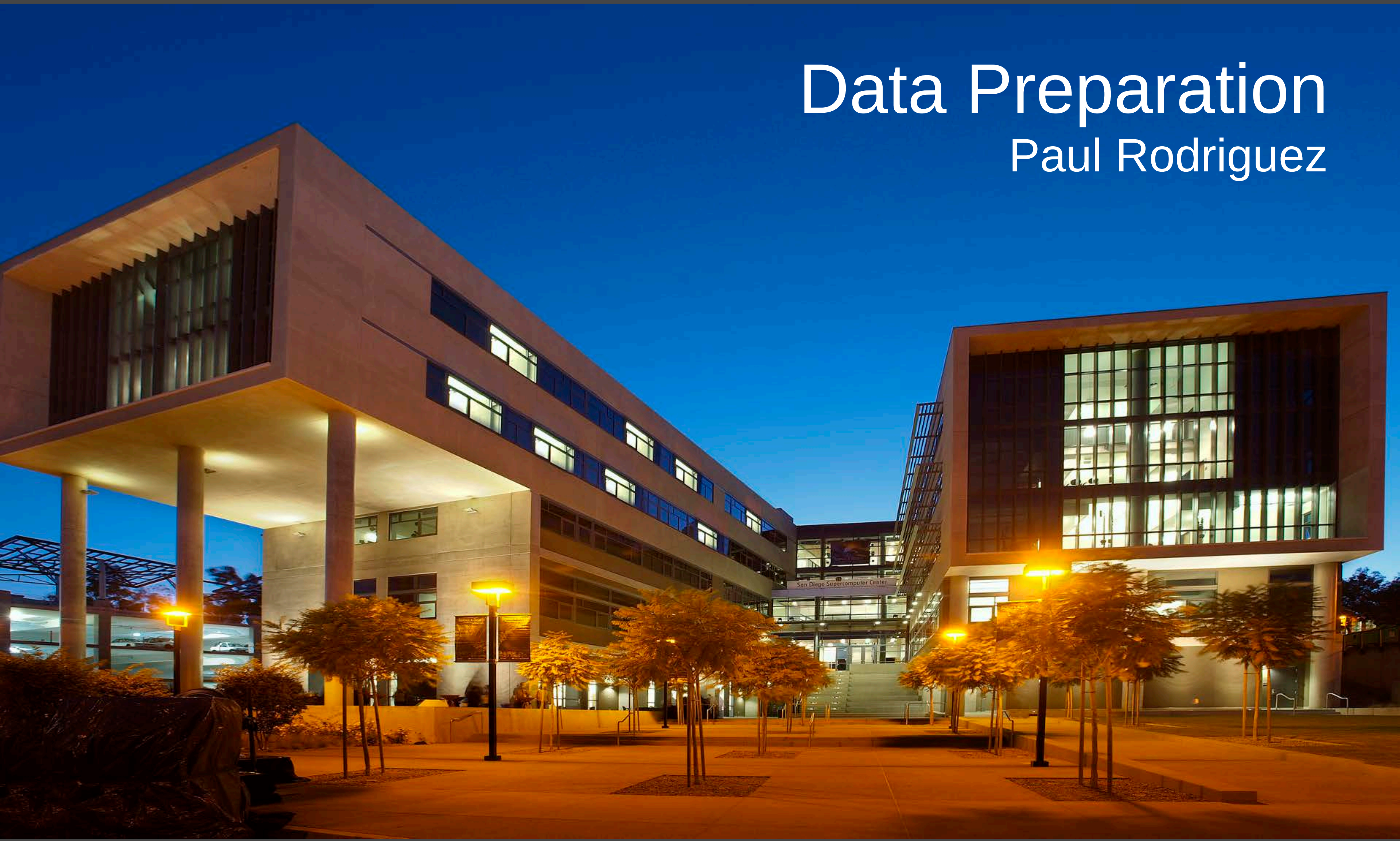
Data Preparation

Paul Rodriguez



SDSC SAN DIEGO SUPERCOMPUTER CENTER

UC San Diego

Overview

- Highlights of data prep
- Variable Selection and Reduction

The Importance of Data Prep

- “Garbage in, garbage out”

- Sometimes takes 60-80% of the whole data mining effort

Working definition

- **Data Preparation:**
  - Cleaning – deal with noise, outliers, missing data

Working definition

- **Data Preparation:**
  - Cleaning – deal with noise, outliers, missing data
  - Filtering – select subsets or summaries

Working definition

- **Data Preparation:**
  - Cleaning – deal with noise, outliers, missing data
  - Filtering – select subsets or summaries
  - Transforming – normalize or enhance

Working definition

- **Data Preparation:**
  - Cleaning – deal with noise, outliers, missing data
  - Filtering – select subsets or summaries
  - Transforming – normalize or enhance
  - Organizing data - aka ‘data wrangling’ or ‘data munging’

Working definition

- **Data Preparation:**
  - Cleaning – deal with noise, outliers, missing data
  - Filtering – select subsets or summaries
  - Transforming – normalize or enhance
  - Organizing data - aka ‘data wrangling’ or ‘data munging’
  - Variable Selection/Dimension Reduction

Working definition

- **Data Preparation:**
  - Cleaning – deal with noise, outliers, missing data
  - Filtering – select subsets or summaries
  - Transforming – normalize or enhance
  - Organizing data - aka ‘data wrangling’ or ‘data munging’
  - Variable Selection/Dimension Reduction

In a nutshell, prepare data for modeling

Missing Data (NA vs NULL)

- **Not applicable - NA**

  e.g. spouse name depends on marital status

- **Not available - NULL**

  unknown

  not entered

- **In R:**

  NA is logically missing (ie the value won't be available): is.na()

  NULL is object not yet defined (ie the object is not available): is.null()

Missing Data

- **What are frequency counts of missing variables?**

    Are entries missing completely at random or contingent on some other variable?

Quick Approaches

- Delete instances

and/or

- Delete attributes with high missing-ness

Simple Imputation

- Use the attribute mean
- Use the attribute mean for each class label

Complicated Imputation

- **Use a model (based on other attributes) to infer missing value**

Complicated Imputation

- **Use a model (based on other attributes) to infer missing value**

Best strategy depends on

time vs accuracy tradeoffs

R and missing data

- **Several packages, such as ‘mice’ , ‘amelia’**
- **Produces multiple data sets**
- **Iterates over missing data estimates and linear model estimates**

  Mice uses Gibbs sampling (slower)

  Amelia uses Expectation Maximization (faster)
- **Beware of correlation in variables**

  Matrices not invertible

R and missing data

- **‘Amelia’ package example**
  - 50 attributes from UN voting data
  - 1K-100K entries missing per col for about 20 cols
  - 300K rows ~ 1 hour on compute node (not run on the user’s PC)

time variable for time series models

cross section to select temporal periods

ignore ‘id’ variables

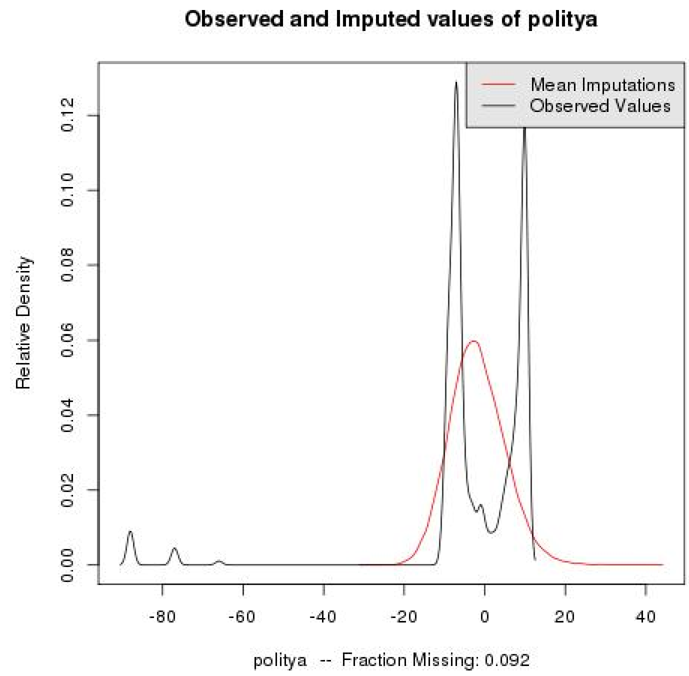
```
# run the imputation
library(‘amelia’)
 a.out <- amelia(data, ts = "year", cs = "dyadid",
                idvars = c("dyadidyr", "cntryera", "statea", "stateb"),
                intercs=FALSE, p2s = 2, m=10, parallel = "multicore”)
```

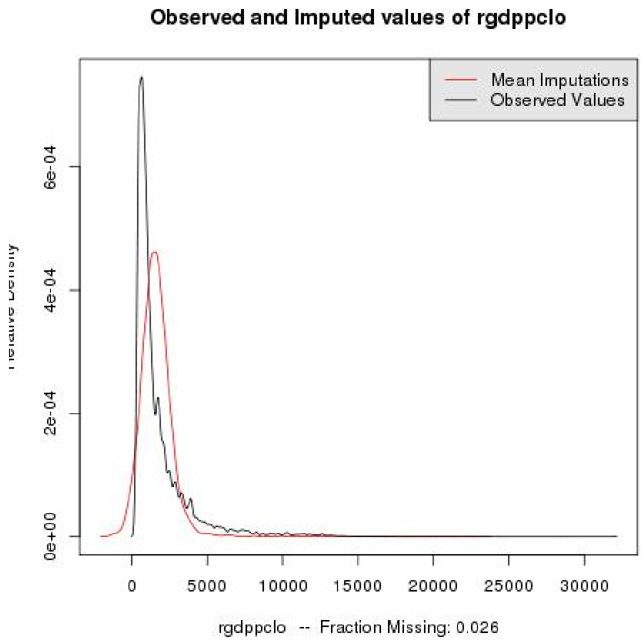
interactions

parallel options

R and missing data

#QA on missing data by comparing density of imputated & original data
compare.density(a.out, var=“politya")
compare.density(a.out,var='rgdpcontg')

