Universität Potsdam
Institut für Informatik
Lehrstuhl Maschinelles Lernen

Text Classification

Tobias Scheffer

Text Classification

- Mapping of text to semantic category:
  - "Amount is due within 10 business days" → *invoice*.
- Can be multi-class classification problems:
  - "I tried this product today and it broke down within a few minutes" → *very negative*.
- Or multiple binary classification problems.
  - "China joins the world trade organization" → {*politics, economics*}.
- Or mapping to nodes in class taxonomy.
  - "wineries, faced with mounting inventory as well as downward price pressure, are forced to reduce their intake of grapes" → economics/agriculture/viniculture

Overview

- Document representation for classification.
- Classification methods.
- Multi-class classification and class taxonomies.
- Evaluation of text classifiers.

Vector-Space Model

- Representations that map documents to a point in a vetor space.
  - $\mathbf{x}_d = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$ ← Hippopotamus

Dictionary: $\begin{bmatrix} \text{Aardvark} \\ \vdots \\ \text{Hippie} \\ \text{Hippopotamus} \\ \text{Hipster} \\ \vdots \\ \text{Zyzzogeton} \end{bmatrix}$

Bag-of-Words Representation

- Term frequency vector: word count for each word .
- High-dimensional but sparse vector.

Hippopotamus

- $\mathbf{x}_d^{\mathrm{TF}} = \begin{pmatrix} 0 \\ \vdots \\ 2 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$

Dictionary: $\begin{bmatrix} \text{Aardvark} \\ \vdots \\ \text{Hippie} \\ \text{Hippopotamus} \\ \text{Hipster} \\ \vdots \\ \text{Zyzzogeton} \end{bmatrix}$

"One hippie was killed by the hippopotamus,
a second hippie survived but was injured."

Bag-of-Words Representation

- Some words and are not relevant for classification.
  - E.g., “was”, “by”, “the”, “a”, “but”.
- Typically, these words occur all the time.
- Often, words that occur in few documents have relevance for classification.
  - E.g., “hippie”, “hippopotamus”, “survived”, “injured”.

“One hippie was killed by the hippopotamus,
a second hippie survived but was injured.”

Inverse Document Frequency

- Measure of how rare a term is.
- $IDF(\text{term}_i) = \log \frac{\text{\#of documents in corpus}}{\text{\#of documents that include term}_i}$

- $IDF(\text{hippopotamus}) = \log \frac{10\,000}{85} = 2.07$
- $IDF(\text{and}) = \log \frac{10\,000}{9\,876} = 0.0054189$

- Inverse-document-frequency vector:
  - $\mathbf{x}^{\text{IDF}} = \begin{pmatrix} IDF(\text{Aarvark}) \\ \vdots \\ IDF(\text{Zyzzogeton}) \end{pmatrix}$

TF-IDF Representation

- Product of term-frquency and inverse-document-frequency vectors.
  - $\mathbf{x}_d^{\text{TFIDF}} = \mathbf{x}_d^{\text{TF}} \odot \mathbf{x}^{\text{IDF}}$

- $$\mathbf{x}_d^{\text{TFIDF}} = \begin{pmatrix} 0 \\ \vdots \\ 2 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \odot \begin{pmatrix} 2.8 \\ \vdots \\ 2.07 \\ 1.68 \\ 0.43 \\ \vdots \\ 2.9 \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 4.14 \\ 1.68 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

"One hippie was killed by the hippopotamus, a second hippie survived but was injured."

TF-IDF Representation

- In linear classifiers, vectors with large norms result in large decision-function values.
- Therefore: normalized TD-IDF representation:
  - $\bar{\mathbf{x}}_d^{\text{TFIDF}} = \frac{\mathbf{x}_d^{\text{TFIDF}}}{|\mathbf{x}_d^{\text{TFIDF}}|}$
- $\bar{\mathbf{x}}_d^{\text{TFIDF}} = \frac{\begin{pmatrix} 0 \\ \vdots \\ 4.14 \\ 1.68 \\ 0 \\ \vdots \\ 0 \end{pmatrix}}{\sqrt{4.14^2+1.68^2+\cdots}}$

TF-IDF Representation

- Several alternative weighting schemes are common.

Term frequency		Document frequency		Normalization	
n (natural)	$\mathrm{tf}_{t,d}$	n (no)	1	n (none)	1
l (logarithm)	$1 + \log(\mathrm{tf}_{t,d})$	t (idf)	$\log \frac{N}{\mathrm{df}_t}$	c (cosine)	$\frac{1}{\sqrt{w_1^2+w_2^2+\ldots+w_M^2}}$
a (augmented)	$0.5 + \frac{0.5\times \mathrm{tf}_{t,d}}{\max_t(\mathrm{tf}_{t,d})}$	p (prob idf)	$\max\{0, \log \frac{N-\mathrm{df}_t}{\mathrm{df}_t}\}$	u (pivoted unique)	$1/u$ (Section 6.4.4)
b (boolean)	$\begin{cases}1 & \text{if } \mathrm{tf}_{t,d} > 0 \\ 0 & \text{otherwise}\end{cases}$			b (byte size)	$1/CharLength^{\alpha}$, $\alpha < 1$
L (log ave)	$\frac{1+\log(\mathrm{tf}_{t,d})}{1+\log(\mathrm{ave}_{t\in d}(\mathrm{tf}_{t,d}))}$				

- Manning, et.al.: Introduction to Information Retrieval: http://nlp.stanford.edu/IR-book/

Bag-of-Words Representations

- Vector space representations disregard any word ordering.
  - “product is broken, not great!” ≈ “product is great, not broken”!

N-Gram Vectors

- One dimension for each term n-gram that occurs at least k times in the corpus.
- $$\mathbf{x}_d = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

"Hippopotamus, a"

Dictionary: $\begin{bmatrix} \text{Aardvark been} \\ \vdots \\ \text{Hippie was} \\ \text{Hippopotamus a} \\ \text{Hipster has} \\ \vdots \\ \text{Zyzzogeton zone} \end{bmatrix}$

"One hippie was killed by the hippopotamus, a second hippie survived but was injured."

N-Gram Vectors

- One dimension for each term n-gram that occurs at least k times in the corpus.
- Number of n-grams grows exponentially in n.
- Semantically similar n-grams have independent dimensions $\rightarrow$ not ideal for generalization.

Skip-Gram Vectors

- n-gram vectors with wild-card symbol.
- Several similar coding schemes are common, for instance orthogonal sparse bigrams (bigrams that occur within k words of each other).

“Hippopotamus * hippie”

- $$\mathbf{x}_d = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

Dictionary:

$$\begin{bmatrix} \text{Aardvark * ant} \\ \vdots \\ \text{Hippie * smoke} \\ \text{Hippopotamus* hippie} \\ \text{Hipster * skinny} \\ \vdots \\ \text{Zyzzogeton * zone} \end{bmatrix}$$

“One hippie was killed by the hippopotamus, a second hippie survived but was injured.”

Skip-Gram Vectors

- Semantically related terms still have unrelated representations.
- Text classifier cannot “know” that when a term correlates with a category, semantically similar terms may also correlate with the same category.

Semantic Representations

- Several attempts at semantic representations have been made, and basically failed.
  - Latent semantic indexing.
  - Latent Dirichlet allocation.
- Classes of “semantically similar” terms are too heterogenous, adds noise to classification problem.

- Continuous-space (“neural”) language models seem to actually work.

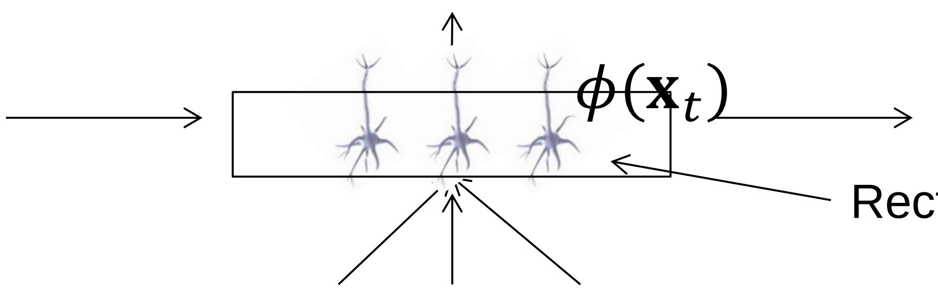
Neural Language Model

Also uses Markov assumption of order $n-1$!

$$\begin{pmatrix} 0.001 \\ \vdots \\ 0.6 \\ 0.1 \\ 0.02 \\ \vdots \\ 0 \end{pmatrix} =$$

Softmax output layer

- Output $\ldots \mathbf{x}_{t-1} \quad \mathbf{x}_t \quad \mathbf{x}_{t+1} \ldots \mathbf{x}_T$



- Input $\mathbf{x}_1 \quad \ldots \mathbf{x}_{t-n+1} \ldots \quad \mathbf{x}_{t-1} \ldots \mathbf{x}_T.$

$$= \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \qquad = \begin{pmatrix} 0 \\ \vdots \\ 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

"Term-frequency vector" for single term

Neural Text Representations

- Process all term n-grams in the document, infer hidden representation $\phi(\mathbf{x}_t)$ for each term.
  - $\mathbf{x}_d^{\mathrm{AVG}} = \frac{1}{T}\sum_i^T \phi(\mathbf{x}_t)$
- Dramatically reduces dimensionality of vector space.
- Can improve text classification accuracy.
  - "Product is broken, not great" and "product is great, not broken" can have distinct representations.
- Context information still limited to n subsequent terms.

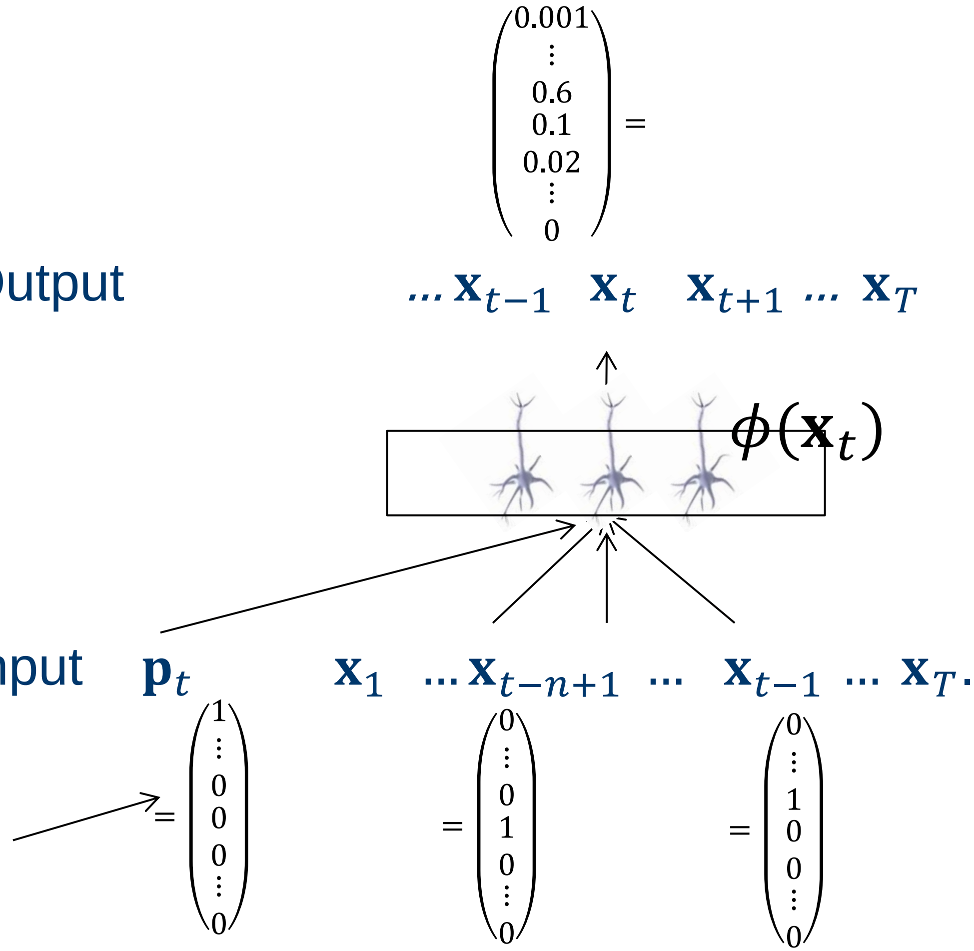
Paragraph Vectors

- Semantic representation of paragraphs.
- Can also be applied with different levels of granularities:
  - Semantic representation of sentences.
  - Semantic representation of documents.

