
IRON
HACK
ENSEMBLE METHODS

ENSEMBLE METHODS

WHAT ARE ENSEMBLE METHODS?

During the Lockdown I decided to improve my cooking skills.

I'm quite happy with my results but I feel I should get external validation. What are my choices?

WHAT ARE ENSEMBLE METHODS?

I'm quite happy with my results but I feel I should get external validation. What are my choices?

1 - Ask Chef Ramsay

WHAT ARE ENSEMBLE METHODS?

I'm quite happy with my results but I feel I should get external validation. What are my choices?

2 - Ask all my family members, friends and students to vote

WHAT ARE ENSEMBLE METHODS?

Ensemble methods base themselves on the idea that the average opinion of several weak predictors will yield a strong predictor.

A Group of predictors is called an Ensemble, thus the Ensemble learning method

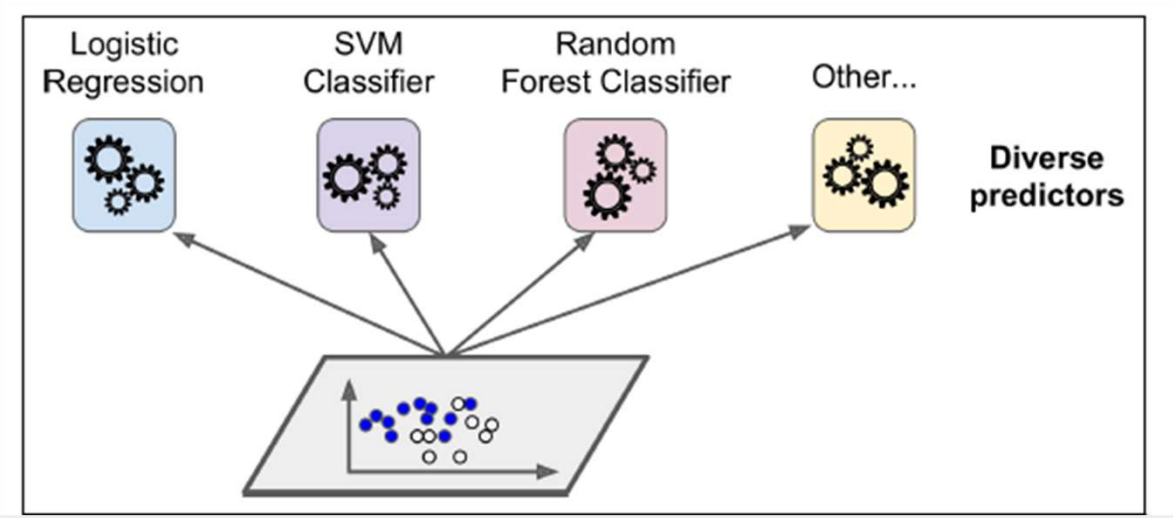
VOTING CLASSIFIERS

Train individually the algorithms that we have seen so far and then take a vote of the answers!

Process of voting classifiers:

- train individually each algorithm

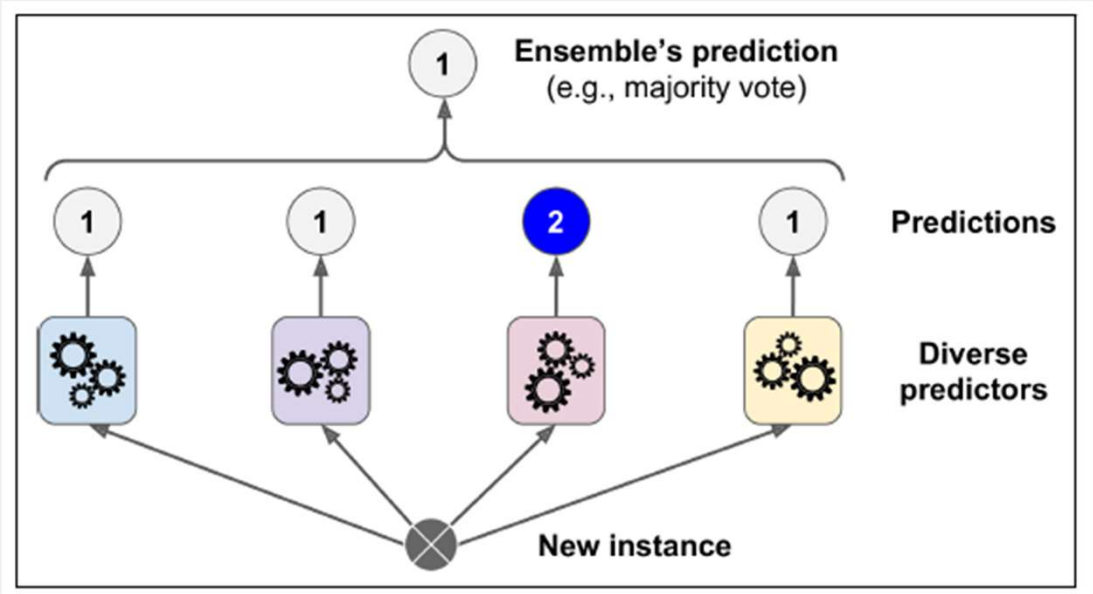
- save all the generated models in a function



VOTING CLASSIFIERS

Train individually the algorithms that we have seen so far and then take a vote of the answers!

Once you receive a new instance (data point) you can then pass it through all the algorithms and perform a majority vote of the answer (classification problem)

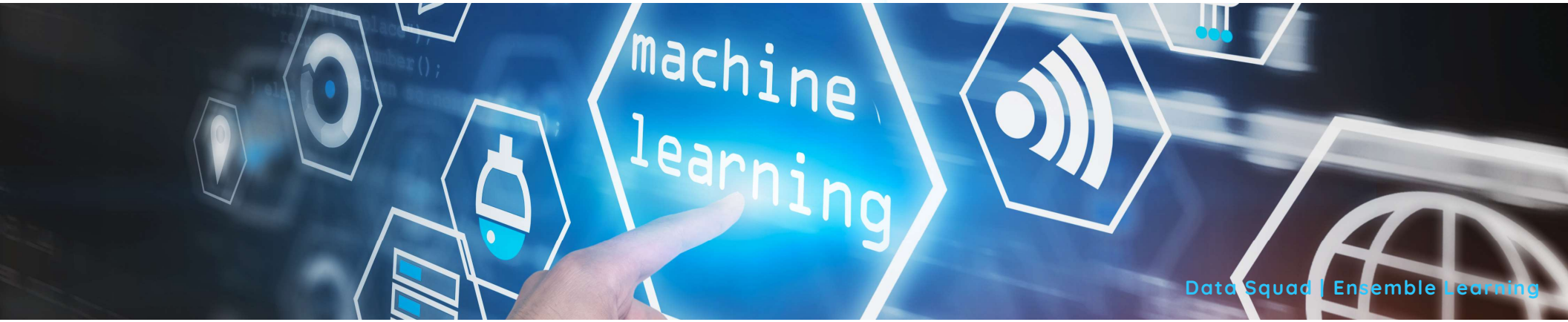


STACKING

Why can't we take this one step further and have our ensemble of weak classifiers and then have one final algorithm that "learns" the best way to perform the final decision?

