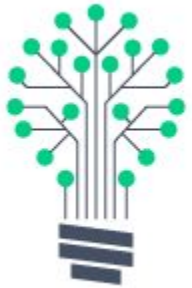


Supervised learning project

Construction et évaluation d'un modèle de classification supervisée

Sandrine Henry - 9/11/2019

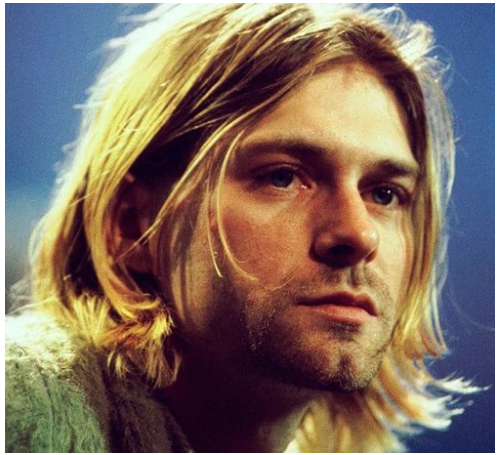
Sommaire

- Contexte
- Source du dataset
- Objectif
- Analyse et nettoyage du dataset
- Machine learning
- Conclusion

Les Etats-Unis en 1994

Président : Bill Clinton (D)
Vice-Président : Al Gore (D)

Mort de Kurt Cobain

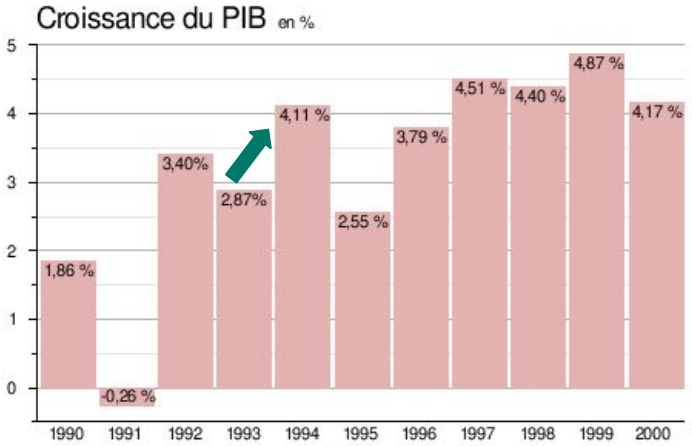


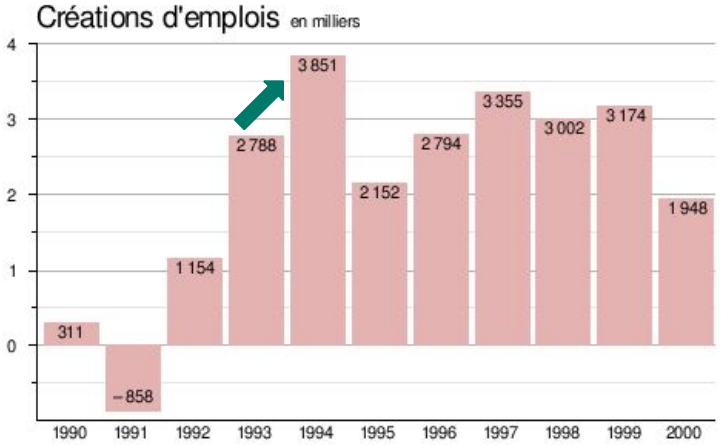
Mais c’est aussi...

… l’année qui connaît un **rebond économique**

- **263,4** millions d’habitants
- **1,2 million d’émigrants** aux États-Unis, dont probablement **300 000 illégaux** (40 % d’Hispaniques, 40 % d’Asiatiques).
- **22,6 millions** de personnes nés à l’étranger, soit 8,7 % de la population totale.
- **4,25 %** = croissance du PIB atteinte au premier semestre 1994, propulsée par une consommation dynamique.
- **3,85 millions d'emplois créés**, un record qui ne fut atteint à nouveau et dépassé qu'en 2015.
- **Retour de la confiance des ménages** américains qui puisent dans leur épargne pour financer leurs achats (notamment de biens durables).

