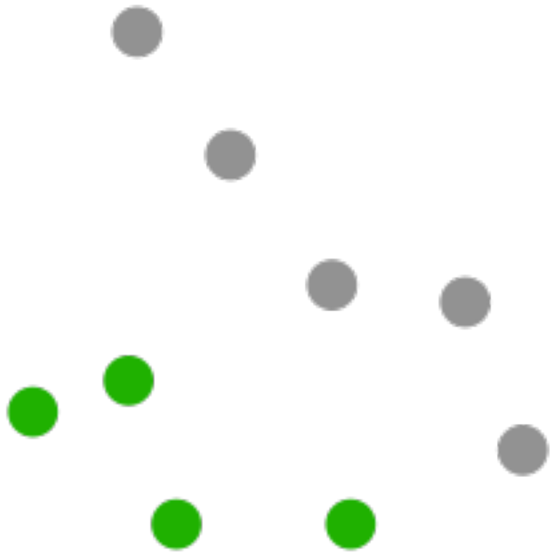
Recap

Families of ML algorithms

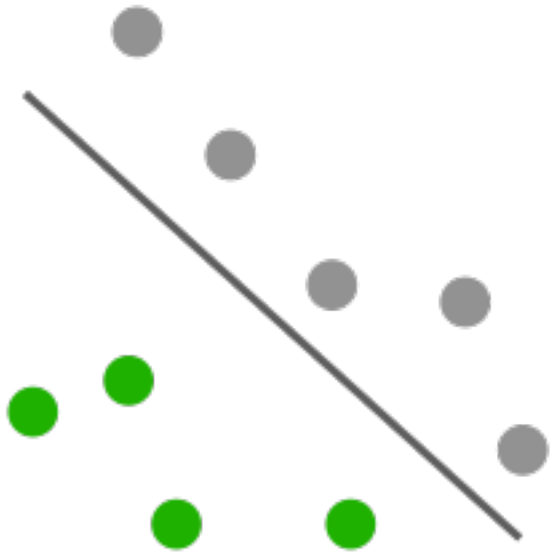
- Linear
- Tree-based
- kNN
- Neural Networks

Linear model

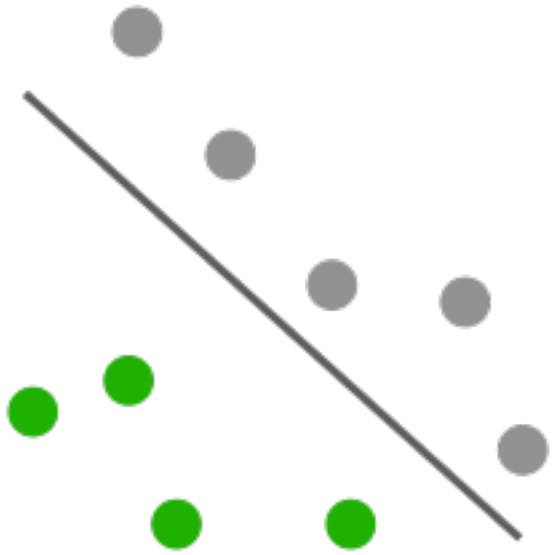
Linear model



Linear model



Linear model

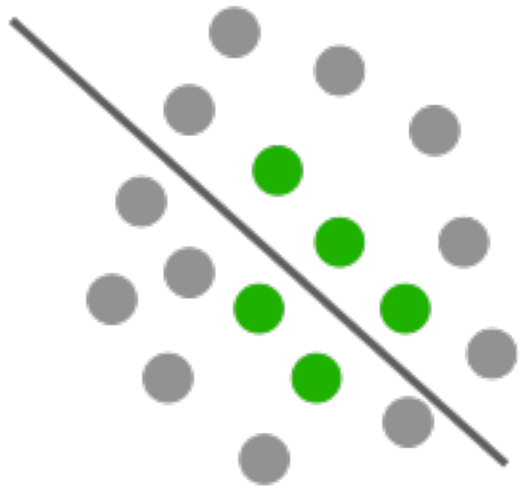


Examples:

- Logistic Regression
- Support Vector Machines

Linear model

I want to emphasize that Linear Models are especially good for sparse high dimensional data. But you should keep in mind the limitations of Linear Models.



