Uses of PCA

Lyle Ungar

Learning objectives

PCA for feature creation

PCR, Semi-supervised learning

PCA for visualization

Eigenfaces: see the worksheet

Eigenwords: word embeddings

PCR: Principal Component Regression

1. Do a PCA on X to get scores Z and loadings V
2. Do OLS regression using Z as features

$$y = w \cdot z \qquad w = (Z^T Z)^{-1} Z^T Y$$

For future predictions, use $z = V^T x$

$$y = w \cdot V^T x$$

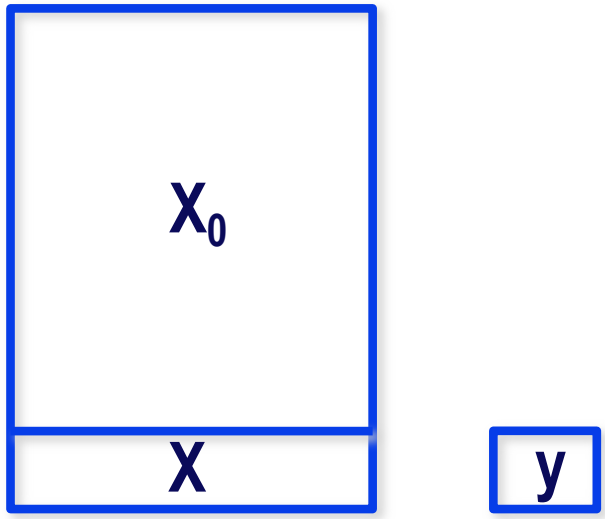
PCR

- How to find z for a new x?
  - $X = ZV^T$
- $x^T V = z^T V^T V = z^T$
- $z = V^T x$

V	$p \times k$
$V^T V = I$	$k \times k$
x^T	$1 \times p$
z^T	$1 \times k$

Semi-supervised learning

- Use unlabeled data X_0 to derive new features $z = \phi(x)$
- Train $y = f(\phi(x), w)$



Semi-supervised PCR

1a. Do a PCA on a big X_u to get loadings V

1b. Project X (with labels y) to get scores Z

2. Do OLS regression using Z as features

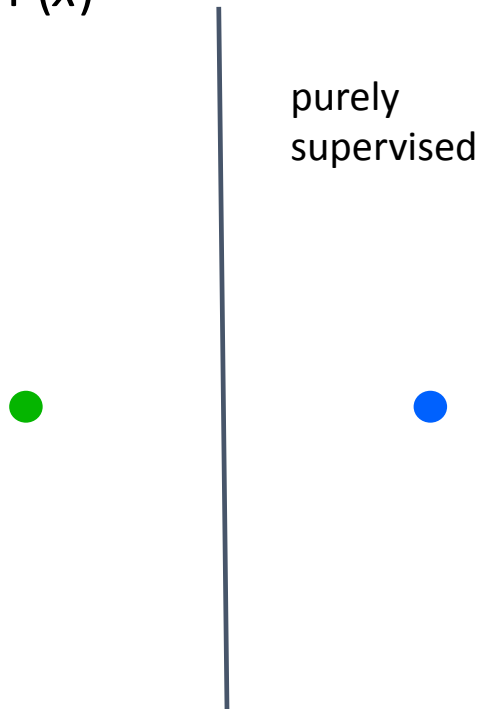
$$y = w \cdot z \qquad w = (Z^T Z)^{-1} Z^T Y$$

For future predictions, use $z = V^T x$

$$y = w \cdot V^T x$$

Semi-Supervised Learning

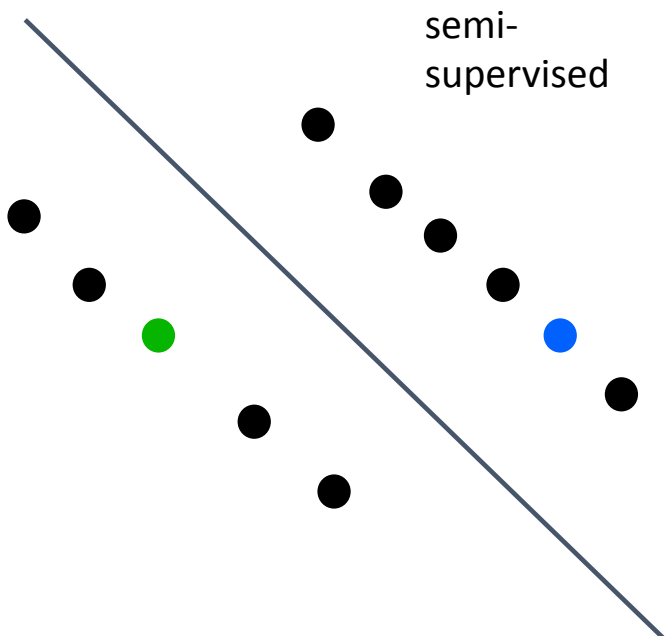
- Hypothesis: P($c|x$) can be more accurately computed using shared structure with P(x)



from Socher and Manning

Semi-Supervised Learning

- Hypothesis: P(*c*|*x*) can be more accurately computed using shared structure with P(*x*)



from Socher and Manning

PCA for visualization

- **Project original observations, *x*, into 2 dimensions**
  - PCA minimizes reconstruction error
  - This is one of many ways of trying to make points that were close in *m* dimensions still be close in 2 dimensions
- **Look at the loadings**
  - Often prefer sparse loadings

