Homework

- Read chapter 3 in *Introduction to Machine Learning with Python*
- Classification assignment due Friday 3/23

Classification

- Supervised learning: we know what the categories are, but we want the program to learn how to assign them
- Unsupervised learning: we want the program to discover the categories
  - Find new representations for texts beyond bag-o-words
  - Discover unknown categories among texts

Feature extraction

- Corpus representation (so far) is the sparse $D \times V$ **document-term matrix *M***

	rude	awful	friendly	service
d_1	2	1	0	4
d_2	3	0	0	1
d_3	0	1	2	0
d_4	1	0	0	1

- **Feature selection** (e.g.: min_df, max_df) chooses terms to include as columns
- **Feature transformation** (e.g.: tf-idf) scales the values to give more weight to some terms or documents

Feature extraction

- Feature extraction creates new features
- Character n-grams

  This shows a very ripe nose with chalky notes

  {*This, a, chalky, nose, notes, ripe, shows, very, with*}

  {*␣cha, ␣sho, ␣ver, This, lky␣, note, ose␣, pe␣n, s␣␣␣, with, ws␣a, y␣ri*}

Feature extraction

- Automated Term Recognition (ATR) used to find technical terms for indexing, machine translation, etc
- Simpler, cheaper methods based on PMI work well for classification
- Mikolov et al. (2013):

$$s(w_i, w_j) = \frac{\mathrm{count}(w_i w_j) - \delta}{\mathrm{count}(w_i) \times \mathrm{count}(w_j)}$$

