Ces notes sont un compte-rendu de la phase de conception et de tests préliminaires de l'algorithme de détection de billets contrefaits.

[SLIDE 2]

On va dans un premier temps s'attacher à décrire notre jeu de données de départ, depuis son inspection initiale jusqu'à quelques statistiques descriptives de base en passant par la méthodologie utilisée pour compléter certaines données manquantes. On va ensuite détailler le premier algorithme de détection sur lequel nous avons travaillé, le K-Means, en présentant la façon dont les paramètres ont été choisis et optimisés ainsi que le clustering auquel ils ont abouti, avant de représenter les clusters produits au travers du filtre de l'analyse en composantes principales et de tester l'algorithme final et d'analyser ses performances. Enfin, on appliquera la même démarche, à savoir choix des variables, optimisation des paramètres et évaluation de la performance de l'algorithme final sur notre 2ème type de détecteur, celui utilisant la méthodologie de la régression logistique, avant de conclure sur les performances respectives et le choix entre nos deux modèles, que nous pourrons également tester en direct.

[SLIDE 3]

Tout d'abord, donc, la vérification de nos données de départ. Elles consistaient en un fichier de 1500 lignes et 7 colonnes de données apparemment correctement typées – 1 donnée booléenne indiquant la nature du billet comme vraie ou fausse et 6 colonnes de variables quantitatives continues représentant les dimensions géométriques de nos billets. Le fichier ne présente aucune valeur négative ni aucun doublon ; on note cependant quelques valeurs *a priori* atypiques mais pas aberrantes qu'il conviendra d'analyser plus en détail. La colonne de données sur la dimension margin_low fait apparaître 37 valeurs manquantes, pour 29 vrais billets et 8 faux billets, qu'il conviendra donc de compléter. Aucune clé primaire n'est présente dans la table, on pourra au besoin utiliser l'index comme clé primaire artificielle pour réaliser des éventuelles jointures au cours de notre analyse.

[SLIDE 4]

Afin de compléter les données manquantes dans notre colonne margin_low, nous avons procédé à une régression linéaire multiple. Ce n'est pas la seule méthode disponible ici, on aurait pu choisir de faire une imputation par la moyenne des billets sur leur catégories respectives vrais/faux par exemple. Les résultats de la régression linéaire multiple sont significatifs, avec une probabilité de la F-statistic très proche de zéro qui nous indique que notre modèle est meilleur qu'un modèle où tous les coefficients appliqués aux différentes variables seraient nuls. Toutes nos variables sont significatives au seuil α = 5%, mais notre modèle a une qualité de prévision médiocre avec un R2 inférieur à 0.5 et le Condition Number très élevé indique un problème de colinéarité potentiel de nos variables, que nous avons en fait pu écarter en calculant les Variance Inflation Factors de chaque variable, qui sont tous inférieurs à 10.

[SLIDE 5]

L'analyse des résidus de notre régression linéaire multiple a révélé que leur moyenne était très proche de zéro, ce qui est un bon résultat, mais les différents tests réalisés n'ont pas permis de confirmer leur homoscédasticité, ni leur distribution selon une loi normale, qui étaient deux hypothèses à vérifier pour valider complètement les résultats de notre régression linéaire multiple. Ce résultat s'observe notamment sur le QQ-plot à droite de cette diapositive, où on voit la queue de distribution à droite s'écarter nettement de la distribution théorique d'une loi normale, et on constate également que les résidus semblent plus dispersés sur les faux billets que sur les vrais, indiquant des dimensions moins régulières.

[SLIDE 6]

L'analyse des résidus studentisés a fait apparaître 7 individus atypiques et influents, mais leur élimination de la régression linéaire ne fait que peu varier les résultats, et nous avons donc décidé de les conserver dans notre analyse. Nous avons également procédé à des régressions linéaires multiples utilisant les modèles Ridge et Lasso, mais celles-ci n'ont pas permis de corriger les problèmes sur les résidus ni d'améliorer significativement le R2.Malgré le fait que nos résidus ne vérifient pas certaines hypothèses statistiques, nous avons néanmoins décidé de procéder au reste de l'analyse en complétant notre jeu de données avec les

valeurs prédites par la régression linéaire multiple pour la variable margin_low. En effet, malgré les problèmes mentionnés, nous avons une moyenne des résidus très proche de zéro, un échantillon suffisamment grand avec 1463 observations, une courbe relativement “en cloche » comme on peut le constater sur le graphique de droite, et tous nos paramètres ainsi que notre modèle lui-même sont significatifs au seuil α = 5% comme on l’a dit, donc nous avons des résultats qui démontrent une robustesse suffisante pour pouvoir continuer notre analyse en les incluant et ce malgré les hypothèses non vérifiées sur les résidus.

[SLIDE 7]

Quelques statistiques à présent pour décrire brièvement notre jeu de données. Comme on peut le voir sur les différents boxplots à gauche de cette diapositive, les faux billets présentent des moyennes supérieures sur les dimensions des marges et des hauteurs, ainsi qu’une longueur moindre que les vrais billets en moyenne. Les vrais et faux billets révèlent une différence beaucoup moins marquée en revanche sur la diagonale. On note par ailleurs la présence de quelques outliers, considérés comme on l’a dit atypiques plutôt qu’aberrants en raison de leur proximité relative avec les moustaches de nos boxplots. La heatmap des corrélations fait quant à elle apparaître une corrélation relativement élevée entre la longueur et la marge du bas sur l’ensemble des billets.

[SLIDE 8]

Le premier algorithme de prédiction que nous avons utilisé pour notre modélisation est celui du K-Means. C’est ce que l’on appelle un algorithme de classification non-supervisé où le modèle va essayer, par itérations successives, de trouver des similarités entre différents sous-groupes de nos billets. Ce n’est donc pas un algorithme de régression mais bien un algorithme de classification qui opère sur des données non-étiquetées, d’où le terme « non-supervisé ». Une des tâches à effectuer en aval de la modélisation sera donc de rattacher les groupes de données formés, les clusters, à une étiquette vraie ou fausse pour qualifier nos billets, et utiliser cet algorithme de classification à des fins de prévision. Avant d’en arriver là, notre première tâche pour optimiser le modèle a consisté à choisir le nombre optimal de sous-groupes à former ; on pourrait se dire intuitivement que comme la variable qu’on cherche à prédire ici est

booléenne, 2 clusters constituent la meilleure option, mais l'intuition n'est pas toujours avérée en matière de statistiques. Afin d'optimiser ce premier paramètre, nous avons donc utilisé plusieurs méthodes pour déterminer le nombre optimal de clusters. La méthode du coude ici à gauche ne s'avère pas très parlante, il n'y a pas de « coude » à proprement parler dans notre courbe qui a une pente plutôt régulière, qui semble toutefois diminuer un peu après 3 clusters mais sans former de vrai « coude ». Les scores de la silhouette et de l'indice de Davies-Bouldin semblent indiquer 3 et 4 clusters comme des possibilités intéressantes, donc nous les avons mises en concurrence avec notre intuition de départ qui était d'appliquer 2 clusters. Nous avons aussi cherché à optimiser les autres paramètres du modèle pour rendre les résultats reproductibles et la convergence aussi rapide que possible, ce qui nous a conduit à initialiser le modèle avec la méthode du K-Means ++, l'utilisation de l'algorithme Elkan et un random state initialisé avec une valeur entière que l'on gardera constante – ici on a choisi 42, la valeur elle-même est sans importance, seule sa constance entre deux calculs du modèle importe. Pour assurer une meilleure comparaison des résultats avec ceux de la régression logistique que nous aborderons plus loin, nous avons également séparé notre jeu de données en un training set et un testing set selon une proportion de 80/20. Les résultats de nos modélisations sur 2, 3 et 4 clusters sont présents dans le tableau de droite, et on constate que du point de vue de tous les indicateurs de performance utilisés, que ce soit le Rand score, sa version ajustée, le Mutual Information Based score et sa version ajustée ou le score de Fowlkes-Mallowes, le modèle à 3 clusters présente les meilleurs résultats.

[SLIDE 9]

Si on analyse à présent les centroïdes de nos différents clusters dans nos 3 modèles et les matrices de confusion qui en résultent sur notre jeu de données d'entraînement, on s'aperçoit qu'un de nos clusters reste relativement inchangé quant à son centroïde alors que le second semble se « décomposer » en plusieurs clusters imbriqués à mesure que le nombre de clusters augmente. On note, comme mentionné dans l'analyse des boxplots précédemment, que l'un de nos clusters a des dimensions moyennes plus fortes sur les marges et les hauteurs (le cluster 2, 2 et 3 dans nos 3 modélisations respectivement), et que ce cluster doit donc correspondre à celui de nos faux billets. Si on analyse les matrices de

confusion qui en résultent, on s'aperçoit qu'ici encore, le modèle à 3 clusters s'impose : il offre le meilleur compromis de performance entre le nombre de faux négatifs (c'est-à-dire le nombre de vrais billets détectés comme faux) et le nombre de faux positifs (à savoir le nombre de faux billets détectés comme vrais). On cherche bien évidemment à minimiser le taux de faux positifs, mais ici encore on retiendra le modèle à 3 clusters car il permet une amélioration de 50% du taux de faux négatifs pour une très légère dégradation de 7.7% du taux de faux positifs, ce qui nous a semblé un compromis acceptable.

[SLIDE 10]

Afin de faciliter la représentation graphique de nos données, nous avons également procédé à une Analyse en Composantes Principales de notre jeu d'entraînement. L'éboulis des valeurs propres, ici sur la gauche, nous indique que 60% de la variance de notre jeu de données sont capturés par nos 2 premiers axes. Si l'on observe la heatmap de notre ACP, on s'aperçoit que notre premier axe est dominé par les marges et la hauteur droite, auxquelles il est positivement corrélé, et la longueur, dont la contribution est négative. Le deuxième axe est quant à lui dominé fortement par la diagonale, la dernière variable, la hauteur gauche, apportant une contribution équivalente aux deux premiers axes. Ces contributions sont représentées graphiquement sur le cercle des corrélations. Si on se souvient des caractéristiques de nos faux billets présentées sur les boxplots du départ, on devrait donc logiquement les retrouver dans la partie droite du premier plan factoriel, et c'est exactement ce que l'on observe si on projette nos 1500 billets de départ sur ce premier plan.

[SLIDE 11]

Si l'on projette à présent nos différents clusters sur ce premier plan factoriel, on voit bien notre cluster de droite représentant les faux billets, avec un cluster des vrais billets à gauche qui se « subdivise » en quelque sorte en sous-clusters à mesure que le nombre de clusters augmente. Nous avons donc utilisé cette propriété pour allouer nos étiquettes vrai/faux à nos clusters et procéder à une évaluation de la performance sur le jeu de test.

[SLIDE 12]

Nous utilisons donc les mêmes scores pour évaluer la performance sur notre jeu test que nous l'avions initialement fait sur le jeu d'entraînement, et nous aboutissons à une performance même légèrement meilleure que sur le jeu d'entraînement, puisque nous n'obtenons qu'un seul faux positif là où nous en attendions presque 4 d'après les résultats sur le jeu d'entrainement. L'application de ce modèle à l'échantillon de production non étiqueté retourne les 3 premier billets comme faux et les 2 derniers comme vrais.

[SLIDE 13]

Intéressons-nous à présent à notre second modèle de détection, celui basé sur la régression logistique. La différence essentielle ici avec le K-Means est qu'il s'agit d'un algorithme de régression, dont le résultat va être le calcul d'une variable aléatoire continue qui est une probabilité, et qu'il ne s'agit dont pas d'un algorithme de classification. De surcroît, cette régression est effectuée sur des données étiquetées, c'est donc un algorithme dit « supervisé ». Notre première étape a consisté à vérifier que toutes les données de nos 6 dimensions étaient pertinentes pour l'analyse, nous avons donc fait tourner un premier « fit » du modèle et bien que nous obtenions un R2 satisfaisant à près de 0.7, on s'aperçoit que 2 de nos variables, diagonal et height_left, ne sont pas significatives au seuil $\alpha = 5\%$. Nous avons donc procédé à leur élimination successive pour aboutir au modèle final avec seulement 4 données indépendantes en entrée, qui coïncidemment se trouvent être les 4 variables les plus fortement pondérées en valeur absolue dans le premier axe factoriel de notre ACP.

[SLIDE 14]

Si nous faisons donc tourner ce modèle sans autre optimisation sur notre jeu d'entraînement, nous obtenons zéro faux positifs et 3 faux négatifs, ce qui nous donne un excellent recall à 100% mais un score de précision à 98.45% : bien que ce soit déjà très bon, nous avons essayé d'améliorer ce score en procédant à l'optimisation des hyperparamètres du modèle. Nous avons pour cela utilisé la fonctionnalité gridsearchCV de Scikit-Learn, qui nous a permis de tester 5 modèles de régression logistique différents avec une variété de paramètres de pénalité sur les coefficients et un nombre d'itérations suffisant pour assurer la convergence du modèle. Ici encore, on a initialisé le random state a 42 pour

assurer la reproductibilité des résultats d'une implémentation du modèle à l'autre. Ces différentes options ont mis en valeur 2 modèles dont les performances étaient supérieures aux autres, le modèle newton-cg et le modèle saga. Ces deux modèles étant vraiment équivalents en termes de performance, nous avons choisi de procéder à la suite de notre analyse avec le modèle saga qui, outre d'être celui préconisé dans un tel cas par la documentation de Scikit-Learn, devrait nous permettre d'obtenir une convergence plus rapide, et donc une meilleure efficacité computationnelle, sur nos données centrées et réduites.

[SLIDE 15]

Nous avons donc appliqué ce modèle de régression logistique saga avec ses hyperparamètres optimaux à notre jeu de données de test, et en fixant le seuil de probabilité à 50%. Nous obtenons ici encore un recall score parfait à 100%, ce qui nous indique que si un billet est vrai, il sera détecté comme tel dans 100% des cas, et que donc réciproquement, si un billet est détecté comme faux, il l'est dans 100% des cas. Le score de précision quant à lui nous indique que 97.94% des faux billets seront détectés comme tels par le modèle, ce qui réciproquement nous indique que si un billet set détecté comme vrai, on a en fait 2.06% de chances de se tromper. L'application de ce modèle à l'échantillon de production non étiqueté retourne ici encore les 3 premier billets comme faux et les 2 derniers comme vrais.

[SLIDE 16]

Pour conclure cette analyse, on voit bien que nos deux algorithmes retournent des résultats identiques sur un petit échantillon de billets aux caractéristiques suffisamment différentes. En pratique, il faut rappeler que par nature, le K-Means est un algorithme de classification non-supervisé qui fonctionne en détectant des similarités entre les données, et que par nature donc il aura du mal à différencier des données si elles sont trop similaires – en plus de la tâche relativement complexe de recherches des similarités entre 6 variables. Différentes méthodes pourraient être appliquées pour approfondir cette analyse et raffiner le modèle du K-Means, comme par exemple traiter le numéro de cluster comme une variable catégorielle et appliquer la méthode du one-hot encoding pour utiliser cette variable dans la régression logistique par exemple. On pourrait également

envisager d'ajouter des données à notre analyse, comme par exemple la valeur nominale du billet, ou des caractéristiques graphiques de pigmentation ou de translucidité du papier pour tenter de maximiser le score de précision. Il n'est pas vraiment pertinent de dire ici qu'un modèle est meilleur que l'autre ou même que nos deux modèles sont équivalents, leurs résultats peuvent être amenés à différer en raison du fait que ces 2 modèles ont été créés pour répondre à des questions différentes et opèrent avec des paramètres et des données de départ différents, et ont des possibilités d'optimisation différentes, donc il est en fait assez remarquable ici qu'ils retournent le même résultat – c'est clairement une illustration des hasards de l'échantillonnage qui nous a fourni des données de production suffisamment différenciées sur ces 5 billets pour être clairement identifiables du point de vue du clustering. Sans dire qu'il s'agit d'un meilleur modèle, donc, on peut toutefois souligner que dans la pratique il est plus courant d'avoir recours à des algorithmes d'apprentissage supervisés comme la régression logistique qu'à des algorithmes non-supervisés comme le clustering pour répondre à ce genre de questions.