Ensemble Methods with Python

by: Sarah Nooravi

Motivation by Example

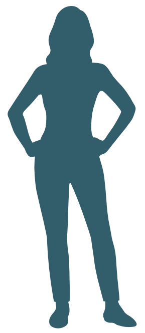
Problem: Set rules for classification of spam emails

Solution: We can generate various rules for classification of spam emails, let's look at the some of them:

- Spam
  - Have total length less than 20 words
  - Have only image (promotional images)
  - Have specific key words like "make money and grow" and "reduce your fat"
  - More miss spelled words in the email
- Not Spam
  - Email from Analytics Vidhya domain
  - Email from family members or anyone from e-mail address book

https://www.analyticsvidhya.com/blog/2015/09/questions-ensemble-modeling/

One vs. Many

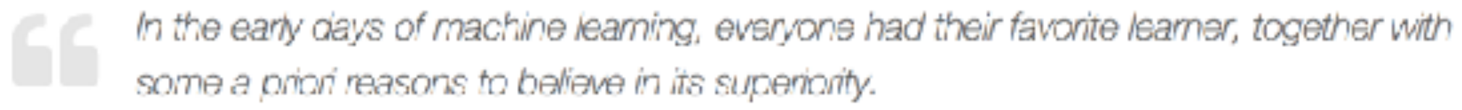


Relating this to real life, a group of people are likely to make better decisions compared to individuals, especially when group members come from **diverse** backgrounds. Basically, an ensemble is a supervised learning technique for combining multiple *weak learners/ models* to produce a *strong learner*.

Note: Ensemble model works better, when we ensemble models with low correlation.

Motivation - Advantages

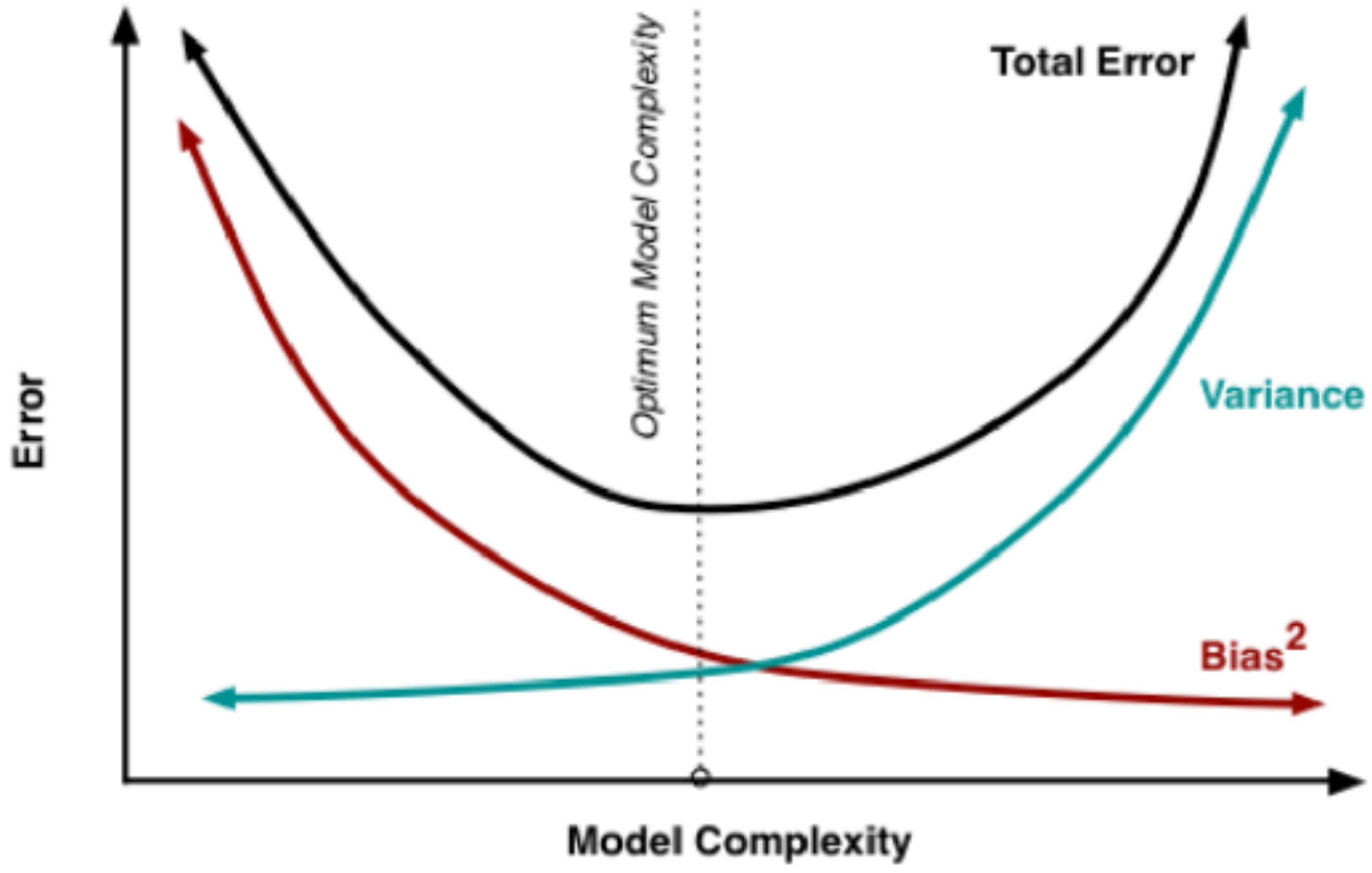
- Better predictions
- More stable model
- Increase accuracy
- Reduces variance error (less noise)
- Important in industries like healthcare where even the smallest amount of improvement in the accuracy of machine learning algorithms can be something truly valuable.


"In the early days of machine learning, everyone had their favorite learner, together with some a priori reasons to believe in its superiority.

The aggregate opinion of a multiple models is less noisy than other models. In finance, we called it "Diversification" a mixed portfolio of many stocks will be much less variable than just one of the stocks alone.

Disadvantages

- Complexity
- Not guaranteed to perform better
- Time and computation expensive
- Loss of interpretability. (Often not preferred in industries where interpretability is more important.)



