Apprentissage automatique

Lecture 3: Régression logistique

Département Génie Informatique, FST de Tanger

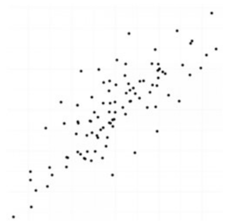
M. AIT KBIR

Plan

- Introduction à la science des données
- Notions d'apprentissage statistique
- Bayes Naïf
- Régression logistique
- Arbres de décision
- K-means
- …
- Labs (NoteBook Jupyter)
- 2 Devoir
- 1 CC

Régression linéaire

Réaliser une analyse de régression d'une variable dépendante y et d'une variable explicative x, consiste tout simplement d'ajuster, le mieux possible, une droite parmi les données. Exemple 2D, points du plan (x, y).



Régression linéaire

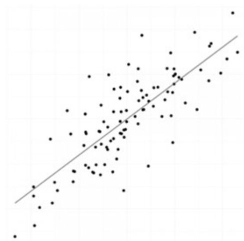
La solution par la méthode des moindres carrés consiste à minimiser la somme suivante:

$$\sum_{i=1}^{N} (y_i - w_0 - w_1 x_i)^2$$

$$w_1 = \frac{\overline{xy} - \bar{x}\bar{y}}{\overline{x^2} - \bar{x}^2} = \frac{cov(x,y)}{var(x)}$$

$$w_0 = \bar{y} - w_1 \bar{x}$$

$$\bar{x} = {\sum_i x_i}/_{N} \quad \bar{y} = {\sum_i y_i}/_{N}$$



Classifieurs géneratifs vs. Discriminatifs

Classifieur génératif, exemple de Bayes naïf:

- Supposer une forme fonctionnelle pour $P(X|Y)$, $P(Y)$
- Estimer les paramètres de $P(X|Y)$, $P(Y)$ à partir de la base des exemples d'apprentissage
- Utiliser la loi de Bayes pour calculer $P(Y|X)$

Classifieur discriminatif, exemple de la régression logistique:

- Supposer une forme fonctionnelle pour $P(Y|X)$
- Estimer les paramètres de $P(Y|X)$ à partir de la base des exemples d'apprentissage

Bayes Naïf

- On fourni:
  - Probabilités à priori $p(y)$
  - Pour chaque attribut x_i on dispose de $p(x_i|y)$
- Loi de Bayes:

$$y^* = \frac{\arg max}{y}\{p(y)p(x_1, x_2, \dots, x_M|y)\}$$

$$= \frac{\arg max}{y}\{\prod_i^M p(x_i|y)p(y)\}$$

Régression Logistique

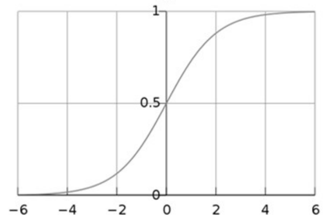
- Soit X les données d'apprentissage et Y les classes d'appartenance (0 ou 1).

$$w = (w_0, w_1, \dots, w_M)^T, X = (1, x_1, x_2 \dots, x_M)^T$$

On cherche à apprendre $p(Y|X)$ à partir des données.

Sigmoid function

$$p(Y = 1|X) = \frac{1}{1 + e^{-w^T X}}$$



Logistic Regression

- Soit $p_w(Y|X)$ l'estimation de $p(Y|X)$ avec le vecteur des paramètres w. On remarque :

$$\log\left(\frac{p_w(Y = 1|X)}{p_w(Y = 0|X)}\right) = w^T X \quad \text{(Fonction linéaire de } X\text{)}$$

Le modèle de régression logistique découle de la volonté de modéliser la probabilité à posteriori d'une classes avec une fonctions linéaire de X.

- Comment trouver w?

