INITIATION AUX LANGAGES PYTHON & R / + SQL

Julie Bonalumi - Othmane Taya - Thi Mai Hien Vu - Sarah Souiber - Camille Malié

MSC2 Data Analytics & Marketing Manager - INSEEC

Juillet 2022

LES SUJETS PRÉSENTÉS

- Exercice 1
- Exercice 2
- Exercice 3
- Exercice Bonus

Exercice 1

Exercice 1

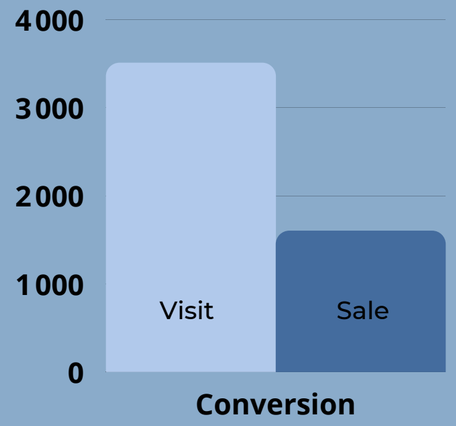
Période d'extraction des données : 9 juillet 2019 au 15 juillet 2019

```
: Qi = [impressions_df.timestamp.min(),impressions_df.timestamp.max()]
  Qi
```

```
[Timestamp('2018-07-09 00:09:28'), Timestamp('2018-07-15 19:05:24')]
```

Performances de la campagne

	Cost	CPM	Nb_impressions	Nb_impressions_unique	Repetition	Nb_conversions	CPA
visit	82.06263	0.711201	115386.0	69238.0	1.666513	3509.0	0.023386
sale	82.06263	0.711201	115386.0	69238.0	1.666513	1604.0	0.051161



La campagne est performante avec un fort nombre d'impression qui entraîne beaucoup de conversions tout en gardant un CPA faible.

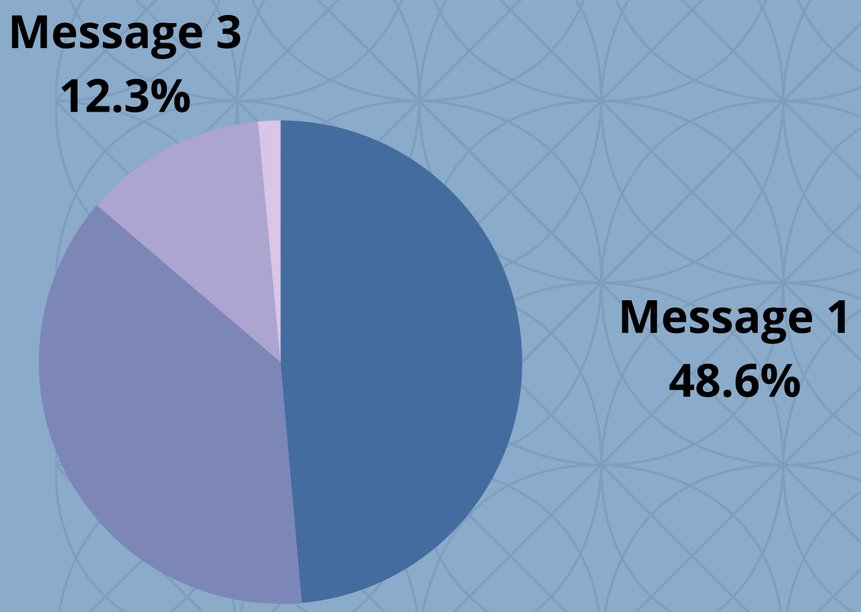
Exercice 1

Exposition aux messages publicitaires

Nombre de messages différents	Nombre de users imprimés	Part sur l'ensemble des users
Message 1	33 643	48.6%
Message 2	26 074	37.7%
Message 3	8 489	12.3%
Message 4	1 032	1.5%

Tableau de résultats dans Python

nb_messages	user_imprime	part total [%]
1	33643	48.590369
2	26074	37.658511
3	8489	12.260608
4	1032	1.490511

