# Machine Learning for Data Mining

## Introduction to Semi-supervised Methods

Andres Mendez-Vazquez

August 21, 2020

# Outline

# Outline

# Introduction

## Semi-supervised learning

Semi-Supervised Learning (SSL) is halfway between supervised and unsupervised learning. I

## Your Data

Some labels are provided, but not for all data

## Thus the data set $X = \{x_1, x_2, \ldots, x_N\}$

It can be divided into two parts:

- The points $X_l = \{x_1, x_2, \ldots, x_l\}$ for which labels are provided $Y_l = \{y_1, y_2, \ldots, y_l\}$
- The points $X_u = \{x_{l+1}, x_{l+2}, \ldots, x_u\}$ where the labels are unknown.

# Introduction

## Semi-supervised learning

Semi-Supervised Learning (SSL) is halfway between supervised and unsupervised learning. I

## Your Data

Some labels are provided, but not for all data

## Thus the data set $X = \{x_1, x_2, \ldots, x_N\}$

It can be divided into two parts:

- The points $X_l = \{x_1, x_2, \ldots, x_l\}$ for which labels are provided $Y_l = \{y_1, y_2, \ldots, y_l\}$
- The points $X_u = \{x_{l+1}, x_{l+2}, \ldots, x_u\}$ where the labels are unknown.

# Introduction

## Semi-supervised learning

Semi-Supervised Learning (SSL) is halfway between supervised and unsupervised learning. I

## Your Data

Some labels are provided, but not for all data

## Thus the data set $X = \{x_1, x_2, ..., x_N\}$

It can be divided into two parts:

- The points $X_l = \{x_1, x_2, ..., x_l\}$ for which labels are provided $Y_l = \{y_1, y_2, ..., y_l\}$
- The points $X_u = \{x_{l+1}, x_{l+2}, ..., x_u\}$ where the labels are unknown.

# Outline

# History

## The earliest idea about using unlabeled data

It was proposed in self-learning!!!

## Definition

This is a wrapper-algorithm that repeatedly uses a supervised learning method.

# History

## The earliest idea about using unlabeled data

It was proposed in self-learning!!!

## Definition

This is a wrapper-algorithm that repeatedly uses a supervised learning method.

# Process

## First

It starts by training on the labeled data only.

## Second

In each step a part of the unlabeled points is labeled according to the current decision function:

$$\text{Given } \boldsymbol{x}_i \rightarrow y_i = f\left(\boldsymbol{x}_i\right) \tag{1}$$

## Third

The supervised method is retrained using its own predictions as additional labeled points.

# Process

## First

It starts by training on the labeled data only.

## Second

In each step a part of the unlabeled points is labeled according to the current decision function:

$$\text{Given } \boldsymbol{x}_i \rightarrow y_l = f\left(\boldsymbol{x}_i\right) \tag{1}$$

## Third

The supervised method is retrained using its own predictions as additional labeled points.

# Process

## First

It starts by training on the labeled data only.

## Second

In each step a part of the unlabeled points is labeled according to the current decision function:

$$\text{Given } \boldsymbol{x}_i \rightarrow y_l = f\left(\boldsymbol{x}_i\right) \tag{1}$$

## Third

The supervised method is retrained using its own predictions as additional labeled points.

# When did this appear?

## We have several authors proposing this idea for a long time

Scudder (1965), Fralick (1967) and Agrawala (1970).

- An unsatisfactory aspect of self-learning is that the effect of the wrapper depends on the supervised method used inside it.

# When did this appear?

## We have several authors proposing this idea for a long time

Scudder (1965), Fralick (1967) and Agrawala (1970).

- An unsatisfactory aspect of self-learning is that the effect of the wrapper depends on the supervised method used inside it.

## Closely related

A problem related to SSL was introduced by Vapnik already several decades ago.

# When did this appear?

## We have several authors proposing this idea for a long time

Scudder (1965), Fralick (1967) and Agrawala (1970).

- An unsatisfactory aspect of self-learning is that the effect of the wrapper depends on the supervised method used inside it.

## Closely related

A problem related to SSL was introduced by Vapnik already several decades ago.

## Transductive learning (Vapnik and Chervonenkis, 1974)

The idea of transduction is to perform predictions only for the test points.

- This is in contrast to inductive learning, where the goal is to output a prediction function which is defined on the entire space!!!

# When did this appear?

## We have several authors proposing this idea for a long time

Scudder (1965), Fralick (1967) and Agrawala (1970).

- An unsatisfactory aspect of self-learning is that the effect of the wrapper depends on the supervised method used inside it.

## Closely related

A problem related to SSL was introduced by Vapnik already several decades ago.

## Transductive learning ((Vapnik and Chervonenkis, 1974)

The idea of transduction is to perform predictions only for the test points.

- This is in contrast to inductive learning, where the goal is to output a prediction function which is defined on the entire space!!

# When did this appear?

## We have several authors proposing this idea for a long time

Scudder (1965), Fralick (1967) and Agrawala (1970).

- An unsatisfactory aspect of self-learning is that the effect of the wrapper depends on the supervised method used inside it.

## Closely related

A problem related to SSL was introduced by Vapnik already several decades ago.

## Transductive learning ((Vapnik and Chervonenkis, 1974)

The idea of transduction is to perform predictions only for the test points.

- This is in contrast to inductive learning, where the goal is to output a prediction function which is defined on the entire space!!!

# Finally in the 1970's SSL took off

## With the problem

Estimating the Fisher linear discriminant rule with unlabeled data was considered.

- Hosmer, 1973; McLachlan, 1977; O'Neill, 1978; McLachlan and Ganesalingam, 1982.

# Finally in the 1970's SSL took off

## With the problem

Estimating the Fisher linear discriminant rule with unlabeled data was considered.

- Hosmer, 1973; McLachlan, 1977; O'Neill, 1978; McLachlan and Ganesalingam, 1982.

## Setting

The setting was in the case where each class conditional density is Gaussian with equal covariance matrix.

# Finally in the 1970's SSL took off

## With the problem

Estimating the Fisher linear discriminant rule with unlabeled data was considered.

- Hosmer, 1973; McLachlan, 1977; O'Neill, 1978; McLachlan and Ganesalingam, 1982.

## Setting

The setting was in the case where each class conditional density is Gaussian with equal covariance matrix.

## Then

The likelihood of the model was calculated using EM... thus the labels (Hidden data) are estimated!!!

# Finally in the 1970's SSL took off

## With the problem

Estimating the Fisher linear discriminant rule with unlabeled data was considered.

- Hosmer, 1973; McLachlan, 1977; O'Neill, 1978; McLachlan and Ganesalingam, 1982.

## Setting

The setting was in the case where each class conditional density is Gaussian with equal covariance matrix.

## Then

The likelihood of the model was calculated using EM... thus the labels (Hidden data) are estimated!!!

# During the 1990's

**The SSL became a subject of great interest**

Mostly due to applications in natural language problems and text classification.

# Outline

# A simple question

**Is semi-supervised learning meaningful?**

This question is about the information that the unlabeled data can provide.

**Making possible to say the following**

Using enough data semi-supervised learning will be superior than supervised learning

# A simple question

**Is semi-supervised learning meaningful?**

This question is about the information that the unlabeled data can provide.

**Making possible to say the following**

Using enough data semi-supervised learning will be superior than supervised learning

# In a more mathematical formulation

## You could say that

1. The knowledge on $p(\boldsymbol{x})$ that one gains through the unlabeled data has to carry information.
2. That is useful in the inference of $p(l_{\boldsymbol{x}}|x)$!!!

## Something Notable

If this is not the case, semi-supervised learning will not yield an improvement over supervised learning.

## It might even happen

Usage unlabeled data degrades the prediction accuracy by misguiding the inference.

# In a more mathematical formulation

## You could say that

1. The knowledge on $p(\boldsymbol{x})$ that one gains through the unlabeled data has to carry information.
2. That is useful in the inference of $p(l_{\boldsymbol{x}}|x)$!!!

## Something Notable

If this is not the case, semi-supervised learning will not yield an improvement over supervised learning.

## It might even happen

Usage unlabeled data degrades the prediction accuracy by misguiding the inference.

# In a more mathematical formulation

## You could say that

1. The knowledge on $p(\boldsymbol{x})$ that one gains through the unlabeled data has to carry information.
2. That is useful in the inference of $p(l_{\boldsymbol{x}}|x)$!!!

## Something Notable

If this is not the case, semi-supervised learning will not yield an improvement over supervised learning.

## It might even happen

Usage unlabeled data degrades the prediction accuracy by misguiding the inference.

# Assumptions to be made

**However, your data must have the four principles**

1. The Semi-Supervised Smoothness Assumption.
2. The Cluster Assumption.
3. The Manifold Assumption.
4. Transduction Principle.

# Assumptions to be made

**However, your data must have the four principles**

1. The Semi-Supervised Smoothness Assumption.
2. The Cluster Assumption.
3. The Manifold Assumption.
4. Transduction Principle.

# Assumptions to be made

## However, your data must have the four principles

1. The Semi-Supervised Smoothness Assumption.
2. The Cluster Assumption.
3. The Manifold Assumption.
4. Transduction Principle.

# Assumptions to be made

**However, your data must have the four principles**

1. The Semi-Supervised Smoothness Assumption.
2. The Cluster Assumption.
3. The Manifold Assumption.
4. Transduction Principle.

# Outline

# Supervised Smoothness Assumption

## Definition

If two points $\boldsymbol{x}_1$, $\boldsymbol{x}_2$ are close, then the corresponding outputs $f(\boldsymbol{x}_1)$ and $f(\boldsymbol{x}_2)$.

- Where $f$ is the supervised algorithm.
- Strictly speaking, this assumption only refers to continuity rather than smoothness.

## Without such assumptions

It would never be possible to generalize from a finite training set to a set of possibly infinitely many unseen test cases.

# Supervised Smoothness Assumption

## Definition

If two points $\boldsymbol{x}_1$, $\boldsymbol{x}_2$ are close, then the corresponding outputs $f(\boldsymbol{x}_1)$ and $f(\boldsymbol{x}_2)$.

- Where $f$ is the supervised algorithm.
- Strictly speaking, this assumption only refers to continuity rather than smoothness.

## Without such assumptions

It would never be possible to generalize from a finite training set to a set of possibly infinitely many unseen test cases.

# Supervised Smoothness Assumption

## Definition

If two points $\boldsymbol{x}_1$, $\boldsymbol{x}_2$ are close, then the corresponding outputs $f(\boldsymbol{x}_1)$ and $f(\boldsymbol{x}_2)$.

- Where $f$ is the supervised algorithm.
- Strictly speaking, this assumption only refers to continuity rather than smoothness.

## Without such assumptions

It would never be possible to generalize from a finite training set to a set of possibly infinitely many unseen test cases.

## Remember?

ill-posed problems.

# Supervised Smoothness Assumption

## Definition

If two points $\boldsymbol{x}_1$, $\boldsymbol{x}_2$ are close, then the corresponding outputs $f(\boldsymbol{x}_1)$ and $f(\boldsymbol{x}_2)$.

- Where $f$ is the supervised algorithm.
- Strictly speaking, this assumption only refers to continuity rather than smoothness.

## Without such assumptions

It would never be possible to generalize from a finite training set to a set of possibly infinitely many unseen test cases.

## Remember?

ill-posed problems.