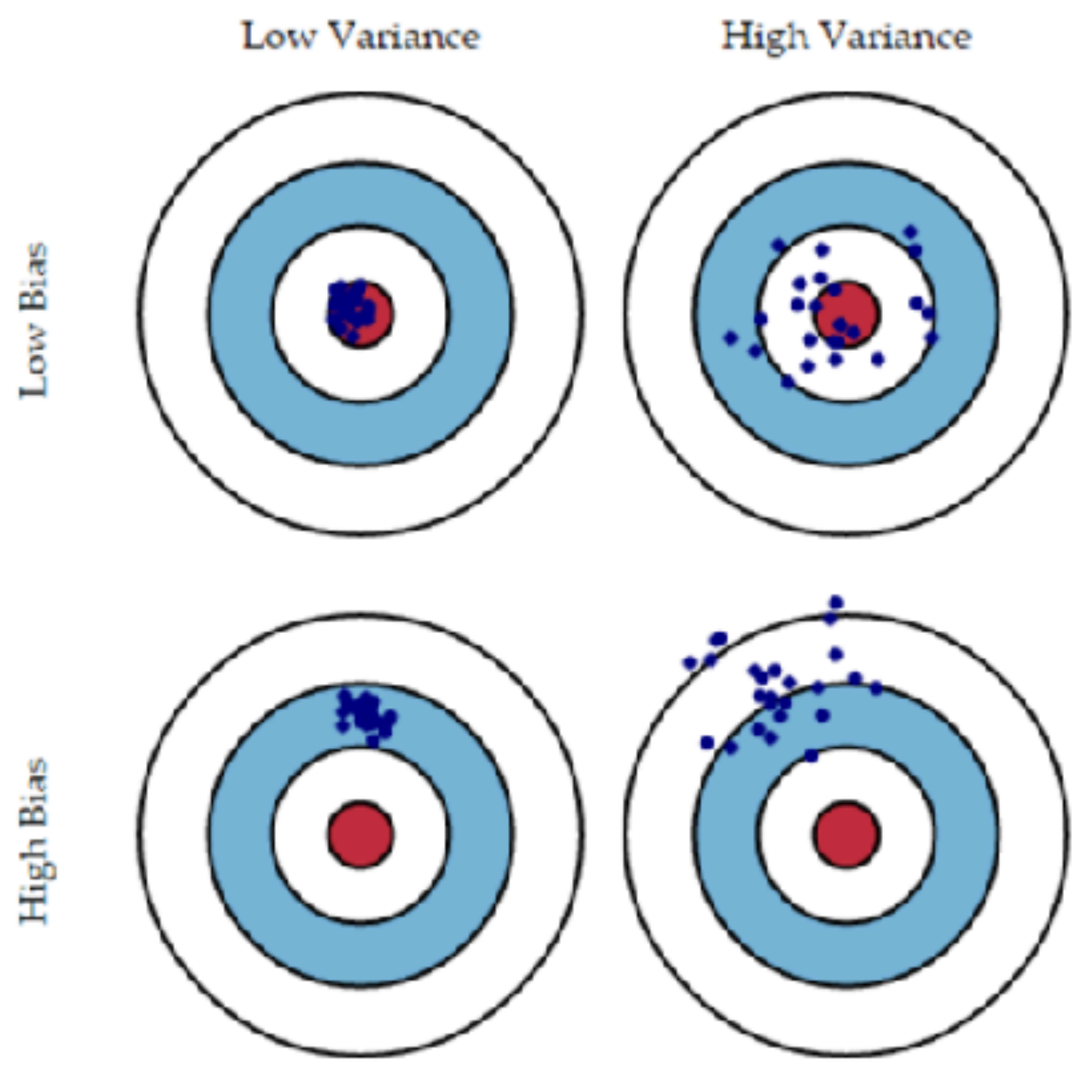
High variance can cause an algorithm to model the random noise in the training data, rather than the intended outputs (**overfitting**)

Modeling Error

- **Bias error** is useful to quantify how much on an average are the predicted values different from the actual value. A high bias error means we have a under-performing model which keeps on missing important trends.
- **Variance** on the other side quantifies how are the prediction made on same observation different from each other. A high variance model will over-fit on your training population and perform badly on any observation beyond training. Following diagram will give you more clarity (Assume that red spot is the real value and blue dots are predictions)

$$Err(x) = \left(E[\hat{f}(x)] - f(x)\right)^2 + E\left[\hat{f}(x) - E[\hat{f}(x)]\right]^2 + \sigma_e^2$$

$$Err(x) = \mathrm{Bias}^2 + \mathrm{Variance} + \mathrm{Irreducible\ Error}$$



Ensemble Methods

A champion model should maintain a balance between these two types of errors. Ensemble learning is one way to execute this trade off analysis.