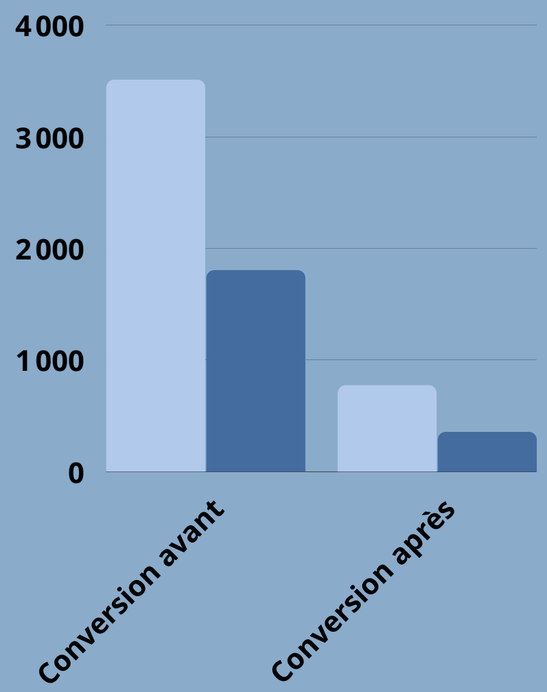


Exercice 1

Nouvelle règle : conversion attribuée si elle a lieu moins de 1 jour après l'exposition publicitaire.

Performances de la campagne avec la nouvelle règle :

	Cost	CPM	Nb_impressions	Nb_impressions_unique	Repetition	Nb_conversions	CPA
visit	82.06263	0.711201	115386.0	69238.0	1.666513	772.0	0.106299
sale	82.06263	0.711201	115386.0	69238.0	1.666513	355.0	0.231162



Avec la nouvelle règle la campagne est moins efficace, on remarque que le nombre de conversion est nettement inférieure par rapport à avant.

Exercice 2

Exercice 2

Comparaison des performances de la campagne entre les 2 grands types de messages

- Type 1 : offre promotionnelle
- Type 2 : offre générique

KPI of generic ads

```
: KPI(impressions_adtypeclean_df.loc(axis=0)[:, "Generic"])
```

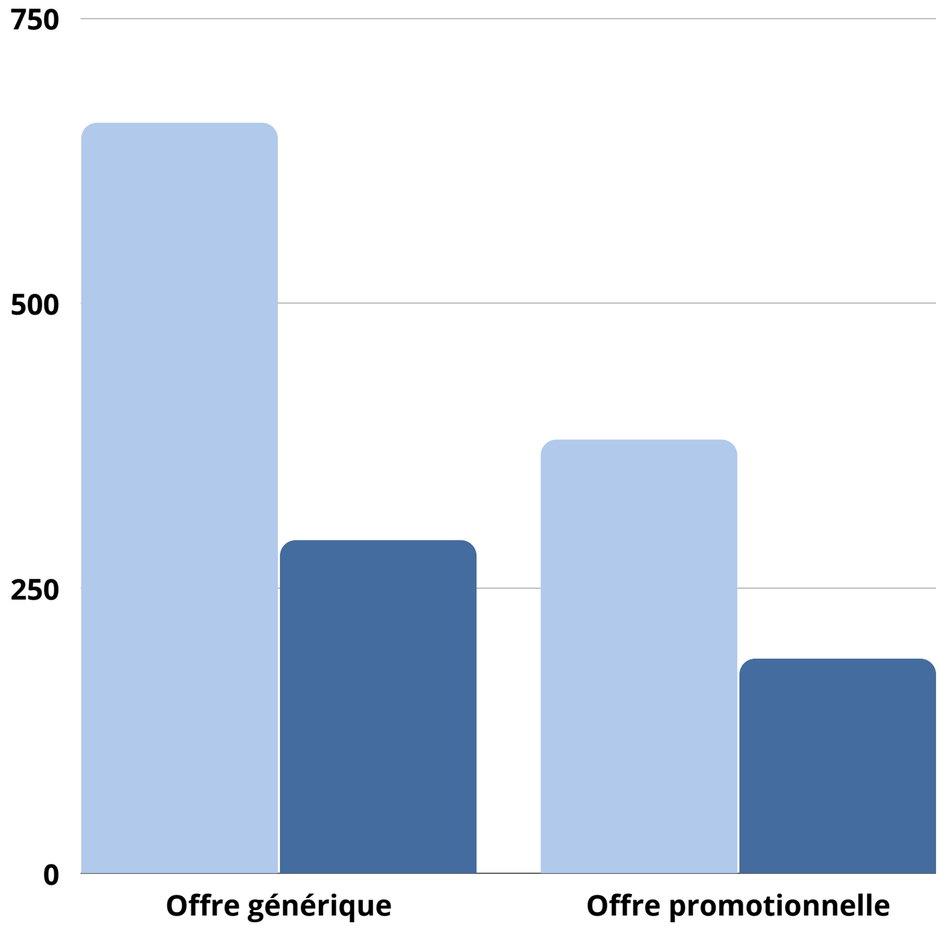
	Cost	CPM	Nb_impressions	Nb_impressions_unique	Repetition	Nb_conversions	CPA
visit	56.14601	0.755717	74295.0	58351.0	1.273243	658.0	0.085328
sale	56.14601	0.755717	74295.0	58351.0	1.273243	292.0	0.192281

