Lifting the Curtain on Machine Learning

Norm Matloff
University of California at Davis

SatRday
UCLA, April 6, 2019

These slides will be available at
`http://heather.cs.ucdavis.edu/satrday.pdf`

Outline of Talk

Outline of Talk

- Introduce 2 R packages.

Outline of Talk

- Introduce 2 R packages.
  - **polyreg**
  - **prVis**

Outline of Talk

- Introduce 2 R packages.
  - **polyreg**
  - **prVis**
- Both are "machine learning (ML) alternatives."

Outline of Talk

- Introduce 2 R packages.
  - **polyreg**
  - **prVis**
- Both are "machine learning (ML) alternatives."
- Taking a critical look at certain aspects of ML:

Outline of Talk

- Introduce 2 R packages.
  - **polyreg**
  - **prVis**
- Both are "machine learning (ML) alternatives."
- Taking a critical look at certain aspects of ML:
  - neural networks (NNs)
  - t-sne (a "nonlinear PCA")

Sources of Confusion

Sources of Confusion

- The press tends to present the message AI

Sources of Confusion

- The press tends to present the message AI = machine learning

Sources of Confusion

- The press tends to present the message
  AI = machine learning = neural networks

Sources of Confusion

- The press tends to present the message
  AI = machine learning = neural networks
- Not true, of course, but the NN people have a knack for getting into the press. :-)

Sources of Confusion

- The press tends to present the message
  AI = machine learning = neural networks
- Not true, of course, but the NN people have a knack for getting into the press. :-)
- The very term *machine learning* already sounds science fiction-ish,

Sources of Confusion

- The press tends to present the message
  AI = machine learning = neural networks
- Not true, of course, but the NN people have a knack for getting into the press. :-)
- The very term *machine learning* already sounds science fiction-ish, and *neural networks* really does.

Sources of Confusion (cont'd)

Sources of Confusion (cont'd)

The NN/ML people tend to invent their own terminology. E.g

statistics-ese	ML-ese
observations	cases
predictors	features
covariates	side information
β_0/intercept	bias
prediction	inference
inference	statistics

Goals

Goals

So, our goals are:

Goals

So, our goals are:

- Show what NNs are actually doing.

Goals

So, our goals are:

- Show what NNs are actually doing.
- Suggest a more straightforward alternative to NNs,

Goals

So, our goals are:

- Show what NNs are actually doing.
- Suggest a more straightforward alternative to NNs, that performs as well or better than NNs yet it is simpler and easier to use.

Goals

So, our goals are:

- Show what NNs are actually doing.
- Suggest a more straightforward alternative to NNs, that performs as well or better than NNs yet it is simpler and easier to use.
- Present a "spinoff" visualization package that serves as an alternative to a popular ML one.

Neural Networks

Neural Networks

- Series of *layers*:

Neural Networks

- Series of *layers*:
  - input (predictors);

Neural Networks

- Series of *layers*:
  - input (predictors);
  - output (prediction);

Neural Networks

- Series of *layers*:
  - input (predictors);
  - output (prediction);
  - ≥ 1 *hidden* layers in between.

Neural Networks

- Series of *layers*:
  - input (predictors);
  - output (prediction);
  - ≥ 1 *hidden* layers in between.
  - Each hidden layer consists of a few/many units (*neurons*).

Neural Networks

- Series of *layers*:
  - input (predictors);
  - output (prediction);
  - ≥ 1 *hidden* layers in between.
  - Each hidden layer consists of a few/many units (*neurons*).
- Inputs to layer i = linear combination of outputs from layer $i-1$.

Neural Networks

- Series of *layers*:
  - input (predictors);
  - output (prediction);
  - ≥ 1 *hidden* layers in between.
  - Each hidden layer consists of a few/many units (*neurons*).
- Inputs to layer i = linear combination of outputs from layer $i-1$.
- Outputs of each layer run through an *activation function*, e.g. logistic, to allow for nonlinearity.

Example: UCI Vertebrae Data

Example: UCI Vertebrae Data

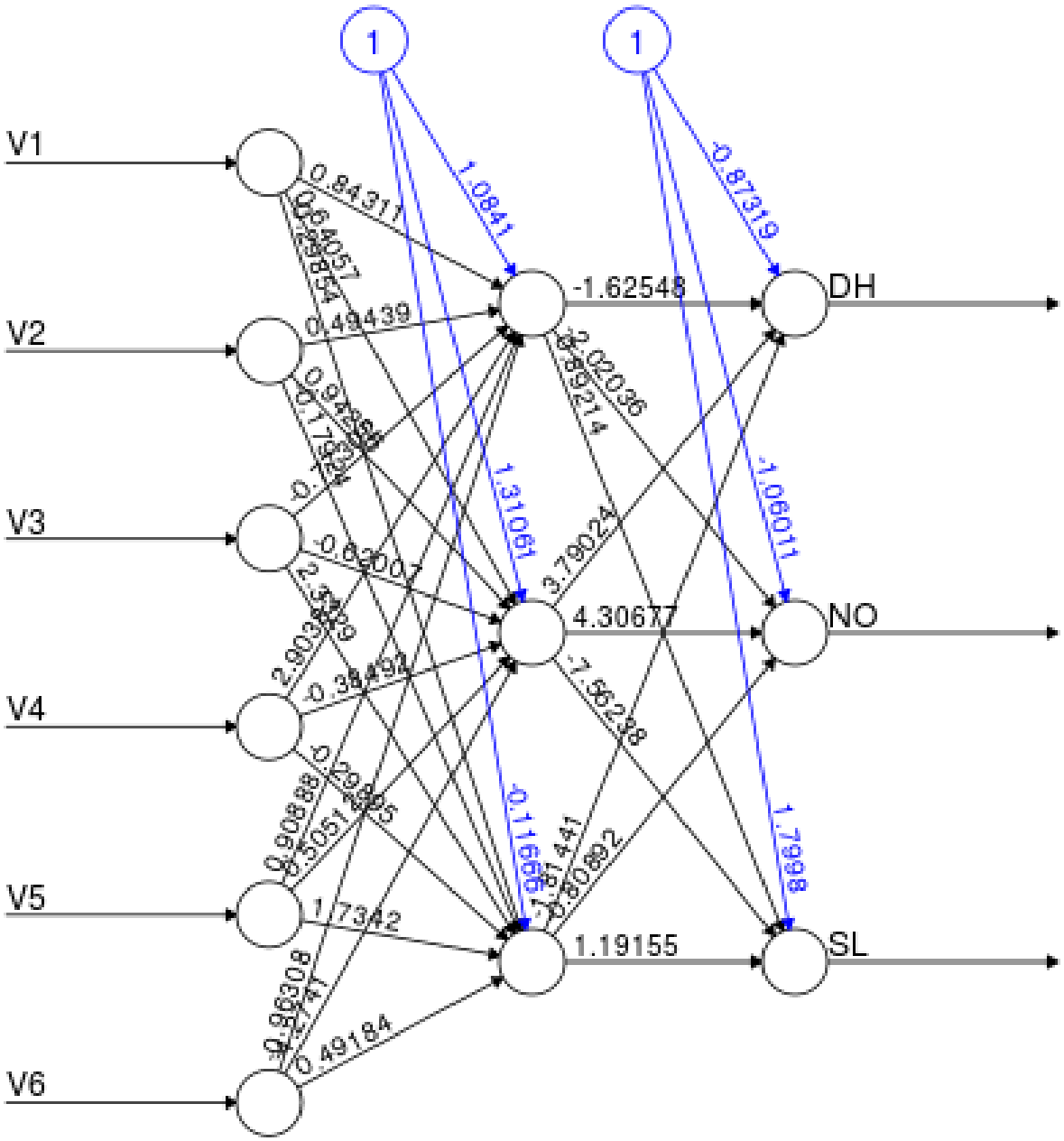
- 6 predictors (various med.), V1, V2,...,V6.
- Predict one of 3 classes, DH, NO, SL. (E.g. NO = normal.)