=> Taux d'intérêt favorables et hausse des revenus encouragent les achats d'automobiles et de biens d'équipement de la maison.





Année	Croissance du PIB	Créations d'emplois (en milliers)	Solde public (en milliards)	Indice de confiance des consommateurs[5]	Évolution de la production industrielle[6]
1990	1,86 %	311	-221,2	81,6	0,99 %
1991	−0,26 %	-858	-269,3	77,5	−1,50 %
1992	3,4 %	1 154	-290,4	77,3	2,87 %
1993	2,87 %	2 788	-255,1	82,8	3,27 %
1994	4,11 %	3 851	-203,2	92,3	5,23 %
1995	2,55 %	2 152	-164,0	92,2	4,65 %
1996	3,79 %	2 794	-107,5	93,6	4,53 %
1997	4,51 %	3 355	-22,0	103,2	7,16 %
1998	4,4 %	3 002	69,2	104,6	5,83 %
1999	4,87 %	3 174	125,6	105,8	4,44 %
2000	4,17 %	1 948	236,4	107,6	3,90 %
Légende	Meilleur résultat de la décennie Pire résultat de la décennie				

Le Dataset

Le dataset sur lequel j'ai travaillé provient du UCI Machine Learning Repository il se compose de données issues du recensement de 1994 aux Etats-unis.

Source : https://archive.ics.uci.edu/ml/datasets/census+income

Objectif

Prédire si le revenu dépasse 50 000 $ / an
en fonction des données du recensement.

Méthode

La colonne « Income » nous permettra de répondre à cette question.
Elle compte 2 valeurs :
>50k$ ou <50k$.

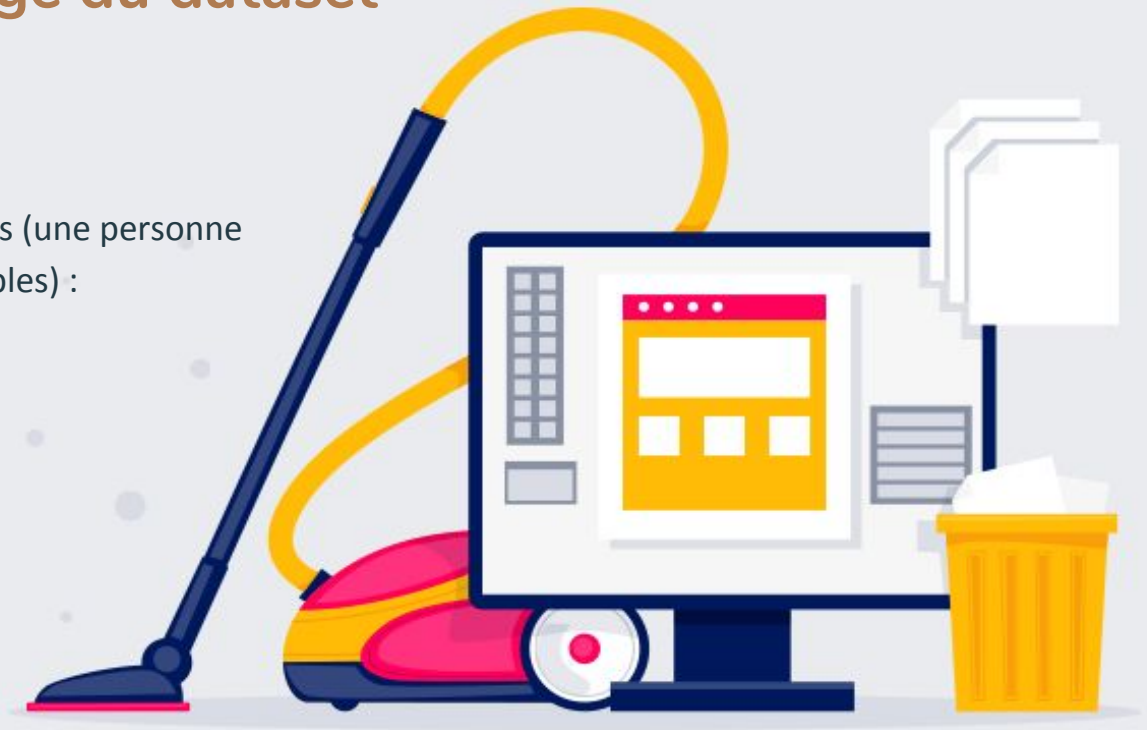
C’est une donnée dite **discrète**, la méthode de ML choisie sera de type **classification** : régression logistique, SVM, forest tree etc…

