

Supervised Learning with Random Forests

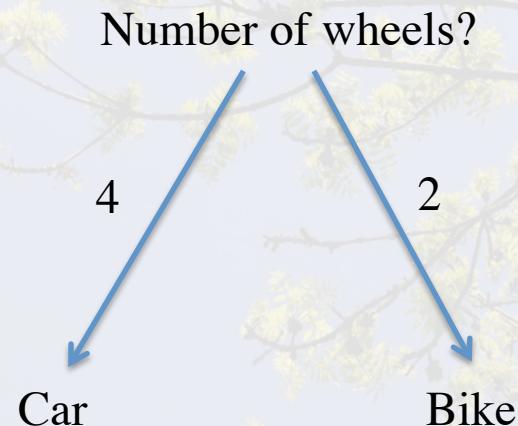
Classification – what is this thing?

Decision trees

Classes: Car or Bike

Features: Number of wheels

Question: How many wheels does it have?



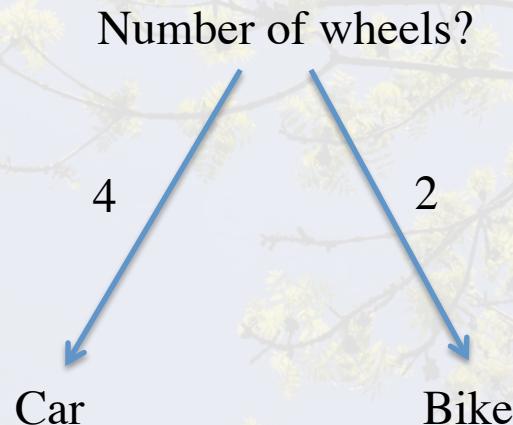
Classification – what is this thing?