- Semantically related paragraphs (or documents, sentences) have similar representation.

Paragraph Vectors

- Output
- Input

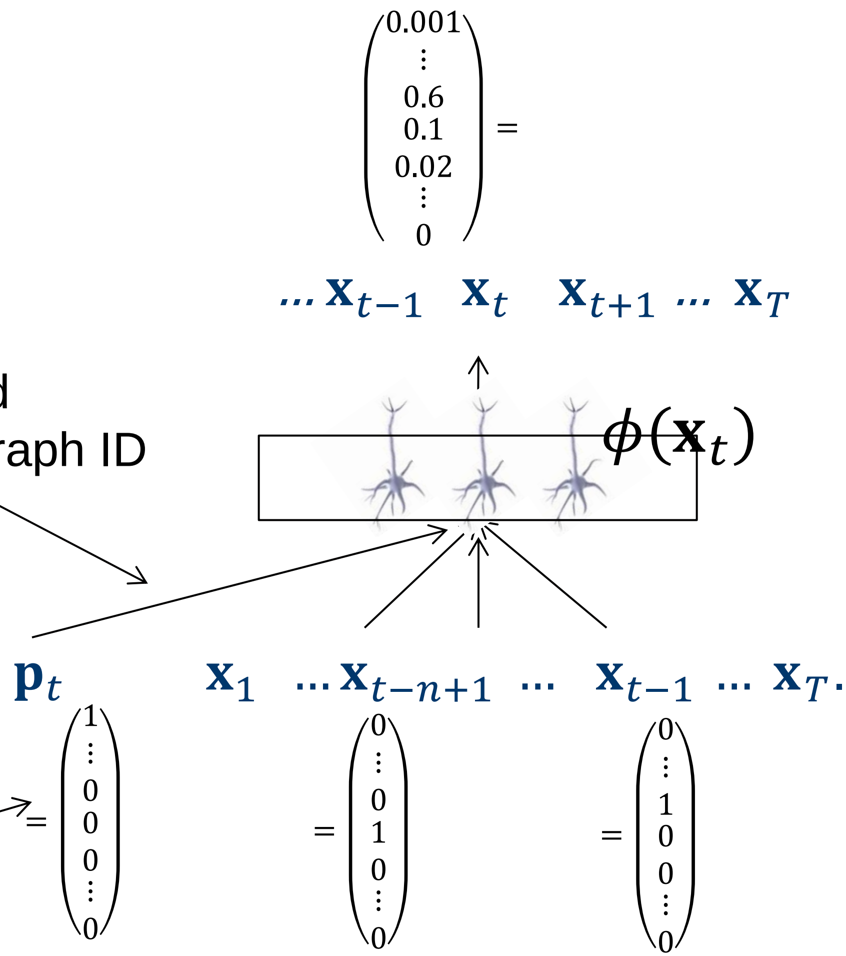


One-hot coded paragraph ID. One dimension per paragraph in the corpus.

Paragraph Vectors

- Output
- Input

Weights between hidden states and paragraph ID: Embedding of paragraph ID in semantic feature space Φ.

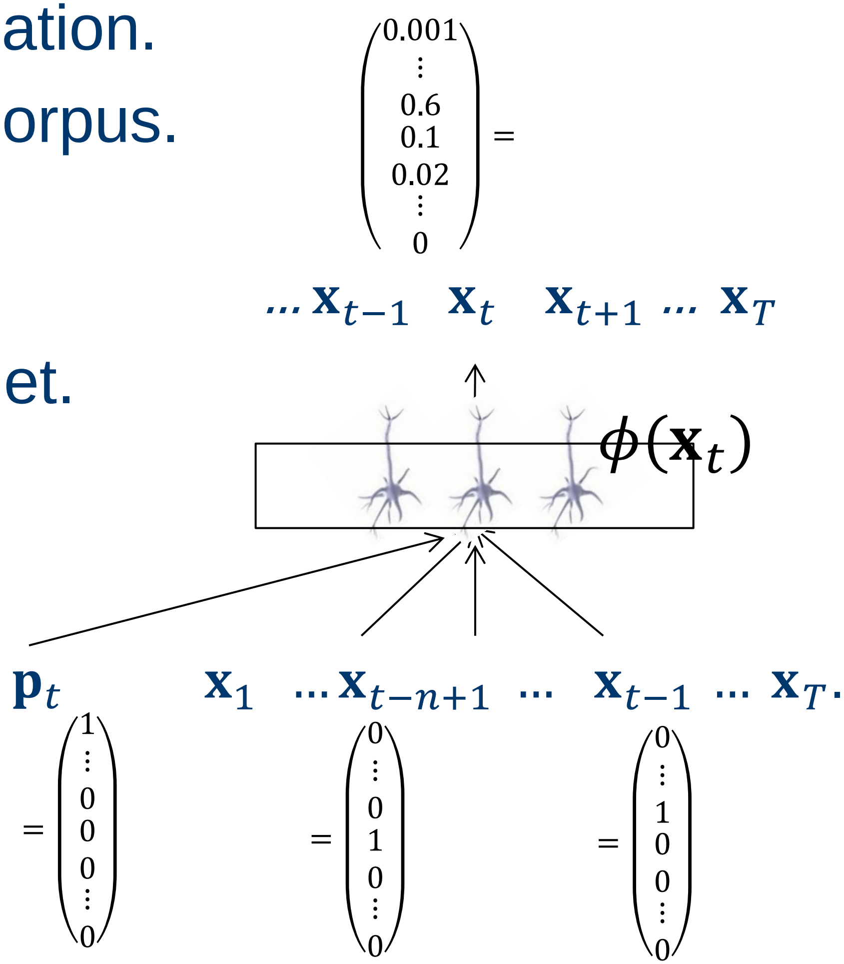


One-hot coded paragraph ID. One dimension per paragraph in the corpus.

Training Paragraph-Vector Models

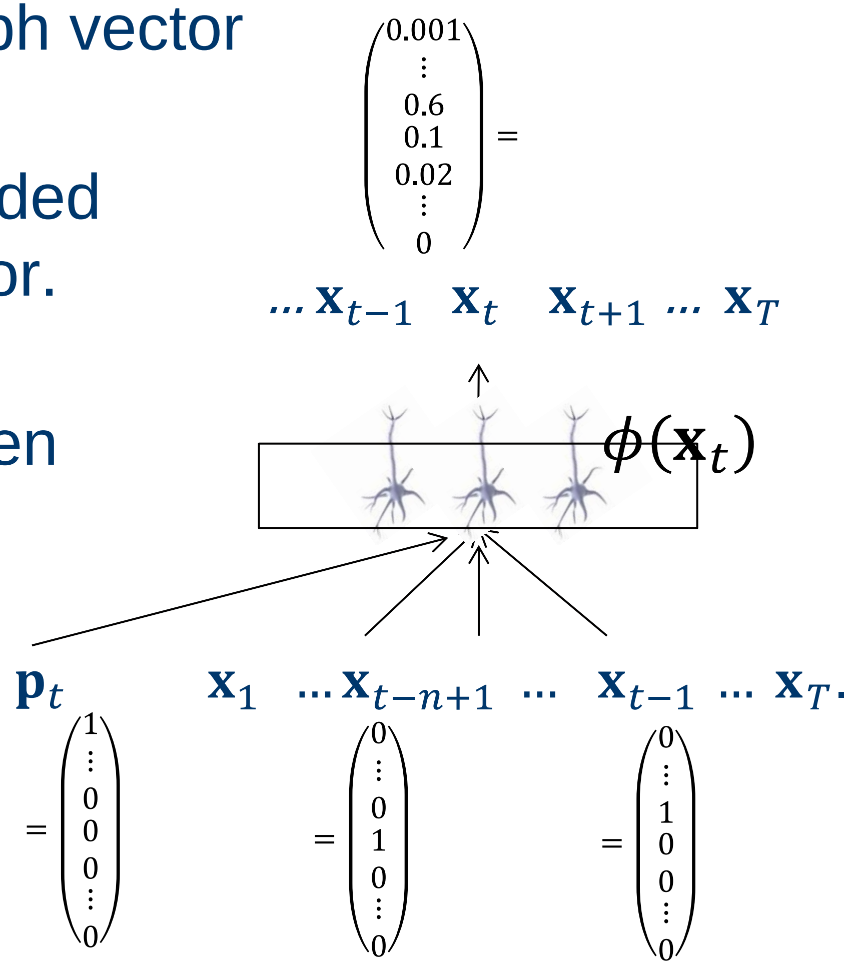
- Train by back propagation.
- Iterate over training corpus.
- Input paragraph ID,
  term $(n-1)$-gram,
- Use n-th term as target.

- → Model is trained to predict next term, given paragraph ID and preceding terms.