Country well-being

```
# import OECD data
df_OECD = pd.read_csv('http://www.cis.upenn.edu/~cis545/OECD_well-being.csv')
df_OECD
X, country = df_OECD.iloc[:,1:].values, df_OECD.iloc[:,0].values
```

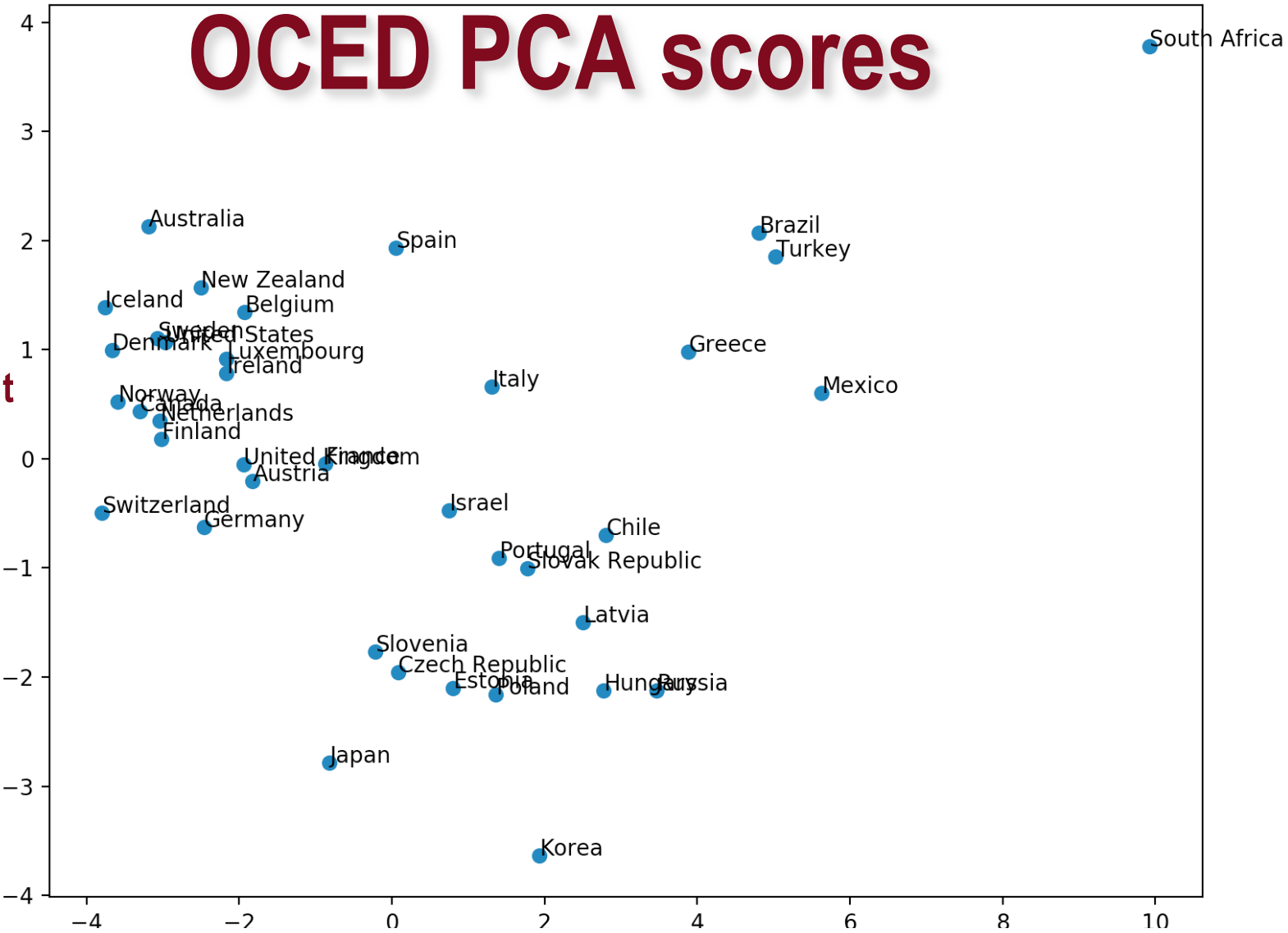
```
sc = StandardScaler()
X_std= sc.fit_transform(X)
```

```
def plot_points(X, labels):
        plt.scatter(X[:,0],X[:,1])
        for i, txt in enumerate(labels):
                plt.annotate(txt, (X[i,0],X[i,1]))
        plt.show()
```