Analyse et nettoyage du dataset

Le dataset contient 32 561 lignes (une personne recensée) et 15 colonnes (variables) :

- 6 numériques
- 9 catégorielles

- 14 sont les features
- 1 la target : “Income”



Analyse et nettoyage du dataset : les colonnes

Socio-demo

age
race : White, Asian-Pac-Islander, Amer-Indian-Eskimo, Other, Black.
sex : Female, Male
native-country : United-States, Cambodia, England, Puerto-Rico, Canada, Germany, …

Education

education : Bachelors, Some-college, 11th, ...
education-num → équivalent numérique

Marital-status

marital-status : Married-civ-spouse, Divorced, Never-married, Separated, Widowed, ...
relationship : Wife, Own-child, Husband, Not-in-family, Other-relative, Unmarried.

Work status

workclass : Private, Self-emp-not-inc, Self-emp-inc,...,Without-pay, Never-worked.
occupation : Tech-support, Craft-repair, Other-service, Sales, Exec-managerial, Prof-specialty, ….
hours-per-week

Capital

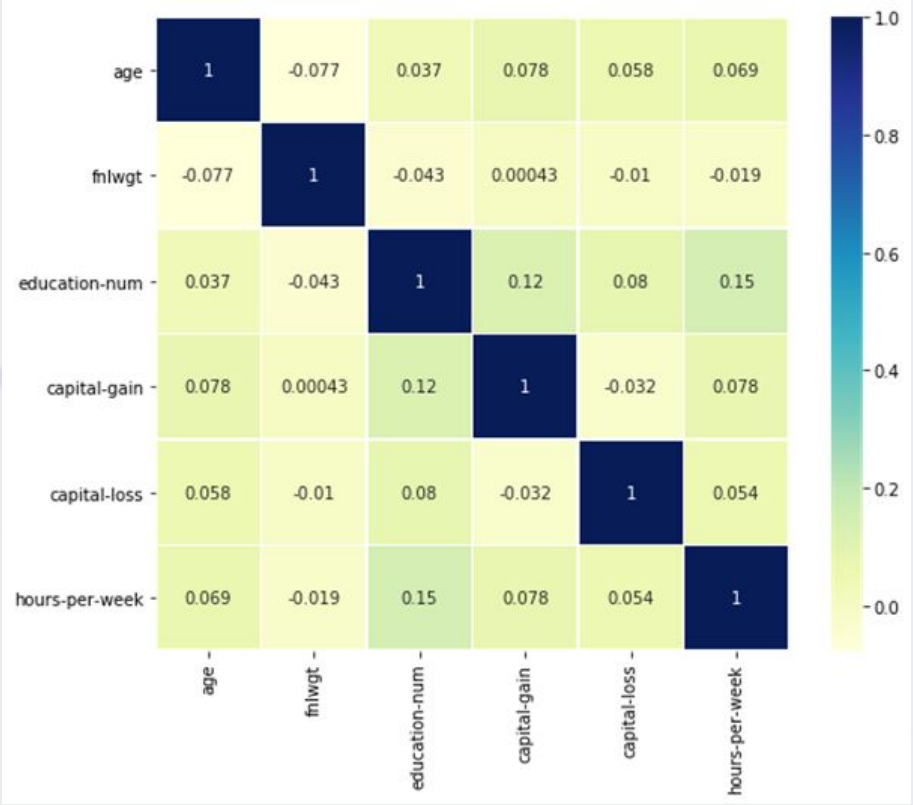
capital-gain
capital-loss

Autre

fnlwgt

Corrélation des valeurs numériques

But : supprimer d'éventuelles colonnes qui auraient un effet sur le modèle.



