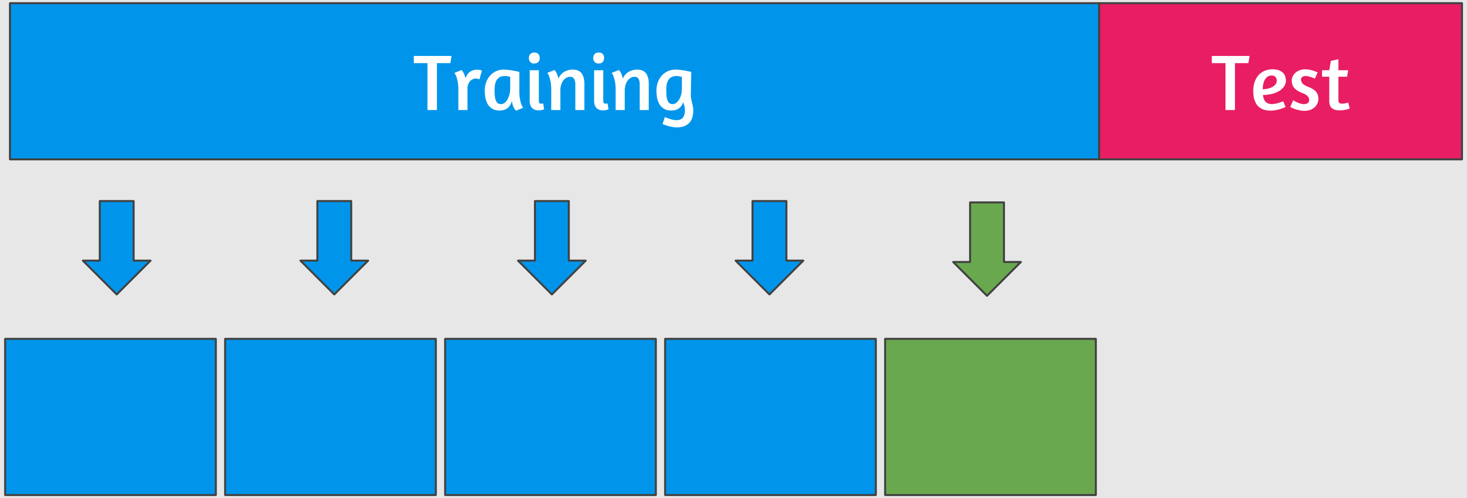
- Use **the same training algorithm** for every predictor, but to train them on different random subsets of the training set.
- **Bagging** (short for Bootstrap Aggregating): sampling is performed **with** replacement.
- **Pasting**: sampling is performed **without** replacement.

