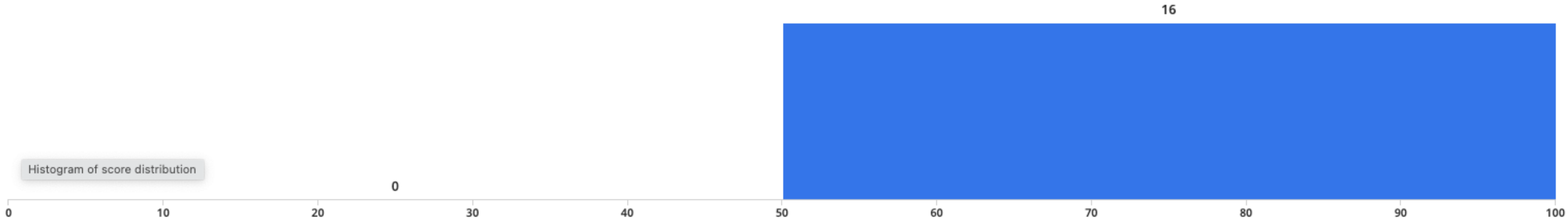
PROPOSAL GRADES POSTED

Review Grades for Project Proposal

● Regrade Requests Open ● Grades Published



Minimum	Median	Maximum	Mean	Std Dev
95.0	98.5	100.0	98.0	1.51

SPOTLIGHT REMINDER

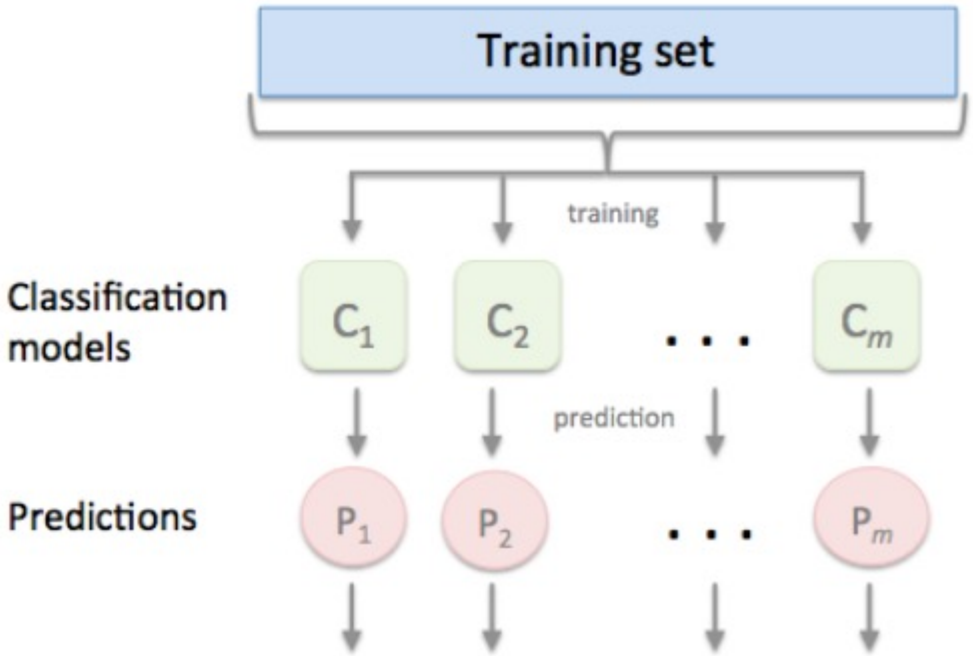
- Spotlight PPT slide due 10/30 on Canvas
- Project spotlight presentation on 11/1
  - 2 min each group
  - 1 min between groups
  - Group order 16 -> 1

ENSEMBLE METHODS

- Ensemble of same classifiers
  - Bagging and random forest
  - Boosting and gradient boosted tree
- Ensemble of different classifiers

VOTING

- Voting ensembles mimic error-correcting codes
  - More voters —> potential better signal to noise
  - Lower correlation between models



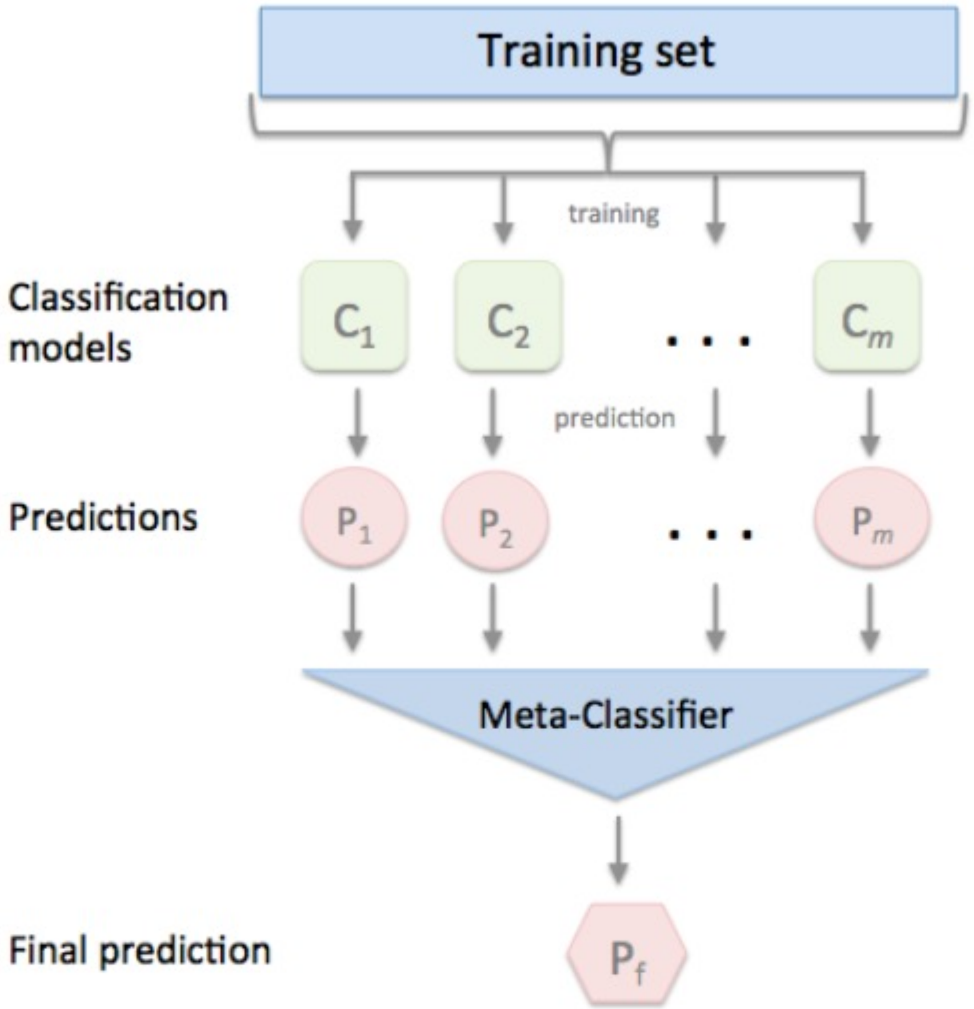
How do we combine the predictions?

STACKING

- Introduced by Wolpert, 1992

- Use a pool of base classifiers and do out-of-fold predictions, then train a meta-classifier to combine their predictions

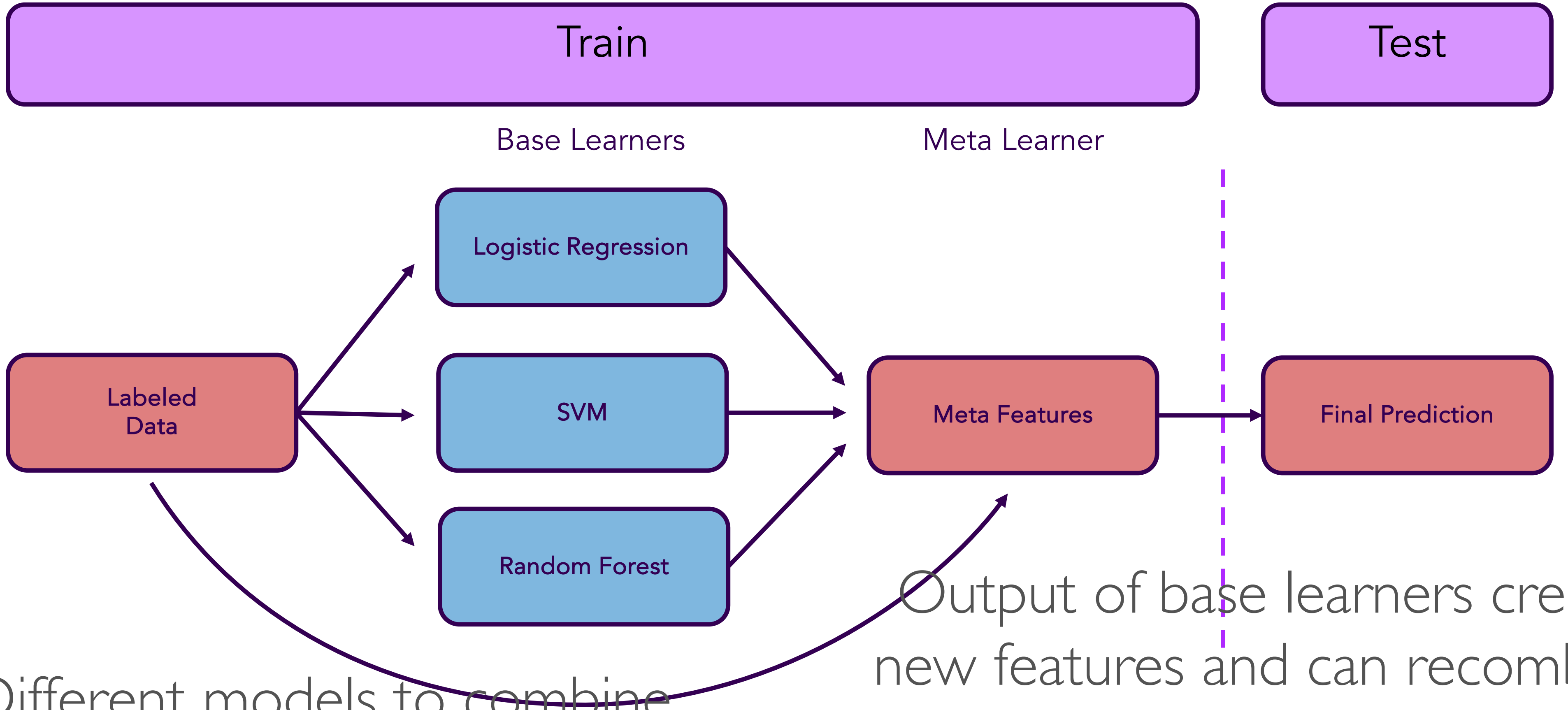
- Stacker model gains information by using first-stage predictions as features