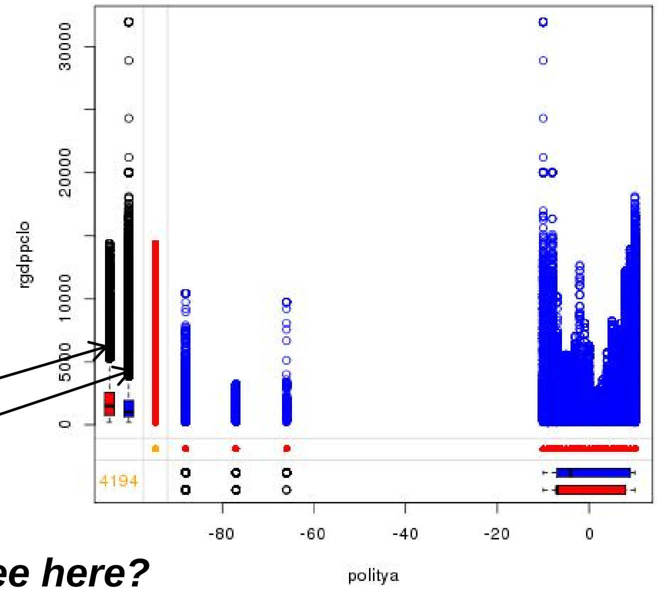




R and missing data

```
# Useful library for printing margin plots, to compare histograms
#                          conditional on missing/non-missing data
library('VIM')
marginplot(gart2use[,c('politya', 'rgdppclo')],
  col=c('blue','red','orange')
```

dist of rgdppclo when
politya missing
politya present



What would you want to see here?

Variable Transformations

- **Engineer new features**
- **Combine attributes**
  e.g. rates and ratios
- **Normalize or Scale data**
- **Discretize data**
  (perhaps more intuitive to deal with binned data)

Feature Engineering is Variable Enhancement

- **Use Domain and world knowledge**

- **Example: given date and location of doctor visits**

  a new variable for Number-of-1st-time-visits

  deduce a new variable for Number-of-visits-over-25-miles

  deduce a new variable for Amount-of-time-between-visits

Re-scaling

- **Mean center** $$x_{new} = x - \mathrm{mean}(x)$$
- **z-score** $$score = \frac{x - \mathrm{mean}(x)}{\mathrm{std}(x)}$$
- **Scale to [0…1]** $$x_{new} = \frac{x - \min(x)}{\max(x) - \min(x)}$$
- **log scaling** $$x_{new} = \log(x)$$

Generally

- Preparing data is based on statistical principles,

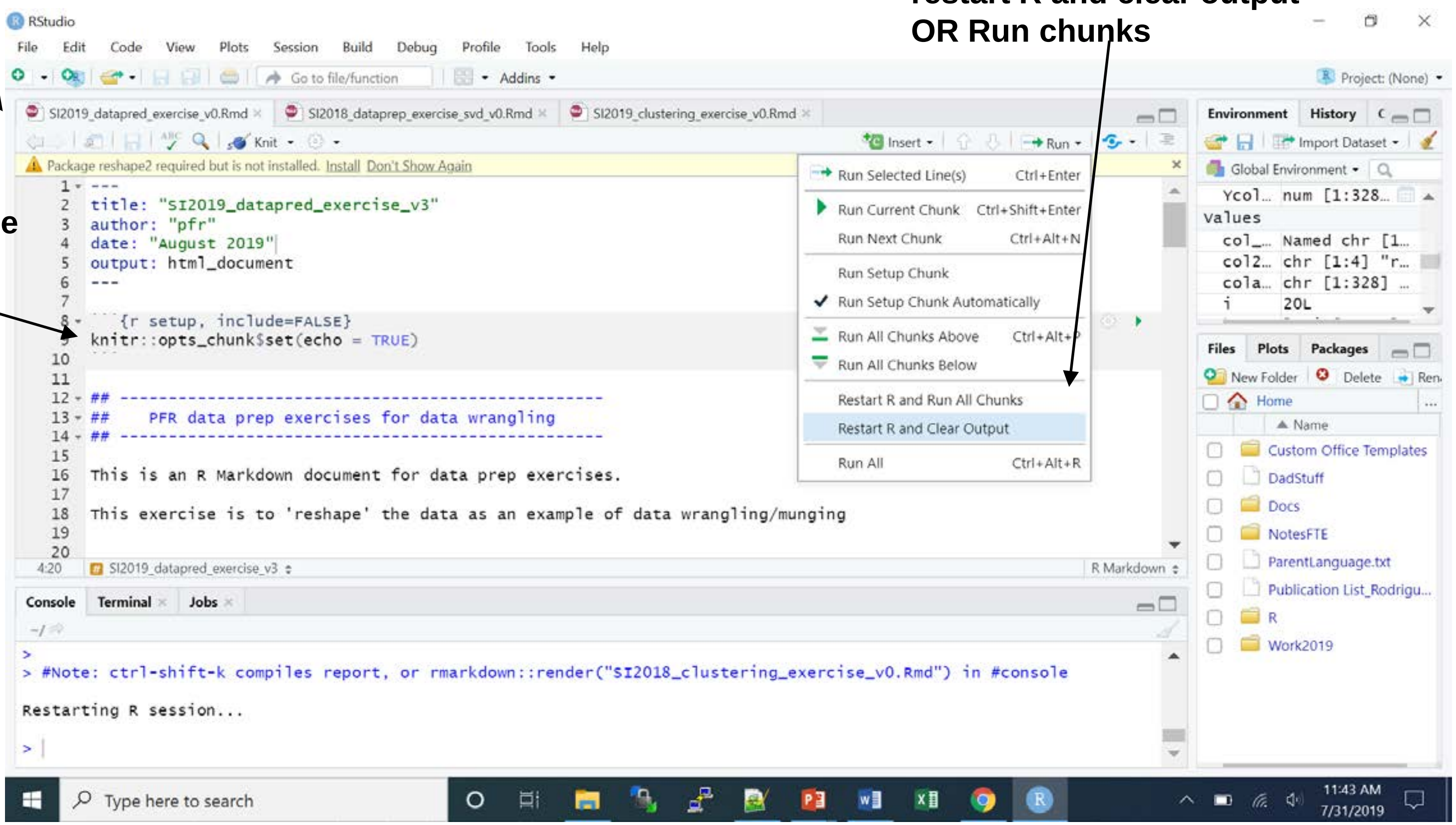
- But also heuristics

Data Wrangling Exercise

Open File -> to load R mark-down exercises ("__.Rmd")

R code chunks inside triple backticks

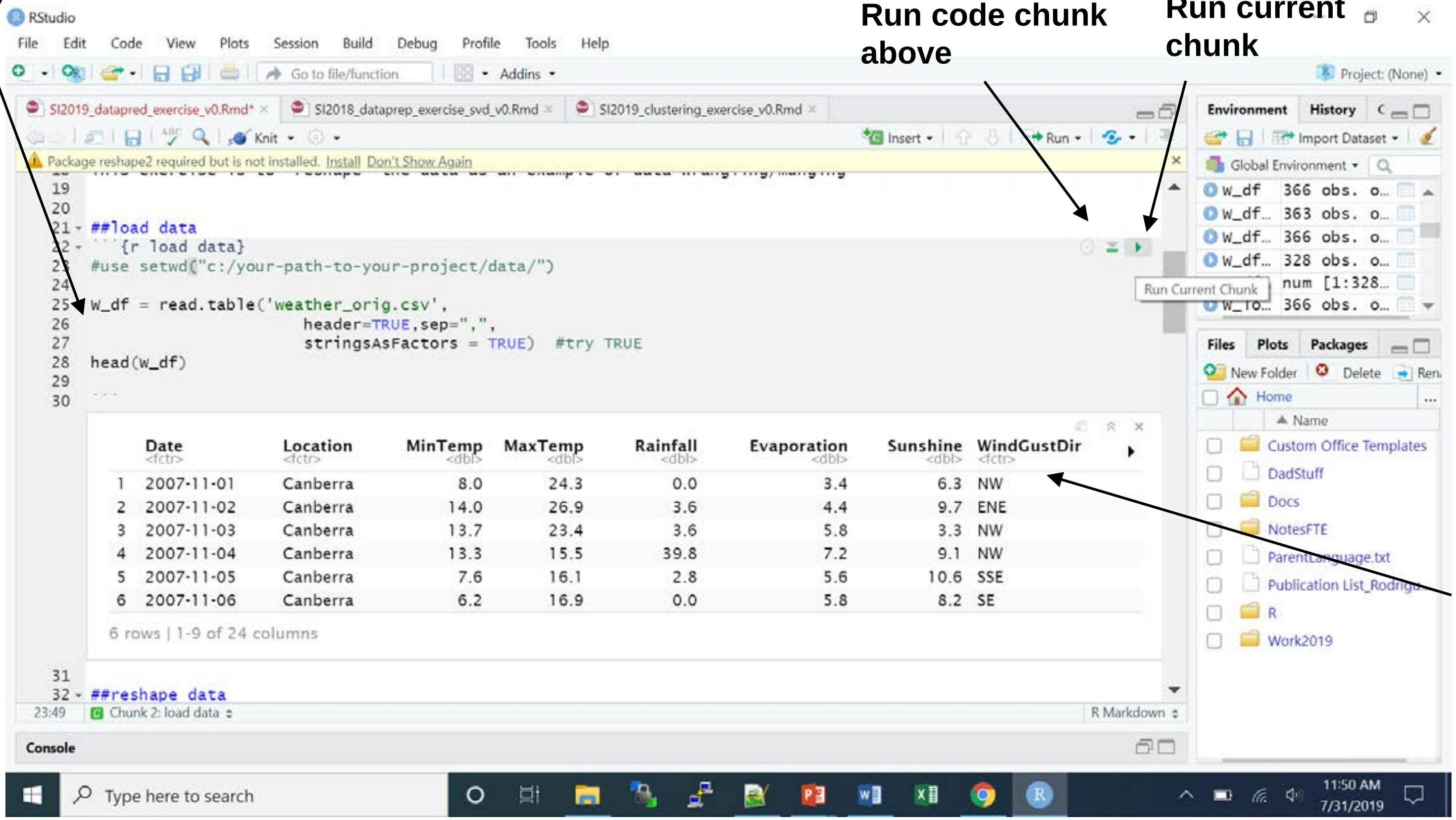
Select Run drop down-> restart R and clear output OR Run chunks