This is an alternative to using a naive majority vote or average

Let's train a model to perform this aggregation

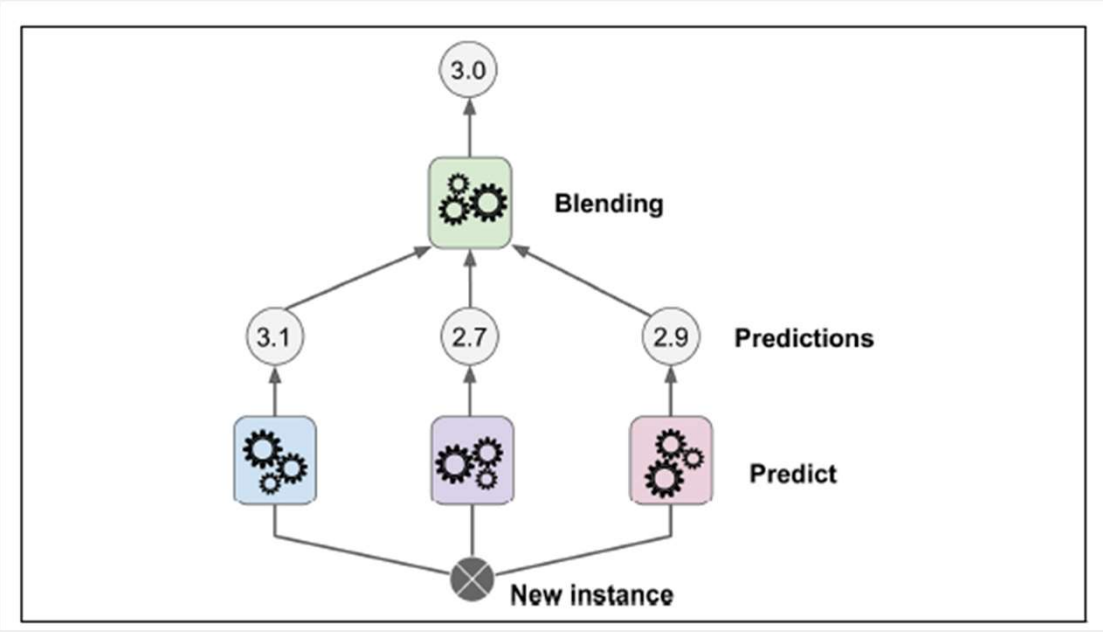


STACKING

Why can't we take this one step further and have our ensemble of weak classifiers and then have one final algorithm that "learns" the best way to perform the final decision?

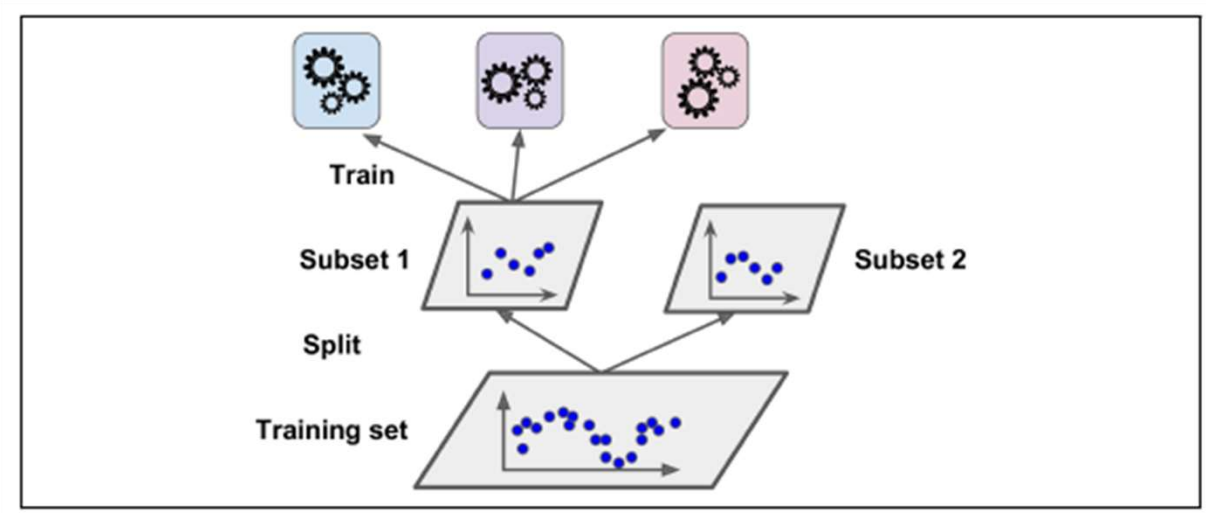
This is an alternative to using a naive majority vote or average

Let's train a model to perform this aggregation



STACKING

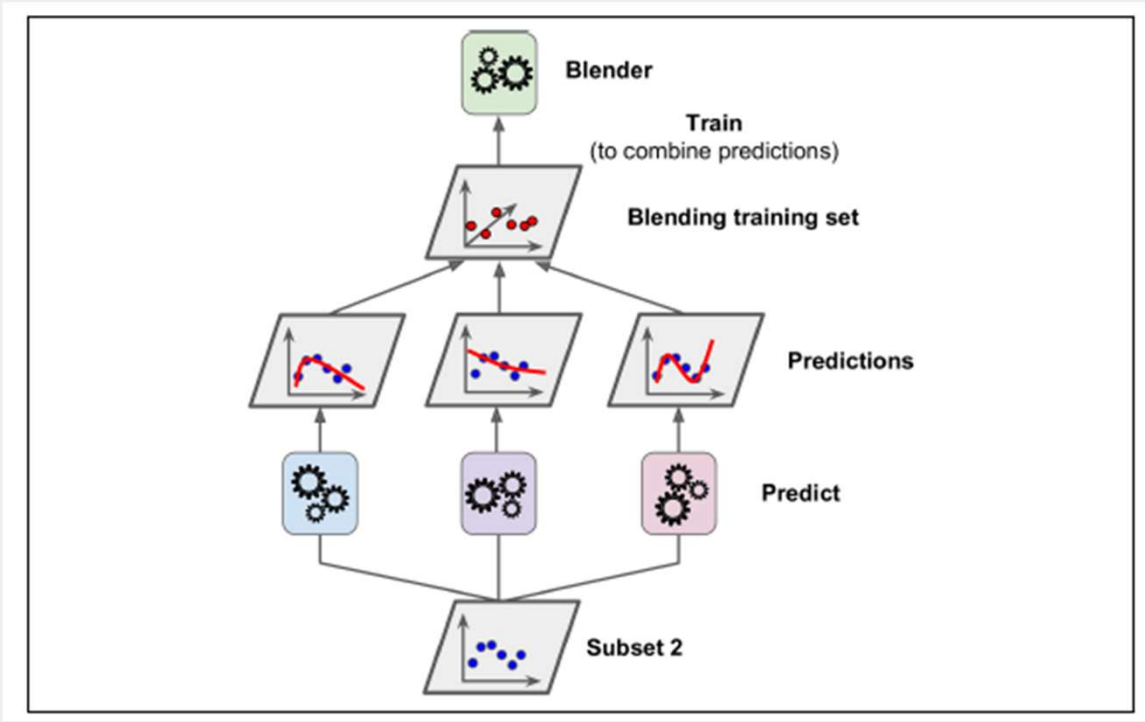
In practice this is done in two steps:



From your training dataset, leave out a subset to then be used to train the blending algorithm.
Perform your usual training

STACKING

In practice this is done in two steps:



Use the second subset that the trained weak predictors have never seen before to make them make predictions.

Finally with these predictions a final belding algorithm will be trained to find the best way to combine the predictions

VOTING CLASSIFIERS

when can this go wrong?

Well, this is a great voting system if all algorithms are truly independent of each other and will not tend to overfit in the same way or have the same weaknesses.

