Support Vector Machines

Computational Linguistics: Jordan Boyd-Graber
University of Maryland
AUGUST 22, 2018

Slides adapted from Tom Mitchell, Eric Xing, and Lauren Hannah

Roadmap

- Classification: machines labeling data for us
- Previously: naïve Bayes and logistic regression
- This time: SVMs
  - (another) example of linear classifier
  - Good classification accuracy
  - Good theoretical properties

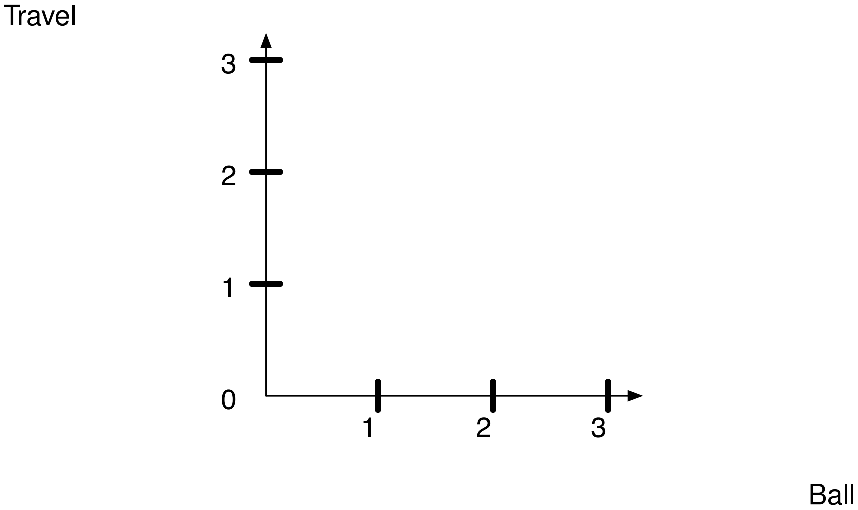
Thinking Geometrically

- Suppose you have two classes: vacations and sports
- Suppose you have four documents

Sports	Vacations
Doc_1: {ball, ball, ball, travel}	Doc_3: {travel, ball, travel}
Doc_2: {ball, ball}	Doc_4: {travel}

- What does this look like in vector space?

Put the documents in vector space



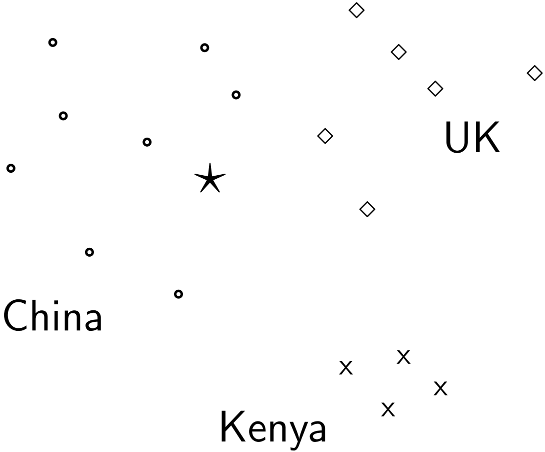
Vector space representation of documents

- Each document is a vector, one component for each term.
- Terms are axes.
- High dimensionality: 10,000s of dimensions and more
- How can we do classification in this space?

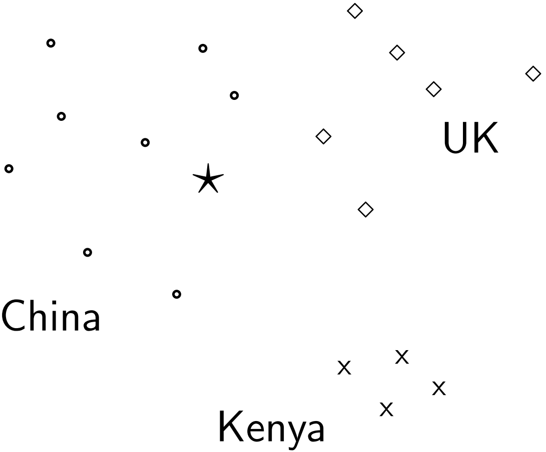
Vector space classification

- As before, the training set is a set of documents, each labeled with its class.
- In vector space classification, this set corresponds to a labeled set of points or vectors in the vector space.
- Premise 1: Documents in the same class form a **contiguous region**.
- Premise 2: Documents from different classes **don't overlap**.
- We define lines, surfaces, hypersurfaces to divide regions.

