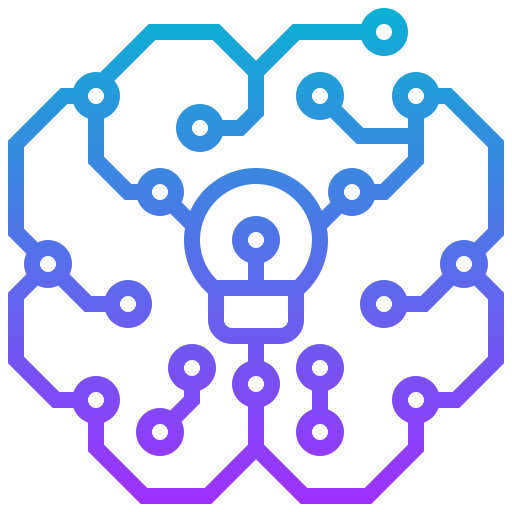
**BÁO CÁO BÀI TẬP LỚN MÔN HỌC MÁY**

**

*Chu Hữu Đăng Trường - Đỗ Minh Nhật - Nguyễn Đức Anh - Nguyễn Nhật Minh*

**MỤC LỤC**

[**PHẦN I - THÔNG TIN CHUNG 2**](#_vsxgca6wmfvc)

[THÔNG TIN THÀNH VIÊN 2](#_g3powt2znmsw)

[PHÂN CÔNG CÔNG VIỆC 2](#_qz73xtxos2nh)

[**PHẦN II - GIỚI THIỆU CHUNG VỀ BÀI TOÁN 3**](#_346olfozm10m)

[MỤC ĐÍCH: ĐÁNH GIÁ CÔNG BẰNG TRONG TUYỂN DỤNG 3](#_3dryal79ily4)

[BÀI TOÁN DỰ ĐOÁN KẾT QUẢ TUYỂN DỤNG 3](#_ha9uji29qbom)

[**PHẦN III - TÌM KIẾM VÀ XỬ LÝ DỮ LIỆU 4**](#_cpu93afiwukj)

[CÁC NGUỒN DỮ LIỆU THU THẬP ĐƯỢC 4](#_pvn36wz06t3l)

[EXPLORATORY DATA ANALYSIS 5](#_joaig8aucjlq)

[DATA PREPROCESSING 11](#_awjvht2f7jrx)

[FEATURE ENGINEERING 11](#_y6csp26gixww)

[**PHẦN IV - MÔ HÌNH SỬ DỤNG 13**](#_5tirbhjnovtq)

[BASELINE 13](#_tfbnqia9havv)

[LINEAR SUPPORT VECTOR MACHINE (SVM) 13](#_21038ysaakhw)

[LINEAR SUPPORT VECTOR MACHINE (SVM) 15](#_5cs2emjajwn7)

[LOGISTIC REGRESSION 15](#_79rt6esvdt39)

[DECISION TREE 16](#_1qrl1t9jvc2a)

[**PHẦN V - CÁC CHỈ SỐ ĐÁNH GIÁ 17**](#_uomghgylled0)

[TỔNG QUAN 17](#_yswkyfv8htwt)

[ĐỘ CHÍNH XÁC (ACCURACY) 17](#_1vzhwgr6clnp)

[AREA UNDER ROC CURVE 18](#_oudxmrrpkvgt)

[F1 SCORE 19](#_o78ubin2tuuo)

[AREA UNDER PR CURVE 20](#_r2qw7pewf6ex)

[DEMOGRAPHIC PARITY 20](#_d9a1mbndfmn7)

[EQUALIZED ODDS 21](#_q6zj6boeam08)

[ABSOLUTE BETWEEN-ROC AREA (ABROCA) 22](#_l01wrgm7eh8r)

[HONORABLE MENTION: PREDICTIVE PARITY 23](#_7hlq5laaz67n)

[BẢNG CHỈ SỐ 23](#_tqcvpei8jf)

[**PHẦN VI - NGUỒN, TÀI LIỆU THAM KHẢO 24**](#_4rqsg6g8o1vw)

# 

# PHẦN I - THÔNG TIN CHUNG

## THÔNG TIN THÀNH VIÊN

Chu Hữu Đăng Trường

* Mã sinh viên: 22022505
* Lớp: AI2, Viện trí tuệ nhân tạo trường Đại học Công nghệ, ĐHQGHN
* Email: [22022505@vnu.edu.vn](mailto:22022505@vnu.edu.vn)

Đỗ Minh Nhật

* Mã sinh viên: 22022537
* Lớp: AI2, Viện trí tuệ nhân tạo, trường Đại học Công nghệ, ĐHQGHN
* Email: [22022537@vnu.edu.vn](mailto:22022537@vnu.edu.vn)

Nguyễn Đức Anh

* Mã sinh viên: 22022504
* Lớp: AI1, Viện trí tuệ nhân tạo, trường Đại học Công nghệ, ĐHQGHN
* Email: [22022504@vnu.edu.vn](mailto:22022504@vnu.edu.vn)

Nguyễn Nhật Minh

* Mã sinh viên: 22022503
* Lớp: AI1, Viện trí tuệ nhân tạo trường Đại học Công nghệ, ĐHQGHN
* Email: [22022503@vnu.edu.vn](mailto:22022503@vnu.edu.vn)

## PHÂN CÔNG CÔNG VIỆC

Chu Hữu Đăng Trường (C)

* Tìm hiểu, code phần chỉ số đánh giá + Logistic Regression

Đỗ Minh Nhật

* Tìm hiểu, code phần xử lý dữ liệu

Nguyễn Đức Anh

* Tìm hiểu, code phần SVM

Nguyễn Nhật Minh

* Tìm hiểu, code phần Decision Tree, Random Forest, Gradient Boosting

# PHẦN II - GIỚI THIỆU CHUNG VỀ BÀI TOÁN

## MỤC ĐÍCH: ĐÁNH GIÁ CÔNG BẰNG TRONG TUYỂN DỤNG

Trong thời đại công nghệ hiện đại, việc sử dụng trí tuệ nhân tạo (AI) trong quá trình tuyển dụng đặt ra những thách thức lớn về công bằng và đạo đức. Một ý tưởng quan trọng là phải đảm bảo rằng quá trình này không tạo ra độ chệch và phân biệt đối xử. Để đảm bảo công bằng, cần xem xét cẩn thận về dữ liệu đào tạo mô hình AI, vì nếu dữ liệu đó chứa đựng độ chệch giới tính, sắc tộc hoặc địa lý, nó có thể dẫn đến quyết định không công bằng.

Đồng thời, việc xây dựng mô hình AI có khả năng giải thích là quan trọng. Transparent AI giúp mọi người hiểu rõ quyết định của hệ thống, từ đó tăng cường niềm tin và minh bạch trong quá trình tuyển dụng. Các chính sách và tiêu chuẩn cần được thiết lập để đảm bảo rằng AI không chỉ hỗ trợ quá trình tuyển dụng mà còn đảm bảo sự công bằng và đạo đức trong mọi khía cạnh.

Với niềm tin trên, nhóm có ý định nghiên cứu sự công bằng trong tuyển dụng, dựa trên bài toán dự đoán kết quả tuyển dụng. Đây là bài toán tiêu biểu cho bài toán phân loại nhị phân.

## BÀI TOÁN DỰ ĐOÁN KẾT QUẢ TUYỂN DỤNG

Mô hình học máy của bài toán này có dữ liệu đầu vào là các trường thông tin tuyển dụng, và đầu ra là kết quả đỗ/trượt.

Thông tin tuyển dụng bao gồm các thuộc tính “nhạy cảm,” hay những giá trị mà chúng ta không thể thay đổi được như tuổi, giới tính, quốc tịch. Vì vậy, ở phần đánh giá mô hình, các chỉ số công bằng sẽ được đưa vào để xem xét sự “công bằng” của mô hình với các thuộc tính trên. Trong bài tập lớn này, thuộc tính giới tính sẽ được chú ý nhiều nhất.

