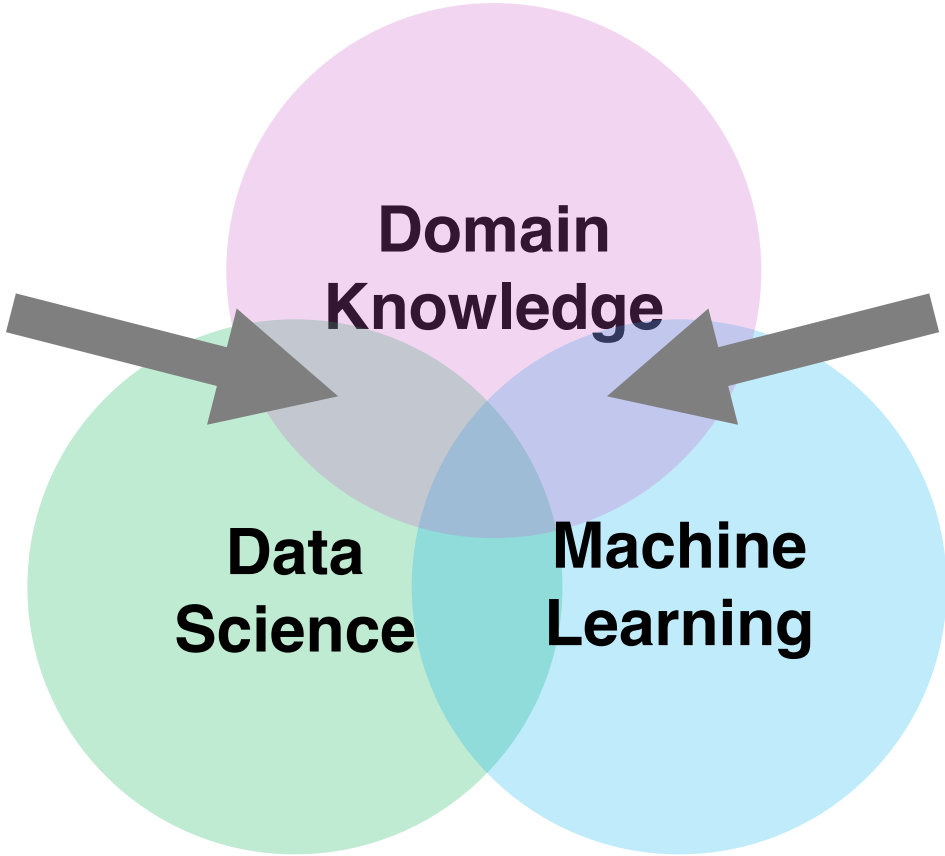
Teaching Domain Experts Data Science: A Progress Report from a Purdue Initiative

Brett M. Savoie
Davidson Assistant Professor of Chemical Engineering, Purdue University

"Teaching Data Science to Students and Teachers II" AIChE National Meeting, Boston MA, 11/10/21

Data Science and Chemical Engineering – Overlaps Matter

Domain Knowledge: The content that most chemical engineering undergraduates already learn.



In the current environment, these overlaps are rapidly growing

Data Science: curating analyzing, storing, and generally making data data useful.

Machine Learning: Using algorithms to develop predictive models from data

Data literate domain experts are essential to realizing these opportunities

Data Science and Chemical Engineering - Contrasts

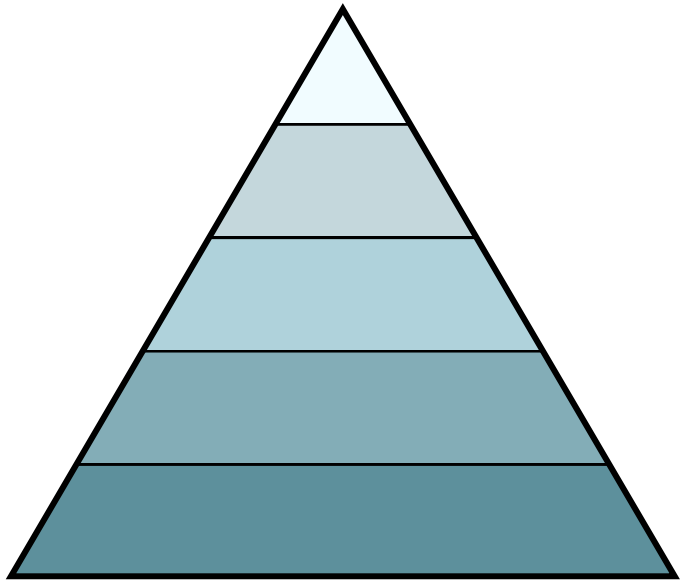
Some contrasts that illustrate the role of domain expertise in applying ML in an engineering context.

Popular ML	ML in Engineering
Data Rich	Data Scarce
Negative results reported	Systematic bias against publishing negative results
Testing and validation are cheap	Testing and validation are expensive
Low stakes for inaccuracies	High stakes for inaccuracies
Weak desire for interpretability	Strong desire for interpretability
Data is typically labeled	Retrospective data is often unlabeled and unstructured
Non-hierarchical. Sloppy data curation and feature selection is forgiven	Intrinsically hierarchical. Data curation and feature selection play a large role.

*These are illustrative not universal.

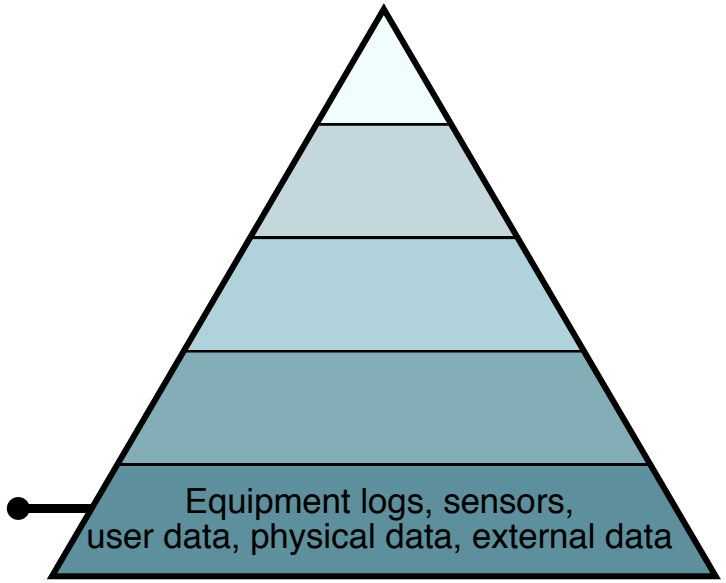
Data Science and Chemical Engineering - Needs

Hierarchy of Data Science Needs:



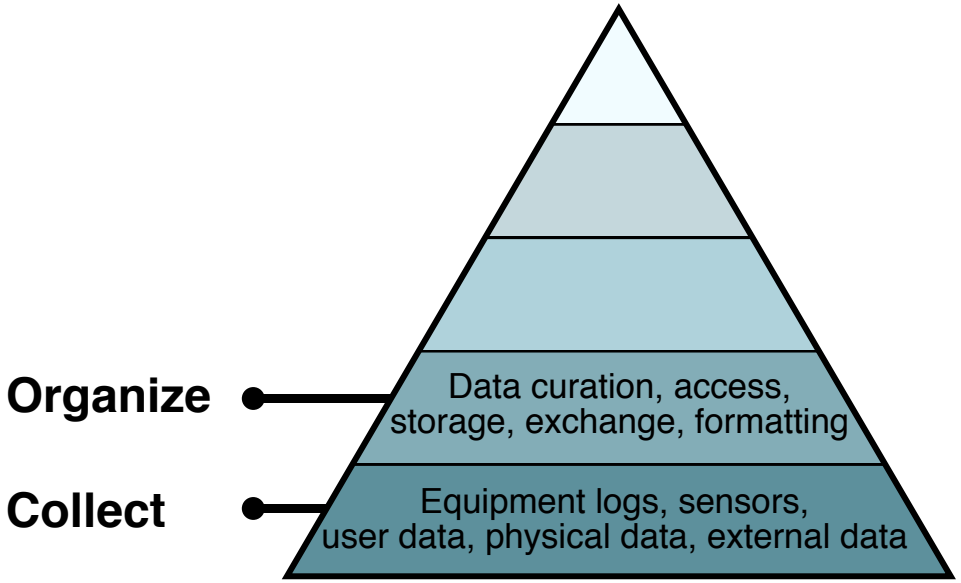
Data Science and Chemical Engineering - Needs

Hierarchy of Data Science Needs:



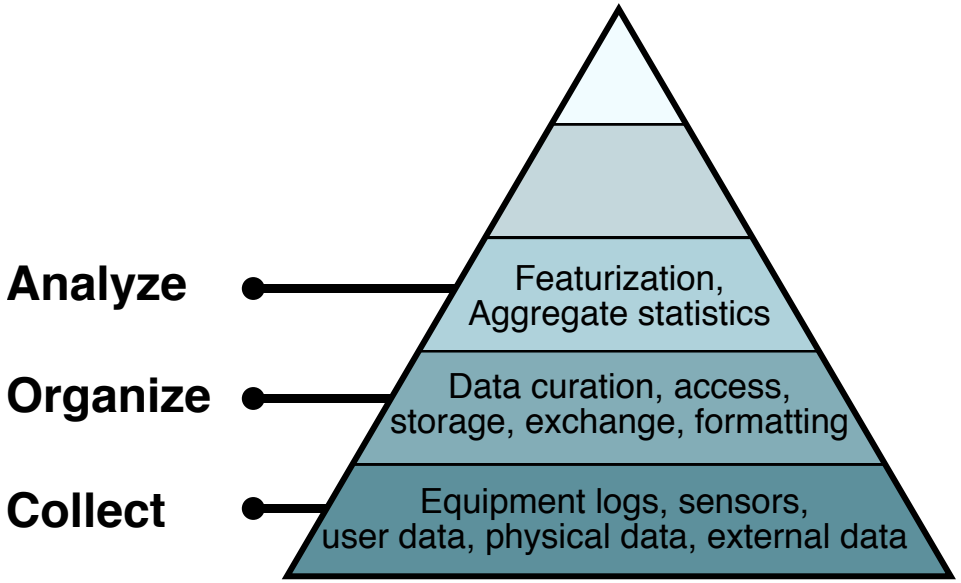
Data Science and Chemical Engineering - Needs

Hierarchy of Data Science Needs:



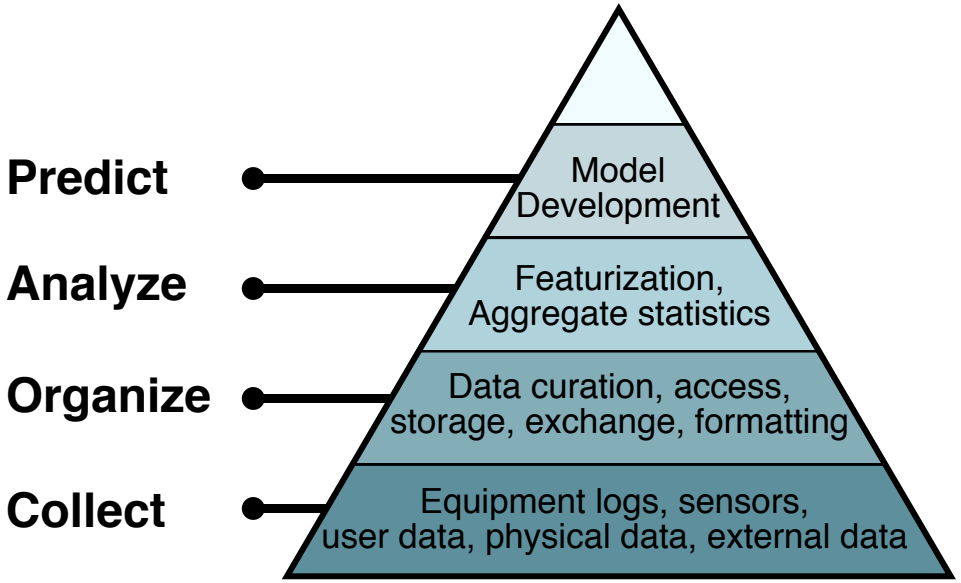
Data Science and Chemical Engineering - Needs

Hierarchy of Data Science Needs:



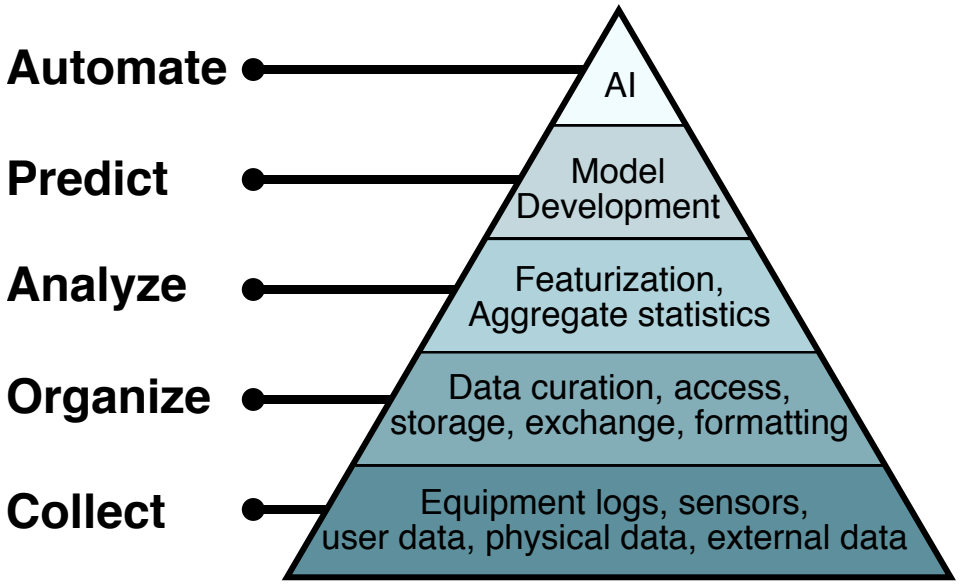
Data Science and Chemical Engineering - Needs

Hierarchy of Data Science Needs:



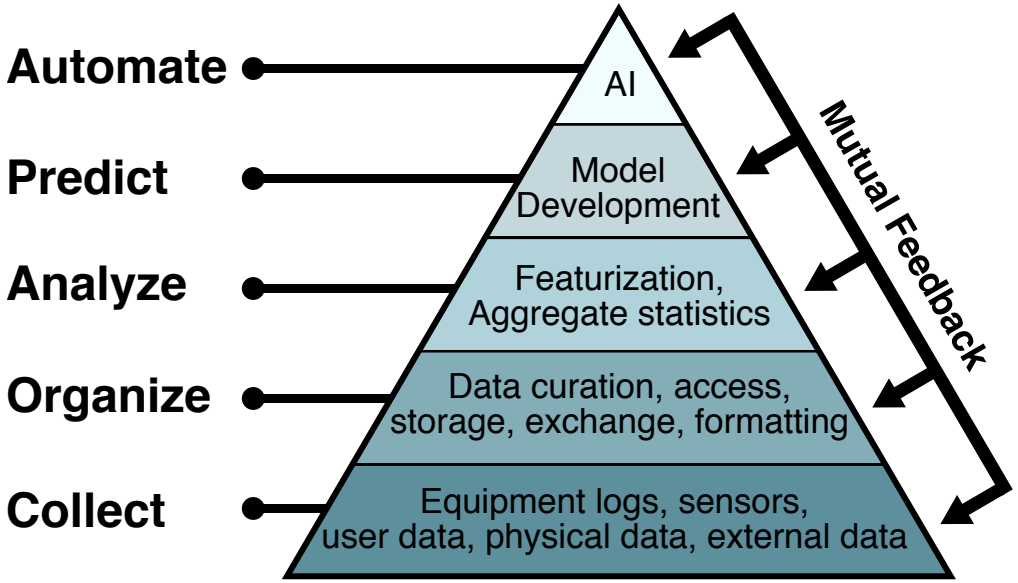
Data Science and Chemical Engineering - Needs

Hierarchy of Data Science Needs:



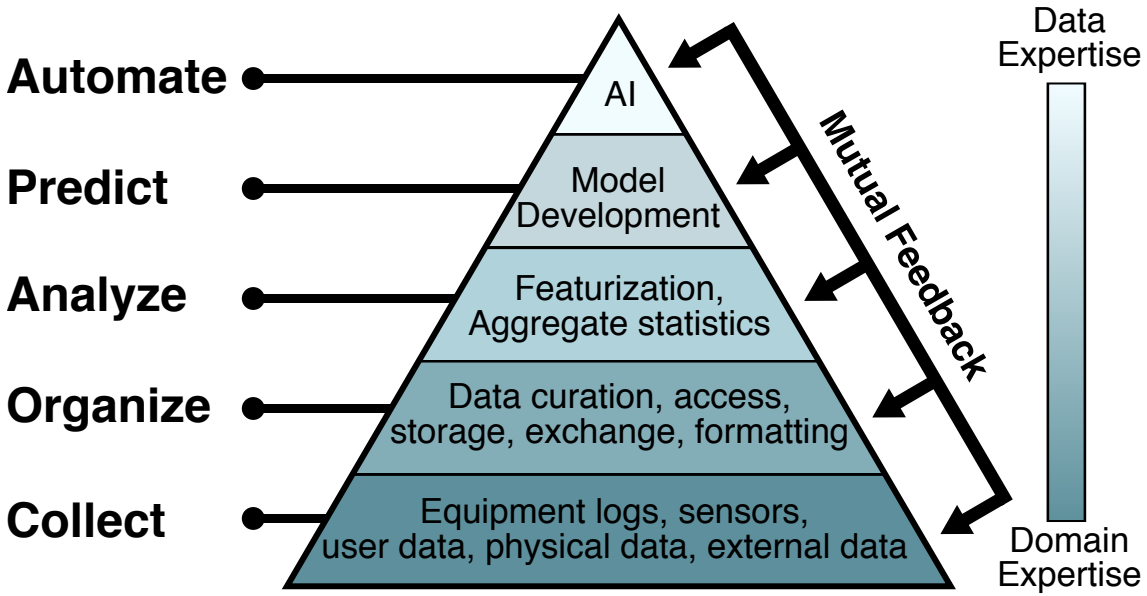
Data Science and Chemical Engineering - Needs

