PROJET PYTHON A4

NGUYEN BINH ARTHUR OULES LOUIS SCELLES ARTHUR

PRÉSENTATION DU DATASET

- Thème : popularité d'articles en ligne (nombre de partages)

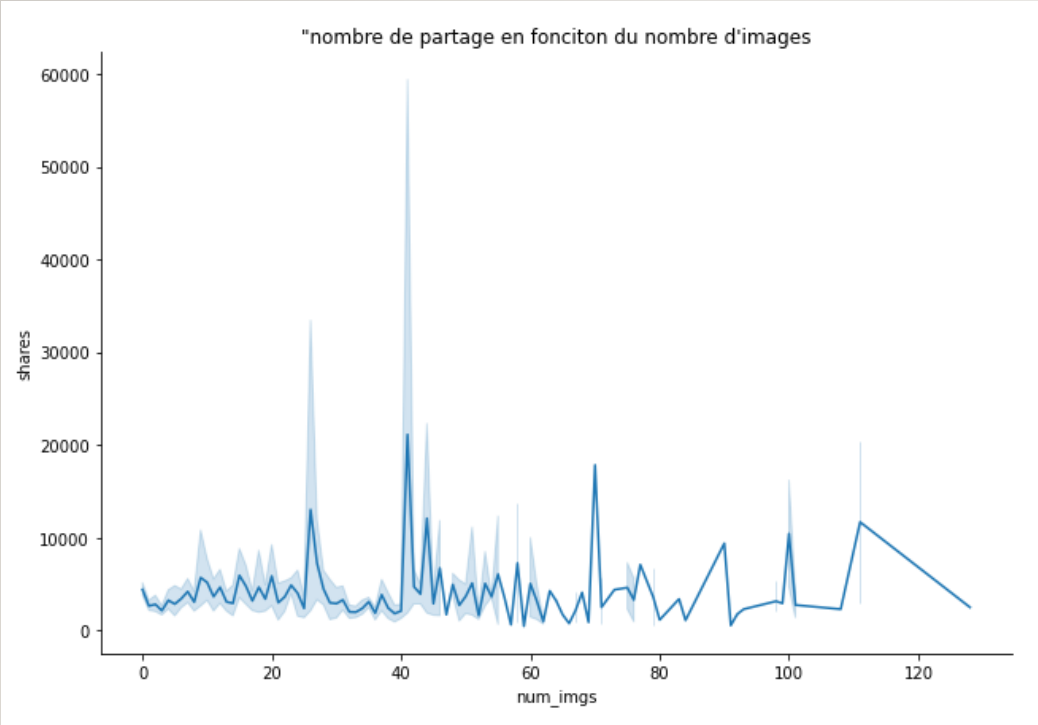
- Quelques paramètres : nombre de mots dans le titre, jour de la semaine de publication de l'article, subjectivité du titre et du texte, etc.