# Supervised Smoothness Assumption

## Definition

If two points $\boldsymbol{x}_1$, $\boldsymbol{x}_2$ are close, then the corresponding outputs $f(\boldsymbol{x}_1)$ and $f(\boldsymbol{x}_2)$.

- Where $f$ is the supervised algorithm.
- Strictly speaking, this assumption only refers to continuity rather than smoothness.

## Without such assumptions

It would never be possible to generalize from a finite training set to a set of possibly infinitely many unseen test cases.

## Remember?

ill-posed problems.

# The Semi-Supervised Smoothness Assumption

**Definition**

If two points $\boldsymbol{x}_1, \boldsymbol{x}_2$ in a ***high-density region*** are close, then so should be the corresponding outputs $f(\boldsymbol{x}_1)$ and $f(\boldsymbol{x}_2)$.

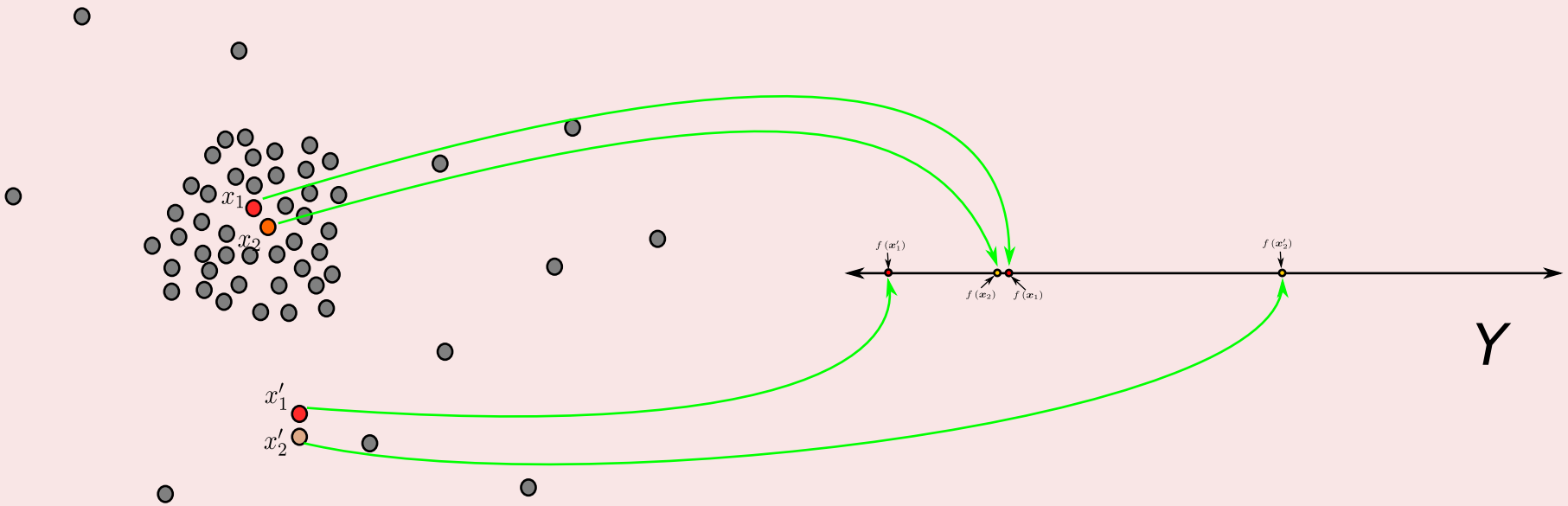
For example

# The Semi-Supervised Smoothness Assumption

## Definition

If two points $\boldsymbol{x}_1, \boldsymbol{x}_2$ in a ***high-density region*** are close, then so should be the corresponding outputs $f(\boldsymbol{x}_1)$ and $f(\boldsymbol{x}_2)$.

## For example



# Implications

## Given Transitivity

The assumption implies that if two points are linked by a path of high density, then their outputs are likely to be close

## Thus

If they are separated by a low-density region, then their outputs need not be close.

# Implications

## Given Transitivity

The assumption implies that if two points are linked by a path of high density, then their outputs are likely to be close

## Thus

If they are separated by a low-density region, then their outputs need not be close.

# Outline

# Introduction

**Now**

Suppose that the points of each class tended to form a cluster.

**Then**

We can use the unlabeled data could help to find the boundary of each cluster more accurately.

**Thus**

One could run a clustering algorithm and use the labeled points to assign a class to each cluster.

# Introduction

**Now**

Suppose that the points of each class tended to form a cluster.

**Then**

We can use the unlabeled data could help to find the boundary of each cluster more accurately.

**Thus**

One could run a clustering algorithm and use the labeled points to assign a class to each cluster.

# Introduction

**Now**

Suppose that the points of each class tended to form a cluster.

**Then**

We can use the unlabeled data could help to find the boundary of each cluster more accurately.

**Thus**

One could run a clustering algorithm and use the labeled points to assign a class to each cluster.

# Thus

**In fact**

This is the earliest form of semi-supervised learning.

# The Cluster Assumption

**Definition**

If points are in the same cluster, they are likely to be of the same class.

**Remark**

This assumption may be considered reasonable on the basis of the sheer existence of classes

**It is more**

if there is a densely populated continuum of objects, it is unlikely that they could be distinguished into different classes.

# The Cluster Assumption

**Definition**

If points are in the same cluster, they are likely to be of the same class.

**Remark**

This assumption may be considered reasonable on the basis of the sheer existence of classes

**It is more**

if there is a densely populated continuum of objects, it is unlikely that they could be distinguished into different classes.

# The Cluster Assumption

**Definition**

If points are in the same cluster, they are likely to be of the same class.

**Remark**

This assumption may be considered reasonable on the basis of the sheer existence of classes

**It is more**

if there is a densely populated continuum of objects, it is unlikely that they could be distinguished into different classes.

# Meaning

## Something Notable

It means that, usually, we do not observe objects of two distinct classes in the same cluster.

## In addition

***Low density separation***: The decision boundary should lie in a low-density region.

## We can see that not?

Thing about this!!!

# Meaning

## Something Notable

It means that, usually, we do not observe objects of two distinct classes in the same cluster.

## In addition

***Low density separation***: The decision boundary should lie in a low-density region.

## We can see that not?

Thing about this!!!

# Meaning

## Something Notable

It means that, usually, we do not observe objects of two distinct classes in the same cluster.

## In addition

***Low density separation***: The decision boundary should lie in a low-density region.

## We can see that not?

Thing about this!!!

# Not only that

**Although the definition are equivalent**

They inspire different algorithms

**Think about this**

Some example?

# Not only that

**Although the definition are equivalent**

They inspire different algorithms

**Think about this**

Some example?

# Outline

# The Manifold Assumption

**Definition**

The (high-dimensional) data lie (roughly) on a low-dimensional manifold.

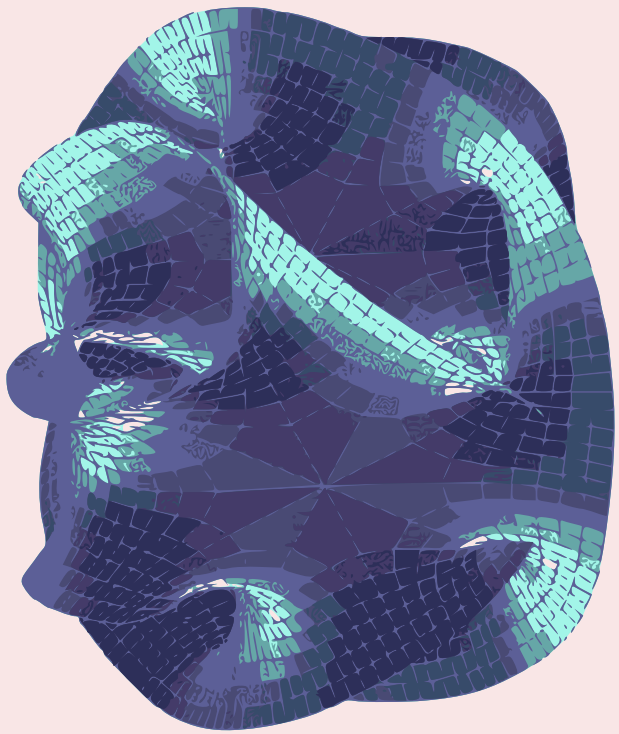
**Example**

# The Manifold Assumption

## Definition

The (high-dimensional) data lie (roughly) on a low-dimensional manifold.

## Example



# How this can help us?

**We have the following WELL KNOWN PROBLEM**

The so-called curse of dimensionality

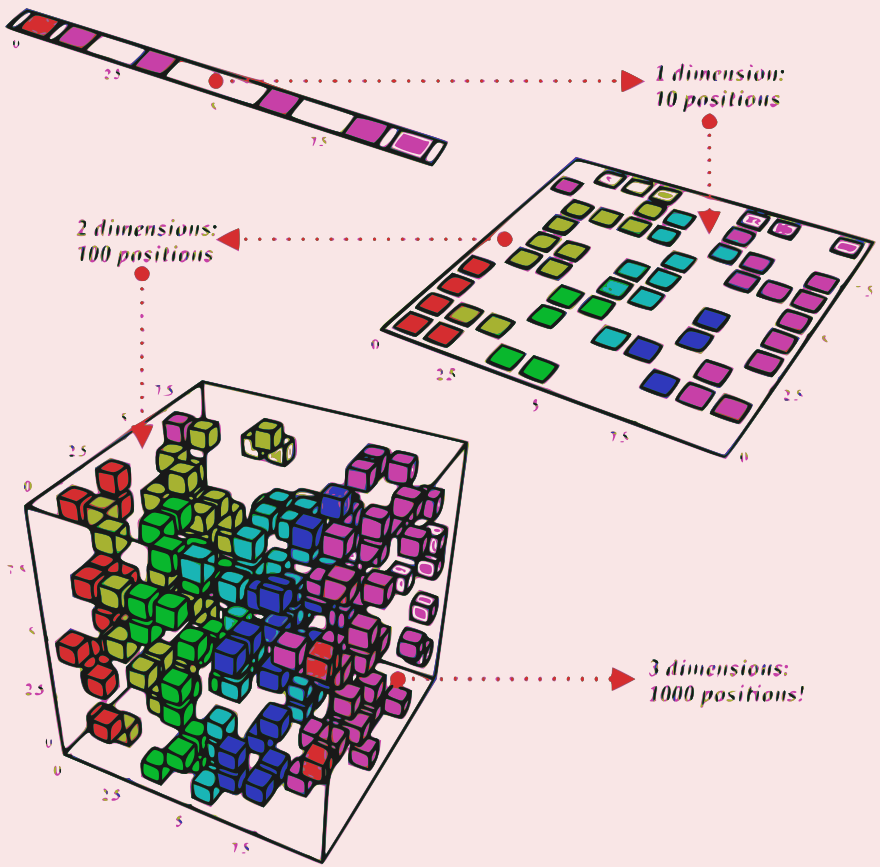
**Example**

# How this can help us?

## We have the following WELL KNOWN PROBLEM

The so-called curse of dimensionality

## Example



# Thus

## Higher Dimensions

A Larger Amount of Data is required!!!

# Thus

## Higher Dimensions

A Larger Amount of Data is required!!!

## However

If the data happen to lie on a low-dimensional manifold the learning algorithm can operate un a space of corresponding dimension.

