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

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

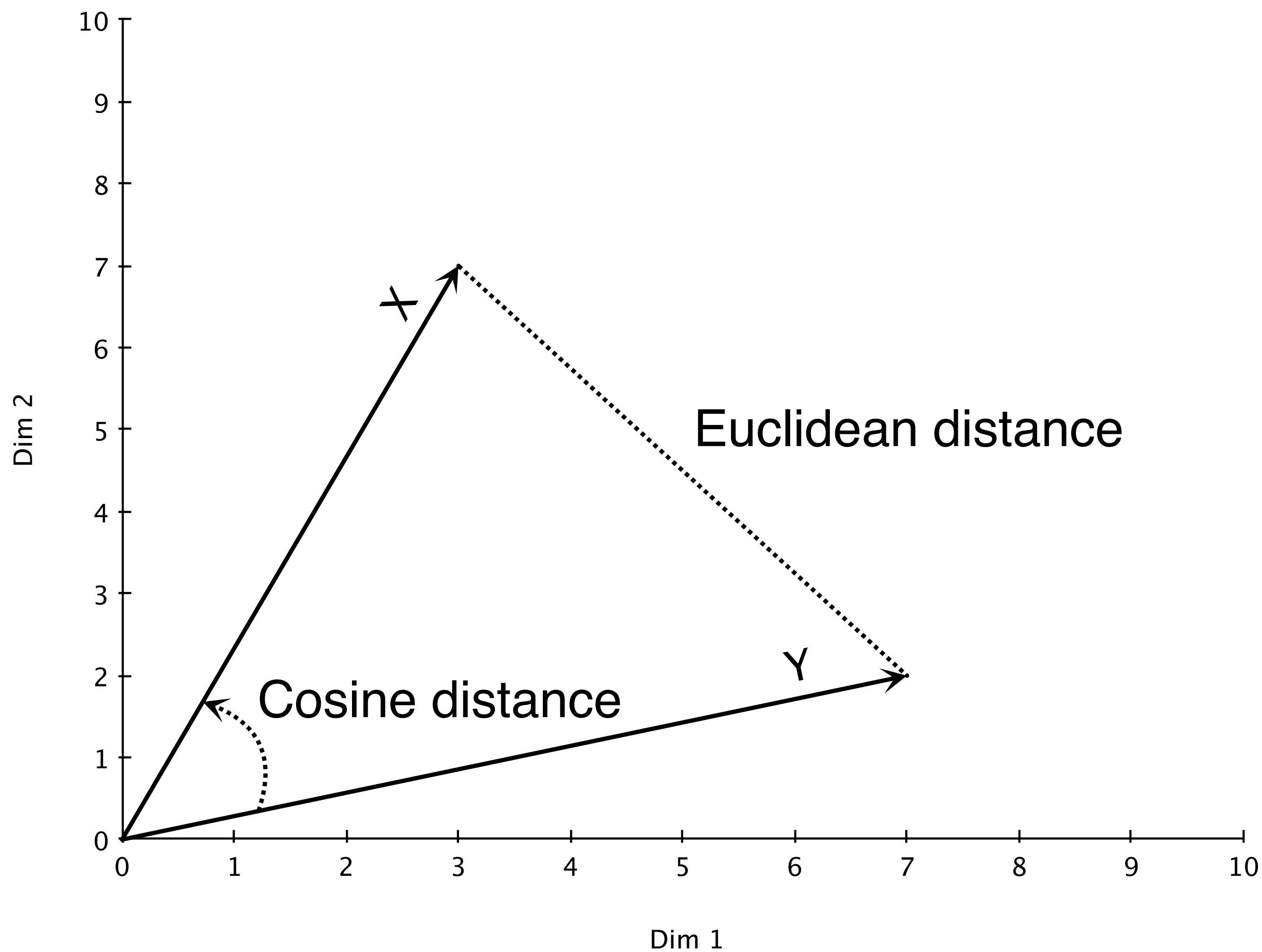
$$X \cdot Y = \sum_{i=1}^{m} x_i y_i$$

- 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!

Clustering

- Classification is **supervised** = useful for when you want to automate a known task
- Clustering is **unsupervised** = useful for when you don't know what you want to do
  - Forensic document analysis
  - Intelligence
  - Social media

Clustering

- Silhouette Coefficient
  - a = the mean distance between a sample and all other points in the same class.
  - b = The mean distance between a sample and all other points in the next nearest cluster.
  - The Silhouette Coefficient s for a single sample is then given as:

$$s = \frac{b - a}{max(a, b)}$$

  - The Silhouette Coefficient for a set of samples is given as the mean of the Silhouette Coefficient for each sample

K Means

- K Means clustering organizes items into *k* clusters represented by centroids
- Each item is in the cluster with the closest centroid
- Start with randomly distributed centroids
- http://stanford.edu/class/ee103/visualizations/kmeans/kmeans.html

K Means

- Lloyd's algorithm
  1. Choose *k* centroids at random
  2. Repeat until converged:
     i. Assign documents to cluster whose centroid is closest
     ii. Recompute cluster centroids
- Results depend (a lot) on initial guess
  - Not guaranteed to converge
  - Won't find an optimal solution
  - Run multiple times and average the solutions?