If they are not independent, you are simply giving more weight to inaccurate predictions.

hence the importance of having diverse classifiers

BAGGING AND PASTING

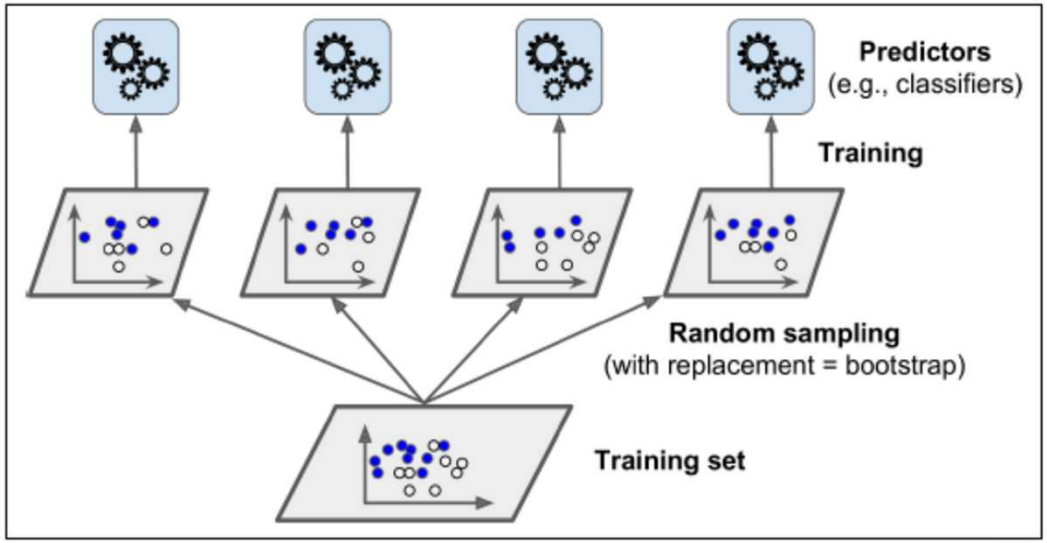
So how can we make sure that we create independent classifiers?

One way is to always use the same algorithm but then train it on a different random sample of the training dataset.

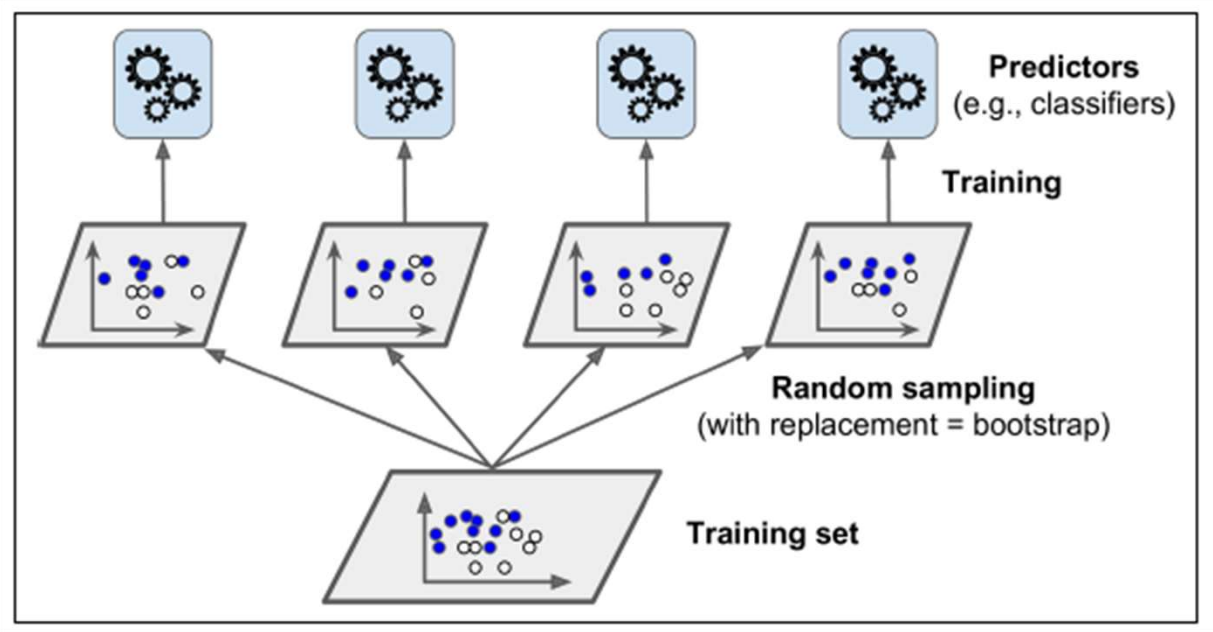
Bagging - When random sample is taken with replacement

Pasting - When random sample is taken without replacement

BAGGING AND PASTING

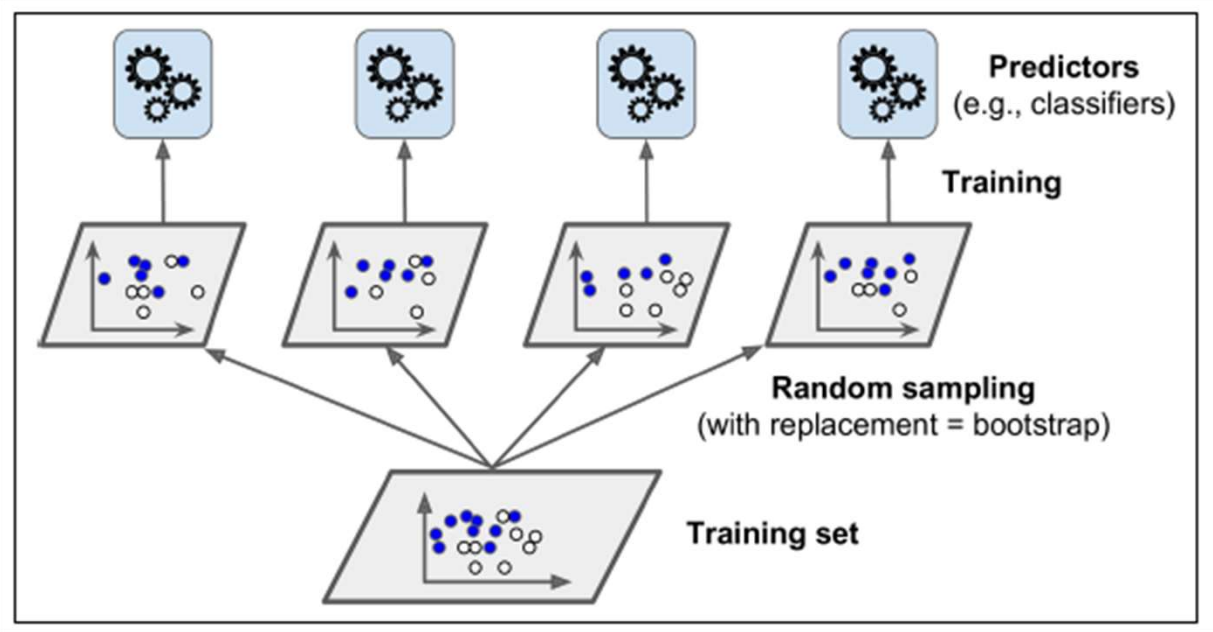


BAGGING AND PASTING



Each individual predictor has a higher bias than if it were trained on the original training set, but the final aggregation reduces the overall bias

BAGGING AND PASTING



Each individual predictor has a higher bias than if it were trained on the original training set, but the final aggregation reduces the overall bias