- Merge bigrams with $s >$ threshold
- Repeat to find 3, . . . word terms

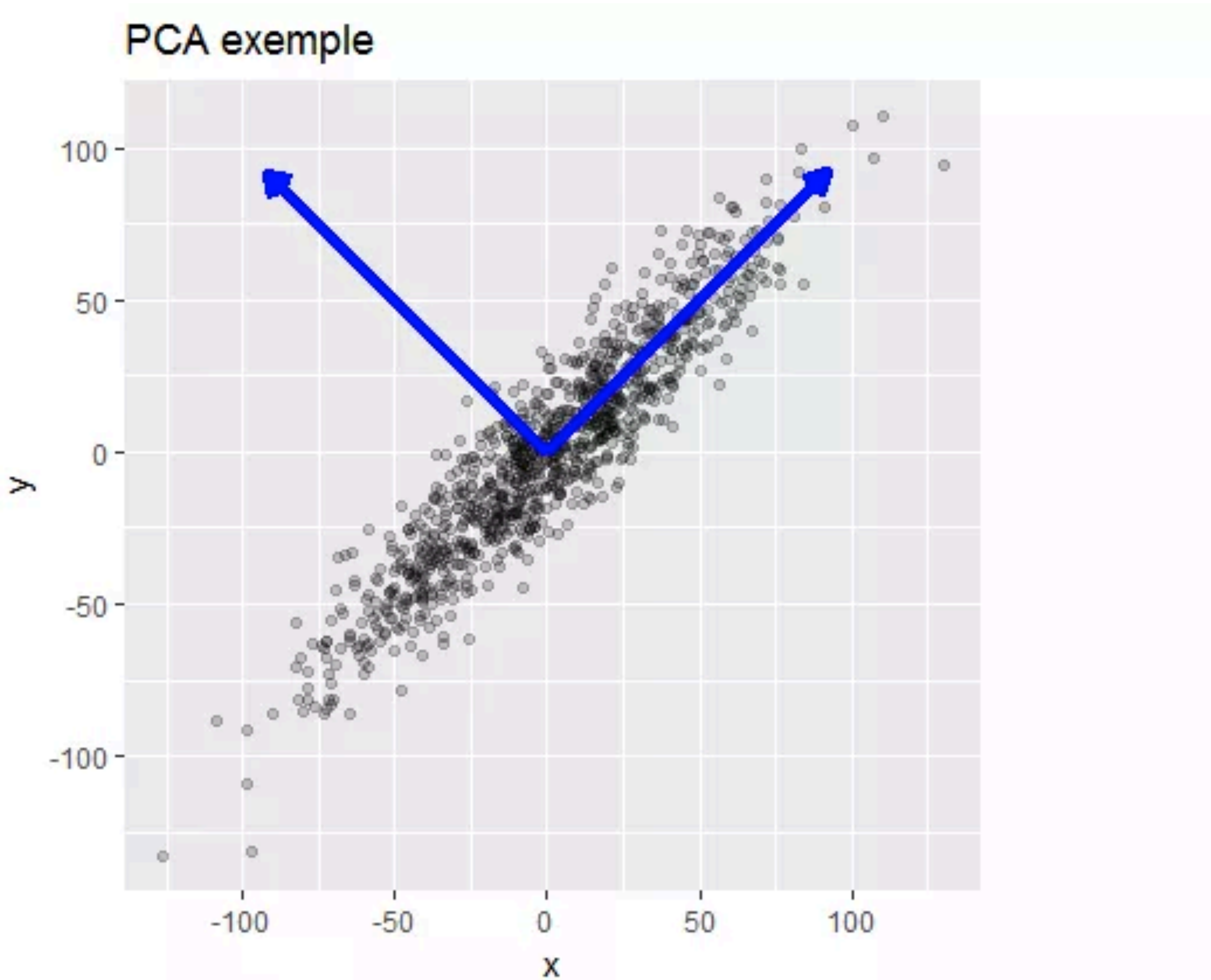
Feature extraction

Consumed_at Maitre d ' with stuffed chilled lobster . **Bright_disc** . **Light_yellow** robe with **clear_rim** . **Clean_nose** , showing restrained aromas of minerals and pear . Light to **medium_-_bodied** on the palate , with crisp acidity , light oak and **similar_flavors** as for the nose . The **mid_-_palate** drops_off and the finish is short . Hopefully some time in the bottle **will_help**