BLENDING

- Close to stacked generalization, but a bit simpler
- Instead of out-of-fold predictions, create small holdout set that the stacker is then trained on this set
- Disadvantages:
  - Less data used overall
  - Final model may overfit to holdout set

EXAMPLE: COMBINING DIFFERENT CLASSIFIERS



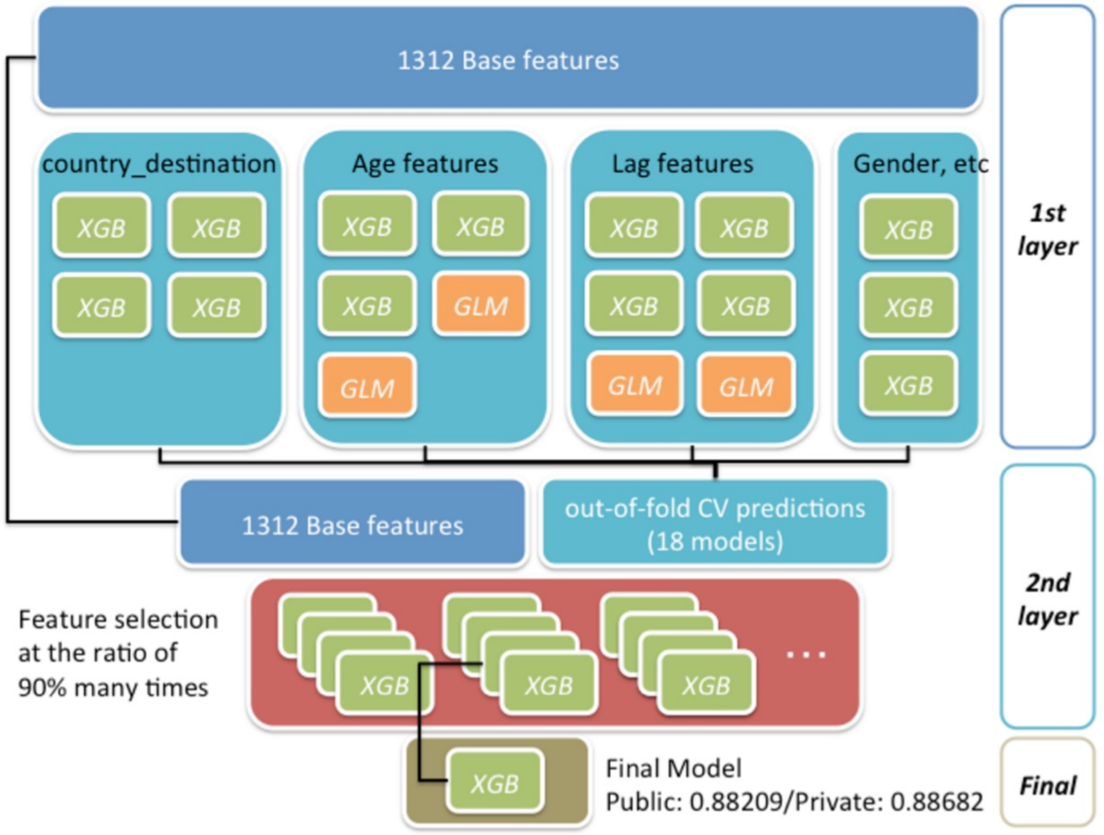
Different models to combine

Output of base learners create new features and can recombine with original data

STACKING AND BLENDING

- Why stop at two stages? Why not combine multiple ensembles models?

EXAMPLE: AIRBNB 2ND PLACE



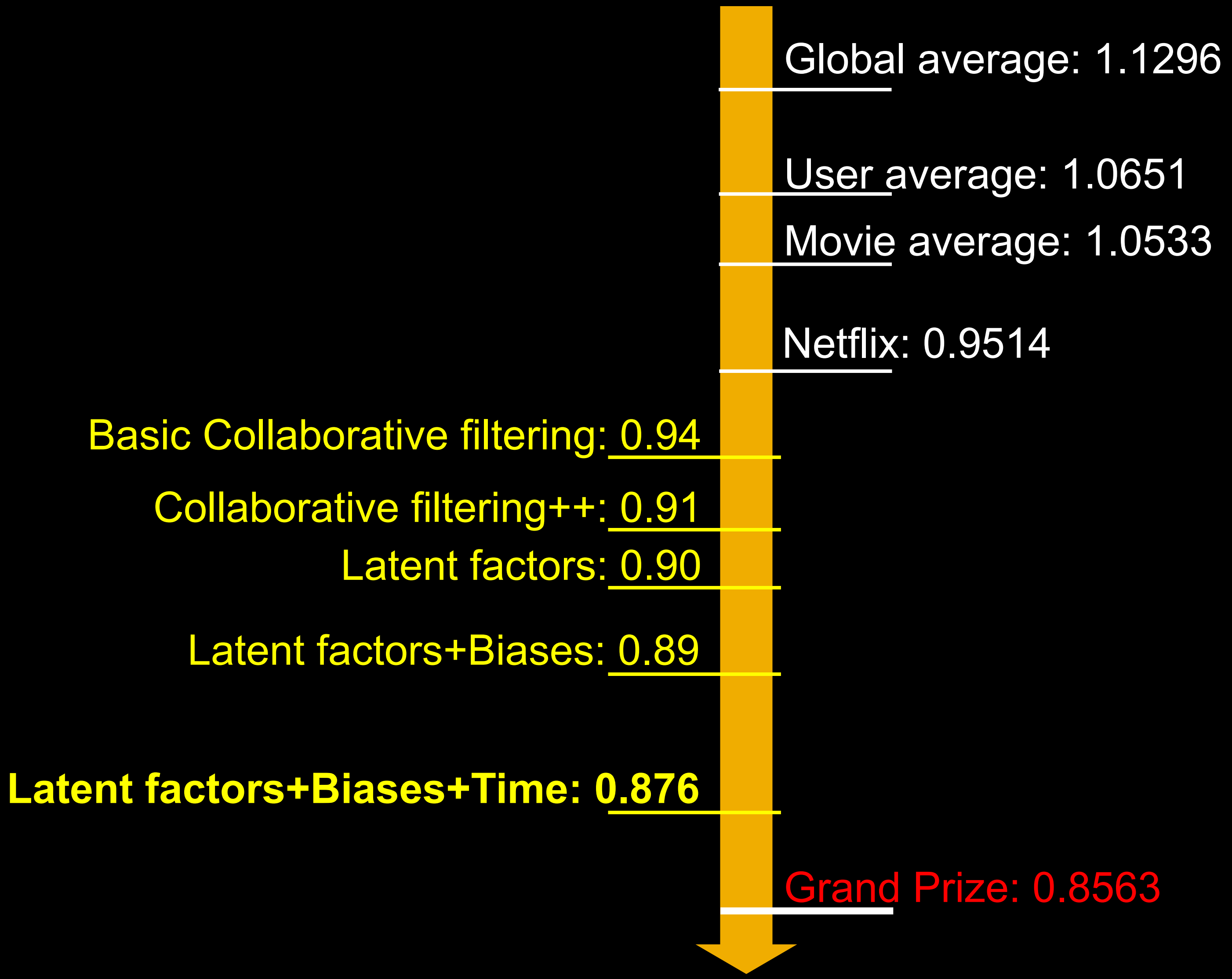
LEARNING ARCHITECTURE

https://github.com/Keiku/kaggle-airbnb-recruiting-new-user-bookings

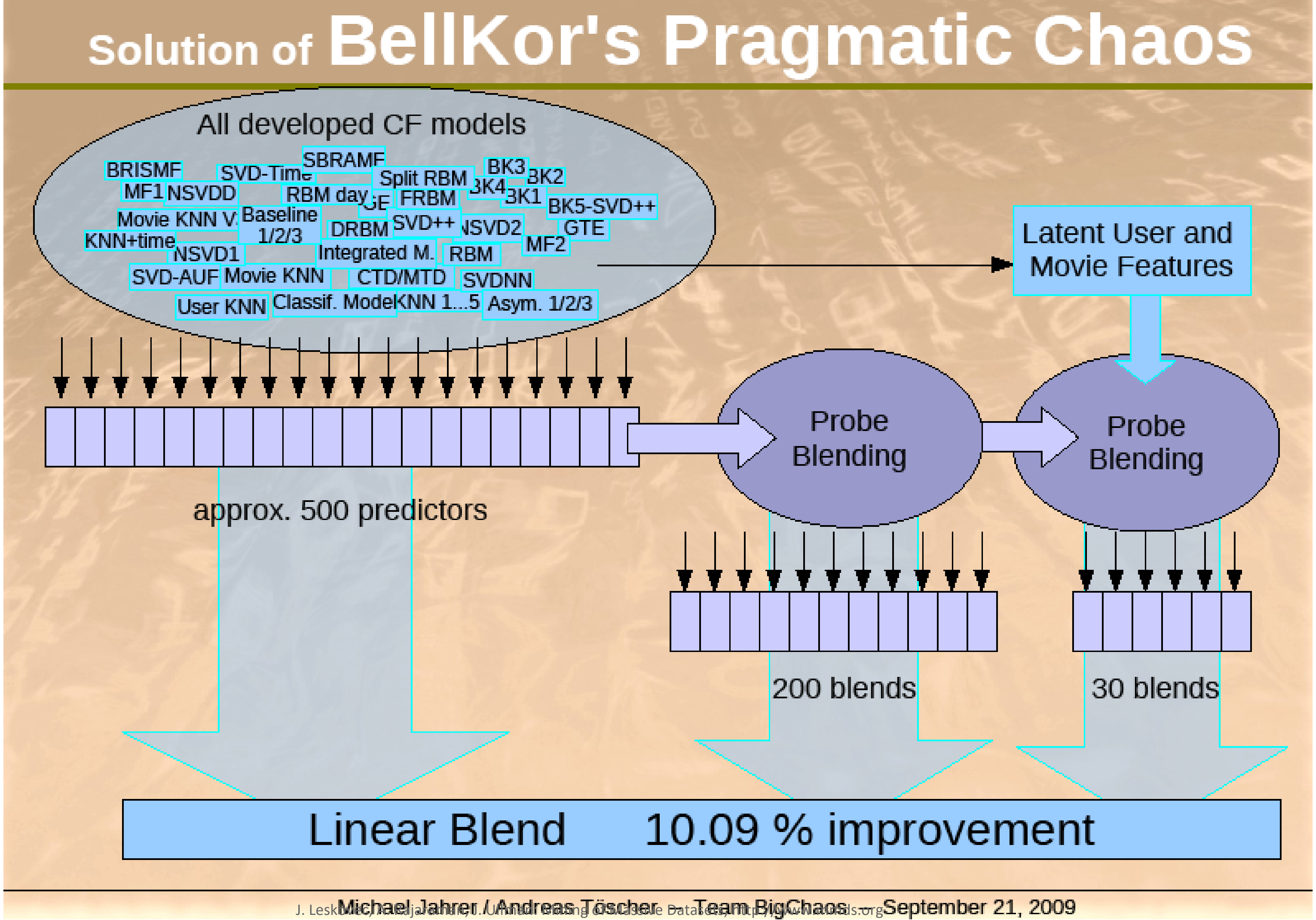
The Netflix Prize

- An open competition 2006 – 2009 held by Netflix
- **Training data**
  - 100 million ratings (with timestamps), 480,000 users, 17,770 movies
  - 6 years of data: 2000-2005
- **Test data**
  - Last few ratings of each user (2.8 million)
- **Evaluation criterion:** Root Mean Square Error (RMSE) $= \frac{1}{|R|}\sqrt{\sum_{(i,x)\in R}(\hat{r}_{xi} - r_{xi})^2}$
- **Competition**
  - **Netflix's system Cinematch RMSE: 0.9514**
  - **$1 million** prize for 10% improvement on Netflix
  - $50,000 progress prize every year