Classes in the vector space

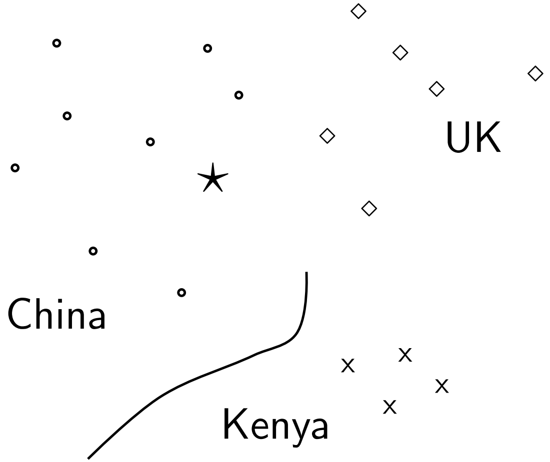


Classes in the vector space



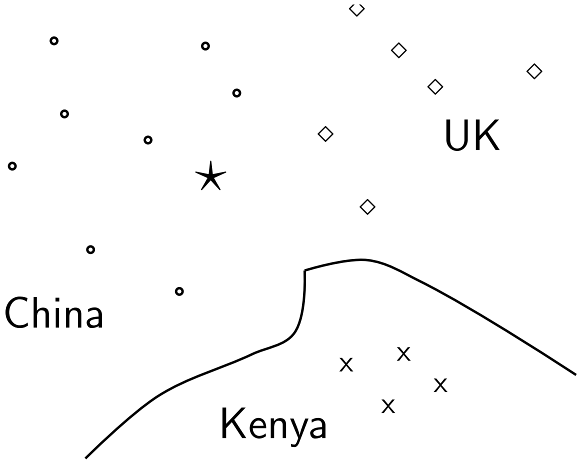
Should the document ⋆ be assigned to `China`, `UK` or `Kenya`?

Classes in the vector space



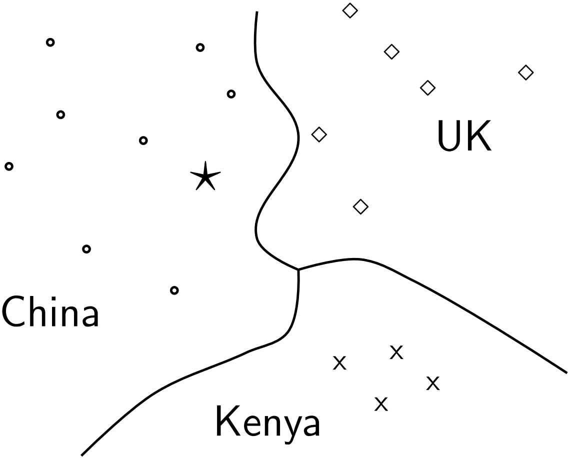
Find separators between the classes

Classes in the vector space



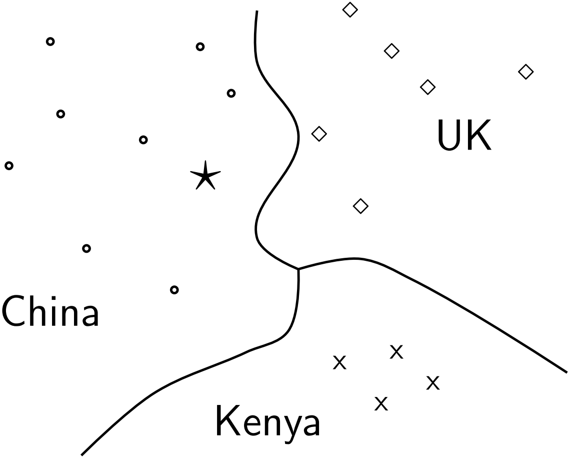
Find separators between the classes

Classes in the vector space



Based on these separators: ⋆ should be assigned to `China`

Classes in the vector space



How do we find separators that do a good job at classifying new documents like $\star$? – Main topic of today

