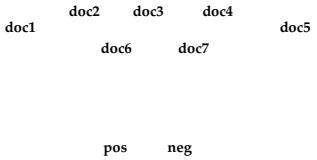
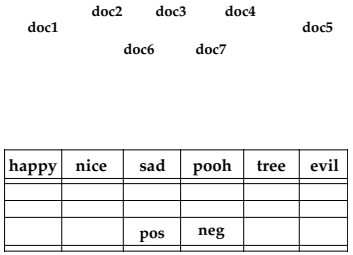
# From Latent Features to Word Embeddings

*Lecture 11: Text Analytics for Big Data*
*Mark Keane, Insight/CSI, UCD*

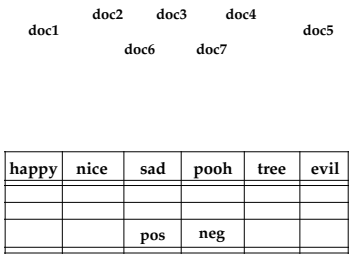
# Intuitive Idea



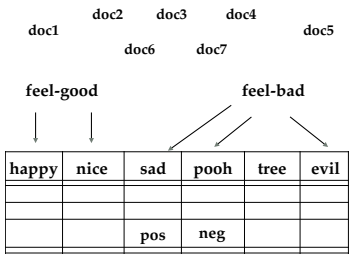

**Imagine *Model-X* that maps word features of docs to sentiment features**



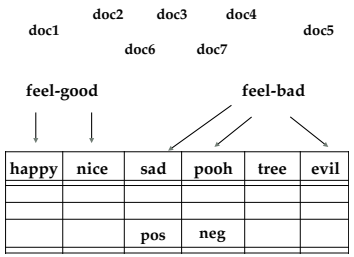

**Model-X** tells if docs are positive or negative;
based on the word-feature in the docs...




But, **Model-X** does not tell us anything deeper about these feature-dimensions; latent/hidden features

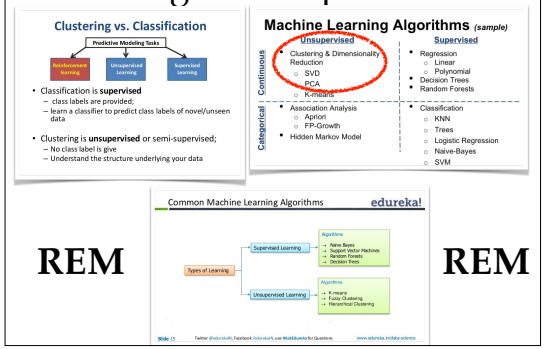



**Latent features** like a feel-good or a feel-bad feature...

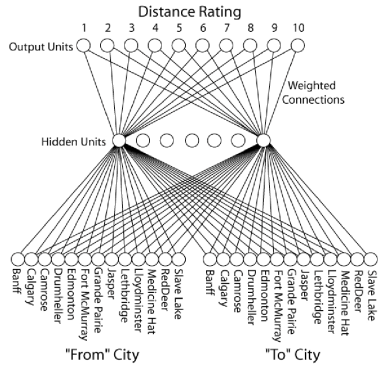



**Methods** exist to find such latent features;
basically, factor-analytic techniques that are unsupervised ML

# Clustering is Unsupervised…




**REM** **REM**




NN Analogous: localist v globalist; coarse coding; hidden feature = dim_red

# Buzz Words & Terminology

- Discovering Latent/Hidden Features, Factors
- Dimensionality Reduction
- Word embeddings and Deep Learning

feature = dimension = variable

factor is linear combination of variables

factor interchangeable with component

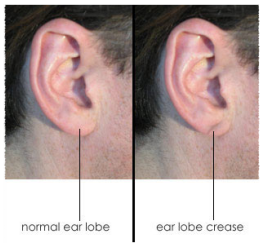
# Four (Linked) Techniques

- PCA: Principal Components Analysis
- SVD: Singular Value Decomposition
- Latent Stuff: LSA, LSI, pLSA, LDA…
- Wonderful World of Word Embeddings

**PCA**

# PCA: The Idea:

**Several low-level features may combine (as a factor / latent feature) to predict some outcome (e.g., smoking, ear-lobe folds, high-BMI may represent genetic sub-group that predicts early-stroke risk)**




**feature = dimension = variable**

**factor is linear combination of variables**

# PCA: The Idea:

- **In multivariate analysis, many *variables* (ie word-features) may be correlated (i.e., feel-good words) in their effect on some outcome/prediction (good/bad)**
- **PCA is a dimension-reduction tool that reduces a large set of *variables* to a smaller set of *factors* that still contains most of the information in the large set**
- **PCA mathematically transforms a (larger) number of (possibly) correlated *variables* into a (smaller) number of uncorrelated *factors* aka Principal Components**

ftp://statgen.ncsu.edu/pub/thorne/molevoclass/AtchleyOct19.pdf

**doc1** **doc2** **doc3** **doc4** **doc5** **doc6** **doc7**

**feel-good** **feel-bad** **tree**

| happy | nice | sad | pooh | evil | tree |
|---|---|---|---|---|---|
| | | | | | |
| | | | | | |
| | | pos | neg | | |

**feel-good words** will be correlated in their effects on the predicted outcome (**pos**), and **feel-bad words** in their effects on predicted outcome (**neg**); BUT the effects of **feel-good** and **feel-bad words** will be uncorrelated (because they are a different class of variables): Components/Factors = FEEL-GOOD/FEEL-BAD

# PCA: The Idea: II

- 1st Principal-Component accounts for most of the variance, the second the next next most and so on
- Components summarise the total variance of data
- NB: NO guarantee Components are interpretable
- Like Factor Analysis but actually different…

ftp://statgen.ncsu.edu/pub/thorne/molevoclass/AtchleyOct19.pdf

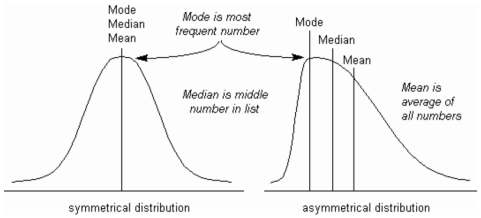

# PCA: Some Stats Nuts & Bolts

- Means & Deviations from it
- Standard Deviations & Variance
- Co-variance & Correlation
- The Matrix Step
- Eigenvectors & Eigenvalues

ftp://statgen.ncsu.edu/pub/thorne/molevoclass/AtchleyOct19.pdf

# I Really Means It !

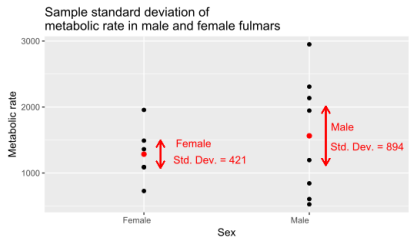
Mean (arithmetic) is the central value of a discrete set of numbers
Got by the sum of the values divided by the number of values




https://en.wikipedia.org/wiki/Mean

# Means & Deviations

Normalised spread of scores around the mean (ie standardised)




https://en.wikipedia.org/wiki/Mean

# Standard Deviation

$$s_x = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n-1}}$$

$n$ = The number of data points
$\bar{x}$ = The mean of the $x_i$
$x_i$ = Each of the values of the data

- **SD** is standardised score of the spread of numbers around the **mean**
- **SD** is a measure that quantifies the amount of variation or dispersion in a set of values
- N or N-1 = pop v sample

$$\sigma = \sqrt{\frac{\sum (x - \bar{x})^2}{n}}$$

$\sigma$ = standard deviation
$\sum$ = sum of
$x$ = each value in the data set
$\bar{x}$ = mean of all values in the data set
$n$ = number of value in the data set

https://en.wikipedia.org/wiki/Standard_deviation

# Standard Deviation

Calculating Standard Deviation (sample)

$$s = \sqrt{\frac{\sum (Xi - \bar{X})^2}{n-1}}$$ ← SS

Step 1: Calculate Sum of Squares

**SS = $\Sigma X^2 - (\Sigma X)^2 / N$**

Square each score and then add up all the squared scores | Add up all the scores, square the sum, divide by the sample size

Step 2: Plug in SS (what you got above) into the numerator of the standard deviation formula

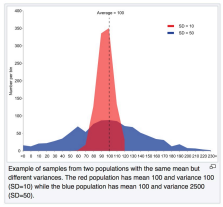
$$s = \sqrt{\frac{SS}{n-1}}$$

Step 3: Interpretation (On average, observations in the sample are (insert standard deviation) units away from the sample mean of (insert mean).

https://en.wikipedia.org/wiki/Standard_deviation

# Variance:

**Variance** not standardised but still a useful measure; sometimes




Example of samples from two populations with the same mean but different variances. The red population has mean 100 and variance 100 (SD=10) while the blue population has mean 100 and variance 2500 (SD=50).

**Sample Variance**

$$s^2 = \frac{\sum (x - \bar{x})^2}{n-1}$$

**Sample Standard Deviation**

$$s = \sqrt{\frac{\sum (x - \bar{x})^2}{n-1}}$$

https://en.wikipedia.org/wiki/Variance

# But: Double Down Donald

We have seen a single set of figures
What about two sets of figures; two measures the same items (e.g. people)

| | *Hours(H)* | *Mark(M)* |
|---|---|---|
| Data | 9 | 39 |
| | 15 | 56 |
| | 25 | 93 |
| | 14 | 61 |
| | 10 | 50 |
| | 18 | 75 |
| | 0 | 32 |
| | 16 | 85 |
| | 5 | 42 |
| | 19 | 70 |
| | 16 | 66 |
| | 20 | 80 |
| Totals | 167 | 749 |
| Averages | 13.92 | 62.42 |

