Machine learning from scratch

Lecture 7: Classification

Alexis Zubiolo
alexis.zubiolo@gmail.com

Data Science Team Lead @ Adcash

March 16, 2017

Introduction

Classification and **regression** are the 2 main aspects of supervised machine learning.

Introduction

Classification and **regression** are the 2 main aspects of supervised machine learning.

Reminder: In classification, y takes **discrete values**: **1 value per class**. Typically, we note $y \in \{1, 2, \ldots, k\}$ where k is the number of classes.

Introduction

Classification and **regression** are the 2 main aspects of supervised machine learning.

Reminder: In classification, y takes **discrete values**: **1 value per class**. Typically, we note $y \in \{1, 2, \ldots, k\}$ where k is the number of classes.

Example:

- For digit recognition, there are **10 classes** (one per digit: 0, 1, 2, . . . , 9)
- For letter recognition, the number of classes depends on the alphabet (e.g. 26 for the latin alphabet).

Support Vector Machines

Support Vector Machines (SVM) is one of the most popular classification algorithms. What we will see in this course is just one way to approach it. Actually, it can be solved/approached in many (equivalent) ways.

Support Vector Machines

Support Vector Machines (SVM) is one of the most popular classification algorithms. What we will see in this course is just one way to approach it. Actually, it can be solved/approached in many (equivalent) ways.

As usual, we will define a dataset $\left\{(\boldsymbol{x}^{(i)}, y^{(i)}), i = 1, \ldots, n\right\}$. Here, $y^{(i)} \in \{1, \ldots, k\}$ is the class.

Support Vector Machines

Support Vector Machines (SVM) is one of the most popular classification algorithms. What we will see in this course is just one way to approach it. Actually, it can be solved/approached in many (equivalent) ways.

As usual, we will define a dataset $\left\{(\boldsymbol{x}^{(i)}, y^{(i)}), i = 1, \ldots, n\right\}$. Here, $y^{(i)} \in \{1, \ldots, k\}$ is the class.

One way to formulate the SVM model is by defining **one linear models per class**: $\theta_1, \theta_2, \ldots, \theta_k$. The goal is to have $\theta_j^T x$ measure the confidence of $\boldsymbol{x}$ having j as a label. This way, the predicted class of $\boldsymbol{x}$ will be the j that maximizes $\theta^T \boldsymbol{x}$.

Support Vector Machines

Let us consider a single sample $(\boldsymbol{x}, y)$. The SVM algorithm will define weights $\theta_1, \theta_2, \ldots, \theta_k$ for each class.

Support Vector Machines

Let us consider a single sample $(\boldsymbol{x}, y)$. The SVM algorithm will define weights $\theta_1, \theta_2, \ldots, \theta_k$ for each class.

Hence, we can define scores $s_j = \theta_j^T \boldsymbol{x}$ for each class j. If the SVM algorithm makes no mistake on the sample $(\boldsymbol{x}, y)$, s_y **should be the highest value**.

Hinge loss

SVM relies on the **hinge loss**, that is

$$\ell(s_j, s_y) = \max(0, s_j - s_y + \Delta)$$

where:

- s_j is the score for some class $j \neq y$
- s_y is the score for the true class y
- Δ is a **hyper-parameter** that quantifies by how much we want s_y to be bigger than s_j for $j \neq y$

Hinge loss

SVM relies on the **hinge loss**, that is

$$\ell(s_j, s_y) = \max(0, s_j - s_y + \Delta)$$

where:

- s_j is the score for some class $j \neq y$
- s_y is the score for the true class y
- Δ is a **hyper-parameter** that quantifies by how much we want s_y to be bigger than s_j for $j \neq y$

This loss has to be computed for all the classes **other than the actual class** y. In the end, the total loss for the sample $(\boldsymbol{x}, y)$ is the sum of the loss for all the $j \neq y$:

$$L = \sum_{j \neq y} \ell(s_j, s_y) = \sum_{j \neq y} \max(0, s_j - s_y + \Delta)$$

Illustration on a simple example

Example: Suppose we have 3 classes (1, 2 and 3). We get a training sample $(\boldsymbol{x}, y)$ within the first class (that is, $y = 0$). We have an SVM algorithm with hyperparameter $\Delta = 10$ that returns scores of 13 for class 0, -7 for class 1, and 11 for class 2.
Question: What is the total loss for this sample?

