Artificial Intelligence and Automation
Training Linear Models

Ph.D. Gerardo Marx Chávez-Campos

Instituto Tecnológico de Morelia: Ing. Mecatrónica

Introduction

Summary

- The Classification/Prediction task is made by **a function that converts** some input in a desired output
- **Error** is the main measure used to determine if our Classification/Prediction task is good
- A problem with the model's adjustments is that the **model is updated** to match the last training example, discarding all previous training examples.
- A good way to fix this is to moderate the updates with a learning rate (α); thus, no single training example totally dominates the learning.

Introduction

For now Machine Learning model and their training are black boxes for now. In this Lecture, we will start by looking at the **Linear Regression model**, one of the simplest models. Thus, we will discuss two different ways to train it:

- using a direct "closed-form" equation that directly computes the model parameters that best fit the model to the training set.
- Using an iterative optimization approach called Gradient Descent (GD) that gradually tweaks the model parameters to minimize the cost function over the training set.

Next, we will look at **Polynomial Regression**, a more complex model that can fit non-linear datasets.

Finally, we will look at two more models that are commonly used for classification tasks: **Logistic Regression** and **Softmax Regression**.

Linear Regression I

In the first laboratory session, we develop a simple regression model of *life satisfaction*:

$$\text{lifeSatis} = \theta_0 + \theta_1 \times \text{GPDperCapita} \quad (1)$$

here θ_0 and θ_1 are the model parameters.

Linear Regression II

More generally, a linear model makes a prediction by simply computing a weighted sum of the input features plus a constant called the bias term (intercept term):

$$\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \cdots + \theta_n x_n \quad (2)$$

with $\hat{y}$ as the predicted value and

- n is the number of features
- x_i is the i^{th} feature
- θ_j as the j^{th} model parameter

Vectorized form

A vectorized form of the **Linear Regressor** is:

$$\hat{y} = h_\theta(x) = \boldsymbol{\theta} \cdot \boldsymbol{x} \tag{3}$$

- $\boldsymbol{\theta}$ is the model's *parameter vector*
- $\boldsymbol{x}$ is the instances's *feature vector*, containing x_0 to x_1, with $x_0 = 1$
- $\boldsymbol{\theta} \cdot \boldsymbol{x}$ is the dot product $\theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \cdots + \theta_n x_n$
- h_θ is hypothesis function, using the model parameter $\boldsymbol{\theta}$

How do we train it?

- Training a model means **setting its parameters** that best fits the training set.
- We need a measure to determine how well (or poorly) the model fits the data
- The **Root Mean Square Error** (RMSE) is the most common measure
- To train the LR Model, you need to find $\boldsymbol{\theta}$ that minimizes the RMSE

The MSE Cost Function

The **Mean Square Error** (MSE) of a Linear Regression hypothesis h_θ on a training set $\boldsymbol{X}$ is calculated using:

$$MSE(\boldsymbol{X}, h_\theta) = \frac{1}{m} \sum_{i=0}^{m} \left(\boldsymbol{\theta} \boldsymbol{x}^{(i)} - y^{(i)} \right)^2 \qquad (4)$$

$$J(\theta) = MSE(\boldsymbol{X}, h_\theta) \qquad (5)$$

The Normal Equation I

To find the value of $\boldsymbol{\theta}$ that minimizes the cost function $J(\theta)$, there is a closed -form solution— in other words, a mathematical equation that gives the result directly. This is called the **Normal Equation**:

$$
\begin{aligned}
\frac{\partial J(\theta)}{\partial \theta} &= 0 \\
\frac{\partial J(\theta)}{\partial \theta} &= \frac{1}{m} \sum_{i=1}^{m} (\boldsymbol{\theta x} - y)^2 = (\boldsymbol{\theta x} - y)^T (\boldsymbol{\theta x} - y) \\
&= \left[(\boldsymbol{\theta x})^T - y^T \right] [\boldsymbol{\theta x} - y]
\end{aligned}
$$

The Normal Equation II

$$
\begin{aligned}
\frac{\partial J(\theta)}{\partial \theta} &= 0 \\
\frac{\partial J(\theta)}{\partial \theta} &= \frac{\partial}{\partial \theta}\left(\boldsymbol{\theta x} - y\right)^2 = \frac{\partial}{\partial \theta}\left(\boldsymbol{\theta x} - y\right)^T\left(\boldsymbol{\theta x} - y\right) \\
&= \frac{\partial}{\partial \theta}\left[(\boldsymbol{\theta x})^T - y^T\right]\left[\boldsymbol{\theta x} - y\right]
\end{aligned}
$$

The Normal Equation III

Theorem. The following properties hold:

$$
\begin{aligned}
(A^T)^T &= A \\
(A+B)^T &= A^T + B^T \\
(kA)^T &= kA^T \\
(AB)^T &= A^T B^T
\end{aligned}
$$

The Normal Equation IV

just considers that $(\boldsymbol{\theta x})^T y = y^T(\boldsymbol{\theta x})$

$$
\begin{aligned}
0 =& \frac{\partial}{\partial \theta}\left[(\boldsymbol{\theta x})^T \boldsymbol{\theta x} - (\boldsymbol{\theta x})^T y - y^T \boldsymbol{\theta x} + y^T y\right] \\
0 =& \frac{\partial}{\partial \theta}\left[\boldsymbol{\theta}^T \boldsymbol{x}^T \boldsymbol{\theta x} - 2(\boldsymbol{\theta x})^T y + y^T y\right] \\
0 =& \frac{\partial}{\partial \theta}\left[\boldsymbol{\theta}^2 \boldsymbol{x}^T \boldsymbol{x} - 2(\boldsymbol{\theta}^T \boldsymbol{x}^T) y\right] \\
0 =& 2\boldsymbol{\theta x}^T \boldsymbol{x} - 2(\boldsymbol{x}^T) y \\
2\boldsymbol{\theta x}^T \boldsymbol{x} =& 2\boldsymbol{x}^T y \\
\boldsymbol{\theta x}^T \boldsymbol{x} =& \boldsymbol{x}^T y \\
\boldsymbol{\theta} =& (\boldsymbol{x}^T \boldsymbol{x})^{-1}(\boldsymbol{x}^T y) \\
\hat{\boldsymbol{\theta}} =& (\boldsymbol{x}^T \boldsymbol{x})^{-1}(\boldsymbol{x}^T y)
\end{aligned}
$$

Referencias

- https://www.geeksforgeeks.org/ml-normal-equation-in-linear-regression/
- https://prutor.ai/normal-equation-in-linear-regression/
- https://towardsdatascience.com/performing-linear-regression-using-the-normal-equation-6372ed3c57
- Géron, Aurélien. "Hands-on machine learning with scikit-learn and tensorflow: Concepts." Tools, and Techniques to build intelligent systems (2017).