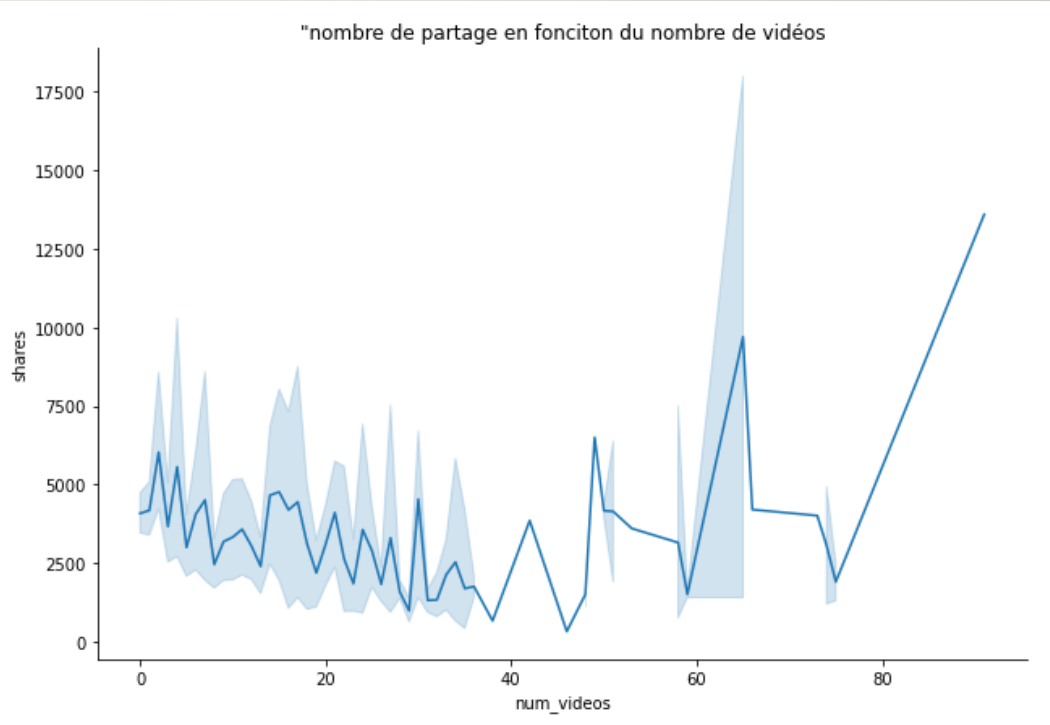
- Lien : https://archive.ics.uci.edu/ml/datasets/Online+News+Popularity

DATA PRE-PROCESSING

- Aucun NAN dans le dataset (mais des valeurs nulles)
- Nous avons renommé les colonnes
- Transformation de multiples booléens en une seule variable
- Des variables que nous ne comprenons pas (ex: LDA)

