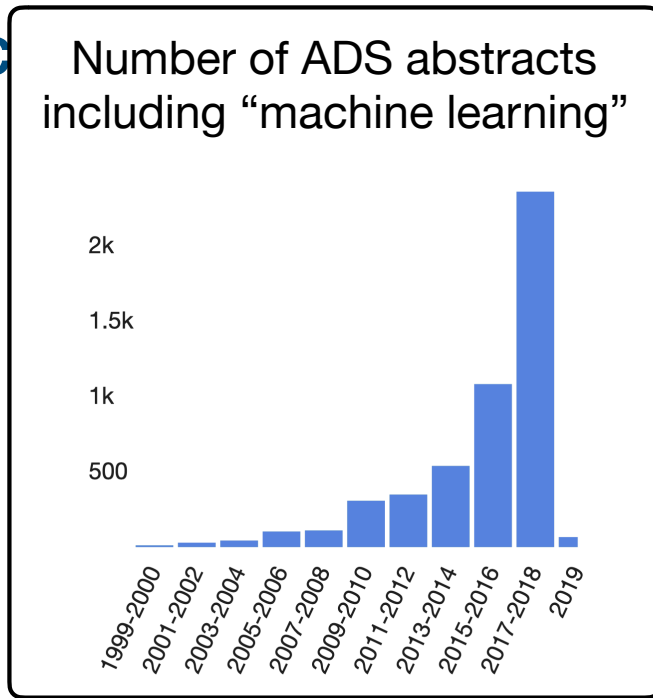


A Typical User's Ground-Level Perspective on Machine Learning in Astronomy

James R. A. Davenport



A Typical User's Ground-Level Perspective on Macroeconomy



Machine Learning is Boring*

Machine Learning is Boring*

*normal, lots of people doing it.
And this is a good thing! (i.e. don't be scared!)

Most of our work isn't "big data"

(i.e. can run most algorithms on your laptop)


Yet sample size is big enough that
interesting/rare things can be found

Our data is becoming better suited for ML

- big datasets (Mario's talk), especially Gaia!
- easier than ever to get data (VizieR/Xmatch, ADS, journals, Github, Zenodo...)
- value-added datasets for surveys (e.g. stellar parameters from SDSS)

ML is easier than ever to use

- robust, open source libraries available
- many programming languages
- many domain (astro) experts & workshops available



[Home](#)
[Installation](#)
[Documentation](#)
[Examples](#)

[Google Custom Search](#)

Previous
Glossary of C...
Next
Isotonic
Regression

scikit-learn v0.20.2
 Other versions

Please **cite us** if you use the software.

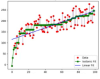
Examples

- Miscellaneous examples
- Examples based on real world datasets
- Biclustering
- Calibration
- Classification
- Clustering
- Pipelines and composite estimators
- Covariance estimation
- Cross decomposition
- Dataset examples
- Decomposition
- Ensemble methods
- Tutorial exercises
- Feature Selection
- Gaussian Process for Machine Learning
- Generalized Linear Models
- Manifold learning
- Gaussian Mixture Models
- Model Selection
- Multicouput methods
- Nearest Neighbors
- Neural Networks
- Preprocessing
- Semi Supervised Classification
- Support Vector Machines
- Working with text documents
- Decision Trees


Examples

Miscellaneous examples

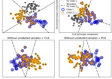
Miscellaneous and introductory examples for scikit-learn.




Isotonic Regression




Face completion with a multi-output estimators



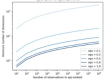
Multilabel classification



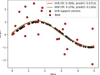
Comparing anomaly detection algorithms for outlier detection on toy datasets




Imputing missing values before building an estimator



The Johnson-Lindenstrauss bound for embedding with random projections





also astroML!
(see Brigitta's talk later)

astroML Home User Guide Book Figures Examples Plots Google Custom Search

AstroML: Machine Learning and Data Mining for Astronomy

News

January 2014: the textbook accompanying astroML is now available! View it on Amazon.

November 2013: astroML 0.2 has been released! Get the source on GitHub

Our Introduction to astroML paper received the CIDU 2012 best paper award.

Links

astroML Mailing List
GitHub Issue Tracker

Videos

Scipy 2012 (15 minute talk)
Scipy 2013 (20 minute talk)

Citing

If you use the software, please consider citing astroML.

AstroML is a Python module for machine learning and data mining built on numpy, scipy, scikit-learn, matplotlib, and astropy, and distributed under the 3-clause BSD license. It contains a growing library of statistical and machine learning routines for analyzing astronomical data in Python, loaders for several open astronomical datasets, and a large suite of examples of analyzing and visualizing astronomical datasets.

Downloads

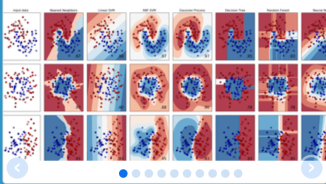
- Released Versions: Python Package Index
- Bleeding-edge Source: github

The goal of astroML is to provide a community repository for fast Python implementations of common tools and routines used for statistical data analysis in astronomy and astrophysics, to provide a uniform and easy-to-use interface to freely available astronomical datasets. We hope this package will be useful to researchers and students of astronomy. If you have an example you'd like to share, we are happy to accept a contribution via a GitHub Pull Request: the code repository can be found at <http://github.com/astroML/astroML>.

Textbook

The astroML project was started in 2012 to accompany the book **Statistics, Data Mining, and Machine Learning in Astronomy** by Zeljko Ivezic, Andrew Connolly, Jacob VanderPlas, and Alex Gray, published by Princeton University Press. The table of contents is available [here \(pdf\)](#), or you can [preview](#) or [purchase](#) the

Problems that ML is good for:



scikit-learn

Machine Learning in Python

- Simple and efficient tools for data mining and data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

Classification

Identifying to which category an object belongs to.
Applications: Spam detection, Image recognition.
Algorithms: SVM, nearest neighbors, random forest, ... — Examples

Regression

Predicting a continuous-valued attribute associated with an object.
Applications: Drug response, Stock prices.
Algorithms: SVR, ridge regression, Lasso, ... — Examples

Clustering

Automatic grouping of similar objects into sets.
Applications: Customer segmentation, Grouping experiment outcomes
Algorithms: k-Means, spectral clustering, mean-shift, ... — Examples

Dimensionality reduction

Reducing the number of random variables to consider.
Applications: Visualization, Increased efficiency
Algorithms: PCA, feature selection, non-negative matrix factorization. — Examples

Model selection

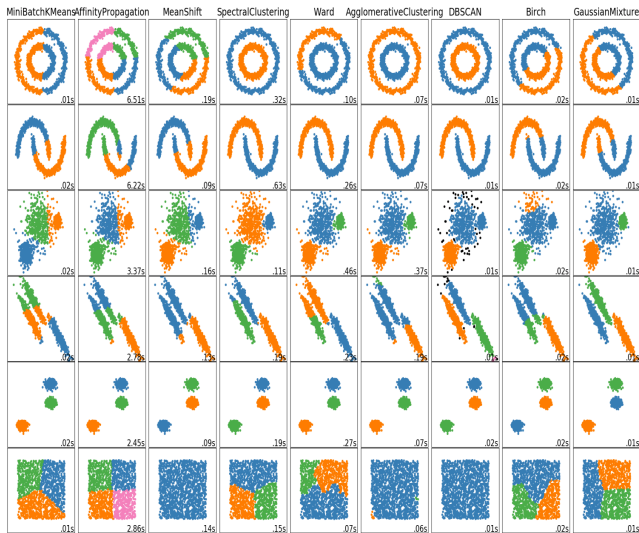
Comparing, validating and choosing parameters and models.
Goal: Improved accuracy via parameter tuning
Modules: grid search, cross validation, metrics. — Examples

Preprocessing

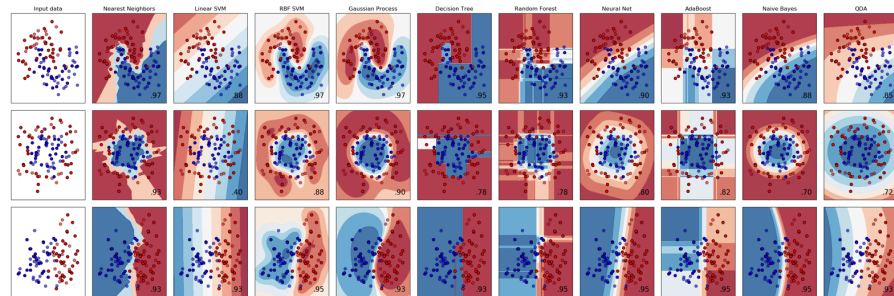
Feature extraction and normalization.
Application: Transforming input data such as text for use with machine learning algorithms.
Modules: preprocessing, feature extraction. — Examples

Each algorithm has specific use cases
(sometimes: just try them all!)

