INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES TOULOUSE

Rapport Apprentissage Supervisé

LEMMERS
Nathan
OSORNIO
Patrick
5-SDBD
2023-2024

Rapport Apprentissage Supervisé

Code source disponible sur : https://github.com/posornio/TPApprentisage2

1) Présentation des données et du dataset

Nous analyserons le dataset ACSincome dans le but de prédire si le salaire d'une personne donnée sera supérieur à 50 000 dollars par an.
Nos modèles seront des modèles de classification où on cherche à prédire la classe 'PINCP', qui détermine si l'individu étudié a un salaire supérieur à 50 000 $ annuels.
On étudie leur âge, leur domaine de travail, leur dernier diplôme scolaire, leur état marital, leur occupation, où ils ont été nait, leur sexe et finalement leur ethnicité.

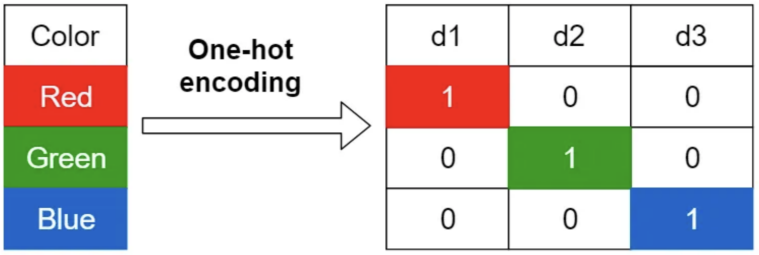
2) Création de pipeline

Pour améliorer nos prédictions, il est impératif de traiter nos données avant de lancer les entraînements. On reconnaît deux types de features à traiter: les classes et les valeurs numériques.
Pour traiter efficacement les valeurs numériques, il est nécessaire de les centrer et réduire. Cependant, certaines données peuvent posséder des valeurs extrêmes ou 'outliers' qui ne correspondent pas aux cas habituels et peuvent donc produire une réduction qui n'est pas représentative de nos données. Nous avons donc choisi d'utiliser un Robust Scaler dont la formule est:

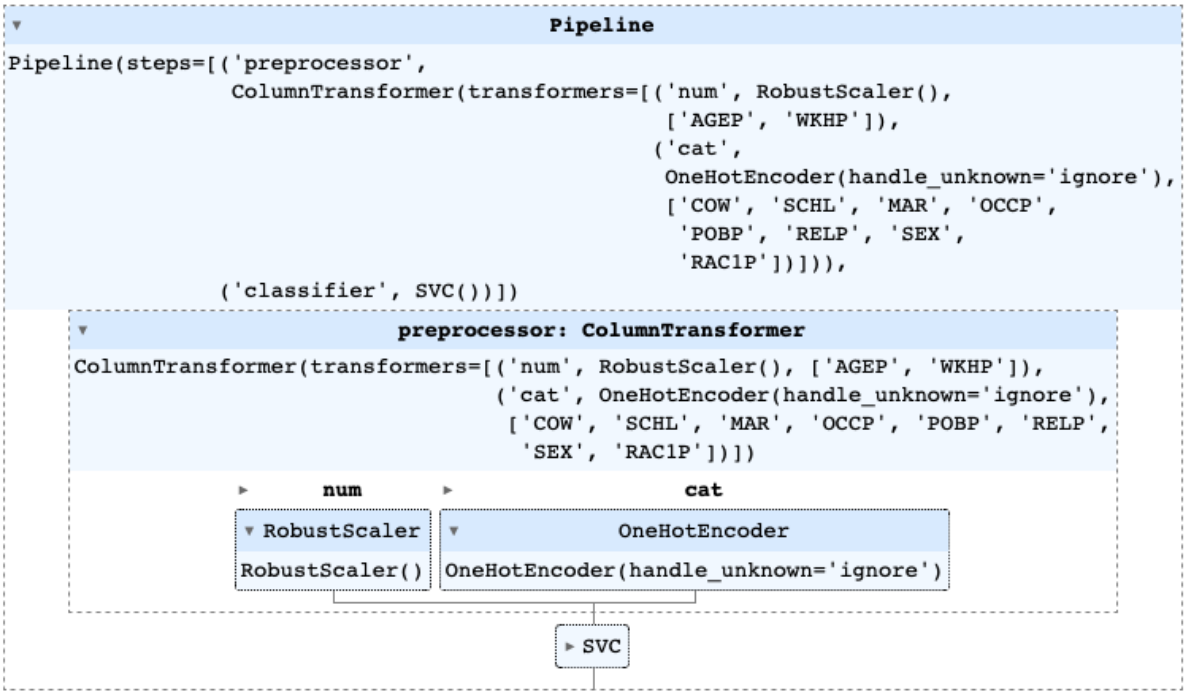
$$X_{\text{scale}} = \frac{x_i - x_{\text{med}}}{x_{75} - x_{25}}$$

Les valeurs seront centrées en fonction de la médiane et non pas à la moyenne, car est moins sensible aux valeurs extrêmes.

Dans un même temps, les valeurs représentant une classe doivent aussi être traités, surtout lorsqu'on traite des classes tels que le sexe ou l'ethnicité. Nos classes ne peuvent pas avoir un ordre empirique, sinon, on risquerait de biaiser notre modèle et lui dire qu'un sexe ou un groupe ethnique vaut plus qu'un autre.
Pour éviter cela, on utilise un OneHotEncoder. Chaque classe est représentée par une colonne dans une matrice, 1 si on appartient à cette classe, 0 sinon.



On à donc créé une pipeline qui fait ce traitement de données avant d'entraîner un modèle. Voici notre exemple de pipeline pour le modèle SVC.