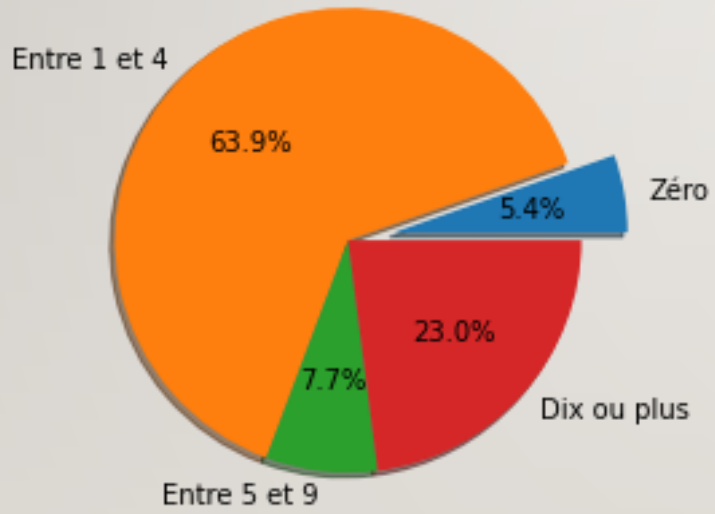
DATA VISUALISATION



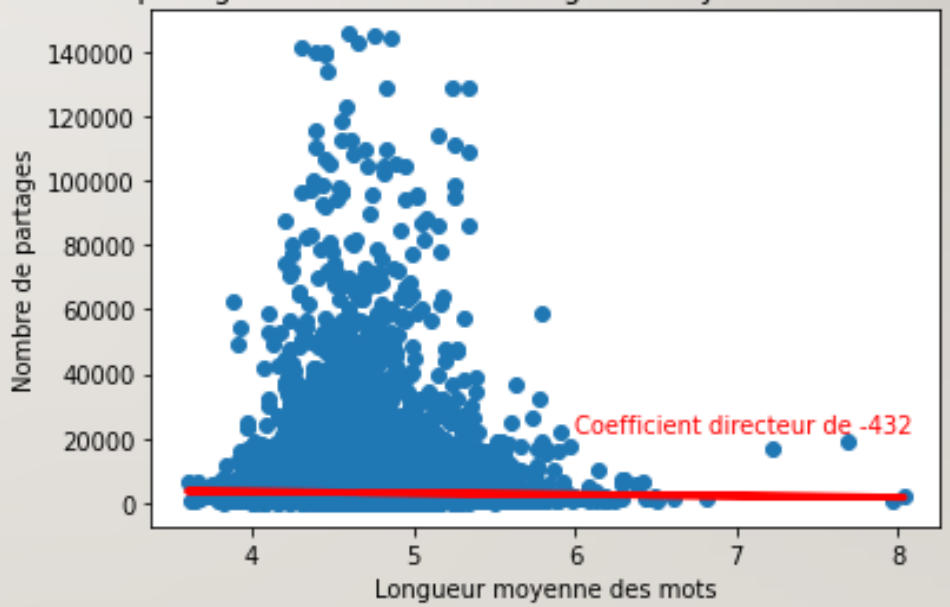


DATA VISUALISATION

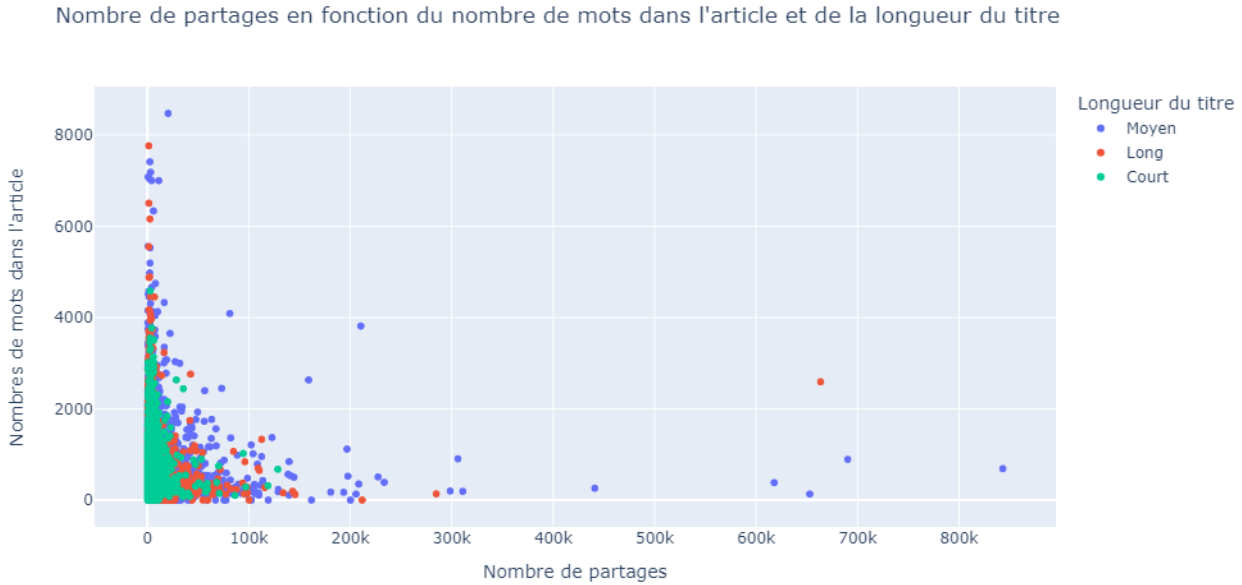
Partages en fonction du nombre d'images et vidéos présents dans l'article