Variables catégorielles + traitement des valeurs manquantes

La variable "native-country"

```
In [8]: print('Nb de pays:',adult_data['native-country'].value_counts().count()) ,adult_data['native-country'].
        value_counts()

        Nb de pays: 42

Out[8]: (None,  United-States                29170
         Mexico                                643
         ?                                     583
         Philippines                           198
         Germany                               137
         Canada                                121
         Puerto-Rico                           114
         El-Salvador                           106
         India                                 100
         Cuba                                   95
         England                                90
         Jamaica                                81
         South                                  80
         China                                  75
         Italy                                  73
         Dominican-Republic                     70
         Vietnam                                67
         Guatemala                              64
         Japan                                  62
         Poland                                 60
         Columbia                               59
         Taiwan                                 51
         Haiti                                  44
         Iran                                   43
         Portugal                               37
         Nicaragua                              34
         Peru                                   31
         France                                 29
         Greece                                 29
         Ecuador                                28
         Ireland                                24
         Hong                                   20
         Cambodia                               19
         Trinadad&Tobago                        19
         Laos                                   18
         Thailand                               18
```

```
Out[13]:  United-States    29170
          America           1536
          Asia               671
          Europe             521
          Name: group_country, dtype: int64
```

Les valeurs manquantes

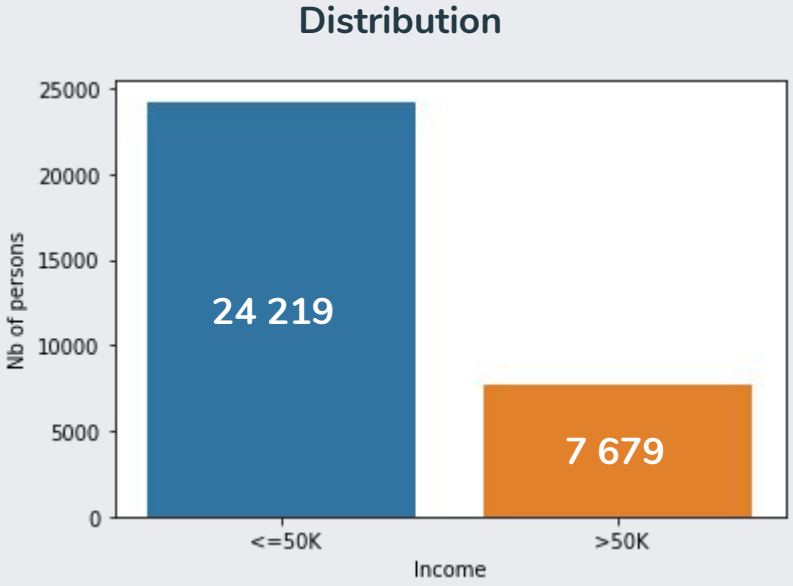
Pour les autres variables, je n'ai rien vu de particulier. Mis à part qu'il faudra vérifier s'il n'y a pas de valeurs manquantes.

```
In [25]: adult_data.isna().sum().sum()

Out[25]: 0
```

Il n'y a pas de valeurs manquantes. Je peux passer à la partie ML

La variable cible : la colonne “Income”



Je passe cette colonne en **ordinal** pour faciliter les calculs.

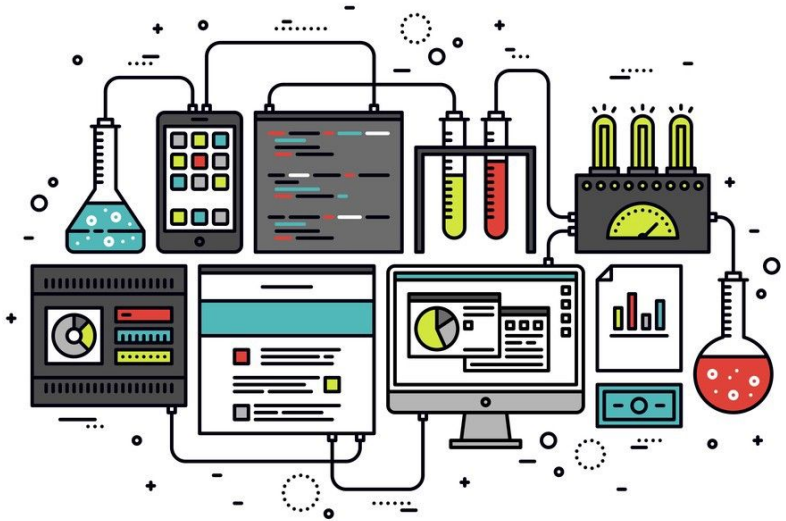
```
adult_data['income_ordinal'] = adult_data['income'].map(lambda x: 1 if x==' >50K' else 0)
adult_data['income_ordinal'].value_counts()
```

```
0    24219
1     7679
Name: income_ordinal, dtype: int64
```