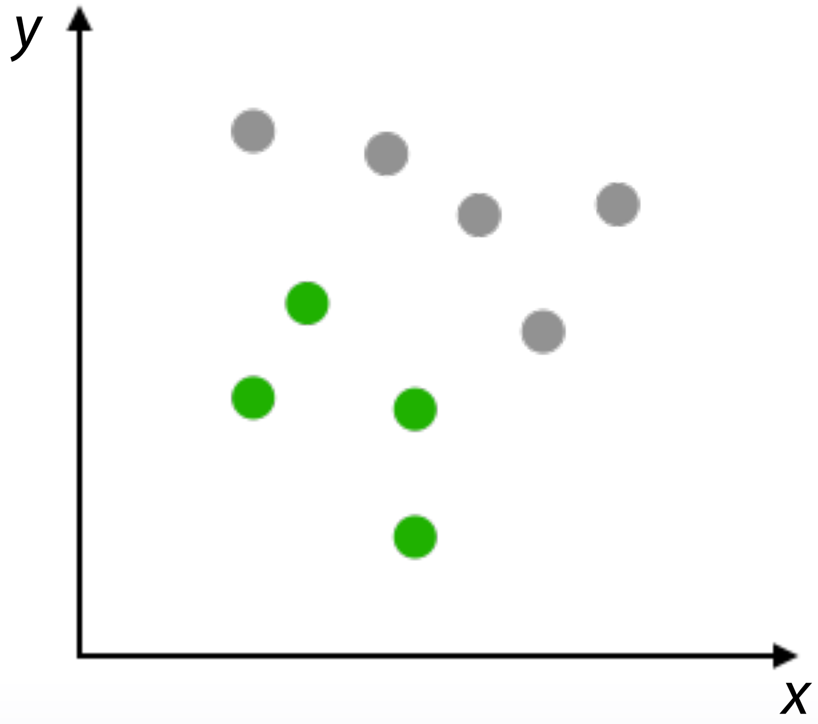
As an example, you can imagine two sets of points that form rings, one inside the other. Although it's pretty obvious how to separate them, Linear Models are not an appropriate choice either and will fail in this case.

Linear model

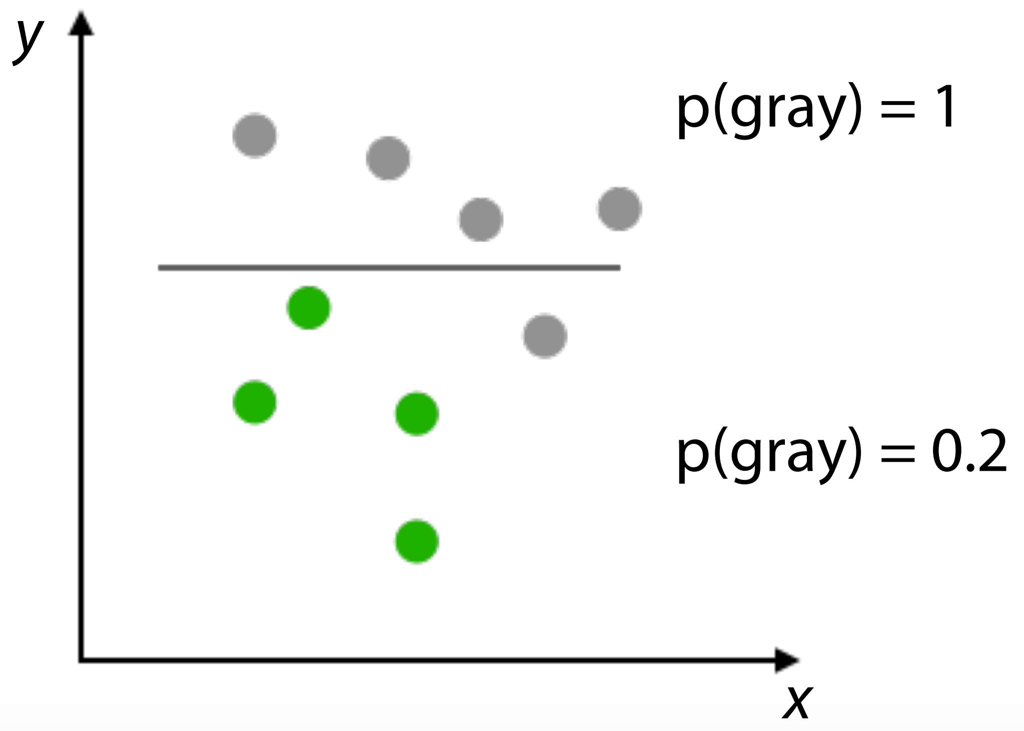
Tree-based: Decision Tree, Random Forest, GBDT

Tree-Based Methods use decision tree as a basic block for building more complicated models.