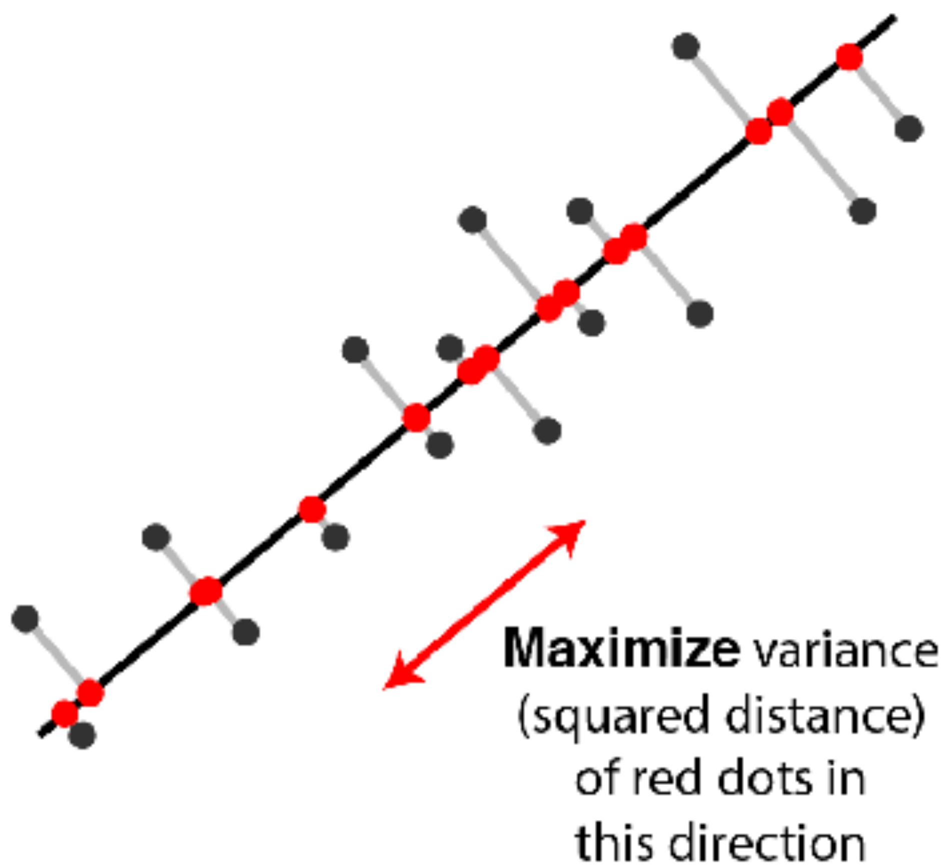
Latent Semantic Indexing

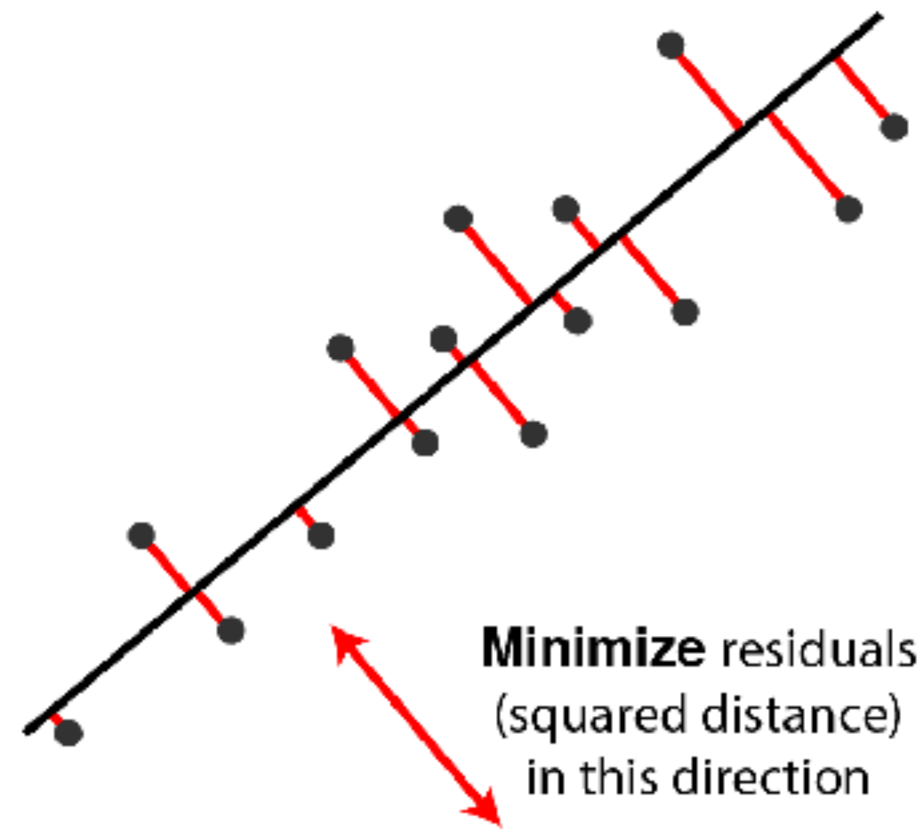
- Dimensionality reduction techniques transform M
- **Latent Semantic Indexing** uses tf-idf + Principal Component Analysis to project M into a lower dimensional space
- Developed in the 1980's for information retrieval

