UNIVERSITY OF TORONTO SOCIAL SCIENCE METHODS WEEK

An Introduction to Machine Learning for Textual Analysis

Contents

UofT Social Science Methods Week

Machine Learning: The Intuition

UNIVERSITY OF TORONTO

Social Science vs. Machine Learning

In the social sciences, we are usually interested in statistical inference. We fit statistical models like

$$y = \mathbf{x}'\beta + \varepsilon$$

and the goal is to test hypotheses about β in the population, using the estimates $\widehat{\beta}$ and the covariance matrix $\widehat{\mathbf{V}}$ (standard errors).

Social Science vs. Machine Learning

In machine learning, the main objective is prediction. We want to find the most accurate model to predict y.

$$\widehat{y} = \mathbf{x}'\widehat{\beta}$$

The Good News

If you've ever predicted values from a fitted regression model, you get the intuition of machine learning, e.g.

- You fitted the model

$$\widehat{y} = 2.5x$$

- For $x = 4$, your prediction is

$$\widehat{y} = 2.5 \times 4 = 10$$

Your model has "learned" to predict y just by observing x.

Categorical Outcomes

The same logic extends to categorical outcomes, for instance

$$\widehat{P}(y = i) = \frac{exp(\mathbf{x}'\widehat{\beta}_i)}{\sum_{k=1}^{K} exp(\mathbf{x}'\widehat{\beta}_k)}$$

and the predicted category is the one with largest $\widehat{P}(y = i)$.

In machine learning, we often call models with categorical outcomes **classifiers**.

UofT Social Science Methods Week

Working with Textual Data

UNIVERSITY OF TORONTO

Textual Data

A common application in the social sciences is to use machine learning for predicting latent classes in text documents. The class could be:

- Identity of the author;
- Gender of the author;
- Psychological characteristics of the author;
- Occupation of the author;
- Topic of a text;
- Sentiment of a text;
- Ideological slant of a text;
- Degree of incivility of a text;
- ...

Textual Data as Numerical Data

The text itself can be transformed into an array of independent variables **x**, called **features**, and the dependent variable *y* is the attribute we try to predict.

- Each **document** is one unit of analysis.
- Each **term** can be encoded as a binary variable (1 if word *x* appears in the document, 0 otherwise), as a count variable, as a sequence.

Term-Document Matrix (TDM)

- The following three documents:

Document	Text
1	“the economy is in crisis”
2	“the economy is in bad shape”
3	“the economy is recovering”

- Can be encoded as a design matrix $\mathbf{X}$ called a term-document matrix (TDM):

Document	bad	crisis	economy	in	is	recovering	shape	the
1	0	1	1	1	1	0	0	1
2	1	0	1	1	1	0	1	1
3	0	0	1	0	1	1	0	1

Term-Document Matrix (TDM) and Classes

- Suppose the following dataset:

Document	Text	Class
1	“the economy is in crisis”	Negative
2	“the economy is in bad shape”	Negative
3	“the economy is recovering”	Positive
...	...	...

- We can fit the model $\mathbf{y} = f(\mathbf{X}\beta)$ as

$$\begin{bmatrix} 0 \\ 0 \\ 1 \\ \vdots \end{bmatrix} = f\left(\begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & \cdots \\ 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & \cdots \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \\ \vdots \end{bmatrix} \right)$$

Predicting the Class of New Documents

- Suppose a dataset of unseen documents:

Document	Text	Class
1	“Canada is facing a crisis”	NA
2	“the plan is taking shape”	NA

- We need to encode the text in the same vector space

Document	bad	crisis	economy	in	is	recovering	shape	the
1	0	1	0	0	1	0	0	0
2	0	0	0	0	1	0	1	1

- Then we predict $\widehat{\mathbf{y}} = f(\mathbf{X}\widehat{\beta})$.

Term-Document Matrix (TDM) and Classes

Document	Text	Class
1	"the economy is in **crisis**"	Negative
2	"the economy is in **bad shape**"	Negative
3	"the economy is **recovering**"	Positive
...	...	...

- Our model may have learned to associate the words **crisis** and **bad** with negative sentiment.
- Our model may have incorrectly learned to associate **shape** with negative sentiment.

Other Things to Consider

- In practice, we improve the accuracy of a classifier in a number of ways, for instance:
  1. Preprocessing the text (removing “stop words”, lemmatization).
  2. Weighted counts (giving more importance to infrequent words).
  3. Regularization (most algorithms use regularization by default).
  4. Feature selection (restrict model to variables that are good predictors).
  5. Using n-grams (considering sequences of words).
  6. Combining features of different types (metadata, other linguistic properties).

1. Text Preprocessing

- The same three documents:

Document	Text	Preprocessed Text
1	“the economy is in crisis”	“economy crisis”
2	“the economy is in bad shape”	“economy bad shape”
3	“the economy is recovering”	“economy recover”

- Can be encoded as preprocessed text by removing stop words and using lemmas:

Document	bad	crisis	economy	recover	shape
1	0	1	1	0	0
2	1	0	1	0	1
3	0	0	1	1	0

- We reduce model variance.

2. Term Frequency–Inverse Document Frequency

A common way to give more importance to infrequent words. Let:

- tf_{id} = count of term i in document d;
- df_i = number of documents that contain the term i;
- N = sample size;
- $idf_i = \ln\left(\frac{N}{df_i}\right)$ or some variant thereof.

The term frequency-inverse document frequency (tf-idf) is:

$$tf_{id} \times idf_i$$

3. Regularization

Regularization techniques such as L1 (Lasso) and L2 (Ridge) penalties are virtually always used by default in machine learning.