Decision trees

Classes: Car or Bike

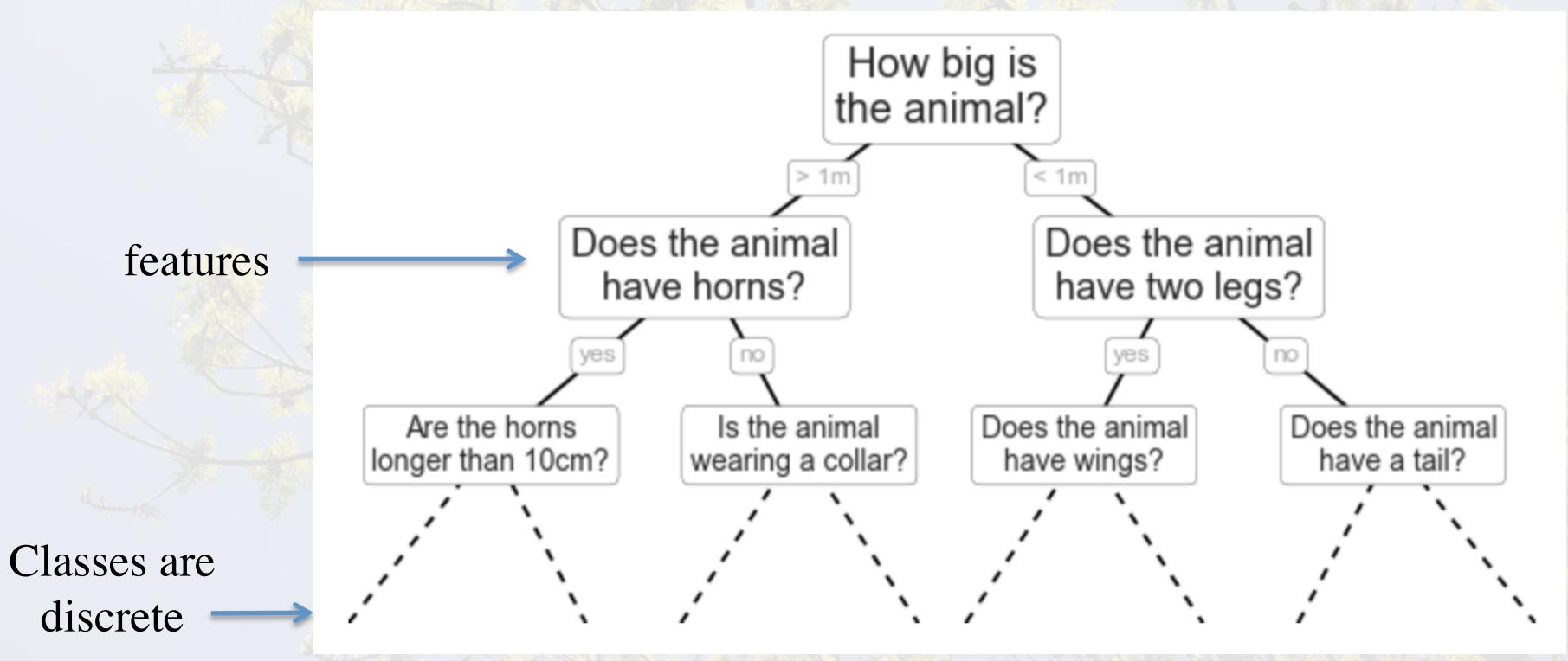
Features: Number of wheels

Question: How many wheels does it have?



Classification – what is this thing?

Decision trees

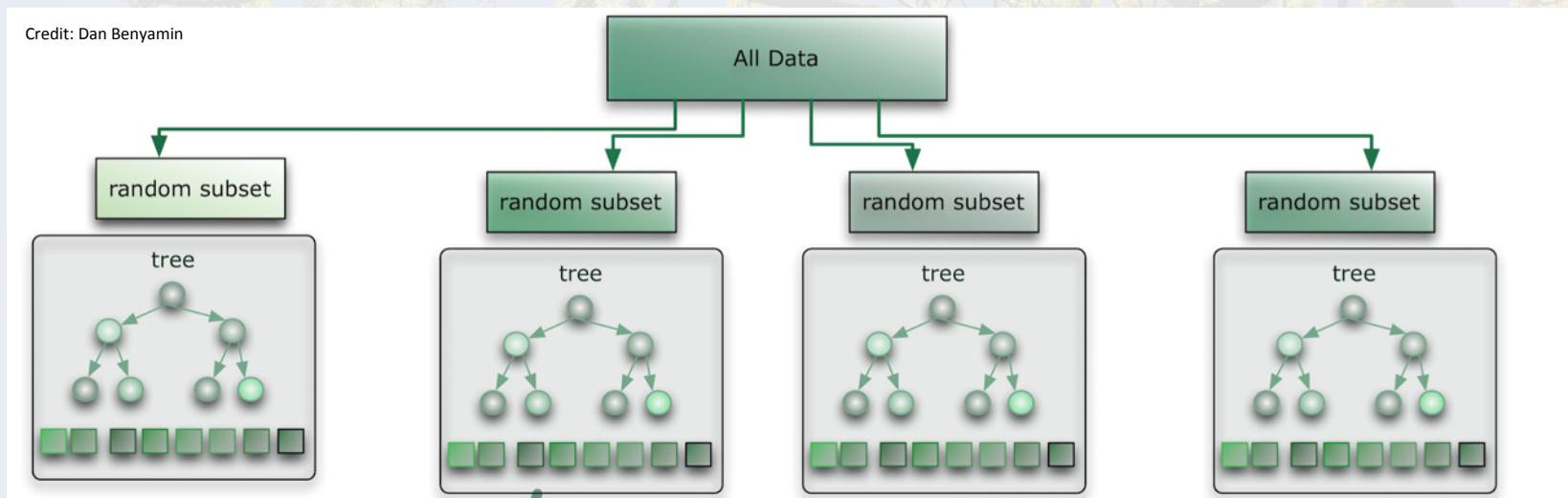


Requires training data such that the algorithm can map decisions to known output classes

Random Forests

Combine many decision trees into a single model

- Each decision tree in the forest considers a random subset of features and data (so is a biased estimator)
- They each capture different trends in the data
- At the end take majority vote from all decision trees for the predicted class





2.5 million sources with photometry points (optical/IR/radio)

Class	u	g	r	i	z	w1	w2	w3	w4
Galaxy/Star/QSO	21.2	19.8	19.1	18.7	18.3	14.3	14.1	10.7	8.9