3) Grid Search

a) Hyperparamètres

Pour trouver le meilleur modèle possible, nous effectuons une recherche en grille (GridSearchCV). Le GridSerachCV effectue une validation par cross validation pour déterminer quel estimateur est le meilleur. Par défaut, on utilise un cross-val de 5. De cette manière, on divise le dataset en 5, et testons sur chacune des 5 fractions en entraînant sur les 4 parties restantes. De plus, on va calculer la justesse (accuracy) de nos modèles un faisant une prédiction avec les données de X_test et en la comparant à y_test.
Finalement, on évalue où notre modèle se trompe en regardant la matrice de confusion de notre prédiction.

	Predicted 0	Predicted 1
Actual 0	TN	FP
Actual 1	FN	TP

TN et TP représentent les négatifs et positifs prédits correctement, et FN et FP les positifs et négatifs prédits incorrectement.

Finalement, on utilisera aussi la métrique “classification_report”, qui donne plusieurs informations tel que le recall (TP/TP+FN), la précision (TP/TP + FP) et le F1 score (moyenne harmonique entre le recall et la précision). Le macro average est la moyenne des métriques alors que le weighted average est la moyenne pondérée qui prend en compte le nombre d’échantillons dans chaque classe. Les classes plus répandues ont un impact plus important. Support indique le numéro d'occurrences dans le dataset.

b) Résultats

Nous commençons par rechercher un modèle de type SVC. On recherche sur le paramètre de régularisation (C) , le kernel utilisé et le paramètre de contraction.
On obtient ce résultat en utilisant ces paramètres

```
param_grid_SVM = {
    'classifier__C' : [0.01,0.1,1,10,100],
    'classifier__kernel' : ["linear", "poly", "rbf", "sigmoid"],
    'classifier__shrinking' : [True, False]
}
```

```
Accuracy test:  0.7836286201022147
[[28152  6288]
 [ 6413 17847]]
              precision    recall  f1-score   support

           0       0.81      0.82      0.82     34440
           1       0.74      0.74      0.74     24260

    accuracy                           0.78     58700
   macro avg       0.78      0.78      0.78     58700
weighted avg       0.78      0.78      0.78     58700
```

On peut noter que le modèle SVM détecte mieux la classe 0, correspondant au salaire bas.

Par la suite, nous avons fait le même processus en utilisant Random Forest en ajustant la taille maximale, le critère, la taille des nœuds et la taille du diviseur de noeuds.

```
param_grid_RF = {
    'classifier__max_depth' : np.arange(1,1501,300),
    'classifier__criterion' : ["gini","entropy","log_loss"],
    'classifier__min_samples_leaf' : np.arange(1,16,2),
    'classifier__min_samples_split': np.arange(2,16,2)
}
```

```
Accuracy test:  0.7899829642248722
[[29912  4910]
 [ 7418 16460]]
              precision    recall  f1-score   support

           0       0.80      0.86      0.83     34822
           1       0.77      0.69      0.73     23878

    accuracy                           0.79     58700
   macro avg       0.79      0.77      0.78     58700
weighted avg       0.79      0.79      0.79     58700
```

Ensuite, on a lancé un GridSearch sur le modèle AdaBoost.
AdaBoost nous permet de commencer la recherche sur un estimateur différent, tels que SVC , RFC, KNeighbors, LogisticRegressor, etc..
On change le nombre maximum d'estimateurs, le poids donné à chaque estimateur et l'algorithme utilisé.

```
param_grid_ADA = {
    'classifier__n_estimators' : np.arange(1,91,30),
    'classifier__learning_rate' : np.arange(0,3, 1),
    'classifier__algorithm' : ['SAMME', 'SAMME.R'],
    'classifier__estimator' : [SVC(), RandomForestClassifier(), KNeighborsClassifier(), LogisticRegression() ]
}
```

```
Accuracy test:  0.7771890971039183
[[29237  5585]
 [ 7494 16384]]
              precision    recall  f1-score   support

           0       0.80      0.84      0.82     34822
           1       0.75      0.69      0.71     23878

    accuracy                           0.78     58700
   macro avg       0.77      0.76      0.77     58700
weighted avg       0.78      0.78      0.78     58700
```

Le dernier modèle testé sera le Gradient Boosting Classifier. Le seul paramètre nouveau est le critère de classification, qui indique quel sous quel critère on va mesurer la qualité d'un split.

