seance_6_mars

July 2, 2023

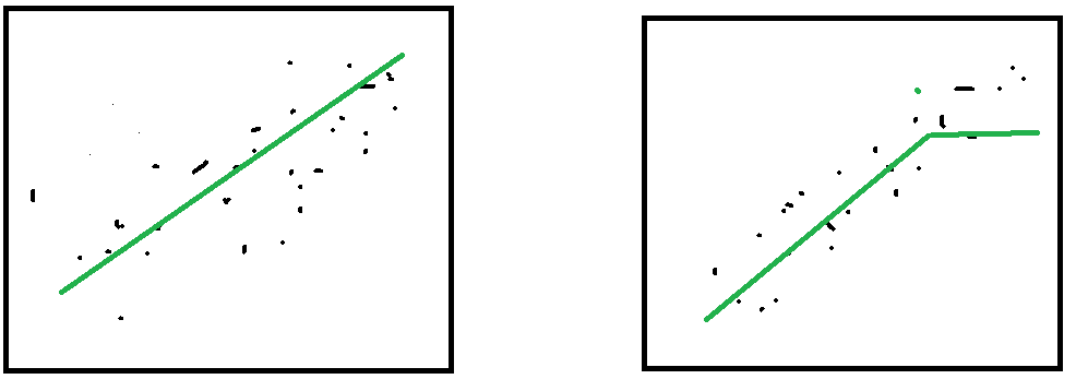
1 Questions séance 6 mars 2023

1.1 régression

Axe des X: valeur attendue, axe des Y: valeur prédites. Premier graphe : il y a 3 mois, second graphe : aujourd'hui
On mesure les performances d'un modèle après quelques mois sur de nouvelles données. Les performances se sont dégradées ? Pourquoi ?

[1]:
```
from IPython.display import Image
Image("seance_6_mars.png")
```

[1]:



Les données évoluent avec le temps. On peut supposer qu'elles croissent avec le temps. Le modèle de prédiction est un modèle borné (arbre de décision par exemple) et sa prédiction ne peut excéder un certain seuil. Il faut soit enlever la tendance dans les données soit mettre le modèle à jour très régulièrement.

1.2 classification

Un problème de classification, la classe A représente 90% des observations, la classe B, 10%. Au delà de quel seuil, pouvez-vous dire qu'un modèle de classification aura de bonnes performances ?
Le plus simple des classifieurs retourne classe tous les éléments dans la classe majoritaire. SA performance ne peut être inférieur à 90% et pourtant il ne prédit pas. Un classifieur doit faire plus de 90% de bonnes classifications pour espérer avoir appris quelque chose.

1.3 le bug de l'an 2000

Un datascientist remarque d'un modèle de prédiction est meilleur si on ajoute une colonne contenant l'année. Cela ne lui pose aucun problème. Il garde la colonne. Le 31 décembre, il réveillonne. En janvier, on mesure à nouveau les performances… Que va-t-on observer ?
L'année ne devrait jamais être utilisée comme variable à l'inverse du mois ou du jour de la semaine. L'année est une donnée qui n'existe que pendant un an et ne revient jamais. Tout apprentissage s'appuie sur le fait que des événements reviennent plusieurs fois au cours du temps. Au lieu de cela, il vaut mieux chercher quelles variables évoluent avec le temps.

1.4 algorithme

Une base d'apprentissage (X_i, y_i) avec $X_i = (x_{i1}, ..., x_{iC})$, i allant de 1 à N, et un modèle f. On fixe i, on choisit une variable c, il faut calculer :

$$\hat{y_{jc}} = \frac{1}{N} \sum_j f(x_{i1}, ..., x_{jc}, ..., x_{iC})$$

[2]:
```
import numpy

def f(x):
    return x.mean()

X = numpy.arange(100).reshape((-1, 2))


def yjc(X, i, c):
    total = 0
    for j in range(X.shape[0]):
        x = X[i].copy()
        x[c] = X[j, c]
        v = f(x)
        total += v

    return total / X.shape[0]


yjc(X, 5, 1)
```

[2]:
```
30.0
```

[3]:
```
```