fairness

January 19, 2024

1 SIN.06023 L'Apprentissage Automatique

1.0.1 Semestre de printemps 2025

1.0.2 Matthew Vowels

(PhD Eng., PhD Appl. Math, MS, MSc, BMus (hons))

Slides available at www.github.com/matthewvowels1/SIN06023

1.0.3 Calendrier des cours

Semaine 1: Révision des concepts clés de l'algèbre linéaire, de la théorie des probabilités et de l'optimisation

Semaine 2: Algorithmes d'apprentissage, capacité et biais-variance, et 'maximum likelihood estimation'

Semaine 3: Répartition et validation train/test, k-fold, optimisation des hyperparamètres

Semaine 4: Apprentissage supervisé et non-supervisé

Semaine 5: Machines vectorielles de support

Semaine 6: Arbres de décision et forêts aléatoires

Semaine 7: Réseaux de neurone: Rétropropagation, règle de chaîne et optimisation stochastique

Semaine 8: Réseaux de neurones

Semaine 9: Réseaux convolutifs

Semaine 10: Biais, équité, et justice dans l'apprentissage automatique

Semaine 11: Apprentissage automatique explicable (importances, SHAP, LIME) et calibration

Semaine 12 : Récapitulation

2 Semaine 10: Biais, équité, et justice dans l'apprentissage automatique

Comme toujours, vous devez accéder au contenu du notebook Jupyter depuis le référentiel et l'ouvrir dans Google Colab (ou localement si vous le souhaitez).

2.0.1 Resources

Solon Barocas, Moritz Hardt, Arvind Narayanan. Fairness and Machine Learning: Limitations and Opportunities https://fairmlbook.org

Meredith Broussard. Artificial Unintelligence: How Computers Misunderstand the World

Keynotes/Tutoriels par Joy Boulamwini, Kate Crawford, Meredith Whitaker et d'autres.

Le contenu de cette conférence est inspiré de la discussion de Moritz Hardt des biais : https://www.youtube.com/watch?v=Igq_S_7IfOU

2.0.2 Qu'est-ce que c'est qu'un biais ?

Le terme 'biais' se réfère généralement à une erreur systématique ou une injustice dans la manière dont un modèle d'apprentissage automatique traite les données, ce qui reflète et perpétue souvent les inégalités sociales existantes

2.0.3 Leçons clés pour aujourd'hui

- L'utilisation (peut-être bien intentionnée) des algorithmes sans prise de conscience des problèmes conduit à des problèmes d'iniquité.
- « Nous n'utilisons pas cette variable » n'est pas un argument.
- Il n'existe pas de solution universelle au problème.
- Il est important de réfléchir à l'application avant le déploiement
- Il existe une interprétation du « droit à l'explication »
- La loi évolue et évolue en réponse aux éventuels problèmes exacerbés par l'IA et l'apprentissage automatique.

2.0.4 Alors, qu'est-ce qu'une *discrimination injustifiée* dans l'apprentissage automatique ?

Deux types :

- Non-pertinence pratique (orientation sexuelle dans les décisions d'emploi ou les recommandations d'emploi)
- Non-pertinence morale – plus compliquée (statut de handicap ou grossesse dans les décisions d'embauche et impact, par exemple, sur le coût pour l'employeur)

Notez qu'on peut avoir un algorithme avec un comportement équitable qui n'est pas juste !

Notez également qu'il existe diverses fonctionnalités sensibles (aux États-Unis par exemple, ce sont des classes protégées) :

- Race
- Genre
- Grossesse
- Religion
- Statut familial

Anciennes cartes « ligne rouge » permettant aux agents de prêt de décider où prêter et où ne pas prêter - ce n'est plus légal aux États-Unis

https://www.youtube.com/watch?v=Igq_S_7IfOU Moritz Hardt

La loi essaie de suivre les progrès…

- Règlement général sur la protection des données (RGPD) : article 22 : accorde aux individus le droit de ne pas être soumis à des décisions basées uniquement sur une prise de décision automatisée, ce qui implique un besoin de transparence et d'explicabilité.