Latent Semantic Indexing



Latent Semantic Indexing





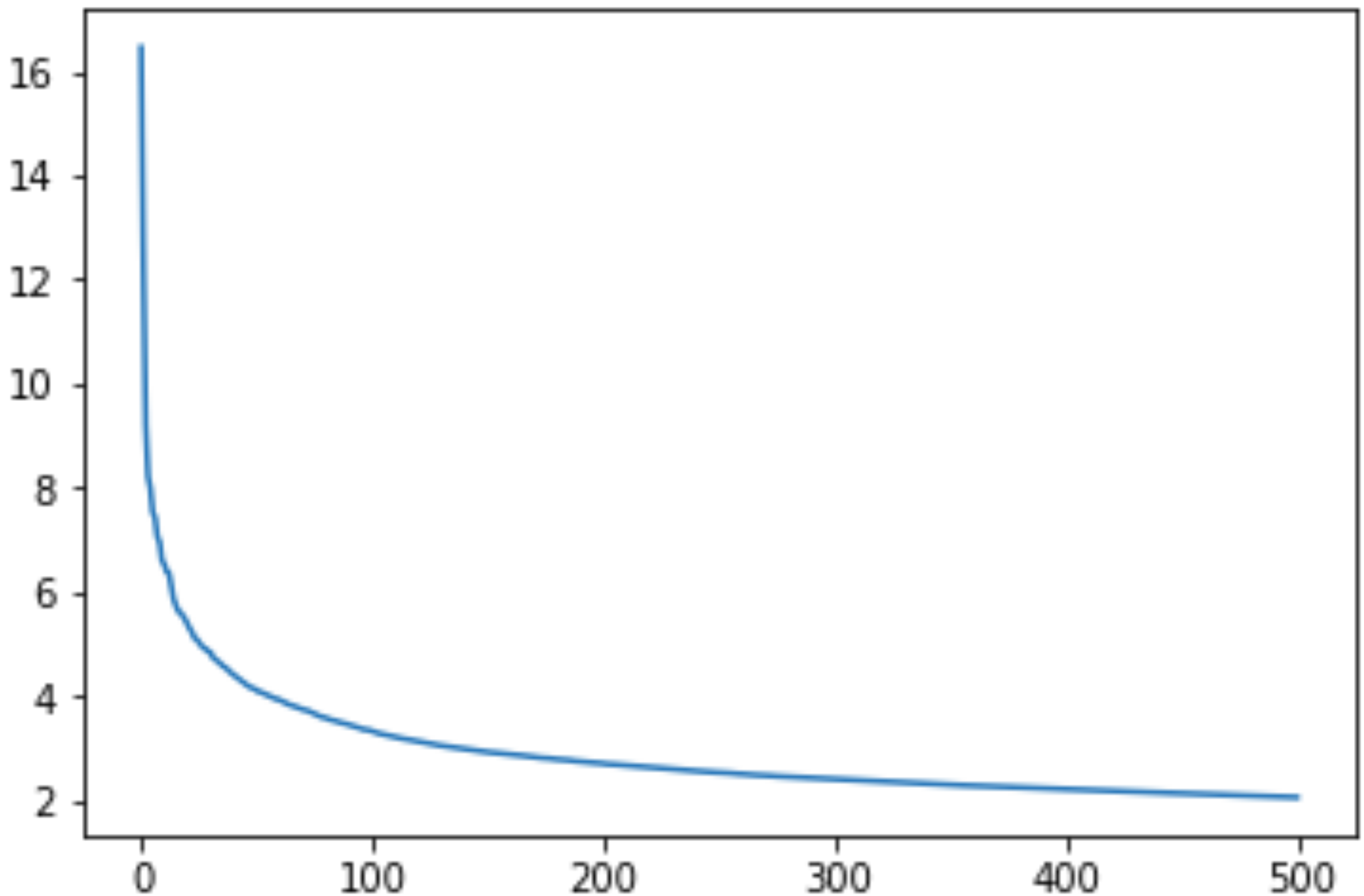
Latent Semantic Indexing

- Apply to Reuters politics dataset

```
model = make_pipeline(CountVectorizer(analyzer=identity),
                      TfidfTransformer(norm='l2', use_idf=True),
                      TruncatedSVD(300, n_iter=25),
                      LogisticRegression())
```

- Improves accuracy from 89.05% to 89.61%

Latent Semantic Indexing



Latent Semantic Indexing

percent U.S. $ 1 new million Clinton state Minister 2

0 1 2 6 3 4 7 5 : 8

Israel Israeli Palestinian Netanyahu peace Arafat Palestinians Jerusalem Hebron East

