

Introduction to Gradient Boosting

Eoin Brazil, PhD, MSc
Proactive Technical Services, MongoDB

Github repo for this talk:

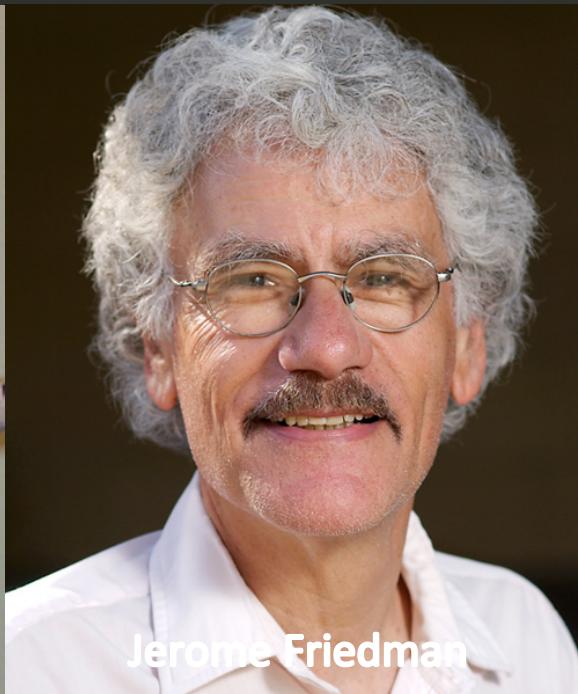
http://github.com/braz/pycon2016_talk/

From Theory
to Python
Libraries

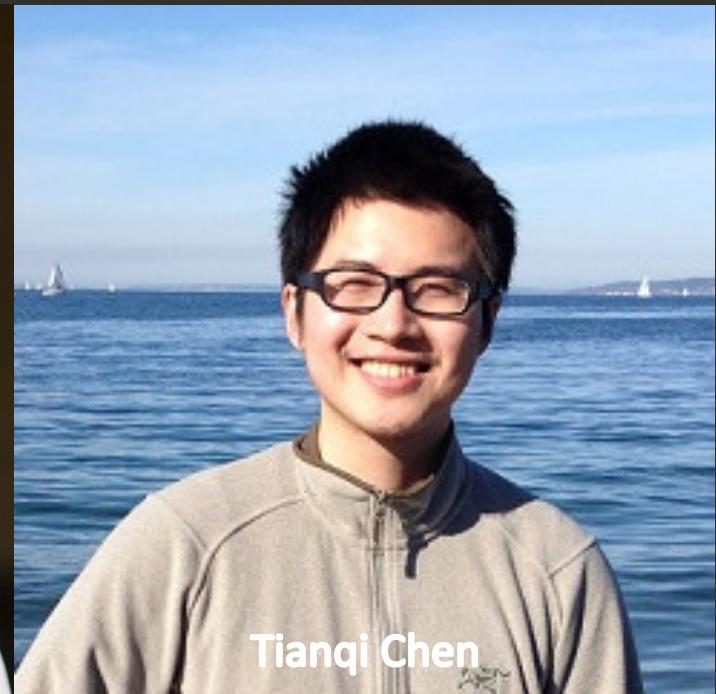
What this talk will cover



Leo Brieman

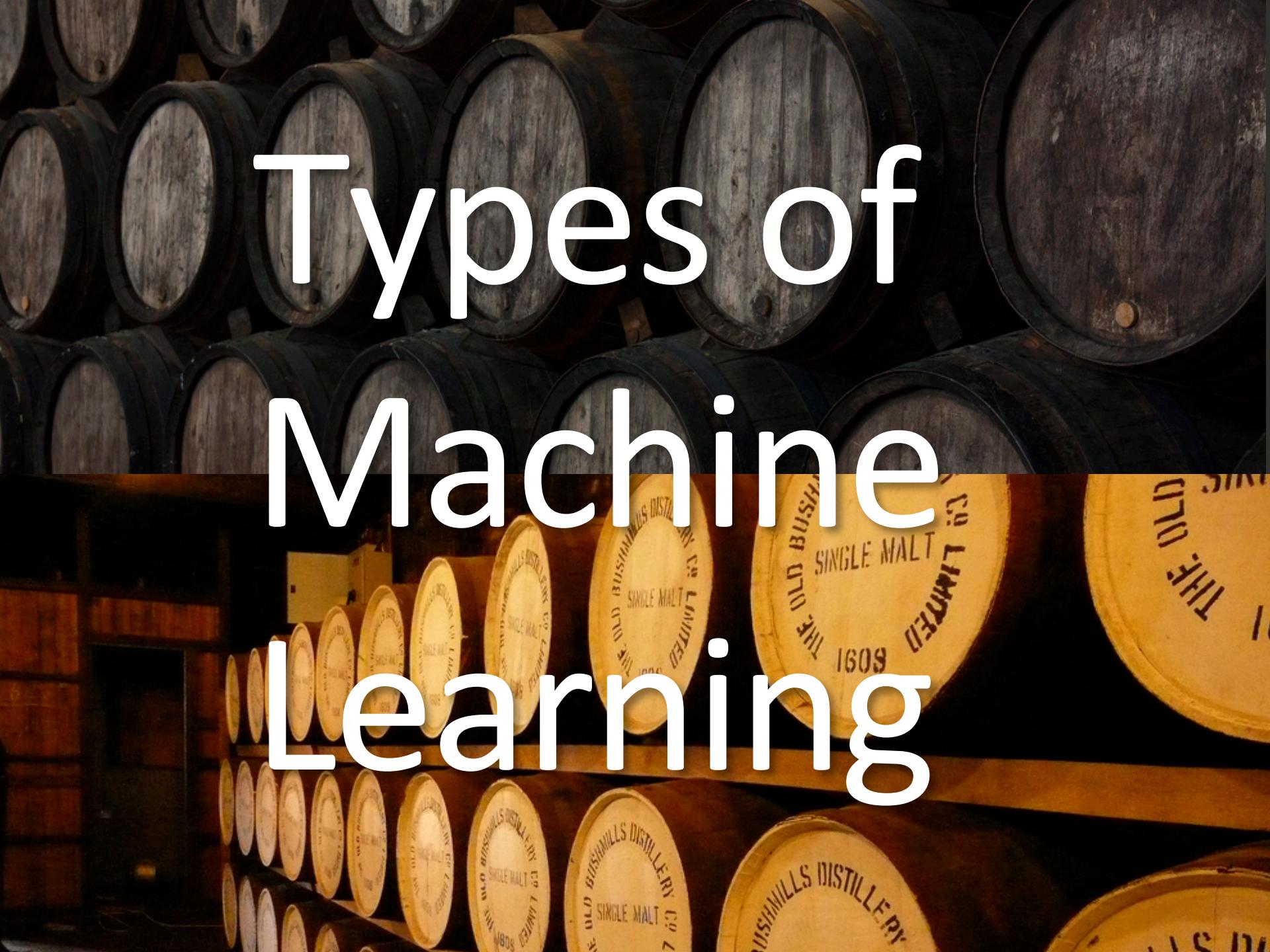


Jerome Friedman



Tianqi Chen

Types of Machine Learning



Iris Dataset

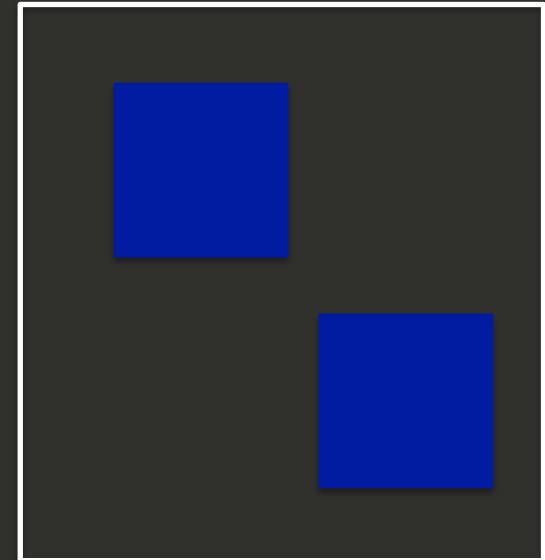
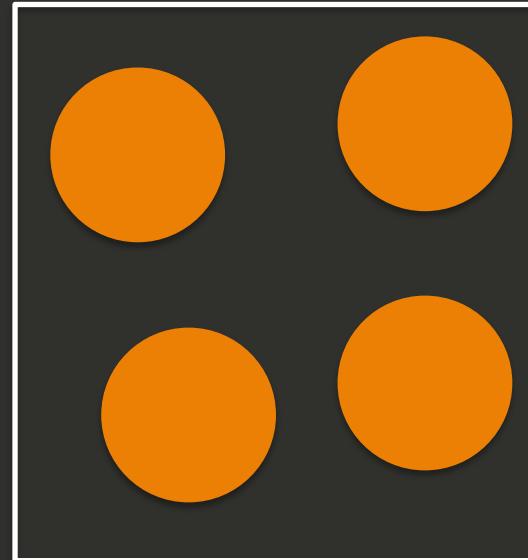
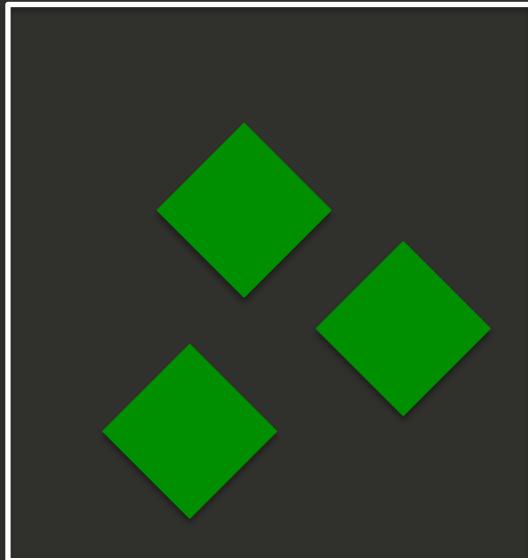


150 observations of 4
variable (sepal l/w
and petal l/w)

