Online Learning

Felipe Almeida
Rio Machine Learning Meetup August / 2016

Introduction, overview and examples

Structure

- Introduction
- Use cases
- Types of Targets
- Approaches
- Current Trends / Related Areas
- Links

Introduction

- Online Learning is generally described as doing machine learning in a *streaming* data setting, i.e. training a model in consecutive rounds

Introduction

- Online Learning is generally described as doing machine learning in a *streaming* data setting, i.e. training a model in consecutive rounds
    - At the beginning of each round the algorithm is presented with an input sample, and must perform a prediction

Introduction

- Online Learning is generally described as doing machine learning in a *streaming* data setting, i.e. training a model in consecutive rounds
    - At the beginning of each round the algorithm is presented with an input sample, and must perform a prediction
    - The algorithm verifies whether its prediction was correct or incorrect, and feeds this information back into the model, for subsequent rounds

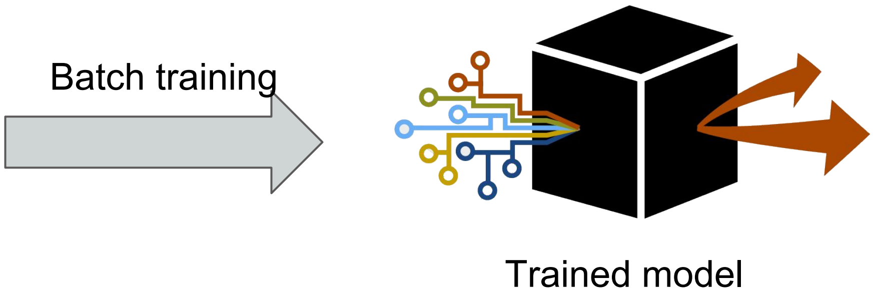
Introduction

Whereas in **batch (or offline) learning** you have access to the whole dataset to train on