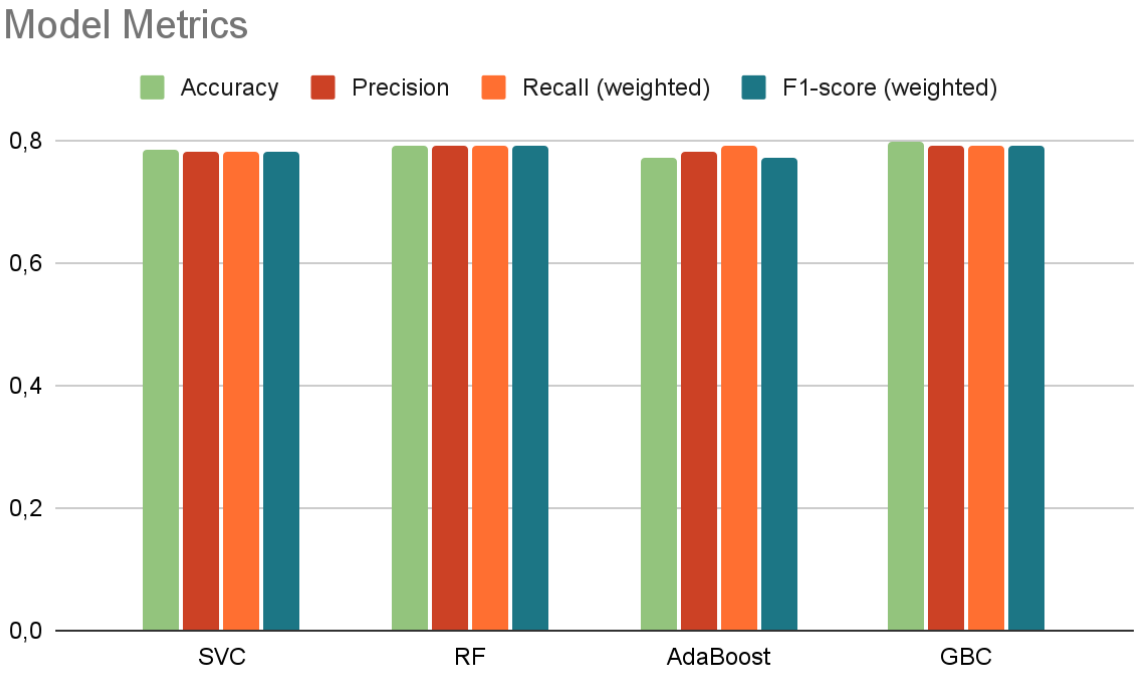
```
param_grid_GBC = {
    'classifier__n_estimators' : np.arange(1,401,50),
    'classifier__learning_rate' : np.arange(0.0,2.0, 0.2),
    'classifier__min_samples_split' : np.arange(2,22, 5),
    'classifier__criterion' : ['friedman_mse', 'squared_error']
}
```

```
Accuracy test:  0.7872572402044293
[[29140  5682]
 [ 6806 17072]]
              precision    recall  f1-score   support

           0       0.81      0.84      0.82     34822
           1       0.75      0.71      0.73     23878

    accuracy                           0.79     58700
   macro avg       0.78      0.78      0.78     58700
weighted avg       0.79      0.79      0.79     58700
```

Notons que l'on pourrait avoir un plus gros écart et des meilleures performances si on n'avait pas découpé initialement le dataset. Tous nos modèles semblent avoir des performances similaires si on se base uniquement sur l'accuracy. Cependant, les performances peuvent ne pas être les mêmes, d'où le besoin d'utiliser plusieurs métriques. Les métriques plus importantes sont souvent dépendantes du contexte, par exemple, il est plus problématique qu'un test Covid soit faussement négatif que faussement positif.

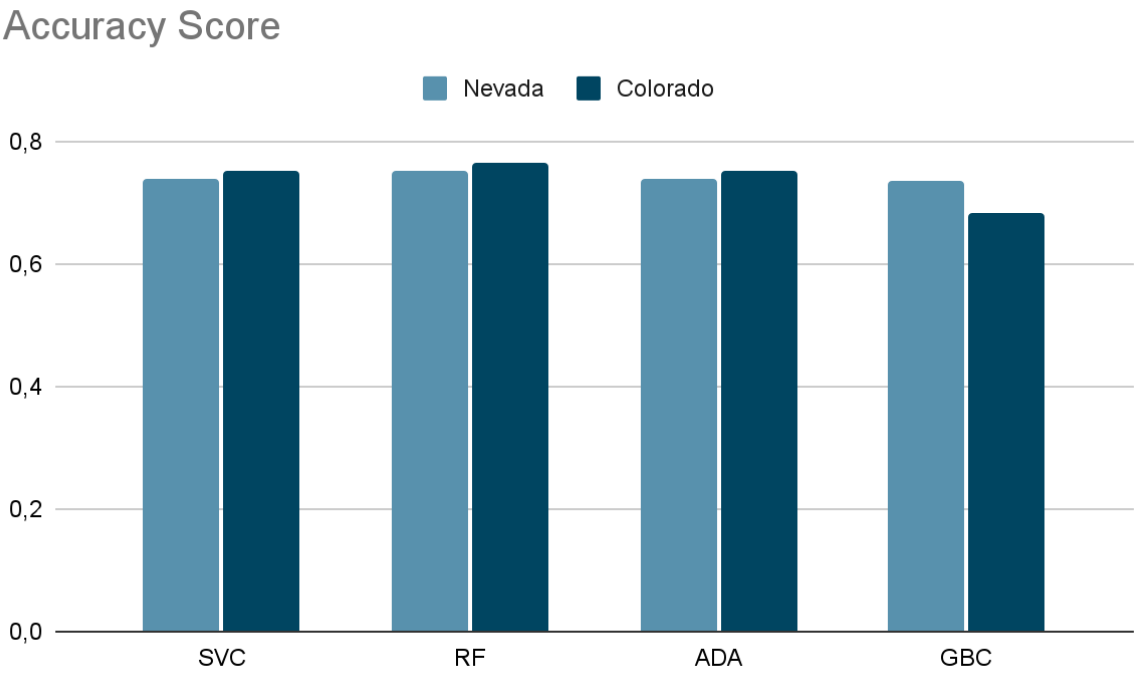
4) Comparaison des modèles

Tous nos modèles semblent avoir des performances similaires si on se base uniquement sur l'accuracy. Cependant, les performances peuvent ne pas être les mêmes, d'où le besoin d'utiliser plusieurs métriques. Les métriques plus importantes sont souvent dépendantes du contexte, par exemple, il est plus problématique qu'un test Covid soit faussement négatif que faussement positif.
Nous comparerons plus précisément les différents modèles dans les parties suivantes, mettant en avant leur précision sur des données venant d'une autre ville, et en observant les biais générés par nos entraînements.



5) Prédiction sur Nevada et Colorado

Pour trouver le modèle le plus performant, nous avons décidé de les évaluer sur des régions différentes telles que Nevada et Colorado.