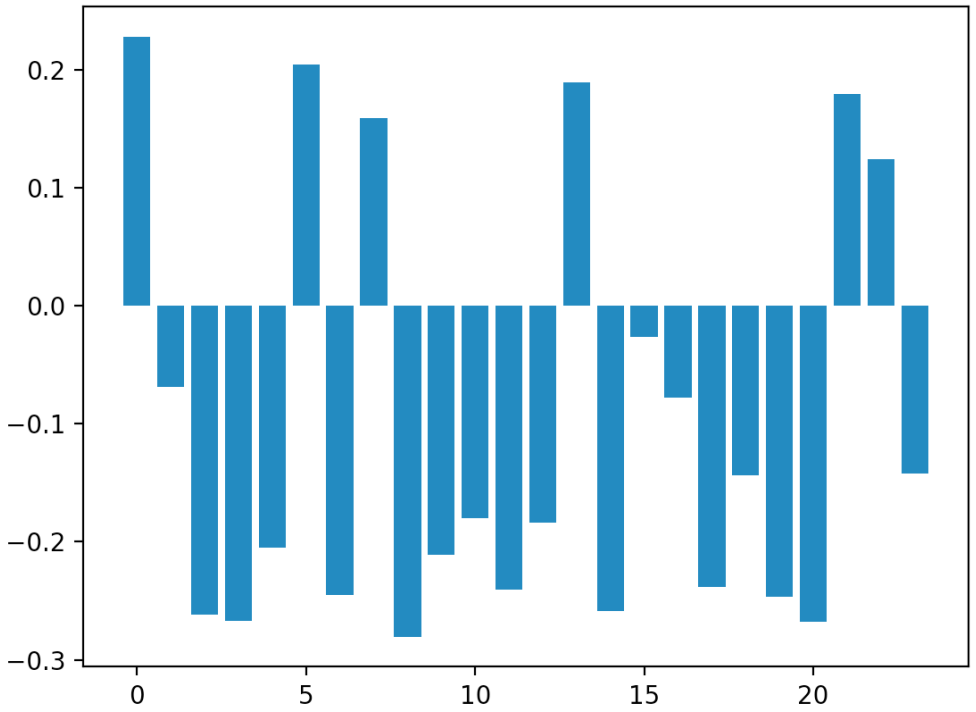
```
# plot PCA
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
pca.fit(X_std)
X_PCA = pca.transform(X_std)
plot_points(X_PCA,country)
```