D features

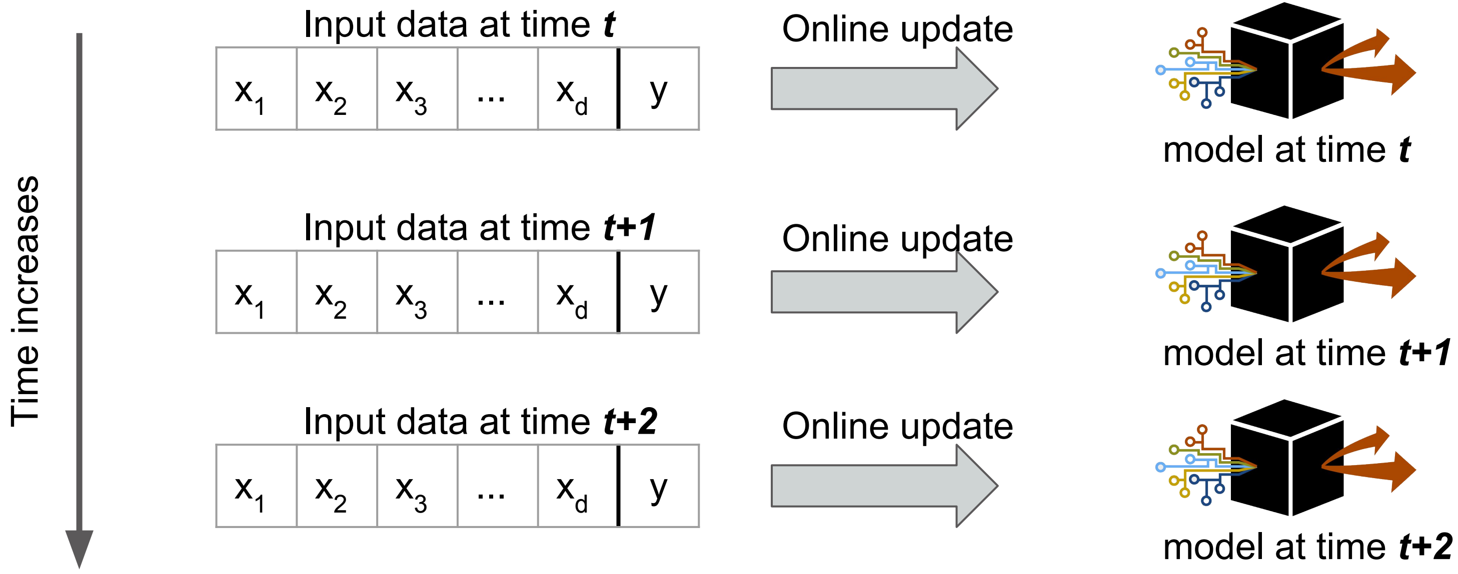
N samples

x_1	x_2	x_3	...	x_d	y
x_1	x_2	x_3	...	x_d	y
x_1	x_2	x_3	...	x_d	y
x_1	x_2	x_3	...	x_d	y
x_1	x_2	x_3	...	x_d	y



Introduction

In **online learning** your model evolves as you see new data, one example at a time



Introduction

- In other words, you need to answer a sequence of questions but you only have access to answers to previous questions

Introduction

- In other words, you need to answer a sequence of questions but you only have access to answers to previous questions

```
Online Learning
```

for $t = 1, 2, \ldots$

  receive question $\mathbf{x}_t \in \mathcal{X}$

  predict $p_t \in D$

  receive true answer $y_t \in \mathcal{Y}$

  suffer loss $l(p_t, y_t)$

Adapted from Shalev-Shwartz 2012

Introduction

- In other words, you need to answer a sequence of questions but you only have access to answers to previous questions

Online Learning

for $t = 1, 2, \ldots$
  receive question $\mathbf{x}_t \in \mathcal{X}$
  predict $p_t \in D$
  receive true answer $y_t \in \mathcal{Y}$
  suffer loss $l(p_t, y_t)$

Each t represents a *trial*

Adapted from Shalev-Shwartz 2012

Introduction

- In other words, you need to answer a sequence of questions but you only have access to answers to previous questions

Online Learning

for $t = 1, 2, \ldots$
  receive question $\mathbf{x}_t \in \mathcal{X}$
  predict $p_t \in D$
  receive true answer $y_t \in \mathcal{Y}$
  suffer loss $l(p_t, y_t)$

Each t represents a *trial*

x_t is the input data for trial *t*

Adapted from Shalev-Shwartz 2012

Introduction

- In other words, you need to answer a sequence of questions but you only have access to answers to previous questions

Online Learning

for $t = 1, 2, \ldots$
  receive question $\mathbf{x}_t \in \mathcal{X}$
  predict $p_t \in D$
  receive true answer $y_t \in \mathcal{Y}$
  suffer loss $l(p_t, y_t)$

Adapted from Shalev-Shwartz 2012

Each *t* represents a *trial*

x_t is the input data for trial *t*

p_t is the prediction for the label corresponding to x_t

Introduction

- In other words, you need to answer a sequence of questions but you only have access to answers to previous questions

Online Learning

for $t = 1, 2, \ldots$
receive question $\mathbf{x}_t \in \mathcal{X}$
predict $p_t \in D$
receive true answer $y_t \in \mathcal{Y}$
suffer loss $l(p_t, y_t)$

Adapted from Shalev-Shwartz 2012

Each *t* represents a *trial*

x_t is the input data for trial *t*

p_t is the prediction for the label corresponding to x_t

This is the difference between what you predicted and the actual label

Introduction

- In other words, you need to answer a sequence of questions but you only have access to answers to previous questions

Online Learning

for $t = 1, 2, \dots$
  receive question $\mathbf{x}_t \in \mathcal{X}$
  predict $p_t \in D$
  receive true answer $y_t \in \mathcal{Y}$
  suffer loss $l(p_t, y_t)$

Each *t* represents a *trial*

x_t is the input data for trial *t*

p_t is the prediction for the label corresponding to x_t

This is the difference between what you predicted and the actual label

After computing the loss, the algorithm will use this information to update the model generating the predictions for the next trials

Introduction: Main Concepts

The main objective of online learning algorithms is to minimize the **regret**.