Netflix Prize



The big picture



DIMENSIONALITY REDUCTION

CS 334: Machine Learning

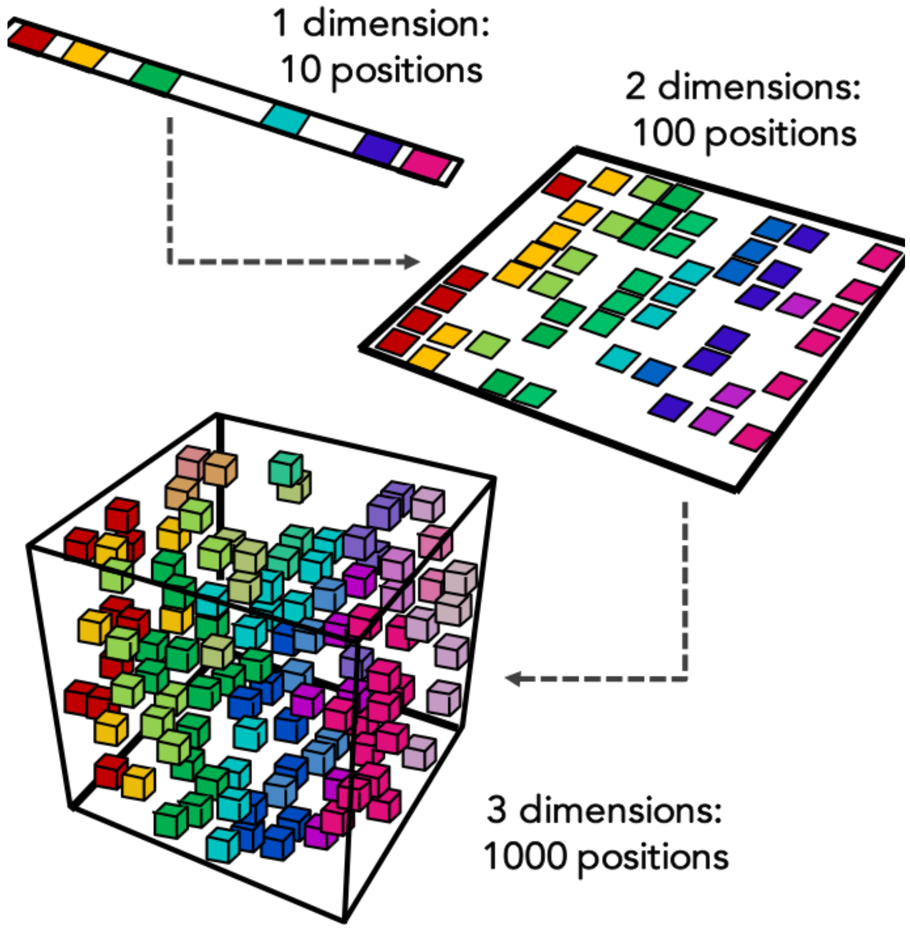
TYPES OF UNSUPERVISED LEARNING

Clustering — identify unknown structure in the data

Dimensionality Reduction — use structural characteristics to simplify data

CURSE OF DIMENSIONALITY

- Increasing features should improve performance right?
- In practice, too many features leads to worse performance
- # of training examples required increases exponentially with the # of features

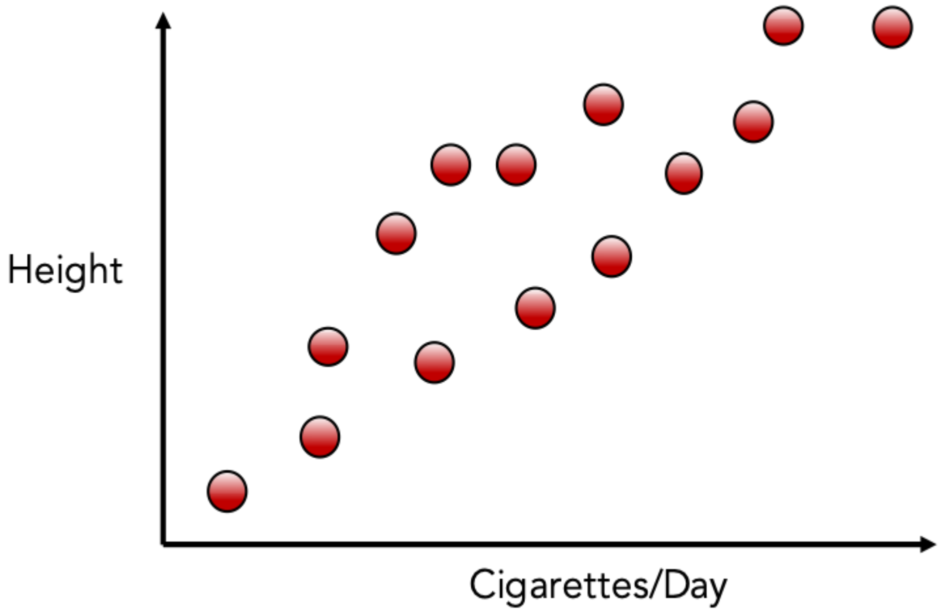


FEATURE SELECTION/REDUCTION

- Feature selection
  - Filter methods (agnostic to the learning algorithm)
  - Wrapper methods (keep model in the loop)
  - Embedded methods (e.g. Lasso regularization)
- Dimension reduction

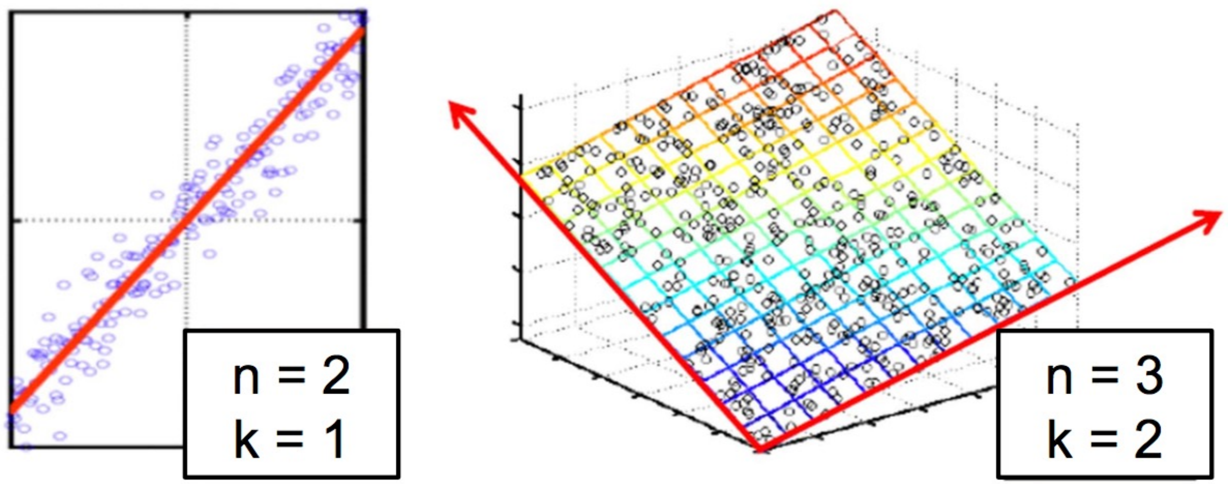
MOTIVATION: EXAMPLE

- Two features: Height and cigarettes per day
- Both features are correlated
- Can we reduce the features to one?



DIMENSIONALITY REDUCTION

- Represent data with fewer dimensions
- Discover "intrinsic dimensionality" of data
- New feature space: linear / non-linear combination of original features