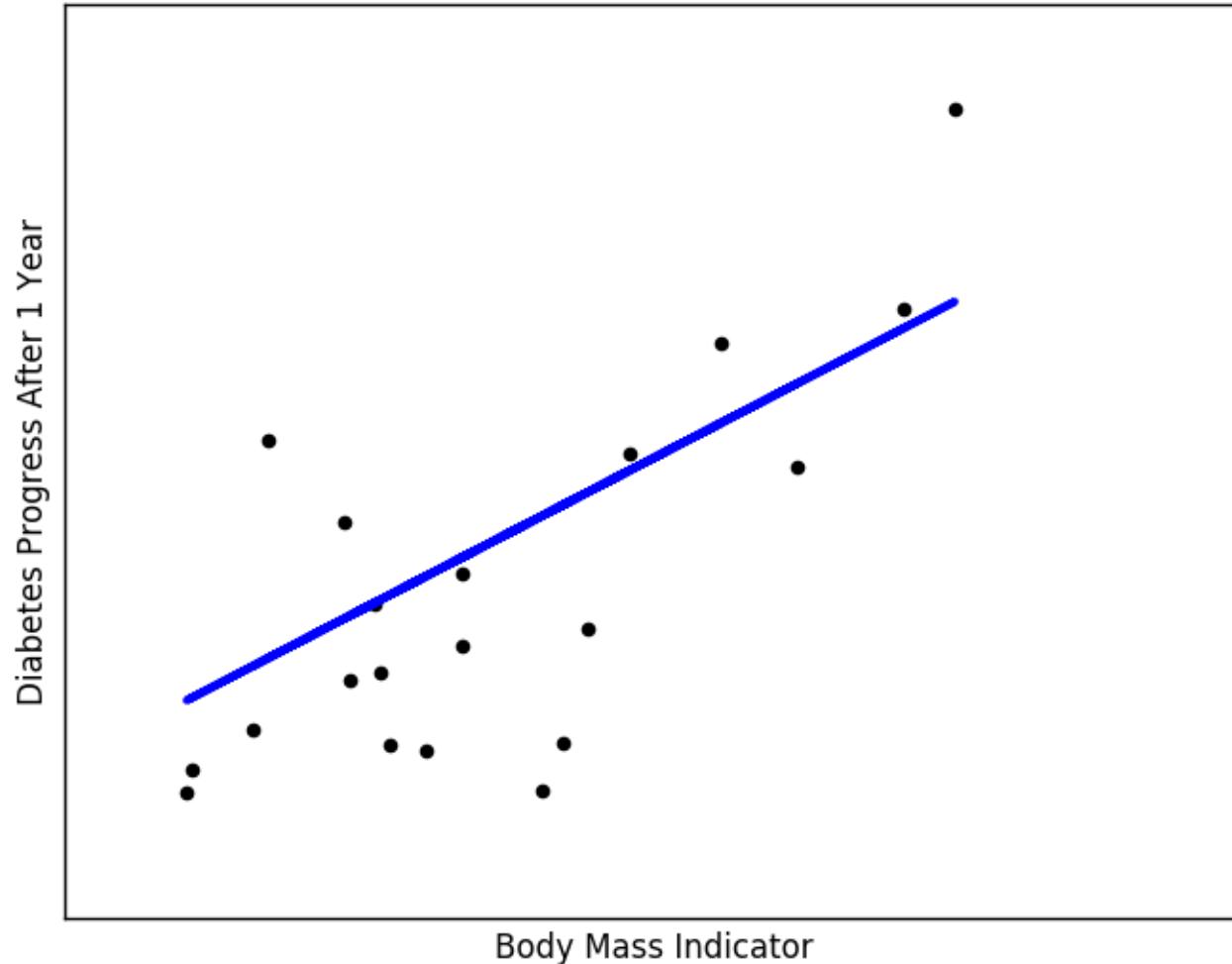
Supervised Techniques

- *Classification*
- Ranking
- Regression

Classification



Regression

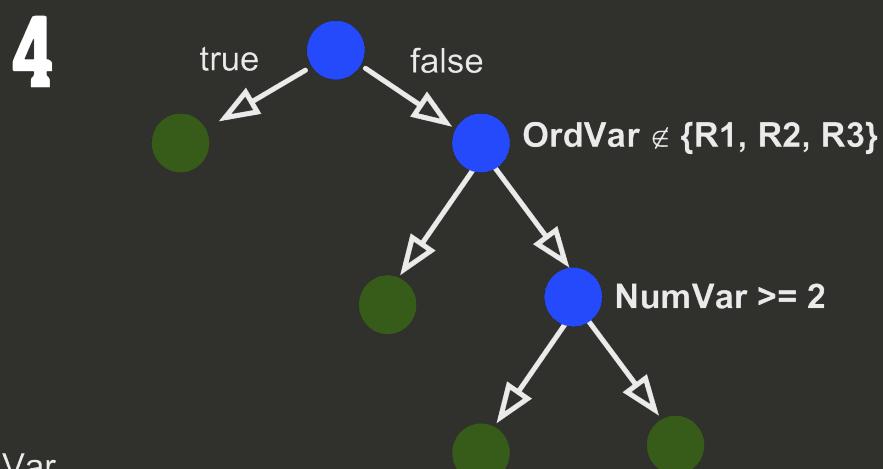
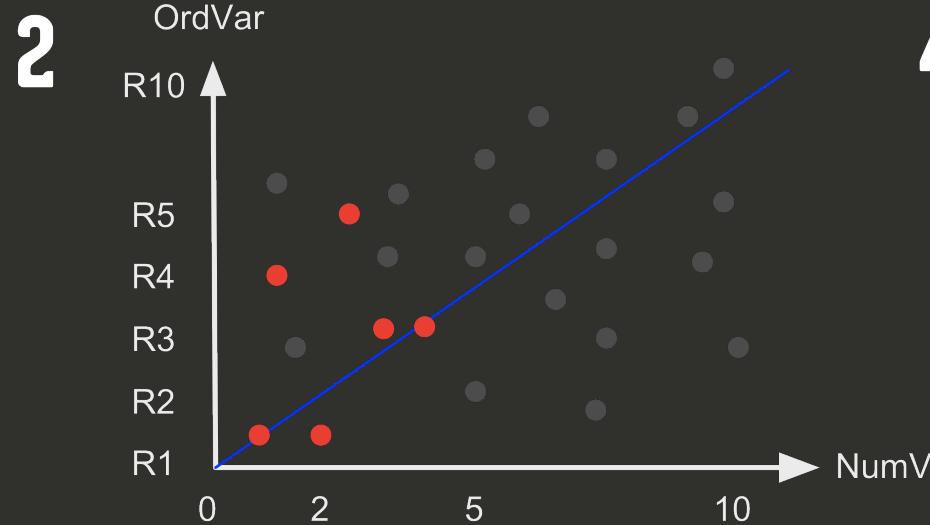
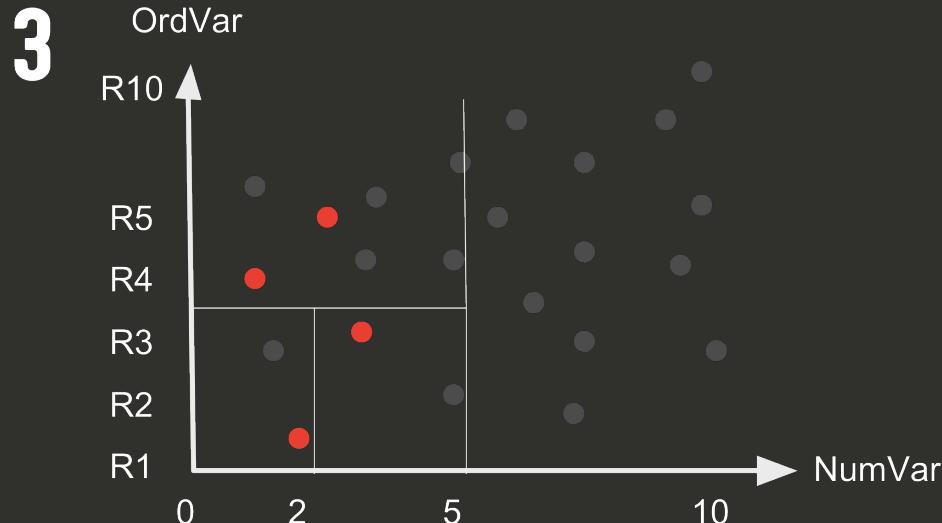


Context
or a bit of
history

Decision Trees

1

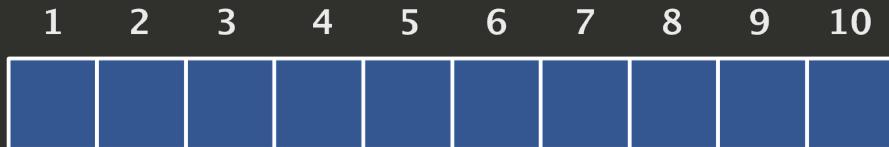
NumVar	OrdVar	CatVar
1	R2	Good
1.5	R4	Bad
2	R1	Bad
2.5	R5	Good
3	R3	Good
3.5	R3	Bad
⋮	⋮	⋮



Refining / Improving

- Bagging