OCED PCA scores

Second principal component

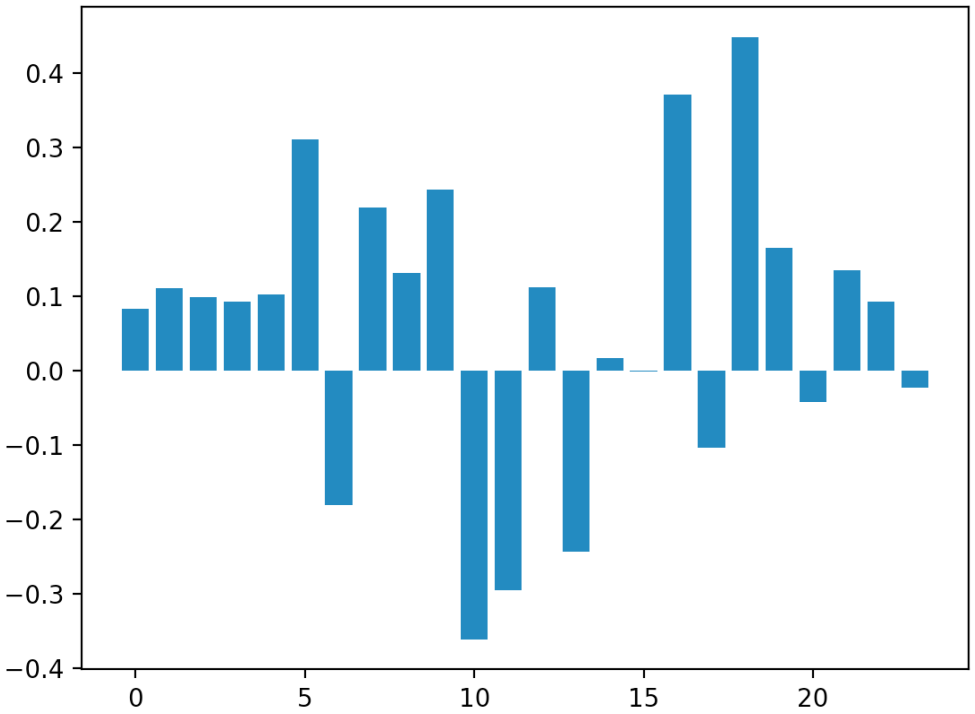


OCED PCA loadings



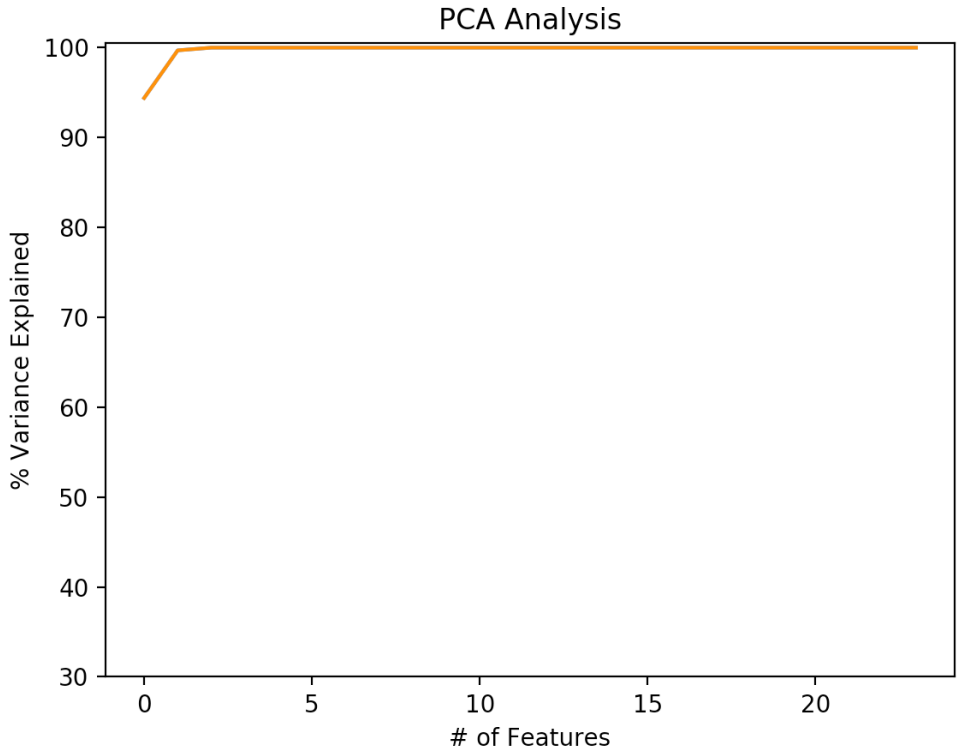
0 Dwellings without basic facilities
1 Housing expenditure
2 Rooms per person
3 Household net adjusted disposable income
4 Household net financial wealth
5 Labour market insecurity
6 Employment rate
7 Long-term unemployment rate
8 Personal earnings
9 Quality of support network
10 Educational attainment
11 Student skills
12 Years in education
13 Air pollution
14 Water quality
15 Stakeholder engagement for developing regulations
16 Voter turnout
17 Life expectancy
18 Self-reported health
19 Life satisfaction
20 Feeling safe walking alone at night
21 Homicide rate
22 Employees working very long hours
23 Time devoted to leisure and personal care

OCED PCA loadings



0 Dwellings without basic facilities
1 Housing expenditure
2 Rooms per person
3 Household net adjusted disposable income
4 Household net financial wealth
5 Labour market insecurity
6 Employment rate
7 Long-term unemployment rate
8 Personal earnings
9 Quality of support network
10 Educational attainment
11 Student skills
12 Years in education
13 Air pollution
14 Water quality
15 Stakeholder engagement for developing regulations
16 Voter turnout
17 Life expectancy
18 Self-reported health
19 Life satisfaction
20 Feeling safe walking alone at night
21 Homicide rate
22 Employees working very long hours
23 Time devoted to leisure and personal care

All the variance is in the first PC



Eigenwords

Learning objectives
Distributional similarity
Word embeddings
SVD on words

Lyle Ungar
University of Pennsylvania

Represent each word by its context

I ate ham
You ate cheese
You ate

context

word	Word Before: ate	cheese	ham	I	You	Word After: ate	cheese	ham	I	You
ate	0	0	0	1	2	0	1	1	0	0
cheese	1	0	0	0	0	0	0	0	0	0
ham	1	0	0	0	0	0	0	0	0	0
I	0	0	0	0	0	1	0	0	0	0
You	0	0	0	0	0	2	0	0	0	0

Distributional Similarity

Hypothesis: Words with similar contexts have similar meanings

Eigenwords: *Word embeddings*

- Project high dimensional context to low dimensional space (SVD/PCA)
- Similar words are close in this low dimensional space

I ate ham
You ate cheese
You ate

	Word Before					Word After				
	ate	cheese	ham	I	You	ate	cheese	ham	I	You
ate	0	0	0	1	2	0	1	1	0	0
cheese	1	0	0	0	0	0	0	0	0	0
ham	1	0	0	0	0	0	0	0	0	0
I	0	0	0	0	0	1	0	0	0	0
You	0	0	0	0	0	2	0	0	0	0

