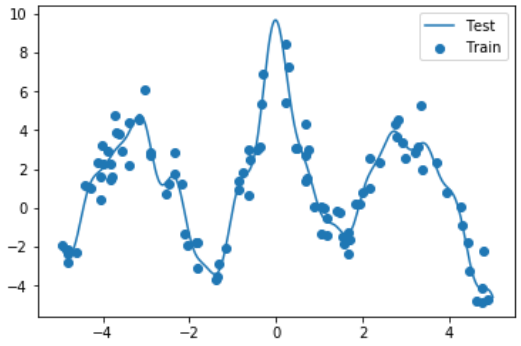
Summary of Unit 5:
Kernel Methods
For Regression and Classification



	SVM	Logistic Regression
Loss	hinge	cross entropy (log loss)
Sensitive to outliers	**Less**	More sensitive
Probabilistic?	No	**Yes**
Multi-class?	Only via separate model for each class (one-vs-all)	**Easy**, using softmax
Kernelizable? **(cover next class)**	Yes, with speed benefits from **sparsity**	Yes

Multi-class SVMs

- How do we extend idea of margin to more than 2 classes? Not so elegant. Two options:

One vs rest

Need to fit C separate models

Pick class with largest f(x)

One vs one

Need to fit C(C-1)/2 models

Pick class with most f(x) “wins”

Multi-class Logistic Regression

- How do we extend LR to more than 2 classes?
- Elegant: Can train weights using same prediction function we'll use at test time

$$\hat{p}(x) = \text{softmax}(w_1^T x, w_2^T x, \ldots w_C^T x)$$

Kernel methods

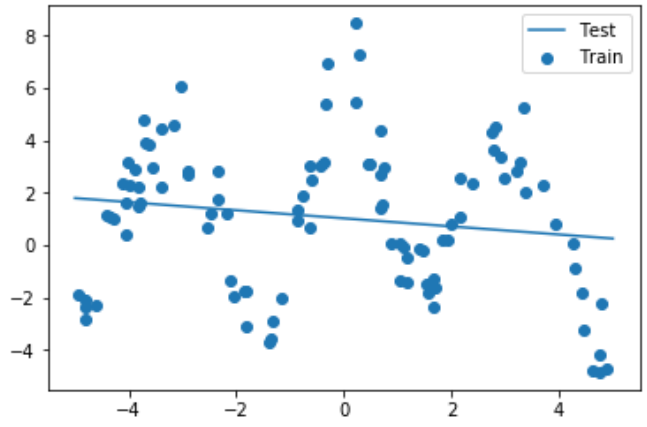
Use kernel functions (similarity function with special properties) to obtain flexible high-dimensional feature transformations without explicit features

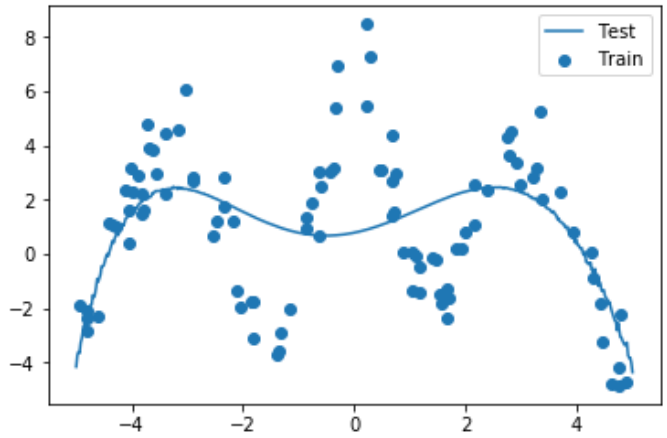
Solve “dual” problem (for parameter alpha), not “primal” problem (for weights w)

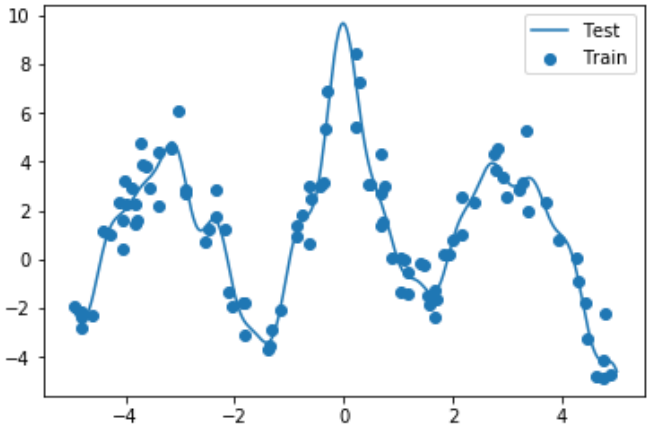
Can use the “kernel trick” for:

* regression
* classification (Logistic Regr. or SVM)

Kernel Methods for Regression







Kernels exist for:

- Periodic regression
- Histograms
- Strings
- Graphs,
- And more!

Review: Key concepts in supervised learning

- Parametric vs nonparametric methods
- Bias vs variance

Parametric vs Nonparametric

- Parametric methods
  - Complexity of decision function fixed in advance and specified by a finite fixed number of parameters, regardless of training data size