“Bagging Predictors”, L. Breiman (1996): http://goo.gl/o42tml

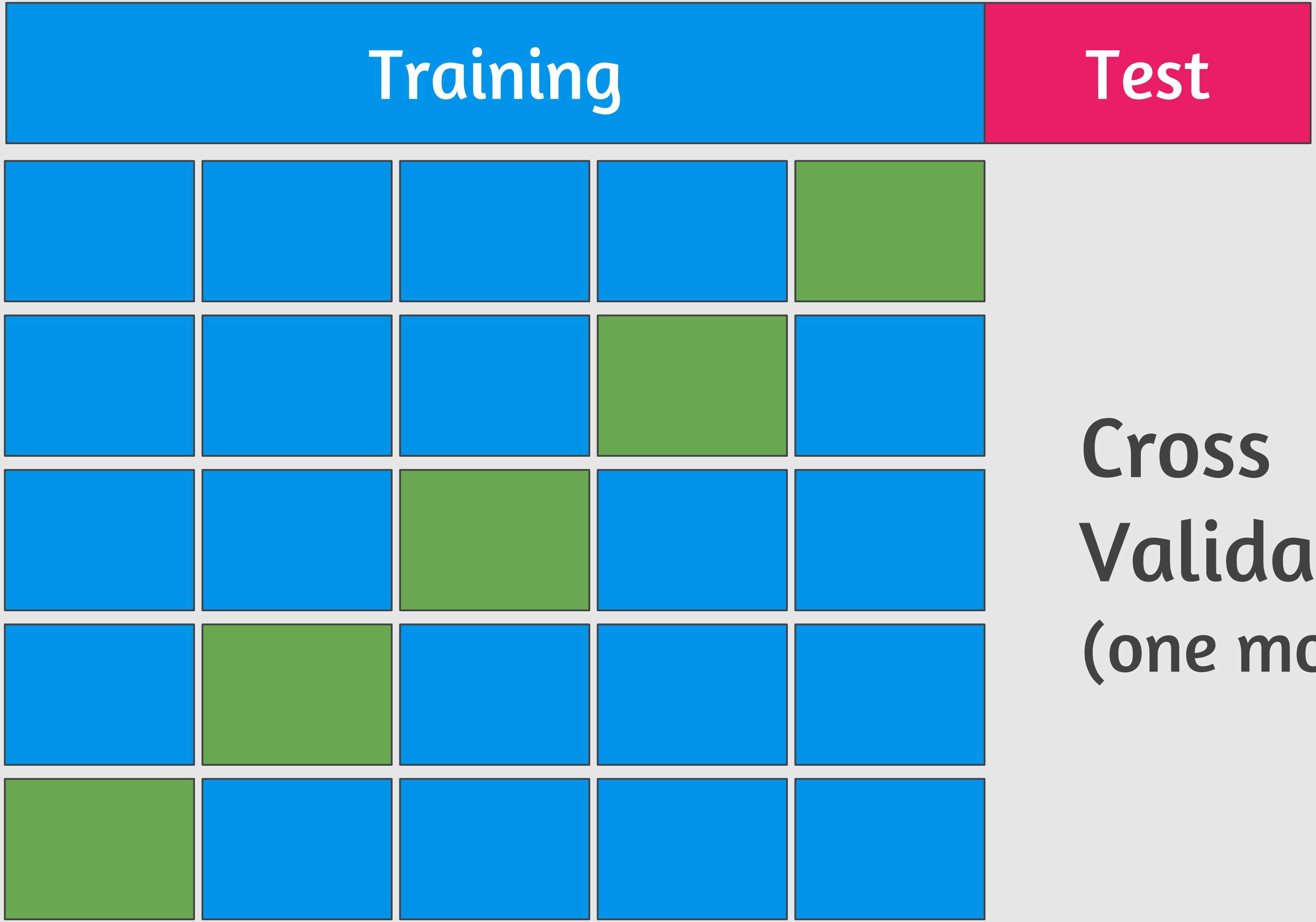
Bagging and Pasting



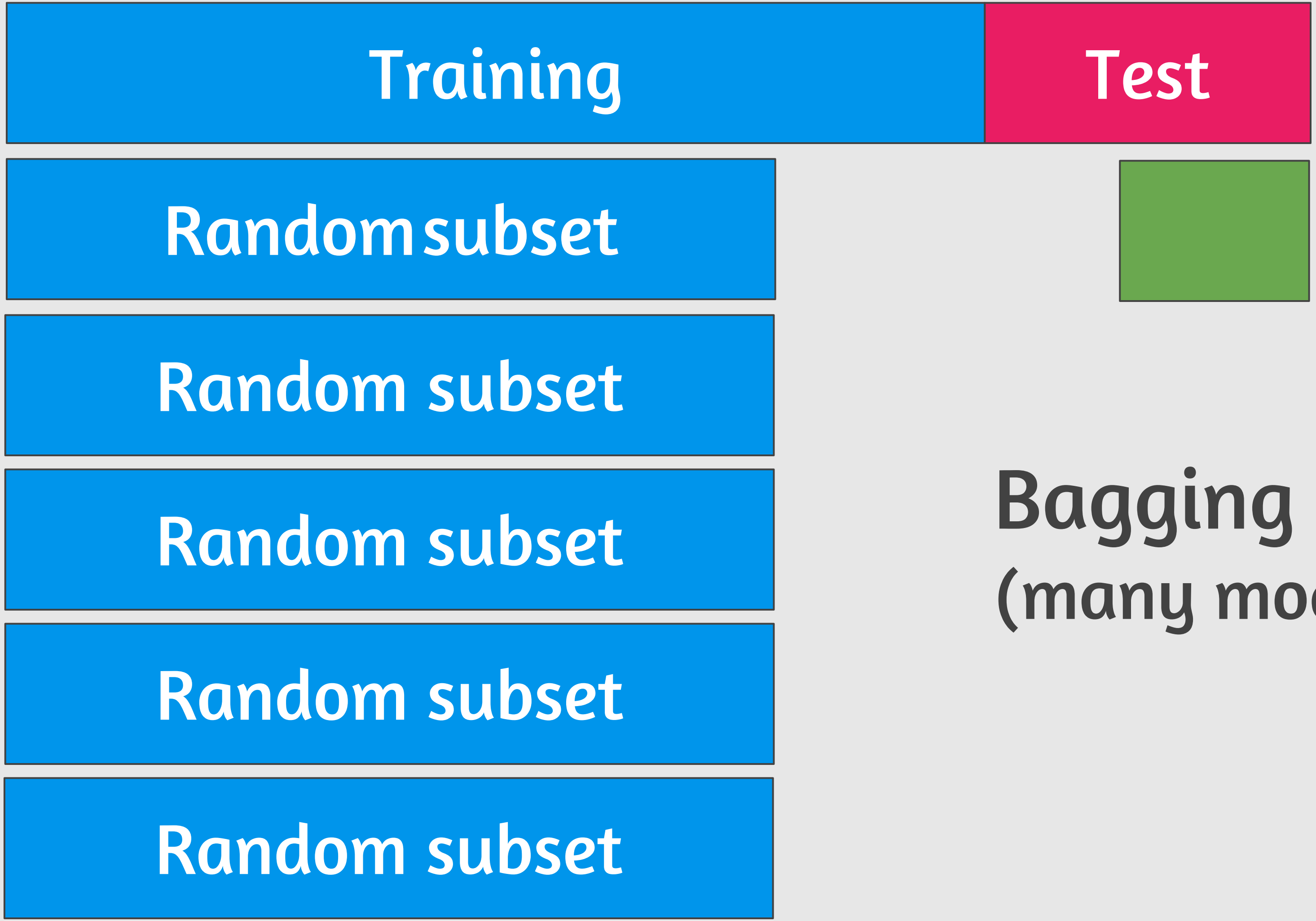
Bagging vs. Cross Validation



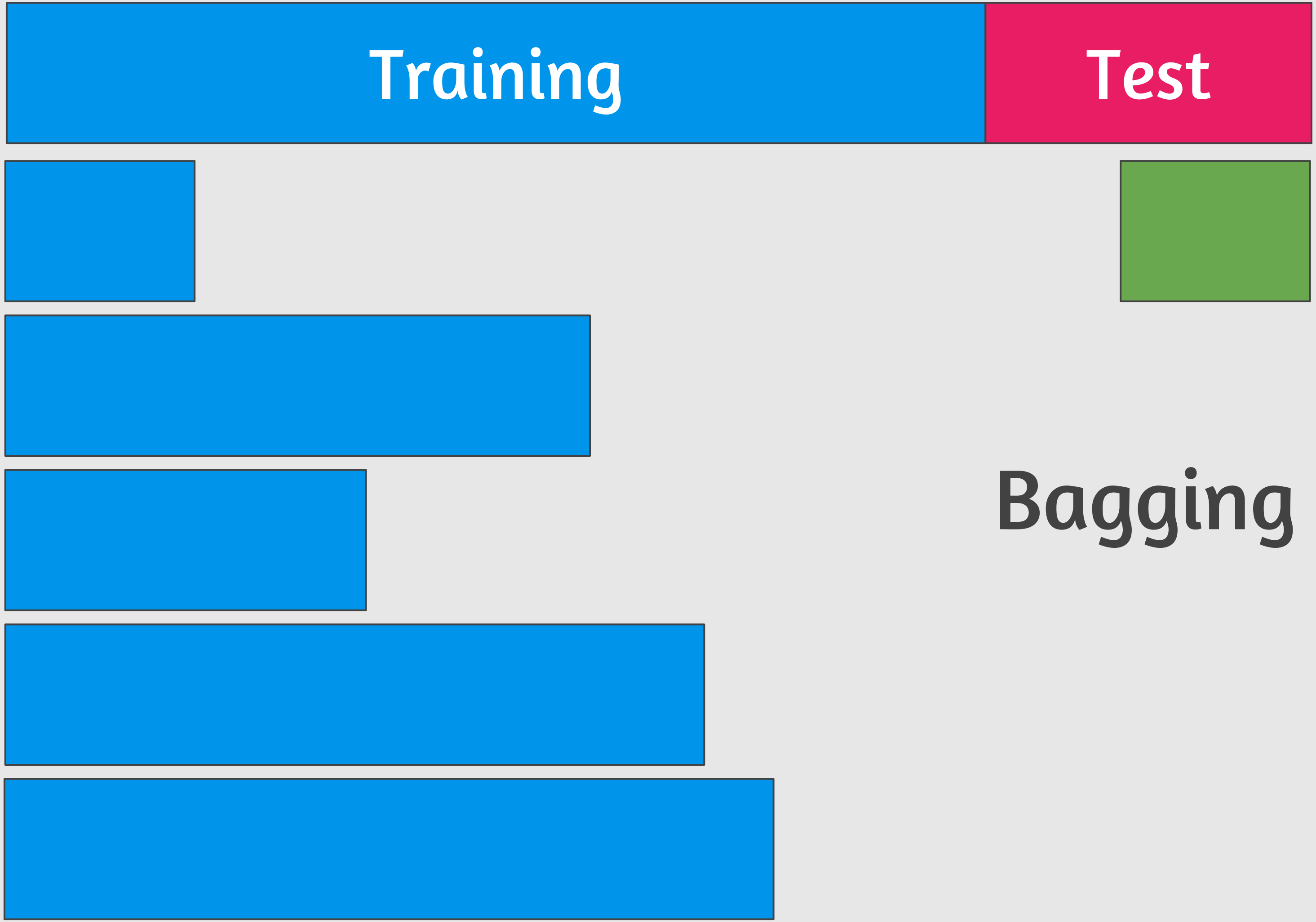
Cross Validation



Cross Validation
(one model)



Bagging
(many models)



Bagging

Bagging and Pasting

- Once all predictors are trained, the ensemble can make a prediction for a new instance by simply aggregating the predictions of all predictors.

- **Bagging and Pasting scale very well.**

Bagging and Pasting

```
from sklearn.ensemble import BaggingClassifier
from sklearn.tree import DecisionTreeClassifier
bag_clf = BaggingClassifier(
        DecisionTreeClassifier(), n_estimators=500,
