Escaping the Black Box

Yellowbrick: A Visual API for Machine Learning

Charla Plenaria

PyData
San Luis 2017

Rebecca Bilbro

Escapando de la caja negra con Yellowbrick

Con más herramientas de aprendizaje automático que nunca, ¿Por qué sigue siendo tan difícil? Los practicantes saben que el aprendizaje automático es más complejo que la simple selección de la mejor herramienta o el mejor algoritmo. Es parte de búsqueda, parte de experiencia, y parte de *suerte*. En lugar de sucumbir al encanto de las cajas negras, en esta charla veremos cómo hacer para reducir la necesidad de *suerte* con Yellowbrick - una API visual para aprendizaje automático.


Once upon a time ...


And then things got ...
easy

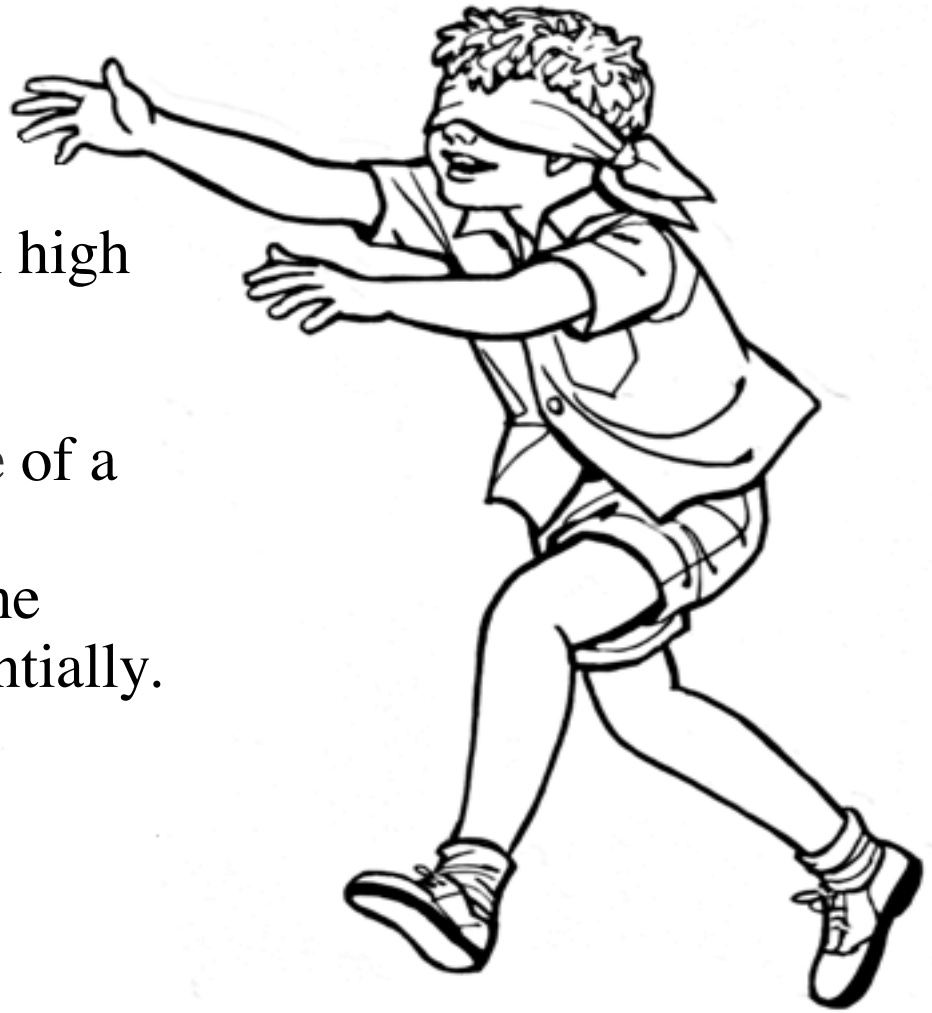
Try them all!

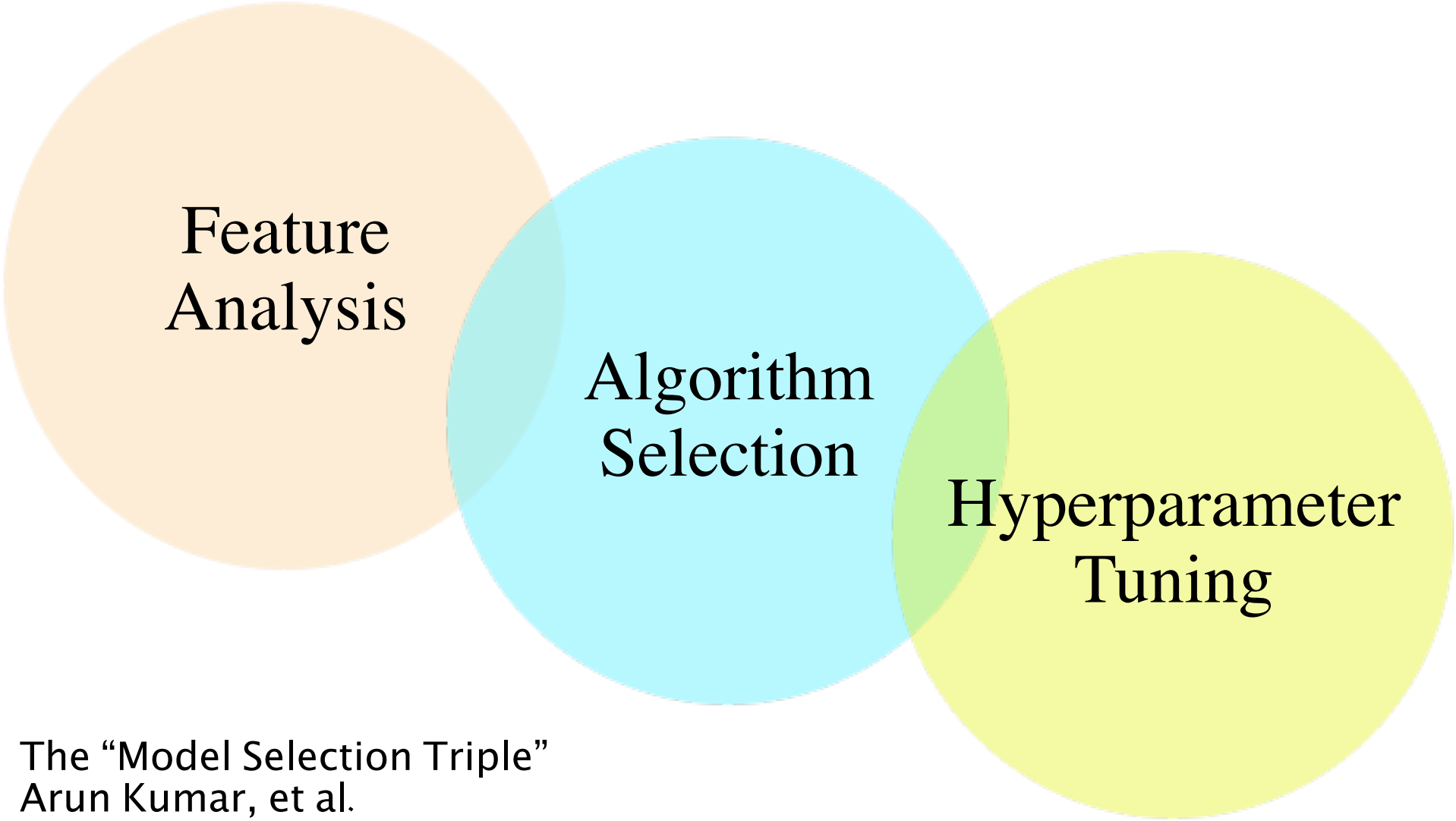

```
from sklearn.svm import SVC
from sklearn.naive_bayes import GaussianNB
from sklearn.ensemble import AdaBoostClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn import model_selection as ms

classifiers = [
  KNeighborsClassifier(5),
  SVC(kernel="linear", C=0.025),
  RandomForestClassifier(max_depth=5),
  AdaBoostClassifier(),
  GaussianNB(),
]

kfold  = ms.KFold(len(X), n_folds=12)
max([
  ms.cross_val_score(model, X, y, cv=kfold).mean
  for model in classifiers
])
```

Machine Learning as a Service

Except ...