    Linear regression
    Logistic regression

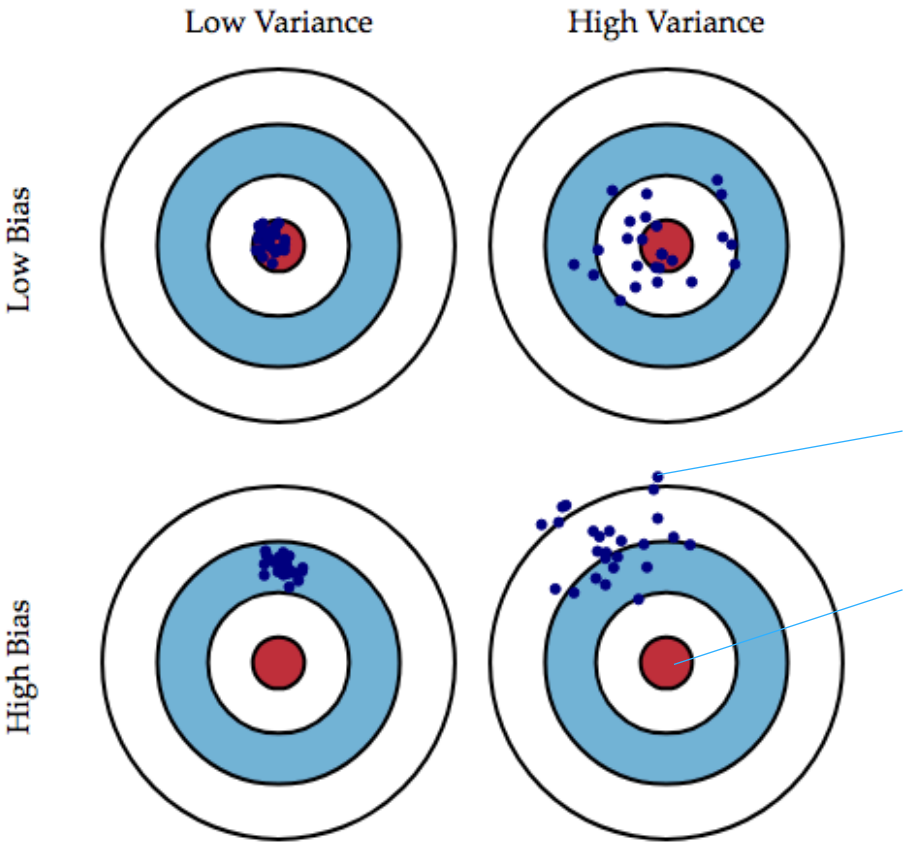
    Neural networks

- Nonparametric methods
  - Complexity of decision function can grow as more training data is observed

    Decision trees
    Ensembles of trees

    Nearest neighbor methods

Bias & Variance



Credit: Scott Fortmann-Roe
http://scott.fortmann-roe.com/docs/BiasVariance.html

Decompose into Bias & Variance

y is known "true" response value at given known heldout input x

$\hat{y}$ is a Random Variable obtained by fitting estimator to random sample of N training data examples, then predicting at x

Bias:
Error from average model to true
How far the average prediction of our model (averaged over all possible training sets of size N) is from true response

$$\underbrace{(\bar{y} - y)^2}_{\text{bias}^2} \qquad \bar{y} \triangleq \mathbb{E}[\hat{y}]$$

Variance:
Deviation over model samples
How far predictions based on a single training set are from the average prediction

$$\text{Var}(\hat{y}) = \mathbb{E}[(\hat{y} - \bar{y})^2] \quad = \mathbb{E}\left[\hat{y}^2\right] - \bar{y}^2$$

Total Error: Bias^2 + Variance

$$\mathbb{E}\Big[\left(\hat{y}(x^{tr}, y^{tr}) - y\right)^2\Big] = \mathbb{E}\Big[\left(\hat{y} - y\right)^2\Big]$$

Expected value is over samples of the observed training set

$$= \mathbb{E}\Big[\hat{y}^2 - 2\hat{y}y + y^2\Big]$$

$$= \mathbb{E}\Big[\hat{y}^2\Big] - 2\bar{y}y + y^2$$

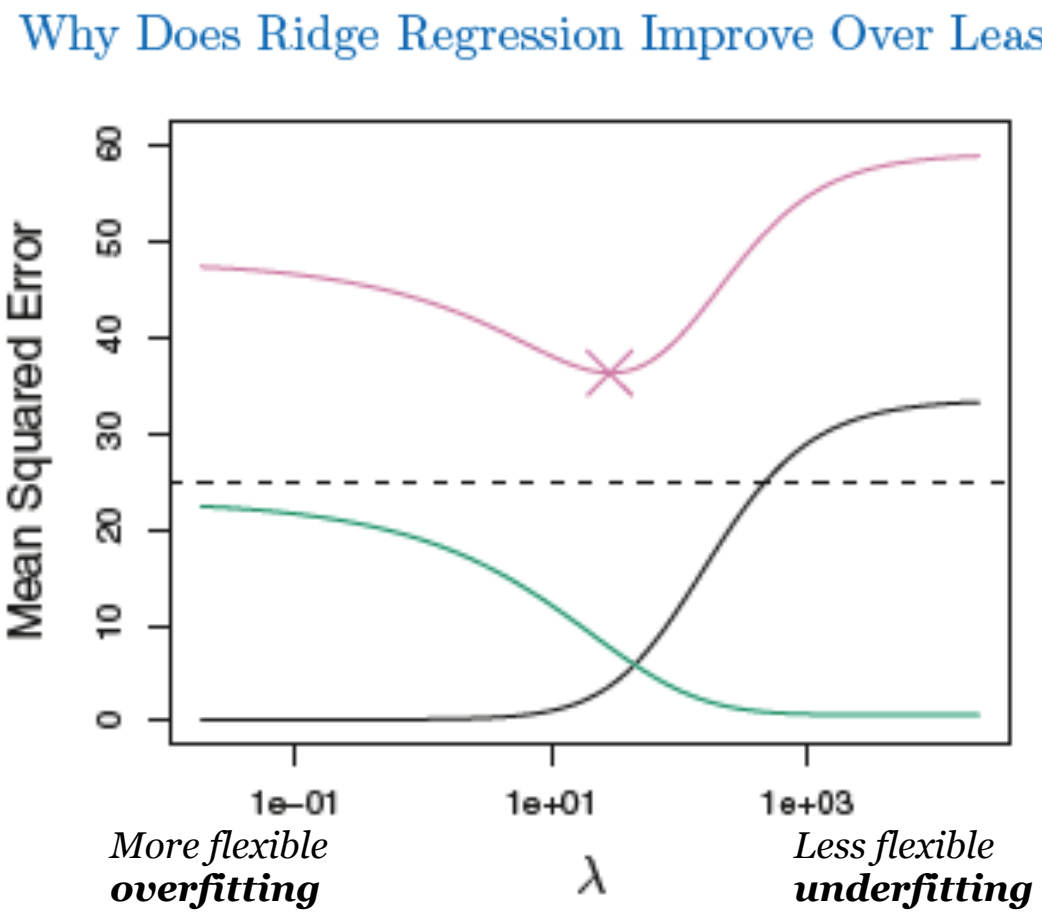
$$= \mathbb{E}\Big[\hat{y}^2\Big] - \bar{y}^2 + \bar{y}^2 - 2\bar{y}y + y^2$$

$$= \mathrm{Var}(\hat{y}) + (\bar{y} - y)^2$$

Variance

bias^2

Toy example: ISL Fig. 6.5



total error

bias

Error due to inability of typical fit (averaged over training sets) to capture true predictive relationship

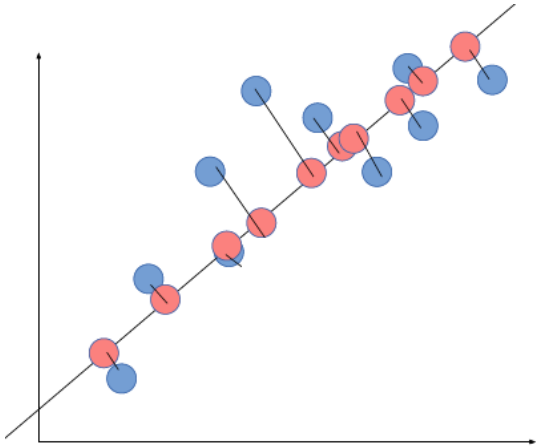
variance

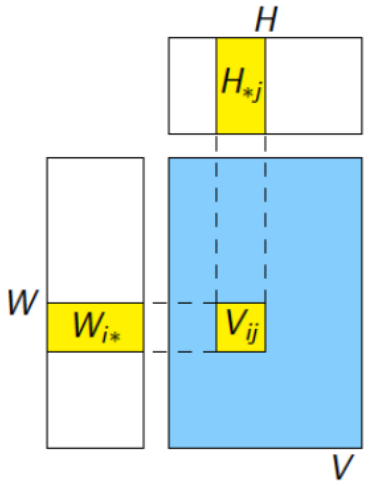
Error due to estimating from a single finite-size training set

All supervised learning methods must manage bias/variance tradeoff. Hyperparameter search is key.