- Thus, avoiding the curse of dimensionality

# Thus

## Higher Dimensions

A Larger Amount of Data is required!!!

## However

If the data happen to lie on a low-dimensional manifold the learning algorithm can operate un a space of corresponding dimension.

- Thus, avoiding the curse of dimensionality

# Facts

## First

Working with manifolds can be seen as approximately implementing the semi-supervised smoothness assumption.

- Algorithms use the metric of the manifold for computing geodesic distances.

# Facts

## First

Working with manifolds can be seen as approximately implementing the semi-supervised smoothness assumption.

- Algorithms use the metric of the manifold for computing geodesic distances.

## Second

If we view the manifold as an approximation of the high-density regions.

- The semi-supervised smoothness assumption reduces to the standard smoothness assumption of supervised learning, applied on the manifold.

# Facts

## First

Working with manifolds can be seen as approximately implementing the semi-supervised smoothness assumption.

- Algorithms use the metric of the manifold for computing geodesic distances.

## Second

If we view the manifold as an approximation of the high-density regions.

- The semi-supervised smoothness assumption reduces to the standard smoothness assumption of supervised learning, applied on the manifold.

## Third

if the manifold is embedded into the high-dimensional input space in a curved fashion (i.e., it is not just a subspace)

- Geodesic distances differ from those in the input space.

# Facts

## First

Working with manifolds can be seen as approximately implementing the semi-supervised smoothness assumption.

- Algorithms use the metric of the manifold for computing geodesic distances.

## Second

If we view the manifold as an approximation of the high-density regions.

- The semi-supervised smoothness assumption reduces to the standard smoothness assumption of supervised learning, applied on the manifold.

## Third

if the manifold is embedded into the high-dimensional input space in a curved fashion (i.e., it is not just a subspace)

- Geodesic distances differ from those in the input space.

# Facts

## First

Working with manifolds can be seen as approximately implementing the semi-supervised smoothness assumption.

- Algorithms use the metric of the manifold for computing geodesic distances.

## Second

If we view the manifold as an approximation of the high-density regions.

- The semi-supervised smoothness assumption reduces to the standard smoothness assumption of supervised learning, applied on the manifold.

## Third

if the manifold is embedded into the high-dimensional input space in a curved fashion (i.e., it is not just a subspace)

- Geodesic distances differ from those in the input space.

# Facts

## First

Working with manifolds can be seen as approximately implementing the semi-supervised smoothness assumption.

- Algorithms use the metric of the manifold for computing geodesic distances.

## Second

If we view the manifold as an approximation of the high-density regions.

- The semi-supervised smoothness assumption reduces to the standard smoothness assumption of supervised learning, applied on the manifold.

## Third

if the manifold is embedded into the high-dimensional input space in a curved fashion (i.e., it is not just a subspace)

- Geodesic distances differ from those in the input space.

# Outline

# Vapnik's principle

## Definition

High-dimensional estimation problems should attempt to follow the following principle:

- When trying to solve some problem, one should not solve a more difficult problem as an intermediate step.

# Vapnik's principle

**Definition**

High-dimensional estimation problems should attempt to follow the following principle:

- When trying to solve some problem, one should not solve a more difficult problem as an intermediate step.

For example in supervised learning

- Generative models estimate the density of x as an intermediate step.
- Discriminative methods directly estimate the labels.

# Vapnik's principle

## Definition

High-dimensional estimation problems should attempt to follow the following principle:

- When trying to solve some problem, one should not solve a more difficult problem as an intermediate step.

## For example in supervised learning

- Generative models estimate the density of x as an intermediate step.
- Discriminative methods directly estimate the labels.

## Given that

Label predictions are only required for a given test set in the transductive setting.

# Vapnik's principle

## Definition

High-dimensional estimation problems should attempt to follow the following principle:

- When trying to solve some problem, one should not solve a more difficult problem as an intermediate step.

## For example in supervised learning

- Generative models estimate the density of x as an intermediate step.
- Discriminative methods directly estimate the labels.

## Given that

Label predictions are only required for a given test set in the transductive setting.

# Vapnik's principle

## Definition

High-dimensional estimation problems should attempt to follow the following principle:

- When trying to solve some problem, one should not solve a more difficult problem as an intermediate step.

## For example in supervised learning

- Generative models estimate the density of x as an intermediate step.
- Discriminative methods directly estimate the labels.

## Given that

Label predictions are only required for a given test set in the transductive setting.

# Therefore

## While an inductive method infers a function

$$f : \mathcal{X} \to \mathcal{Y} \tag{2}$$

Over the entire space to obtain inferences $f(x_i)$ over inputs $x_i$.

# Therefore

## While an inductive method infers a function

$$f : \mathcal{X} \to \mathcal{Y} \tag{2}$$

Over the entire space to obtain inferences $f(\boldsymbol{x}_i)$ over inputs $\boldsymbol{x}_i$.

## This is different of Transduction

Transduction consists of directly estimating the finite set of test labels,

$$f : \mathcal{X}_u \to \mathcal{Y} \tag{3}$$

Note: This is only defined on the test set

# Therefore

## While an inductive method infers a function

$$f : \mathcal{X} \to \mathcal{Y} \tag{2}$$

Over the entire space to obtain inferences $f(\boldsymbol{x}_i)$ over inputs $\boldsymbol{x}_i$.

## This is different of Transduction

Transduction consists of directly estimating the finite set of test labels,

$$f : \mathcal{X}_u \to \mathcal{Y} \tag{3}$$

Note: This is only defined on the test set

# Therefore

## While an inductive method infers a function

$$f : \mathcal{X} \to \mathcal{Y} \tag{2}$$

Over the entire space to obtain inferences $f(\boldsymbol{x}_i)$ over inputs $\boldsymbol{x}_i$.

## This is different of Transduction

Transduction consists of directly estimating the finite set of test labels,

$$f : \mathcal{X}_u \to \mathcal{Y} \tag{3}$$

Note: This is only defined on the test set

# Transduction VS SSL

**Remark**

Note that transduction is not the same as SSL:

- Some semi-supervised algorithms are transductive, but others are inductive.

# Transduction VS SSL

**Remark**

Note that transduction is not the same as SSL:

- Some semi-supervised algorithms are transductive, but others are inductive.

# What?

## To see why the difference, imagine the following example

Suppose we are given a transductive algorithm which produces a solution superior to an inductive algorithm trained on the same labeled data (Discarding the unlabeled data).

# What?

## To see why the difference, imagine the following example

Suppose we are given a transductive algorithm which produces a solution superior to an inductive algorithm trained on the same labeled data (Discarding the unlabeled data).

## This difference might be due to the following points

- Transduction follows Vapnik's principle more closely than induction does.
- The transductive algorithm takes advantage of the unlabeled data in a way similar to semi-supervised learning algorithms.

# What?

## To see why the difference, imagine the following example

Suppose we are given a transductive algorithm which produces a solution superior to an inductive algorithm trained on the same labeled data (Discarding the unlabeled data).

## This difference might be due to the following points

- Transduction follows Vapnik's principle more closely than induction does.
- The transductive algorithm takes advantage of the unlabeled data in a way similar to semi-supervised learning algorithms.

## Remark

- There is ample evidence for improvements being due to the second of these points.
- It seems that there are no empirical results that selectively support the first point.

# What?

## To see why the difference, imagine the following example

Suppose we are given a transductive algorithm which produces a solution superior to an inductive algorithm trained on the same labeled data (Discarding the unlabeled data).

## This difference might be due to the following points

- Transduction follows Vapnik's principle more closely than induction does.
- The transductive algorithm takes advantage of the unlabeled data in a way similar to semi-supervised learning algorithms.

## Remark

- There is ample evidence for improvements being due to the second of these points.
- It seems that there are no empirical results that selectively support the first point.

# What?

## To see why the difference, imagine the following example

Suppose we are given a transductive algorithm which produces a solution superior to an inductive algorithm trained on the same labeled data (Discarding the unlabeled data).

## This difference might be due to the following points

- Transduction follows Vapnik's principle more closely than induction does.
- The transductive algorithm takes advantage of the unlabeled data in a way similar to semi-supervised learning algorithms.

## Remark

- There is ample evidence for improvements being due to the second of these points.
- It seems that there are no empirical results that selectively support the first point.

# Outline

# Setup

## Since SSL methods are supervised learning techniques

The SSL's methods can be divided into

- Generative
- Low-Density Separation
- Graph-Based methods

Although, there is the need to be careful given the unlabeled data!!!

# Setup

## Since SSL methods are supervised learning techniques

The SSL's methods can be divided into

- Generative
- Low-Density Separation
- Graph-Based methods

Although, there is the need to be careful given the unlabeled data!!!

# Setup

## Since SSL methods are supervised learning techniques

The SSL's methods can be divided into

- Generative
- Low-Density Separation
- Graph-Based methods

Although, there is the need to be careful given the unlabeled data!!!

# Setup

## Since SSL methods are supervised learning techniques

The SSL's methods can be divided into

- Generative
- Low-Density Separation
- Graph-Based methods

Although, there is the need to be careful given the unlabeled data!!!

# Setup

## Since SSL methods are supervised learning techniques

The SSL's methods can be divided into

- Generative
- Low-Density Separation
- Graph-Based methods

Although, there is the need to be careful given the unlabeled data!!!

# Outline

# Introduction

## Classes are modeled using a distribution $P(\boldsymbol{x}|y)$

For this, we use a family of models parametrized by

$$\{P(\boldsymbol{x}|y,\theta)\} \tag{4}$$

## Furthermore

The class of priors $P(y)$ for the labels is model by

$$\pi_y = P(y|\boldsymbol{\pi}) \text{ with } \boldsymbol{\pi} = (\pi_y)_y \tag{5}$$

## This is known as the joint density model

After all we are using a full joint density $P(\boldsymbol{x}, y)$ by

$$P(\boldsymbol{x}|y,\boldsymbol{\theta})\, P(y|\boldsymbol{\pi}) = P(\boldsymbol{x}|y,\boldsymbol{\theta})\, \pi_y$$

# Introduction

## Classes are modeled using a distribution $P(\boldsymbol{x}|y)$

For this, we use a family of models parametrized by

$$\{P(\boldsymbol{x}|y,\theta)\} \tag{4}$$

## Furthermore

The class of priors $P(y)$ for the labels is model by

$$\pi_y = P(y|\boldsymbol{\pi}) \text{ with } \boldsymbol{\pi} = (\pi_y)_y \tag{5}$$

## This is known as the joint density model

After all we are using a full joint density $P(\boldsymbol{x}, y)$ by
$P(\boldsymbol{x}|y,\boldsymbol{\theta})\, P(y|\boldsymbol{\pi}) = P(\boldsymbol{x}|y,\boldsymbol{\theta})\, \pi_y$

# Introduction

## Classes are modeled using a distribution $P(\boldsymbol{x}|y)$

For this, we use a family of models parametrized by

$$\{P(\boldsymbol{x}|y,\theta)\} \tag{4}$$

## Furthermore

The class of priors $P(y)$ for the labels is model by

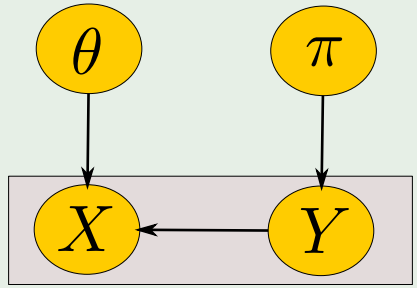
$$\pi_y = P(y|\boldsymbol{\pi}) \text{ with } \boldsymbol{\pi} = (\pi_y)_y \tag{5}$$