        max_samples=100, bootstrap=True, n_jobs=-1
    )
bag_clf.fit(X_train, y_train)
y_pred = bag_clf.predict(X_test)
```

Bagging and Pasting

- **Random Patches Ensemble** method: sampling **both** training instances and features.
- This is particularly useful when dealing with high-dimensional inputs.

“Ensembles on Random Patches”, G. Louppe and P. Geurts (2012): http://goo.gl/B2EcM2

Boosting

Boosting

- The general idea of most boosting methods is **to train predictors sequentially**, each trying to correct its predecessor.

Boosting

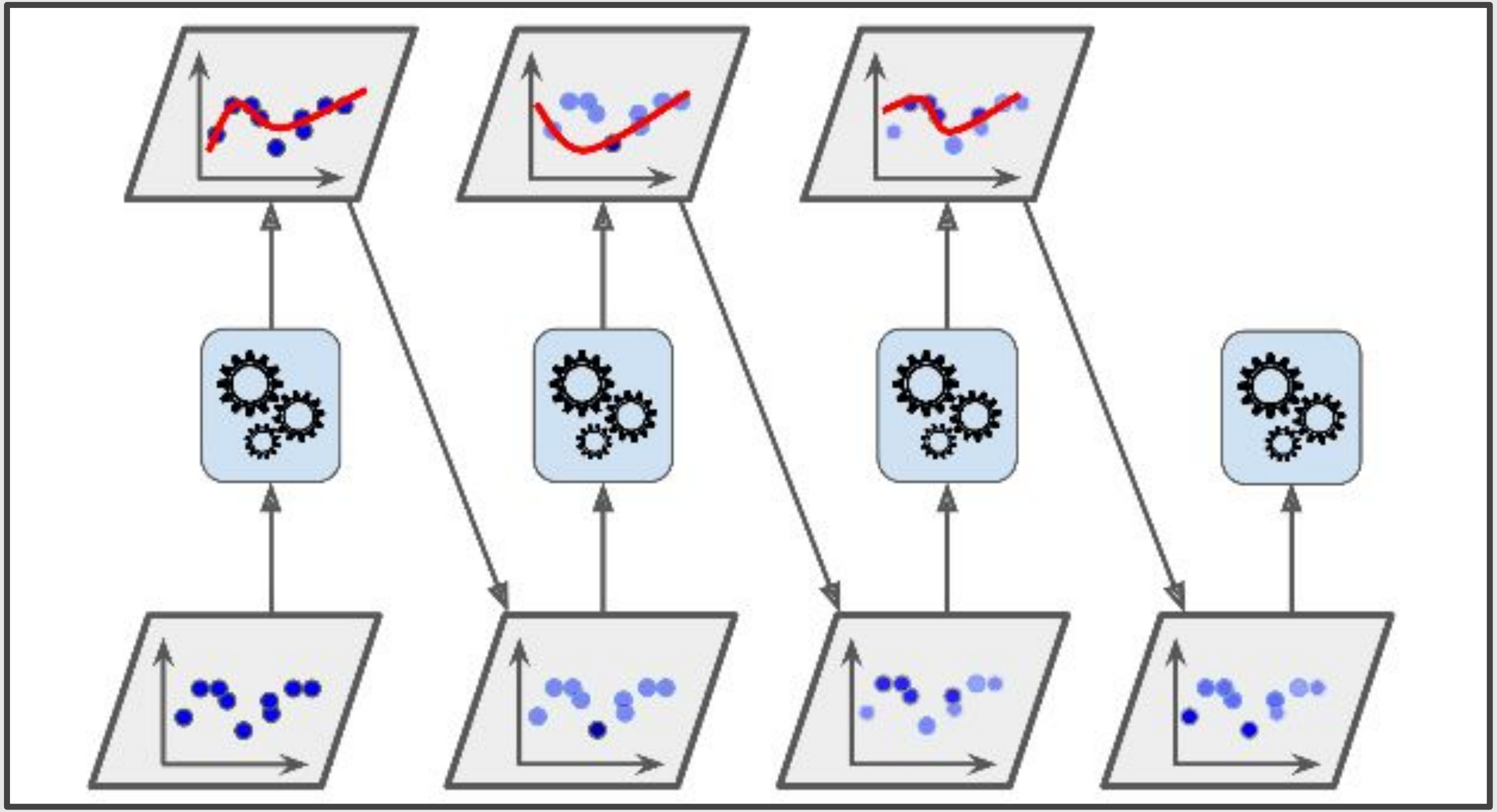
- The general idea of most boosting methods is **to train predictors sequentially**, each trying to correct its predecessor.
- Most popular: AdaBoost and Gradient Boost.