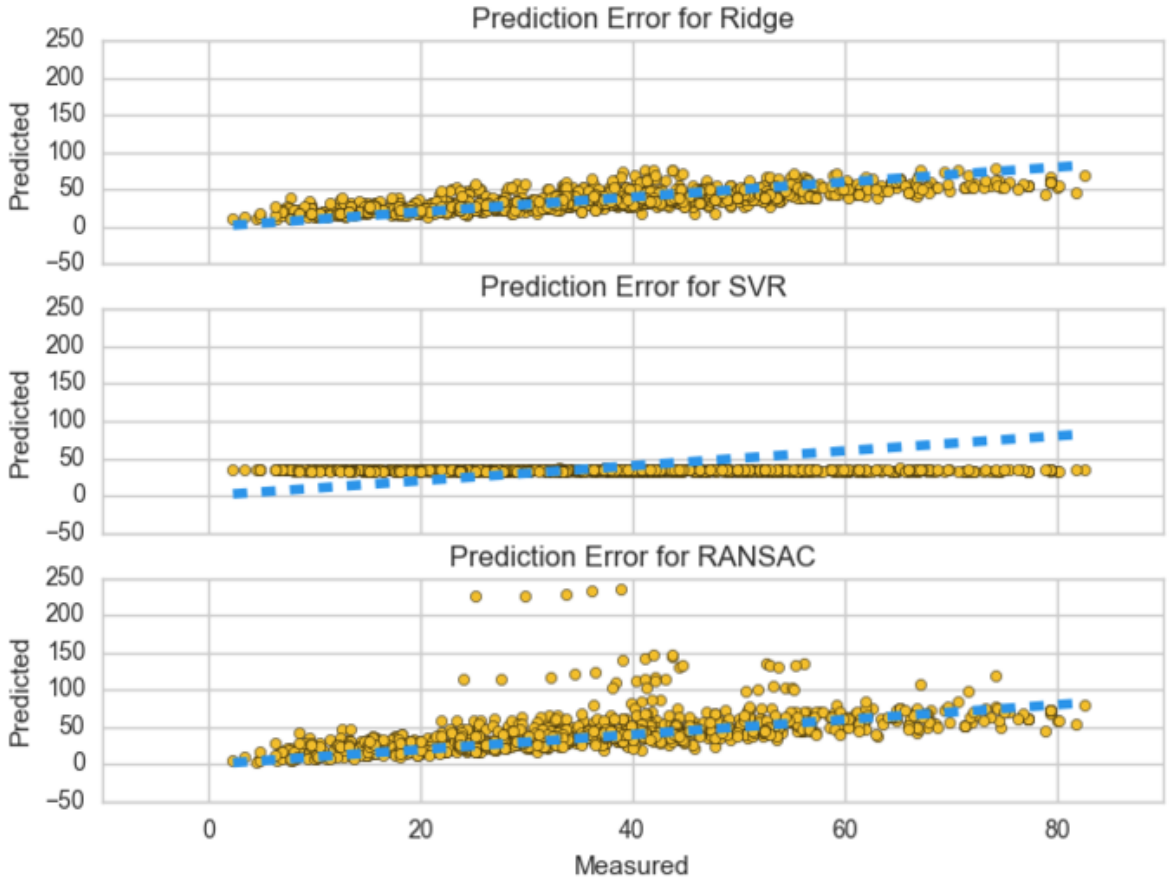
- Search is **difficult,** particularly in high dimensional space.
- Even with clever optimization techniques, there is **no guarantee** of a solution.
- As the search space gets larger, the amount of **time** increases exponentially.





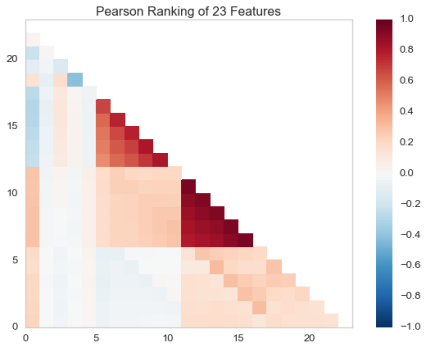
The “Model Selection Triple”
Arun Kumar, et al.

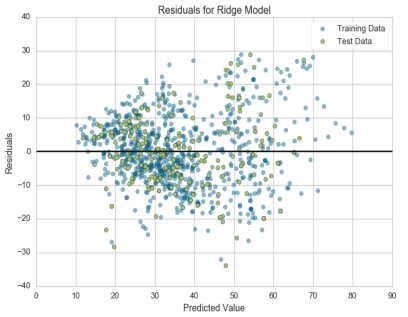
Solution: Visual Steering

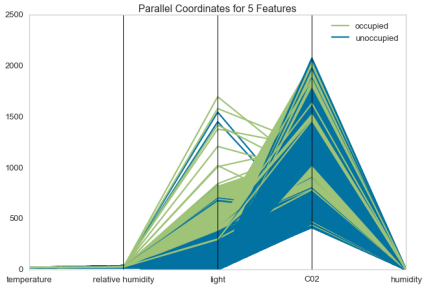


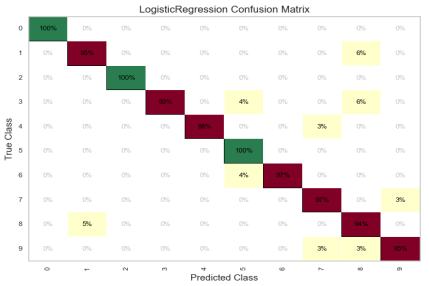
Enter Yellowbrick

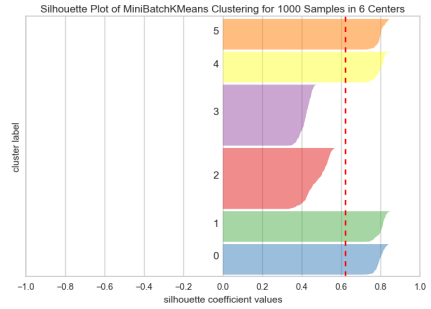
- Extends the Scikit-Learn API.
- Enhances the model selection process.
- Tools for feature visualization, visual diagnostics, and visual steering.
- Not a replacement for other visualization libraries.

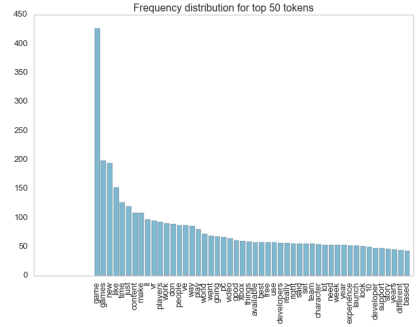










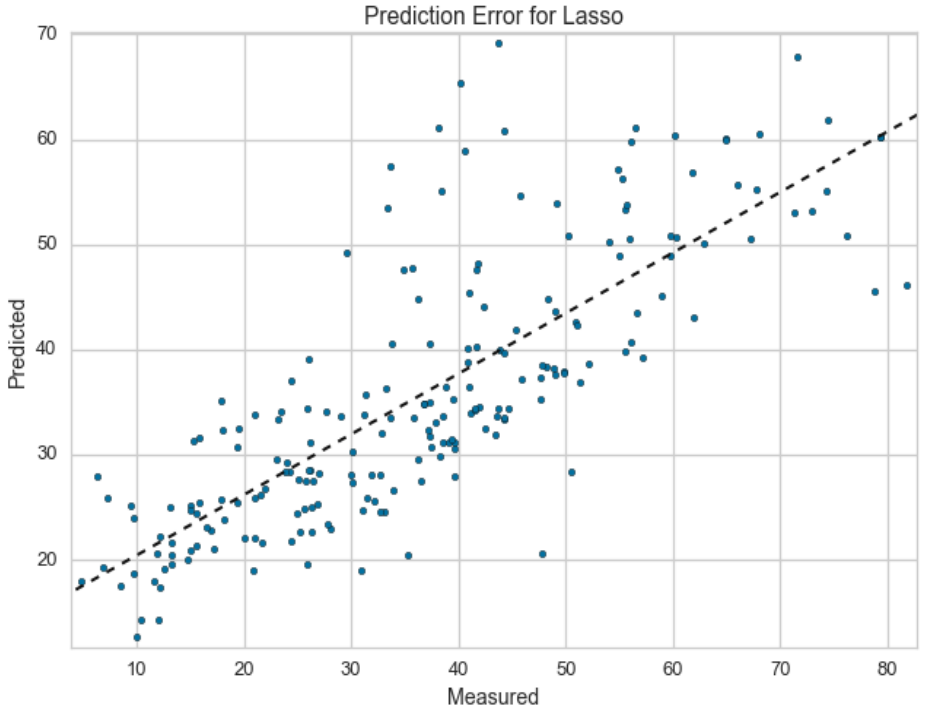


Scikit-Learn Estimator Interface

```
# Import the estimator
from sklearn.linear_model import Lasso

# Instantiate the estimator
model = Lasso()

# Fit the data to the estimator
model.fit(X_train, y_train)

# Generate a prediction
model.predict(X_test)
```

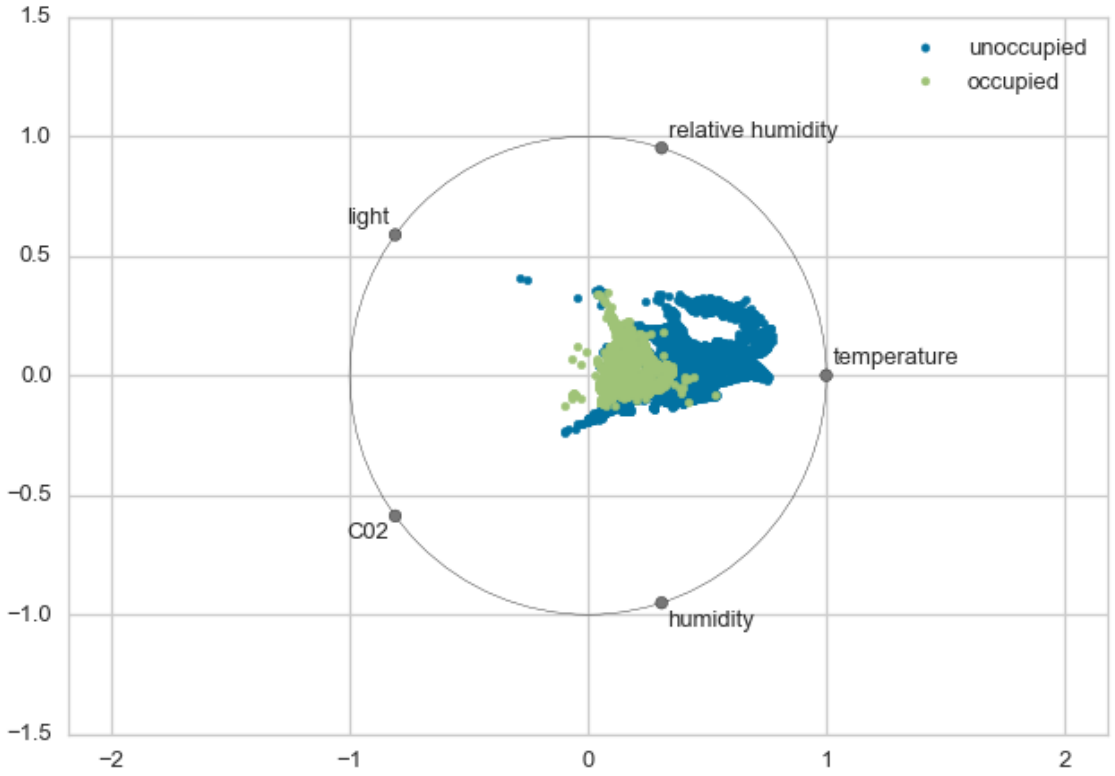
Yellowbrick Visualizer Interface

```
# Import the model and visualizer
from sklearn.linear_model import Lasso
from yellowbrick.regressor import PredictionError

# Instantiate the visualizer
visualizer = PredictionError(Lasso())

# Fit
visualizer.fit(X_train, y_train)

# Score and visualize
visualizer.score(X_test, y_test)
visualizer.poof()
```