## This is known as the joint density model

After all we are using a full joint density $P(\boldsymbol{x},y)$ by $P(\boldsymbol{x}|y,\boldsymbol{\theta})\,P(y|\boldsymbol{\pi}) = P(\boldsymbol{x}|y,\boldsymbol{\theta})\,\pi_y$

# Example

## Graphical Model of the Joint Density




Thus for any fixed $\hat{\theta}$ and $\hat{\pi}$, and a labels $y \in \{1, \ldots, M\}$

$$P\left(y|x, \hat{\theta}, \hat{\pi}\right) = \frac{P\left(x|y, \hat{\theta}\right)\hat{\pi}_y}{\sum_{y'=1}^{M} P\left(x|y', \hat{\theta}\right)\hat{\pi}_{y'}} \tag{6}$$

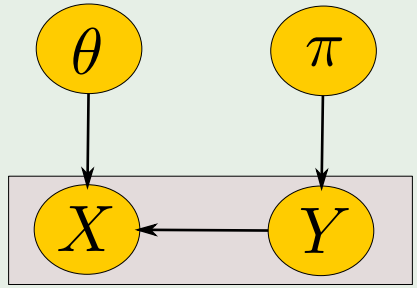
Alternatively

One can obtain the Bayesian predictive distribution $P\left(y|x, D_l\right)$ ($D_l$ the labeled data) by averaging

$$\frac{P\left(y|x, \theta, \pi\right)}{P\left(\theta, \pi|D_l\right)} \tag{7}$$

# Example

## Graphical Model of the Joint Density




## Thus for any fixed $\hat{\boldsymbol{\theta}}$ and $\hat{\pi}$, and a labels $y \in \{1, ..., M\}$

$$P\left(y|\boldsymbol{x}, \hat{\boldsymbol{\theta}}, \hat{\pi}\right) = \frac{P\left(\boldsymbol{x}|y, \hat{\boldsymbol{\theta}}\right)\hat{\pi}_y}{\sum_{y'=1}^{M} P\left(\boldsymbol{x}|y', \hat{\boldsymbol{\theta}}\right)\hat{\pi}_{y'}} \tag{6}$$

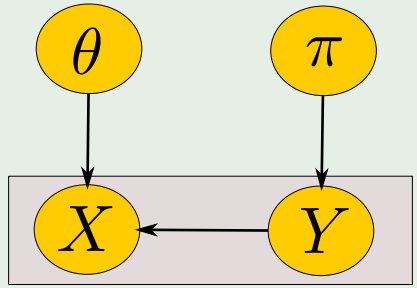
## Alternatively

One can obtain the Bayesian predictive distribution $P\left(y|x, D_l\right)$ ($D_l$ the labeled data) by averaging

$$\frac{P\left(y|\boldsymbol{x}, \boldsymbol{\theta}, \boldsymbol{\pi}\right)}{P\left(\boldsymbol{\theta}, \boldsymbol{\pi}|D_l\right)} \tag{7}$$

# Example

## Graphical Model of the Joint Density




## Thus for any fixed $\hat{\boldsymbol{\theta}}$ and $\hat{\pi}$, and a labels $y \in \{1, ..., M\}$

$$P\left(y|\boldsymbol{x}, \hat{\boldsymbol{\theta}}, \hat{\pi}\right) = \frac{P\left(\boldsymbol{x}|y, \hat{\boldsymbol{\theta}}\right)\hat{\pi}_y}{\sum_{y'=1}^{M} P\left(\boldsymbol{x}|y', \hat{\boldsymbol{\theta}}\right)\hat{\pi}_{y'}} \tag{6}$$

## Alternatively

One can obtain the Bayesian predictive distribution $P\left(y|\boldsymbol{x}, D_l\right)$ ($D_l$ the labeled data) by averaging

$$\frac{P\left(y|\boldsymbol{x}, \boldsymbol{\theta}, \pi\right)}{P\left(\boldsymbol{\theta}, \pi|D_l\right)} \tag{7}$$

# Clearly

**The marginal $P(\boldsymbol{x}|\boldsymbol{\theta}, \pi)$ can be seen as**

$$P(\boldsymbol{x}|\boldsymbol{\theta}, \pi) = \sum_{y=1}^{M} P(\boldsymbol{x}|y, \boldsymbol{\theta}) \pi_y \tag{8}$$

**If labeled and unlabeled data are available**

A natural criterion emerges as the joint log likelihood of both $D_l$ and $D_u$:

$$\sum_{i=1}^{n} \log \pi_{y_i} P(\boldsymbol{x}_i|y_i, \theta) + \sum_{i=n+1}^{n+m} \log \left[ \sum_{y=1}^{M} \pi_y P(\boldsymbol{x}_i|y, \theta) \right] \tag{9}$$

# Clearly

**The marginal $P(\boldsymbol{x}|\boldsymbol{\theta}, \pi)$ can be seen as**

$$P(\boldsymbol{x}|\boldsymbol{\theta}, \pi) = \sum_{y=1}^{M} P(\boldsymbol{x}|y, \boldsymbol{\theta}) \pi_y \tag{8}$$

**If labeled and unlabeled data are available**

A natural criterion emerges as the joint log likelihood of both $D_l$ and $D_u$:

$$\sum_{i=1}^{n} \log \pi_{y_i} P(\boldsymbol{x}_i|y_i, \boldsymbol{\theta}) + \sum_{i=n+1}^{n+m} \log \left[ \sum_{y=1}^{M} \pi_y P(\boldsymbol{x}_i|y, \boldsymbol{\theta}) \right] \tag{9}$$

# Thus

## Essentially

It is an issue of maximum likelihood in the presence of missing data by threating $y$ as a latent variable.

# Thus

## Essentially

It is an issue of maximum likelihood in the presence of missing data by threating $y$ as a latent variable.

## Clearly

We can deal with this problem using EM algorithm.

# Thus

## Essentially

It is an issue of maximum likelihood in the presence of missing data by threating $y$ as a latent variable.

## Clearly

We can deal with this problem using EM algorithm.

## However

Some researchers have been quick in hailing this strategy as an obvious solution SSL:

- Problem maximizing the joint likelihood of a finite sample need not lead to a small classification error.

# Thus

## Essentially

It is an issue of maximum likelihood in the presence of missing data by threating $y$ as a latent variable.

## Clearly

We can deal with this problem using EM algorithm.

## However

Some researchers have been quick in hailing this strategy as an obvious solution SSL:

- Problem maximizing the joint likelihood of a finite sample need not lead to a small classification error.

# Outline

# Introduction

**We want to**

To push the decision boundary away from the unlabeled points.

**The most common approach**

To use a maximum margin algorithm as SVM.

**This method is known as**

The method of maximizing the margin for unlabeled as well as labeled points is called the Transductive SVM (TSVM).

# Introduction

**We want to**

To push the decision boundary away from the unlabeled points.

**The most common approach**

To use a maximum margin algorithm as SVM.

**This method is known as**

The method of maximizing the margin for unlabeled as well as labeled points is called the Transductive SVM (TSVM).

# Introduction

**We want to**

To push the decision boundary away from the unlabeled points.

**The most common approach**

To use a maximum margin algorithm as SVM.

**This method is known as**

The method of maximizing the margin for unlabeled as well as labeled points is called the Transductive SVM (TSVM).

# Thus

## The process using a SVM is as follow

1. Starting from the SVM solution as trained on the labeled data only.
2. The unlabeled points are labeled by SVM predictions.
3. The SVM is retrained on all points.

Thus: This is iterated while the weight of the unlabeled points is slowly increased.

# Thus

## The process using a SVM is as follow

1. Starting from the SVM solution as trained on the labeled data only.
2. The unlabeled points are labeled by SVM predictions.
3. The SVM is retrained on all points.

Thus: This is iterated while the weight of the unlabeled points is slowly increased.

# Thus

## The process using a SVM is as follow

1. Starting from the SVM solution as trained on the labeled data only.
2. The unlabeled points are labeled by SVM predictions.
3. The SVM is retrained on all points.

Thus: This is iterated while the weight of the unlabeled points is slowly increased.

# Thus

## The process using a SVM is as follow

1. Starting from the SVM solution as trained on the labeled data only.
2. The unlabeled points are labeled by SVM predictions.
3. The SVM is retrained on all points.

Thus: This is iterated while the weight of the unlabeled points is slowly increased.

# Alternatives

## Two alternatives to the TSVM are

1. Probability framework.
2. Information theoretic framework.

# Alternatives

## Two alternatives to the TSVM are

1. Probability framework.
2. Information theoretic framework.

## For example

We can use binary Gaussian process classification

- By introducing a null class that occupies the space between the two regular classes.

# Alternatives

## Two alternatives to the TSVM are

1. Probability framework.
2. Information theoretic framework.

## For example

We can use binary Gaussian process classification

- By introducing a null class that occupies the space between the two regular classes.

## Another example

Using Entropy Minimization, it is possible to push the class-conditional probabilities $P(y|x)$ to 0 or 1 at unlabeled and labeled points.

- Given the smoothness assumption, the probability will tend to 0 or 1 in any high-density region.
- While class boundaries correspond to intermediate probabilities.

# Alternatives

## Two alternatives to the TSVM are

1. Probability framework.
2. Information theoretic framework.

## For example

We can use binary Gaussian process classification

- By introducing a null class that occupies the space between the two regular classes.

## Another example

Using Entropy Minimization, it is possible to push the class-conditional probabilities $P(y|x)$ to 0 or 1 at unlabeled and labeled points.

- Given the smoothness assumption, the probability will tend to 0 or 1 in any high-density region.
- While class boundaries correspond to intermediate probabilities.

# Alternatives

## Two alternatives to the TSVM are

1. Probability framework.
2. Information theoretic framework.

## For example

We can use binary Gaussian process classification

- By introducing a null class that occupies the space between the two regular classes.

## Another example

Using Entropy Minimization, it is possible to push the class-conditional probabilities $P(y|\boldsymbol{x})$ to 0 or 1 at unlabeled and labeled points.

- Given the smoothness assumption, the probability will tend to 0 or 1 in any high-density region.
- While class boundaries correspond to intermediate probabilities.

# Alternatives

## Two alternatives to the TSVM are

1. Probability framework.
2. Information theoretic framework.

## For example

We can use binary Gaussian process classification

- By introducing a null class that occupies the space between the two regular classes.

## Another example

Using Entropy Minimization, it is possible to push the class-conditional probabilities $P(y|\boldsymbol{x})$ to 0 or 1 at unlabeled and labeled points.

- Given the smoothness assumption, the probability will tend to 0 or 1 in any high-density region.
- While class boundaries correspond to intermediate probabilities.

# Alternatives

## Two alternatives to the TSVM are

1. Probability framework.
2. Information theoretic framework.

## For example

We can use binary Gaussian process classification

- By introducing a null class that occupies the space between the two regular classes.

## Another example

Using Entropy Minimization, it is possible to push the class-conditional probabilities $P(y|\boldsymbol{x})$ to 0 or 1 at unlabeled and labeled points.

- Given the smoothness assumption, the probability will tend to 0 or 1 in any high-density region.
- While class boundaries correspond to intermediate probabilities.