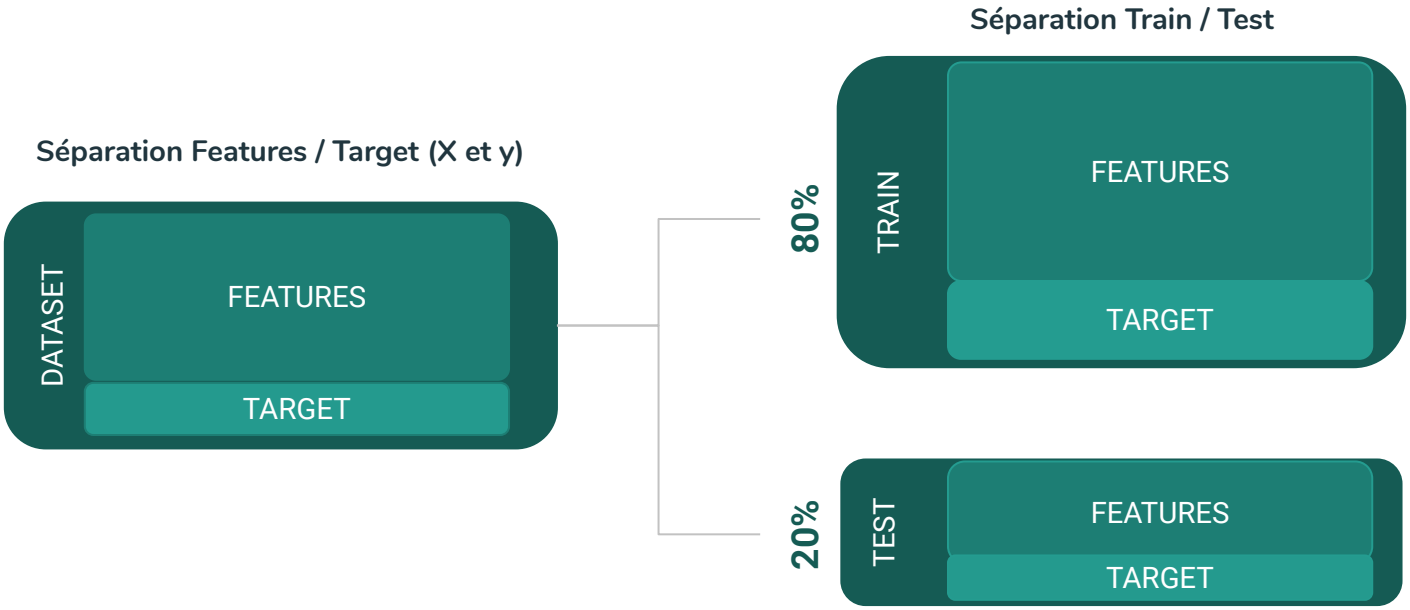
MACHINE LEARNING

Machine learning : Rappel du processus

- Phase d'***apprentissage*** : construction du modèle
- Phase de ***test*** : consiste à prédire l'étiquette des données de test, connaissant le modèle préalablement appris.
- ***Comparaison*** des prédictions obtenues grâce au modèle avec les vrais étiquettes pour juger la performance du modèle.