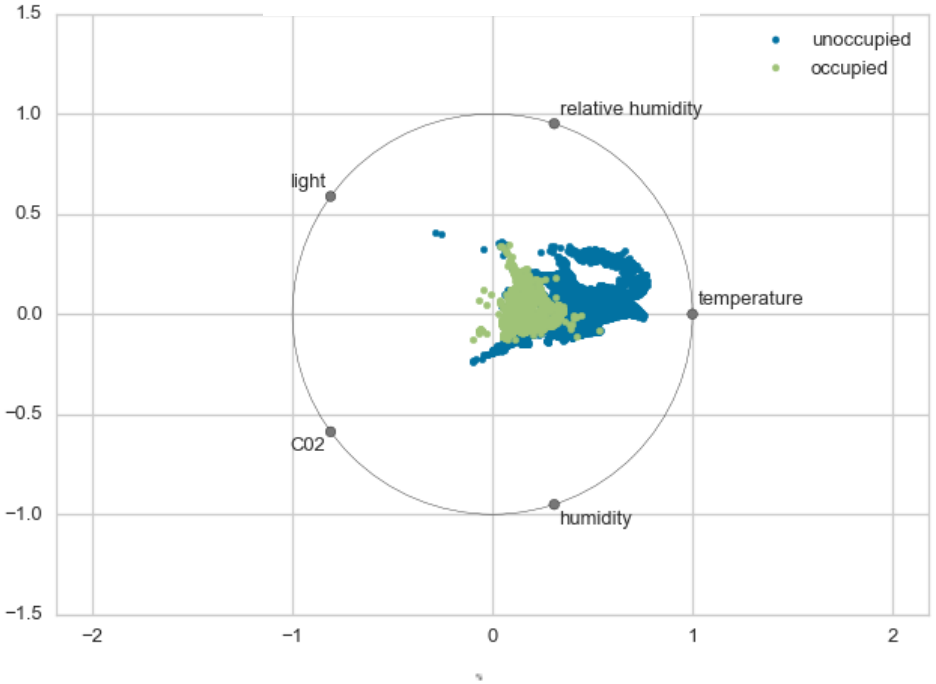
How do I select the right features?

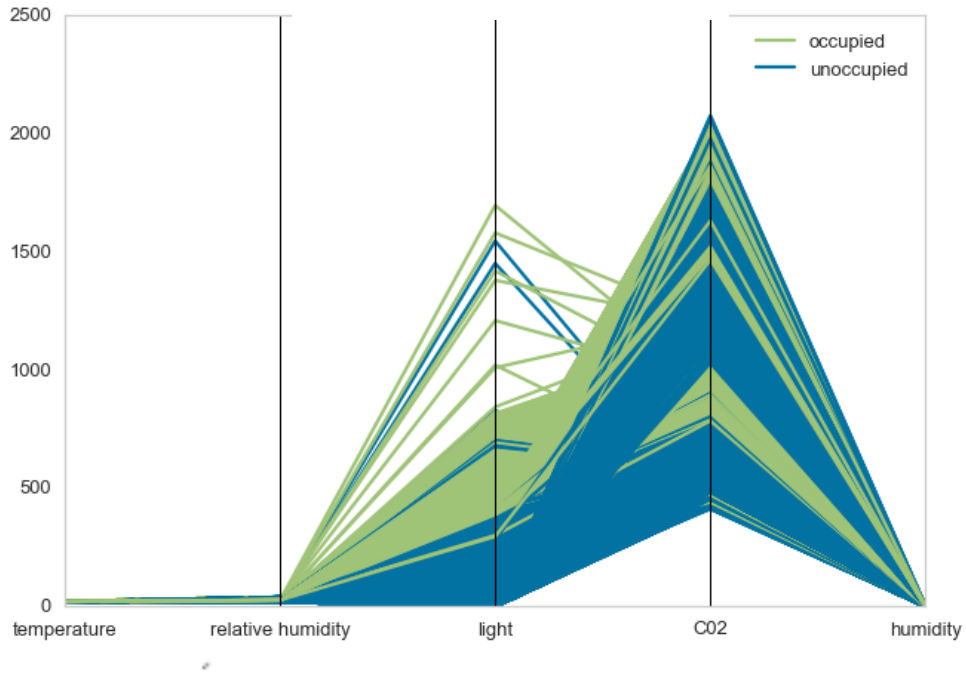
Is this room occupied?



- Given labelled data with amount of light, heat, humidity, etc.
- Which features are most predictive?
- How hard is it going to be to distinguish the empty rooms from the occupied ones?