# Outline

# Graph-Based Methods

## This has been an active area of research in SSL

The common denominator of these methods is that

- The data are represented by the nodes of a graph.
- The edges are the pairwise distance between nodes.
  - Missing edges correspond to infinite distances.

# Graph-Based Methods

## This has been an active area of research in SSL

The common denominator of these methods is that

- The data are represented by the nodes of a graph.
- The edges are the pairwise distance between nodes.
  - Missing edges correspond to infinite distances.

## Now

If the distance of two points is computed by minimizing the aggregate path distance over all paths connecting the two points.

- This is an approximation to the geodesic distance of the two points!!!

# Graph-Based Methods

## This has been an active area of research in SSL

The common denominator of these methods is that

- The data are represented by the nodes of a graph.
- The edges are the pairwise distance between nodes.
  - Missing edges correspond to infinite distances.

## Now

If the distance of two points is computed by minimizing the aggregate path distance over all paths connecting the two points.

- This is an approximation to the geodesic distance of the two points!!!

## Then, the graph

It can be argued to build on the manifold assumption.

# Graph-Based Methods

## This has been an active area of research in SSL

The common denominator of these methods is that

- The data are represented by the nodes of a graph.
- The edges are the pairwise distance between nodes.
  - Missing edges correspond to infinite distances.

## Now

If the distance of two points is computed by minimizing the aggregate path distance over all paths connecting the two points.

- This is an approximation to the geodesic distance of the two points!!!

## Then, the graph

It can be argued to build on the manifold assumption.

# Graph-Based Methods

## This has been an active area of research in SSL

The common denominator of these methods is that

- The data are represented by the nodes of a graph.
- The edges are the pairwise distance between nodes.
  - Missing edges correspond to infinite distances.

## Now

If the distance of two points is computed by minimizing the aggregate path distance over all paths connecting the two points.

- This is an approximation to the geodesic distance of the two points!!!

Then, the graph

It can be argued to build on the manifold assumption.

# Graph-Based Methods

## This has been an active area of research in SSL

The common denominator of these methods is that

- The data are represented by the nodes of a graph.
- The edges are the pairwise distance between nodes.
  - Missing edges correspond to infinite distances.

## Now

If the distance of two points is computed by minimizing the aggregate path distance over all paths connecting the two points.

- This is an approximation to the geodesic distance of the two points!!!

## Then, the graph

It can be argued to build on the manifold assumption.

# Graph-Based Methods

## This has been an active area of research in SSL

The common denominator of these methods is that

- The data are represented by the nodes of a graph.
- The edges are the pairwise distance between nodes.
  - Missing edges correspond to infinite distances.

## Now

If the distance of two points is computed by minimizing the aggregate path distance over all paths connecting the two points.

- This is an approximation to the geodesic distance of the two points!!!

## Then, the graph

It can be argued to build on the manifold assumption.

# Now

## Most graph methods

They refer to the graph by utilizing the Laplacian Matrix.

# Now

## Most graph methods

They refer to the graph by utilizing the Laplacian Matrix.

## Setup

Let be $G = (V, E)$ a graph with real edge weights given by weight function $w : E \to \mathbb{R}$ with:

- $w(e)$ represents the similarity of the incident nodes.
- A missing edge correspond to zero similarity.

# Now

## Most graph methods

They refer to the graph by utilizing the Laplacian Matrix.

## Setup

Let be $G = (V, E)$ a graph with real edge weights given by weight function $w : E \to \mathbb{R}$ with:

- $w(e)$ represents the similarity of the incident nodes.
- A missing edge correspond to zero similarity.

## Now

The weighted adjacency matrix $W$ of graph $G$ is defined by

$$W_{ij} = \begin{cases} w(e) & \text{if } e = (i, j) \in E \\ 0 & \text{if } e = (i, j) \notin E \end{cases} \tag{10}$$

# Now

## Most graph methods

They refer to the graph by utilizing the Laplacian Matrix.

## Setup

Let be $G = (V, E)$ a graph with real edge weights given by weight function $w : E \to \mathbb{R}$ with:

- $w(e)$ represents the similarity of the incident nodes.
- A missing edge correspond to zero similarity.

## Now

The weighted adjacency matrix $W$ of graph $G$ is defined by

$$W_{ij} = \begin{cases} w(e) & \text{if } e = (i, j) \in E \\ 0 & \text{if } e = (i, j) \notin E \end{cases} \tag{10}$$

# Now

## Most graph methods

They refer to the graph by utilizing the Laplacian Matrix.

## Setup

Let be $G = (V, E)$ a graph with real edge weights given by weight function $w : E \to \mathbb{R}$ with:

- $w(e)$ represents the similarity of the incident nodes.
- A missing edge correspond to zero similarity.

## Now

The weighted adjacency matrix $\boldsymbol{W}$ of graph $G$ is defined by

$$W_{ij} = \begin{cases} w(e) & \text{if } e = (i,j) \in E \\ 0 & \text{if } e = (i,j) \notin E \end{cases} \tag{10}$$

# Now

## Most graph methods

They refer to the graph by utilizing the Laplacian Matrix.

## Setup

Let be $G = (V, E)$ a graph with real edge weights given by weight function $w : E \to \mathbb{R}$ with:

- $w(e)$ represents the similarity of the incident nodes.
- A missing edge correspond to zero similarity.

## Now

The weighted adjacency matrix $\boldsymbol{W}$ of graph $G$ is defined by

$$\boldsymbol{W}_{ij} = \begin{cases} w(e) & \text{if } e = (i,j) \in E \\ 0 & \text{if } e = (i,j) \notin E \end{cases} \tag{10}$$

# The diagonal matrix $\boldsymbol{D}$

**Definition**

The diagonal matrix $\boldsymbol{D}$ defined by $\boldsymbol{D}_{ii} = \sum_j \boldsymbol{W}_{ij}$ is called the degree matrix of $G$.

# The diagonal matrix $D$

**Definition**

The diagonal matrix $D$ defined by $D_{ii} = \sum_j W_{ij}$ is called the degree matrix of $G$.

**Although there are different ways of defining the Laplacian Matrix**

We decided to use the normalized and unormalized versions.

**Unnormalized Matrix $L = D - W$**

# The diagonal matrix $\boldsymbol{D}$

**Definition**

The diagonal matrix $\boldsymbol{D}$ defined by $\boldsymbol{D}_{ii} = \sum_j \boldsymbol{W}_{ij}$ is called the degree matrix of $G$.

**Although there are different ways of defining the Laplacian Matrix**

We decided to use the normalized and unormalized versions.

**Laplacian Matrix**

- Normalized Matrix: $\mathcal{L} = \boldsymbol{I} - \boldsymbol{D}^{-1/2}\boldsymbol{W}\boldsymbol{D}^{-1/2}$.
- Unormalized Matrix: $L = D - W$.

# The diagonal matrix $\boldsymbol{D}$

**Definition**

The diagonal matrix $\boldsymbol{D}$ defined by $\boldsymbol{D}_{ii} = \sum_j \boldsymbol{W}_{ij}$ is called the degree matrix of $G$.

**Although there are different ways of defining the Laplacian Matrix**

We decided to use the normalized and unormalized versions.

**Laplacian Matrix**

- Normalized Matrix: $\mathcal{L} = \boldsymbol{I} - \boldsymbol{D}^{-1/2}\boldsymbol{W}\boldsymbol{D}^{-1/2}$.
- Unormalized Matrix: $L = \boldsymbol{D} - \boldsymbol{W}$.

# This methods are split into two versions

**Version One**

Methods that penalize non-smoothness along the edges of a weighted graph.

**Version Two**

Methods that transfers notions of smoothness from the continuous case onto graphs as the discrete case.

# This methods are split into two versions

**Version One**

Methods that penalize non-smoothness along the edges of a weighted graph.

**Version Two**

Methods that transfers notions of smoothness from the continuous case onto graphs as the discrete case.

# Thus

## We will a two methods

1. Semi-Supervised Text Classification Using EM.
2. Transductive Support Vector Machines.

# Thus

## We will a two methods

1. Semi-Supervised Text Classification Using EM.
2. Transductive Support Vector Machines.

# Outline

# A Generative Model for Text

**We assume documents are generated by a *mixture of multinomials* model**

Each mixture component corresponds to a class.

# The Multinomial Distribution

## Definition

$(u_1, u_2, ..., u_k)$ is said to b follow a multinomial distribution with parameters $(\mathfrak{N}, p_1, p_2, ..., p_k)$.

$$p\left(x_1, x_2, ..., x_k | \mathfrak{N}, p_1, p_2, ..., p_k\right) = \frac{\mathfrak{N}!}{x_1! x_2! \cdots x_k!} p_1^{x_1} p_2^{x_2} \cdots p_k^{x_k} \tag{11}$$

# The Multinomial Distribution

## Definition

$(u_1, u_2, ..., u_k)$ is said to b follow a multinomial distribution with parameters $(\mathfrak{N}, p_1, p_2, ..., p_k)$.

$$p\left(x_1, x_2, ..., x_k | \mathfrak{N}, p_1, p_2, ..., p_k\right) = \frac{\mathfrak{N}!}{x_1! x_2! \cdots x_k!} p_1^{x_1} p_2^{x_2} \cdots p_k^{x_k} \tag{11}$$

## Under the following conditions

1. Number of trials $\mathfrak{N} > 0$.
2. $x_i \in \{0, 1, 2, ..., \mathfrak{N}\}$.
3. $\sum_{i=1}^{k} x_i = \mathfrak{N}$.
4. $\sum_{i=1}^{k} p_i = 1$.

# The Multinomial Distribution

## Definition

$(u_1, u_2, ..., u_k)$ is said to b follow a multinomial distribution with parameters $(\mathfrak{N}, p_1, p_2, ..., p_k)$.

$$p\left(x_1, x_2, ..., x_k | \mathfrak{N}, p_1, p_2, ..., p_k\right) = \frac{\mathfrak{N}!}{x_1! x_2! \cdots x_k!} p_1^{x_1} p_2^{x_2} \cdots p_k^{x_k} \tag{11}$$

## Under the following conditions

1. Number of trials $\mathfrak{N} > 0$.
2. $x_i \in \{0, 1, 2, ..., \mathfrak{N}\}$.
3. $\sum_{i=1}^{k} x_i = \mathfrak{N}$.
4. $\sum_{i=1}^{k} p_i = 1$.

# The Multinomial Distribution

## Definition

$(u_1, u_2, ..., u_k)$ is said to b follow a multinomial distribution with parameters $(\mathfrak{N}, p_1, p_2, ..., p_k)$.

$$p\left(x_1, x_2, ..., x_k | \mathfrak{N}, p_1, p_2, ..., p_k\right) = \frac{\mathfrak{N}!}{x_1! x_2! \cdots x_k!} p_1^{x_1} p_2^{x_2} \cdots p_k^{x_k} \tag{11}$$

## Under the following conditions

1. Number of trials $\mathfrak{N} > 0$.
2. $x_i \in \{0, 1, 2, ..., \mathfrak{N}\}$.
3. $\sum_{i=1}^{k} x_i = \mathfrak{N}$.
4. $\sum_{i=1}^{k} p_i = 1$.