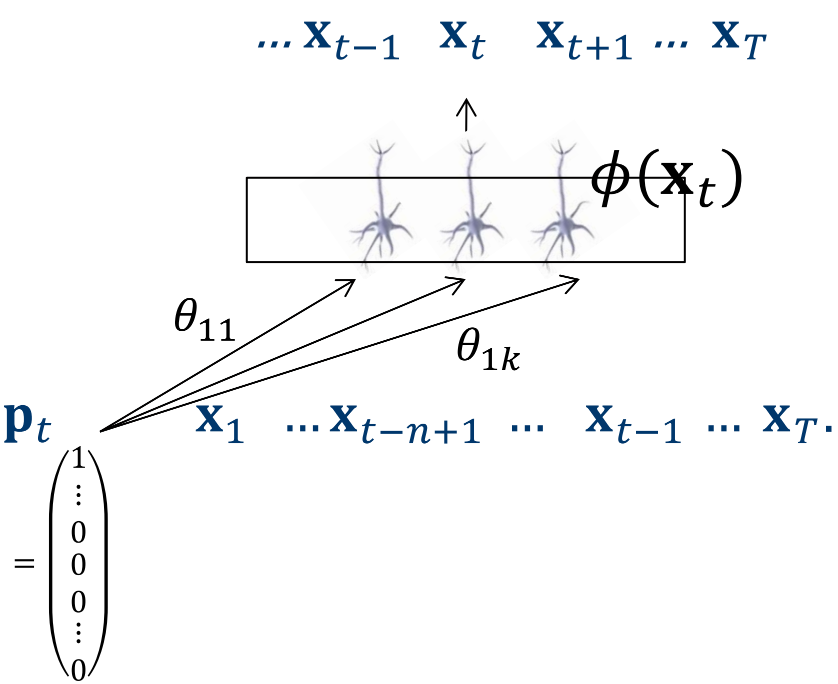
Applying Paragraph-Vector Models

- Inference of paragraph vector for new paragraph.
- New dimension is added to paragraph ID vector.
- Weights from new paragraph ID to hidden units are trained on n-grams from new paragraph.
- Other weights are frozen.



Paragraph Vector Representation

- Weights from paragraph ID to hidden units for document d are used as representation of d.
  - $\mathbf{x}_d^{\text{PAR}} = \begin{pmatrix} \theta_{d1} \\ \vdots \\ \theta_{dk} \end{pmatrix}$

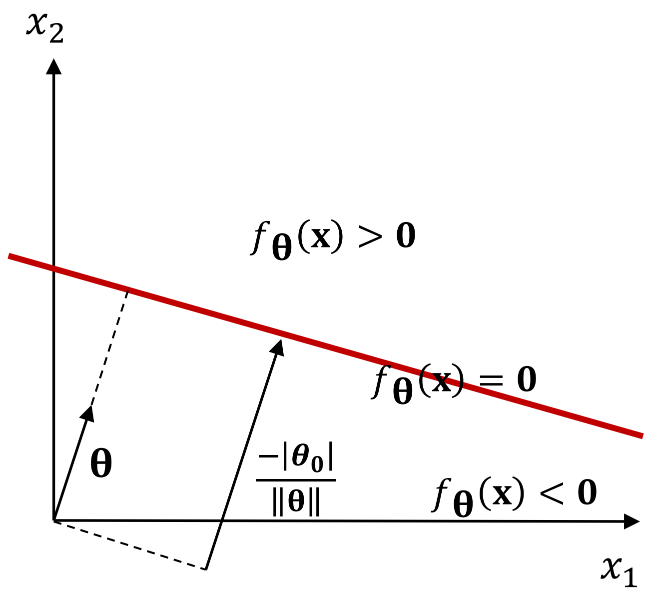


Overview

- Document representation for classification.
- Classification methods.
- Multi-class classification and class taxonomies.
- Evaluation of text classifiers.

Text Classification

- Most often, linear classification models are used for text classification.
- Example: $X = \mathbb{R}^2$
- Decision function:
  $f_{\boldsymbol{\theta}}(\mathbf{x}) = \mathbf{x}^{\mathrm{T}}\boldsymbol{\theta} + \theta_0$
- For binary classification,
  $y \in \{+1, -1\}$:
  $y_{\boldsymbol{\theta}}(\mathbf{x}) = \mathrm{sign}\left(f_{\boldsymbol{\theta}}(\mathbf{x})\right)$



Scheffer: Language Technology

Regularized Empirical Risk Minimization

- Solve

$$\underset{\boldsymbol{\theta}}{\operatorname{argmin}} \sum_{i=1}^{n} \ell(f_{\boldsymbol{\theta}}(\mathbf{x}_i), y_i) + \lambda\, \Omega(\boldsymbol{\theta})$$

- Loss function $\ell(f_{\boldsymbol{\theta}}(\mathbf{x}_i), y_i)$: cost of the model's output $f_{\boldsymbol{\theta}}(\mathbf{x})$ when the true value is y.
  - The empirical risk is $\hat{R}_n(\boldsymbol{\theta}) = \sum_{i=1}^{n} \ell(f_{\boldsymbol{\theta}}(\mathbf{x}_i), y_i)$
- Regularizer $\Omega(\boldsymbol{\theta})$ & trade-off parameter $\lambda \geq 0$:
  - Background information about preferred solutions
  - Provides numerical stability (Tikhonov-Regularizer)
  - allows for tighter error bounds (PAC-Theory)

Regularized Empirical Risk Minimization

- Linear classification model: minimize

$$\boldsymbol{L}(\boldsymbol{\theta}) = \sum_{i=1}^{n} \ell\left(\mathbf{x}^{\mathrm{T}}\boldsymbol{\theta}, y_i\right) + \lambda\, \Omega(\boldsymbol{\theta})$$

- Gradient:
  - Vector of the derivatives with respect to each individual parameter
  - Direction of the steepest increase of the function $L(\boldsymbol{\theta})$.

$$\nabla L(\boldsymbol{\theta}) = \begin{pmatrix} \frac{\partial L(\boldsymbol{\theta})}{\partial \theta_1} \\ \vdots \\ \frac{\partial L(\boldsymbol{\theta})}{\partial \theta_m} \end{pmatrix}$$

Regularized Empirical Risk Minimization

- Linear classification model: minimize

$$L(\boldsymbol{\theta}) = \sum_{i=1}^{n} \ell\left(\mathbf{x}^{\mathrm{T}}\boldsymbol{\theta}, y_i\right) + \lambda\,\Omega(\boldsymbol{\theta})$$

- Gradient descent method:

RegERM(*Data:* $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$)

  Set $\boldsymbol{\theta}^0 = \mathbf{0}$ and $t = 0$

  DO

    Compute gradient $\nabla L(\boldsymbol{\theta}^t)$

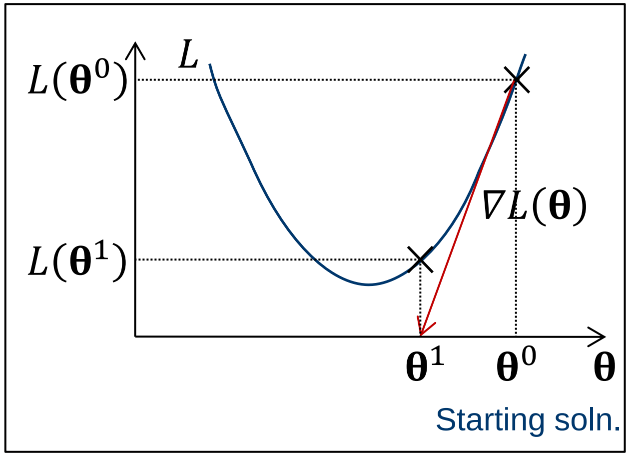
    Compute step size α^t

    Set $\boldsymbol{\theta}^{t+1} = \boldsymbol{\theta}^t - \alpha^t \nabla L(\boldsymbol{\theta}^t)$

    Set $t = t + 1$

  WHILE $\|\boldsymbol{\theta}^t - \boldsymbol{\theta}^{t+1}\| > \varepsilon$

RETURN $\boldsymbol{\theta}^t$



Regularized Empirical Risk Minimization

- Linear classification model: minimize

$$L(\boldsymbol{\theta}) = \sum_{i=1}^{n} \ell\left(\mathbf{x}^{\mathrm{T}}\boldsymbol{\theta}, y_i\right) + \lambda\, \Omega(\boldsymbol{\theta})$$

- Large training sets: stochastic gradient descent.

RegERM-Stoch(*Data:* $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)$)

    Set $\boldsymbol{\theta}^0 = \mathbf{0}$ and $t = 0$

    DO

        Shuffle data randomly

        FOR $i = 1, \dots, n$

            Compute subset gradient $\nabla_{\mathbf{x}_i} L(\boldsymbol{\theta}^t)$

            Compute step size α^t

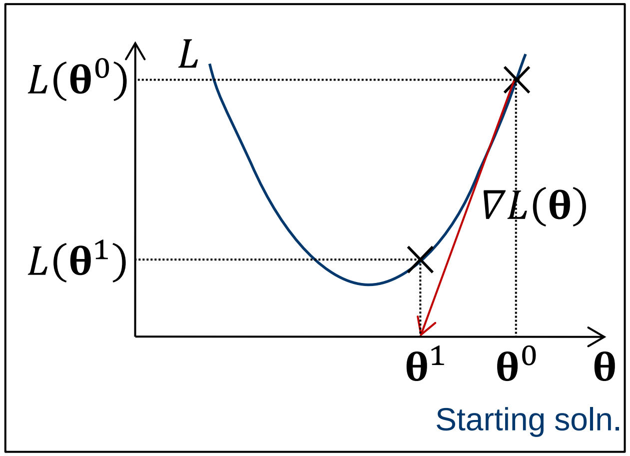
            Set $\boldsymbol{\theta}^{t+1} = \boldsymbol{\theta}^t - \alpha^t \nabla_{\mathbf{x}_i} L(\boldsymbol{\theta}^t)$

            Set $t = t + 1$

        END

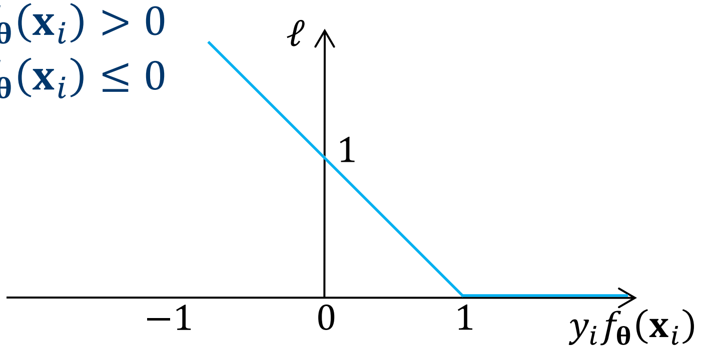
    WHILE $\|\boldsymbol{\theta}^t - \boldsymbol{\theta}^{t+1}\| > \varepsilon$

RETURN $\boldsymbol{\theta}^t$



ERM: Support Vector Machine (SVM)

- Class $y \in \{-1, +1\}$
- Loss function:
  - $$\ell_h(f_{\boldsymbol{\theta}}(\mathbf{x}_i), y_i) = \begin{cases} 1 - y_i f_{\boldsymbol{\theta}}(\mathbf{x}_i) & \text{if } 1 - y_i f_{\boldsymbol{\theta}}(\mathbf{x}_i) > 0 \\ 0 & \text{if } 1 - y_i f_{\boldsymbol{\theta}}(\mathbf{x}_i) \leq 0 \end{cases}$$
    $$= \max\left(0, 1 - y_i f_{\boldsymbol{\theta}}(\mathbf{x}_i)\right)$$



- Regularizer:
  - $\Omega_2(\boldsymbol{\theta}) = \boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{\theta} = \sum_{j=1}^{m} |\theta_j|^2 = \|\boldsymbol{\theta}\|_2^2$

Scheffer: Language Technology

Support Vector Machine (SVM)

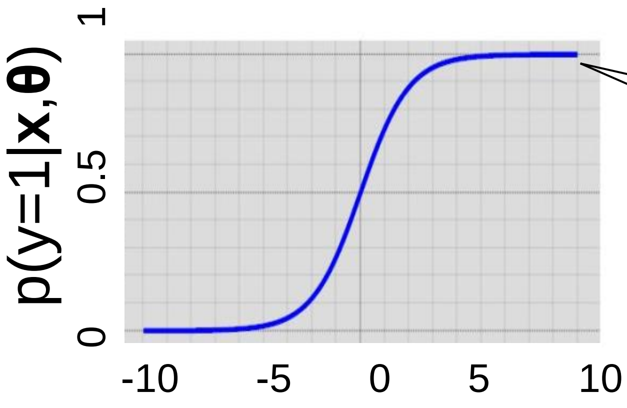
- $\boldsymbol{L}(\boldsymbol{\theta})$ can be minimized using stochastic gradient descent method ("Pegasos")
  - Very fast, often used in practice
- $\boldsymbol{L}(\boldsymbol{\theta})$ can be minimized using gradient descent method ("Primal SVM")

Logistic Regression

- „Logistic regression" is a model for classification!
- For now, binary classification with $y_i \in \{-1,1\}$.

