Inspire…Educate…Transform.

Statistics and Probability in Decision Modeling

Principal Components Analysis (PCA), Regularization (Ridge, LASSO and Elastic Nets Regression), Time Series Forecasting (Part 1)

Dr. Sridhar Pappu
Executive VP – Academics, INSOFE

January 20, 2018

Slides on PCA and Regularization courtesy Dr. Anand Jayaraman


FLASHBACK
I REMEMBER IT LIKE IT WAS YESTERDAY...
CMCOTT. 04/13/12 #138

VIF, GVIF $\left(\frac{1}{2*df}\right)$

Predicting Coronary Heart Disease Case

```
Call:
glm(formula = TenYearCHD ~ ., family = binomial, data = train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.9392  -0.5998  -0.4211  -0.2771   2.8632

Coefficients:
                 Estimate Std. Error z value Pr(>|z|)
(Intercept)     -8.360272   0.864696  -9.668  < 2e-16 ***
male             0.524080   0.130836   4.006 6.19e-05 ***
age              0.065429   0.008049   8.129 4.34e-16 ***
education       -0.041105   0.059185  -0.695 0.487366
currentSmoker    0.120498   0.187629   0.642 0.520735
cigsPerDay       0.016471   0.007488   2.200 0.027825 *
BPMeds           0.169118   0.282140   0.599 0.548898
prevalentStroke  1.156666   0.560179   2.065 0.038940 *
prevalentHyp     0.307077   0.166034   1.849 0.064389 .
diabetes        -0.319937   0.392574  -0.815 0.415087
totChol          0.003799   0.001330   2.856 0.004290 **
sysBP            0.011144   0.004446   2.507 0.012188 *
diaBP           -0.001861   0.007760  -0.240 0.810517
BMI              0.008812   0.015662   0.563 0.573702
heartRate       -0.007273   0.005131  -1.418 0.156296
glucose          0.009227   0.002752   3.353 0.000798 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2176.6  on 2565  degrees of freedom
Residual deviance: 1919.9  on 2550  degrees of freedom
  (402 observations deleted due to missingness)
AIC: 1951.9
```

- Education as **numeric** variable

```
> framingham = read.csv("framingham.csv")
> str(framingham)
'data.frame':   4240 obs. of  16 variables:
 $ male           : int  1 0 1 0 0 0 0 0 1 1 ...
 $ age            : int  39 46 48 61 46 43 63 45 52 43 ...
 $ education      : int  4 2 1 3 3 2 1 2 1 1 ...
```

- car package gives the following VIF figures

```
> car::vif(framinghamLog)
     male       age     education
 1.247670  1.278996  1.057810
```

CSE 7302c

VIF, GVIF, GVIF $\left(\frac{1}{2*df}\right)$

Predicting Coronary Heart Disease Case

```
Call:
glm(formula = TenYearCHD ~ ., family = binomial, data = train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.9315  -0.5948  -0.4196  -0.2733   2.8925

Coefficients:
                 Estimate Std. Error z value Pr(>|z|)
(Intercept)     -8.286327   0.856000  -9.680  < 2e-16 ***
male             0.506116   0.131991   3.834 0.000126 ***
age              0.064020   0.008135   7.869 3.56e-15 ***
education2      -0.210890   0.148310  -1.422 0.155038
education3      -0.120464   0.174595  -0.690 0.490220
education4      -0.082216   0.199469  -0.412 0.680212
currentSmoker    0.125147   0.187693   0.667 0.504921
cigsPerDay       0.016589   0.007482   2.217 0.026611 *
BPMeds           0.177341   0.282452   0.628 0.530094
prevalentStroke  1.158996   0.563826   2.056 0.039822 *
prevalentHyp     0.308709   0.166304   1.856 0.063412 .
diabetes        -0.318608   0.393231  -0.810 0.417807
totChol          0.003860   0.001334   2.894 0.003801 **
sysBP            0.011195   0.004451   2.515 0.011893 *
diaBP           -0.001726   0.007766  -0.222 0.824120
BMI              0.007535   0.015680   0.481 0.630828
heartRate       -0.007132   0.005136  -1.389 0.164946
glucose          0.009221   0.002748   3.356 0.000791 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2176.6  on 2565  degrees of freedom
Residual deviance: 1918.2  on 2548  degrees of freedom
  (402 observations deleted due to missingness)
AIC: 1954.2

Number of Fisher Scoring iterations: 5
```

- Education as **categorical** variable

```
> framingham$education = factor(framingham$education)
> str(framingham)
'data.frame':   4240 obs. of  16 variables:
 $ male           : int  1 0 1 0 0 0 0 0 1 1 ...
 $ age            : int  39 46 48 61 46 43 63 45 52 43 ...
 $ education      : Factor w/ 4 levels "1","2","3","4": 4 2
1 3 3 2 1 2 1 1 ...
```

- car package gives the following (G)VIF figures

```
> car::vif(framinghamLog)
                    GVIF Df GVIF^(1/(2*Df))
male            1.268872  1        1.126442
age             1.302896  1        1.141445
education       1.121533  3        1.019300
```

- 👍 Use the **square** of the GVIF $\left(\frac{1}{2*df}\right)$ value and apply the VIF rule of thumb

CSE 7302c

Present Day

Dealing with a large number of dimensions

DIMENSIONALITY REDUCTION

CSE 7302c

Understanding the concept

PRINCIPAL COMPONENTS ANALYSIS (PCA)

CSE 7302c

PCA

- Linear transformation technique used for dimensionality reduction
- Converts a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables (principal components)
- Explains a large part of the variation in data using a small number of principal components

CSE 7302c

PCA Applications

- Commonly used in stock market predictions, analysis of gene expression (process of using information from a gene to synthesize proteins, etc.) data, etc.
- Many applications in taxonomy, finance, biology, neuroscience, pharmacy, agriculture, ecology, health, architecture, etc.

CSE 7302c

Algorithmic Trading

- Analyze patterns in economic data and past stock prices to forecast future price movements
- Target Variable: Stock Price Direction (Up or Down), e.g., Bank Nifty

CSE 7302c

Algorithmic Trading

- Predictor Variables: 352 different variables quantifying economic data and price movements
  - Past data of Bank Nifty and quantifying patterns therein
  - Quantifying patterns in the US market on the previous data
  - Patterns in the variations in Dollar-Rupee exchange rate
  - Interest rate changes in India
  - Interest rate changes in US
- The patterns, not values, over the past week, month, etc. are studied after downloading all this data

Algorithmic Trading

- Many of the predictors may be irrelevant
- Many of the predictors may be similar (correlated)
- Correlation could be complex, e.g., 5 variables put together in some manner may be correlated to another variable

- Approximately 14 hours to do the simulation

CSE 7302c

Algorithmic Trading

- Is there a way to figure out which variables are important and which are not essential?

CSE 7302c

Which is the relevant variable?

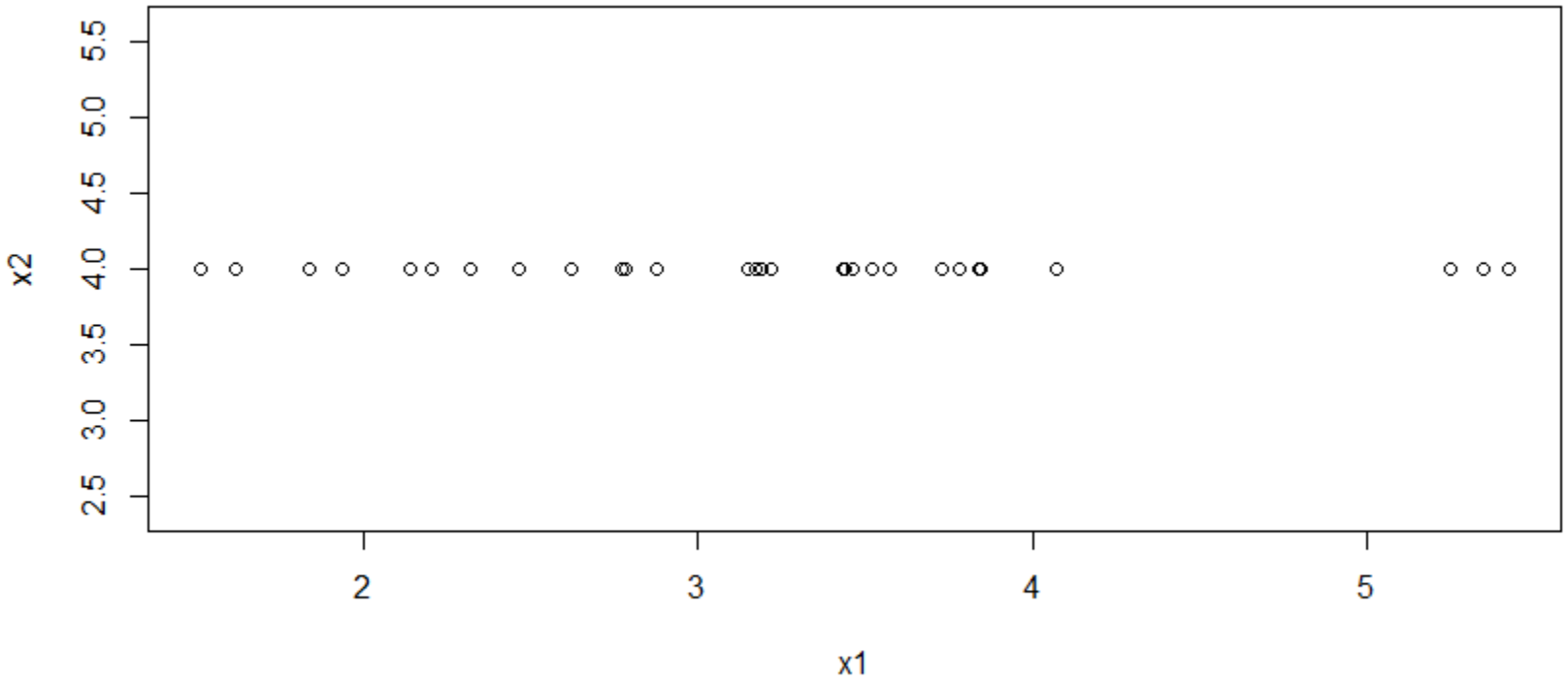
y	x1	x2
21	2.62	4
21	2.875	4
22.8	2.32	4
21.4	3.215	4
18.7	3.44	4
18.1	3.46	4
14.3	3.57	4
24.4	3.19	4
22.8	3.15	4
19.2	3.44	4
17.8	3.44	4
16.4	4.07	4
17.3	3.73	4
15.2	3.78	4
10.4	5.25	4
10.4	5.424	4
14.7	5.345	4
32.4	2.2	4

There is no variation in x2. All the variation is only along x1. So *y* cannot be explained by x2.

sd(x1) = 0.98
sd(x2) = 0

We didn't even need to look at *y* to make the conclusion!

CSE 7302c



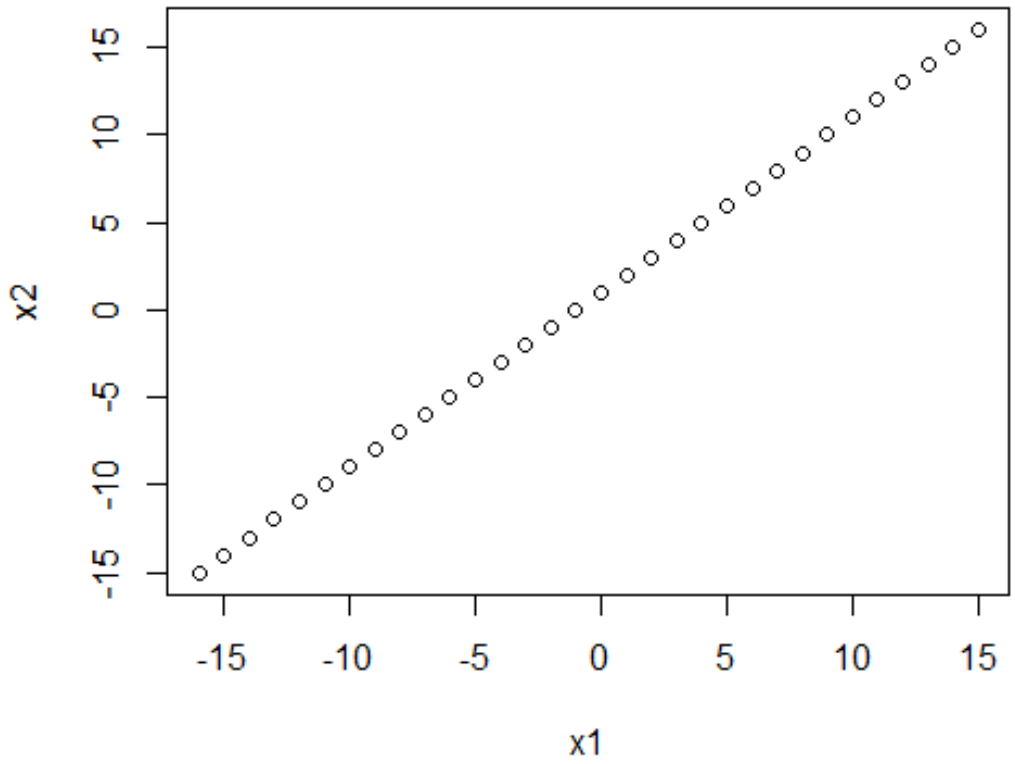
All the variation in the **explanatory variables** is in x1 direction only.

This direction is called the dominant **Principal Component** of the explanatory variables.

The x2 direction is redundant, since there is no variation along that axis.

How many relevant features are here?

y	x1	x2
21	-16	-15
21	-15	-14
22.8	-14	-13
21.4	-13	-12
18.7	-12	-11
18.1	-11	-10
14.3	-10	-9
24.4	-9	-8
22.8	-8	-7
19.2	-7	-6
17.8	-6	-5
16.4	-5	-4
17.3	-4	-3
15.2	-3	-2
10.4	-2	-1
10.4	-1	0
14.7	0	1
32.4	1	2
30.4	2	3
33.9	3	4



This seems to have variation in both x1 and x2. However, the data points in x1 and x2 are perfectly correlated.

CSE 7302c

How many relevant features are here?

y	x1	x2
21	-16	-15
21	-15	-14
22.8	-14	-13
21.4	-13	-12
18.7	-12	-11
18.1	-11	-10
14.3	-10	-9
24.4	-9	-8
22.8	-8	-7
19.2	-7	-6
17.8	-6	-5
16.4	-5	-4
17.3	-4	-3
15.2	-3	-2
10.4	-2	-1
10.4	-1	0
14.7	0	1
32.4	1	2
30.4	2	3
33.9	3	4

If we create new variables

X1 = x2 + x1

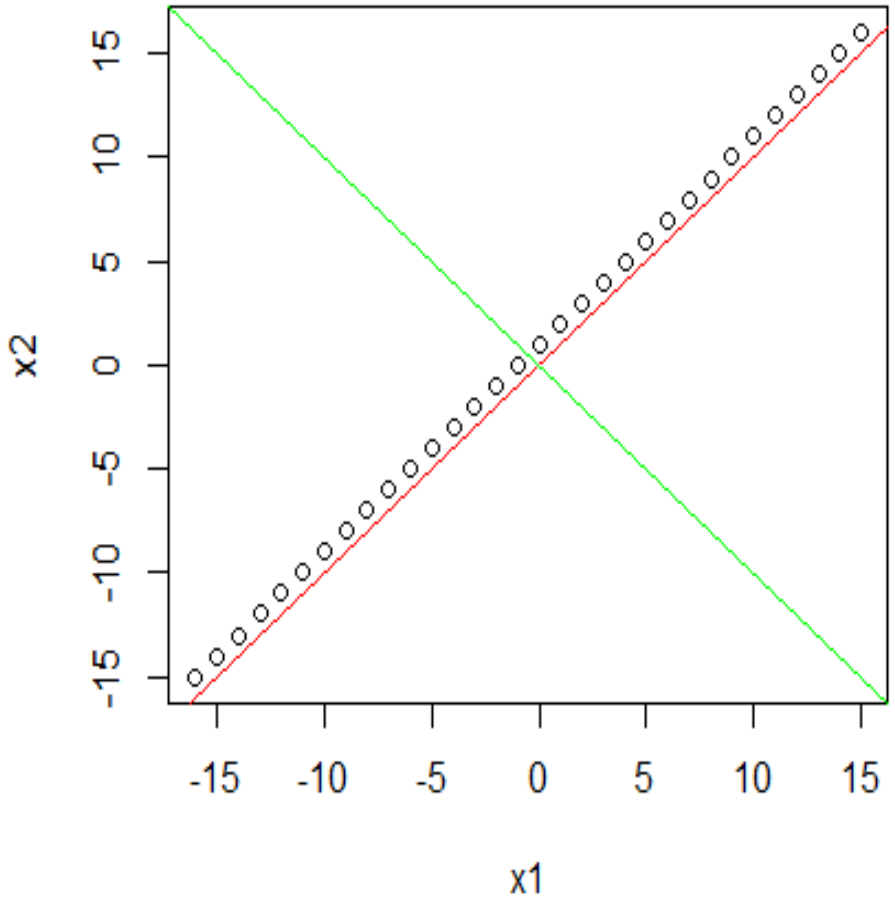
X2 = x2 – x1

In terms of new variables X1 and X2 it is clear that there is only one true relevant feature.

y	X1	X2
21	-31	1
21	-29	1
22.8	-27	1
21.4	-25	1
18.7	-23	1
18.1	-21	1
14.3	-19	1
24.4	-17	1
22.8	-15	1
19.2	-13	1
17.8	-11	1
16.4	-9	1
17.3	-7	1
15.2	-5	1
10.4	-3	1
10.4	-1	1
14.7	1	1
32.4	3	1
30.4	5	1

CSE 7302c

Transformed Variables: Interpretation



CSE 7302c

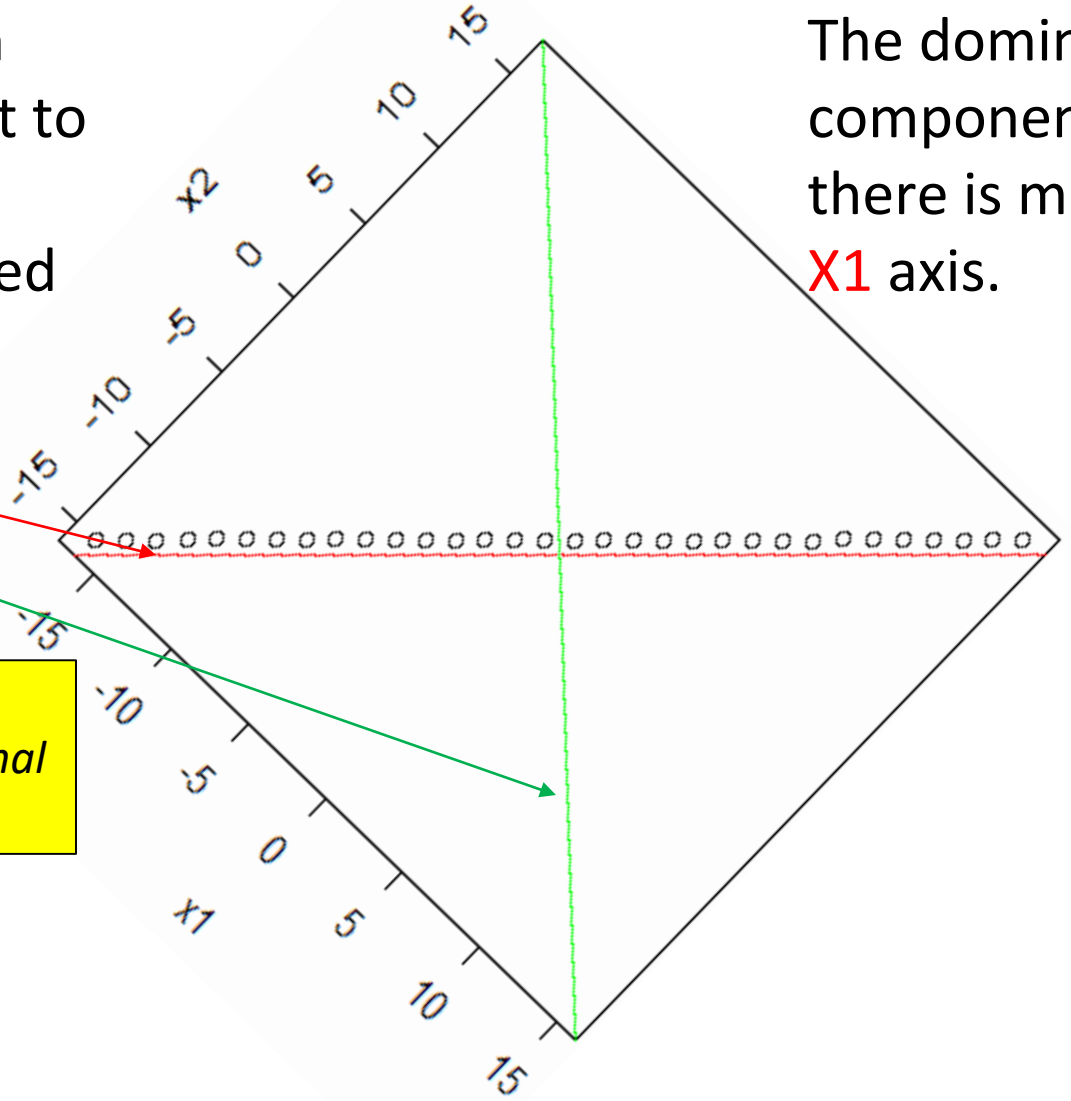
Transformed Variables: Interpretation

The transformation we did is equivalent to viewing the data points from a rotated coordinate system.