Clustering



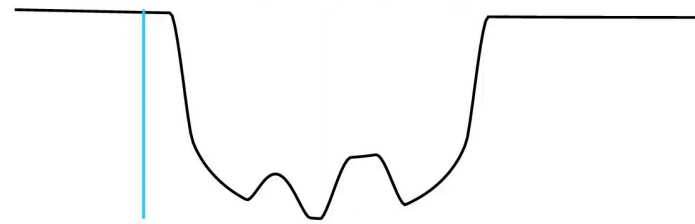
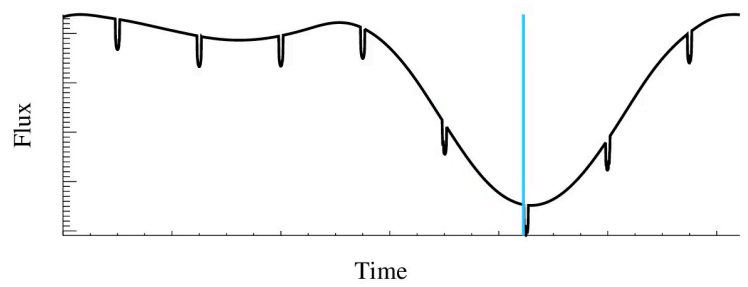
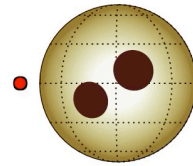
Classification



**Goal: demonstrate 3 real problems
that ML can be used for**

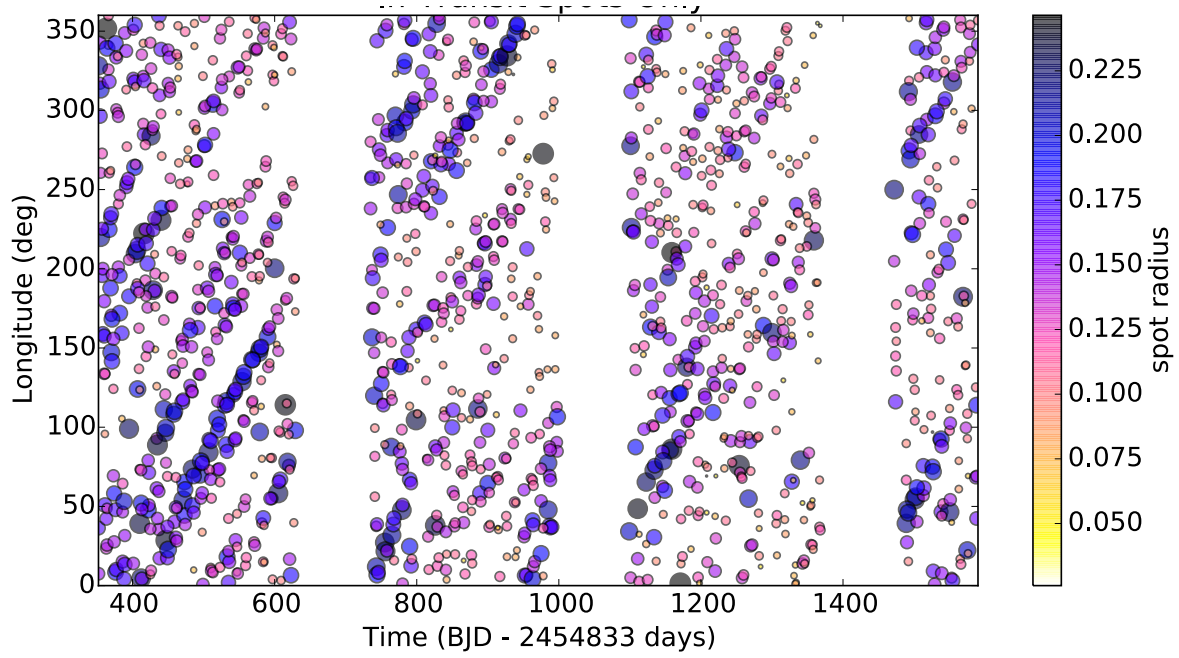
Example 1: Clustering starspot evolution tracks

Kepler 17



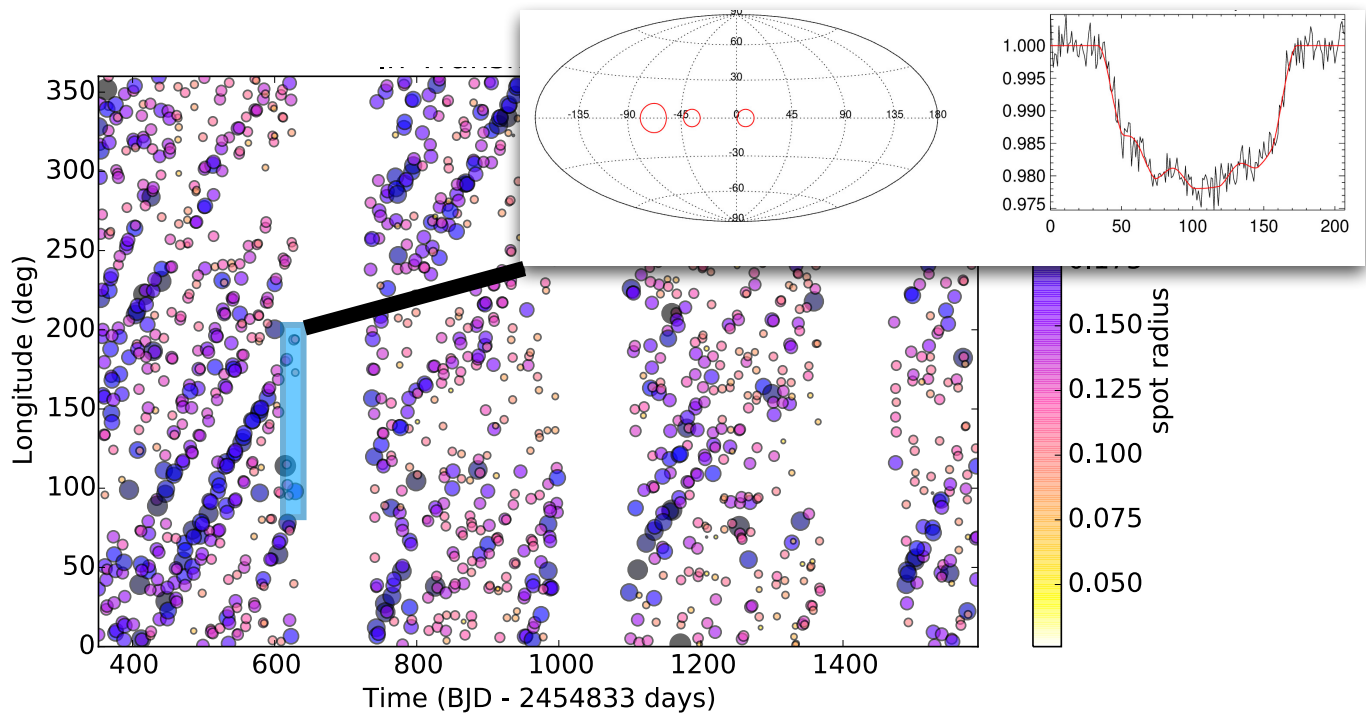
Example 1: Clustering starspot evolution tracks