- Boosting
- Random Forests

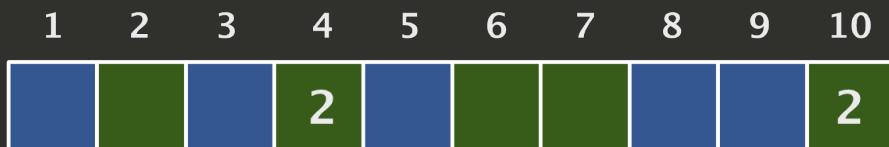
Bagging



Sample

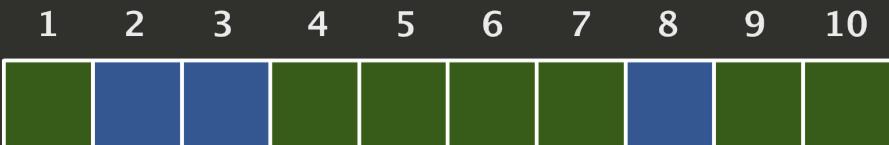


Not
Sampled



Samples the original training set uniformly with replacement to generate new bootstrapped training sets. This reduces variance and helps to avoid overfitting.

**Training set 1 –
Bootstrap samples 7**



**Training set 2 –
Bootstrap samples 7**

⋮



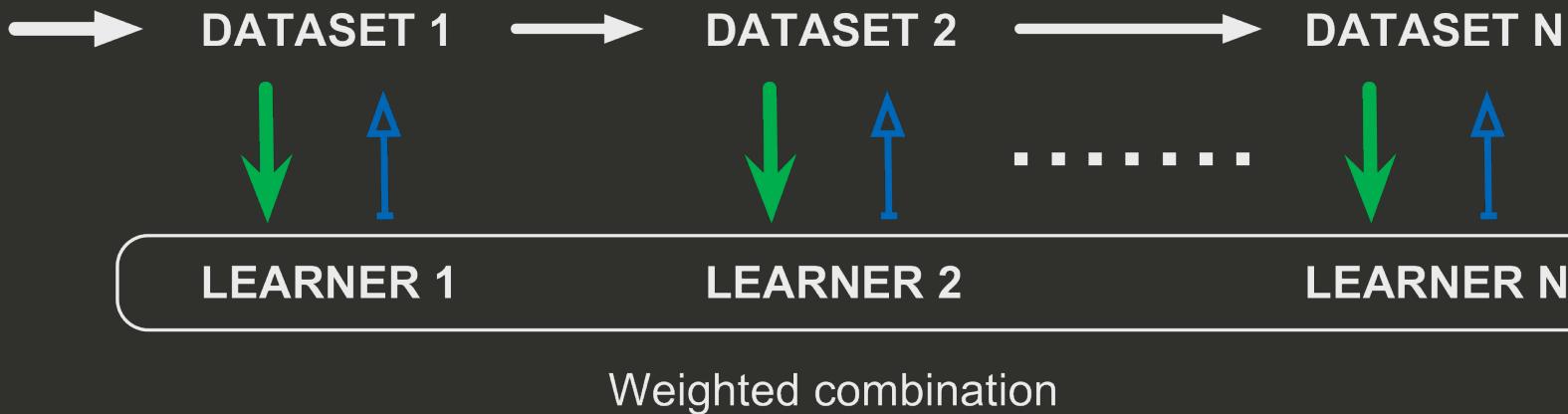
**Training set N –
Bootstrap samples 7**