Yellowbrick Feature Visualizers

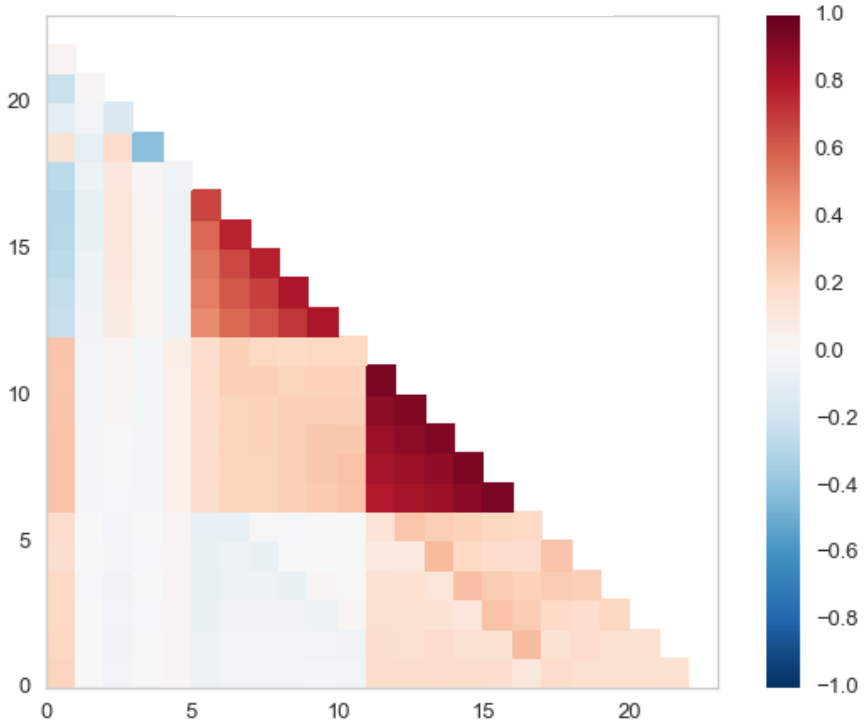


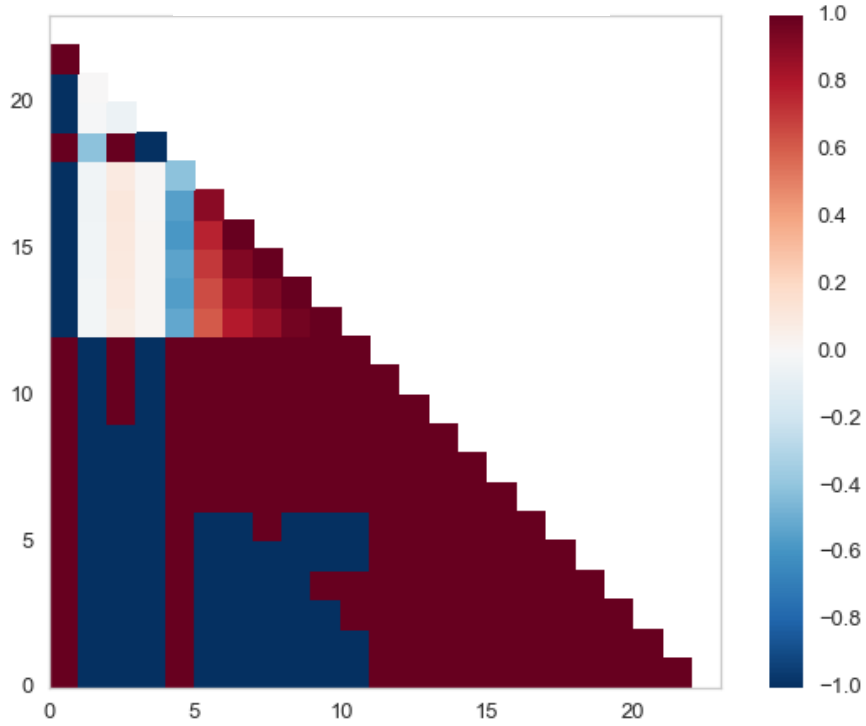


Use radviz or parallel coordinates to look for class separability

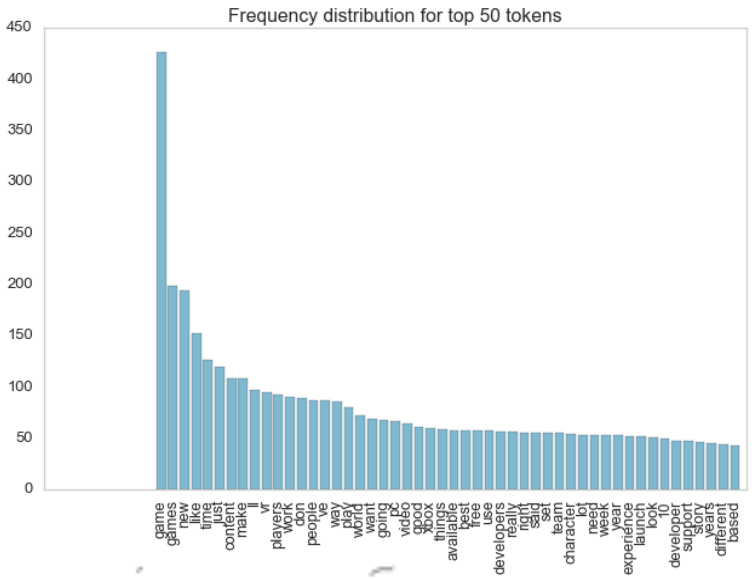
Yellowbrick Feature Visualizers

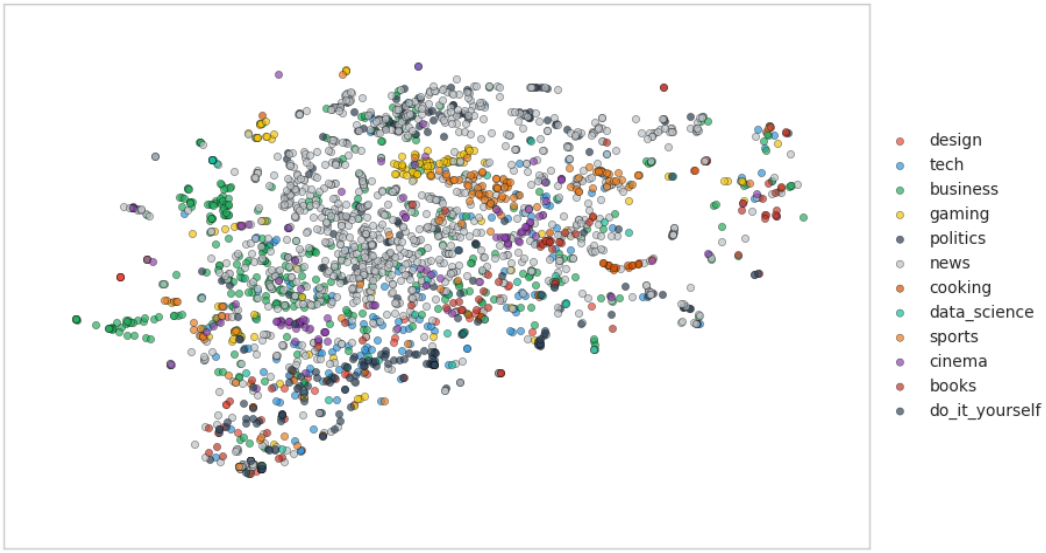
Use Rank2D for pairwise feature analysis





…for text, too!



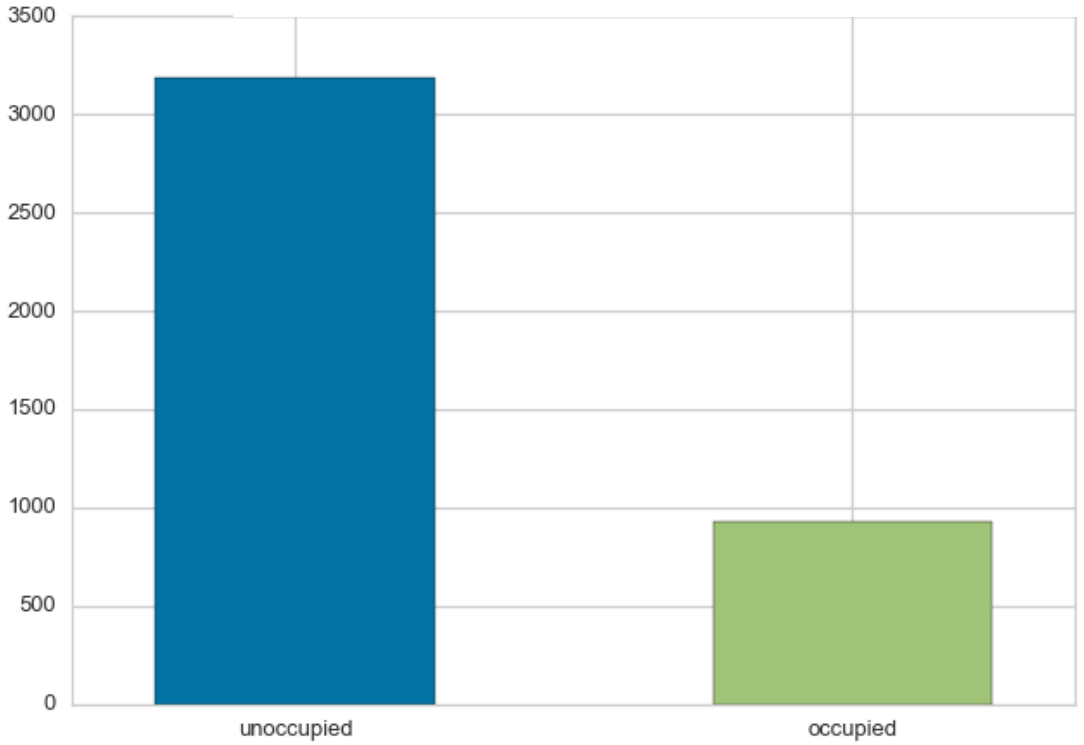


Visualize top tokens, document distribution & part-of-speech tagging

```
Algebra ( from Arabic `` al-jabr '' meaning `` reunion of broken parts '' ) is one of the bro
ad parts of mathematics , together with number theory , geometry and analysis . In its most g
eneral form , algebra is the study of mathematical symbols and the rules for manipulating the
se symbols ; it is a unifying thread of almost all of mathematics .
```

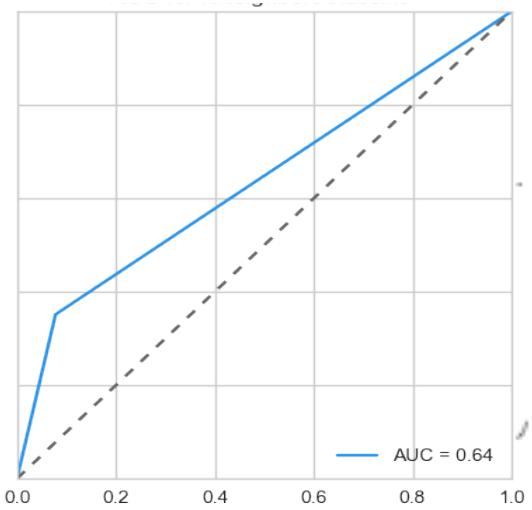
What's the best model?

Why isn’t my model predictive?

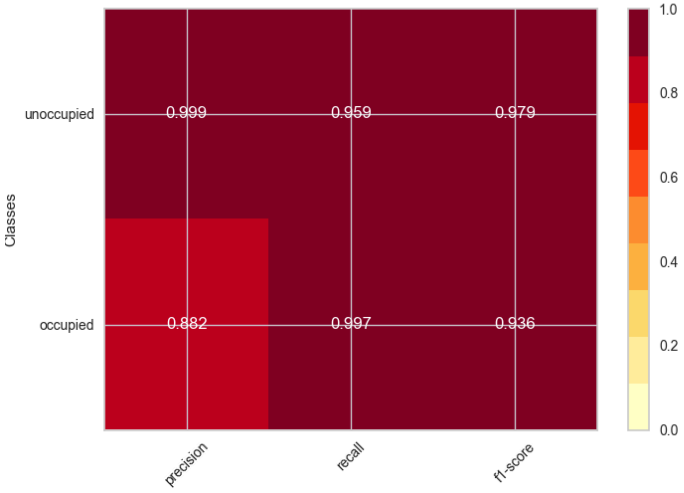


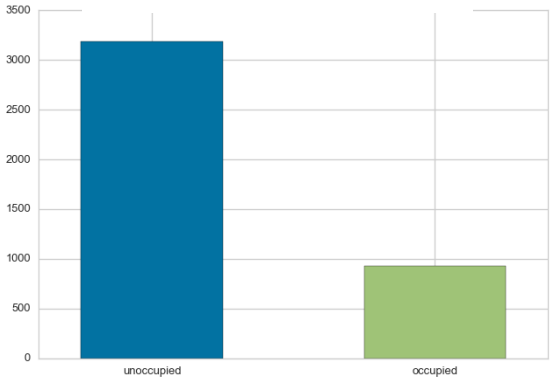
- What to do with a low-accuracy classifier?
- Check for class imbalance.
- Visual cue that we might try stratified sampling, oversampling, or getting more data.