Linear classifiers

- Definition:
  - A linear classifier computes a linear combination or weighted sum $\sum_i \beta_i x_i$ of the feature values.
  - Classification decision: $\sum_i \beta_i x_i > \beta_0$? ($\beta_0$ is our bias)
  - ...where β_0 (the threshold) is a parameter.
- We call this the **separator** or **decision boundary**.
- We find the separator based on training set.
- Methods for finding separator: logistic regression, naïve Bayes, linear SVM
- Assumption: The classes are **linearly separable**.

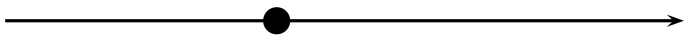
Linear classifiers

- Definition:
  - A linear classifier computes a linear combination or weighted sum $\sum_i \beta_i x_i$ of the feature values.
  - Classification decision: $\sum_i \beta_i x_i > \beta_0$? ($\beta_0$ is our bias)
  - ...where β_0 (the threshold) is a parameter.
- We call this the **separator** or **decision boundary**.
- We find the separator based on training set.
- Methods for finding separator: logistic regression, naïve Bayes, linear SVM
- Assumption: The classes are **linearly separable**.
- Before, we just talked about equations. What's the geometric intuition?

A linear classifier in 1D

- A linear classifier in 1D is a point x described by the equation $\beta_1 x_1 = \beta_0$

A linear classifier in 1D

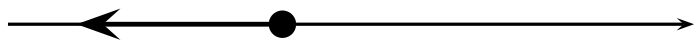


- A linear classifier in 1D is a point x described by the equation $\beta_1 x_1 = \beta_0$
- $x = \beta_0 / \beta_1$

A linear classifier in 1D

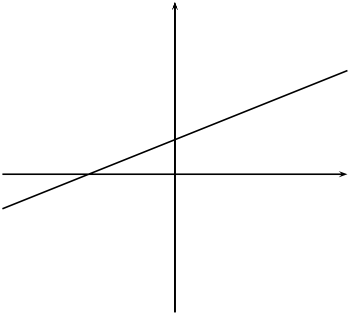
- A linear classifier in 1D is a point x described by the equation $\beta_1 x_1 = \beta_0$
- $x = \beta_0 / \beta_1$
- Points (x_1) with $\beta_1 x_1 \geq \beta_0$ are in the class c.

A linear classifier in 1D



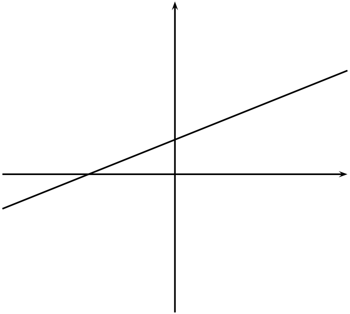
- A linear classifier in 1D is a point x described by the equation $\beta_1 x_1 = \beta_0$
- $x = \beta_0 / \beta_1$
- Points (x_1) with $\beta_1 x_1 \geq \beta_0$ are in the class c.
- Points (x_1) with $\beta_1 x_1 < \beta_0$ are in the complement class $\overline{c}$.

A linear classifier in 2D



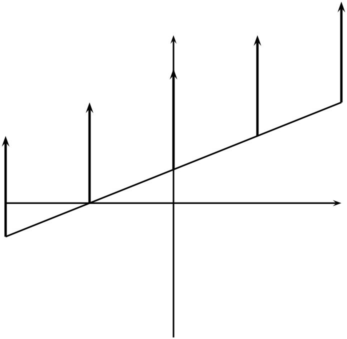
- A linear classifier in 2D is a line described by the equation $\beta_1 x_1 + \beta_2 x_2 = \beta_0$