⋮

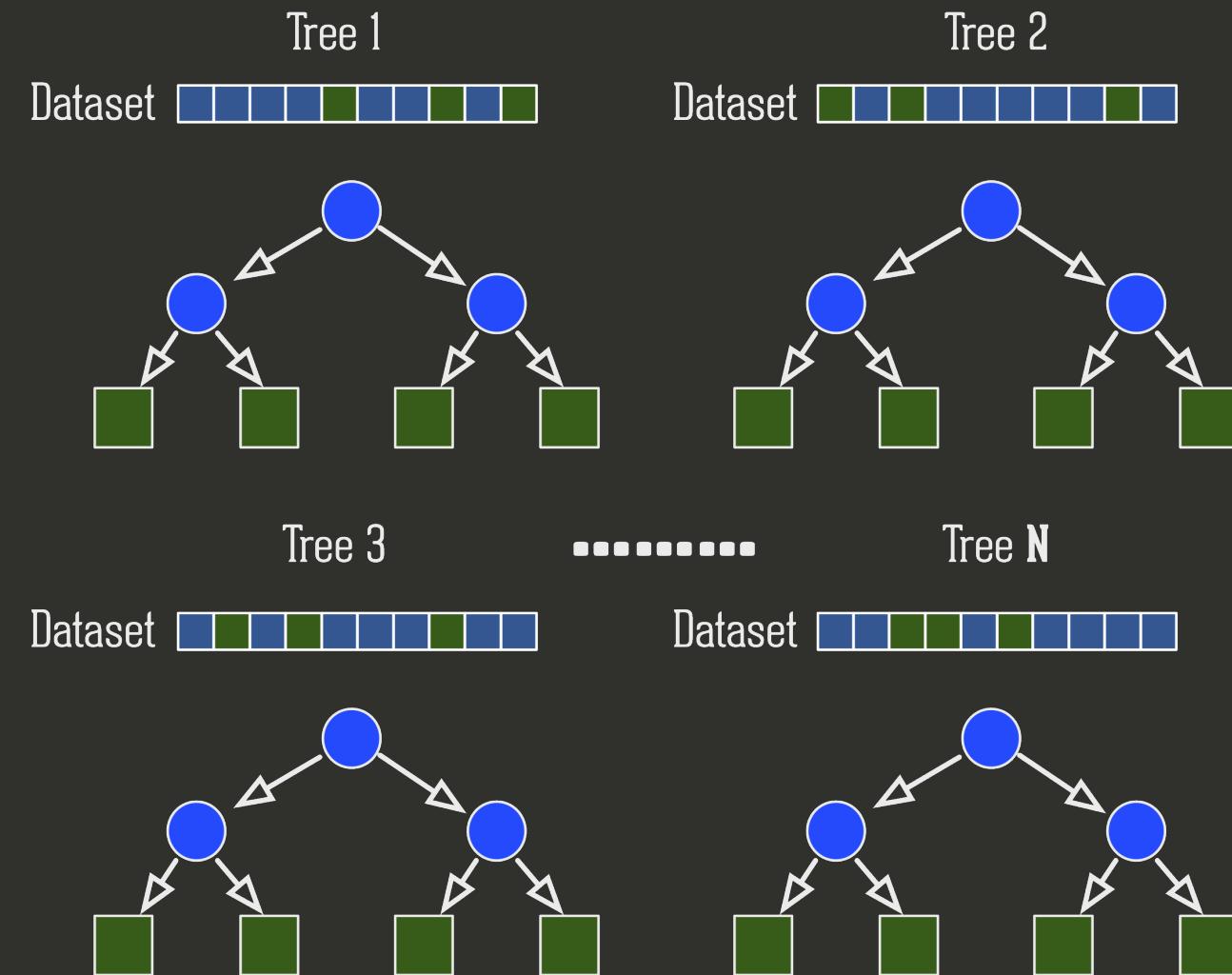
Boosting

Adds weak learners, so new learners pick up the slack. Incrementally increase the accuracy of model as weak learners focus more on misclassified examples from the previous weak learners.

NumVar	OrdVar	CatVar	▼
1	R2	Good	
1.5	R4	Bad	
2	R1	Bad	
2.5	R5	Good	
3	R3	Good	
3.5	R3	Bad	
⋮	⋮	⋮	⋮



Random Forests



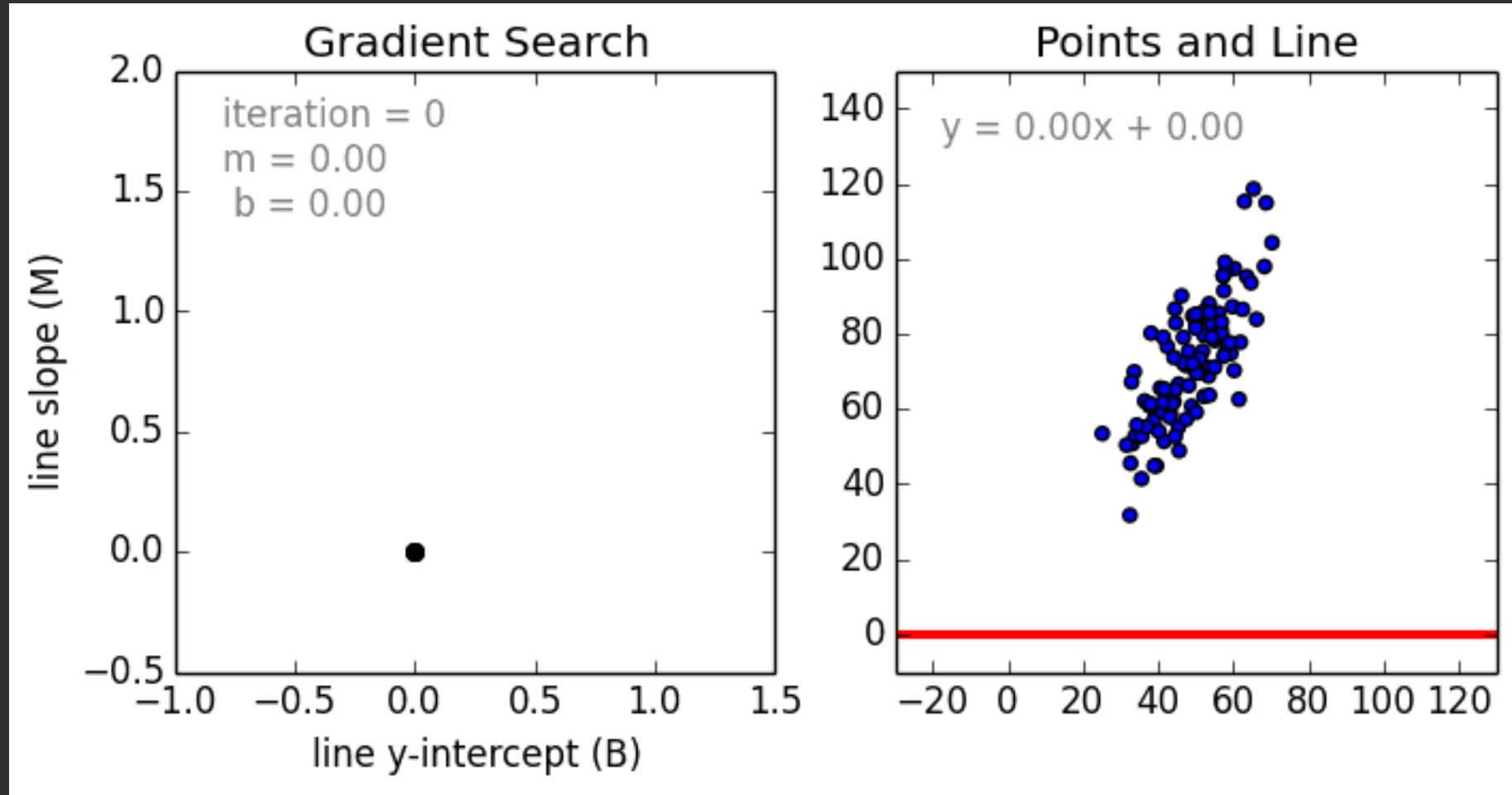
Traverse each tree and
at each node in a tree:

- i. Select **m** random predictor variables from available set
- ii. Use the variable with best split [use objective function]
- iii. Move to next node in tree

Gradient Boosting

Gradient
Descent +
Boosting

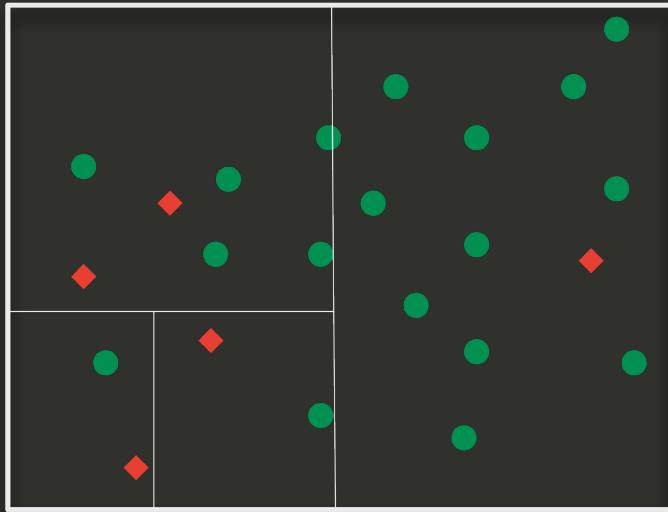
Gradient Descent



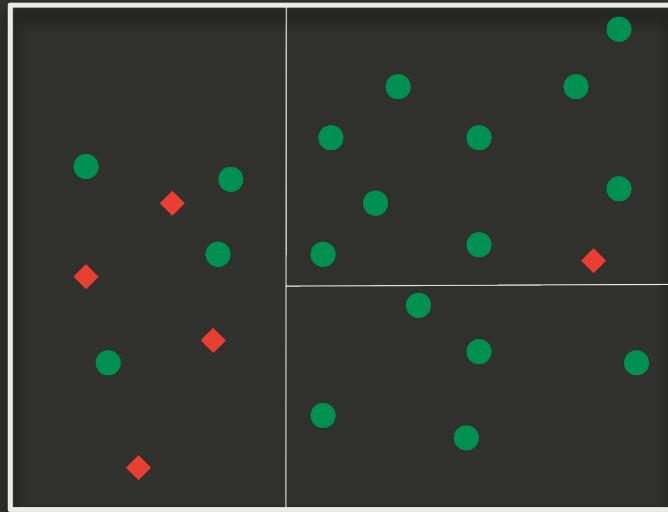
<https://github.com/mattnedrich/GradientDescentExample>

Gradient Boosting

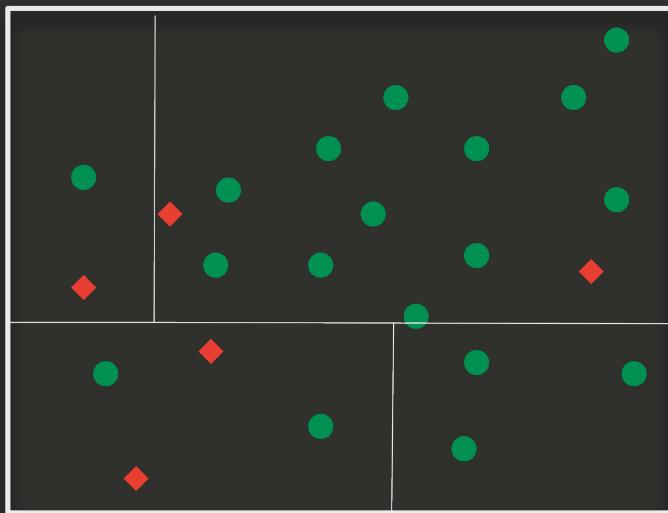
1



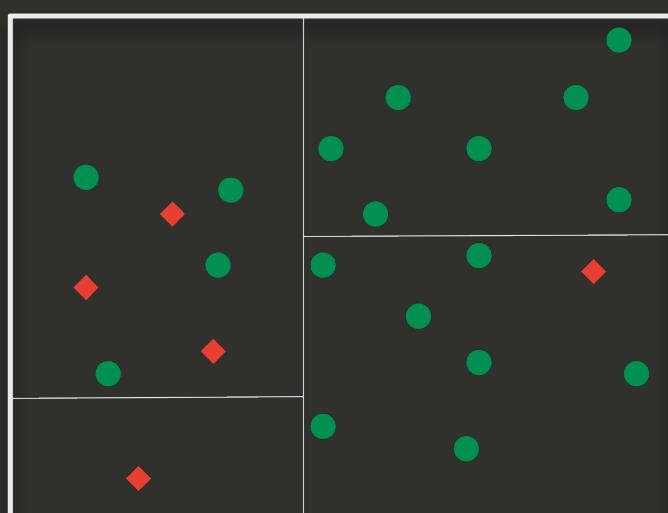
3



2



4



Tunables

- Number of trees and the learning rate
- *Across all trees*
 - subsampling rows
 - different loss function
- *Per tree*
 - maximum tree depth
 - minimum samples to split a node
 - minimum samples in a leaf node
 - subsampling features