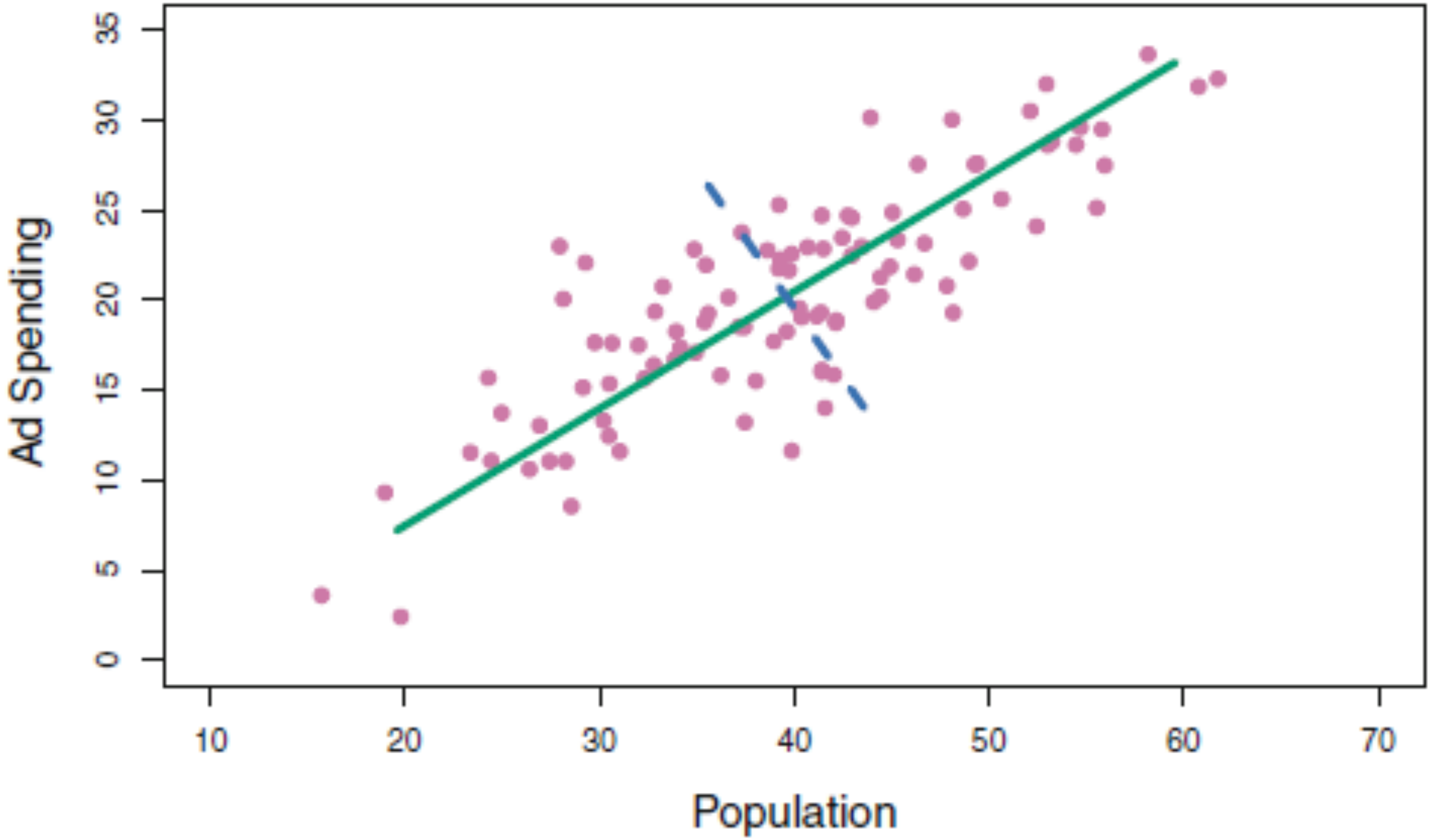
Slide by Yi Zhang

WHY DIMENSIONALITY REDUCTION

- Simplified data processing
- Noise reduction (removing feature redundancy)
- Easier learning — less parameters
- Robust learning — numerical stability due to less correlations
- Easier visualization — show high dimensional data in 2D or 3D

GROUP ACTIVITY

WHICH IS THE BEST PROJECTION AND WHY?

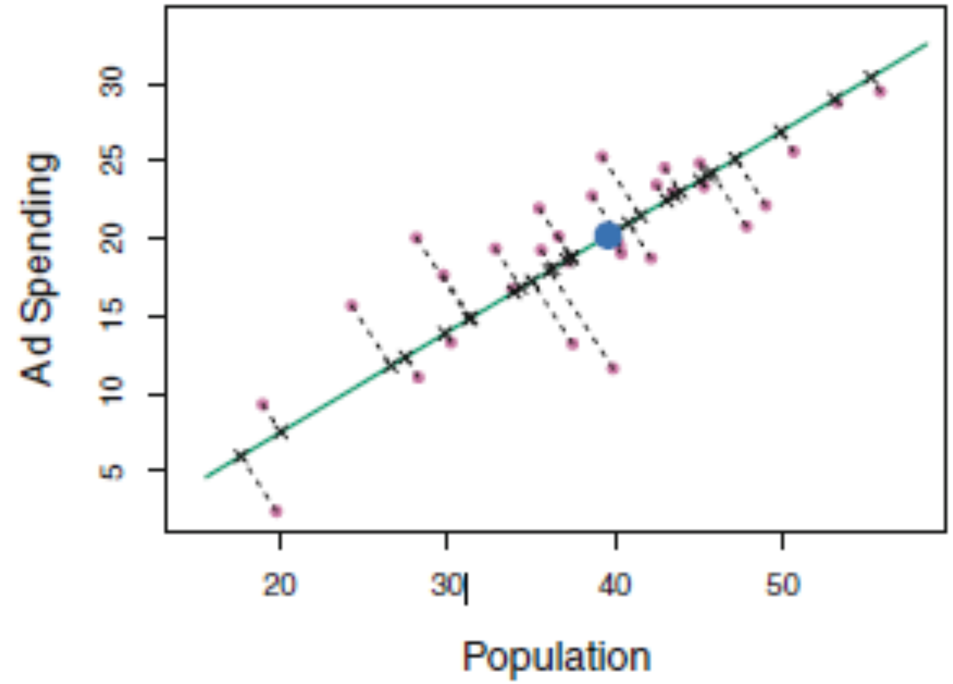


PRINCIPAL COMPONENT ANALYSIS (PCA)

- Developed by Pearson in 1901
- Popular and widely studied
- Finds sequence of linear combinations of the features (also known as principal components) that have maximal variance and are uncorrelated

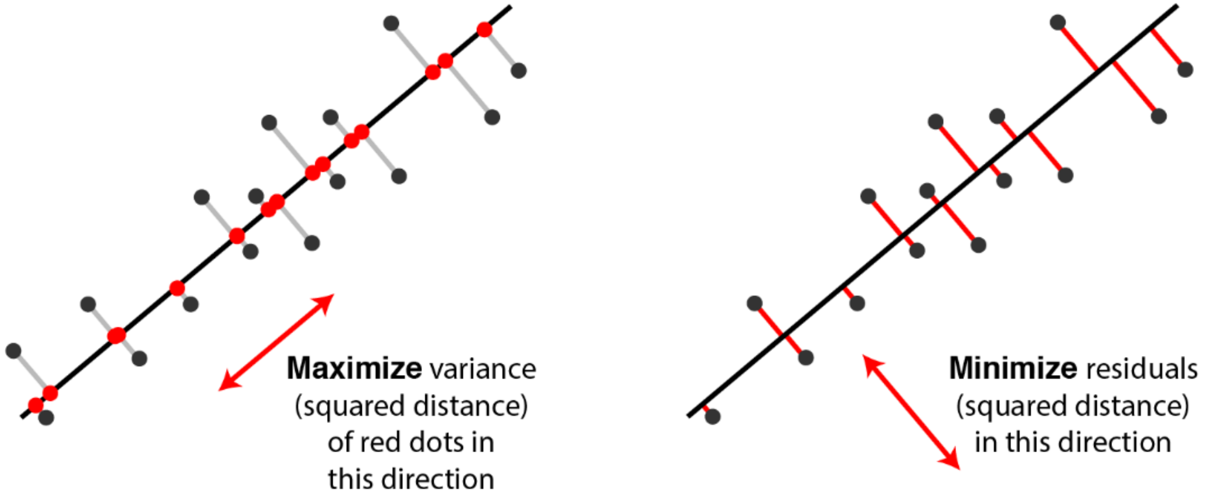
PRINCIPAL COMPONENT ANALYSIS (PCA)

- First principal component
  - Yields the highest variance of the projection
  - Minimizes sum of squared perpendicular distances (reconstruction error between data and projected points)



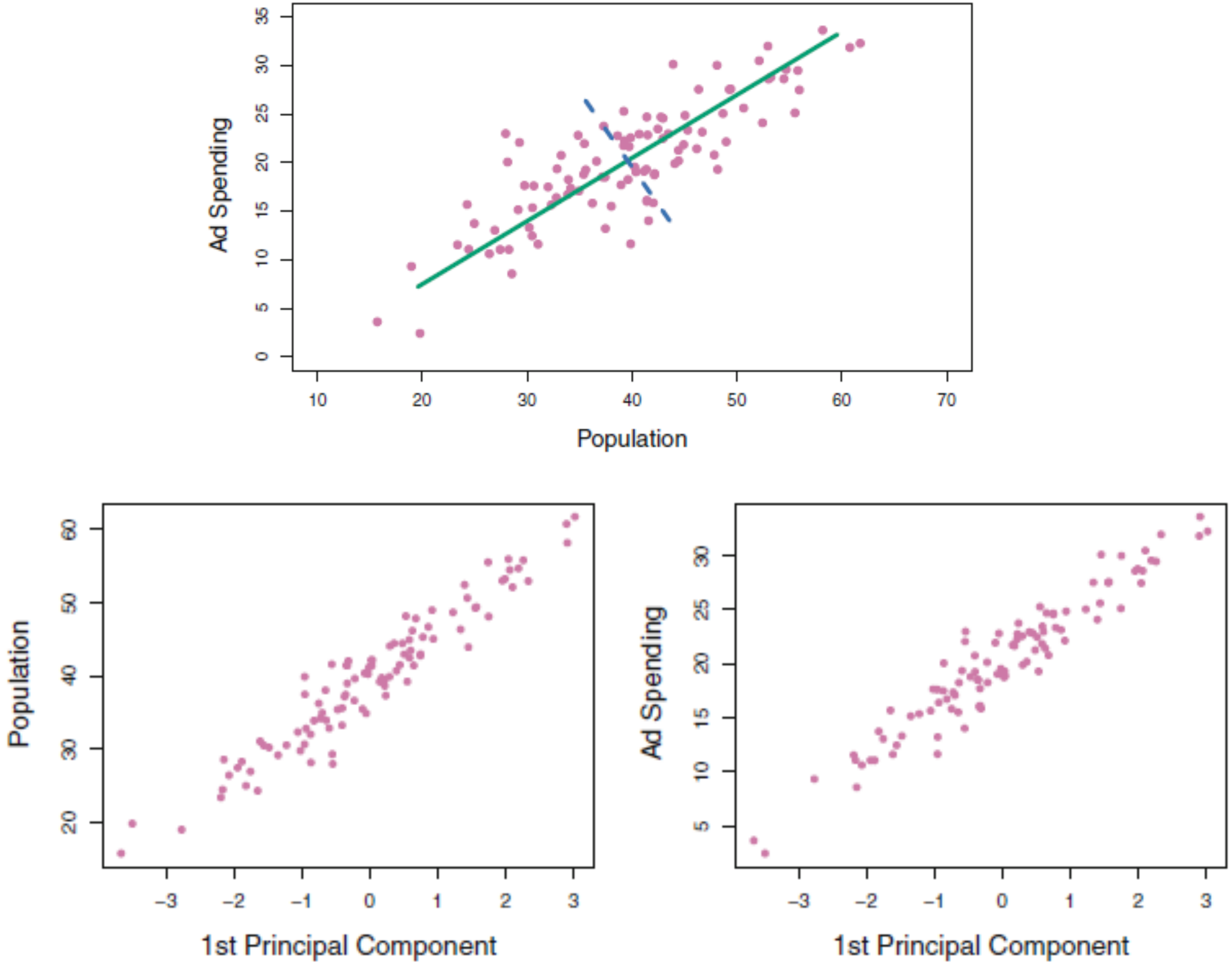
TWO INTERPRETATIONS OF PCA

- Maximize the variance of projection along principal component
- Minimize the reconstruction error between original data and projected components