Example of optimization problem for a L2 penalized linear regression:

$$(\widehat{\alpha}, \widehat{\boldsymbol{\beta}}) = \underset{\alpha,\boldsymbol{\beta}}{\arg\min} \sum_{i=1}^{n} (y_i - \alpha - \boldsymbol{x}_i'\boldsymbol{\beta})^2 + \lambda \sum_{\beta \in \boldsymbol{\beta}} \beta^2$$

4. Feature Selection

- The same three documents:

Document	Text	Preprocessed Text
1	“the economy is in crisis”	“economy crisis”
2	“the economy is in bad shape”	“economy bad shape”
3	“the economy is recovering”	“economy recover”

- Can be shrunken by removing poor predictors, e.g. shape, economy:

Document	bad	crisis	recover
1	0	1	0
2	1	0	0
3	0	0	1

- We again reduce model variance.

5. N-Grams

- The same three documents:

Document	Text	Preprocessed Text
1	“the economy is in crisis”	“economy crisis”
2	“the economy is in bad shape”	“economy bad shape”
3	“the economy is recovering”	“economy recover”

- We can include bi-grams (sequences of two words) and n-grams as features. Here’s an abbreviated example.

Document	bad	crisis	recover	bad shape	economy crisis
1	0	1	0	0	1
2	1	0	0	1	0
3	0	0	1	0	0

- This may increase variance, but we can restrict the features to the best n-gram predictors.

6. Combining Features

- The same three documents:

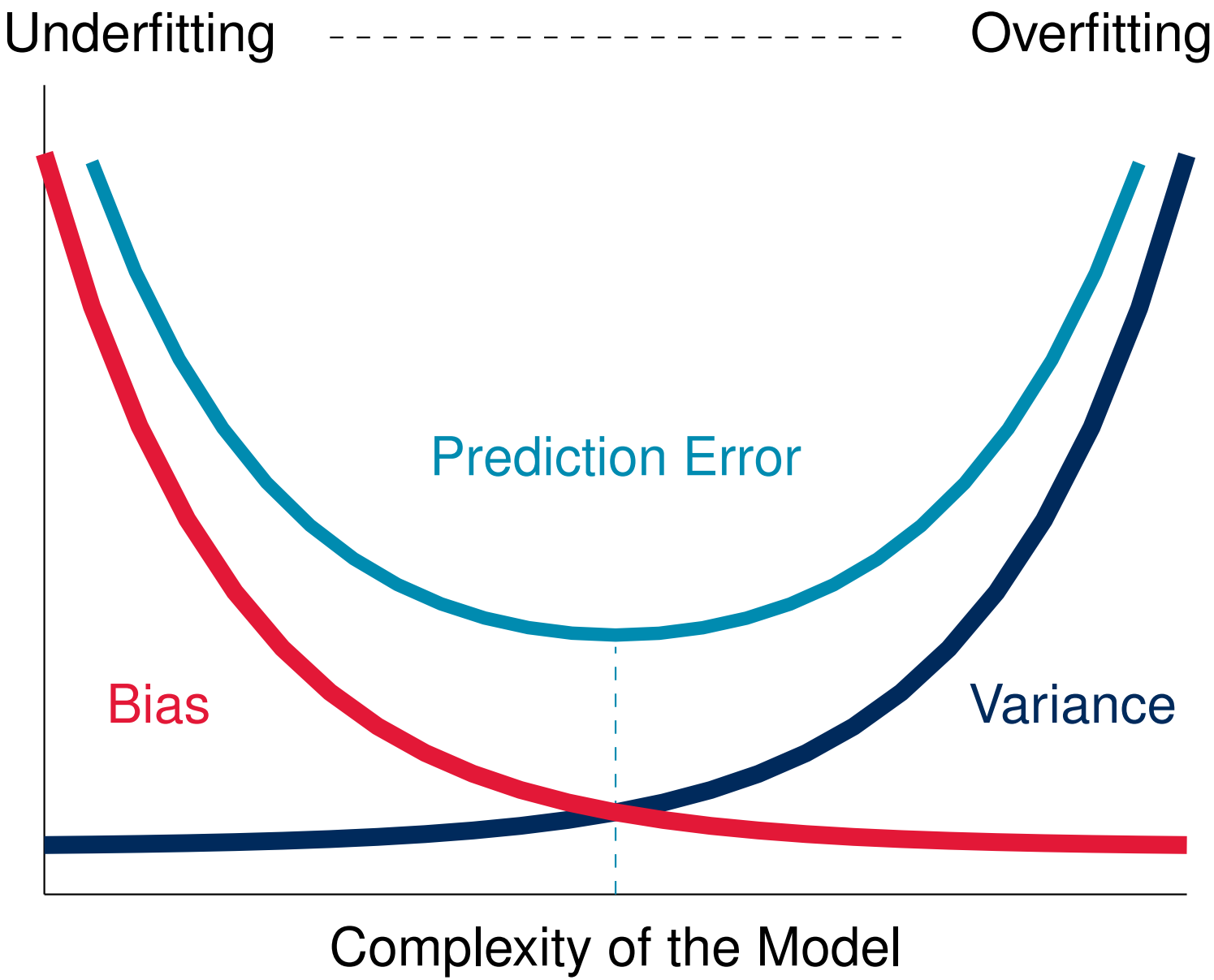
Document	Text	Preprocessed Text	Source
1	“the economy is in crisis”	“economy crisis”	Social Media
2	“the economy is in bad shape”	“economy bad shape”	Blog
3	“the economy is recovering”	“economy recover”	Social Media

- Combining features is just appending columns to the design matrix.

Document	bad	crisis	recover	social_media	word_count
1	0	1	0	1	2
2	1	0	0	0	3
3	0	0	1	1	2

- Theory may help to determine which features are relevant. We want these features to be observed in the new documents to which we plan to apply the model.