Red = X1 axis

Green = X2 axis

Note both X1 and X2 are combinations of the original variables, x1 and x2.



The dominant Principal component (along which there is maximum variation) is X1 axis.

CSE 7302c

Standardization is a Prerequisite

y	x1	x2
21	-16	-15
21	-15	-14
22.8	-14	-13
21.4	-13	-12
18.7	-12	-11
18.1	-11	-10
14.3	-10	-9
24.4	-9	-8
22.8	-8	-7
19.2	-7	-6
17.8	-6	-5
16.4	-5	-4
17.3	-4	-3
15.2	-3	-2
10.4	-2	-1
10.4	-1	0
14.7	0	1
32.4	1	2
30.4	2	3
33.9	3	4

Suppose ***y*** is **miles per gallon**, **x1** is **weight of a car** and **x2** is **horsepower**. Does it make sense to transform variables (x1+x2 or x1-x2) with different scales and units?

No. Before running PCA, it is a must to standardize all variables.

y	X1	X2
21	-31	1
21	-29	1
22.8	-27	1
21.4	-25	1
18.7	-23	1
18.1	-21	1
14.3	-19	1
24.4	-17	1
22.8	-15	1
19.2	-13	1
17.8	-11	1
16.4	-9	1
17.3	-7	1
15.2	-5	1
10.4	-3	1
10.4	-1	1
14.7	1	1
32.4	3	1
30.4	5	1

Principal Components Analysis

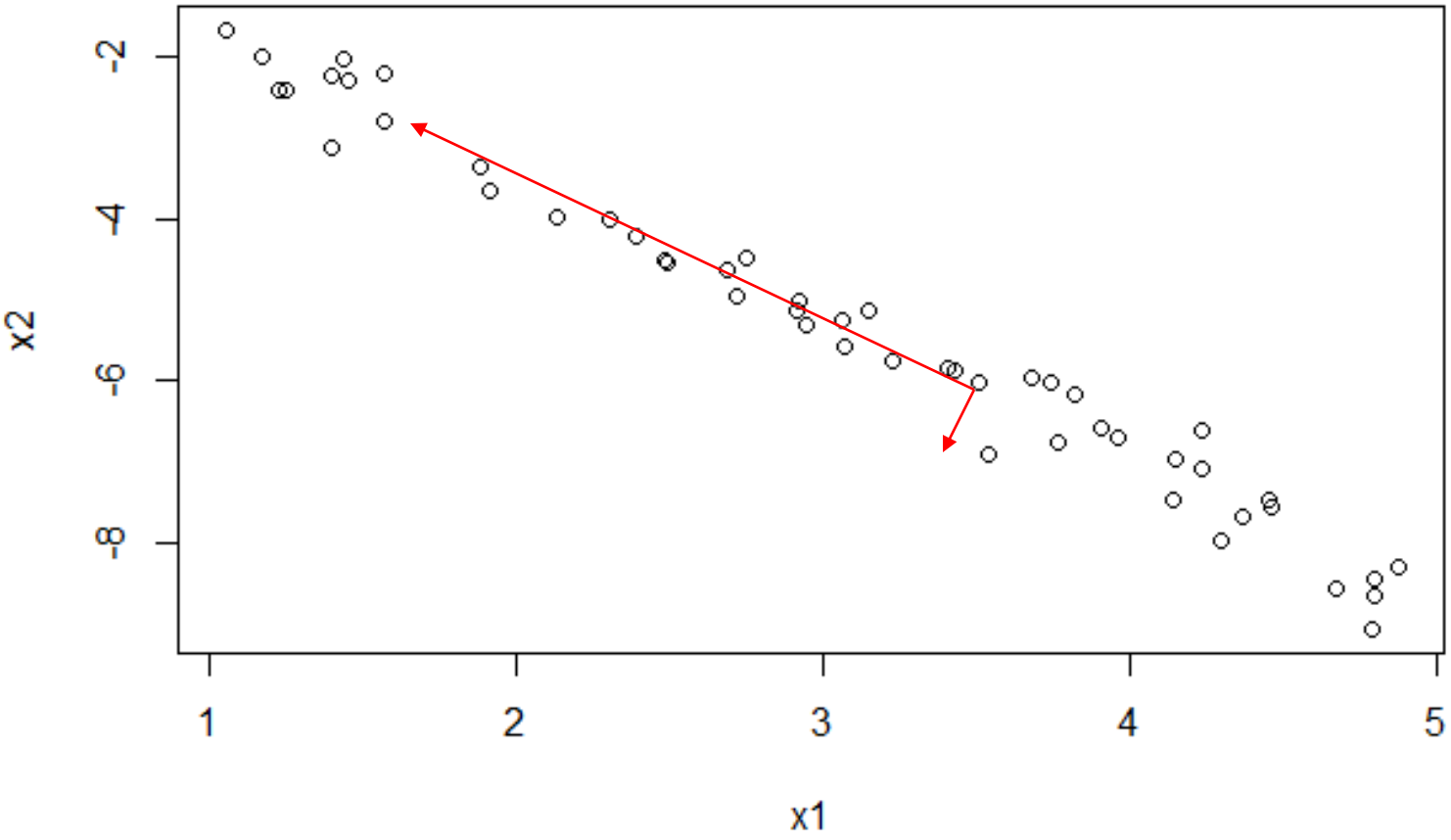
PCA is the method which allows you to identify the “directions” in which most of the variation in the data is present. Note we do not consider ***y*** at all in this analysis.

Equivalently, it can be thought of as a method to identify the “directions” along which there is least variation (or least useful info). Identifying this would allow us to drop these irrelevant directions from our regressions/model building.

CSE 7302c

Principal Component Directions

y	x1	x2
-42.4957832	4.797034	-8.651047
-2.0248387	1.220774	-2.428135
-1.9379560	1.437964	-2.037009
-13.0806296	2.719718	-4.960111
-35.1968731	4.302478	-7.967436
-31.7650916	4.139152	-7.471322
-22.4288444	3.681965	-5.951819
-21.4937406	3.512183	-6.016649
-0.9004686	1.050161	-1.683525
-3.3677489	1.398850	-3.111843
-26.3779407	3.907309	-6.573550
-34.6572233	4.460774	-7.552211
-24.1480305	3.819653	-6.157824
-23.1244874	3.744704	-6.023781
-2.3255245	1.452426	-2.284816
-29.7199029	4.153706	-6.950889
-2.1284295	1.393772	-2.233364
-16.0110878	3.060388	-5.258223

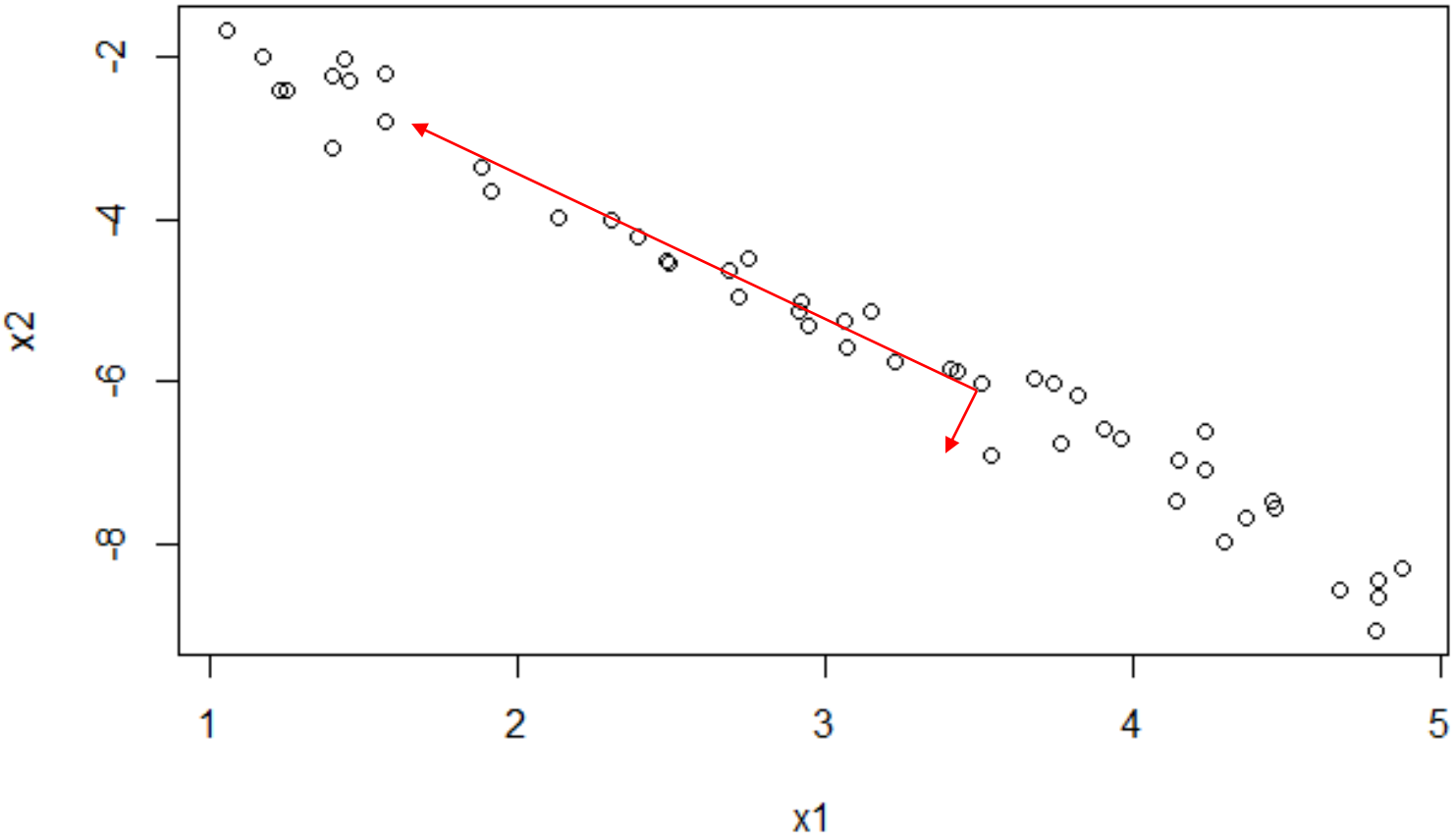


Can you identify the Principal Components?

CSE 7302c

Dimension Reduction

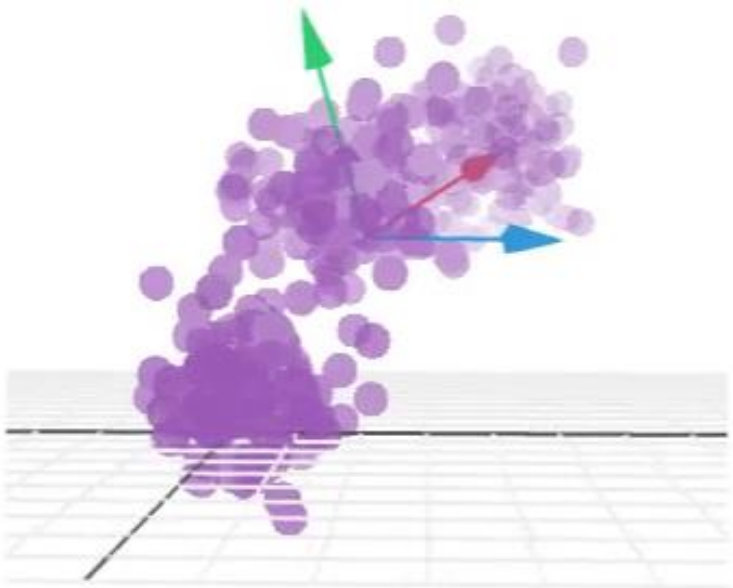
y	x1	x2
-42.4957832	4.797034	-8.651047
-2.0248387	1.220774	-2.428135
-1.9379560	1.437964	-2.037009
-13.0806296	2.719718	-4.960111
-35.1968731	4.302478	-7.967436
-31.7650916	4.139152	-7.471322
-22.4288444	3.681965	-5.951819
-21.4937406	3.512183	-6.016649
-0.9004686	1.050161	-1.683525
-3.3677489	1.398850	-3.111843
-26.3779407	3.907309	-6.573550
-34.6572233	4.460774	-7.552211
-24.1480305	3.819653	-6.157824
-23.1244874	3.744704	-6.023781
-2.3255245	1.452426	-2.284816
-29.7199029	4.153706	-6.950889
-2.1284295	1.393772	-2.233364
-16.0110878	3.060388	-5.258223

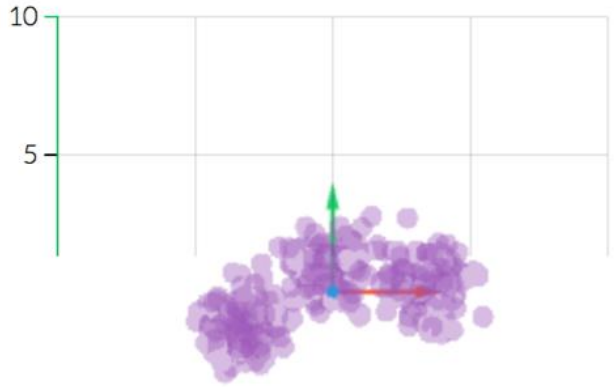


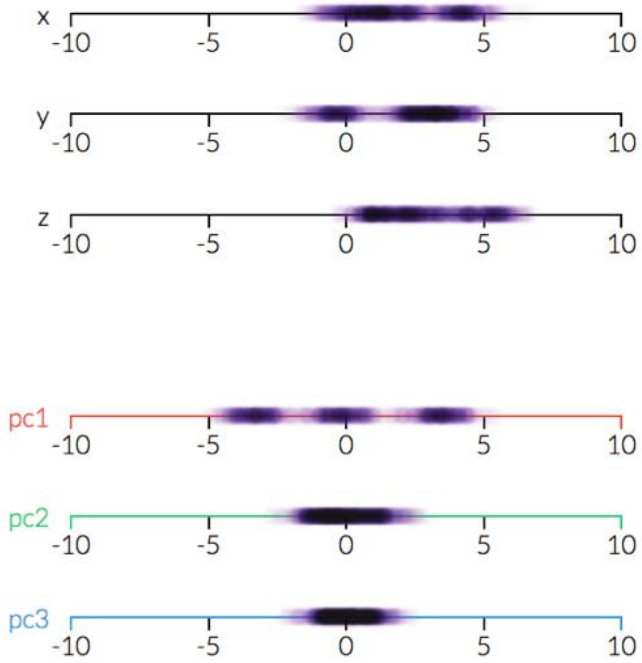
If we had to simplify and pick only one feature, it is easier to do so in the transformed variables. We can keep the dominant principal component and skip the non-dominant component.

CSE 7302c

An example in 3D







Source: http://setosa.io/ev/principal-component-analysis/
Last accessed: January 19, 2018

CSE 7302c

Principal Components Analysis – Summarizing the Concept

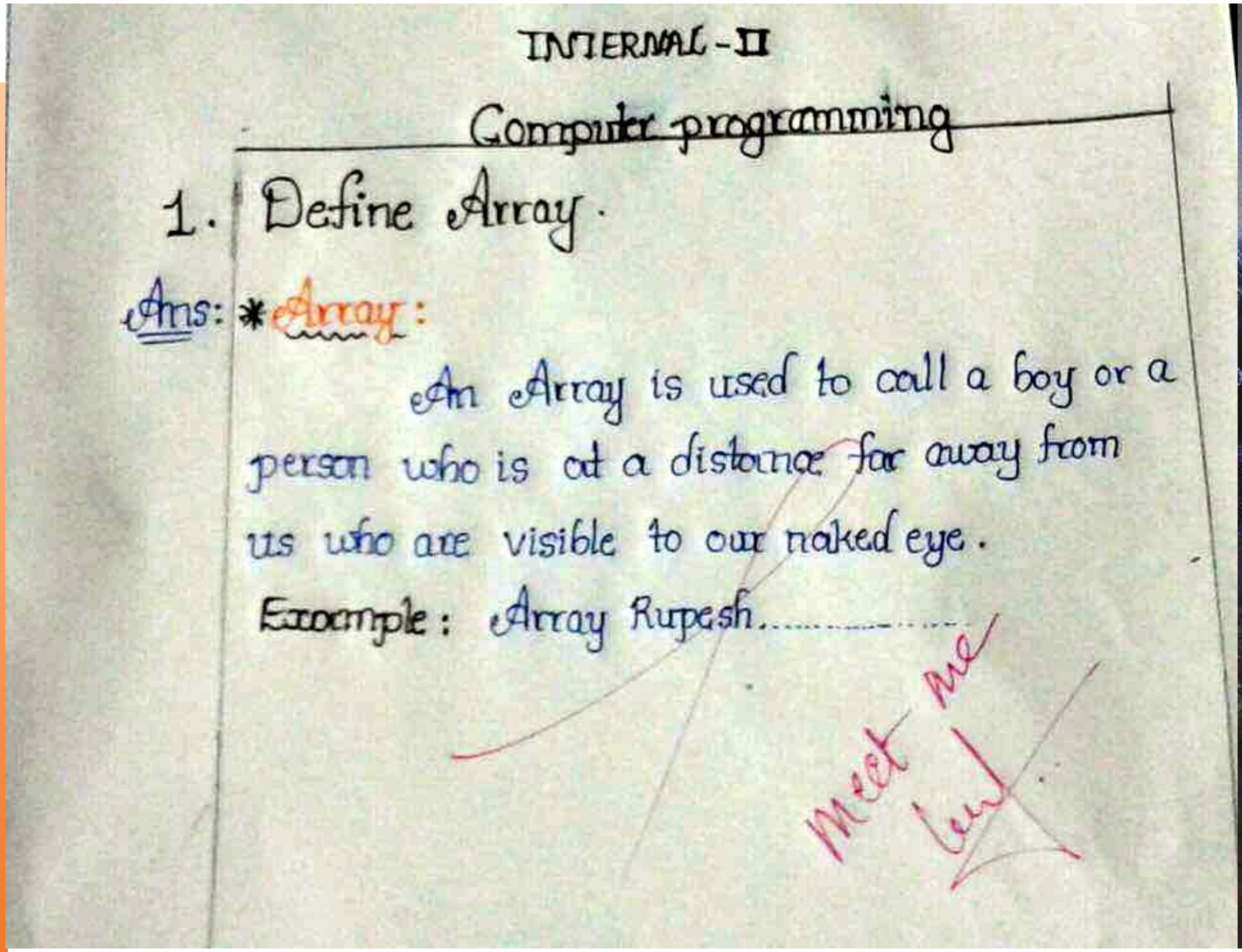
- PCA is a way for identifying the directions of largest to smallest variance
- It also tells how to transform the data into the new coordinate system
- Once we transform the data into the new coordinate system, we can choose to drop the variables which do not have much variance
- This gives a way to **reduce dimensions** and focus on ones with largest variance
- PCA is a **dimensionality reduction** technique, not a **feature selection** method, i.e., we don't select variables but we combine them to find fewer dimensions explaining most variance in the data

CSE 7302c

Array, Vector, Matrix…THE MATH

METHODOLOGY

Array! Matrix! And other Nightmares!