Kepler 17



Example 1: Clustering starspot evolution tracks

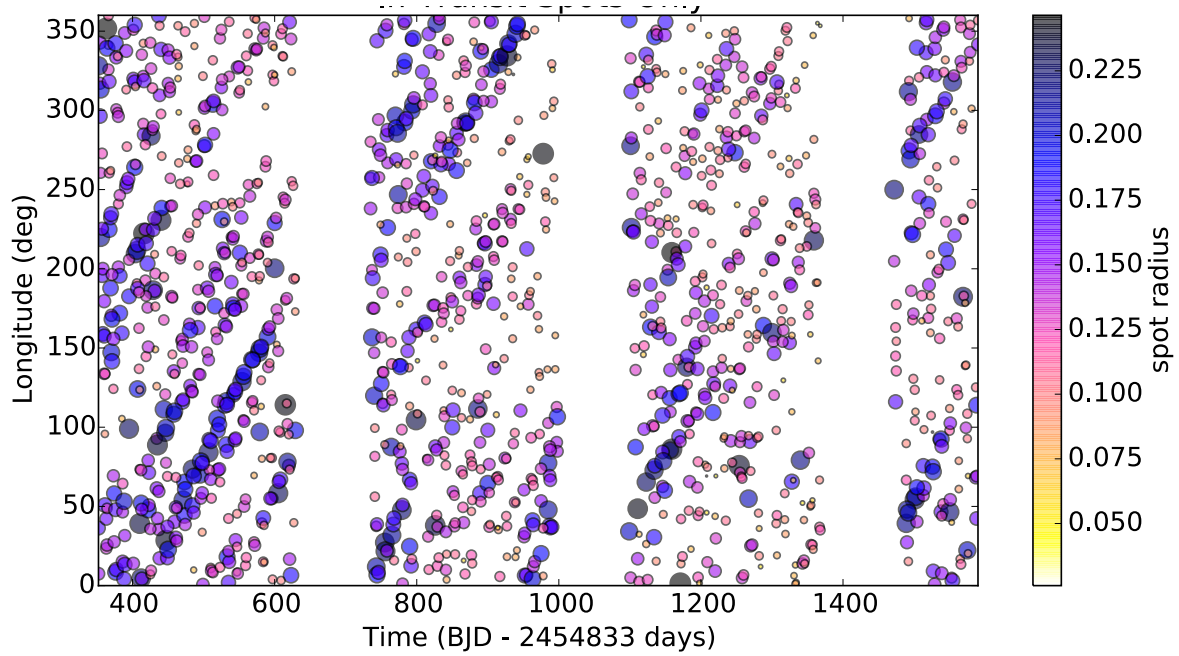
Kepler 17



Example 1: Clustering starspot evolution tracks

Kepler 17

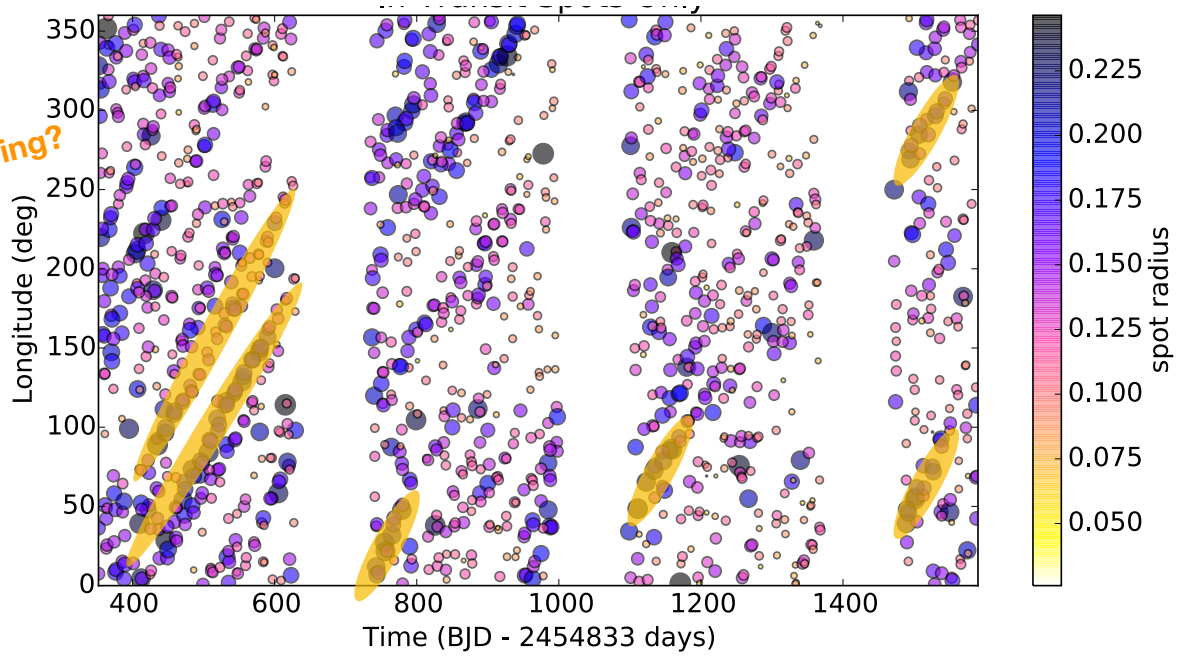
Question: Which tracks are starspots, how do they emerge/decay?



Example 1: Clustering starspot evolution tracks

Kepler 17

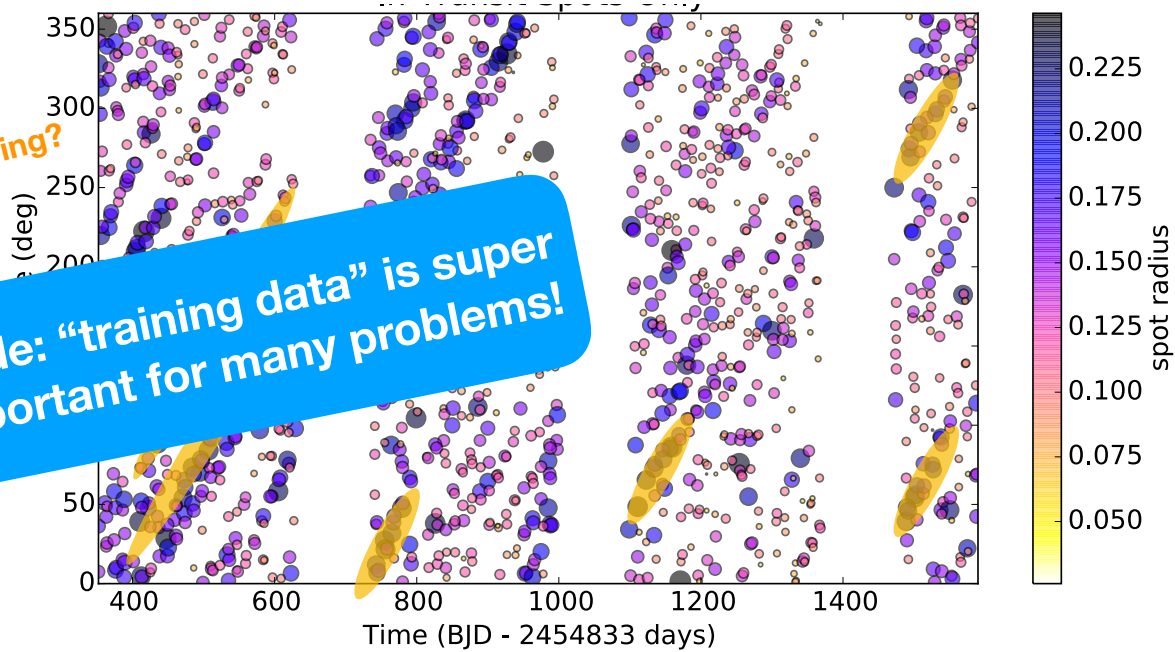
Question: Which tracks are starspots, how do they emerge/decay?