Bias-Variance Trade-Off



UofT Social Science Methods Week

Machine Learning Classifiers

UNIVERSITY OF TORONTO

Popular ML Models for Textual Data

Logistic Regression

Bernoulli Naive Bayes

Multinomial Naive Bayes

Decision Trees

K-Nearest Neighbours (k-NN)

Support Vector Machines (SVM)

Multi-Layer Perceptrons

Convolutional Neural Networks (CNN)

Recurrent Neural Networks (RNN)

Example: Support Vector Machines

Support Vector Machines

Support vector machines (SVMs) have become one of the most popular models for textual data.

- Model dates from the 1960s.
- Popularized by Cortes and Vapnik ("Support-Vector Networks", *Machine Learning*, 1995)
- To introduce SVMs, we need to cover the concept of **hyperplane** and the **maximum margin classifier**.

Hyperplane

Hyperplane

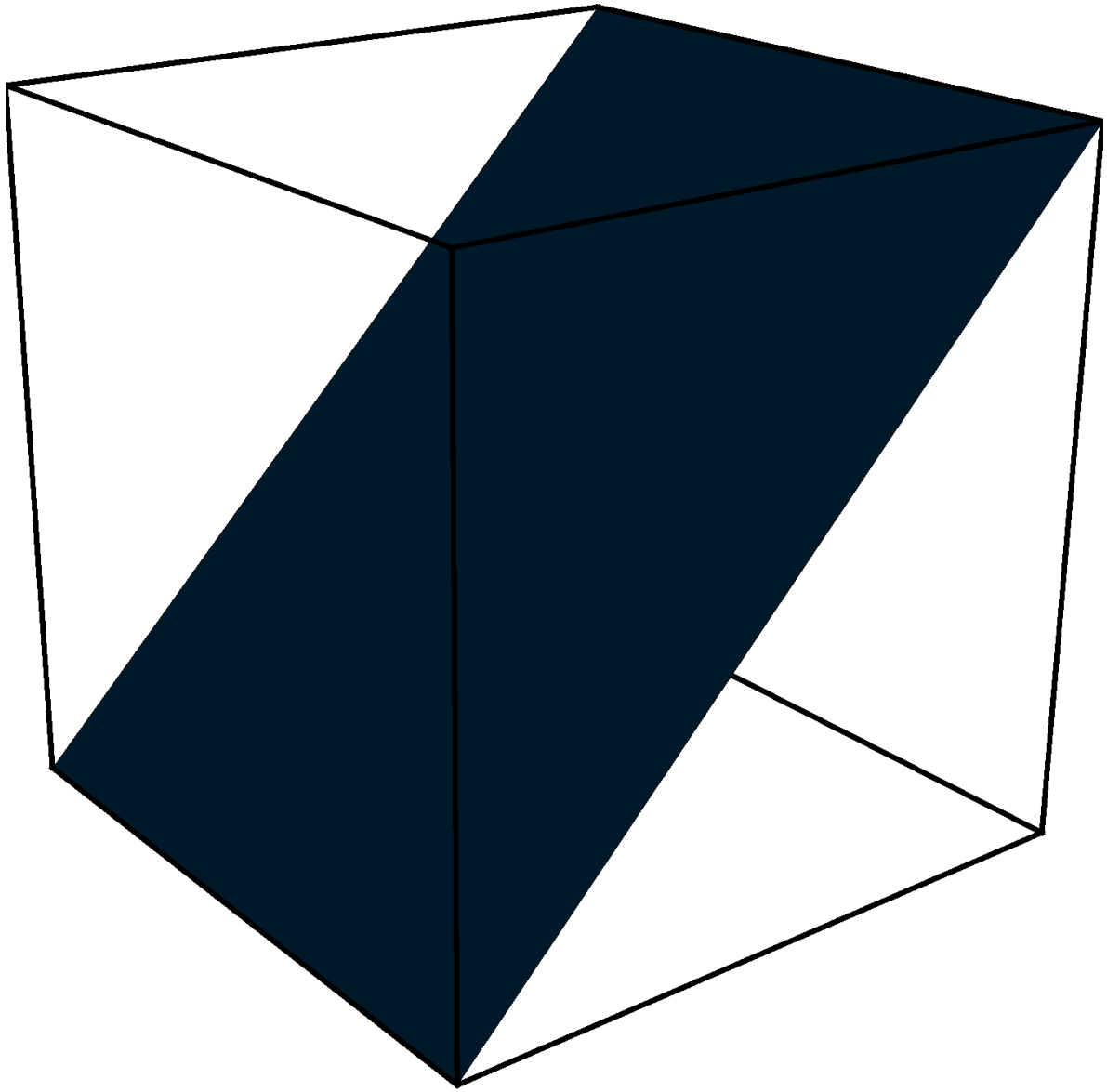
For a p-dimensional space, a hyperplane is a flat subspace with $p - 1$ dimensions. Hyperplanes can be expressed as equations of the type:

$$\alpha + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p = 0$$

A vector $\boldsymbol{x} = (x_1, x_2, ..., x_p)$ that respects this equation lies on the hyperplane.

- For a 2D space, a hyperplane is a line.
- For a 3D space, a hyperplane is a 2D plane.
- ...