Illustration on a simple example

Example: Suppose we have 3 classes (1, 2 and 3). We get a training sample $(\boldsymbol{x}, y)$ within the first class (that is, $y = 0$). We have an SVM algorithm with hyperparameter $\Delta = 10$ that returns scores of 13 for class 0, -7 for class 1, and 11 for class 2.
Question: What is the total loss for this sample?

Solution:

$$\begin{aligned}
L &= \ell(s_1, s_0) + \ell(s_2, s_0) \\
&= \max(0, -7 - 13 + 10) + \max(0, 11 - 13 + 10) \\
&= \max(0, -10) + \max(0, +8) \\
&= 0 + 8 \\
&= 8
\end{aligned}$$

Optimization problem formulation

We saw that for a single sample $(\boldsymbol{x}, y)$, the training loss is:

$$L = \sum_{j \neq y} \ell(s_j, s_y) = \sum_{j \neq y} \max(0, s_j - s_y + \Delta)$$

where $s_j = \theta_j^T \boldsymbol{x}$ and $s_y = \theta_y^T \boldsymbol{x}$.

Optimization problem formulation

We saw that for a single sample $(\boldsymbol{x}, y)$, the training loss is:

$$L = \sum_{j \neq y} \ell(s_j, s_y) = \sum_{j \neq y} \max(0, s_j - s_y + \Delta)$$

where $s_j = \theta_j^T \boldsymbol{x}$ and $s_y = \theta_y^T \boldsymbol{x}$.

As for the linear regression, when given a training set $\left\{(\boldsymbol{x}^{(i)}, y^{(i)}), i = 1, \ldots, n\right\}$, we just need to sum the losses for all the training samples:

$$\begin{aligned} J(\theta_1, \theta_2, \ldots, \theta_k) &= \sum_{i=1}^{n} L_i \\ &= \sum_{i=1}^{n} \sum_{j \neq y_i} \max(0, \theta_j^T \boldsymbol{x}^{(i)} - \theta_{y_i}^T \boldsymbol{x}^{(i)} + \Delta) \end{aligned}$$

Overfitting and regularization

As for regression tasks, **outliers** can have a dramatic impact on the model we train. We saw that it usually resulted in high weights and that **adding a regularization term** could help.

$$R(\theta_1, \theta_2, \ldots, \theta_k) = \sum_{j=1}^{k} \|\theta_j\|_2^2$$

Overfitting and regularization

As for regression tasks, **outliers** can have a dramatic impact on the model we train. We saw that it usually resulted in high weights and that **adding a regularization term** could help.

$$R(\theta_1, \theta_2, \ldots, \theta_k) = \sum_{j=1}^{k} \|\theta_j\|_2^2$$

We can do the same for the SVM problem:

$$J(\theta_1, \theta_2, \ldots, \theta_k) = \sum_{i=1}^{n} \sum_{j \neq y_i} \max(0, \theta_j^T \boldsymbol{x}^{(i)} - \theta_{y_i}^T \boldsymbol{x}^{(i)} + \Delta) + \lambda \sum_{j=1}^{k} \|\theta_j\|_2^2$$

where

$$\|\theta_j\|_2^2 = \sum_{m=1}^{d} \theta_{j,m}^2$$

(θ_j is a vector, and $\theta_{j,m}$ its mth value.)

Optimization

We have defined a cost function J we would like to minimize. Again, we can apply an algorithm like gradient descent to find its optimal value. For this, we need to compute J's **gradients**.

Optimization

We have defined a cost function J we would like to minimize. Again, we can apply an algorithm like gradient descent to find its optimal value. For this, we need to compute J's **gradients**.