A linear classifier in 2D



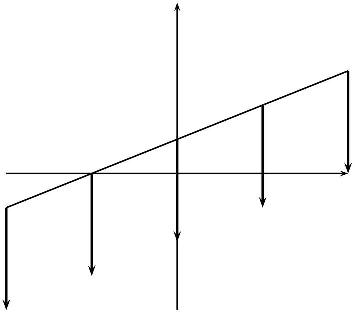
- A linear classifier in 2D is a line described by the equation $\beta_1 x_1 + \beta_2 x_2 = \beta_0$
- Example for a 2D linear classifier

A linear classifier in 2D



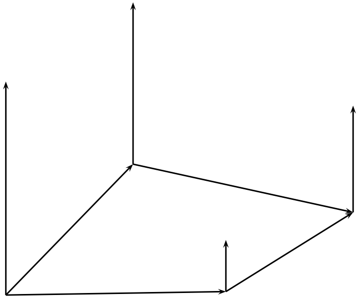
- A linear classifier in 2D is a line described by the equation $\beta_1 x_1 + \beta_2 x_2 = \beta_0$
- Example for a 2D linear classifier
- Points $(x_1 \ x_2)$ with $\beta_1 x_1 + \beta_2 x_2 \geq \beta_0$ are in the class c.

A linear classifier in 2D



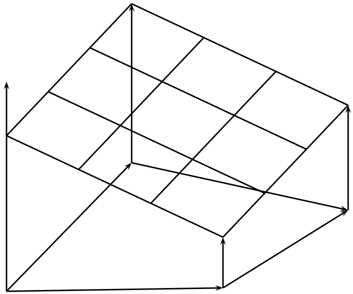
- A linear classifier in 2D is a line described by the equation $\beta_1 x_1 + \beta_2 x_2 = \beta_0$
- Example for a 2D linear classifier
- Points $(x_1 \; x_2)$ with $\beta_1 x_1 + \beta_2 x_2 \geq \beta_0$ are in the class c.
- Points $(x_1 \; x_2)$ with $\beta_1 x_1 + \beta_2 x_2 < \beta_0$ are in the complement class $\overline{c}$.

A linear classifier in 3D



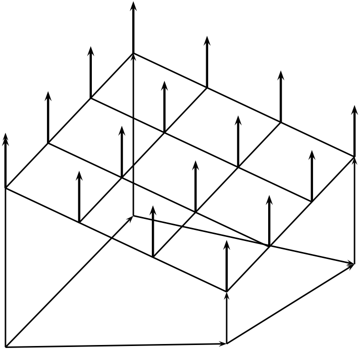
- A linear classifier in 3D is a plane described by the equation $\beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 = \beta_0$

A linear classifier in 3D



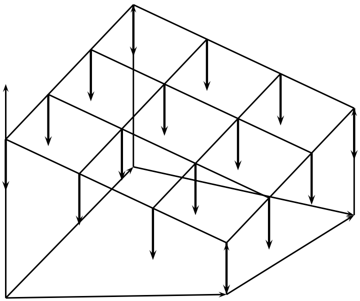
- A linear classifier in 3D is a plane described by the equation $\beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 = \beta_0$
- Example for a 3D linear classifier

A linear classifier in 3D



- A linear classifier in 3D is a plane described by the equation $\beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 = \beta_0$
- Example for a 3D linear classifier
- Points $(x_1 \ x_2 \ x_3)$ with $\beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 \geq \beta_0$ are in the class c.

A linear classifier in 3D



- A linear classifier in 3D is a plane described by the equation $\beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 = \beta_0$
- Example for a 3D linear classifier
- Points $(x_1 \ x_2 \ x_3)$ with $\beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 \geq \beta_0$ are in the class c.
- Points $(x_1 \ x_2 \ x_3)$ with $\beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 < \beta_0$ are in the complement class $\overline{c}$.

Naive Bayes and Logistic Regression as linear classifiers

Multinomial Naive Bayes is a linear classifier (in log space) defined by:

$$\sum_{i=1}^{M} \beta_i x_i = \beta_0$$

where $\beta_i = \log[\hat{P}(t_i|c)/\hat{P}(t_i|\bar{c})]$, $x_i =$ number of occurrences of t_i in d, and $\beta_0 = -\log[\hat{P}(c)/\hat{P}(\bar{c})]$. Here, the index i, $1 \leq i \leq M$, refers to terms of the vocabulary.