INTERNAL - II

Computer programming

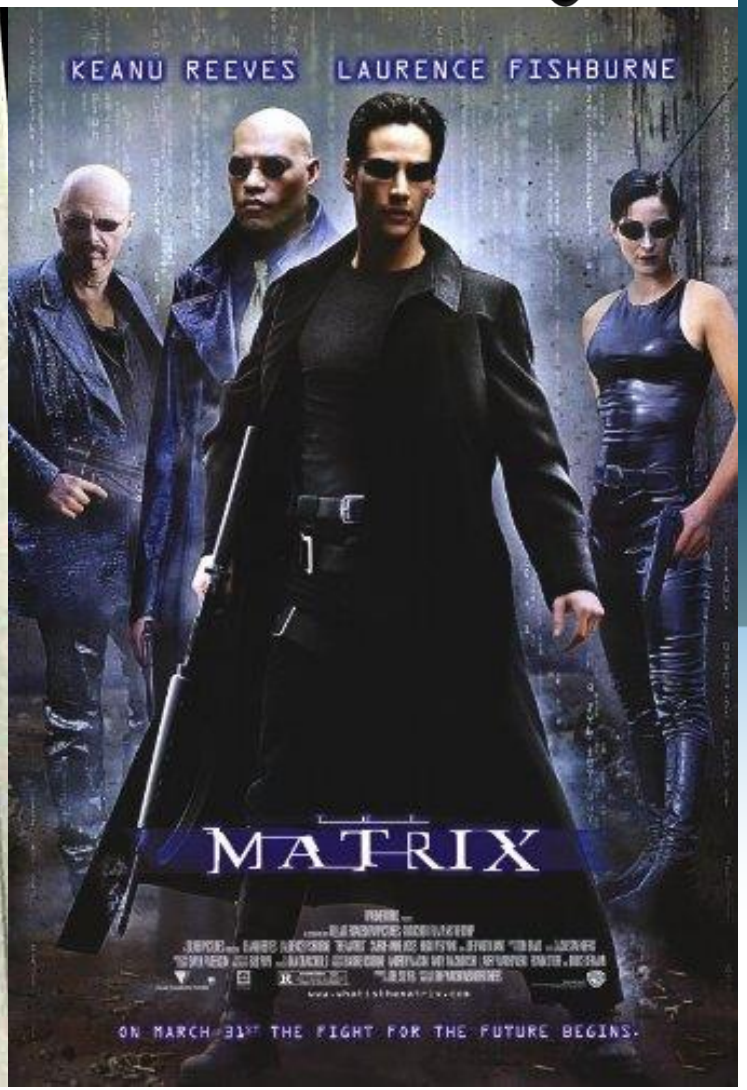
1. Define Array.

Ans: * Array :

An Array is used to call a boy or a person who is at a distance far away from us who are visible to our naked eye.

Example : Array Rupesh.....

meet me



Array! Matrix! They are SIMPLE!

What is an array?

Dimensions	Example	Terminology
1	0 1 2	Vector (1D Array)
2	0 1 2 3 4 5 6 7 8	Matrix (2D Array)
3	0 1 2 3 4 5 6 7 8	3D Array (3rd order Tensor)
N	... ...	ND Array

WineClass	Alcohol	MalicAcid	Ash	AlkalinityAsh
barolo	14.23	1.71	2.43	15.6
barolo	13.2	1.78	2.14	11.2
barolo	13.16	2.36	2.67	18.6
barolo	14.37	1.95	2.5	16.8
barolo	13.24	2.59	2.87	21
barolo	14.2	1.76	2.45	15.2
barolo	14.39	1.87	2.45	14.6
barolo	14.06	2.15	2.61	17.6
barolo	14.83	1.64	2.17	14
barolo	13.86	1.35	2.27	16
barolo	14.1	2.16	2.3	18
barolo	14.12	1.48	2.32	16.8
barolo	13.75	1.73	2.41	16
barolo	14.75	1.73	2.39	11.4
barolo	14.38	1.87	2.38	12

Source: http://www.java67.com/2014/08/what-is-array-data-structure-in-java.html
Last accessed: July 07, 2017

CSE 7302c

Regression in Matrix Form

Market Price ($1000) (y)	Area (sqft) (x1)	Age of House (years) (x2)
63	1605	35
65.1	2489	45
69.9	1553	20
76.8	2404	32
73.9	1884	25
77.9	1558	14
74.9	1748	8
78	3105	10
79	1682	28
83.4	2470	30
79.5	1820	2
83.9	2143	6
79.7	2121	14
84.5	2485	9
96	2300	19
109.5	2714	4
102.5	2463	5
121	3076	7
104.9	3048	3
128	3267	6
129	3069	10
117.9	4765	11
140	4540	8

	Coefficients	*Standard Error*	*t Stat*	*P-value*
Intercept	57.35074586	10.00715186	5.73097587	1.31298E-05
Area (sqft) (x1)	0.017718036	0.00314562	5.632605205	1.63535E-05
Age of House (years) (x2)	-0.666347946	0.227996703	-2.922620973	0.008417613

$$\hat{Y} = X\beta$$

$$\begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \hat{y}_3 \\ \hat{y}_4 \end{bmatrix} = \begin{bmatrix} 1 & 1605 & 35 \\ 1 & 2489 & 45 \\ 1 & 1553 & 20 \\ 1 & 2404 & 32 \end{bmatrix} \begin{bmatrix} 57.3507 \\ 0.0177 \\ -0.6663 \end{bmatrix}$$

$$Y = X\beta + \varepsilon$$

CSE 7302c

PCA Methodology

Start with analyzing the covariance matrix of the features.

x1	x2
4.797034	-8.651047
1.220774	-2.428135
1.437964	-2.037009
2.719718	-4.960111
4.302478	-7.967436
4.139152	-7.471322
3.681965	-5.951819
3.512183	-6.016649
1.050161	-1.683525
1.398850	-3.111843
3.907309	-6.573550
4.460774	-7.552211
3.819653	-6.157824
3.744704	-6.023781
1.452426	-2.284816
4.153706	-6.950889

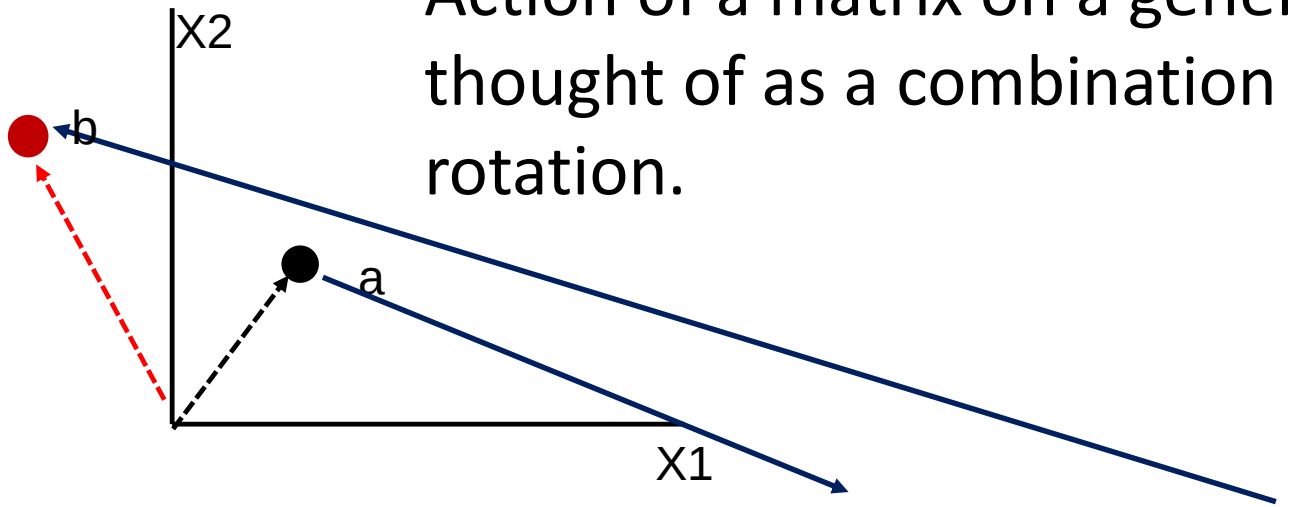
$$C = \begin{bmatrix} cov(x_1, x_1) & cov(x_1, x_2) \\ cov(x_2, x_1) & cov(x_2, x_2) \end{bmatrix}$$

```
> cov(d[,c("x1","x2")])
           x1         x2
x1  1.358045 -2.35219
x2 -2.352190  4.18192
```

The covariance matrix contains information about both correlations and the “special directions” of maximal variances.

CSE 7302c

Matrix as a Transformation on a Vector



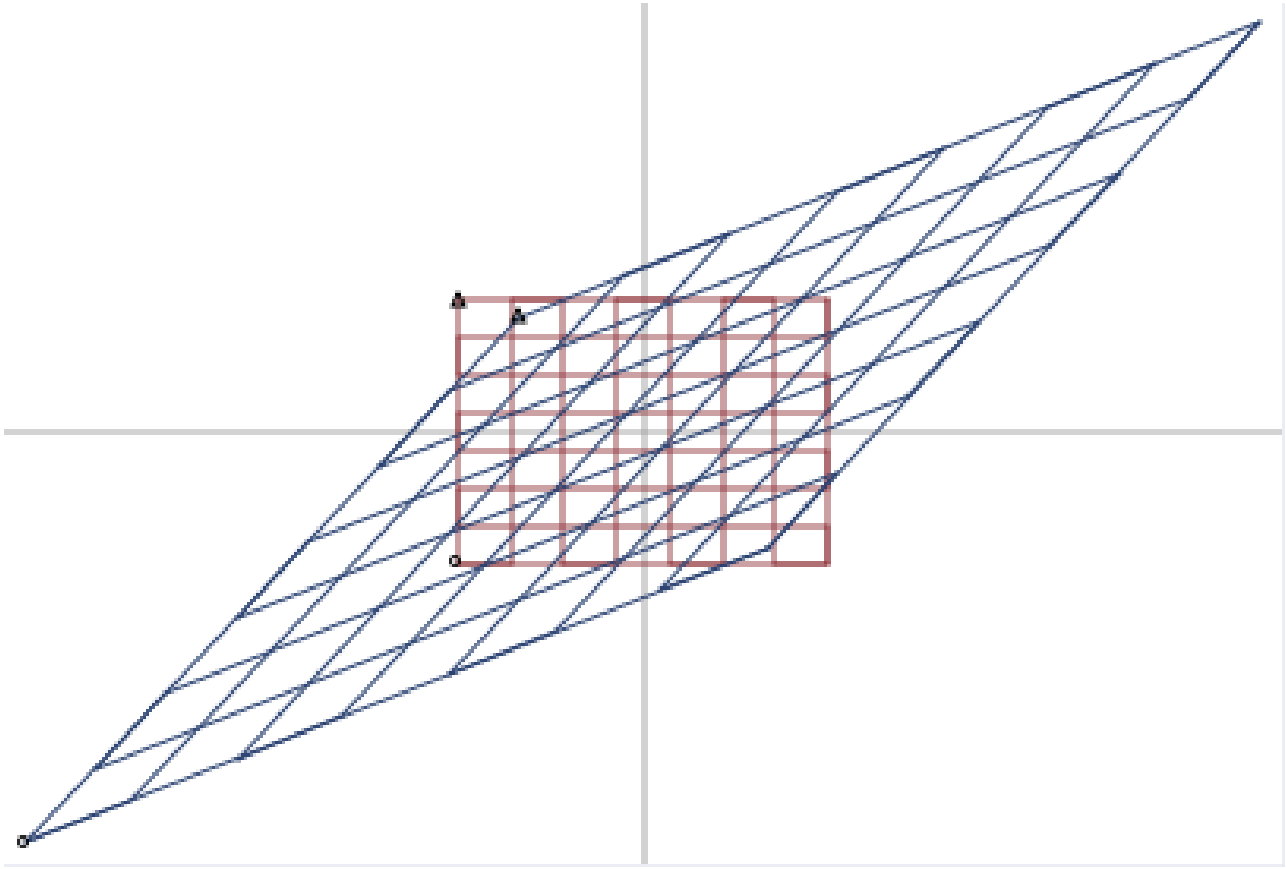
Action of a matrix on a general vector can be thought of as a combination of stretch and rotation.

$$\begin{bmatrix} 1.36 & -2.35 \\ -2.35 & 4.18 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} -0.99 \\ 1.83 \end{bmatrix}$$

```
> cov(d[,c("x1","x2")])
           x1       x2
x1  1.358045 -2.35219
x2 -2.352190  4.18192
```

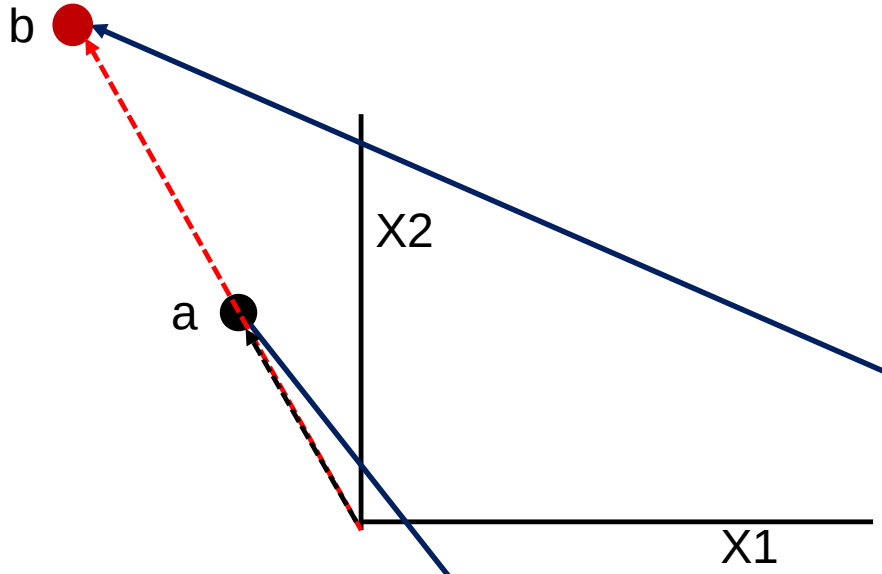
CSE 7302c

A Transformation Matrix that Stretches, Rotates and Skews



CSE 7302c

Matrix as a Transformation on a Vector



$$\begin{bmatrix} 1.36 & -2.35 \\ -2.35 & 4.18 \end{bmatrix} \begin{bmatrix} -0.49 \\ 0.87 \end{bmatrix} = \begin{bmatrix} -2.715 \\ 4.795 \end{bmatrix} = 5.51 \begin{bmatrix} -0.49 \\ 0.87 \end{bmatrix}$$

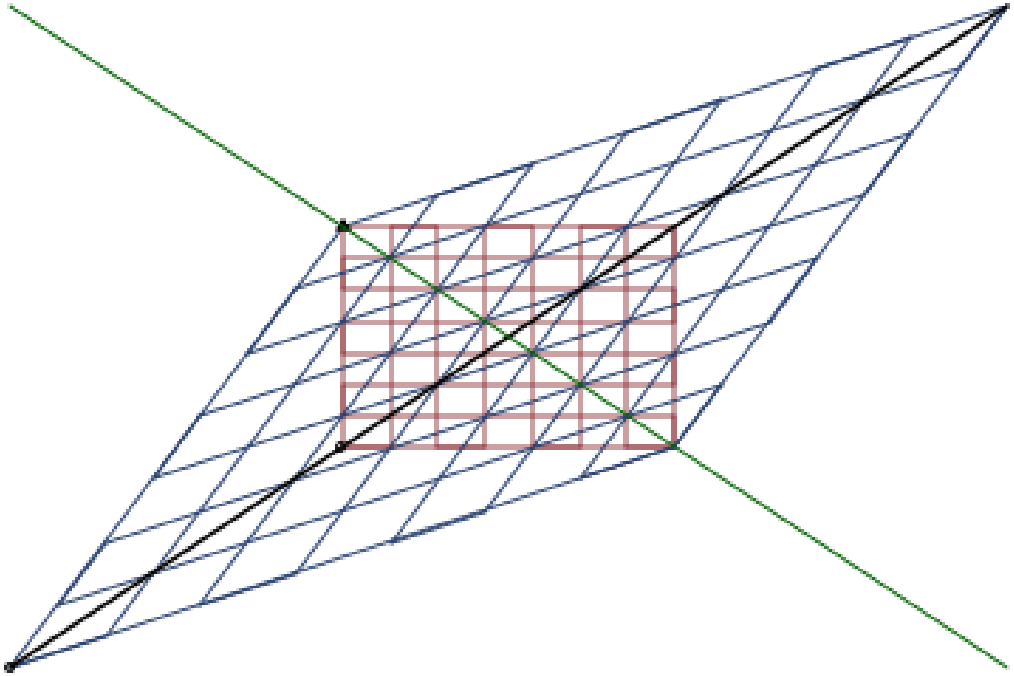
For a given matrix there are special directions along which its effect is only to stretch (without rotation). These special directions are called **Eigendirections** or **Eigenvectors**. The amount by which it stretches is called the **Eigenvalue**.

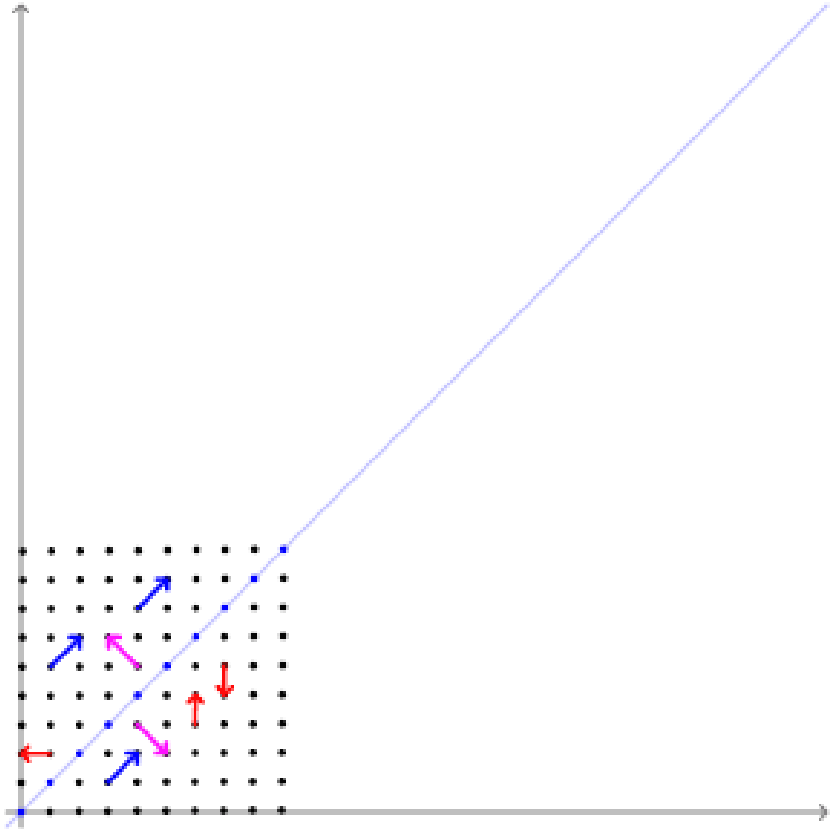
CSE 7302c

Eigenvectors

Eigen in German means "very own". For example, "mein eigenes auto" means "my very own car".

So, an eigenvector describes the relationship with itself and not with anything else.





Source: https://deeplearning4j.org/eigenvector
Last accessed: July 07, 2017

CSE 7302c

Eigenvectors Mathematics

- The eigenvectors and eigenvalues of matrix **A** are defined to be the nonzero **x** and λ values, respectively, that solve

$$\mathbf{Ax} = \lambda\mathbf{x}$$ **(A is just stretching)**

$$\begin{bmatrix} 1.36 & -2.35 \\ -2.35 & 4.18 \end{bmatrix} \begin{bmatrix} -0.49 \\ 0.87 \end{bmatrix} = 5.51 \begin{bmatrix} -0.49 \\ 0.87 \end{bmatrix}$$

- For a *n*-dim square matrix, there are at most *n* eigenvectors and *n* eigenvalues.

CSE 7302c

Eigenvectors and PCA

- Eigenvectors are the Principal Component directions
- Eigenvalues are the magnitude of stretch
- Eigenvalues represent the variance along those (new) directions.