Hyperplane



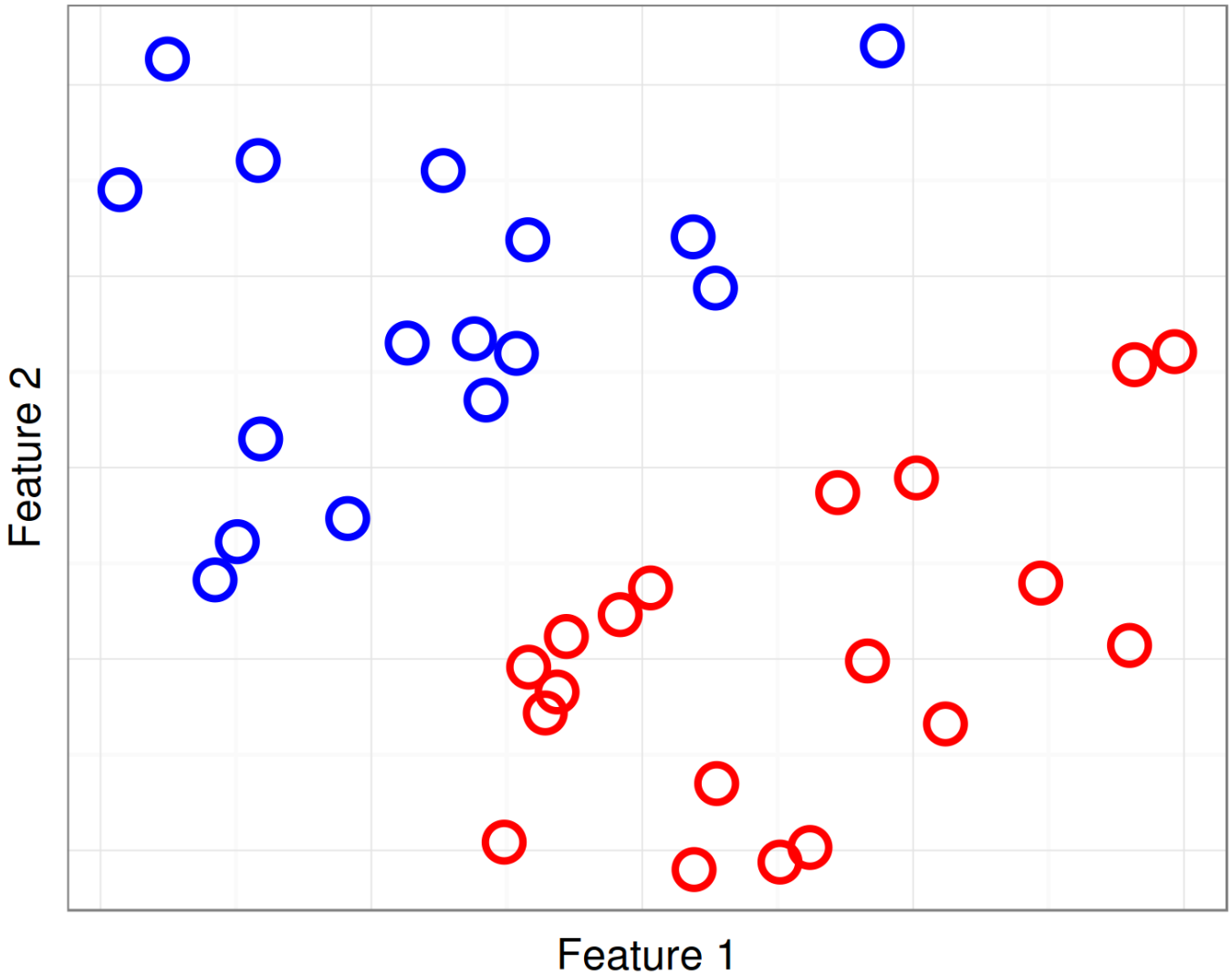
Separating Hyperplane

A hyperplane works as a useful boundary to classify documents. For instance, we could attribute a class to documents above the hyperplane, and another class for documents below:

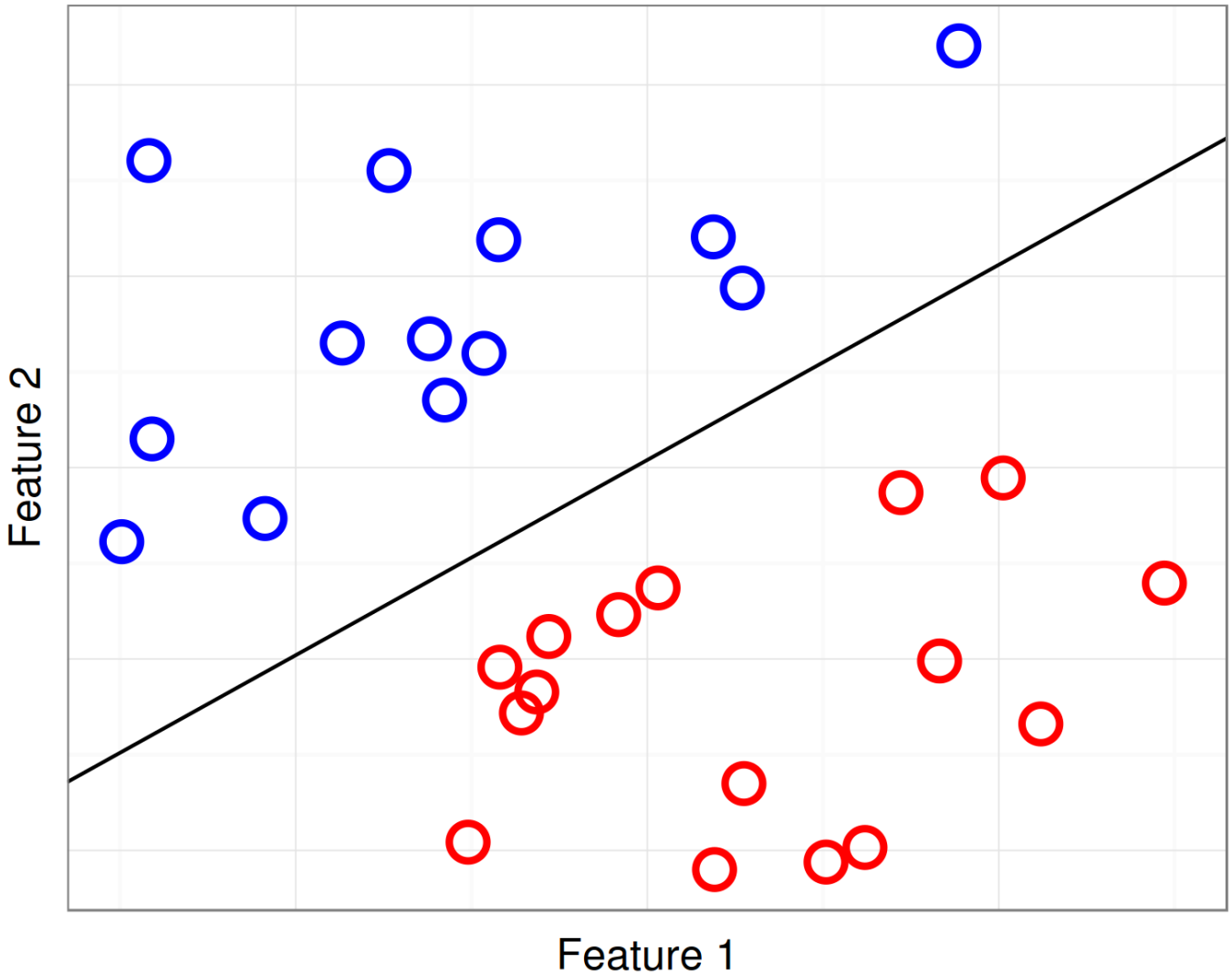
$$y = \begin{cases} 1 & \text{if } \alpha + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p \geq 0 \\ -1 & \text{if } \alpha + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p < 0 \end{cases}$$