KPI of promotional ads

```
: KPI(impressions_adtypeclean_df.loc(axis=0)[:, "Promotional"])
```

	Cost	CPM	Nb_impressions	Nb_impressions_unique	Repetition	Nb_conversions	CPA
visit	25.91662	0.630713	41091.0	36147.0	1.136775	380.0	0.068202
sale	25.91662	0.630713	41091.0	36147.0	1.136775	188.0	0.137854

La campagne de message générique est plus efficace de la campagne promotionnelle.
De plus, le retour sur l'investissement est meilleur car une campagne générique coûte deux fois plus cher qu'une campagne promotionnelle.



Exercice 2

Limites dans la comparaison des performances et méthodologie permettant d’éviter ces biais

On trouve que le coût global des campagnes génériques est élevé mais le CPA est plus fort.

En plus, on constate que plus qu'on dépense pour les campagnes, plus qu'on va recevoir les impressions et les conversions. Donc l'efficacité de campagne est trop influencée par les Spends. Mais en réalité, il faut tenir compte des autres KPIs pour évaluer les performances des campagnes.

Exercice 3

Exercice 3 : Machine learning classification supervisée

Méthode 1 : L'arbre de décision

Un arbre de décision est un schéma représentant les résultats possibles d'une série de choix interconnectés. Il permet à une personne ou une organisation d'évaluer différentes actions possibles en fonction de leur coût, leur probabilité et leurs bénéfices.

Avantages

- Capacité à combiner les résultats des arbres pour obtenir un résultat final plus fiable
- Ils sont faciles à comprendre et à utiliser
- De nouvelles options peuvent être ajoutées aux arbres existants
- Ils permettent de sélectionner l'option la plus appropriée parmi de nombreuses options
- Il est facile de les associer à d'autres outils de prise de décision

Inconvénients

- Peut devenir rapidement complexe lorsqu'il y a trop d'options
- Lors de la gestion de données de catégorie comportant plusieurs niveaux, le gain d'information est biaisé en faveur des attributs disposant du plus de niveaux

Méthode 2 : Le k-plus proche voisin

L’algorithme kNN suppose que des objets similaires existent à proximité. En d’autres termes, des éléments similaires sont proches les uns des autres.

Avantages

- Il est très simple et facile à mettre en œuvre.
- Il n’est pas nécessaire de construire un modèle, d’ajuster plusieurs paramètres ou de faire des hypothèses supplémentaires.
- L’algorithme est polyvalent : classification, régression et recherche d’informations

Inconvénients

- L’algorithme ralentit à mesure que le nombre d’observations ou de variables dépendantes/indépendantes augmente
- Le choix de la méthode de calcul de la distance ainsi que le nombre de voisins K peut ne pas être évident
- L'étape de prédiction est lente lorsqu'il y a de la complexité

MÉTHODE L'ARBRE DE DÉCISION

ÉTAPE 1 : L'IDÉE

Il faut avant toute chose avoir l'idée qui ménera à l'utilisation de l'arbre.
Cette idée constituera le noeud de décision.
Ensuite, il sera possible d'ajoutez des branches simples qui correspondent aux différentes décisions possibles.