Read file into dataframe and show the first few (head) rows

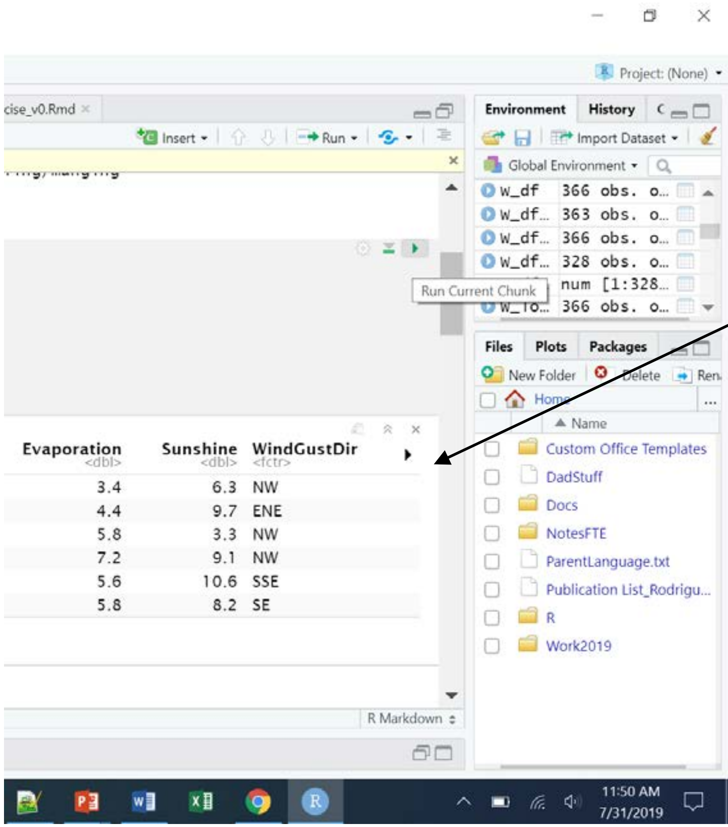
Run code chunk above

Run current chunk

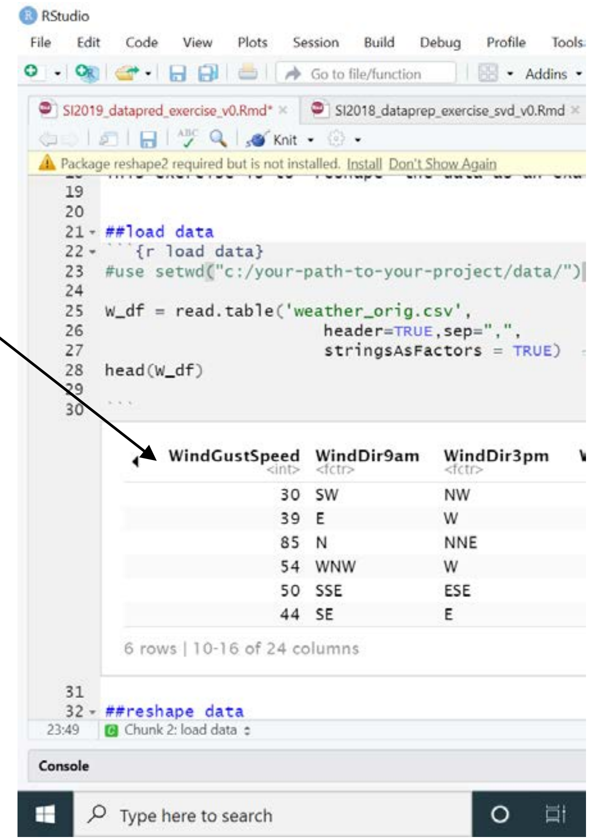


Notice “WindGustDir” describes a category. Scroll to next column to see its value.

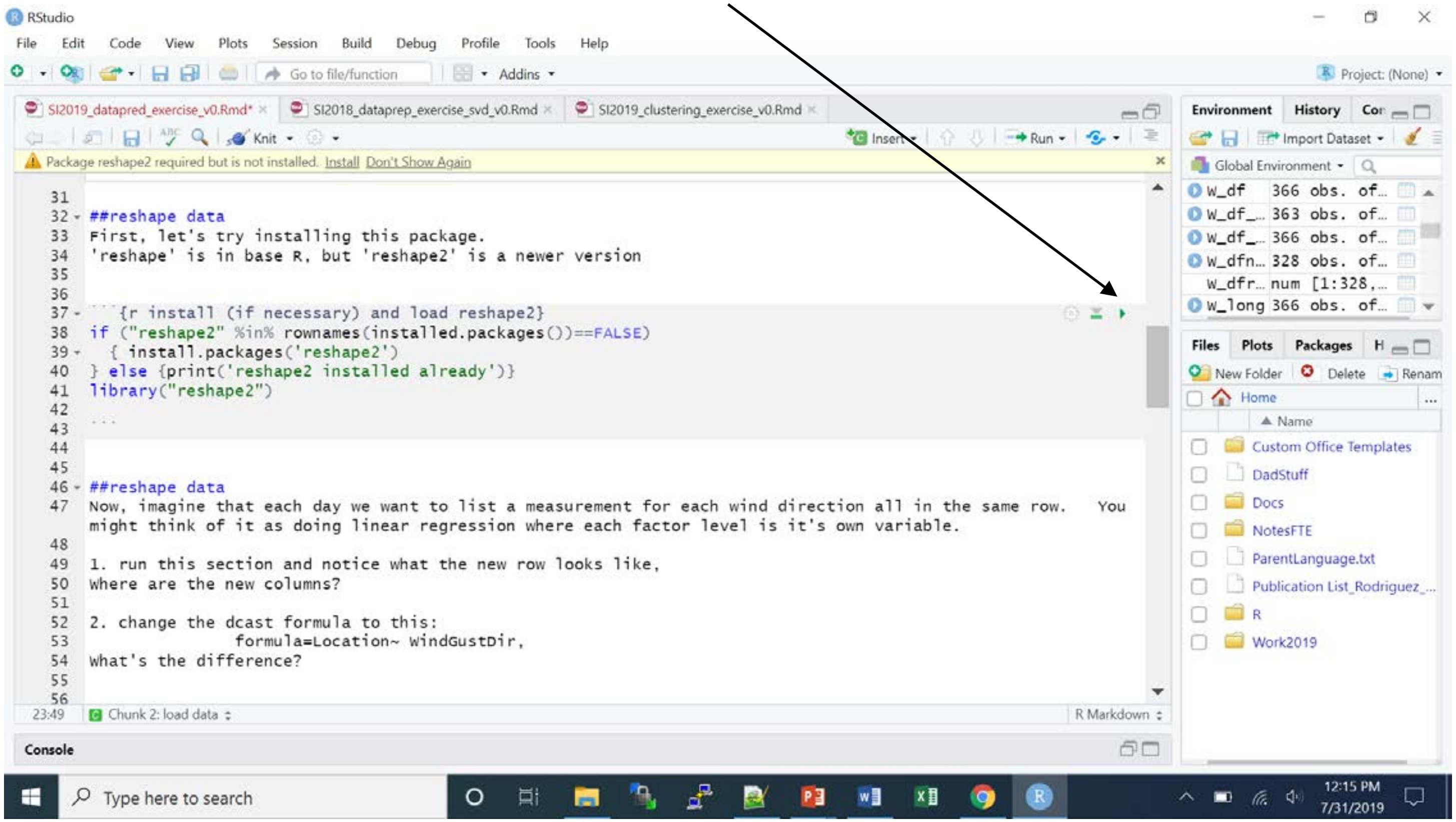
Let's consider WindGustDir & WindGustSpeed as a repeated measurement. Assume we want code these as binary factors for a linear model, i.e. we want to put each direction in its own column.



Scroll to next column to see its value.



Run this chunk to install "reshape2" package (if it is not already installed) and load it into this R session



Long to Wide transform

A1 fx Date

	A	B	C	D	E	F	G	H	I	J
1	Date	Location	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustDi	WindGustSp	WindDir9am
2	11/1/2007	Canberra	8	24.3	0	3.4	6.3	NW	30	SW
3	11/2/2007	Canberra	14	26.9	3.6	4.4	9.7	ENE	39	E
4	11/3/2007	Canberra	13.7	23.4	3.6	5.8	3.3	NW	85	N

date, location and the rest identify the row

WindGustDir entries are labels for the repeated measures

```
# long to wide: ie 'cast' repeated measure into wide table
W_long    =dcast(W_df,
               formula=Date+Location+ ...~ WindGustDir,
                      #date, location and the rest are not repeated
                      #WindGustDir entries are labels for the repeated measures
                fill=0,     #this could be 0 or NA, for example.
                value.var="WindGustSpeed")
                      #this variable has the repeated measurement values

```

this could be 0 or NA

indicate variable that has the repeated measurement values

Transformed Data Matrix

Now: WindGustDir values each have their own column. These are all new columns AFTER the last column of original data frame. Only one value in each row is non-zero.