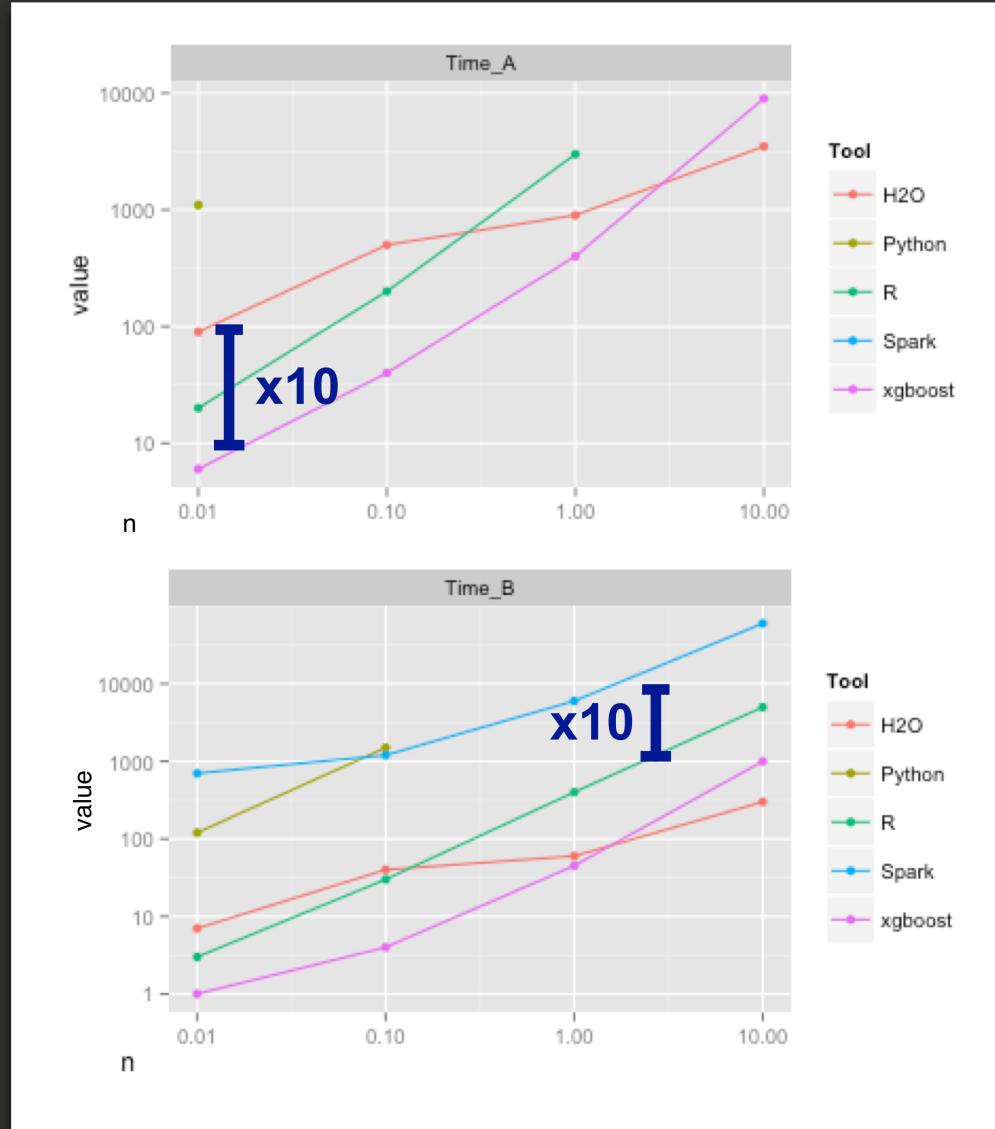


dmlc
XGBoost

H₂O.ai



Which to choose ?



[https://github.com/
szilard/benchm-ml](https://github.com/szilard/benchm-ml)

7.5 hrs / 79.8 AUC H2O
14 hrs / 81.1 AUC xgboost

Caveat for xgboost

Models by representing all problems as a regression predictive modeling problem and only takes numerical values as input.

1 Libraries

```
import urllib.request
import pandas
import xgboost
from sklearn import cross_validation
from sklearn.metrics import accuracy_score
from sklearn.preprocessing import LabelEncoder
```

```
# URL for the Iris dataset (UCI Machine Learning Repository)
url = "http://archive.ics.uci.edu/ml/machine-learning-databases/iris/
iris.data"
```

```
# download the file
raw_data = urllib.request.urlopen(url)
```

```
# load the CSV file as a numpy matrix
data = pandas.read_csv(raw_data, header=None)
dataset = data.values
```

```
# split data into X and y
X = dataset[:,0:4]
Y = dataset[:,4]
```

```
# encode string class values as integers
label_encoder = LabelEncoder()
label_encoder = label_encoder.fit(Y)
label_encoded_y = label_encoder.transform(Y)
```

```
seed = 7
test_size = 0.33
X_train, X_test, y_train, y_test = cross_validation.train_test_split(X,
label_encoded_y, test_size=test_size, random_state=seed)
```

```
# fit model
model = xgboost.XGBClassifier()
model.fit(X_train, y_train)
```

```
# make predictions for test data
y_pred = model.predict(X_test)
predictions = [round(value) for value in y_pred]
```

```
# evaluate predictions
accuracy = accuracy_score(y_test, predictions)
print("Accuracy: %.2f%%" % (accuracy * 100.0))
```

2 Data Processing

3 Modelling & Prediction

A better example

```
import numpy as np
import urllib.request
import xgboost
from sklearn import cross_validation
from sklearn.metrics import accuracy_score

# URL for the Pima Indians Diabetes dataset = "http://
archive.ics.uci.edu/ml/machine-learning-databases/pima-
indians-diabetes/pima-indians-diabetes.data"

# download the file
raw_data = urllib.request.urlopen(url)
```

A better example

```
# load the CSV file as a numpy matrix
dataset = np.loadtxt(raw_data, delimiter=",")
print(dataset.shape)

X = dataset[:,0:8]
Y = dataset[:,8]

# split data into train and test sets
seed = 7
test_size = 0.33
X_train, X_test, y_train, y_test =
cross_validation.train_test_split(X, Y, test_size=test_size,
random_state=seed)
```

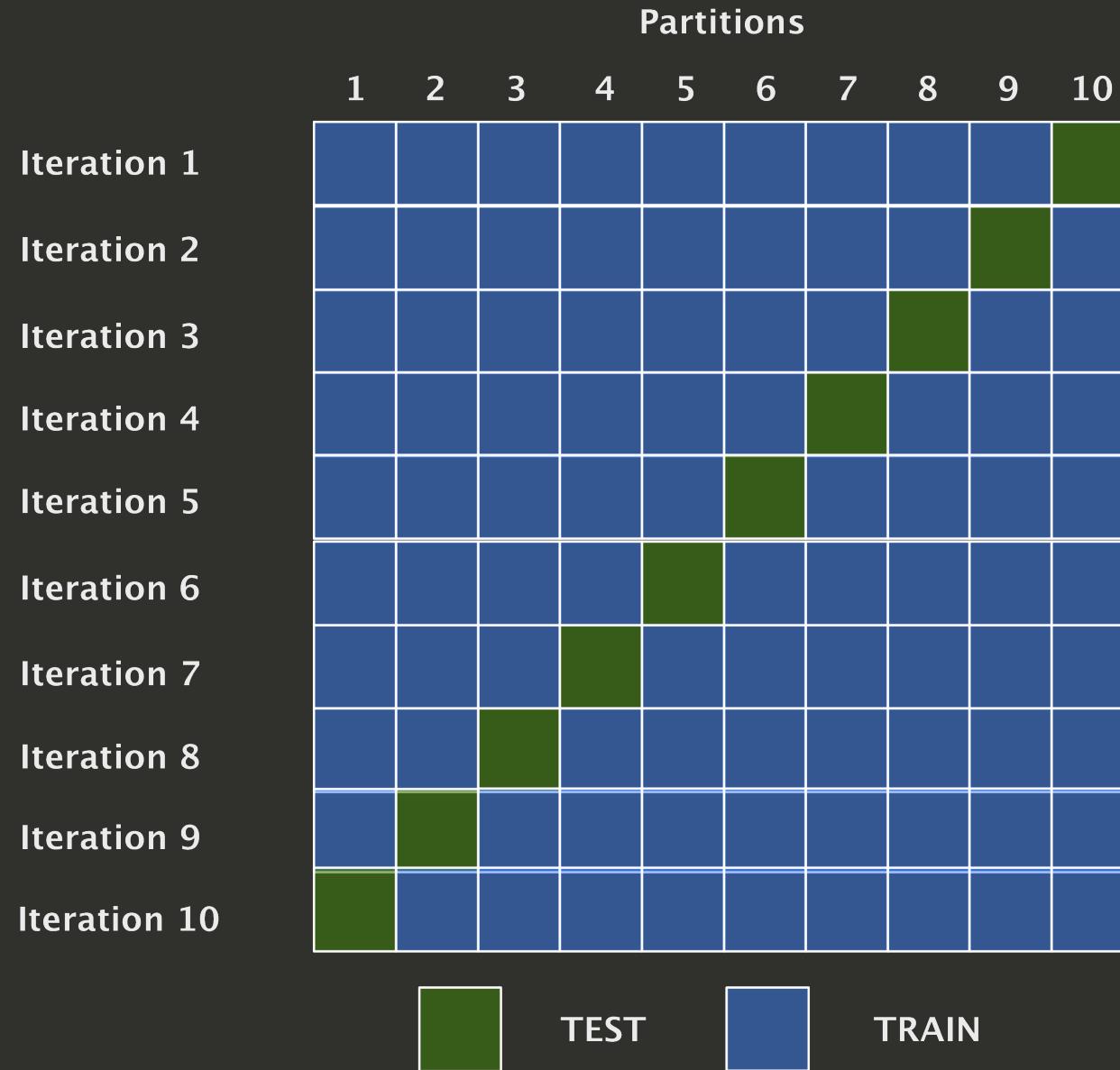
A better example

```
model = xgboost.XGBClassifier()  
model.fit(X_train, y_train)
```

```
# make predictions for test data  
y_pred = model.predict(X_test)  
predictions = [round(value) for value in y_pred]  
  
# evaluate predictions  
accuracy = accuracy_score(y_test, predictions)  
print("Accuracy: %.2f%%" % (accuracy * 100.0))
```