Introduction: Main Concepts

The main objective of online learning algorithms is to minimize the **regret**.

The **regret** is the difference between the performance of:

- the online algorithm
- an ideal algorithm that has been able to train on the whole data seen so far, in batch fashion

Introduction: Main Concepts

In other words, the main objective of an online machine learning algorithm is to try to perform as closely to the corresponding offline algorithm as possible.

Introduction: Main Concepts

In other words, the main objective of an online machine learning algorithm is to try to perform as closely to the corresponding offline algorithm as possible.

This is measured by the **regret**.

Use cases

- Online algorithms are useful in at least two scenarios:

Use cases

- Online algorithms are useful in at least two scenarios:

  - When your data is too large to fit in the memory
    - So you need to train your model one example at a time

Use cases

- Online algorithms are useful in at least two scenarios:

  - When your data is too large to fit in the memory
    - So you need to train your model one example at a time

  - When new data is constantly being generated, and/or is dependent upon time

Use cases

Some cases where data is constantly being generated and you need quick predictions:

Use cases

Some cases where data is constantly being generated and you need quick predictions:

- Real-time Recommendation
- Fraud Detection
- Spam detection
- Portfolio Selection
- Online ad placement

Types of Targets

There are **two** main ways to think about an online learning problem, as far as the target functions (that we are trying to learn) are concerned:

Types of Targets

There are **two** main ways to think about an online learning problem, as far as the target functions (that we are trying to learn) are concerned:

- **Stationary Targets**
  - The target function you are trying to learn does **not** change over time (but may be stochastic)

Types of Targets

There are **two** main ways to think about an online learning problem, as far as the target functions (that we are trying to learn) are concerned:

- **Stationary Targets**
  - The target function you are trying to learn does **not** change over time (but may be stochastic)
- **Dynamic Targets**
  - The process that is generating input sample data is assumed to be non-stationary (i.e. may change over time)
  - The process may even be adapting to your model (i.e. in an **adversarial** manner)

Example I: Stationary Targets

For stationary targets, the input-generating process is a single, but unknown, function of the attributes.

Example I: Stationary Targets

For stationary targets, the input-generating process is a single, but unknown, function of the attributes.

- Ex.: Some process generates, at each time step *t*, inputs of the form (x_1, x_2, x_3) where each attribute is a *bit*, and the label **y** is the result of $(x_1 \vee (x_2 \wedge x_3))$:

Example I: Stationary Targets

For stationary targets, the input-generating process is a single, but unknown, function of the attributes.

- Ex.: Some process generates, at each time step *t*, inputs of the form (x_1, x_2, x_3) where each attribute is a *bit*, and the label **y** is the result of (x_1 v (x_2 ^ x_3)):

	x_1	x_2	x_3	**y**
Input at time ***t***	1	0	1	1
Input at time ***t+1***	0	1	1	1
Input at time ***t+2***	0	0	0	0
Input at time ***t+3***	1	0	0	1

Example I: Stationary Targets

For stationary targets, the input-generating process is a single, but unknown, function of the attributes.

- Ex.: Some process generates, at each time step *t*, inputs of the form (x_1, x_2, x_3) where each attribute is a *bit*, and the label **y** is the result of $(x_1 \vee (x_2 \wedge x_3))$:

	x_1	x_2	x_3	**y**
Input at time ***t***	1	0	1	1
Input at time ***t+1***	0	1	1	1
Input at time ***t+2***	0	0	0	0
Input at time ***t+3***	1	0	0	1

From the point of view of the online learning algorithm, obviously!