Example 1: Clustering starspot evolution tracks

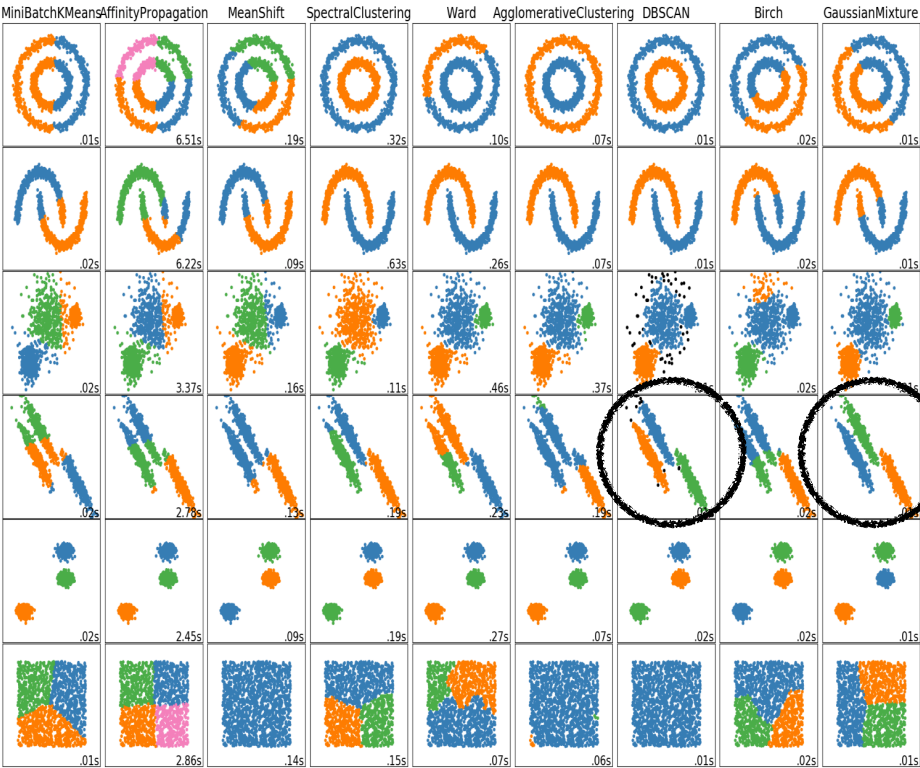
Kepler 17

Question: Which tracks are starspots, how do they emerge/decay?



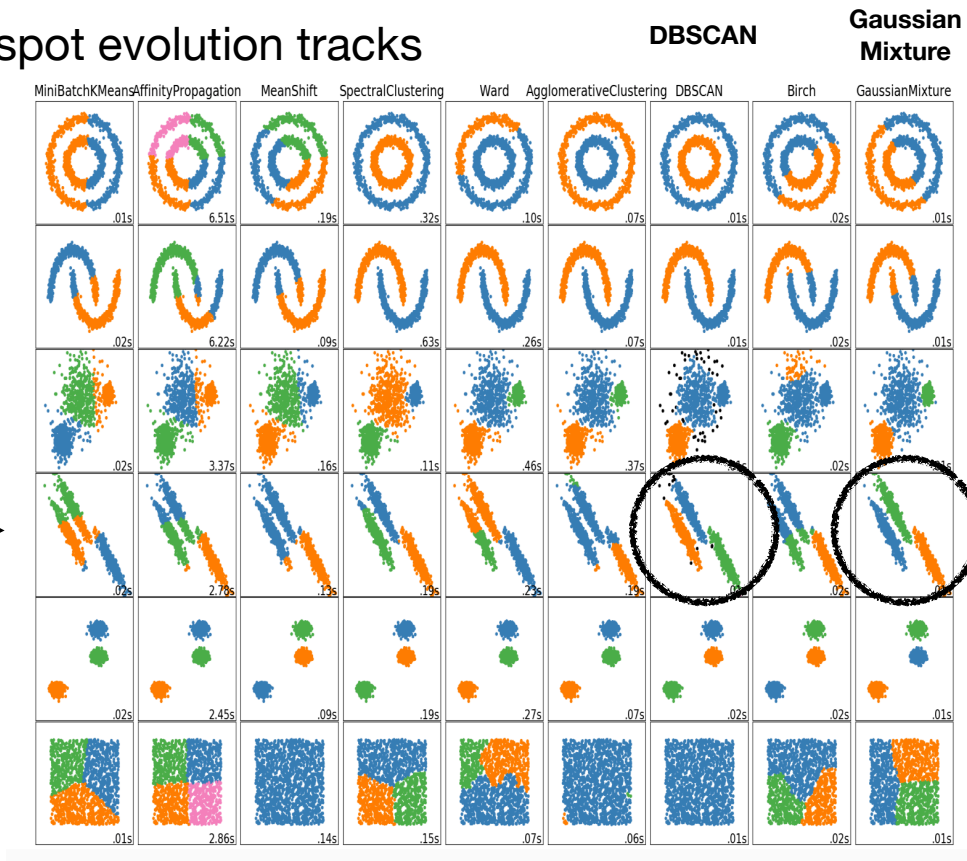
Example 1: Clustering starspot evolution tracks

DBSCAN **Gaussian Mixture**



Example 1: Clustering starspot evolution tracks

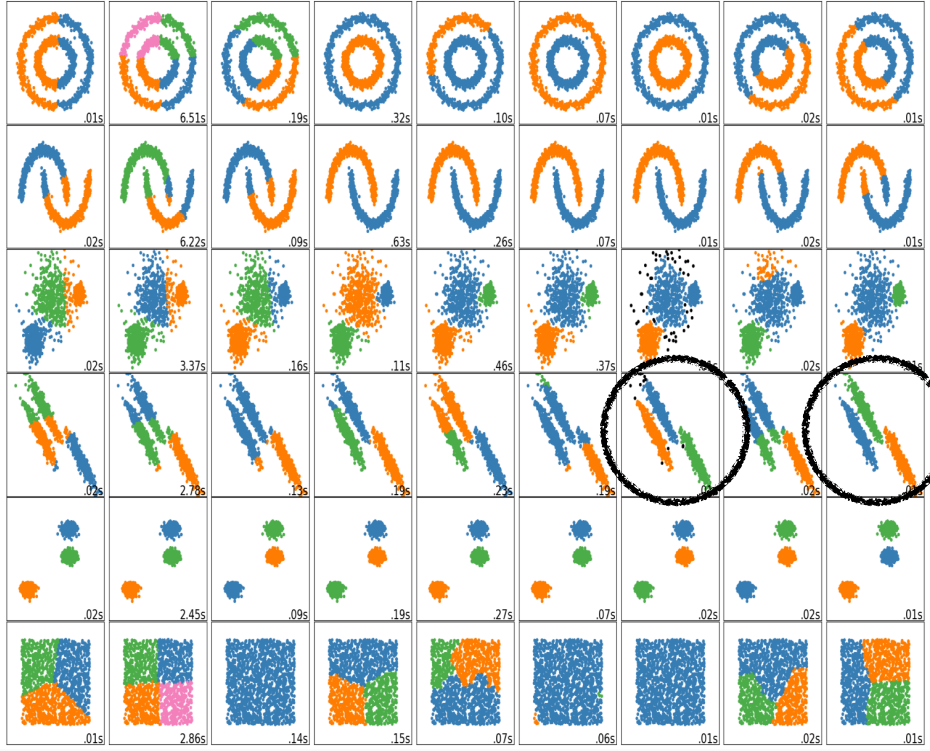
need to predefine Nclusters?