# PHẦN III - TÌM KIẾM VÀ XỬ LÝ DỮ LIỆU

## CÁC NGUỒN DỮ LIỆU THU THẬP ĐƯỢC

|  | Utrecht | Job Placement/Campus Recruitment |
| --- | --- | --- |
| Nguồn dữ liệu | [Utrecht Fairness Recruitment dataset](https://www.kaggle.com/datasets/ictinstitute/utrecht-fairness-recruitment-dataset) | [Job Placement Dataset | Kaggle](https://www.kaggle.com/datasets/ahsan81/job-placement-dataset)  &  [Campus Recruitment](https://www.kaggle.com/datasets/benroshan/factors-affecting-campus-placement) |
| Features | 15 (chi tiết ở dưới) | 15 (chi tiết ở dưới) |
| Số hàng (samples) | 4,000 | 215 |
| Further notes |  | 2 bộ dữ liệu giống hệt nhau, trừ 2 trường *specialisation* và *salary* |

### **Utrecht**

Chi tiết của các trường:

* Id: a unique identifier. Do not use in your algorithm
* gender: male, female, or other
* age: age in years
* nationality: Current nationality. Dutch, German or Belgian
* sport: The first/main sport the candidate listed on their CV
* ind-university\_grade: University grade (percentage) of the candidate
* ind-debateclub: Whether the candidate participated in a debating / social club
* ind-programming\_exp: Whether the candidate has programming experience
* ind-international\_exp: Whether the candidate has international experience
* ind-entrepeneur\_exp: Whether the candidate has run their own company
* ind-languages: number of additional languages 0,1,2,3,4 spoken fluently by candidate
* ind-exact\_study: whether they studied physics, engineering or other science-oriented study
* ind-degree: Highest completed degree. phd, bachelor or master level
* company: the company that the candidate applied to
* decision: target value, to be predicted. whether the candidate was hired.

### **Job Placement/Campus Recruit**

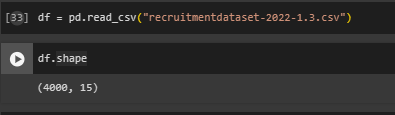
Chi tiết của các trường:

* gender: Gender of the candidate
* ssc\_p: Senior secondary exams percentage (10th Grade)
* ssc\_b: Board of education for ssc exams
* hsc\_p: Higher secondary exams percentage (12th Grade)
* hsc\_b: Board of education for hsc exams
* hsc\_s: Subject of study for hsc
* degree\_p: Percentage of marks in undergrad degree
* degree\_t: Undergrad degree majors
* workex: Past work experience
* etest\_p: Aptitude test percentage
* specialisation: Post Graduation(MBA)- Specialization
* mba\_p: Percentage of marks in MBA degree
* status: Status of placement. (Placed / Not Placed)
* salary: Salary offered by corporate to candidates

Nhìn chung các dữ liệu trên đều tương đối đầy đủ và không có missing values khi kiểm tra bằng method **isnull()**

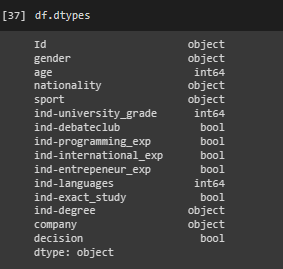
## EXPLORATORY DATA ANALYSIS

Trước hết ta sẽ EDA với dữ liệu Utrecht, đầu tiên với method shape ta thấy dữ liệu đã cho có 4000 dòng và 15 cột (như hình 3.1)



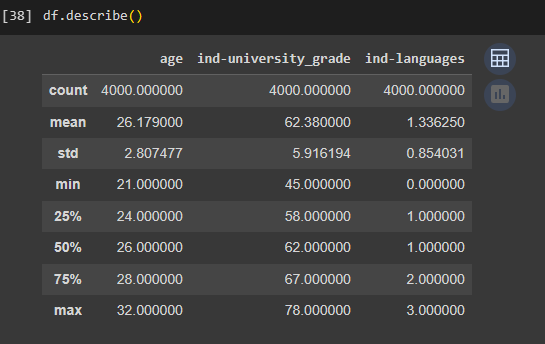
*Hình 3.1*

Kế đến ta kiểm tra kiểu dữ liệu của từng trường với method dtypes trong dataset trên, kết quả cho ra được như hình 3.2

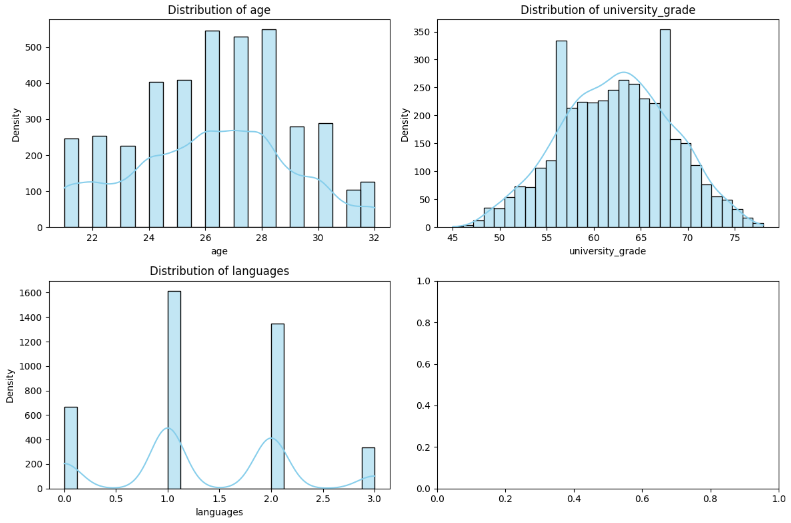


*Hình 3.2*

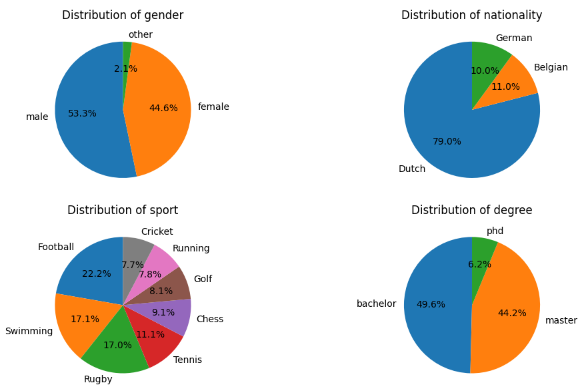
Cuối cùng ta sử dụng hàm describe để để thực hiện các thống kê cơ bản về dữ liệu, cũng như vẽ biểu đồ phân phối của các trường kết quả thu được như hình 3.3 3.4 và 3.5



*Hình 3.3*

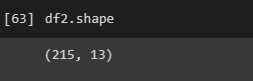


*Hình 3.4*

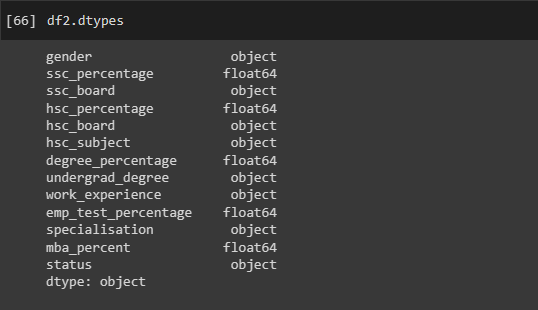
**

*Hình 3.5*

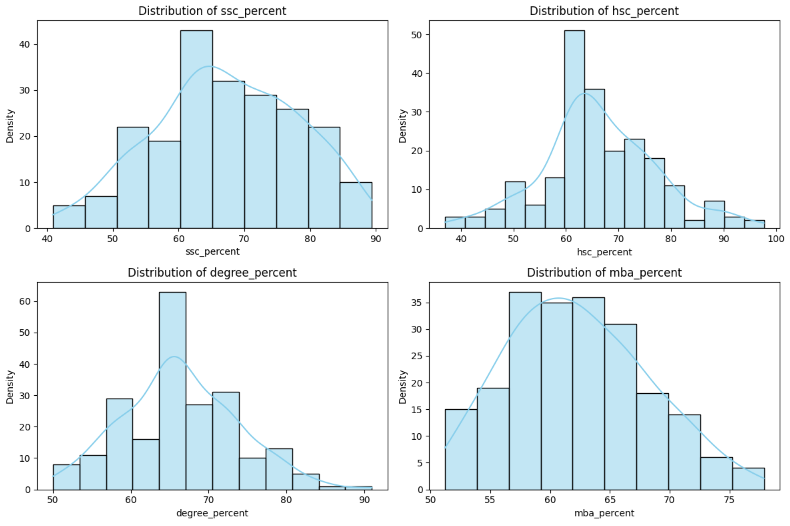
Tương tự, ta thực hiện EDA với dataset Job Placement/Campus recruitment ta thu được kết quả như các hình 3.6, 3.7, 3.8, 3.9 và 3.10