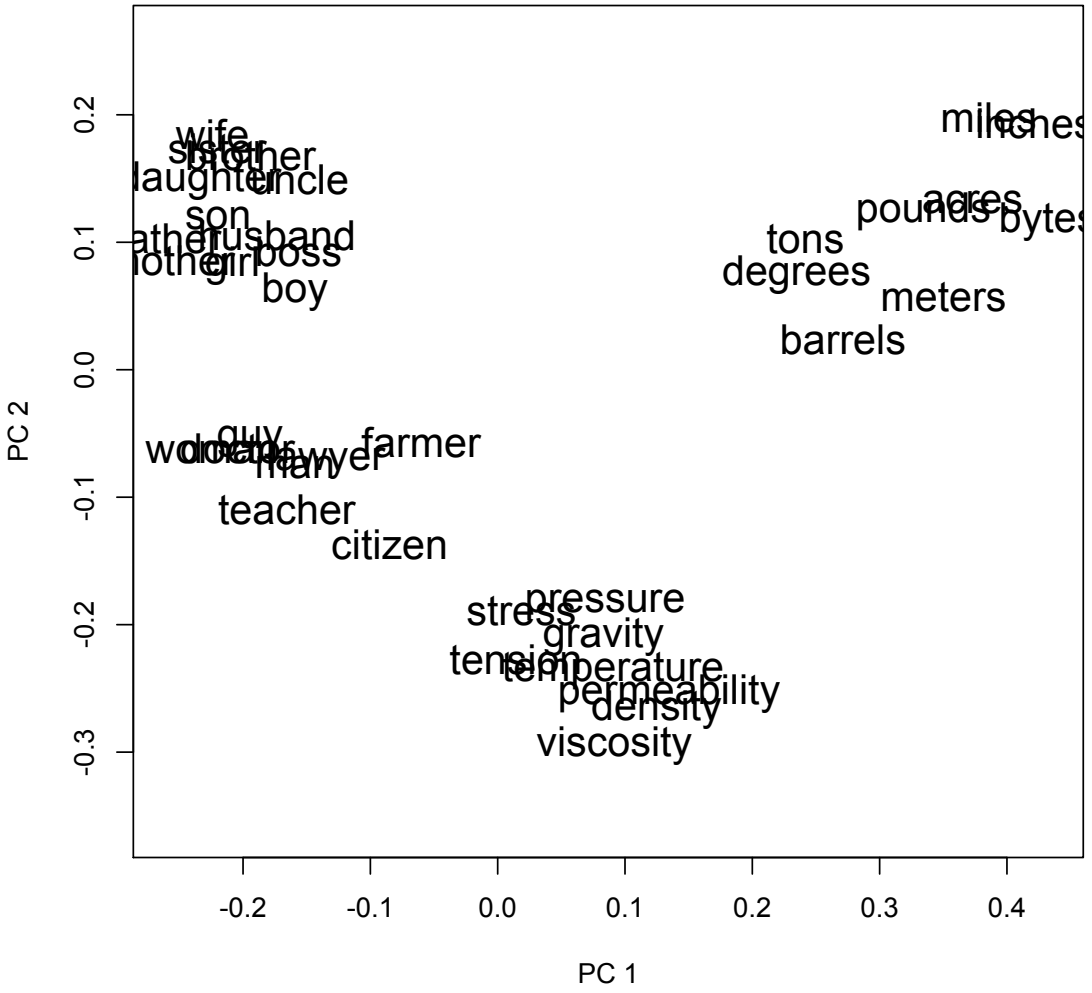
Eigenwords as SVD

- **Left singular vectors are *eigenwords***
  - a vector representing each word – ***"word embeddings"***
- **Right singular vectors times context give *eigentokens***
  - vectors mapping contexts to the latent space