Example: UCI Vertebrae Data

- 6 predictors (various med.), V1, V2,...,V6.
- Predict one of 3 classes, DH, NO, SL. (E.g. NO = normal.)
- Many R packages, e.g. **kerasformula**, **MXNet**.

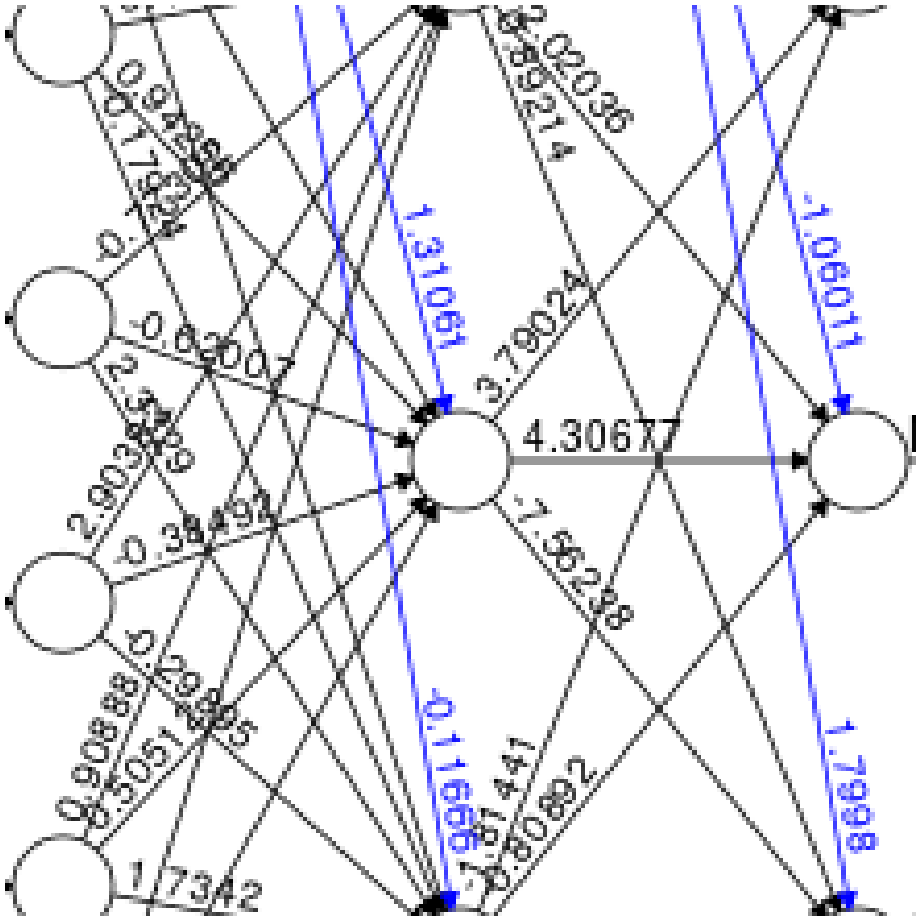
Norm Matloff University of California at Davis

UCI Vert. (cont'd.)



Closeup: 2nd Neuron in 2nd Layer

Closeup: 2nd Neuron in 2nd Layer



Input to this neuron: $\ldots + 0.94V2 - 0.62V3 - 0.38V4 + \ldots$

This neuron then feeds that lin. comb. into logistic, which is then input to all neurons in next layer, with weights 3.79, 4.31 and 7.56.

History of NNs

History of NNs

- Treated largely as a curiosity through the 1990s.
- Then in the 2000s, "NN+" models, e.g. CNN, won a number of major competitions, a huge boost to their popularity.
- But also many dismiss them as hype.
- Some say NNs work poorly on their data; others counter, "You're not using them right."

Contributions of Our Work

Contributions of Our Work

https://arxiv.org/abs/1806.06850

- We present an informal argument that NNs — in essence — actually are polynomial regression (PR). Acronym: NN=PR

Contributions of Our Work

https://arxiv.org/abs/1806.06850

- We present an informal argument that NNs — in essence — actually are polynomial regression (PR). Acronym: NN=PR
- We use this to speculate and then confirm a surprising multicollinearity property of NNs.

Contributions of Our Work

https://arxiv.org/abs/1806.06850

- We present an informal argument that NNs — in essence — actually are polynomial regression (PR). Acronym: NN=PR
- We use this to speculate and then confirm a surprising multicollinearity property of NNs.
- NN=PR suggests that one might simply fit a polynomial model in the first place, bypassing NNs.

Contributions of Our Work

https://arxiv.org/abs/1806.06850

- We present an informal argument that NNs — in essence — actually are polynomial regression (PR). Acronym: NN=PR
- We use this to speculate and then confirm a surprising multicollinearity property of NNs.
- NN=PR suggests that one might simply fit a polynomial model in the first place, bypassing NNs.
- Thus avoid NN's problems, e.g. choosing numerous hyperparameters, nonconvergence and so on.

Contributions of Our Work

https://arxiv.org/abs/1806.06850

- We present an informal argument that NNs — in essence — actually are polynomial regression (PR). Acronym: NN=PR
- We use this to speculate and then confirm a surprising multicollinearity property of NNs.
- NN=PR suggests that one might simply fit a polynomial model in the first place, bypassing NNs.
- Thus avoid NN's problems, e.g. choosing numerous hyperparameters, nonconvergence and so on.
- Tried many datasets. In all cases, **PR meets or beats NNs in predictive accuracy**.

Contributions of Our Work

https://arxiv.org/abs/1806.06850

- We present an informal argument that NNs — in essence — actually are polynomial regression (PR). Acronym: NN=PR
- We use this to speculate and then confirm a surprising multicollinearity property of NNs.
- NN=PR suggests that one might simply fit a polynomial model in the first place, bypassing NNs.
- Thus avoid NN's problems, e.g. choosing numerous hyperparameters, nonconvergence and so on.
- Tried many datasets. In all cases, **PR meets or beats NNs in predictive accuracy**.
- Developed many-featured R pkg., **polyreg**.