ÉTAPE 2 : LDES NŒUDS D’OPPORTUNITÉ ET DE DÉCISION

Une fois que l'idée est en place, il faut ajouter les noeuds de décision à chaque décision pour développer l'arbre.
Un nœud d’opportunité peut s’accompagner d’une branche d’alternative, puisque certaines décisions ont plusieurs résultats possibles/potentiels.

ÉTAPE 3 : DÉVELOPPER L'ARBRE

Il faut ajouter des noeuds de décision et d'opportunité pour développer l'arbre jusqu'à son maximum.
Ensuite, vous avez des noeuds finaux pour présenter la fin du processus de création.
Une fois que l'arbre est terminé, il est possible d'analyser chaque décision.

MÉTHODE L'ARBRE DE DÉCISION

ÉTAPE 4 : CALCULER LES VALEURS

L'arbre doit être associé à des données quantitatives.
Il est envisageable d'estimer la valeur de chaque décision.
Pour calculer la valeur attendue de chaque option :

- Valeur attendue (VA) = (premier résultat possible x probabilité de survenue du résultat) + (deuxième résultat possible x probabilité de survenue du résultat) - coût

ÉTAPE 5 : ANALYSE

Une fois que les résultats attendus sont représentés pour chaque décision, il faut déterminer celle qui vous convient le mieux en fonction du niveau de risque prêt à prendre.
La valeur attendue la plus élevée n'est pas toujours celle à privilégier.
Il faut déterminer la meilleure façon d'analyser les décisions sur l'arbre.