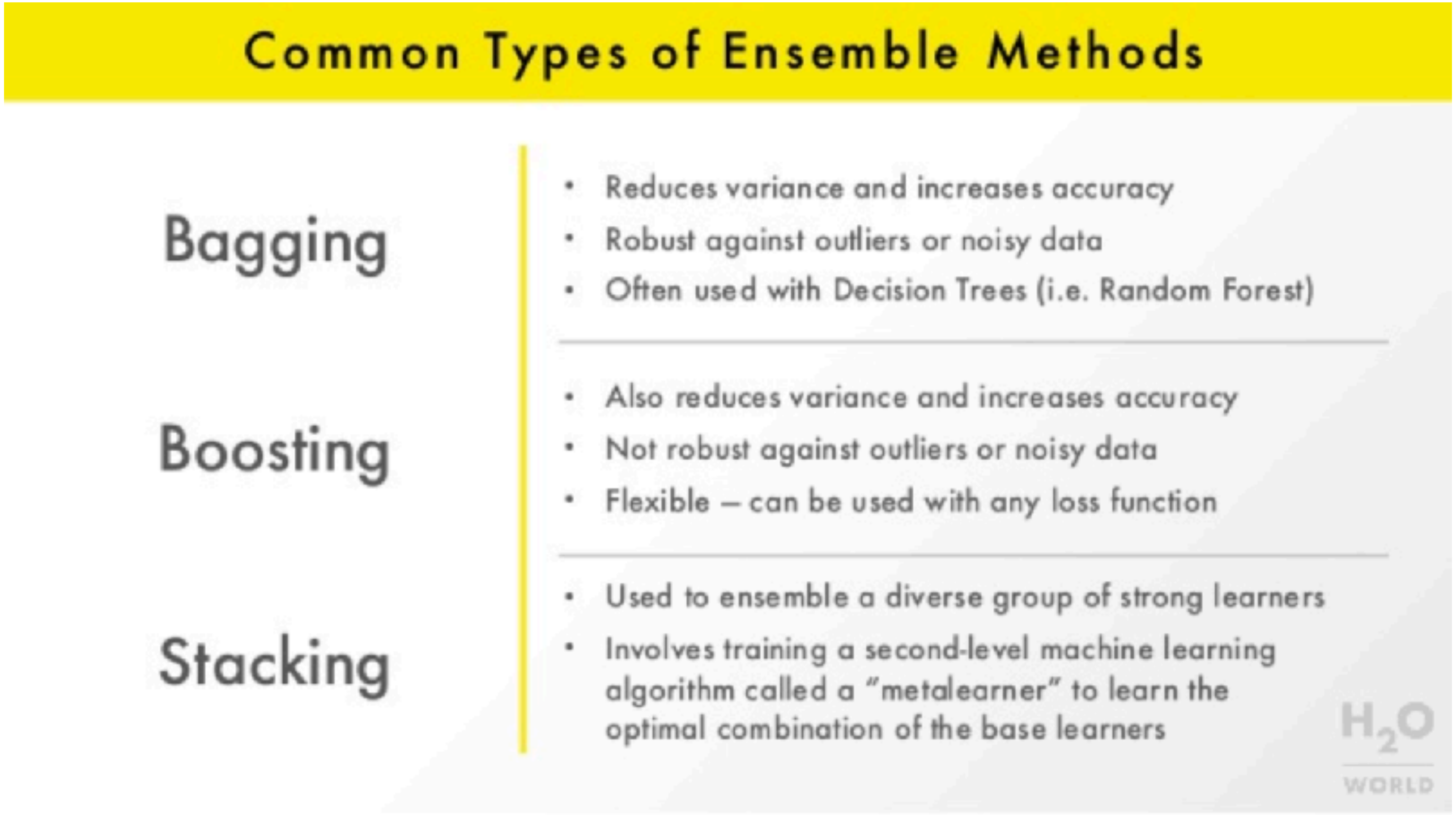
Ensemble Methods

- **Bagging**
- **Boosting**
- **Stacking**

Ensemble Methods

- **Bagging**: generate different samples of the training data, prepare the learner on each and combine the predictions using voting.
- **Boosting**: weight training instances by their difficulty during training to put special focus on those difficult to classify instances.
- **Stacking**: use a higher-level classifier to learn how to best combine the predictions of other classifier

Ensemble Methods



Common Types of Ensemble Methods

Bagging	• Reduces variance and increases accuracy • Robust against outliers or noisy data • Often used with Decision Trees (i.e. Random Forest)
Boosting	• Also reduces variance and increases accuracy • Not robust against outliers or noisy data • Flexible – can be used with any loss function
Stacking	• Used to ensemble a diverse group of strong learners • Involves training a second-level machine learning algorithm called a "metalearner" to learn the optimal combination of the base learners

H2O WORLD

https://www.slideshare.net/0xdata/h2o-world-ensembles-with-erin-ledell

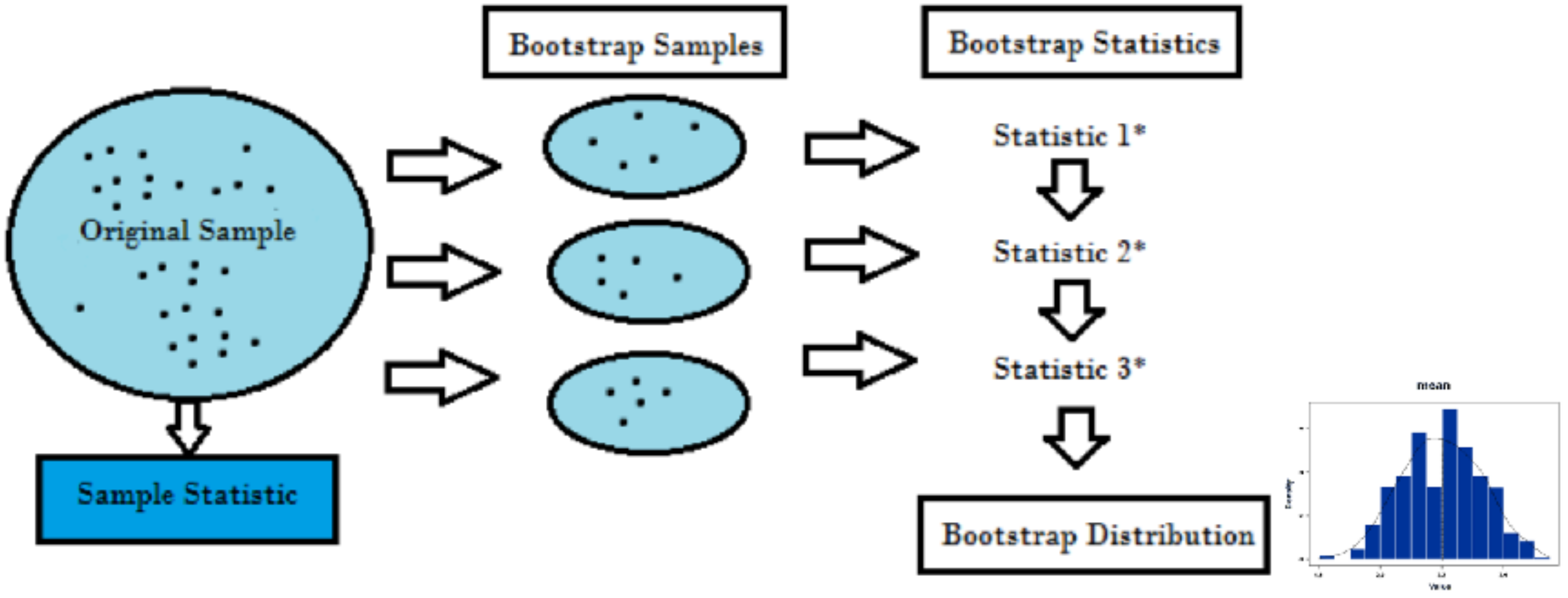
Making Predictions

- **Classification**
  - Majority Voting: Every model makes a prediction (votes) for each test instance and the final output prediction is the one that receives more than half of the votes.
  - Weighted Voting: In weighted voting you count the prediction of the better models multiple times.
- **Regression**
  - Simple Average: In simple averaging method, for every instance of test dataset, the average predictions are calculated.
  - Weighted Average: The prediction of each model is multiplied by the weight and then their average is calculated.

Bagging

Bootstrap Overview

- Bootstrapping is loosely based on the law of large numbers, which states that if you sample over and over again, your data should approximate the true population data



http://blog.minitab.com/blog/the-statistics-game/understanding-bootstrapping-and-the-central-limit-theorem

Bagging (Bootstrap Aggregation) Overview

- Can be used to *reduce the variance* for those algorithm that have high variance.
- Operates via equal weighting of models.
- Employs multiple instances of same classifier for one dataset.
- Builds models of smaller datasets by sampling with replacement.
- Works best when classifier is **unstable** (decision trees, for example), as this instability creates models of differing accuracy and results to draw majority from.