AdaBoost [Freund and Schapire, 1997]

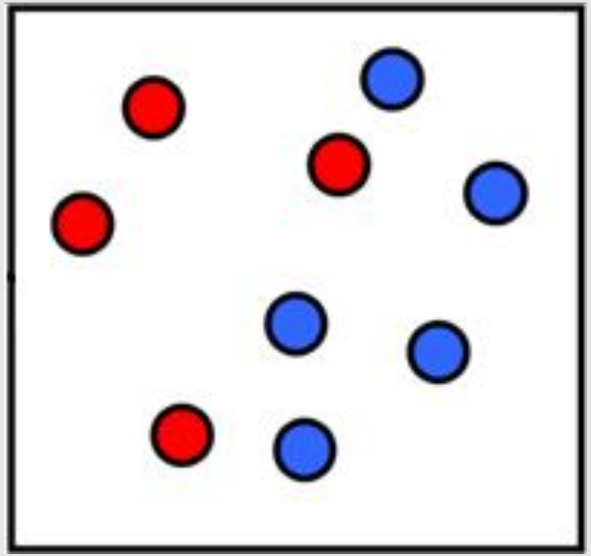
- One way for a new predictor to correct its predecessor is to pay a bit **more attention** to the training instances that **the predecessor underfitted**.

“A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting”, Y. Freund and R. Schapire (1997): http://goo.gl/OIduRW