# Covariance…

- With two, we can ask do they vary together, in some way
- What about their Covariance?
- This is a measure of their co-variability

| | *Hours(H)* | *Mark(M)* |
|---|---|---|
| Data | 9 | 39 |
| | 15 | 56 |
| | 25 | 93 |
| | 14 | 61 |
| | 10 | 50 |
| | 18 | 75 |
| | 0 | 32 |
| | 16 | 85 |
| | 5 | 42 |
| | 19 | 70 |
| | 16 | 66 |
| | 20 | 80 |
| Totals | 167 | 749 |
| Averages | 13.92 | 62.42 |

$$var(X) = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(X_i - \bar{X})}{(n-1)}$$

$$cov(X,Y) = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{(n-1)}$$

# Covariance Matrix: For 2

Can compute the covariance of two variables to each other

…and to themselves

…and put them in a *covariance matrix*

| Student | Hours Studied | Exam Mark |
|---|---|---|
| John | 9 | 39 |
| Jimmy | 15 | 56 |
| Mary | 25 | 93 |
| Jane | 14 | 61 |
| Aditi | 10 | 50 |
| Tarquin | 18 | 75 |
| Ashok | 0 | 32 |
| Plembo | 16 | 85 |
| Mark | 5 | 42 |
| Derek | 19 | 70 |
| Torens | 16 | 66 |
| Ibble | 20 | 80 |
| **Mean** | *13.92* | *62.42* |
| **Sum** | *167* | *749* |
| **N** | *12* | *12* |

| | | |
|---|---|---|
| **Covar_S (Hs x Em)** | 122.95 | |
| **Covar_S (Hs x Hs)** | 47.72 | **aka Var(Hs)** |
| **Covar_S (Em x Em)** | 370.08 | **aka Var(Em)** |

| | HoursStudied | ExamMark |
|---|---|---|
| HoursStudied | 47.72 | 122.95 |
| ExamMark | 122.95 | 370.08 |

**COVARIANCE MARIX FOR TWO VARIABLES**

# Covariance Matrix: For 3

Can compute the covariance of three variables to each other

…and to themselves

…and put them in a *covariance matrix*

| Student | Hours Studied | Exam Mark | Height |
|---|---|---|---|
| John | 9 | 39 | 112 |
| Jimmy | 15 | 56 | 118 |
| Mary | 25 | 93 | 120 |
| Jane | 14 | 61 | 100 |
| Aditi | 10 | 50 | 93 |
| Tarquin | 18 | 75 | 145 |
| Ashok | 0 | 32 | 167 |
| Plembo | 16 | 85 | 112 |
| Mark | 5 | 42 | 111 |
| Derek | 19 | 70 | 102 |
| Torens | 16 | 66 | 115 |
| Ibble | 20 | 80 | 116 |
| **Mean** | *13.92* | *62.42* | *117.58* |
| **Sum** | *167* | *749* | *1411* |
| **N** | *12* | *12* | *12* |

| | | |
|---|---|---|
| **Covar_S (Hs x Em)** | 122.95 | |
| **Covar_S (Hs x He)** | -43.04 | |
| **Covar_S (Em x He)** | -70.27 | |
| **Covar_S (Hs x Hs)** | 47.72 | **aka Var(Hs)** |
| **Covar_S (Em x Em)** | 370.08 | **aka Var(Em)** |
| **Covar_S (He x He)** | 406.45 | **aka Var(He)** |

| | HoursStudied | ExamMark | Height |
|---|---|---|---|
| HoursStudied | 47.72 | 122.95 | -43.04 |
| ExamMark | 122.95 | 370.08 | -70.27 |
| Height | -43.04 | -70.27 | 406.45 |

**COVARIANCE MARIX FOR THREE VARIABLES**

# Correlation

- With two, we can also ask do they *co-relate*
- Do they correspond in the way the change
- Pearson's *r* is a measure of correlation

| | Hours(H) | Mark(M) |
|---|---|---|
| Data | 9 | 39 |
| | 15 | 56 |
| | 25 | 93 |
| | 14 | 61 |
| | 10 | 50 |
| | 18 | 75 |
| | 0 | 32 |
| | 16 | 85 |
| | 5 | 42 |
| | 19 | 70 |
| | 16 | 66 |
| | 20 | 80 |
| Totals | 167 | 749 |
| Averages | 13.92 | 62.42 |

$$r = \frac{\sum (x-\bar{x})(y-\bar{y})}{\sqrt{\sum (x-\bar{x})^2 \sum (y-\bar{y})^2}}$$

$$r = \frac{n(\Sigma xy) - (\Sigma x)(\Sigma y)}{\sqrt{[\,n\Sigma x^2 - (\Sigma x)^2\,]\,[\,n\Sigma y^2 - (\Sigma y)^2\,]}}$$

# Correlation Matrix: For 3

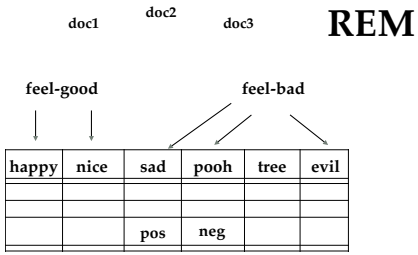
- We can also compute the correlation of two/ more variables to one another
- …and to themselves
- …and put them in a *correlation matrix*

| Student | Hours Studied | Exam Mark | Height |
|---|---|---|---|
| John | 9 | 39 | 112 |
| Jimmy | 15 | 56 | 118 |
| Mary | 25 | 93 | 120 |
| Jane | 14 | 61 | 100 |
| Aditi | 10 | 50 | 93 |
| Tarquin | 18 | 75 | 145 |
| Ashok | 0 | 32 | 167 |
| Plembo | 16 | 85 | 112 |
| Mark | 5 | 42 | 111 |
| Derek | 19 | 70 | 102 |
| Torens | 16 | 66 | 115 |
| Ibble | 20 | 80 | 116 |
| **Mean** | *13.92* | *62.42* | *117.58* |
| **Sum** | *167* | *749* | *1411* |
| **N** | *12* | *12* | *12* |

| | |
|---|---|
| **r (Hs x Em)** | 0.93 |
| **r (Hs x He)** | -0.31 |
| **r (Em x He)** | -0.18 |
| **r (Hs x Hs)** | 1.00 |
| **r (Em x Em)** | 1.00 |
| **r (He x He)** | 1.00 |

| | HoursStudied | ExamMark | Height |
|---|---|---|---|
| HoursStudied | 1.00 | 0.93 | -0.31 |
| ExamMark | 0.93 | 1.00 | -0.18 |
| Height | -0.31 | -0.18 | 1.00 |

**CORRELATION MARIX FOR THREE VARIABLES**




**Methods** exist to find such latent features
**by examining patterns of covariance and correlation**

# Like This…

- We can compute the correlation of two variables to one another
- …and to themselves
- …and put them in a *correlation matrix*

| | doc1 | doc2 | doc3 |
|---|---|---|---|
| happy | 10 | 12 | 1 |
| nice | 16 | 14 | 0 |
| sad | 2 | 2 | 10 |
| pooh | 1 | 1 | 9 |
| evil | 1 | 1 | 12 |
| tree | 1 | 1 | 1 |
| dog | 1 | 0 | 0 |
| cat | 0 | 1 | 0 |
| donkey | 5 | 5 | 5 |
| spigot | 1 | 1 | 1 |
| **Mean** | *3.80* | *3.80* | *4.40* |
| **Sum** | *38* | *38* | *44* |
| **N** | *10* | *10* | *10* |

| | |
|---|---|
| **r (doc 1 x doc2** | 0.98 |
| **r (doc 1 x doc3)** | -0.30 |
| **r (doc 2 x doc3** | -0.30 |
| **r (doc1 x doc1)** | 1.00 |
| **r (doc2 x doc2)** | 1.00 |
| **r (doc3 x doc3)** | 1.00 |

| | doc1 | doc2 | doc3 |
|---|---|---|---|
| doc1 | 1.00 | 0.98 | -0.30 |
| doc2 | 0.98 | 1.00 | -0.30 |
| doc3 | -0.30 | -0.30 | 1.00 |

**CORRELATION MARIX FOR THREE VARIABLES**

# Oh…and…eigen-whaaaa?

Will all be done using matrix multiplication

and the like…

finding factors to "explain" the variance in

these matrices

…matrices produced by multiplying original

by transposed ones…

$A^T.A$ and $A.A^T$

# Eigenvectors & Eigenvalues

- Given an input matrix **X**, an eigenvector of the matrix is a non-zero vector **v** that satisfies the equation:

$$\mathbf{X}v = \lambda v$$

where the corresponding number $\lambda$ is called an eigenvalue.

- Eigendecomposition is the factorisation of a matrix into its eigenvalues and eigenvectors.

$$\mathbf{X} = \begin{pmatrix} 1.0000 & 0.5000 & 0.3330 & 0.2500 \\ 0.5000 & 1.0000 & 0.6667 & 0.5000 \\ 0.3333 & 0.6667 & 1.0000 & 0.7500 \\ 0.2500 & 0.5000 & 0.7500 & 1.0000 \end{pmatrix}$$

4 x 4 symmetric

$$\Lambda = \begin{pmatrix} 2.5361 & 0 & 0 & 0 \\ 0 & 0.8483 & 0 & 0 \\ 0 & 0 & 0.4078 & 0 \\ 0 & 0 & 0 & 0.2077 \end{pmatrix}$$

4

Eigenvectors and values exist in pairs: every eigenvector has a corresponding eigenvalue.

$$\mathbf{V} = \begin{pmatrix} -0.37775 & -0.81052 & -0.44217 & -0.06988 \\ -0.53223 & -0.18762 & 0.74199 & 0.36221 \\ -0.56139 & 0.30099 & 0.04872 & -0.76926 \\ -0.50881 & 0.46611 & -0.50155 & 0.52169 \end{pmatrix}$$

4

From D. Greene's course…

- Each eigenvector has a direction - i.e. it is a dimension.
- Eigenvectors of symmetric matrices are orthogonal to each other - i.e. they point in completely different directions.
- Each eigenvalue is a number indicating how much variance there is in the data in that direction.
- The eigenvector with the largest eigenvalue has the most variance, making it a useful dimension for projection, and so on…

- Principal Components (PCs): New dimensions constructed as linear combinations of the original features, which are uncorrelated with one another. Constructed from eigenvectors.