Notation and Acronyms

Notation and Acronyms

- n cases; p predictors
- polynomials of degree d
- *PR*: polynomial regression
- *NN=PR*: Neural Networks Are Essentially Polynomial Regression

polyreg

polyreg

- R package.

polyreg

- R package.
- Motivated by NN=PR: use PR instead of NNs.

polyreg

- R package.
- Motivated by NN=PR: use PR instead of NNs.
- Generates all possible d-degree polynomials in p variables.

polyreg

- R package.
- Motivated by NN=PR: use PR instead of NNs.
- Generates all possible d-degree polynomials in p variables. (Not so easy. Must skip, e.g., powers of dummy variables.)

polyreg

- R package.
- Motivated by NN=PR: use PR instead of NNs.
- Generates all possible d-degree polynomials in p variables. (Not so easy. Must skip, e.g., powers of dummy variables.)
- Has dimension reduction options.

polyreg

- R package.
- Motivated by NN=PR: use PR instead of NNs.
- Generates all possible d-degree polynomials in p variables. (Not so easy. Must skip, e.g., powers of dummy variables.)
- Has dimension reduction options.
- *github.com/matloff/polyreg*

Key polyreg functions

Key polyreg functions

```
polyFit(function (xy, deg, maxInteractDeg = deg,
    use = "lm", pcaMethod = NULL, pcaLocation =
    "front", pcaPortion = 0.9, glmMethod = "one",
    return_xy = FALSE, returnPoly = FALSE)

predict.polyFit(object, newdata, ...)
```

E.g. if choose dimension reduction by PCA in **polyFit()**, **predict()** will automatically take care of it.

Various other dim. reduction, helper functions.

NN=PR

NN=PR

- Consider toy example:

NN=PR

- Consider toy example:
- Activation function $a(t) = t^2$.

NN=PR

- Consider toy example:
- Activation function $a(t) = t^2$.
- Say $p = 2$ predictors, u and v.

NN=PR

- Consider toy example:
- Activation function $a(t) = t^2$.
- Say $p = 2$ predictors, u and v.
- Output of Layer 1 is all quadratic functions of u, v.

NN=PR

- Consider toy example:
- Activation function $a(t) = t^2$.
- Say $p = 2$ predictors, u and v.
- Output of Layer 1 is all quadratic functions of u, v.
- Output of Layer 2 is all quartic ($d = 4$) functions of u, v.

NN=PR

- Consider toy example:
- Activation function $a(t) = t^2$.
- Say $p = 2$ predictors, u and v.
- Output of Layer 1 is all quadratic functions of u, v.
- Output of Layer 2 is all quartic ($d = 4$) functions of u, v.
- Etc.

NN=PR

- Consider toy example:
- Activation function $a(t) = t^2$.
- Say $p = 2$ predictors, u and v.
- Output of Layer 1 is all quadratic functions of u, v.
- Output of Layer 2 is all quartic ($d = 4$) functions of u, v.
- Etc.
- Polynomial regression!

NN=PR

- Consider toy example:
- Activation function $a(t) = t^2$.
- Say $p = 2$ predictors, u and v.
- Output of Layer 1 is all quadratic functions of u, v.
- Output of Layer 2 is all quartic ($d = 4$) functions of u, v.
- Etc.
- Polynomial regression!
- **Important note:** The degree of the fitted polynomial in NN grows with each layer.

NN=PR: General Activation Functions

NN=PR: General Activation Functions

- Clearly this analysis for $a(t) = t^2$ extends to any polynomial activation function.

NN=PR: General Activation Functions

- Clearly this analysis for $a(t) = t^2$ extends to any polynomial activation function.
- What about transcendental $a()$?

NN=PR: General Activation Functions

- Clearly this analysis for $a(t) = t^2$ extends to any polynomial activation function.
- What about transcendental $a()$? Computer implementatations often use a Taylor series rep., i.e. a polynomial!

NN=PR: General Activation Functions

- Clearly this analysis for $a(t) = t^2$ extends to any polynomial activation function.
- What about transcendental $a()$? Computer implementatations often use a Taylor series rep., i.e. a polynomial!
- What about reLU? Same analysis, but now have piecewise polynomials, so NN=PPR.

NN=PR: General Activation Functions

- Clearly this analysis for $a(t) = t^2$ extends to any polynomial activation function.
- What about transcendental $a()$? Computer implementatations often use a Taylor series rep., i.e. a polynomial!
- What about reLU? Same analysis, but now have piecewise polynomials, so NN=PPR.
- Even without Taylor series etc.] any reasonable activation function is "close" to a polynomial.

NN=PR: General Activation Functions

- Clearly this analysis for $a(t) = t^2$ extends to any polynomial activation function.
- What about transcendental $a()$? Computer implementatations often use a Taylor series rep., i.e. a polynomial!
- What about reLU? Same analysis, but now have piecewise polynomials, so NN=PPR.
- Even without Taylor series etc.] any reasonable activation function is "close" to a polynomial.
- Hence NN=PR.

Implications of NN=PR

Implications of NN=PR

- Use our understanding of PR to gain insights into NNs.
- Heed the "advice" of NN=PR, and use PR instead of NNs!

Implications of NN=PR

- Use our understanding of PR to gain insights into NNs.
- Heed the "advice" of NN=PR, and use PR instead of NNs!
  - No dealing with numerous hyperparameters.

Implications of NN=PR

- Use our understanding of PR to gain insights into NNs.
- Heed the "advice" of NN=PR, and use PR instead of NNs!
  - No dealing with numerous hyperparameters.
  - No convergence issues.

Implications of NN=PR

- Use our understanding of PR to gain insights into NNs.
- Heed the "advice" of NN=PR, and use PR instead of NNs!
  - No dealing with numerous hyperparameters.
  - No convergence issues.
  - No "fake minima" (NN iteration settles on a local min).

Implications of NN=PR

- Use our understanding of PR to gain insights into NNs.
- Heed the "advice" of NN=PR, and use PR instead of NNs!
  - No dealing with numerous hyperparameters.
  - No convergence issues.
  - No "fake minima" (NN iteration settles on a local min).

Possible drawbacks/remedies of PR:

Implications of NN=PR

- Use our understanding of PR to gain insights into NNs.
- Heed the "advice" of NN=PR, and use PR instead of NNs!
  - No dealing with numerous hyperparameters.
  - No convergence issues.
  - No "fake minima" (NN iteration settles on a local min).

Possible drawbacks/remedies of PR:

- Large memory requirement.

Implications of NN=PR

- Use our understanding of PR to gain insights into NNs.
- Heed the "advice" of NN=PR, and use PR instead of NNs!
  - No dealing with numerous hyperparameters.
  - No convergence issues.
  - No "fake minima" (NN iteration settles on a local min).

Possible drawbacks/remedies of PR:

- Large memory requirement. Maybe use R's **bigmemory** package (with backing store).

Implications of NN=PR

- Use our understanding of PR to gain insights into NNs.
- Heed the "advice" of NN=PR, and use PR instead of NNs!
  - No dealing with numerous hyperparameters.
  - No convergence issues.
  - No "fake minima" (NN iteration settles on a local min).

Possible drawbacks/remedies of PR:

- Large memory requirement. Maybe use R's **bigmemory** package (with backing store).
- Run time (worse than NN????).

Implications of NN=PR

- Use our understanding of PR to gain insights into NNs.
- Heed the "advice" of NN=PR, and use PR instead of NNs!
  - No dealing with numerous hyperparameters.
  - No convergence issues.
  - No "fake minima" (NN iteration settles on a local min).

Possible drawbacks/remedies of PR:

- Large memory requirement. Maybe use R's **bigmemory** package (with backing store).
- Run time (worse than NN????). C code, and/or GPU.

Some of Our Experimental Results

Some of Our Experimental Results

- Compared PR vs. NNs on a wide variety of datasets.

Some of Our Experimental Results

- Compared PR vs. NNs on a wide variety of datasets.
  - PR: plain or with PCA beforehand
  - KF: **kerasformula**, R NN pkg.
  - DN: **deepnet**, R NN pkg.

Some of Our Experimental Results

- Compared PR vs. NNs on a wide variety of datasets.
  - PR: plain or with PCA beforehand
  - KF: **kerasformula**, R NN pkg.
  - DN: **deepnet**, R NN pkg.
- Calculated accuracy (mean abs. prediction error, prop. of correct classification).

Some of Our Experimental Results

- Compared PR vs. NNs on a wide variety of datasets.
  - PR: plain or with PCA beforehand
  - KF: **kerasformula**, R NN pkg.
  - DN: **deepnet**, R NN pkg.
- Calculated accuracy (mean abs. prediction error, prop. of correct classification).
- No data cleaning.

Some of Our Experimental Results

- Compared PR vs. NNs on a wide variety of datasets.
  - PR: plain or with PCA beforehand
  - KF: **kerasformula**, R NN pkg.
  - DN: **deepnet**, R NN pkg.
- Calculated accuracy (mean abs. prediction error, prop. of correct classification).
- No data cleaning.
- In every single dataset, **PR matched or exceeded the accuracy of NNs**.

Some of Our Experimental Results

- Compared PR vs. NNs on a wide variety of datasets.
  - PR: plain or with PCA beforehand
  - KF: **kerasformula**, R NN pkg.
  - DN: **deepnet**, R NN pkg.
- Calculated accuracy (mean abs. prediction error, prop. of correct classification).
- No data cleaning.
- In every single dataset, **PR matched or exceeded the accuracy of NNs**.
- **Warning:** Beware of "p-hacking" effects. Don't take timings rankings overly seriously.

Programmer/Engineer Wages

Programmer/Engineer Wages

setting	accuracy
PR, 1	25595.63
PR, 2	24930.71
PR, 3,2	24586.75
PR, 4,2	**24570.04**
KF, default	27691.56
KF, layers 5,5	26804.68
KF, layers 2,2,2	27394.35
KF, layers 12,12	27744.56

Prog./Eng. Occupation

Prog./Eng. Occupation

setting	accuracy
PR, 1	0.3741
PR, 2	**0.3845**
KF, default	0.3378
KF, layers 5,5	0.3398
KF, layers 500	0.3401
KF, layers 5,5; dropout 0.1	0.3399
KF, layers 256,128; dropout 0.8	0.3370

Million Song Data, predict year

Million Song Data, predict year

setting	accuracy
PR, 1, PCA	7.7700
PR, 2, PCA	**7.5758**
KF, default	8.4300
KF, layers 5,5	7.9381
KF, layers 2,2	8.1719
DN, layers 2,2	7.8809
DN, layers 3,2	7.9458
DN, layers 3,3	7.8060
DN, layers 2,2,2	8.7796

UCI Forest Cover Data, predict type

UCI Forest Cover Data, predict type

setting	accuracy
PR, 1	0.69
PR, 3	**0.80**
KF, layers 5,5	0.72
reader report, NN	0.75

NYC Taxi Data, predict trip time

NYC Taxi Data, predict trip time

setting	accuracy
PR, 1	**580.6935**
PR, 2	591.1805
DN, layers 5,5	592.2224
DN, layers 5,5,5	623.5437
DN, layers 2,2,2	592.0192

Note: Sorely needs data cleaning.

What about Image Classification?

What about Image Classification?

- A work in progress.

What about Image Classification?

- A work in progress.
- Source of NN pride, in CNNs.

What about Image Classification?

- A work in progress.
- Source of NN pride, in CNNs.
- What about PR?

What about Image Classification?

- A work in progress.
- Source of NN pride, in CNNs.
- What about PR? Should do as well, due to NN=PR.

What about Image Classification?

- A work in progress.
- Source of NN pride, in CNNs.
- What about PR? Should do as well, due to NN=PR.
- **MYTH:** CNNs do well because of NN.

What about Image Classification?

- A work in progress.
- Source of NN pride, in CNNs.
- What about PR? Should do as well, due to NN=PR.
- **MYTH:** CNNs do well because of NN. No, they do well because of "C."

What about Image Classification?

- A work in progress.
- Source of NN pride, in CNNs.
- What about PR? Should do as well, due to NN=PR.
- **MYTH:** CNNs do well because of NN. No, they do well because of "C."
- "C" is **standard old-fashioned image ops, not NN —**

What about Image Classification?

- A work in progress.
- Source of NN pride, in CNNs.
- What about PR? Should do as well, due to NN=PR.
- **MYTH:** CNNs do well because of NN. No, they do well because of "C."
- "C" is **standard old-fashioned image ops, not NN** — tiling, filtering etc.

What about Image Classification?

- A work in progress.
- Source of NN pride, in CNNs.
- What about PR? Should do as well, due to NN=PR.
- **MYTH:** CNNs do well because of NN. No, they do well because of "C."
- "C" is **standard old-fashioned image ops, not NN** — tiling, filtering etc.
- So in principle PR should perform as well.

What about Image Classification?

- A work in progress.
- Source of NN pride, in CNNs.
- What about PR? Should do as well, due to NN=PR.
- **MYTH:** CNNs do well because of NN. No, they do well because of "C."
- "C" is **standard old-fashioned image ops, not NN** — tiling, filtering etc.
- So in principle PR should perform as well.
- But so far we have not had a chance to do much with "C."

What about Image Classification?

- A work in progress.
- Source of NN pride, in CNNs.
- What about PR? Should do as well, due to NN=PR.
- **MYTH:** CNNs do well because of NN. No, they do well because of "C."
- "C" is **standard old-fashioned image ops, not NN** — tiling, filtering etc.
- So in principle PR should perform as well.
- But so far we have not had a chance to do much with "C."
- Have just done non="C", using PCA for dimension reduction.

What about Image Classification?

- A work in progress.
- Source of NN pride, in CNNs.
- What about PR? Should do as well, due to NN=PR.
- **MYTH:** CNNs do well because of NN. No, they do well because of "C."
- "C" is **standard old-fashioned image ops, not NN** — tiling, filtering etc.
- So in principle PR should perform as well.
- But so far we have not had a chance to do much with "C."
- Have just done non="C", using PCA for dimension reduction.
- Respectable, e.g. 98.7% on MNIST, but need to do serious use of "C."

Nonlinear "PCA"

Nonlinear "PCA"

- PCA may be OK for dimension reduction.

Nonlinear "PCA"

- PCA may be OK for dimension reduction.
- But we also want visualization in 2-D.

Nonlinear "PCA"

- PCA may be OK for dimension reduction.
- But we also want visualization in 2-D. And nonlinear data is a challenge.

Nonlinear "PCA"

- PCA may be OK for dimension reduction.
- But we also want visualization in 2-D. And nonlinear data is a challenge.
- ML favorite is t-sne.

Nonlinear "PCA"

- PCA may be OK for dimension reduction.
- But we also want visualization in 2-D. And nonlinear data is a challenge.
- ML favorite is t-sne. Similar but much faster is UMAP.

Nonlinear "PCA"

- PCA may be OK for dimension reduction.
- But we also want visualization in 2-D. And nonlinear data is a challenge.
- ML favorite is t-sne. Similar but much faster is UMAP.
- Our idea: Form polynomials, then do PCA. Our package: **prVis**.

Nonlinear "PCA"

- PCA may be OK for dimension reduction.
- But we also want visualization in 2-D. And nonlinear data is a challenge.
- ML favorite is t-sne. Similar but much faster is UMAP.
- Our idea: Form polynomials, then do PCA. Our package: **prVis**.
- *github.com/matloff/prVis*

Example: Swiss Roll

Example: Swiss Roll

- Artificial data, due to D. Surendran.

Example: Swiss Roll

- Artificial data, due to D. Surendran.
- Designed to be a mixture of 4 components.

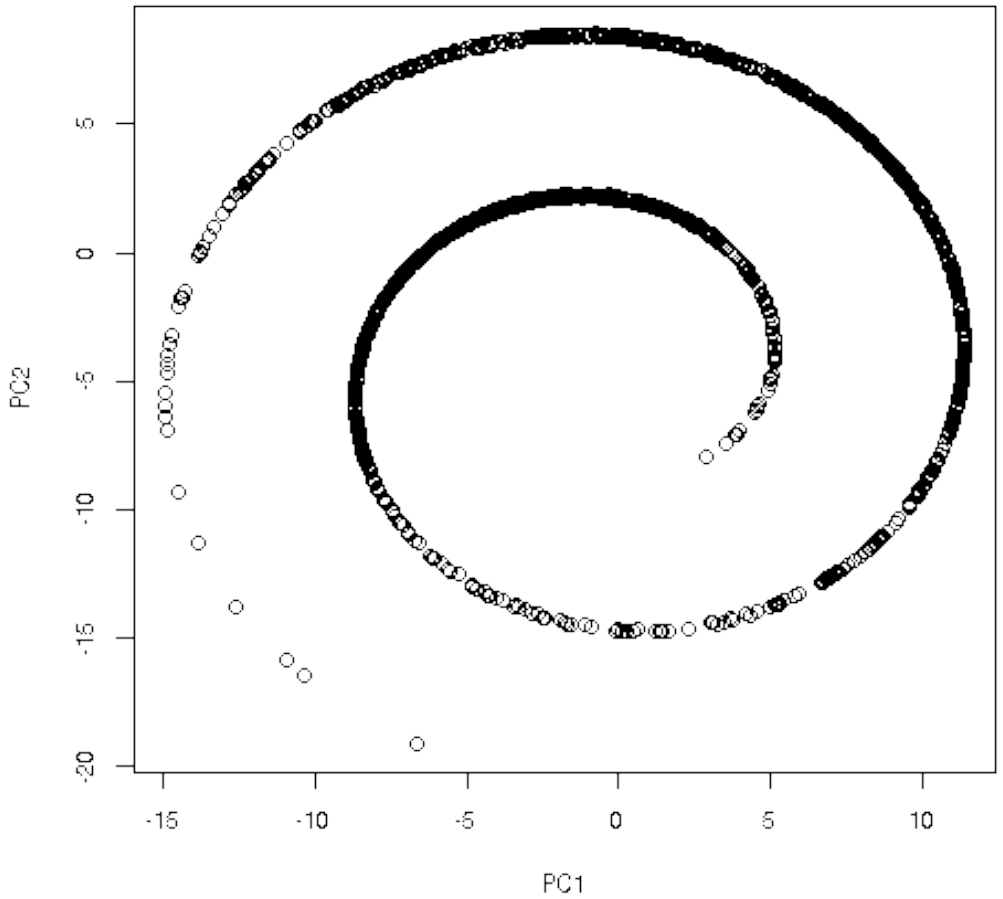
Example: Swiss Roll

- Artificial data, due to D. Surendran.
- Designed to be a mixture of 4 components.
- **The Test**: Will any of these visualization tools detect that?

Example: Swiss Roll

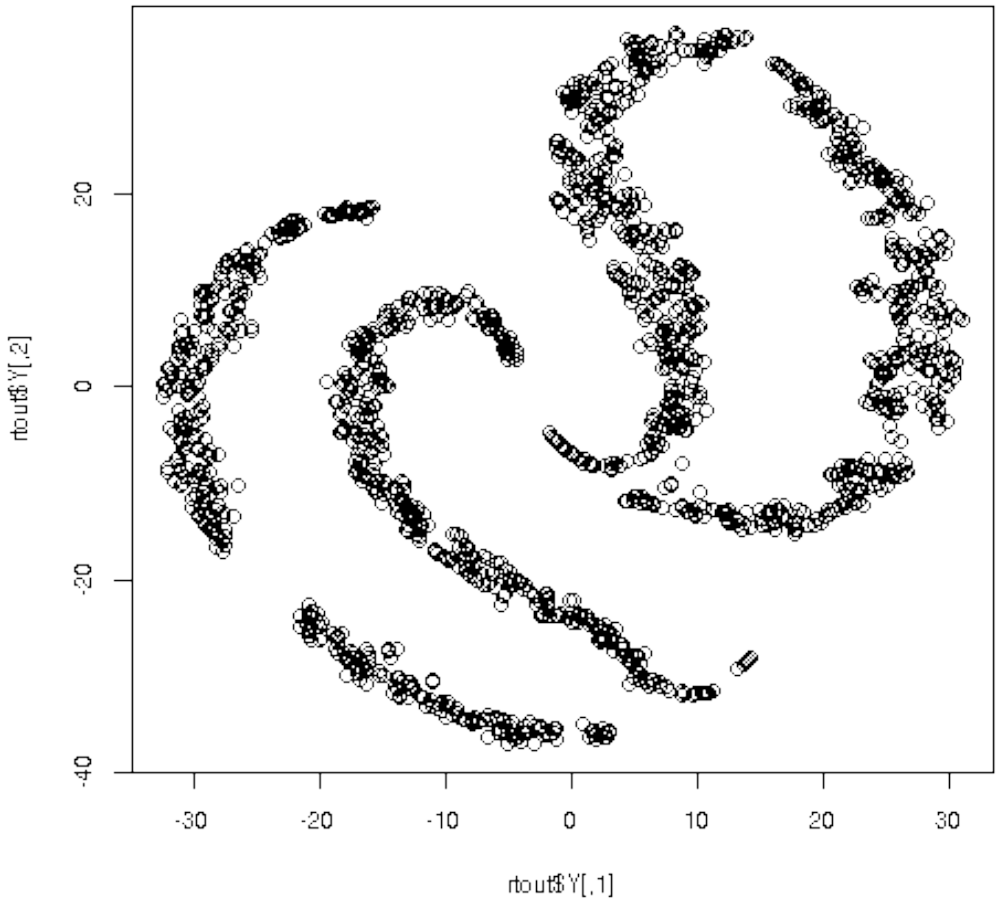
- Artificial data, due to D. Surendran.
- Designed to be a mixture of 4 components.
- **The Test**: Will any of these visualization tools detect that?
- Let's temporarily pretend we don't know there are 4.

PCA



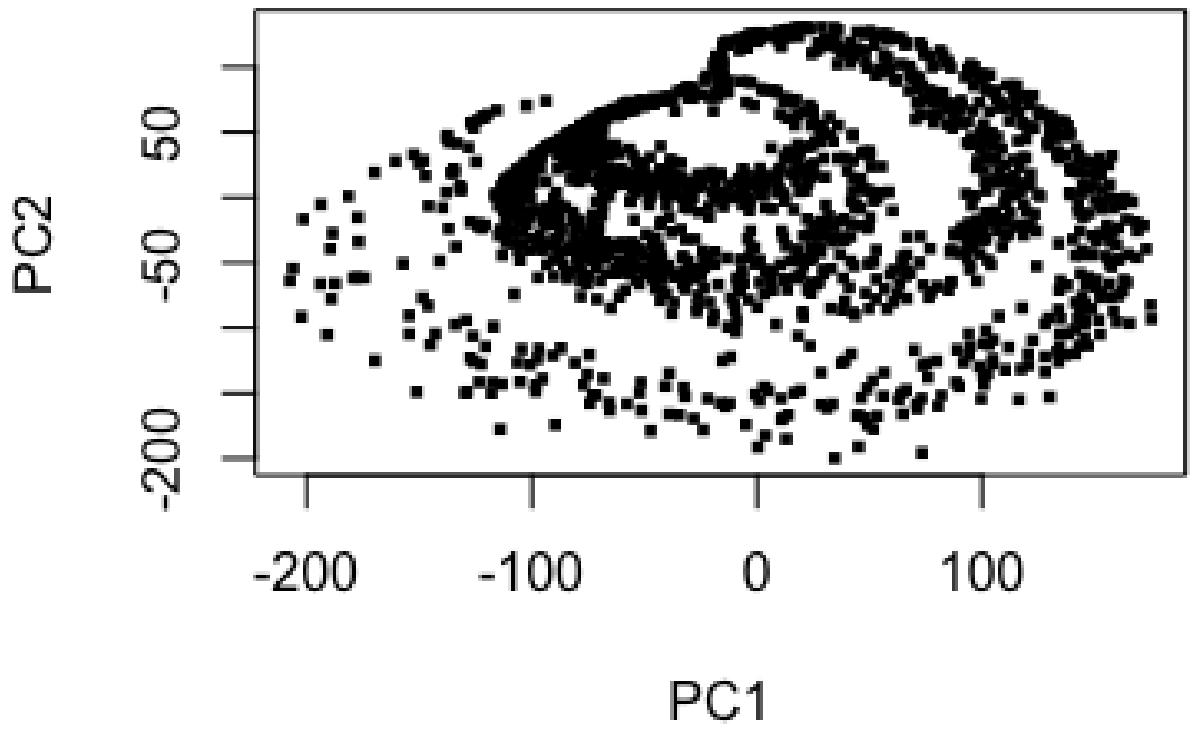
No clue at all that there are 4 components.

t-sne



3 components? 4? 5? Even 1? Not clear.

prVis



Fairly clear there are 4 components.

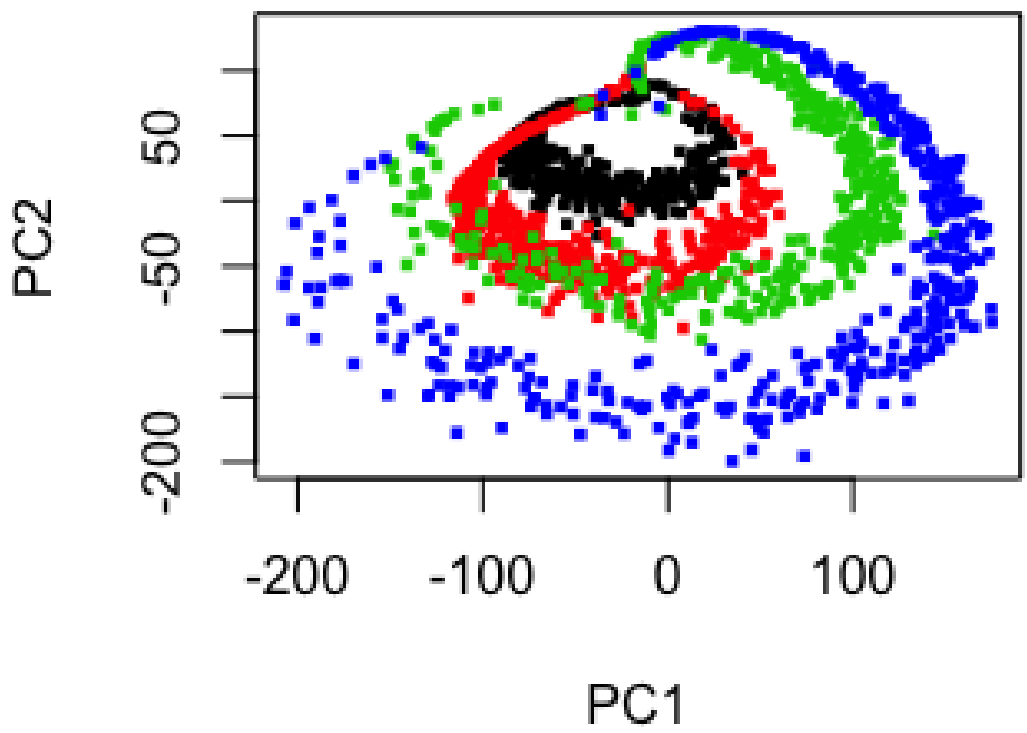
prVis — Reveal

prVis — Reveal

Now let's un-pretend, color coding the known components.