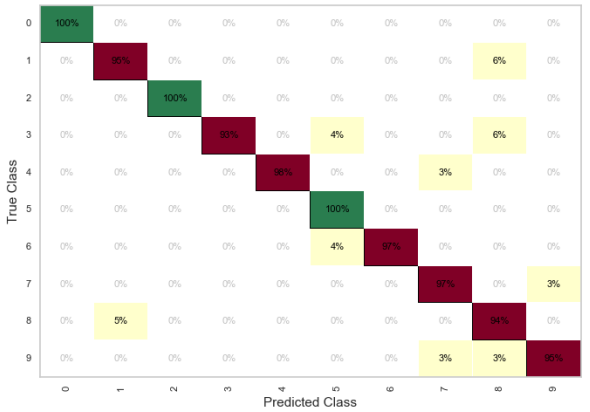
Yellowbrick Score Visualizers



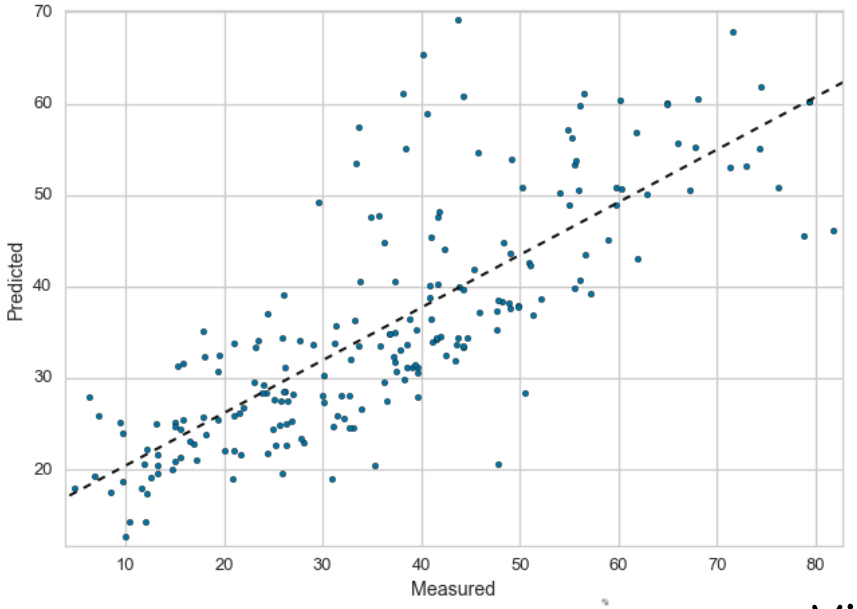
Visualize accuracy and begin to diagnose problems







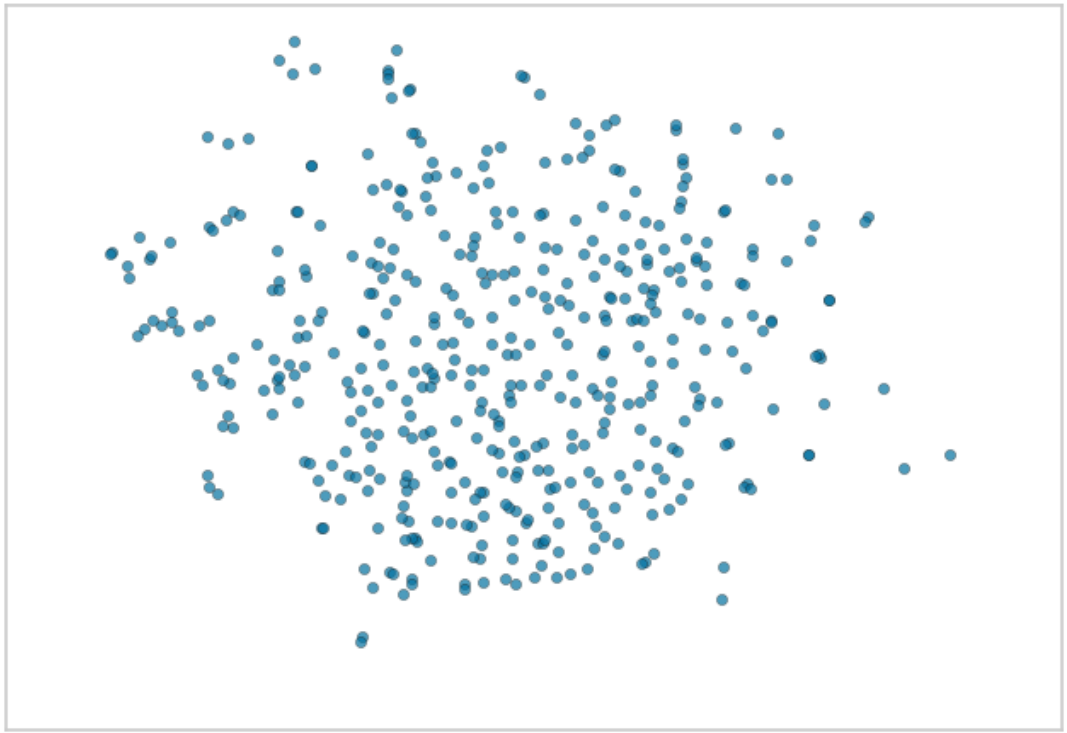
Yellowbrick Score Visualizers



Visualize the distribution of error to diagnose heteroscedasticity

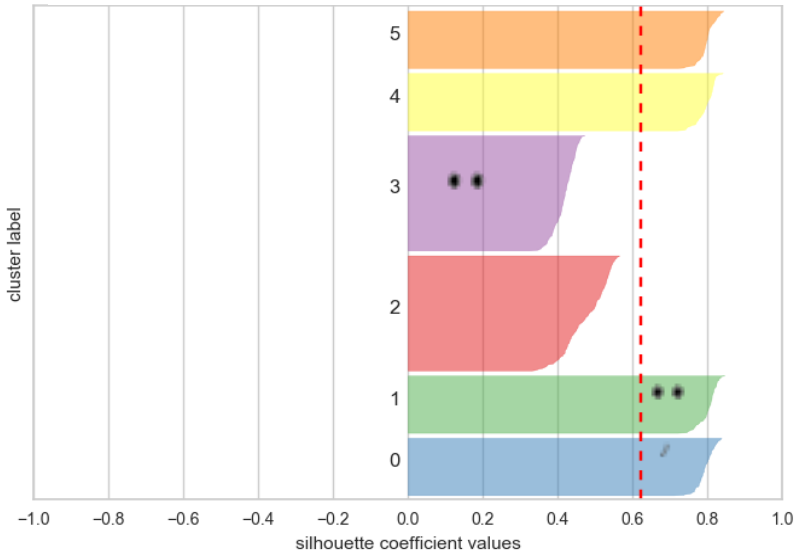
How do I tune this model?

What's the right k?



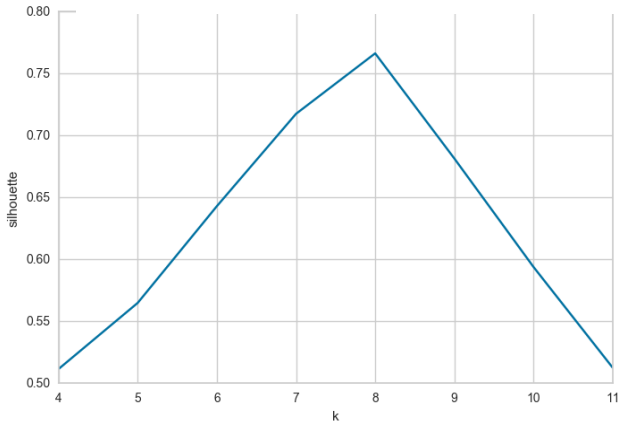
- How many clusters do you see?
- How do you pick an initial value for k in k-means clustering?
- How do you know whether to increase or decrease k?
- Is partitive clustering the right choice?