can be made very performant if process if parallelized

BAGGING AND PASTING - EXAMPLE

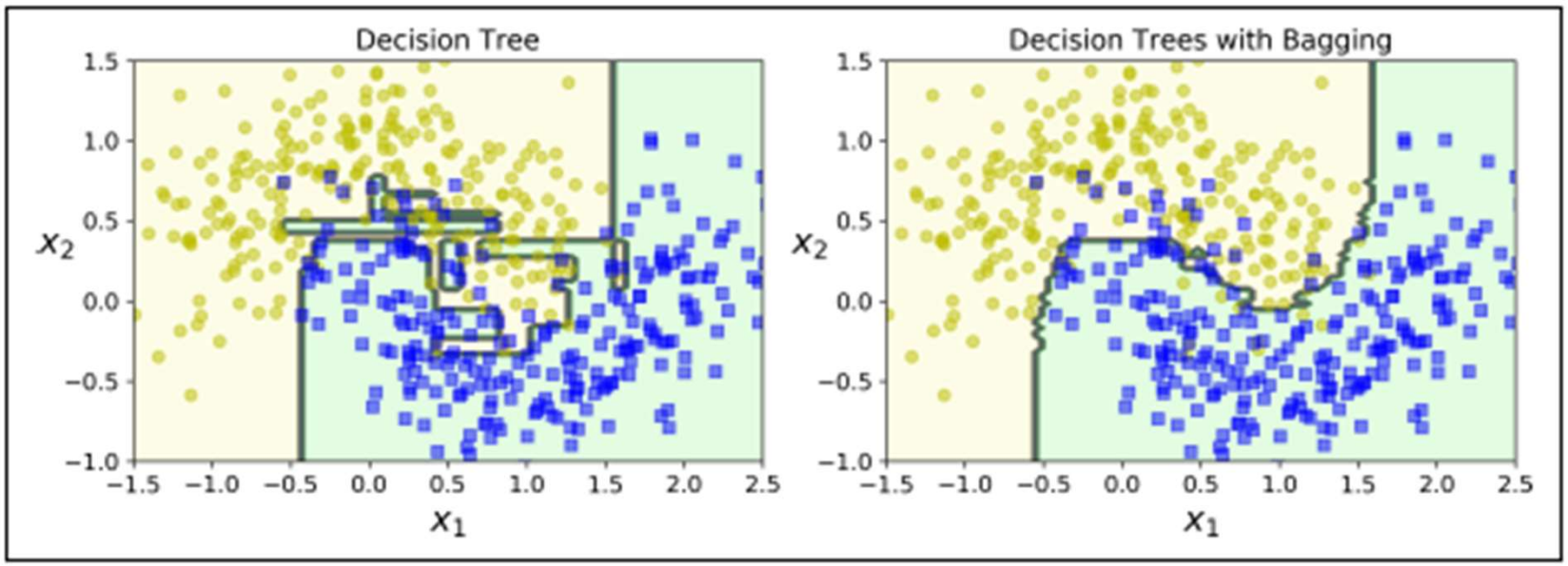


Figure 7-5. A single Decision Tree versus a bagging ensemble of 500 trees

RANDOM PATCHES METHOD

The previous methods randomized the training data used. What else could we randomize?

RANDOM PATCHES METHOD

The previous methods randomized the training data used. What else could we randomize?

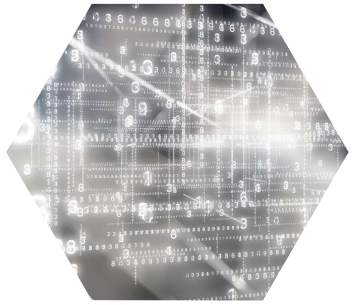
The FEATURES

RANDOM PATCHES METHOD

The previous methods randomized the training data used. What else could we randomize?

The FEATURES

By repeating the algorithm training process randomizing both the training data subset but also the features chosen to perform the training is called the Random Patches Method



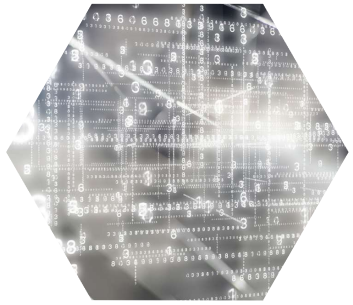
RANDOM PATCHES METHOD

The previous methods randomized the training data used. What else could we randomize?

The FEATURES

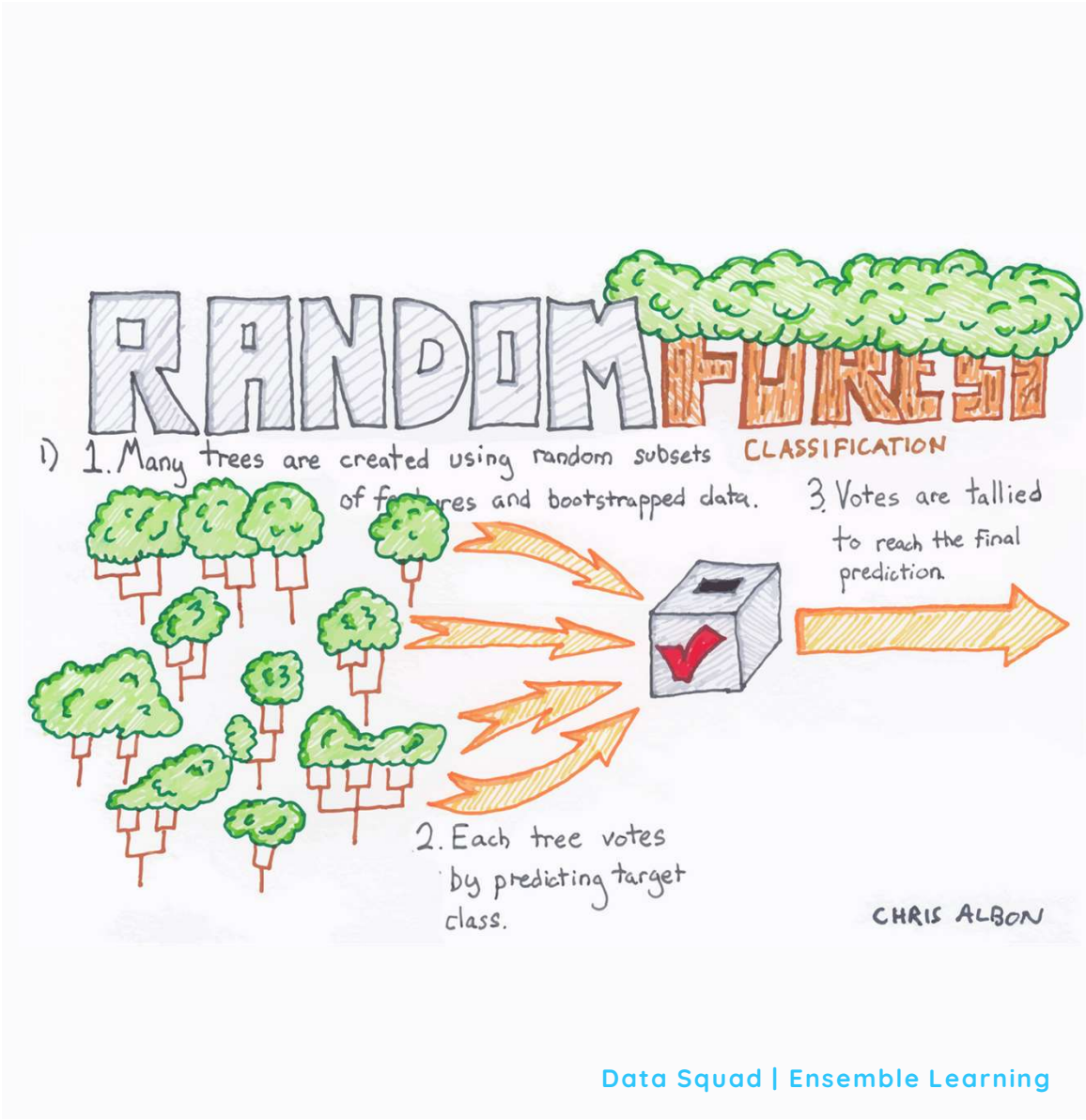
By repeating the algorithm training process randomizing both the training data subset but also the features chosen to perform the training is called the Random Patches Method

Becomes more powerful when we have a very high dimensional problem (many features!). This is the case in image processing for example (each pixel is a feature)