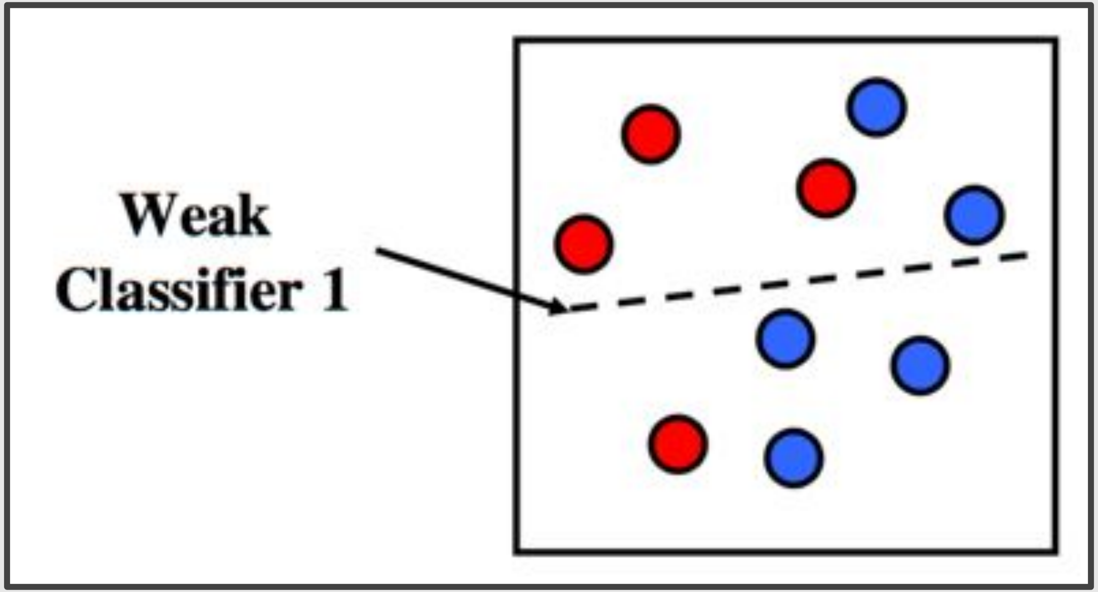
AdaBoost



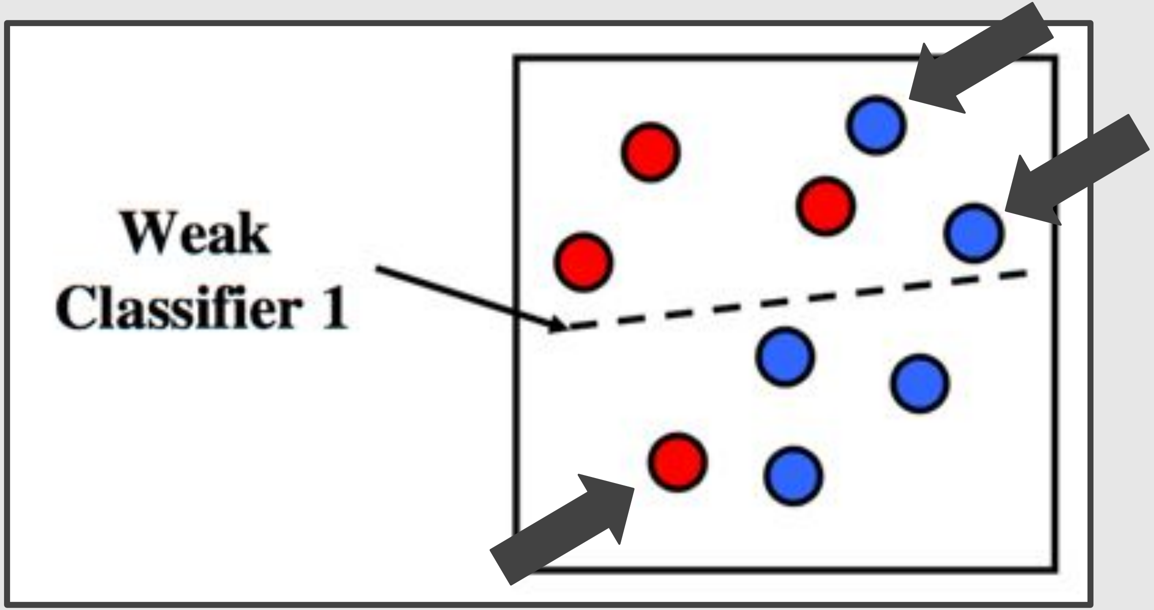
AdaBoost



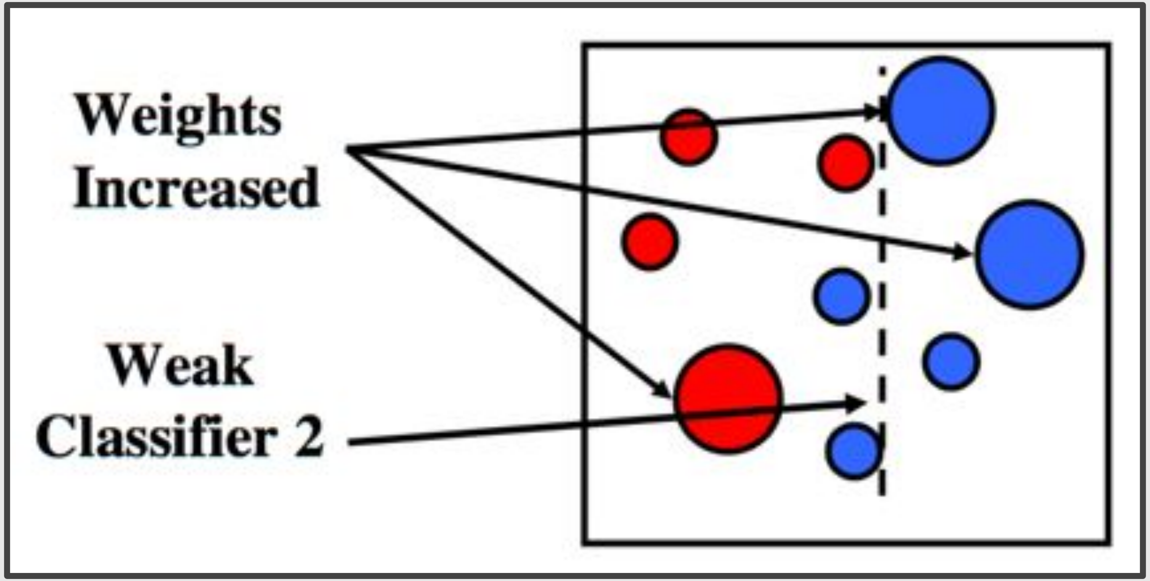
AdaBoost



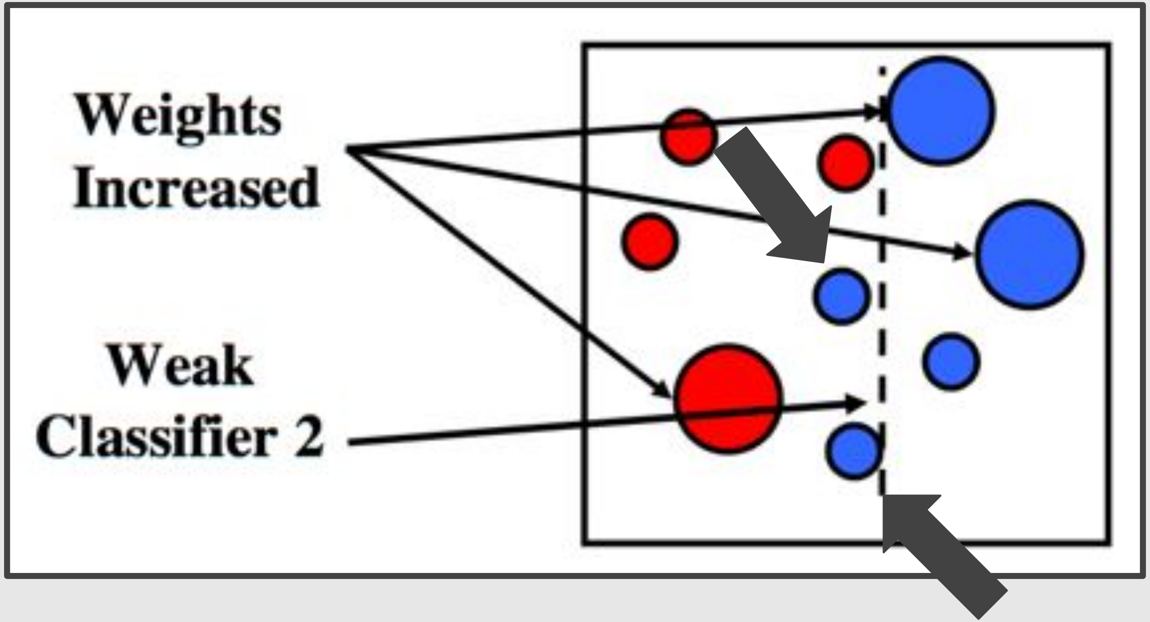
AdaBoost



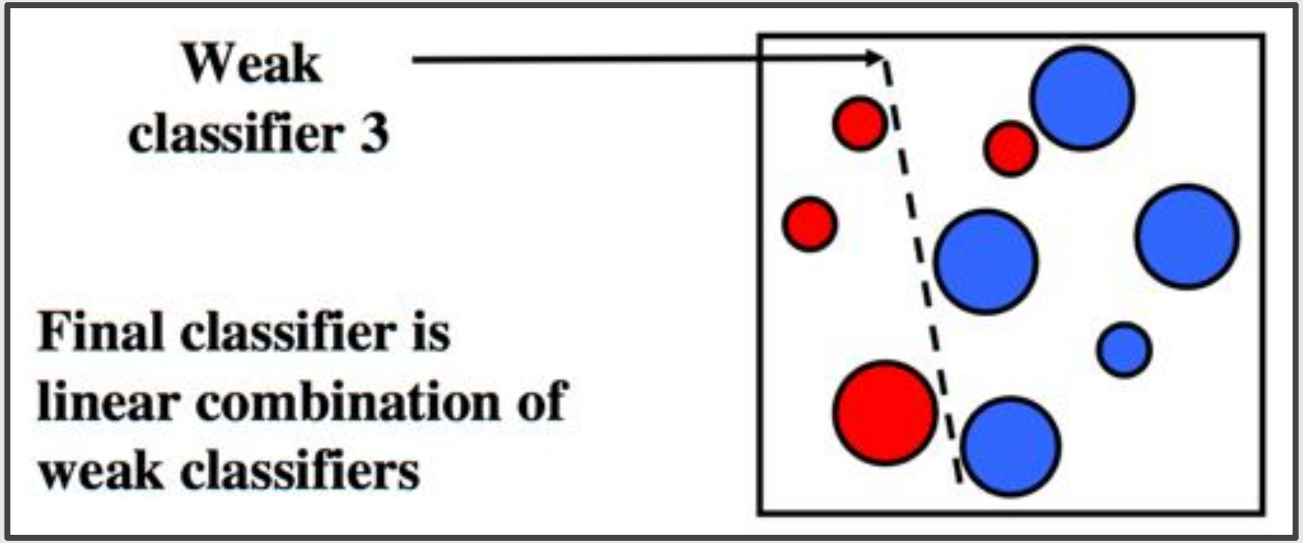
AdaBoost



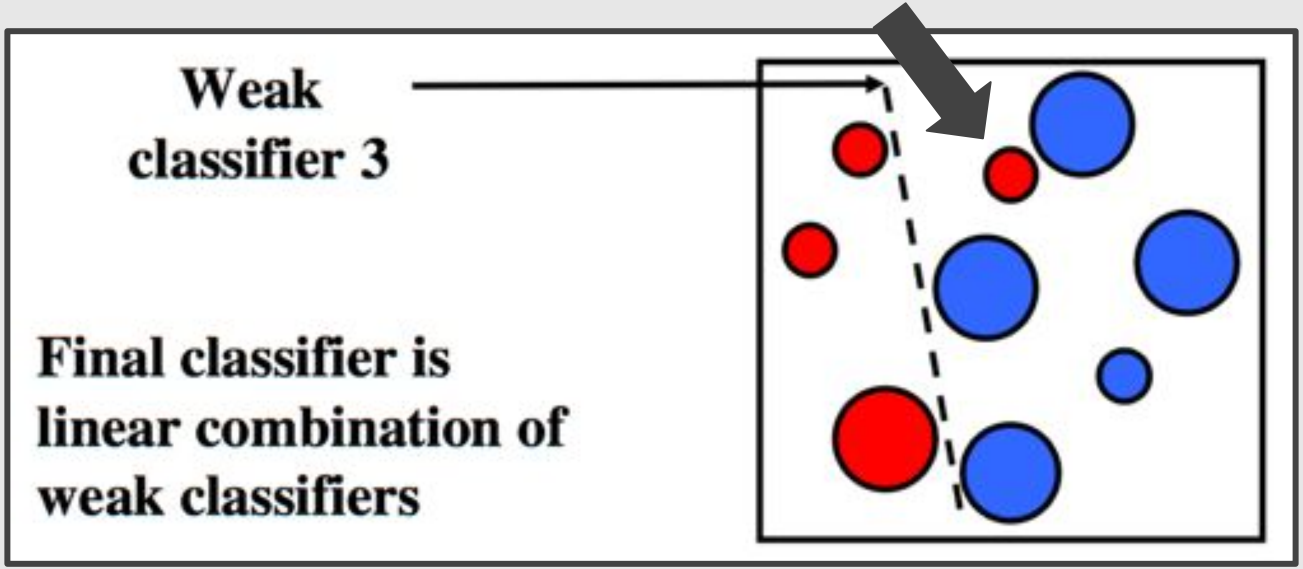
AdaBoost



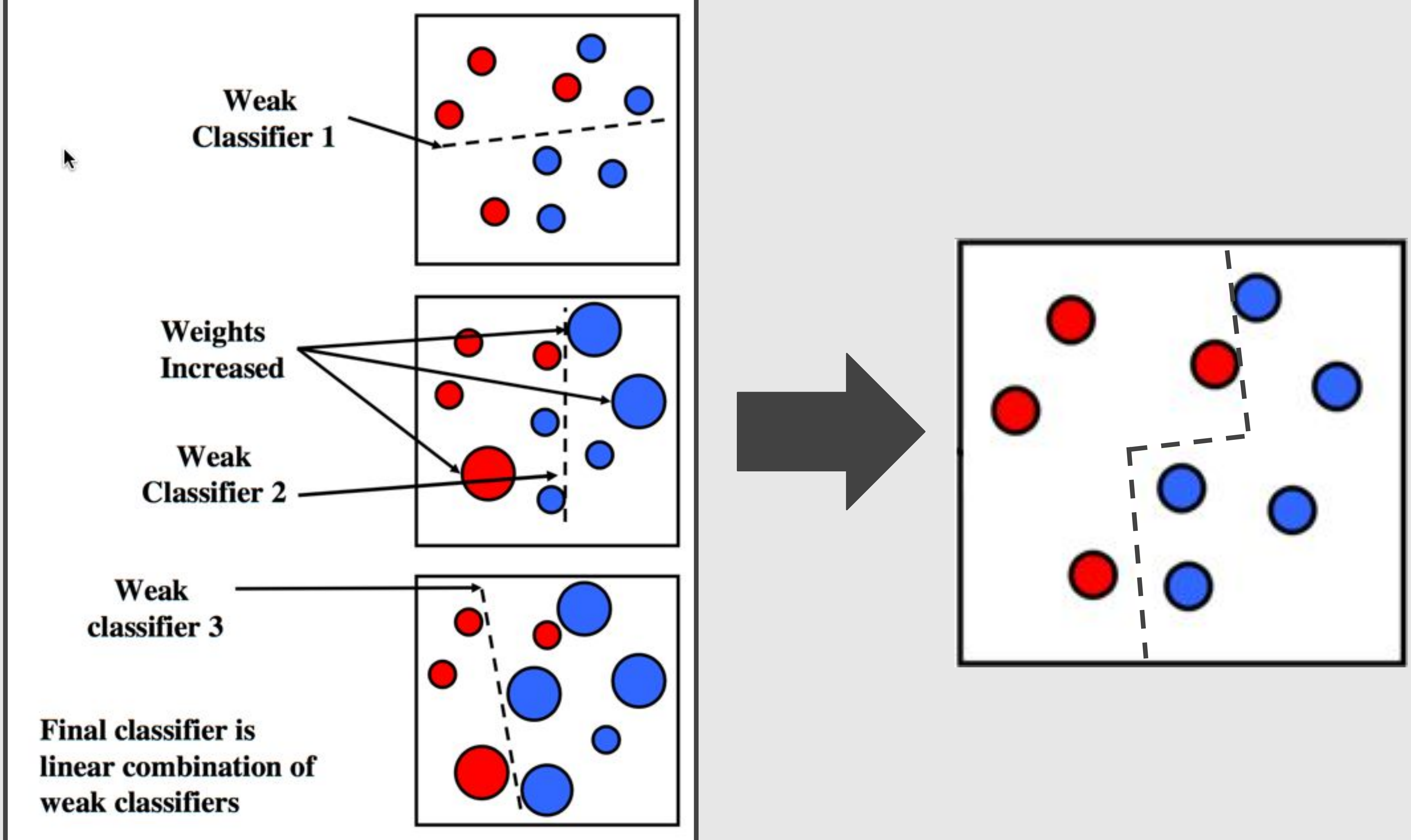
AdaBoost


Weak
classifier 3
Final classifier is
linear combination of
weak classifiers

AdaBoost



Weak
classifier 3
Final classifier is
linear combination of
weak classifiers


Weak Classifier 1
Weights Increased
Weak Classifier 2
Weak classifier 3
Final classifier is linear combination of weak classifiers

AdaBoost

1. Assign every observation, x_i, an initial weight value, $w_i = \frac{1}{n}$, where n is the total number of observations.
2. Train a "weak" model. (most often a decision tree)
3. For each observation:
    3.1. If predicted incorrectly, w_i is increased
    3.2. If predicted correctly, w_i is decreased
4. Train a new weak model where observations with greater weights are given more priority.
5. Repeat steps 3 and 4 until observations perfectly predicted or a preset number of trees are trained.

ChrisAlbon

Gradient Boosting [Breiman, 1997]

- Instead of tweaking the instance weights at every iteration like AdaBoost does, this method fit the new predictor to the **residual errors** made by the previous predictor.