- Need: model for $p(y \mid \mathbf{x}, \boldsymbol{\theta})$
  - Model defines probability $p(y=1 \mid \mathbf{x}, \boldsymbol{\theta})$.
  - Probability $p(y=-1 \mid \mathbf{x}, \boldsymbol{\theta}) = 1 - p(y=1 \mid \mathbf{x}, \boldsymbol{\theta})$.

- Idea: transformation of a linear model $\mathbf{x}^{\mathrm{T}}\boldsymbol{\theta}$.



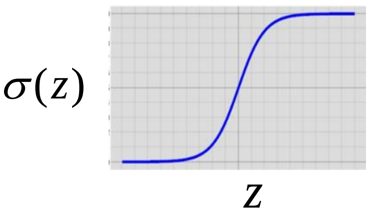
$\mathbf{x}^{\mathrm{T}}\boldsymbol{\theta}$

Sigmoid function („Squashing function") maps interval $[-\infty,\infty]$ to [0,1].

Logistic Regression

- Model logistic regression
  - Given by parameter vector $\boldsymbol{\theta} \in \mathbb{R}^m$.
  - Defines conditional distribution $p(y \mid \mathbf{x}, \boldsymbol{\theta})$ by

$$p(y=1 \mid \mathbf{x}, \boldsymbol{\theta}) = \sigma(\mathbf{x}^{\mathrm{T}} \boldsymbol{\theta}) = \frac{1}{1+\exp(-\mathbf{x}^{\mathrm{T}} \boldsymbol{\theta})}$$



$$p(y=-1 \mid \mathbf{x}, \boldsymbol{\theta}) = 1 - p(y=1 \mid \mathbf{x}, \boldsymbol{\theta})$$

- Prediction function $f_{\boldsymbol{\theta}} : \mathbb{R}^m \to \{0,1\}$:

$$f_{\boldsymbol{\theta}}(\mathbf{x}) = \begin{cases} 1: & \sigma(\mathbf{x}^{\mathrm{T}} \boldsymbol{\theta}) \geq 0.5 \\ 0: & \text{sonst} \end{cases}$$

Learning Logistic Regression Models

- MAP model: minimize regularized loss.

$$\boldsymbol{\theta}_{\text{MAP}} = \arg\max_{\boldsymbol{\theta}} P(\mathbf{y} \mid \mathbf{X}, \boldsymbol{\theta})\, p(\boldsymbol{\theta})$$

$$= \arg\min_{\boldsymbol{\theta}} \underbrace{\sum_{i=1}^{n} \log\left(1 + \exp(-y_i \mathbf{x}_i^{\mathrm{T}} \boldsymbol{\theta})\right)}_{\text{loss function}} + \underbrace{\frac{1}{2\sigma_p^2} |\boldsymbol{\theta}|^2}_{\text{regularizer}}$$

- Convex optimization problem, global minimum.
- Compare earlier lecture on „Linear models“.

Overview

- Document representation for classification.
- Classification methods.
- Multi-class classification and class taxonomies.
- Evaluation of text classifiers.

Multi-Class Classification

- Text classification problems with more than 2 classes.
  - $Y = \{1, \dots, k\}$
- Problem: we cannot separate k classes with a single hyperplane.
- Idea: Each class y has a separate function $f_{\boldsymbol{\theta}}(\mathbf{x}, y)$ that is used to predict how likely y is given $\mathbf{x}$.
  - Each function is modeled as linear.
  - We predict class y with the highest scoring function for $\mathbf{x}$.

Multi-Class Classification

- Decision functions:

$$f_{\boldsymbol{\theta}}(\mathbf{x}, y) = \mathbf{x}^{\mathrm{T}} \boldsymbol{\theta}^{y}$$

- Classifier:

$$y_{\boldsymbol{\theta}}(\mathbf{x}) = \underset{y \in Y}{\operatorname{argmax}} f_{\boldsymbol{\theta}}(\mathbf{x}, y)$$

- Model parameters (k classes):

$$\boldsymbol{\theta} = \begin{pmatrix} \boldsymbol{\theta}^{1} \\ \vdots \\ \boldsymbol{\theta}^{k} \end{pmatrix}$$

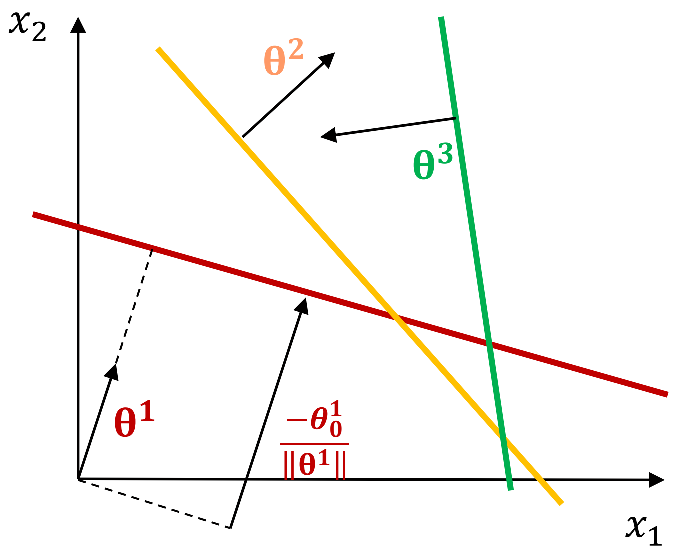
Multi-Class Classification

- Decision functions:

$$f_{\boldsymbol{\theta}}(\mathbf{x}, y) = \mathbf{x}^{\mathrm{T}}\boldsymbol{\theta}^{y}$$

- Classifier:

$$y_{\boldsymbol{\theta}}(\mathbf{x}) = \underset{y \in Y}{\operatorname{argmax}} f_{\boldsymbol{\theta}}(\mathbf{x}, y)$$



Scheffer: Language Technology

Multi-Class Classification

- Decision function:

$$f_{\boldsymbol{\theta}}(\mathbf{x}, y) = \mathbf{x}^{\mathrm{T}} \boldsymbol{\theta}^{y} \text{ with } \boldsymbol{\theta} = \begin{pmatrix} \boldsymbol{\theta}^{1} \\ \vdots \\ \boldsymbol{\theta}^{k} \end{pmatrix}$$

- Decision function in terms of a joint feature mapping of input and output:

$$f_{\boldsymbol{\theta}}(\mathbf{x}, y) = \Phi(\mathbf{x}, y)^{\mathrm{T}} \boldsymbol{\theta} \text{ with } \Phi(\mathbf{x}, y) = \begin{pmatrix} \mathbf{x}[y = 1] \\ \vdots \\ \mathbf{x}[y = k] \end{pmatrix}$$

Multi-Class Classification

- Decision function in terms of a joint feature mapping of input and output:

$$f_{\boldsymbol{\theta}}(\mathbf{x}, y) = \Phi(\mathbf{x}, y)^{\mathrm{T}}\boldsymbol{\theta} \text{ with } \Phi(\mathbf{x}, y) = \begin{pmatrix} \mathbf{x}[y = 1] \\ \vdots \\ \mathbf{x}[y = k] \end{pmatrix}$$

- Example: 3-class classification

$$f_{\boldsymbol{\theta}}(\mathbf{x}, 2) = \Phi(\mathbf{x}, 2)^{\mathrm{T}}\boldsymbol{\theta}$$

$$= (\mathbf{x}[2 = 1] \quad \mathbf{x}[2 = 2] \quad \mathbf{x}[2 = 3]) \begin{pmatrix} \boldsymbol{\theta}^1 \\ \boldsymbol{\theta}^2 \\ \boldsymbol{\theta}^3 \end{pmatrix}$$

$$= (0 \quad \mathbf{x} \quad 0) \begin{pmatrix} \boldsymbol{\theta}^1 \\ \boldsymbol{\theta}^2 \\ \boldsymbol{\theta}^3 \end{pmatrix} = \mathbf{x}^{\mathrm{T}}\boldsymbol{\theta}^2$$

Multi-Class Classification

- Decision function in terms of a joint feature mapping of input and output:
  $f_{\boldsymbol{\theta}}(\mathbf{x}, y) = \Phi(\mathbf{x}, y)^{\mathrm{T}}\boldsymbol{\theta}$ with $\Phi(\mathbf{x}, y) = \Phi(\mathbf{x}) \times \Lambda(y)$,
  $\Phi(\mathbf{x}) = \mathbf{x}$, and $\Lambda(y) = \begin{pmatrix} [y = 1] \\ \vdots \\ [y = k] \end{pmatrix}$
- Example: 3-class classification

$$\Phi(\mathbf{x}, y) = \Phi(\mathbf{x}) \times \Lambda(y)$$

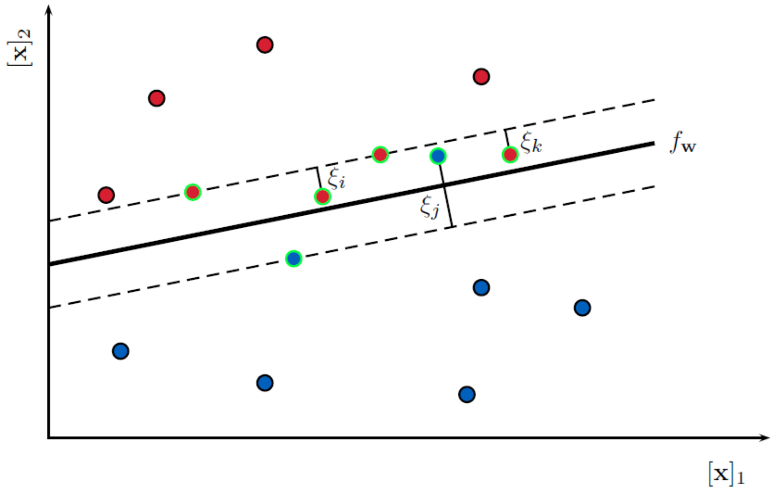
$$= \mathbf{x} \times \begin{pmatrix} [y = 1] \\ [y = 2] \\ [y = 3] \end{pmatrix} = \begin{pmatrix} \mathbf{x}[y = 1] \\ \mathbf{x}[y = 2] \\ \mathbf{x}[y = 3] \end{pmatrix}$$