RainToday <fctr>	RISK_MM <dbl>	RainTomorrow <fctr>	E <dbl>	ENE <dbl>	ESE <dbl>	N <dbl>	NE <dbl>	NNE <dbl>	NNW <dbl>
No	3.6	Yes	0	0	0	0	0	0	0
Yes	3.6	Yes	0	39	0	0	0	0	0
Yes	39.8	Yes	0	0	0	0	0	0	0
Yes	2.8	Yes	0	0	0	0	0	0	0
Yes	0.0	No	0	0	0	0	0	0	0
No	0.2	No	0	0	0	0	0	0	0

64:39 Chunk 4: reshape long to wide R Markdown

Type here to search

Extra to try:

```
# wide to long: ie 'melt' repeated measure into long
#            table
W_melt   =melt(W_cast,
        na.rm=TRUE,
        measure.vars=c(23:39),
        variable.name="WindGustDir_cast")
```

The repeated measures are now in columns 23 to 30

pause

Reading Material

- **Data Preparation for Data Mining by Dorian Pyle**
  - http://www.ebook3000.com/Data-Preparation-for-Data-Mining_88909.html
- **Data mining – Practical Machine learning tools and techniques by Witten & Frank**
  - http://books.google.com

Many Variables

- **More variables => more information, but also more noise and more ways of interactions**

- **2 ways to handle many variables**
  - Variable Selection
  - Dimension reduction methods

Variable selection vs Dimensionality Reduction

- **Prior to algorithm, depends on data and number of variables (P)**
  - For large P, with noise particular to variables, try variable selection
  - For large P, diffuse noise, try dimension reduction by Matrix Factorization

Variable selection

- **Heuristic methods:**

  remove variables with low correlations to outcome

  (other criteria: information gain, sensitivity, etc…)

- **Step wise: add 1 variable at a time and test algorithm on samples**

Variable selection

- **Some algorithms are robust to extra noise variables**

- E.g. Least Angle Regression (L_1 penalty),

penalize small effect sizes (zero them out)

- E.g. Random Forest outputs 'importance'

low importance implies small error effect in the model when removed or permuted

Matrix Factorization:

Given a numeric matrix, can we reduce the number of columns?

Matrix Factorization:

Given a numeric matrix, can we reduce the number of columns?

- Yes, if features are constant or redundant

Matrix Factorization:

Given a numeric matrix, can we reduce the number of columns?

- Yes, if features are constant or redundant
- Yes, if features only contribute noise

Matrix Factorization:

Given a numeric matrix, can we reduce the number of columns?

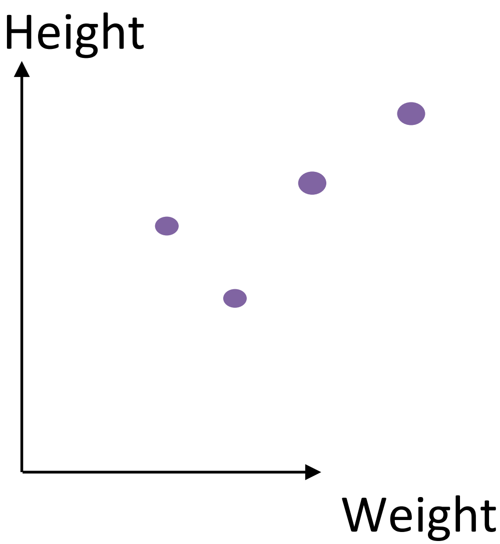
- Yes, if features are constant or redundant
- Yes, if features only contribute noise

Conversely, want features that contribute to variations of the data

Think graphically!
An example in 2D:

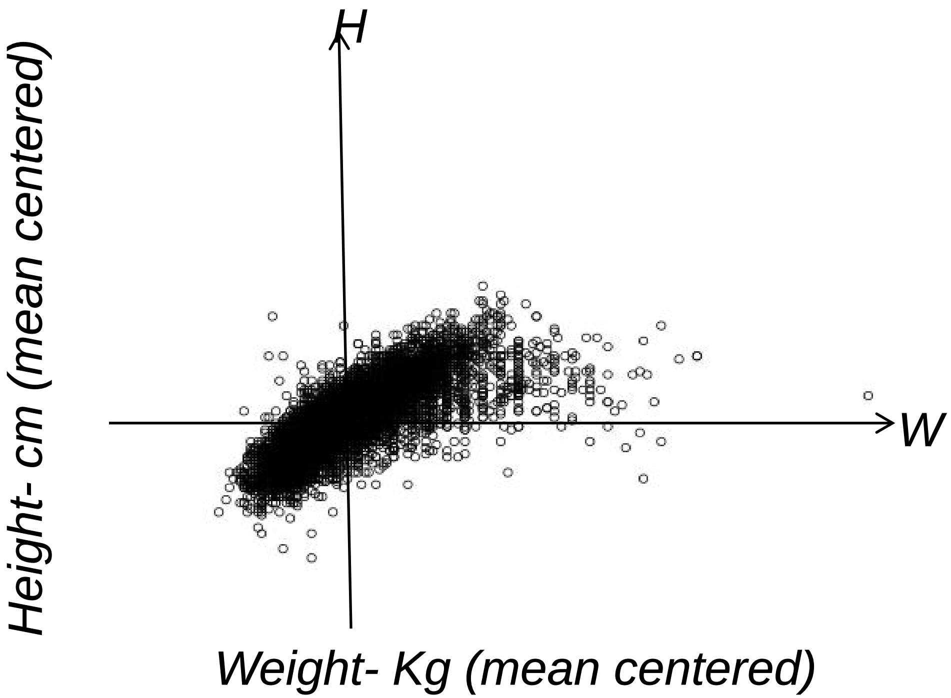
	Weight	Height
S1	50	179
S2	66	175
S3	74	180
S4	94	192

PLOT



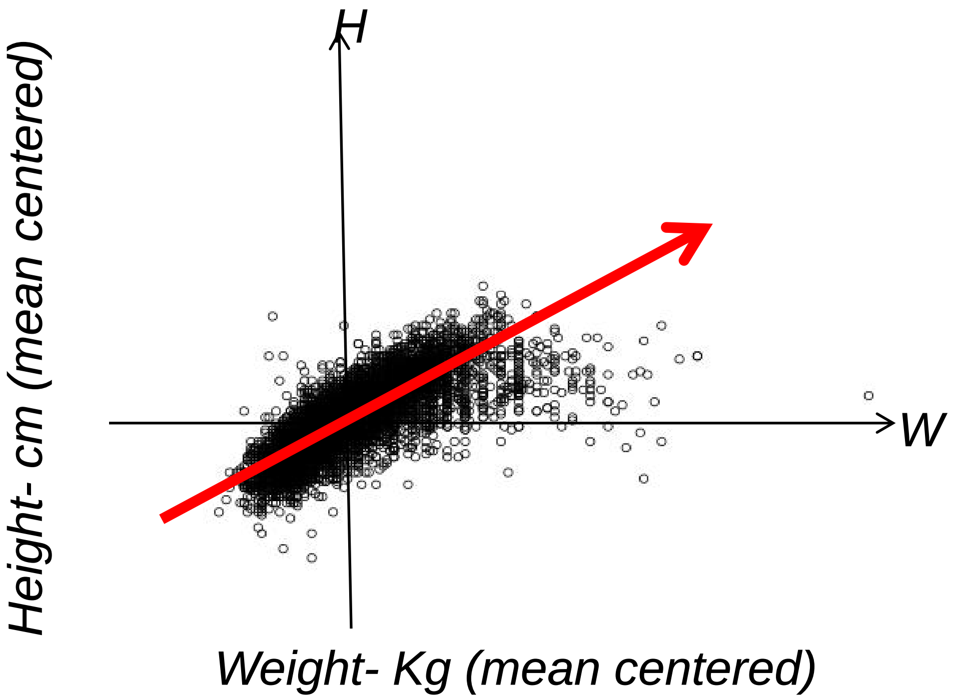
this is the input space, each row is a coordinate point

Example: Athletes' Height by Weight



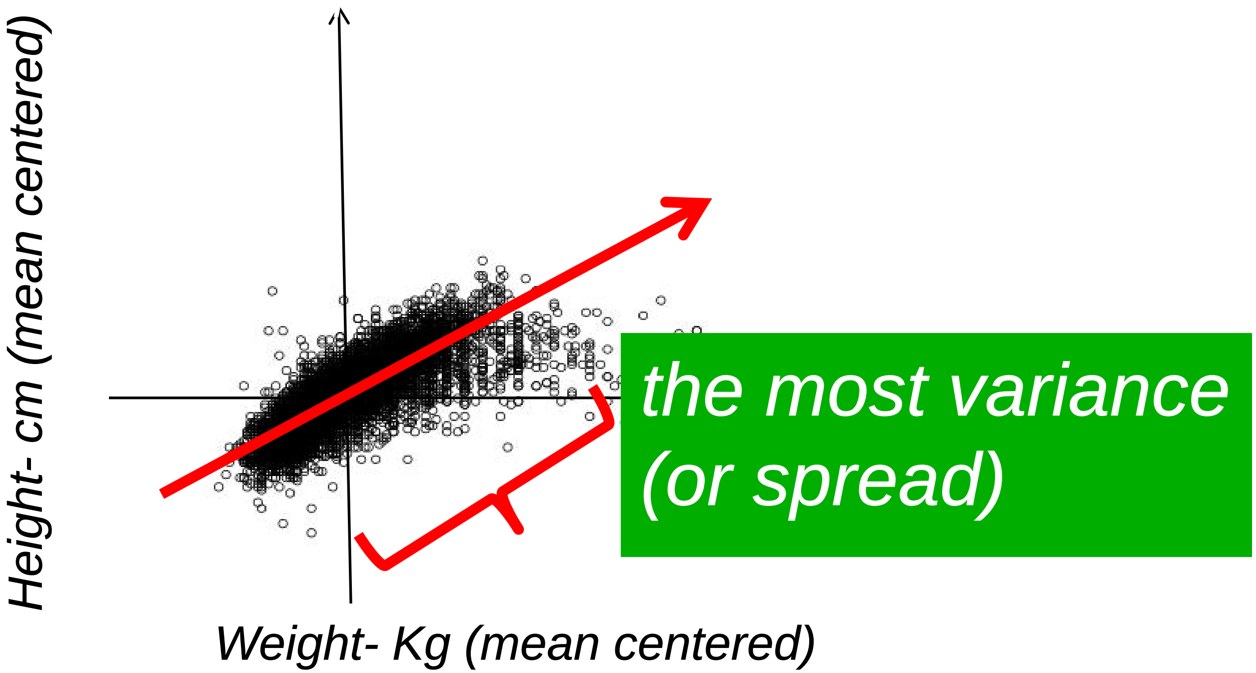
Find a line that aligns with the data.