Gradient Boosting [Breiman, 1997]

- Instead of tweaking the instance weights at every iteration like AdaBoost does, this method fit the new predictor to the **residual errors** made by the previous predictor.

- Instead of training on a newly sample distribution, the weak learner trains on the remaining errors.

Gradient Boosting [Breiman, 1997]

1. Fit a simple linear regressor or decision tree on data
[call x as input and y as output]

2. Calculate error residuals. Actual target value, minus predicted target value
[e1 = y - y_predicted1]

3. Fit a new model on error residuals as target variable with same input variables
[call it e1_predicted]

4. Add the predicted residuals to the previous predictions
[y_predicted2 = y_predicted1 + e1_predicted]

5. Fit another model on residuals that is still left, i.e. **[e2 = y - y_predicted2]** and repeat steps 2 to 5 until it starts overfitting or the sum of residuals become constant.

Gradient Boosting [Breiman, 1997]

- Instead of tweaking the instance weights at every iteration like AdaBoost does, this method fit the new predictor to the **residual errors** made by the previous predictor.
- Instead of training on a newly sample distribution, the weak learner trains on the remaining errors.
- XGboost [Chen and Guestrin, 2016]: Extreme Gradient Boosting

Stacking

Stacking [Wolpert, 1992]

- Stacking (short for Stacked Generalization)

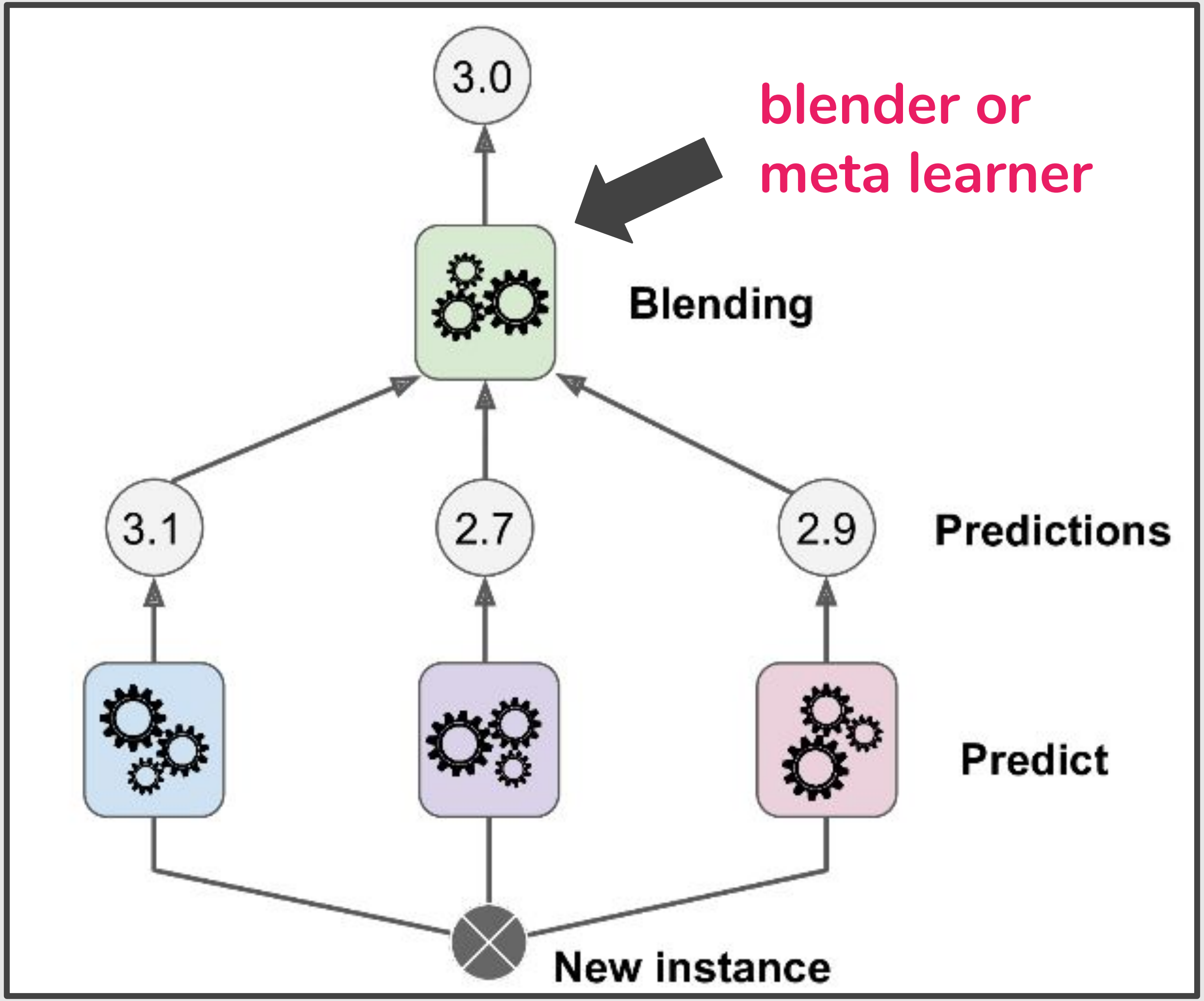
“Stacked Generalization”, D. Wolpert (1992): http://goo.gl/9I2NBw

Stacking [Wolpert, 1992]

- Stacking (short for Stacked Generalization)
- Instead of using trivial functions (such as hard voting) to aggregate the predictions of all predictors in an ensemble, we **train a model to perform this aggregation**.