DBSCAN

Gaussian Mixture

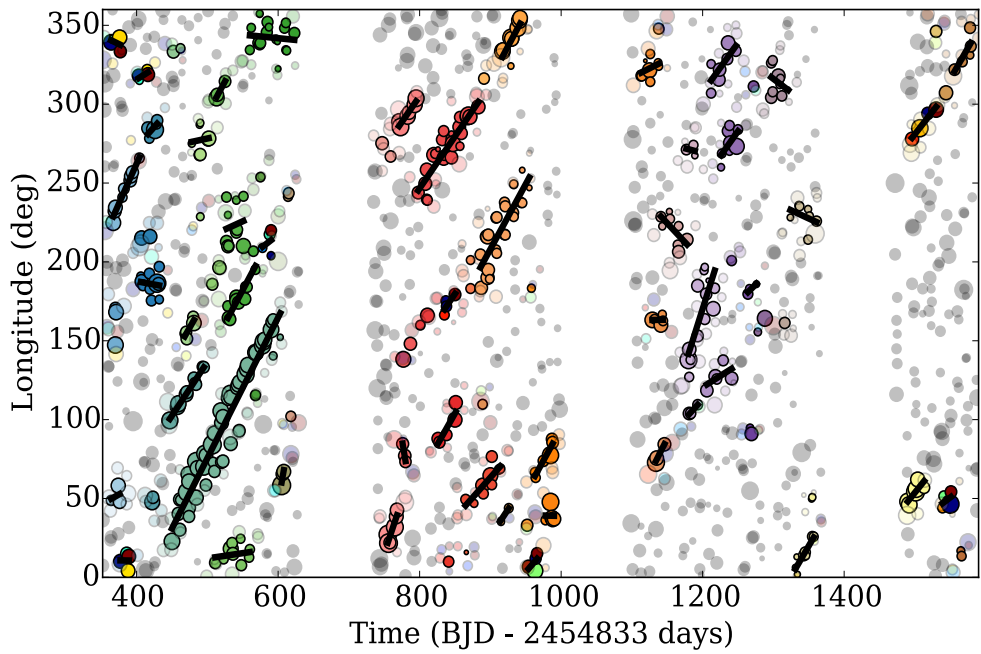
MiniBatchKMeans AffinityPropagation MeanShift SpectralClustering Ward AgglomerativeClustering DBSCAN Birch GaussianMixture



Example 1: Clustering starspot evolution tracks

Kepler 17

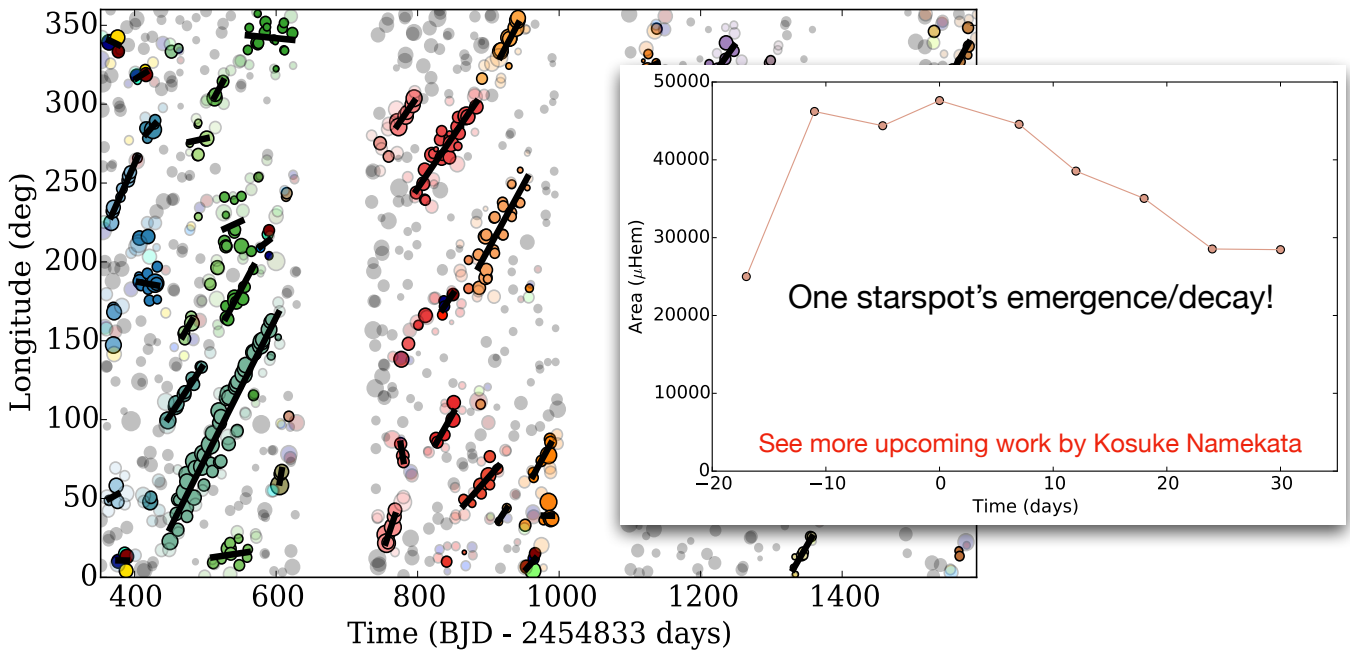
DBSCAN: Density-based spatial clustering of applications with noise



Example 1: Clustering starspot evolution tracks

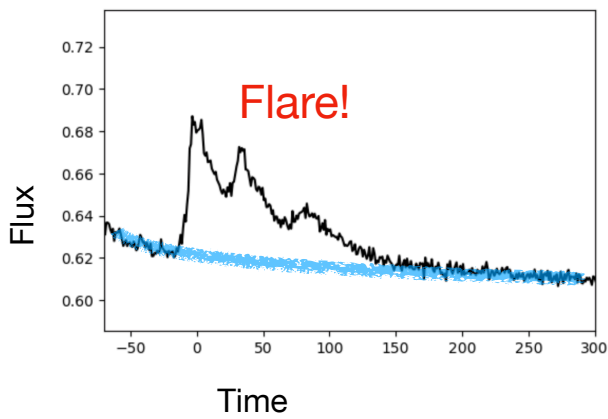
Kepler 17

DBSCAN: Density-based spatial clustering of applications with noise



Example 2: Modeling a complex stellar flare

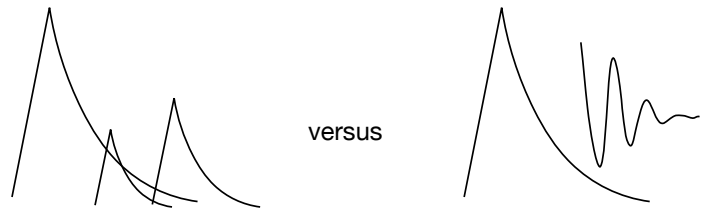
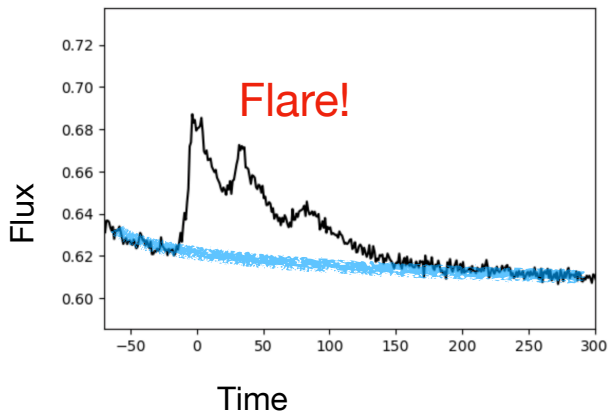
Question: Is there (quasi-) sinusoidal behavior in the flare decay?