Logistic regression is the same (we only put it into the logistic function to turn it into a probability).

Naive Bayes and Logistic Regression as linear classifiers

Multinomial Naive Bayes is a linear classifier (in log space) defined by:

$$\sum_{i=1}^{M} \beta_i x_i = \beta_0$$

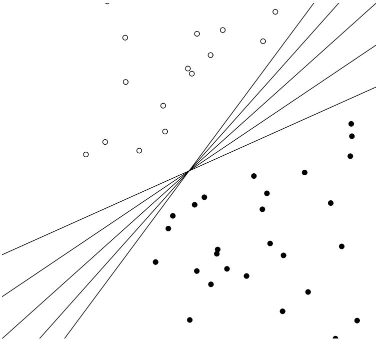
where $\beta_i = \log[\hat{P}(t_i|c)/\hat{P}(t_i|\bar{c})]$, $x_i =$ number of occurrences of t_i in d, and $\beta_0 = -\log[\hat{P}(c)/\hat{P}(\bar{c})]$. Here, the index i, $1 \leq i \leq M$, refers to terms of the vocabulary.

Logistic regression is the same (we only put it into the logistic function to turn it into a probability).

Takeway

Naïve Bayes, logistic regression and SVM are all linear methods. They choose their hyperplanes based on different objectives: joint likelihood (NB), conditional likelihood (LR), and the margin (SVM).

Which hyperplane?



Which hyperplane?

- For linearly separable training sets: there are **infinitely** many separating hyperplanes.
- They all separate the training set perfectly ...
- ...but they behave differently on test data.
- Error rates on new data are low for some, high for others.
- How do we find a low-error separator?

Support vector machines

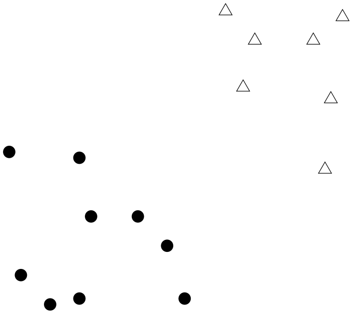
- Machine-learning research in the last two decades has improved classifier effectiveness.
- New generation of state-of-the-art classifiers: support vector machines (SVMs), boosted decision trees, regularized logistic regression, neural networks, and random forests
- Applications to IR problems, particularly text classification

SVMs: A kind of large-margin classifier

Vector space based machine-learning method aiming to find a decision boundary between two classes that is maximally far from any point in the training data (possibly discounting some points as outliers or noise)

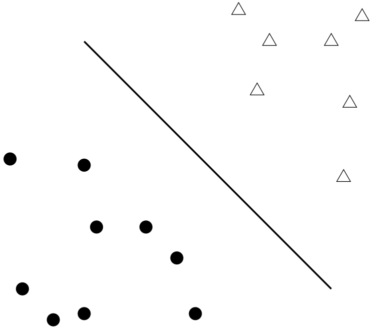
Support Vector Machines

- 2-class training data



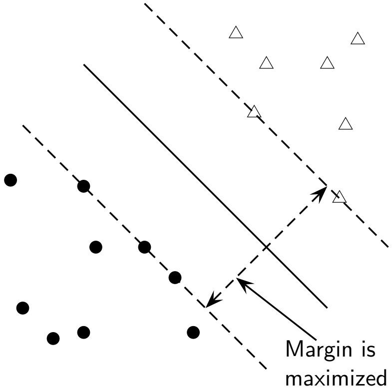
Support Vector Machines

- 2-class training data
- decision boundary $\longrightarrow$ **linear separator**



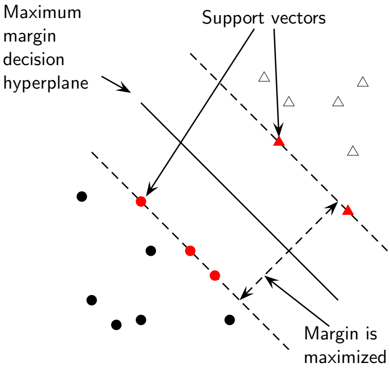
Support Vector Machines

- 2-class training data
- decision boundary → **linear separator**
- criterion: being maximally far away from any data point → determines classifier **margin**