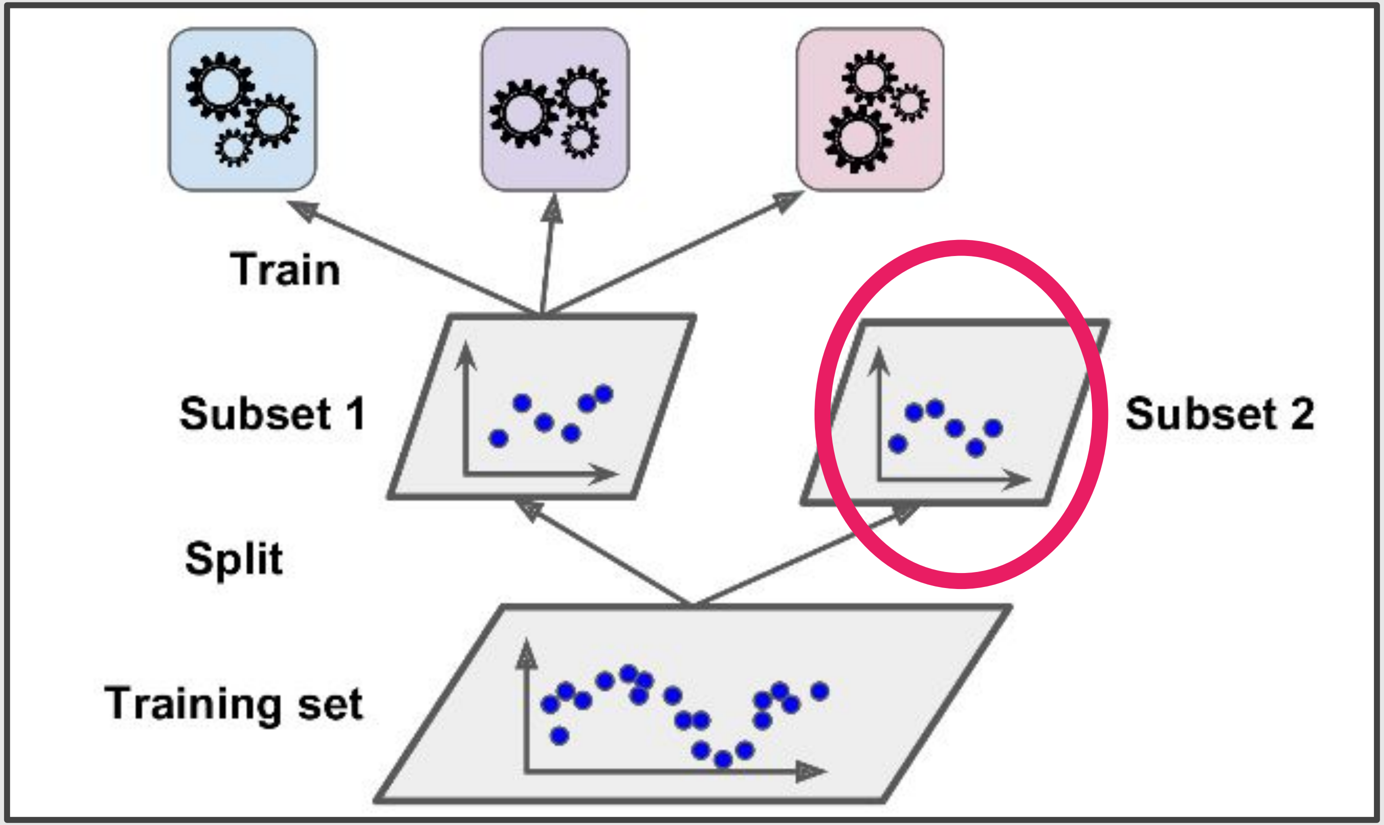
"Stacked Generalization", D. Wolpert (1992): http://goo.gl/9I2NBw

Stacking

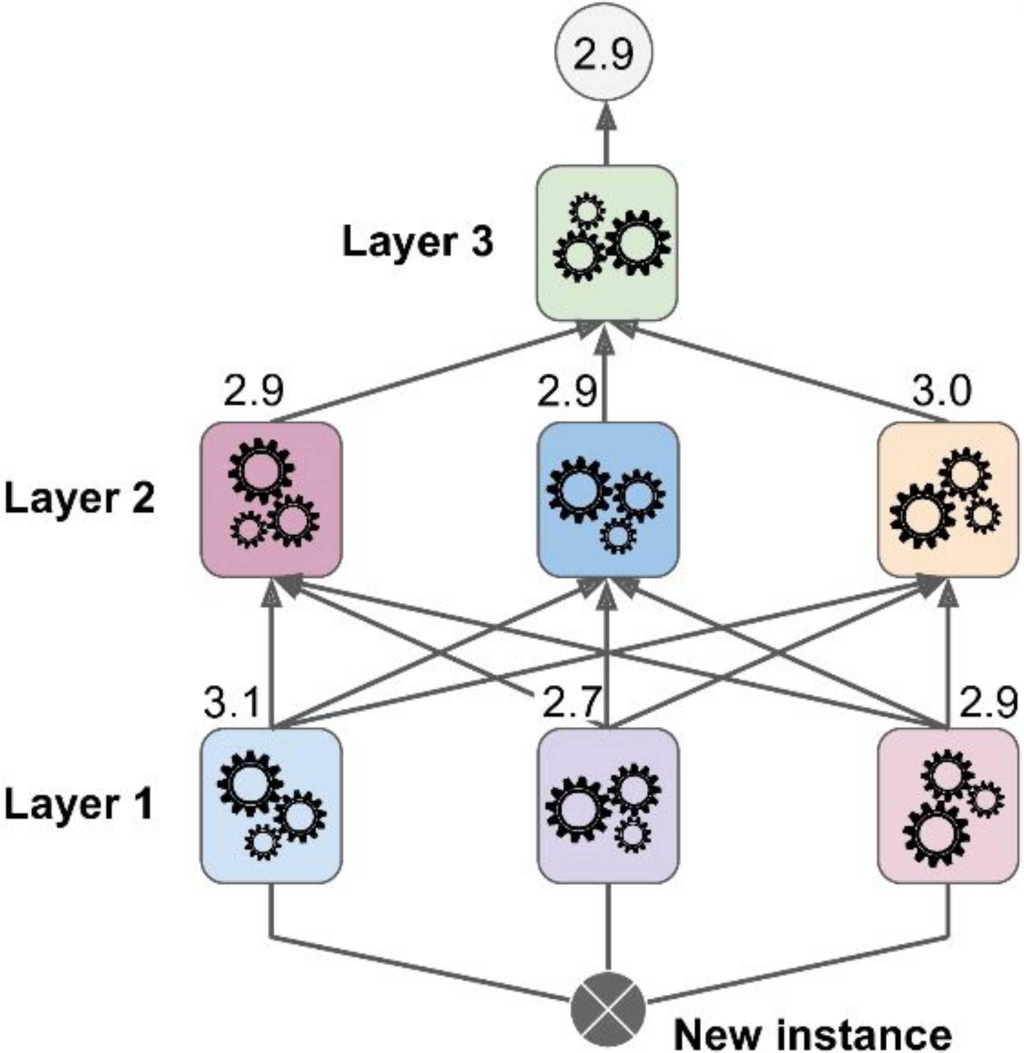


3.0
blender or
meta learner
Blending
3.1
2.7
2.9
Predictions
Predict
New instance

To train the blender, a common approach is to use a **hold-out set**.



Multi-layer Stacking Ensemble



References

Machine Learning Books

- Hands-On Machine Learning with Scikit-Learn and TensorFlow, Chap. 6 & 7
- Pattern Recognition and Machine Learning, Chap. 14
- Pattern Classification, Chap 8 & 9 (Sec. 9.5)