Binary SVM: Optimization Problem

- minimize

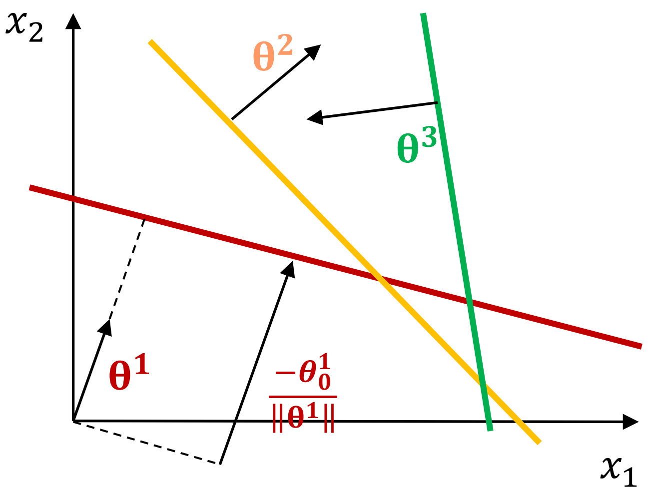
$$\boldsymbol{L}(\boldsymbol{\theta}) = \sum_{i=1}^{n} \left[\max\left(0, 1 - y_i \mathbf{x}_i{}^{\mathrm{T}} \boldsymbol{\theta}\right)\right] + \lambda \boldsymbol{\theta}^{\mathrm{T}} \boldsymbol{\theta}$$

- Equivalent to: minimize
  $\boldsymbol{L}(\boldsymbol{\theta}) = \sum_{i=1}^{n} \xi_i + \lambda \boldsymbol{\theta}^{\mathrm{T}} \boldsymbol{\theta}$ subject to the constraints
  - $y_i f_{\boldsymbol{\theta}}(\mathbf{x}_i) \geq 1 - \xi_i$
  - $\xi_i \geq 0$



Multi-Class SVM: Optimization Problem

- minimize
  $$L(\boldsymbol{\theta}) = \sum_{i=1}^{n} \left[\max_{y \neq y_i} \left(0, f_{\boldsymbol{\theta}}(\mathbf{x}_i, y) + 1 - f_{\boldsymbol{\theta}}(\mathbf{x}_i, y_i)\right)\right] + \lambda \boldsymbol{\theta}^{\mathrm{T}} \boldsymbol{\theta}$$
- Minimize
  $L(\boldsymbol{\theta}) = \sum_{i=1}^{n} \xi_i + \lambda \boldsymbol{\theta}^{\mathrm{T}} \boldsymbol{\theta}$ subject to the constraints
  - $\forall y \neq y_i: f_{\boldsymbol{\theta}}(\mathbf{x}_i, y_i) \geq f_{\boldsymbol{\theta}}(\mathbf{x}_i, y) + 1 - \xi_i$
  - $\xi_i \geq 0$



SVM-Struct: Learning

- Large-margin optimization criterion: minimize

$$\boldsymbol{L}(\boldsymbol{\theta}) = \sum_{i=1}^{n} \left[\max_{y \neq y_i} \left(0, f_{\boldsymbol{\theta}}(\mathbf{x}_i, \mathbf{y}) + 1 - f_{\boldsymbol{\theta}}(\mathbf{x}_i, \mathbf{y}_i) \right) \right] + \lambda \boldsymbol{\theta}^{\mathrm{T}} \boldsymbol{\theta}$$

- Equivalent to minimize $\boldsymbol{L}(\boldsymbol{\theta}) = \sum_{i=1}^{n} \xi_i + \lambda \boldsymbol{\theta}^{\mathrm{T}} \boldsymbol{\theta}$ subject to the constraints
  - $\forall \mathbf{y} \neq \mathbf{y}_i: f_{\boldsymbol{\theta}}(\mathbf{x}_i, \mathbf{y}_i) \geq f_{\boldsymbol{\theta}}(\mathbf{x}_i, \mathbf{y}) + 1 - \xi_i$
  - $\xi_i \geq 0$

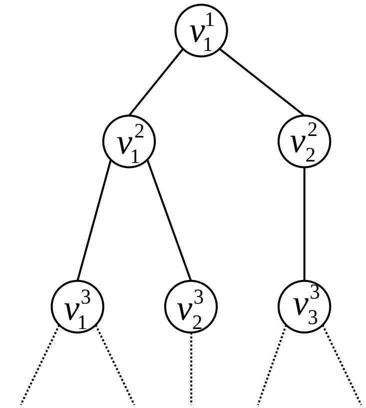
- $\forall \mathbf{y} \neq \mathbf{y}_i$: number of constraints is exponential in T.
- Iterative training: explicit training constraint is added when some $\mathbf{y} \neq \mathbf{y}_i$ violates margin during training.

SVM-Struct: Learning Algorithm

- Minimize
  $\boldsymbol{L}(\boldsymbol{\theta}) = \sum_{i=1}^{n} \xi_i + \lambda \boldsymbol{\theta}^{\mathrm{T}} \boldsymbol{\theta}$ subject to the constraints
  - $\forall \mathbf{y} \neq y_i: f_{\boldsymbol{\theta}}(\mathbf{x}_i, \mathbf{y}_i) \geq f_{\boldsymbol{\theta}}(\mathbf{x}_i, \mathbf{y}) + 1 - \xi_i$
  - $\xi_i \geq 0$

- Start with empty working set of constraints
- Iterate over training instances
  - Find $\arg\max_{\bar{\mathbf{y}} \neq \mathbf{y}_i} f_{\boldsymbol{\theta}}(\mathbf{x}_i, \bar{\mathbf{y}})$
  - While $f_{\boldsymbol{\theta}}(\mathbf{x}_i, \mathbf{y}_i) < f_{\boldsymbol{\theta}}(\mathbf{x}_i, \bar{\mathbf{y}}) + 1 - \xi_i$, add this constraint to the working set of training constraints and solve minimization problem over current working set (e.g., using stochastic gradient descent).

Classification with Class Taxonomies

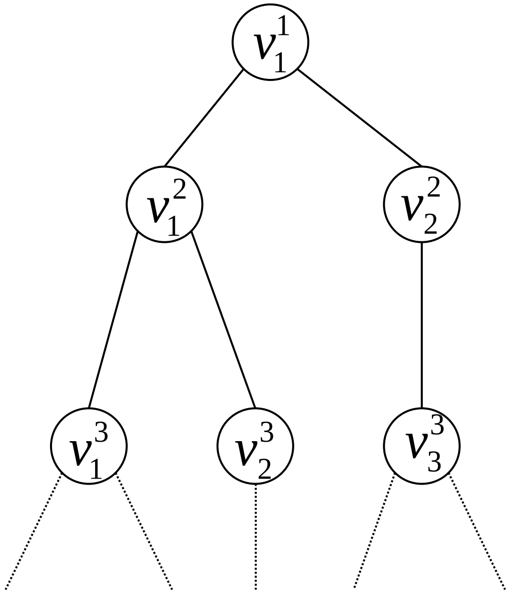


- Taxonomic tree of classes
  - $\hat{\mathbf{y}} = \arg\max_{\mathbf{y}} f_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{y})$
  - $f_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{y}) = \boldsymbol{\theta}^{\mathrm{T}} \Phi(\mathbf{x}, \mathbf{y})$
  - $\mathbf{y} = (y^1, \dots, y^d)$
  - $\Lambda(\mathbf{y}) = \begin{pmatrix} \Lambda(y^1) \\ \vdots \\ \Lambda(y^d) \end{pmatrix}$, where $\Lambda(y^i) = \begin{pmatrix} y_i = v_1 \\ \vdots \\ y_i = v_k \end{pmatrix}$
  - $\Phi(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x}) \otimes \Lambda(\mathbf{y}) = \phi(\mathbf{x}) \otimes \begin{pmatrix} \Lambda(y^1) \\ \vdots \\ \Lambda(y^d) \end{pmatrix} = \phi(\mathbf{x}) \otimes \begin{pmatrix} [\![y^1 = v_1^1]\!] \\ \vdots \\ [\![y^1 = v_{n_1}^1]\!] \\ \vdots \\ [\![y^d = v_1^d]\!] \\ \vdots \\ [\![y^d = v_{n_d}^d]\!] \end{pmatrix}$

Classification with Class Taxonomies

- Let $\mathbf{x}$ be a document
- $\mathbf{y} = \left(v_1^1, v_2^2, v_3^3\right)^T$ is a path in a subject taxonomy tree

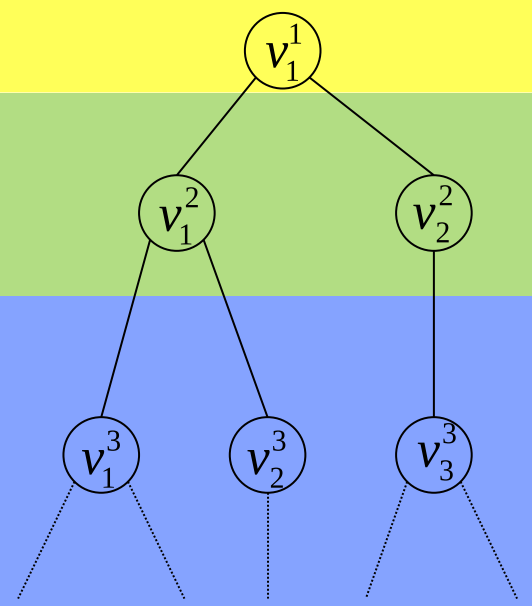
$$\Phi(\mathbf{x},\mathbf{y}) = \begin{pmatrix} [\![y^1 = v_1^1]\!]\mathbf{x} \\ [\![y^2 = v_1^2]\!]\mathbf{x} \\ [\![y^2 = v_2^2]\!]\mathbf{x} \\ [\![y^3 = v_1^3]\!]\mathbf{x} \\ [\![y^3 = v_2^3]\!]\mathbf{x} \\ [\![y^3 = v_3^3]\!]\mathbf{x} \\ \vdots \end{pmatrix} = \begin{pmatrix} \mathbf{x} \\ \mathbf{0} \\ \mathbf{x} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{x} \\ \vdots \end{pmatrix}$$



Classification with Class Taxonomies

- Let $\mathbf{x}$ be a document
- $\mathbf{y} = \left(v_1^1, v_2^2, v_3^3\right)^T$ is a path in a subject taxonomy tree

$$\Phi(\mathbf{x},\mathbf{y}) = \begin{pmatrix} [\![y^1 = v_1^1]\!]\mathbf{x} \\ [\![y^2 = v_1^2]\!]\mathbf{x} \\ [\![y^2 = v_2^2]\!]\mathbf{x} \\ [\![y^3 = v_1^3]\!]\mathbf{x} \\ [\![y^3 = v_2^3]\!]\mathbf{x} \\ [\![y^3 = v_3^3]\!]\mathbf{x} \\ \vdots \end{pmatrix} = \begin{pmatrix} \mathbf{x} \\ \mathbf{0} \\ \mathbf{x} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{x} \\ \vdots \end{pmatrix}$$



Overview

- Document representation for classification.
- Classification methods.
- Multi-class classification and class taxonomies.
- Evaluation of text classifiers.