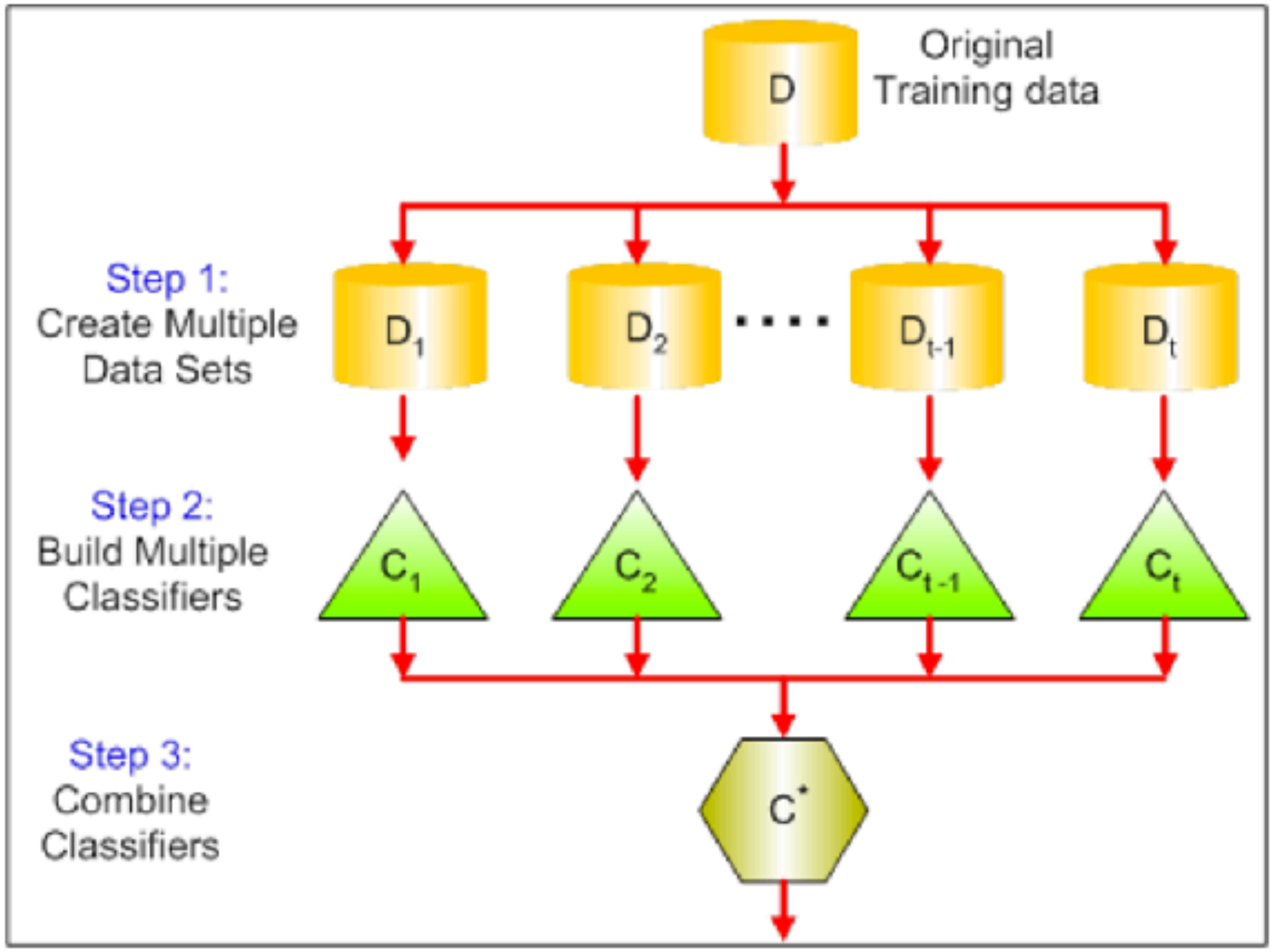
Bagging - How it works

- Step 1: Create random samples of the training data set (sub-sets of training data)
- Step 2: Build a classifier for each sample.
- Step 3: Results of the classifiers are combined using **simple average** (regression) or **majority voting** (classification).
- Most common: Random Forest

Why does bagging work ?

- Main reason for error in learning is due to noise ,bias and variance.
- Noise is error by the target function
- Bias is where the algorithm can not learn the target.
- Variance comes from the sampling, and how it affects the learning algorithm
- Does bagging minimizes these errors ?
- Yes
- Averaging over bootstrap samples can reduce error from variance especially in case of unstable classifiers

Bagging - How it works



Boosting

Boosting Overview

- The term “Boosting” refers to a family of algorithms which converts weak learners to strong learners
- To convert a weak learner (a base learner) to a strong one, we’ll combine the prediction of each weak learner using methods like:
  - Using **average/ weighted average** (regression)
  - Considering prediction has **higher vote** (classification)

Boosting - How it works

- Step 1: The base learner takes all the distributions and assigns equal weight or attention to each observation
- Step 2: If there is any prediction error caused by the first base learning algorithm, then we pay higher attention to observations having prediction error. Then we apply the next base learning algorithm.
- Step 3: Iterate step 2 till the limit of base learning algorithm is reached or higher accuracy is achieved.