<https://github.com/RileyWClarke/QPP-GP>

Example 2: Modeling a complex stellar flare

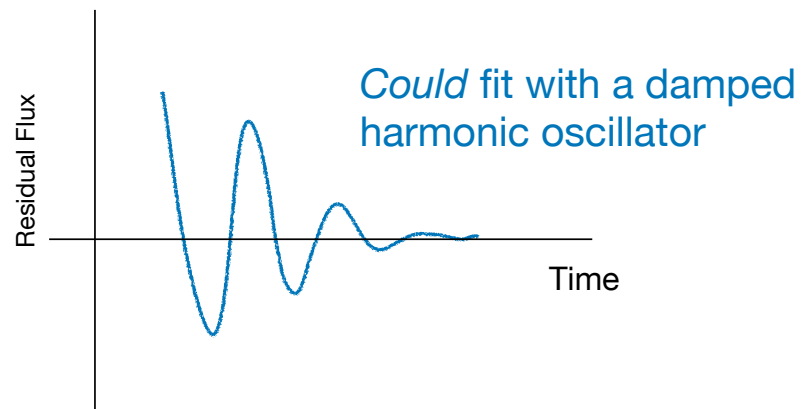
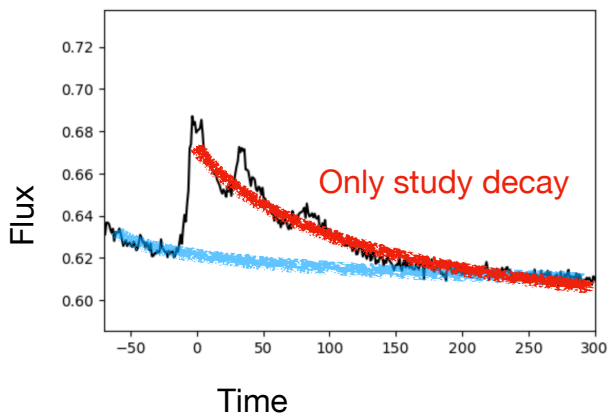
Question: Is there (quasi-) sinusoidal behavior in the flare decay?



<https://github.com/RileyWClarke/QPP-GP>

Example 2: Modeling a complex stellar flare

Question: Is there (quasi-) sinusoidal behavior in the flare decay?

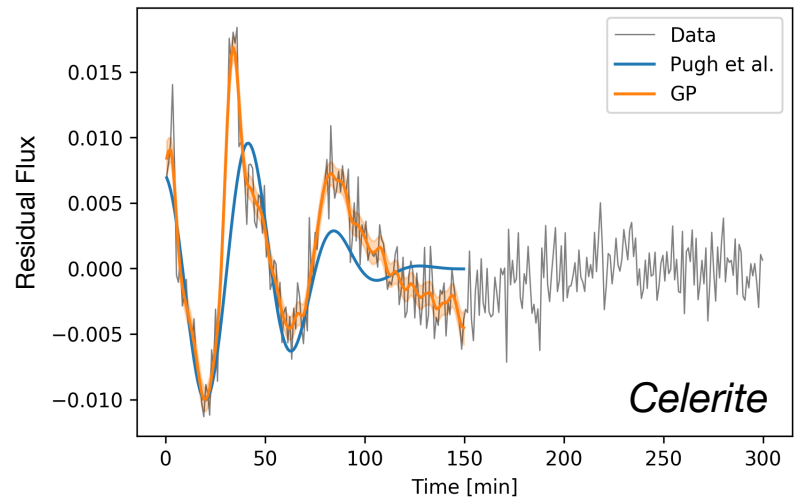


Difficult to classify sinusoidal vs. stochastic, & strict vs quasi sinusoid

<https://github.com/RileyWClarke/QPP-GP>

Example 2: Modeling a complex stellar flare

Gaussian Process

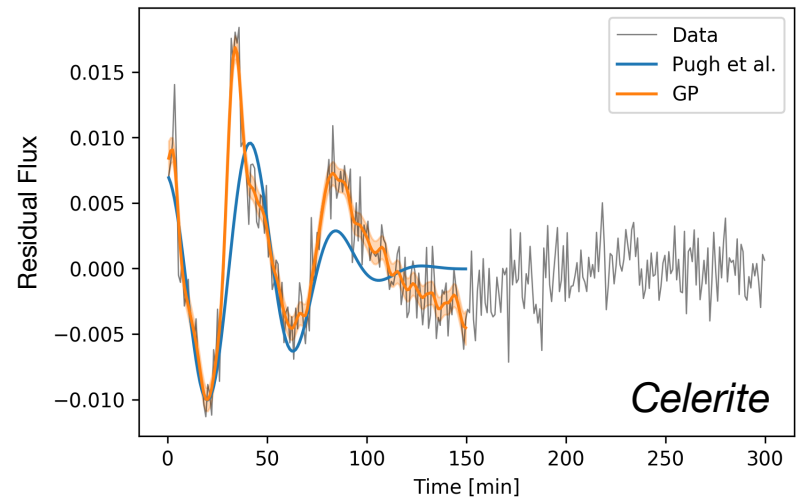
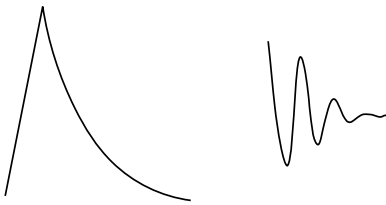


<https://github.com/RileyWClarke/QPP-GP>

Example 2: Modeling a complex stellar flare

Gaussian Process

Use an exponential +
simple-harmonic-oscillator *kernel*



<https://github.com/RileyWClarke/QPP-GP>

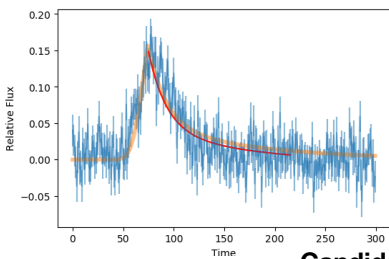
Example 2: Modeling a complex stellar flare

Gaussian Process

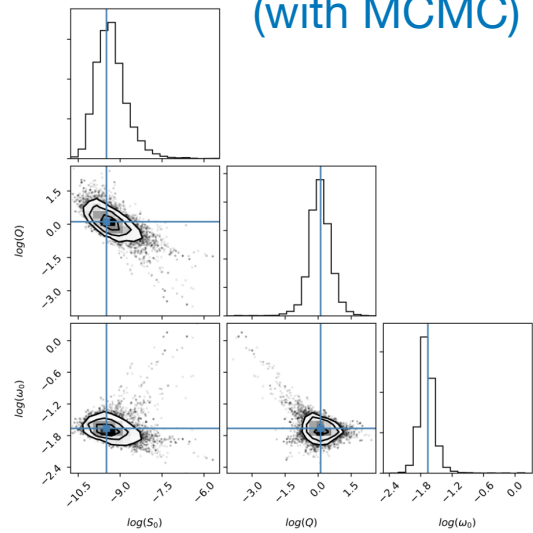
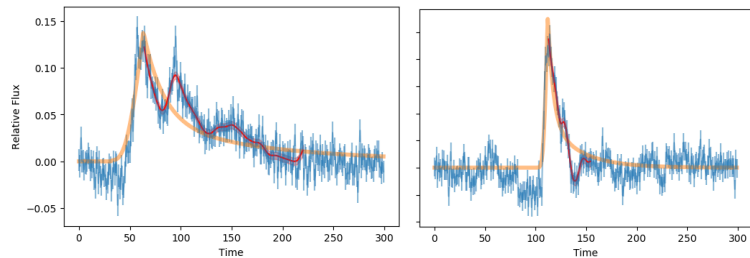
No Period

Objective search

Robust uncertainties!
(with MCMC)