RANDOM FORESTS

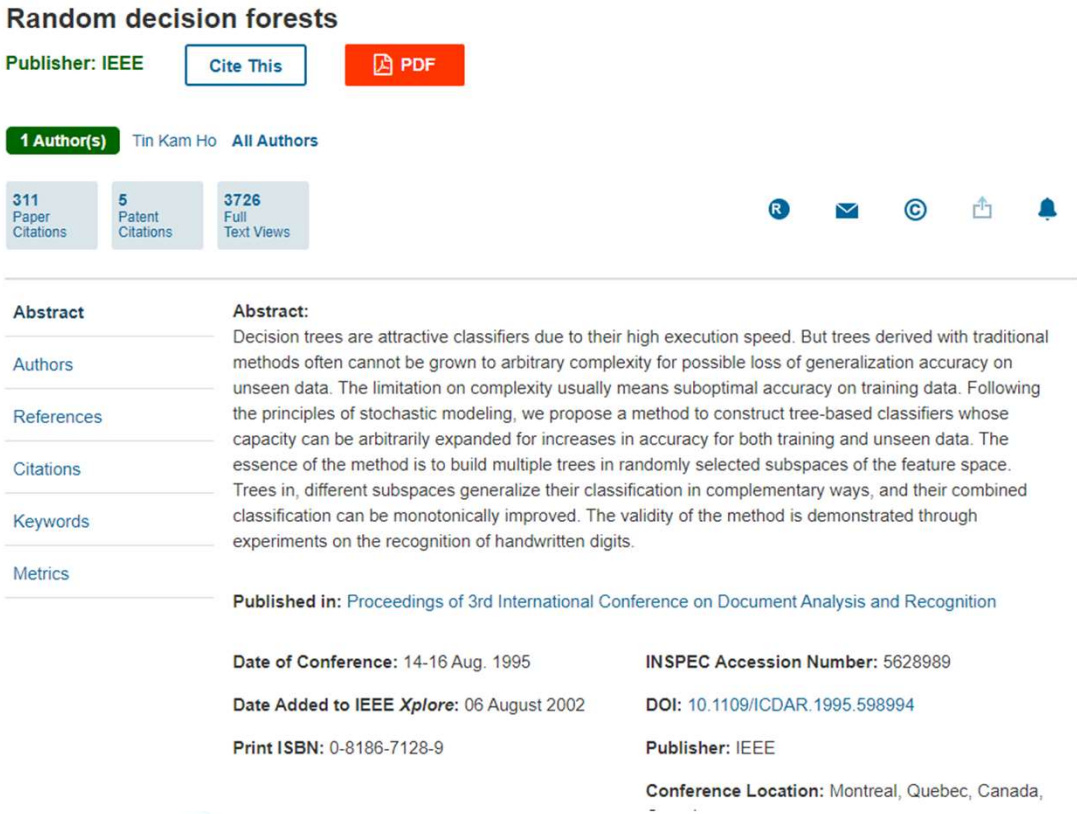
Chris Albon said it all!



RANDOM FORESTS

The previous methods randomized the training data used. What else could we randomize?The original paper comes from 1995 and the abstract actually gives us a lot of intuition about what's going on!

in each tree of the forest, only a random number of features are consideres for the splitting process!



Random decision forests

Publisher: IEEE | Cite This | PDF

1 Author(s) Tin Kam Ho All Authors

311 Paper Citations | 5 Patent Citations | 3726 Full Text Views

Abstract | Authors | References | Citations | Keywords | Metrics

Abstract:
Decision trees are attractive classifiers due to their high execution speed. But trees derived with traditional methods often cannot be grown to arbitrary complexity for possible loss of generalization accuracy on unseen data. The limitation on complexity usually means suboptimal accuracy on training data. Following the principles of stochastic modeling, we propose a method to construct tree-based classifiers whose capacity can be arbitrarily expanded for increases in accuracy for both training and unseen data. The essence of the method is to build multiple trees in randomly selected subspaces of the feature space. Trees in, different subspaces generalize their classification in complementary ways, and their combined classification can be monotonically improved. The validity of the method is demonstrated through experiments on the recognition of handwritten digits.

Published in: Proceedings of 3rd International Conference on Document Analysis and Recognition

Date of Conference: 14-16 Aug. 1995
Date Added to IEEE *Xplore*: 06 August 2002
Print ISBN: 0-8186-7128-9
INSPEC Accession Number: 5628989
DOI: 10.1109/ICDAR.1995.598994
Publisher: IEEE
Conference Location: Montreal, Quebec, Canada,

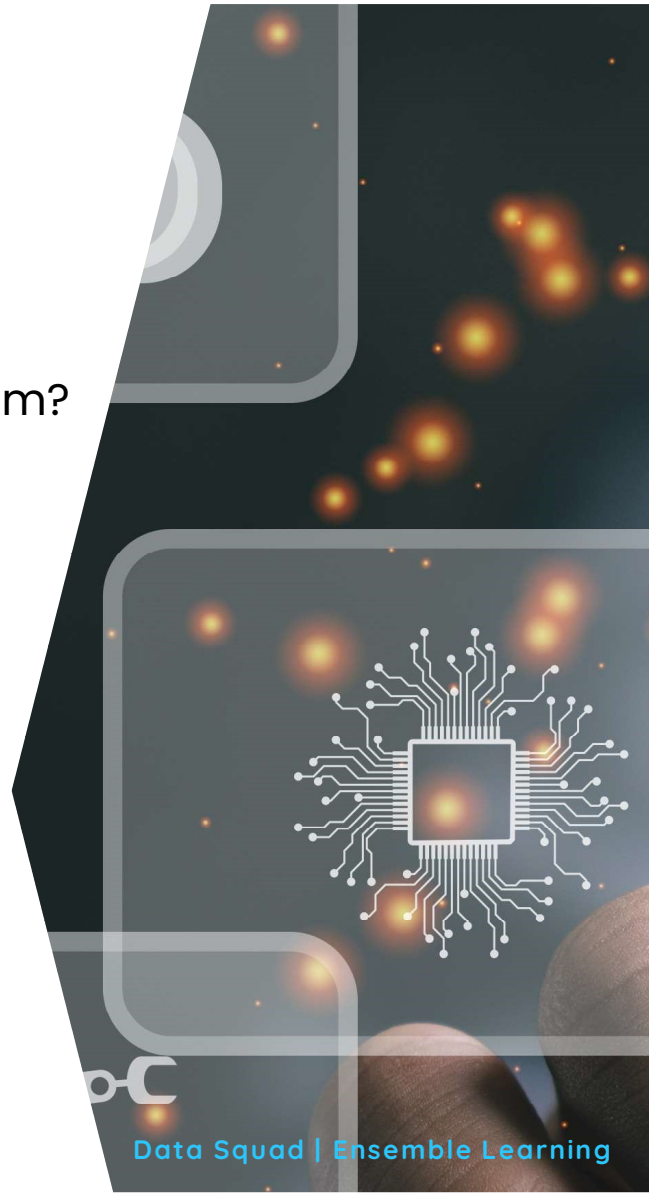
RANDOM FORESTS

What are the hyperparameters of a random forest algorithm?

n_estimators -> how many trees do we want to initialize?