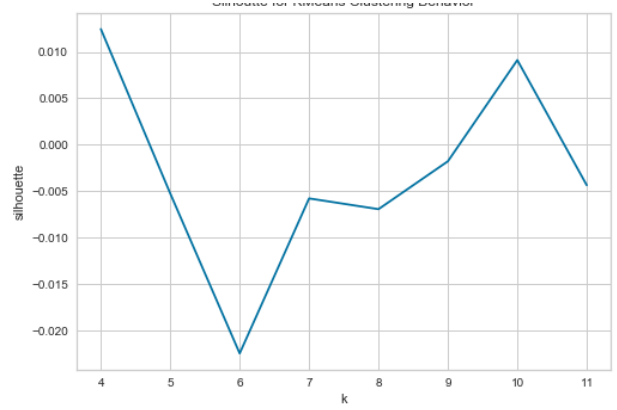
Hyperparameter Tuning



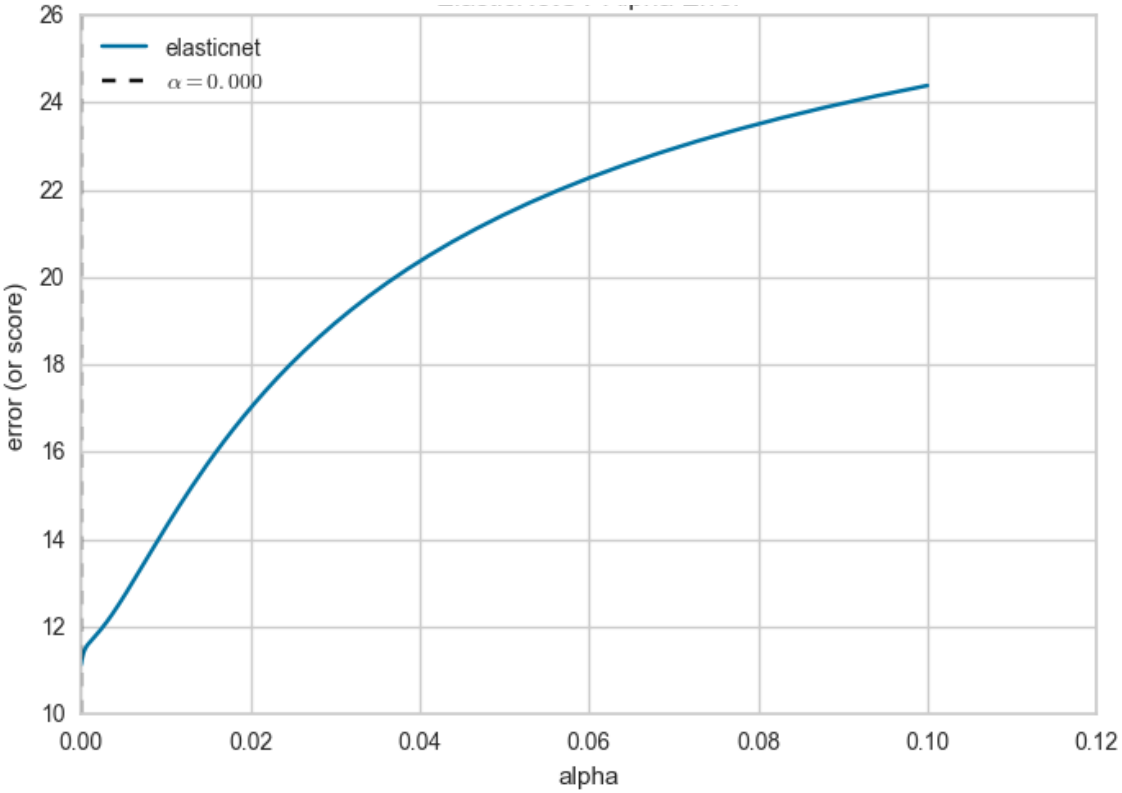
higher silhouette scores mean denser, more separate clusters

The elbow shows the best value of k...
Or suggests a different algorithm





Hyperparameter Tuning



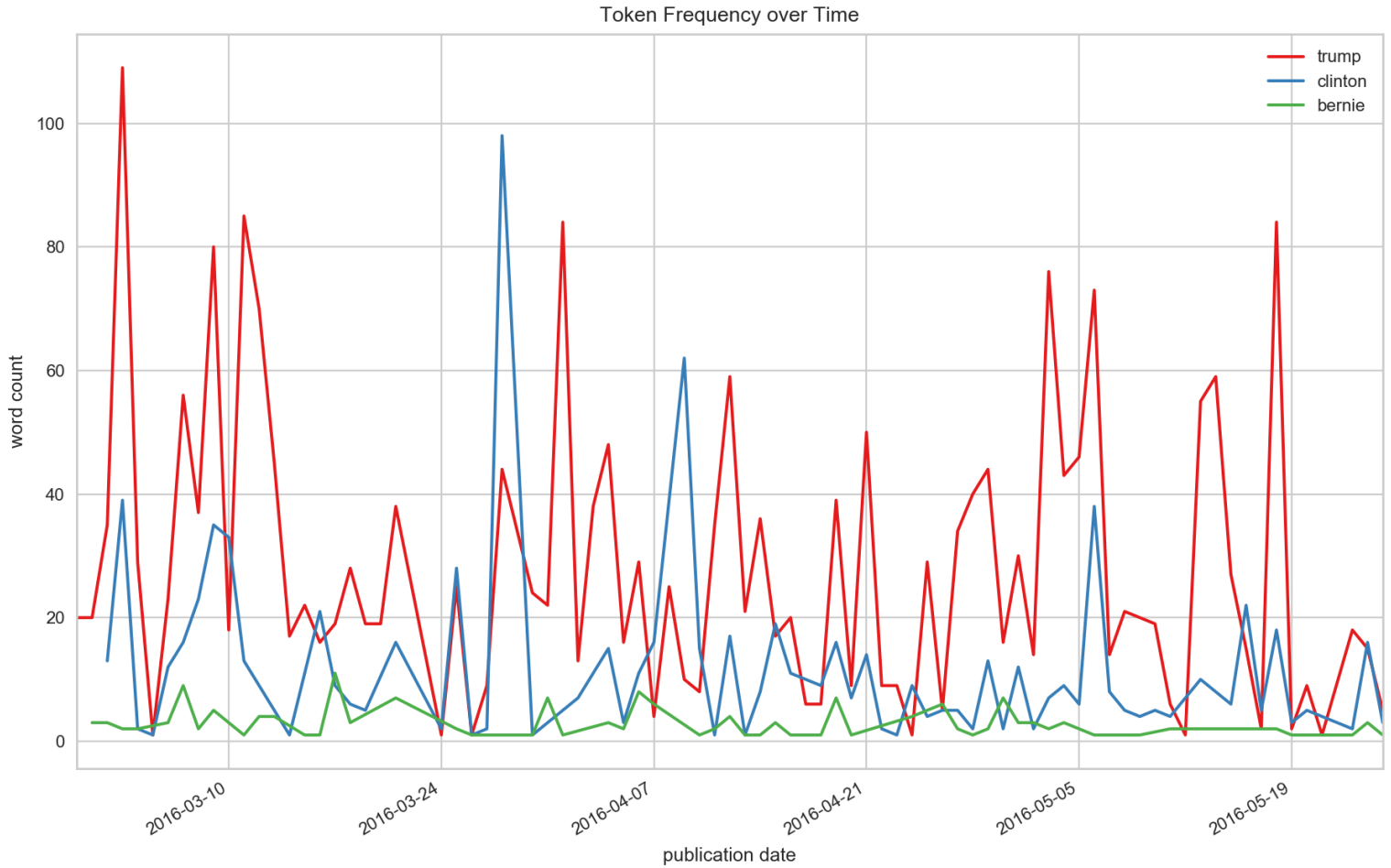
Should I use Lasso, Ridge, or ElasticNet?

Is regularization even working?

What’s next?

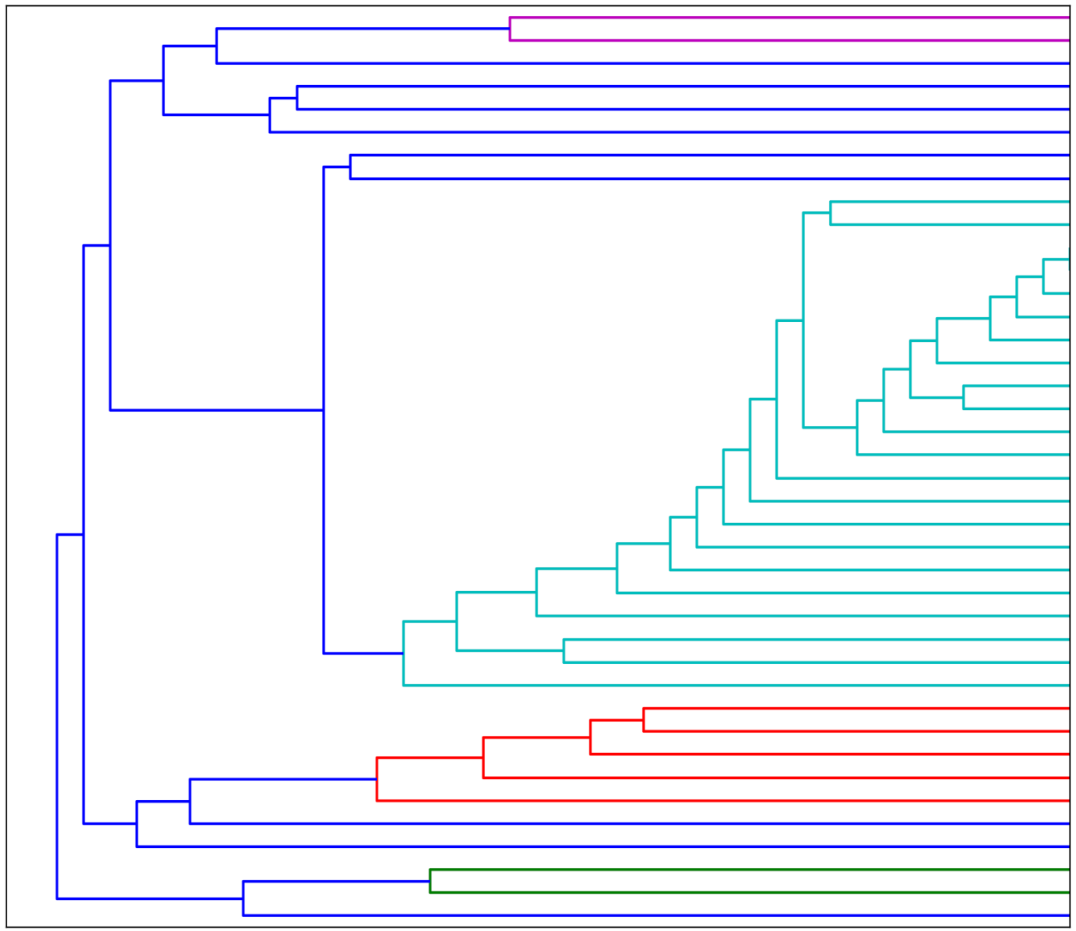
Some ideas…

How does token frequency change over time, in relation to other tokens?