# PCA: The Idea:

**REM**

- In multivariate analysis, many *variables* (ie word-features) may be correlated (i.e., feel-good words) in their effect on some outcome/prediction (good/bad)
- PCA is a dimension-reduction tool that reduces a large set of *variables* to a smaller set of *factors* that still contains most of the information in the large set
- PCA mathematically transforms a (larger) number of (possibly) correlated *variables* into a (smaller) number of uncorrelated *factors* aka Principal Components

ftp://statgen.ncsu.edu/pub/thorne/molevoclass/AtchleyOct19.pdf

**doc1** **doc2** **doc3** **doc4** **doc5** **doc6** **doc7**

**REM**

**feel-good** **feel-bad** **tree**

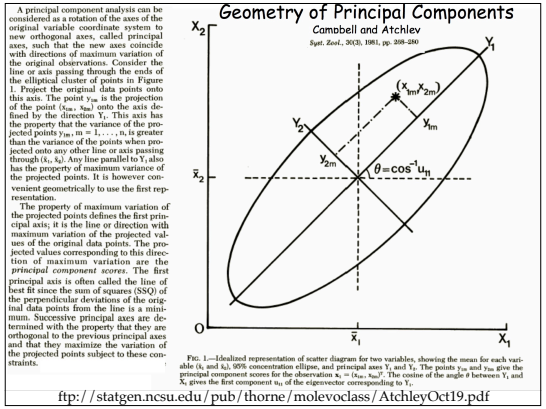
| happy | nice | sad | pooh | evil | tree |
|---|---|---|---|---|---|
| | | | | | |
| | | | | | |
| | | pos | neg | | |

**feel-good words** will be correlated in their effects on the predicted outcome (**pos**), and **feel-bad words** in their effects on predicted outcome (**neg**); BUT the effects of **feel-good** and **feel-bad words** will be uncorrelated (because they are a different class of variables): Components/Factors = FEEL-GOOD/FEEL-BAD

# PCA: The Idea: II **REM**

- 1st Principal-Component accounts for most of the variance, the 2nd the next most and so on
- Components summarise total variance of data
- No guarantee Components are interpretable
- Components reduce dimensionality of data

ftp://statgen.ncsu.edu/pub/thorne/molevoclass/AtchleyOct19.pdf



A principal component analysis can be considered as a rotation of the axes of the original variable coordinate system to new orthogonal axes, called principal axes, such that the new axes coincide with directions of maximum variation of the original observations. Consider the line or axis passing through the ends of the elliptical cluster of points in Figure 1. Project the original data points onto this axis. The point $y_{1m}$ is the projection of the point $(x_{1m}, x_{2m})$ onto the axis defined by the direction $Y_1$. This axis has the property that the variance of the projected points $y_{1m}$, $m = 1, \ldots, n$, is greater than the variance of the points when projected onto any other line or axis passing through $(\bar{x}_1, \bar{x}_2)$. Any line parallel to $Y_1$ also has the property of maximum variance of the projected points. It is however convenient geometrically to use the first representation.

The property of maximum variation of the projected points defines the first principal axis; it is the line or direction with maximum variation of the projected values of the original data points. The projected values corresponding to this direction of maximum variation are the *principal component scores*. The first principal axis is often called the line of best fit since the sum of squares (SSQ) of the perpendicular deviations of the original data points from the line is a minimum. Successive principal axes are determined with the property that they are orthogonal to the previous principal axes and that they maximize the variation of the projected points subject to these constraints.

Geometry of Principal Components

Campbell and Atchley

*Syst. Zool.*, 30(3), 1981, pp. 268–280



FIG. 1.—Idealized representation of scatter diagram for two variables, showing the mean for each variable ($\bar{x}_1$ and $\bar{x}_2$), 95% concentration ellipse, and principal axes $Y_1$ and $Y_2$. The points $y_{1m}$ and $y_{2m}$ give the principal component scores for the observation $\mathbf{x}_m = (x_{1m}, x_{2m})'$. The cosine of the angle $\theta$ between $Y_1$ and $X_1$ gives the first component $u_{11}$ of the eigenvector corresponding to $Y_1$.

ftp://statgen.ncsu.edu/pub/thorne/molevoclass/AtchleyOct19.pdf

# PCA: The Maths Bit

PCA is performed on a square, symmetric matrix:

- **Covariance Matrix**: based on comparing the variance between two variables
- **Correlation Matrix**: based on correlating two variables (e.g., Pearson's correlation)
- NB: these are related to one another: CorrMatrix is scaled version of CovMatrix

ftp://statgen.ncsu.edu/pub/thorne/molevoclass/AtchleyOct19.pdf

# PCA: Steps…

- Step1: Subtract means from raw data
- Step2: Do COV/CORR matrix for variables
- Step3: Find matrix eigenvectors+eigenvalues
- Step4: Order eigenvector(s) from high->low
- Step5*: Apply & Re-plot data

* **Optional; eigenvectors adopted as new representation of data**

http://www.iro.umontreal.ca/~pift6080/H09/documents/papers/pca_tutorial.pdf

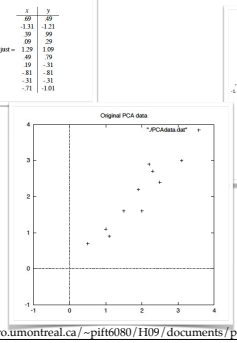
# PCA: Step 1: Subtract Means

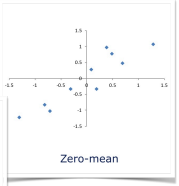
| x | y |
|---|---|
| 2.5 | 2.4 |
| 0.5 | 0.7 |
| 2.2 | 2.9 |
| 1.9 | 2.2 |
| 3.1 | 3.0 |
| 2.3 | 2.7 |
| 2 | 1.6 |
| 1 | 1.1 |
| 1.5 | 1.6 |
| 1.1 | 0.9 |

Data =

| x | y |
|---|---|
| .69 | .49 |
| -1.31 | -1.21 |
| .39 | .99 |
| .09 | .29 |
| 1.29 | 1.09 |
| .49 | .79 |
| .19 | -.31 |
| -.81 | -.81 |
| -.31 | -.31 |
| -.71 | -1.01 |

DataAdjust =



Zero-mean

http://www.iro.umontreal.ca/~pift6080/H09/documents/papers/pca_tutorial.pdf

# PCA: Step 2: Covariance Matrix

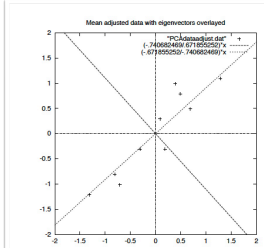
$$cov = \begin{pmatrix} .616555556 & .615444444 \\ .615444444 & .716555556 \end{pmatrix}$$

- As the non-diagonal elements in this covariance matrix are positive, we should expect that both the x and y variable increase together
- Aka positively, not negatively correlated

.

http://www.iro.umontreal.ca/~pift6080/H09/documents/papers/pca_tutorial.pdf

# PCA: Step 3: Get Eigen…Stuff




$$eigenvalues = \begin{pmatrix} .0490833989 \\ 1.28402771 \end{pmatrix}$$

$$eigenvectors = \begin{pmatrix} -.735178656 & -.677873399 \\ .677873399 & -.735178656 \end{pmatrix}$$

A·v=λ·v. In this equation A is an n-by-n **matrix**, v is a non-zero n-by-1 vector and λ is a scalar (which may be either real or complex). Any value of λ for which this equation has a solution is known as an **eigenvalue** of the **matrix** A. It is sometimes also called the characteristic value.

a scalar λ is called an **eigenvalue** of the n × n matrix A if there is a nontrivial solution x of Ax = λx. Such an x is called an **eigenvector** corresponding to the **eigenvalue** λ.

Figure 3.2: A plot of the normalised data (mean subtracted) with the eigenvectors of the covariance matrix overlayed on top.

# PCA: Step 3: Get Eigen…Stuff

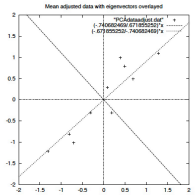



Figure 3.2: A plot of the normalised data (mean subtracted) with the eigenvectors of the covariance matrix overlayed on top.

- eigenvectors are plotted as diagonal dotted lines on the plot.
- Note they are perpendicular to each other.
- Note one of the eigenvectors goes through the middle of the points, like drawing a line of best fit.
- The second eigenvector gives us the other, less important, pattern in the data, that all the points follow the main line, but are off to the side of the main line by some amount.

http://www.iro.umontreal.ca/~pift6080/H09/documents/papers/pca_tutorial.pdf

# PCA: Step 4: Order

$$eigenvalues = \begin{pmatrix} .0490833989 \\ 1.28402771 \end{pmatrix}$$

$$eigenvectors = \begin{pmatrix} -.735178656 & -.677873399 \\ .677873399 & -.735178656 \end{pmatrix}$$

**order**

- Order eigenvectors to apply them
- By highest-lowest eigenvalue
- Choose eigenvectors to use

$$FeatureVector = (eig_1\ eig_2\ eig_3\ ....\ eig_n)$$

Given our example set of data, and the fact that we have 2 eigenvectors, we have two choices. We can either form a feature vector with both of the eigenvectors:

$$\begin{pmatrix} -.677873399 & -.735178656 \\ -.735178656 & .677873399 \end{pmatrix}$$

or, we can choose to leave out the smaller, less significant component and only have a single column:

$$\begin{pmatrix} -.677873399 \\ -.735178656 \end{pmatrix}$$

http://www.iro.umontreal.ca/~pift6080/H09/documents/papers/pca_tutorial.pdf

# PCA: Step 4: Order

$$eigenvalues = \begin{pmatrix} .0490833989 \\ 1.28402771 \end{pmatrix}$$

$$eigenvectors = \begin{pmatrix} -.735178656 & -.677873399 \\ .677873399 & -.735178656 \end{pmatrix}$$

- Order eigenvectors to apply them
- By highest-lowest eigenvalue
- Choose eigenvectors to use

$$FeatureVector = (eig_1\ eig_2\ eig_3\ ....\ eig_n)$$

Given our example set of data, and the fact that we have 2 eigenvectors, we have two choices. We can either form a feature vector with both of the eigenvectors:

$$\begin{pmatrix} -.677873399 & -.735178656 \\ -.735178656 & .677873399 \end{pmatrix}$$ **choose**

or, we can choose to leave out the smaller, less significant component and only have a single column:

$$\begin{pmatrix} -.677873399 \\ -.735178656 \end{pmatrix}$$

http://www.iro.umontreal.ca/~pift6080/H09/documents/papers/pca_tutorial.pdf

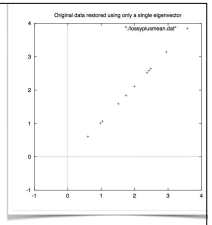
# PCA Step 5:Apply 1

| x | y |
|---|---|
| 2.5 | 2.4 |
| 0.5 | 0.7 |
| 2.2 | 2.9 |
| 1.9 | 2.2 |
| 3.1 | 3.0 |
| 2.3 | 2.7 |
| 2 | 1.6 |
| 1 | 1.1 |
| 1.5 | 1.6 |
| 1.1 | 0.9 |

Data =

| x | y |
|---|---|
| .69 | .49 |
| -1.31 | -1.21 |
| .39 | .99 |
| .09 | .29 |
| 1.29 | 1.09 |
| .49 | .79 |
| .19 | -.31 |
| -.81 | -.81 |
| -.31 | -.31 |
| -.71 | -1.01 |