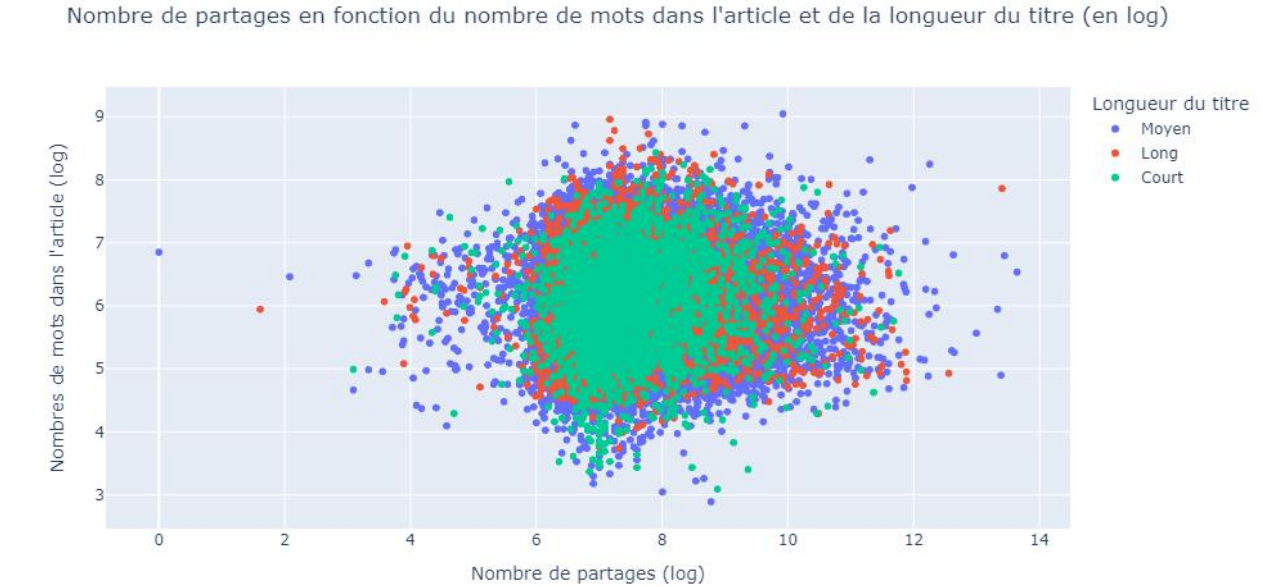


Nombre de partages en fonction de la longueur moyenne des mots dans l'article

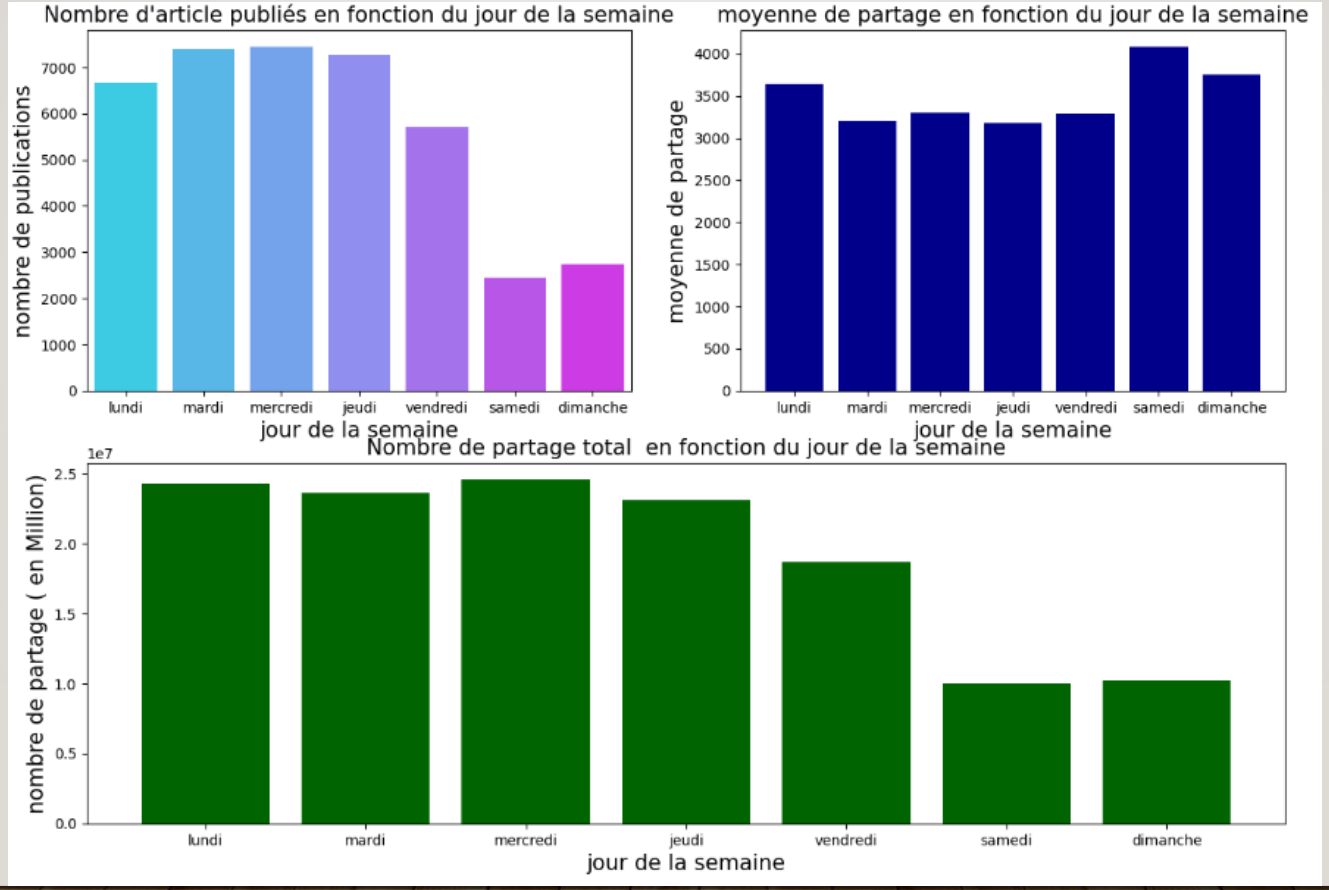


DATA VISUALISATION

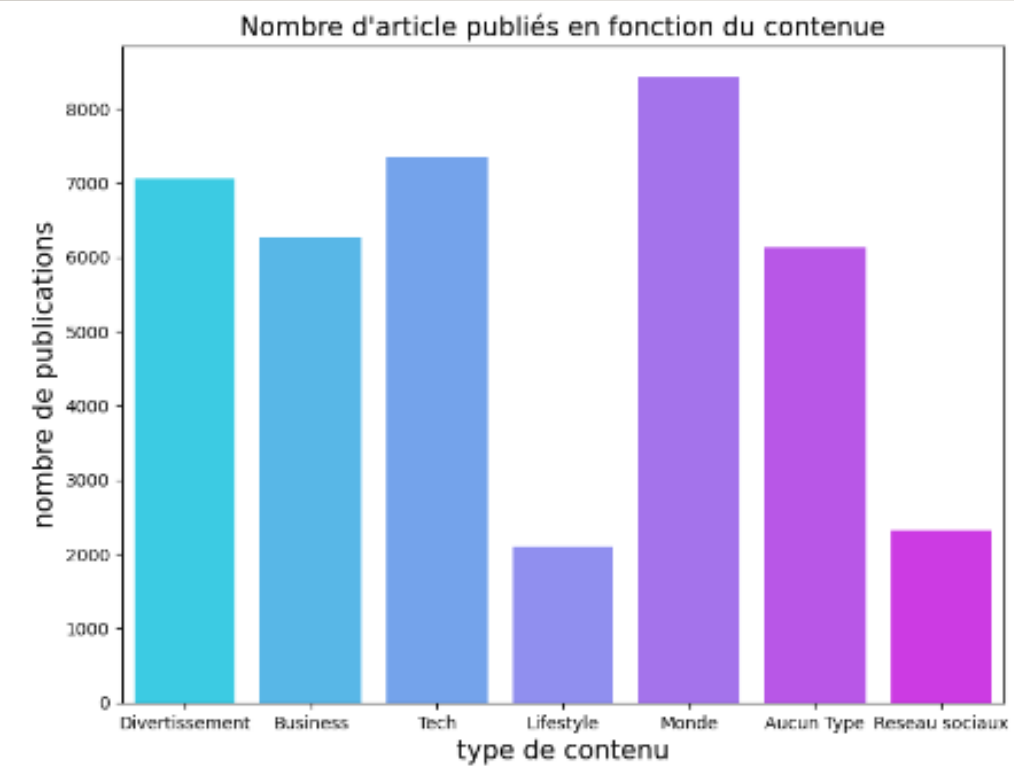


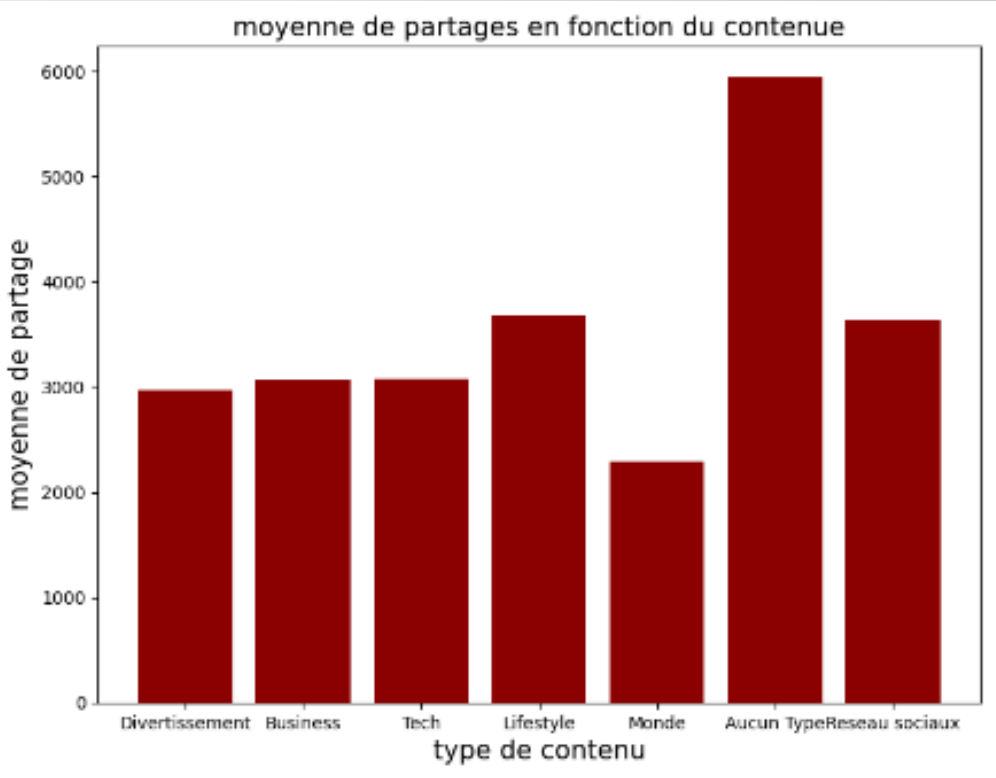


DATA VISUALISATION

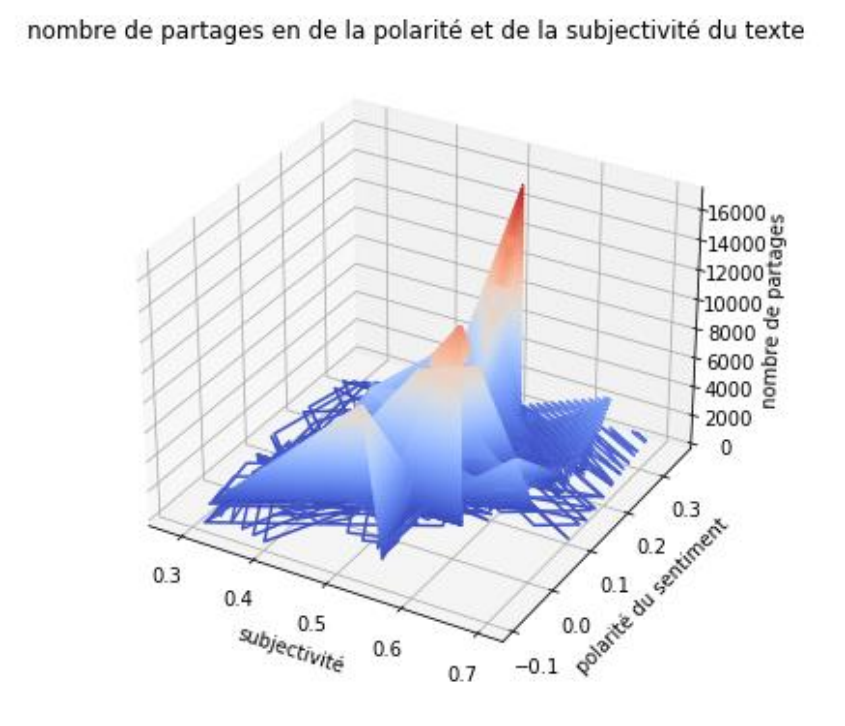