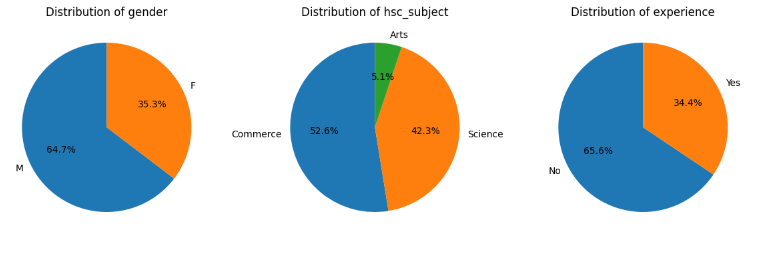
*Hình 3.6: Số lượng hàng và trường của data*

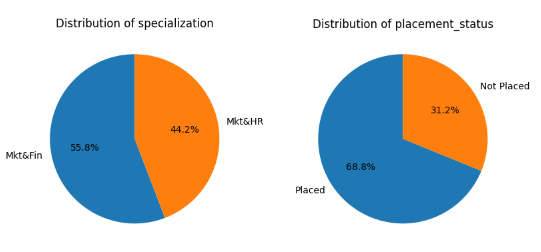
**

*Hình 3.7: Kiểu dữ liệu của các trường trong data*

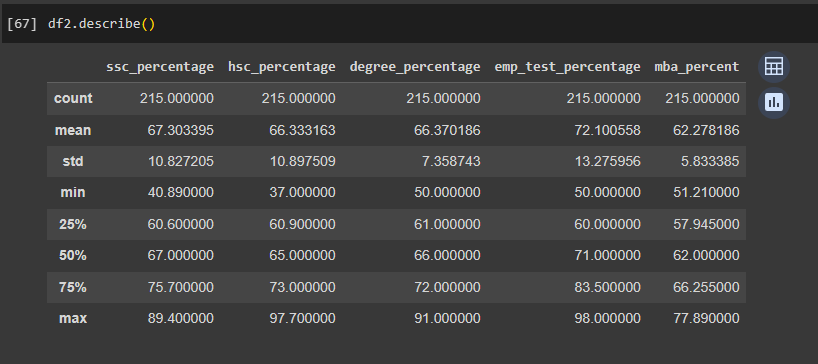
**

*Hình 3.8: Phân phối của các trường dữ liệu có dạng numerical*





*Hình 3.9: Phân phối của các trường dữ liệu có dạng categorical*

**

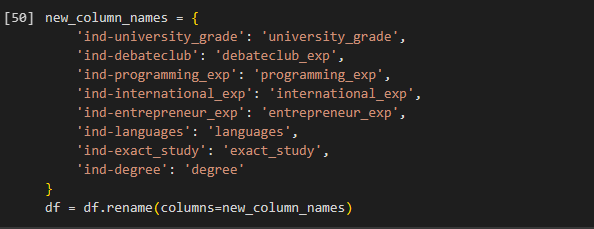
*Hình 3.10: Một số thống kê cơ bản về các trường có dạng numerical trong data*

Từ những phân phối thống kê trên chúng ta thấy không nên tiến hành Data Integration với 2 nguồn dataset đã cho mà chỉ nên sử dụng dataset có cỡ lớn nhất là dataset utrecht do có sự sai khác về các trường dữ liệu mà ta không thể xử lý được. (Ngoài ra còn 1 lý do sẽ được trình bày ở mục **FEATURE ENGINEERING**)

## DATA PREPROCESSING

Data Cleaning: Loại bỏ những hàng bị lặp lại, kiểm tra những trường dữ liệu bị thiếu giá trị và xử lý

Rename: Sửa lại tên các trường cần thiết để tiện thao tác (như hình 3.11)



*Hình 3.11: Một số thống kê cơ bản về các trường có dạng numerical trong data*

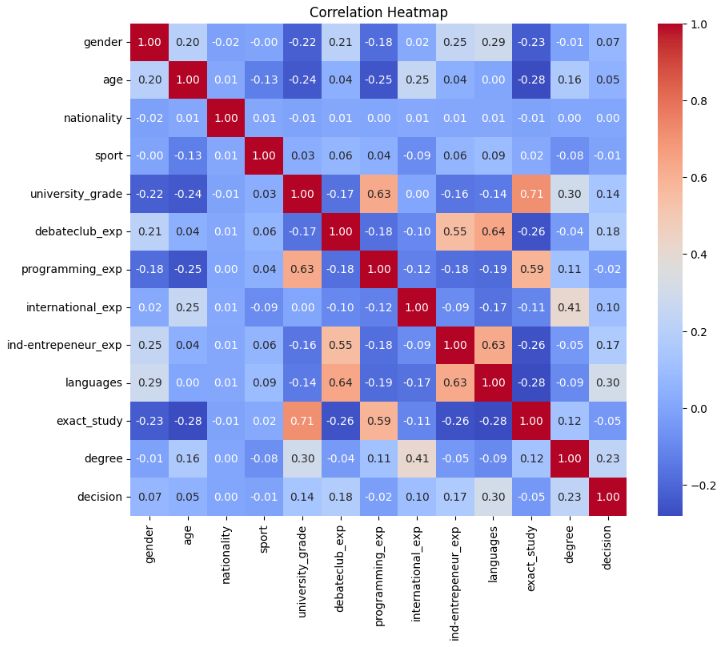
One-hot encoding: Các features ‘gender’ và ‘degree’ được one-hot encoding từ dạng categorical thành vector để fit vào mô hình học máy được sử dụng ở những bước tiếp theo

Standardize: Chuẩn hóa GPA thành phân phối với mean = 0 và variance = 1 bằng công thức (hoặc sử dụng z-score cho trường GPA vì phân phối trường này tương tự phân phối chuẩn)