Evaluating Text Classifiers

- Accuracy (proportion of documents which are classified correctly) is often used for multi-class text classification.
- Downside: rare classes have only small influence.
- Classifier that only recognizes frequent classes can have a high accuracy.

Evaluating Text Classifiers

- Decision function $f_\theta(\mathbf{x})$ returns continuous value.
- Decision rule for binary classification:

$$y_\theta(\mathbf{x}) = \begin{cases} +1 & \text{if } f_\theta(\mathbf{x}) \geq \theta_0 \\ -1 & \text{if } f_\theta(\mathbf{x}) < \theta_0 \end{cases}$$

- By adjusting threshold θ_0 decision rule can be made more sensitive or more conservative.
- Decision function for each category can be evaluated separately.

Precision and Recall

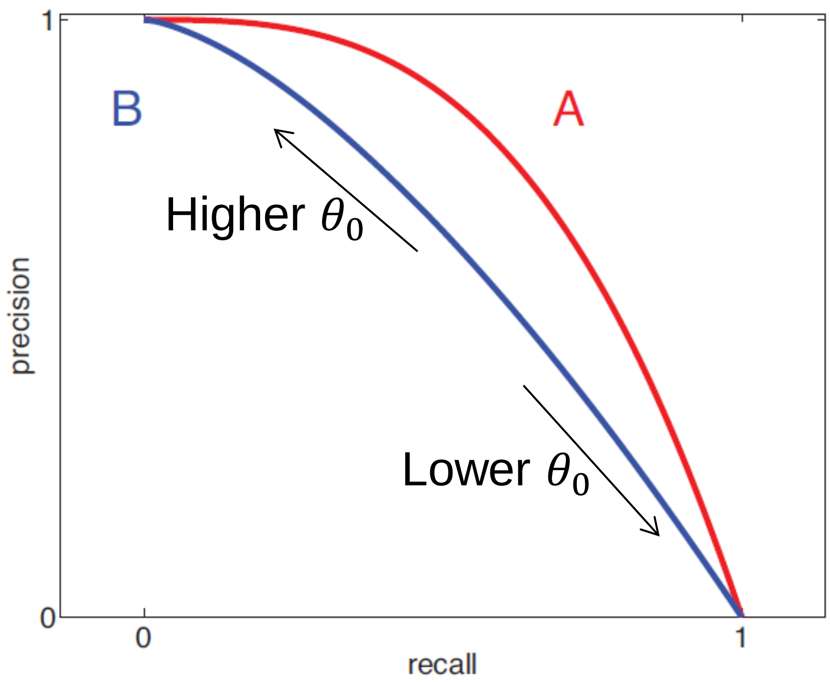
- Performance measure for binary classification.
  - Example: class *invoice* against all other classes.
  - Dociment $\mathbf{x}_i$ is in category if $y_i = +1$.
  - Classifier recognizes category if $y_\theta(\mathbf{x}_i) = +1$.
- True positives:
  - Document in category ($y_i = +1$), classifier recognizes ($y_\theta(\mathbf{x}_i) = +1$)
- False positives:
  - Document not in catrgory ($y_i = -1$), but classifier thinks it is ($y_\theta(\mathbf{x}_i) = +1$).
- True negatives:
  - Patient is healthy Document not in catrgory ($y_i = -1$), classifier recognizes ($y_\theta(\mathbf{x}_i) = -1$)
- False negatives:
  - Document in category ($y_i = +1$), classifier misses ($y_\theta(\mathbf{x}_i) = -1$)

Precision and Recall

- Let n_{TP} be the number of true positives.
- Let n_{FP} be the number of false positives.
- Let n_{TN} be the number of true negatives.
- Let n_{FN} be the number of false negatives.
- Precision: $P = \frac{n_{TP}}{n_{TP}+n_{FP}}$
  - "Rate of true positives among all instances that are classified as positives"
  - Answers: "How accurate is classifier when it says +1?"
- Recall: $R = \frac{n_{TP}}{n_{TP}+n_{FN}}$
  - "Rate of true positives among all positive instances"
  - Answers: "How many of the positive instances does the classifier detect?"

Precision-Recall Curves

- Evaluates decision function $f_\theta(\mathbf{x})$ independent of threshold θ_0.
- Shows which pairs of precision and recall can be obtained by varying threshold θ_0.
- Each point on the curve is a classification rule with a particular values of θ_0.
- Which decision function is better – A or B?



F Measures

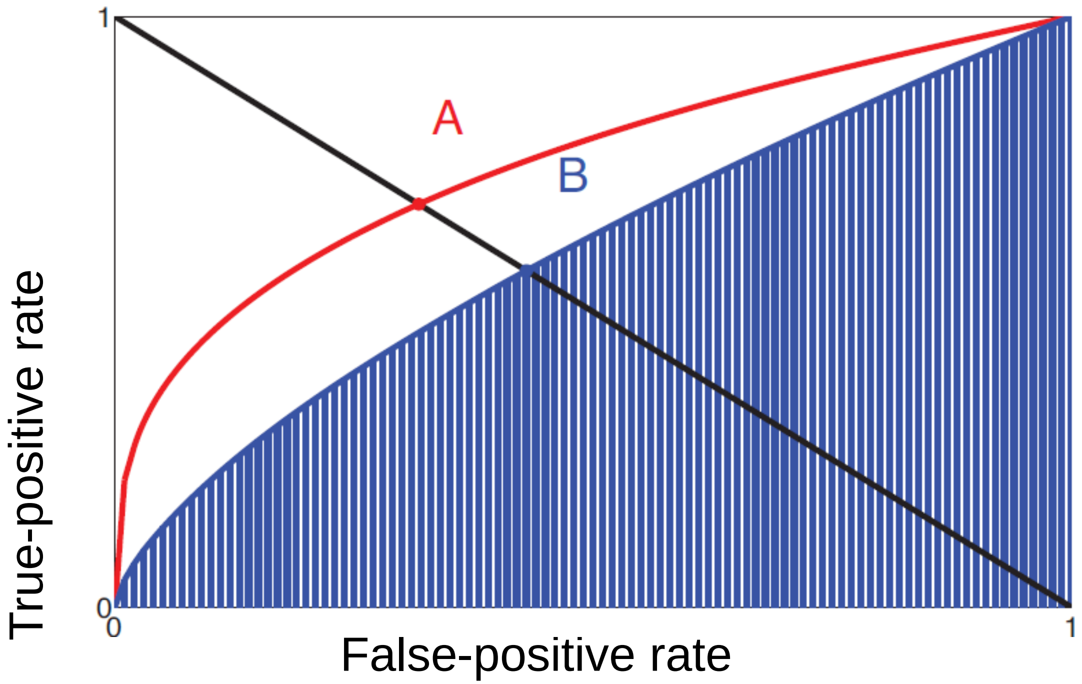
- F_α measures combine precision and recall values into single value:

$$F_\alpha = \frac{n_{TP}}{\alpha(n_{TP} + n_{FP}) + (1-\alpha)(n_{TP} + n_{FN})}$$

- $\alpha = 1$: Precision
- $\alpha = 0$: Recall
- $\alpha = 0.5$: “F-measure”, harmonic mean of precision and recall.
- Alternative definition: F_β measures.
  - Relationship: $\alpha = \frac{1}{1+\beta}$

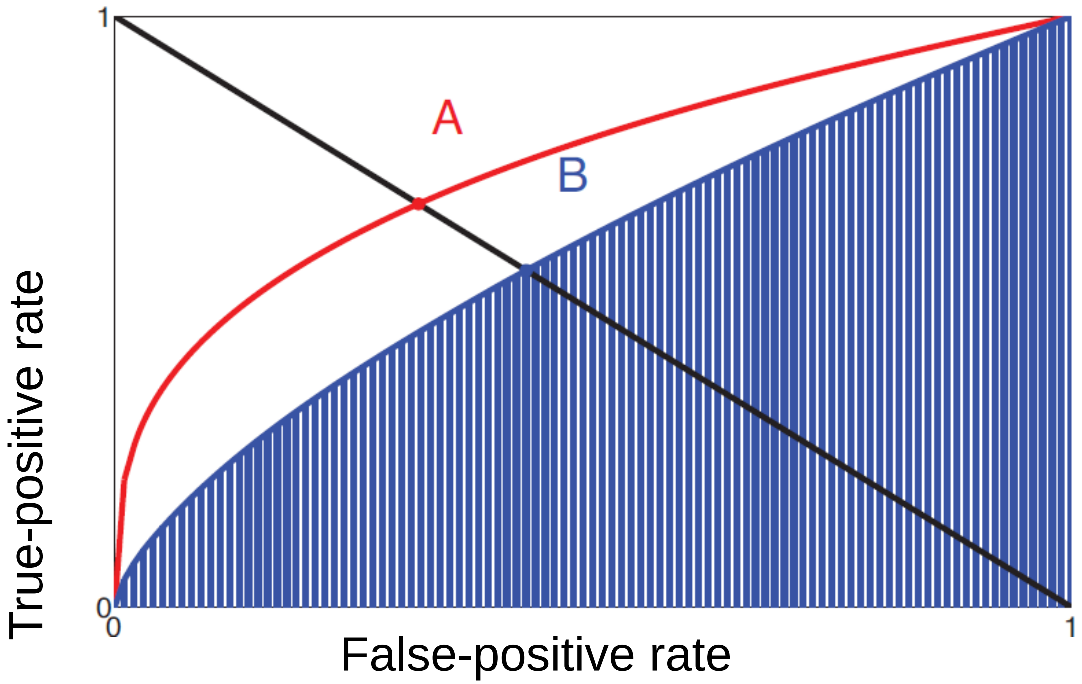
ROC Analysis

- Alternative measure of how well the decision function separates positive from negative instances, independent of any threshold value θ_0.