Tree-based: Decision Tree, Random Forest, GBDT

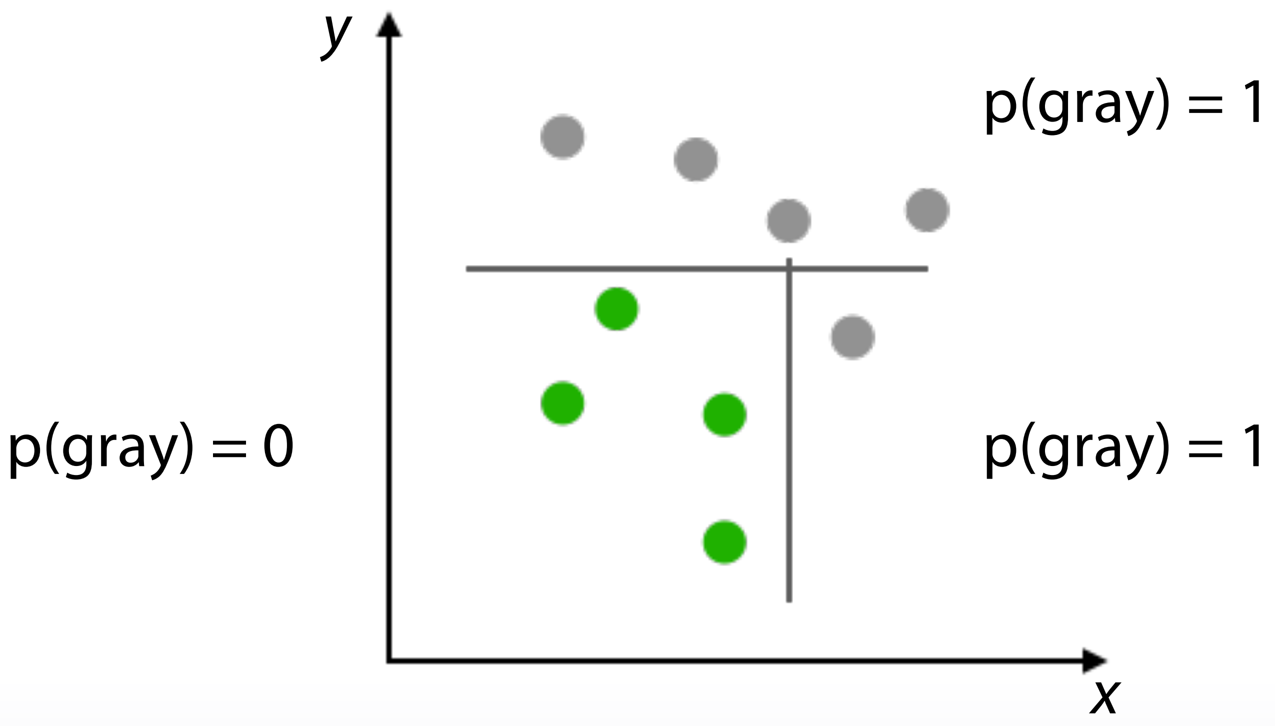


Tree-based: Decision Tree, Random Forest, GBDT

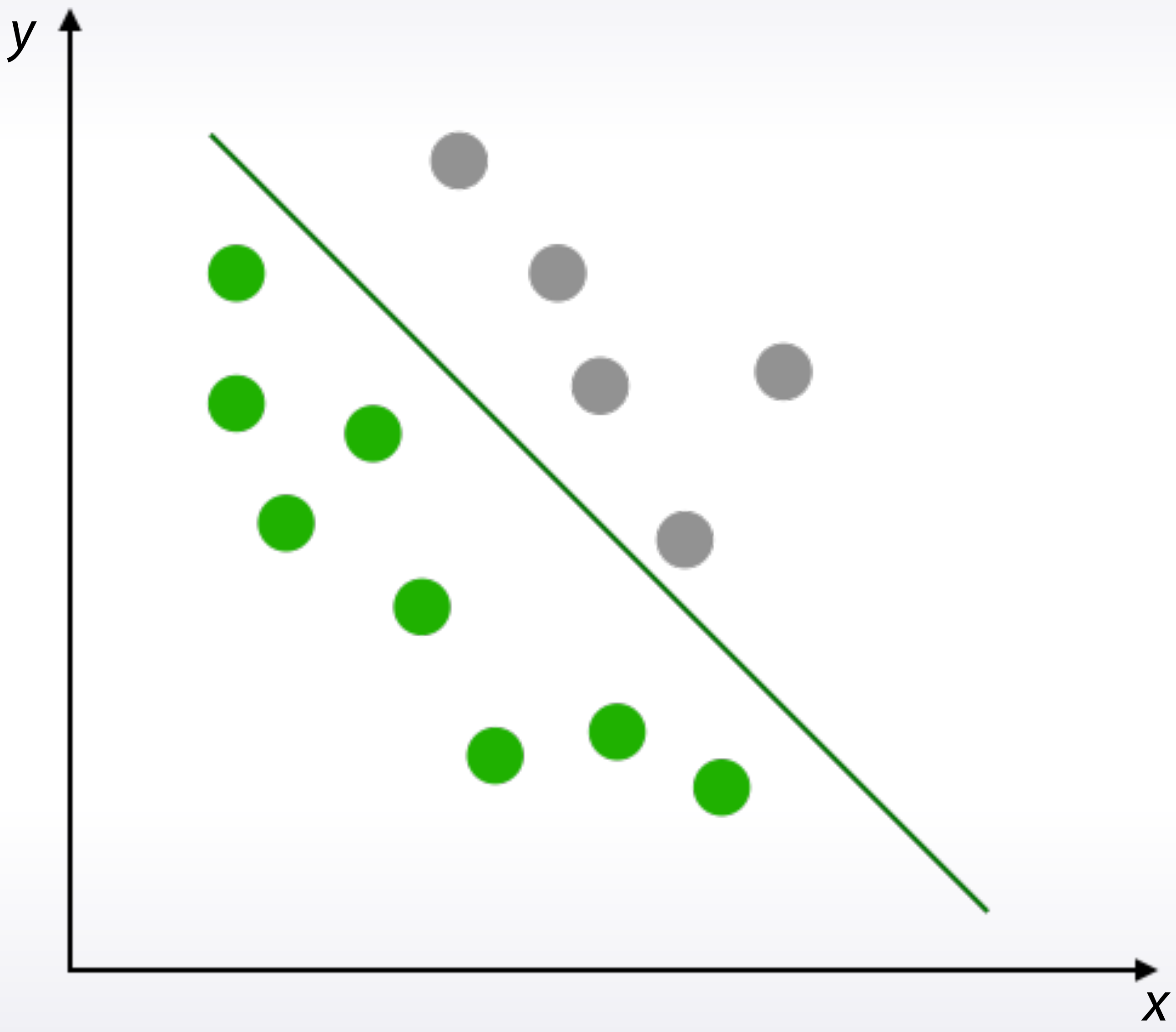


Tree-based: Decision Tree, Random Forest, GBDT

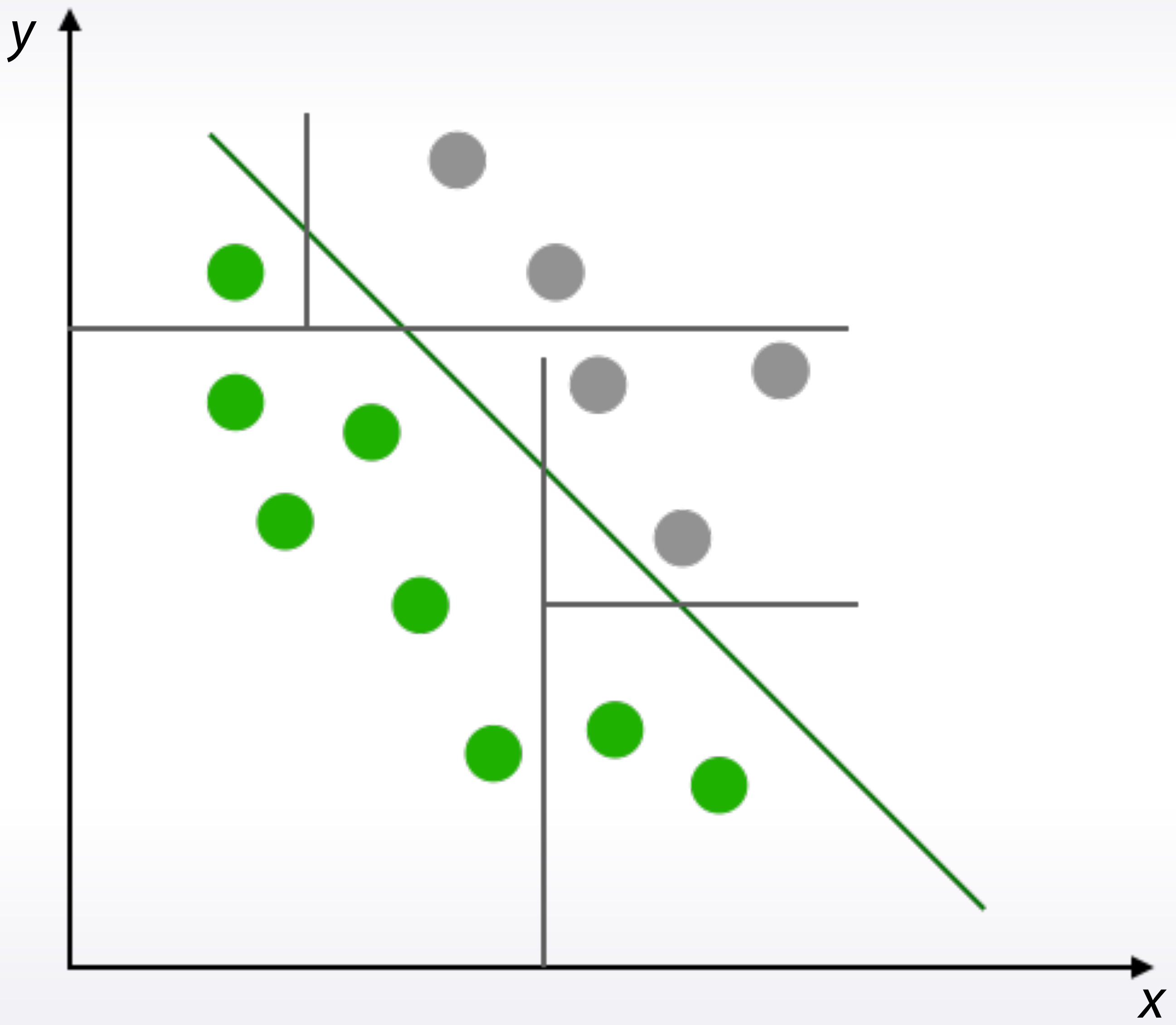
In general, tree-based models are very powerful and can be a good default method for tabular data. In almost every competitions, winners use this approach. But keep in mind that for Tree-Based Methods, it's hard to capture linear dependencies since it requires a lot of splits.