Example II: Dynamic Targets

Spam filtering

The objective of a good spam filter is to accurately model the following function:

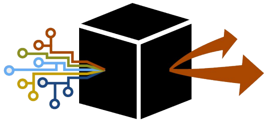
Example II: Dynamic Targets

Spam filtering

The objective of a good spam filter is to accurately model the following function:

x_1	x_2	x_3	x_4	...	x_d

$\to$



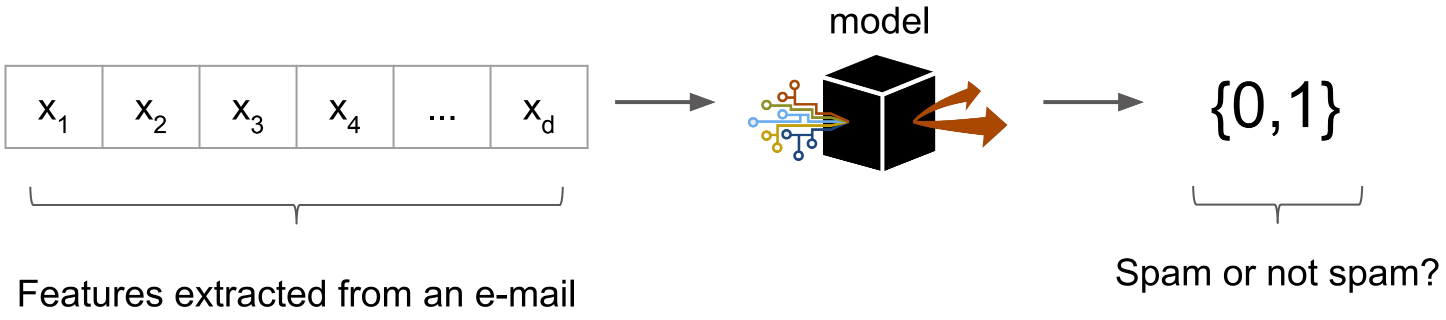
$\to$

$\{0,1\}$

Example II: Dynamic Targets

Spam filtering

The objective of a good spam filter is to accurately model the following function:



Example II: Dynamic Targets

Spam filtering

- So suppose you have learned that the presence of the word `“Dollars”` implies that an e-mail is likely spam.

Example II: Dynamic Targets

Spam filtering

- So suppose you have learned that the presence of the word “`Dollars`” implies that an e-mail is likely spam.

- Spammers have noticed that their scammy e-mails are falling prey to spam filters so they change tactics:

Example II: Dynamic Targets

Spam filtering

- So suppose you have learned that the presence of the word `“Dollars”` implies that an e-mail is likely spam.

- Spammers have noticed that their scammy e-mails are falling prey to spam filters so they change tactics:
  - So instead of using the word `“Dollars”` they start using the word `“Euro”`, which fools your filter but also accomplishes their goal (have people read the e-mail)

Approaches

A couple of approaches have been proposed in the literature:

- Online Learning from Expert Advice

Approaches

A couple of approaches have been proposed in the literature:

- Online Learning from Expert Advice
- Online Learning from Examples

Approaches

A couple of approaches have been proposed in the literature:

- Online Learning from Expert Advice
- Online Learning from Examples
- General algorithms that may also be used in the online setting

Approaches: Expert Advice

In this approach, it is assumed that the algorithm has multiple *oracles* (or experts at its disposal), which its can use to produce its output, in each trial.

Approaches: Expert Advice

In this approach, it is assumed that the algorithm has multiple *oracles* (or experts at its disposal), which its can use to produce its output, in each trial.

In other words, the task of this online algorithm is simply to learn which of the experts it should use.

Approaches: Expert Advice

In this approach, it is assumed that the algorithm has multiple *oracles* (or experts at its disposal), which its can use to produce its output, in each trial.

In other words, the task of this online algorithm is simply to learn which of the experts it should use.

The simplest algorithm in this realm is the **Randomized Weighted Majority Algorithm**

Approaches: Expert Advice

Randomized Weighted Majority Algorithm

- Every expert has a *weight* (starting at 1)

- For every trial:
  - Randomly select an expert (larger weight => more likely)
  - Use that expert's output as your prediction
  - Verify the correct answer
  - For each expert:
    - If it was mistaken, decrease its weight by a constant factor

Approaches: Learning from Examples

Learning from examples is different from using Expert Advice inasmuch as we don’t need to previously define prebuild experts we will derive our predictions from.

Approaches: Learning from Examples

Learning from examples is different from using Expert Advice inasmuch as we don’t need to previously define prebuild experts we will derive our predictions from.