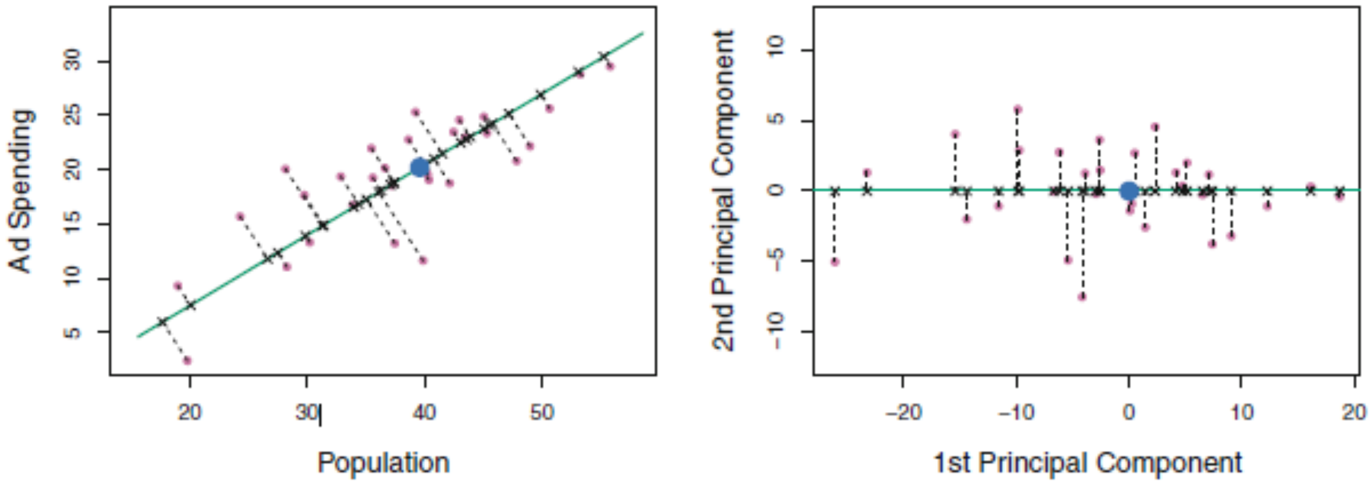
http://alexhwilliams.info/itsneuronalblog/2016/03/27/pca/#some-things-you-maybe-didnt-know-about-pca

1ST PRINCIPAL COMPONENT

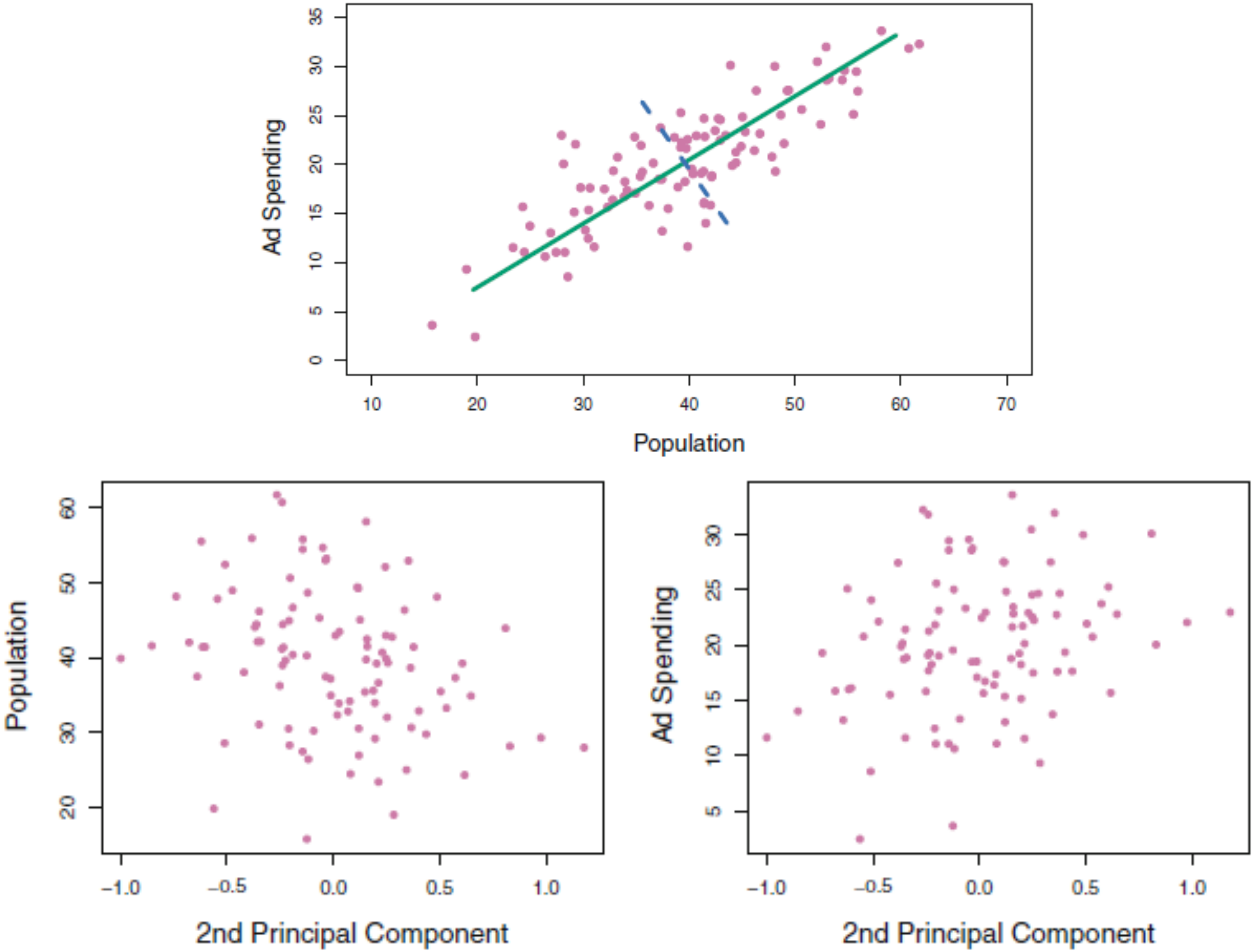


2ND PRINCIPAL COMPONENT

- Second principal component
  - Orthogonal to the first principal component
  - And has largest variance
- In general, we can construct up to p principal components (p features)



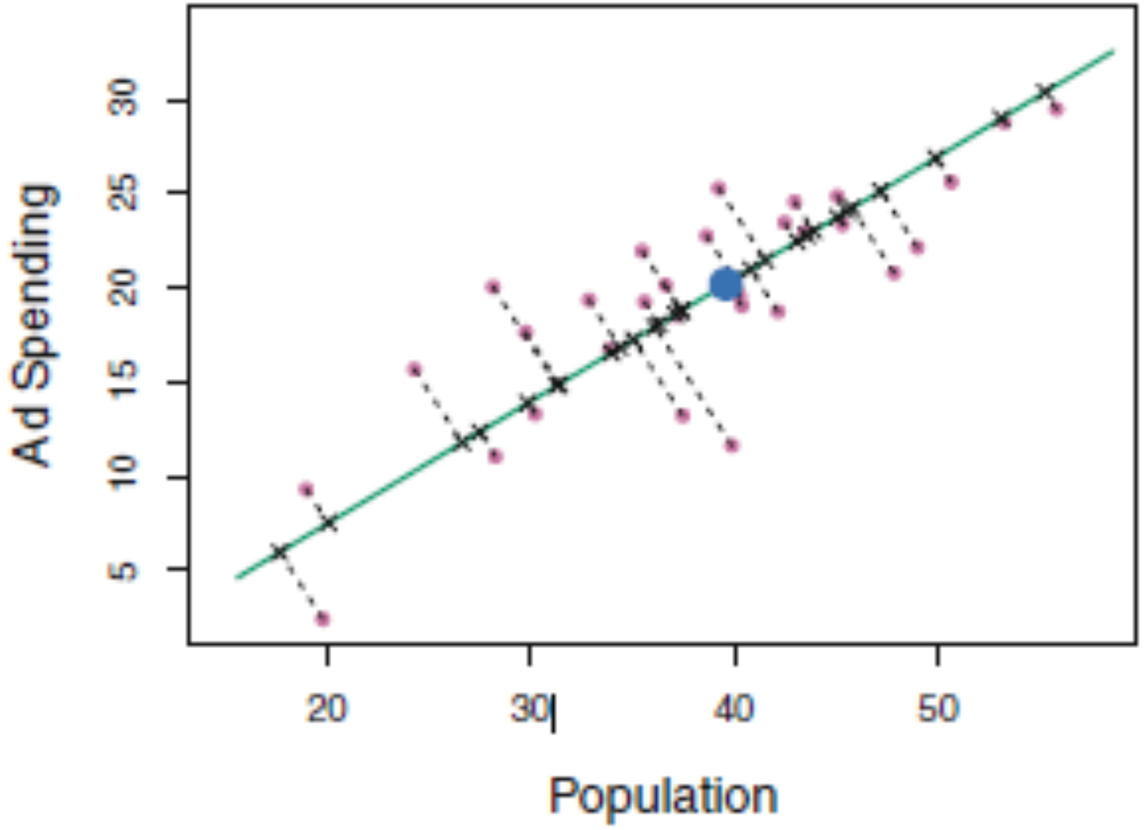
2ND PRINCIPAL COMPONENT



PRINCIPAL COMPONENT ANALYSIS (PCA)

- Principal component loadings
  - Determines the axis
  - v = (0.839, 0.544)
- Principal component scores
  - Projected position of each point on the axis
  - Z = Xv

  (Data is centered first)



$$Z_1 = 0.839 \times (\text{pop} - \overline{\text{pop}}) + 0.544 \times (\text{ad} - \overline{\text{ad}}).$$