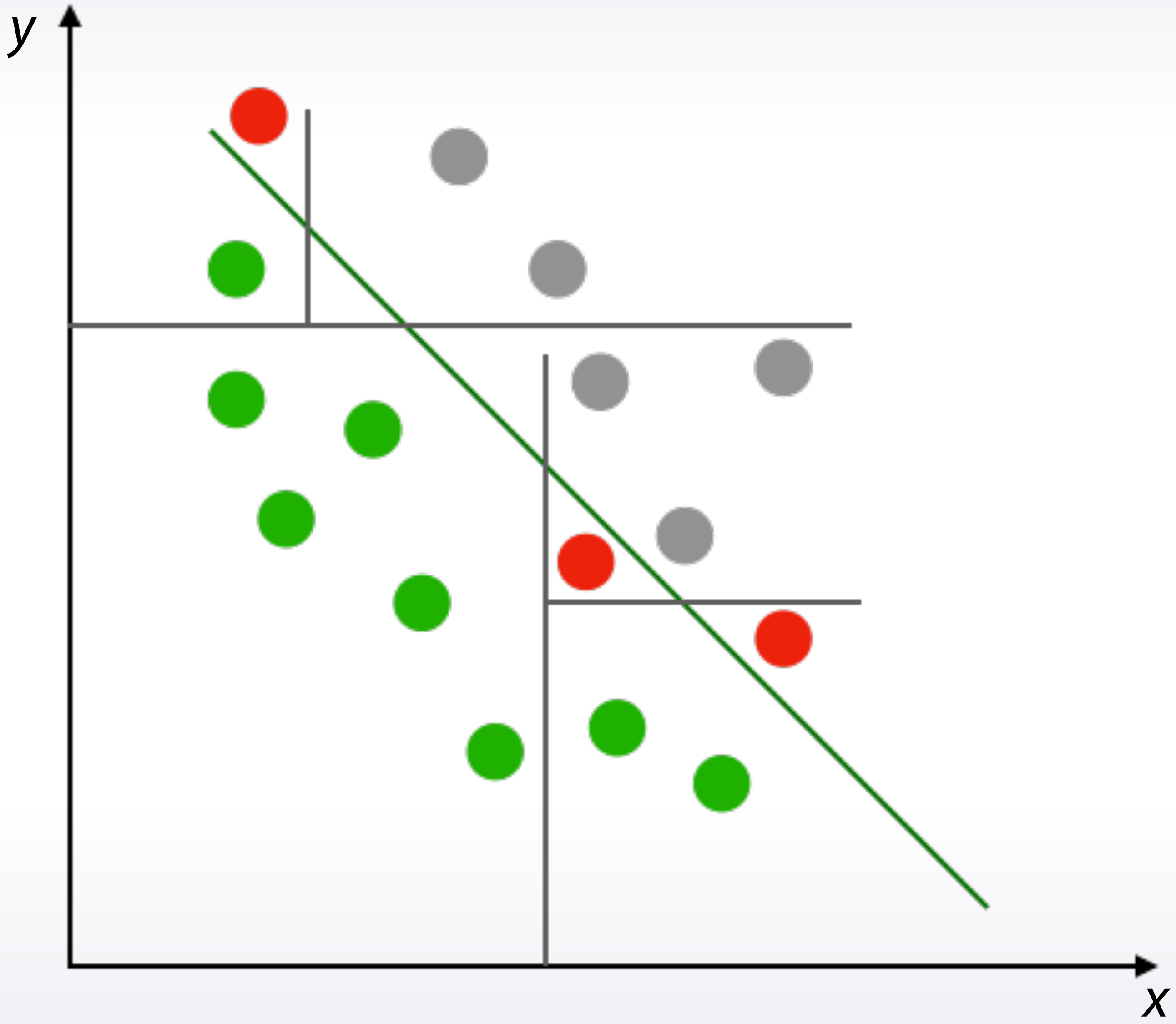
Tree-based: Decision Tree, Random Forest, GBDT