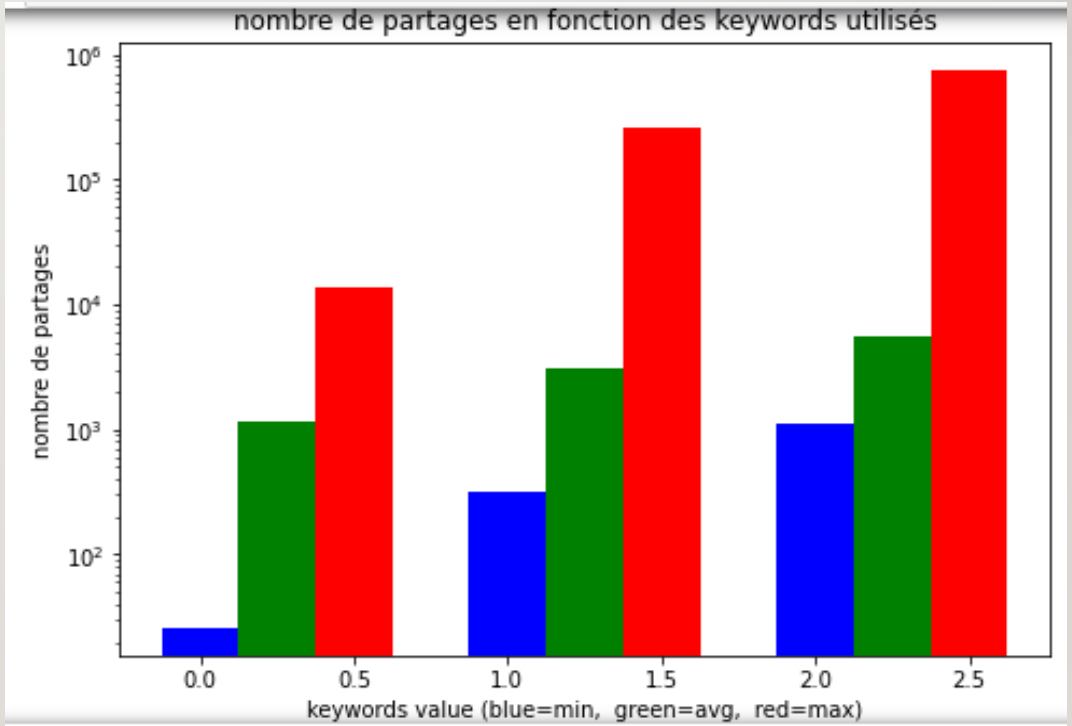
DATA VISUALISATION





DATA VISUALISATION





MODELING

```
train, test = train_test_split(data2, test_size=0.2)
X_train = train.drop("popularity", axis=1)
Y_train = train["popularity"]
X_test  = test.drop("popularity", axis=1)
Y_test= test["popularity"]
X_test
```

Entrée [56]:

```
random_forest = RandomForestClassifier(n_estimators=100)
random_forest.fit(X_train, Y_train)

pred_R = random_forest.predict(X_test)
trainpred_R = random_forest.predict(X_train)

acc_R = accuracy_score(Y_test,pred_R)
f_R = fbeta_score(Y_test,pred_R,beta=1)
AUC_R = roc_auc_score(Y_test,pred_R)

trainacc_R = accuracy_score(Y_train,trainpred_R)
trainf_R = fbeta_score(Y_train,trainpred_R,beta=1)
trainAUC_R = roc_auc_score(Y_train,trainpred_R)

print(acc_R,f_R,AUC_R,trainacc_R,trainf_R,trainAUC_R)
confusion_matrix(Y_test,pred_R)
confusion_matrix(Y_train,trainpred_R)
```

```
0.6609912977676883 0.6557377049180327 0.6609032089335223 1.0 1.0 1.0
```