CSE 7302c

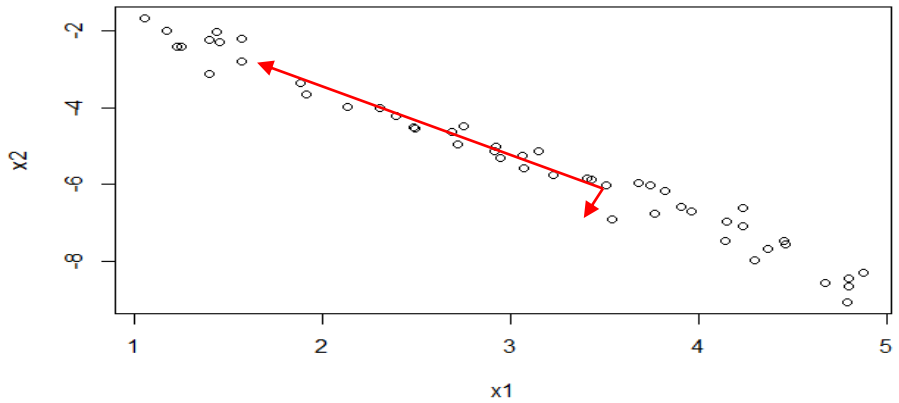
```
> c <- cov(d[,c("x1","x2")])
> c
          x1        x2
x1  1.358045 -2.35219
x2 -2.352190  4.18192
> eigen(c)
$values
[1] 5.51340440 0.02656068

$vectors
           [,1]       [,2]
[1,] -0.4926140 -0.8702479
[2,]  0.8702479 -0.4926140
```

$$E_1 = \begin{bmatrix} -0.49 \\ 0.87 \end{bmatrix} \text{ with } \lambda_1 = 5.51$$

Dominant Principal Component

$$E_2 = \begin{bmatrix} -0.87 \\ -0.49 \end{bmatrix} \text{ with } \lambda_2 = 0.02$$



The relative size of eigenvalues says variance in one direction is much higher than in the other.

```
> c <- cov(d[,c("x1","x2")])
> c
          x1       x2
x1  1.358045 -2.35219
x2 -2.352190  4.18192
> eigen(c)
$values
[1] 5.51340440 0.02656068

$vectors
           [,1]       [,2]
[1,] -0.4926140 -0.8702479
[2,]  0.8702479 -0.4926140
```

Remember, in the other example, when we transformed the data points from original variables x1, x2 into new transformed variables, X1 (x2+x1) and X2 (x2-x1), we could reduce the dimensions?

This matrix of eigenvectors as a whole also allows you to transform each one of our data-points into new variables X1 and X2.

Transform into Principal Components

y	x1	x2
-42.4957832	4.797034	-8.651047
-2.0248387	1.220774	-2.428135
-1.9379560	1.437964	-2.037009
-13.0806296	2.719718	-4.960111
-35.1968731	4.302478	-7.967436
-31.7650916	4.139152	-7.471322
-22.4288444	3.681965	-5.951819
-21.4937406	3.512183	-6.016649
-0.9004686	1.050161	-1.683525
-3.3677489	1.398850	-3.111843
-26.3779407	3.907309	-6.573550
-34.6572233	4.460774	-7.552211
-24.1480305	3.819653	-6.157824
-23.1244874	3.744704	-6.023781
-2.3255245	1.452426	-2.284816
-29.7199029	4.153706	-6.950889
-2.1284295	1.393772	-2.233364
-16.0110878	3.060388	-5.258223

Any record in our data set (x_1,x_2) when multiplied by the matrix of eigenvectors, we get the new coordinates in the rotated principal component axis.

$$\begin{bmatrix} X_1 \\ X_2 \end{bmatrix} = \begin{bmatrix} -0.49 & -0.87 \\ 0.87 & -0.49 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$\begin{bmatrix} X_1 \\ X_2 \end{bmatrix} = \begin{bmatrix} -0.49\, x_1 - 0.87 x_2 \\ 0.87 x_1 - 0.49 x_2 \end{bmatrix}$$

CSE 7302c

x1	x2
4.797034	-8.651047
1.220774	-2.428135
1.437964	-2.037009
2.719718	-4.960111
4.302478	-7.967436
4.139152	-7.471322
3.681965	-5.951819
3.512183	-6.016649
1.050161	-1.683525
1.39885	-3.111843
3.907309	-6.57355
4.460774	-7.552211
3.819653	-6.157824
3.744704	-6.023781
1.452426	-2.284816
4.153706	-6.950889
1.393772	-2.233364
3.060388	-5.258223
4.788337	-9.048708
2.918199	-5.015753

X1	X2
-9.891641591	0.087018102
-2.714449748	0.133757285
-2.481064002	-0.247926
-5.656297344	0.076591242
-9.053105345	0.180648073
-8.540906504	0.07838948
-6.993345492	-0.272272944
-6.966126674	-0.09258435
-1.982408107	-0.084572421
-3.39716793	0.315591153
-7.645413199	-0.162104692
-8.769735486	-0.161654336
-7.240447948	-0.29061469
-7.086876382	-0.291421939
-2.703841707	-0.138438327
-8.095170283	-0.19064869
-2.630171931	-0.112740783
-6.083547498	-0.073021965
-10.23342098	0.290480024
-5.802494197	-0.068726403

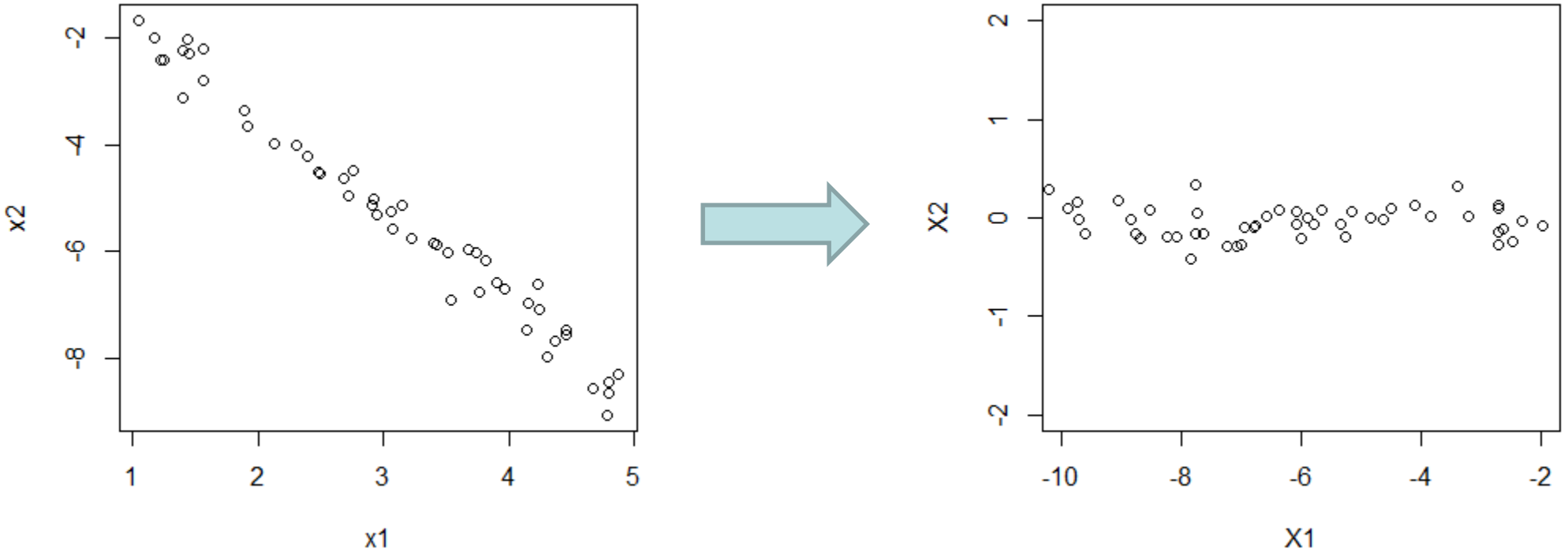
```
> eigen(c)
$values
[1] 5.51340440 0.02656068
```

Var(X1)=5.5134

Var(X2)=0.02656

Remember: The eigenvalues are exactly the same values!

Data Transformed into the Basis of Principal Components

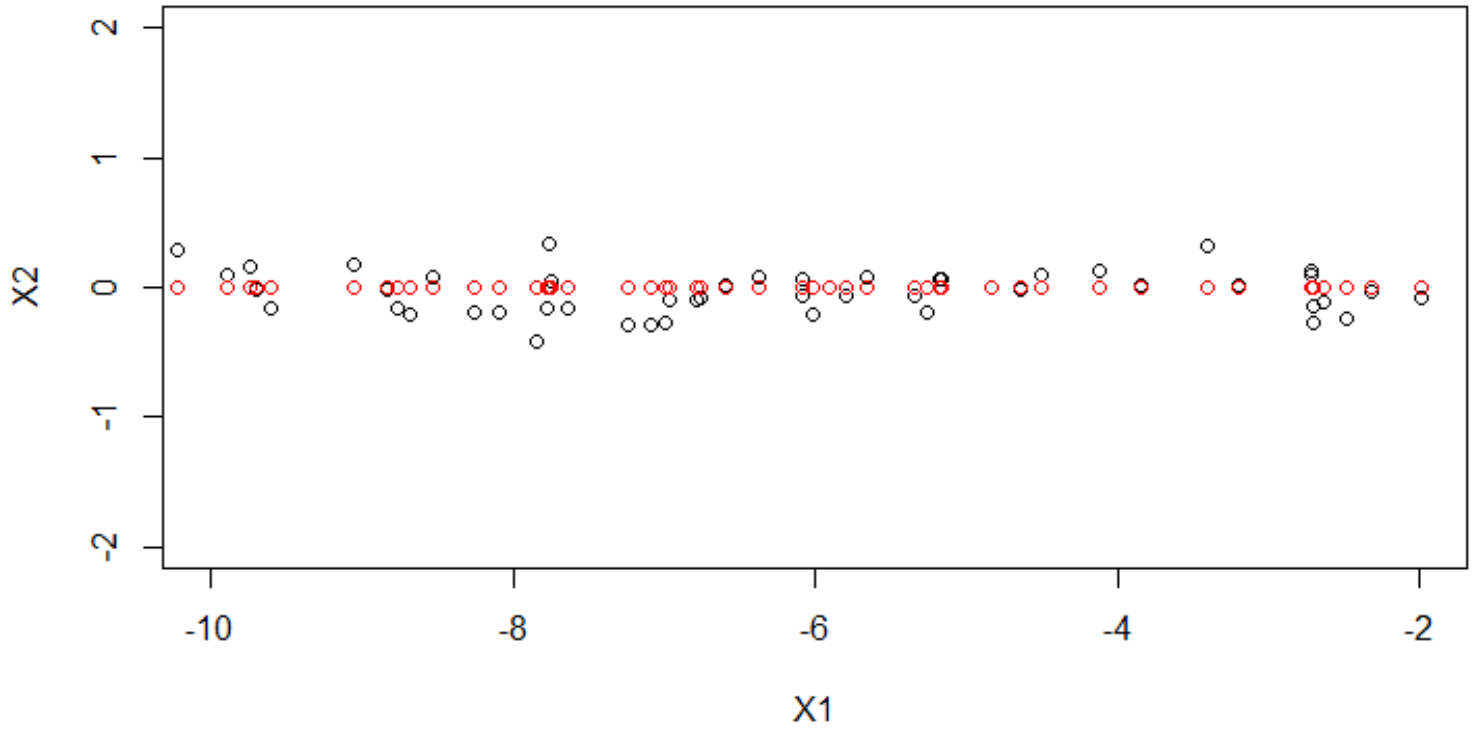


Since X2 variable has small variance, we can now drop it by setting it to zero.

CSE 7302c

Dimensionality Reduction

X2 has been set to zero. Red dots are the approximated points.



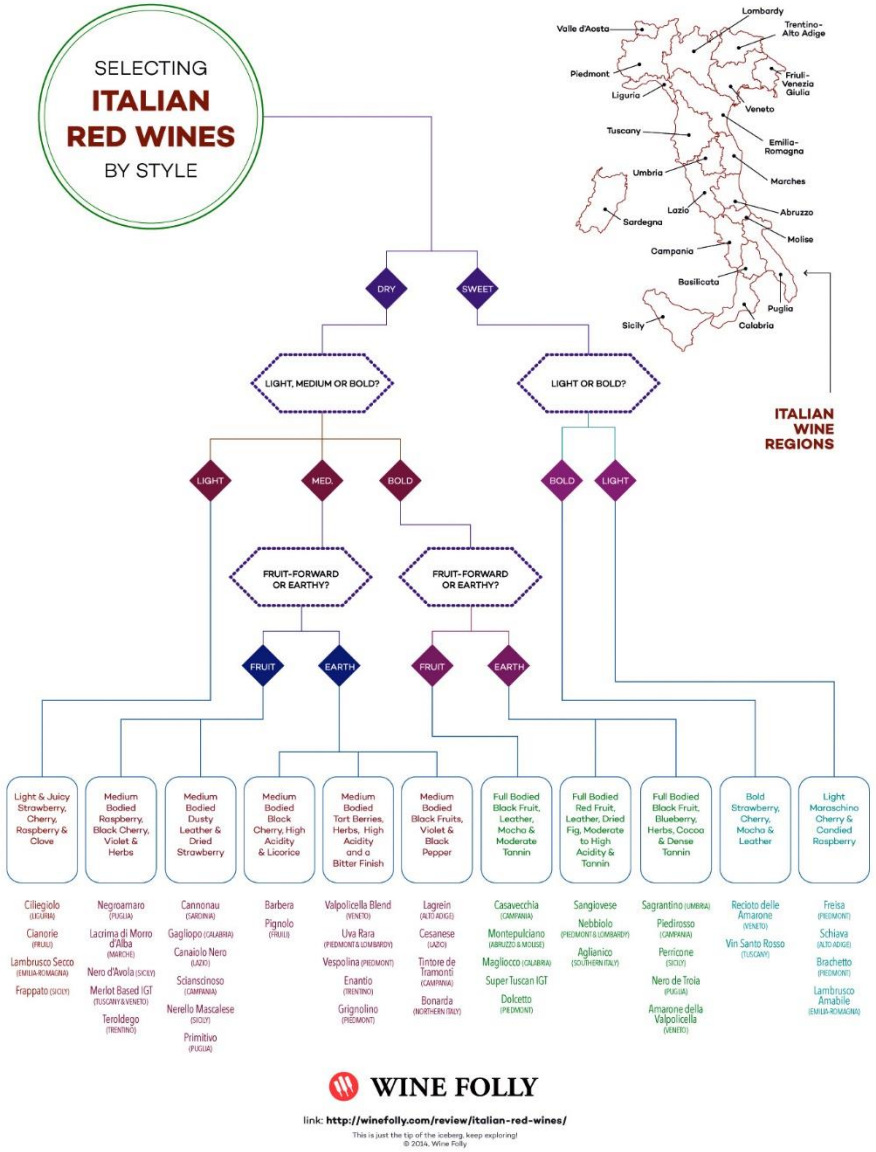
So now instead of doing regression of y vs (x1,x2), we are going to do regression of y vs X1 . This is the point of doing PCA! It allows us to ignore variables with low variance.

CSE 7302c

Lets now apply the idea to an example with more features.

CSE 7302c

Chemical Analysis of Wine



We have chemical analysis of 178 different wines produced by vineyards in Italy.

The wines come from 3 different grapes – Barolo, Grignolino, Barbera.

We are trying to identify the type of grape from 13 features.

CSE 7302c

Data Structure

WineClass	Alcohol	MalicAcid	Ash	AlkalinityAsh	Mg	Phenols	Flavanoids	NonFlavanoidPhenols	Proanthocyanins	Color	Hue	ODRatio	Proline
barolo	14.23	1.71	2.43	15.6	127	2.8	3.06	0.28	2.29	5.64	1.04	3.92	1065
barolo	13.2	1.78	2.14	11.2	100	2.65	2.76	0.26	1.28	4.38	1.05	3.4	1050
barolo	13.16	2.36	2.67	18.6	101	2.8	3.24	0.3	2.81	5.68	1.03	3.17	1185
barolo	14.37	1.95	2.5	16.8	113	3.85	3.49	0.24	2.18	7.8	0.86	3.45	1480
barolo	13.24	2.59	2.87	21	118	2.8	2.69	0.39	1.82	4.32	1.04	2.93	735
barolo	14.2	1.76	2.45	15.2	112	3.27	3.39	0.34	1.97	6.75	1.05	2.85	1450
barolo	14.39	1.87	2.45	14.6	96	2.5	2.52	0.3	1.98	5.25	1.02	3.58	1290
barolo	14.06	2.15	2.61	17.6	121	2.6	2.51	0.31	1.25	5.05	1.06	3.58	1295
barolo	14.83	1.64	2.17	14	97	2.8	2.98	0.29	1.98	5.2	1.08	2.85	1045
barolo	13.86	1.35	2.27	16	98	2.98	3.15	0.22	1.85	7.22	1.01	3.55	1045
barolo	14.1	2.16	2.3	18	105	2.95	3.32	0.22	2.38	5.75	1.25	3.17	1510
barolo	14.12	1.48	2.32	16.8	95	2.2	2.43	0.26	1.57	5	1.17	2.82	1280
barolo	13.75	1.73	2.41	16	89	2.6	2.76	0.29	1.81	5.6	1.15	2.9	1320
barolo	14.75	1.73	2.39	11.4	91	3.1	3.69	0.43	2.81	5.4	1.25	2.73	1150
barolo	14.38	1.87	2.38	12	102	3.3	3.64	0.29	2.96	7.5	1.2	3	1547
barolo	13.63	1.81	2.7	17.2	112	2.85	2.91	0.3	1.46	7.3	1.28	2.88	1310
barolo	14.3	1.92	2.72	20	120	2.8	3.14	0.33	1.97	6.2	1.07	2.65	1280
barolo	13.83	1.57	2.62	20	115	2.95	3.4	0.4	1.72	6.6	1.13	2.57	1130
barolo	14.19	1.59	2.48	16.5	108	3.3	3.93	0.32	1.86	8.7	1.23	2.82	1680
barolo	13.64	3.1	2.56	15.2	116	2.7	3.03	0.17	1.66	5.1	0.96	3.36	845
barolo	14.06	1.63	2.28	16	126	3	3.17	0.24	2.1	5.65	1.09	3.71	780
barolo	12.93	3.8	2.65	18.6	102	2.41	2.41	0.25	1.98	4.5	1.03	3.52	770
barolo	13.71	1.86	2.36	16.6	101	2.61	2.88	0.27	1.69	3.8	1.11	4	1035
barolo	12.85	1.6	2.52	17.8	95	2.48	2.37	0.26	1.46	3.93	1.09	3.63	1015
barolo	13.5	1.81	2.61	20	96	2.53	2.61	0.28	1.66	3.52	1.12	3.82	845
barolo	13.05	2.05	3.22	25	124	2.63	2.68	0.47	1.92	3.58	1.13	3.2	830

Dimension: 178 X 14. Number of Features = 13

PCA Steps - R

- PCA is done only on predictor numerical variables
- Convert categories to dummy numerical variables
- If there are values of different order of magnitude, scale them.

```
> WineData <- read.csv("wine.csv")
> str(WineData)
'data.frame':   178 obs. of  14 variables:
 $ WineClass          : Factor w/ 3 levels "barbera","barolo",..: 2 2 2 2 2 2 2 2 2 2 ...
 $ Alcohol            : num  14.2 13.2 13.2 14.4 13.2 ...
 $ MalicAcid          : num  1.71 1.78 2.36 1.95 2.59 1.76 1.87 2.15 1.64 1.35 ...
 $ Ash                : num  2.43 2.14 2.67 2.5 2.87 2.45 2.45 2.61 2.17 2.27 ...
 $ AlkalinityAsh      : num  15.6 11.2 18.6 16.8 21 15.2 14.6 17.6 14 16 ...
 $ Mg                 : int  127 100 101 113 118 112 96 121 97 98 ...
 $ Phenols            : num  2.8 2.65 2.8 3.85 2.8 3.27 2.5 2.6 2.8 2.98 ...
 $ Flavanoids         : num  3.06 2.76 3.24 3.49 2.69 3.39 2.52 2.51 2.98 3.15 ...
 $ NonFlavanoidPhenols: num  0.28 0.26 0.3 0.24 0.39 0.34 0.3 0.31 0.29 0.22 ...
 $ Proanthocyanins    : num  2.29 1.28 2.81 2.18 1.82 1.97 1.98 1.25 1.98 1.85 ...
 $ Color              : num  5.64 4.38 5.68 7.8 4.32 6.75 5.25 5.05 5.2 7.22 ...
 $ Hue                : num  1.04 1.05 1.03 0.86 1.04 1.05 1.02 1.06 1.08 1.01 ...
 $ ODRatio            : num  3.92 3.4 3.17 3.45 2.93 2.85 3.58 3.58 2.85 3.55 ...
 $ Proline            : int  1065 1050 1185 1480 735 1450 1290 1295 1045 1045 ...
> # PCA analysis is done only on the predictors
> wine.predictors <- WineData[,-1]
>
> # Since the predictors are of completely different magnitude,
> # we need scale them before the analysis.
> scaled.Predictors <- scale(wine.predictors)
```