Hierarchy of Data Science Needs:



Data Science and Chemical Engineering - Needs

Hierarchy of Data Science Needs:

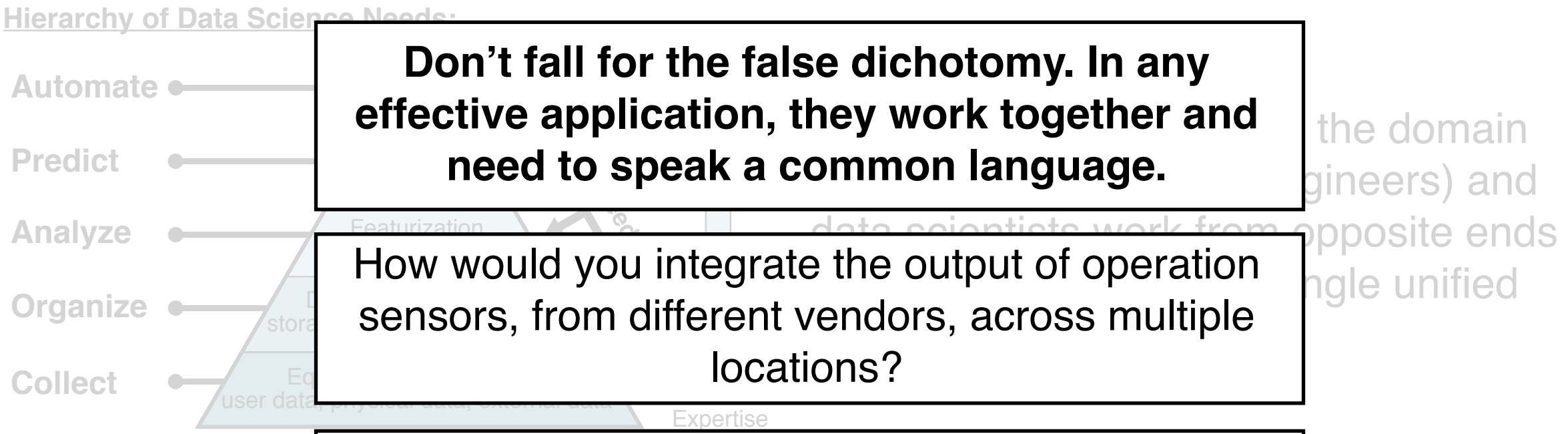


Domain experts (like chemical engineers) and data scientists often work at opposite ends of what needs to be a single unified framework

What does this mean for education?

Is it easier to teach an engineer data science, or a data scientist engineering?

Data Science and Chemical Engineering - Needs



Don’t fall for the false dichotomy. In any effective application, they work together and need to speak a common language.

How would you integrate the output of operation sensors, from different vendors, across multiple locations?

After building a data flow infrastructure, how would you model it in real time to optimize economic, safety, and quality constraints?

How would the model results be validated and implemented into an application?

Domain Expertise is Needed to Evaluate ML Solutions

Challenge: ML is surprisingly versatile when it comes to gaming objective functions.

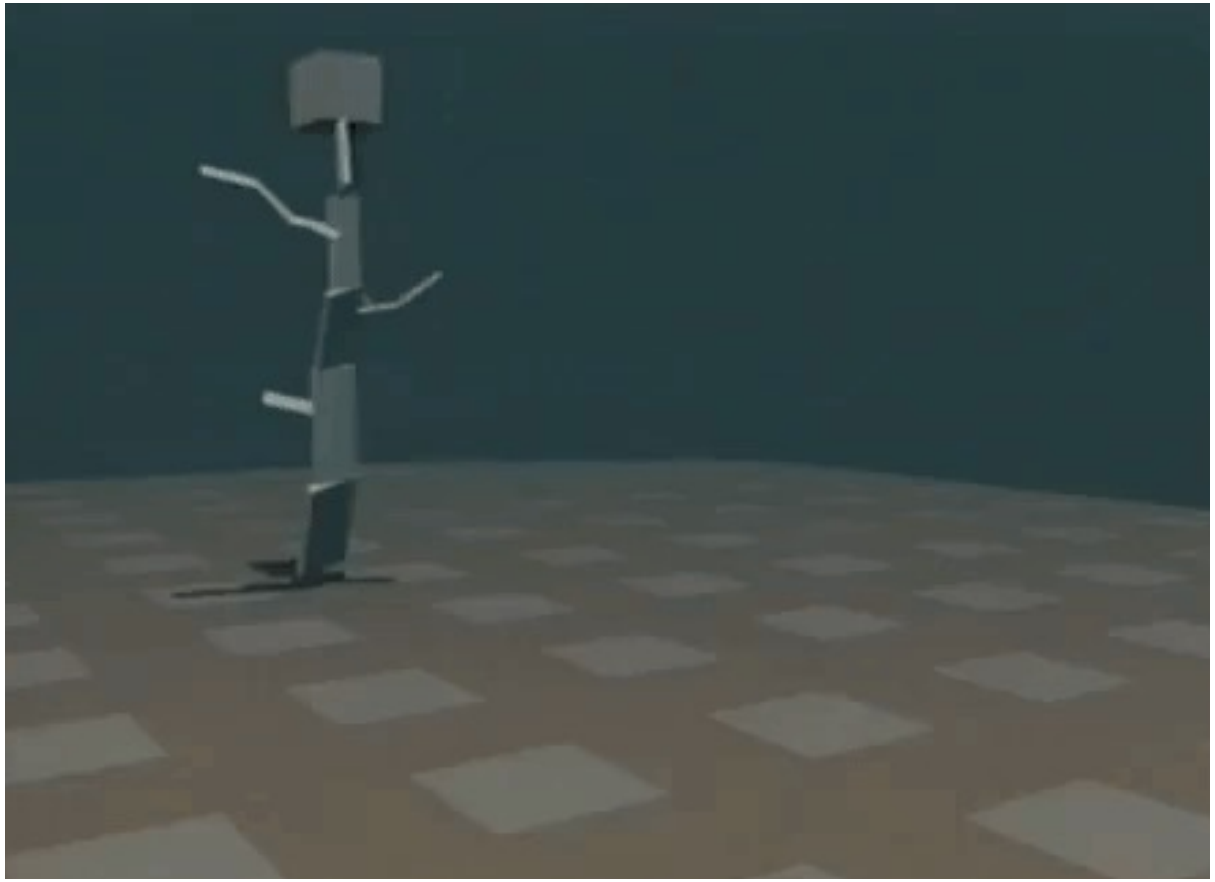
Classic Example: Karl Sims was trying to evolve machines that could walk to a finish line.

Domain Expertise is Needed to Evaluate ML Solutions

Challenge: ML is surprisingly versatile when it comes to gaming objective functions.

Classic Example: Karl Sims was trying to evolve machines that could walk to a finish line.

This case of objective function gaming is obvious because everyone is a “domain expert” when it comes to this task, but objective gaming in technical applications is much more difficult to catch



Sims, K; “Evolving 3D Morphology and Behavior by Competition” Artificial Life 1994

Data Science and Chemical Engineering – Curriculum Changes

CHE 597: Data Science in Chemical Engineering
The intent of this course is to present data analysis and machine learning from a practical perspective focused on applications, use-cases, and the limitations of various approaches to problems in chemical engineering.

CHE 320: Statistical Modeling and Quality Enhancement
Statistical modeling methods, design of experiments, error analysis, curve fitting and regression, analysis of variance, confidence intervals, quality control and enhancement: emphasizes preparation for designing chemical engineering laboratory experiments and analyzing data.

CHE 550: Optimization in Chemical Engineering
Survey of the basic computational tools for solving nonlinear constrained and unconstrained optimization problems. Emphasis on methods applicable to problems arising in chemical plant design, process operations and scheduling, parameter estimation, and waste energy reutilization.