DataAdjust =




**multiplied-by**

$$\begin{pmatrix} -.677873399 \\ -.735178656 \end{pmatrix}$$

Transformed Data (Single eigenvector)

| x' |
|---|
| -.827970186 |
| 1.77758033 |
| -.992197494 |
| -.274210416 |
| -1.67580142 |
| -.912949103 |
| .0991094375 |
| 1.14457216 |
| .438046137 |
| 1.22382056 |

http://www.iro.umontreal.ca/~pift6080/H09/documents/papers/pca_tutorial.pdf

# PCA Step 5:Apply 1

| x | y |
|---|---|
| .69 | .49 |
| -1.31 | -1.21 |
| .39 | .99 |
| .09 | .29 |
| 1.29 | 1.09 |
| .49 | .79 |
| .19 | -.31 |
| -.81 | -.81 |
| -.31 | -.31 |
| -.71 | -1.01 |

DataAdjust =

$$\begin{pmatrix} -.677873399 \\ -.735178656 \end{pmatrix} =$$

Transformed Data (Single eigenvector)

| x' |
|---|
| -.827970186 |
| 1.77758033 |
| -.992197494 |
| -.274210416 |
| -1.67580142 |
| -.912949103 |
| .0991094375 |
| 1.14457216 |
| .438046137 |
| 1.22382056 |

10X2 Matrix multiplied by 2x1 vector, giving a 10 x 1 vector

http://www.iro.umontreal.ca/~pift6080/H09/documents/papers/pca_tutorial.pdf

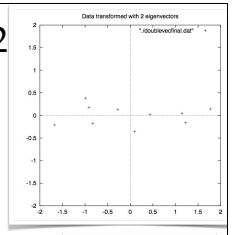
# PCA Step 5:Apply2

| x | y |
|---|---|
| 2.5 | 2.4 |
| 0.5 | 0.7 |
| 2.2 | 2.9 |
| 1.9 | 2.2 |
| 3.1 | 3.0 |
| 2.3 | 2.7 |
| 2 | 1.6 |
| 1 | 1.1 |
| 1.5 | 1.6 |
| 1.1 | 0.9 |

Data =

| x | y |
|---|---|
| .69 | .49 |
| -1.31 | -1.21 |
| .39 | .99 |
| .09 | .29 |
| 1.29 | 1.09 |
| .49 | .79 |
| .19 | -.31 |
| -.81 | -.81 |
| -.31 | -.31 |
| -.71 | -1.01 |

DataAdjust =




multiplied-by

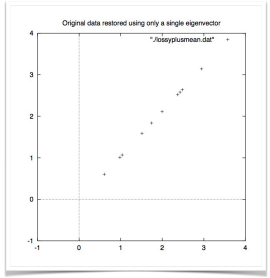
$$\begin{pmatrix} -.677873399 & -.735178656 \\ -.735178656 & .677873399 \end{pmatrix}$$

| x' | y' |
|---|---|
| -.827970186 | -.175115307 |
| 1.77758033 | .142857227 |
| -.992197494 | .384374989 |
| -.274210416 | .130417207 |
| -1.67580142 | -.209498461 |
| -.912949103 | .175282444 |
| .0991094375 | -.349824698 |
| 1.14457216 | .0464172582 |
| .438046137 | .0177646297 |
| 1.22382056 | -.162675287 |

10X2 Matrix multiplied by 2x2 vector, giving a 10 x 2 vector



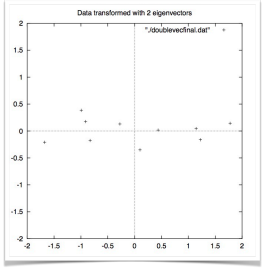
# PCA: Step 5: Re-plot-1




- Using single eigenvector
- 2nd Component effects are removed
- Reduced to single dimension
- Data adjusted to reveal new patterns



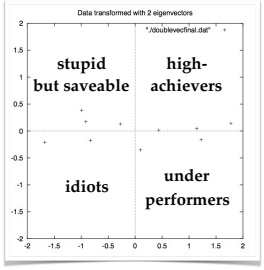
# PCA: Step 5: Re-plot-2




- Using two eigenvectors
- Still cast along two dimensions
- But, adjusted to reveal new patterns



# PCA: Step 5: Re-plot Data (2)




~ **Could reveal new groupings of data-items**

# PCA: Steps… **REM**

**Step1: Subtract means from raw data**

**Step2: Do COV/CORR matrix for variables**

**Step3: Find matrix eigenvectors+eigenvalues**

**Step4: Order eigenvector(s) from high->low**

**Step5*: Apply & Re-plot data**

* Optional; eigenvectors adopted as new representation of data



# PCA: Stop at Step: Use Eigen

**Step1: Subtract means from raw data**

**Step2: Do COV/CORR matrix for variables**

**Step3: Find matrix eigenvectors+eigenvalues**

**Step4: Order eigenvector(s) from high->low**

$$eigenvalues = \begin{pmatrix} .0490833989 \\ 1.28402771 \end{pmatrix}$$

$$eigenvectors = \begin{pmatrix} -.735178656 & -.677873399 \\ .677873399 & -.735178656 \end{pmatrix}$$

~ **a new coding of the old data**

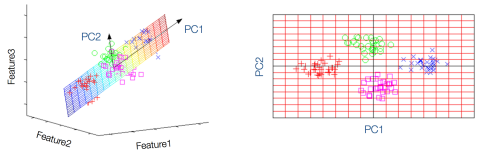
# PCA: Uses of …

- PCA can be used as a method for feature extraction; finding a set of features to use in your model
- PCA can speed up an ML algorithm by reducing the number of input features/variables
- PCA can aid visualisation; reducing 4-D to 2-D
- Like Factor Analysis but actually different; most think they are the same

http://www.iro.umontreal.ca/~pift6080/H09/documents/papers/pca_tutorial.pdf

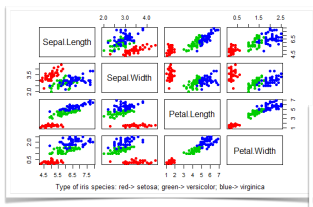
# PCA: Prog Example

- PCA can aid visualisation; 4-D to 2-D
- Prog Eg using Irises Database



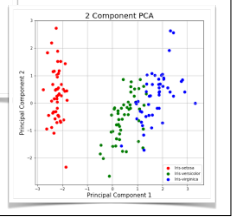

http://www.iro.umontreal.ca/~pift6080/H09/documents/papers/pca_tutorial.pdf

# PCA: Prog Example


From 4-D


...To 2-D

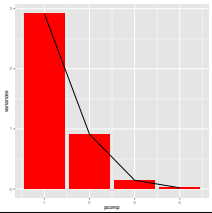


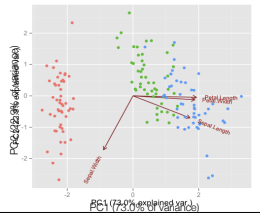

# PCA: Prog Example

We generally select the *k* PCs with the highest variance.

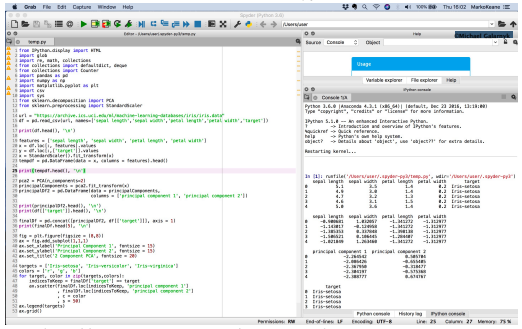
**Example:** Apply PCA to *Iris* data set, and examine amount of variance in each PC.

The first two PCs account for ~96% of the variance in the data.



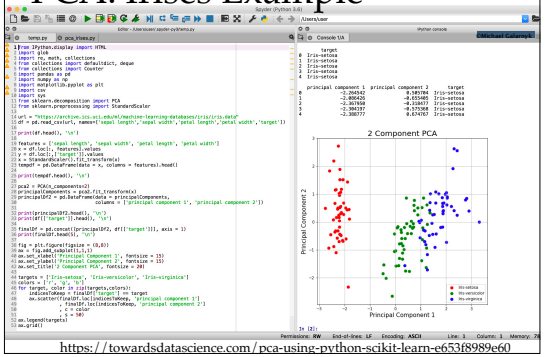




# PCA: Irises Example



https://towardsdatascience.com/pca-using-python-scikit-learn-e653f8989e60

# PCA: Irises Example




https://towardsdatascience.com/pca-using-python-scikit-learn-e653f8989e60

# PCA: Some Issues

- PCA is not scale-invariant; hence, must scale data first
- Components explain largest variance assumed to be of most interest; but this may not always be the case
- Eigenvectors are orthogonal transformations of original variables; so, other rotations are not entertained
- Good for distributions that are well-captured by mean-vectors/co-variance matrices; some distributions are not…
- PSA only achieves dimensionality reduction, when variables are correlated; otherwise, just orders by variances

https://towardsdatascience.com/pca-using-python-scikit-learn-e653f8989e60

# PCA: Factors Affecting Retweets

Want to be Retweeted? Large Scale Analytics on Factors Impacting Retweet in Twitter Network

Bongwon Suh, Lichan Hong, Peter Pirolli, and Ed H. Chi
Palo Alto Research Center, Inc.
Palo Alto, CA, U.S.A.
{suh, hong, pirolli, echi}@parc.com

TABLE I. TWEET FEATURES

| | |
|---|---|
| *URL* | # of URLs in a tweet |
| *Hashtag* | # of hashtags in a tweet |
| *Mention* | # of usernames specified in a tweet excluding ones used for making a retweet (e.g. via @username, RT: @username) |
| *Follower* | # of users who follows the author of a tweet |
| *Followee (Friend)* | # of friends that the author is following |
| *Days* | # of days since the author created Twitter account |
| *Status* | # of tweets made by the author since the creation of the account |
| *Favorite* | # of favorited tweets by a user |
| *Retweet* | # of retweets recorded for a given tweet |

V. EXPLORATORY PRINCIPAL COMPONENTS ANALYSIS

We performed a Principal Component Analysis (PCA) with the 10K data set with the nine features. PCA is a data reduction technique in which possibly correlated features (e.g., those in Table I) are transformed into a smaller number of factors called principal components. The technique is often used in an attempt to reveal internal (underlying) structure that maximally accounts for the variance in the data set. Our PCA analysis

Suh, B., Hong, L., Pirolli, P., & Chi, E. H. (2010, August). Want to be retweeted? large scale analytics on factors impacting retweet in twitter network. In *2010 IEEE Second International Conference on Social Computing* (pp. 177-184). IEEE.