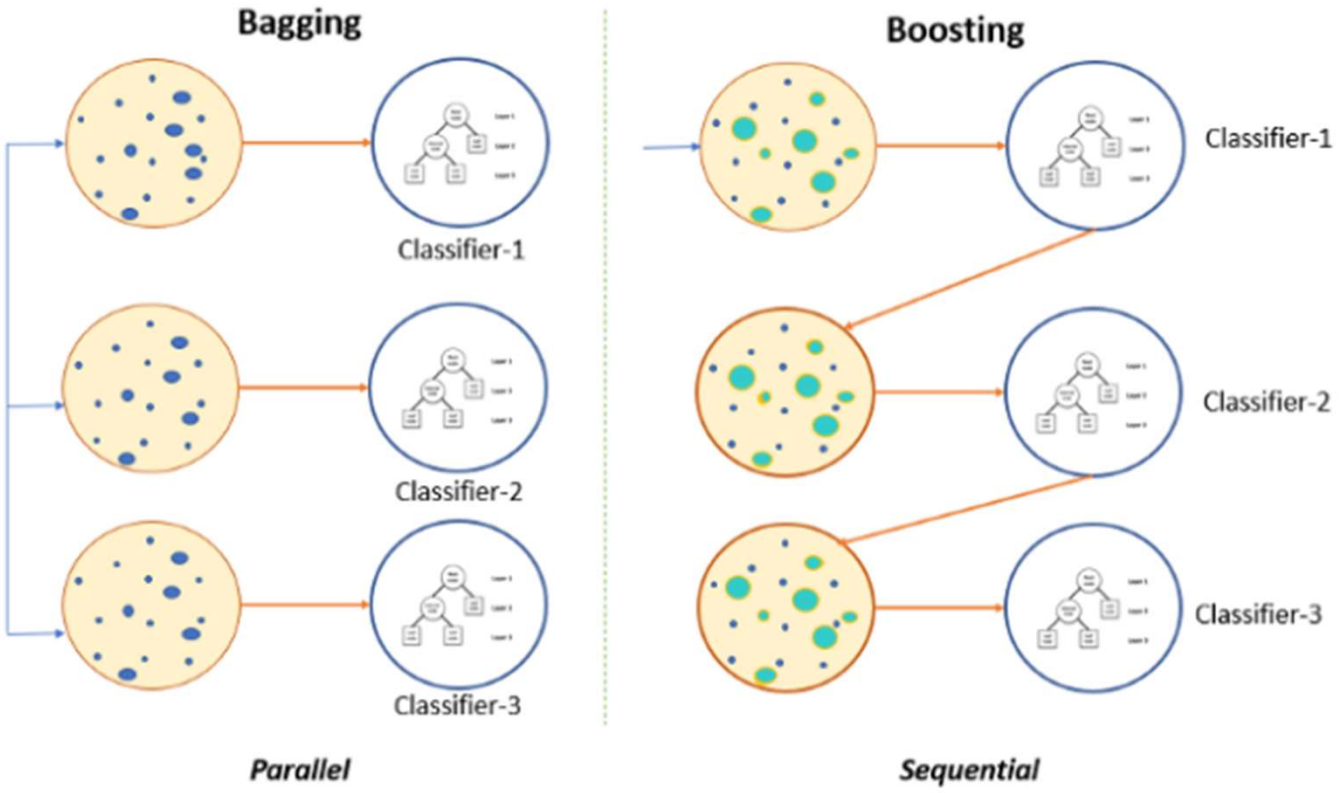
max_leaf_nodes -> how many final leafs do we want?

max_depth -> how deep do I want my tree to be?



JUPYTER (Random Forest)

BOOSTING


Bagging
Classifier-1
Classifier-2
Classifier-3
Parallel
Boosting
Classifier-1
Classifier-2
Classifier-3
Sequential

BOOSTING

Any Ensemble method that can combine weak learners into a strong learner

The main idea is to train predictors sequentially whereby the next tries to "correct" its predecessor

There are two common methods that are the most popular:

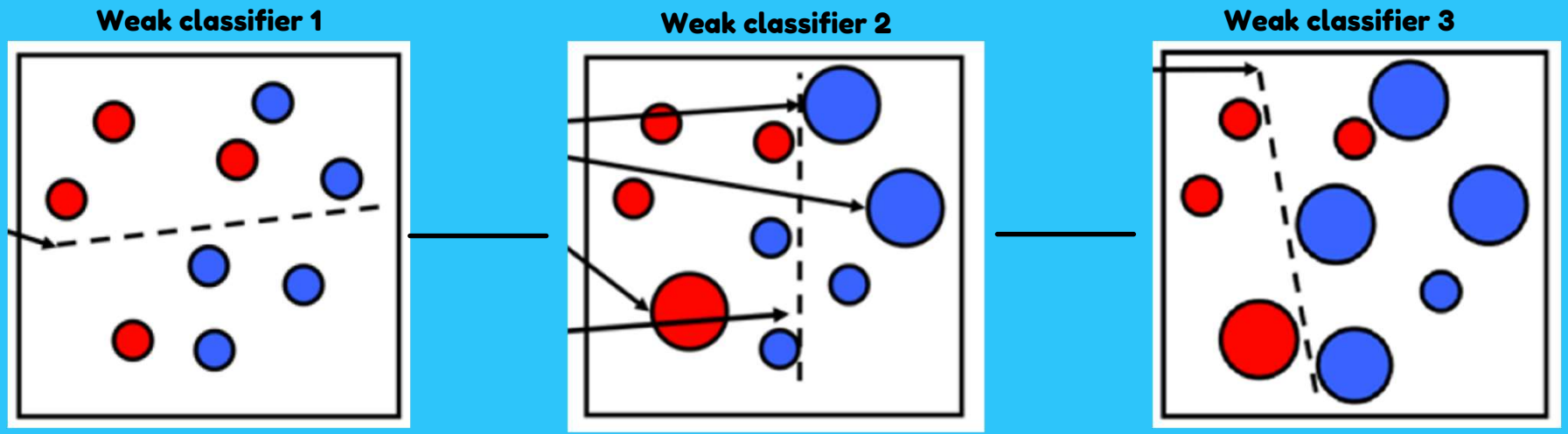
- AdaBoost (Adaptive Boosting)

- Gradient Bosting

ADA BOOSTING

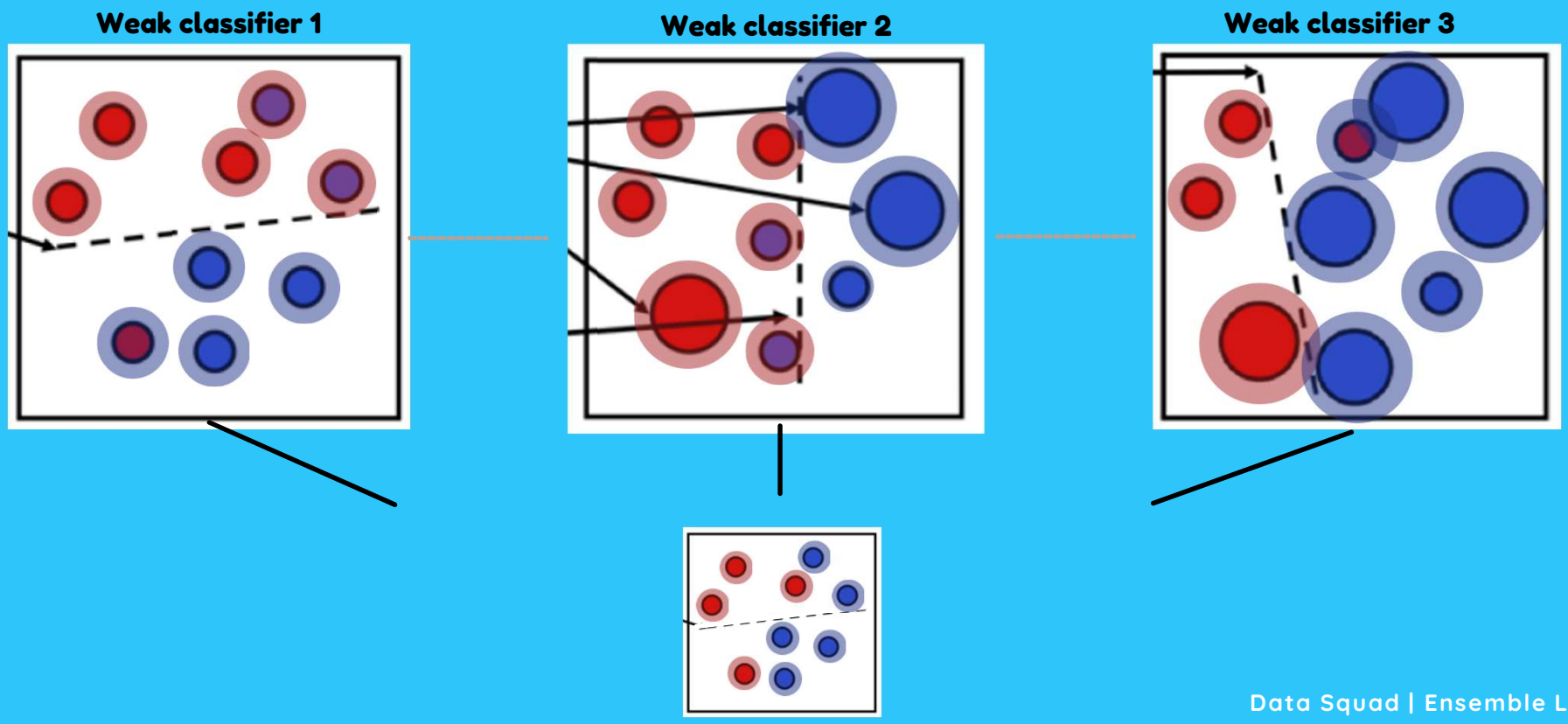
In adaboosting each new classifier looks at the previous and tries to see on which training instances did he seem to underfit

This results in a new prediction focusing on those cases.