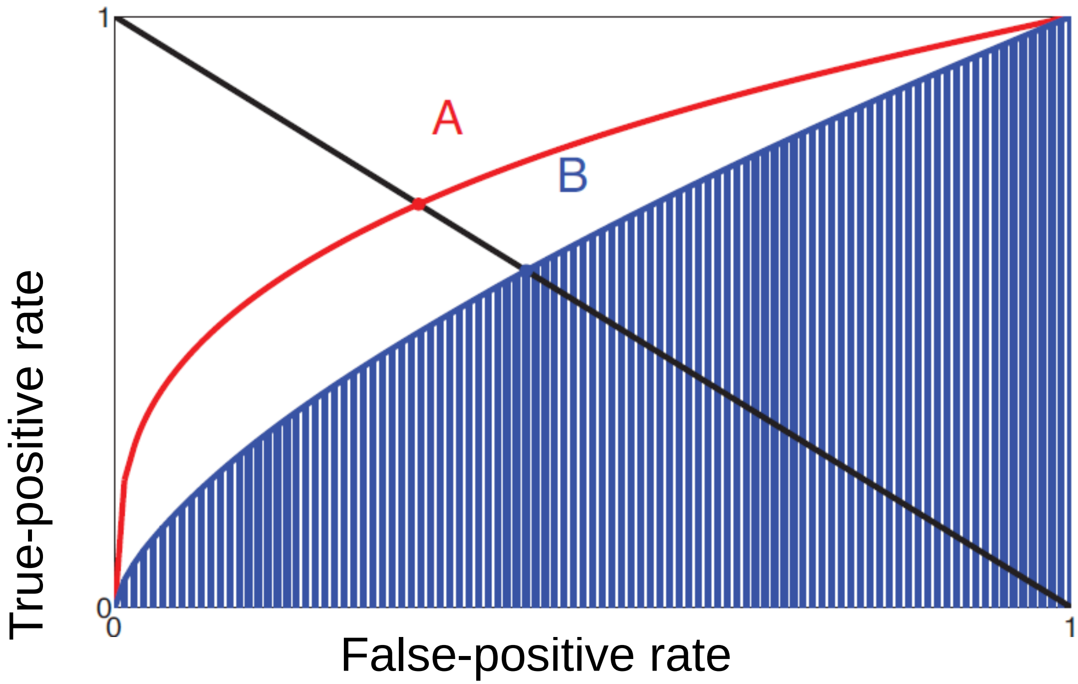
ROC Analysis

- Each curve characterizes a decision function f_θ.
- Each point is a classification rule for a value of θ_0.
- Which is better, A or B?
- $r_{TP} = \frac{n_{TP}}{n_{TP}+n_{FN}}$
- $r_{FP} = \frac{n_{FP}}{n_{FP}+n_{TN}}$



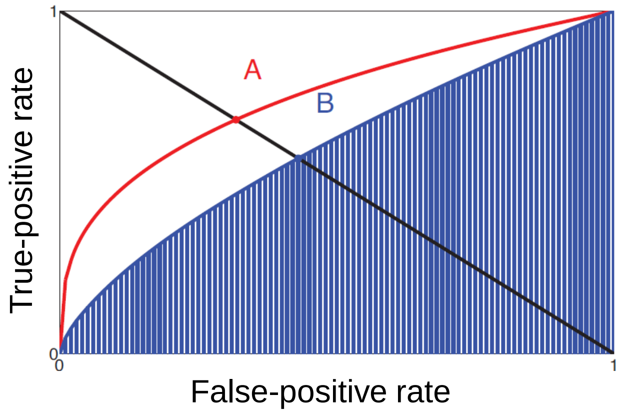
ROC Analysis

- Equal error rate (EER): value $r_{TP} = 1 - r_{FP}$.
- Scalar aggregate of curve: Area under ROC curve (AUC).



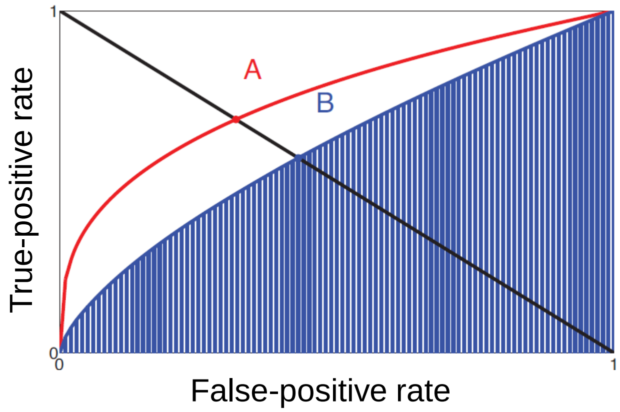
ROC Analysis

- Area under the ROC curve (AUC):
  - Let $\mathbf{x}_+$ be a randomly drawn positive instance.
  - Let $\mathbf{x}_-$ be a randomly drawn negative instance.
  - $AUC(\theta) = P(f_\theta(\mathbf{x}_+) > f_\theta(\mathbf{x}_-))$.



ROC Analysis

- ROC analysis is often used
  - When positive instances are rare (accuracy of 99.9% is meaningless if positive class is extremely rare)
  - When no meaningful probability of meeting positive instances can be defined (prior probability of news categories changes every day based on events).



Drawing ROC Curves

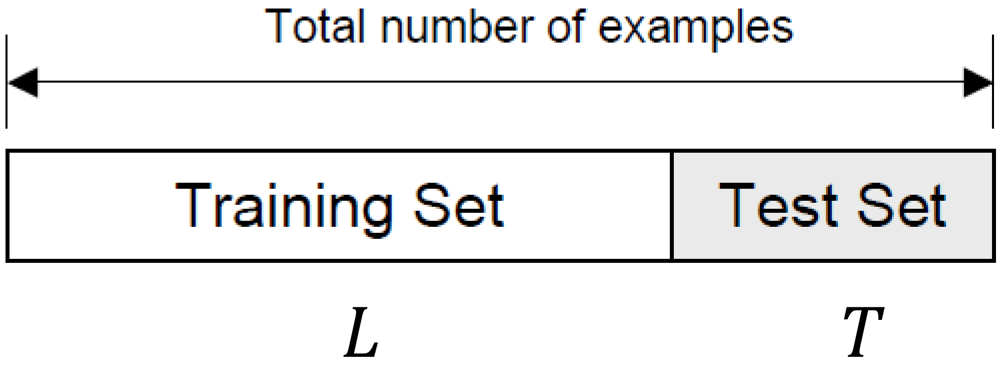
- For all positive examples X_p in test set:
  - Insert $f(x_p)$ in decreasing order in ordered list L_p.
- For all negative examples X_n in test set:
  - Insert $f(x_n)$ in decreasing order in ordered list L_n.
- Let TP = FP = 0.
- Repeat as long as L_p and L_n are not empty:
  - If $L_p \rightarrow$ element > $L_n \rightarrow$ element, then increment(TP) and $L_p = L_p \rightarrow$ Next.
  - Elsif $L_n \rightarrow$ element < $L_p \rightarrow$ element, then increment(FP) and $L_n = L_n \rightarrow$ Next.
  - Else increment(TP, FP), $L_p = L_p \rightarrow$ Next, $L_n = L_n \rightarrow$ Next.
  - Plot next point (FP, TP).

Evaluation Protocols

- Usually, model f_θ is not given and evaluation data cannot be drawn from $p(\mathbf{x}, y)$.
- Typical case, data $S = (\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)$ and learning method are given.
- Data S have to be used for training and evaluation.
- Desired output: model f_θ and risk estimate.
- Cannot evaluate on training data because performance on training data is always high (higher than on unseen test data).

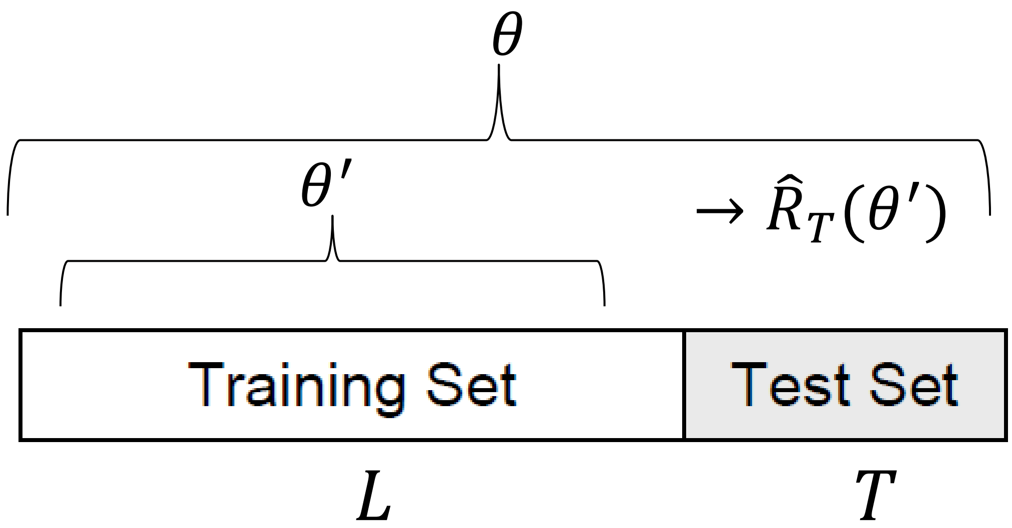
Holdout Testing

- Idea: error estimation on independent test data
- Given: data $S = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$
- Divide the data into
  - Training data $L = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$and
  - Test data $T = (\mathbf{x}_{m+1}, y_{m+1}), \ldots, (\mathbf{x}_n, y_n)$



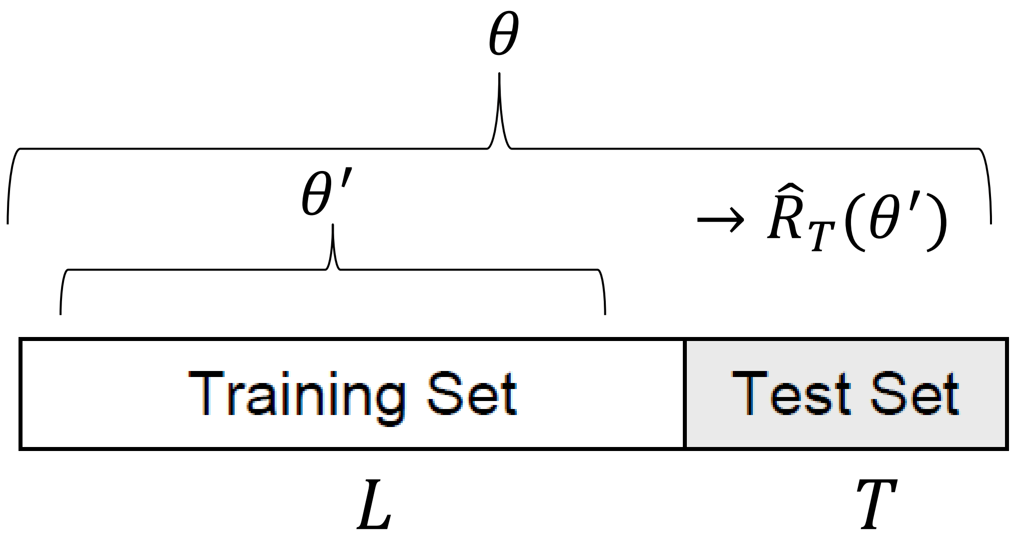
Holdout Testing

- Start learning algorithm with data L and obtain model $f_{\theta'}$ from it.
- Determine performance $\hat{R}_T(\theta')$ on data T.
- Start learning algorithm with all data S and obtain Model f_θ from it.
- Output: model f_θ & $\hat{R}_T(\theta')$ as the estimator of $R(\theta)$.