Some ideas…

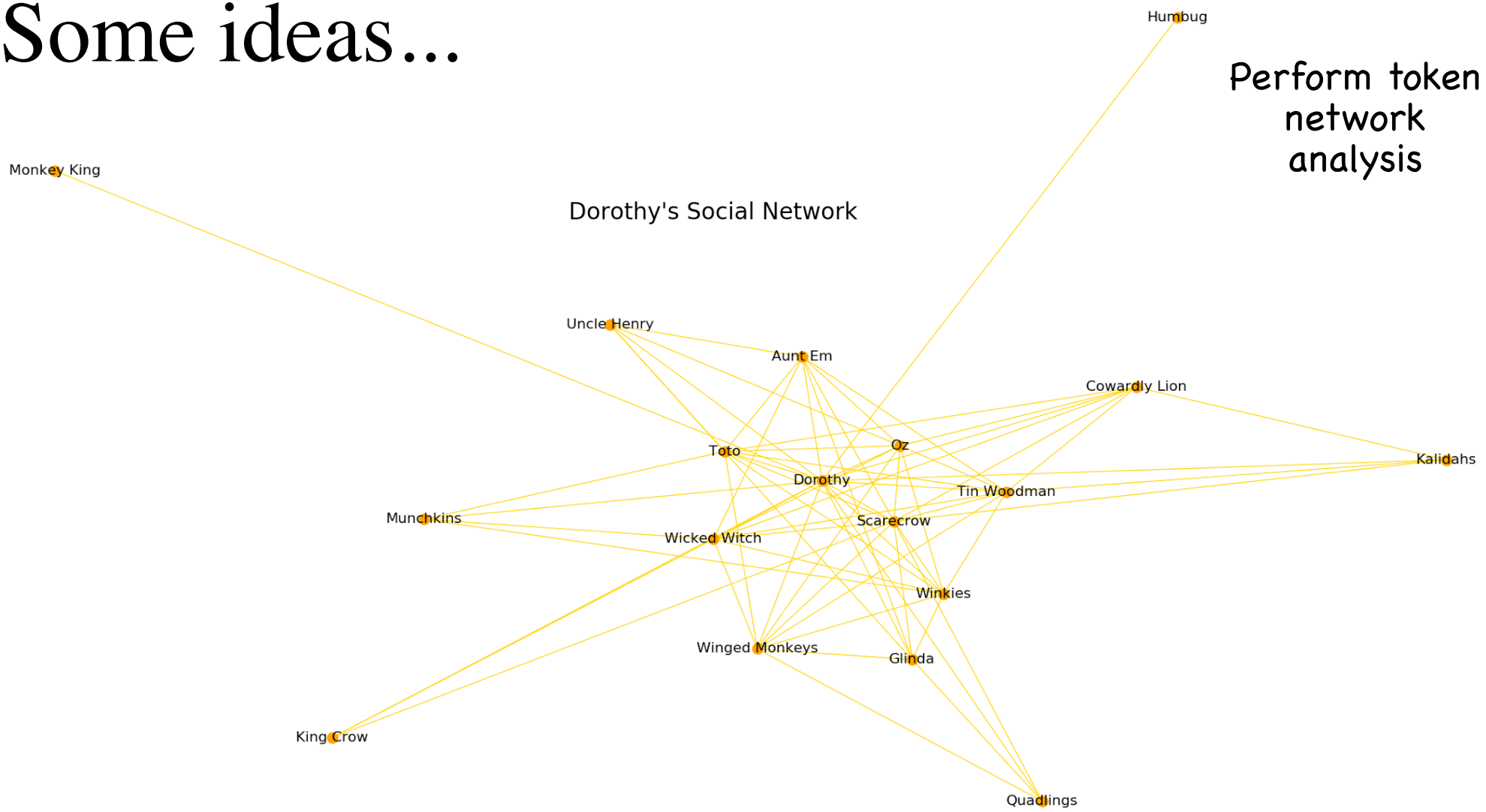
View the corpus hierarchically, after clustering



['say', 'apple', 'fbi', 'government', 'comey', 'tuesday', 'phone', 'iphone', 'ask', 'san']
['say', 'apple', 'fbi', 'encryption', 'comey', 'could', 'drive', 'iphone', 'copy', 'phone']
['tax', 'republican', 'say', 'vote', 'new', 'california', 'gop', 'healthcare', 'message', 'state']
['unit', 'study', 'combat', 'woman', 'marine', 'injury', 'say', 'effectiveness', 'cruz', 'corp']
['kylie', 'jenner', 'minogue', 'trademark', 'notice', 'fashion', 'year', 'note', 'post', 'attempt']
['brain', 'sleep', 'hurley', 'find', 'say', 'iq', 'people', 'score', 'cognitive', 'new']
['billion', '$', 'last', 'year', 'net', 'worth', 'photo', 'forbes', 'spot', 'say']
['superyacht', 'supercar', 'mono', 'photo', '11', 'fit', 'yacht', 'marine', 'edition', 'briggs']
['country', 'africa', 'say', 'erdogan', 'turkey', 'terrorism', 'council', 'new', 'security', 'attack']
['pipeline', 'uganda', 'oil', 'kenyan', 'also', 'tanzanian', 'statement', 'tanzania', 'say', 'route']
['china', 'south', 'carter', 'sea', 'ash', 'foreign', 'beijing', 'defense', 'secretary', 'action']
['billion', 'germany', 'euro', 'trading', 'worth', '2015', 'statement', 'data', 'france', 'partner']
['qualcomm', 'sec', 'company', 'pay', 'hire', 'agree', 'american', 'reuters', 'chinese', 'relative']
['county', 'alcohol', 'sale', 'alabama', 'vote', 'wet', 'go', 'city', 'ashland', 'last']
['school', 'officer', 'video', 'baltimore', 'police', 'affiliate', 'cnn', 'report', 'man', 'happen']
['shoe', 'tell', 'baltimore', 'slow', 'tsa', 'woman', 'sun', 'romero', 'gun', 'carry']
['jalalabad', 'consulate', 'attack', 'afghan', 'group', 'area', 'responsibility', 'tell', 'taliban', 'target']
['china', 'philippine', 'ship', 'atoll', 'report', 'chinese', 'dispute', 'presence', 'say', 'fisherman']
['laden', 'bin', 'qaeda', 'al', 'discuss', 'document', 'mauritania', 'propose', 'million', 'also']
['jones', 'lord', 'trump', 'say', 'klan', 'wright', 'know', 'kkk', 'cnn', 'race']
['court', 'abortion', 'justice', 'case', 'texas', 'scalia', 'clinic', 'appeal', 'woman', 'state']
['ml', 'concede', 'united', 'galaxy', 'go', 'c', 'year', 'stage', 'p', '1']
['facebook', 'work', 'company', 'platform', 'say', 'like', 'codorniou', 'one', 'make', 'workplace']
['zuma', 'south', 'say', 'african', 'africa', 'economy', 'parliament', 'wreck', 'party', 'jacob']
['say', 'msnbc', 'perry', 'harris', 'network', 'show', 'one', 'leave', 'write', 'non']
['mauritania', 'aqim', 'qaeda', 'al', 'say', 'group', 'u', 'government', 'would', 'deal']
['first', 'coach', 'harbaugh', 'spring', 'year', 'new', 'game', 'say', 'park', 'baseball']
['christie', 'chris', 'trump', 'see', 'man', 'eye', 'look', 'come', 'would', 'must']
['say', 'hamilton', 'still', 'officer', 'year', 'crystal', 'old', 'ronald', 'family', '1']
['p', 'gershon', 'grant', 'conduct', 'di', '7', '2017', 'dec', 'work', 'lasso']
['iowa', 'indiana', 'point', '3', '14', 'williams', 'big', 'hawkeyes', 'ten', 'hoosier']
['russell', 'game', 'lakers', 'rebound', 'point', 'night', 'run', 'career', 'second', 'pointer']
['point', 'game', 'score', 'warrior', 'season', 'bull', 'shot', 'miami', 'green', 'three']
['anthony', 'point', 'knicks', 'say', 'game', 'team', 'get', 'fan', 'back', 'lead']
['heat', 'bull', 'game', 'chicago', 'point', 'miami', 'whiteside', 'say', 'score', 'since']
['roseman', 'eagle', 'bradford', 'quarterback', 'kelly', 'free', 'pederson', 'like', 'year', 'lot']
['space', 'kelly', 'crew', 'second', 'nasa', '10', 'cst', 'spacecraft', 'march', 'gmt']
['trump', 'clinton', 'rubio', 'tuesday', 'least', 'delegate', 'say', 'cruz', 'state', 'party']
['trump', 'state', 'win', 'delegate', 'cruz', 'rubio', 'party', 'sander', 'race', 'republican']
['trump', 'say', 'republican', 'go', 'party', 'make', 'new', 'get', 'state', 'rubio']