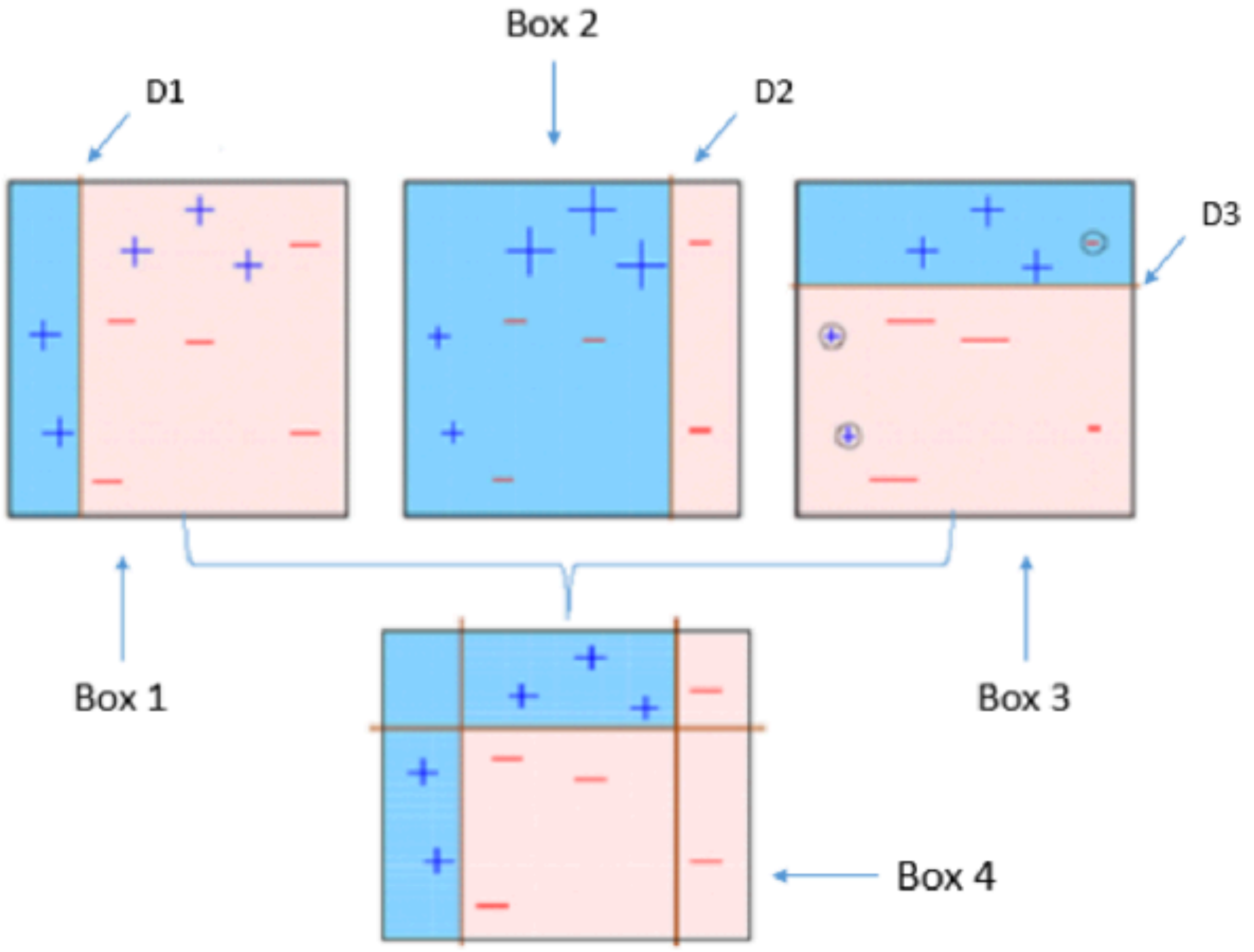
Types of Boosting Algorithms

- AdaBoost (Adaptive Boosting)
- Gradient Tree Boosting
- XGBoost

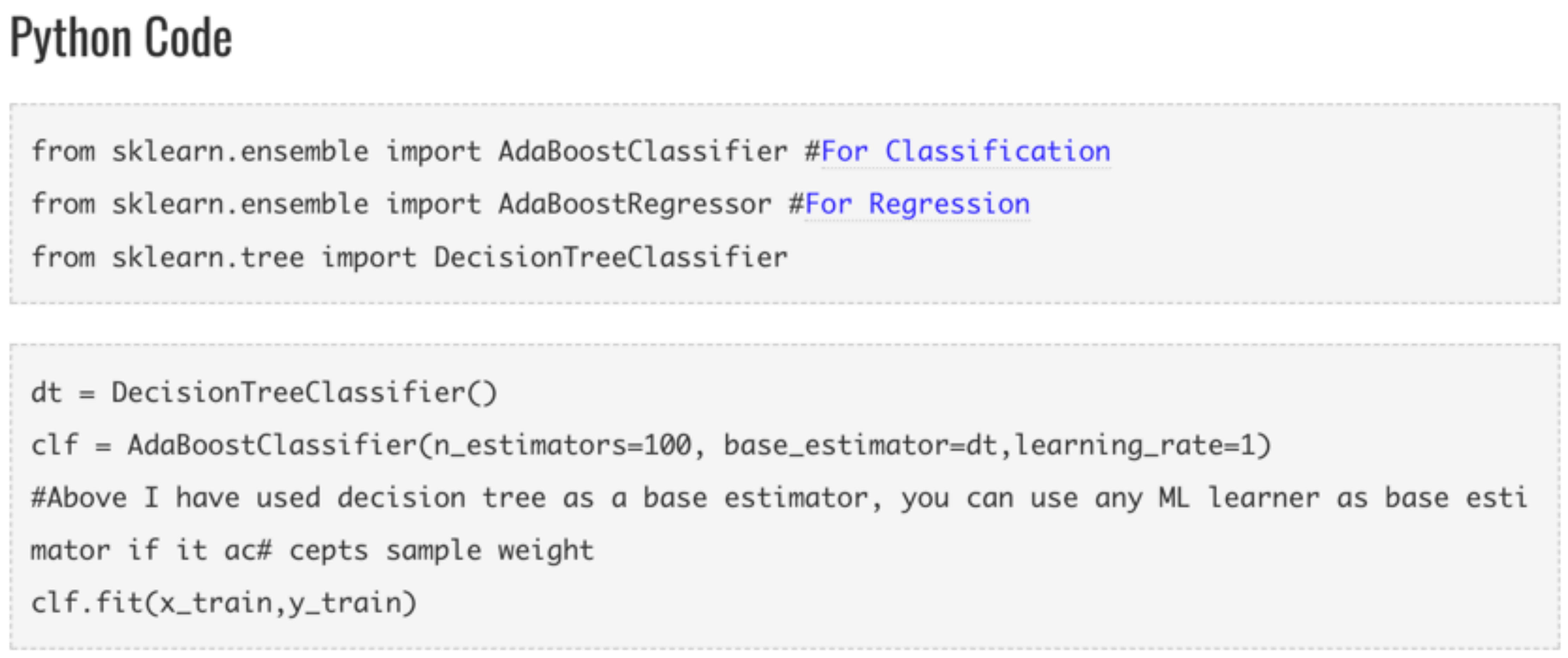
https://www.analyticsvidhya.com/blog/2015/11/quick-introduction-boosting-algorithms-machine-learning/