We call this a separating hyperplane.

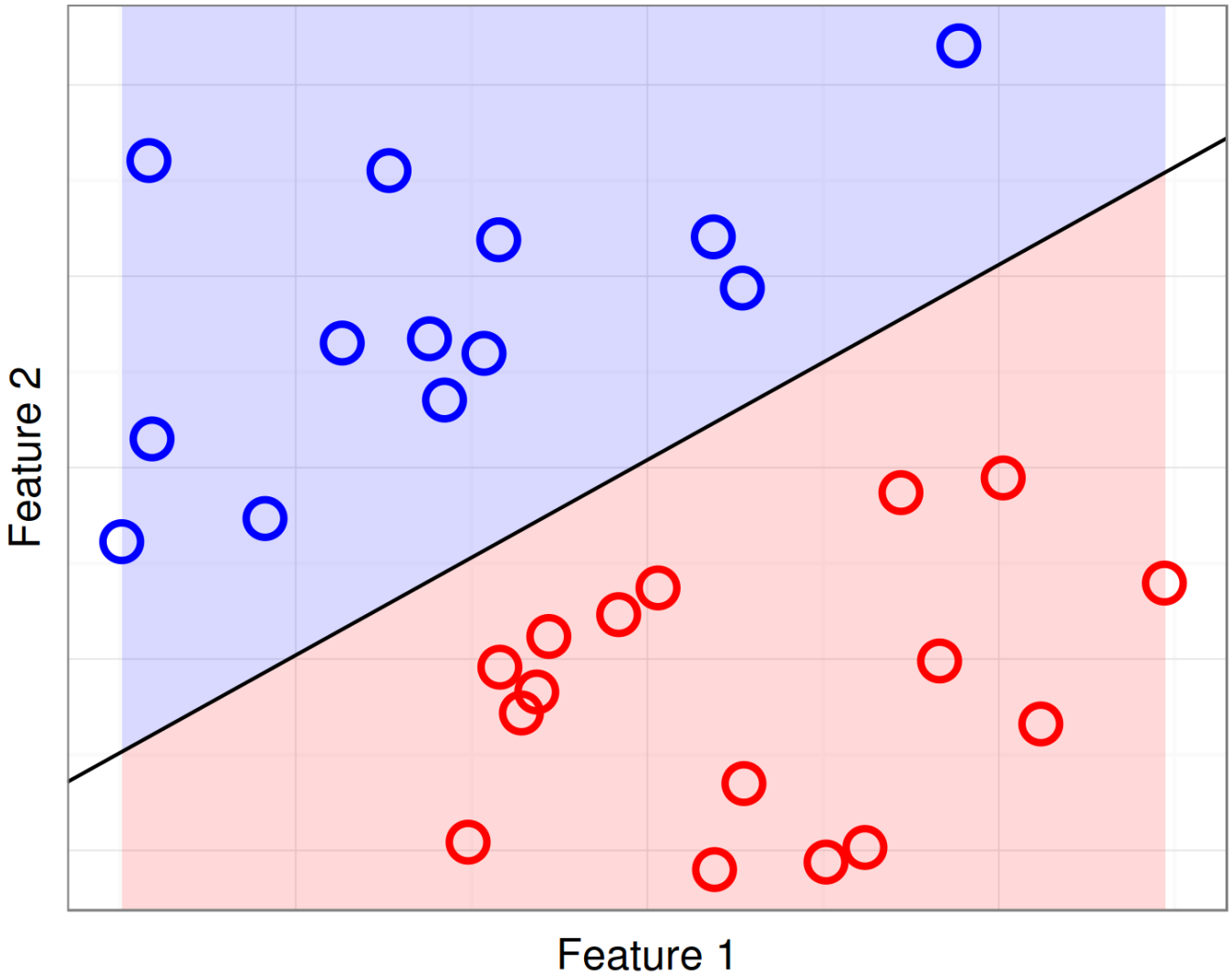
Separating Hyperplane



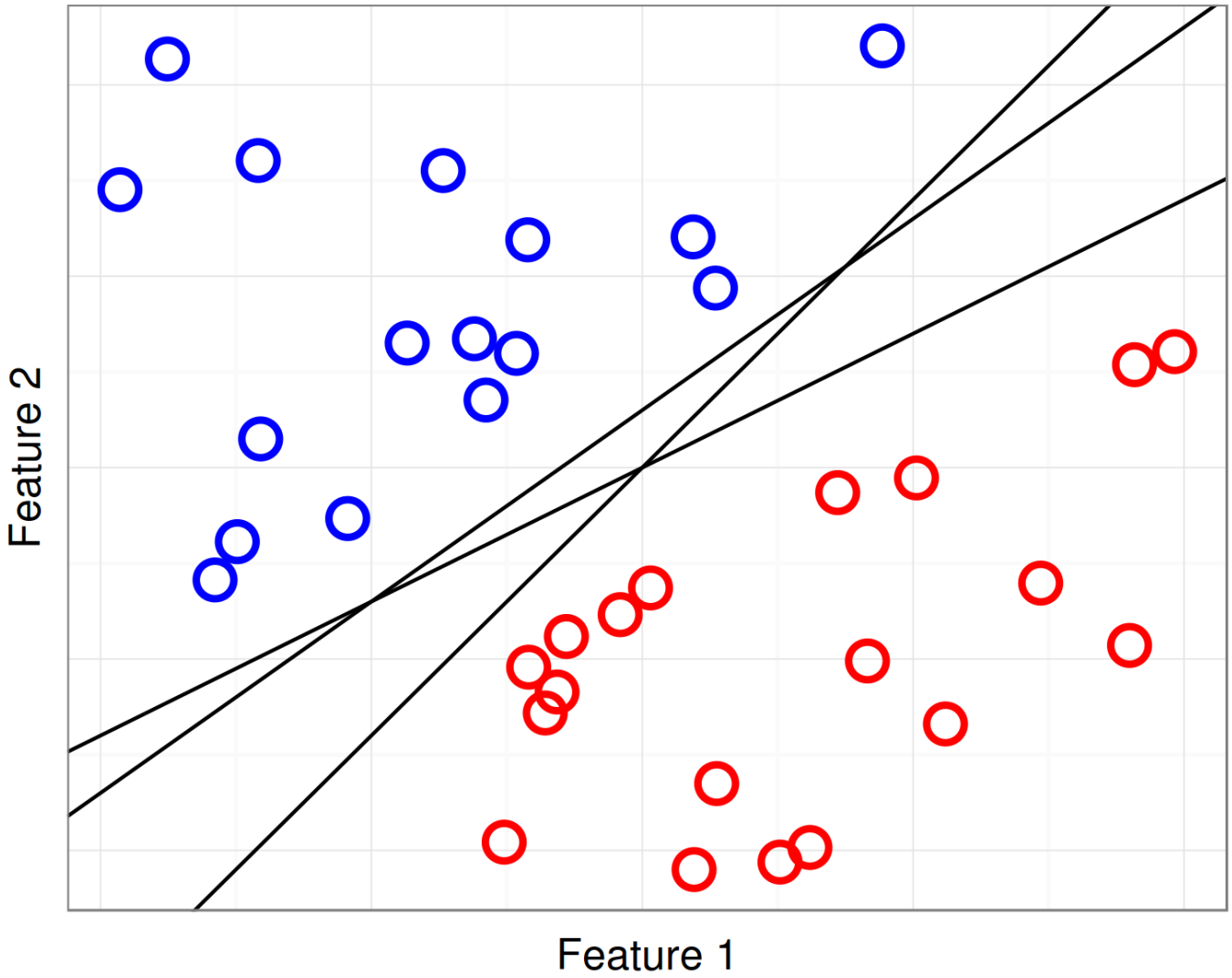
Separating Hyperplane



Separating Hyperplane



What if Multiple Separating Hyperplanes?



Maximum Margin Classifier