LES OUTILS

GitMind

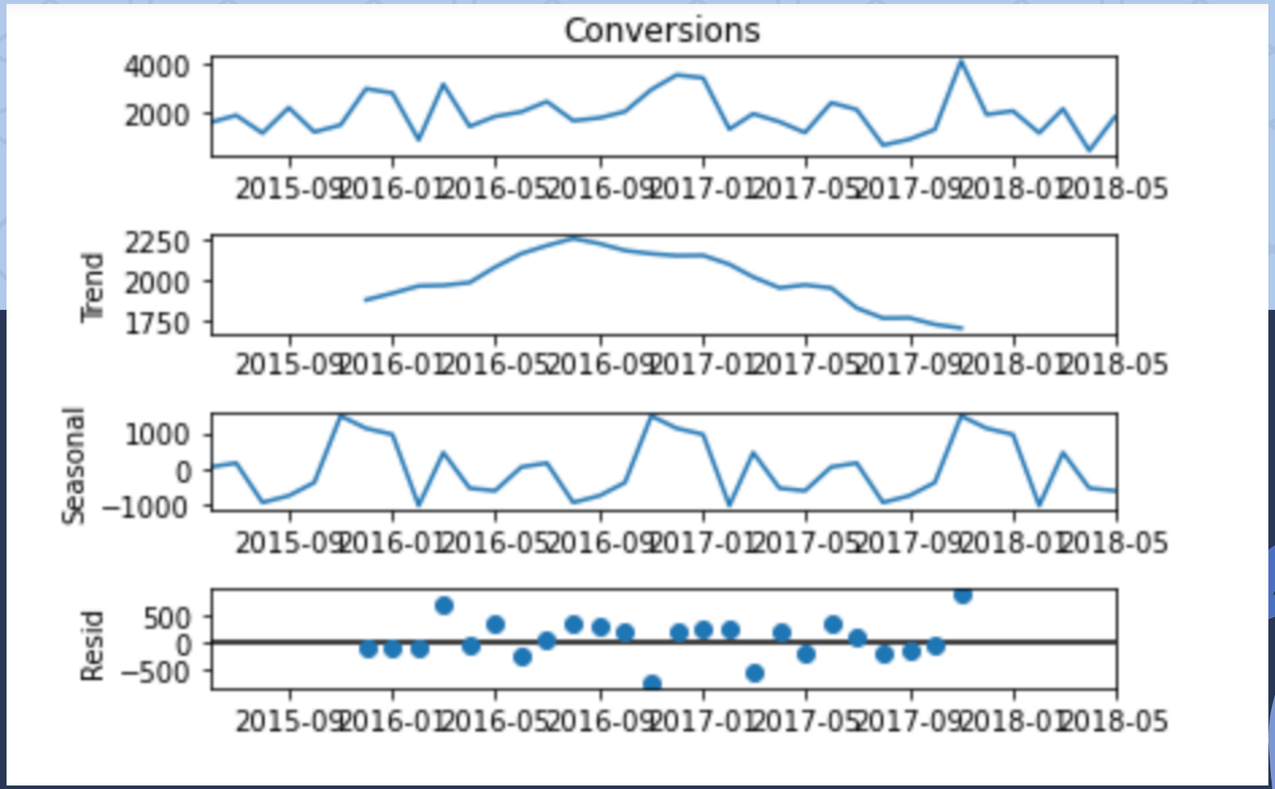
Scapple

mind meister

Exercice bonus

On propose d'utiliser la Régression comme la méthode de prévision.
Tout d'abord, on commence par le nettoyage de la colonne Mois et regarde les caractéristiques des données de type **Séries Chronologiques Multivariées.**
On remarque qu'il y a une **tendance saisonnelle** assez forte. Autrement dit, en regardant la 3e courbe, le nombre de conversions est plus élevé au début de chaque année

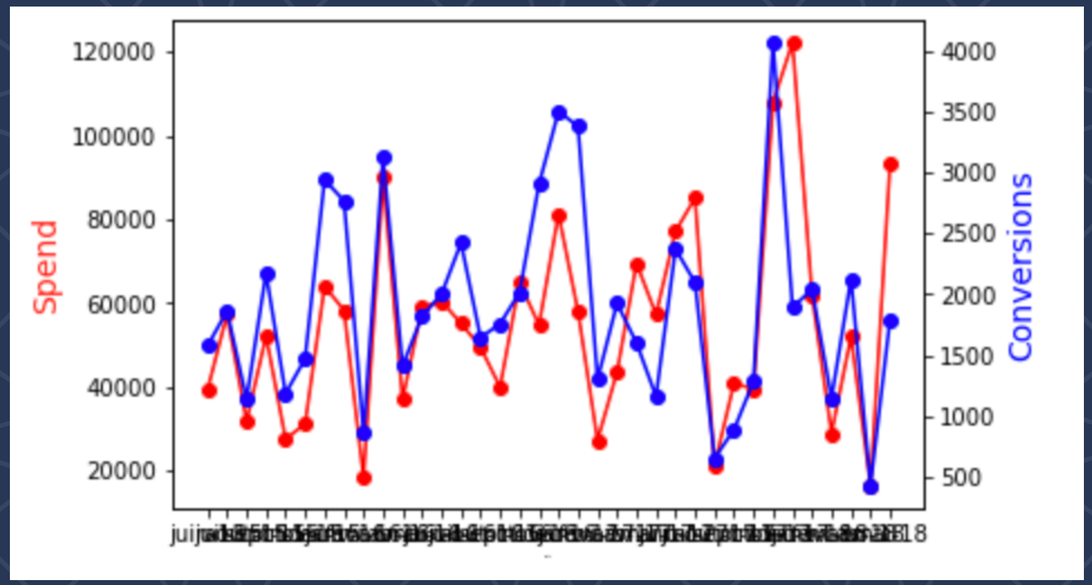
Time	Spend	Conversions
2015-06-01	39456.23	1578
2015-07-01	57338.93	1864
2015-08-01	31802.36	1138
2015-09-01	52195.88	2176
2015-10-01	27642.56	1185
2015-11-01	31215.60	1471
2015-12-01	64036.93	2942
2016-01-01	58342.15	2768
2016-02-01	18326.47	863
2016-03-01	90135.24	3127