# PCA: Factors Affecting Retweets

TABLE II. FACTORS (PRINCIPLE COMPONENTS) FROM THE ANALYSIS

| Factor | Eigenvalue | % Variance | Cumulative % Variance |
|---|---|---|---|
| 1 | 1.52 | 16.94 | 16.94 |
| 2 | 1.31 | 14.52 | 31.46 |
| 3 | 1.16 | 12.88 | 44.34 |
| 4 | 0.98 | 10.90 | 55.25 |
| 5 | 0.94 | 10.48 | 65.73 |
| 6 | 0.89 | 9.94 | 75.67 |
| 7 | 0.78 | 8.70 | 84.37 |
| 8 | 0.71 | 7.89 | 92.26 |
| 9 | 0.70 | 7.74 | 100.00 |

TABLE III. FACTOR LOADINGS

| Tweet Features | Factor 1 | Factor 2 | Factor 3 |
|---|---|---|---|
| **Retweet** | 0.2870 | -0.0021 | 0.4459 |
| **Hashtag** | 0.1709 | -0.4035 | -0.1333 |
| **Mention** | -0.1353 | 0.4169 | 0.2732 |
| **URL** | 0.3422 | -0.6417 | -0.1883 |
| **Days** | 0.1868 | 0.3117 | -0.0795 |
| **Favorite** | 0.3030 | 0.5347 | -0.5033 |
| **Follower** | 0.5403 | 0.0325 | 0.5064 |
| **Followee** | 0.6382 | 0.0237 | 0.2524 |
| **Status** | 0.5561 | 0.2966 | -0.4343 |

We interpret Factor 1 (Figure 1) as capturing the degree to which tweet authors are "broadcasters". Factor 1 is correlated with number of followees, number of followers, and number of tweets (status). Retweets correlate 0.29 with Factor 1.

We interpret Factor 2 (Figures 1 and 2) as a content factor separating tweets that contain URLs and hashtags from those with mentions instead. URLs and hashtags correlate positively with Factor 2, whereas mentions have a negative correlation. This makes sense given that tweets are limited in the amount of content they can communicate, so having one kind of content (e.g., URLs) will tend to be exclusive of another (e.g., mentions).

We interpret Factor 3 (Figure 2) as distinguishing types of different users, specifically separating tweet authors who get retweeted frequently and have lots of followers, from those who tweet frequently (status) and have many favorites. Factor 3 is positively associated with retweeting and number of followers, and negatively associated with status and favorites. One example that fits this category would be Bill Gates (@billgates). He made only 128 tweets so far and has no favorite items but he has more then 780 thousand followers.

In summary, Factor 1 is strongly associated with features representing the author profile: numbers of followees, number of followers, and status all load heavily. Factor 2 is associated with content features, showing strong correlations with mentions and negative correlations with URLs and hashtags. Factor 3 is strongly associated with retweets correlated with high number of followers but negatively correlated with number of favorites and status.

Suh, B., Hong, L., Pirolli, P., & Chi, E. H. (2010, August). Want to be retweeted? large scale analytics on factors impacting retweet in twitter network. In *2010 IEEE Second International Conference on Social Computing* (pp. 177-184). IEEE.

**SVD**

# SVD: History

- IQ Testing in Psychology; get people to do tasks and puzzles (WWII streaming)
- Find that people who do well on one task, do well on another too; task perf. is correlated
- Proposed general factor, *g*, general intelligence that lies behind this commonality
- A pure statistical concept…

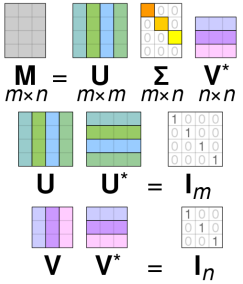
# SVD: The Idea: I

- SVD is closely related to PCA in the factor analytic tradition; basically, a similar move
- Another dimensionality reduction tool
- But, based on matrix algebra, for decomposing the matrix into constituent matrices; for rows and columns

ftp://statgen.ncsu.edu/pub/thorne/molevoclass/AtchleyOct19.pdf

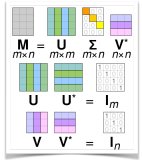
# SVD: Maths

- SVD decomposes a matrix into constituent parts (factorisation)
- M is the *m* x *n* matrix we want to decompose
- U is *m* x *m* matrix
- ∑ (sigma) is *m* x *n* diagonal matrix
- V* is *n* x *n* matrix




https://en.wikipedia.org/wiki/Singular_value_decomposition

# SVD: Maths

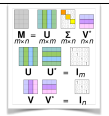



- Any matrix can be approximated by one of a lower rank; rank is akin to no of dimensions/factors required to account for data
- U and V are computed from multiplying original matrix by its transpositions, M.M$^T$ and M$^T$.M, respectively; then getting eigenvectors and eigenvalues of the resultant matrices
- Singular values (in ∑) are square roots of the eigenvalues of the cross-product matrices (U, V)
- Singular values can be applied to column/row variables to determine relative "importance" identifying factors (i.e., how much they account for variance)

http://web.mit.edu/be.400/www/SVD/Singular_Value_Decomposition.htm

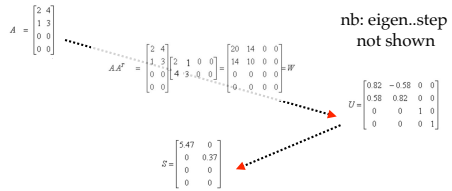
# SVD: Maths:

$A_{nxp} = U_{nxn} S_{nxp} V^T_{pxp}$




**U** is computed from A.A$^T$ to get eigenvectors and eigenvalues of resultant matrices

nb: eigen..step not shown

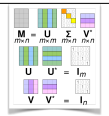



**S is the square root of the eigenvalues from $AA^T$ or $A^TA$**

http://web.mit.edu/be.400/www/SVD/Singular_Value_Decomposition.htm

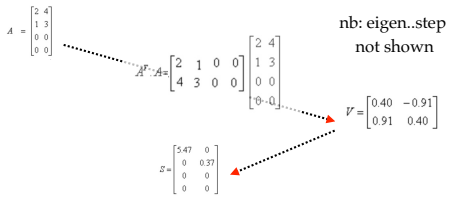
# SVD: Maths:

$A_{nxp} = U_{nxn} S_{nxp} V^T_{pxp}$




**V** is computed from A$^T$.A to get eigenvectors and eigenvalues of resultant matrices

nb: eigen..step not shown




**S is the square root of the eigenvalues from $AA^T$ or $A^TA$**

http://web.mit.edu/be.400/www/SVD/Singular_Value_Decomposition.htm

# SVD: Maths: ∑ : **Singular Values**

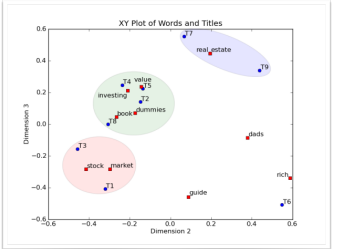
- Square of the singular values are eigenvalues
- Convey relative importance of different eigenvectors

$$S = \begin{bmatrix} 5.47 & 0 \\ 0 & 0.37 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

# SVD: Eg…see later

| Index Words | Titles | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | T1 | T2 | T3 | T4 | T5 | T6 | T7 | T8 | T9 |
| book | | 1 | 1 | | | | | | |
| dads | | | | | | 1 | | | 1 |
| dummies | 1 | | | | | | 1 | | |
| estate | | | | | | 1 | | | 1 |
| guide | 1 | | | | | 1 | | | |
| investing | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| market | 1 | | 1 | | | | | | |
| real | | | | | | 1 | | | 1 |
| rich | | | | | | 2 | | | 1 |
| stock | 1 | | 1 | | | | 1 | | |
| value | | | | 1 | 1 | | | | |

| | | | |
|---|---|---|---|
| book | 0.15 | -0.27 | 0.04 |
| dads | 0.24 | 0.38 | -0.09 |
| dummies | 0.13 | -0.17 | 0.07 |
| estate | 0.18 | 0.19 | 0.45 |
| guide | 0.22 | 0.09 | -0.46 |
| investing | 0.74 | -0.21 | 0.21 |
| market | 0.18 | -0.3 | -0.28 |
| real | 0.18 | 0.19 | 0.45 |
| rich | 0.36 | 0.59 | -0.34 |
| stock | 0.25 | -0.42 | -0.28 |
| value | 0.12 | -0.14 | 0.23 |



**Latent Semantic Indexing/Analysis**

## Lets Distinguish…

- LSI (LSA) as a technique for processing term-doc matrices
- LSI (Latent Semantic Indexing) was the version of this used to build new indices for retrieval systems, typically
- LSA (Latent Semantic Analysis) was the version of this used to process terms-docs (e.g., for paraphrase id)
- LSA-Theory (Plato's Problem) was an attempted to build a new LSA-based-vector-representation for words/paras to use for semantic tasks (Landauer & Dumais, 1997)

## LSI/LSA = SVD(Term-Doc Matrix)

- LSI (LSA) is SVD on a Term-Doc VSM to find latent/hidden associations between words, usually using co-occurrence
- If two words occur together frequently then they are related to some common topic (meaning by the company it keeps)
- 'goal', 'net', 'post' will have associated clusters of words about the soccer topic, even though "soccer" may not be explicitly mentioned, the latent-factor could be (interpreted as) SOCCER
- LSI uses a weighted term-document matrix, performing a SVD on the matrix, and using the resultant matrix to identify the concept-groupings contained in the text.