Maximum (or Maximal) Margin Classifier

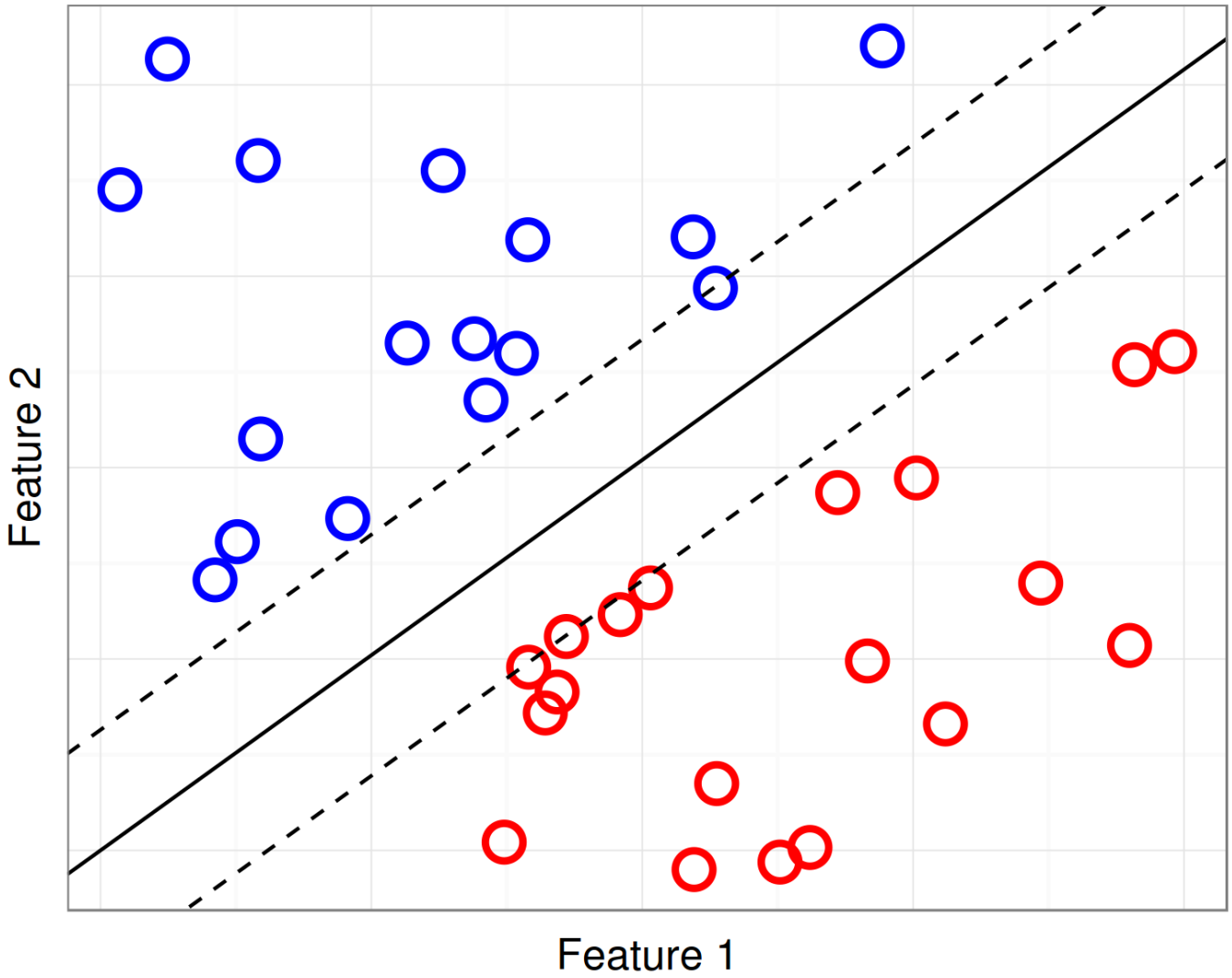
The **maximum margin classifier** uses the separating hyperplane that maximizes the margin between observations with binary classes $y \in \{-1, 1\}$. The optimization problem is:

$$\underset{\alpha,\boldsymbol{\beta}}{\arg\max}\, M$$

such that

$$\sum_{\beta \in \boldsymbol{\beta}} \beta^2 = 1; \quad y_i(\alpha + \boldsymbol{x}_i'\boldsymbol{\beta}) \geq M \ \forall \ i = 1, ..., n$$

Maximum Margin Classifier



Maximum Margin Classifier: Decision Rule

Maximum Margin Classifier: Decision Rule

Once we have estimates of the hyperplane with the maximum margin, we classify new documents based on whether they fall above or below the hyperplane:

$$\widehat{y} = \begin{cases} 1 & \text{if } (\widehat{\alpha} + \boldsymbol{x}'\widehat{\boldsymbol{\beta}}) \geq 0 \\ -1 & \text{if } (\widehat{\alpha} + \boldsymbol{x}'\widehat{\boldsymbol{\beta}}) < 0 \end{cases}$$

Maximum Margin Classifier and Support Vectors

- We call the points on the margin boundaries (dashed lines in figure above) the **support vectors**.

Limitation of Maximum Margin Classifier

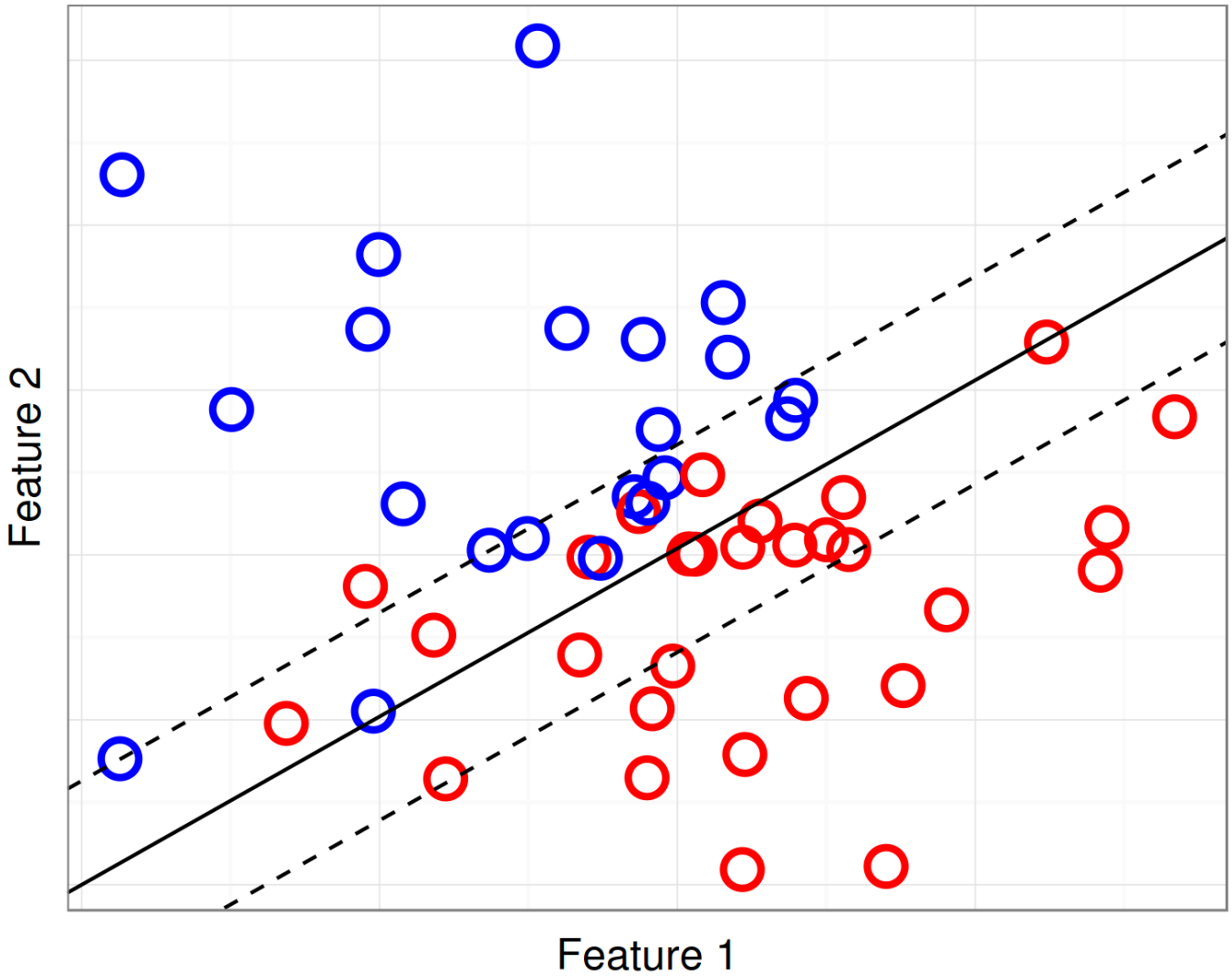
We require that all observations fall on either side of the hyperplane (linearly separable).