Example: Athletes' Height by Weight



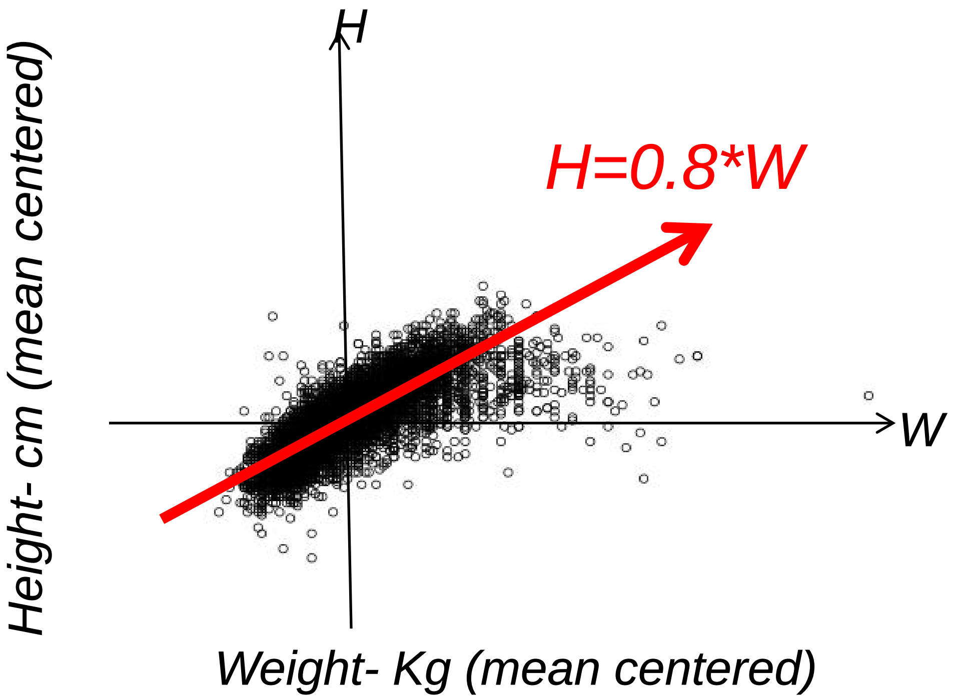
Find a line that aligns with the data.

Example: Athletes' Height by Weight

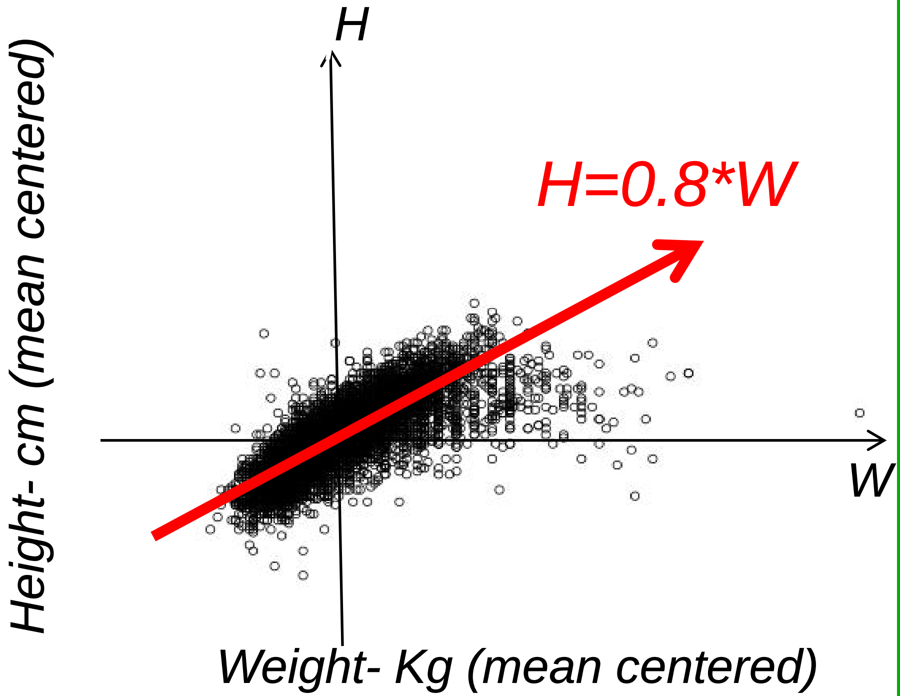


Find a line that aligns with the data.

Example: Athletes' Height by Weight



Find a line that aligns with the data.

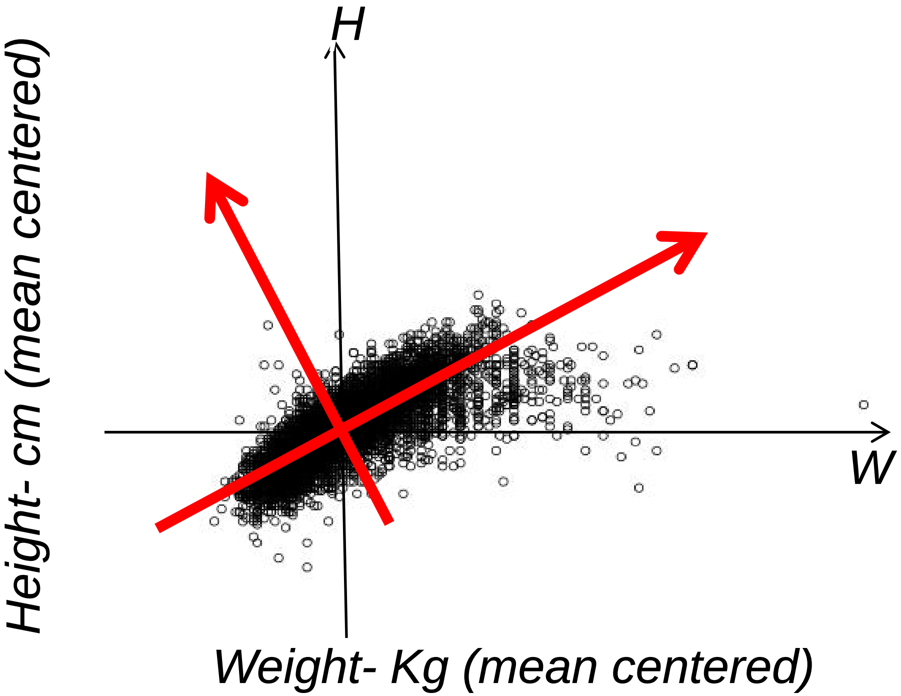


Note that ***(0,0) and (1,0.8) are points on the line that are combinations of H,W***

- OR -

|1 |
| 0.8 |
describes a vector in HxW space

Find a line that aligns with the data.



| -0.8 |
| 1.0 |
describes a 2nd vector, orthogonal to 1st one

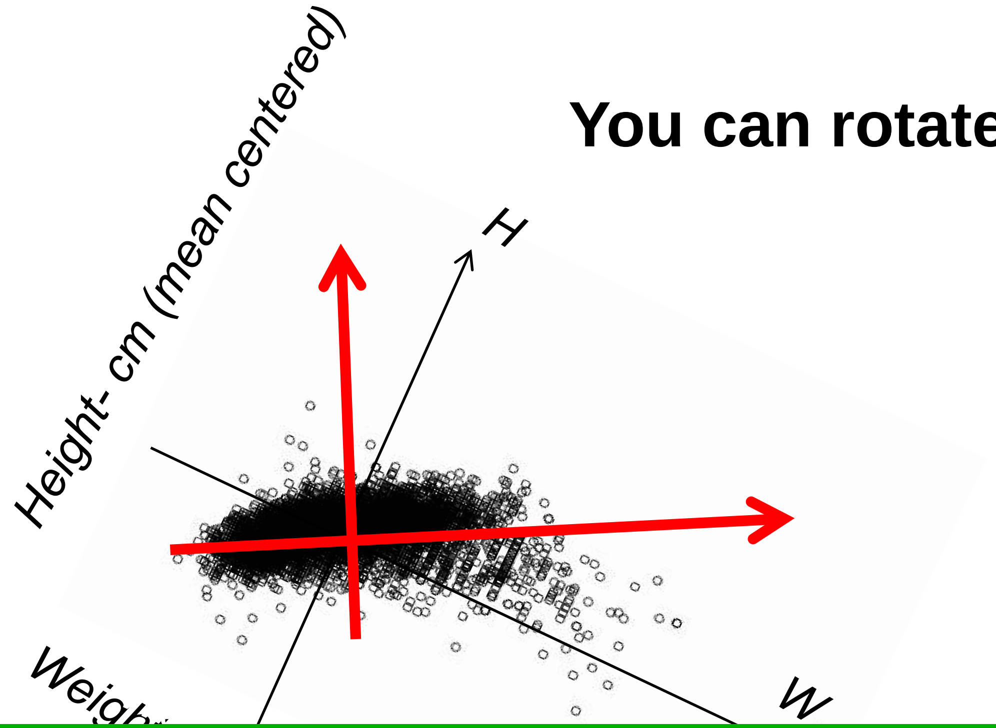
- AND -

Both vectors are columns in a matrix:

$$\begin{vmatrix} 1 & -0.8 \\ 0.8 & 1 \end{vmatrix}$$

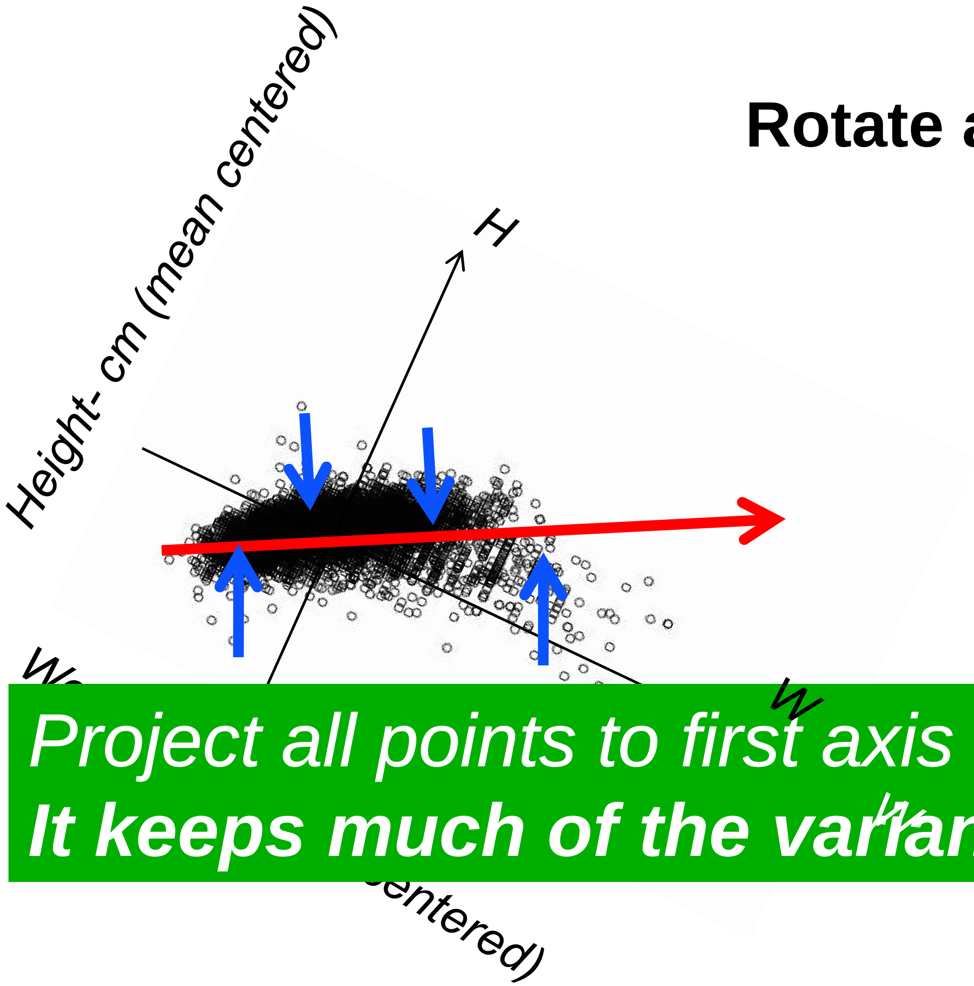
The next direction of most variance.

You can rotate the axes



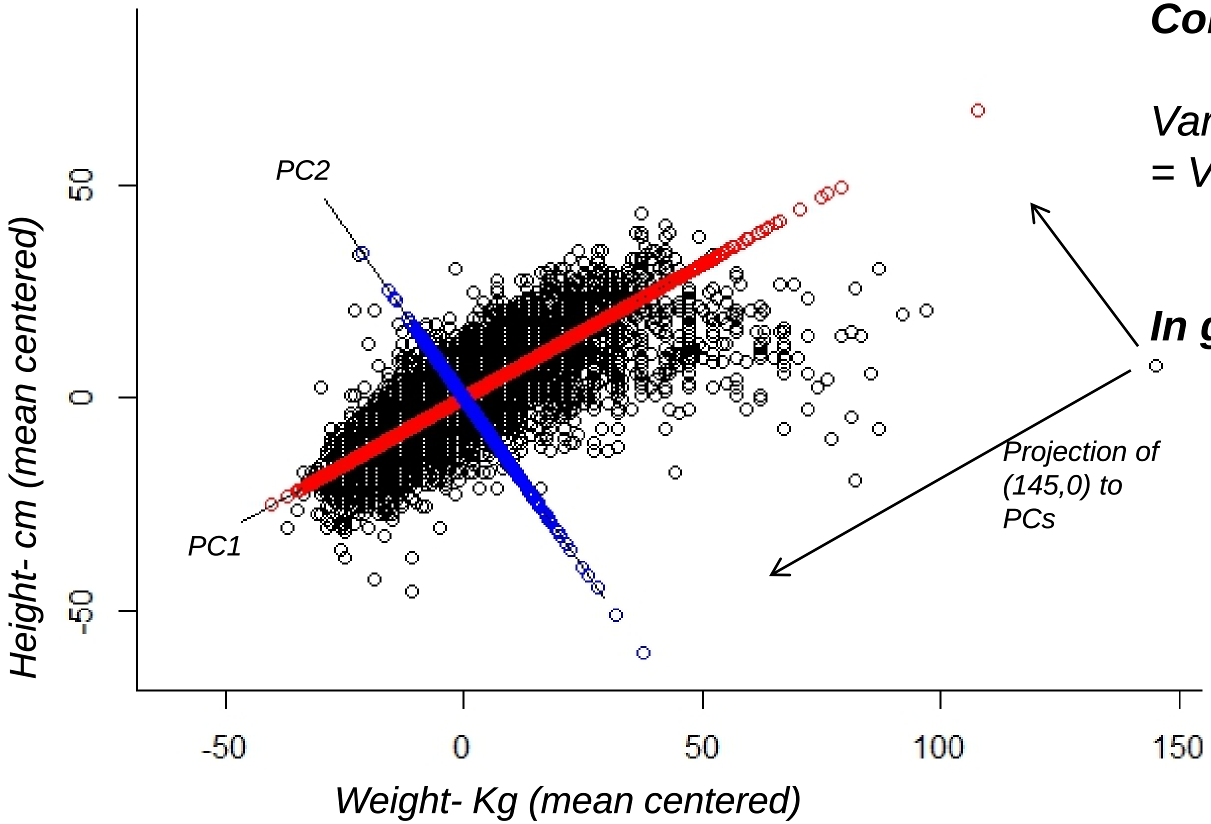
New axes (i.e., new features or latent factors) are combinations of old axis (i.e., old features or observed factors)

Rotate axis



Project all points to first axis
It keeps much of the variance

Note: factorization conserves and reorders variance



Total Variance Conserved:

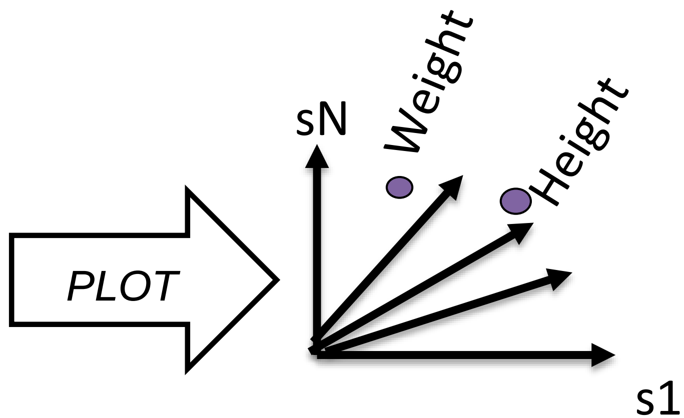
Var in Weight + Var in Height = Var in PC1 + Var in PC2

In general:

Var in PC1>
Var in PC2>
Var in PC3…

2D data transposed to 2 points in high dimensional space

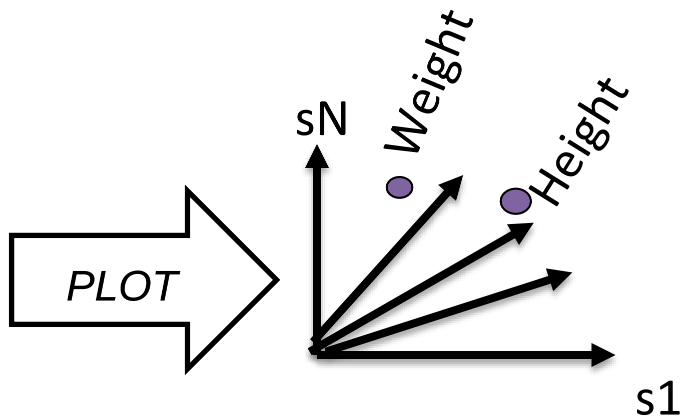
	s1	s2	s3	…
Height	179	175	180	…
Weight	50	66	74	…



We can also work in the transposed space (aka 'output' space or 'column' space)

2D data transposed to 2 points in high dimensional space

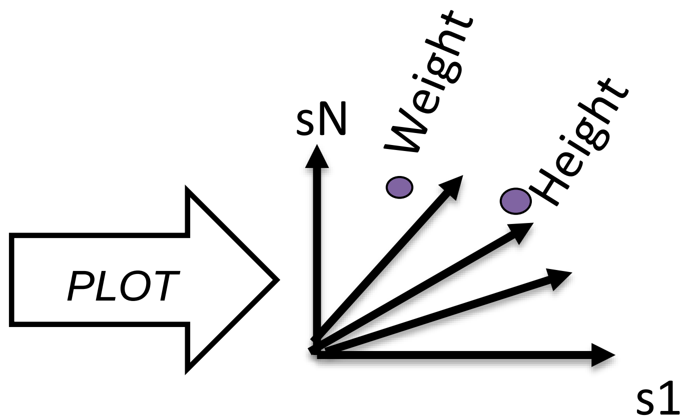
	s1	s2	s3	…
Height	179	175	180	…
Weight	50	66	74	…



How do we find factors?