Accuracy: 77.95%

Cross-validation



Cross-validation

```
import numpy as np  
from sklearn.model_selection import KFold
```

```
X = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

```
kf = KFold(n_splits=10)  
for train, test in kf.split(X):  
    print("%s %s" % (train, test))
```

Cross-validation

[2 3 4 5 6 7 8 9 10]	[0 1]
[0 1 3 4 5 6 7 8 9 10]	[2]
[0 1 2 4 5 6 7 8 9 10]	[3]
[0 1 2 3 5 6 7 8 9 10]	[4]
[0 1 2 3 4 6 7 8 9 10]	[5]
[0 1 2 3 4 5 7 8 9 10]	[6]
[0 1 2 3 4 5 6 8 9 10]	[7]
[0 1 2 3 4 5 6 7 9 10]	[8]
[0 1 2 3 4 5 6 7 8 10]	[9]
[0 1 2 3 4 5 6 7 8 9]	[10]

Scikit Only

```
from sklearn.ensemble import  
GradientBoostingClassifier #For Classification  
from sklearn.ensemble import  
GradientBoostingRegressor #For Regression
```

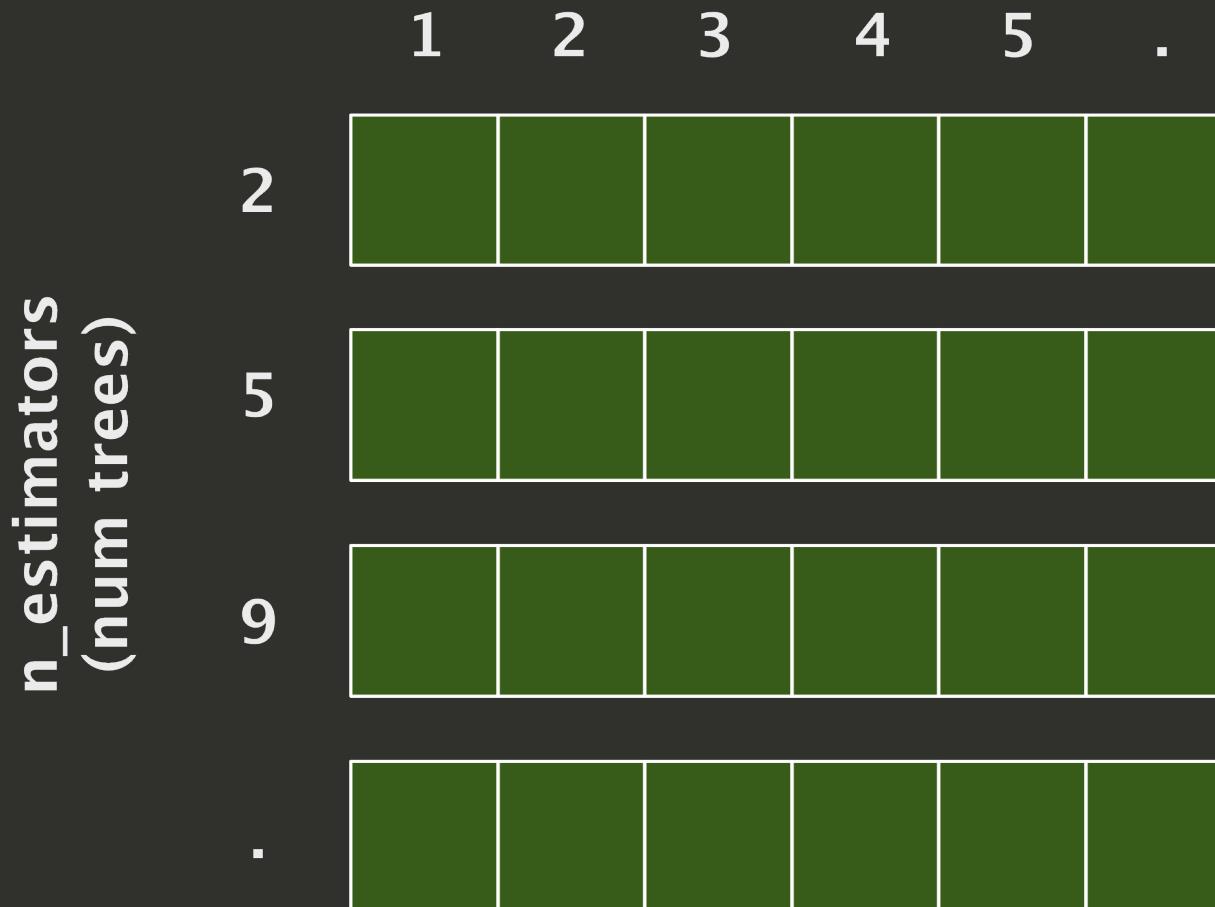
```
clf = GradientBoostingClassifier(n_estimators=100,  
learning_rate=1.0, max_depth=1)
```