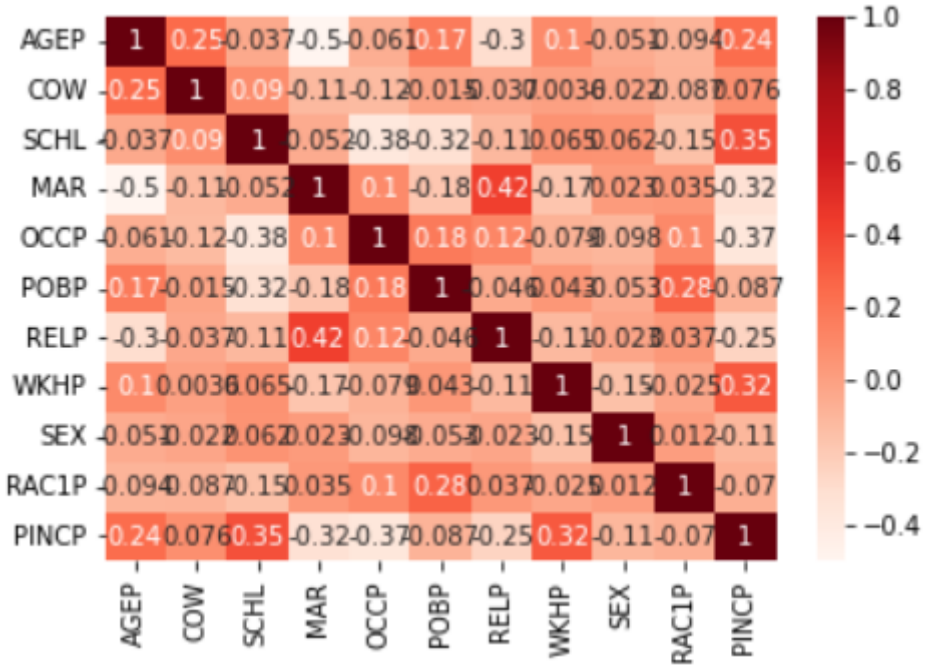
Les modèles sont moins performants sur des datasets différents, cela peut être dû à une différence de patrons dans ces régions ou des raisons culturelles ou sociales.
Cependant, ce test nous permet de voir que le modèle entraîné avec l'algorithme Random Forest est celui qui s'adapte le mieux.

6) Explicabilité des modèles

Dans le reste de ce rapport, ADA fait référence à AdaBoostClassifier, RF à RandomForest, SVM à Support Vector Machine, et GBC à Gradient Boosting Classifier.

a) Matrice de corrélation du dataset

Afin de comprendre les liens entre plusieurs features, il est possible de calculer une matrice de corrélation. Celle-ci, se calcule via la fonction corr(), qui nous renvoie une matrice de valeur telle que nous l'avons ci-dessous :

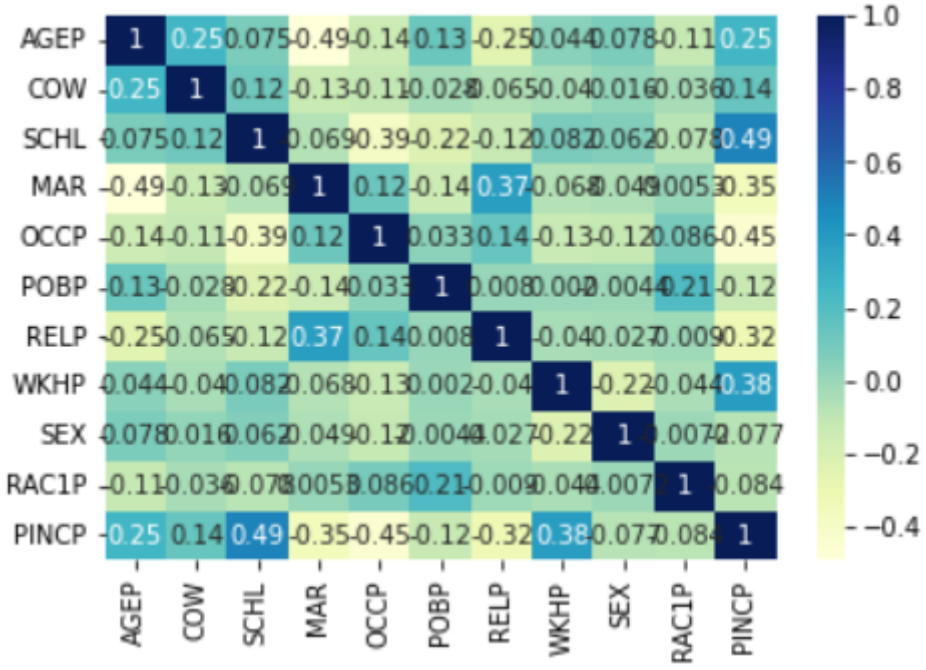


Tout d'abord, on observe que cette matrice est symétrique, puisque les mêmes features se retrouvent en abscisse et en ordonnée. Ensuite, la diagonale est toujours égale à 0, puisque les features sont évidemment corrélées avec elles-mêmes.
On observe que plus la valeur est éloignée de 0, plus les features sont liées. En effet, si le résultat est proche de 1, alors les valeurs sont proportionnelles, et si le résultat est proche de -1, alors elles sont inversement proportionnelles. Elles sont donc liées dans les deux cas.

Finalement, en ajoutant le label à notre matrice, on peut observer les corrélations entre celui-ci et les autres features. Dans notre cas, le label correspond à la colonne PINCP. Et on peut observer qu'il est corrélé avec OCCP, SCHL et WKHP. Ces features correspondent à l'occupation, la graduation scolaire, et le nombre moyen d'heures par semaine de travail.