Train-Test split: Tách bộ dataset thành train-test data theo tỉ lệ 80-20

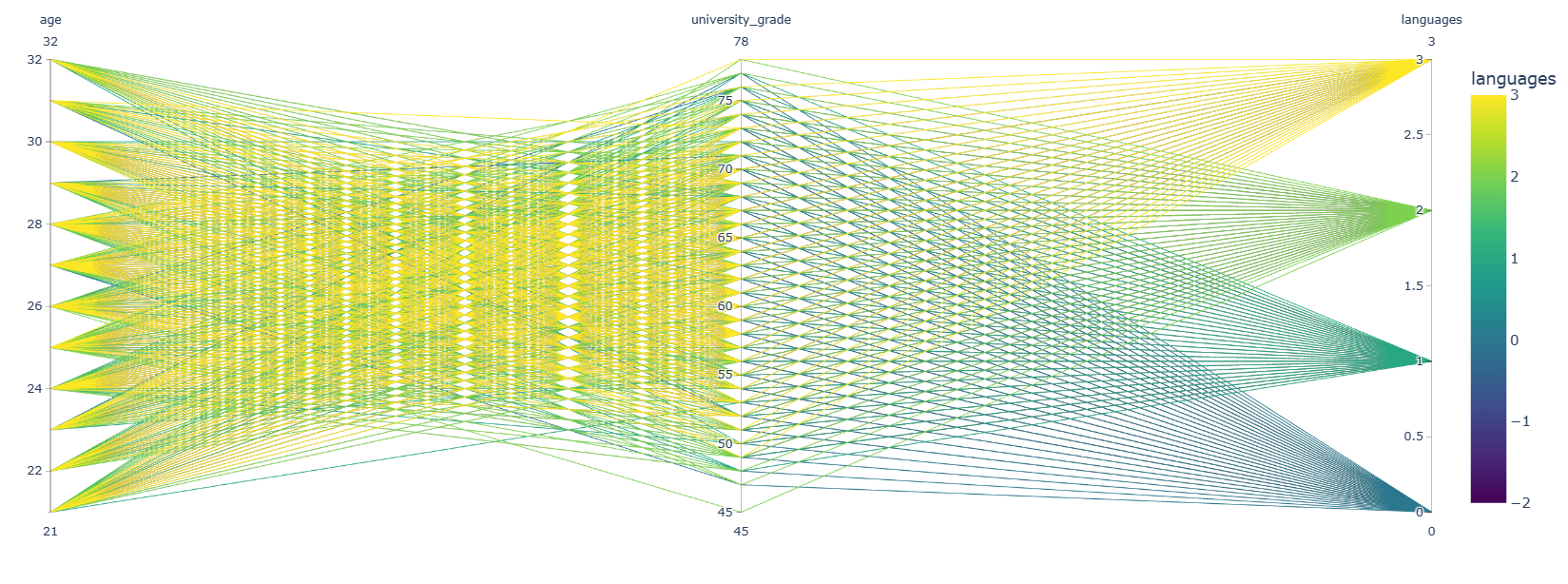
## FEATURE ENGINEERING

Trước hết ta phải nắm rõ từng mối quan hệ của các trường dữ liệu, để làm được điều này đầu tiên ta plot một heatmap để xem xét mối tương quan giữa các trường dữ liệu (hình 3.12)



*Hình 3.12: Ma trận thể hiện mối tương quan giữa các trường dữ liệu*

Ngoài ra ta tiếp tục khảo sát mối tương quan giữa các trường dữ liệu với Parallel Coordinate Plot (hình 3.13)



*Hình 3.13: Parallel Coordinate Plot*

Ma trận 3.12 cho ta thấy một mối tương quan mạnh giữa các trường:

* university\_grade và exact\_study
* university\_grade và programming\_exp
* debateclub\_exp và language
* debateclub\_exp và ind-entrepeneur\_exp

Điều này gợi ý việc bỏ một vài trường dữ liệu có tương quan với các trường dữ liệu khác khác khi huấn luyện mô hình với việc kết hợp xem biểu đồ của MI score.

Từ đồ thị 3.13 ta thấy các trường categorical không tồn tại thêm bất kỳ outlier nào vậy nên việc sử dụng một số thuật như OCSVM hay Isolation Forest để tìm và loại bỏ outlier là không cần thiết.

Sau khi đã có cái nhìn khái quát về quan hệ giữa các trường dữ liệu, tiếp theo chúng ta sẽ bắt đầu chọn lọc những trường dữ liệu phù hợp để phát triển mô hình.

Ở đây chúng ta sẽ sử dụng MI (Mutual Information Score) có công thức như sau:

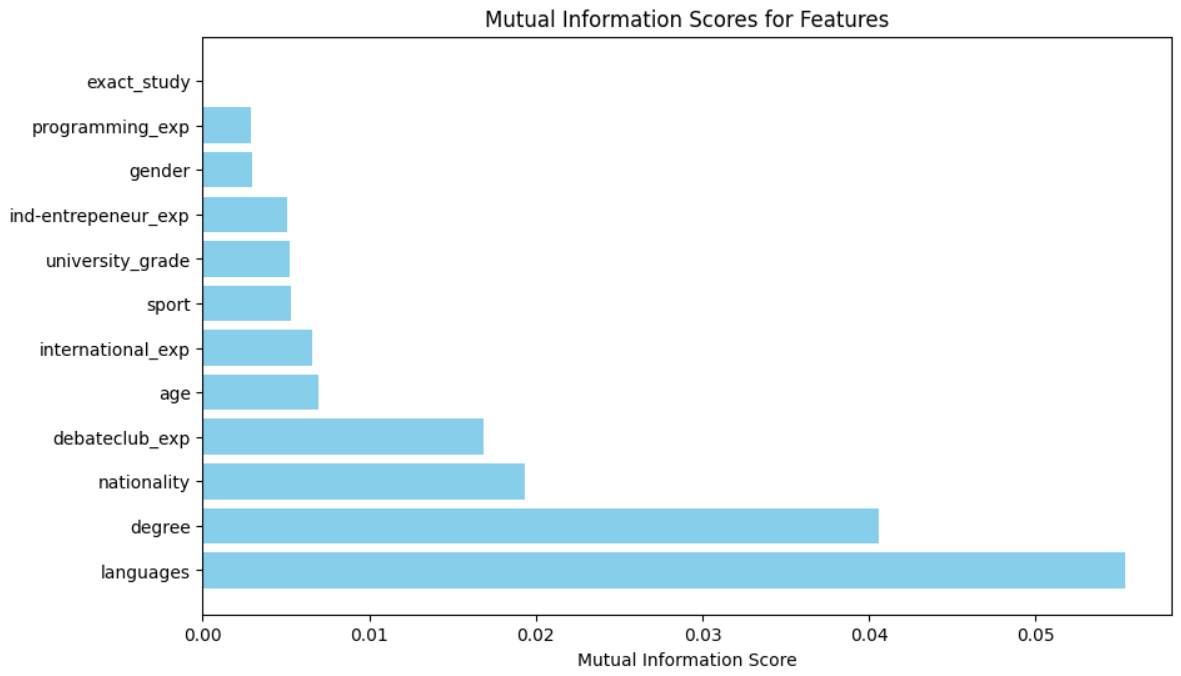


MI Score sẽ giúp ta tìm ra những trường dữ liệu có mối tương quan với trường dữ liệu bài toán dự đoán hơn các trường dữ liệu khác từ đó cho ta phương hướng phát triển trường dữ liệu nào để sử dụng cho mô hình dự đoán. Để dễ minh hoạ ta plot một biểu đồ hình cột về MI score của các trường dữ liệu (Hình 3.14).

Trong hình 3.14, ta dễ dàng thấy MI score của exact\_study là 0 điều này có nghĩa là exact\_study hoàn toàn độc lập với trường dữ liệu mình muốn dự đoán, do đó ta sẽ bỏ trường dữ liệu này đi khi xây dựng mô hình.

Ngoài ra, chúng ta cũng nên lưu ý MI Score của tất cả các trường trên đều rất thấp điều này ngầm ám chỉ các trường dữ liệu trên đều tạo ra những ‘weak classifier’ nên khả năng cao ta phải biến đổi, tạo trường dữ liệu mới hoặc sử dụng ensemble method (Ở đây, ta sẽ chọn phương pháp tiếp cận ensemble method và built-in feature importance ở các mô hình để chọn trường dữ liệu do các dữ liệu dạng categorical rất khó biến đổi tạo ra trường dữ liệu mới mới giúp tăng hiệu suất mô hình).

Cũng từ hình 3.14 ta thấy trường degree, languages và nationality chắc chắn sẽ được model của ta sử dụng do có MI scores cao hơn các trường khác, mặt khác với dataset Job Placement 3 trường trên không tồn tại hoặc khó biến đổi thành một trường giống 1 trong 3 trường trên nên ta sẽ không tiến hành Data Integration mà chỉ tiến hành huấn luyện mô hình trên duy nhất tập data Utrecht.



*Hình 3.14: MI score của các trường dữ liệu*

# PHẦN IV - MÔ HÌNH SỬ DỤNG

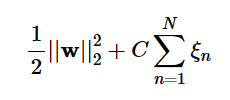
## BASELINE

Do tập dữ liệu có phần data False nhiều gấp đôi data True, nên lấy baseline là đánh trượt hết tất cả. Khi đó accuracy cũng đã được khoảng 66.7% (dù các chỉ số đánh giá khác không tốt như True Positive Rate = 0%)

## LINEAR SUPPORT VECTOR MACHINE (SVM)

Đây là một bài toán binary classification nên chúng ta sẽ sử dụng SVM để xem liệu có thể tìm được một hyperplane chia data thành hai phần. Tuy nhiên với bộ dữ liệu nhiều chiều như này khó có thể tìm được một hyperplane thỏa mãn vì nó linearly inseparable, tức là nếu ta bắt buộc phải chia thành 2 phần mà không có điểm nào nằm sai phía thì sẽ không thể chia. Tuy nhiên ta có thể linh hoạt để một số điểm nhiễu nằm khác phía, để có thể có một mặt phẳng chia được bộ dữ liệu tốt hơn baseline.