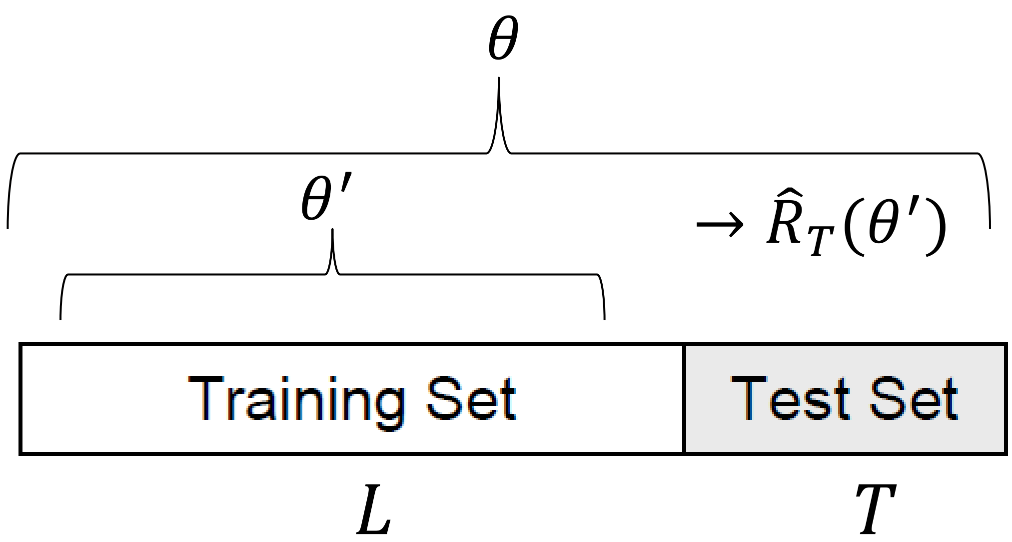
Holdout Testing: Analysis

- Is the estimator $\hat{R}_T(\theta')$ of the risk of model $R(\theta)$
  - unbiased,
  - optimistic,
  - pessimistic?
- Hint: the more training data, the better the model.



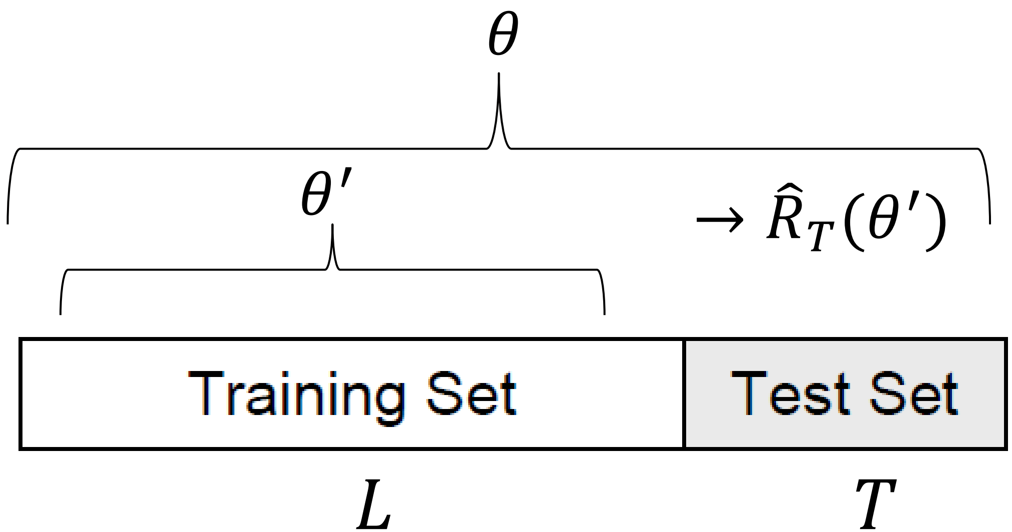
Holdout Testing: Analysis

- Estimate $\hat{R}_T(\theta')$ is obtained on a small part of the available data.
- Therefore, its variance is relatively high, especially if the overall sample is small.
- Holdout testing is used in practice for large available samples.



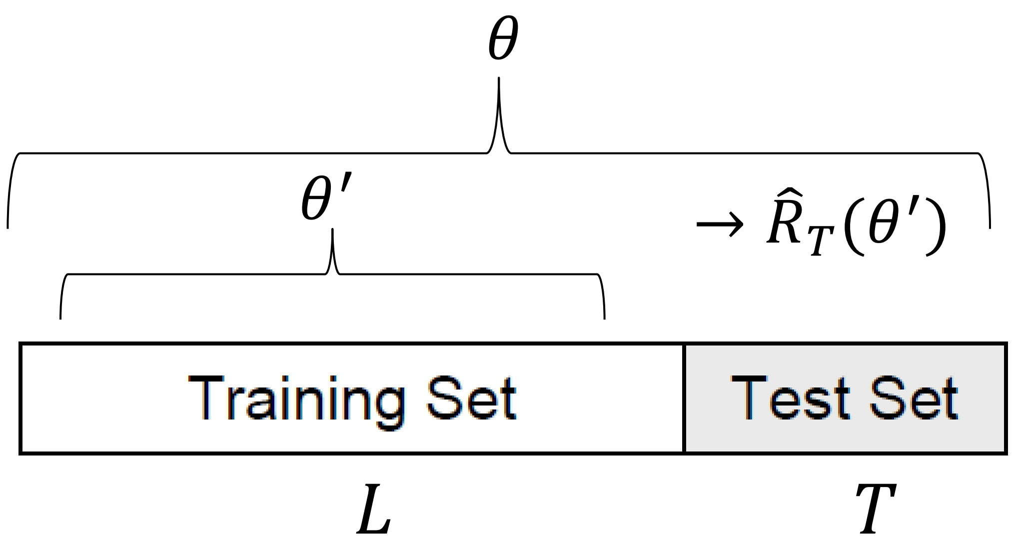
Holdout Testing: Analysis

- Using empirical risk $\hat{R}_T(\theta')$ is an **pessimistic** estimator of the risk $R(\theta)$.
- Because θ' is trained with fewer training instances than θ.



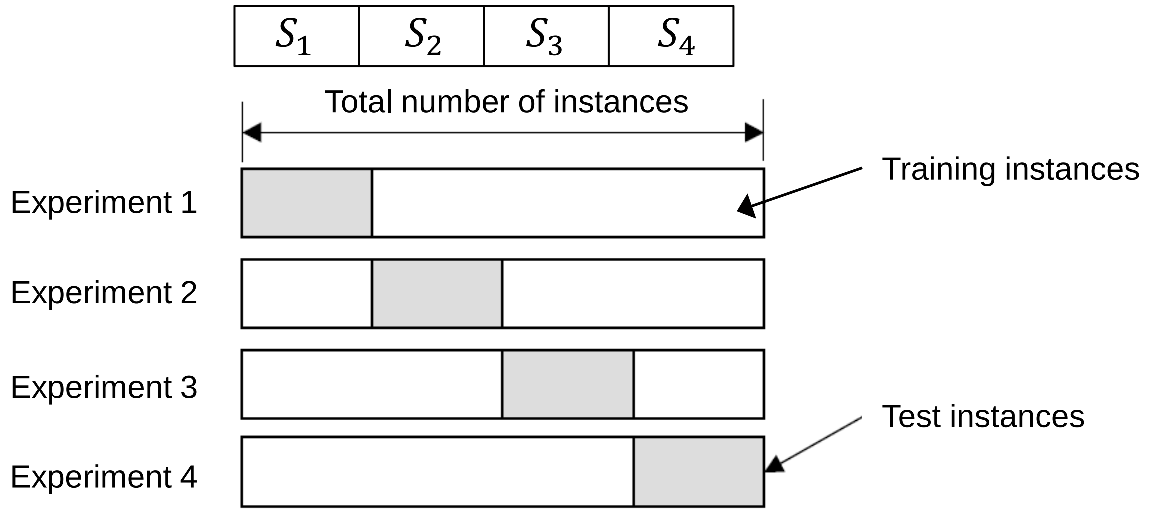
Holdout Testing: Analysis

- One could instead return model θ'.
- Empirical risk $\hat{R}_T(\theta')$ would be an unbiased estimate of $R(\theta')$.
- But since θ' was trained on fewer data, it would result in an inferior model.



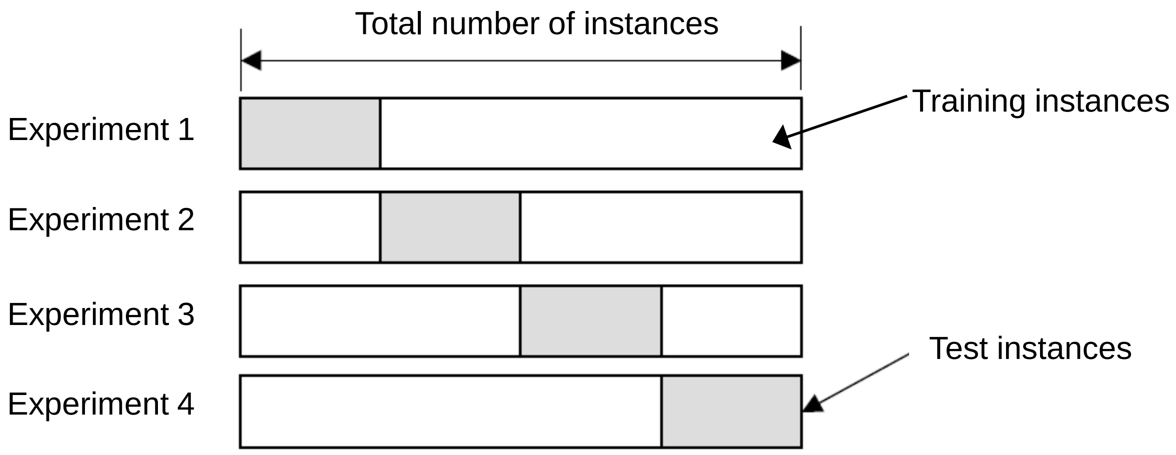
K-Fold Cross Validation

- Given: data $S = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$
- Partition S into k equally sized portions $S_1, \ldots, S_k$.
- Repeat for $i = 1 \ldots k$
  - Train f_{θ_i} with training set $S = S \backslash S_i$.
  - Calculate empirical risk $\hat{R}_{S_i}(\theta_i)$ on S_i.
- Calculate average $\hat{R}_S = \frac{1}{k} \sum_i \hat{R}_{S_i}(\theta_i)$



Cross Validation

- Then, train f_θ on all data S.
- Return model f_θ and estimator $\hat{R}_S$.



Summary

- Document representation for classification:
  - Bag-of-words, TF-IDF,
  - Neural language models.
- Classification methods:
  - Linear models, regularized empirical risk minimization,
  - Logiscic regression
- Multi-class classification and class taxonomies.
- Evaluation of text classifiers:
  - Precision-recall, ROC curves,
  - Hold-out testing, cross validation.