## LSI/A is based on VSMs

- LSI/A begins by constructing a term-document matrix, A, to identify the occurrences of the m unique terms within a collection of *n* documents.
- In a term-document matrix, each term/word is a row, and each document/text-item is a column
- This matrix is usually very large and very sparse; aka something you want to reduce
- SVD can give you a new matrix to encode terms/docs

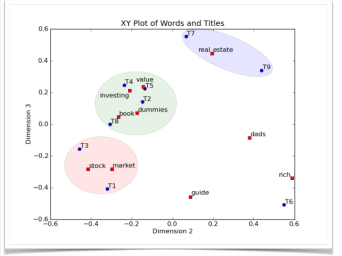
## LSI/LSA

- Gives you a new set of vectors for the terms and documents; re-coding them in terms of the factors (eigenvector and eigenvalues found)
- These new vectors can be clustered; to show new, deeper groupings or combine for D-reduction
- Also, new recoding of words may constitute a latent model of meaning (LSA-Theory)

## From This to This…

| Index Words | Titles | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | T1 | T2 | T3 | T4 | T5 | T6 | T7 | T8 | T9 |
| book | | | 1 | 1 | | | | | |
| dads | | | | | | 1 | | | 1 |
| dummies | | 1 | | | | | 1 | | |
| estate | | | | | | 1 | | | 1 |
| guide | 1 | | | | | 1 | | | |
| investing | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| market | 1 | | 1 | | | | | | |
| real | | | | | | 1 | | | 1 |
| rich | | | | | | 2 | | | 1 |
| stock | 1 | | 1 | | | | 1 | | |
| value | | | | 1 | 1 | | | | |

| | | | |
|---|---|---|---|
| book | 0.15 | -0.27 | 0.04 |
| dads | 0.24 | 0.38 | -0.09 |
| dummies | 0.13 | -0.17 | 0.07 |
| estate | 0.18 | 0.19 | 0.45 |
| guide | 0.22 | 0.09 | -0.46 |
| investing | 0.74 | -0.21 | 0.21 |
| market | 0.18 | -0.3 | -0.28 |
| real | 0.18 | 0.19 | 0.45 |
| rich | 0.36 | 0.59 | -0.34 |
| stock | 0.25 | -0.42 | -0.28 |
| value | 0.12 | -0.14 | 0.23 |



# LSA: Eg: Data:Book Titles

i. The Neatest Little Guide to Stock Market Investing
ii. Investing For Dummies, 4th Edition
iii. The Little Book of Common Sense Investing: The Only Way to Guarantee Your Fair Share of Stock Market Returns
iv. The Little Book of Value Investing
v. Value Investing: From Graham to Buffett and Beyond
vi. Rich Dad's Guide to Investing: What the Rich Invest in, That the Poor and the Middle Class Do Not!
vii. Investing in Real Estate, 5th Edition
viii. Stock Investing For Dummies
ix. Rich Dad's Advisors: The ABC's of Real Estate Investing: The Secrets of Finding Hidden Profits Most Investors Miss

https://technowiki.wordpress.com/2011/08/27/latent-semantic-analysis-lsa-tutorial/

# LSA: Eg: Data:Code

```
from numpy import zeros
from scipy.linalg import svd
from math import log  # needed for TFIDF
from numpy import asarray, sum

titles = ["The Neatest Little Guide to Stock Market Investing",
          "Investing For Dummies, 4th Edition",
          "The Little Book of Common Sense Investing: The Only Way to Guarantee
            Your Fair Share of Stock Market Returns",
          "The Little Book of Value Investing",
          "Value Investing: From Graham to Buffett and Beyond",
          "Rich Dad's Guide to Investing: What the Rich Invest in, That the \
         Poor and the Middle Class Do Not!",
          "Investing in Real Estate, 5th Edition",
          "Stock Investing For Dummies",
          "Rich Dad's Advisors: The ABC's of Real Estate Investing: The \
         Secrets of Finding Hidden Profits Most Investors Miss"
          ]

stopwords = ['and','edition','for','in','little','of','the','to']
ignorechars = ''',:'!''
```

https://technowiki.wordpress.com/2011/08/27/latent-semantic-analysis-lsa-tutorial/

# LSA: Eg: Term:Doc Matrix

| Index Words | Titles | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | T1 | T2 | T3 | T4 | T5 | T6 | T7 | T8 | T9 |
| book | | | 1 | 1 | | | | | |
| dads | | | | | | 1 | | | 1 |
| dummies | | 1 | | | | | | 1 | |
| estate | | | | | | | 1 | | 1 |
| guide | 1 | | | | | 1 | | | |
| investing | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| market | 1 | | 1 | | | | | | |
| real | | | | | | | 1 | | 1 |
| rich | | | | | | 2 | | | 1 |
| stock | 1 | | 1 | | | | | 1 | |
| value | | | | 1 | 1 | | | | |

NB: Often Tf-Idf-ed

tf-idf defined in prog but not used

```
[[ 0. 0. 1. 1. 0. 0. 0. 0. 0.]
 [ 0. 0. 0. 0. 0. 1. 0. 0. 1.]
 [ 0. 1. 0. 0. 0. 0. 0. 1. 0.]
 [ 0. 0. 0. 0. 0. 0. 1. 0. 1.]
 [ 1. 0. 0. 0. 0. 0. 1. 0. 0.]
 [ 1. 1. 1. 1. 1. 1. 1. 1. 1.]
 [ 1. 0. 1. 1. 0. 0. 0. 0. 0.]
 [ 0. 0. 0. 0. 0. 0. 1. 0. 1.]
 [ 0. 0. 0. 0. 0. 2. 0. 0. 1.]
 [ 1. 0. 1. 0. 0. 0. 0. 1. 0.]
 [ 0. 0. 0. 1. 1. 0. 0. 0. 0.]]
```

https://technowiki.wordpress.com/2011/08/27/latent-semantic-analysis-lsa-tutorial/

# LSA: SVD Step

```
def calc(self):
    self.U, self.S, self.Vt = svd(self.A)
```

| | SV | EigenV |
|---|---|---|
| SV1 | 3.308 | 10.94 |
| SV2 | 2.755 | 7.59 |
| SV3 | 1.880 | 3.53 |
| SV4 | 1.790 | 3.20 |
| SV5 | 1.386 | 1.92 |
| SV6 | 0.949 | 0.90 |
| SV7 | 0.795 | 0.63 |
| SV8 | 0.448 | 0.20 |
| SV9 | 0.278 | 0.08 |

All 9 Singular Values (sq-root of eigenvalues)

nb: the values given above taken from the program outputs and do not correspond to those in the document

https://technowiki.wordpress.com/2011/08/27/latent-semantic-analysis-lsa-tutorial/

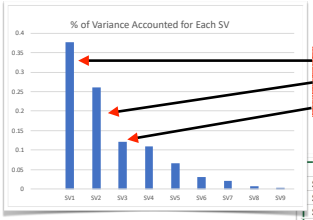
# Recall…Explaining the Variance

- Recall, in PCA, the eigenvalues are the variances for the latent variables/components;
- So, can be used to assess how much variance a PC explains (PC1-variance*/Total-variance **)
- Square of the singular value (SV) == eigenvalue
- So, an ordering of SVs tells us about the relative importance of the latent-variables

*ie, its eigenvalue, ** ie. sum of all eigenvalues

# LSA: SVD Step

```
def calc(self):
    self.U, self.S, self.Vt = svd(self.A)
```




SV1 = freq of word in title
SV2 = semantic1 dimension
SV3 = semantic2 dimension

| | SV | EigenV | % Var |
|---|---|---|---|
| SV1 | 3.308 | 10.94 | 0.38 |
| SV2 | 2.755 | 7.59 | 0.26 |
| SV3 | 1.880 | 3.53 | 0.12 |
| SV4 | 1.790 | 3.20 | 0.11 |
| SV5 | 1.386 | 1.92 | 0.07 |
| SV6 | 0.949 | 0.90 | 0.03 |
| SV7 | 0.795 | 0.63 | 0.02 |
| SV8 | 0.448 | 0.20 | 0.01 |
| SV9 | 0.278 | 0.08 | 0.00 |

nb. values taken from program

https://technowiki.wordpress.com/2011/08/27/latent-semantic-analysis-lsa-tutorial/

# LSA: SVD Step

$$A_{nxp} = U_{nxn}\, S_{nxp}\, V^{T}_{pxp}$$

```
def calc(self):
    self.U, self.S, self.Vt = svd(self.A)
```

**U**

| | SV1 | SV2 | SV3 |
|---|---|---|---|
| book | 0.15 | -0.27 | 0.04 |
| dads | 0.24 | 0.38 | -0.09 |
| dummies | 0.13 | -0.17 | 0.07 |
| estate | 0.18 | 0.19 | 0.45 |
| guide | 0.22 | 0.09 | -0.46 |
| investing | 0.74 | -0.21 | 0.21 |
| market | 0.18 | -0.3 | -0.28 |
| real | 0.18 | 0.19 | 0.45 |
| rich | 0.36 | 0.59 | -0.34 |
| stock | 0.25 | -0.42 | -0.28 |
| value | 0.12 | -0.14 | 0.23 |

**S**

| | | |
|---|---|---|
| 3.91 | 0 | 0 |
| 0 | 2.61 | 0 |
| 0 | 0 | 2 |

| SV1 | SV2 | SV3 |
|---|---|---|

**Vt**

| T1 | T2 | T3 | T4 | T5 | T6 | T7 | T8 | T9 | |
|---|---|---|---|---|---|---|---|---|---|
| 0.35 | 0.22 | 0.34 | 0.26 | 0.22 | 0.49 | 0.28 | 0.29 | 0.44 | SV1 |
| -0.32 | -0.15 | -0.46 | -0.24 | -0.14 | 0.55 | 0.07 | -0.31 | 0.44 | SV2 |
| -0.41 | 0.14 | -0.16 | 0.25 | 0.22 | -0.51 | 0.55 | 0 | 0.34 | SV3 |

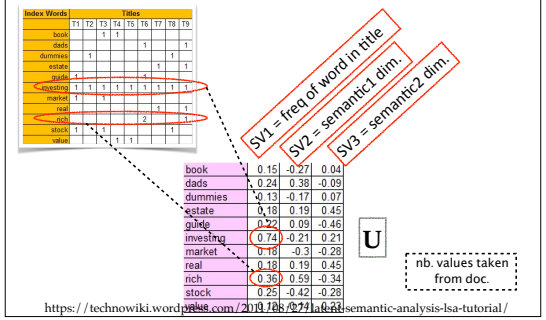
3 highest/important singular values/dims

nb: the values given above taken from the document below, do not correspond to the actual program outputs

https://technowiki.wordpress.com/2011/08/27/latent-semantic-analysis-lsa-tutorial/

# LSA: SVD Step

```
def calc(self):
    self.U, self.S, self.Vt = svd(self.A)
```




| | | | |
|---|---|---|---|
| book | 0.15 | -0.27 | 0.04 |
| dads | 0.24 | 0.38 | -0.09 |
| dummies | 0.13 | -0.17 | 0.07 |
| estate | 0.18 | 0.19 | 0.45 |
| guide | 0.22 | 0.09 | -0.46 |
| investing | 0.74 | -0.21 | 0.21 |
| market | 0.18 | -0.3 | -0.28 |
| real | 0.18 | 0.19 | 0.45 |
| rich | 0.36 | 0.59 | -0.34 |
| stock | 0.25 | -0.42 | -0.28 |
| value | 0.12 | -0.14 | 0.23 |

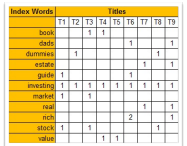
**U**

nb. values taken from doc.