Fine-tune hyperparameter: hyperparameter C trong bài toán SVM có tác dụng phạt những điểm nằm sai vị trí, biểu diễn bằng công thức dưới đây:

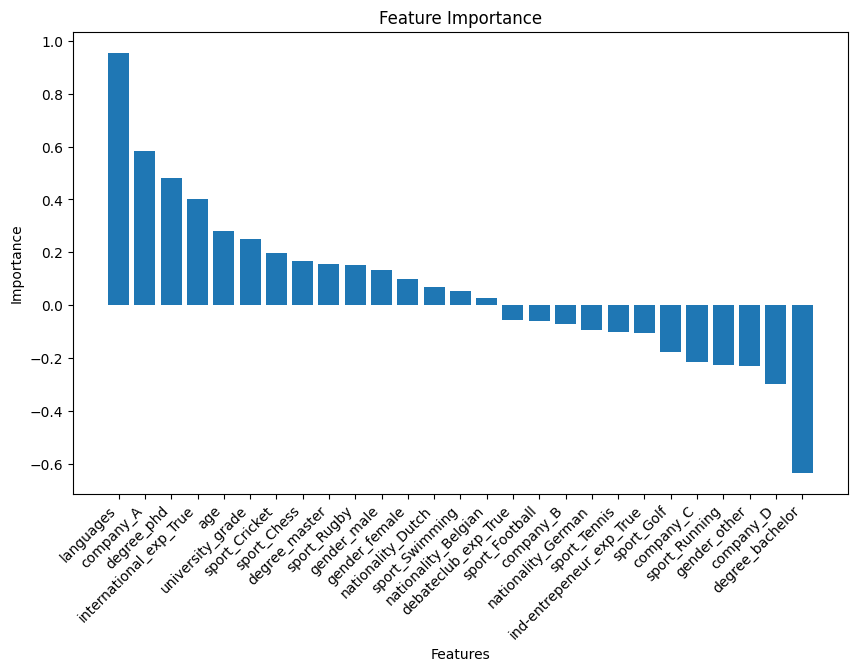


Nếu C nhỏ thì model sẽ linh hoạt hơn với các điểm nằm sai vùng phân chia. Ngược lại nếu C lớn và quá lớn, model sẽ cho ra nghiệm của bài toán Hard Margin SVM, ở đây đồng nghĩa với việc chia tất cả dữ liệu về phía false (tương tự với baseline). Để chọn tham số C phù hợp thì ta sẽ thực hiện grid search (từ 10^-2 đến 10^2) và cross validation (5-fold) để tìm ra bộ tham số tối ưu.

Sau khi chạy grid search, ta tìm được C=10 là tối ưu cho model này.

Sau khi chạy model ngoài việc đưa ra kết quả True / False, ta cần cả confident score dưới dạng xác suất thuộc (0, 1) để có thể tính một số evaluation khác nhau nên ta dùng decision\_function của scikit learn để tính ra khoảng cách từ 1 điểm dữ liệu tới mặt chia cắt, rồi dùng hàm sigmoid để đưa nó về xác suất (0, 1).

Chạy feature importance của model, ta thu được biểu đồ:



Có thể thấy ngôn ngữ, bằng PhD, điểm GPA đại học đều có positive contribution tới việc dự toán một người có trúng tuyển. Gender male và female cũng gần nhau và đều dương, điều này cho thấy model khá fair về mặt giới tính.

## 

## NON-LINEAR SUPPORT VECTOR MACHINE (SVM)

Dù đã tối ưu hyperparameter C để có thể giúp model linear SVM tăng accuracy nhưng mức trần vẫn chưa thể vượt qua 73%.

Có lẽ vì dữ liệu thuần túy vẫn là không linearly separable, ta sẽ thử dùng kernel khác để xử lý, cụ thể, ta dùng kernel RBF (Radial Basic Function):

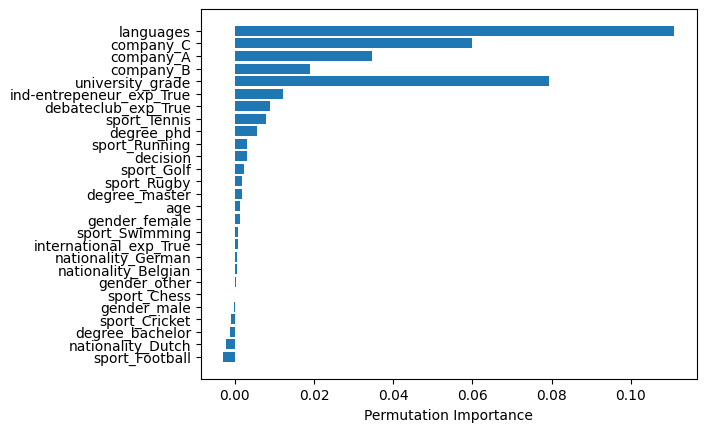


Fine-tune hyperparameter: Có thể thấy hyperparameter ở kia sẽ có tác dụng quyết định độ lớn của hàm nhân. Khi gamma tăng, hàm nhân sẽ nhỏ lại, model sẽ tập trung vào các điểm giá trị gần nhau, và ngược lại, model sẽ quan tâm đến cả những điểm ở xa. Như vậy gamma quá to sẽ có thể giúp model chính xác hơn nhưng cũng làm nó dễ overfitting hơn. Để chọn hyperparameter phù hợp (bao gồm cả C đã trình bày ở trên) thì ta sẽ thực hiện grid search và cross validation để tìm ra bộ tham số tối ưu.

Sau khi chạy grid search với 5-fold cross validation trên training data, ta tìm được C=1 và gamma=0.1. Accuracy tăng rõ ràng lên gần 84%!

Tương tự bên trên, ta lại tính confidence score bằng cách nhân với hàm sigmoid.

Do model sử dụng hàm kernel RBF, không thể tính toán feature importance một cách chính xác dùng hàm có sẵn của SVM, nên ta sử dụng permutation importance. Định nghĩa baseline metric là model gốc, sau đấy ngẫu nhiên tráo các feature, feature nào ảnh hưởng đến model nhiều nhất tức là feature quan trọng nhất. Do quá trình này là ngẫu nhiên nên ta chọn lặp lại khoảng 20 lần để ra kết quả chính xác.



Có thể thấy ngôn ngữ và điểm GPA vẫn chiếm vai trò quan trọng trong việc dự đoán. Ngoài ra model này, gender female có độ quan trọng cao hơn male một chút, nhưng không đáng kể.

## LOGISTIC REGRESSION

Với bài toán logistic regression, chúng ta sử dụng hàm logistic để chuyển đổi tổng trọng số của các đặc trưng đầu vào thành một giá trị xác suất. Hàm logistic được định nghĩa bởi công thức:

Trong đó:

- là xác suất để là true, hay ở đây là ứng viên đỗ

- là các trọng số của mô hình mà ta cần tối ưu

- là các features của bộ dữ liệu.

Để tìm các trọng số tối ưu, chúng ta sử dụng phương pháp tối ưu hóa hàm chi phí (cost function), mặc định của scikit learn là hàm binary cross-entropy.

Trong đó:

- là số lượng samples trong bộ dữ liệu.

- là giá trị dự đoán xác suất true của .

- là ground truth của .