Machine learning : Préparation des données



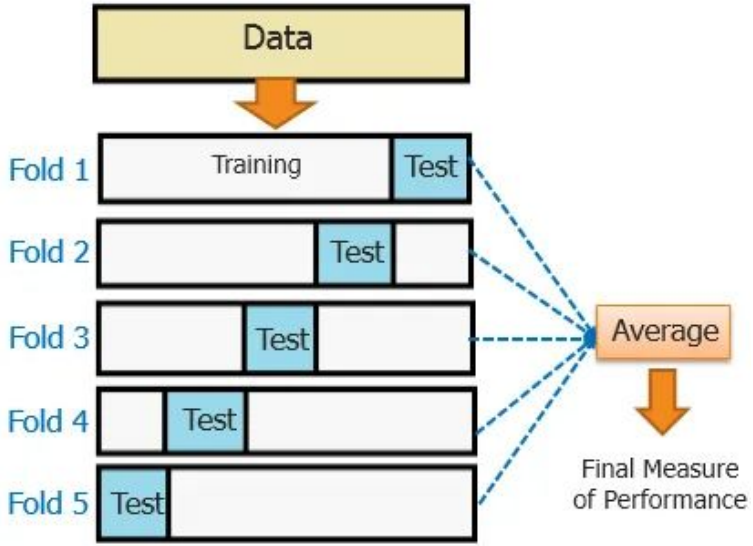
Sélection du modèle via la Cross-Validation 1/3

Je décide de passer par la méthode de **cross-validation** pour choisir mon modèle de prédiction.

La validation croisée (en français) est une méthode qui permet d'estimer les performances de généralisation de plusieurs modèles à partir de l'ensemble d'apprentissage.

Cette estimation pourra être comparée à l'estimation obtenue sur l'ensemble de test mis de côté au départ.

La cross validation consiste à recommencer sur plusieurs découpages train/test différents du jeu de données initial de manière à **s'assurer que la prédiction est *stable* et éviter l'*overfitting***.



Sélection du modèle via la Cross-Validation 2/3

```
from sklearn.model_selection import cross_val_score
from sklearn import metrics

print('Liste des scores existants dans le module sklearn:\n\n',metrics.SCORERS.keys())
```

```
Liste des scores existants dans le module sklearn:

 dict_keys(['explained_variance', 'r2', 'max_error', 'neg_median_absolute_error', 'neg_mean_absolute_err
or', 'neg_mean_squared_error', 'neg_mean_squared_log_error', 'accuracy', 'roc_auc', 'balanced_accuracy',
'average_precision', 'neg_log_loss', 'brier_score_loss', 'adjusted_rand_score', 'homogeneity_score', 'co
mpleteness_score', 'v_measure_score', 'mutual_info_score', 'adjusted_mutual_info_score', 'normalized_mut
ual_info_score', 'fowlkes_mallows_score', 'precision', 'precision_macro', 'precision_micro', 'precision_
samples', 'precision_weighted', 'recall', 'recall_macro', 'recall_micro', 'recall_samples', 'recall_weig
hted', 'f1', 'f1_macro', 'f1_micro', 'f1_samples', 'f1_weighted', 'jaccard', 'jaccard_macro', 'jaccard_m
icro', 'jaccard_samples', 'jaccard_weighted'])
```

Critère de sélection de modèle choisi : **Accuracy**

Nb de bonnes prédictions (TP)

l'ensemble des prédictions (TP+FP+TN+FN).

Je vais tester les modèles suivant : **LogisticRegression**, **DecisionTreeClassifier**, **KNeighborsClassifier** et **RandomForestClassifier**.

```
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier

print("Rf:")
accuracy = cross_val_score(RandomForestClassifier(n_estimators=100, max_depth=2,random_state=0), X_trai
n, y_train, scoring='accuracy', cv = 10)
print(accuracy)
print("Accuracy of Random Forest is: " , accuracy.mean()*100)

print("\n\nLog:")
#print(cross_val_score(log_class, X_train, y_train, scoring='accuracy', cv = 10))
accuracy = cross_val_score(LogisticRegression(solver='liblinear'), X_train, y_train, scoring='accuracy
', cv = 10)
print(accuracy)
print("Accuracy of LogisticRegression is: " , accuracy.mean()*100)

print("\n\nK Neighbors:")
#print(cross_val_score(knn_class, X_train, y_train, scoring='accuracy', cv = 10))
accuracy = cross_val_score(KNeighborsClassifier(n_neighbors=3), X_train, y_train, scoring='accuracy', c
v = 10)
print(accuracy)
print("Accuracy of K Neighbors is: " , accuracy.mean()*100)

print("\n\nDecision Tree:")
#print(cross_val_score(dtc_class, X_train, y_train, scoring='accuracy', cv = 10))
accuracy = cross_val_score(DecisionTreeClassifier(),X_train, y_train, scoring='accuracy', cv = 10)
print(accuracy)
print("Accuracy of Decision Tree is: " , accuracy.mean()*100)
```