# The Multinomial Distribution

## Definition

$(u_1, u_2, ..., u_k)$ is said to b follow a multinomial distribution with parameters $(\mathfrak{N}, p_1, p_2, ..., p_k)$.

$$p\left(x_1, x_2, ..., x_k | \mathfrak{N}, p_1, p_2, ..., p_k\right) = \frac{\mathfrak{N}!}{x_1! x_2! \cdots x_k!} p_1^{x_1} p_2^{x_2} \cdots p_k^{x_k} \tag{11}$$

## Under the following conditions

1. Number of trials $\mathfrak{N} > 0$.
2. $x_i \in \{0, 1, 2, ..., \mathfrak{N}\}$.
3. $\sum_{i=1}^{k} x_i = \mathfrak{N}$.
4. $\sum_{i=1}^{k} p_i = 1$.

# Assumptions

**Now, assume**

- $M$ is the number of classes.

# Assumptions

## Now, assume

- $M$ is the number of classes.
- And you have a vocabulary $\mathfrak{X}$ of size $|\mathfrak{X}|$.
- Each document $\mathfrak{x}_i$ has $|\mathfrak{x}_i|$ words in it.

# Assumptions

## Now, assume

- $M$ is the number of classes.
- And you have a vocabulary $\mathfrak{X}$ of size $|\mathfrak{X}|$.
- Each document $\boldsymbol{x}_i$ has $|\boldsymbol{x}_i|$ words in it.

## How do we create a document using this model?

As it stands, it looks difficult!!!

# Assumptions

## Now, assume

- $M$ is the number of classes.
- And you have a vocabulary $\mathfrak{X}$ of size $|\mathfrak{X}|$.
- Each document $\boldsymbol{x}_i$ has $|\boldsymbol{x}_i|$ words in it.

## How do we create a document using this model?

As it stands, it looks difficult!!!

# Outline

# Model

## First

We roll a biased $M$-sided die to determine the class of our document.

# Model

## First

We roll a biased $M$-sided die to determine the class of our document.

## Second

Then, we pick up the biased $|\mathfrak{X}|$-sided die that corresponds to the chosen class.

# Model

**First**

We roll a biased $M$-sided die to determine the class of our document.

**Second**

Then, we pick up the biased $|\mathfrak{X}|$-sided die that corresponds to the chosen class.

**Third**

We roll this die $|\boldsymbol{x}_i|$ times, and count how many times each word occurs.

- These word counts form the generated document.

# Model

## First

We roll a biased $M$-sided die to determine the class of our document.

## Second

Then, we pick up the biased $|\mathfrak{X}|$-sided die that corresponds to the chosen class.

## Third

We roll this die $|\boldsymbol{x}_i|$ times, and count how many times each word occurs.

- These word counts form the generated document.

# Formally

## Every document

It is generated according to a probability distribution with parameter $\theta$.

## The probability distribution for the documents

It consists of a mixture of components $c_j \in \{1, 2, ..., M\}$.

## Thus

A document $x_i$ is created by first selecting a mixture component according to the mixture weights (Class probability), $P(c_j|\theta)$.

# Formally

## Every document

It is generated according to a probability distribution with parameter $\theta$.

## The probability distribution for the documents

It consists of a mixture of components $c_j \in \{1, 2, ..., M\}$.

## Thus

A document $x_i$ is created by first selecting a mixture component according to the mixture weights (Class probability), $P(c_j|\theta)$.

# Formally

## Every document

It is generated according to a probability distribution with parameter $\theta$.

## The probability distribution for the documents

It consists of a mixture of components $c_j \in \{1, 2, ..., M\}$.

## Thus

A document $\boldsymbol{x}_i$ is created by first selecting a mixture component according to the mixture weights (Class probability), $P(c_j|\theta)$.

# Next

## We can use this mixture component

This selected mixture component to to generate a document according to its own parameters, with distribution $P(\boldsymbol{x}_i|c_j,\theta)$

## Thus the likelihood to see a particular document $\boldsymbol{x}_i$ is

$$P(\boldsymbol{x}_i|\theta) = \sum_{j\in\{1,2,\ldots,M\}} P(c_j|\theta)\,P(\boldsymbol{x}_i|c_j,\theta) \tag{12}$$

## Each document has a class label

We assume a one-to-one correspondence between mixture model components and classes.

# Next

## We can use this mixture component

This selected mixture component to to generate a document according to its own parameters, with distribution $P(\boldsymbol{x}_i|c_j,\theta)$

## Thus the likelihood to see a particular document $\boldsymbol{x}_i$ is

$$P(\boldsymbol{x}_i|\theta) = \sum_{j\in\{1,2,...,M\}} P(c_j|\theta)\, P(\boldsymbol{x}_i|c_j,\theta) \tag{12}$$

## Each document has a class label

We assume a one-to-one correspondence between mixture model components and classes.

# Next

## We can use this mixture component

This selected mixture component to to generate a document according to its own parameters, with distribution $P(\boldsymbol{x}_i|c_j,\theta)$

## Thus the likelihood to see a particular document $\boldsymbol{x}_i$ is

$$P(\boldsymbol{x}_i|\theta) = \sum_{j\in\{1,2,...,M\}} P(c_j|\theta)\, P(\boldsymbol{x}_i|c_j,\theta) \tag{12}$$

## Each document has a class label

We assume a one-to-one correspondence between mixture model components and classes.

# Now

**Thus**

We use $c_j$ to indicate the $j$th mixture component as well the $j$th class.

**Not only that**

The class label for a particular document $x_i$ is written $y_i$.

**Thus**

If document $x_i$ was generated by mixture component $c_j$ we say $y_i = c_j$.

# Now

**Thus**

We use $c_j$ to indicate the $j$th mixture component as well the $j$th class.

**Not only that**

The class label for a particular document $\boldsymbol{x}_i$ is written $y_i$.

**Thus**

If document $x_i$ was generated by mixture component $c_j$ we say $y_i = c_j$.

# Now

**Thus**

We use $c_j$ to indicate the $j$th mixture component as well the $j$th class.

**Not only that**

The class label for a particular document $\boldsymbol{x}_i$ is written $y_i$.

**Thus**

If document $x_i$ was generated by mixture component $c_j$ we say $y_i = c_j$.

# Outline

# Now

## A document, $\boldsymbol{x}_i$ , is a vector of word counts

Thus, we write $x_{it}$ to be the number of times word $w_t$ occurs in document $\boldsymbol{x}_i$.

## Length of the Document

When a document is to be generated by a particular mixture component the document length is chosen independently of the component.

$$|\boldsymbol{x}_i| = \sum_{t=1}^{|\mathbf{x}|} x_{it} \tag{13}$$

## Then

The selected mixture component is used to generate a document of the specified length, by drawing from its multinomial distribution

# Now

## A document, $\boldsymbol{x}_i$ , is a vector of word counts

Thus, we write $x_{it}$ to be the number of times word $w_t$ occurs in document $\boldsymbol{x}_i$.

## Length of the Document

When a document is to be generated by a particular mixture component the document length is chosen independently of the component.

$$|\boldsymbol{x}_i| = \sum_{t=1}^{|\mathfrak{x}|} x_{it} \tag{13}$$

## Then

The selected mixture component is used to generate a document of the specified length, by drawing from its multinomial distribution

# Now

## A document, $\boldsymbol{x}_i$ , is a vector of word counts

Thus, we write $x_{it}$ to be the number of times word $w_t$ occurs in document $\boldsymbol{x}_i$.

## Length of the Document

When a document is to be generated by a particular mixture component the document length is chosen independently of the component.

$$|\boldsymbol{x}_i| = \sum_{t=1}^{|\mathfrak{x}|} x_{it} \tag{13}$$

## Then

The selected mixture component is used to generate a document of the specified length, by drawing from its multinomial distribution

# Thus

**The probability of a document given a mixture component in terms of its constituent features**

$$P(\boldsymbol{x}_i|c_j,\theta) \propto P(|\boldsymbol{x}_i|) \prod_{w_t \in \mathfrak{X}} P(w_t|c_j,\theta)^{x_{it}} \tag{14}$$

**Under the standard naive assumption**

The words of a document are conditionally independent of the other words in the same document, given the class label.

**Thus**

The parameters of an individual mixture component define a multinomial distribution over words.

# Thus

**The probability of a document given a mixture component in terms of its constituent features**

$$P\left(\boldsymbol{x}_i|c_j,\theta\right) \propto P\left(|\boldsymbol{x}_i|\right) \prod_{w_t \in \mathfrak{X}} P\left(w_t|c_j,\theta\right)^{x_{it}} \tag{14}$$

**Under the standard naive assumption**

The words of a document are conditionally independent of the other words in the same document, given the class label.

**Thus**

The parameters of an individual mixture component define a multinomial distribution over words.

# Thus

**The probability of a document given a mixture component in terms of its constituent features**

$$P(\boldsymbol{x}_i|c_j,\theta) \propto P(|\boldsymbol{x}_i|) \prod_{w_t \in \mathfrak{X}} P(w_t|c_j,\theta)^{x_{it}} \tag{14}$$

**Under the standard naive assumption**

The words of a document are conditionally independent of the other words in the same document, given the class label.

**Thus**

The parameters of an individual mixture component define a multinomial distribution over words.

# Meaning

## The collection of word probabilities

Each written $\theta_{w_t|c_j}$ such that $\theta_{w_t|c_j} \equiv P(w_t|c_j, \theta)$ where $t \in \{1, 2, ..., |\mathfrak{X}|\}$ and $\sum_t P(w_t|c_j, \theta) = 1$.

## In addition

Since we assume that for all classes, document length is identically distributed, it does not need to be parameterized for classification.

## What else we need?

We only need the mixture weights (Class probabilities), $\theta_{c_j} \equiv P(c_j|\theta)$.

# Meaning

## The collection of word probabilities

Each written $\theta_{w_t|c_j}$ such that $\theta_{w_t|c_j} \equiv P(w_t|c_j, \theta)$ where $t \in \{1, 2, ..., |\mathfrak{X}|\}$ and $\sum_t P(w_t|c_j, \theta) = 1$.

## In addition

Since we assume that for all classes, document length is identically distributed, it does not need to be parameterized for classification.

## What else we need?

We only need the mixture weights (Class probabilities), $\theta_{c_j} \equiv P(c_j|\theta)$.

# Meaning

## The collection of word probabilities

Each written $\theta_{w_t|c_j}$ such that $\theta_{w_t|c_j} \equiv P(w_t|c_j, \theta)$ where $t \in \{1, 2, ..., |\mathfrak{X}|\}$ and $\sum_t P(w_t|c_j, \theta) = 1$.

## In addition

Since we assume that for all classes, document length is identically distributed, it does not need to be parameterized for classification.

## What else we need?

We only need the mixture weights (Class probabilities), $\theta_{c_j} \equiv P(c_j|\theta)$.