We need, however, to know what **Concept Class** we want to search over.

Approaches: Learning from Examples

A **Concept Class** is a set of of functions (concepts) that subscribe to a particular model.

Approaches: Learning from Examples

A **Concept Class** is a set of of functions (concepts) that subscribe to a particular model.

Some examples of concept classes are:

- The set of all monotone disjunctions of N variables
- The set of non-monotone disjunctions of N variables
- Decision lists with N variables
- Linear threshold formulas
- DNF (disjunctive normal form) formulas

Approaches: Learning from Examples

The **Winnow Algorithm** is one example of a simple algorithm that learns monotone disjunctions online.

Approaches: Learning from Examples

The **Winnow Algorithm** is one example of a simple algorithm that learns monotone disjunctions online.

In other words, it learns any concept (function), provided the concept belongs to the **Concept Class** of monotone disjunctions.

Approaches: Learning from Examples

The **Winnow Algorithm** is one example of a simple algorithm that learns monotone disjunctions online.

In other words, it learns any concept (function), provided the concept belongs to the **Concept Class** of monotone disjunctions.

It also uses weights, as in the previous example.

Approaches: Learning from Examples

Winnow algorithm

- Initialize all weights (w_1, w_2,... w_n) to 1
- Given a new example:
  - Predict 1 if $w^T x > n$
  - Predict 0 otherwise
- Check the true answer
- For each input attribute:
  - If algorithm predicted 1 but true answer was 0, double the value of every weight corresponding to an attribute = 1
  - If algorithm predicted 0 but true answer was 1, halve the value of each weight corresponding to an attribute = 0

Approaches: Learning from Examples

Winnow algorithm

We aimed too low, let's try to make our guess higher

- Initialize all weights (w_1, w_2,... w_n) to 1
- Given a new example:
  - Predict 1 if $w^Tx > n$
  - Predict 0 otherwise
- Check the true answer
- For each input attribute:
  - If algorithm predicted 1 but true answer was 0, double the value of every weight corresponding to an attribute = 1
  - If algorithm predicted 0 but true answer was 1, halve the value of each weight corresponding to an attribute = 0

Approaches: Learning from Examples

Winnow algorithm

- Initialize all weights (w_1, w_2,... w_n) to 1
- Given a new example:
  - Predict 1 if $w^T x > n$
  - Predict 0 otherwise
- Check the true answer
- For each input attribute:
  - If algorithm predicted 1 but true answer was 0, double the value of every weight corresponding to an attribute = 1
  - If algorithm predicted 0 but true answer was 1, halve the value of each weight corresponding to an attribute = 0

We aimed too high, let's try to make our guess lower

Approaches: Other Approaches

More general algorithms can also be used in an online setting, such as:

Approaches: Other Approaches

More general algorithms can also be used in an online setting, such as:

- Stochastic Gradient Descent

Approaches: Other Approaches

More general algorithms can also be used in an online setting, such as:

- Stochastic Gradient Descent
- Perceptron Learning Algorithm

Current Trends / Related Areas

Adversarial Machine Learning

- Refers to scenarios where your input-generating process is an adaptive adversary
- Applications in:
  - Information Security
  - Games

Current Trends / Related Areas

One-shot Learning

- Refers to scenarios where your must perform predictions after seeing just a few, or even a single input sample
- Applications in:
  - Computer Vision

Links

- http://ttic.uchicago.edu/~shai/papers/ShalevThesis07.pdf
- Blum 1998 Survey Paper
- UofW CSE599S Online Learning
- Machine Learning From Streaming data
- Twitter Fighting Spam with BotMaker
- CS229 - Online Learning Lecture
- Building a real time Recommendation Engine with Data Science
- Online Optimization for Large Scale Machine Learning by prof A. Banerjee
- Learning, Regret, Minimization and Equilibria

Links

- https://github.com/JohnLangford/vowpal_wabbit
- Shai Shalev-Shwartz 2011 Survey Paper
- Hoi et al 2014 - LIBOL
- MIT 6.883 Online Methods in Machine Learning