Kong Hong China Chinese Taiwan Beijing Zaire rebels military refugees

Kong Hong China percent Chinese Israel Beijing tax Taiwan Palestinian

NATO Yeltsin Russia 0 Russian Moscow 1 alliance summit Clinton

6 7 beat 4 NATO Yeltsin Russia 5 U.S. Russian

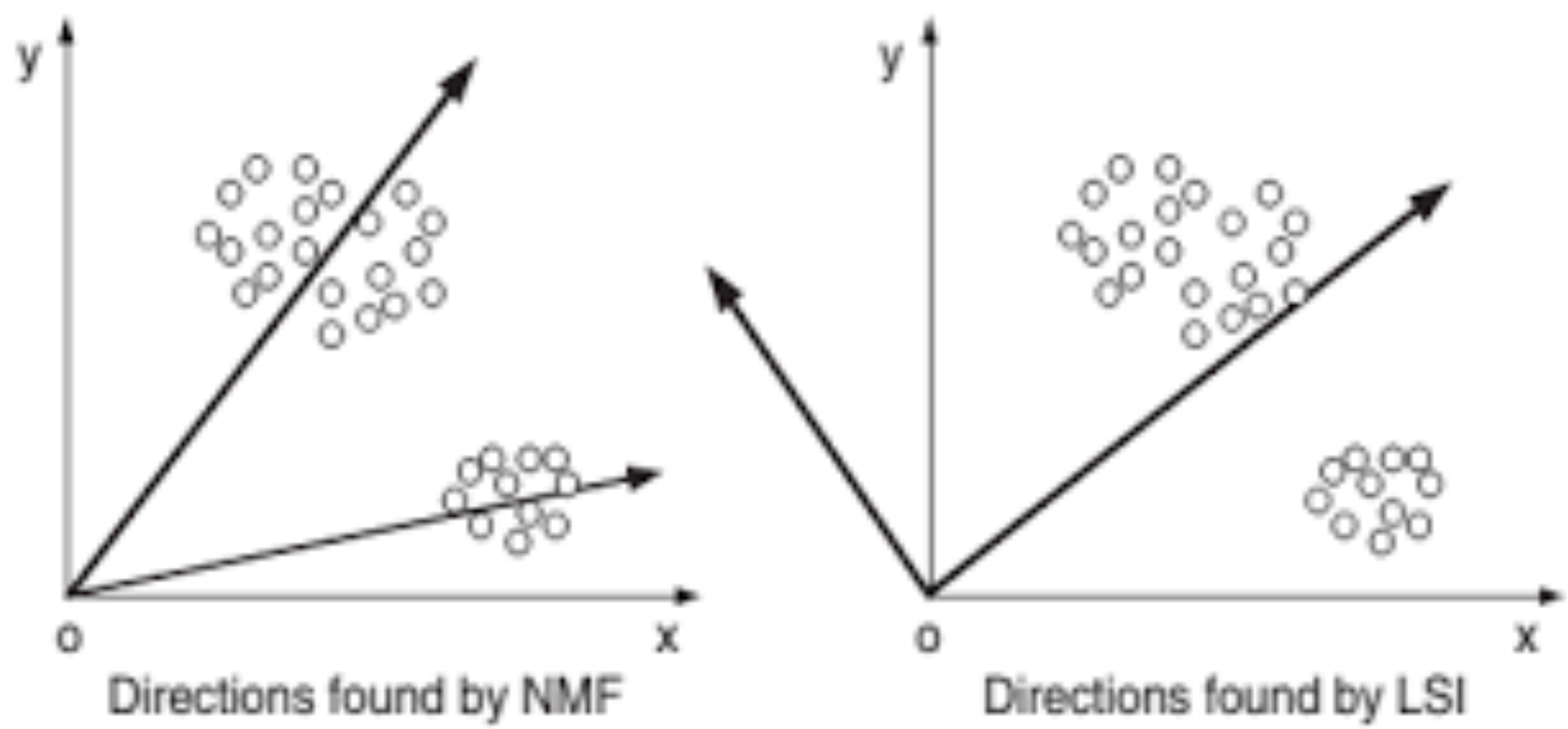
Zaire percent refugees 6 tax Iraq Mobutu Kabila budget rebels

party election Labour percent Zaire opposition elections Party parliament Mobutu

Zaire Mobutu Kabila refugees Rwanda South Rwandan Africa Zairean Clinton

Feature extraction

- Other dimensionality reduction and manifold learning techniques can also be used
- Non-negative Matrix Factorization find non-negative but not necessarily orthogonal dimensions



NMF

Labour party election Party opposition vote parliament Major elections coalition

0 1 2 3 4 5 Results matches played soccer

Israel Palestinian Israeli Netanyahu Arafat peace Palestinians Jerusalem Hebron East