In this case, there will be **several gradients** (1 per class because we have 1 θ_j vector per class). We will note $\nabla_{\theta_j} J$ the gradient with respect to θ_j.

Optimization

We have defined a cost function J we would like to minimize. Again, we can apply an algorithm like gradient descent to find its optimal value. For this, we need to compute J's **gradients**.

In this case, there will be **several gradients** (1 per class because we have 1 θ_j vector per class). We will note $\nabla_{\theta_j} J$ the gradient with respect to θ_j.

Due to the gradient linearity, we can compute the gradient of each component **separately**. Hence, we need to compute:

- L's gradients
- R's gradients

L's gradients

Reminder: For a sample $(\boldsymbol{x}, y)$, L is given by:

$$L = \sum_{j \neq y} \max(0, \theta_j^T \boldsymbol{x} - \theta_y^T \boldsymbol{x} + \Delta)$$

L's gradients

Reminder: For a sample $(\boldsymbol{x}, y)$, L is given by:

$$L = \sum_{j \neq y} \max(0, \theta_j^T \boldsymbol{x} - \theta_y^T \boldsymbol{x} + \Delta)$$

There are 2 different cases:

L's gradients

Reminder: For a sample $(\boldsymbol{x}, y)$, L is given by:

$$L = \sum_{j \neq y} \max(0, \theta_j^T \boldsymbol{x} - \theta_y^T \boldsymbol{x} + \Delta)$$

There are 2 different cases:

- $j \neq y$:

$$\nabla_{\theta_j} L = \begin{cases} \boldsymbol{x} & \text{if} \quad \theta_j^T \boldsymbol{x} - \theta_y^T \boldsymbol{x} + \Delta > 0 \\ 0 & \text{otherwise.} \end{cases}$$

L's gradients

Reminder: For a sample $(\boldsymbol{x}, y)$, L is given by:

$$L = \sum_{j \neq y} \max(0, \theta_j^T \boldsymbol{x} - \theta_y^T \boldsymbol{x} + \Delta)$$

There are 2 different cases:

- $j \neq y$:

$$\nabla_{\theta_j} L = \begin{cases} \boldsymbol{x} & \text{if} \quad \theta_j^T \boldsymbol{x} - \theta_y^T \boldsymbol{x} + \Delta > 0 \\ 0 & \text{otherwise.} \end{cases}$$

- $j = y$:

$$\nabla_{\theta_j} L = p\boldsymbol{x}$$

  where p is the **number of times the desired margin is not satisfied**, that is the number of $j \neq y$ such that

$$\theta_j^T \boldsymbol{x} - \theta_y^T \boldsymbol{x} + \Delta > 0$$

R's gradients

Reminder: R is given by:

$$R(\theta_1, \theta_2, \ldots, \theta_k) = \sum_{j=1}^{k} \|\theta_j\|_2^2$$

Question: What is $\nabla_{\theta_j} R$?

R's gradients

Reminder: R is given by:

$$R(\theta_1, \theta_2, \ldots, \theta_k) = \sum_{j=1}^{k} \|\theta_j\|_2^2$$

Question: What is $\nabla_{\theta_j} R$?

Answer:

$$\nabla_{\theta_j} R = 2\theta_j$$

R's gradients

Reminder: R is given by:

$$R(\theta_1, \theta_2, \ldots, \theta_k) = \sum_{j=1}^{k} \|\theta_j\|_2^2$$

Question: What is $\nabla_{\theta_j} R$?

Answer:

$$\nabla_{\theta_j} R = 2\theta_j$$

Conclusion: We have the analytic expression of the gradients of L and R and are able to apply the gradient descent (stochastic or batch) to it.

Conclusion

In this lecture, we've seen Classification models with the example of SVMs:

- How to define the problem
- How to compute the gradient
- How to train an SVM model for multiclass classification

Conclusion

In this lecture, we've seen Classification models with the example of SVMs:

- How to define the problem
- How to compute the gradient
- How to train an SVM model for multiclass classification

Next time, we'll go a bit further and have a **practical session** about classification.

Thank you! Questions?