prVis — Reveal

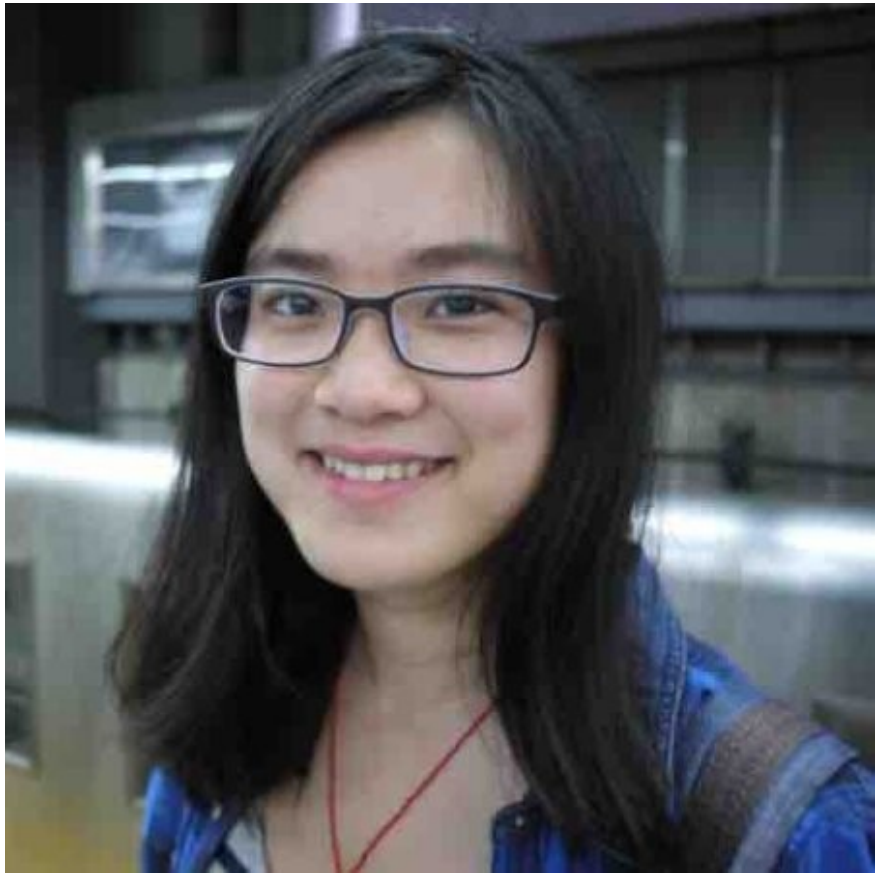
Now let's un-pretend, color coding the known components.



Yep!

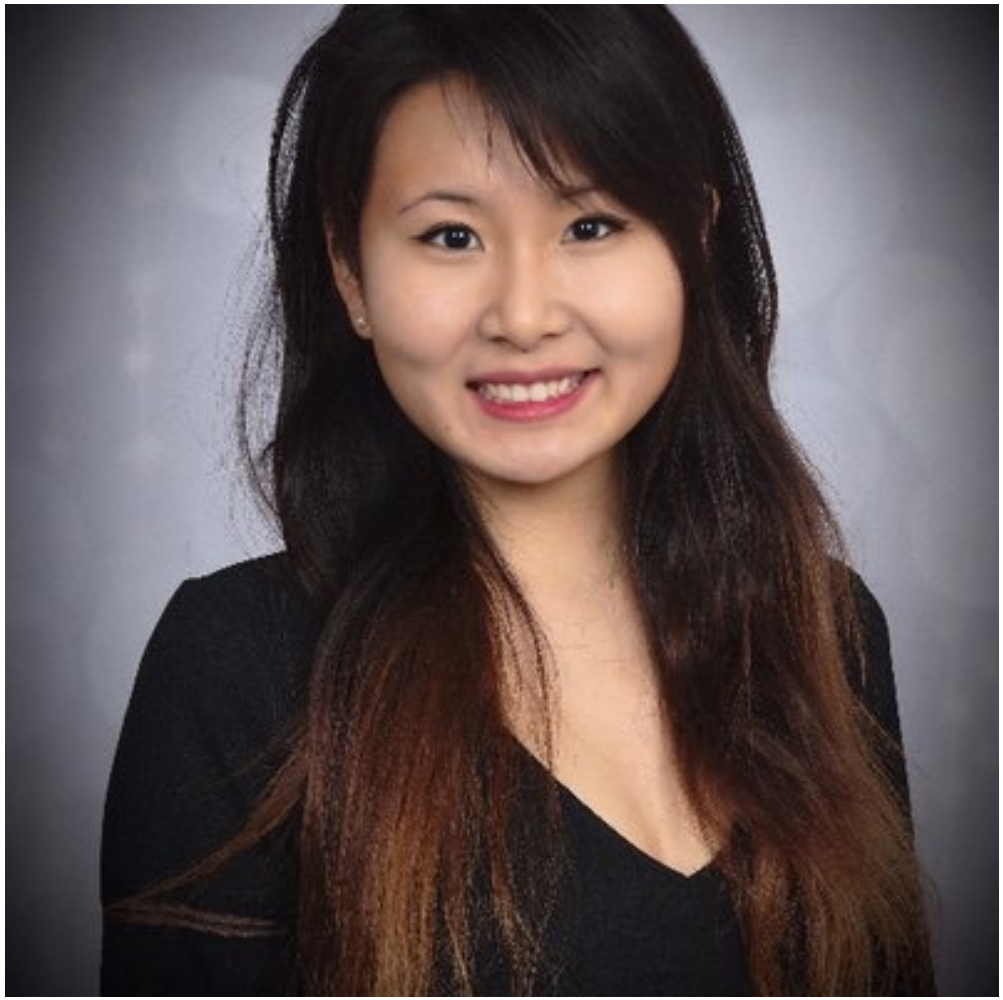
The Team!

The Team!