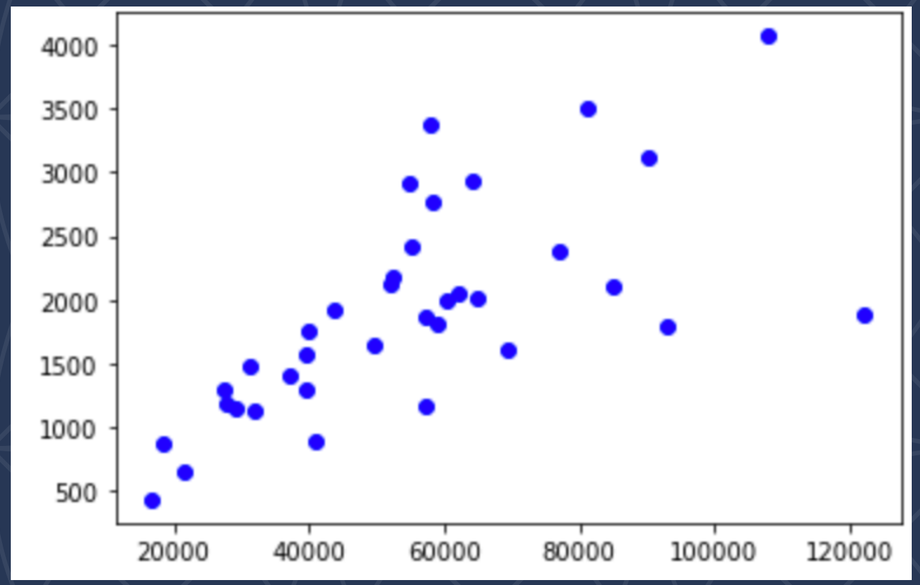


On constate également qu'il y a une corrélation entre le montant de Spend et le nombre de Conversions. On a réalisé un test du coefficient de corrélation et nous avons obtenu 0.74. Plus le coefficient est proche de 1, plus la relation linéaire positive entre les variables est forte.

```
Entrée [41]: stats.spearmanr(forecast_df["Spend"], forecast_df["Conversions"]).correlation

  Out[41]: 0.7402831402831404
```





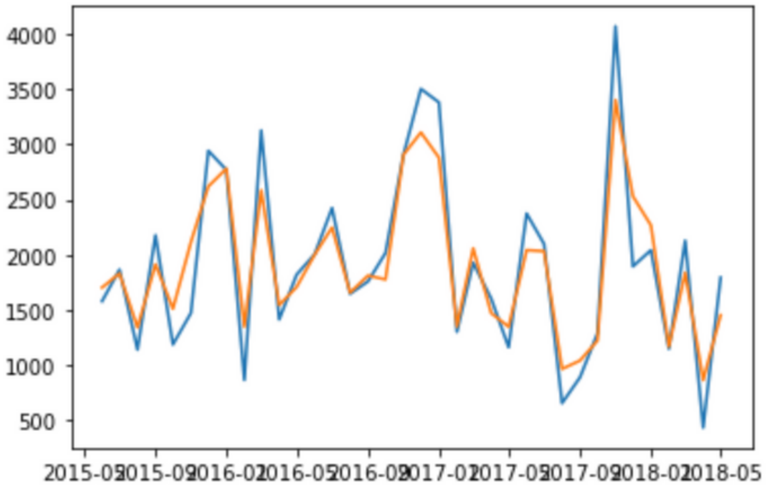
Nuage de points et corrélation

Entrée [45]:
```
from sklearn.ensemble import RandomForestRegressor
# fit the model
model = RandomForestRegressor()
model.fit(X_train, Y_train)

# predict on the same period
preds = model.predict(X_train)

# plot what has been learned
plt.plot(forecast_df.index, Y_train)
plt.plot(forecast_df.index, preds)
```

Out[45]: [<matplotlib.lines.Line2D at 0x1a5da7f62b0>]



Exercice bonus bonus

On a choisi **RandomForestRegressor** pour construire le modèle dont les entrées sont les mois et années. On n'utilise pas le Spend car on n'a pas le montant pour 2 mois prochains.

Après l'apprentissage, il y a un très bon accord entre la courbe de prédiction (en orange) et cela de conversions de la même période.

En plus, on a un R2 score de 86% qui montre une grande pertinence du modèle.

Entrée [47]:
```
score = model.score(X_train, Y_train)
print("R-squared:", score)
```

```
R-squared: 0.8582393900746712
```

Voici la prédiction pour le juin et juillet 2018

Entrée [48]:
```
X_test = np.array([[6,2018],[7,2018]]).T
# predict on the same period
preds = model.predict(X_test)
preds
```

Out[48]: array([1701.56, 2520.06])