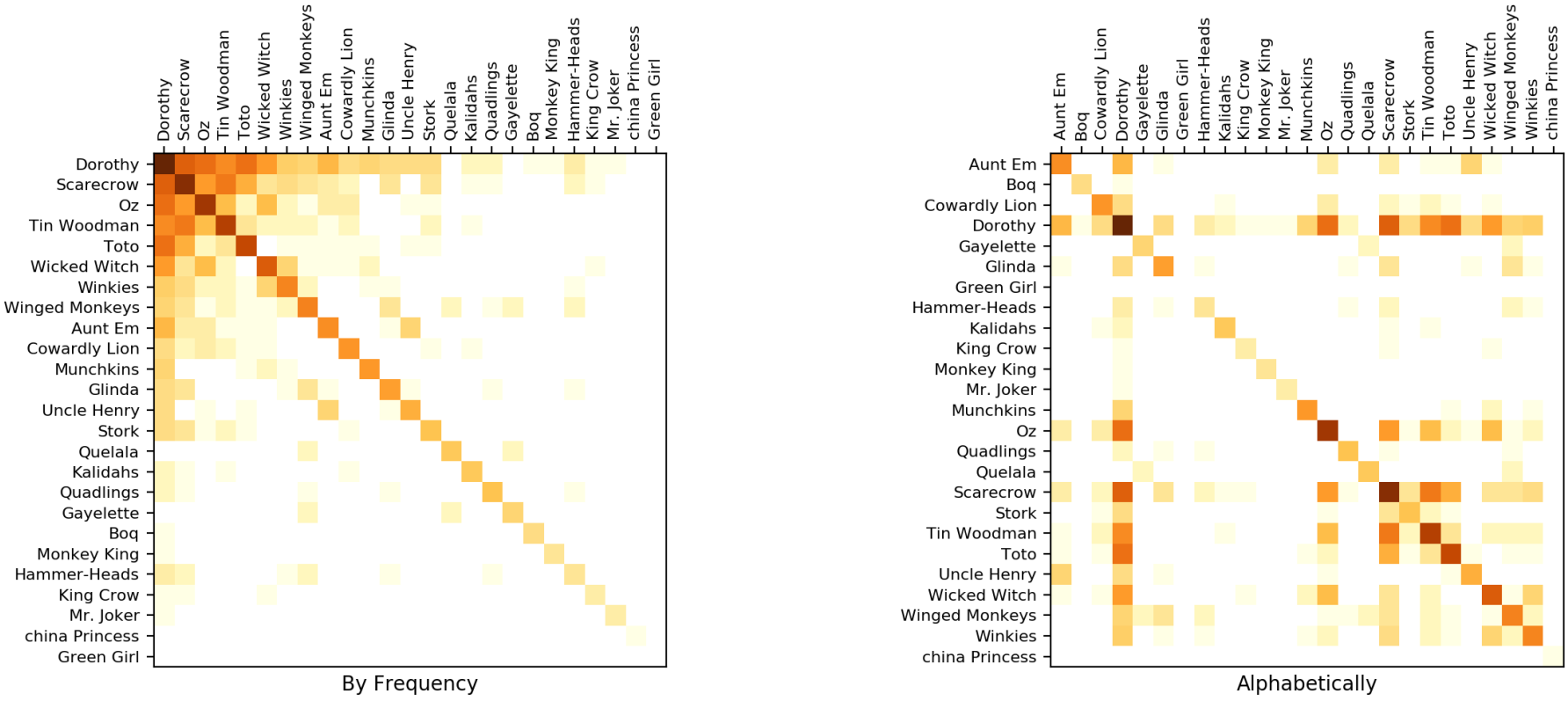
Some ideas…

Perform token network analysis



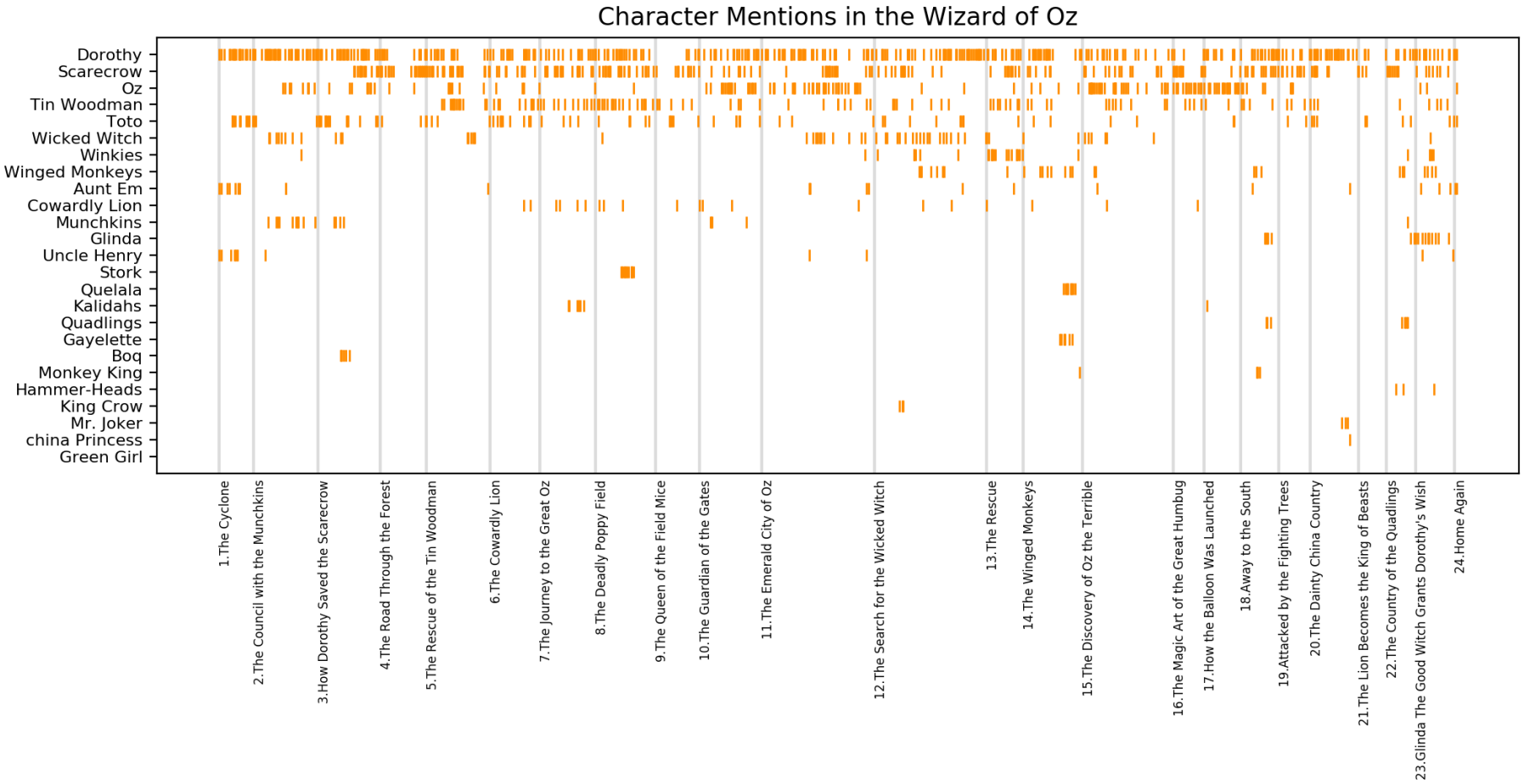
Some ideas…

Plot token co-occurrence



Some ideas…

Get an x-ray of the text



Do you have an idea?

Estimators

The main API implemented by Scikit-Learn is that of the estimator. An estimator is any object that learns from data;

it may be a classification, regression or clustering algorithm, or a transformer that extracts/filters useful features from raw data.

```
class Estimator(object):

    def fit(self, X, y=None):
        """
        Fits estimator to data.
        """
        # set state of self
        return self

    def predict(self, X):
        """
        Predict response of X
        """
        # compute predictions pred
        return pred
```

Transformers

Transformers are special cases of Estimators -- instead of making predictions, they transform the input dataset X to a new dataset X'.

```
class Transformer(Estimator):

  def transform(self, X):
    """
    Transforms the input data.
    """
    # transform X to X_prime
    return X_prime
```

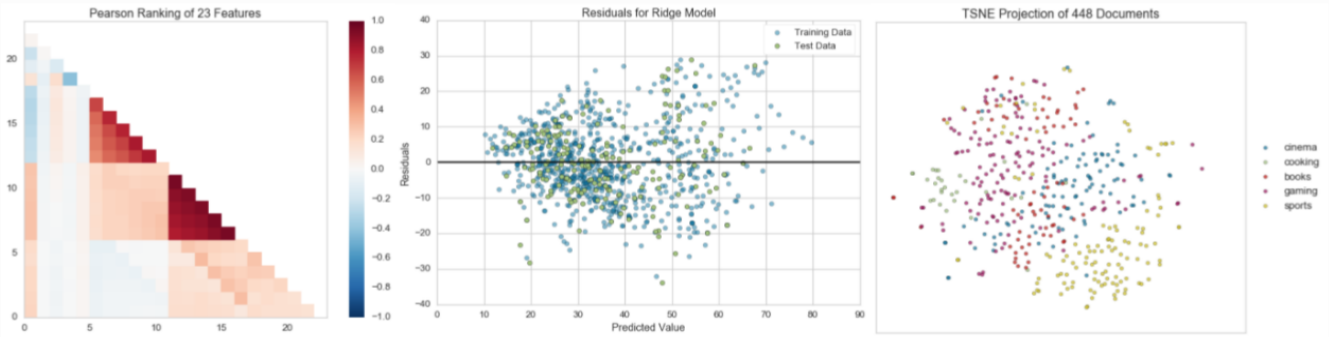
Visualizers

A visualizer is an estimator that produces visualizations based on data rather than new datasets or predictions.

Visualizers are intended to work in concert with Transformers and Estimators to shed light onto the modeling process.

```
class Visualizer(Estimator):

  def draw(self):
      """
      Draw the data
      """
      self.ax.plot()

  def finalize(self):
      """
      Complete the figure
      """
      self.ax.set_title()

  def poof(self):
      """
      Show the figure
      """
      plt.show()
```

Yellowbrick: Machine Learning Visualization



Yellowbrick is a suite of visual diagnostic tools called "Visualizers" that extend the Scikit-Learn API to allow human steering of the model selection process. In a nutshell, Yellowbrick combines Scikit-Learn with Matplotlib in the best tradition of the Scikit-Learn documentation, but to produce visualizations for *your* models! For more on Yellowbrick, please see the About.

If you're new to Yellowbrick, checkout the Quick Start or skip ahead to the Model Selection Tutorial. Yellowbrick is a rich library with many Visualizers being added on a regular basis. For details on specific Visualizers and extended usage head over to the Visualizers and API. Interested in contributing to Yellowbrick? Checkout the contributing guide . If you've signed up to do user testing, head over to the User Testing Instructions (and thank you!).

Thank you!

Twitter: twitter.com/rebeccabilbro

Github: github.com/rebeccabilbro

Email: rebecca.bilbro@bytecubed.com