Choisir les paramètres maximisent la fonction de vraisemblance suivante:

$$w^* = \frac{\arg max}{w}\left\{\prod_i^N p_w(Y^i \mid X^i)\right\}$$

Ou encore:

$$w^* = \frac{\arg max}{w}\left\{\sum_{i=1}^N log(p_w(Y^i \mid X^i))\right\}$$

Optimisation par la montée du gradient

On utilise l'optimisation par la technique de monté du gradient pour calculer les poids w. On commence par des poids initiaux, puis on met à jour de manière répétitive les poids dans la direction du gradient positif, selon la règle de mise à jour suivante:

$$w(t+1) = w(t) + \alpha \sum_{i} X^i (Y^i - \hat{p}_w(Y = 1 | X^i))$$

Régression logistique avec régularisation

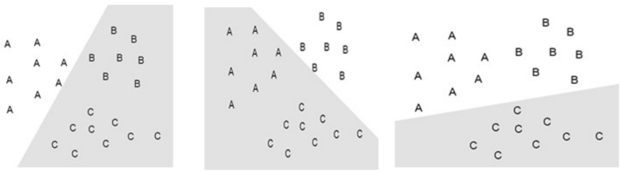
Pour éviter le sur-apprentissage surtout avec une grande dimensionnalité on peut maximiser la variante suivante du critère qui pénalise les grandes valeurs de w.

$$w^* = \frac{\arg max}{w} \left\{ \sum_{i=1}^{N} log(p_w(Y^i \mid X^i)) \right\} - \frac{\lambda}{2} \|w\|^2$$

$$w(t+1) = w(t) + \alpha \sum_i X^i (Y^i - \hat{p}_w(Y = 1 | X^i)) \text{- } \alpha \lambda w(t)$$

Régression logistique: Plusieurs classes

Pour un problème à plusieurs classes on peut adopter une stratégie dite un contre tous. Où, dans chaque étape on applique la régression logistique pour extraire une classe.



Résumé

- En général la différence entre Bayes naïf et la régression logistique, on suppose :
  - BN: attributs indépendants → $P(X|Y)$
  - RL: Forme connu de $P(Y|X)$, aucune sur $P(X|Y)$
- RL est un classifieur linéaire:
  - Hyperplan comme frontière
- RL se base sur l'optimisation de la vraisemblance conditionnelle par la technique de monté du gradient:
  - Concavité permet un maximum global.

Régression logistique : Classification des Iris

```python
import numpy as np
import matplotlib.pyplot as plt
from sklearn import datasets
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
iris = datasets.load_iris()

X = iris.data[:, :]; Y = iris.target

# Entrainement
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.2, random_state=42)
model = LogisticRegression(max_iter=600)
model.fit(X_train, y_train)

#Généralisation
y_pred = model.predict(X_test)

# Calcul des probalilités
model.predict_proba (X_test)
```

```
array([[3.78358899e-03, 8.27188710e-01, 1.69027701e-01],
       [9.46705321e-01, 5.32944791e-02, 2.00096212e-07],
       [8.71887948e-09, 1.55742200e-03, 9.98442569e-01],
       [6.43118193e-03, 7.92114071e-01, 2.01454747e-01],
       [1.44027006e-03, 7.74330539e-01, 2.24229191e-01],
       [9.55766730e-01, 4.42330929e-02, 1.76938399e-07],
       [7.75947735e-02, 9.08094963e-01, 1.43102636e-02],
       [1.61308470e-04, 1.55730935e-01, 8.44107756e-01],
       [2.20735286e-03, 7.62727710e-01, 2.35064937e-01],
       [2.83112843e-02, 9.45792578e-01, 2.58961376e-02],
       [4.39374992e-04, 2.43340887e-01, 7.56219738e-01],
       [9.68318132e-01, 3.16817900e-02, 7.81058015e-08],
       [9.72932553e-01, 2.70674132e-02, 3.33529973e-08],
       [9.62104872e-01, 3.78950166e-02, 1.10981334e-07],
        …
       [9.66848470e-01, 3.31513941e-02, 1.35862948e-07],
       [9.56312125e-01, 4.36876426e-02, 2.32547789e-07]])
```


Régression logistique

Voir le livre de Peter Harrington,
chapitre 5 "Logistic regression", page 83.