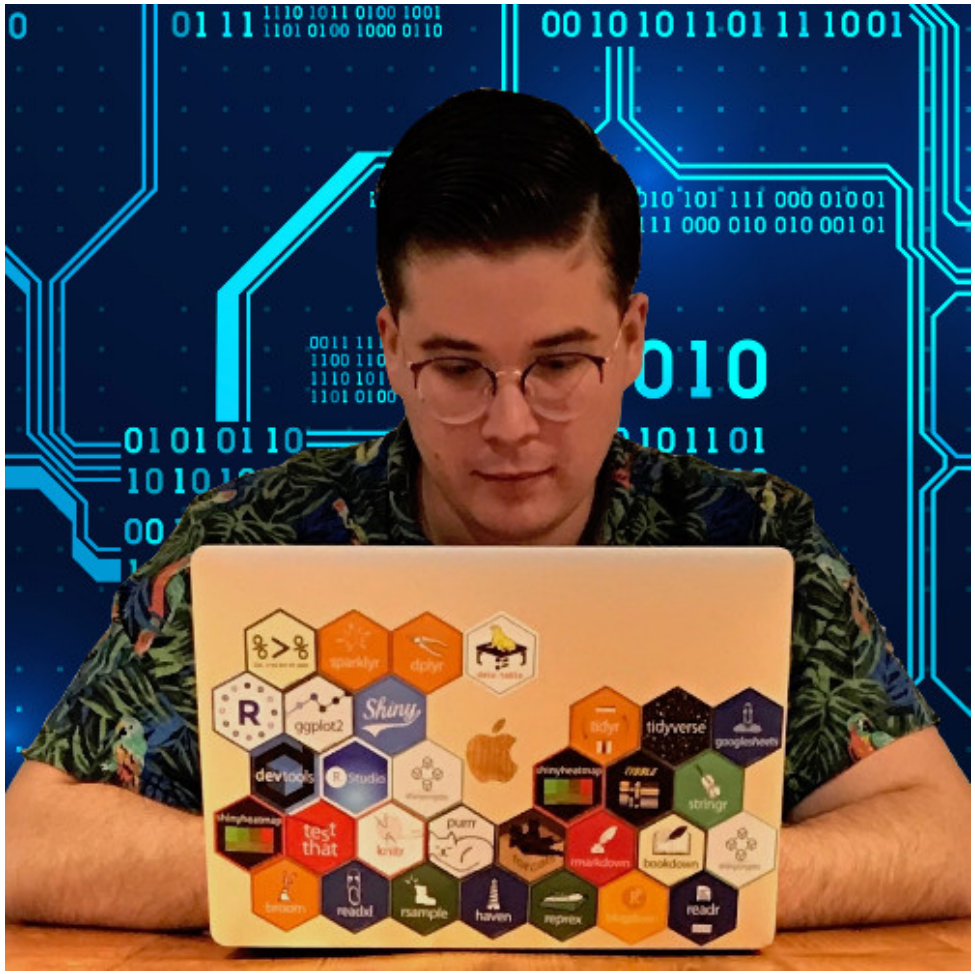
Xi Cheng

The Team! (contd.)



Tiffany Jiang

The Team! (contd.)



Bohdan Khomtchouk

The Team! (contd.)

Matt Kotila

The Team! (contd.)

Pete Mohanty

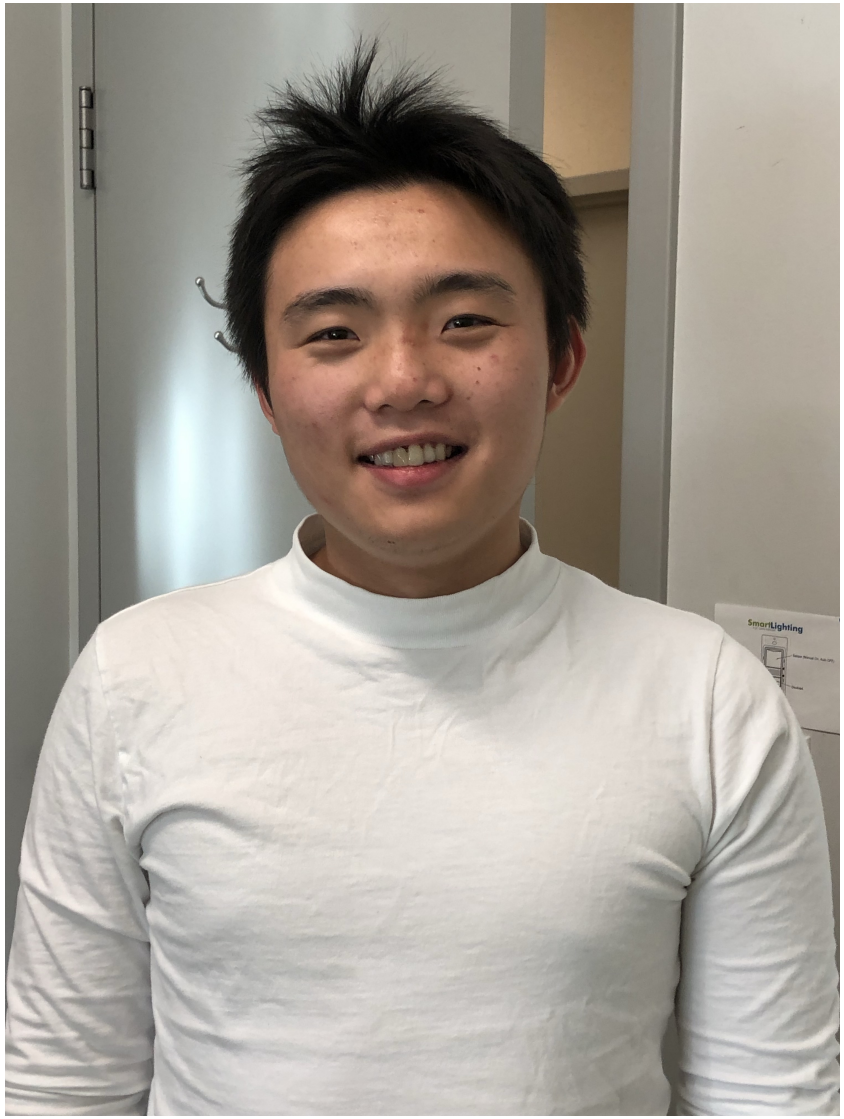
The Team! (contd.)

Robert Tucker

The Team! (contd.)

Robin Yancey

The Team! (contd.)



Allan Zhao

Thanks

Thanks

Thanks to all!

Appendix

Appendix

Backup slides:

Multicollinearity in NNs

Multicollinearity in NNs

- Test of a good theory: Does it predict new phenomena?

Multicollinearity in NNs

- Test of a good theory: Does it predict new phenomena? E.g. Einstein "solar eclipse experiment."

Multicollinearity in NNs

- Test of a good theory: Does it predict new phenomena? E.g. Einstein "solar eclipse experiment."
- PR is well known to be prone to multicollinearity.

Multicollinearity in NNs

- Test of a good theory: Does it predict new phenomena? E.g. Einstein "solar eclipse experiment."
- PR is well known to be prone to multicollinearity.
- The higher the degree in PR, the worse the multicollinearity.

Multicollinearity in NNs

- Test of a good theory: Does it predict new phenomena? E.g. Einstein "solar eclipse experiment."
- PR is well known to be prone to multicollinearity.
- The higher the degree in PR, the worse the multicollinearity.
- Thus NN=PR predicts that **the outputs of the NN layers will have multicollinearity**, with each layer having great amounts of multicollinearity.

Multicollinearity in NNs

- Test of a good theory: Does it predict new phenomena? E.g. Einstein "solar eclipse experiment."
- PR is well known to be prone to multicollinearity.
- The higher the degree in PR, the worse the multicollinearity.
- Thus NN=PR predicts that **the outputs of the NN layers will have multicollinearity**, with each layer having great amounts of multicollinearity.
- Is it true?

Multicollinearity in NNs

- Test of a good theory: Does it predict new phenomena? E.g. Einstein "solar eclipse experiment."
- PR is well known to be prone to multicollinearity.
- The higher the degree in PR, the worse the multicollinearity.
- Thus NN=PR predicts that **the outputs of the NN layers will have multicollinearity**, with each layer having great amounts of multicollinearity.
- Is it true? Yes!

Multicollinearity Example:

Multicollinearity Example:

MNIST data, NN via R **keras** package.

Multicollinearity Example:

MNIST data, NN via R **keras** package.
Use VIF as measure of multicollinearity.

Multicollinearity Example:

MNIST data, NN via R **keras** package.
Use VIF as measure of multicollinearity.

layer	% VIFs > 10	mean VIF
1	0.0078125	4.3537
2	0.9921875	46.84217
3	1	5.196113×10^{13}