Tufts COMP 135: Introduction to Machine Learning
https://www.cs.tufts.edu/comp/135/2020f/

Dimensionality Reduction & Embedding





Prof. Mike Hughes

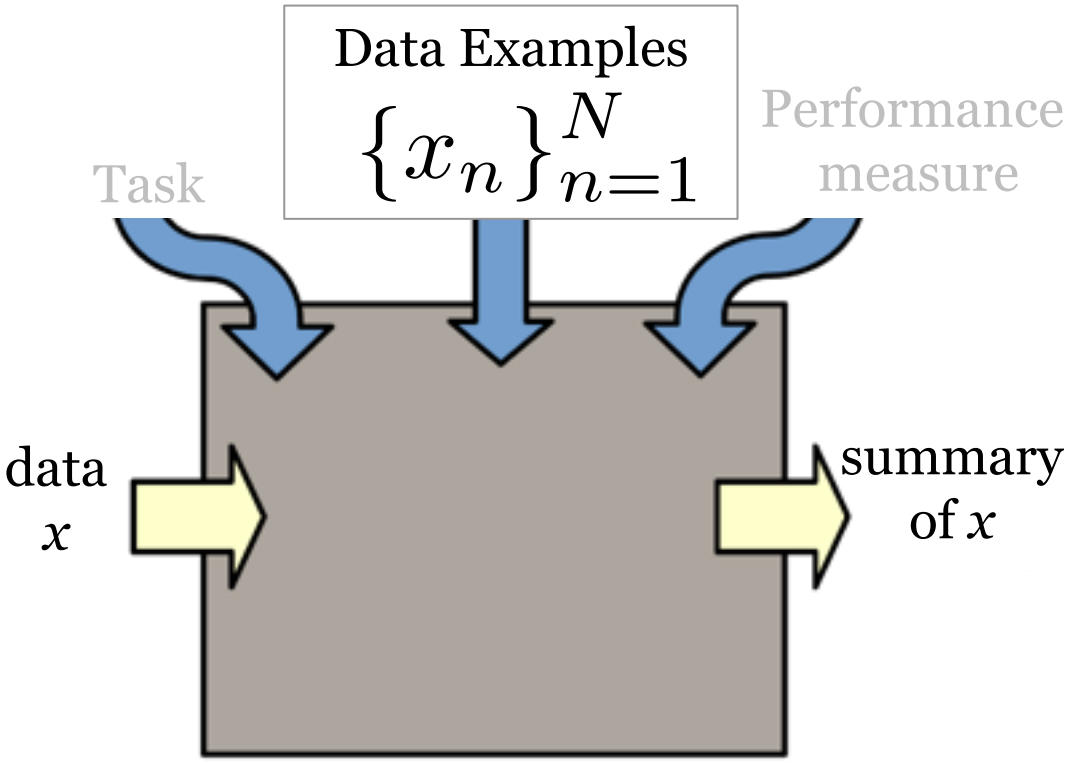
Many ideas/slides attributable to:
Liping Liu (Tufts), Emily Fox (UW)
Matt Gormley (CMU)

What will we learn?

Supervised Learning

Unsupervised Learning

Reinforcement Learning



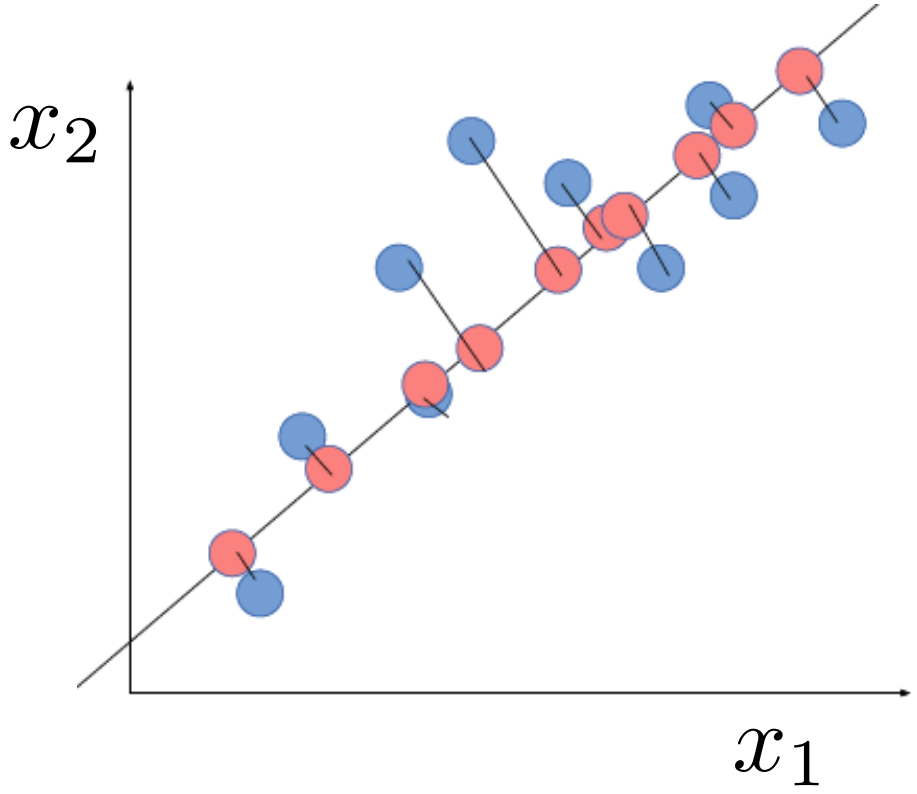
Task: Embedding

Supervised Learning

Unsupervised Learning

embedding

Reinforcement Learning



Dim. Reduction/Embedding
Unit Objectives

- Goals of dimensionality reduction
  - Reduce feature vector size (keep signal, discard noise)
  - "Interpret" features: visualize/explore/understand
- Common approaches
  - Principal Component Analysis (PCA)
  - word2vec and other neural embeddings
- Evaluation Metrics
  - Storage size
  - "Interpretability"
  - Reconstruction error