Tree-based: Decision Tree, Random Forest, GBDT



Tree-based: Decision Tree, Random Forest, GBDT



Tree-based methods

scikit learn

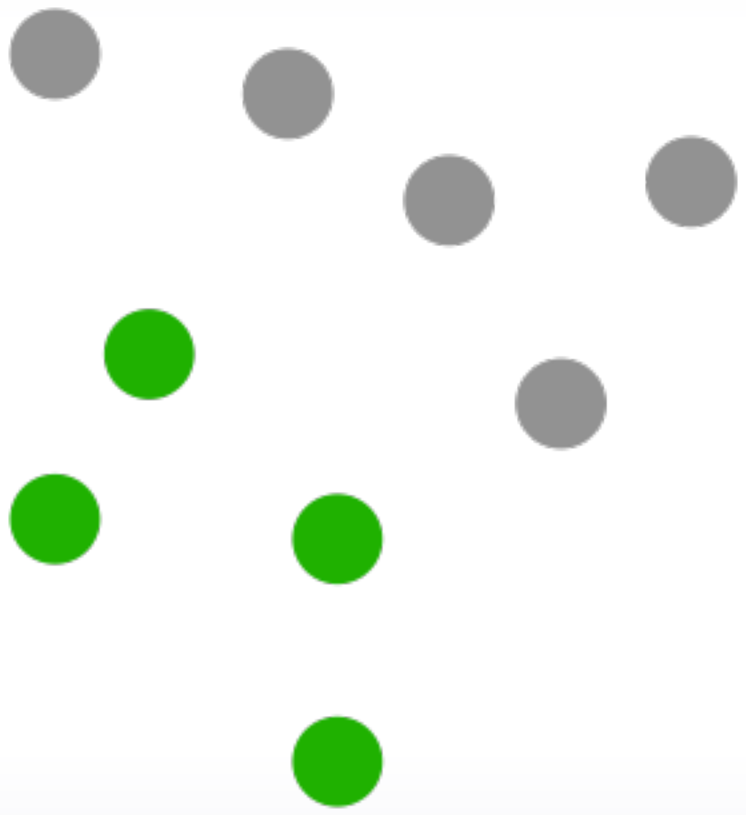
dmlc XGBoost

Microsoft / LightGBM

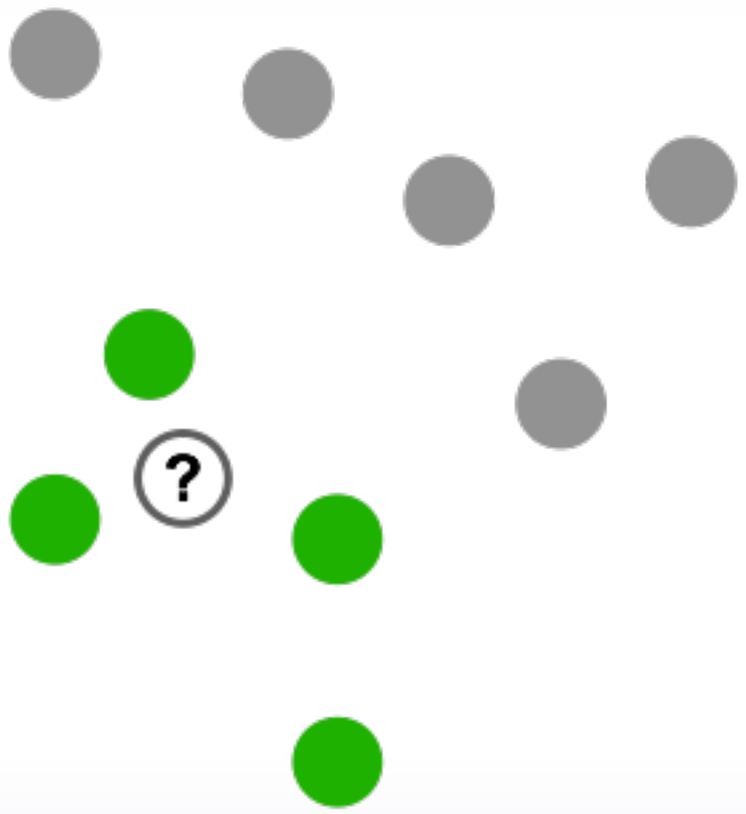
Scikit-Learn contains quite good implementation of random forest which I personally prefer. All the Scikit-Learn contain implementation of gradient boost decision trees. I prefer to use libraries like XGBoost and LightGBM for their higher speed and accuracy.