To discuss - why is this a "weak" classifier

ADA BOOSTING

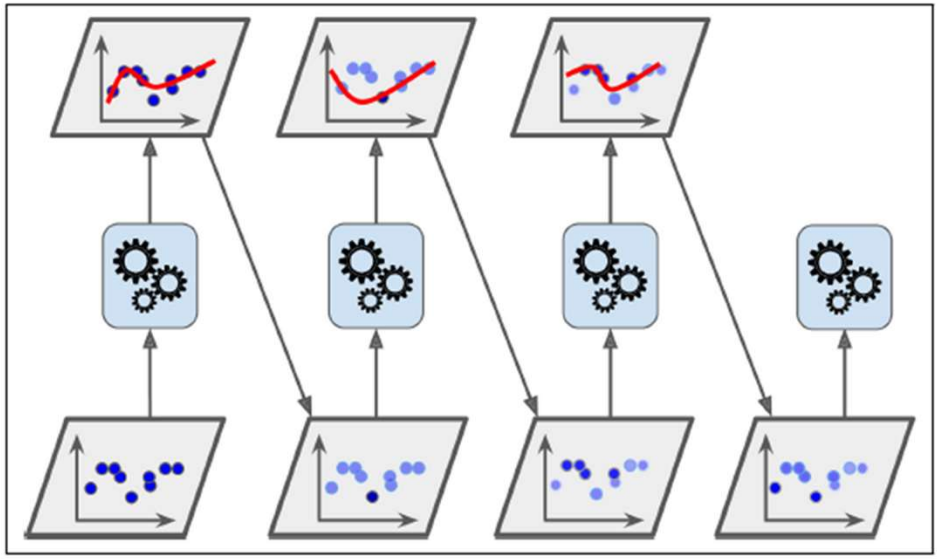


ADA BOOSTING

GRADIENT BOOSTING

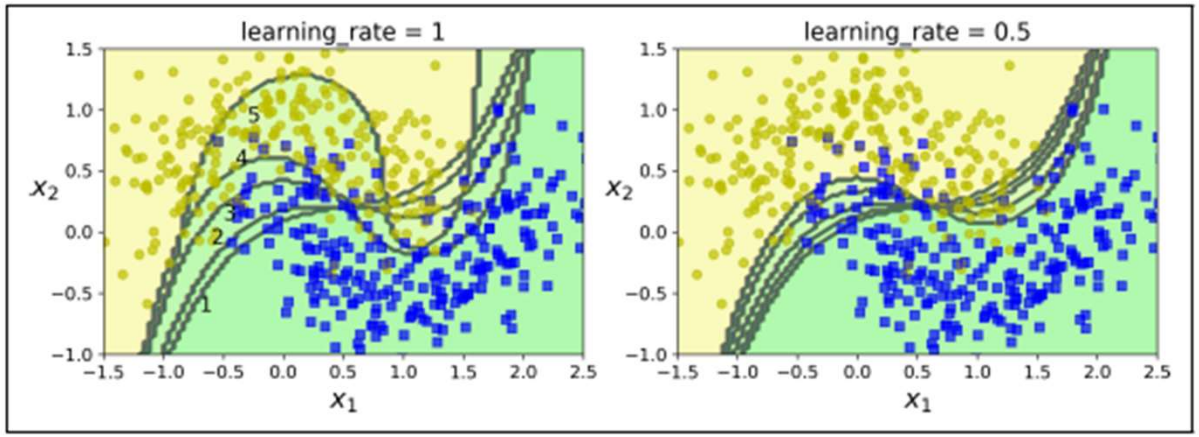
In adaboosting each new classifier looks at the previous and tries to see on which training instances did he seem to underfit

This results inanew prediction focusing on those cases.



[AdaBosting Paper]

ADA BOOSTING



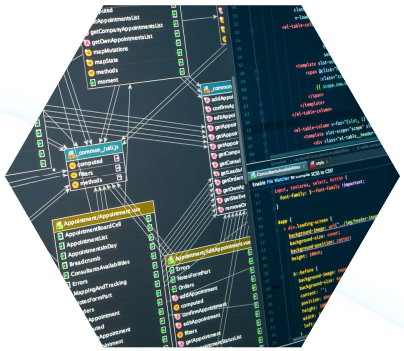
Once all predictors are trained we still have an ensemble!!

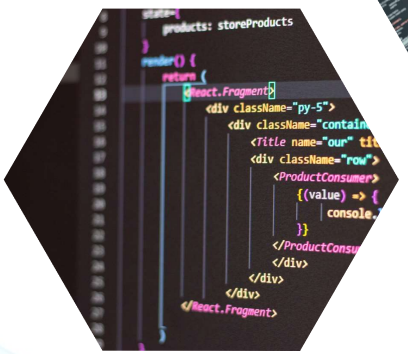
However, the voting method is made giving different weight to each method depending on the overall accuracy

These weights are themselves optimized in the machine learning process

ADA BOOSTING - CODE SNIP

```
from sklearn.ensemble import AdaBoostClassifier

ada_clf = AdaBoostClassifier(
    DecisionTreeClassifier(max_depth=1), n_estimators=200,
    algorithm="SAMME.R", learning_rate=0.5)
ada_clf.fit(X_train, y_train)
```





GRADIENT BOOSTING

Similarly to AdaBoost, Gradient Boosting works be sequentially adding predictors to an ensemble each correcting its predecessor.

The difference is that Gradient Boosting tries to fit the new predictor to the residual errors made by the previous predictor

GRADIENT BOOSTING

Sounds confusing, in practice what does this mean?If I have a training dataset X and a target variable y:

```
tree_reg1 = DecisionTreeRegressor(max_depth=2)
tree_reg1.fit(X, y)
```

```
y2 = y - tree_reg1.predict(X)
tree_reg2 = DecisionTreeRegressor(max_depth=2)
tree_reg2.fit(X, y2)
```

```
y3 = y2 - tree_reg2.predict(X)
tree_reg3 = DecisionTreeRegressor(max_depth=2)
tree_reg3.fit(X, y3)
```

```
y_pred = sum(tree.predict(X_new) for tree in (tree_reg1, tree_reg2, tree_reg3))
```

GRADIENT BOOSTING

Sounds confusing, in practice what does this mean?If I have a training dataset X and a target variable y:

```
tree_reg1 = DecisionTreeRegressor(max_depth=2)
tree_reg1.fit(X, y)
```

I am now going to see the mistakes made by my fit

GRADIENT BOOSTING

Sounds confusing, in practice what does this mean?If I have a training dataset X and a target variable y:

```
tree_reg1 = DecisionTreeRegressor(max_depth=2)
tree_reg1.fit(X, y)
```

```
y2 = y - tree_reg1.predict(X)
tree_reg2 = DecisionTreeRegressor(max_depth=2)
tree_reg2.fit(X, y2)
```

I am now going to see the mistakes made by my fit

Compute the array of those mistakes

and then try to create a model that based on the input features X, predicts the errors!!!