AdaBoost

AdaBoost Explained



AdaBoost in Python



Python Code

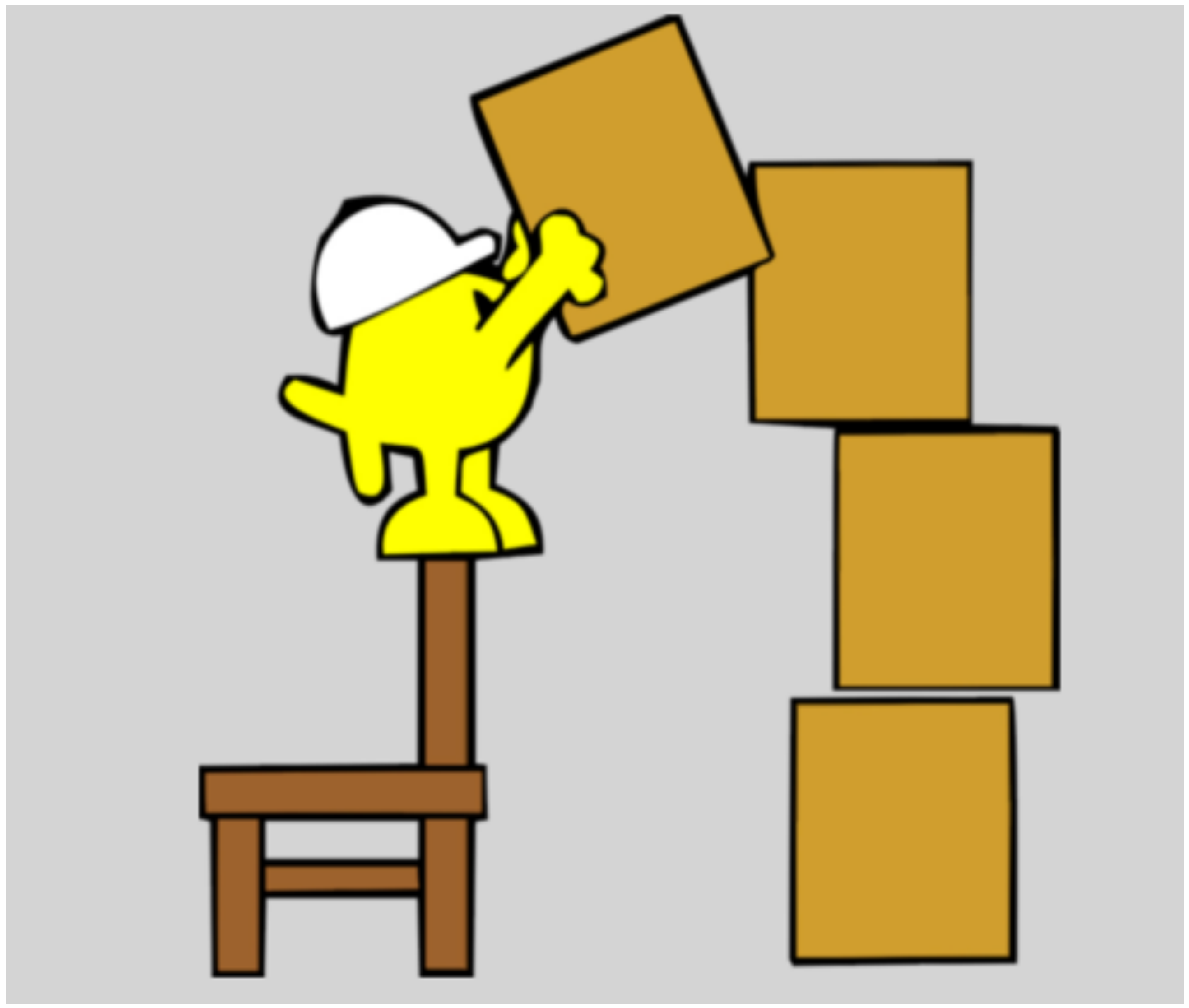
```
from sklearn.ensemble import AdaBoostClassifier #For Classification
from sklearn.ensemble import AdaBoostRegressor #For Regression
from sklearn.tree import DecisionTreeClassifier
```

```
dt = DecisionTreeClassifier()
clf = AdaBoostClassifier(n_estimators=100, base_estimator=dt,learning_rate=1)
#Above I have used decision tree as a base estimator, you can use any ML learner as base esti
mator if it ac# cepts sample weight
clf.fit(x_train,y_train)
```

Tuning Parameters

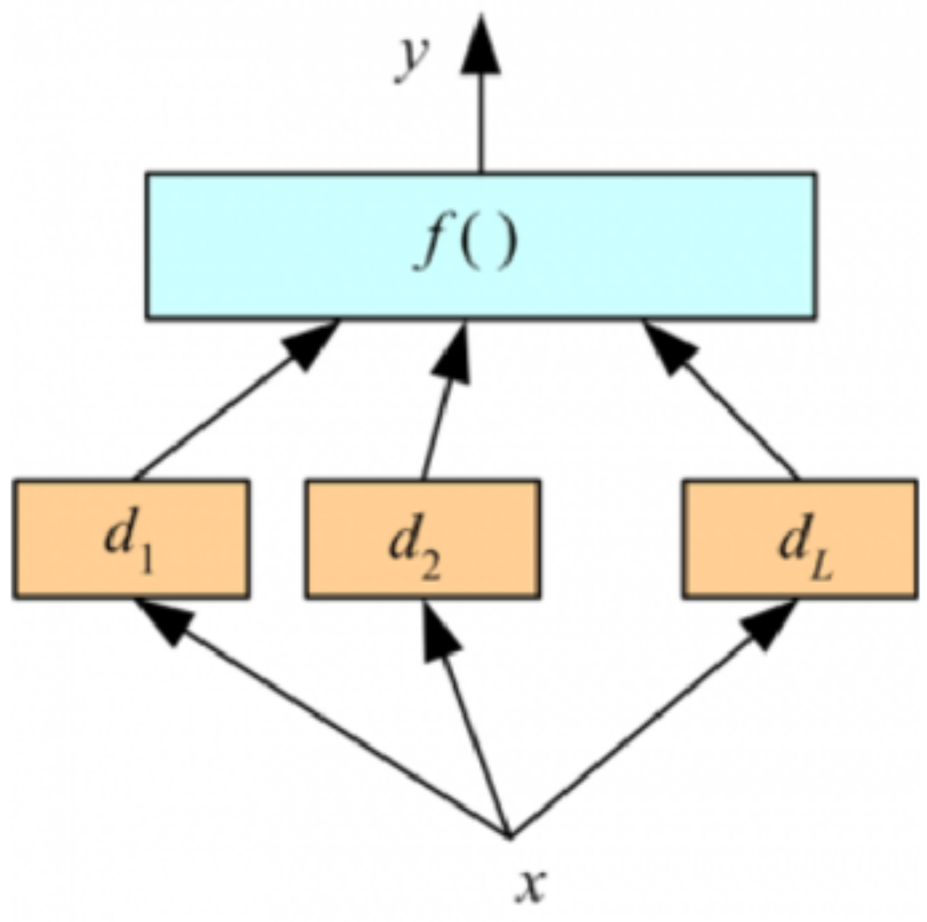
- n_estimators: controls the number of weak learners
- learning_rate: controls the contribution of weak learners in the final combination. There is a trade-off between learning_rate and n_estimators
- base_estimators: helps to specify different ml algorithm (default is decision tree)

Stacking



Stacking Overview

- Method used to combine information from multiple predictive models to form a new model. Using the **predictions** of the base models as **features** (i.e. meta features) for the stacked model.