2D data transposed to 2 points in high dimensional space

	s1	s2	s3	…
Height	179	175	180	…
Weight	50	66	74	…



How do we find factors?

Same process as before, but now factors are combinations of the s1…sN axes

- Best Known Factorization Algorithms:
    - SVD (singular value decomposition)
    - PCA (principle component analysis)

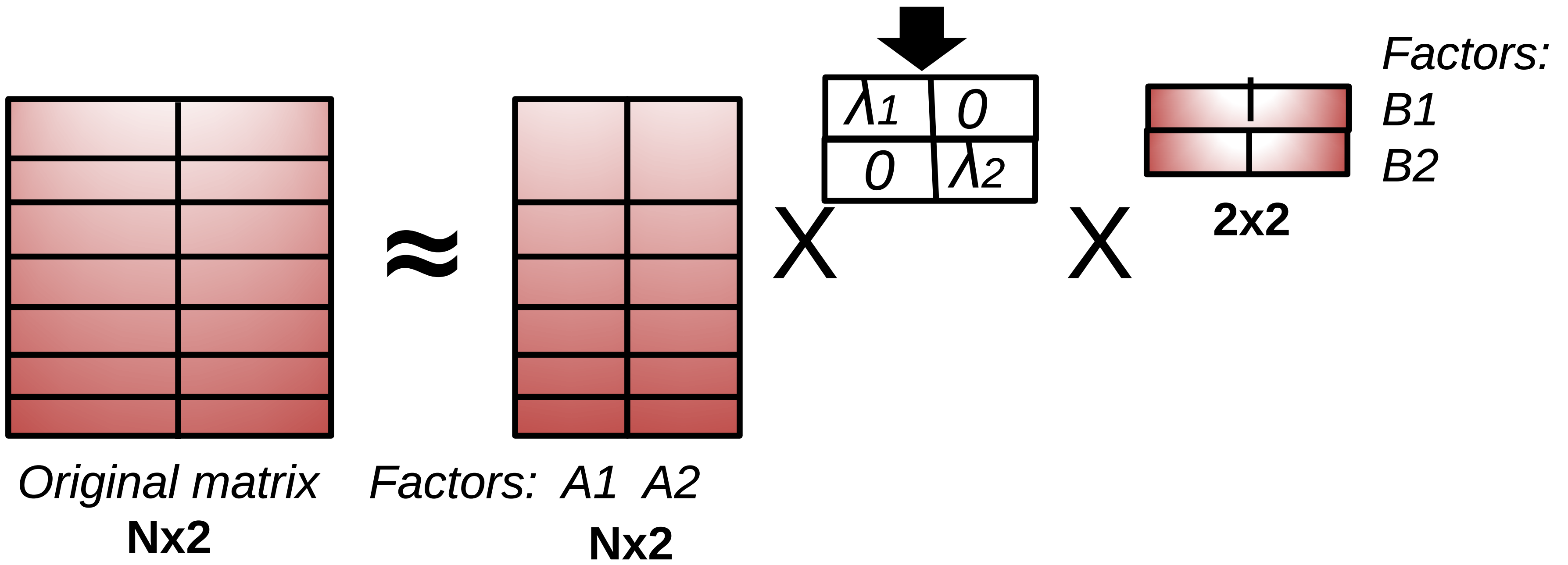
- Best Known Factorization Algorithms:

    SVD (singular value decomposition)

    PCA (principle component analysis)

SVD more generally works on non square matrices

SVD produces factors (ie column vectors) and 'singular' values that are multiplied to recreate original matrix



- More generally:

Factorization Algorithms may vary depending on criterion for how factors ‘align’ with data.

- More generally:

Factorization Algorithms may vary depending on criterion for how factors ‘align’ with data.

- Number of factors to use depends on tradeoff of good approximation vs good dimensional reduction

Can use cross validation or heuristics to choose.

Summary: Principle Components

- **Can choose *k* heuristically as approximation improves, or choose *k* so that high percent (ie 80-95%) of data variance accounted for**
- **aka Singular Value Decomposition**

  PCA on square matrices only

  SVD gives same vectors on square matrices
- **Works for numeric data only**
- **For higher dimensional data, use factors to visualize 2 factors at a time**

Pause

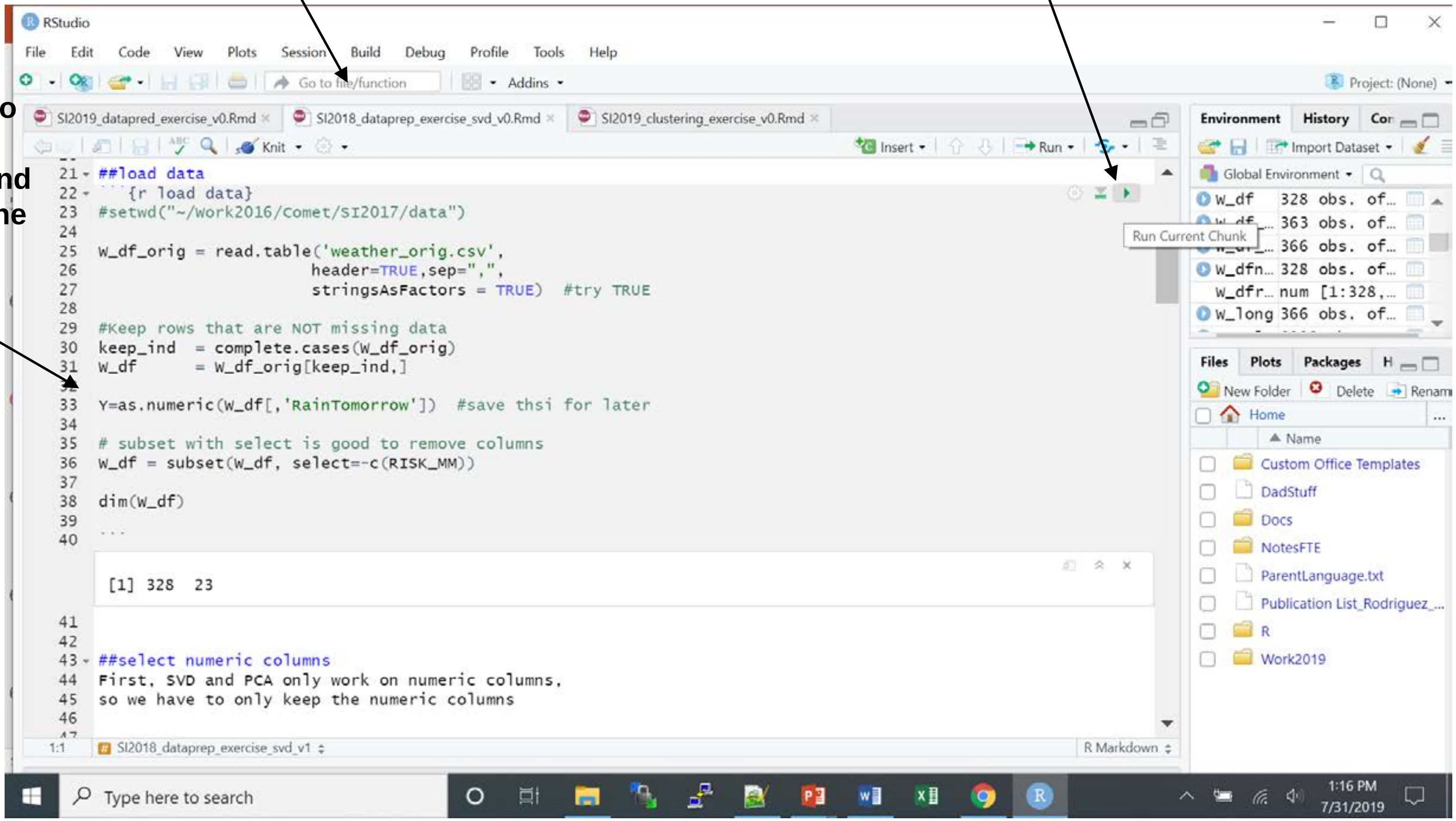
SVD Exercise

- Overview
  - Run on numeric fields of weather data
  - Run SVD and select smaller number of dimensions
  - Run linear model with original and reduced data