kNN-based methods

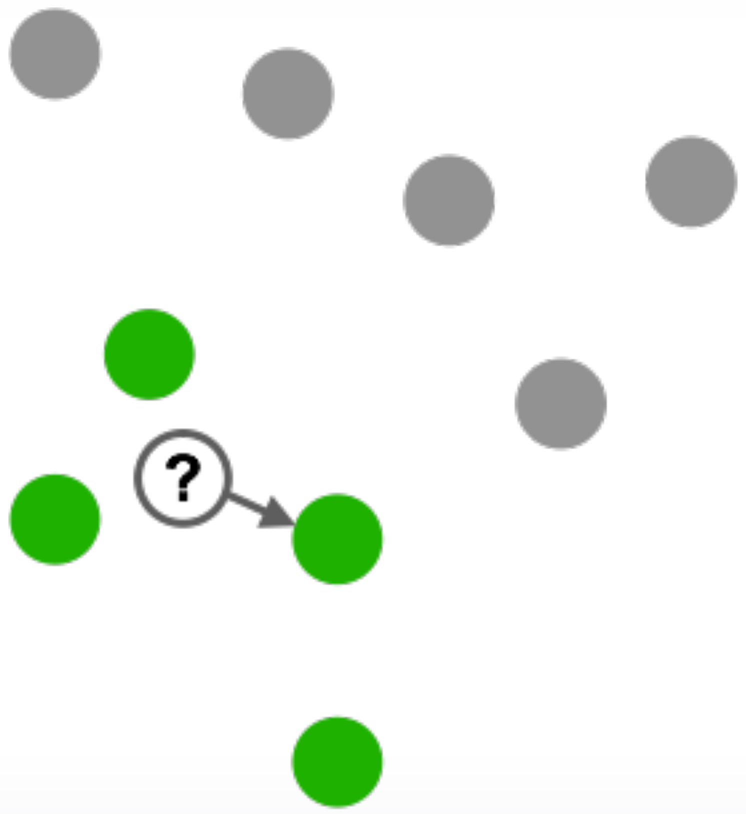
kNN-based methods



kNN-based methods

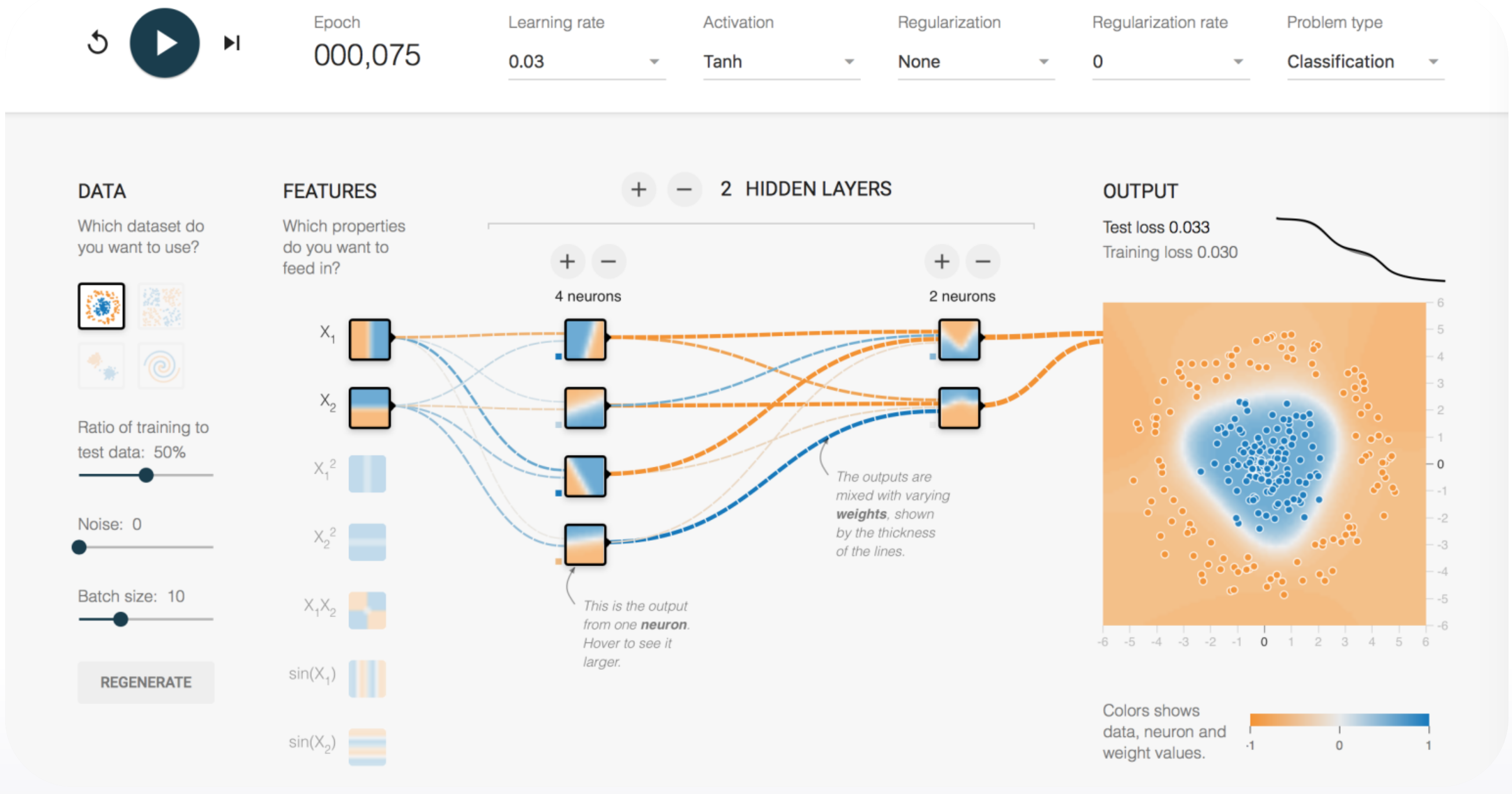


kNN-based methods



kNN-based methods

Neural Networks



Epoch 000,075

Learning rate 0.03

Activation Tanh

Regularization None

Regularization rate 0

Problem type Classification

DATA

Which dataset do you want to use?

Ratio of training to test data: 50%

Noise: 0

Batch size: 10

FEATURES

Which properties do you want to feed in?

2 HIDDEN LAYERS

4 neurons

2 neurons

*This is the output from one **neuron**. Hover to see it larger.*

*The outputs are mixed with varying **weights**, shown by the thickness of the lines.*

OUTPUT

Test loss 0.033

Training loss 0.030

Colors shows data, neuron and weight values.

Tensorflow Playground, http://playground.tensorflow.org

Neural Networks

TensorFlow

K

dmlc mxnet

PYTORCH