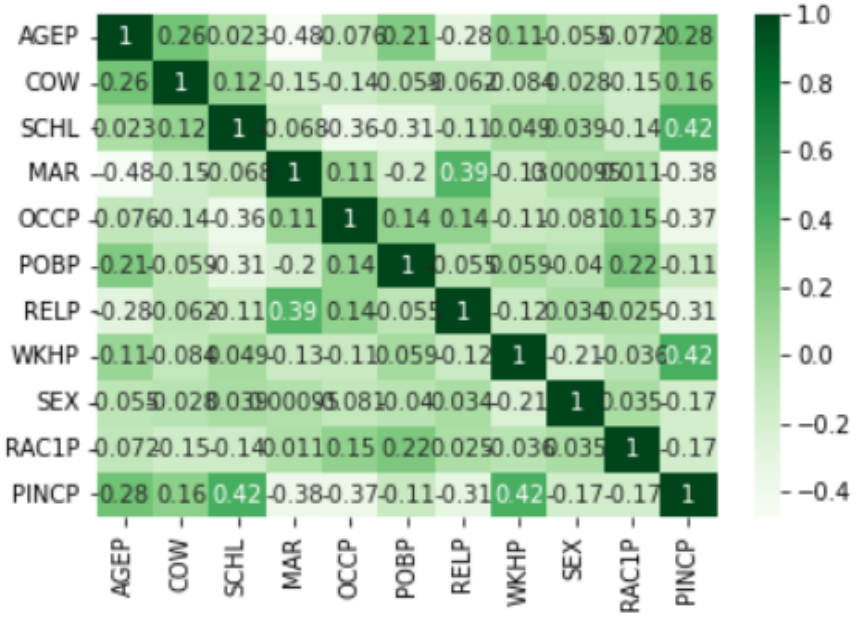
b) Matrice de corrélation sur nos prédictions

Il semble désormais intéressant d'observer le résultat des matrices de corrélations de nos modèles. Pour cela, il suffit de changer la colonne du label, ici PINCP, par la prédiction de notre modèle. Voici le premier résultat dans le cas de SVM :



On observe une grande similarité dans le cas des features importantes, mais avec un score plus grand, ce qui implique que notre modèle à surinterpréter l'importance de nos meilleures features. On l'observe notamment avec SCHL, qui a un score de 0.49, alors qu'il n'était que de 0.35 dans les données initiales.

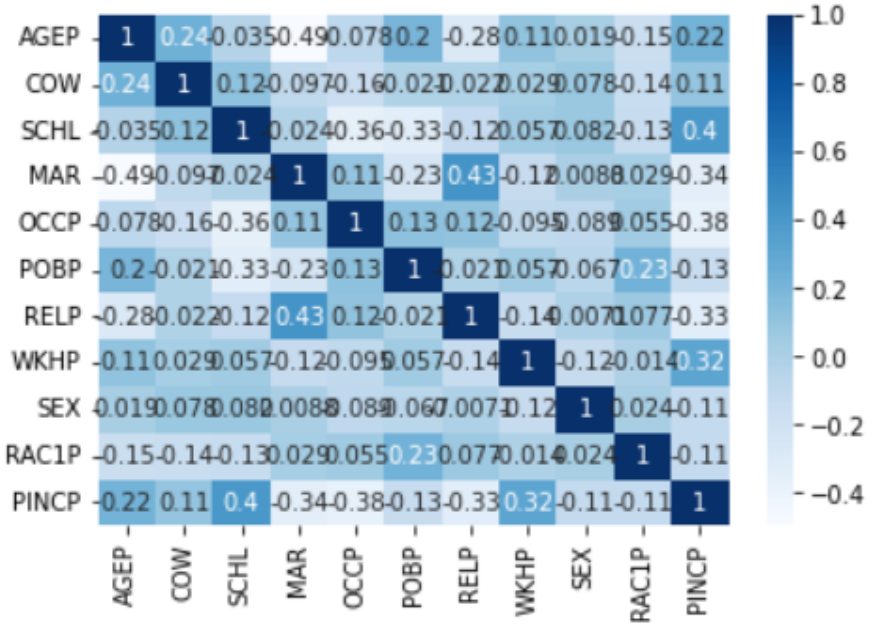
Observons maintenant le résultat pour le modèle ADA :



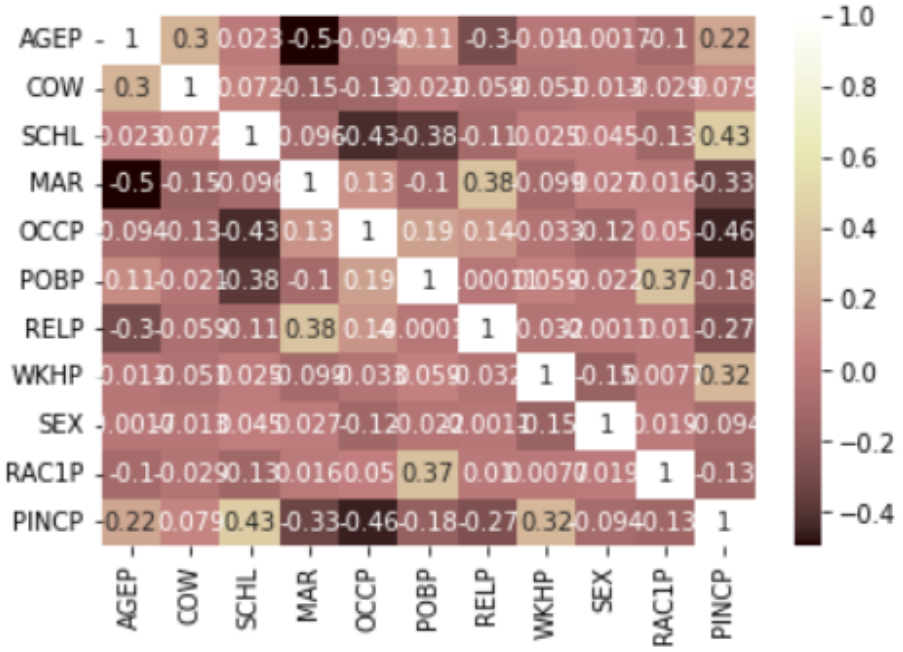
On retrouve le même phénomène que pour SVM, mais aussi que la feature MAR, correspondant au statut marital, est l'une des features principale, devançant OCCP. Notre modèle semble avoir surestimé l'importance de cette feature, tout comme celle du genre, SEX, et de la race, RAC1P.