Nghiệm của bài toán Logistic Regression chính là một hyperplane cắt qua tập dữ liệu do nó có dạng với

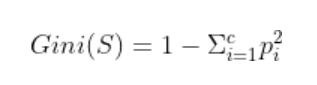
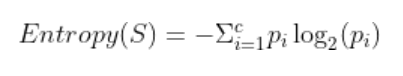
* là vector
* X là vector
* c là hằng số

Đây chính là dạng công thức của 1 hyperplane trong không gian n chiều. Như vậy, ý tưởng của Logistic Regression cũng như của linear SVM, là tìm một hyperplane có thể cắt qua bộ data để chia thành 2 phần. Tuy nhiên như đã chứng minh ở phần SVM, bộ data này là linearly inseparable nên cách làm này sẽ không cho độ chính xác cao. Kể cả khi có fine-tune với các hàm loss khác ngoài cross entropy thì việc cho 1 mặt phẳng cắt qua bộ dữ liệu như này cũng không thể cho ra một model tốt.

## DECISION TREE

Đây là một bài toán binary classification nên chúng ta có thể sử dụng model decision tree. Khi fit model decision tree với hàm mất mát Entropy cho kết quả tốt hơn Gini, điều này có thể được lí giải ở một số chỗ:

* Entropy, so với Gini, có khả năng xây dựng ranh giới phân loại phi tuyến tốt hơn:

****

Entropy có khả năng tăng nhanh hơn Gini khi xác suất của các lớp xấp xỉ nhau, điều này có thể dẫn đến khả năng xây dựng ranh giới phi tuyến tốt hơn cho dữ liệu không linearly separable.

Đối với dữ liệu không linearly separable, sự không chắc chắn thường cao, và Entropy có thể là lựa chọn tốt hơn vì nó tập trung vào việc giảm sự không chắc chắn này, giúp cây quyết định tìm ra các ranh giới phân loại phức tạp hơn.

* Quan tâm đến tính tổng quát và công bằng:

Entropy giúp đảm bảo rằng cây quyết định không chủ yếu tập trung vào các thuộc tính cụ thể mà có thể gây ra độ chệch và không công bằng.

Với thuộc tính nhạy cảm như giới tính, quan trọng để mô hình không chủ quan và không phụ thuộc nhiều vào giới tính để đưa ra quyết định.

Việc accuracy tăng khi thay đổi từ gini sang entropy cho thấy một số các features có ảnh hưởng nhất định đến prediction => Unbalance, không công bằng.

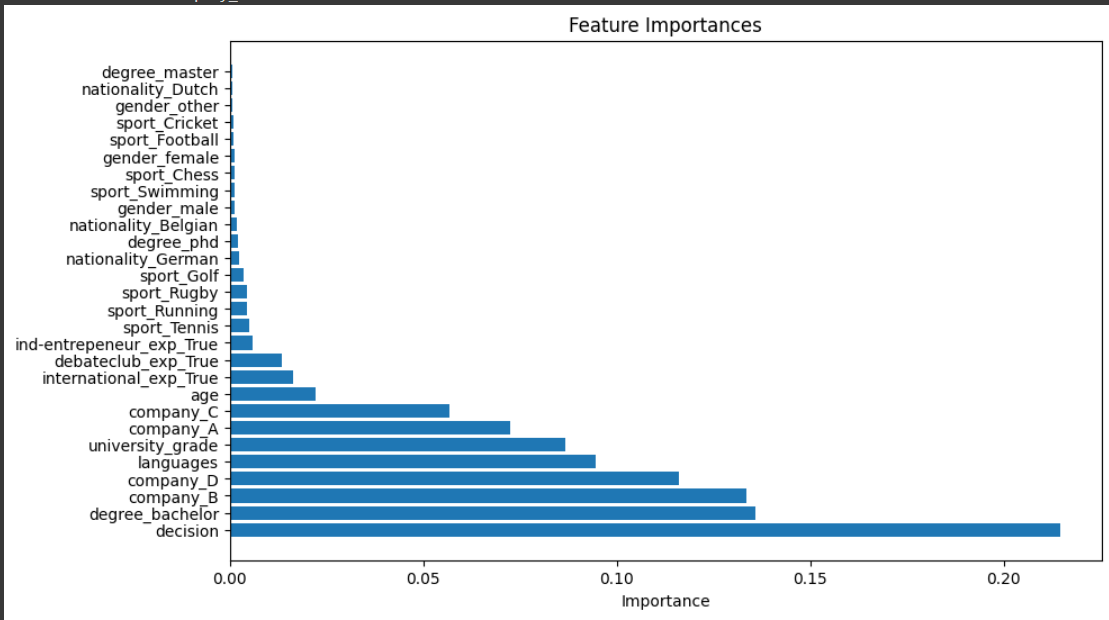
Random Forest Sử dụng RandomizedSearch để fine tune parameters cho kết quả cải thiện đáng kể.

Gradient Boosting cho kết quả tốt nhất do:

* Khả năng dự đoán tốt hơn khi combine các model học yếu như decision tree.
* Khả năng xử lý các quan hệ phi tuyến bằng cách ensemble các decision tree.
* Khả năng xử lý class imbalance tốt, tránh bias cho majority class.

Gradient Boosting feature importances:

* Trước khi hiệu chỉnh data có sự imbalance về contribution giữa các features: gender\_male có percentage cao hơn gender\_female
* Sau khi hiệu chinh data:



# PHẦN V - CÁC CHỈ SỐ ĐÁNH GIÁ

## TỔNG QUAN

Vì mục đích của nhóm là đạt được sự “công bằng” (fairness,) nên nhóm đã sử dụng nhiều chỉ số đánh giá, được chia làm 2 loại:

* Chỉ số mục tiêu (goal metrics): chỉ số mà nhóm sẽ cố gắng tối đa hóa
* Nhóm sử dụng các chỉ số mục tiêu là *F1 score*, *Accuracy, PR curve,* và *AUC score*
* Chỉ số ngưỡng (threshold metrics): chỉ số mà các model sẽ cần phải đạt được hoặc vượt qua. Nhóm chỉ số này sẽ đảm bảo sự “công bằng” cho các models
* Nhóm sử dụng các chỉ số ngưỡng *Demographic parity, Equalized odds,* và *ABROCA score*.
* Với *Demographic parity* và *ABROCA score*, các chỉ số này sẽ có ngưỡng là 0.2 (20%) lấy từ U.S. Equal Employment Opportunity Commission. Vì  *Equalized odds* có 2 phần (TPR và FPR) nên nhóm đã sử dụng ngưỡng 0.3 (0.2 \* 1.5)

## ĐỘ CHÍNH XÁC (ACCURACY)

Là tỉ lệ số lượng ứng viên trong tập test được dự đoán tuyển dụng đỗ hay trượt chính xác, được tính theo công thức sau:

## AREA UNDER ROC CURVE

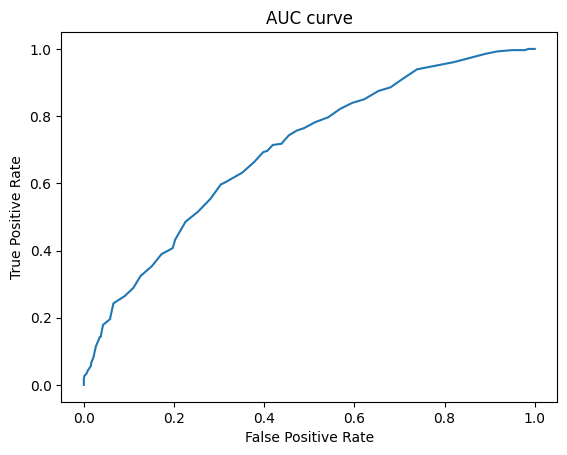
### Receiver operating characteristic (ROC) curve

True Positive Rate (TPR,) hay recall, là tỉ lệ những sample có ground truth là true sẽ được đánh nhãn true bởi model. Ở trường hợp này, đó là tỉ lệ những ứng viên đỗ mà được model gán nhãn là đỗ:

False Positive Rate (TPR) là tỉ lệ những sample có ground truth là false nhưng bị gán nhầm là true. Ở trường hợp này, đó là tỉ lệ những ứng viên trượt mà được model gán nhãn là đỗ:

Đây là một đồ thị của True Positive Rate (TPR) và False Positive Rate (FPR):

Với mỗi giá trị threshold từ 0 đến 1, chúng ta có các TPR và FPR khác nhau, và đồ thị ROC curve là một hàm (xấp xỉ) biểu diễn các điểm TPR và FPR đó.