# Outline

# Thus

**The complete collection of model parameters, $\theta$, defines a set of multinomials and class probabilities**

$$\theta = \left\{ \theta_{w_t|c_j} | w_t \in \mathfrak{X}, c_j \in \{1, ..., M\}, \theta_{c_j} : c_j \in \{1, ..., M\} \right\} \tag{15}$$

**Thus, we get**

$$P(x_i|\theta) = P(|x_i|) \sum_{j \in \{1,2,...,M\}} P(c_j|\theta) \prod_{w_t \in \mathfrak{X}} P(w_t|c_j, \theta)^{n_{it}} \tag{16}$$

# Thus

**The complete collection of model parameters, $\theta$, defines a set of multinomials and class probabilities**

$$\theta = \left\{ \theta_{w_t|c_j} | w_t \in \mathfrak{X}, c_j \in \{1, ..., M\}, \theta_{c_j} : c_j \in \{1, ..., M\} \right\} \tag{15}$$

**Thus, we get**

$$P(\boldsymbol{x}_i|\theta) = P(|\boldsymbol{x}_i|) \sum_{j \in \{1,2,...,M\}} P(c_j|\theta) \prod_{w_t \in \mathfrak{X}} P(w_t|c_j, \theta)^{x_{it}} \tag{16}$$

# Outline

# Now

## Given the generative model

We need to estimate the values $\hat{\theta}$.

## For Naive Bayes

We can use MAP to find the $\arg\max_\theta P(\theta|X, Y)$ by using the likelihood of the data and a prior.

## We know that the commonly used conjugate prior distribution for multinomial distributions is the Dirichlet Prior

$$P\left(\theta_{w_t|c_j}|\alpha\right) \propto \prod_{w_t \in \mathfrak{X}} P(w_t|c_j)^{\alpha_t - 1} \tag{17}$$

Where $\alpha_t$ are constants greater than zero.

# Now

## Given the generative model

We need to estimate the values $\hat{\theta}$.

## For Naive Bayes

We can use MAP to find the $\arg\max_\theta P(\theta|X,Y)$ by using the likelihood of the data and a prior.

We know that the commonly used conjugate prior distribution for multinomial distributions is the Dirichlet Prior

$$P\left(\theta_{w_t|c_j}|\alpha\right) \propto \prod_{w_t \in \mathfrak{X}} P(w_t|c_j)^{\alpha_t - 1} \tag{17}$$

Where $\alpha_t$ are constants greater than zero.

# Now

## Given the generative model

We need to estimate the values $\hat{\theta}$.

## For Naive Bayes

We can use MAP to find the $\arg\max_\theta P(\theta|X,Y)$ by using the likelihood of the data and a prior.

## We know that the commonly used conjugate prior distribution for multinomial distributions is the Dirichlet Prior

$$P\left(\theta_{w_t|c_j}|\alpha\right) \propto \prod_{w_t \in \mathfrak{X}} P(w_t|c_j)^{\alpha_t - 1} \tag{17}$$

Where $\alpha_t$ are constants greater than zero.

# The Dirichlet Distribution

**Let $\Theta = \{\theta_1, \theta_2, ..., \theta_m\}$**

We write:

$$\Theta \sim Dirichlet\left(\alpha_1, \alpha_2, ..., \alpha_m\right) \tag{18}$$

**The pmf looks like**

$$P\left(\theta_1, \theta_2, ..., \theta_m\right) = \frac{\Gamma\left(\sum_k \alpha_k\right)}{\prod_k \Gamma\left(\alpha_k\right)} \prod_{k=1}^{m} \theta_k^{\alpha_k - 1} \tag{19}$$

# The Dirichlet Distribution

**Let $\Theta = \{\theta_1, \theta_2, ..., \theta_m\}$**

We write:

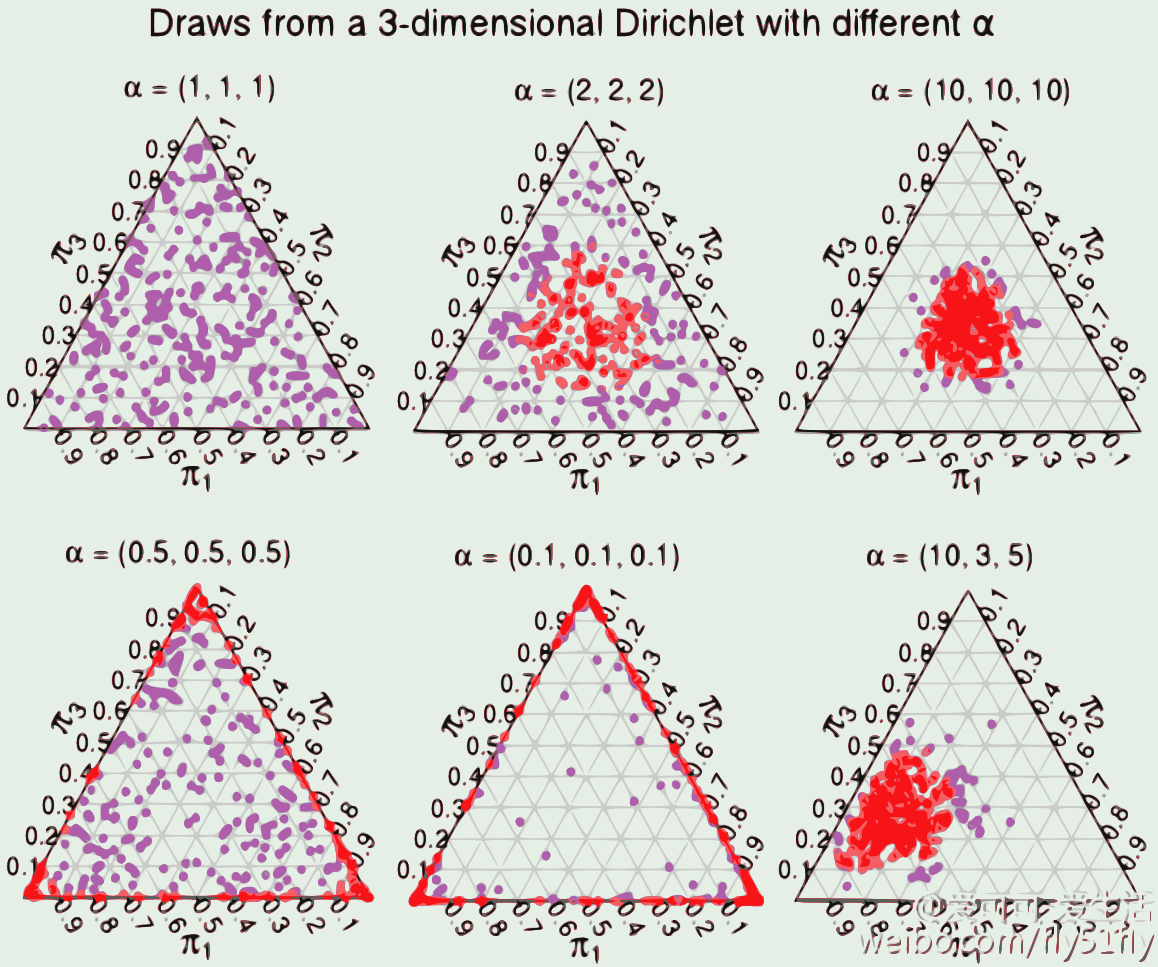
$$\Theta \sim Dirichlet\left(\alpha_1, \alpha_2, ..., \alpha_m\right) \tag{18}$$

**The pmf looks like**

$$P\left(\theta_1, \theta_2, ..., \theta_m\right) = \frac{\Gamma\left(\sum_k \alpha_k\right)}{\prod_k \Gamma\left(\alpha_k\right)} \prod_{k=1}^{m} \theta_k^{\alpha_k - 1} \tag{19}$$

# Examples

## Dirichlet distributions under different $\alpha's$



# In addition

## Dirichlet distributions have two parameters

- The scale or concentration $\sigma = \sum_t \alpha_t$.
- The base measure $(\alpha'_1, ..., \alpha'_k)$ with $\alpha'_t = \frac{\alpha_t}{\sigma}$.

# In addition

## Dirichlet distributions have two parameters

- The scale or concentration $\sigma = \sum_t \alpha_t$.
- The base measure $(\alpha'_1, ..., \alpha'_k)$ with $\alpha'_t = \frac{\alpha_t}{\sigma}$.

# Thus

**We can set all $\alpha_t = 1$**

This prior favors the uniform distribution.

**Thus**

The parameter estimation formulas that result from maximization with the data and our prior are the familiar smoothed ratios of empirical counts.

**We have that**

$$\hat{\theta}_{w_t|c_j} = P\left(w_t|c_j, \hat{\theta}\right) = \frac{1 + \sum_{x_i \in X} \delta_{ij} x_{it}}{|\mathfrak{X}| + \sum_{s=1}^{|\mathfrak{X}|} \sum_{x_i \in X} \delta_{ij} x_{is}} \tag{20}$$

# Thus

**We can set all $\alpha_t = 1$**

This prior favors the uniform distribution.

**Thus**

The parameter estimation formulas that result from maximization with the data and our prior are the familiar smoothed ratios of empirical counts.

**We have that**

$$\hat{\theta}_{w_t|c_j} = P\left(w_t|c_j, \hat{\theta}\right) = \frac{1 + \sum_{x_i \in X} \delta_{ij} x_{it}}{|\mathfrak{X}| + \sum_{s=1}^{|\mathfrak{X}|} \sum_{x_i \in X} \delta_{ij} x_{is}} \tag{20}$$

# Thus

## We can set all $\alpha_t = 1$

This prior favors the uniform distribution.

## Thus

The parameter estimation formulas that result from maximization with the data and our prior are the familiar smoothed ratios of empirical counts.

## We have that

$$\hat{\theta}_{w_t|c_j} \equiv P\left(w_t|c_j, \hat{\theta}\right) \equiv \frac{1 + \sum_{x_i \in X} \delta_{ij} x_{it}}{|\mathfrak{X}| + \sum_{s=1}^{|\mathfrak{X}|} \sum_{x_i \in X} \delta_{ij} x_{is}} \tag{20}$$

# Where

## You have that

$$\delta_{ij} = \begin{cases} 1 & \text{if } y_i = c_j \\ 0 & \text{if } y_i \neq c_j \end{cases} \tag{21}$$

## In addition

$$\hat{\theta}_{c_j} = P\left(c_j|\hat{\theta}\right) = \frac{1 + \sum_{i=1}^{|X|} \delta_{ij}}{M + |X|} \tag{22}$$

## Now

Given estimates of these parameters calculated from labeled training documents...

# Where

## You have that

$$\delta_{ij} = \begin{cases} 1 & \text{if } y_i = c_j \\ 0 & \text{if } y_i \neq c_j \end{cases} \tag{21}$$

## In addition

$$\hat{\theta}_{c_j} \equiv P\left(c_j|\hat{\theta}\right) = \frac{1 + \sum_{i=1}^{|X|} \delta_{ij}}{M + |X|} \tag{22}$$

## Now

Given estimates of these parameters calculated from labeled training documents...

# Where

## You have that

$$\delta_{ij} = \begin{cases} 1 & \text{if } y_i = c_j \\ 0 & \text{if } y_i \neq c_j \end{cases} \tag{21}$$

## In addition

$$\hat{\theta}_{c_j} \equiv P\left(c_j|\hat{\theta}\right) = \frac{1 + \sum_{i=1}^{|X|} \delta_{ij}}{M + |X|} \tag{22}$$

## Now

Given estimates of these parameters calculated from labeled training documents.

# Thus

## Something Notable

It is possible to turn the generative model backward and calculate the probability that a particular mixture component generated a given document to perform classification.

## Because

# Thus

## Something Notable

It is possible to turn the generative model backward and calculate the probability that a particular mixture component generated a given document to perform classification.