Example: 2D viz. of movies

Example Factors

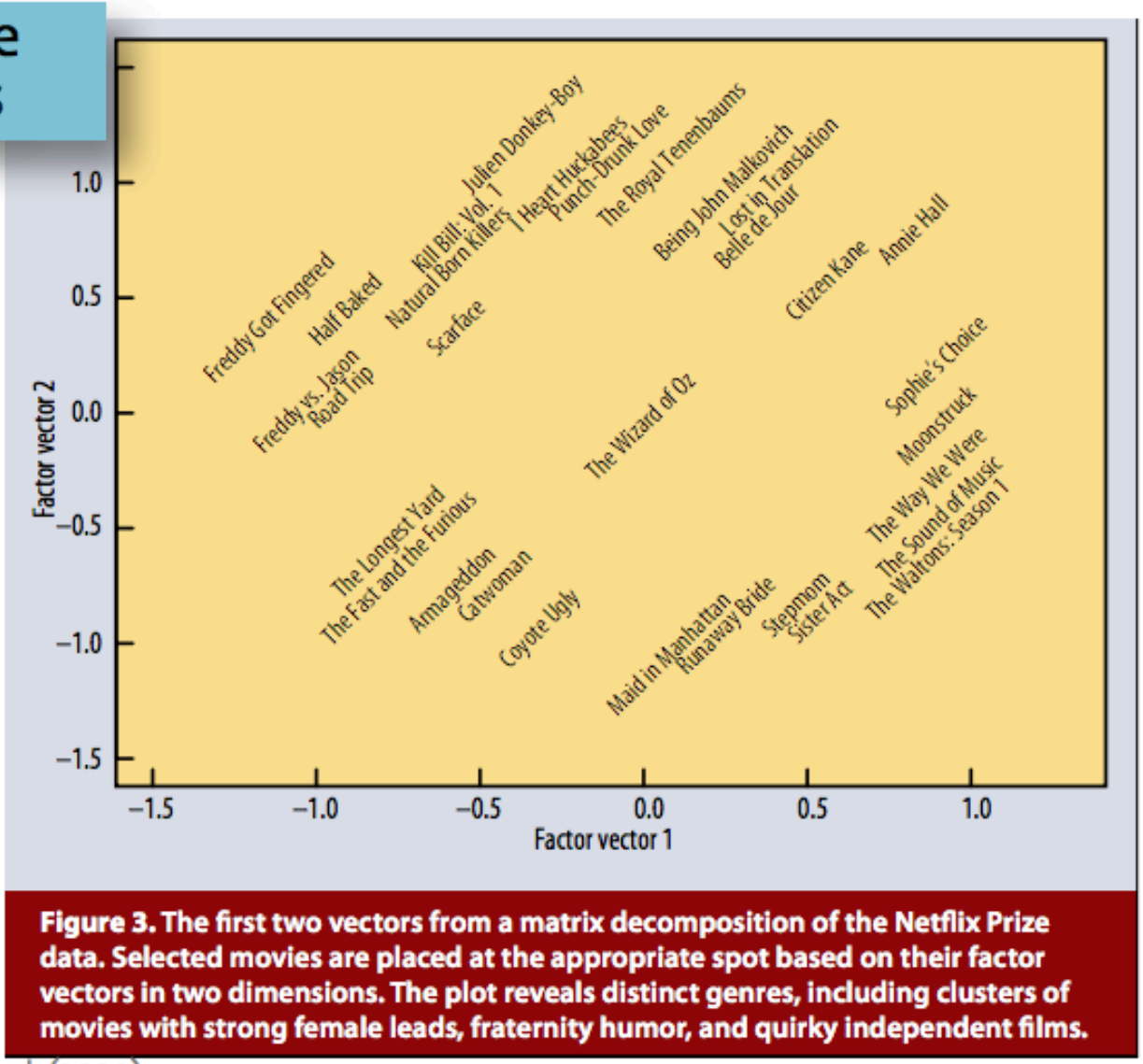


Figure 3. The first two vectors from a matrix decomposition of the Netflix Prize data. Selected movies are placed at the appropriate spot based on their factor vectors in two dimensions. The plot reveals distinct genres, including clusters of movies with strong female leads, fraternity humor, and quirky independent films.

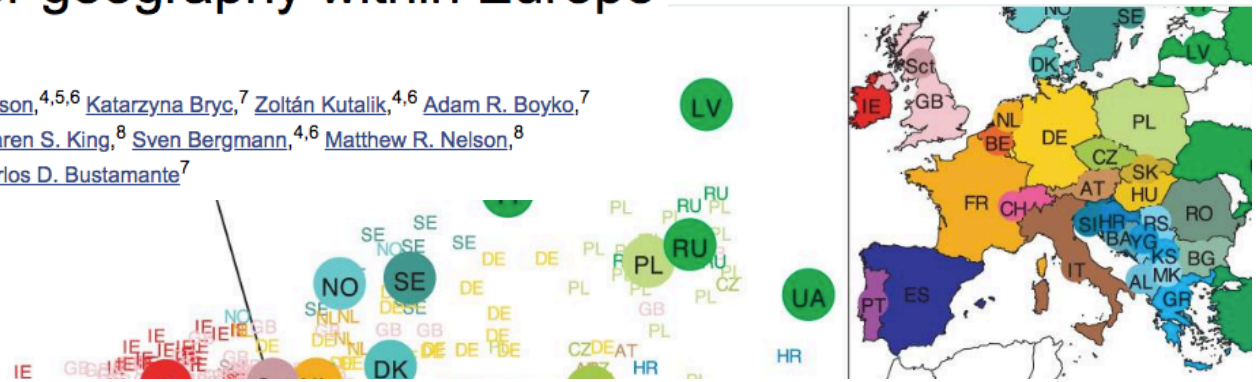
Figure from Koren et al. (2009)