En comparaison, les modèles RF et GBC semblent avoir bien analysé les features, et se retrouvent avec une matrice de corrélation proche de l'originale. On note cependant une légère différence pour RAC1P.

RF :



GBC :



c) Permutation importance de nos modèles

Une autre manière de mesurer l'importance des features pour un modèle est d'utiliser la fonction permutation_importance de sklearn. Celle-ci permet de tester le modèle en supprimant les colonnes une à une, pour mesurer l'importance de celle-ci dans la prédiction. Ainsi, cette méthode est indépendante du modèle, et permet de détecter efficacement les features, en observant l'erreur mesurée quand le modèle n'a plus une certaine feature.

Voici le premier résultat pour le modèle ADA :

```
SCHL    0.074 +/- 0.013
WKHP    0.050 +/- 0.011
RELP    0.029 +/- 0.010
POBP    0.029 +/- 0.009
MAR     0.025 +/- 0.009
SEX     0.016 +/- 0.007
```

Nous observons que 2 features semblent être primordiales. La première est SCHL, qui est liée à la graduation scolaire. Tandis que le second est WKHP, pour le taux de travail par semaine. Cela conforte le résultat de la matrice de corrélation, où ces deux features étaient les plus importantes. De plus, OCCP ne se trouve pas dans les 6 features les plus importantes, comme mis en avant dans la partie précédente pour ce modèle. On note aussi la présence de la donnée SEX, que nous reverrons dans la partie suivante.

Observons maintenant le résultat pour les modèles SVM, GBC, et RF.

SVM

```
SCHL    0.072 +/- 0.012
WKHP    0.034 +/- 0.009
OCCP    0.028 +/- 0.008
RELP    0.023 +/- 0.009
POBP    0.020 +/- 0.007
```

GBC

```
OCCP    0.111 +/- 0.010
SCHL    0.089 +/- 0.010
WKHP    0.061 +/- 0.008
POBP    0.039 +/- 0.009
AGEP    0.027 +/- 0.008
RELP    0.024 +/- 0.005
COW     0.020 +/- 0.005
MAR     0.019 +/- 0.007
```

RF

```
SCHL    0.104 +/- 0.009
OCCP    0.064 +/- 0.005
WKHP    0.064 +/- 0.008
MAR     0.059 +/- 0.009
RELP    0.055 +/- 0.007
POBP    0.038 +/- 0.007
AGEP    0.035 +/- 0.008
SEX     0.032 +/- 0.007
RAC1P   0.024 +/- 0.007
COW     0.016 +/- 0.005
```

Ces résultats semblent cohérents avec ceux de la matrice de corrélation. La contribution de SCHL, WKHP et OCCP ont bien été mises en avant par nos modèles, même si leurs valeurs diffèrent.

d) Conclusion explicabilité

Dans cette partie, nous avons montré deux manières différentes d'observer les features mises en avant dans nos modèles. Ces deux méthodes présentées sont indépendantes du modèle choisi, et sont donc très utiles pour des modèles complexes.
Nos modèles ont donc été capable de déterminer assez précisément la contribution de certaines données, et l'on observe que les deux qui ont été meilleurs, ici RF et GBC, sont aussi ceux avec les meilleurs résultats dans la prédiction sur les données de test, ce qui confirme leur précision dans la compréhension et le traitement des données.

7) Equité des modèles

Mesurer l'équité de nos modèles est l'une des parties les plus importantes de leurs évaluations. En effet, les méthodes d'apprentissage supervisé fonctionnent avec des gros

jeux de données, et visent à être efficaces sur la majorité. Cependant, cela peut facilement mener à des inégalités de traitement quand l'on travaille sur des minorités.
Dans notre cas, nous allons évaluer l'équité de notre modèle sur deux features, celle du genre, SEX, et de l'origine, RAC1P.

a) Importance du genre

i) Matrice de confusion et métrique d'équité statistique

Pour commencer, nous avons voulu évaluer l'équité de notre modèle via une matrice de confusion. Pour cela, nous avons divisé nos données de test et de train en deux parties, une première partie pour les hommes, et une deuxième pour les femmes. Il est nécessaire de préciser que dans notre cas, 53,49% des données étaient des hommes, et 46,51%, des femmes.
Avec nos deux matrices de confusions, nous sommes capables de calculer 3 métriques d'équité statistiques différentes. Tout d'abord, le **taux de prédiction positives**, qui calcule le nombre de fois où notre modèle à fait une prédiction positive. Nous avons ensuite **le taux de positif détecté**, c'est-à -dire à son taux de détection des résultats positifs. Et **taux de négatif détecté**, qui correspond à sa détection de résultats négatifs.