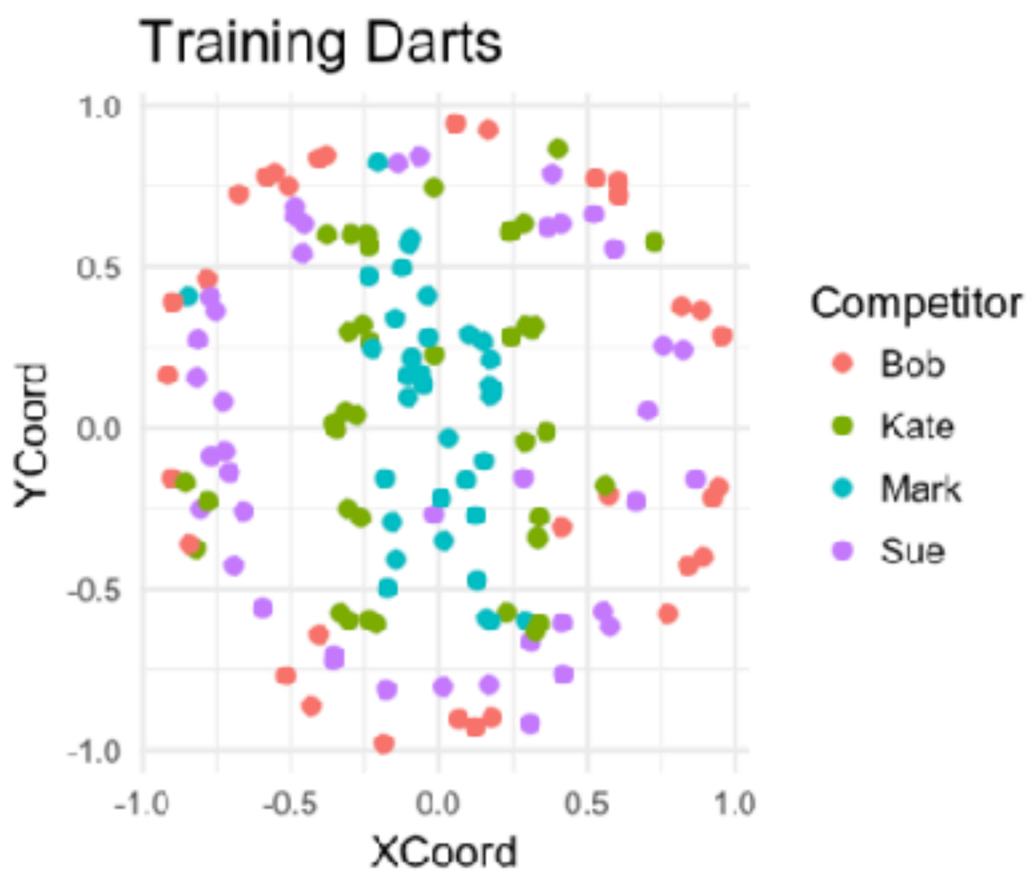


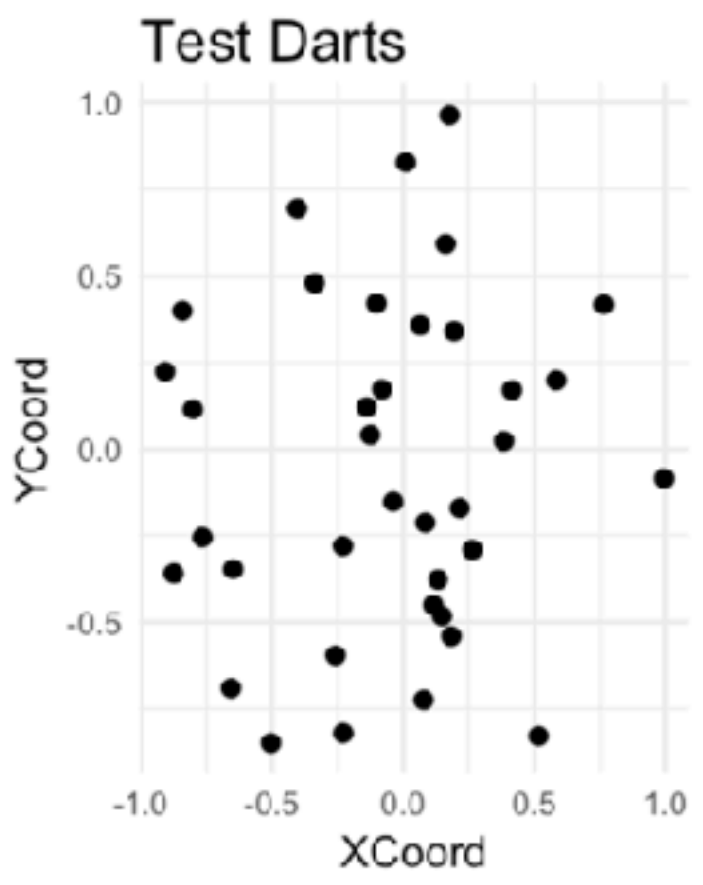
Stacking - How it works

- Step 1: Use multiple (diverse) base classifiers to predict the class
- Step 2: A new learner is used to combine their predictions

Stacking Explained - 1

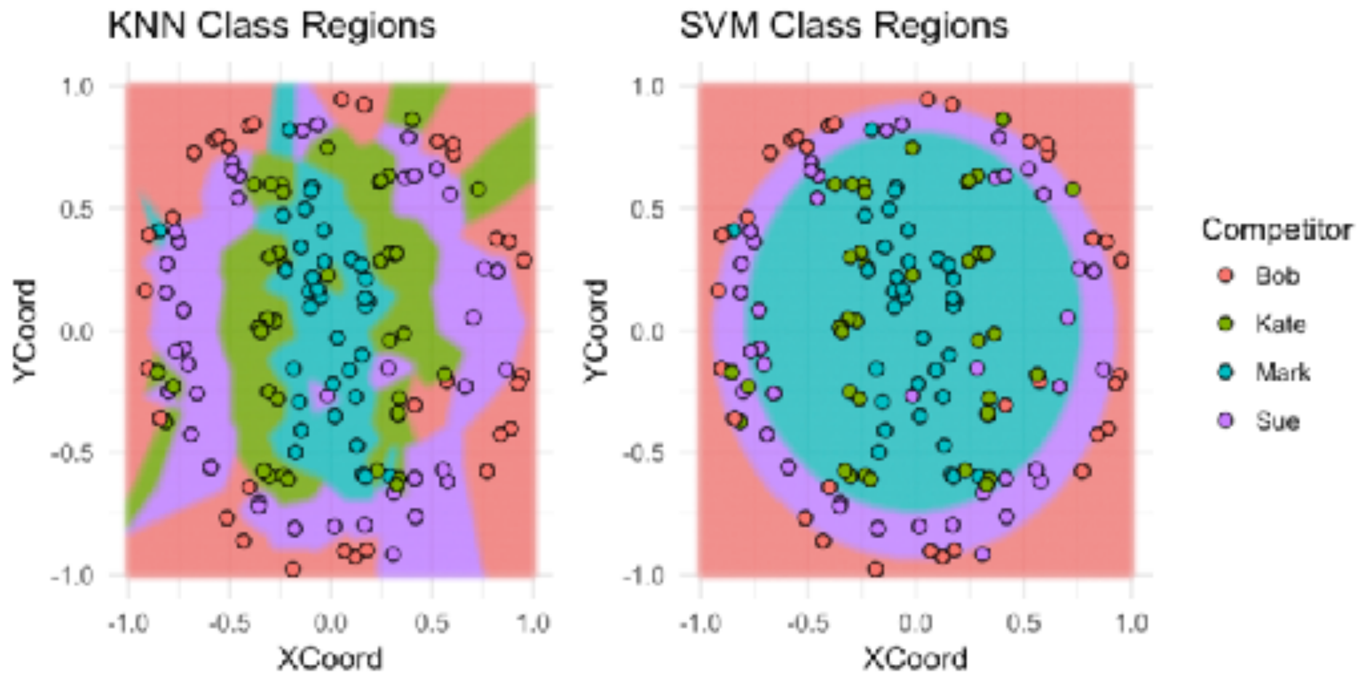
Example: Suppose four people throw a combined 187 darts at a board. For 150 of those we get to see who threw each dart and where it landed. For the rest, we only get to see where the dart landed. Our task is to guess who threw each of the unlabelled darts based on their landing spot.