Concevez un système capable d'évaluer s'il vaut la peine de livrer le jour même dans une région particulière ou non.

$$f : \mathbf{X} \rightarrow y$$

Exemple issu du monde réel... Couverture de livraison Amazon le jour même

https://www.bloomberg.com/graphics/2016-amazon-same-day/

Exemple issu du monde réel... Couverture de livraison Amazon le jour même

https://www.bloomberg.com/graphics/2016-amazon-same-day/

2.0.5 Leçon 1: L'utilisation (peut-être bien intentionnée) des algorithmes sans prise de conscience des problèmes conduit à des problèmes d'iniquité.

Une grande question à laquelle sont confrontés (surtout) les ingénieurs aux États-Unis :

- Devraient-ils collecter l'origine ethnique comme variable ?

L'origine ethnique est codée indirectement dans les autres variables (une variable latente), peut-être via des mécanismes sociaux inéquitables que nous ne voulons pas perpétuer

Donc, même sans l'utiliser comme variable, cela peut toujours être un problème.

De plus, si nous ne collectons pas l'origine ethnique, nous n'avons aucune possibilité d'évaluer si notre modèle prend indirectement des décisions en l'utilisant.

2.0.6 Leçon 2: Donc « Nous n'utilisons pas cette variable » n'est pas un argument.

2.0.7 En anglais on dirait que 'there is no fairness without awareness' (il n'y a pas de justice sans conscience)

2.0.8 Alors, comment décidons-nous d'inclure ou non une variable ?

Deux types de non-pertinence (dans le choix d'un employé, par. ex.).

Si une variable appartient à l'une de ces deux catégories, ce n'est pas une bonne idée de l'utiliser comme prédicteur. Cependant, cela peut quand même être important à considérer !

1) Non-pertinence pratique (orientation sexuelle dans les décisions d'emploi ou les recommandations d'emploi)

2) Non-pertinence morale – plus compliquée (statut de handicap ou grossesse dans les décisions d'embauche et impact, par exemple, sur le coût pour l'employeur)

Notez que certaines variables sont importants à prendre en compte pour d'autres raisons.

Par exemple, l'origine ethnique peut être un prédicteur justifiable 'pratique' dans certains processus de décision médicale en raison de certains liens génétiques).

2.0.9 Alors, comment pouvons-nous évaluer et tester l'équité ?

- Critères d'équité dans la classification

(Modélisation causale et explicabilité la semaine prochaine)

2.1 Les deux critères populaires

1) Taux positif égal: un nombre égal de cas « acceptés » dans tous les groupes

$$p(\hat{Y} = 1|G = a) = p(\hat{Y} = 1|G = b)$$

Motivation : "Tous les groupes ont un droit égal à l'acceptation."

2.2 Les deux critères populaires

2) Égalisation des taux d'erreur: nombre de faux positifs et de négatifs égal dans tous les groupes

$$p(\hat{Y} = 1|Y = 0, G = a) = p(\hat{Y} = 1|Y = 0, G = b)$$

$$p(\hat{Y} = 0|Y = 1, G = a) = p(\hat{Y} = 0|Y = 1, G = b)$$

Motivation : "Qui supporte le coût d'une mauvaise classification ? Historiquement, les taux d'erreur plus élevés se produisent davantage dans les groupes marginalisés. "

2.2.1 1 Taux positif égal

Pour le decision $\hat{Y}$, l'appartenance au groupe G:

$$p(\hat{Y} = 1|G = a) = p(\hat{Y} = 1|G = b)$$

[4]:
```
import numpy as np
from typing import Dict, Tuple


def calculate_equal_positive_rate(predictions: np.ndarray, groups: np.ndarray)␣
 ↪-> Dict[str, float]:
    """
    Evaluate the equal positive rate criterion for a decision classifier,
    considering only the group membership.
```

```
    :param predictions: A NumPy array containing binary predictions from the␣
 ↪classifier.
    :param groups: A NumPy array indicating the group membership of each␣
 ↪prediction.
    :return: A dictionary with the positive rate for each group.
    """
    unique_groups = np.unique(groups)
    positive_rates: Dict[str, float] = {}

    for group in unique_groups:
        # Mask for the current group
        mask = (groups == group)

        # Calculate positive rate for the group
        positive_rate = predictions[mask].mean()
        positive_rates[group] = positive_rate

    return positive_rates
```