Voici le type de matrice de confusion :

		Predicted	
		Negative **(N)** -	Positive **(P)** +
Actual	Negative -	True Negative **(TN)**	**False Positive (FP) Type I Error**
	Positive +	**False Negative (FN) Type II Error**	True Positive **(TP)**

En se basant sur cette structure, voici le détails de nos calculs par rapport à la matrice de confusion :

Taux de prédiction négative : (True negative + False Negative)/Total
Taux de négatif détecté : True Negative / (True Negative + False Positive)
Taux de positif détecté : True Positive / (True Positive + False Negative)

Voici le résultat obtenu en détails :

			ADA	RF	GBC	SVM
Taux de prédiction négatives	Femmes	Train	0,71	0,71	0,67	0,62
		Test	0,71	0,7	0,67	0,61
	Hommes	Train	0,55	0,67	0,56	0,57
		Test	0,55	0,57	0,55	0,56
Taux de négatifs détecté	Femmes	Train	0,87	0,89	0,86	0,81
		Test	0,88	0,89	0,86	0,81
	Hommes	Train	0,81	0,83	0,81	0,83
		Test	0,8	0,82	0,81	0,83
Taux de positifs détectés	Femmes	Train	0,6	0,63	0,67	0,74
		Test	0,6	0,63	0,68	0,75
	Hommes	Train	0,74	0,73	0,74	0,73
		Test	0,75	0,74	0,74	0,72

Ici, on observe que notre modèle a tendance à donner un salaire plus bas aux femmes, car le taux de prédiction négative est particulièrement élevé chez les femmes et non chez les hommes. Cependant, on observe aussi que le taux de détection des négatifs et des positifs est plutôt stable pour les deux. Cela reste cohérent avec les features importantes mises en avant pour ce modèle dans la partie précédente, le genre n'étant pas décisif.
Dans notre cas, il est à noter que le modèle le plus équitable semble être celui de SVM, même si son accuracy est plus basse, elle est plus stable entre les hommes et les femmes.

ii) Training sans la feature SEX et résultats

Afin d'affiner nos résultats, nous avons décidé de tester un training sans la colonne SEX pour notre meilleur modèle, GBC. Ainsi, nous avons appliqué les mêmes paramètres et effectué un training. Rappelons que notre accuracy était précédemment de 0.80.
Voici le nouveau résultat obtenu et mesuré avec les mêmes métriques que dans la partie précédente, en remettant le genre correspondant à chaque exemple :

```
Matrice de confusion homme :
 [[31952  6620]
 [ 9388 24435]]
Matrice de confusion femme :
 [[35033  6903]
 [ 6251 16383]]
Taux de prédiction négative pour hommes :  0.5710339111817114
Taux de négatif détecté :  0.8283729129938816
Taux de positive détecté :  0.7224373946722644
Taux de prédiction négative pour femmes :  0.6393681276134427
Taux de négatif détecté :  0.8353920259442961
Taux de positive détecté :  0.7238225678183264
Accuracy test:  0.7841396933560477
```

On observe une amélioration du biais, le taux de prédiction négative a baissé chez les femmes. Cela a mené à une détection des négatifs plus faible et à un augmentation de détection des positifs, équilibrant ainsi les données, et réduisant le biais qui était observé.
Il serait désormais intéressant de faire la même chose sur le modèle RF, qui avait noté une utilisation de la feature SEX comme importante dans la partie précédente. Voici le résultat après un training sans la feature SEX sur l'ensemble de test :

```
Matrice de confusion homme :
 [[33529  5153]
 [ 9580 24025]]
Matrice de confusion femme :
 [[36415  5658]
 [ 6465 16140]]
Taux de prédiction négative pour hommes :  0.5963589580422483
Taux de négatif détecté :  0.8667855850266274
Taux de positive détecté :  0.7149233744978426
Taux de prédiction négative pour femmes :  0.662976591731346
Taux de négatif détecté :  0.8655194542818435
Taux de positive détecté :  0.7140013271400133
Accuracy test:  0.8050596252129472
```

On note que le taux de prédiction négative a grandement diminué chez les femmes, passant de 0,7 à 0,59. De plus, l'accuracy semble n'avoir pas changé et l'équité semble exemplaire, avec une détection de négatif et de positif égal entre les hommes et les femmes.

iii) Bilan d'équité du genre

Dans cette partie, nous avons mis en avant l'un des biais de notre algorithme. Bien que léger, celui-ci semble avoir impacté certains de nos modèles, principalement ADA. Cependant, notre modèle le plus précis, construit via GBC, semble avoir été le moins touché par ces biais. On l'observe notamment dans cette dernière partie avec le training sans la colonne SEX, qui donne pourtant un résultat très similaire.

b) Importance de l'ethnicité

i) Détection des valeurs de RAC1P biaisées

De même que pour le genre, l'origine de la personne peut avoir biaisé notre algorithme, et il est important de le mesurer. Cependant, il semble trop lourd d'utiliser la même méthode que pour le SEX pour les mettre en avant, car il n'y a pas que 2 valeurs possibles pour RAC1P.
Pour cela, nous avons décidé de tester nos modèles en fixant les valeurs de RAC1P à chaque valeur possible. Ainsi, nous avons pu détecter dans quels cas l'accuracy devenait basse.
Nous avons remarqué qu'en fixant cette feature, l'accuracy n'était pas trop perturbée, sauf pour *Hawaï* et *Other race alone*. Dans ces cas, l'accuracy en test tendait à baisser significativement, et c'est donc sur ces deux valeurs que nous avons concentré le reste de nos mesures, même s' il serait plus précis de le faire pour chaque valeur.

ii) Matrice de confusion et métrique d'équité statistique

Nous avons appliqué la même méthode que dans la partie précédente pour afficher les résultats pour les valeurs de race de *hawaï* et *other race alone.*
Voici notre résultat :