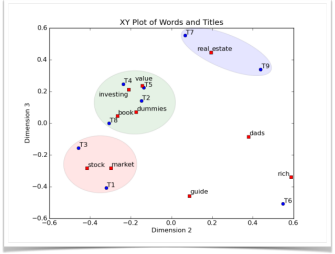
https://technowiki.wordpress.com/2011/08/27/latent-semantic-analysis-lsa-tutorial/

**So, what does LSA give you?**

# New Clusters of New Vectors



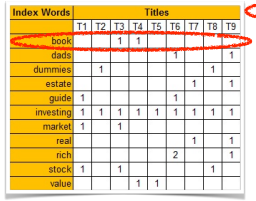

| | | | |
|---|---|---|---|
| book | 0.15 | -0.27 | 0.04 |
| dads | 0.24 | 0.38 | -0.09 |
| dummies | 0.13 | -0.17 | 0.07 |
| estate | 0.18 | 0.19 | 0.45 |
| guide | 0.22 | 0.09 | -0.46 |
| investing | 0.74 | -0.21 | 0.21 |
| market | 0.18 | -0.3 | -0.28 |
| real | 0.18 | 0.19 | 0.45 |
| rich | 0.36 | 0.59 | -0.34 |
| stock | 0.25 | -0.42 | -0.28 |
| value | 0.12 | -0.14 | 0.23 |




nb. values taken from doc.

https://technowiki.wordpress.com/2011/08/27/latent-semantic-analysis-lsa-tutorial/

# New/Better Word Vectors !



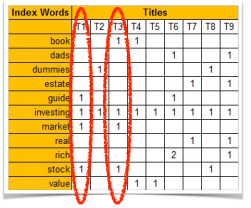

| | | | |
|---|---|---|---|
| book | 0.15 | -0.27 | 0.04 |
| dads | 0.24 | 0.38 | -0.09 |
| dummies | 0.13 | -0.17 | 0.07 |
| estate | 0.18 | 0.19 | 0.45 |
| guide | 0.22 | 0.09 | -0.46 |
| investing | 0.74 | -0.21 | 0.21 |
| market | 0.18 | -0.3 | -0.28 |
| real | 0.18 | 0.19 | 0.45 |
| rich | 0.36 | 0.59 | -0.34 |
| stock | 0.25 | -0.42 | -0.28 |
| value | 0.12 | -0.14 | 0.23 |

...that could surface new similarities between words

nb. values taken from doc.

https://technowiki.wordpress.com/2011/08/27/latent-semantic-analysis-lsa-tutorial/

# New/Better Doc Vectors !




| T1 | T2 | T3 | T4 | T5 | T6 | T7 | T8 | T9 |
|---|---|---|---|---|---|---|---|---|
| 0.35 | 0.22 | 0.34 | 0.26 | 0.22 | 0.49 | 0.28 | 0.29 | 0.44 |
| -0.32 | -0.15 | -0.46 | -0.24 | -0.14 | 0.55 | 0.07 | -0.31 | 0.44 |
| -0.41 | 0.14 | -0.16 | 0.25 | 0.22 | -0.51 | 0.55 | 0 | 0.34 |

...that could surface new similarities between titles/docs

nb. values taken from doc.

https://technowiki.wordpress.com/2011/08/27/latent-semantic-analysis-lsa-tutorial/

# New Latent Dimensions

- Original dimensions (words as counts) are now expressed in latent, semantic dimensions (words as SV1, SV2, SV3)
- By selecting SVs, we can reduce/enlarge dimensions
- NB: One can adjust the original matrix values using the SVD matrices (as in PCA)

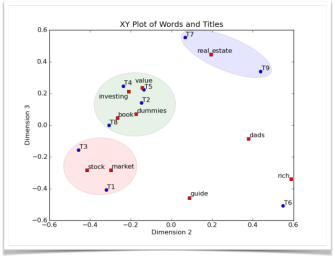
nb. values taken from doc.

| Index Words | Titles | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | T1 | T2 | T3 | T4 | T5 | T6 | T7 | T8 | T9 |
| book | | | 1 | 1 | | | | | |
| dads | | | | | | 1 | | | 1 |
| dummies | | 1 | | | | | | 1 | |
| estate | | | | | | | 1 | | 1 |
| guide | 1 | | | | | 1 | | | |
| investing | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| market | 1 | | 1 | | | | | | |
| real | | | | | | | 1 | | 1 |
| rich | | | | | | 2 | | | 1 |
| stock | 1 | | 1 | | | | | 1 | |
| value | | | | 1 | 1 | | | | |

| | | | |
|---|---|---|---|
| book | 0.15 | -0.27 | 0.04 |
| dads | 0.24 | 0.38 | -0.09 |
| dummies | 0.13 | -0.17 | 0.07 |
| estate | 0.18 | 0.19 | 0.45 |
| guide | 0.22 | 0.09 | -0.46 |
| investing | 0.74 | -0.21 | 0.21 |
| market | 0.18 | -0.3 | -0.28 |
| real | 0.18 | 0.19 | 0.45 |
| rich | 0.36 | 0.59 | -0.34 |
| stock | 0.25 | -0.42 | -0.28 |
| value | 0.12 | -0.14 | 0.23 |

https://technowiki.wordpress.com/2011/08/27/latent-semantic-analysis-lsa-tutorial/

# New Clusters of New Vectors




nb. values taken from doc.

https://technowiki.wordpress.com/2011/08/27/latent-semantic-analysis-lsa-tutorial/

*Memory & Cognition*
*2004, 32 (2), 185-197*

## But, Mark, what about

What plausibly affects plausibility? Concept coherence and distributional word coherence as factors influencing plausibility judgments

LOUISE CONNELL and MARK T. KEANE
*University College Dublin, Dublin, Ireland*

Our goal was to investigate the basis of human plausibility judgments. Previous research had suggested that plausibility is affected by two factors: concept coherence (the inferences made between parts of a discourse) and word coherence (the distributional properties of the words used). In two experiments, participants were asked to rate the plausibility of sentence pairs describing events. In the first, we manipulated concept coherence by using different inference types to link the sentences in a pair (e.g., causal or temporal). In the second, we manipulated word coherence by using latent semantic analysis, so two sentence pairs describing the same event had different distributional properties. The results showed that inference type affects plausibility; sentence pairs linked by causal inferences were rated highest, followed by attributal, temporal, and unrelated inferences. The distributional manipulations had no reliable effect on plausibility ratings. We conclude that the processes involved in rating plausibility are based on evaluating concept coherence, not word coherence.

## LSA-Theory ?

Plausibility has the hallmarks of a phenomenon so pervasive and important that, like air, no one notices it. Time and again, many cognitive accounts appeal to the idea of plausibility without specifying its cognitive basis. Most of these accounts are grounded in operational definitions of plausibility, such as, for example, the plausibility ratings that people give to some set of stimuli. Neither do

digm for exploring plausibility judgments. But first, let us reflect on the centrality of plausibility before exploring its theoretical bases more fully.

**The Centrality of Plausibility**

The centrality of plausibility is demonstrated by the diversity of cognitive acts in which it has been shown to

# Lets Distinguish… **REM**

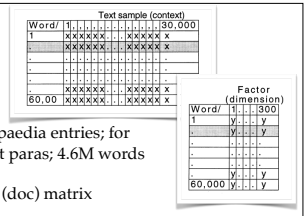
- LSI (LSA) as a technique for processing term-doc matrices
- LSI (Latent Semantic Indexing) was the version of this used to build new indices for retrieval systems, typically
- LSA (Latent Semantic Analysis) was the version of this used to process terms-docs (e.g., for paraphrase id)
- LSA-Theory (Plato's Problem) to build a new LSA-based-vector-representation for all words to use for semantic tasks (Landauer & Dumais, 1997)

Landauer, T. K., & Dumais, S. T. (1997). A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review, 104*(2), 211-240.

# LSA-Theory:Distributional Semantics

- A word's meaning is determined by the company it keeps…meaning emerges from distributional factors
- Meaning of a word, can be cast as some collection of the words it is typically seen with (co-occurrence)
- So, if we can analyse the statistical co-occurrence of words together to find latent-variables; those variables could define meaning-features of all words
- WOW ! Sheeeehshhhh !

# LSA Theory:




- Corpus: 37,000+ encyclopaedia entries; for young US students; short paras; 4.6M words
- A 60,000 (word) x 37,000 (doc) matrix
- Get frequency count of a word in a doc-context, *f(w, d)*, and then *log(f(w, d))/entropy(d)* [odd and unclear??]
- Now SVA the lot ! Select top 300 singular values; 300 is found to be optimal for synonym tests
- Apply these to original weighted-word counts; to create new 60,000 x 300 matrix encoding all the words

http://lsa.colorado.edu/papers/plato/plato.annote.html

# LSA Theory: Bang !

300-dimension word-vectors tells you an awful lot about a word's meaning, it predicts:

- Associations: night->day (and reaction times)
- Synonyms: punch -> strike
- Measure of word and sentence coherence:
  - "football, score, keeper" V "football, egg, vomit"
- New vectors for words…KNOWLEDGE !

# AND…

- LSA gives rise to probabilistic LSA (pLSA); Bayesian approach to considering the hidden factors of the term:document matrix
- pLSA leads to **Latent Dirichlet Allocation (LDA)**, which comes to be used heavily for topic modelling
- Any given word within a doc could belong to a topic/factor and the set of topics can be used to define the document, just as the set of factors characterise a given doc; LDA finds these topics, n.b., familiar issues around knowing *k* no. of topics in advance
- Leads to other method: Expectation Maximisation, Non-Non-Negative Matrix Factorisation, all heavily used in Text Analytics

# NB: LSA:Dimensional Reduction

### Real-World Example

Finally, I applied LDA to a set of Sarah Palin's emails a little while ago (see here for the blog post, or here for an app that allows you to browse through the emails by the LDA-learned categories), so let's give a brief recap. Here are some of the topics that the algorithm learned:

- **Trig/Family/Inspiration**: family, web, mail, god, son, from, congratulations, children, life, child, down, trig, baby, birth, love, you, syndrome, very, special, bless, old, husband, years, thank, best, …
- **Wildlife/BP Corrosion**: game, fish, moose, wildlife, hunting, bears, polar, bear, subsistence, management, area, board, hunt, wolves, control, department, year, use, wolf, habitat, hunters, caribou, program, denby, fishing, …
- **Energy/Fuel/Oil/Mining:** energy, fuel, costs, oil, alaskans, prices, cost, nome, now, high, being, home, public, power, mine, crisis, price, resource, need, community, fairbanks, rebate, use, mining, villages, …
- **Gas**: gas, oil, pipeline, agia, project, natural, north, producers, companies, tax, company, energy, development, slope, production, resources, line, gasline, transcanada, said, billion, plan, administration, million, industry, …
- **Education/Waste**: school, waste, education, students, schools, million, read, email, market, policy, student, year, high, news, states, program, first, report, business, management, bulletin, information, reports, 2008, quarter, …
- **Presidential Campaign/Elections**: mail, web, from, thank, you, box, mccain, sarah, very, good, great, john, hope, president, sincerely, wasilla, work, keep, make, add, family, republican, support, doing, p.o, …

http://blog.echen.me/2011/08/22/introduction-to-latent-dirichlet-allocation/

**Latent Stuff**

## Word Embeddings

# Word Embeddings…

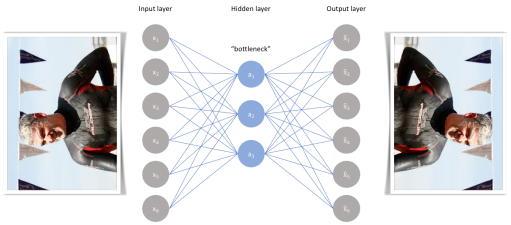
- LSA variant…computed differently…Google enter the fray…
- Creates multi-dimensional word vectors in a computationally efficient manner (for all); Vectors manifest semantic relations
- KING:QUEEN::PRINCE:? (using vectors offsets)
- A number of current competitors; explosion since 2015
  - word2vec
  - GloVe
  - Others….