Candidate Periods

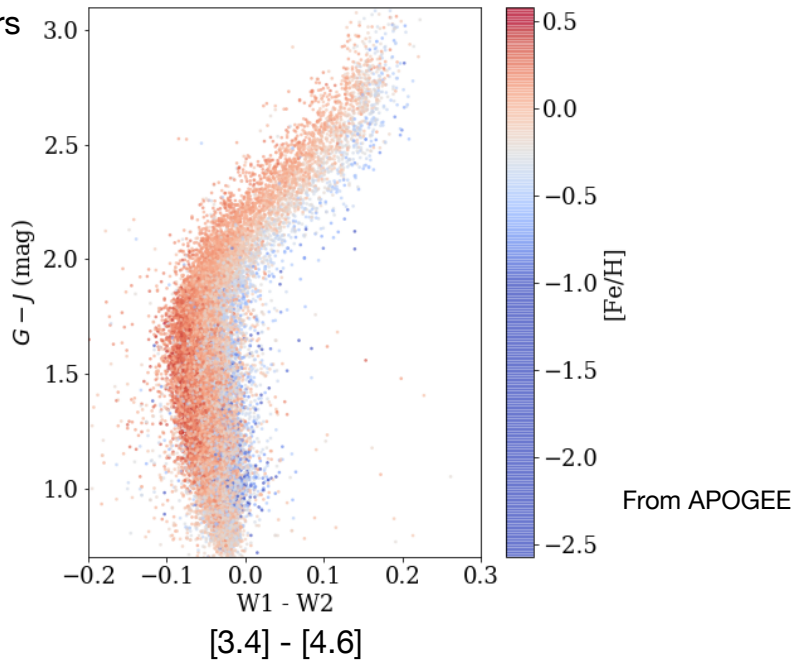


<https://github.com/RileyWClarke/QPP-GP>



Example 3: Modeling photometric metallicities

Observation: [Fe/H] gradient in stars

Gaia DR2 + (WISE + 2MASS) + APOGEE



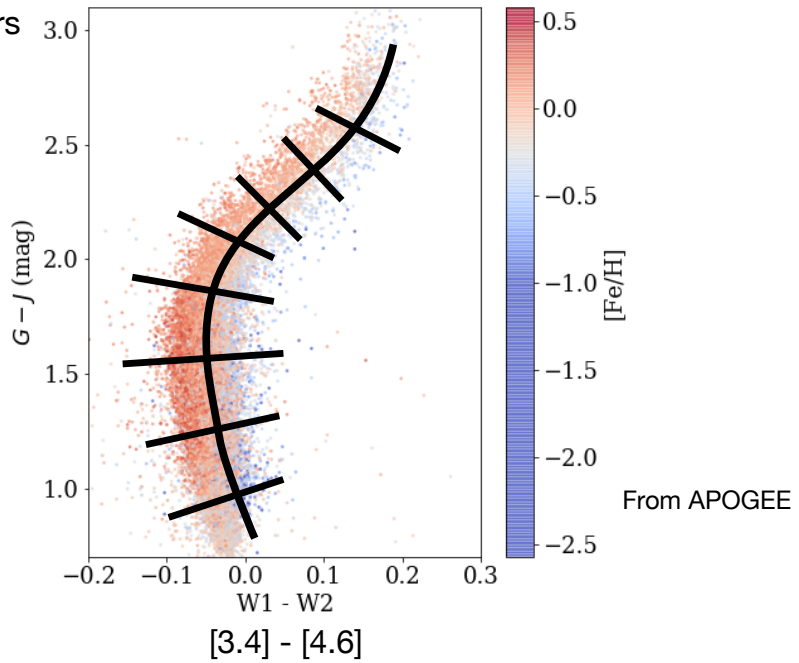
<https://github.com/jradavenport/ingot/>

  jradavenport



Example 3: Modeling photometric metallicities

Observation: $[Fe/H]$ gradient in stars

We could build a complex polynomial or spline model



<https://github.com/jradavenport/ingot/>

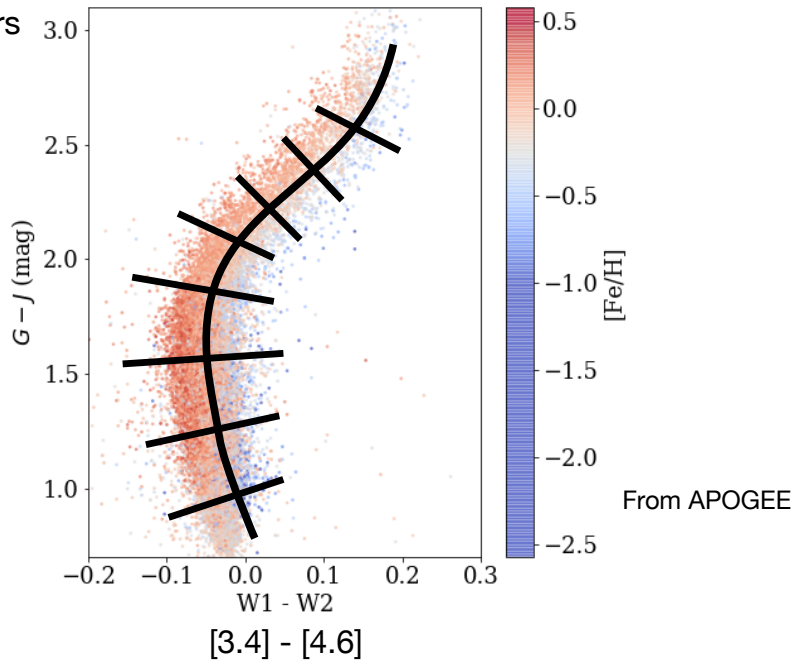
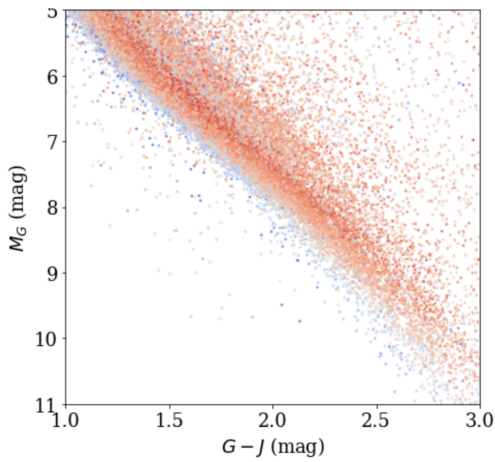
  jradavenport

Example 3: Modeling photometric metallicities



Observation: $[Fe/H]$ gradient in stars

We could build a complex polynomial or spline model

Tedious, and difficult to add additional dimensions!



<https://github.com/jradavenport/ingot/>

  jradavenport

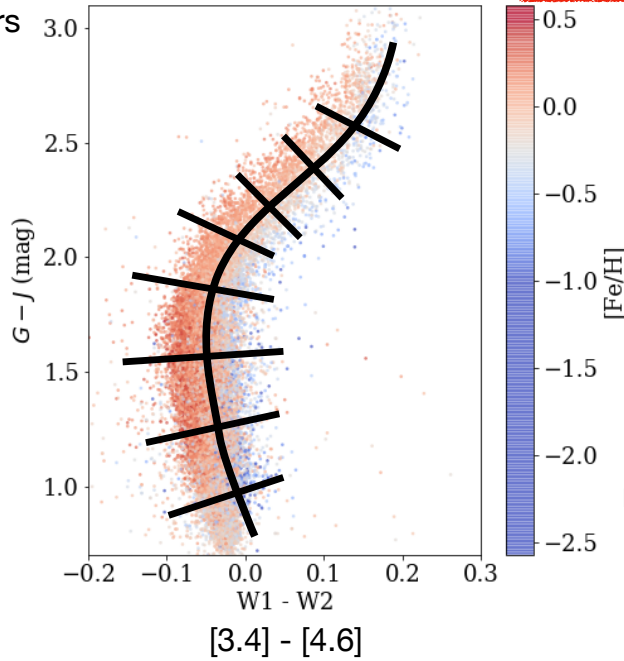
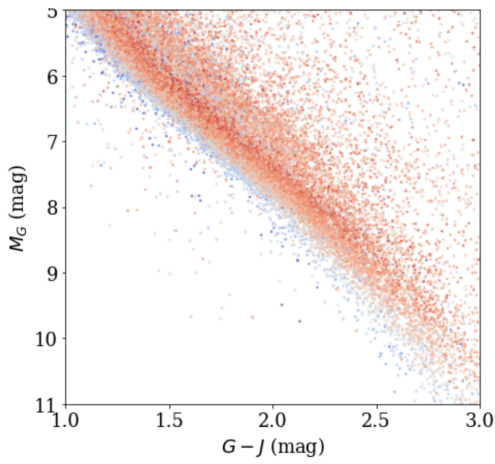
Example 3: Modeling photometric metallicities

Or use a simple, flexible ML model!

Observation: $[Fe/H]$ gradient in stars



We could build a complex polynomial or spline model

Tedious, and difficult to add additional dimensions!