CHE 557: Intelligent Systems in Process Engineering
Introduction to artificial intelligence concepts and techniques and their application to important problems in process systems engineering.

Data Science and Chemical Engineering - Syllabus

CHE 597: Data Science in Chemical Engineering
The intent of this course is to present data analysis and machine learning from a practical perspective focused on applications, use-cases, and the limitations of various approaches to problems in chemical engineering.

Mean Enrollment: ~25 (about evenly split between graduate and undergraduate)

Topics in the Syllabus:

1) Relevant Python (standard library, pandas, numpy, scikit-learn, keras/tensorflow)
2) Data curation (missing values, outlier removal, dimension-reduction)
3) Training/Validation/Testing procedures
4) Optimization (gradient-based, heuristic/global)
5) Over-training concepts (regularization, cross-validation, bias/variance trade-off)
6) Supervised Learning (decision trees, random forests, boosting, deep learning)
7) Unsupervised Learning (k-means, hierarchical agglomeration, PCA)
8) Active Learning (Gaussian Processes, data selection)
9) Transfer Learning (Delta models)

Course Format – Lectures + Jupyter

Lectures + Jupyter Notebooks: The code that is used to generate all of the lecture figures can be found in complementary notebooks

These notebooks consistently rate highly in student reviews

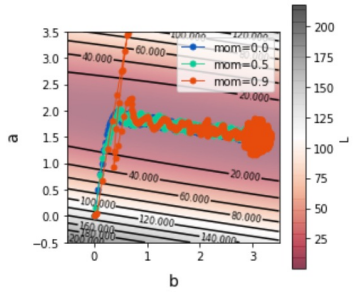
jupyter Optimization Last Checkpoint: 04/02/2020 (autosaved)

File Edit View Insert Cell Kernel Navigate Widgets Help

Run Markdown

Contents

In [30]:
```
# For illustration purposes, we'll use this version to return all of the updates
# (not just every epoch like the above version)
def MBSGD_all(params,grad,N,alpha=0.002,thresh=0.00001,max=1E6,decay=0.0,n=1,mom=0.0):
    m = np.zeros(len(params[0]))
    for i in range(int(max)):
        a = alpha*np.exp(-decay*i)
        for j in range(int(N/n)):
            m = mom*m - a*grad(params[-1] + mom*m,np.random.randint(N,size=n))
            params += [params[-1] + m]
        if np.sum((params[-1]-params[-2])**(2.0))**(0.5) < thresh:
            return np.array(params)
    return np.array(params)

# Perform GD-mom optimization mom=0.9
mom_0 = MBSGD_all([[0,0]],grad_stoch_0,len(x),alpha=0.002,thresh=1E-5,n=1,mom=0,max=1E3)
mom_5 = MBSGD_all([[0,0]],grad_stoch_0,len(x),alpha=0.002,thresh=1E-5,n=1,mom=0.5,max=1E3)
mom_9 = MBSGD_all([[0,0]],grad_stoch_0,len(x),alpha=0.002,thresh=1E-5,n=1,mom=0.9,max=1E3)

# Compare the trajectories on the same optimization surface
surf_plot(np.linspace(a-2,a+2, 101),np.linspace(a-2,a+2, 101),L,[mom_0,mom_5,mom_9],labels=["mom=0.0","mom=0.5","mom=0.
```



Plotting every update you can see that the momentum trajectories actually follow the GD pathways and do not cut the corner as it appeared in the previous example. You can also see that as the momentum factor increases (`mom`) the trajectory tends to overshoot the minimum. This is consistent with the momentum concept of a ball rolling down a gradient, crossing the minimum, and going up the other side before turning around (experiment with different `mom` values to see how this varies). Also note that the final uncertainty in parameters increases with the momentum factor when we don't use the learning rate decay.

As with learning schedules, many algorithms employ momentum concepts, so what you have learned with respect to GD should give you an intuition for how momentum can affect convergence in other cases.

1.5 Adaptive Learning Rates

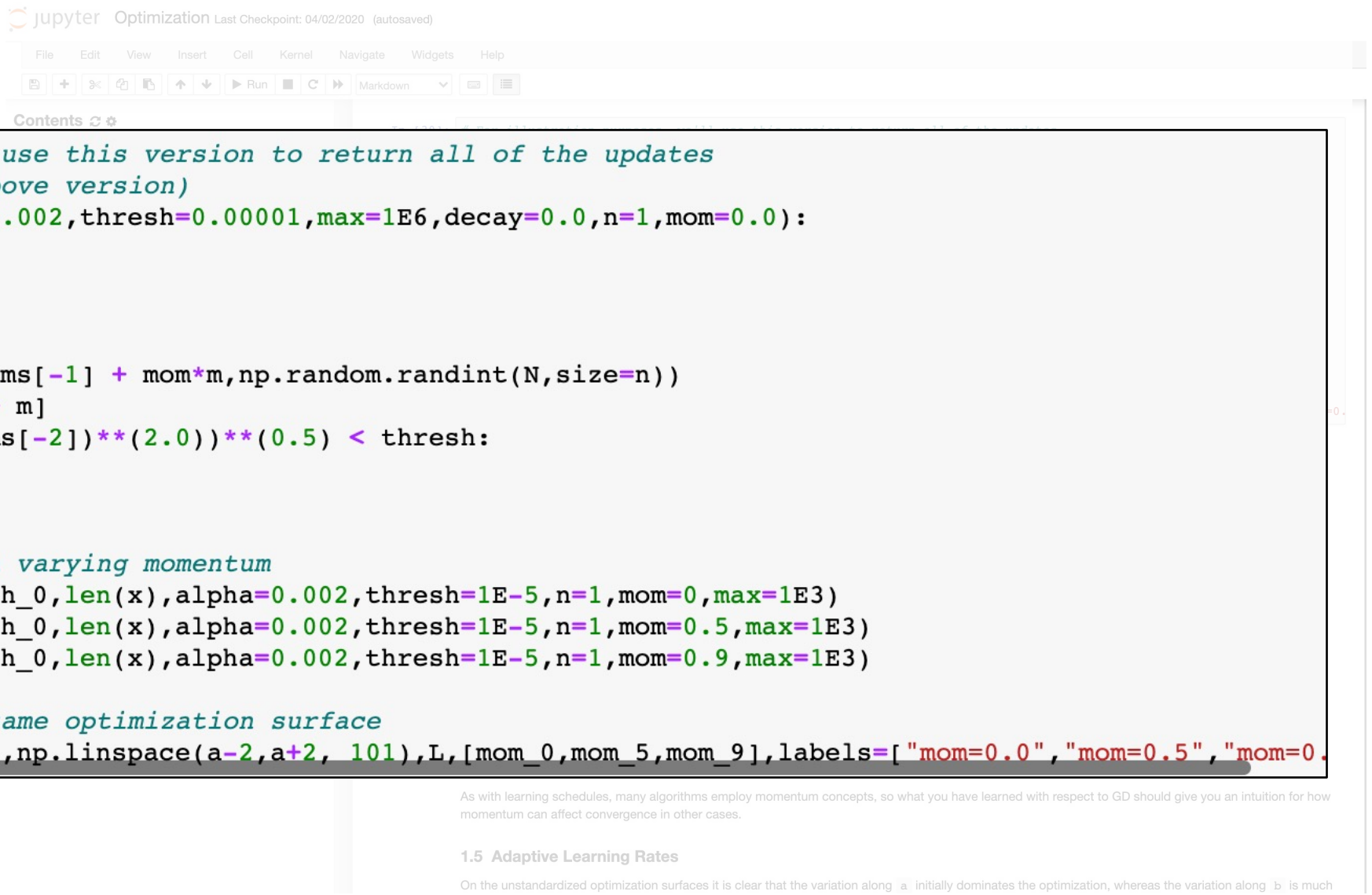
On the unstandardized optimization surfaces it is clear that the variation along `a` initially dominates the optimization, whereas the variation along `b` is much

Course Format – Lectures + Jupyter

Lectures + Jupyter Notebooks: The

highly in student reviews

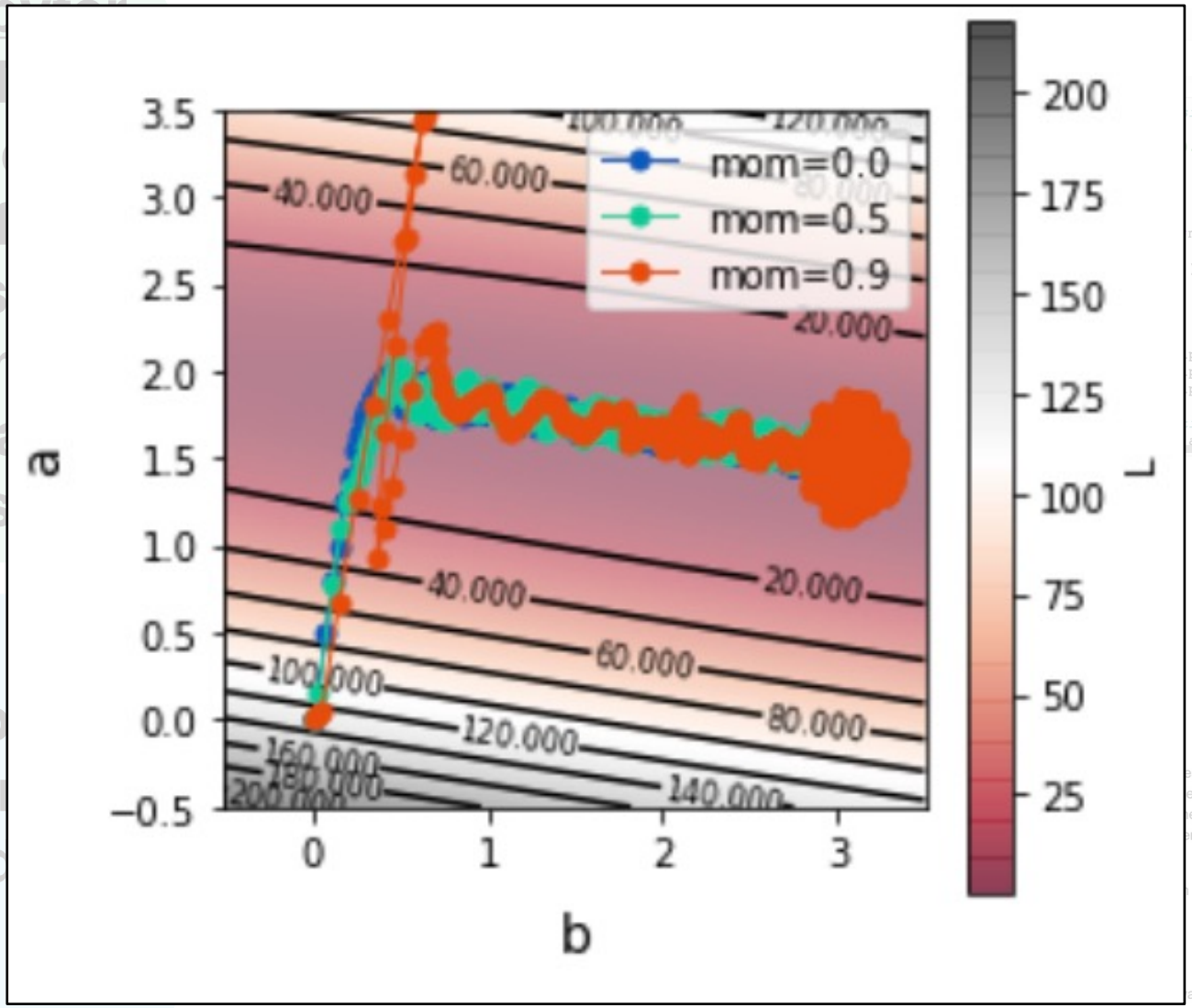
```
# For illustration purposes, we'll use this version to return all of the updates
# (not just every epoch like the above version)
def MBSGD_all(params,grad,N,alpha=0.002,thresh=0.00001,max=1E6,decay=0.0,n=1,mom=0.0):
    m = np.zeros(len(params[0]))
    for i in range(int(max)):
        a = alpha*np.exp(-decay*i)
        for j in range(int(N/n)):
            m = mom*m - a*grad(params[-1] + mom*m,np.random.randint(N,size=n))
            params += [params[-1] + m]
        if np.sum((params[-1]-params[-2])**(2.0))**(0.5) < thresh:
            return np.array(params)
    return np.array(params)

# Perform GD-mom optimizations with varying momentum
mom_0 = MBSGD_all([[0,0]],grad_stoch_0,len(x),alpha=0.002,thresh=1E-5,n=1,mom=0,max=1E3)
mom_5 = MBSGD_all([[0,0]],grad_stoch_0,len(x),alpha=0.002,thresh=1E-5,n=1,mom=0.5,max=1E3)
mom_9 = MBSGD_all([[0,0]],grad_stoch_0,len(x),alpha=0.002,thresh=1E-5,n=1,mom=0.9,max=1E3)

# Compare the trajectories on the same optimization surface
surf_plot(np.linspace(a-2,a+2, 101),np.linspace(a-2,a+2, 101),L,[mom_0,mom_5,mom_9],labels=["mom=0.0","mom=0.5","mom=0.
```



Course Format – Lectures + Jupyter

Lectures + Jup
Notebooks: T
code that is us
generate all of
lecture figures
be found in
complementa
notebooks

These notebo
consistently
highly in stud
reviews



Course Format – Lectures + Jupyter

Lectures + Jupyter

jupyter Optimization Last Checkpoint: 04/02/2020 (autosaved)

File Edit View Insert Cell Kernel Navigate Widgets Help

1.4 Momentum

In the simple linear case maybe it has occured to you that the gradient from the previous epoch might be useful for predicting the direction we should go in the current epoch. More generally, there are many advantages to mixing in the previous gradient evaluation with the current evaluation. In analogy with kinematics, this concept is called momentum--meaning if your optimizer is already heading in a certain direction, it will have a tendency to continue heading in that direction depending on the value of a parameter (η). The momentum correction ($\mathbf{m}$) consists of just one additional term ($\eta\mathbf{m}$):

$$\mathbf{m}_{i+1} = \eta\mathbf{m}_i - \alpha\nabla_\beta f(\boldsymbol{\beta}_i)$$

$$\boldsymbol{\beta}_{i+1} = \boldsymbol{\beta}_i + \mathbf{m}_{i+1}$$

Where η is a number between 0 and 1, and $\mathbf{m}_0 = 0$. When $\eta = 0$ we recover the original GD algorithm. When $\eta = 0.9$ you can think of this like mixing in information from the last ten gradient calculations. From the equations, you can also see that momentum can be stacked on top of either the GD, SGD, and the mini-batch variants, since these just affect how the gradient is calculated.

You might have noticed that the gradient for the momentum correction is evaluated at the "old" position $\boldsymbol{\beta}^i$ in the above equation. An even better variant (Nesterov momentum) performs the gradient calculation at the approximate position $\boldsymbol{\beta}_i + \eta\mathbf{m}_i$ (see lecture notes for longer discussion, but in short this is better):

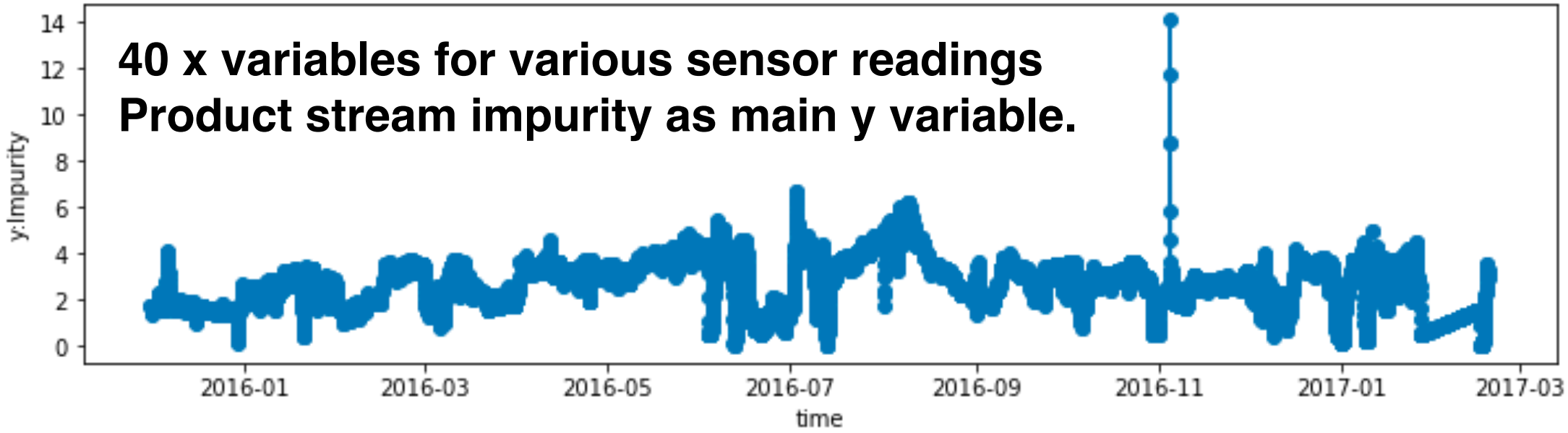
$$\mathbf{m}_{i+1} = \eta\mathbf{m}_i - \alpha\nabla_\beta f(\boldsymbol{\beta}_i + \eta\mathbf{m}_i)$$
$$\boldsymbol{\beta}_{i+1} = \boldsymbol{\beta}_i + \mathbf{m}_{i+1}$$

Adding nesterov momentum to our MBSGD function just requires two lines, (1) to initialize the `m` vector to 0, (2) the `m` update:

On the unstandardized optimization surfaces it is clear that the variation along `a` initially dominates the optimization, whereas the variation along `b` is much

Example Problem – Dow Process Dataset

Dow has opened up a sample dataset from one of its plants (*huge thanks to Leo Chiang@Dow* for curating this).

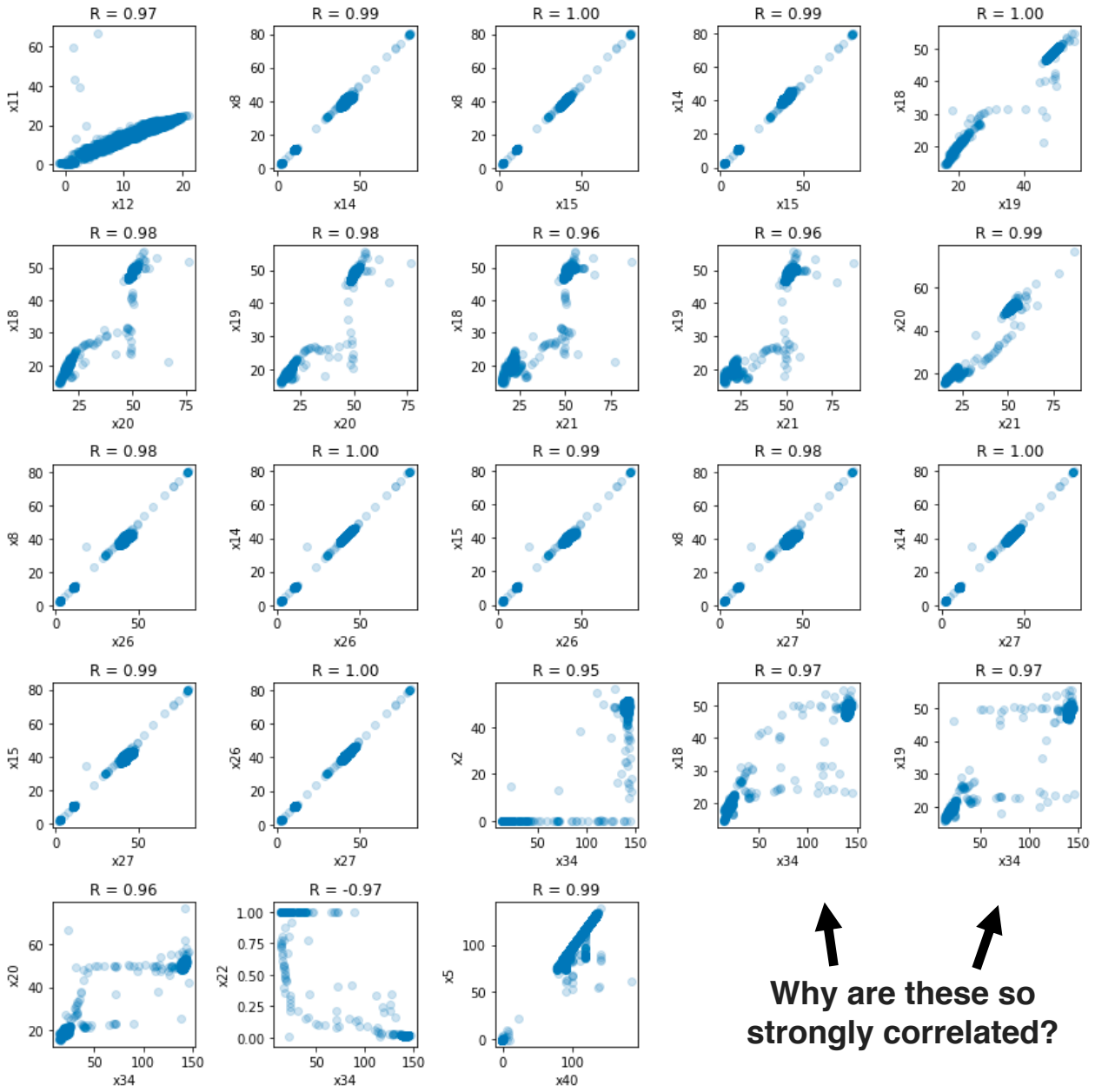


This dataset has lots of missing values and we know that it is polluted with periods of time when some of the unit operations were off for maintenance.

Dow Dataset - Correlations

This problem is introduced right after the python review to discuss data curation, statistics, and outlier removal. We then circle back to it when we cover principle component analysis and again when we discuss supervised learning.

As part of the data investigation we look at the parity plots for all x-variables exhibiting a correlation coefficient >0.95

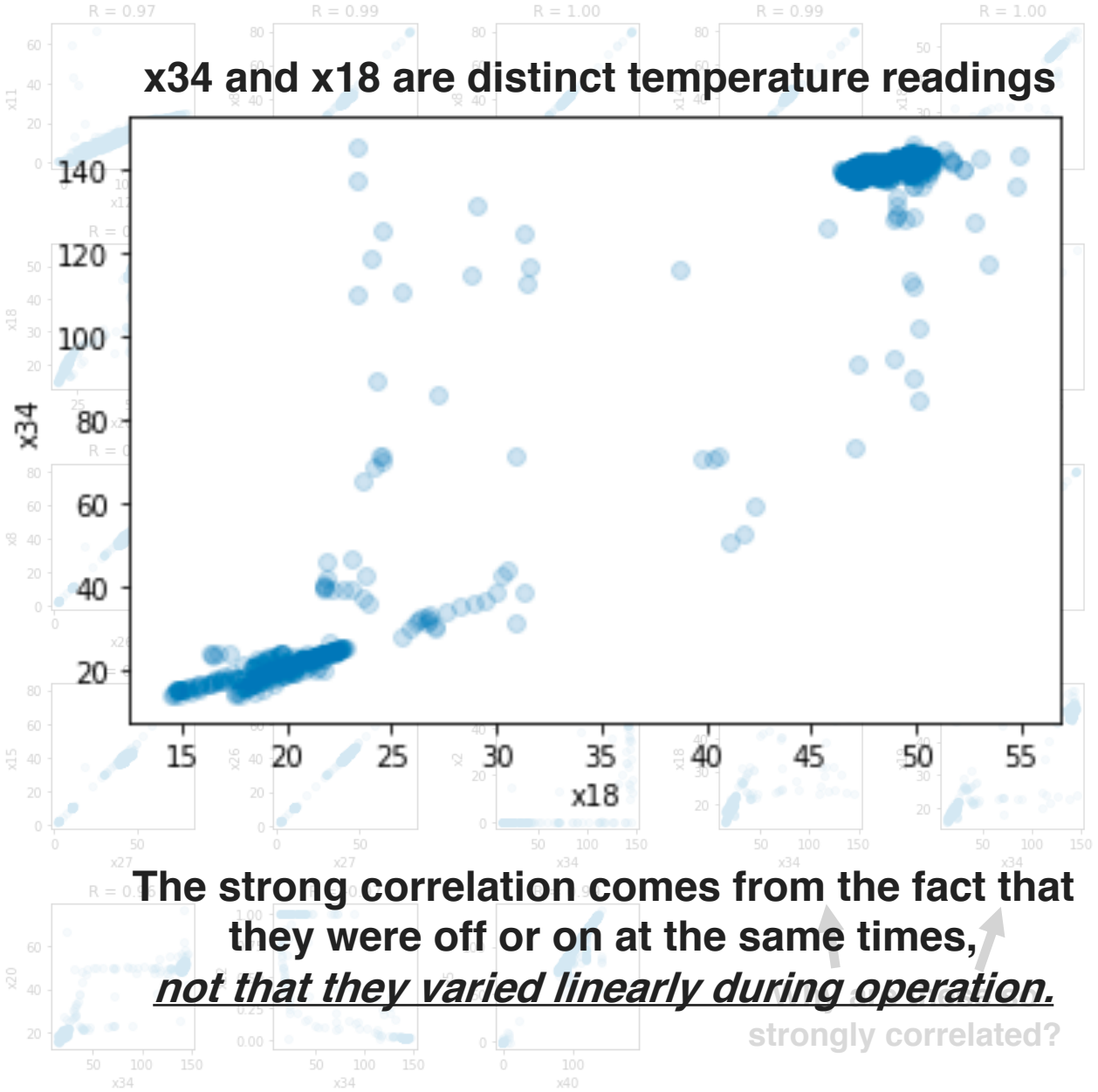


Dow Dataset - Correlations

This problem is introduced right after the python review to discuss data curation, statistics, and outlier removal. We then circle back to it when we cover principle component analysis and again when we discuss supervised learning.

As part of the data investigation we look at the parity plots for all x-variables exhibiting a correlation coefficient >0.95

x34 and x18 are distinct temperature readings



The strong correlation comes from the fact that they were off or on at the same times,

not that they varied linearly during operation.

strongly correlated?

Dow Dataset – Principle Component Analysis

When we cover PCA, we implement it on our own and revisit the Dow problem.

From PCA lecture and notebook

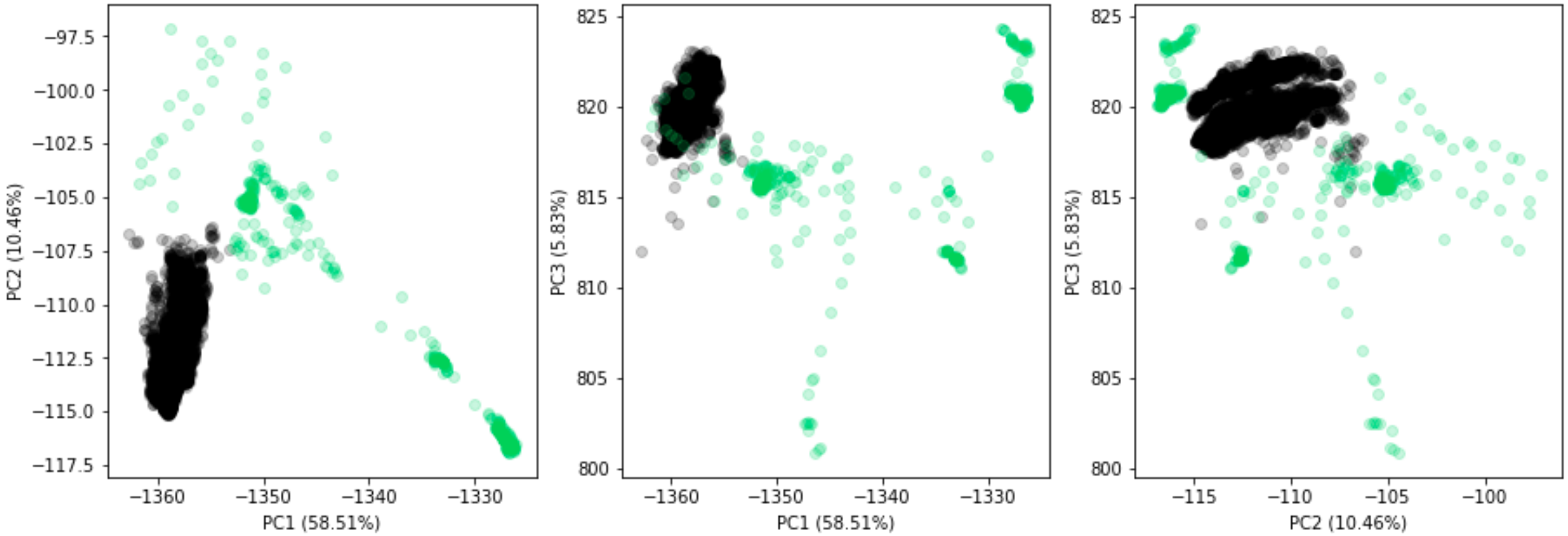
```
# Our own PCA function
def pca(X,std=True):

  # remove mean from each column
  Xmean = X.mean(axis=0)
  TMP = X - Xmean

  # Normalize columns by standard deviation
  if std:
        TMP = TMP/TMP.std(axis=0)

  # Compute the covariance matrix
  C = np.dot(TMP.T, TMP) / ((len(TMP[0])-1)*len(TMP))

  # Perform the Eigen decomposition
  evals, evecs = np.linalg.eig(C)
  ind = np.argsort(evals)[::-1] # sort descending
  evals = evals[ind] # sort evals
  evecs = evecs[:,ind] # sort evecs

  # Project X onto PC space and add back the mean
  X_pca = np.dot(TMP, evecs)
  X_pca = X_pca + np.dot(Xmean,evecs)

  return X_pca,evecs,evals
```

Besides the mean-centering and normalization, it is just an eigendecomposition and projection.

Note: we typically normalize the columns by their variance to control for differences in units.

Dow Dataset – Principle Component Analysis

When we cover PCA, we implement it on our own and revisit the Dow problem.

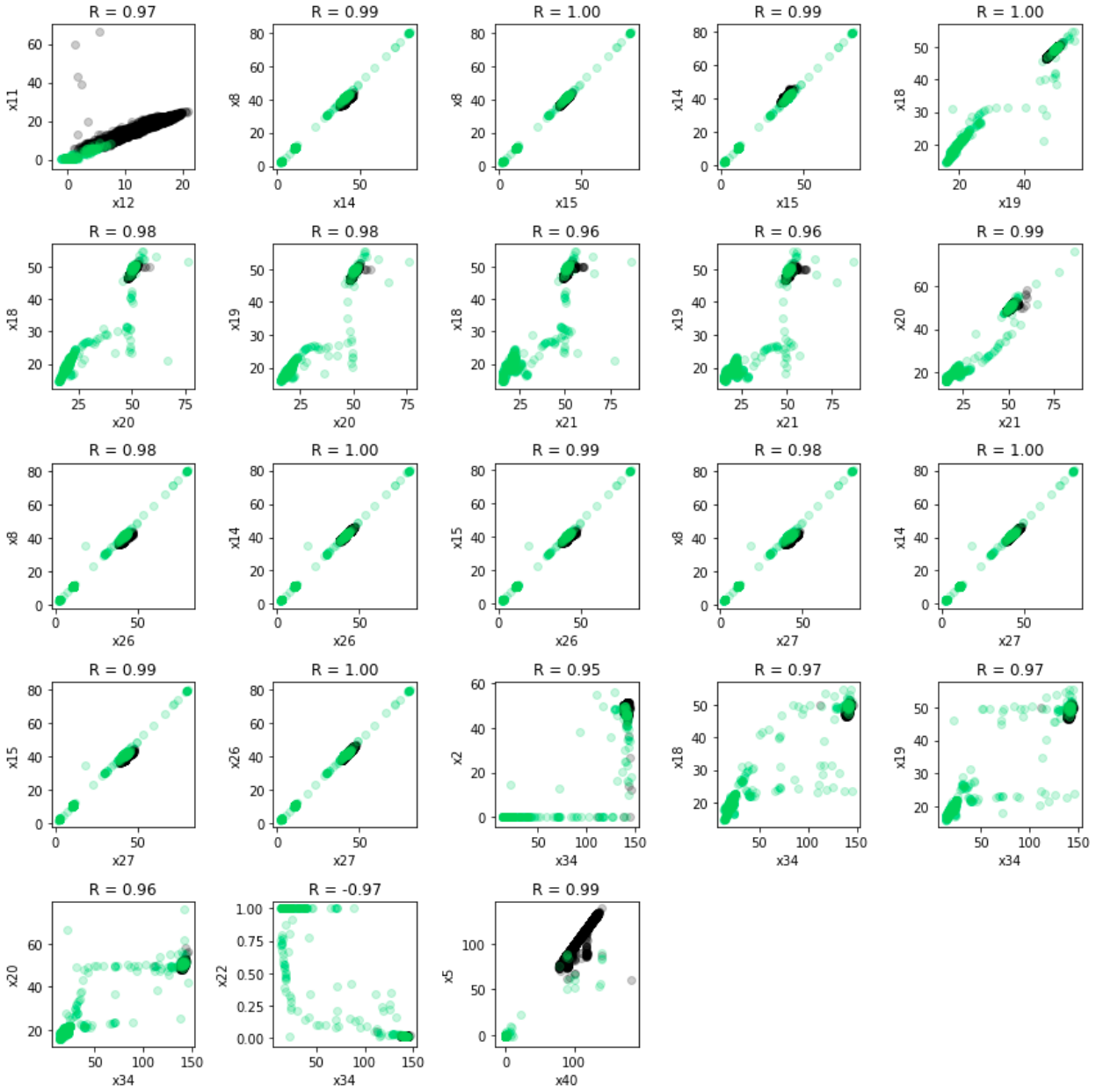


Dataset plotted along the top three PCs; clustering based on PC1/PC2 thresholds to isolate lower-left cluster (96% of datapoints) from the diagonal (4% of datapoints)

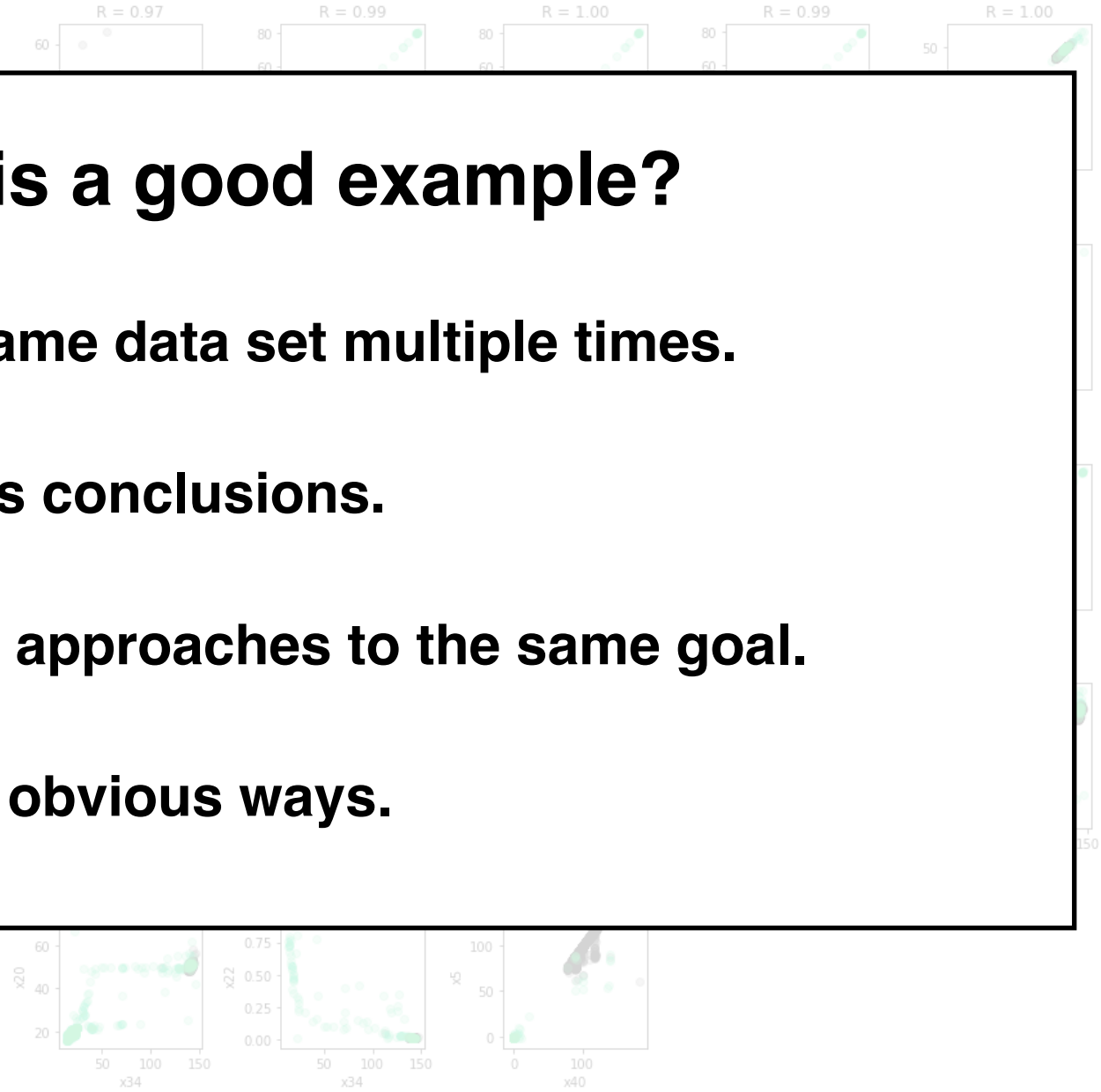
Dow Dataset – Principle Component Analysis

PCA has "automatically" identified the situations when unit ops were off.

What was previously established by inspection, can also be established from PCA (and other clustering techniques)



Dow Dataset – Principle Component Analysis



What makes this a good example?

- It's useful to engage with the same data set multiple times.
- Spurious correlations. Spurious conclusions.
- We are able to present multiple approaches to the same goal.
- Leverages domain expertise in obvious ways.

Points for Discussion

I would like to see more ChE specific datasets. Chemistry is way ahead of us in this regard (and I use them in my course).

I would like to dispense with the Python introduction, but at least 1/2 of my students would be completely lost without it.

Deep learning is only covered in a token fashion. A dedicated course is really needed.

We're still balancing "implement on your own" against using standard libraries. We take a bimodal approach.

How do you teach a subject that is evolving as quickly as ML?

Acknowledgements

Researchers:

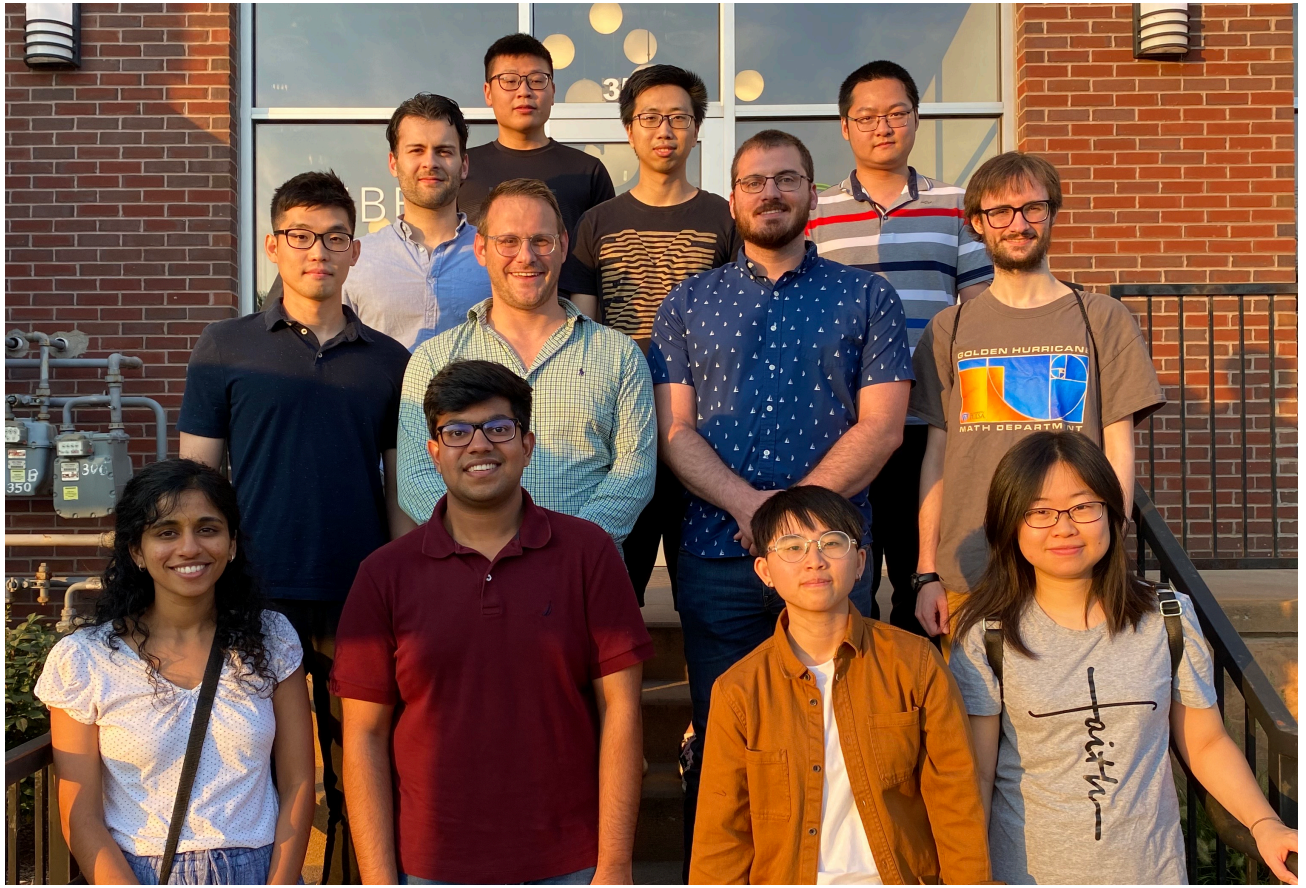
Ryan Brown, Robert MacKnight

Collaborators:

Letian Dou (Perovskites; DOE-SETO)
Bryan Boudouris (Radical Polymers; AFOSR&DOE)
Jianguo Mei (Conjugated Polymers; MURI)
Julia Laskin (Mass Spec Degradation; MURI)
Geoffrey Hutchison (U. Pitt; MURI)
Olexandr Isayev (Carnegie Mellon; MURI)
Jie Xu (ANL; Automated Characterization; MURI)

Funding:

The Camille & Henry Dreyfus Foundation, Inc.



Science to Shape the Future of the Army · Army Research Office · 1951

Air Force Office of Scientific Research · United States Air Force

NSF

P2SAC
Purdue Process Safety & Assurance Center

U.S. Department of Energy

ORAU

Office of Naval Research · ONR · Science & Technology

ACS PRF