			ADA	RF	GBC	SVM
Taux de prédiction négatives	Hawaï	Train	0,77	0,78	0,92	0,73
		Test	0,79	0,82	0,93	0,76
	Other race alone	Train	0,91	0,95	0,88	0,88
		Test	0,65	0,65	0,6	0,58
Taux de négatifs détecté	Hawaï	Train	0,88	0,9	0,96	0,85
		Test	0,87	0,9	0,96	0,84
	Other race alone	Train	0,96	0,94	0,94	0,94
		Test	0,84	0,84	0,8	0,78
Taux de positifs détectés	Hawaï	Train	0,47	0,47	0,18	0,56
		Test	0,4	0,4	0,16	0,45
	Other race alone	Train	0,27	0,18	0,38	0,34
		Test	0,69	0,7	0,77	0,79

Dans cette partie, les biais sont bien plus marqués. En effet, même si cela reste très variable entre les modèles, le taux de prédiction négatives est particulièrement haut, ce qui implique que notre modèle avait tendance à supposer un salaire bas. Ensuite, le taux de négatifs détecté est anormalement élevé comparé à l'accuracy des modèles, et cela s'explique par un taux de positif détecté très bas. Nos modèles ont donc tendance à toujours donner un salaire bas à nos exemples, même quand cela n'est pas correct.
Cette fois encore, c'est SVM qui se distingue, avec des résultats biaisés, mais moins que pour les autres modèles.

iii) Training sans la feature RCA1P et résultats

Afin d'affiner nos résultats, nous avons de nouveau effectué un training sur notre modèle présentant la meilleure accuracy jusqu'à présent, GBC. Ce training a été effectué sans la colonne RAC1P. Voici les résultats obtenus :

```
Matrice de confusion hawai :
 [[128  16]
 [ 17  35]]
Matrice de confusion other race alone :
 [[1336  242]
 [ 274  640]]
Taux de prédiction négative pour hawai :  0.7397959183673469
Taux de négatif détecté :  0.8888888888888888
Taux de positif détecté :  0.6730769230769231
Taux de prédiction négative pour other race alone :  0.6460674157303371
Taux de négatif détecté :  0.8466413181242078
Taux de positif détecté :  0.700218818380744
Accuracy test:  0.7878023850085178
```

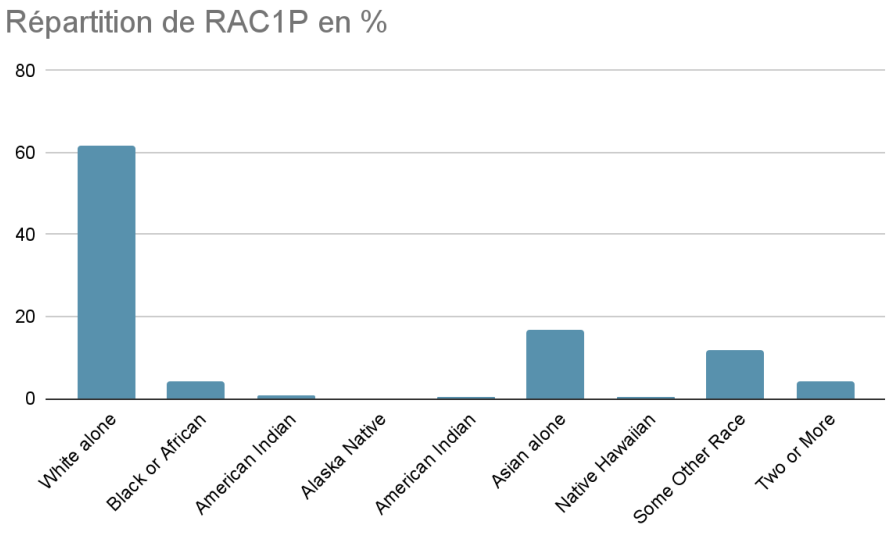
On observe ici une augmentation de l'équité statistique via un taux de prédiction négative qui a baissé pour *Hawaï* et *Other race alone,* avec une accuracy pourtant stable comparé à l'entraînement avec la feature RAC1P.
De plus, le même phénomène que pour le genre est observé, la détection de négatif a diminué, pendant que la détection de positif a grandement augmenté.
On constate donc que supprimer cette feature a permis une réduction significative du biais induit.
Comme dans la partie précédente, nous avons décidé de reproduire la même expérience avec le modèle RF, qui avait mis en avant son utilisation de la feature RAC1P. Voici le résultat obtenu après training sur l'ensemble de test :

```
Matrice de confusion hawai :
 [[106  20]
 [ 22  32]]
Matrice de confusion other race alone :
 [[1471  169]
 [ 277  619]]
Taux de prédiction négative pour hawai :  0.7111111111111111
Taux de négatif détecté :  0.8412698412698413
Taux de positif détecté :  0.5925925925925926
Taux de prédiction négative pour other race alone :  0.6892744479495269
Taux de négatif détecté :  0.8969512195121951
Taux de positif détecté :  0.6908482142857143
Accuracy test:  0.8068143100511074
```

L'accuracy du modèle reste inchangée, même si on observe une légère amélioration au niveau des détection de positifs, le modèle semble toujours très biaisé. Avec un taux de négatif supérieur à celui positif. Nous verrons dans la conclusion une des causes potentielles de cela.

iv) Bilan pour l'équité

Il a été intéressant de voir et d'évaluer les différents biais de nos modèles. Nous avons découvert à quel point il était complexe d'évaluer l'impact de certaines données précises sur un training, et comprenons désormais l'importance que ces inégalités peuvent mener par la suite, via des prédictions bonnes seulement pour la majorité.
Si nous voulions améliorer la précision de nos modèles tout en réduisant ces biais, il semble nécessaire d'améliorer la qualité du dataset. En effet, nous avons remarqué que notre modèle était bien plus impacté par la feature RAC1P que par SEX, et cela est en partie dû à la répartition de nos données. Voici la répartition des données dans RAC1P :



Avec une telle répartition, il semble problématique d'utiliser cette feature dans le training de nos modèles.