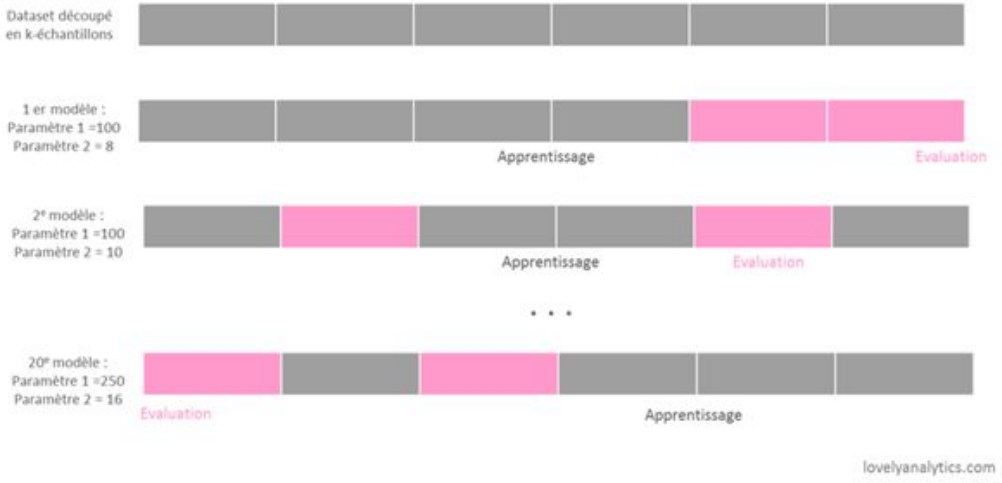
Sélection du modèle via la Cross-Validation 3/3

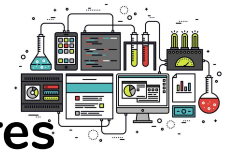
Random Forest: [0.76811594 0.76929103 0.76920063 0.77625392 0.77468652 0.77037618 0.76871815 0.76832615 0.76675813 0.77224618] **Accuracy of Random Forest is:** **77.03972821957758**	K Neighbors: [0.76184881 0.75597336 0.75352665 0.75352665 0.75744514 0.74177116 0.76048608 0.76754214 0.74794198 0.75891807] **Accuracy of K Neighbors is:** **75.58980041578805**
Logistic Regression: [0.80101841 0.78926753 0.78605016 0.80525078 0.79153605 0.79663009 0.80086241 0.79459036 0.79223834 0.80517444] **Accuracy of LogisticRegression is:** **79.62618565675866**	Decision Tree: [0.81629456 0.80846063 0.80525078 0.82131661 0.81347962 0.81543887 0.81066249 0.81575853 0.81144649 0.80399843] **Accuracy of Decision Tree is:** **81.22107018316989**

Au regarde des scores je présélectionne les modèles **Decision Tree** et **Logistic Regression**.

Grid search : Optimisation du modèle

Principe : tester une série de paramètres et comparer les performances pour en déduire le meilleur paramétrage.





Grid Search & Cross-Validation #2 incluant les hyper-paramètres

Decision Tree

Evaluation de 3 paramètres :

- **criterion** : la qualité du split.
- **splitter** : la stratégie de la méthode du split.
- **max_depth** : la profondeur maximale de l'arbre.