Support Vector Classifiers

Support Vector Classifiers

Support vector classifiers are a generalization of the maximum margin classifier, allowing for *soft margins* (constraints). We allow some observations to be on the wrong side of the hyperplane.

Soft Margin (Support Vector) Classifier



Support Vector Classifier

The optimization problem can be modified to include an error around the margin

$$\arg\max_{\alpha,\boldsymbol{\beta}} M$$

such that

$$\sum_{\beta\in\boldsymbol{\beta}} \beta^2 = 1; \quad y_i(\alpha + \boldsymbol{x}_i'\boldsymbol{\beta}) \geq M - \varepsilon_i;$$

$$\varepsilon_i \geq 0; \quad \sum_{i=1}^{n} \varepsilon_i \leq C$$

Support Vector Machines

More generally, **support vector machines** (SVMs) are estimated with a **kernel**, or enlarged feature space, to allow for non-linear decision boundaries.

- The SVM kernel is a transformation of the hyperplane equation sometimes denoted as:

$$f(\boldsymbol{x}) = \alpha + h(\boldsymbol{x})'\boldsymbol{\beta}$$

- Common kernels: linear, polynomial, radial, sigmoid.

UofT Social Science Methods Week

Validating the Models

UNIVERSITY OF TORONTO

Cross-Validation

Since the goal is to predict accurately, conventions for assessing goodness-of-fit are more rigorous in machine learning: we rely upon cross-validation techniques.

- **Training** sample: The subset used to fit the model.
- **Testing** sample: The subset used to assess the model.

Confusion Matrix/Classification Table

		Observed	
		Positive	Negative
Predicted	Positive	50	15
	Negative	5	30

- **True Positives**: 50
- **True Negatives**: 30
- **False Positives (Type I Error)**: 15
- **False Negatives (Type II Error)**: 5

Accuracy (Percent Correctly Predicted)

		Observed	
		Positive	Negative
Predicted	Positive	50	15
	Negative	5	30

- **True Positives**: 50
- **True Negatives**: 30

Accuracy

$$\text{Accuracy} = \frac{\text{True Positives} + \text{True Negatives}}{\text{Total Observations}} = \frac{50 + 30}{100} = 80\%$$

Precision

		Observed	
		Positive	Negative
Predicted	Positive	50	15
	Negative	5	30

Precision

The rate of true positives to all predicted positive outcomes:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} = \frac{50}{65} = 0.769$$

- Precision tells how well our model avoids false positives.

Recall (Sensitivity)

		Observed	
		Positive	Negative
Predicted	Positive	50	15
	Negative	5	30

Recall (Sensitivity)

The rate of true positives to all observed positive outcomes:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} = \frac{50}{55} = 0.909$$

- Recall tells how well our model avoids false negatives.

F_1 Score (or F–Score)

- Widely used in computer science and other fields as a measure of accuracy.
- The F_1 score or F score averages (harmonic mean) precision and recall.
- Computed as

$$F_1 = 2 \times \left(\frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \right)$$

F_1 Score

		Observed	
		Positive	Negative
Predicted	Positive	50	15
	Negative	5	30

- Precision: $50/(50+15) = 0.769$
- Recall: $50/(50+5) = 0.909$.

$$F_1 = 2 \times \left(\frac{0.769 \times 0.909}{0.769 + 0.909} \right) = 0.833.$$

- The closer to 1, the better.

Concluding Thoughts

Multi-Class Problems Aren't a Problem

- Most models can be fit easily with multiple classes.
- Standard packages will run one-vs-one or one-vs-all algorithms.
- Accuracy statistics need to be adjusted accordingly.

Using Predictions for Social Science Research

- Predictions can be useful to create dependent variables of interest, e.g. explaining variations in sentiment.
- If predictions are used as independent variables, accuracy matters to avoid endogeneity issues under the form of measurement error.

Selecting a Sample for Annotations

- Pay attention to the selection method used when annotating a sample for machine learning.
- Select examples at random if possible, to ensure that the distribution of words is representative of the population for which you seek to make predictions.

Unsupervised Learning

There are many models that do not require annotations to find latent classes in a dataset or corpus.

Popular examples:

- K-Means Clustering
- Hierarchical Clustering
- Latent Dirichlet Allocation
- Non-Negative Matrix Factorization
- Principal Component Analysis

Advanced Topics

- Hyperparameter optimization.
- Ensemble and bagging estimators.
- Unbalanced class problems.
- Artificial neural networks.

References

- Machine Learning:

  Trevor Hastie, Robert Tibshirani and Jerome Friedman. 2016. *The Elements of Statistical Learning*. Springer.

- Artificial Neural Networks:

  Ian Goodfellow, Yoshua Bengio and Aaron Courville. 2016. *Deep Learning*. MIT Press.