## Because

$$P\left(y_i = c_j|\boldsymbol{x}_i, \hat{\theta}\right) = \frac{P\left(c_j|\hat{\theta}\right) P\left(\boldsymbol{x}_i|c_j, \hat{\theta}\right)}{P\left(\boldsymbol{x}_i|\hat{\theta}\right)}$$

# Thus

## Something Notable

It is possible to turn the generative model backward and calculate the probability that a particular mixture component generated a given document to perform classification.

## Because

$$P\left(y_i = c_j|\boldsymbol{x}_i, \hat{\theta}\right) = \frac{P\left(c_j|\hat{\theta}\right) P\left(\boldsymbol{x}_i|c_j, \hat{\theta}\right)}{P\left(\boldsymbol{x}_i|\hat{\theta}\right)}$$

$$= \frac{P\left(c_j|\hat{\theta}\right) \prod_{w_t \in \mathfrak{x}} P\left(w_t|c_j, \hat{\theta}\right)^{x_{it}}}{\sum_{k=1}^{M} P\left(c_k|\hat{\theta}\right) \prod_{w_t \in \mathfrak{x}} P\left(w_t|c_k, \hat{\theta}\right)^{x_{it}}}$$

# Final Document classification

## Finally

If our task is to classify document $\boldsymbol{x}_i$ in some class, we take $\arg\max_j P\left(y_i = c_j | \boldsymbol{x}_i, \hat{\theta}\right)$ as such a a class.

# Outline

# What to do?

**In the semi-supervised setting with labeled and unlabeled data**

We still like to find the MAP parameter estimates.

**Given that there is no labeled data**

We do not have labeled data, thus the closed-form equations from the previous section are not applicable

**For this, we can use the EM technique**

The EM technique as applied to the case of labeled and unlabeled data with naive Bayes yields a straightforward algorithm.

# What to do?

**In the semi-supervised setting with labeled and unlabeled data**

We still like to find the MAP parameter estimates.

**Given that there is no labeled data**

We do not have labeled data, thus the closed-form equations from the previous section are not applicable

**For this, we can use the EM technique**

The EM technique as applied to the case of labeled and unlabeled data with naive Bayes yields a straightforward algorithm.

# What to do?

**In the semi-supervised setting with labeled and unlabeled data**

We still like to find the MAP parameter estimates.

**Given that there is no labeled data**

We do not have labeled data, thus the closed-form equations from the previous section are not applicable

**For this, we can use the EM technique**

The EM technique as applied to the case of labeled and unlabeled data with naive Bayes yields a straightforward algorithm.

# EM Process

## First

A naive Bayes classifier is built in the standard supervised fashion from the limited amount of labeled training data.

## Second

Then, we perform classification of the unlabeled data with the naive Bayes model.

- For this, we use the probabilities associated with each class.

## Third

We rebuild a new naive Bayes classifier using all the data, labeled and unlabeled, using the estimated class probabilities as true class label.

# EM Process

## First

A naive Bayes classifier is built in the standard supervised fashion from the limited amount of labeled training data.

## Second

Then, we perform classification of the unlabeled data with the naive Bayes model.

- For this, we use the probabilities associated with each class.

## Third

We rebuild a new naive Bayes classifier using all the data, labeled and unlabeled, using the estimated class probabilities as true class label.

# EM Process

## First

A naive Bayes classifier is built in the standard supervised fashion from the limited amount of labeled training data.

## Second

Then, we perform classification of the unlabeled data with the naive Bayes model.

- For this, we use the probabilities associated with each class.

## Third

We rebuild a new naive Bayes classifier using all the data, labeled and unlabeled, using the estimated class probabilities as true class label.

# EM Process

## Meaning

This means that the unlabeled documents are treated as several fractional documents according to these estimated class probabilities.

Iterating the process of classifying the unlabeled data and rebuilding the model

Until it converges to a stable classifier and set of labels for the data.

# EM Process

**Meaning**

This means that the unlabeled documents are treated as several fractional documents according to these estimated class probabilities.

**Iterating the process of classifying the unlabeled data and rebuilding the model**

Until it converges to a stable classifier and set of labels for the data.

# Outline

# EM Algorithm

## Basic EM algorithm for semi-supervised learning of a text classifier

Input: Collections $X_l$ of labeled documents and $X_u$ of unlabeled documents.

1. Build an initial naive Bayes classifier, with $\hat{\theta}$, from the labeled documents.
2. Use MAP to find an estimation $\hat{\theta} = \arg\max_\theta P(X_l|\theta) P(\theta)$.
3. loop while $l(\theta|X, Y)$ improves (The Log probability of all data and prior)
4. (E step) Use the current parameter, $\hat{\theta}$, to estimate component membership
5. of each unlabeled data i.e. $P\left(c_j|x_i, \hat{\theta}\right)$.
6. (M step) Re-estimate the parameter, $\hat{\theta}$, given the estimated component
7. membership of each document, $\hat{\theta} = \arg\max_\theta P(X, Y|\theta) P(\theta)$.
8. Output: A stable classifier parameter $\hat{\theta}$ used to take an unlabeled document
9. to predict a class label.

# EM Algorithm

## Basic EM algorithm for semi-supervised learning of a text classifier

Input: Collections $X_l$ of labeled documents and $X_u$ of unlabeled documents.

1. Build an initial naive Bayes classifier, with $\hat{\theta}$, from the labeled documents.
2. Use MAP to find an estimation $\hat{\theta} = \arg\max_{\theta} P(X_l|\theta) P(\theta)$.
3. loop while $l(\theta|X, Y)$ improves (The Log probability of all data and prior)
4. (E step) Use the current parameter, $\hat{\theta}$, to estimate component membership of each unlabeled data i.e. $P\left(c_j|x_i, \hat{\theta}\right)$.
5. (M step) Re-estimate the parameter, $\hat{\theta}$, given the estimated component membership of each document, $\hat{\theta} = \arg\max_{\theta} P(X, Y|\theta) P(\theta)$.
6. **Output:** A stable classifier parameter $\hat{\theta}$ used to take an unlabeled document to predict a class label.

# EM Algorithm

## Basic EM algorithm for semi-supervised learning of a text classifier

Input: Collections $X_l$ of labeled documents and $X_u$ of unlabeled documents.

1. Build an initial naive Bayes classifier, with $\hat{\theta}$, from the labeled documents.
2. Use MAP to find an estimation $\hat{\theta} = \arg\max_\theta P(X_l|\theta) P(\theta)$.
3. loop while $l(\theta|X, Y)$ improves (The Log probability of all data and prior)
4. (E step) Use the current parameter, $\hat{\theta}$, to estimate component membership of each unlabeled data i.e. $P(c_j|x_i, \hat{\theta})$.
5. (M step) Re-estimate the parameter, $\hat{\theta}$, given the estimated component membership of each document, $\hat{\theta} = \arg\max_\theta P(X, Y|\theta) P(\theta)$.
6. Output: A stable classifier parameter $\hat{\theta}$ used to take an unlabeled document to predict a class label.

# EM Algorithm

## Basic EM algorithm for semi-supervised learning of a text classifier

Input: Collections $X_l$ of labeled documents and $X_u$ of unlabeled documents.

1. Build an initial naive Bayes classifier, with $\hat{\theta}$, from the labeled documents.
2. Use MAP to find an estimation $\hat{\theta} = \arg\max_\theta P(X_l|\theta) P(\theta)$.
3. loop while $l(\theta|X, Y)$ improves (The Log probability of all data and prior)
4. (E step) Use the current parameter, $\hat{\theta}$, to estimate component membership of each unlabeled data i.e. $P(c_j|x_i, \hat{\theta})$.
5. (M step) Re-estimate the parameter, $\hat{\theta}$, given the estimated component membership of each document, $\hat{\theta} = \arg\max_\theta P(X, Y|\theta) P(\theta)$.
6. Output: A stable classifier parameter $\hat{\theta}$ used to take an unlabeled document to predict a class label.

# EM Algorithm

## Basic EM algorithm for semi-supervised learning of a text classifier

Input: Collections $X_l$ of labeled documents and $X_u$ of unlabeled documents.

1. Build an initial naive Bayes classifier, with $\hat{\theta}$, from the labeled documents.
2. Use MAP to find an estimation $\hat{\theta} = \arg\max_\theta P(X_l|\theta) P(\theta)$.
3. loop while $l(\theta|X, Y)$ improves (The Log probability of all data and prior)
4. (**E step**) **Use the current parameter, $\hat{\theta}$, to estimate component membership**
5. **of each unlabeled data i.e. $P\left(c_j|\boldsymbol{x}_i, \hat{\theta}\right)$.**
6. (M step) Re-estimate the parameter, $\hat{\theta}$, given the estimated component
7. membership of each document, $\hat{\theta} = \arg\max_\theta P(X, Y|\theta) P(\theta)$.
8. Output: A stable classifier parameter $\hat{\theta}$ used to take an unlabeled document
9. to predict a class label.

# EM Algorithm

## Basic EM algorithm for semi-supervised learning of a text classifier

Input: Collections $X_l$ of labeled documents and $X_u$ of unlabeled documents.

1. Build an initial naive Bayes classifier, with $\hat{\theta}$, from the labeled documents.
2. Use MAP to find an estimation $\hat{\theta} = \arg\max_\theta P(X_l|\theta) P(\theta)$.
3. loop while $l(\theta|X, Y)$ improves (The Log probability of all data and prior)
4. (**E step**) **Use the current parameter, $\hat{\theta}$, to estimate component membership**
5. **of each unlabeled data i.e. $P\left(c_j|x_i, \hat{\theta}\right)$.**
6. (**M step**) **Re-estimate the parameter, $\hat{\theta}$, given the estimated component**
7. **membership of each document, $\hat{\theta} = \arg\max_\theta P(X, Y|\theta) P(\theta)$.**
8. Output: A stable classifier parameter $\hat{\theta}$ used to take an unlabeled document
9. to predict a class label.

# EM Algorithm

## Basic EM algorithm for semi-supervised learning of a text classifier

Input: Collections $X_l$ of labeled documents and $X_u$ of unlabeled documents.

1. Build an initial naive Bayes classifier, with $\hat{\theta}$, from the labeled documents.
2. Use MAP to find an estimation $\hat{\theta} = \arg\max_\theta P(X_l|\theta) P(\theta)$.
3. loop while $l(\theta|X, Y)$ improves (The Log probability of all data and prior)
4. (**E step**) **Use the current parameter, $\hat{\theta}$, to estimate component membership**
5. **of each unlabeled data i.e. $P\left(c_j|\boldsymbol{x}_i, \hat{\theta}\right)$.**
6. (**M step**) **Re-estimate the parameter, $\hat{\theta}$, given the estimated component**
7. **membership of each document, $\hat{\theta} = \arg\max_\theta P(X, Y|\theta) P(\theta)$.**
8. **Output:** A stable classifier parameter $\hat{\theta}$ used to take an unlabeled document
9. to predict a class label.

# The Log Probability of all Data and Prior

## We have that

$$l(\theta|X,Y) = \log(P(\theta)) + \sum_{\boldsymbol{x}_i \in X_u} \log\left[\sum_{j\in\{1,\ldots,M\}} P(c_j|\theta)\, P(\boldsymbol{x}_i|c_j,\theta)\right] + \ldots$$
$$\sum_{\boldsymbol{x}_i \in X_l} \log\left[P(y_i = c_j|\theta)\, P(\boldsymbol{x}_i|y_i = c_j,\theta)\right]$$