Agglomerative clustering

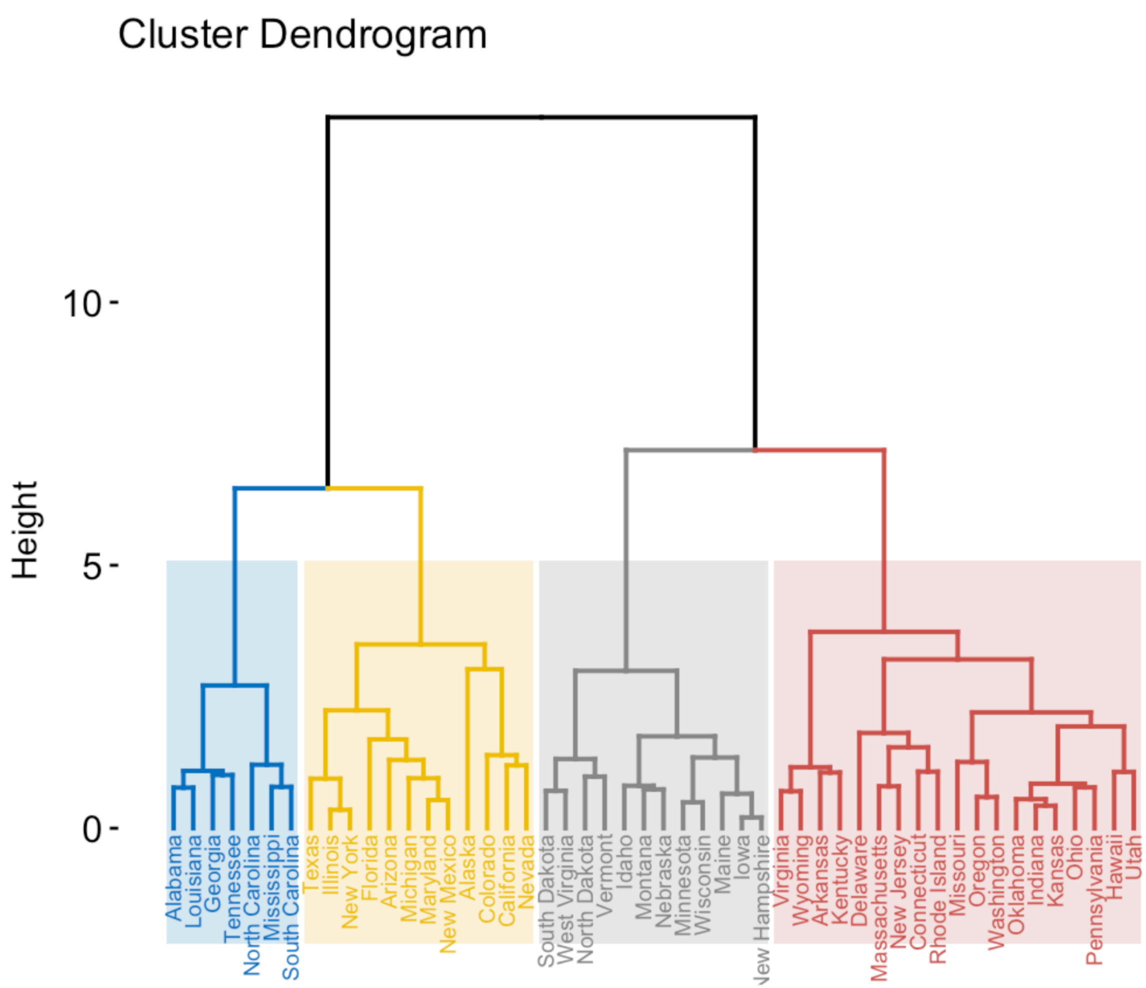
- Weaknesses of K-means:
  - Depends on knowing the number of clusters there are in the data
  - Each item is assigned to one (and only one) cluster
  - No relations among clusters
- Agglomerative clustering starts with an initial cluster assignment (maybe each item is a cluster of one) and progressively merges clusters until

Agglomerative clustering

- **Distance** metrics (cosine, Euclidean, etc) relate pairs of items
- **Linkage** functions determine which clusters to merge at each step
  - **Complete** = merge clusters with the smallest maximum distance
  - **Average** = merge clusters with the smallest average distance
  - **Centroid** = merge clusters with the closest centroids
  - **Ward's** = merge clusters such that the variance within all clusters increases the least (encourages balanced clusters)

Agglomerative clustering

- A **dendrogram** visualizes the hierarchical structure produced by agglomerative clustering



r/conspiracy

r/conspiracy

- We clustered texts on the basis of a document-term matrix
- We can cluster names using a **term-term matrix**
- Start with a binary document vector **d**:

	t_1	t_2	t_3	t_4
d_1	1	0	0	1

r/conspiracy

- We clustered texts on the basis of a document-term matrix
- We can cluster names using a **term-term matrix**
- Take the outer product $\mathbf{d} \otimes \mathbf{d}$

		t_1	t_2	t_3	t_4
		1	0	0	1
t_1	1	1	0	0	1
t_2	0	0	0	0	0
t_3	0	0	0	0	0
t_4	1	1	0	0	1

r/conspiracy

- We clustered texts on the basis of a document-term matrix
- We can cluster names using a **term-term matrix**
- Sum for all the documents in the collection
- Or, use matrix multiplication: $D^{\mathrm{T}}D$