CSE 7302c

Principal Components in R

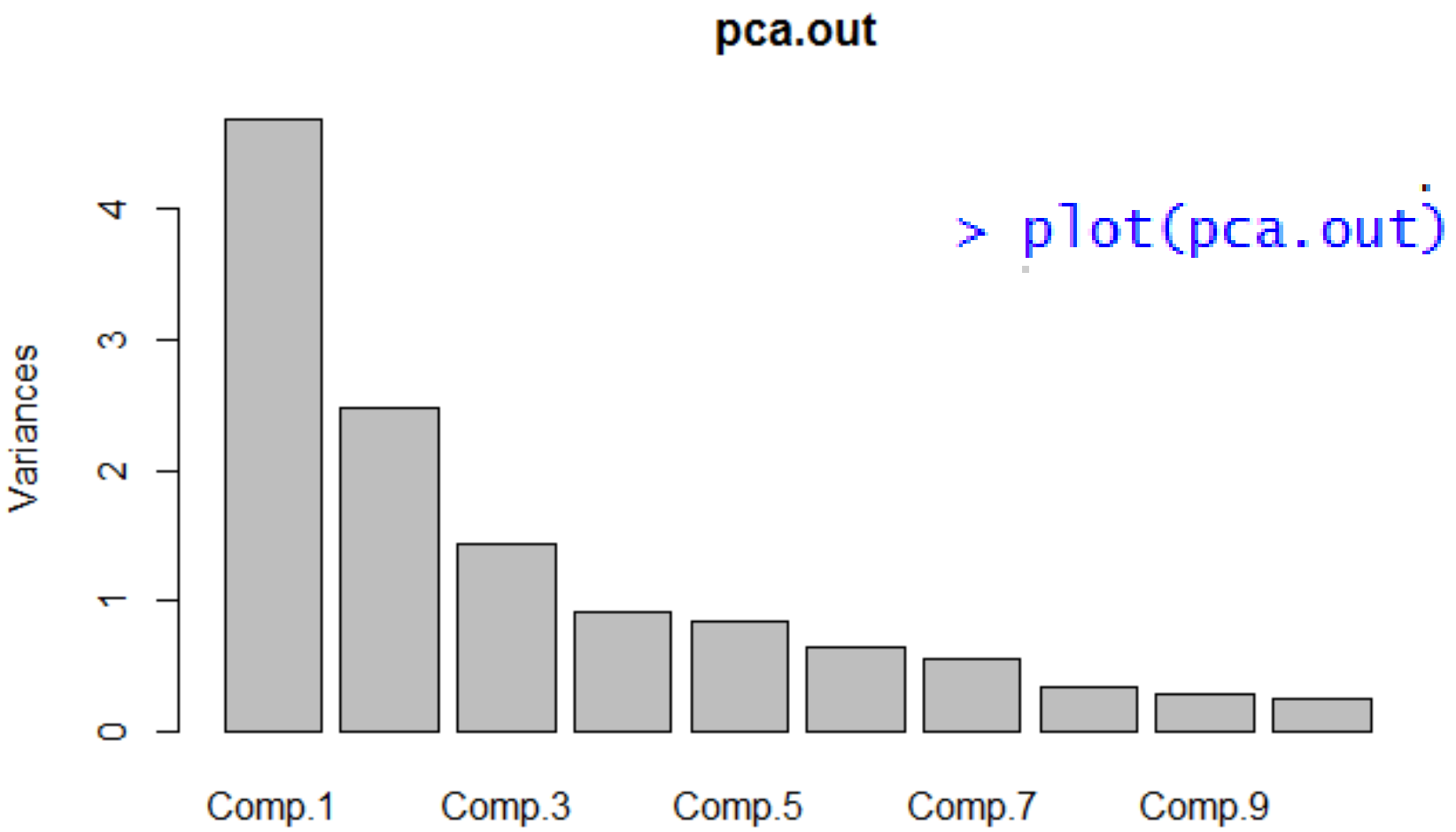
- *princomp* performs a principal components analysis on a given numeric data matrix

- It outputs are:
  - *loadings*: The coefficients for linear transformations into the new variables
  - *scores*: The input features into the transformed bases
  - It also gives a summary of percentage of variance explained by each Principal Component

CSE 7302c

```
> # compute PCs
> pca.out = princomp(scaled.Predictors)
> summary(pca.out)
Importance of components:
                          Comp.1    Comp.2    Comp.3    Comp.4     Comp.5     Comp.6     Comp.7
Standard deviation     2.1631951 1.5757366 1.1991447 0.9559347 0.92110518 0.79878171 0.74022473
Proportion of Variance 0.3619885 0.1920749 0.1112363 0.0706903 0.06563294 0.04935823 0.04238679
Cumulative Proportion  0.3619885 0.5540634 0.6652997 0.7359900 0.80162293 0.85098116 0.89336795
                           Comp.8     Comp.9    Comp.10    Comp.11    Comp.12     Comp.13
Standard deviation     0.58867607 0.53596364 0.49949266 0.47383559 0.40966094 0.320619963
Proportion of Variance 0.02680749 0.02222153 0.01930019 0.01736836 0.01298233 0.007952149
Cumulative Proportion  0.92017544 0.94239698 0.96169717 0.97906553 0.99204785 1.000000000
```

- There are 13 principal components (since there were 13 initial features).
- The output principal components are ordered according to the percentage of variance explained.
- The top row gives the (sqrt of) eigenvalues, the 2nd row gives the percentage of explained variance.
- First 8 components explain over 90% of the variance!


pca.out
> plot(pca.out)
Variances
0
1
2
3
4
Comp.1
Comp.3
Comp.5
Comp.7
Comp.9

CSE 7302c

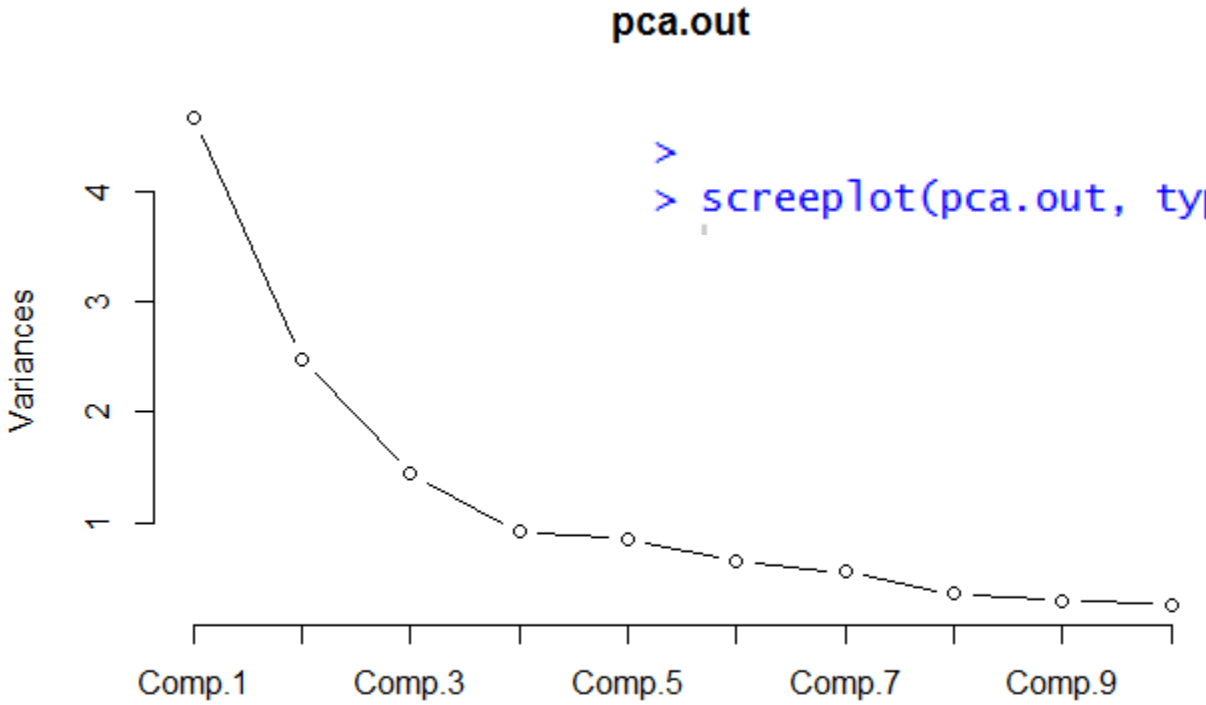
Dimensionality Reduction

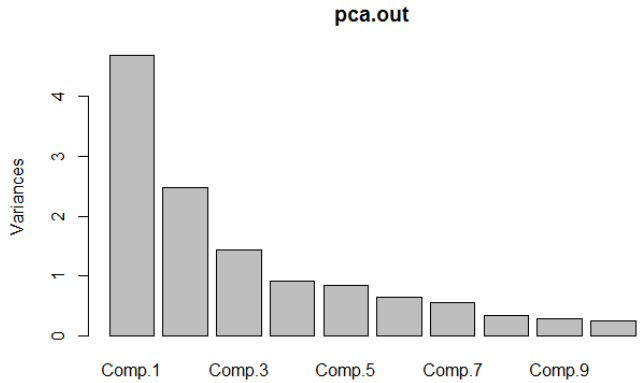
- Let us say we set a threshold of 80% variance explanatory power.
- Keeping the first 5 principal components explains 80% of variance.
- So, we can choose to ignore the remaining 8 components and build our model.

```
>
> compressed_features = pca$scores[,1:5]
> library(nnet)
> multout.pca <- multinom(WineData$wine.class ~ compressed_features)
```

CSE 7302c

How Many Components to Choose?





Rule of thumb: Pick the point, which is at the “elbow” in the plot.
Over here, it happens at around 4th or 5th component.

CSE 7302c

PCA Performance: Multinomial Classification

Regular

```
> summary(multout.full)
Call:
multinom(formula = WineClass ~ scaled.Predictors, data = WineData)

Coefficients:
          (Intercept) scaled.PredictorsAlcohol scaled.PredictorsMalicAcid scaled.PredictorsAsh scaled.PredictorsAlkalinityAsh
barolo         -0.5736                 11.552                    -1.888                5.39                        -13.075
grignolino      4.8870                 -7.864                    -3.600              -11.81                          5.087
          scaled.PredictorsMg scaled.PredictorsPhenols scaled.PredictorsFlavanoids scaled.PredictorsNonFlavanoidPhenols
barolo                 2.5506                   8.113                      11.892                             -5.154
grignolino             0.3456                  -1.243                       9.796                              4.570
          scaled.PredictorsProanthocyanins scaled.PredictorsColor scaled.PredictorsHue scaled.PredictorsODRatio
barolo                               3.339                0.9285                5.60                     9.196
grignolino                           2.597              -19.8481               12.53                     5.140
          scaled.PredictorsProline
barolo                       14.49
grignolino                  -11.35

Std. Errors:
          (Intercept) scaled.PredictorsAlcohol scaled.PredictorsMalicAcid scaled.PredictorsAsh scaled.PredictorsAlkalinityAsh
barolo         119427                  167686                     58547              109387                         106846
grignolino      12729                    4156                      1606                8713                           3680
          scaled.PredictorsMg scaled.PredictorsPhenols scaled.PredictorsFlavanoids scaled.PredictorsNonFlavanoidPhenols
barolo                  79264                   151221                      80842                             137785
grignolino               7269                     4339                       8241                               3672
          scaled.PredictorsProanthocyanins scaled.PredictorsColor scaled.PredictorsHue scaled.PredictorsODRatio
barolo                               82716                114552               74901                     135373
grignolino                            5388                  7194                5822                       3090
          scaled.PredictorsProline
barolo                       99521
grignolino                   10364

Residual Deviance: 0.0001846
AIC: 56
```

AIC = 56

With PCA

```
> summary(multout.pca)
Call:
multinom(formula = WineData$WineClass ~ compressed_features)

Coefficients:
          (Intercept) compressed_featuresComp.1 compressed_featuresComp.2 compressed_featuresComp.3 compressed_featuresComp.4
barolo          11.21                    -33.58                     3.355                     3.839                     1.303
grignolino      19.30                    -14.06                    44.527                    14.878                     2.494
          compressed_featuresComp.5
barolo                       20.489
grignolino                    7.312

Std. Errors:
          (Intercept) compressed_featuresComp.1 compressed_featuresComp.2 compressed_featuresComp.3 compressed_featuresComp.4
barolo          647.5                    395.65                     262.6                     44.72                     89.35
grignolino      258.6                     36.12                     394.5                    165.18                     78.81
          compressed_featuresComp.5
barolo                       294.0
grignolino                   127.4

Residual Deviance: 0.07747
AIC: 24.08
```

AIC = 24.08

AIC in Multinomial Logistic Regression is given by $AIC = D + 2kr$, where *k* is the # of parameters (including intercept) and *r* is the # of categories minus 1.

PCA does not always guarantee better performance. The main use is dimensionality reduction.

CSE 7302c

PCA: Visualization

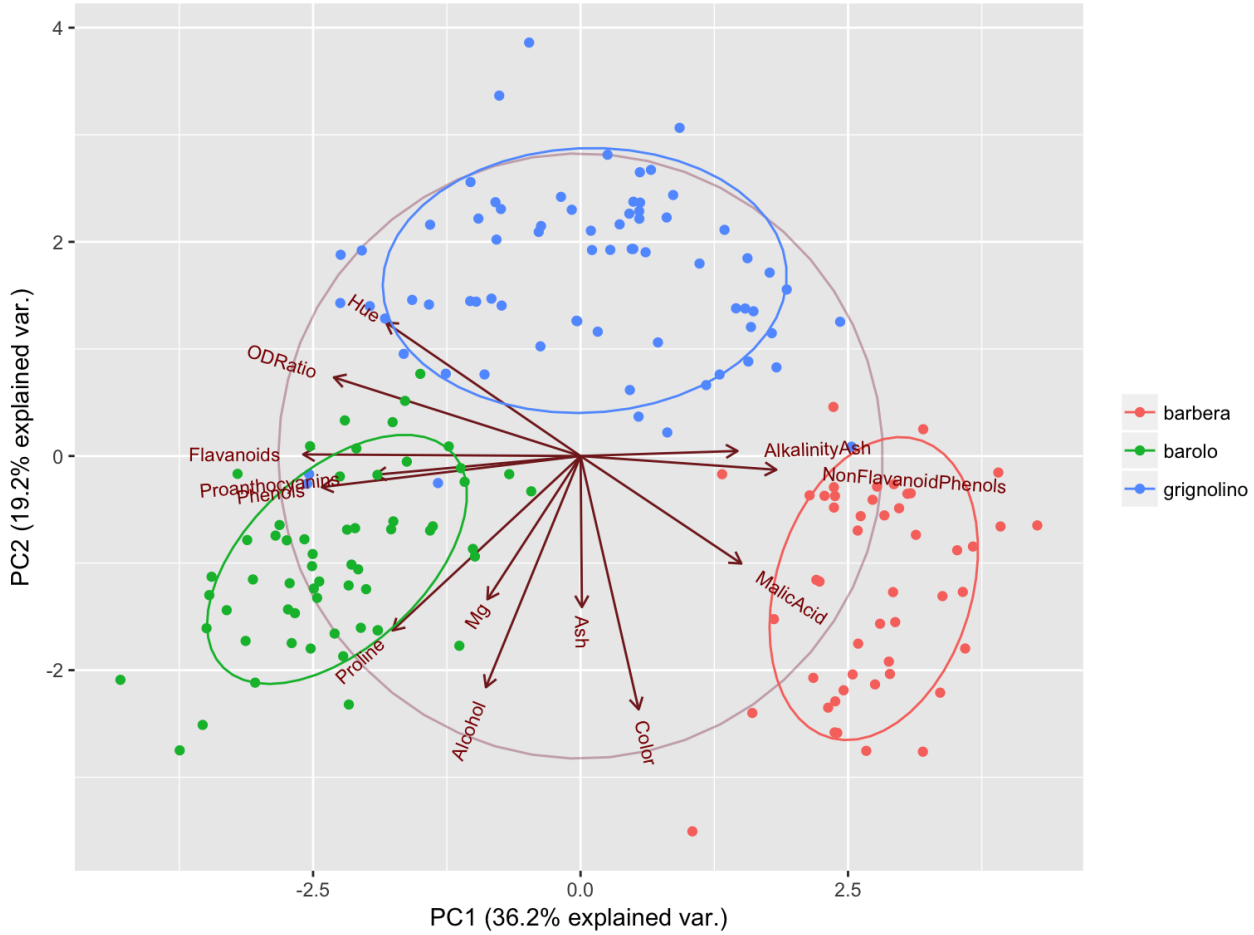
- Many times visualizing the spread in the data-points using only the first 2 Principal Components can provide useful information about the spread of the data

- The function *biplot* lets you do that

- Also check *ggbiplot* in the *library(ggbiplot)*

For installing ggbiplot see: http://www.vince.vu/software/#ggbiplot

Projection Using First 2 Components



Each class nicely gets segmented when viewed from the Principal Component basis.

biplot is also occasionally used to identify outliers.

```
> library(ggbiplot)
> g <- ggbiplot(pca.out, obs.scale = 1, var.scale = 1,
+               groups = WineData$WineClass, ellipse = TRUE
, circle = TRUE)
> g  + scale_color_discrete(name = '')
```

CSE 7302c

1000 Genomes Single Chromosome PCA Example

- Computing principal components of genomic variant data across one chromosome from 2504 people from the 1000 genomes project.
- Effective clustering of people by ethnicity (5 ethnicities) using PCA.
- A sparse matrix with 2504 rows (people) by 18,12,841 columns (variants).

PEL	Peruvians from Lima, Peru	AMR	1	1	1
GIH	Gujarati Indian from Houston, Texas	SAS	1	1	1
PJL	Punjabi from Lahore, Pakistan	SAS	1	1	1
BEB	Bengali from Bangladesh	SAS	1	1	1
STU	Sri Lankan Tamil from the UK	SAS	1	1	1
ITU	Indian Telugu from the UK	SAS	1	1	1

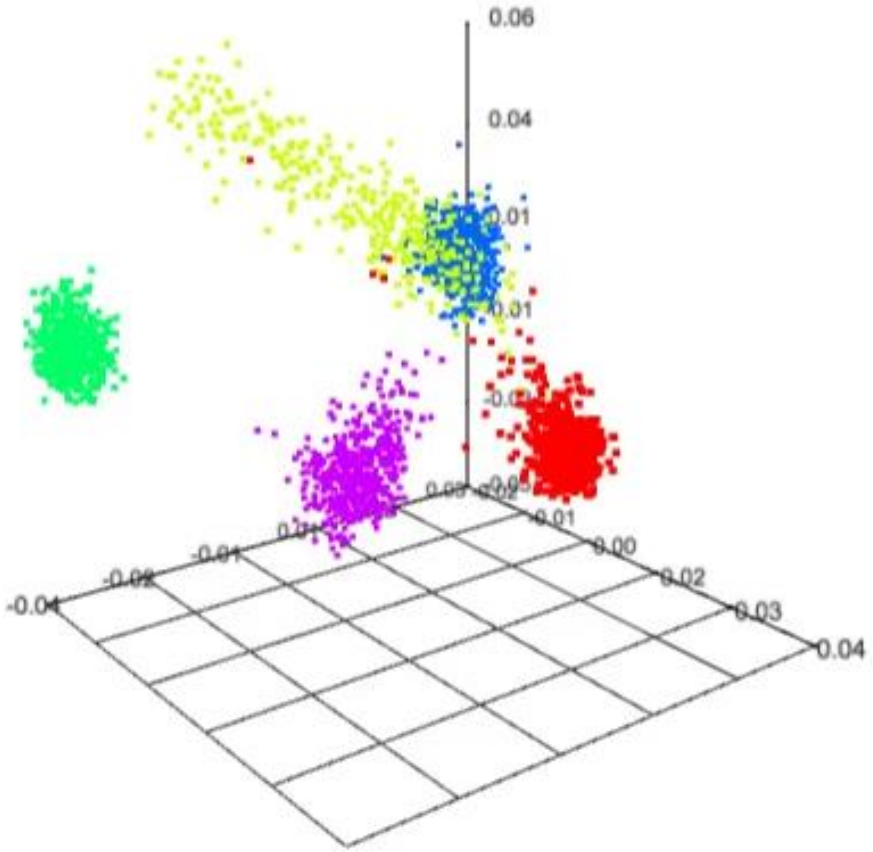
These populations have been divided into 5 super populations

- **AFR**, African
- **AMR**, Ad Mixed American
- **EAS**, East Asian
- **EUR**, European
- **SAS**, South Asian

When the code **ALL** is used this means that all individuals from that release are being considered.

Source: https://bwlewis.github.io/1000_genomes_examples/PCA.html and http://www.internationalgenome.org/faq/which-populations-are-part-your-study/
Last accessed: September 09, 2017

1000 Genomes Single Chromosome PCA Example

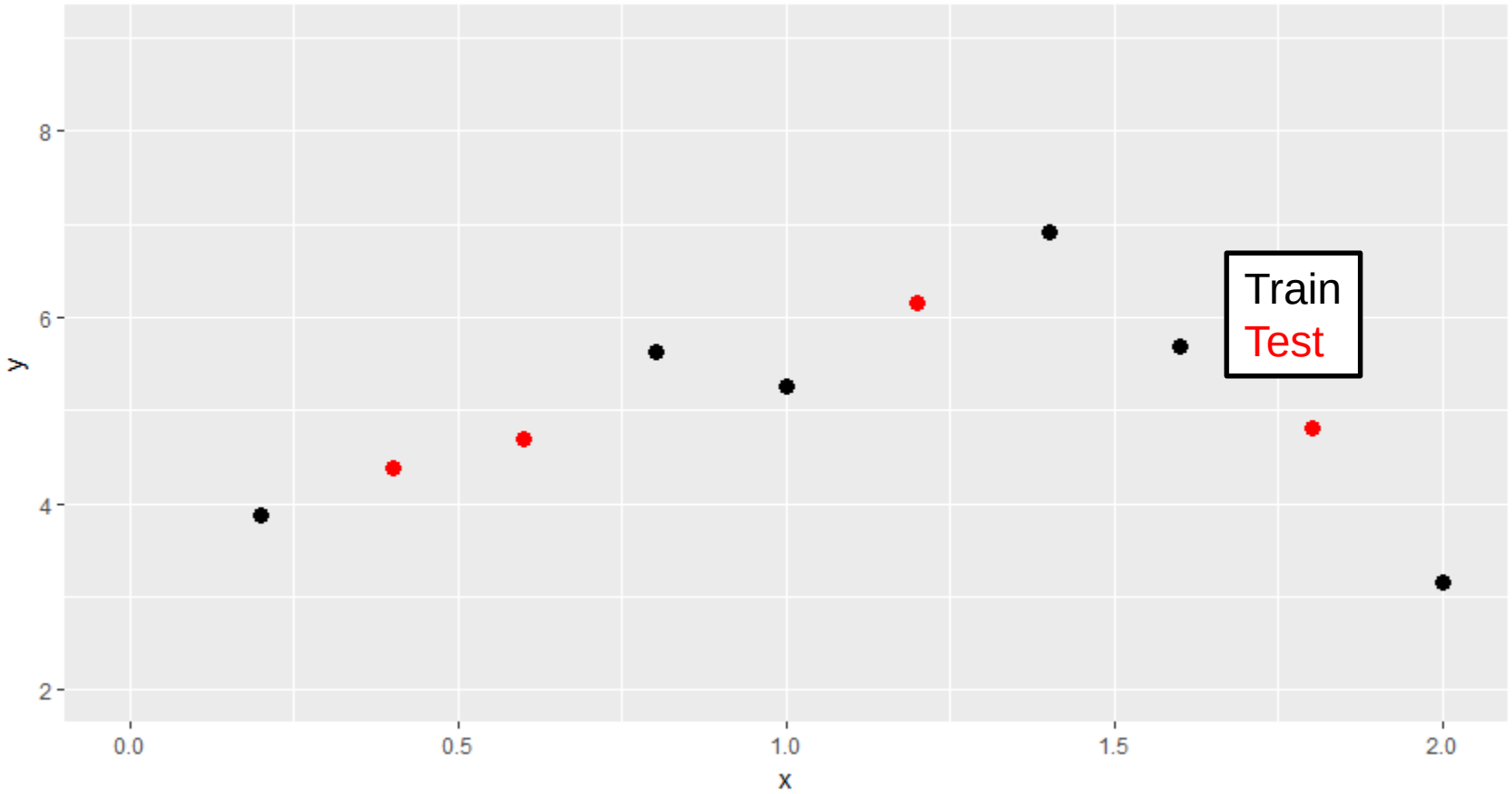


Source: https://bwlewis.github.io/1000_genomes_examples/PCA.html
Last accessed: September 09, 2017

CSE 7302c

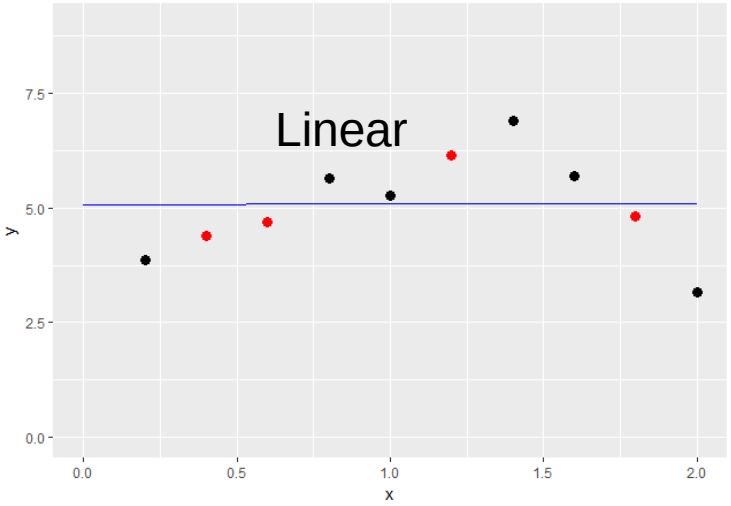
Advanced Regression Methods: Ridge Regression, LASSO Regression, Elastic Nets

REGULARIZATION



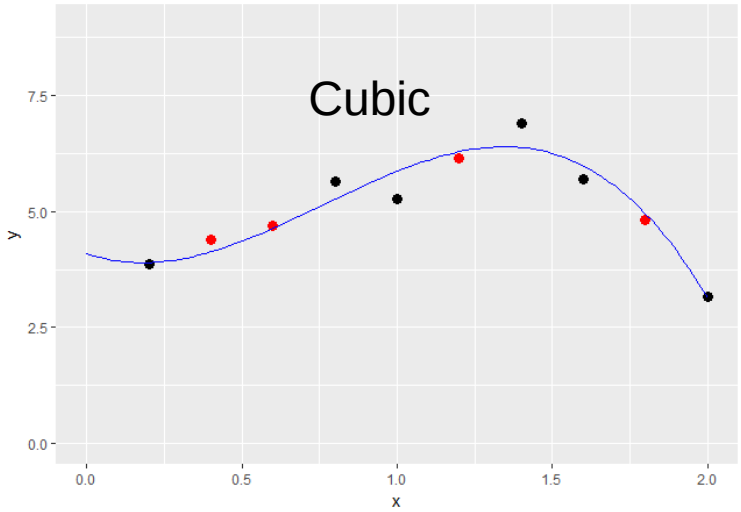
Decision point: What order polynomial should we use for the fit?

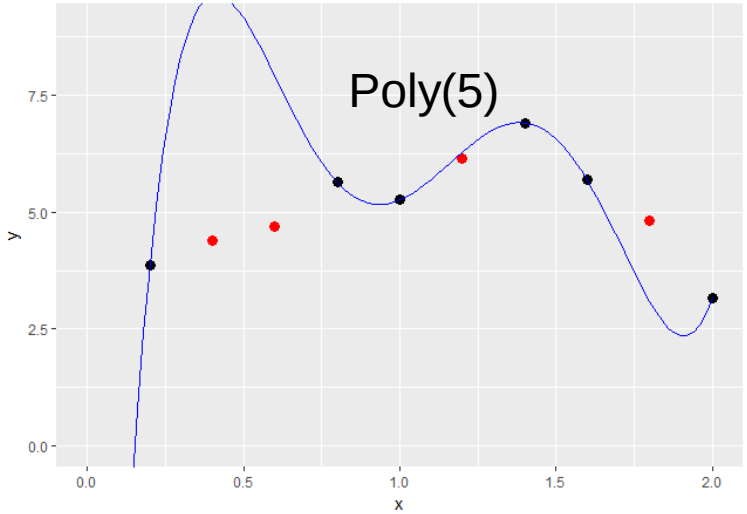
Problem with Overfitting



TrainErr=1.23
TestErr=0.7

TrainErr=0.38
TestErr=0.17





TrainErr=0
TestErr=3.2

CSE 7302c

Overfitting Problem

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 x^4 + \cdots$$

How many β_i do we need for a reasonable fit, while at the same time minimizing the risk of overfitting?

In general, given a regression problem with several features, is there a way to determine how many β_i do we need for a reasonable fit, while at the same time minimizing the risk of overfitting?

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_4 + \cdots$$

CSE 7302c

Regularization

- Standard Least Squares Regression involves minimizing SSE

$$\text{SSE} = \sum_{j=1}^{N} (y_j - \sum_{i=0}^{d} \beta_i \cdot x_i)^2 = (\mathbf{y} - \mathbf{X\beta})^{\mathrm{T}} \cdot (\mathbf{y} - \mathbf{X\beta})$$

$$\mathbf{y} = \text{vector of all training responses } y_j$$

$$\mathbf{X} = \text{matrix of all training samples } \mathbf{x}_j$$

- We can reduce overfitting by penalizing large coefficients (**must standardize** in order to be able to compare/add βs)

CSE 7302c

Ridge Regression

$$\min_{\beta}\left(\sum_{j=1}^{N}(y_j-\sum_{i=0}^{d}\beta_i\cdot x_i)^2+\lambda\sum_{i=1}^{d}\beta_i{}^2\right)$$

- Penalization of large coefficients through L2 norm (another name for Least Squares Error)
- Regularization parameter λ controls the extent of penalization
- $\lambda = 0$ corresponds to Least Squares Regression

CSE 7302c

Ridge Regression

- Occam's Razor

$$\min_{\beta}\left(\sum_{j=1}^{N}(y_j-\sum_{i=0}^{d}\beta_i\cdot x_i)^2+\lambda\sum_{i=1}^{d}\beta_i^{\,2}\right)$$

- Least Squares Regression
  - Fails when the number of features *p* is larger than number of data points *n*, e.g., in genetics studies. For example, there are infinite solutions for $5=\beta_1*3+\beta_2*2$ (two unknowns and one equation)

CSE 7302c

Ridge Regression

$$\min_{\beta}\left(\sum_{j=1}^{N}(y_j-\sum_{i=0}^{d}\beta_i\cdot x_i)^2+\lambda\sum_{i=1}^{d}\beta_i^{\,2}\right)$$

- Least Squares Regression
  - Fails when we have perfect multicollinearity

y	x1	x2
0	0	0
2	1	1
4	2	2
6	3	3

$y = x_1 + x_2$ or $y = 0.5x_1 + 1.5x_2$ or $y = 2x_1$ and so on

- Ridge Regression allows for a unique solution even in above situation. Which equation will you choose if you add the constraint to minimize $\sum_{i=1}^{d} b_i^{\,2}$ as well?

CSE 7302c

Ridge Regularization

The penalty term “regularizes” the problem when there is perfect multicollinearity

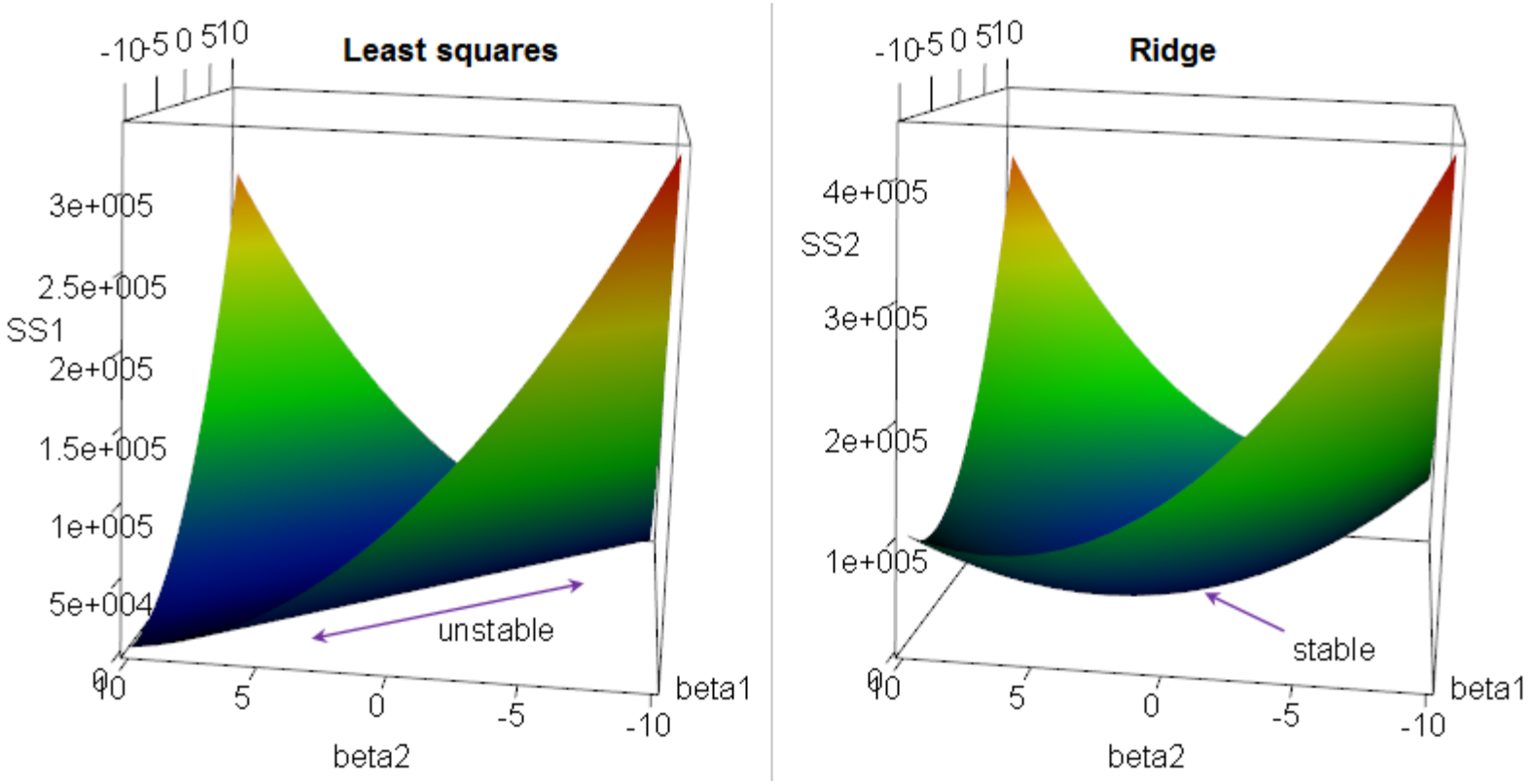
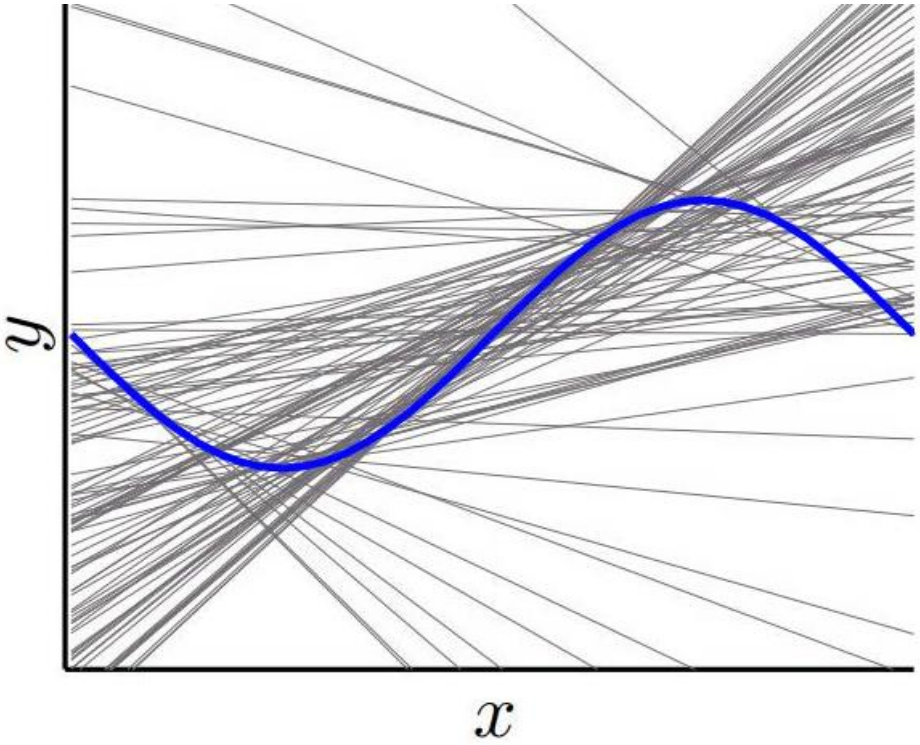


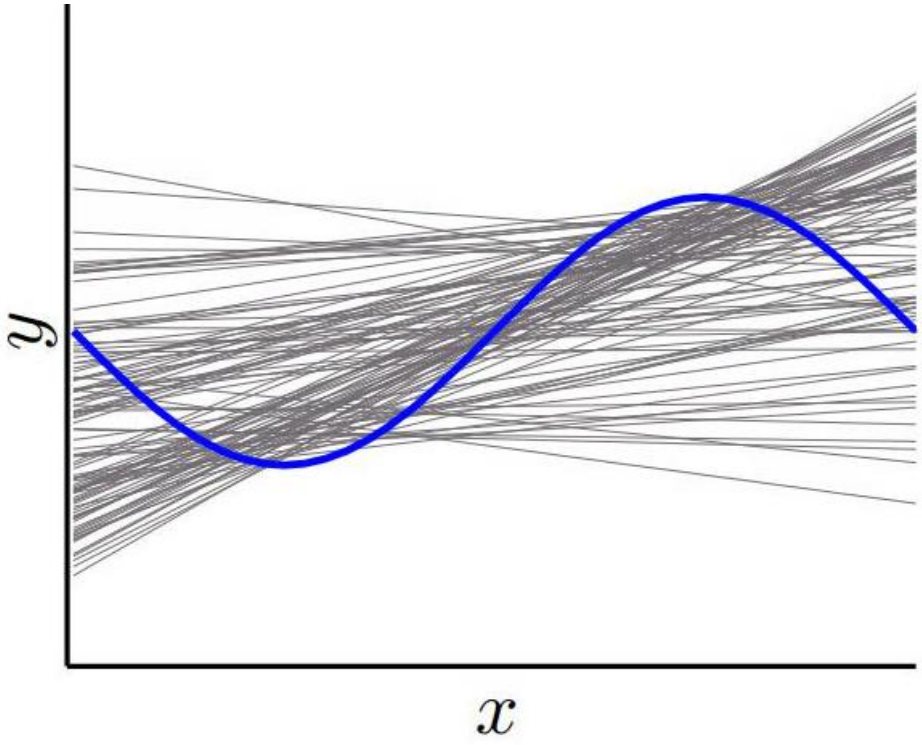
Image Source: https://stats.stackexchange.com/questions/151304/why-is-ridge-regression-called-ridge-why-is-it-needed-and-what-happens-when