# Embeddings: The Idea

- Co-occurrence analysis: Counting/Prediction
- *Counting*; basically, analyses a co-occurrence matrix using SVD
- *Prediction*: uses NNs to learn word-vectors based on co-occurrence, local-context sampling
- *Both*: GloVe (from Stanford)

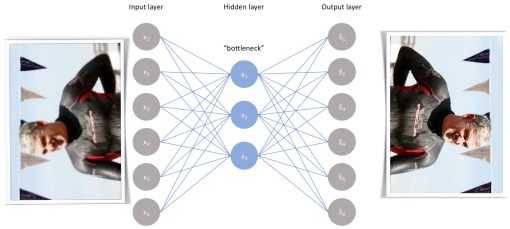
# Embeddings: Prediction

**NNs learn patterns: e.g., autoencoder**



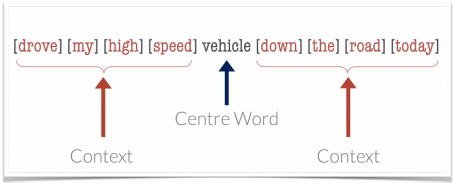

# Embeddings: Prediction

**NNs can learn to complete patterns…**




# Embeddings: Predictions

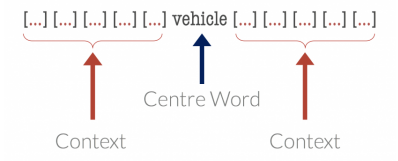
So, can we learn to predict a word from the context-words around it or predict the contexts from the word ?




Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. (2011). Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, *12*(Aug), 2493-2537.

# Embeddings: Predictions

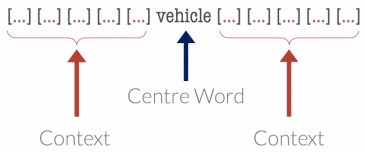
So, can we learn to predict a word from the context-words around it or predict the contexts from the word ? Generalised…




Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. (2011). Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, *12*(Aug), 2493-2537.

# Embeddings: Predictions

Slide a window over the text, or sample windows, noting word-contexts for each word to learn a vector




Collobert, R., & Weston, J. (2008, July). A unified architecture for natural language processing: Deep neural networks with multitask learning. In Proceedings of the 25th international conference on Machine learning (pp. 160-167). ACM.

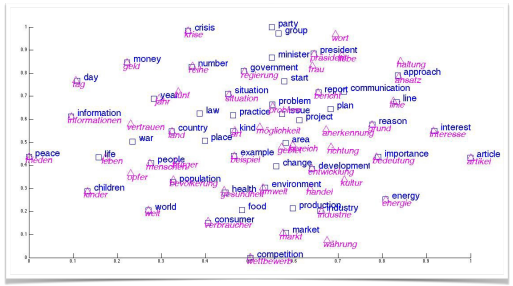
# Embeddings: Prediction

- Collobert & Weston (2008) using deep learning methods; archane, hard to compute
- Mikilova et al (2013) word2vec brings it to the masses (softmax, CBOW, Skip-gram)
- Vectors are better than ones from LSA-Count
- Area explode with many different techniques

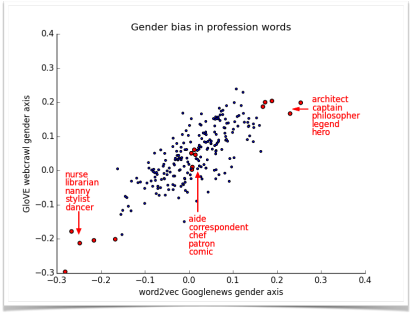
Mikolov, T., Chen, K., Corrado, G. and Dean J. (2013). Efficient estimation of word representations in vector space. CoRR, abs/1301.3781.

Goldberg, Y., & Levy, O. (2014). word2vec Explained: deriving Mikolov et al.'s negative-sampling word-embedding method. *arXiv preprint arXiv:1402.3722*.

# Using vectors: Languages




# Using vectors: Sexism..




# Using vectors: Trajectories..

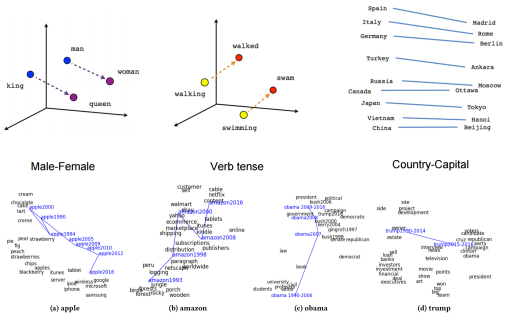



Figure 1: Trajectories of brand names and people through time: `apple`, `amazon`, `obama`, and `trump`.

# Embeddings: Benefits

- Uses real world text-sets (all of wikipedia); so avoids pre-processing : *thnx*, *thanks*, *plz*, *thanx*, get similar vectors
- Normalised vectors mean that transforms in the space are meaningful" man->woman, boy->girl;
- GloVe makes all more formal

https://www.shanelynn.ie/get-busy-with-word-embeddings-introduction/

# Embeddings: Our Work

- Leavy, Pine & Keane (2017). Government report on clerical sexual abuse, Ryan Report.
- Patterns of “sacking” but not called that: “dispensation”, “vows”, and other indirections (e.g., latin)
- word2vec generate synonyms that enabled automated query expansion for search

Leavy, S., Pine, E., & Keane, M. (2017). Mining the Cultural Memory of Irish Industrial Schools Using Word Embedding and Text Classification. In *DH-2017*.

# Embeddings: Our Work…

| 1987 | reagan | koch | soviet | iran_contra | navratilova | yuppie | walkman |
|---|---|---|---|---|---|---|---|
| 1988 | reagan | koch | soviet | iran_contra | sabatini | yuppie | tape_deck |
| 1989 | bush | koch | soviet | iran_contra | navratilova | yuppie | walkman |
| 1990 | bush | dinkins | soviet | iran_contra | navratilova | yuppie | headphones |
| 1991 | bush | dinkins | soviet | iran_contra | navratilova | yuppie | cassette_player |
| 1992 | bush | dinkins | russian | iran_contra | sabatini | yuppie | walkman |
| 1993 | clinton | dinkins | russian | iran_contra | navratilova | yuppie | cd_player |
| 1994 | clinton | mr_giuliani | russian | iran_contra | sanchez_vicario | yuppie | walkman |
| 1995 | clinton | giuliani | russian | white_house | graf | yuppie | cassette_player |
| 1996 | clinton | giuliani | russian | whitewater | graf | yuppie | walkman |
| 1997 | clinton | giuliani | russian | hingis | hingis | yuppie | headphones |
| 1998 | clinton | giuliani | russian | lewinsky | hingis | yuppie | headphones |
| 1999 | clinton | mayor_giuliani | russian | white_house | hingis | yuppie | buttons |
| 2000 | clinton | giuliani | russian | white_house | hingis | yuppie | headset |
| 2001 | bush | giuliani | russian | iran_contra | capriati | yuppie | headset |
| 2002 | bush | bloomberg | russian | white_house | hingis | gen_x | mp3_player |
| 2003 | bush | bloomberg | russian | white_house | agassi | hipsters | walkman |
| 2004 | bush | bloomberg | north_korean | iran_contra | federer | gen_x | headphones |
| 2005 | bush | bloomberg | north_korean | white_house | roddick | geek | ear_buds |
| 2006 | bush | bloomberg | iranian | white_house | hingis | teen | headset |
| 2007 | bush | bloomberg | iranian | capitol_hill | federer | dads | ipod |

Table 1: Examples of words from 1987 and their analogues over time. Each column corresponds to a single point in vector space, and each row shows the word closest to that point in a given year.

Szymanski, T. (2017). Temporal word analogies: Identifying lexical replacement with diachronic word embeddings. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)* (Vol. 2, pp. 448–453).

## Embeddings: Some Issues

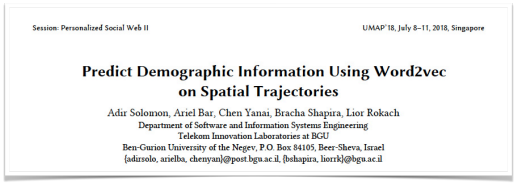
- Note, people typically use the vectors to do sthm-else; like idea of transfer learning
- Different runs produce different answers as the text is sampled in various ways…
- But, algorithmic stability + new metrics

Wendlandt, L., Kummerfeld, J. K., & Mihalcea, R. (2018). Factors Influencing the Surprising Instability of Word Embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)* (Vol. 1, pp. 2092-2102).

## Embeddings: Why Its Hot !

- Delivers very good "semantic" vectors
- Easy of use by anyone; eg. recomputing LSA tests in psych showing better results
- Many things can be treated as sentence: places I go to each day, genes…

## Why Its Hot !: Other Sequences



Session: Personalized Social Web II UMAP'18, July 8–11, 2018, Singapore

**Predict Demographic Information Using Word2vec on Spatial Trajectories**

Adir Solomon, Ariel Bar, Chen Yanai, Bracha Shapira, Lior Rokach
Department of Software and Information Systems Engineering
Telekom Innovation Laboratories at BGU
Ben-Gurion University of the Negev, P.O. Box 84105, Beer-Sheva, Israel
{adirsolo, arielba, chenyan}@post.bgu.ac.il, {bshapira, liorrk}@bgu.ac.il

## Conclusions

- We stared by considering how we can get at hidden-variables by dimensional reduction
- Continuous evolution from early work though PCA to SVD to word2vec…
- At the beginning of a very exciting step…in capturing the meaning of words