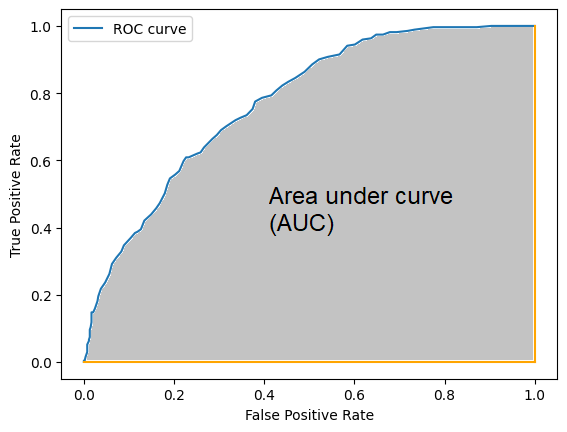
Một ví dụ của ROC curve

### AUC-ROC score

Là diện tích phần dưới của ROC curve.

Với một model rất tốt, thì tỉ lệ TPR phải càng cao càng tốt (gần 1.0) và tỉ lệ FPR phải càng thấp càng tốt (gần 0.0)

Dưới góc độ AUC-ROC score, thì model đánh nhãn tốt khi AUC-ROC score càng cao càng tốt



Hình ảnh minh họa cho AUC score

Việc sử dụng AUC-ROC score cho bộ dữ liệu có khoảng 1/2 tỉ lệ đỗ/trượt ở ground truth có thể sẽ không quá hiệu quả, vì vậy nhóm đã sử dụng thêm chỉ số AUC-PR ở dưới.

## F1 SCORE

### Precision và Recall

Precision là tỉ lệ số đánh nhãn true đúng trong thực tế. Ở bài toán này, đó là tỉ lệ những ứng viên được mô hình đánh nhãn đỗ mà thực sự đỗ:

Recall, hay true positive rate, là tỉ lệ những sample có ground truth là true sẽ được đánh nhãn true bởi model. Ở trường hợp này, đó là tỉ lệ những ứng viên đỗ mà được model gán nhãn là đỗ:

### Công thức tính F1 score

F1 score là trung bình điều hòa của Precision và Recall:

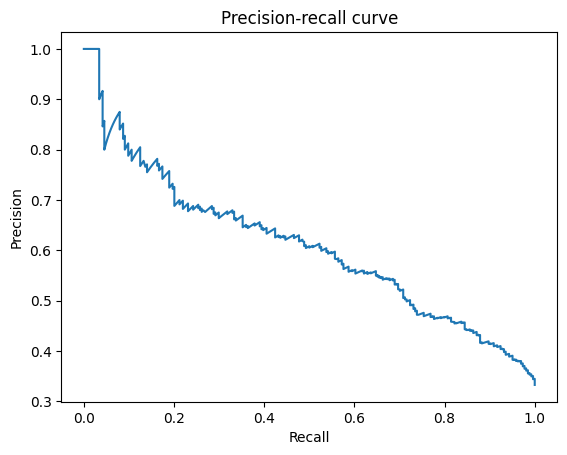
Vì ở đây nhóm trượt đỗ quan trọng như nhau nên nhóm quyết định để F1, chứ không thay đổi F-beta score.

F1 score dùng để tìm threshold hợp lý, do bài toán này có tỉ lệ true/false khoảng 1/2, nên việc dùng F1 score cùng với accuracy sẽ làm cho kết quả trở nên tổng quan hơn.

## AREA UNDER PR CURVE

### **Precision-Recall (PR) curve**

Đây là một đồ thị của Precision và Recall. Với mỗi giá trị threshold từ 0 đến 1, chúng ta có các Precision và Recall khác nhau, và đồ thị PR curve là một hàm (xấp xỉ) biểu diễn các điểm Precision và Recall khác đó:



Ví dụ của một Precision-Recall curve

### **AUC-PR score**

Là diện tích phần dưới của ROC curve.

Với một model rất tốt, thì tỉ lệ Precision phải càng cao càng tốt (gần 1.0) và tỉ lệ Recall cũng phải càng cao càng tốt (gần 0.0)

Dưới góc độ AUC-PR score, thì model đánh nhãn tốt khi AUC score càng cao càng tốt.

Việc sử dụng AUC-PR để kèm theo AUC-ROC score là do bộ dữ liệu có 1/2 tỉ lệ đỗ/trượt ở ground truth, nên AUC-ROC có thể sẽ không quá hiệu quả khi đứng riêng lẻ.

## DEMOGRAPHIC PARITY

Định nghĩa của Demographic parity, hay sự bình đẳng về nhân khẩu học, là tỉ lệ dự đoán true/false của bất cứ nhóm nào phải bằng nhau. Trong toán học, ta định nghĩa rằng:

with are groups in

Tuy nhiên, trong thực tế, không thể nào có thể đạt được 100% sự cân bằng như trên mà không đánh đổi đi một số chỉ số cần tối ưu khác (như accuracy.) Vì vậy, trong bài toán này, chúng em sử dụng Demographic Parity như là một chỉ số ngưỡng (threshold metric) như sau:

với

Chúng em để dự đoán do đây là bài toán phân loại nhị phân.

## EQUALIZED ODDS

Định nghĩa của Equalized odds, hay tỉ lệ thành công cân bằng, là tỉ lệ dự đoán true/false dựa trên điều kiện ground truth của bất cứ nhóm nào phải bằng nhau. Trong toán học, ta định nghĩa rằng:

with are groups in

Tuy nhiên, trong thực tế, không thể nào có thể đạt được 100% sự cân bằng như trên mà không đánh đổi đi một số chỉ số cần tối ưu khác (như accuracy.) Vì vậy, trong bài toán này, chúng em sử dụng Equalized odds như là một chỉ số ngưỡng (threshold metric) như sau:

với là tập các giá trị mà có thể xảy ra

Vì các giá trị có thể xảy ra là (true/false,) nên ta có thể viết lại biểu thức của Equalized odds như sau (vì là bài toán nhị phân nên ta giả sử :

Như vậy, công thức cuối cùng chúng em sử dụng là:

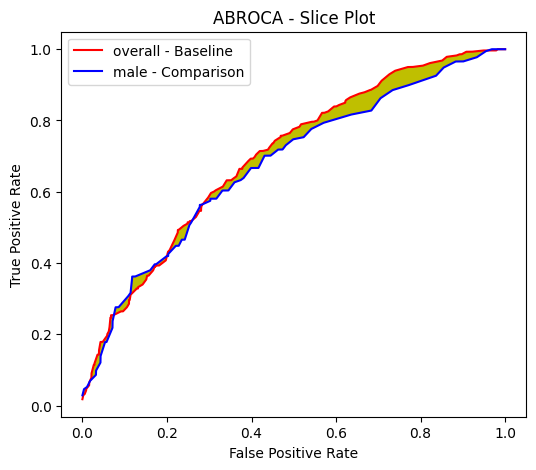
Các chỉ số TPR và FPR đều rất phụ thuộc vào threshold, vì vậy chỉ số ngưỡng này giúp chúng em quyết định threshold nào giúp cho model “công bằng.” Việc loại model không công bằng cũng sẽ sử dụng Equalized odds, nhưng sẽ kết hợp với một phiên bản AUC ở phần dưới.

## ABSOLUTE BETWEEN-ROC AREA (ABROCA)

Chỉ số Equalized odds của cùng một model có thể thay đổi qua threshold, vì vậy để đo sự “công bằng” mà không tính tới threshold thì chúng em sẽ sử dụng ABROCA score.

ABROCA đo lường sự khác biệt giữa ROC curve của các nhóm trên tất cả các threshold (0 đến 1) có thể. Cụ thể, nó đo sự khác biệt tuyệt đối giữa hai đường cong để nắm bắt trường hợp các đường cong có thể cắt nhau.

Ví dụ dưới đây thể hiện điểm ABROCA cần tính, đó là phần tô vàng sẫm:



ABROCA score giữa nhóm nam và nhóm toàn bộ model

Định nghĩa cụ thể của ABROCA score như sau:

Việc tính diện tích này sẽ đảm bảo model được “công bằng” một cách tổng quan hơn.