[5]:
```
np.random.seed(0)

N = 100

# Générer quelques exemples de données
y_true = np.random.randint(0, 2, size=N)

groups = np.random.choice(['A', 'B'], size=N)
print('Example group assignment:',groups[:5])

y_pred = np.random.randint(0, 2, size=N)  #  Au lieu de cela, écoutez vos␣
 ↪prédictions réelles
print('Example predictions:', y_pred[:5])

result = calculate_equal_positive_rate(y_pred, groups)  # nous n'utilisons pas␣
 ↪les vraies étiquettes
print('Per-group positive rate:', result)
```

```
Example group assignment: ['B' 'A' 'A' 'B' 'A']
Example predictions: [1 0 1 0 0]
Per-group positive rate: {'A': 0.5272727272727272, 'B': 0.4888888888888889}
```

2.2.2 1 Taux positif égal : faiblesses !

On peut avoir un nombre élevé de vrais positifs dans un groupe et un nombre élevé de faux positifs dans un autre, et remplir le critère de taux de positivité égale.

Il existe des solutions dégénérées qui satisfont au critère.

2.2.3 2 Égalisation des taux d'erreur

Assurer un taux de faux positifs égal:

$$p(\hat{Y} = 1|Y = 0, G = a) = p(\hat{Y} = 1|Y = 0, G = b)$$

et garantir un taux de faux négatifs égal:

$$p(\hat{Y} = 0|Y = 1, G = a) = p(\hat{Y} = 0|Y = 1, G = b)$$

[6]:
```python
def calculate_false_rates(y_true: np.ndarray, y_pred: np.ndarray, groups: np.
↪ndarray) -> Tuple[Dict[str, float], Dict[str, float]]:

    """
    Calculate the false positive and false negative rates for each group.

    :param y_true: A NumPy array containing the true labels.
    :param y_pred: A NumPy array containing the predicted labels.
    :param groups: A NumPy array indicating the group membership of each label.
    :return: A dictionary with the false positive and false negative rates for␣
↪each group.
    """
    unique_groups = np.unique(groups)
    fp_rates: Dict[str, float] = {}
    fn_rates: Dict[str, float] = {}

    for group in unique_groups:
        mask = (groups == group)

        # False positives: where true label is 0 but predicted label is 1
        false_positives = np.sum((y_true[mask] == 0) & (y_pred[mask] == 1))

        # False negatives: where true label is 1 but predicted label is 0
        false_negatives = np.sum((y_true[mask] == 1) & (y_pred[mask] == 0))

        # Total negatives and positives
        total_negatives = np.sum(y_true[mask] == 0)
        total_positives = np.sum(y_true[mask] == 1)

        fp_rate = false_positives / total_negatives if total_negatives > 0 else␣
↪0
        fn_rate = false_negatives / total_positives if total_positives > 0 else␣
↪0

        fp_rates[group] = fp_rate
        fn_rates[group] = fn_rate
```

```
    return fp_rates, fn_rates
```

[7]:
```
fp_rates, fn_rates = calculate_false_rates(y_true, y_pred, groups)
print('False positive rates:', fp_rates)
print('False negative rates:', fn_rates)
```

```
False positive rates: {'A': 0.375, 'B': 0.6}
False negative rates: {'A': 0.3548387096774194, 'B': 0.6}
```

2.2.4 2 Égalisation des taux d'erreur : faiblesses !

Ce critère ne peut être évalué qu' avoir les vraies données - c'est un critère 'post-hoc'.

Une telle évaluation peut arriver « trop tard ».

Parfois, nous ne pouvons jamais l'évaluer… (par exemple, l'embauche)

Cela ne nous aide pas non plus à éviter les solutions dégénérées…

2.2.5 3 Calibration

Pour une fonction de score $R(\mathbf{X})$ qui pourrait simplement être une estimation de $\mathbb{E}[Y|\mathbf{X}]$, un score R est calibré si :

$$p(Y = 1|R = r) = r$$

Nous pouvons calibrer par groupe :

$$p(Y = y|R = r, G = a) = r$$

Nous voulons donc que notre score se comporte comme une probabilité au sein de chaque groupe et signifie donc ce qu'il est censé signifier.

Cela découle de $Y \perp\!\!\!\perp G|R$.

2.2.6 3 Calibration

En mots : si mon objectif est de prédire Y, après avoir observé le score R il n'est pas nécessaire de connaître l'appartenance au groupe. Un r = 0,8 signifie 0,8 quel que soit le groupe.

