# WEM - Laboratoire 2

## 1. Classification des « pièges-à-clics »

### 1. Est-ce que le changement du paramètre split ratio influence la performance du modèle ?

Pour pouvoir déterminer ceci, nous avons exécuté le process plusieurs fois avec plusieurs split ratio différents :

| **Split ratio** | **Accuracy** | **Précision** | **Recall** |
|---|---|---|---|
| 0.5 | 88.90 | 89.17 | 88.60 |
| 0.7 | 90.23 | 91.42 | 88.83 |
| 0.8 | 90.61 | 91.33 | 89.77 |
| 0.9 | 91.08 | 91.57 | 90.53 |

Nous pouvons en déduire que oui, le changement du paramètre split ratio influence la performance du modèle. La meilleure valeur semble être autour de 0.9.

### 2. Veuillez changer le bloc « Split Validation » avec le bloc « Cross Validation » en utilisant le même classificateur pour l'apprentissage et les mêmes opérateurs dans la partie d'évaluation. Qu'est-ce que vous pouvez constater des résultats correspondants ?

Contrairement au split validation, les résultats ont un pourcentage qui accompagne le résultat moyen de l'accuracy. Ceci est dû au fait que la cross validation effectue plusieurs folds sur les données et que les accuracy entre les différents folds varient de +/-1,2% dans notre cas.

**accuracy: 89.60% +/- 1.20% (micro average: 89.60%)**

### 3. Dans le bloc « Process Documents from Data » nous n'avons pas mis d'étape de stemming. Est-ce que l'ajout de ce prétraitement a un impact sur les résultats obtenus?

Oui, cela semble réduire légèrement l'accuracy, mais également le pourcentage qui l'accompagne.

**accuracy: 87.47% +/- 0.76% (micro average: 87.47%)**

### 4. Jusqu'à présent nous avons utilisé un classificateur bayésien. Veuillez essayer d'autres familles de classificateurs, quel est l'impact sur le résultat obtenu ?

Nous avons essayé plusieurs types de classificateurs :

| **Classificateur** | **Accuracy** |
|---|---|
| K-NN 5 | 88.46 |
| Decision Tree | 61.73 |

# 2. Analyse des sentiments sur des commentaires

### 1. Vous commenterez ensuite les résultats obtenus par rapport aux étiquettes existantes sur le dataset. Quelle est l'influence des différentes étapes de « textprocessing » sur le résultat que vous obtenez ?

Nous avons un process qui applique les étapes suivantes : Tokenize, Transform Cases, Filter Stopwords, Filter tokens (by length). Nous avons analysé l'impact de chaque étape sur le résultat :

- La tokenization, qui consiste à segmenter le texte en unités discrètes, peut influencer l'identification des nuances de sentiment dans les tweets.
- La transformation des cas, qui consiste à convertir toutes les lettres en majuscules ou en minuscules, peut normaliser le texte, mais peut aussi entraîner la perte de certaines informations importantes.
- Le filtrage des stopwords, qui supprime les mots très courants sans signification particulière, peut réduire le bruit dans les données, mais peut par ailleurs entraîner la perte de certains mots-clés importants.
- Le filtrage des tokens par longueur, qui supprime les mots trop courts ou trop longs pour être significatifs dans le contexte de l'analyse de sentiment, peut éliminer le bruit dans les données, mais peut également entraîner la perte de certains mots-clés importants.

2. Dans une deuxième étape, un ensemble de données ‘CoronaTwitterComments_3labels.xlsx’ complété avec des commentaires neutres (étiqueté avec 0) va être considéré. Veuillez adapter le process créé précédemment pour 3 étiquettes (classes) et commenter les nouveaux résultats.

Nous avons importé le nouveau set de données et adapté le process. Nous ne voyons cependant pas de différence significative entre les deux résultats.

# 3. Système de recommandation des films

1. Est-ce que le rapport de partition de l’ensemble des données initiales (bloc « Split Data ») influence la performance du modèle ? Veuillez expliquer votre réponse.

Oui, si on réduit la taille du example set que l’on passe pour créer le modèle, les résultats sont moins bons (0.5 de AUC par exemple avec 30% des données utilisées pour l’example set). Ils sont meilleurs si on donne plus de données à l’example set (0.8 de AUC avec 70% des données utilisées pour l’example set). Ceci est dû au fait qu’avec plus de données présentes pour créer le modèle KNN celui-ci est plus précis, car il se base sur ces valeurs fournies pour effectuer sa prédiction.

2. Qu’est-ce que vous pouvez constater si vous changez la valeur de k de l’opérateur « Item k-NN » ou si vous utilisez un k-nn pondéré ?

La valeur idéale de K semble être autour de 180 (voir tableau ci-dessous).

| **Valeur de K** | **AUC** |
|---|---|
| 20 | 0.751 |
| 80 | 0.830 |
| 150 | 0.849 |
| 180 | 0.852 |

3. Veuillez changer le bloc « Item k-NN » avec le bloc « User k-NN ». Qu’est-ce que vous pouvez constater des résultats correspondants ? Que pouvez-vous constater en utilisant les autres approches de filtrage collaborative ?

Les résultats paraissent être meilleurs si on utilise un bloc User k-NN (~0.897 de AUC peu importe la valeur de K). Avec le bloc “Most Popular” nous obtenons un AUC de 0.865. Avec

le bloc “Weighted Regularization Matrix Factorization” nous obtenons un meilleur résultat (0.907).

### 4. Choisissez la méthode qui donne la meilleure performance, et, à l’aide du bloc « Model Combiner », combinez le avec la méthode « User k-NN ». Qu’est-ce que vous pouvez constater des résultats correspondants ? Les résultats changent-ils si « Item k-NN » est ajouté à la combinaison précédente ?

On obtient exactement les mêmes résultats en combinant le bloc “Weighted Regularization Matrix Factorization” avec un “Item k-NN”.

### 5. Qu’est-ce que vous pouvez constater sur les résultats obtenus ?

Les prédictions ne sont pas si correctes en moyenne. Le modèle semble cependant marcher relativement bien, et ne se trompe que de 0.5 sur la note.

### 6. En évaluant la performance du modèle construit, quelles sont vos observations sur les résultats en changeant les paramètres du k-nn (notamment la valeur de k et le mode de corrélation) ?

Le mode de corrélation “cosine” paraît être plus adapté à notre ensemble de données. On obtient au mieux un RMSE de 0.839 avec K = 180 et mode de corrélation = cosine.

### 7. Veuillez changer le bloc « Item k-NN » avec le bloc « User k-NN ». Qu’est-ce que vous pouvez constater des résultats correspondants ? Qu’est-ce que vous pouvez constater en utilisant les autres approches de filtrage collaborative pour la prédiction ?

On obtient avec le bloc “User k-NN” un RMSE moins bon qu’avec le bloc “Item k-NN”. En essayant le reste des blocs disponibles dans cette catégorie, nous n’avons pas trouvé une meilleure alternative à “Item k-NN”. Dans nos tests, tous les autres blocs obtenaient un RMSE plus haut que 0.839.

# 4. Règles d’association sur des achats en ligne

### 1. Constatez-vous des changements pour des différents paramètres des blocs « FP-Growth » et « Create Association Rules » ? Veuillez commenter le paramétrage choisi et les résultats obtenus.

| **Support** | **Confidence** | **Number of rules** |
|---|---|---|

| | | |
|---|---|---|
| 0.05 | 0.5 | 130 |
| 0.05 | 0.8 | 17 |
| 0.03 | 0.5 | 2108 |
| 0.03 | 0.8 | 462 |
| 0.03 | 0.9 | 65 |

Plus le support est petit, plus on augmente la quantité d'articles liés entre eux. On peut ensuite faire une prédiction conditionnelle sur ces connexions et déduire d'autres articles connectés. La Confidence nous permet de filtrer les connexions estimées qui ont une fiabilité suffisamment élevée. Dans notre cas, nous n'obtenons aucune règle avec un support plus grand que 0.07. Dans le tableau ci-dessus, nous avons effectué plusieurs tests à plusieurs valeurs différentes. Nous pouvons voir qu'en diminuant le support, on obtient plus de règles.

### 2. Est-il possible d'utiliser une/des autre/s colonne/s à partir des données initiales pour produire des règles intéressantes ?

Oui, nous avons par exemple utilisé le Pays. Avec cette nouvelle sélection, nous avons plus facilement obtenu des règles (avec un support à 0.5 et une confidence à 0.9 nous obtenions 16 règles).

# 5. Clustering

### 1. Constatez-vous des changements pour des différents nombres d'itérations (max runs) de kmeans ?

| **Max run** | **Cluster_0** | **Cluster_1** | **Cluster_2** | **Cluster_3** | **Cluster_4** |
|---|---|---|---|---|---|
| 2 | 4239 | 4117 | 721 | 136 | 153 |
| 4 | 15 | 7145 | 136 | 141 | 1929 |
| 10 | 15 | 7145 | 136 | 141 | 1929 |
| 50 | 15 | 7145 | 136 | 141 | 1929 |

Nous pouvons constater des changements autant dans le nombre de données par cluster que dans la position des centroïdes jusqu'à 3 itérations. À partir de 4 itérations, nous obtenons les mêmes valeurs pour les différents clusters avec aucune amélioration au fil des itérations

2. Comment l'ensemble des données est partitionné en imposant seulement 2 clusters ? Comment les résultats changent-ils en augmentant le nombre des clusters ? Veuillez commenter les résultats obtenus en termes des centroïdes (centre de masse de chaque cluster), la moyenne des distances de chaque centroïde et, si pertinent, le genre des applications regroupées dans les clusters.

En utilisant k = 2, on obtient les résultats suivants:

| **Cluster** | **Items** | **Avg. within centroid distance** |
|---|---|---|
| 0 | 136 | -44.814 |
| 1 | 9230 | -3.221 |

Avec seulement 2 clusters, on peut constater que pour le cluster 1, la moyenne des distances vaut -44.814 et pour le second cluster -3.221. Cela montre que les points du cluster 1 sont plus proches de leur centroïde respectif que dans le cluster 0 et que le cluster 1 est dense. Il n'y a pas un regroupement notable au niveau des genres des applications, bien que les genres 'COMMUNICATION' et 'GAME' soient les plus représentés dans le cluster 0.

En augmentant le nombre de cluster, on obtient les moyennes des distances des centroïdes suivantes :

| K | **Avg. within centroid distance** |
|---|---|
| 3 | Avg. within centroid distance_cluster_0: -44.814<br>Avg. within centroid distance_cluster_1: -10.592<br>Avg. within centroid distance_cluster_2: -2.279 |
| 5 | Avg. within centroid distance_cluster_0: -10.706<br>Avg. within centroid distance_cluster_1: -0.587<br>Avg. within centroid distance_cluster_2: -44.814<br>Avg. within centroid distance_cluster_3: -6.132<br>Avg. within centroid distance_cluster_4: -1.275 |

Généralement, en augmentant le nombre de clusters, la moyenne des distances diminue, du fait que les données sont réparties dans plus de groupes. Avec K=3, on remarque que le cluster 0 reste inchangé et que l'algorithme a pu former deux nouveaux clusters à partir du cluster 1.

On remarque notamment un cluster qui est toujours éloigné (cluster 0 pour K=3, cluster 2 pour K=5, cluster 6 pour K=6) du reste des données.

Quant aux genres, même en essayant d'avoir un K égal au nombre de genres, on ne trouve pas un cluster avec un seul genre distinct, les genres étant généralement mélangés au sein d'un cluster

| | |
|---|---|
| 10 | Avg. within centroid distance_cluster_0: -10.706<br>Avg. within centroid distance_cluster_1: -5.214<br>Avg. within centroid distance_cluster_2: -0.197<br>Avg. within centroid distance_cluster_3: -13.333<br>Avg. within centroid distance_cluster_4: -0.242<br>Avg. within centroid distance_cluster_5: -1.196<br>Avg. within centroid distance_cluster_6: -31.913<br>Avg. within centroid distance_cluster_7: -1.421<br>Avg. within centroid distance_cluster_8: -1.166<br>Avg. within centroid distance_cluster_9: -0.159 |
| 20 | Avg. within centroid distance_cluster_0: -0.200<br>Avg. within centroid distance_cluster_1: -0.282<br>Avg. within centroid distance_cluster_2: -0.053<br>Avg. within centroid distance_cluster_3: -2.878<br>Avg. within centroid distance_cluster_4: -14.885<br>Avg. within centroid distance_cluster_5: -5.082<br>Avg. within centroid distance_cluster_6: -0.094<br>Avg. within centroid distance_cluster_7: -0.073<br>Avg. within centroid distance_cluster_8: -0.314<br>Avg. within centroid distance_cluster_9: -1.161<br>Avg. within centroid distance_cluster_10: -0.948<br>Avg. within centroid distance_cluster_11: -0.370<br>Avg. within centroid distance_cluster_12: -0.039<br>Avg. within centroid distance_cluster_13: -0.095<br>Avg. within centroid distance_cluster_14: -0.026<br>Avg. within centroid distance_cluster_15: -12.449<br>Avg. within centroid distance_cluster_16: -2.173<br>Avg. within centroid distance_cluster_17: -0.264<br>Avg. within centroid distance_cluster_18: -10.706<br>Avg. within centroid distance_cluster_19: -0.080 |

## 3. Comment les résultats changent-ils en utilisant d'autres algorithmes de clustering ?

Comme algorithme, nous avons utilisé k-Medoids (K-médoïdes) et avons obtenu les résultats suivants :

| **Max run** | **Cluster_0** | **Cluster_1** | **Cluster_2** | **Cluster_3** | **Cluster_4** |
|---|---|---|---|---|---|
| 4 | 2225 | 4265 | 1176 | 143 | 1557 |
| 6 | 225 | 4265 | 116 | 143 | 1557 |
| 7 | 143 | 2727 | 3835 | 432 | 2229 |
| 10 | 143 | 2727 | 3835 | 432 | 2229 |

Contrairement à l'algorithme de k-Means, on peut constater que 3 clusters principaux se sont formés, là où k-Means en avait 1 avec notamment la majorité des données. Cet algorithme de k-Medoids nécessite légèrement plus d'itérations, à savoir 7, avant de converger et également plus de temps de calcul (plusieurs minutes avec k-Medoids contre quelques secondes avec k-Means).

En augmentant le nombre de clusters, on retrouve généralement toujours des clusters prédominants (notamment 4 avec k = 10)