Lots of trees working on random subsets of features and samples



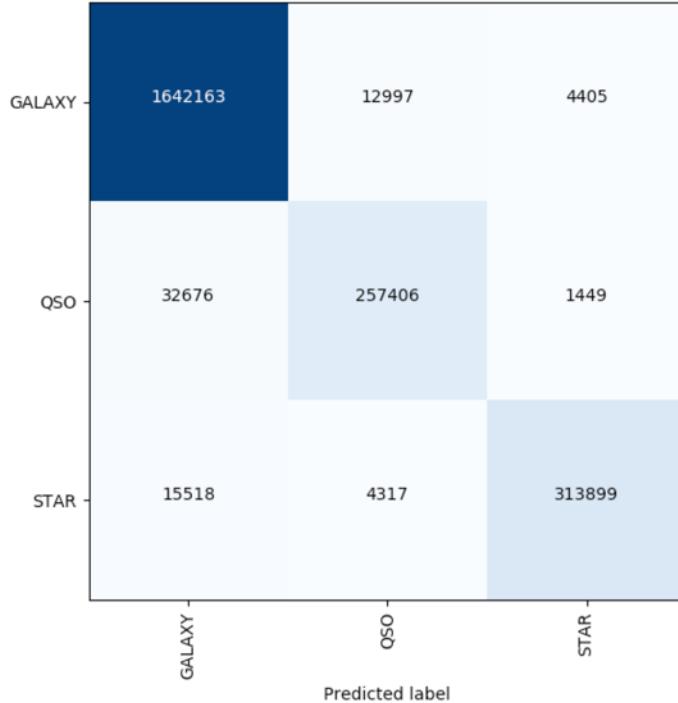
Ensemble of trees decides the most likely class at the end



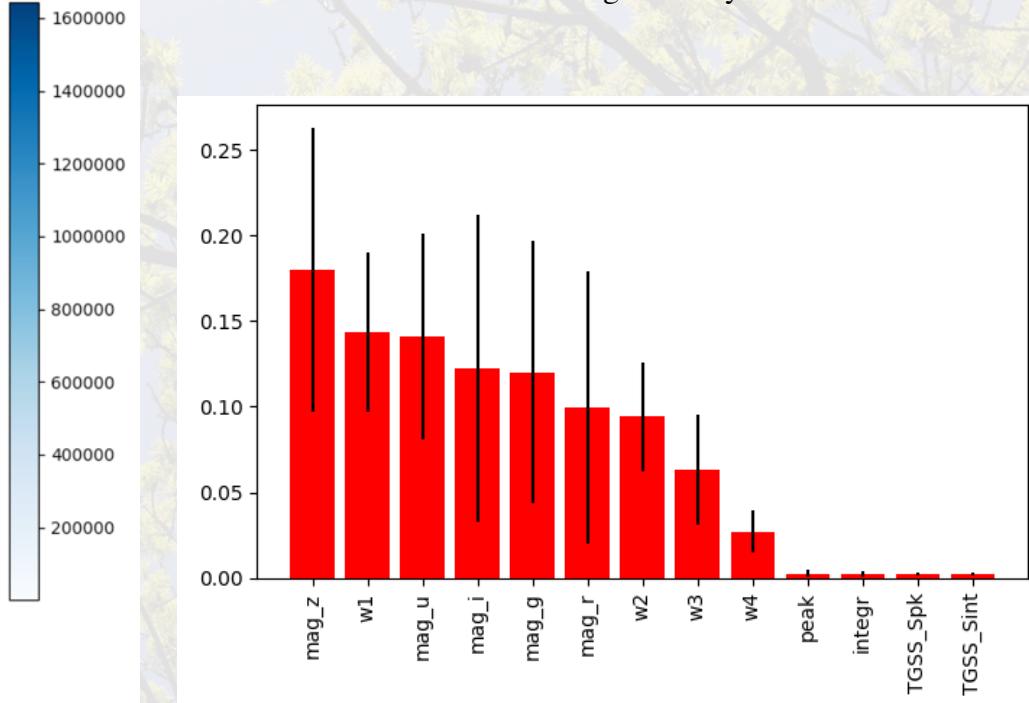
2.5 million sources with 13 photometry points (optical/IR/radio)