Si un modèle est calibré, cela signifie que le score reflète fidèlement la probabilité du véritable résultat Y, quelle que soit l'appartenance au groupe $G = a$ ou $G = b$ ou autre. Il garantit la fiabilité et la « fiabilité » du classificateur entre les groupes.

Par exemple. r = 0,8 signifie un taux d'insuffisance cardiaque de 80 % en moyenne par rapport aux personnes qui reçoivent un score de 0,8 - leur appartenance à un groupe ne change rien à cela.

[8]:
```
# Code... à voir la semaine prochaine 'Platt scaling'!
```

2.2.7 3 Calibration : faiblesses !

Cela ne change rien à la pérennisation des inégalités liées aux coûts réels (par exemple grossesses ou handicaps de productivité).

Calibration ne résout pas le problème de pertinence morale dont nous avons parlé.

2.2.8 Faiblesses des deux

Ils sont incompatibles!

Exemple simple…

On peut utiliser ces critères pour détecter des problèmes, mais ils ne vous indiquent pas quelle devrait être la solution. En effet, ils sont également tous deux incompatibles !

Changez le contexte mais gardez les chiffres les mêmes…

En moyenne, les hommes seront moins qualifiés, ce qui entraînera un pire bilan pour ce groupe, créant ainsi un nouveau déséquilibre lié au groupe. Mais… nous pourrions ainsi bénéficier d'une meilleure diversité.

Leçon 3: Il n'existe pas de solution universelle au problème.

Autres avertissements :

Le respect des critères d'équité à l'aide de l'apprentissage automatique peut souvent conduire à des « solutions paresseuses » et à des problèmes d'équité des **sous**groupes.

2.2.9 Des solutions globales nécessitaient une compréhension détaillée des problèmes sociétaux, moraux et équitables sous-jacents liés au domaine.

2.2.10 Par conséquent, les ingénieurs devraient s'adresser à des experts du domaine !

Considérez « Défaut de comparaître devant le tribunal ».

Une approche en cours de promotion : prédire l'absence de comparution et de détention si le risque est élevé (cela peut durer longtemps - cela est dévastateur pour les familles et le soutien social, et coûteux).

Alternative : mettre en œuvre des mesures pour atténuer les problèmes. Reconnaître que les personnes ne se présentent pas au tribunal en raison de problèmes de transport, du manque de services de garde d'enfants, d'horaires de travail ou d'un trop grand nombre de rendez-vous au tribunal.

Une solution d'apprentissage automatique peut chercher à réduire les coûts mais en fin de compte faire le contraire.

2.3 Résumé

- L'utilisation (peut-être bien intentionnée) des algorithmes sans prise de conscience des problèmes conduit à des problèmes d'iniquité.
- « Nous n'utilisons pas cette variable » n'est pas un argument.
- Il n'existe pas de solution universelle au problème.

Les algorithmes débiaisés/équitables sont difficiles à concevoir, et même leur définition est difficile, nécessitant des connaissances spécialisées dans le domaine et la prise en compte de principes pratiques, éthiques et moraux.

Enfin, nous sommes limités par notre accès uniquement à la distribution conjointe $p(X, Y, G...)$. L'inférence causale est une option pour explorer statistiquement ces problèmes plus profonds. Il existe un fossé sémantique difficile à combler.

2.3.1 La semaine prochaine:

- Explicabilité de l'apprentissage automatique et perspective causale

[9]:
```
from IPython.display import Image, display

def finished_lecture(lecture_state: int = 0, thank_you_image_url: str = None):
    # Check if the input is an integer and either 0 or 1
    if not isinstance(lecture_state, int) or lecture_state not in [0, 1]:
        raise ValueError("condition must be an integer and either 0 or 1")

    if lecture_state == 1:
        print('Merci et je serai heureux de répondre à vos questions !')
        display(Image(url='https://raw.githubusercontent.com/matthewvowels1/
 ↪SIN06023/main/merci.png'))
    elif lecture_state == 0:
        print('Conférence en cours...')
        return


state = 1

finished_lecture(lecture_state=state)
```

```
Merci et je serai heureux de répondre à vos questions !
```

```
<IPython.core.display.Image object>
```

[ ]:

[ ]: