Challenge - Prédiction d'embauche

(Il s'agit d'un exercice basé sur un jeu de donné purement fictif)

Compte rendu de Sébastien Chapeland

Table des matières

1. Description du contexte

"Le jeu de données contenu dans data.csv décrit des candidatures au poste de chercheur d'or chez Orfée. L'objectif consiste à prédire le succès ou l'échec d'une candidature."
Dans ce contexte, il faut distinguer les différents enjeux liés à la tâche. Le taux d'erreur nul (100% de succès) n'étant pas atteignable, il faut choisir quel type d'erreur doit être minimisé :

- Rejeter un bon candidat à tort (erreur de première espèce)
- ou recruter un mauvais candidat à tort (erreur de 2e espèce)

Ainsi nous pourrons opter pour un classifieur plus ou moins "optimiste" qui va systématiquement rejeter plus de candidats (à tort) mais acceptera des mauvais candidats avec un faible taux d'erreur ou inversement ou un classifeur équilibré entre les deux suivant le besoin. Pour cette raison il sera bon d'étudier non pas la justesse du modèle (accuracy) mais plutôt le taux de Précision et de Rappel. Il y aura nécessairement un trade-off à faire pour minimiser un typer d'erreur plutôt qu'un autre.

2. Exploration des données

Une première approche du jeu de données nous permet de visualiser le type de données auxquels nous avons à faire. On constate 20.000 observations. Des types de données différents :

- Catégorielles (cheveux, sexe, diplôme, spécialité, dispo)
- Numériques continues (salaire, note)
- Numériques discrètes (âge, expérience)
- Format date (date)

Certaines données sont manquantes, on peut voir l'apparition de NaN. Cela peut poser problème par la suite, il faudra pouvoir les corriger. Il est intéressant aussi de constater que le jeu de données est très déséquilibré : il n'y a que 11,5% de cas positifs.

Choix de la métrique d'évaluation :

Dans ce cas particulier il faudra choisir une méthode de scoring qui prenne en compte un tel déséquilibre. Une bonne métrique dans ce cas est l'aire sous la courbe de Precision-Recall (rappel en abscisse et précision en ordonnées) qui est plus appropriée que la courbe ROC. C'est celle-là que nous choisirons.

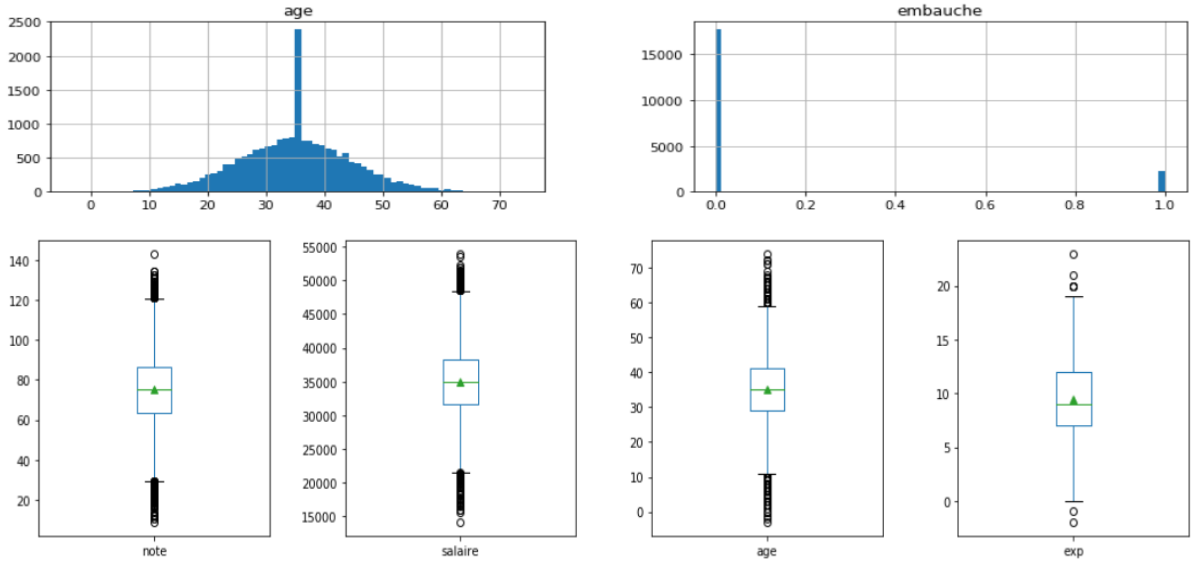
3. Distribution des données

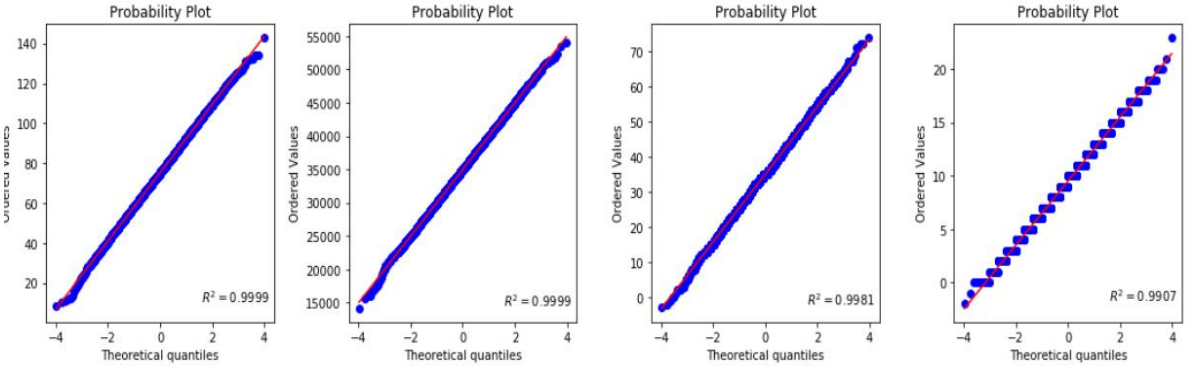
Données Quantitatives

A l'observation de la distribution des données quantitatives on constate que les données semblent gaussiennes. Néanmoins on peut voir un pic au niveau de l'âge autour de la valeur 35 ans qui pourrait contredire cette hypothèse. La distribution normale des données est souvent une hypothèse préalable à d'autres tests que nous réaliserons par la suite, notamment pour l'analyse de la variance (ANOVA). Pour appuyer cette hypothèse, nous pouvons tracer un QQ-plot qui permet une visualisation simple et rapide de la normalité. Néanmoins il ne s'agit pas d'un test fiable. Pour cette raison nous procèderons à un test de normalité des variables. Certaines variables n'étant pas continues, nous utiliserons le test de "scipy.stats.normaltest" qui est basé sur la méthode D'Agostino et Pearson qui permet de tester la normalité dans le cas de variables discrètes contrairement à un test de Shapiro-Wilk ou Jarque-Bera par exemple.
H0: Les variables proviennent d'une loi normale :

- si la p-value est inférieure au risque alpha de 5%, alors l'hypothèse nulle est rejetée.
- si la p-value est supérieure au risque alpha de 5%, alors on ne doit pas rejeter l'hypothèse nulle

Au regard des p-value nous conclurons que les variables proviennent bien d'une loi normale, sauf pour la variable (âge).

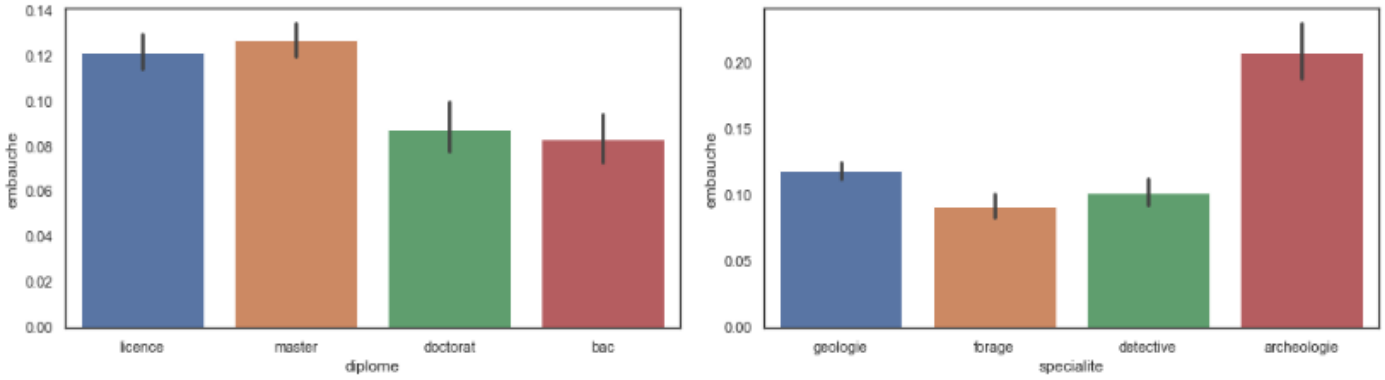




Variable note : p-value = 0.8230 > 0.05, on accepte l'hypothèse de normalité
Variable salaire : p-value = 0.9496 > 0.05, on accepte l'hypothèse de normalité
Variable âge : p-value = 0.0005 << 0.05, on rejette l'hypothèse de normalité
Variable exp : p-value = 0.2642 > 0.05, on accepte l'hypothèse de normalité

Données Qualitatives

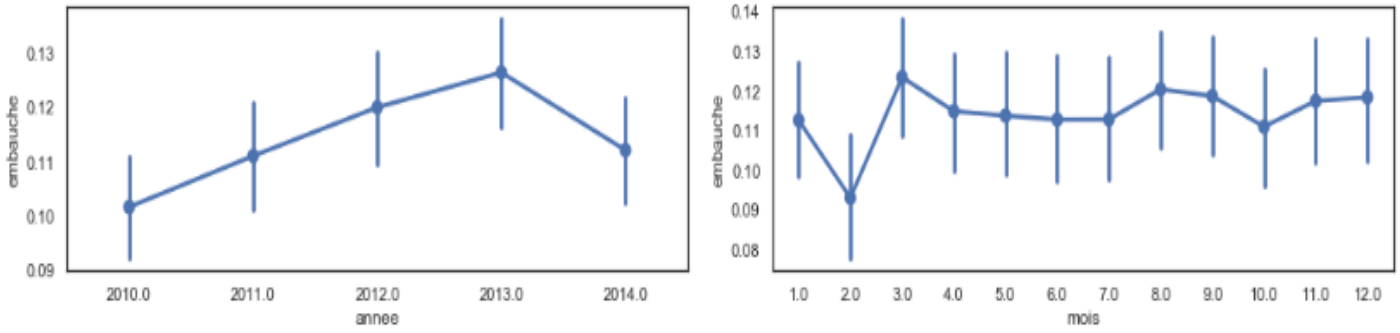
En observant la réparation des données qualitatives, nous remarquons que les données sont déséquilibrées avec notamment une fréquence nettement inférieure pour les classes : "roux" (cheveux), "F" (sexe), "bac" et "doctorat" (diplôme), "oui" (dispo) et une très faible fréquence de la classe "archéologie" en comparaison à la classe "géologie" (spécialité).
On observe aussi un déséquilibre dans la proportion d'embauche de ces classes, particulièrement élevé dans les catégories (spécialité) et (diplôme). Avec une surreprésentation de la classe "archéologie" à plus de 20% contre une moyenne de 10% pour les autres classes. Et une surreprésentation des classes "master" et "licence" à 12% contre 8% en moyenne pour les autres. Ces écarts observés serviront à apporter plus d'informations pour entrainer le modèle, notamment en créant un ranking des classes par catégorie.



Proportion d'embauche en fonction du diplôme et de la spécialité

Données Temporelles

Après un retraitement de la variable (date) pour la séparer en "année", "mois" et "jour", nous observons comme pour les données qualitatives un déséquilibre dans la proportion d'embauche. On observe par exemple que la proportion d'embauche est plus élevée en 2013 que les autres années ou plus faible au mois de février que les autres mois. De même, ces écarts observés serviront à apporter plus d'informations pour entrainer le modèle.



Proportion d'embauche en fonction de l'année et du mois

Traitement des outliers

Dans le jeu de données nous observons des valeurs aberrantes, comme notamment des âges et des expériences qui prennent des valeurs négatives. Cependant comme il s'agit d'un exercice, il ne faut pas chercher de véritable cohérence avec un cas concret. Ici nous avons des données normalement distribuées et les outliers correspondent aux queux d'une loi normale. Les supprimer enlèverait de l'information au modèle. Pour cette raison j'ai considéré qu'il valait mieux les conserver. Néanmoins, si l'on voulait traiter un cas réel, on pourrait appliquer la méthode de Tukey qui supprime les valeurs situées en dehors de 1,5 fois l'intervalle interquartile. C'est ce que fait la fonction "detect_outliers" (cf. notebook).

4. Analyse bivariée

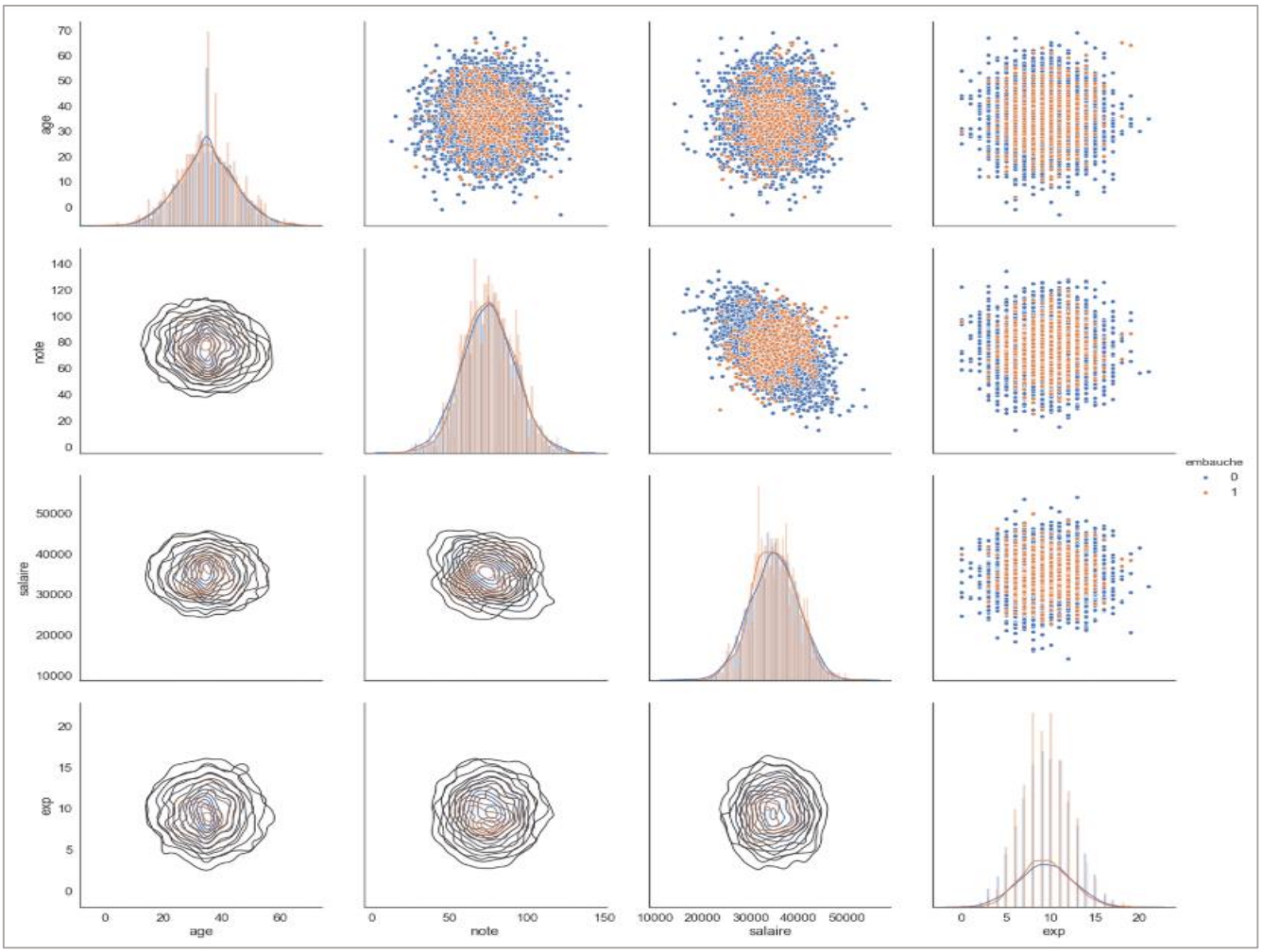
Pour traiter de la dépendance des variables deux à deux, nous avons pu vérifier l'hypothèse que les données quantitatives (à l'exception de l'âge) suivaient une distribution Gaussienne. Pour pouvoir appliquer des tests par intervalle de confiance il faudrait en toute rigueur vérifier aussi l'homoscédasticité et l'indépendance des observations que nous ne traiterons pas ici.

Quantitative vs Quantitatives : Test de corrélation de Pearson

Pour la dépendance de variables quantitative nous utilisons un test de corrélation de Pearson.
H0 : La corrélation entre les deux variables vaut zéro. On rejettera l'hypothèse si la p-value est inférieure au risque alpha de 5%.

Observations des variables deux à deux :

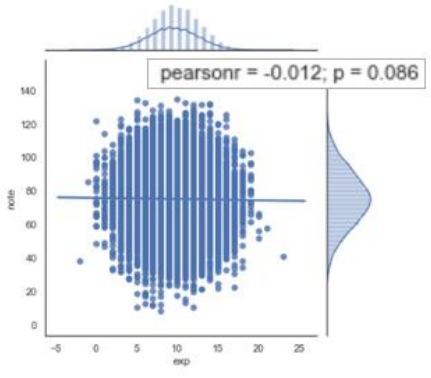
- Le premier constat que l'on fait graphiquement en voyant les Pairplot est que la note et le salaire décrivent un ovale. Nous voudrons donc tester le degré de corrélation de ces variables.
- Ensuite on constate que la distribution des données est la même pour les observations positives (jaune) et pour les observations négatives (bleu). Il n'est donc pas possible d'établir un critère de distinction par simple observation des variables unes à unes.
- Enfin on remarque que les observations positives et négatives sont concentriques et les positives ont un écart-type plus resserrées que les négatives. Pour cette raison et afin de mieux séparer les observations, on va vouloir projeter les donner dans un nouvel espace qui maximise la variance à l'aide d'une PCA.



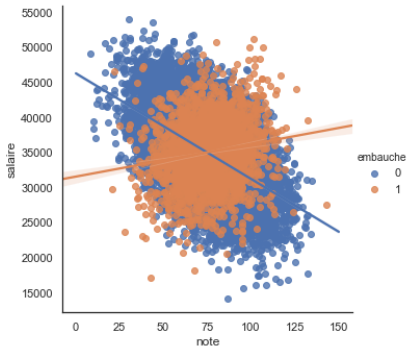
Grid Plot des variables note, salaire, exp et age. Bleu : labels négatifs. Jaune : labels positifs

Expérience vs Note

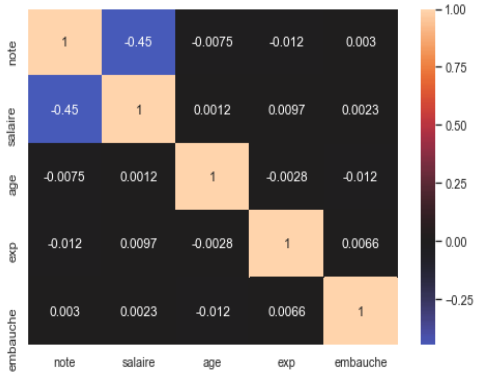
Ici nous souhaitons vérifier s'il existe une corrélation entre la variable (exp) et la variable (note). Le coefficient de corrélation est proche de zéro avec une p-value > 5%. On ne peut donc pas rejeter l'hypothèse à 95% que le coefficient de corrélation soit nul. Les deux variables n'ont pas de dépendance significative.



Pairplot Exp vs Note



Pairplot Salaire vs Note



Matrice des corrélations

Dépendance des variables deux à deux

Grâce à la matrice des corrélations nous pouvons constater, comme cela a été mentionné précédemment, une corrélation entre les variables (note) et (salaire). Il y a effectivement une corrélation négative modérée que l'on ne peut pas rejeter (coef. corr = -0.45). Mais cette corrélation n'est pas suffisamment significative pour pouvoir être exploité.
Nous pouvons aussi constater que la corrélation n'est pas la même en fonction de la sous-population. Tandis que la demande de salaire à tendance à baisser plus la note est élevée pour les labels négatifs, l'effet inverse est observé pour les labels positifs.

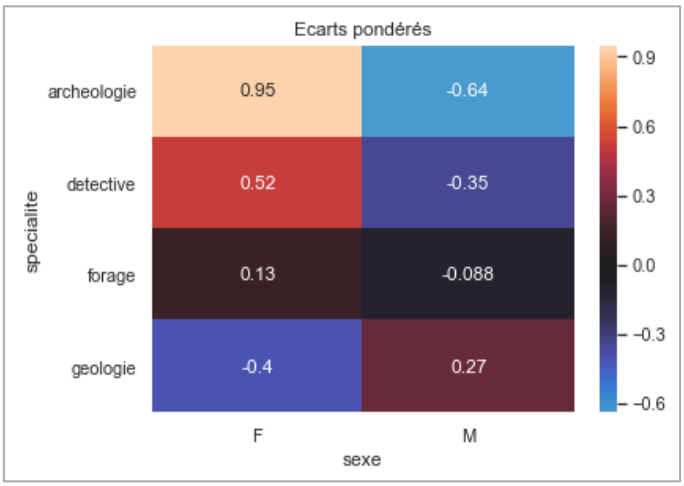
Qualitative vs Qualitative : Table de contingence et test du Chi2

Pour mesurer la dépendance de deux variables qualitatives, on procède généralement avec un test du Chi2 effectué sur la table de contingence des deux variables à tester. Pour effectuer ce test, la fonction "cat_dependance" (cf. notebook) qui fait appelle au module chi2_contengency de Scipy, renvoie la valeur du Chi2, la p-value, les degrés de liberté, la matrice de contingence et la matrice des écarts pondérés.
H0 : Les variables sont indépendantes. On rejettera cette hypothèse avec un risque alpha à 5% si la p-value < 0.05

Spécialité vs Sexe

La p-value du test du Chi2 est proche de zéro, par conséquent on rejette l'hypothèse nulle d'indépendance. Comme nous pouvons le voir dans la table d'écarts pondérés, il y a une surreprésentation des femmes dans les spécialités archéologie (+0.96) et détective (+0.52) et une sous-représentation en géologie (-0.4) et inversement pour les hommes.

```
Table de contingence :
-----------------------------------
sexe              F      M    All
specialite
archeologie    1081    298   1379
detective      2522   1614   4136
forage         1964   2351   4315
geologie       2403   7576   9979
All            7970  11839  19809
-----------------------------------
```



Dépendance des variables deux à deux

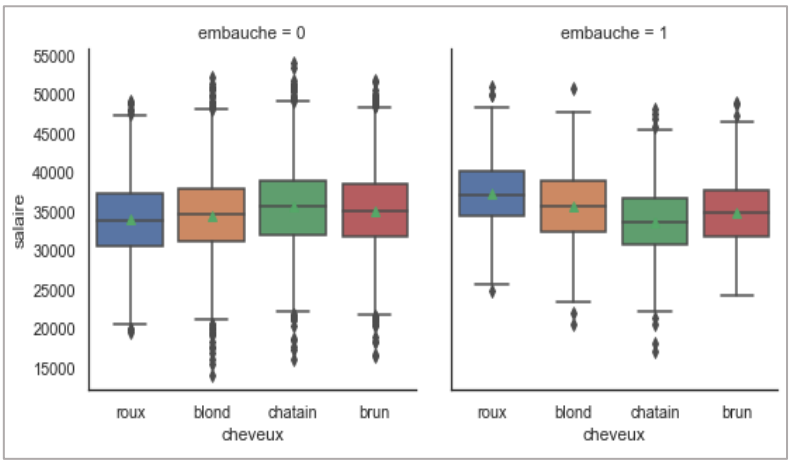
Nous pouvons effectuer la même procédure pour l'ensemble des variables catégorielles deux à deux. Il faudra néanmoins émettre une réserve sur la validité du test concernant la variable (âge) qui ne provient pas d'une loi normale. On pourra se référer à la table de résumer sur les dépendances pour une meilleure lecture.

Quantitative vs Qualitative : ANOVA et test de Fisher

Pour mesurer la dépendance de deux variables de types différents, catégoriel et qualitatif, il est d'usage de faire une ANOVA suivi d'un test de Fisher. Pour effectuer ce test, la fonction "quali_quanti" qui utilise le module anoval_lm de Scipy, renvoie une table contenant la valeur du test de Fisher ainsi que la p-value.
H0 : Les moyennes sont les mêmes et donc les variables sont indépendantes. On rejettera cette hypothèse avec un risque alpha à 5% si la p-value < 0.05

Cheveux vs Salaire

La p-value du test de Fisher est proche de zéro. On rejette l'hypothèse nulle que les moyennes sont les mêmes. Les variables sont donc dépendantes. Nous pouvons en conclure qu'il y a un effet significatif des cheveux sur le salaire demandé. On peut constater visuellement cette dépendance sur les boites de Tuckey.



Résumé ANOVA :

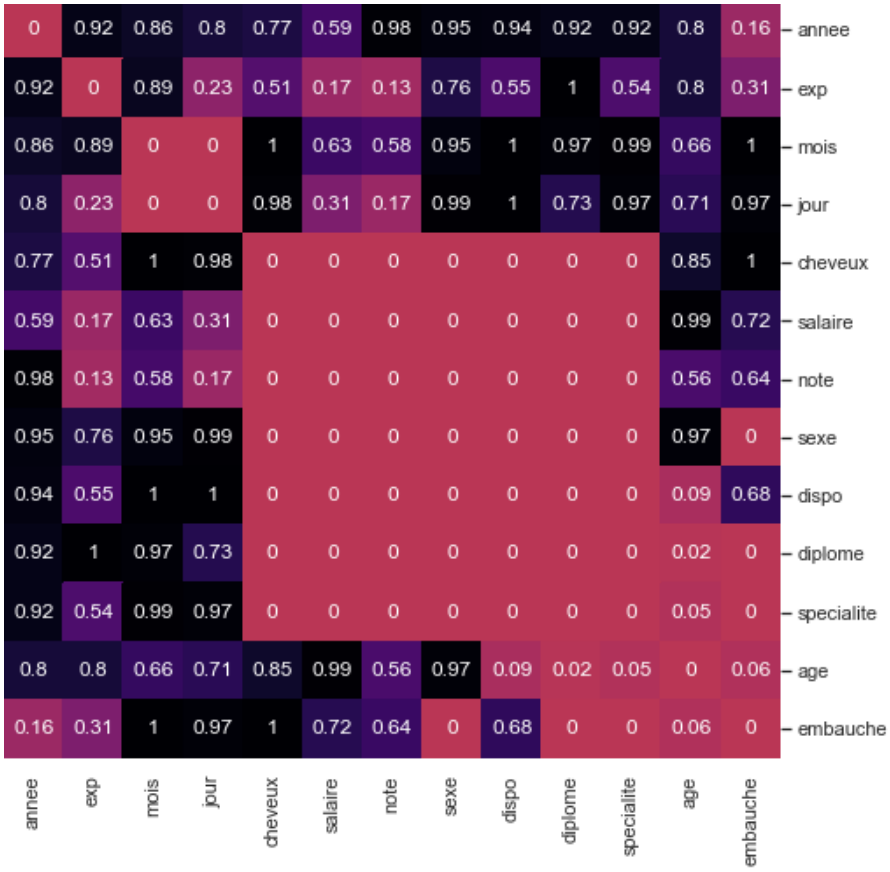
	sum_sq	df	F	PR(>F)
C(cheveux)	2.065100e+09	3.0	27.592636	8.494522e-18
Residual	4.744261e+11	19017.0	NaN	NaN

Dépendances des variables deux à deux :

Il existe probablement des dépendances entre d'autres variables. Il est intéressant de les tester. Il faudra néanmoins émettre une réserve sur la validité du test concernant la variable (âge) qui ne provient pas d'une loi normale. On pourra se référer à la table de résumé sur les dépendances pour une meilleur lecture.

Résumé sur les dépendances bivariées :

Cette matrice résume les dépendances entre les variables deux à deux en fonction des p-values aux tests d'hypothèses. Les p-values proches de zéro (variables dépendantes) apparaissent en rouge.



Résumé des dépendances bivariées

5. Traitement des valeurs manquantes

Bien souvent il est nécessaire de traiter les valeurs manquantes (NaN) pour deux raisons principales :

1- Cela peut aider à améliorer le modèle en apportant des informations manquantes.
2- Souvent nous n'avons pas le choix pour pouvoir faire une prédiction sur une observation.

Dans notre cas nous considérons que l'on veut pouvoir prédire toutes les observations d'un jeu de données test sans en supprimer aucune (bien que cela détériore un peu les prédictions dans ce cas précis). C'est une décision arbitraire. Nous observons 1181 valeurs manquantes (ou 999 si on ne considère qu'une seule fois la date) soit près de 5% du jeu de données. Afin de combler ces vides il existe plusieurs méthodes. La plus simple consiste à prendre la médiane de l'ensemble des observations, ou la classe avec la plus grande fréquence dans le cas des données qualitatives, pour remplacer les valeurs manquantes. Néanmoins cette méthode est peu précise. Il est possible de l'améliorer en utilisant les relations de dépendance entre les variables trouvées précédemment. C'est ce qui est utilisé ici. Par exemple, comme nous l'avons vu sur les boites de Tuckey, la médiane des notes des doctorants est aux alentours de 90 tandis que celles des Bacs est plutôt autour de 60, il serait donc plus judicieux de remplacer un NaN d'un doctorant par la note 90 plutôt que par la médiane globale qui est 75. Les fonctions "replace_nan_quanti" et "replace_nan_quali" (cf. notebook) vont chercher toutes les observations similaires à celles ayant une valeur manquante et les remplacer par la médiane commune (ou plus grande fréquence).

6. Machine Learning

Feature Engineering

Afin de faciliter l'entrainement du modèle, il est préférable de lui apporter le plus d'information possible. Pour ce faire nous procédons à plusieurs transformations successives :

1. Ordonnancer (ranking) les catégories nominales en fonction de leur moyenne d'embauche.
2. Les concaténer avec les catégories nominales OneHotEncodés (encodage binaire).
3. Centrer-Réduire les données (standardisation) pour qu'elles soient comparables sur une même échelle de grandeurs.
4. Projection des données centrées-réduites en composantes principales pour maximiser la variance (PCA).
5. Réduction de dimension en fonction de la variance expliquée par les vecteurs propres.

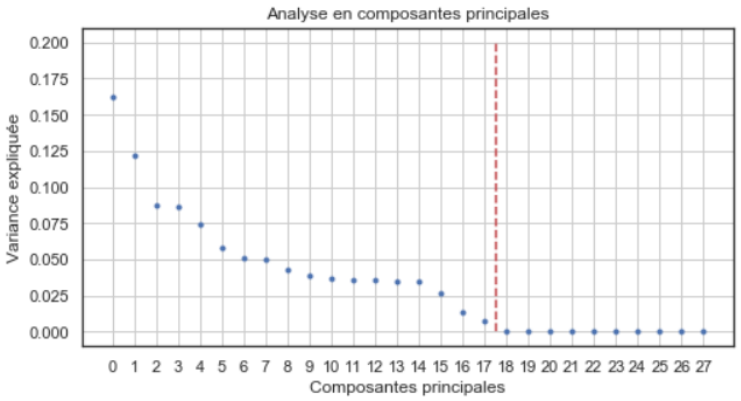
Afin de pouvoir tester notre modèle nous le séparons en deux parties : train 70% / test 30%.
Il faut prendre soin de conserver le taux de classes positives dans le train et dans le test, pour cela nous utilisons la méthode "stratify". Par ailleurs, il est nécessaire de séparer le jeu de données avant de centrer-réduire les variables et de faire la PCA, afin que le modèle ne puisse pas apprendre les caractéristiques intrinsèques des variables comme la moyenne et la variance, ce qui aurait pour effet de biaiser positivement les résultats obtenus.

➔ Dimension du nouveau dataframe après encodage : (20000, 29)
➔ Dimension du train et du test : X_train: (14000, 28), y_train: (14000,), X_test: (6000, 28), y_test: (6000,)

PCA

Comme nous pouvons le voir, les 9 derniers vecteurs propres ont des valeurs propres proches de zéro. Ces composantes n'expliquent donc pas la variance du modèle et n'apportent pas d'information utiles et peuvent même réduire l'interprétation. Nous allons les supprimées et ne conserver que les 18 premières composantes :

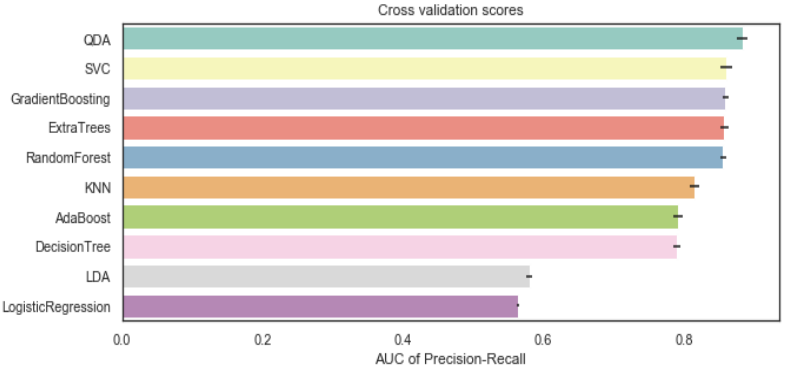
➔ Dimension après PCA : X_train_pca : (14000, 18)



Sélection de modèles et optimisation

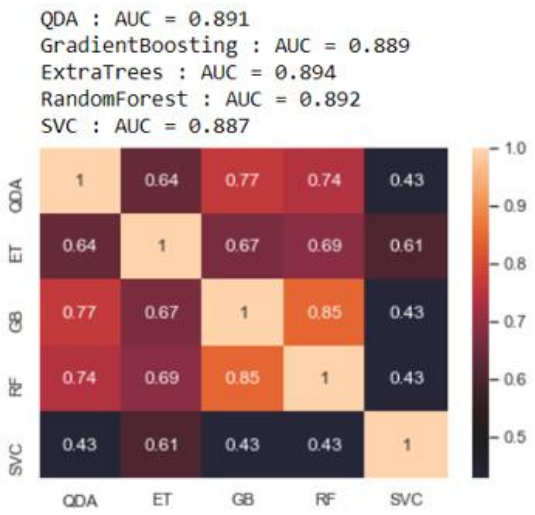
Pour sélectionner le modèle qui sera le plus efficace pour prédire les données du test, nous en testons plusieurs et analysons les scores obtenus. Le score choisi ici est l'aire sous la courbe de Precision-Recall (AUC). J'ai volontairement choisi des classifieurs classiques de la librairie Sklearn pour éviter l'installation de packages lourds comme Keras ou XGBoost et pouvant être longs à entrainer. 10 classifieurs sont testés ici, sans optimisation des paramètres hormis le poids des classes pour les équilibrer. Nous pourrions ne conserver que le meilleur classifeur et l'optimiser. Cependant il est souvent plus efficace d'en sélectionner plusieurs et d'assembler leurs prédictions par un vote à la majorité (ou moyenné) afin d'améliorer la précision, c'est la méthode d'« ensembling ».

