Apprentissage et Reconnaissance des Formes

[4I802]

Rapport du projet

RÉALISÉ PAR
BIZZOZZÉRO NICOLAS
&
ADOUM ROBERT

Table des matières

1 Préambule : Régression linéaire, régression ridge et LASSO

1.1 Calculs préliminaires

Soient f_w la fonction de prédiction et $\hat{y}$ l'ensemble des vrais labels de la base d'apprentissage.

1.1.1 Régularisation L2

On souhaite minimiser la fonction de coût suivante :

$$L_2(w) = \frac{1}{2N}\sum_{i=1}^{N}\left(\hat{y}_i - f_w(x_i)\right)^2 + \alpha||w||_2^2$$

On va donc l'optimiser par descente de gradient, en utilisant le gradient suivant :

$$\frac{\partial L_2}{\partial w} = \frac{1}{N}\sum_{i=1}^{N}\left(\hat{y}_i - f_w(x_i)\right)x_i + 2\alpha w$$

1.1.2 Régularisation L1

On souhaite minimiser la fonction de coût suivante :

$$L_1(w) = \frac{1}{2N}\sum_{i=1}^{N}\left(\hat{y}_i - f_w(x_i)\right)^2 + \alpha||w||_1$$

On va donc l'optimiser par descente de gradient, en utilisant le gradient suivant :

$$\frac{\partial L_1}{\partial w} = \frac{1}{N}\sum_{i=1}^{N}\left(\hat{y}_i - f_w(x_i)\right)x_i + \alpha \cdot sign(w)$$

1.2 Description du protocole

Le classifieur utilisé est, comme décrit dans l'énoncé, une simple **régression linéaire** adaptée à la classification binaire par la méthode du **plug-in**. Nous comparerons différentes fonctions de coût : **MSE**, **régularisation** L_2 et **régularisation** L_1, ainsi que plusieurs valeurs du coefficient α. De plus, nous comparons aussi nos résultats avec les implémentations *LinearRegression* et *Lasso* de la bibliothèque **sklearn** et adaptées par la *méthode du plug-in*.

La base de données **USPS** étant déjà séparée en une base d'apprentissage et une base de test, nous pouvons facilement calculer le score obtenu sur cette même base de test. Nous proposons de comparer les résultats obtenus en classifiant **chaque classe contre chaque autre** dans un premier temps, puis dans du **1 contre tous** par la suite. Enfin, nous étudierons l'apparence du vecteur de poids obtenu par chaque méthode de classification en comptant le **nombre de poids nuls** ainsi que la **moyenne de la valeur absolue des poids**.

1.3 Analyse des résultats

Classifieur	Score moyen	Nombre de 0	moyenne de $\|w\|$
Régression linéaire plug-in, MSE	0.5290	0	0.25045
Régression linéaire plug-in, L_2**,** $\alpha = 0.00$	0.4853	0	0.24862
Régression linéaire plug-in, L_2**,** $\alpha = 0.25$	0.5179	0	0.00285
Régression linéaire plug-in, L_2**,** $\alpha = 0.50$	0.5256	0	0.00308
Régression linéaire plug-in, L_2**,** $\alpha = 0.75$	0.5276	0	0.00265
Régression linéaire plug-in, L_2**,** $\alpha = 1.00$	0.5095	0	0.00258
Régression linéaire plug-in, L_1**,** $\alpha = 0.00$	0.5638	0	0.24974
Régression linéaire plug-in, L_1**,** $\alpha = 0.25$	0.5179	0	0.00387
Régression linéaire plug-in, L_1**,** $\alpha = 0.50$	0.5016	0	0.00473
Régression linéaire plug-in, L_1**,** $\alpha = 0.75$	0.5225	0	0.00573
Régression linéaire plug-in, L_1**,** $\alpha = 1.00$	0.5229	0	0.00642
sklearn.LinearRegression plug-in, $\alpha = 1.00$	0.9732	17	99305970.4
sklearn.Lasso plug-in, $\alpha = 0.10$	0.9535	10832	0.00444
sklearn.Lasso plug-in, $\alpha = 1.00$	0.5607	11520	0.00000

Résultats obtenus pour chaque classifieur sur la base **USPS** en **classe contre classe**

Classifieur	Score moyen	Nombre de 0	moyenne de $\|w\|$
Régression linéaire plug-in, MSE	0.8022	0	0.25628
Régression linéaire plug-in, L_2**,** $\alpha = 0.00$	0.8089	0	0.24960
Régression linéaire plug-in, L_2**,** $\alpha = 0.25$	0.7428	0	0.00421
Régression linéaire plug-in, L_2**,** $\alpha = 0.50$	0.7517	0	0.00317
Régression linéaire plug-in, L_2**,** $\alpha = 0.75$	0.9000	0	0.00512
Régression linéaire plug-in, L_2**,** $\alpha = 1.00$	0.8263	0	0.00279
Régression linéaire plug-in, L_1**,** $\alpha = 0.00$	0.8152	0	0.25186
Régression linéaire plug-in, L_1**,** $\alpha = 0.25$	0.7354	0	0.00524
Régression linéaire plug-in, L_1**,** $\alpha = 0.50$	0.6530	0	0.00643
Régression linéaire plug-in, L_1**,** $\alpha = 0.75$	0.5762	0	0.00547
Régression linéaire plug-in, L_1**,** $\alpha = 1.00$	0.5982	0	0.00702
sklearn.LinearRegression plug-in, $\alpha = 1.00$	0.9690	0	0.03485
sklearn.Lasso plug-in, $\alpha = 0.10$	0.9204	2452	0.00125
sklearn.Lasso plug-in, $\alpha = 1.00$	0.9000	2560	0.00000

Résultats obtenus pour chaque classifieur sur la base **USPS** en **1 contre tous**

On remarque que nos régressions appliquées à la classification par la méthode **plug-in** sont peu efficaces. Celles de *sklearn* en revanche obtiennent un score correct. De plus, toujours sur les implémenta-

tions de *sklearn*, nous voyons que le **Lasso** augmente énormément le nombre de poids nuls et fait que les poids non-nuls se rapprochent beaucoup plus de 0 qu'avec une simple régression linéaire. Nous en déduisons que le **Lasso** permet d'obtenir un vecteur de poids très sparse, utile lorsqu'on ne veut prendre en compte que quelques dimensions.

2 LASSO et Inpainting

2.1 Introduction

2.1.1 Principe

Le principe de l'**Inpainting** est qu'une partie d'une image, un **patch**, peut être approximé par une **combinaison linéaire d'autres patchs** de l'image. Cela permet ainsi de pouvoir restituer une partie manquante d'une image, de la débruiter, ou encore de supprimer de plus larges objets (défauts du visage, touristes, ...).

2.1.2 Déroulement

2.1.3 Applications

Marche bien pour recouvrir de larges zones. Permet de retirer des objets dans un but artistique (touristes sur une photo, défaut sur un visage).