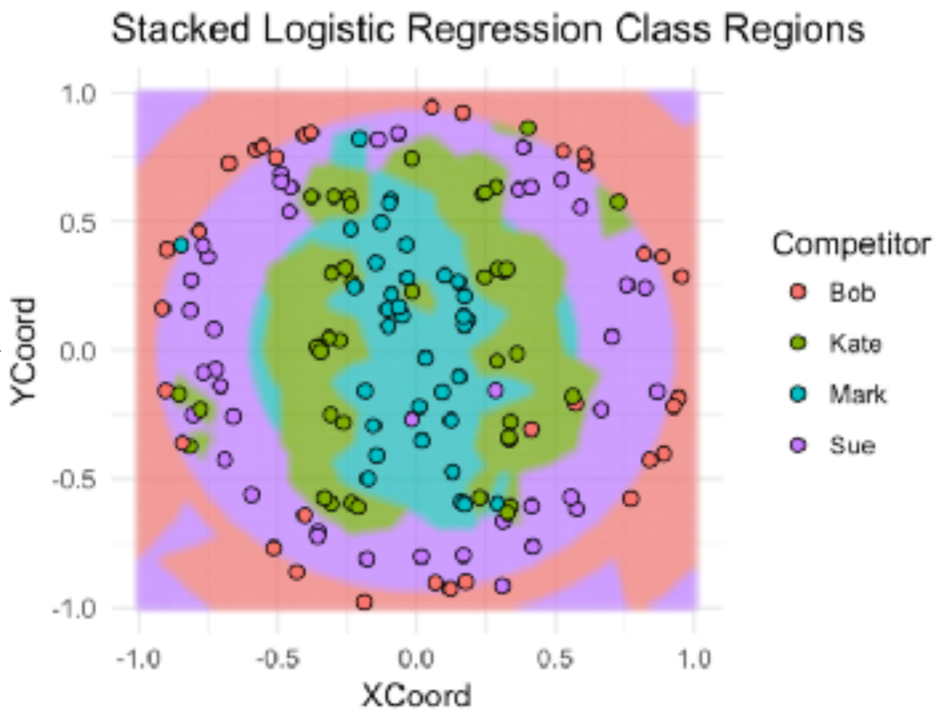




Stacking Explained - 2

ID	XCoord	YCoord	DistFromCenter	M1	M2	Pred	Competitor
6	0.06	0.36	0.36	Mark	Mark	Mark	Mark
12	-0.77	-0.26	0.81	Kate	Sue	Sue	Sue
22	0.18	-0.54	0.57	Mark	Sue	Mark	Mark
...	...	...	...	...	...	...	...
178	0.01	0.83	0.83	Sue	Sue	Sue	Sue
184	0.58	0.2	0.62	Sue	Mark	Sue	Sue
185	0.11	-0.45	0.46	Mark	Mark	Mark	Mark





Rules of Thumb

- The key to creating a powerful ensemble is **model diversity**.
    - An ensemble with two techniques that are very similar in nature will perform poorly than a more diverse model set.Better predictions typically result when combining predictions generated by **different** ML algorithms (rather than many predictions from the same algorithm). For example, the predictions of a random forest, a KNN, and a Naive Bayes may be combined to create a stronger final prediction set as compared to combining three random forest model.

Challenges

- **How do we determine the optimal weights for base models in an ensemble?** In general, we assume equal weights for all models and take the average of the predictions. But is this the best way to deal with this challenge? Other methods are:
    - Find the collinearity between base learners and based on this table, determine the base models to ensemble. After that look at the cross-validation score (ratio of score) of identified base models to find the weights.
    - Find the algorithm to return the optimal weight for base learners.
    - We can also use methods like:
        A. Forward Selection of learners
        B. Selection with replacement
        C. Bagging of Ensemble methods

https://www.analyticsvidhya.com/blog/2015/09/questions-ensemble-modeling/

Practice

1. Go to: https://github.com/snooravi/meetups
2. Clone/download the directory to your local computer
3. Open jupyter notebook (run ‘jupyter notebook’ in CLI)
4. Navigate to: ensemble_methods/
   Bagging%20using%20Sonar%20Data.ipynb

More Resources

1. Bagging from Scratch: https://machinelearningmastery.com/implement-bagging-scratch-python/
2. Bootstrapping and CLT: http://blog.minitab.com/blog/the-statistics-game/understanding-bootstrapping-and-the-central-limit-theorem
3. Ensemble ML In Python and scikit learn: https://machinelearningmastery.com/ensemble-machine-learning-algorithms-python-scikit-learn/
4. Kaggle Ensemble Guide: https://mlwave.com/kaggle-ensembling-guide/
5. http://www.kdnuggets.com/2016/11/data-science-basics-intro-ensemble-learners.html
6. Basics of Ensemble Learning in Plain English: https://www.analyticsvidhya.com/blog/2015/08/introduction-ensemble-learning/
7. Stacking: http://blog.kaggle.com/2016/12/27/a-kagglers-guide-to-model-stacking-in-practice/
8. 5 Questions on Ensembles everyone should know: https://www.analyticsvidhya.com/blog/2015/09/questions-ensemble-modeling/