Example: Genes vs. geography

Genes mirror geography within Europe

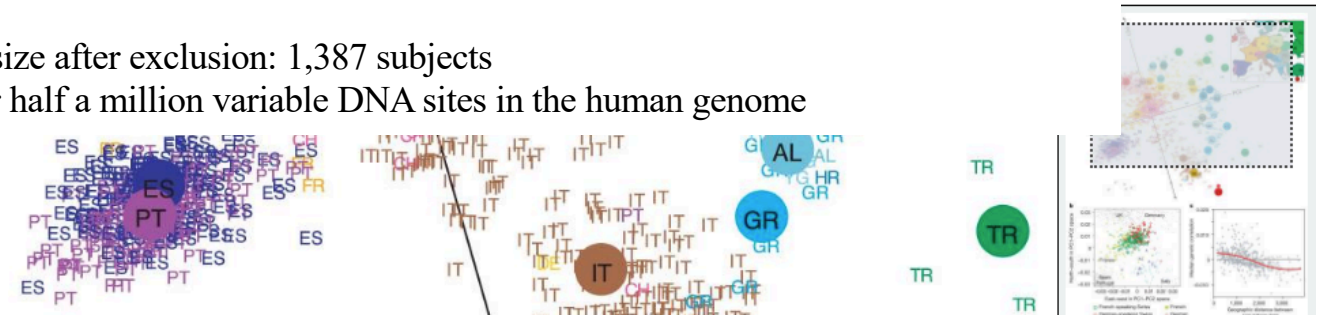
Nature, 2008

John Novembre,[1,2] Toby Johnson,[4,5,6] Katarzyna Bryc,[7] Zoltán Kutalik,[4,6] Adam R. Boyko,[7] Adam Auton,[7] Amit Indap,[7] Karen S. King,[8] Sven Bergmann,[4,6] Matthew R. Nelson,[8] Matthew Stephens,[2,3] and Carlos D. Bustamante[7]



Where possible, we based the geographic origin on the observed country data for grandparents. We used a 'strict consensus' approach: if all observed grandparents originated from a single country, we used that country as the origin. If an individual's observed grandparents originated from different countries, we excluded the individual. Where grandparental data were unavailable, we used the individual's country of birth.

Total sample size after exclusion: 1,387 subjects
Features: over half a million variable DNA sites in the human genome

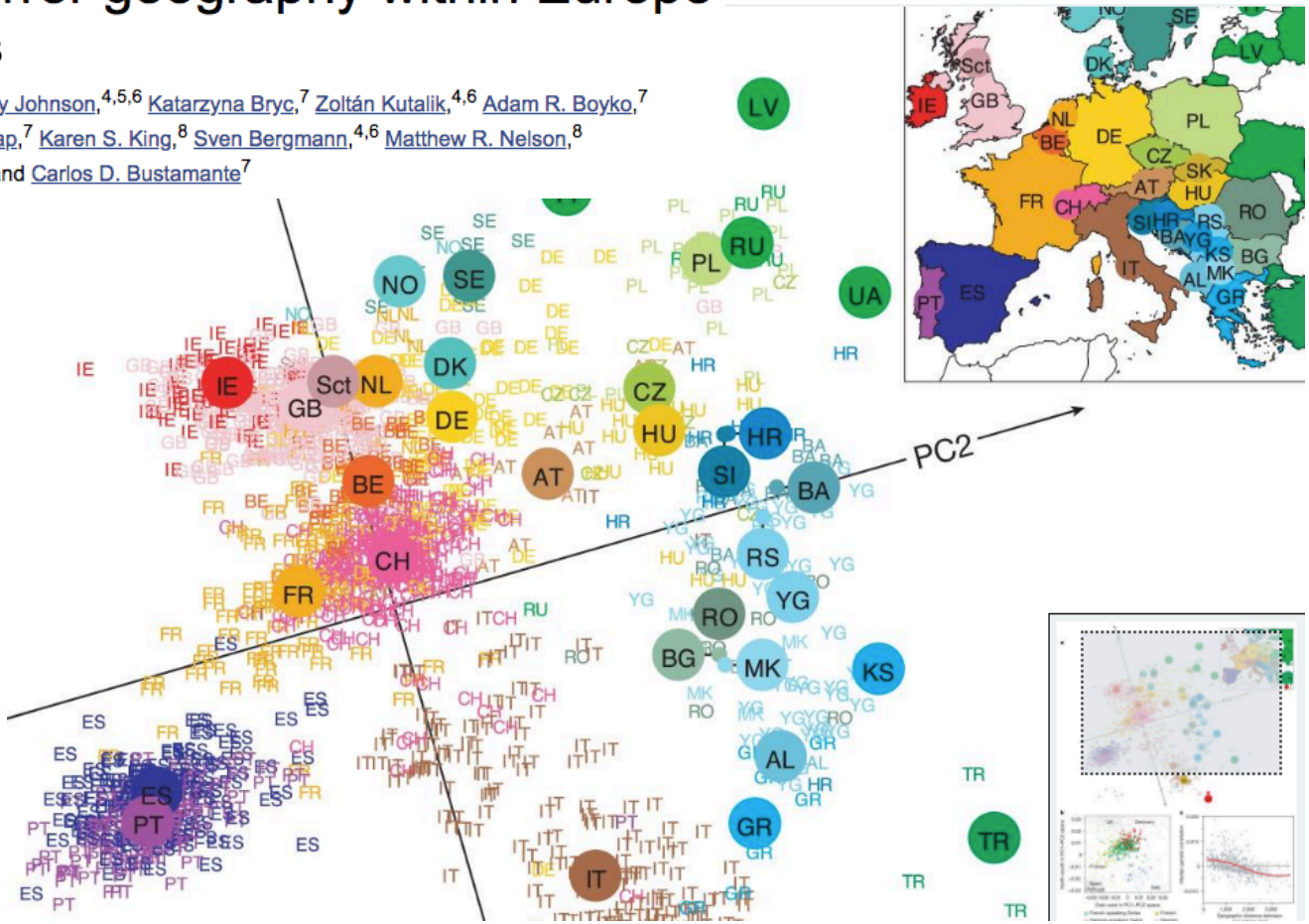


Example: Genes vs. geography

Genes mirror geography within Europe

- Nature, 2008

John Novembre,[1,2] Toby Johnson,[4,5,6] Katarzyna Bryc,[7] Zoltán Kutalik,[4,6] Adam R. Boyko,[7] Adam Auton,[7] Amit Indap,[7] Karen S. King,[8] Sven Bergmann,[4,6] Matthew R. Nelson,[8] Matthew Stephens,[2,3] and Carlos D. Bustamante[7]



Example: Eigen Clothing

A 2D Human Body Model Dressed in Eigen Clothing

Peng Guan* Oren Freifeld† Michael J. Black*

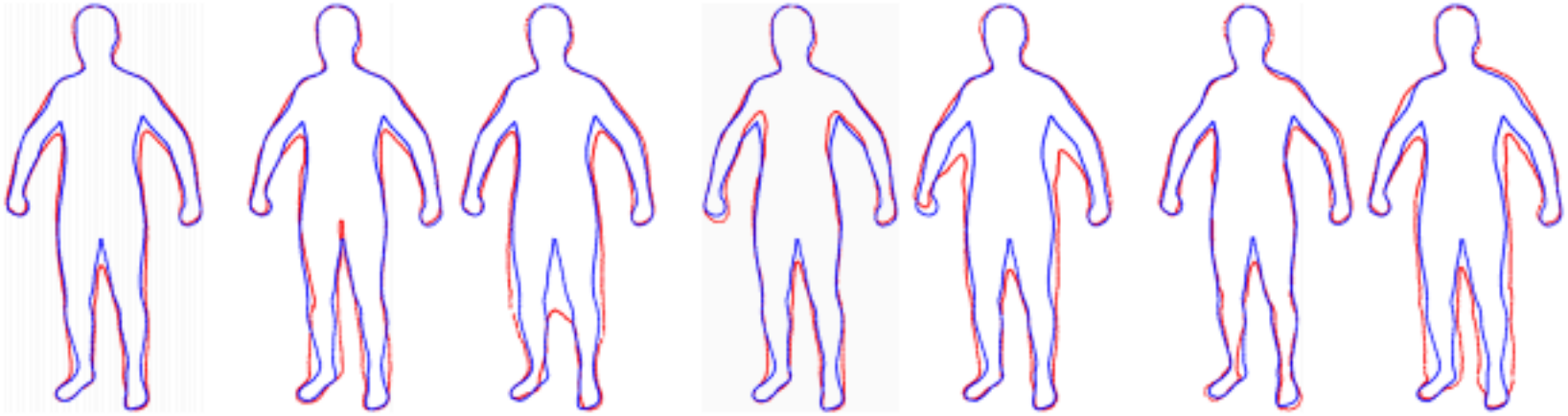


Fig. 4. Eigen clothing. The blue contour is always the same naked shape. The red contour shows the mean clothing contour (a) and ±3 std from the mean for several principal components (b)-(d).

principal components accounting for around 90% of the variance to define the eigen-clothing model. Figure 4 shows the mean and first few clothing eigenvectors for the real data set. This illustrates how the principal components can account for various garments such as long pants, skirts, baggy shirts, etc. Note that

Centering the Data
Goal: each feature's mean = 0.0

Why center?

- Think of mean vector as simplest possible "reconstruction" of a dataset
- No example specific parameters, just one F-dim vector

$$\min_{m \in \mathbb{R}^F} \quad \sum_{n=1}^{N} (x_n - m)^T (x_n - m)$$

$$m^* = \text{mean}(x_1, \ldots x_N)$$

Mean reconstruction

Ex: Viola Jones data set

- 24x24 images of faces = 576 dimensional measurements

Mean

original

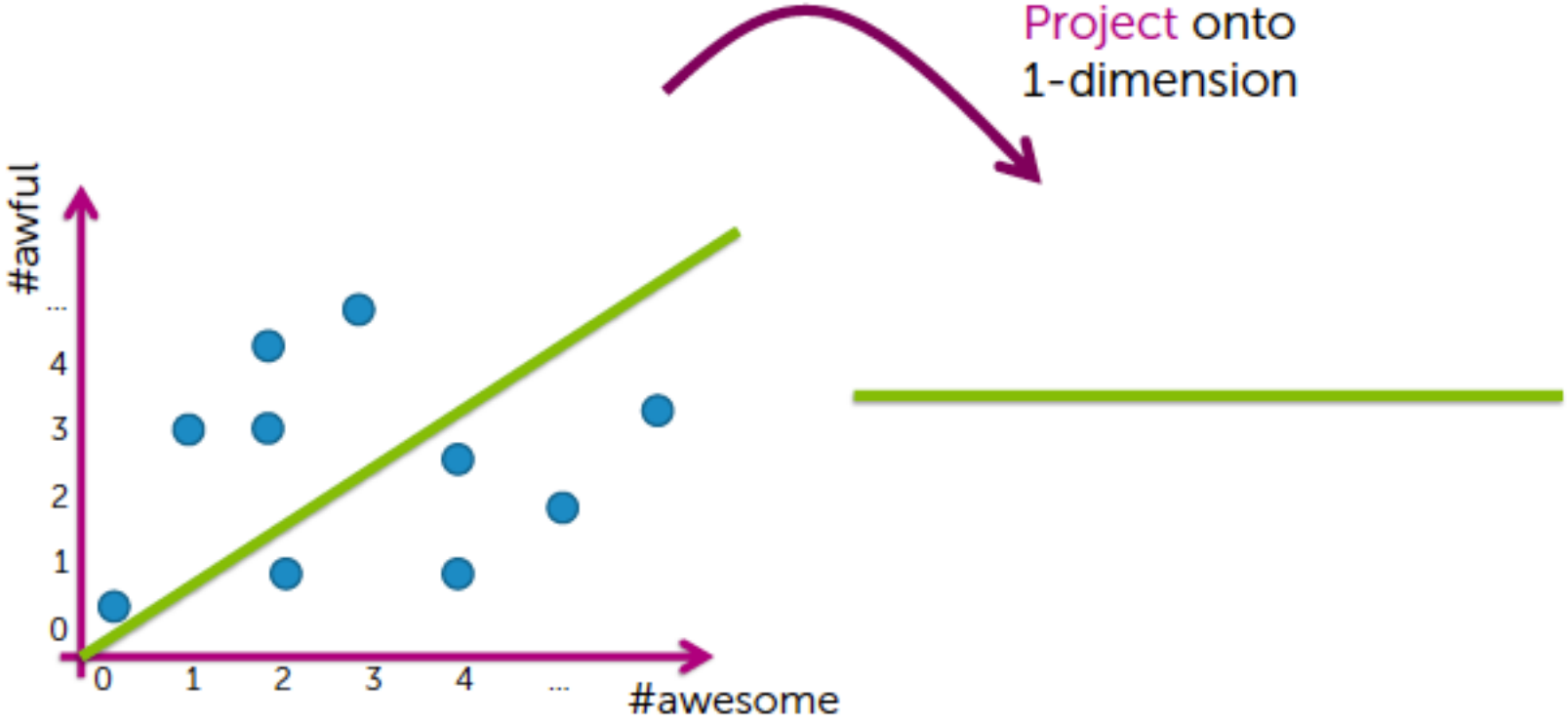
Xi

reconstructed

Mean

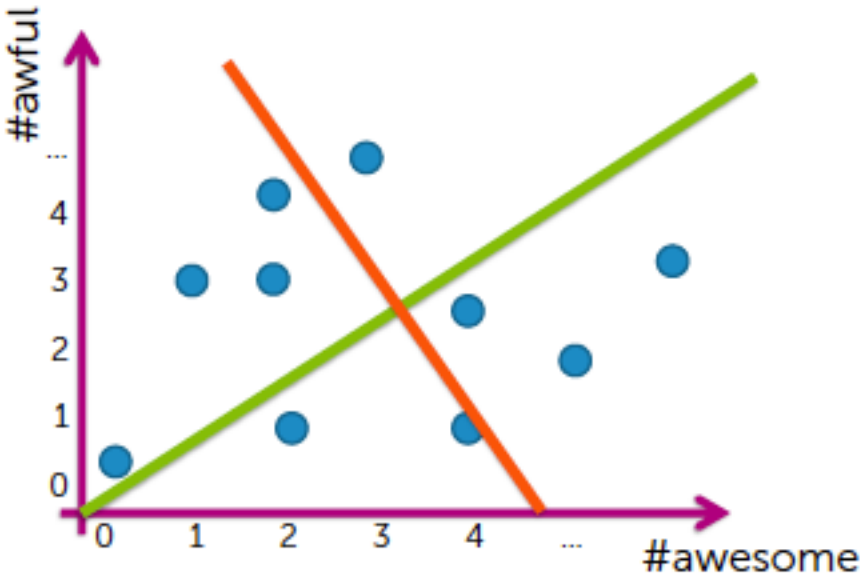
Principal Component Analysis

Linear Projection to 1D



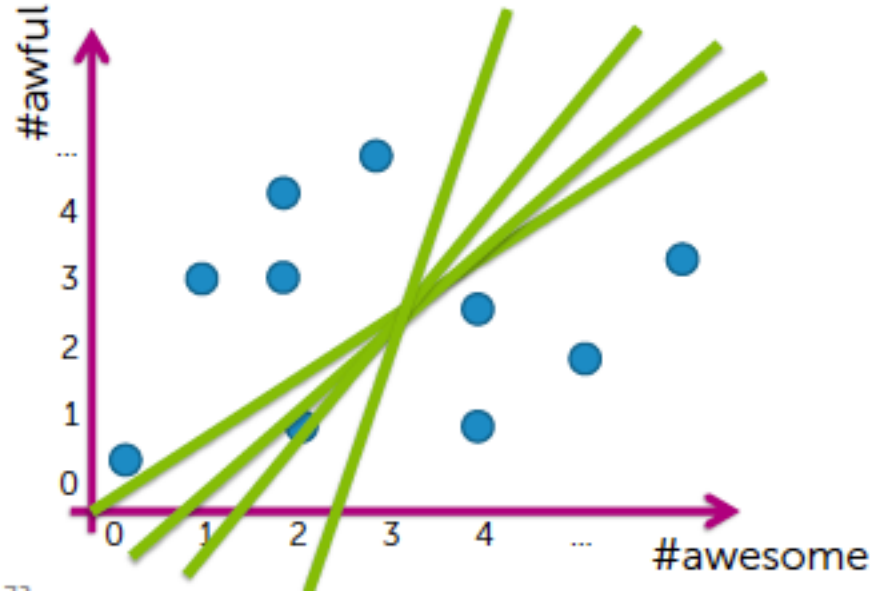
Reconstruction from 1D to 2D

2D Orthogonal Basis



Perfect reconstruction!

Which 1D projection is best?



Idea: Minimize reconstruction error

K-dim Reconstruction with PCA

$$x_i = W z_i + m$$

F vector	F x K	K vector	F vector
High-dim. data	Weights	Low-dim vector	"mean" vector

Problem: Over-parameterized. Too many possible solutions!

If we scale z x2, we can scale W / 2 and get **equivalent** reconstruction

We need to constrain the magnitude of weights.
Let's make all the weight vectors be unit vectors: ||W||_2 = 1

Principal Component Analysis

Training step: .fit()

- Input:
  - X : training data, N x F
    - N high-dim. example vectors
  - K : int, number of components
    - Satisfies 1 <= K <= F

- Output: Trained parameters for PCA
  - m : mean vector, size F
  - W : learned basis of weight vectors, F x K
    - One F-dim. vector (magnitude 1) for each component
    - Each of the K vectors is orthogonal to every other

Principal Component Analysis

Transformation step: .transform()

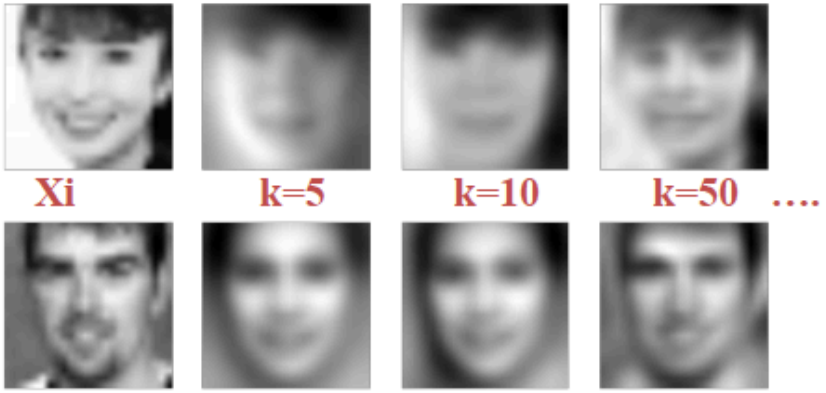
- Input:
  - X : training data, N x F
    - N high-dim. example vectors
  - Trained PCA "model"
    - m : mean vector, size F
    - W : learned basis of eigenvectors, F x K
      - One F-dim. vector (magnitude 1) for each component
      - Each of the K vectors is orthogonal to every other
- Output:
  - Z : projected data, N x K

Example: EigenFaces

Ex: Viola Jones data set

- 24x24 images of faces = 576 dimensional measurements
- Take first K PCA components

Projecting data onto first k dimensions

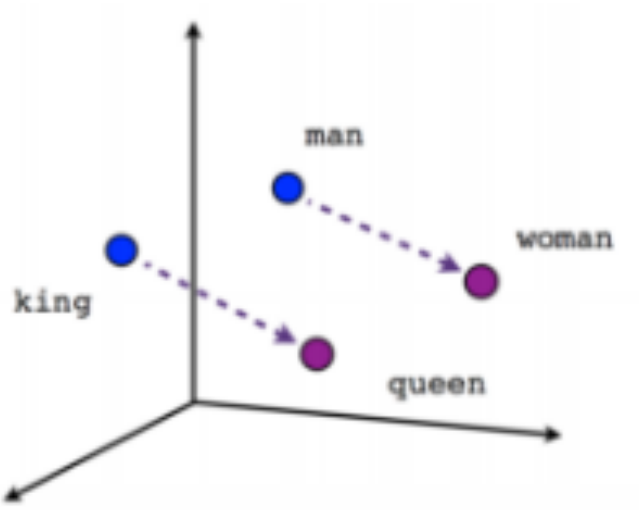


Word Embeddings

Word Embeddings (word2vec)