A l'issue de ce test nous ne conservons que les 5 meilleurs modèles : Quadratic Discriminant Analysis (QDA), Extra Trees (ET), Random Forest (RF), Support Vector Machine (SVC), Gradient Boosting (GB). Afin d'améliorer l'efficacité des modèles retenues, nous optimisons leurs hyperparamètres en en testant plusieurs et nous conserverons les meilleurs. Nous faisons cela à l'aide d'un GridSearch Cross-Validation avec un K-fold de 3.



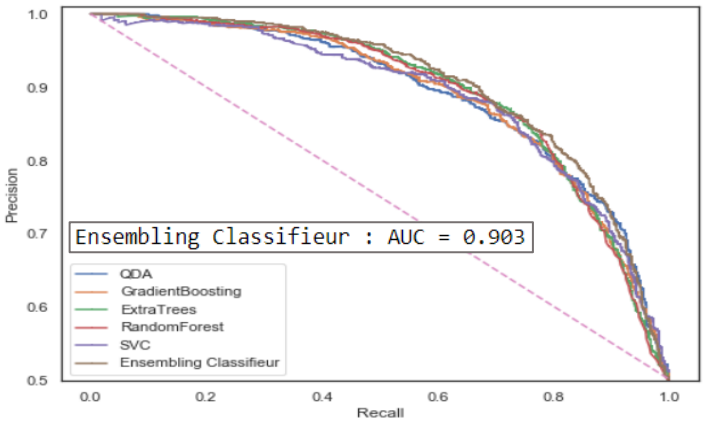
Prédictions

Nous pouvons visualiser ici les résultats des modèles en mesurant l'aire sous la courbe Precision-Recall. Le meilleur score est réalisé par l'Extra Tree avec une aire sous la courbe de 89,4%.
Pour que l'assemblage des modèles soit efficace, il faut que les prédictions soient les plus précisent possible mais elles doivent être aussi suffisamment différentes. Pour cela nous pouvons observer la matrice des corrélations des prédictions des modèles. On constate que les prédictions sont fortement corrélées (sauf pour le SVC), mais avec suffisamment de variance pour ne pas être identiques ce qui permettra à l'assemblage d'améliorer les résultats.



Ensembling

L'ensembling consiste à récupérer les prédictions de différents classifieurs et à les assembler. Ici nous faisons cela en moyennant les prédictions « soft » (probabilistes) des précédents modèles. Par ailleurs j'ai dupliqué le classifieur QDA en modifiant les poids afin de créer plus de variance dans les prédictions. Et j'ai accordé plus d'importance à l'Extra Tree lors de l'assemblage car il avait obtenu le meilleur score. Comme nous pouvons le constater, l'ensembling a un effet positif sur les prédictions en améliorant le score d'aire sous la courbe : 90,3% .
Nous obtenons en sorti un modèle que j'appelle « pessimiste » car il ne favorise pas spécialement les bons candidats (labels positifs). Ce modèle est néanmoins très efficace dans un cas d'usage particulier qui serait de filtrer les mauvais candidats (labels négatifs). En effet, il classe bien à 98% les vrais négatifs. Ce qui signifie que le risque d'embaucher un mauvais candidat à tort n'est que de 2%, ce qui limite le risque de l'entreprise. Le trade-off cependant est d'éliminer la moitié des bons candidats. Ce modèle serait bien adapté dans le cas d'une entreprise qui aurait beaucoup de candidatures.
Nous verrons deux autres modèles par la suite.



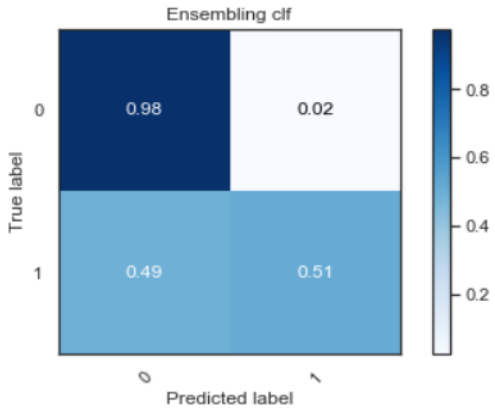


Fig. : Matrice de confusion

D'autres modèles pour d'autres usages

Voici 2 autres modèles provenant d'un même classifieur QDA mais entrainé différemment pour des usages différents :

- Equilibré : Il va classer les bons candidats à 80% mais ne saura pas en détecter 20% et il classera les mauvais candidats à 80% aussi mais avec le risque d'en embaucher 20% à tort ce qui peut coûter très cher à une entreprise. D'autant plus que la quantité de candidats non embauchés est environ neuf fois supérieur aux candidats embauchés, les 20% n'ont donc pas le même ordre de grandeur.
- Optimiste : Il va pouvoir capter 95% des bons profils mais il faudra faire le tri par la suite car il y aura beaucoup de retours négatifs. Ce modèle serait bien adapté à une entreprise qui souhaite embaucher mais qui aurait du mal à trouver de bons profils.