Hyper-paramètres retenus :

- criterion : **'entropy'**
- splitter : **'best'**
- max_depth : **10**

Accuracy of Decision Tree is:
85.5003499642416

Logistic Regression

Evaluation 2 paramètres :

- **C** : Inverse de la force de régularisation; doit être un flottant positif. Des valeurs plus petites spécifient une régularisation plus forte.
- **penalty** : norme utilisée dans la pénalisation

Hyper-paramètres retenus :

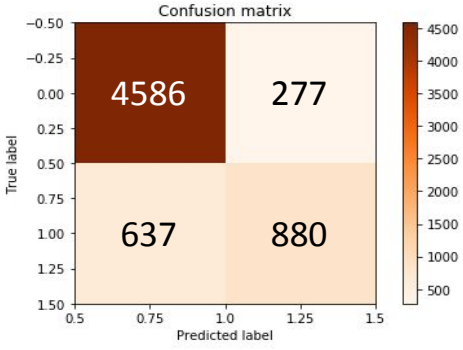
- C : **1.0**
- penalty : '**l1**'

Accuracy of LogisticRegression is:
85.21046728477069

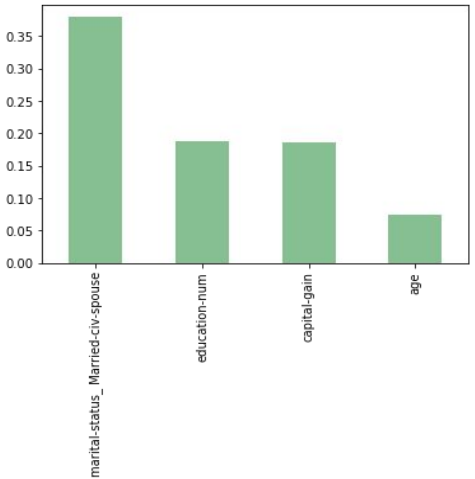
Au regarde des scores le modèle le plus probant est celui du **Decision Tree**.

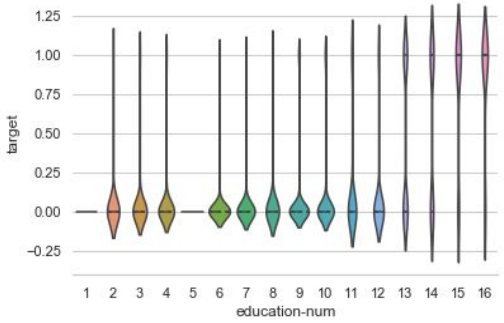
Travail avec le modèle de Decision Tree 3/3 : Graphiques

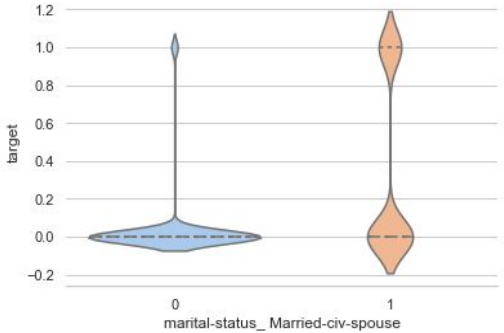
Matrice de confusion



Feature importances







Comparaison Train, Test

Matrices de Confusion

TRAIN

[[18453 903]
[2477 3685]]

TEST

[[4586 277]
[637 880]]

	accuracy score	precision score	recall score	F1 score
train	86.75	80.32	59.8	68.56
test	85.67	76.06	58.01	65.82

Un modèle qui effectivement apporte une amélioration dans la classification des True Positive mais une moins bonne prédiction des True Negative.

Des scores qui permettent de dire qu'il n'y a ni overfitting, ni underfitting.

Piste d'avancement :

- faire une PCA pour réduire le nombre de colonnes / T-sne
- tester une autre modèle
- regarder les lignes où il y a les erreurs

Conclusion

Plus le niveau d'étude est élevé, plus l'éventualité est grande d'avoir un revenu de plus de 50k$.

De même, l'âge augmente la chance de gagner plus.

Enfin, d'après ce modèle de classification, c'est d'abord le fait d'être marié qui est un facteur déterminant du revenu.

https://github.com/sandrineh/data-labs/tree/master/module-3/supervised-learning-project/your-code