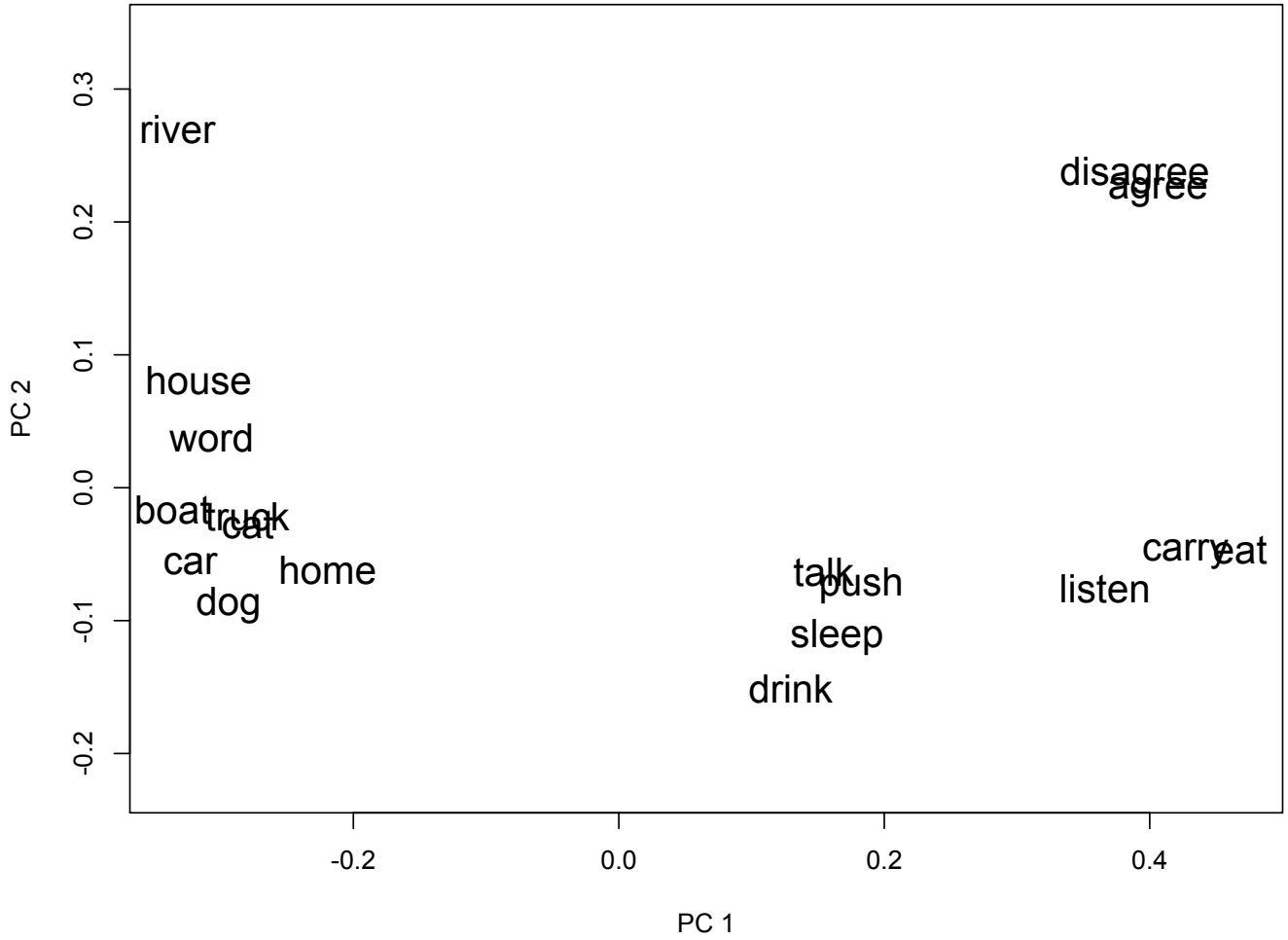
I ate ham
You ate cheese
You ate

	Word Before					**Word After**				
	ate	cheese	ham	I	You	ate	cheese	ham	I	You
ate	0	0	0	1	2	0	1	1	0	0
cheese	1	0	0	0	0	0	0	0	0	0
ham	1	0	0	0	0	0	0	0	0	0
I	0	0	0	0	0	1	0	0	0	0
You	0	0	0	0	0	2	0	0	0	0