Trained on 250k samples, classifies remaining 2.25 million with 97% accuracy*

Confusion matrix, without normalization



*Assessing accuracy metrics is for another talk



Python implementation

Load in libraries from sklearn:

```
>>> from sklearn.ensemble import RandomForestClassifier  
>>> from sklearn.model_selection import train_test_split  
>>> from sklearn.metrics import accuracy_score
```

Load data in:

```
>>> all_features = [[mag_u, mag_g, ...], [mag_u, mag_g, ...], ...] #photometry data  
>>> all_classes = [Galaxy,Star, ...] #class labels
```

split your data set up into two parts, one for training, one for testing:

```
>>> features_train, features_test, classes_train, classes_test = train_test_split(all_features,  
all_classes, test_size=0.5)
```

Initialise classifier (n_estimators is number of trees):

```
>>> forest = RandomForestClassifier(n_estimators=100)
```

fit to the train data:

```
>>> forest = forest.fit(features_train, classes_train)
```

Predict classes from your test data

```
>>> classes_predicted= forest.predict(features_test)
```

Check accuracy:

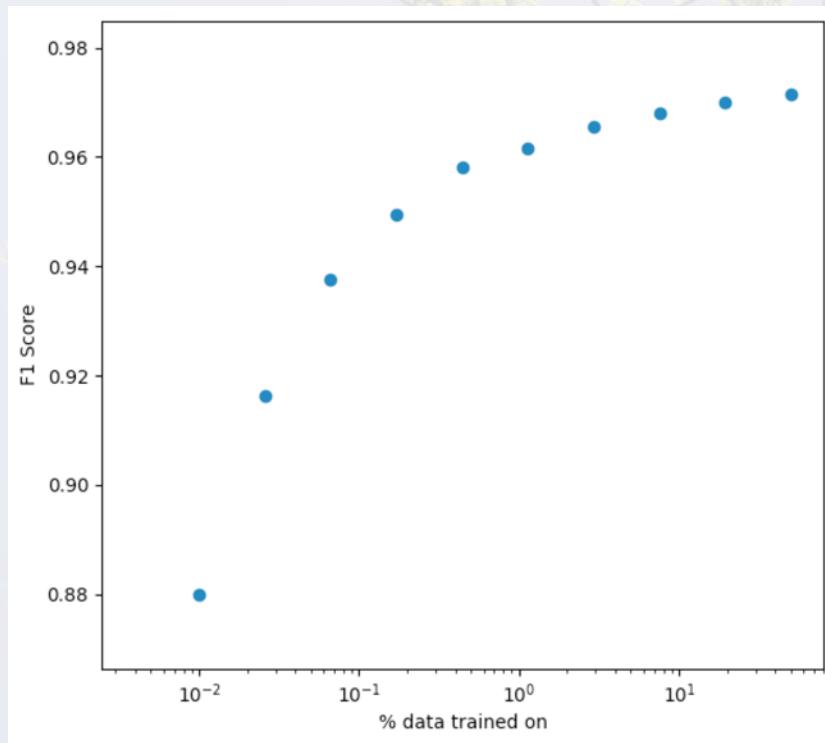
```
>>> accuracy = accuracy_score(classes_test, classes_predicted)
```

Tuning hyper parameters

2.5 million sources with 13 photometry points (optical/IR/radio) – 97% accuracy in classifications