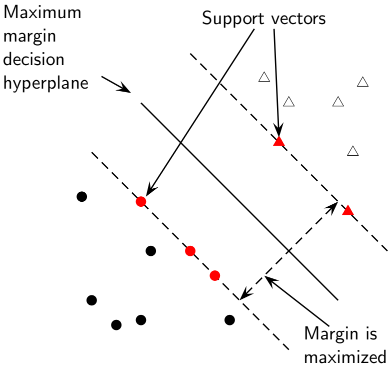
Support Vector Machines

- 2-class training data
- decision boundary → **linear separator**
- criterion: being maximally far away from any data point → determines classifier **margin**
- linear separator position defined by **support vectors**



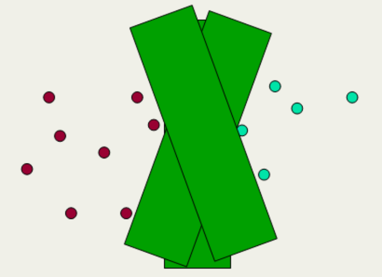
Why maximize the margin?

- Points near decision surface → uncertain classification decisions
- A classifier with a large margin is always confident
- Gives classification safety margin (measurement or variation)



Why maximize the margin?

- SVM classifier: large margin around decision boundary
- compare to decision hyperplane: place fat separator between classes
  - unique solution
- decreased memory capacity
- increased ability to correctly generalize to test data



Equation

- Equation of a hyperplane

$$\vec{w} \cdot x_i + b = 0 \tag{1}$$

- Distance of a point to hyperplane

$$\frac{|\vec{w} \cdot x_i + b|}{||\vec{w}||} \tag{2}$$

- The margin ρ is given by

$$\rho \equiv \min_{(x,y) \in S} \frac{|\vec{w} \cdot x_i + b|}{||\vec{w}||} = \frac{1}{||\vec{w}||} \tag{3}$$

Equation

- Equation of a hyperplane

$$\vec{w} \cdot x_i + b = 0 \tag{1}$$

- Distance of a point to hyperplane

$$\frac{|\vec{w} \cdot x_i + b|}{||\vec{w}||} \tag{2}$$

- The margin ρ is given by

$$\rho \equiv \min_{(x,y)\in S} \frac{|\vec{w} \cdot x_i + b|}{||\vec{w}||} = \frac{1}{||\vec{w}||} \tag{3}$$

- This is because for any point on the marginal hyperplane, $\vec{w} \cdot x + b = \pm 1$

Optimization Problem

We want to find a weight vector $\vec{w}$ and bias b that optimize

$$\min_{\vec{w},b} \frac{1}{2}||w||^2 \tag{4}$$

subject to $y_i(\vec{w} \cdot x_i + b) \geq 1, \forall i \in [1, m]$.

Choosing what kind of classifier to use

When building a text classifier, first question: **how much training data is there currently available?**

- None?
- Very little?
- A fair amount?
- A huge amount

Choosing what kind of classifier to use

When building a text classifier, first question: **how much training data is there currently available?**

- None? **Hand write rules or use active learning**
- Very little?
- A fair amount?
- A huge amount

Choosing what kind of classifier to use

When building a text classifier, first question: **how much training data is there currently available?**

- None? **Hand write rules or use active learning**
- Very little? **Naïve Bayes**
- A fair amount?
- A huge amount

Choosing what kind of classifier to use

When building a text classifier, first question: **how much training data is there currently available?**

- None? **Hand write rules or use active learning**
- Very little? **Naïve Bayes**
- A fair amount? **SVM**
- A huge amount

Choosing what kind of classifier to use

When building a text classifier, first question: **how much training data is there currently available?**

- None? **Hand write rules or use active learning**
- Very little? **Naïve Bayes**
- A fair amount? **SVM**
- A huge amount **Doesn't matter, use whatever works**

SVM extensions: What's next

- Finding solutions
- Slack variables: not perfect line
- Kernels: different geometries