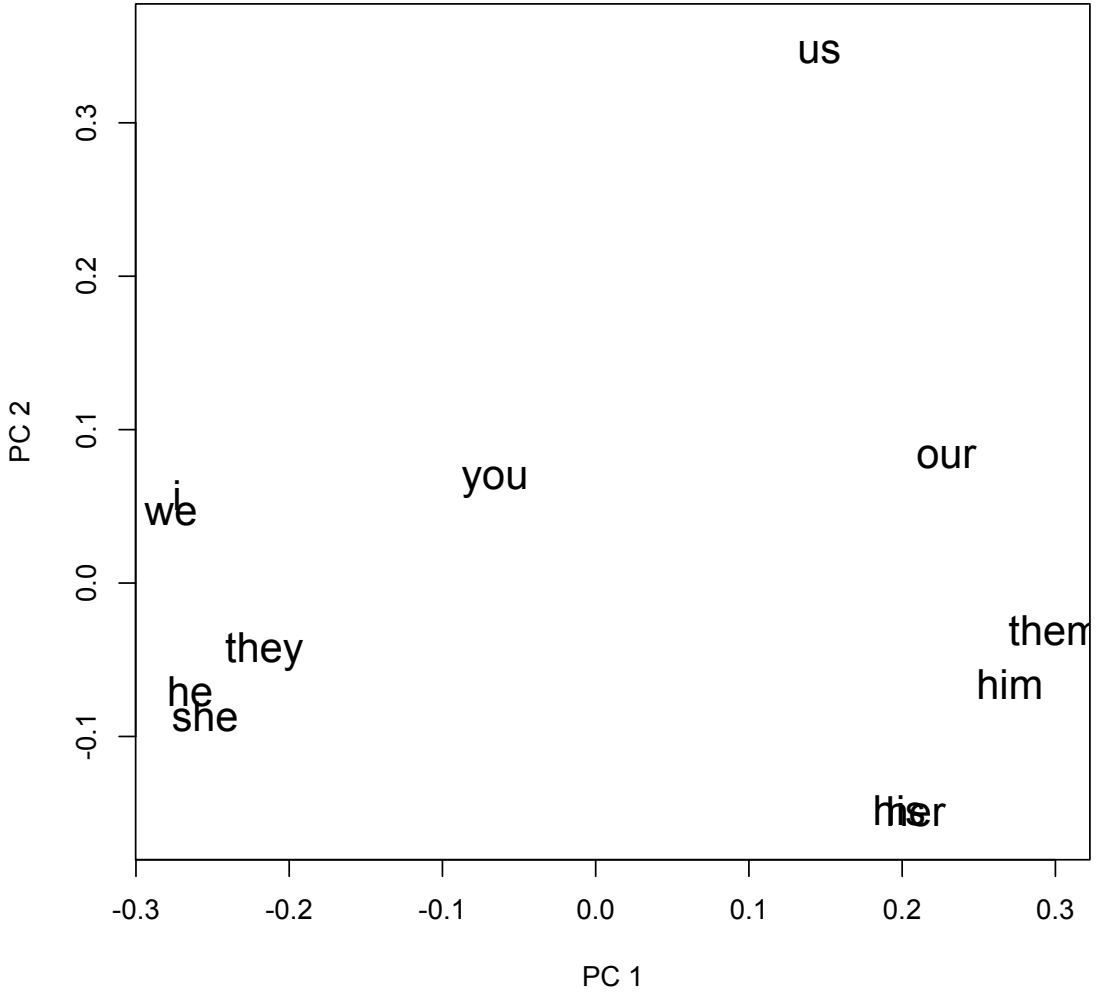
Similar words are close



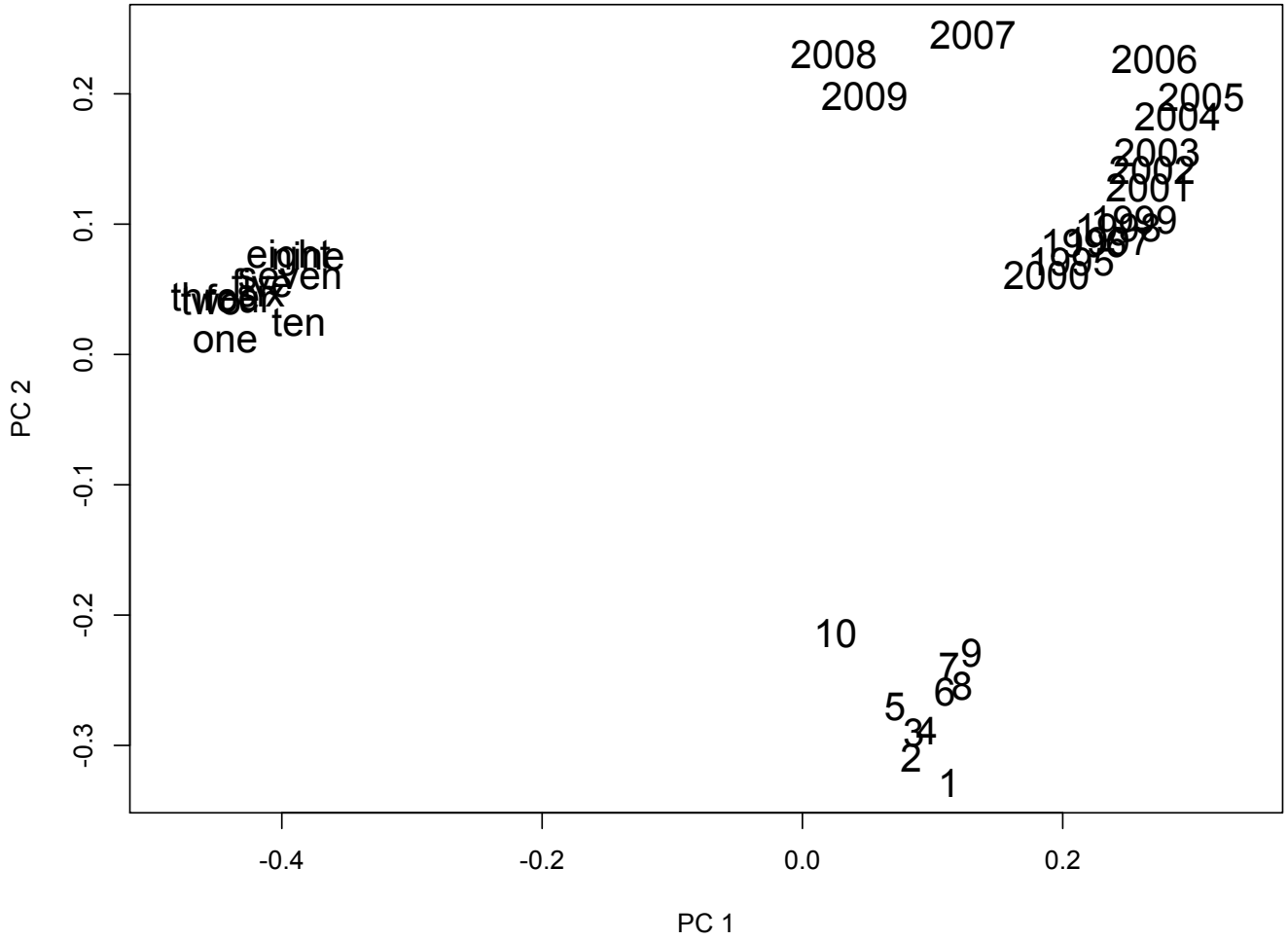
Nouns and verbs



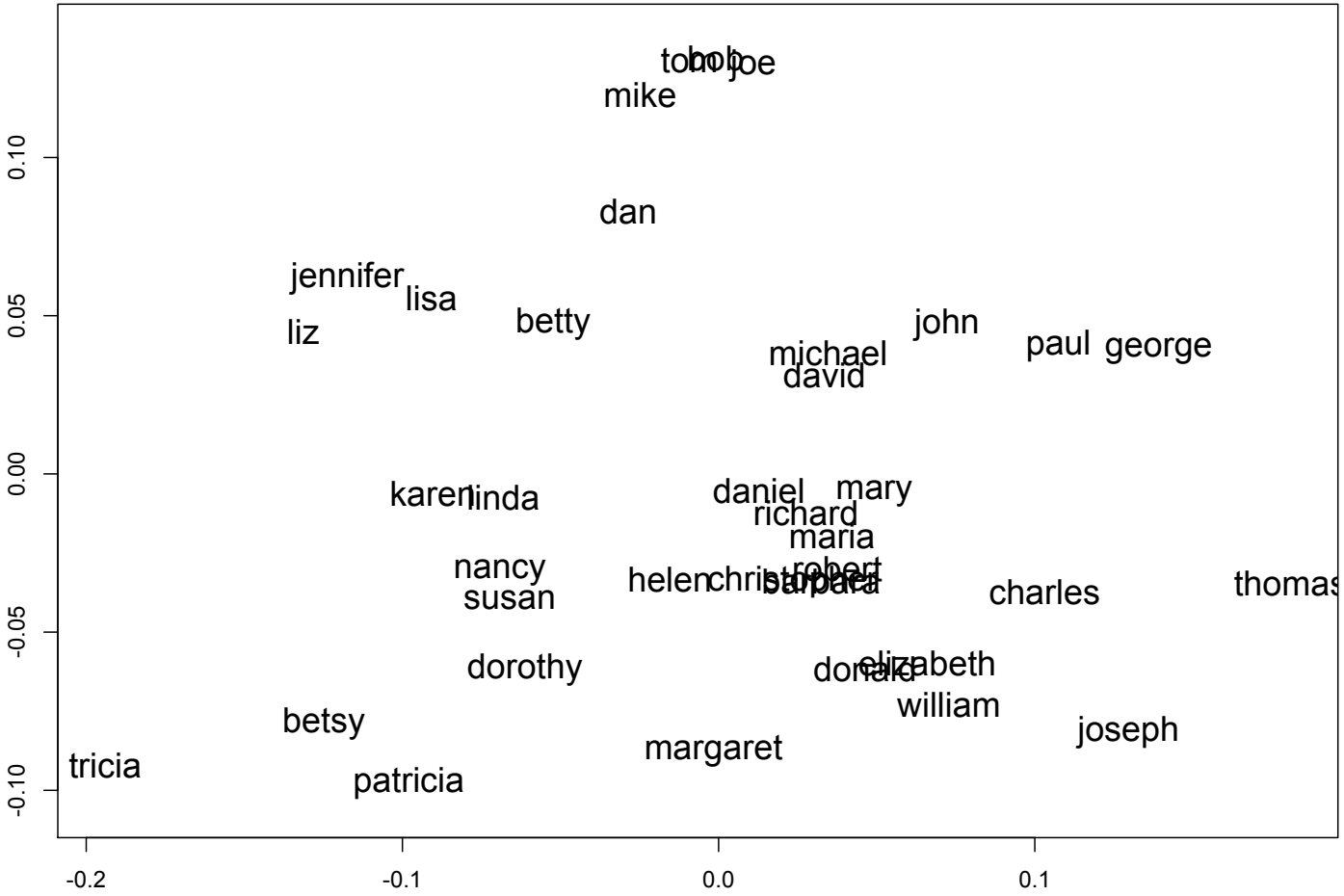
Pronouns


us
you
our
i
we
they
he
she
them
him
his
her
PC 2
PC 1
0.3
0.2
0.1
0.0
-0.1
-0.3
-0.2
-0.1
0.0
0.1
0.2
0.3

Numbers


2008
2007
2009
2006
2005
2004
2003
2002
2001
2000
eight
seven
two
one
ten
10
9
7
8
6
5
4
3
2
1
PC 2
PC 1
0.2
0.1
0.0
-0.1
-0.2
-0.3
-0.4
-0.2
0.0
0.2

Names


tom
bob
joe
mike
dan
jennifer
lisa
liz
betty
john
paul
george
michael
david
karen
linda
daniel
mary
richard
maria
nancy
susan
helen
barbara
charles
thomas
dorothy
donald
elizabeth
william
betsy
margaret
joseph
tricia
patricia
0.10
0.05
0.00
-0.05
-0.10
-0.2
-0.1
0.0
0.1

Word Sense Disambiguation

- **Estimate "state vector" ("contextualized embedding") for a word using right singular vectors**
- **Similar meanings will again be close.**
  - The ships dock in the port.
  - The port is loaded onto ships and sent to America
  - The meat is tender.
  - I have tender feelings for her.
  - The company will tender an offer.

Use eigenwords/eigentokens in supervised learning

- **'Similar' words have embeddings that are close**
- **Predict labels for tokens based on their estimated "state vector"**
  - Part of speech
  - Named entity type (person, place, thing…)
  - Word sense ("meaning") disambiguation
- **Or embed sentences**

Word2vec

- ***Word embeddings*, often found by deep learning, are very popular now**
  - Word2Vec has similar performance to the simpler eigenwords
- **Deep learning (contextualized) versions such as BERT and friends work better**
  - To be covered later

What you should know

- **PCR**
  - Use principal components from training to find scores on test data
- **Interpretation: Scores and loadings**
  - percentage of variance explained
- **Word embeddings**
  - Context free (left singular vectors: words)
  - Context sensitive (right singular vectors: context)