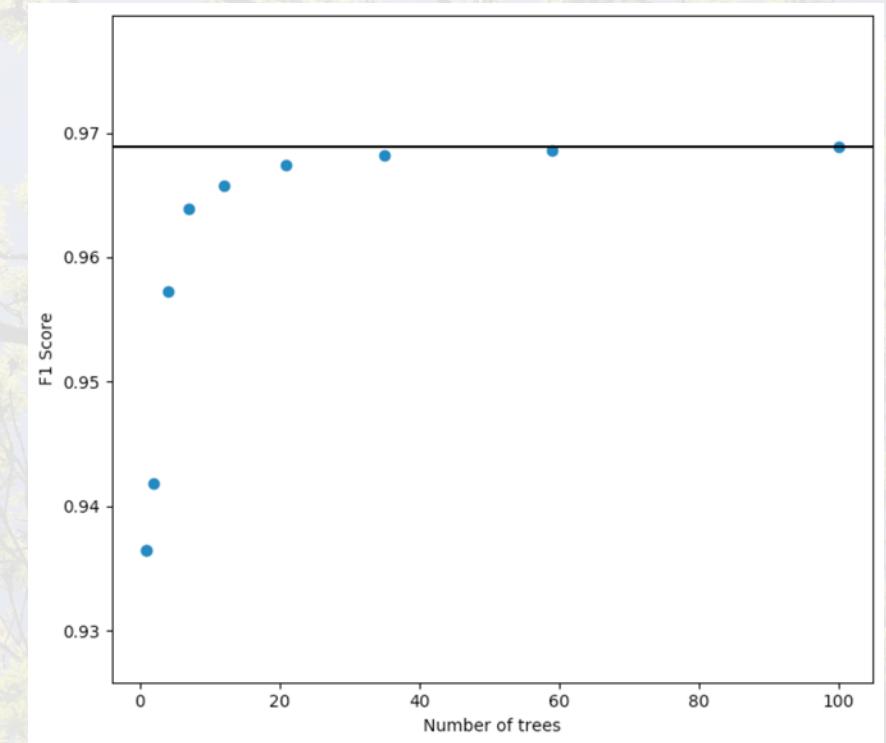
How much data needed to train on?