GRADIENT BOOSTING

Sounds confusing, in practice what does this mean?If I have a training dataset X and a target variable y:

```
tree_reg1 = DecisionTreeRegressor(max_depth=2)
tree_reg1.fit(X, y)
```

Cool, now why don't I repeat that and tree to see the error I have in predicting the errors

```
y2 = y - tree_reg1.predict(X)
tree_reg2 = DecisionTreeRegressor(max_depth=2)
tree_reg2.fit(X, y2)
```

GRADIENT BOOSTING

Sounds confusing, in practice what does this mean?If I have a training dataset X and a target variable y:

```
tree_reg1 = DecisionTreeRegressor(max_depth=2)
tree_reg1.fit(X, y)
```

Cool, now why don't I repeat that and tree to see the error I have in predicting the errors

```
y2 = y - tree_reg1.predict(X)
tree_reg2 = DecisionTreeRegressor(max_depth=2)
tree_reg2.fit(X, y2)
```

```
y3 = y2 - tree_reg2.predict(X)
tree_reg3 = DecisionTreeRegressor(max_depth=2)
tree_reg3.fit(X, y3)
```

GRADIENT BOOSTING

Sounds confusing, in practice what does this mean?If I have a training dataset X and a target variable y:

```
tree_reg1 = DecisionTreeRegressor(max_depth=2)
tree_reg1.fit(X, y)
```

Finally, I have 3 models:
one for y
one for the error in y
one for the error in the error in y

```
y2 = y - tree_reg1.predict(X)
tree_reg2 = DecisionTreeRegressor(max_depth=2)
tree_reg2.fit(X, y2)
```

```
y3 = y2 - tree_reg2.predict(X)
tree_reg3 = DecisionTreeRegressor(max_depth=2)
tree_reg3.fit(X, y3)
```

pass a new datapoint through the 3 and add the results.

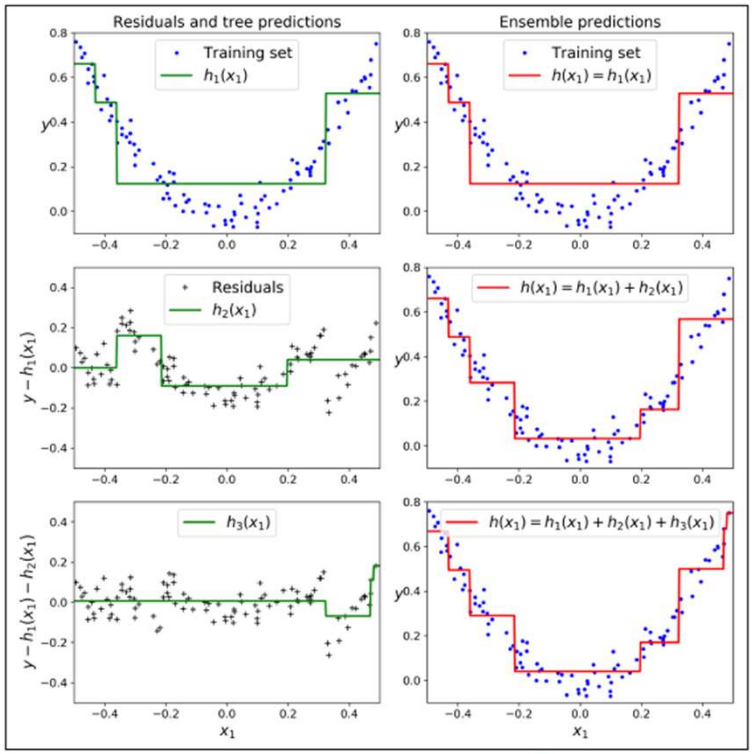
```
y_pred = sum(tree.predict(X_new) for tree in (tree_reg1, tree_reg2, tree_reg3))
```

GRADIENT BOOSTING - CODE SNIP

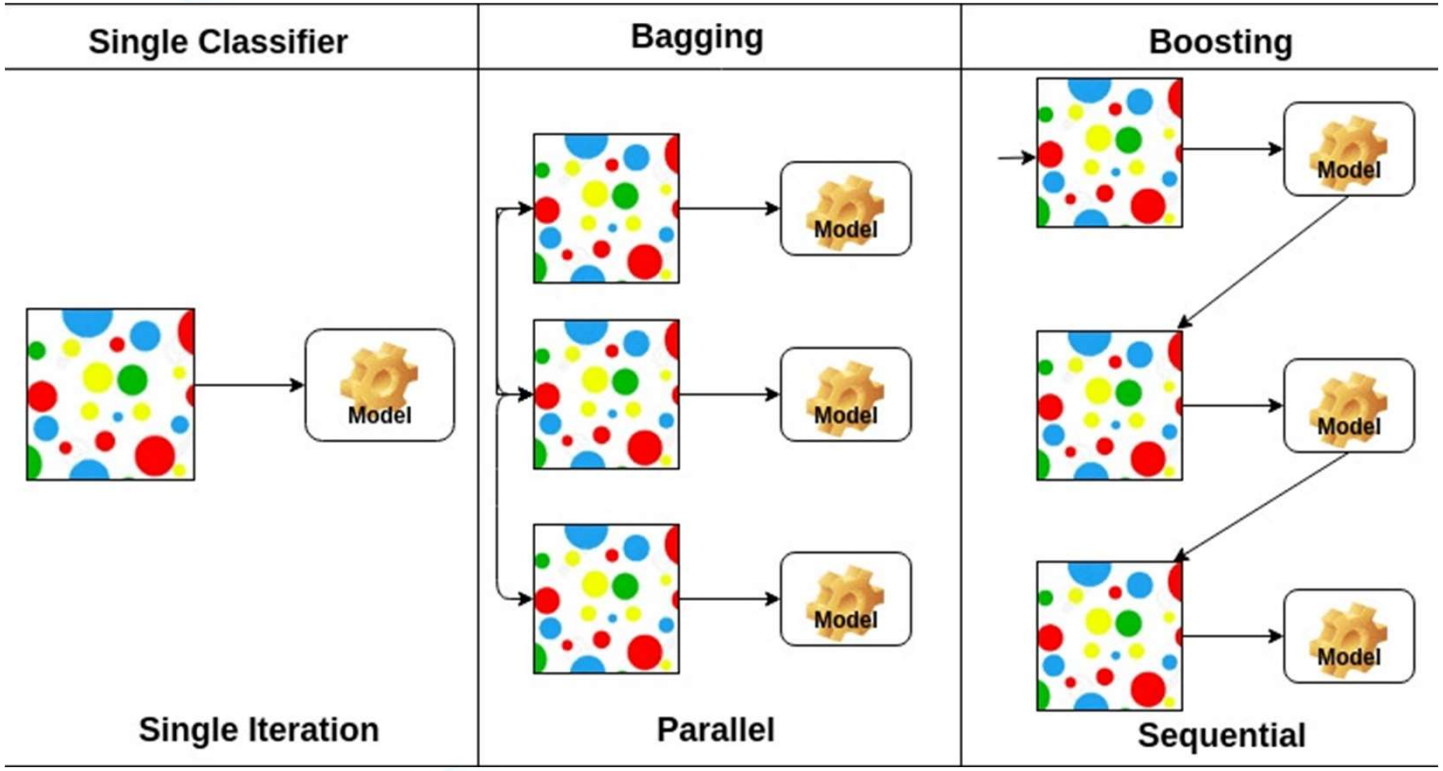
```
from sklearn.ensemble import GradientBoostingRegressor

gbrt = GradientBoostingRegressor(max_depth=2, n_estimators=3, learning_rate=1.0)
gbrt.fit(X, y)
```

This code is equivalent to the previous examples

GRADIENT BOOSTING - VISUAL MEANING



SUMMING UP


Single Classifier
Bagging
Boosting
Model
Model
Model
Model
Model
Model
Model
Model
Single Iteration
Parallel
Sequential

ANY QUESTIONS ?