## HONORABLE MENTION: PREDICTIVE PARITY

Định nghĩa của Predictive Parity, hay sự cân bằng trong dự đoán, là tỉ lệ ground truth dựa trên dự đoán của tất cả các nhóm phải bằng nhau. Trong toán học, ta định nghĩa rằng:

Tuy nhiên, trong thực tế, không thể nào có thể đạt được 100% sự cân bằng như trên mà không đánh đổi đi một số chỉ số cần tối ưu khác (như accuracy.) Vì vậy, trong bài toán này, chúng em sử dụng Predictive Parity như là một chỉ số ngưỡng (threshold metric) như sau:

với là tập các giá trị mà có thể xảy ra

Vì các giá trị có thể xảy ra là (true/false,) nên ta có thể viết lại biểu thức của Predictive Parity như sau (vì là bài toán nhị phân nên ta giả sử :

Sau khi thử nghiệm thì chúng em nhận ra rằng là việc sử dụng NPV và precision có thể dẫn tới chia cho không khi sử dụng threshold quá nhỏ hoặc quá lớn (do FN + TN = 0 khi threshold quá nhỏ và FP + TP = 0 khi threshold quá lớn,) nên chúng em đã không sử dụng threshold này để làm

## BẢNG CHỈ SỐ

| **Model** | **threshold** | **F1** | **ACC** | **AUC-ROC** | **AUC-PR** | **DP** | **EO** | **ABROCA** | **Sensitive** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Full false** | N/A | N/A | 0.66 | 0.5 | 0.5 | 0 | 0 | 0 | N/A |
| **Logistic Regression\*** | 0.3 | 0.58 | 0.69 | 0.75 | 0.6 | 0.15 | 0.31 | 0.1 | Gender |
| 0.07 | 0.22 | 0.04 | Nationality |
| 0.15 | 0.31 | 0.05 | Age |
| **Logistic Regression**  **(no sensitive)** | 0.3 | 0.6 | 0.70 | 0.75 | 0.61 | 0.04 | 0.05 | 0.04 | Gender |
| 0.09 | 0.15 | 0.10 | Nationality |
| 0.11 | 0.07 | 0.22 | Age |
| **SVM** | 0.3 | 0.57 | 0.66 | 0.75 | 0.61 | 0.21 | 0.10 | 0.05 | Gender |
| 0.12 | 0.23 | 0.05 | Nationality |
| 0.16 | 0.31 | 0.06 | Age |
| **SVM**  **(no sensitive)** | 0.3 | 0.58 | 0.68 | 0.75 | 0.61 | 0.10 | 0.17 | 0.10 | Gender |
| 0.04 | 0.06 | 0.05 | Nationality |
| 0.13 | 0.26 | 0.07 | Age |
| **SVM rbf** | 0.3 | 0.74 | 0.79 | 0.91 | 0.84 | 0.08 | 0.15 | 0.02 | Gender |
| 0.03 | 0.04 | 0.04 | Nationality |
| 0.13 | 0.18 | 0.03 | Age |
| **SVM rbf**  **(no sensitive)** | 0.3 | 0.75 | 0.81 | 0.91 | 0.84 | 0.21 | 0.44 | 0.16 | Gender |
| 0.05 | 0.11 | 0.04 | Nationality |
| 0.11 | 0.14 | 0.05 | Age |
| **Decision Tree (incl. no sensitive)** | 0.3 | 0.58 | 0.70 | 0.74 | 0.58 | 0.11 | 0.23 | 0.11 | Gender |
| 0.12 | 0.25 | 0.05 | Nationality |
| 0.17 | 0.25 | 0.05 | Age |
| **Random Forest** | 0.3 | 0.79 | 0.84 | 0.93 | 0.86 | 0.22 | 0.53 | 0.15 | Gender |
| 0.01 | 0.08 | 0.06 | Nationality |
| 0.13 | 0.17 | 0.06 | Age |
| **Random Forest**  **(no sensitive)** | 0.3 | 0.79 | 0.84 | 0.92 | 0.86 | 0.23 | 0.55 | 0.11 | Gender |
| 0.05 | 0.23 | 0.05 | Nationality |
| 0.11 | 0.14 | 0.03 | Age |
| **Gradient Boosting** | 0.3 | 0.79 | 0.84 | 0.92 | 0.86 | 0.22 | 0.53 | 0.15 | Gender |
| 0.01 | 0.08 | 0.05 | Nationality |
| 0.17 | 0.13 | 0.06 | Age |
| **Gradient Boosting**  **(no sensitive)** | 0.3 | 0.8 | 0.85 | 0.93 | 0.88 | 0.28 | 0.63 | 0.20 | Gender |
| 0.06 | 0.10 | 0.03 | Nationality |
| 0.12 | 0.16 | 0.06 | Age |

Như vậy, chúng em chọn mô hình SVM\_rbf, vì các chỉ số ngưỡng (fairness) đều dưới mức cho phép, và có các chỉ số tối ưu cao nhất. Với các mô hình bỏ các sensitive features, chúng em hiện đang tạm thời bỏ qua do chỉ số tối ưu quá thấp, còn các model khác thì lại vi phạm các chỉ số ngưỡng.

Ngoài ra, chúng em còn đưa ra một số kết luận sau:

* Gender dễ bị bias nhất, age thì có bị ảnh hưởng một chút
* Nationality thì không bị chịu ảnh hưởng quá nhiều, có thể là do quốc tịch đều thuộc quốc gia thu nhập cao
* Khi sử dụng các thuật toán nhằm tăng độ chính xác của mô hình lên, đã gây nên sự unfair trong model → bản chất model là unfair

# PHẦN VI - NGUỒN, TÀI LIỆU THAM KHẢO

Fairness of Machine Learning

* Link: <https://arxiv.org/pdf/2012.15816.pdf>
* Focus: Part 2.1, page 3

A survey on datasets for fairness‐aware machine learning

* Link: [A survey on datasets for fairness‐aware machine learning - Le Quy - 2022 - WIREs Data Mining and Knowledge Discovery - Wiley Online Library](https://wires.onlinelibrary.wiley.com/doi/10.1002/widm.1452#widm1452-bib-0068)
* Focus: Part 2.3

Fair Clustering via Equitable Group Representations

* Link: <https://dl.acm.org/doi/pdf/10.1145/3442188.3445913>

Fairlearn

* Main: [Fairlearn](https://fairlearn.org/)
* Evaluation metrics: [Common fairness metrics](https://fairlearn.org/main/user_guide/assessment/common_fairness_metrics.html)

Logistic Regression

* Hosmer, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). \*Applied Logistic Regression.\* Wiley.
* Cox, D. R. (1958). \*The regression analysis of binary sequences.\* Journal of the Royal Statistical Society: Series B (Methodological), 20(2), 215-242.

Support Vector Machine (SVM)

* Cortes, C., & Vapnik, V. (1995). Support-vector networks. Machine learning, 20(3), 273-297.
* Hearst, M. A., Dumais, S. T., Osuna, E., Platt, J., & Scholkopf, B. (1998). Support vector machines. IEEE Intelligent Systems and their Applications, 13(4), 18-28.

Support Vector Machine with Radial Basis Function Kernel (SVM rbf)

* Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In Proceedings of the fifth annual workshop on Computational learning theory (pp. 144-152). ACM.
* Scholkopf, B., & Smola, A. J. (2002). Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. MIT Press.

Decision Tree (including no sensitive)

* Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). Classification and regression trees. CRC press.
* Quinlan, J. R. (1986). Induction of decision trees. Machine learning, 1(1), 81-106.

Random Forest

* Breiman, L. (2001). Random forests. Machine learning, 45(1), 5-32.
* Liaw, A., & Wiener, M. (2002). Classification and regression by randomForest.\* R news, 2(3), 18-22.

Gradient Boosting

* Friedman, J. H. (2001). \*Greedy function approximation: A gradient boosting machine.\* The Annals of Statistics, 29(5), 1189-1232.
* Chen, T., & Guestrin, C. (2016). \*XGBoost: A Scalable Tree Boosting System.\* In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 785-794).