PROJECTION ONTO UNIT VECTORS

- Definition of dot product:

$$\mathbf{A} \cdot \mathbf{B} = ||\mathbf{A}||_2||\mathbf{B}||_2 \cos\theta$$

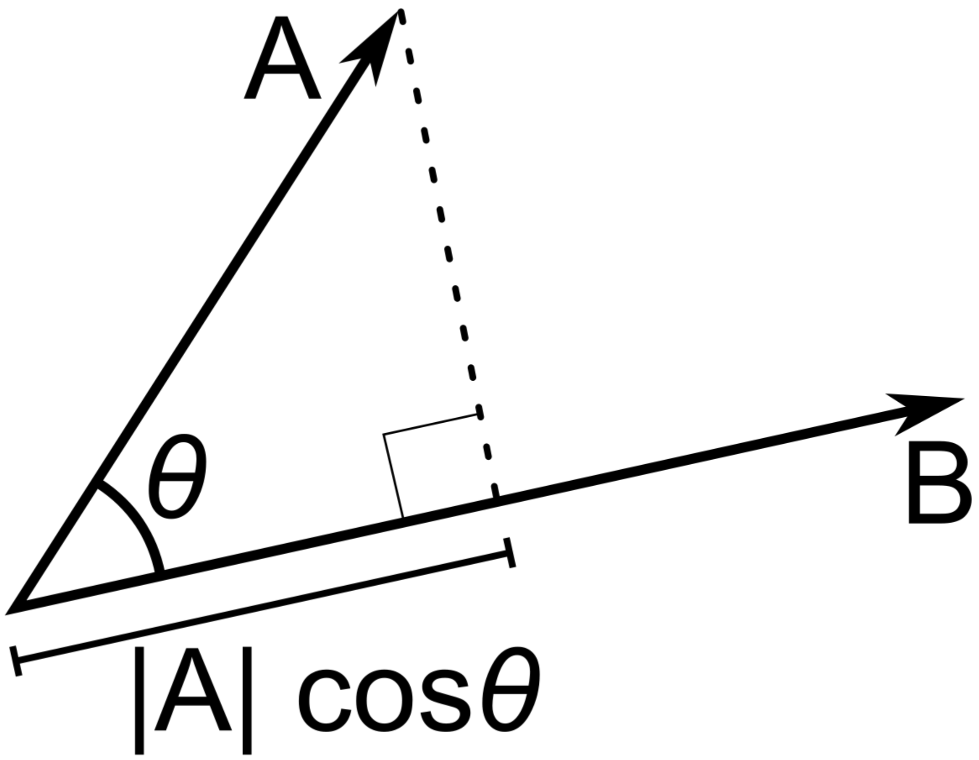
- If B is a unit vector, dot product is length of the projection

$$\mathbf{A} \cdot \mathbf{B} = ||\mathbf{A}||_2 \cos\theta$$

- Projection of A onto B (vector):

$$\boxed{(\mathbf{A} \cdot \mathbf{B})}\mathbf{B}$$

Coefficient / score



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

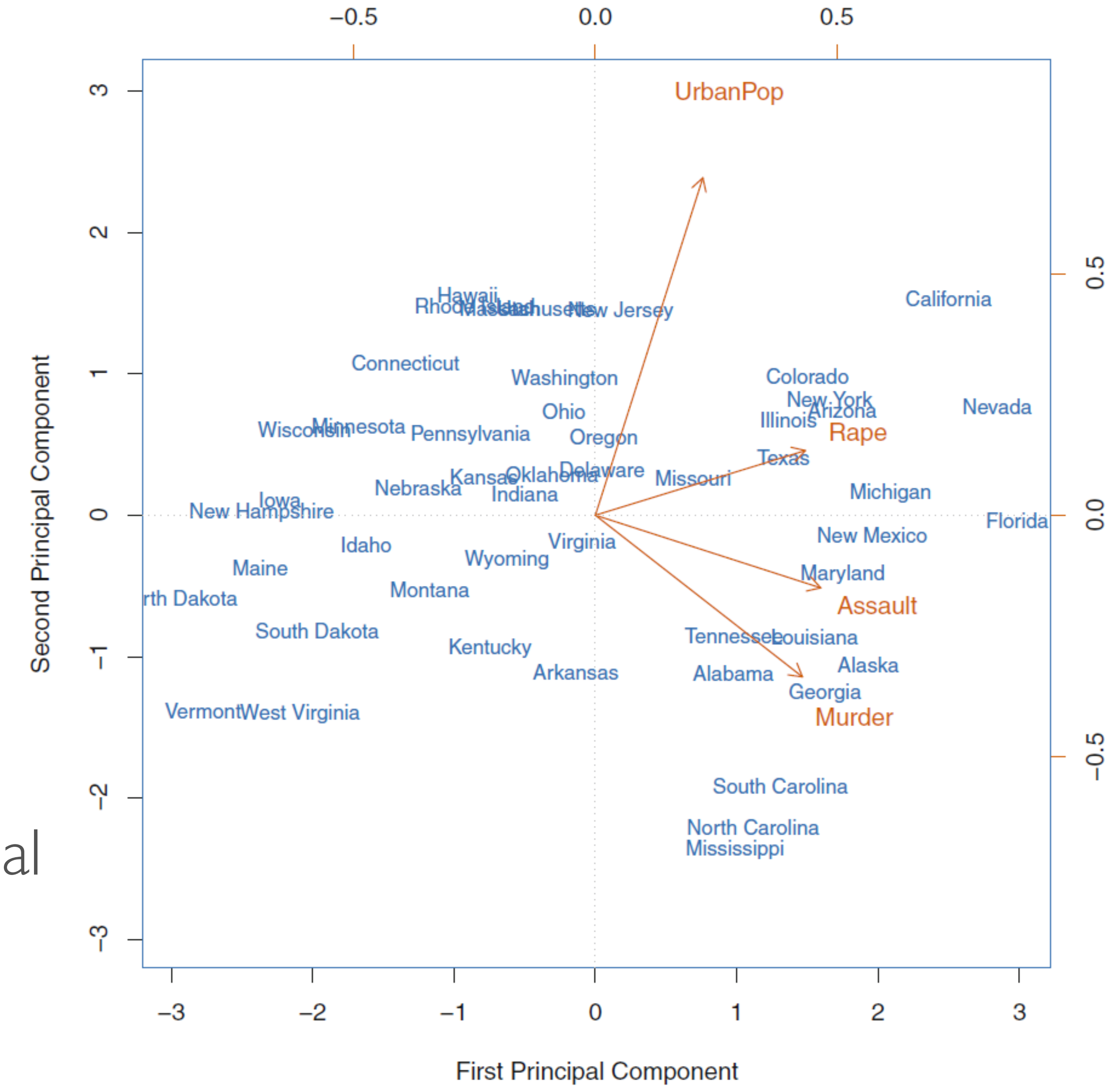
EXAMPLE: US ARRESTS DATASET

- 50 states with 4 features
  - Number of arrests per 100,000 residents for Murder, Assault, Rape
  - Percent of population living in urban areas (UrbanPop)

1		Murder	Assault	UrbanPop	Rape
2	Alabama	13.2	236	58	21.2
3	Alaska	10	263	48	44.5
4	Arizona	8.1	294	80	31
5	Arkansas	8.8	190	50	19.5
6	California	9	276	91	40.6
7	Colorado	7.9	204	78	38.7
8	Connecticut	3.3	110	77	11.1
9	Delaware	5.9	238	72	15.8
10	Florida	15.4	335	80	31.9
11	Georgia	17.4	211	60	25.8

EXAMPLE: US ARRESTS (BIPLOT)

	PC1	PC2
Murder	0.5358995	−0.4181809
Assault	0.5831836	−0.1879856
UrbanPop	0.2781909	0.8728062
Rape	0.5434321	0.1673186



Biplot: displays both principal component scores and principal component loadings.

PCA: 1ST PC

- 1st PC of X is unit vector that maximizes the sample variance compared to all other unit vectors

$$\mathbf{v}_1 = \mathrm{argmax}_{||\mathbf{v}||_2=1} (\mathbf{X}\mathbf{v})^\top (\mathbf{X}\mathbf{v})$$

- 1st PC score: $\mathbf{X}\mathbf{v}_1$
- Variance explained by first PC: $(\mathbf{X}\mathbf{v}_1)^\top (\mathbf{X}\mathbf{v}_1)/n$

PCA: NEXT PC

- Idea: Successively find orthogonal directions of highest variance
- Why orthogonal?
  - Want to minimize redundancy
  - Want to look at variance in different direction
  - Computation is easier

PCA: 2ND PC

- 2nd PC of X is unit vector that is orthogonal to the 1st PC such that it maximizes the sample variance compared to all other unit vectors that are orthogonal to the 1st PC

$$\mathbf{v}_2 = \mathrm{argmax}_{||\mathbf{v}||_2=1, \mathbf{v}^\top \mathbf{v}_1 = 0} (\mathbf{X}\mathbf{v})^\top (\mathbf{X}\mathbf{v})$$

- 2nd PC score: $\mathbf{X}\mathbf{v}_2$

- Variance explained by 2nd PC: $(\mathbf{X}\mathbf{v}_2)^\top (\mathbf{X}\mathbf{v}_2)/n$

PCA: PROBLEM FORMULATION

- Recall 1st and 2nd PC

$$\mathbf{v}_1 = \underset{||\mathbf{v}||_2=1}{\operatorname{argmax}} (\mathbf{X}\mathbf{v})^\top (\mathbf{X}\mathbf{v})$$

$$\mathbf{v}_2 = \underset{||\mathbf{v}||_2=1, \mathbf{v}^\top \mathbf{v}_1 = 0}{\operatorname{argmax}} (\mathbf{X}\mathbf{v})^\top (\mathbf{X}\mathbf{v})$$

- Find orthonormal vectors that maximizes variance (assuming **X** is zero-centered)

PCA: PROBLEM FORMULATION

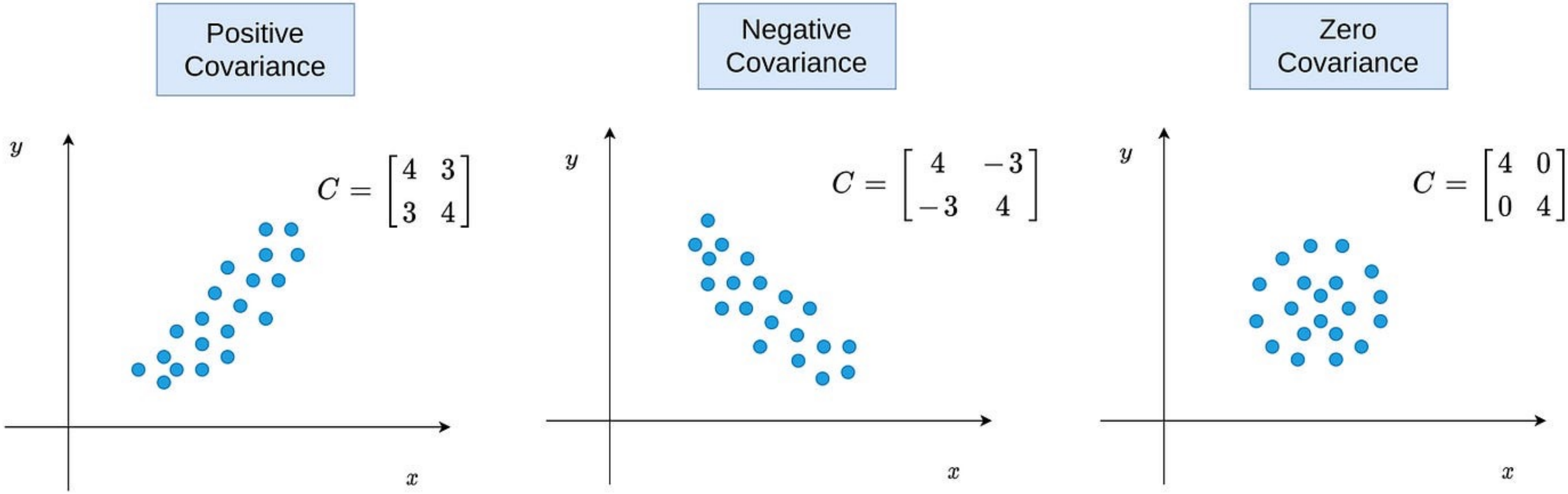
- Given a feature matrix **X** with n data points, find **W** such that $||\mathbf{W}||_2 = 1$ and the Var(**XW**) is maximized and **W** consists of orthonormal vectors

$$\begin{aligned}\mathrm{Var}(\mathbf{XW}) &= \frac{1}{N}(\mathbf{W}^\top(\mathbf{X}-\mu_\mathbf{X})^\top(\mathbf{X}-\mu_\mathbf{X})\mathbf{W}) \\ &= \mathbf{W}^\top \boxed{\mathbf{\Sigma}_\mathbf{X}} \mathbf{W}\end{aligned}$$

Sample covariance matrix

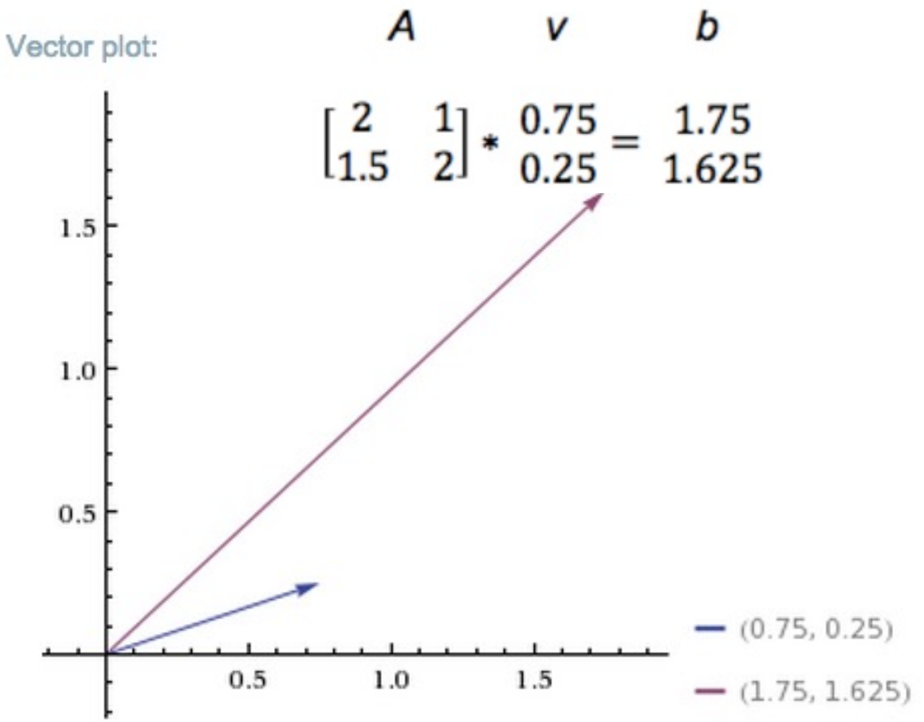
- The solution is the Eigenvectors of the covariance matrix

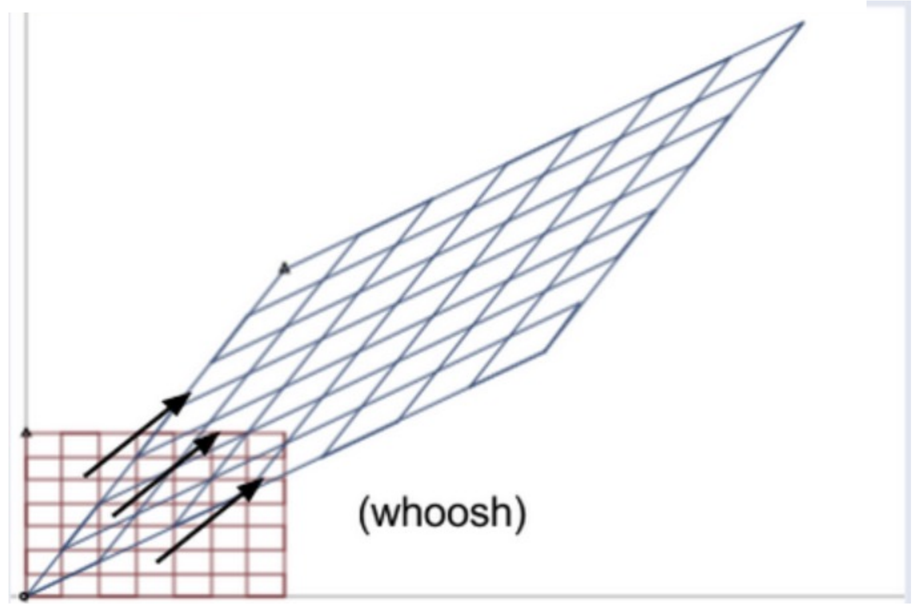
EXAMPLE: COVARIANCE MATRIX



REVIEW: EIGENVALUES + EIGENVECTORS

- Eigenvector and eigenvalue $\mathbf{A}\mathbf{x} = \lambda\mathbf{x}$
- Analogy: Matrix is a gust of wind (invisible force with visible result)
  - Eigenvector is like a weathervane which tells you the direction the wind is blowing in
  - Eigenvalue is just the scalar coefficient
- Eigenvectors of a symmetric matrix are orthonormal





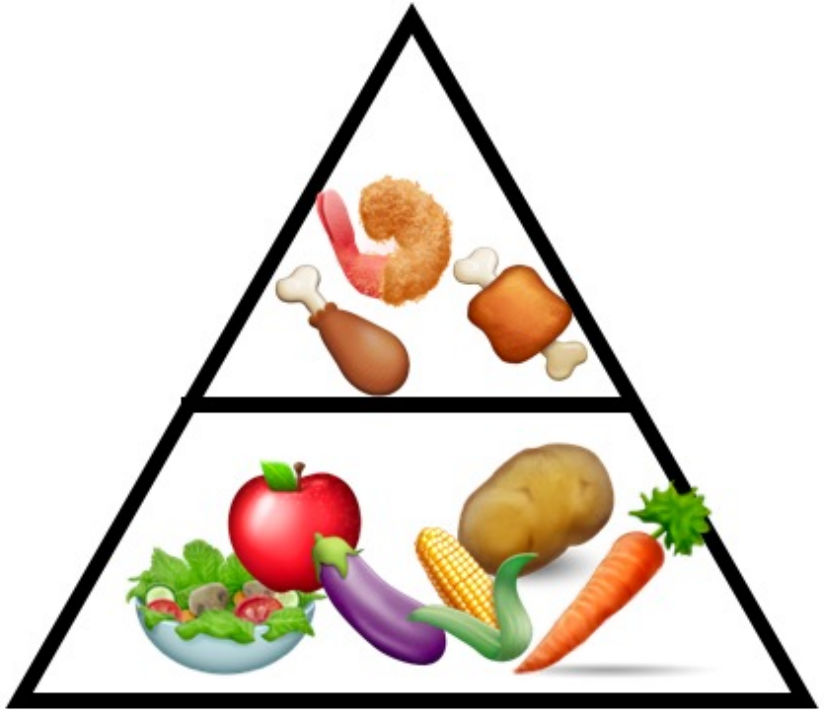
https://wiki.pathmind.com/eigenvector

BASIC PCA ALGORITHM

- Start with a zero-centered m x n data matrix **X**
- Compute covariance matrix
- Find eigenvectors of covariance matrix
- PCs: k eigenvectors with highest eigenvalues (variance)

EXAMPLE: FOOD NUTRITION

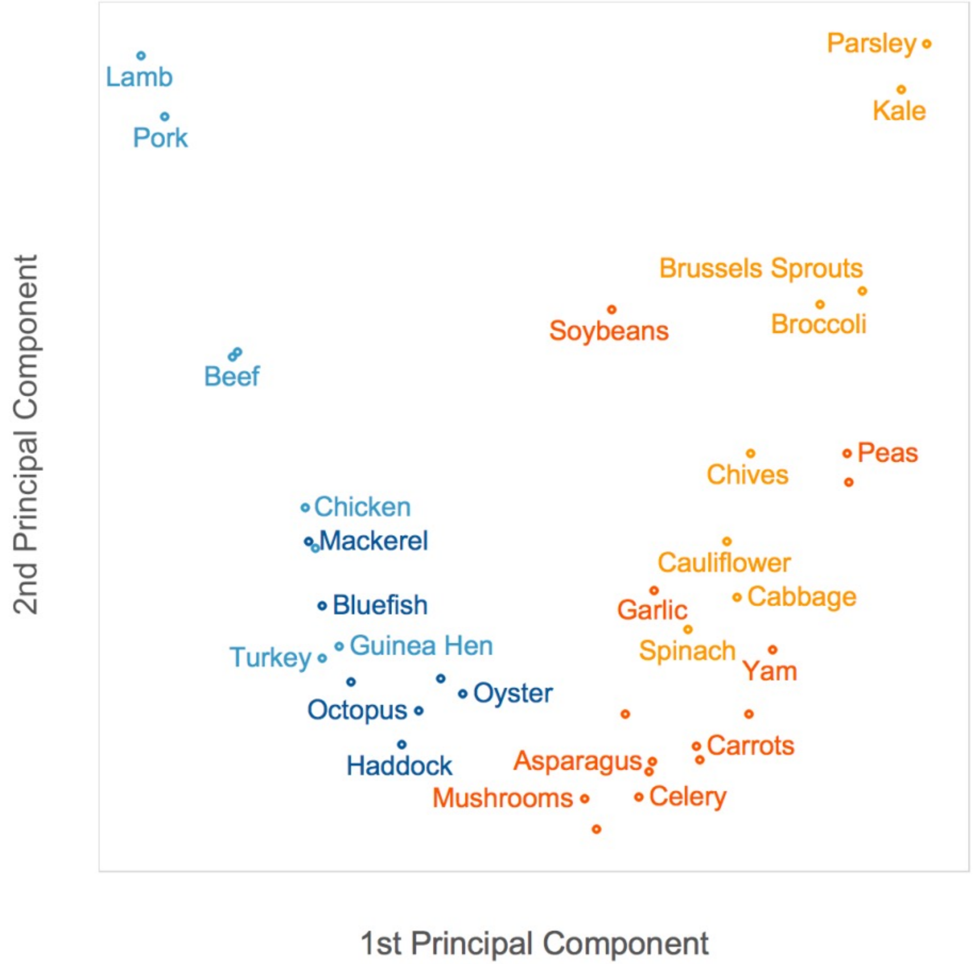
- What is the best way to differentiate food items?
  - Vitamin content
  - Protein levels
  - Fat
  - Fiber



https://algobeans.com/2016/06/15/principal-component-analysis-tutorial/

EXAMPLE: PCA

	PC1	PC2	PC3	PC4
Fat	-0.45	0.66	0.58	0.18
Protein	-0.55	0.21	-0.46	-0.67
Fiber	0.55	0.19	0.43	-0.69
Vitamin C	0.44	0.70	-0.52	0.22



https://algobeans.com/2016/06/15/principal-component-analysis-tutorial/

PCA: # OF PCS?

- How many components are sufficient to summarize the data?

PROPORTION VARIANCE EXPLAINED

- Total variance in data (assuming zero mean):

$$\mathrm{Var}(\mathbf{X}) = \sum_{j=1}^{p} \frac{1}{n} \sum_{i=1}^{n} x_{ij}^2$$

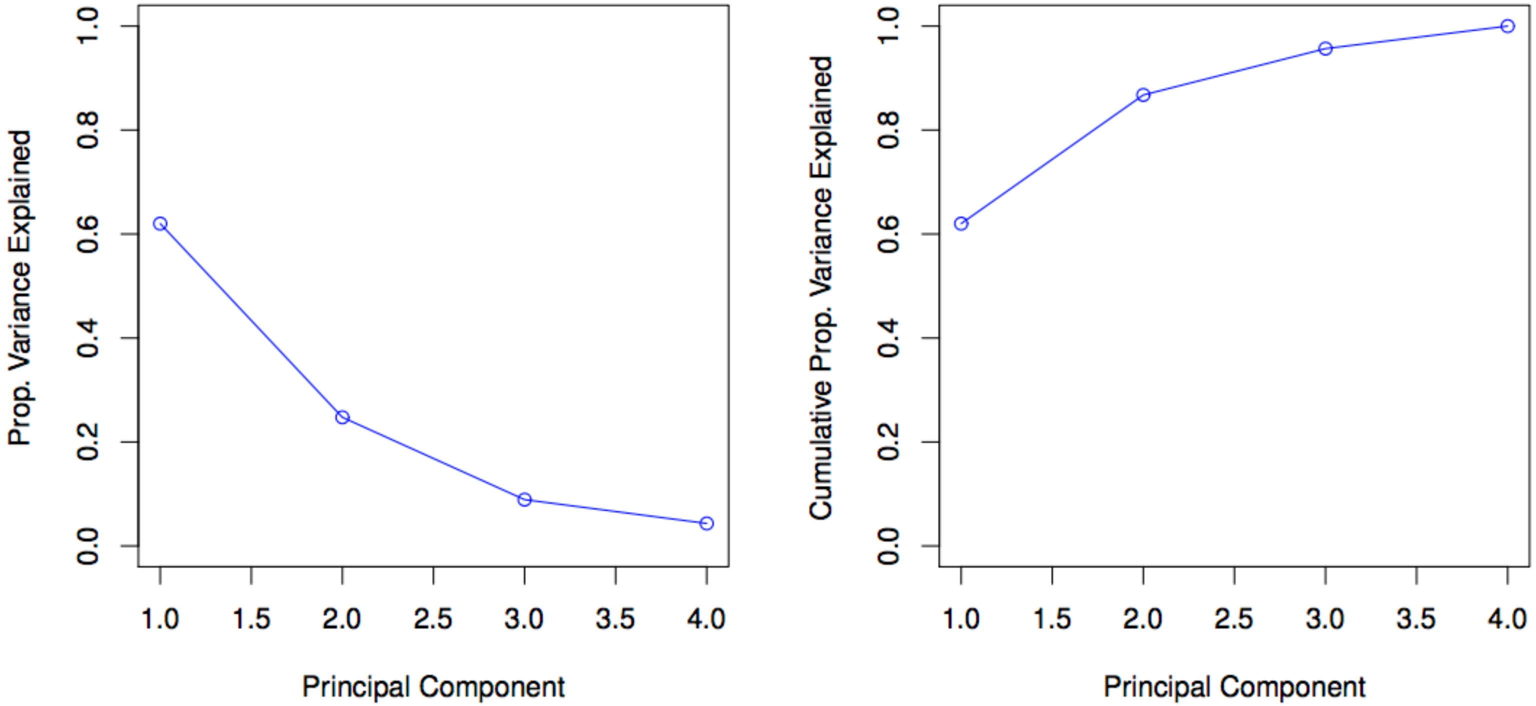
- Variance explained by the m^th component:

$$\mathrm{Var}(\mathbf{W}_m) = \frac{1}{n} \sum_{i=1}^{n} w_{im}^2$$

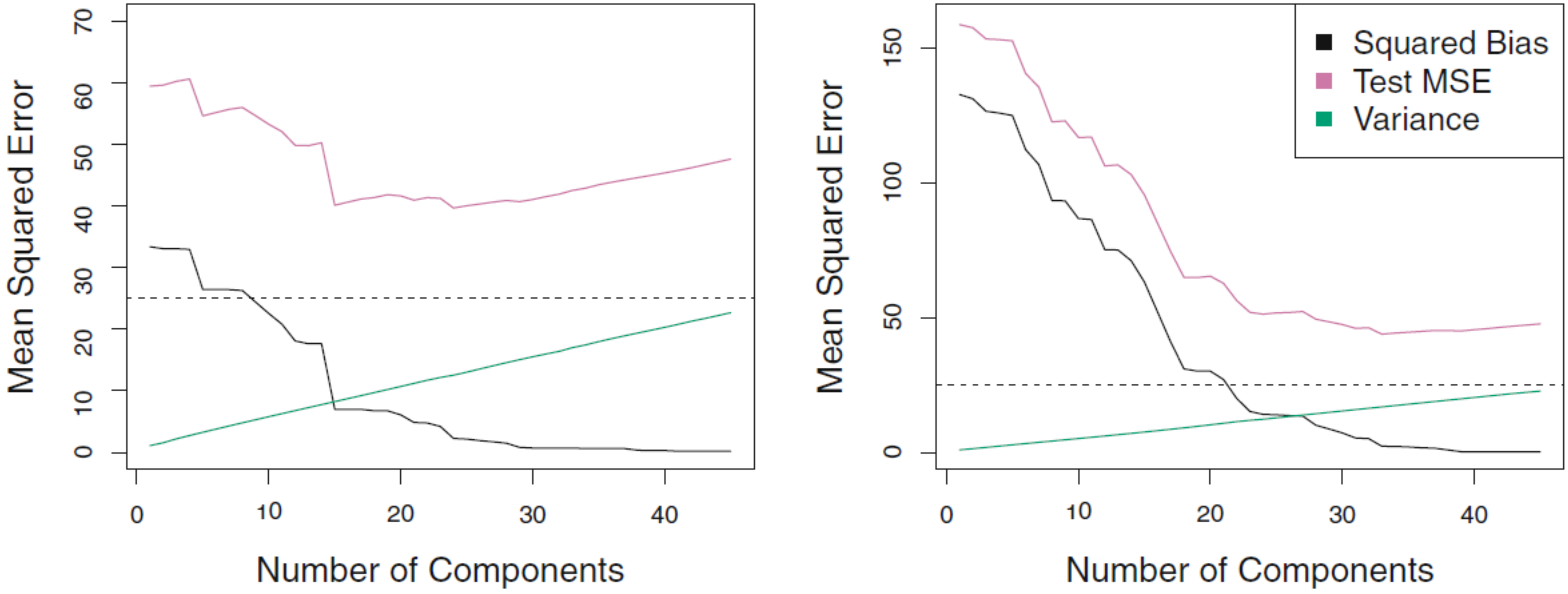
- Proportion of variance explained by m^th component :

$$\mathrm{PVE}_m = \frac{\mathrm{Var}(\mathbf{W}_m)}{\mathrm{Var}(\mathbf{X})}$$

PCA: SCREE PLOT (UNSUPERVISED)

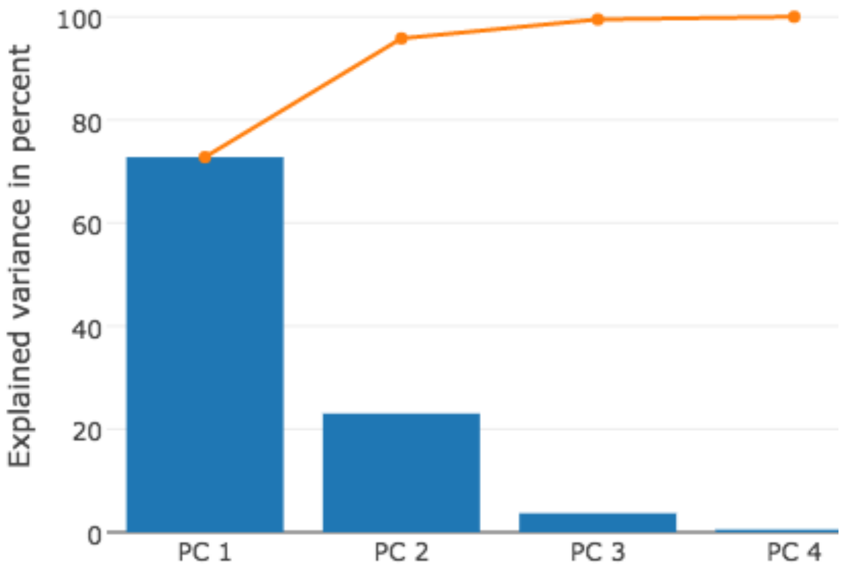


USE PCA FOR SUPERVISED LEARNING



PCA: INTERPRETATION

- If variances of PCs drop off quickly, then X is highly collinear
- Reduce dimensionality of data by keeping only the PCs with highest variance



https://plot.ly/ipython-notebooks/principal-component-analysis/

PCA: SKLEARN

```
from sklearn.decomposition import PCA as sklearnPCA
sklearn_pca = sklearnPCA(n_components=2)
Y_sklearn = sklearn_pca.fit_transform(X_std)
```

HOMEWORK #5

- Due 11/16 @ 11:59 PM ET on Gradescope
- 2 questions
  - PCA
  - Almost Random Forest

DEMO: PCA-EXAMPLE.IPYNB

HTTPS://COLAB.RESEARCH.GOOGLE.COM/DRIVE/12VDO-JD0PVFLVZZLI2FT50DYESJ8V6WW