From APOGEE

<https://github.com/jradavenport/ingot/>

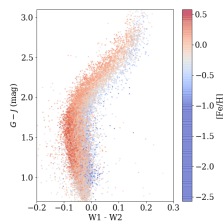
  jradavenport

Example 3: Modeling photometric metallicities

KNearestNeighbors

Xdata = (G-J, W1-W2)

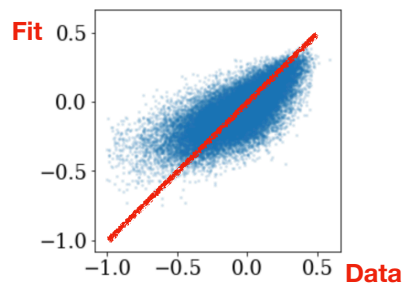
Ydata = [Fe/H]



```
In [43]: model = KNeighborsRegressor(n_neighbors=5)
         model.fit(Xdata, Ydata)
         newY = model.predict(Xdata)

         plt.figure(figsize=(4,4))
         plt.scatter(Ydata, newY, s=3, alpha=0.14)
         # plt.xlim(-.5, .5)
         plt.ylim(plt.xlim())
```

Out[43]: (-1.0819262972727413, 0.6647018304720514)

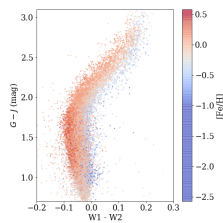


Example 3: Modeling photometric metallicities

KNearestNeighbors

Xdata = (G-J, W1-W2)

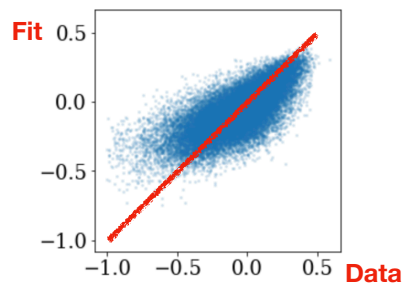
Ydata = [Fe/H]



```
In [43]: model = KNeighborsRegressor(n_neighbors=5)
         model.fit(Xdata, Ydata)
         newY = model.predict(Xdata)

         plt.figure(figsize=(4,4))
         plt.scatter(Ydata, newY, s=3, alpha=0.14)
         # plt.xlim(-.5, .5)
         plt.ylim(plt.xlim())
```

Out[43]: (-1.0819262972727413, 0.6647018304720514)

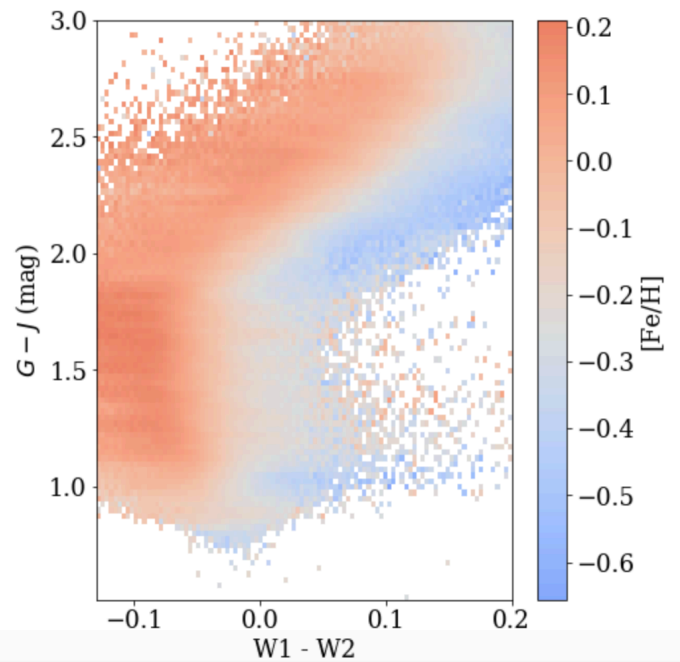


Example 3: Modeling photometric metallicities



KNearestNeighbors

Result: a simple to use “surface”,
no tweaking for shape/order,
extend to additional dimensions easily

1 Million new stars with no spectra



<https://github.com/jradavenport/ingot/>

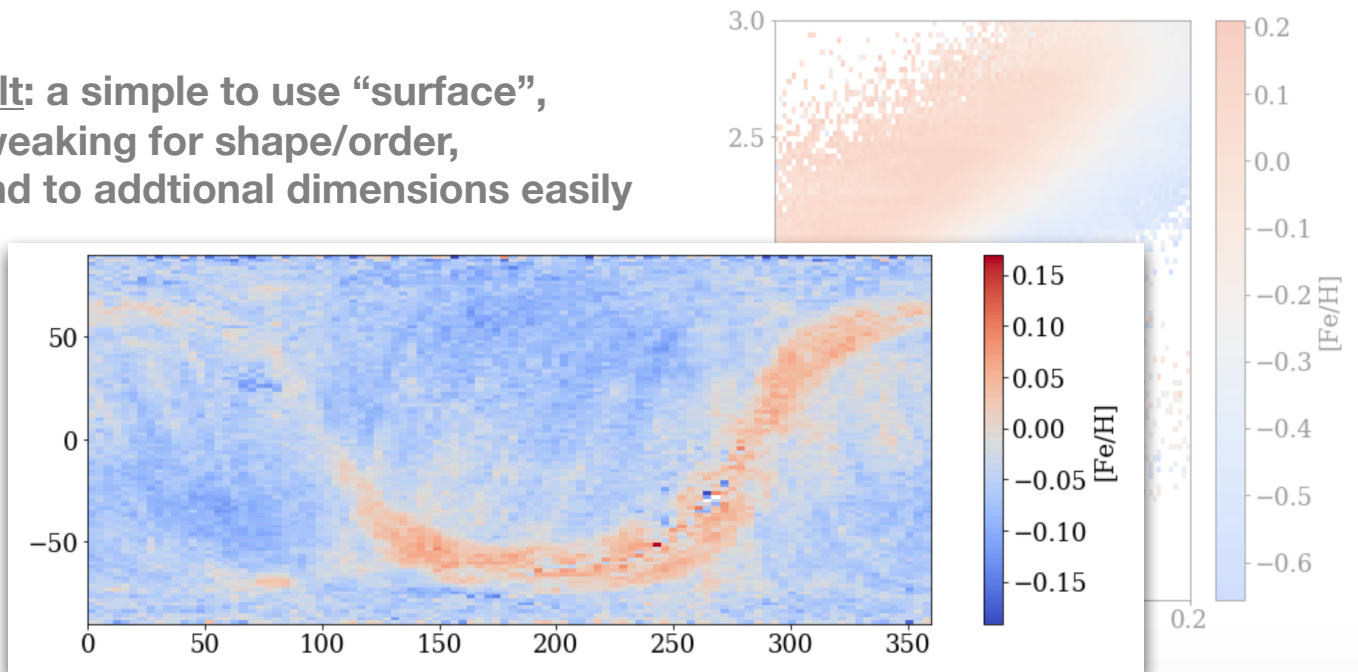
  jradavenport

Example 3: Modeling photometric metallicities



KNearestNeighbors

Result: a simple to use “surface”,
no tweaking for shape/order,
extend to additional dimensions easily

1 Million new stars with no spectra

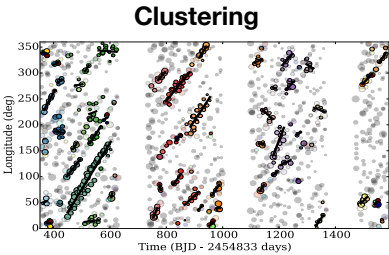
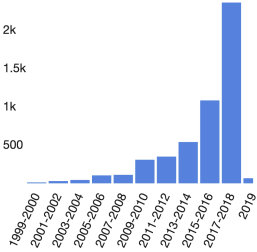


<https://github.com/jradavenport/ingot/>

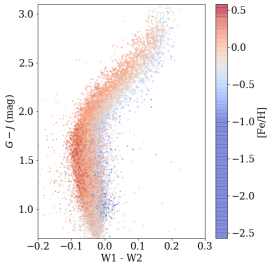
  jradavenport

Conclusions

ML is easier and more “boring” than ever!



Regression



GP's