Lasagne

No Free Lunch Theorem

No Free Lunch Theorem

"Here is no method which **outperforms all others for all tasks**"

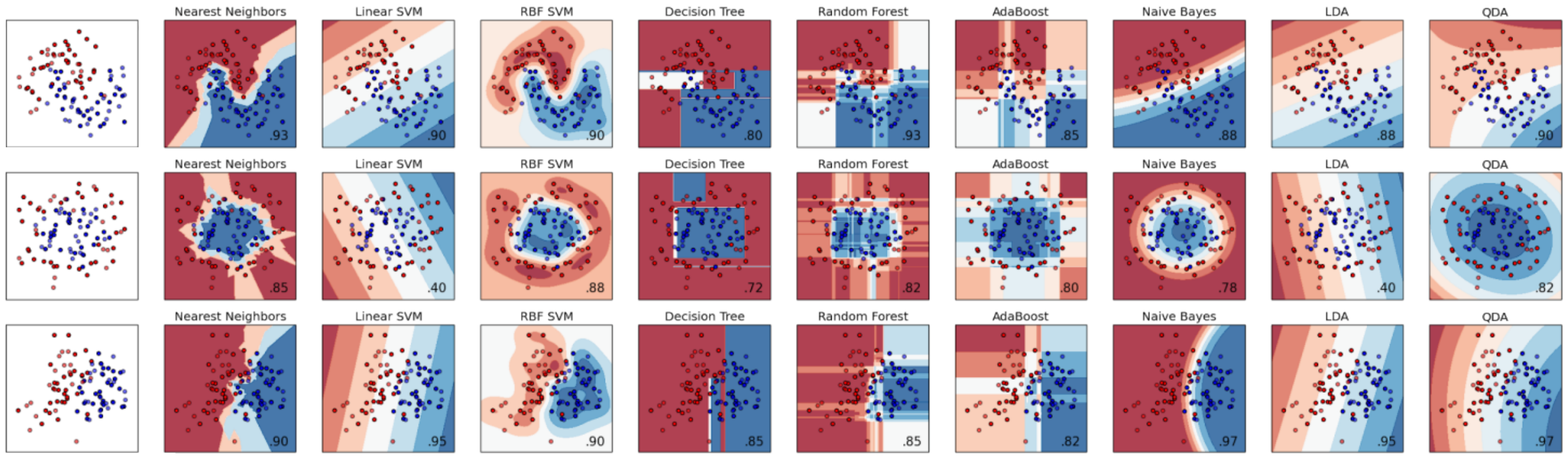
No Free Lunch Theorem

“Here is no method which **outperforms all others for all tasks**”

or

“For every method **we can construct a task** for which **this particular method will not be the best**”

Decision surfaces



Classifier comparison, http://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html

Conclusion

- There is no “silver bullet” algorithm
- Linear models split space into 2 subspaces
- Tree-based methods splits space into boxes
- k-NN methods heavy rely on how to measure points “closeness”
- Feed-forward NNs produce smooth non-linear decision boundary

The most powerful methods are
Gradient Boosted Decision Trees and **Neural Networks**.
But you shouldn’t underestimate the others