China Chinese Beijing Deng rights Jiang human Xiaoping trade visit

Kong Hong China Tung British handover Chinese territory colony legislature

NATO Russia alliance expansion Moscow enlargement Poland Europe summit Russian

6 beat 7 4 3 2 5 tennis denotes Spain

Cup club season league team match World England striker players

$ million billion money pounds bonds state pay oil fund

refugees Zaire Rwanda Rwandan Hutu Zairean rebels eastern U.N. Tutsi

Distributional semantics

- M represents documents via words, and M^T represents words via documents
- Words that occur in similar sets of documents probably have (broadly) related meanings
- Distributional hypothesis
  - The meaning of a word is the sum of all the contexts where it can be used
  - JR Firth (1957): "We shall know a word by the company it keeps"

Distributional semantics

- Latent Semantic Analysis uses PCA on a **term-term matrix**
- N is the number of times word c_3 occurs within a k-word context window of t_3

	c_1	c_2	c_3	c_4
t_1				
t_2				
t_3			N	
t_4				

- Term vectors are distributional representations of word meanings

Distributional semantics

- Early versions used PCA, NMF, etc.
- Current “word embedding” systems (word2vec, GloVe, fastText) use deep learning techniques
  - CBOW (Continuous Bag Of Words) uses a non-uniform context window
  - SGNS (Skipgrams+Negative Sampling) learn to predict a missing word

    Intense aromas of _____ fruits and tobacco . → *dark*

Clustering

- For classification, the categories are defined by the task and known in adavance
- Not always the case!
- Clustering algorithms are unsupervised methods for grouping similar texts into categories
- Depends on notion of ‘similar’

Distance metrics

- A distance metric $d(x, y)$ must satisfy:
  - non-negative: $d(x, y) \geq 0$
  - $d(x, y) = 0$ iff x and y are the same
  - symmetric: $d(x, y) = d(y, x)$
  - triangle inequality: $d(x, z) \leq d(x, y) + d(y, z)$
- Jaccard distance

$$J(X, Y) = 1 - \frac{|X \cap Y|}{|X \cup Y|}$$

Distance metrics

- Documents as term vectors
- Inner product or 'dot product' of two vectors is defined as