```
clf.fit(X_train, y_train)
```

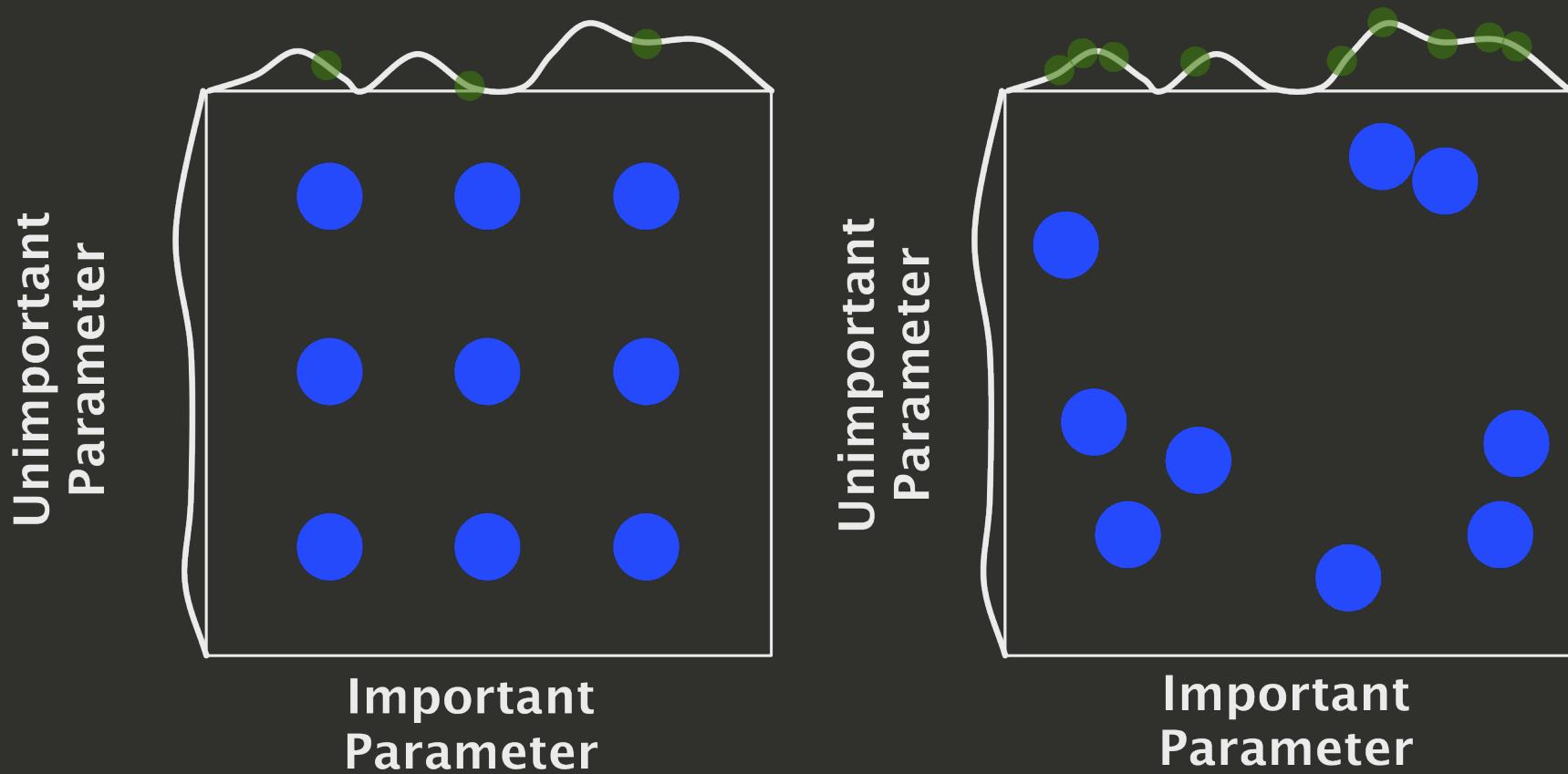
Getting the
best results

Which Parameters

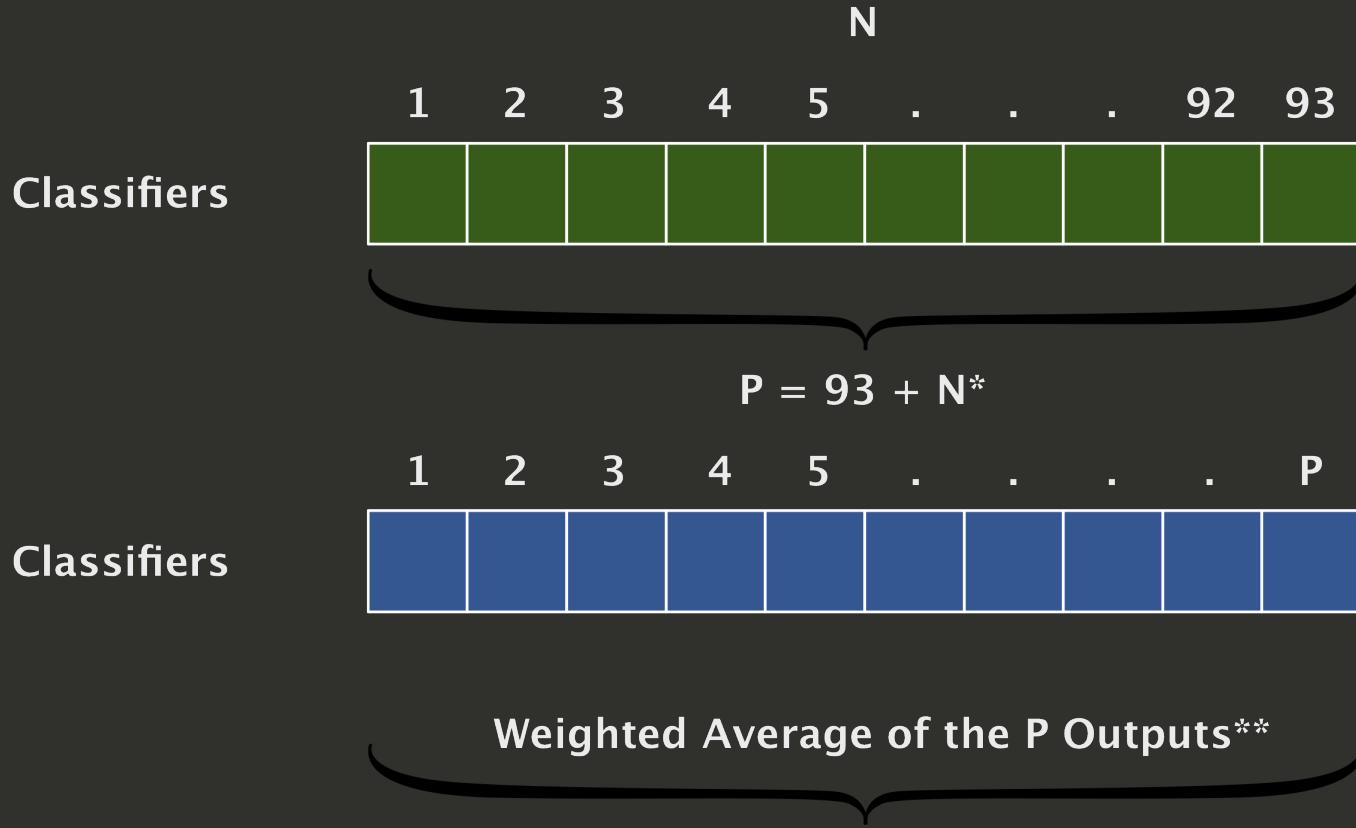
max_depth (max num of nodes in tree)



Kinds of Parameter Search



Stacking, Blending, & Averaging

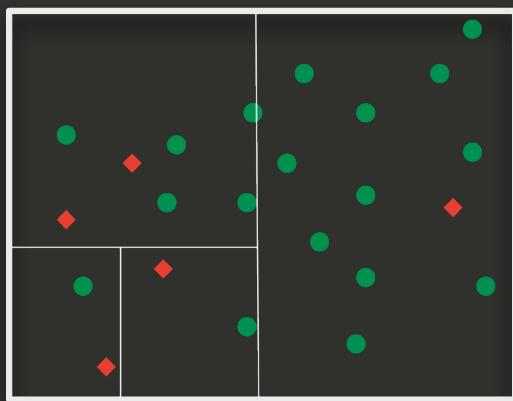


Christophe Bourguignon

* Added N features using logit
(log odds) of prediction rather
than the prediction itself

**Harmonic mean (for average)

Recap



Unimportant
Parameter

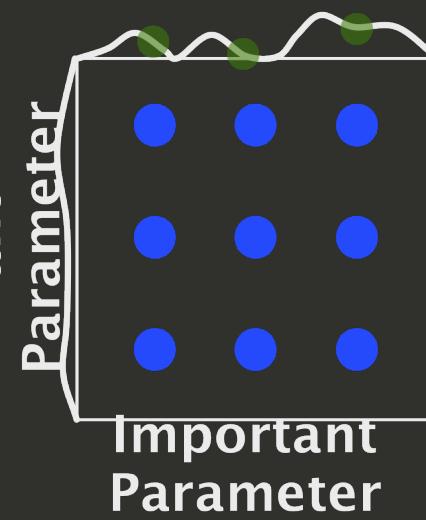


Photo / Image Credits



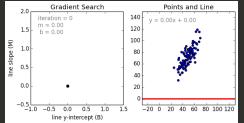
Yves Cosentino

<https://www.flickr.com/photos/31883499@N05/3015866093>



Jordi Payà

<https://www.flickr.com/photos/arg0s/6705230505/>



Matt Nedrich

<https://github.com/mattnedrich/GradientDescentExample>



Various

https://en.wikipedia.org/wiki/Iris_flower_data_set



Chia Ying Yang

<https://www.flickr.com/photos/enixii/17074838535/>

Link to Slides & Code

http://github.com/braz/pycon2016_talk/