Load SVD R mark-down exercise

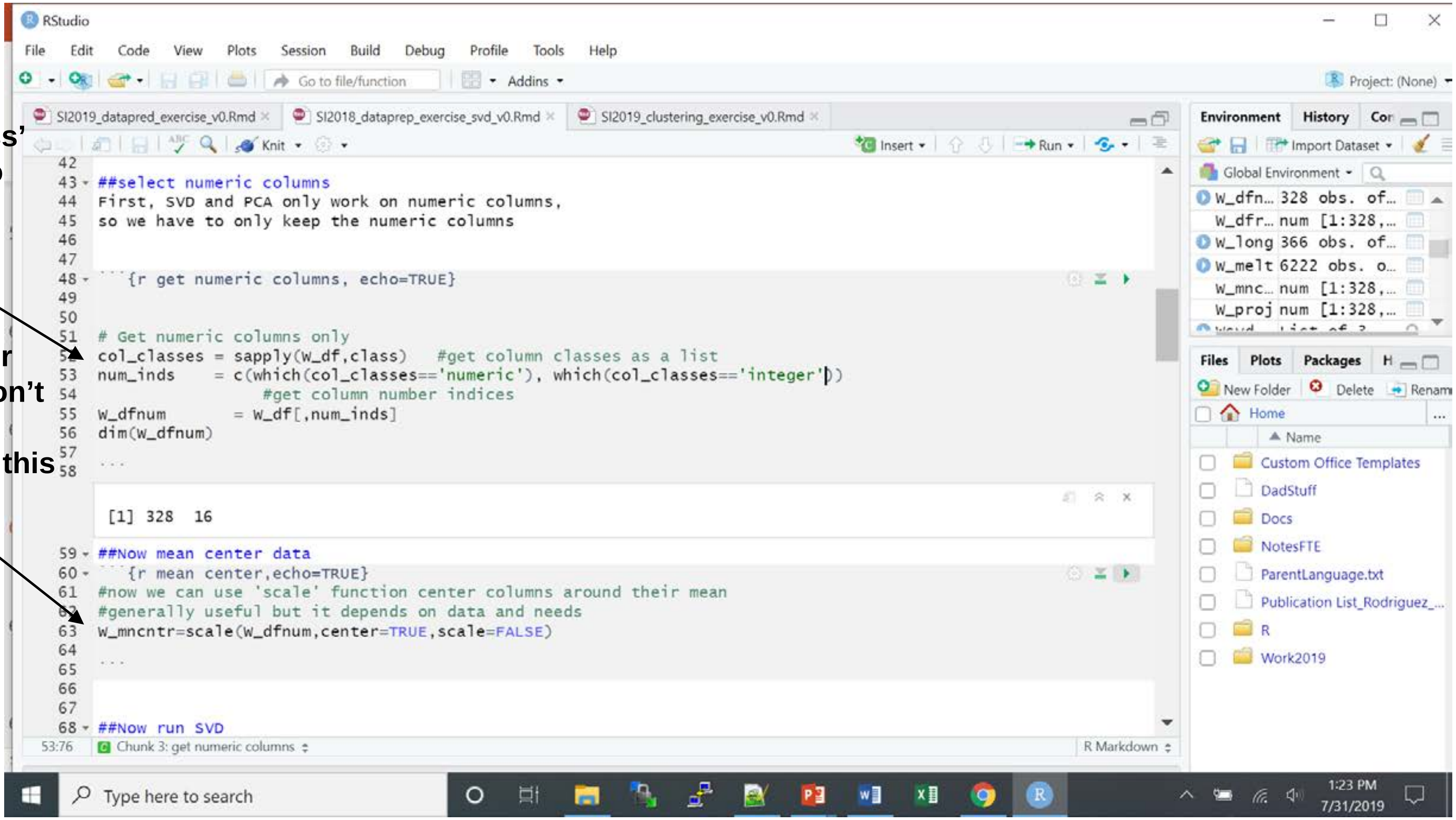
Find "load data" chunk. Run chunks above and current chunk

Take complete rows (ie no missing values), and take out the target variable



Take numeric variables - apply 'class' function to each column

Mean center data, but don't scale the variance in this case

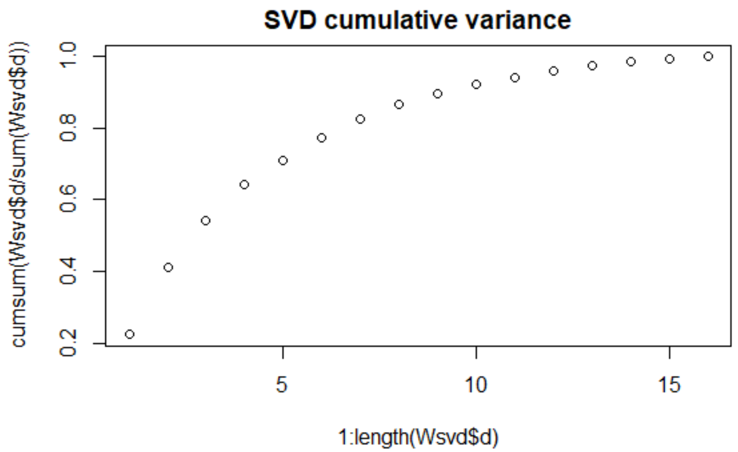


Later, we'll compare SVD components with Clustering

```
#W_num is only numeric or integer fields of Weather data
> Wsvd=svd(W_num)

> str(Wsvd)
List of 3
 $ d: num [1:9] 27442.7 231.2 96.4 68.2 44.5 ...
 $ u: num [1:363, 1:9] -0.0524 -0.0521 -0.052 -0.0519 -0.0525 ...
 $ v: num [1:9, 1:9] -0.005042 -0.014276 -0.000969 -0.00314 -0.005491 ...
```

Exercise highlights

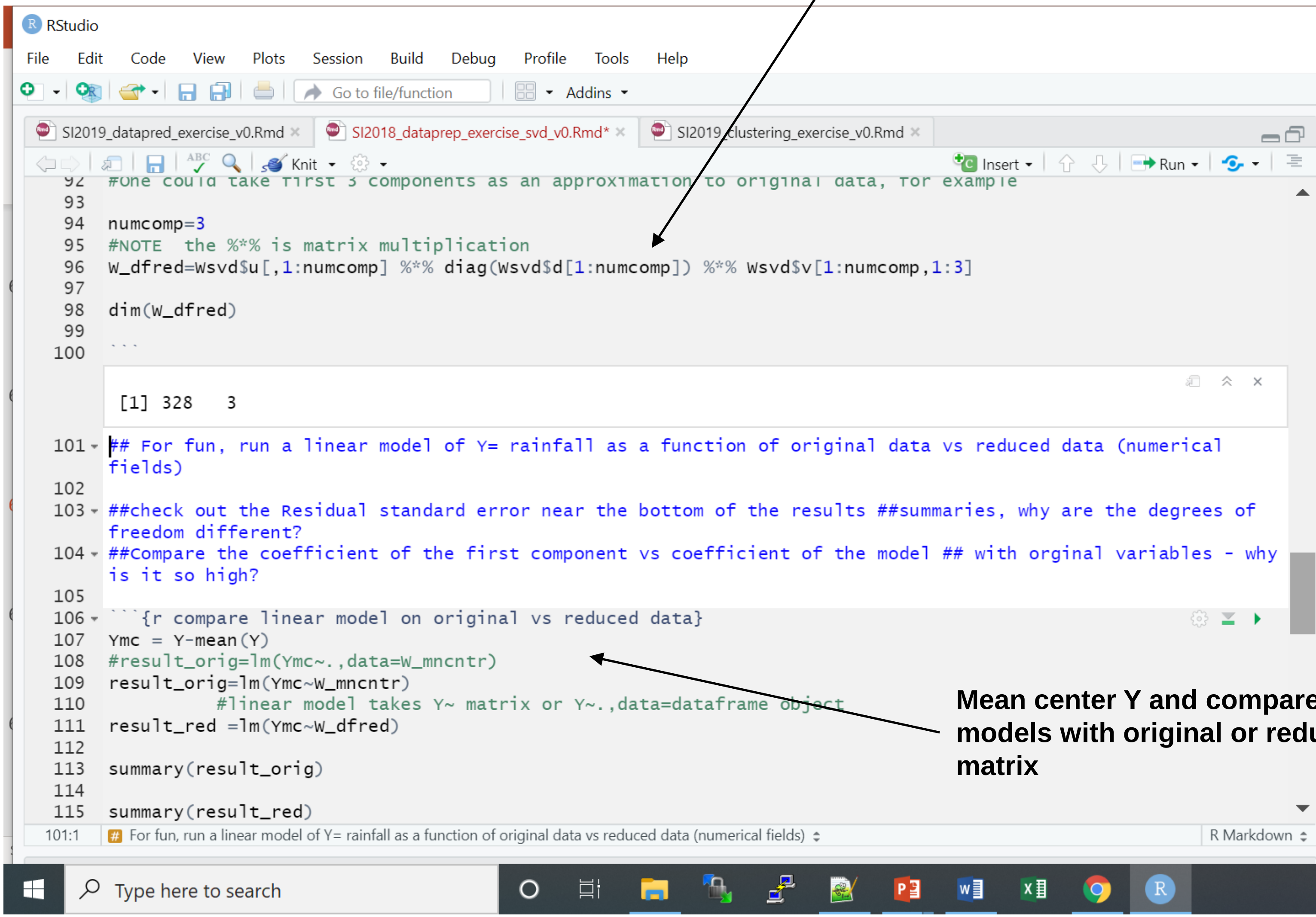


Compare Linear Model results, using Y = raintomorrow:

look for residual standard error values and degree of freedom,

look at coefficient estimates

Take 3 SVD components, rebuild a Nx3 reduced dimension data matrix



Mean center Y and compare linear models with original or reduced data matrix

```
Call:
lm(formula = Ymc ~ W_mncntr)
Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)         -1.124e-15  1.641e-02   0.000 1.000000
W_mncntrMinTemp      -1.368e-02  1.013e-02  -1.350 0.177844
W_mncntrMaxTemp       1.035e-02  2.010e-02   0.515 0.607120
W_mncntrRainfall      4.269e-03  4.471e-03   0.955 0.340442
W_mncntrEvaporation   2.690e-02  1.010e-02   2.663 0.008137 **
W_mncntrSunshine     -3.446e-02  9.898e-03  -3.482 0.000570 ***
W_mncntrPressure9am   6.569e-02  1.325e-02   4.960 1.16e-06 ***
W_mncntrPressure3pm  -8.047e-02  1.337e-02  -6.021 4.89e-09 ***

Residual standard error: 0.2971 on 311 degrees of freedom
```

```
Call:
lm(formula = Ymc ~ W_dfred)
Coefficients: (13 not defined because of singularities)
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -4.874e-16  1.808e-02   0.000 1.000000
W_dfred1     4.519e+00  1.242e+00   3.638 0.000320 ***
W_dfred2     4.650e+00  1.307e+00   3.559 0.000429 ***
W_dfred3     1.580e+00  4.357e-01   3.627 0.000333 ***
W_dfred4           NA         NA      NA       NA
W_dfred5           NA         NA      NA       NA

Residual standard error: 0.3274 on 324 degrees of freedom
```

- **end**