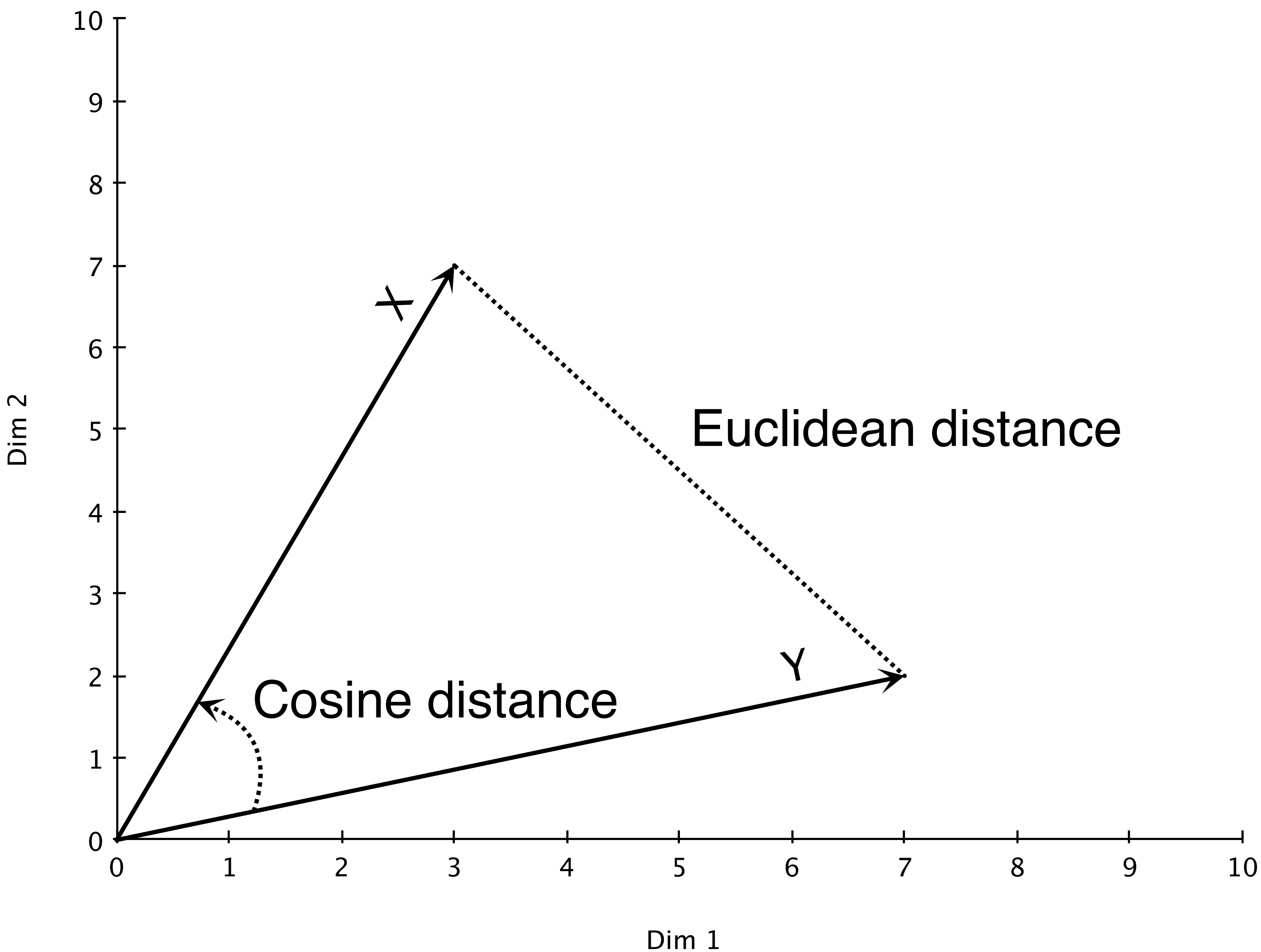
$$X \cdot Y = \sum_{i=1}^{m} x_i y_i$$

- Jaccard distance (for binary term vectors):

$$d(X, Y) = 1 - \frac{X \cdot Y}{X^2 + Y^2 - X \cdot Y}$$

Distance metrics

- Documents as term vectors



Distance metrics

- Euclidean distance

$$d(X,Y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

- Cosine distance

$$\begin{aligned} d(X,Y) &= 1 - \frac{X \cdot Y}{\|X\| \, \|Y\|} \\ &= 1 - \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2} \sqrt{\sum_{i=1}^{n} y_i^2}} \end{aligned}$$

Distance metrics

- Euclidean distance
  - ranges from 0 to ∞
  - depends on absolute term frequencies and the total number of words in document
- Cosine distance
  - ranges from 0 to 1
  - only depends on relative word frequencies
- If document vectors are pre-normalized to length 1, then cosine distance is just the dot product $X \cdot Y$
- Distance and similarity are related, but not the same!

Assignment

- Your task: write a program that can guess from a review what kind of wine is being talked about (*Cabernet Sauvignon, Merlot, Chardonnay, Sauvignon Blanc*)
- See 09-wine-project on github
- Subtask 1:
  - Read texts
  - Tokenize texts using spaCy
- Subtask 2:
  - Build baseline classifier using DummyClassifier
  - Evaluate using 10-fold cross-validation

Assignment

- Subtask 3:
  - Build a logistic regression classifier using LogisticRegression
  - Evaluate using 10-fold cross-validation over the same training/validation splits as you used for the baseline
- Subtask 4:
  - Build the best classifier you can using any method
  - Use GridSearchCV to find optimal settings for hyperparameters
  - Again, evaluate using 10-fold cross-validation over the same training/validation splits as you used for the baseline

Assignment

- Subtask 4:
  - Error analysis
    - What kinds of reviews is your classifier bad at classifying, and why?
  - Discussion
    - What have you learned about the task?
    - Is guessing the wine variety from a review hard or easy? What are the hard parts?
    - What would you need to do to score better than 90% accuracy?
- Turn in notebook via github by next Friday 3/23