CSE 7302c

Effect of Regularization



without regularization

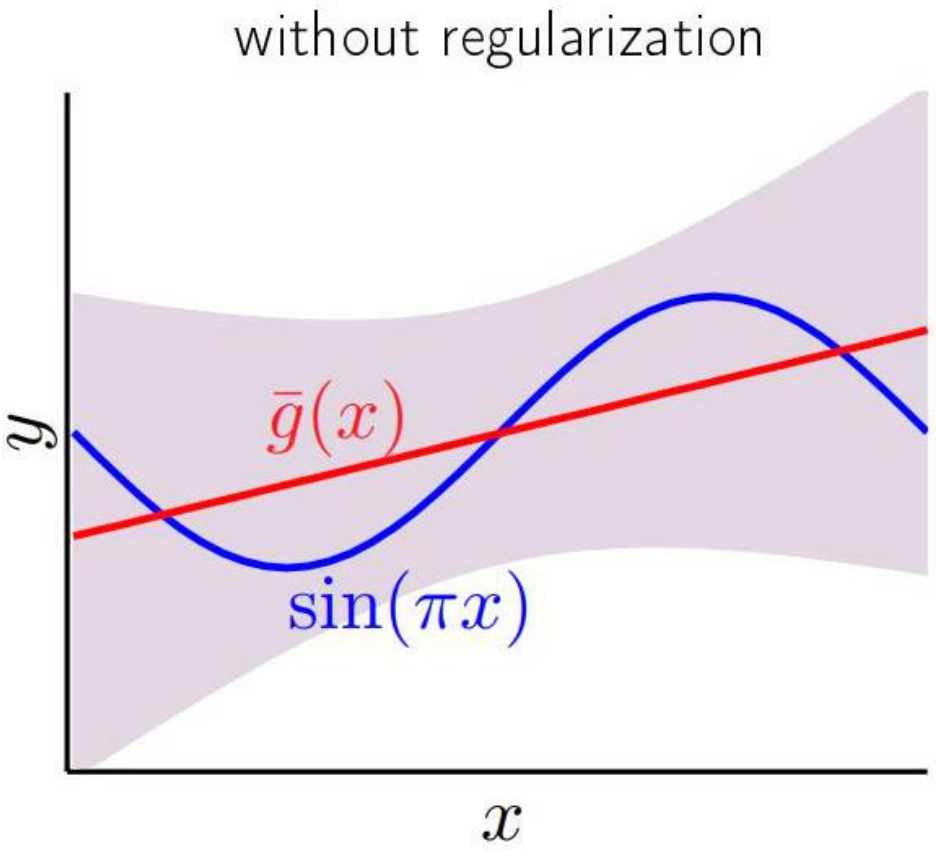


with regularization

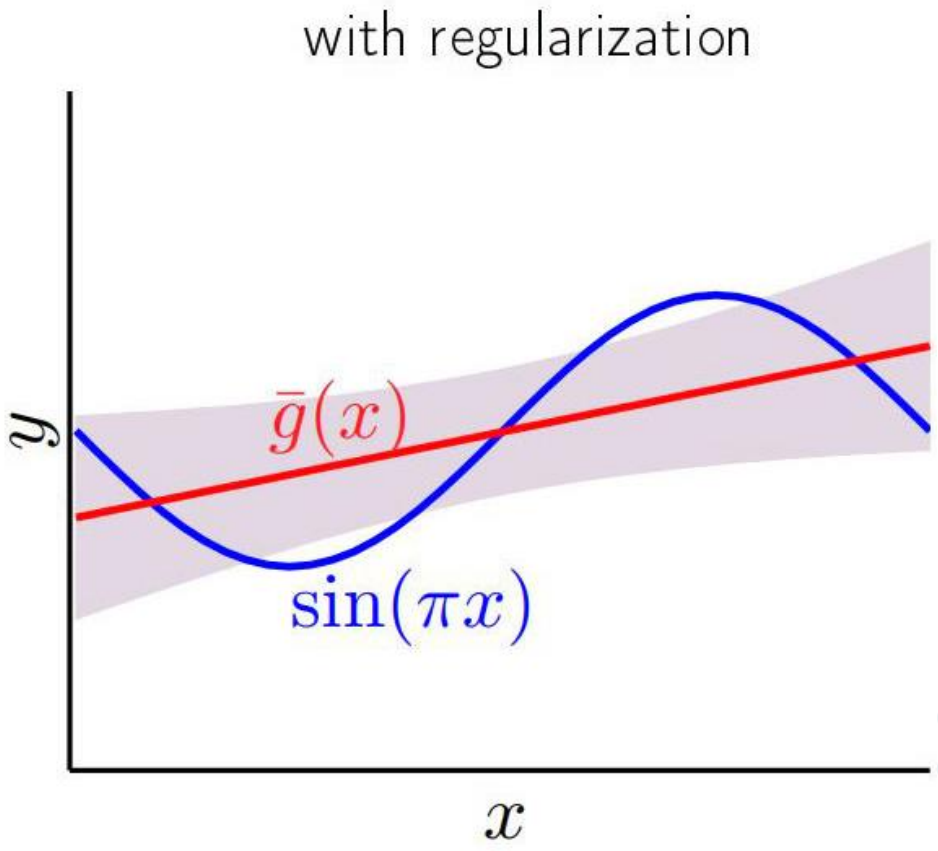
Source: http://work.caltech.edu/slides/slides12.pdf

CSE 7302c

Effect of Regularization



bias = **0.21** var = **1.69**



bias = **0.23** var = **0.33**

Source: http://work.caltech.edu/slides/slides12.pdf

CSE 7302c

LASSO Regularization

$$\min_{\beta}\left(\sum_{j=1}^{N}(y_j-\sum_{i=0}^{d}\beta_i\cdot x_i)^2+\lambda\sum_{i=1}^{d}|\beta_i|\right)$$

- Penalization of large coefficients through L1 norm (another name for Least Absolute Error)

- Least Absolute Shrinkage and Selection Operator (LASSO)

CSE 7302c

LASSO Regularization

$$\min_{\beta}\left(\sum_{j=1}^{N}(y_j-\sum_{i=0}^{d}\beta_i\cdot x_i)^2+\lambda\sum_{i=1}^{d}|\beta_i|\right)$$

y	x1	x2
0	0	0
2	1	1
4	2	2
6	3	3

Not useful when there is strong multicollinearity.

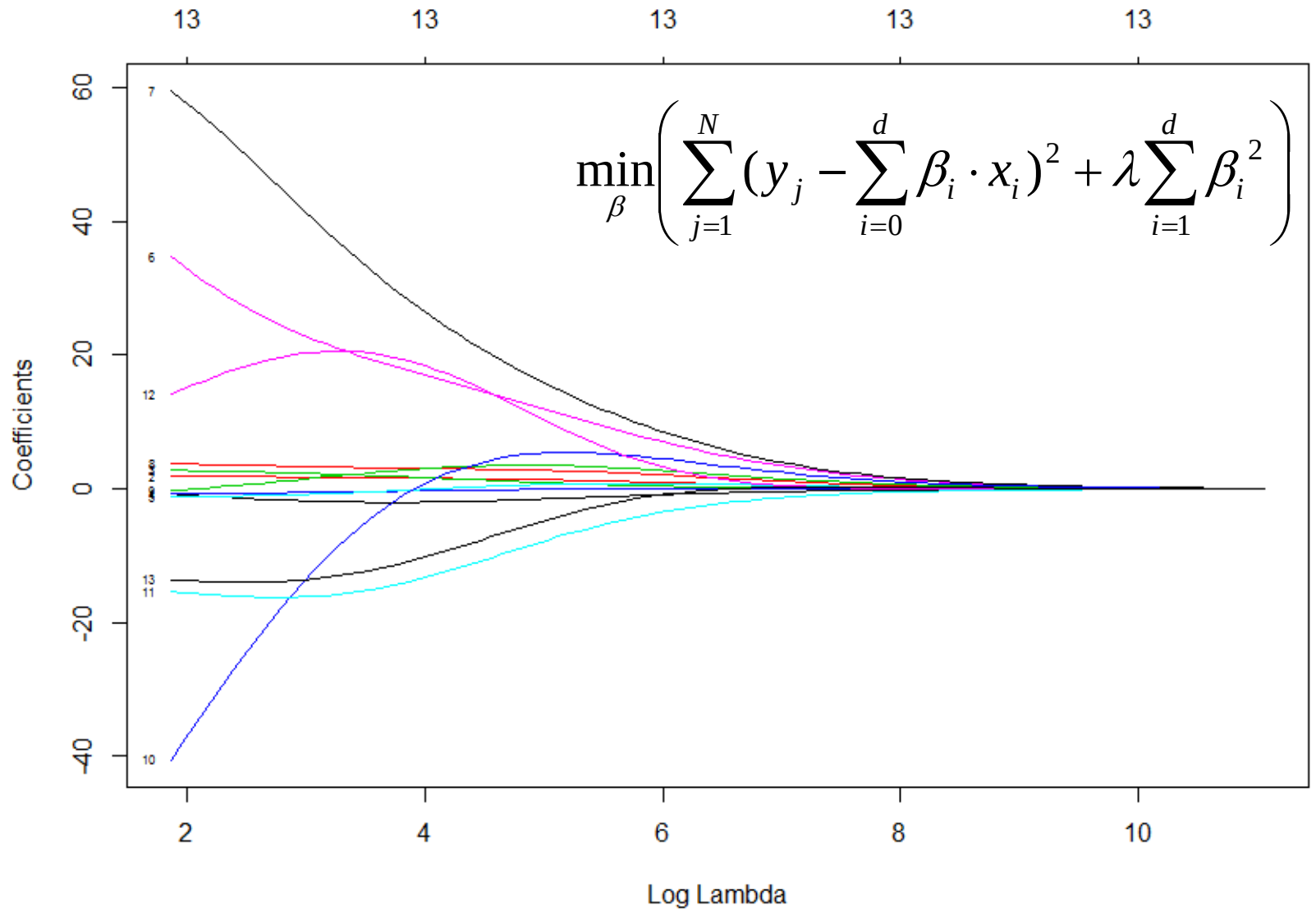
$y = x_1 + x_2$ or $y = 0.5x_1 + 1.5x_2$ or $y = 2x_1$ and so on

CSE 7302c

Regularization – Toy Purchase Data

CSE 7302c

Ridge: Coefficients



$$\min_{\beta}\left(\sum_{j=1}^{N}(y_j-\sum_{i=0}^{d}\beta_i\cdot x_i)^2+\lambda\sum_{i=1}^{d}\beta_i^{\,2}\right)$$

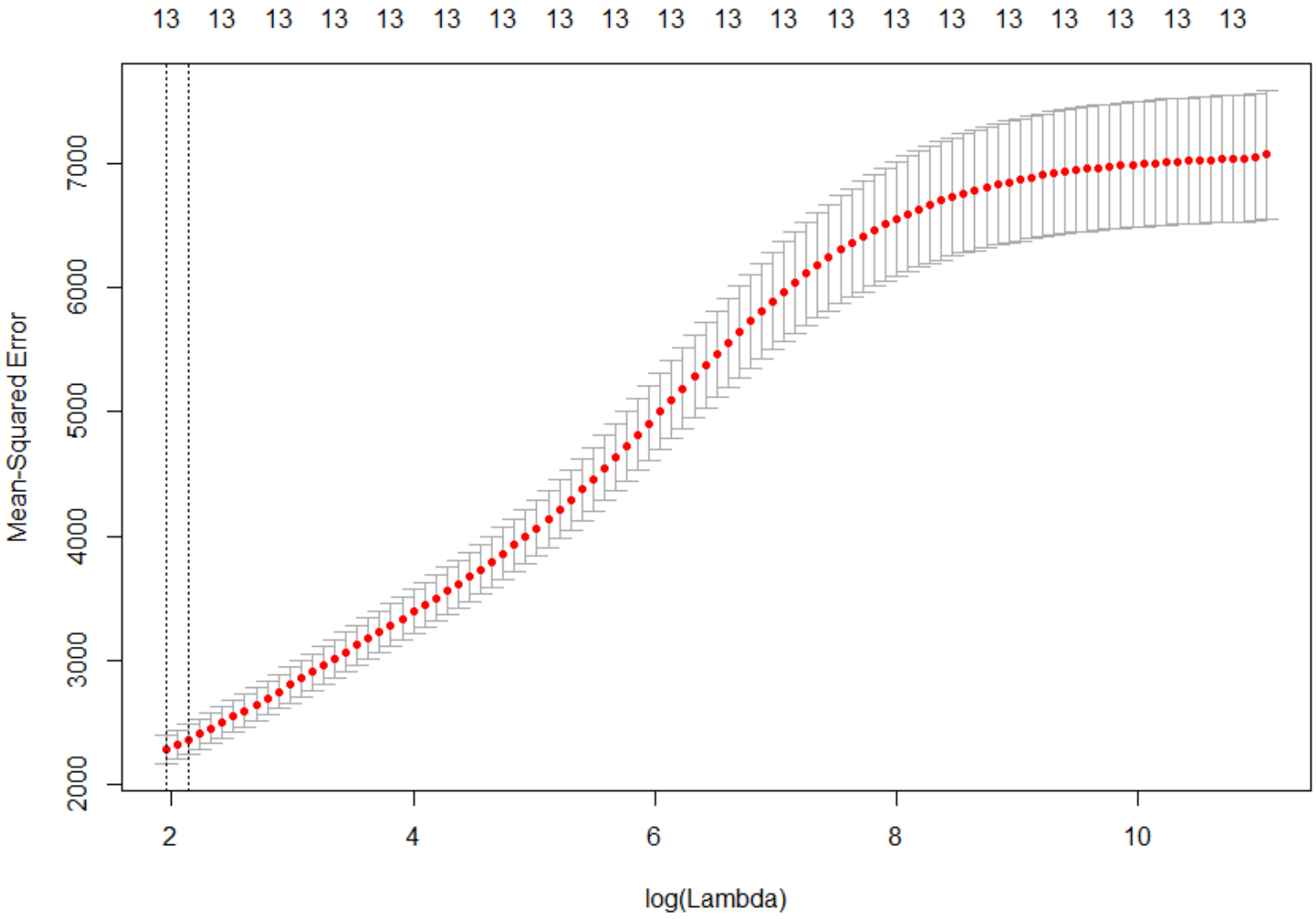
CSE 7302c

How to choose λ

- Choose a value for λ.
- Run *k*-fold cross validation and calculate mean "squared" error.
- Repeat the above 2 steps for various λ values.
- Pick λ that gives least error. R recommends the range from min λ to 1 stdev away.
- Pick λ by looking at the coefficients corresponding to these λ values, in addition to the minimum error.

CSE 7302c

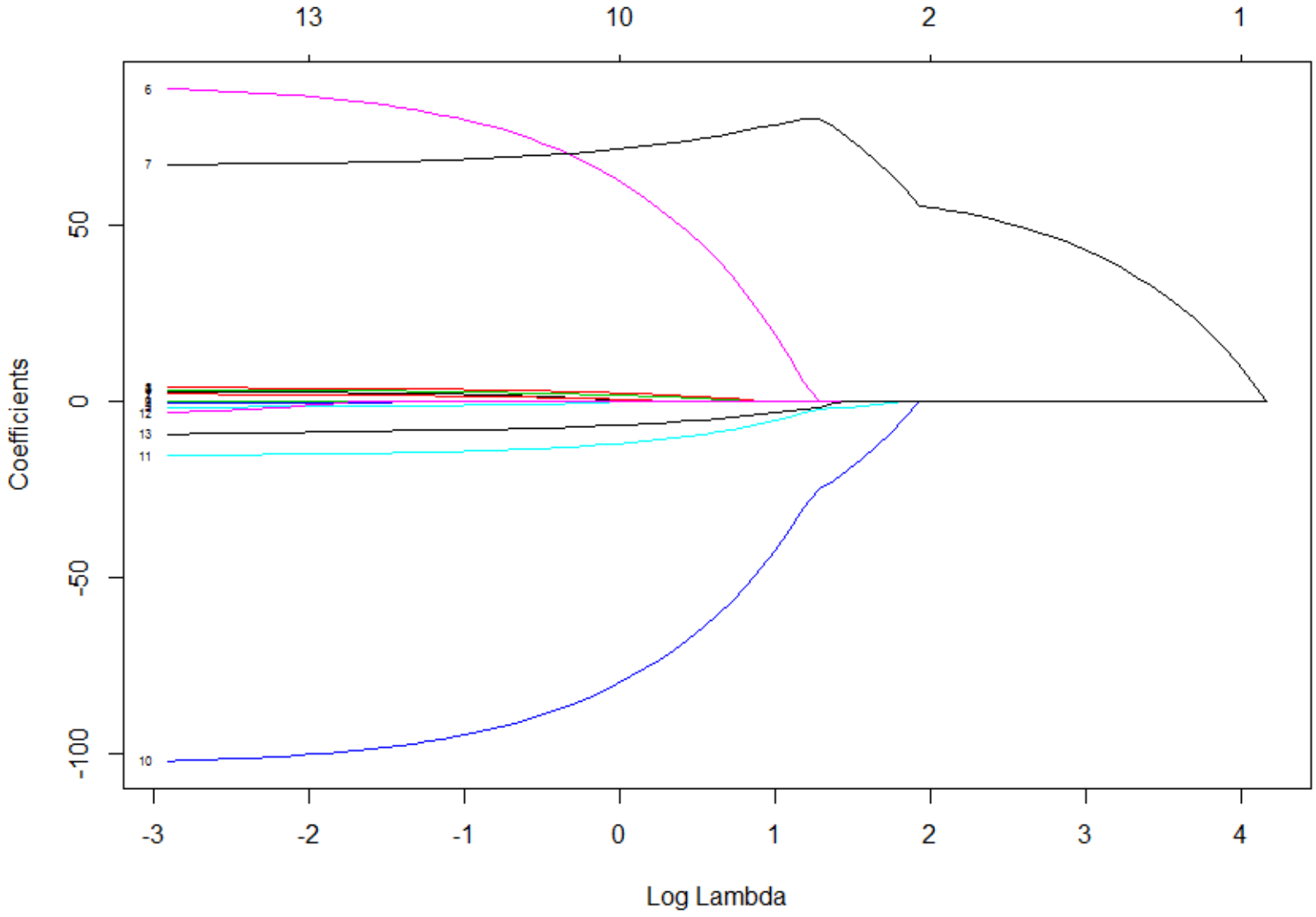
Ridge: MSE



CSE 7302c

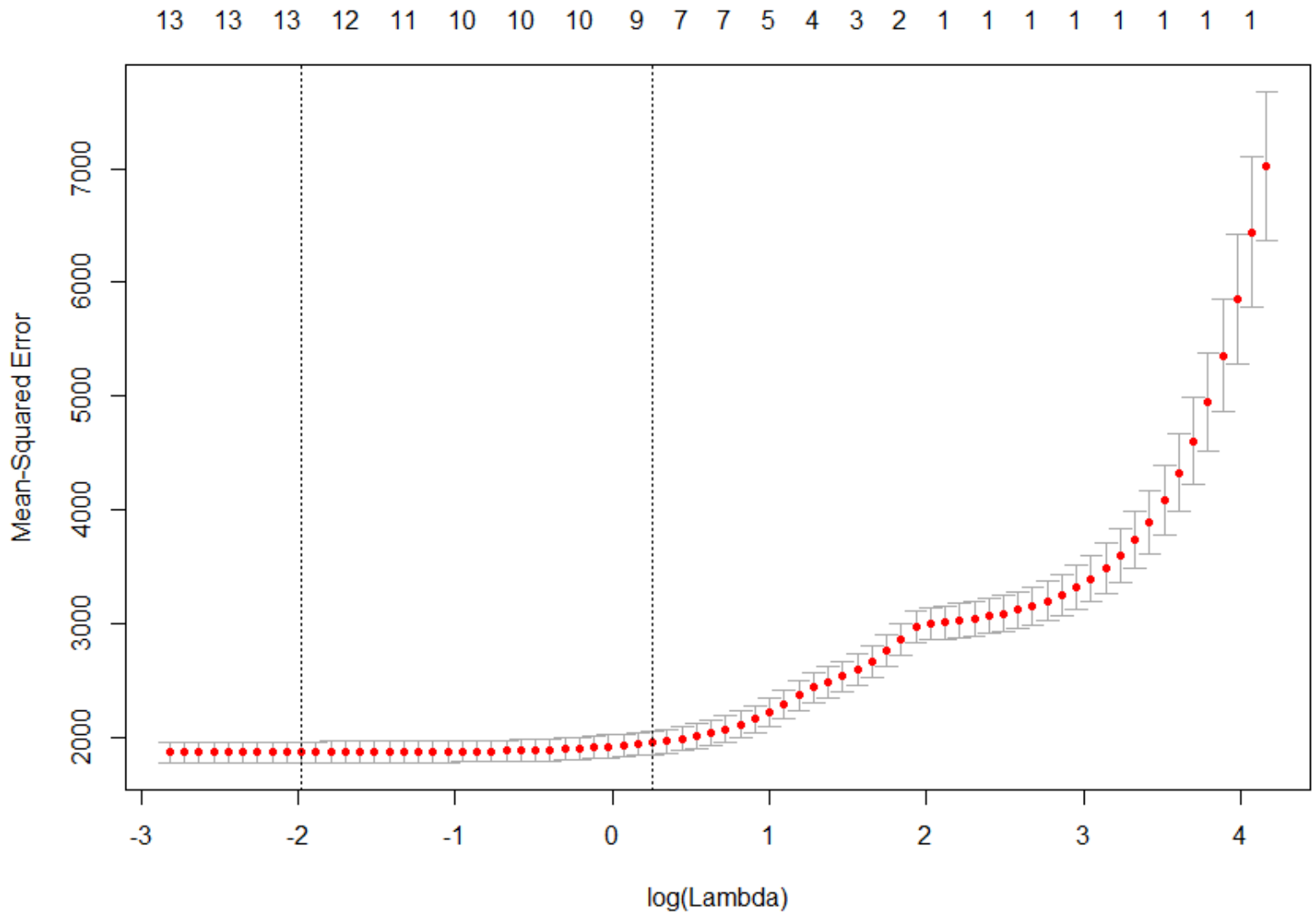
LASSO: Coefficients

$$\min_{\beta}\left(\sum_{j=1}^{N}(y_j-\sum_{i=0}^{d}\beta_i\cdot x_i)^2+\lambda\sum_{i=1}^{d}|\beta_i|\right)$$



CSE 7302c

LASSO: MSE



CSE 7302c

Ridge vs Lasso Regression

Ridge	LASSO
• Reduces the magnitude of coefficients (does not make them zero) • Solution is unique – however, doesn't do feature selection since it keeps all the terms • Stable even if $n<<p$ • Very stable even in multicollinear case	• Values of irrelevant coefficients set to zero • Good for feature selection • Doesn't do well when $n<<p$ • Out of sample performance can be impacted for multicollinear case

CSE 7302c

Elastic-net

- A mix-between Ridge and LASSO regression

$$\min_{\beta}\left(\sum_{j=1}^{N}(y_j-\sum_{i=0}^{d}\beta_i\cdot x_i)^2+\lambda\left(\sum_{i=1}^{d}(\alpha|\beta_i|+(1-\alpha)\beta_i^2)\right)\right)$$

- α=0 is Ridge
- α=1 is LASSO
- α=0.5 is a equal mix of the two

CSE 7302c

- Keep in mind that Ridge regression can't zero out coefficients; thus, you either end up including all the coefficients in the model, or none of them.