Goal: map each word in vocabulary to an embedding vector

- Preserve semantic meaning in this new vector space



Male-Female

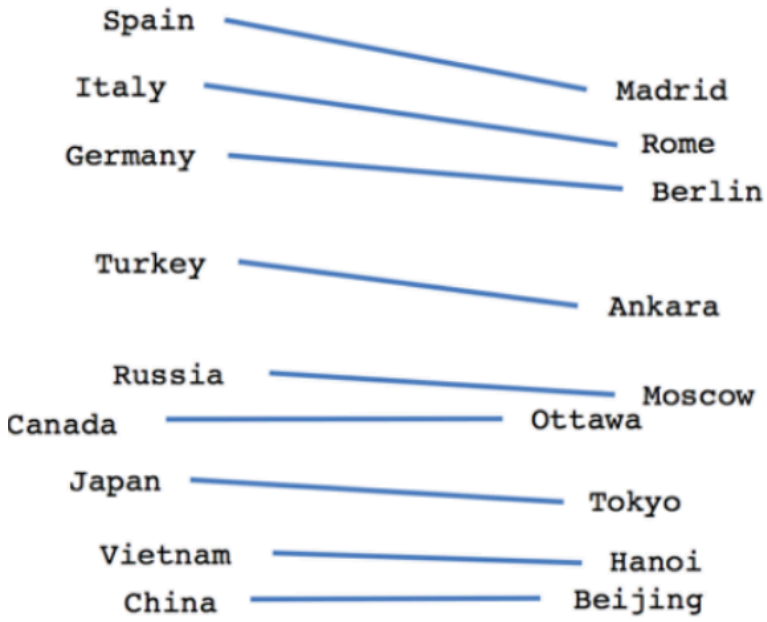
Verb tense

vec(swimming) – vec(swim) + vec(walk) = vec(walking)

Word Embeddings (word2vec)

Goal: map each word in vocabulary to an embedding vector

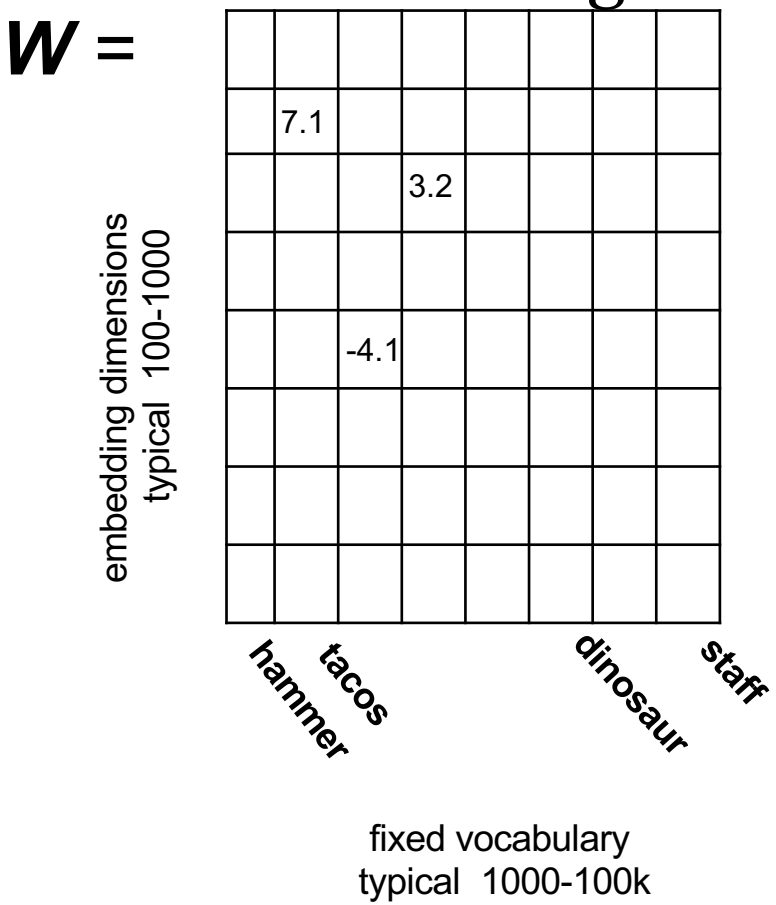
- Preserve semantic meaning in this new vector space



Country-Capital

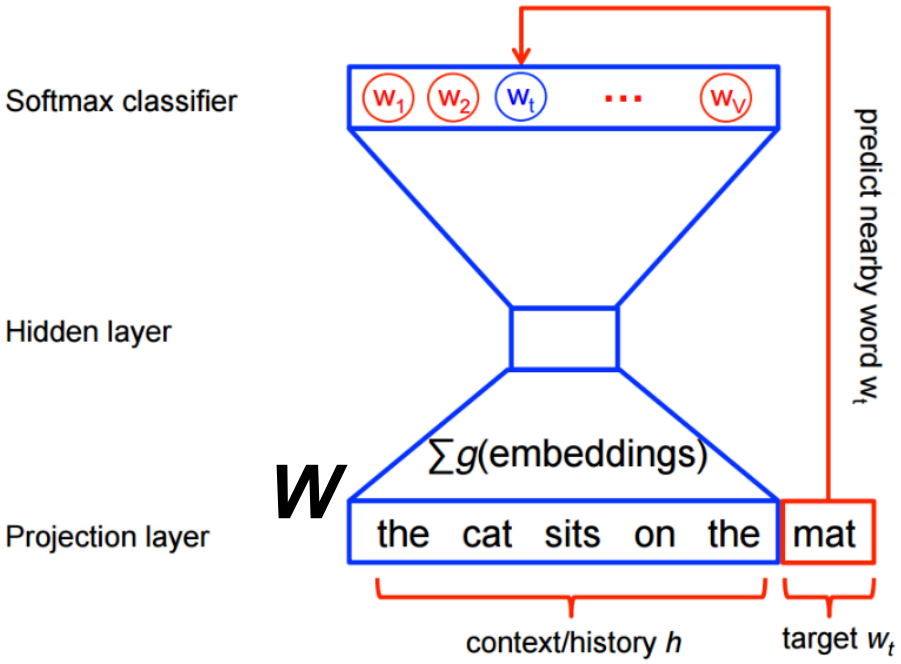
How to embed?

Goal: learn weights



Training

Reward embeddings that predict nearby words in the sentence.



Credit:
https://www.tensorflow.org/tutorials/representation/word2vec

Dim. Reduction/Embedding
Unit Objectives

- Goals of dimensionality reduction
  - Reduce feature vector size (keep signal, discard noise)
  - "Interpret" features: visualize/explore/understand
- Common approaches
  - Principal Component Analysis (PCA)
  - word2vec and other neural embeddings
- Evaluation Metrics
  - Storage size
  - "Interpretability"

- Reconstruction error