Depends on the complexity of the data and classifications

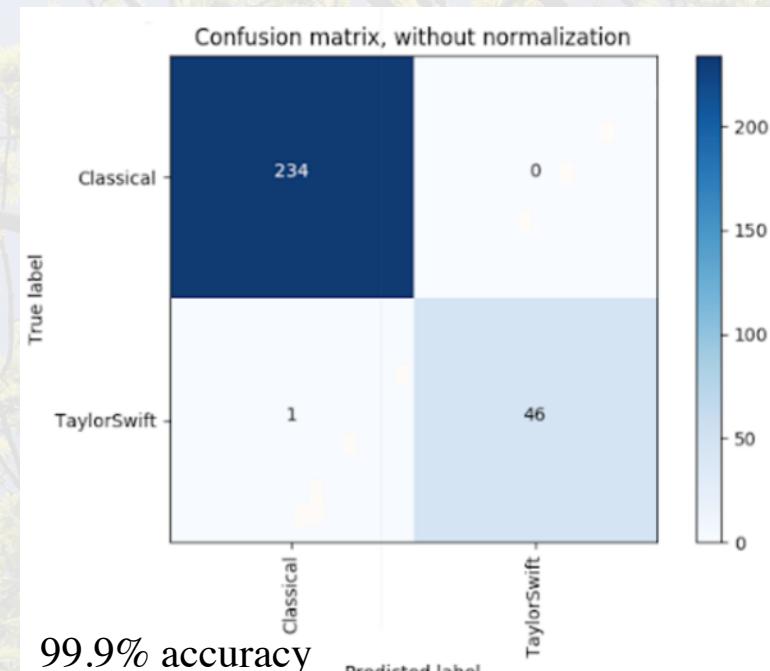
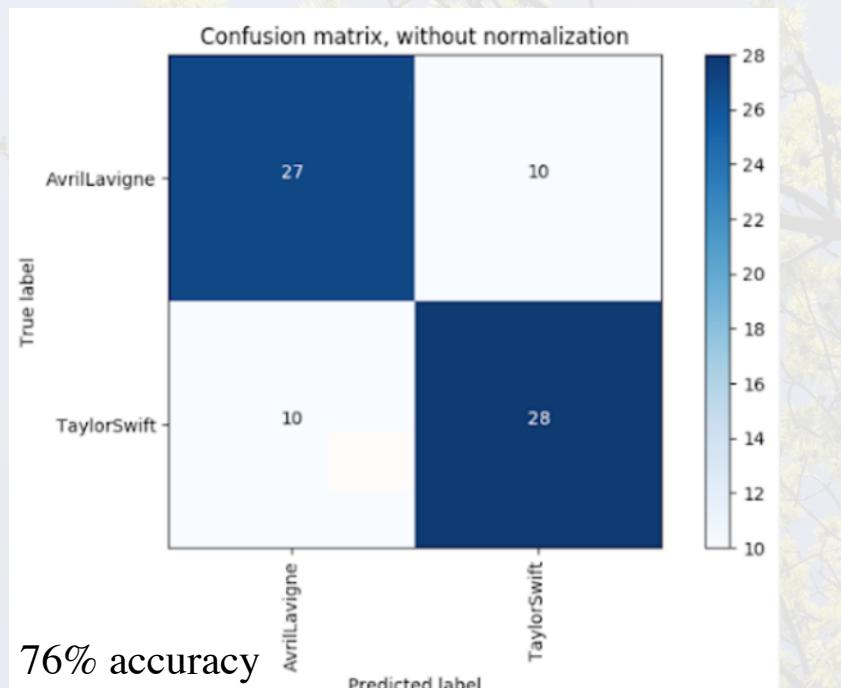
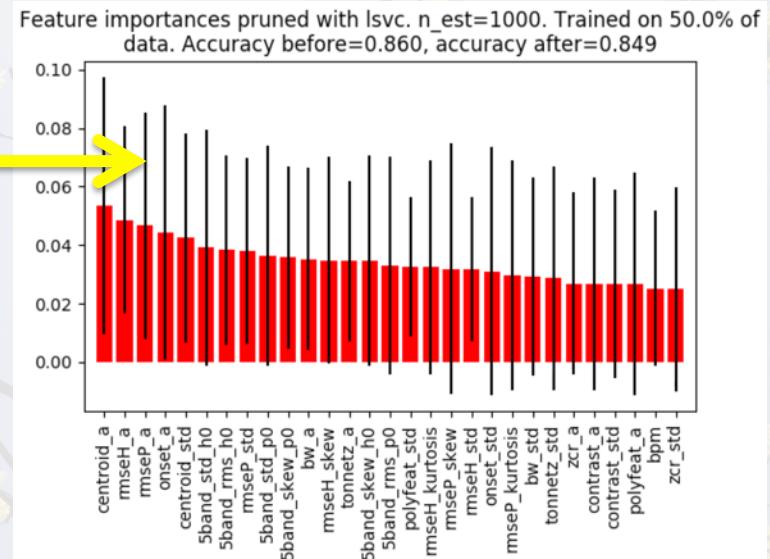
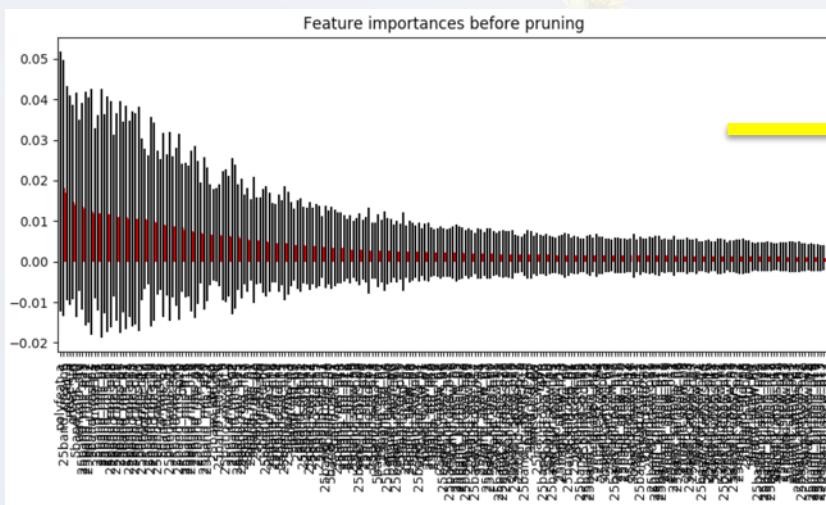


Number of trees (estimators)?

The more features you have the more you'll need



Random Forests - Music



More info at: github.com/informationcake/music-machine-learning

Thanks for coming ☺