- In contrast, the LASSO does both parameter shrinkage and variable selection automatically.

- If some of your covariates are highly correlated, you may want to look at the Elastic Net instead of the LASSO.

CSE 7302c

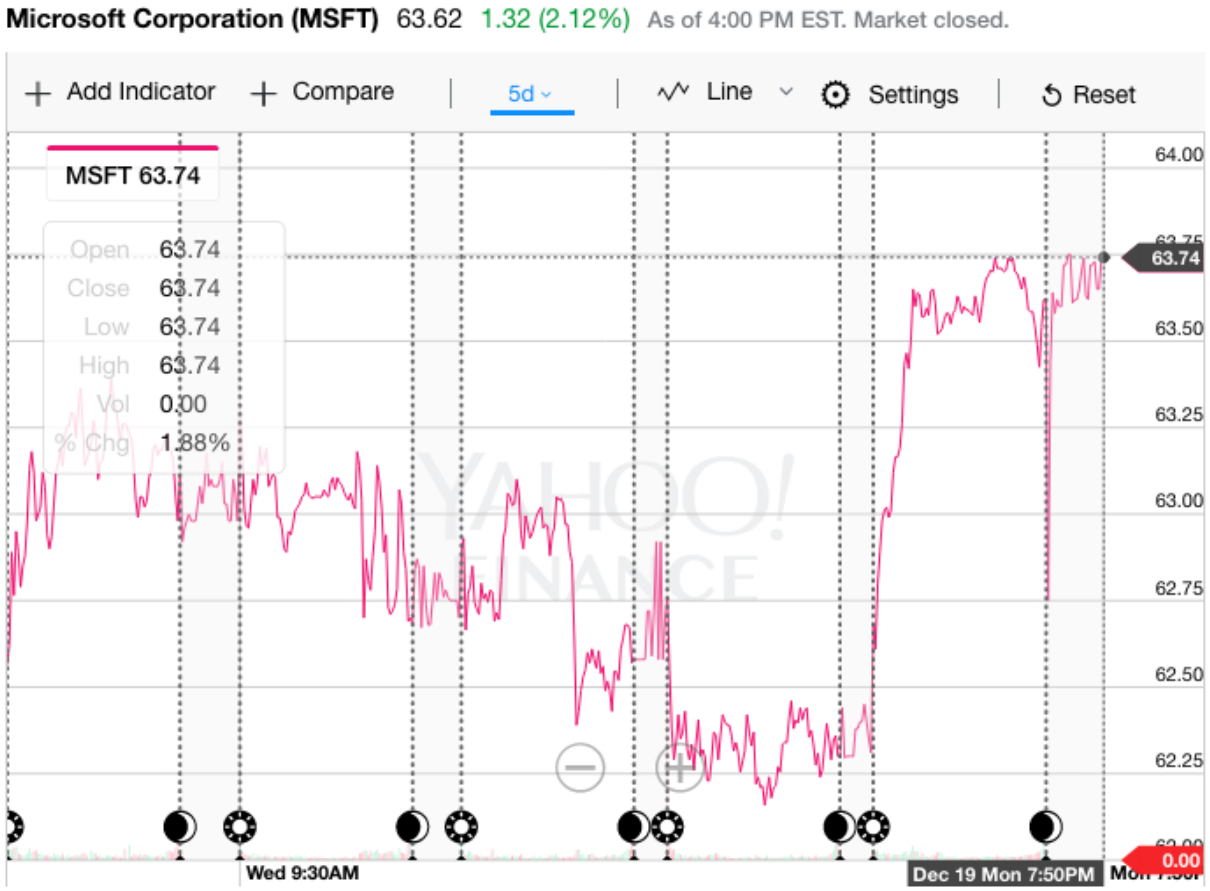
TIME SERIES FORECASTING

CSE 7302c

Why Time Series

Causal independent variables are

- Unknown to us
- Not available
- Might not fit the data well
- Difficult to forecast



Typical Time Series

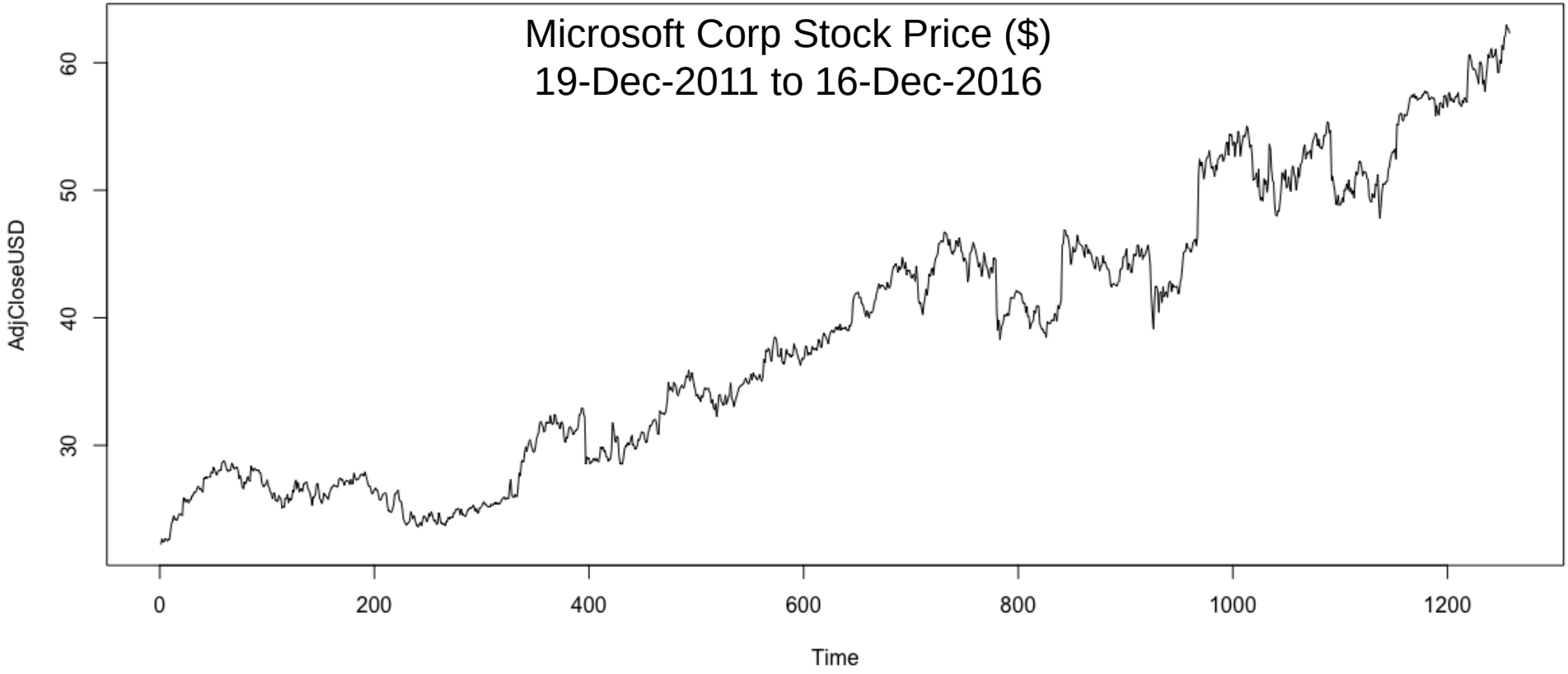
$$\hat{y}_{t+1} = f(y_t, y_{t-1}, y_{t-2} \ldots) + f(x_1, x_2, x_3 \ldots)$$

f can be linear or nonlinear

CSE 7302c

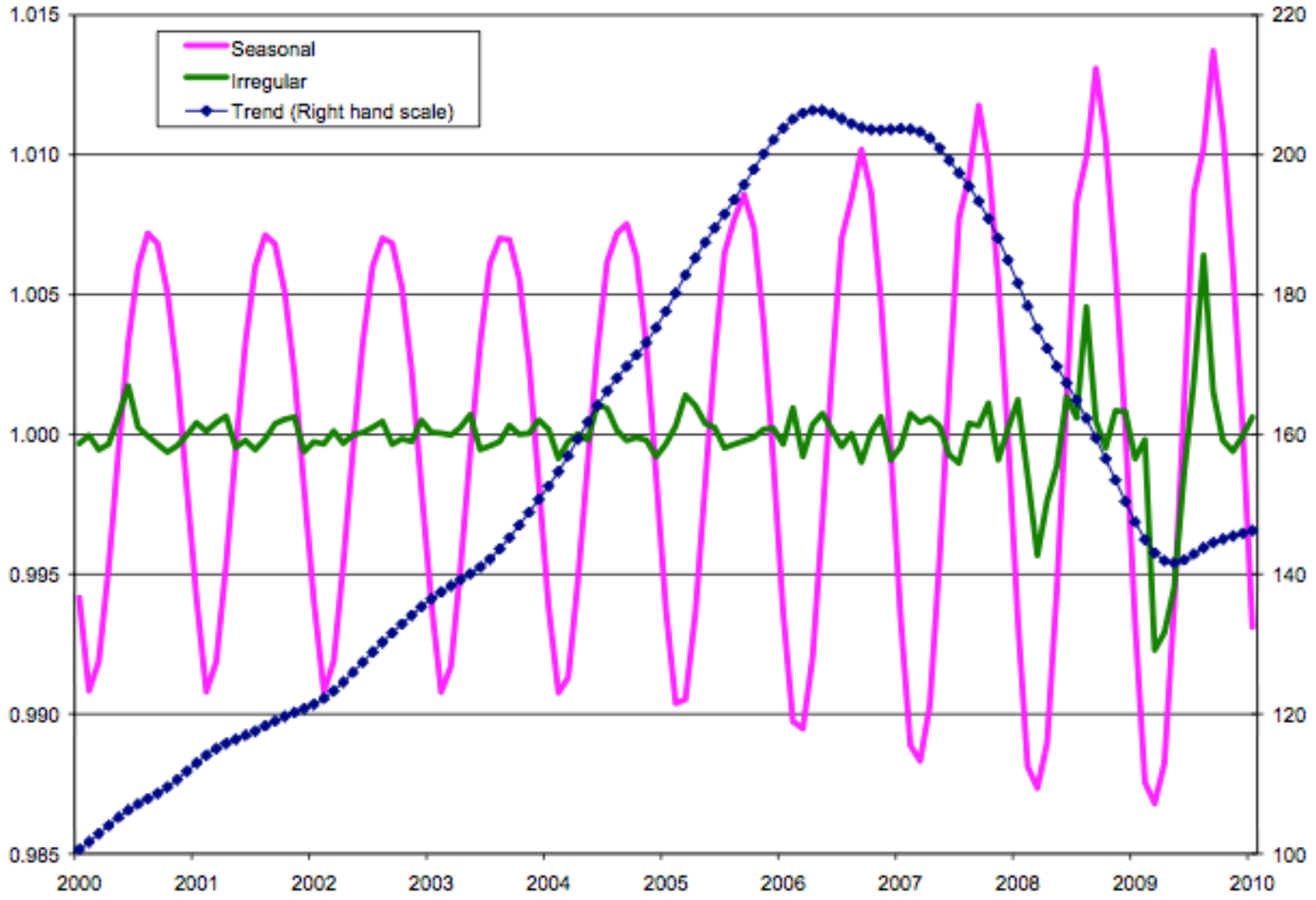
Components of Time Series

- Trend
- Seasonality
- Random component



CSE 7302c

Trend, Seasonality and Randomness

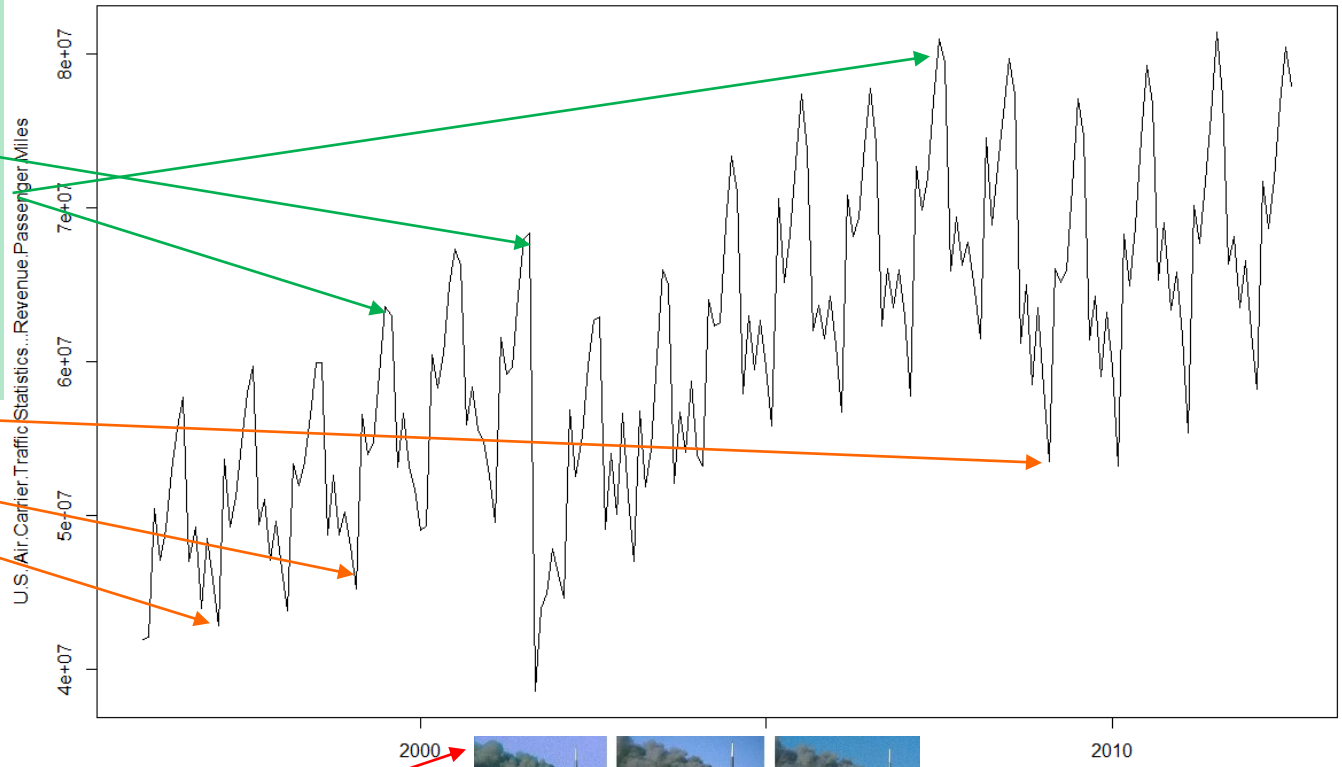


US Air Carrier Traffic – Revenue Passenger Miles ('000)

R code: milestimeseries <- ts(miles, frequency = 12, start = c(1996,1))

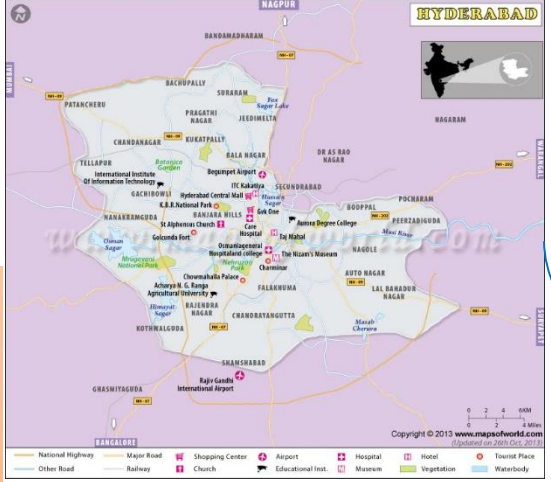
RPM

```
> milestimeseries <- ts(miles, frequency = 12, start = c(1996,1))
> milestimeseries
          Jan      Feb      Mar      Apr      May      Jun      Jul
1996 41972194 42054796 50443045 47112397 49118248 52880510 55664750
1997 45850623 42838949 53620994 49282817 51191842 54707221 57995025
1998 46514139 43769273 53361926 51968480 53515798 56460422 59939170
1999 47988560 45241211 56555731 53920855 54674958 59213000 63572248
2000 49045412 49306303 60443541 58286680 60533783 64903295 67346377
2001 52634354 49532578 61575055 59151645 59662416 64353323 67965298
2002 46224031 44615129 56897729 52542164 55116060 59745343 62664511
2003 51197175 47040806 56766580 51857453 54335598 60272900 65962215
2004 53979786 53179693 64035864 62340117 62530704 68866398 73335888
2005 59629608 55795165 70595861 65145552 68268899 72952959 77432998
2006 61035027 56729212 70799794 68120559 69352606 74099239 77798621
2007 63016013 57793832 72700241 69836156 71933109 76926452 80988340
2008 64667106 61504426 74575531 68906882 72725750 76162105 79707545
2009 58373786 53506580 66027341 65166300 65868254 71350227 77136799
2010 59651061 53240066 68307090 64953250 68850904 74474550 79304441
2011 61630362 55391206 70158268 67683558 71711448 76057910 81423215
2012 61940180 58243763 71696039 68669228 71887523 76760759 80499353
          Aug      Sep      Oct      Nov      Dec
1996 57723208 47035464 49263120 43937074 48539606
1997 59715433 49418190 51058879 47056048 49654209
```

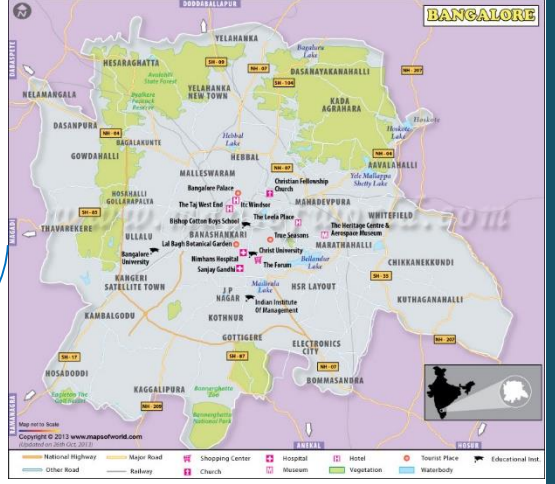


Data sources:
http://www.bts.gov/xml/air_traffic/src/index.xml
and https://datamarket.com/data/set/281x/us-air-carrier-traffic-statistics-revenue-passenger-miles
Last accessed: 31-Mar-2016

	Aug	Sep	Oct	Nov	Dec
1996	57723208	47035464	49263120	43937074	48539606
1997	59715433	49418190	51058879	47056048	49654209
1998	59927214	48751280	52578217	48734375	50208641
1999	63003663	53131972	56653901	53215500	51746821
2000	66256804	55900504	58373996	55590325	54822970
2001	68377080	38601868	43964788	44915764	47836501
2002	62944816	49096035	54019748	50106814	56656594
2003	64989766	52121480	56724551	54128776	58739845
2004	70961522	57881042	63021142	59453943	62680310



Inspire...Educate...Transform.



HYDERABAD

2nd Floor, Jyothi Imperial, Vamsiram Builders, Old Mumbai Highway, Gachibowli, Hyderabad - 500 032
+91-9701685511 (Individuals)
+91-9618483483 (Corporates)

BENGALURU

Floors 1-3, L77, 15th Cross Road, 3A Main Road, Sector 6, HSR Layout, Bengaluru – 560 102
+91-9502334561 (Individuals)
+91-9502799088 (Corporates)

Social Media

Web:	http://www.insofe.edu.in
Facebook:	https://www.facebook.com/insofe
Twitter:	https://twitter.com/Insofeedu
YouTube:	http://www.youtube.com/InsofeVideos
SlideShare:	http://www.slideshare.net/INSOFE
LinkedIn:	http://www.linkedin.com/company/international-school-of-engineering

This presentation may contain references to findings of various reports available in the public domain. INSOFE makes no representation as to their accuracy or that the organization subscribes to those findings.