Out[56]:

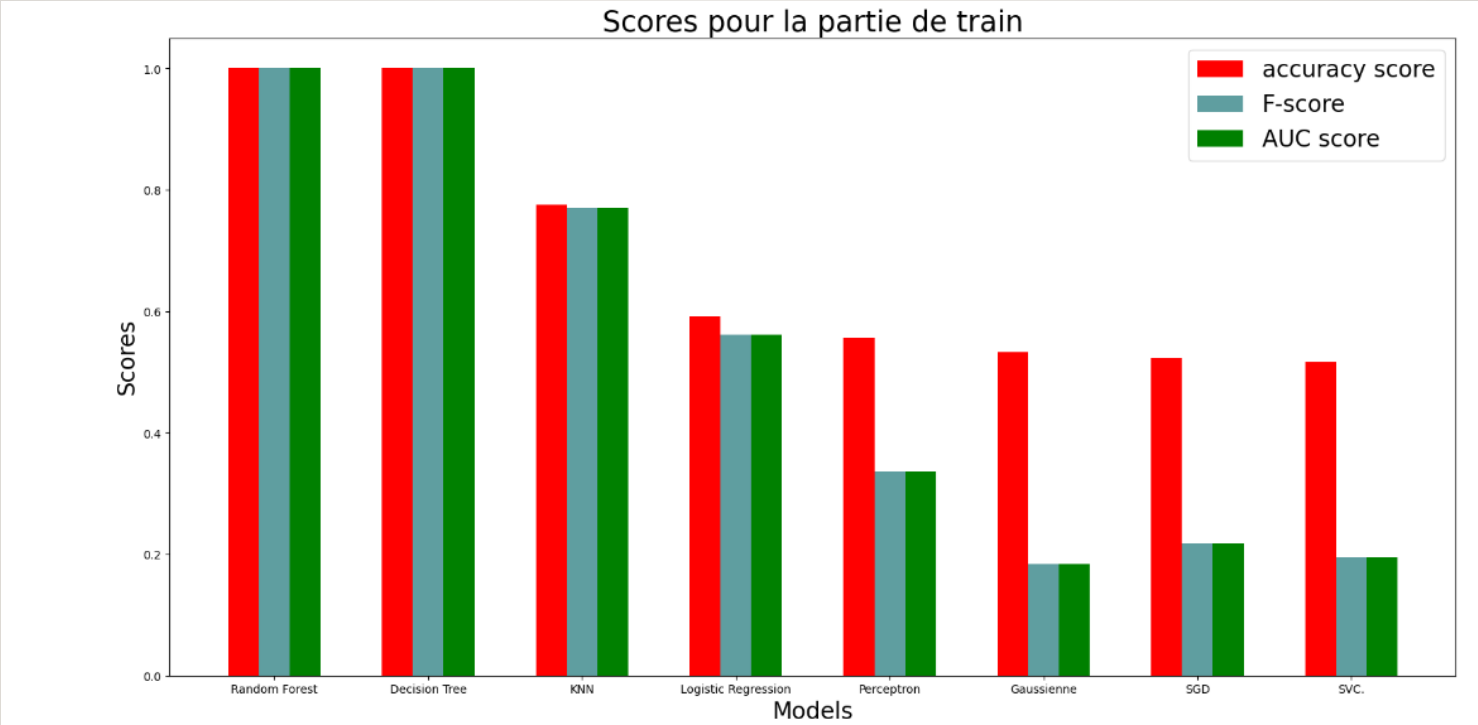
```
array([[16072,     0],
       [    0, 15643]], dtype=int64)
```

MODELING

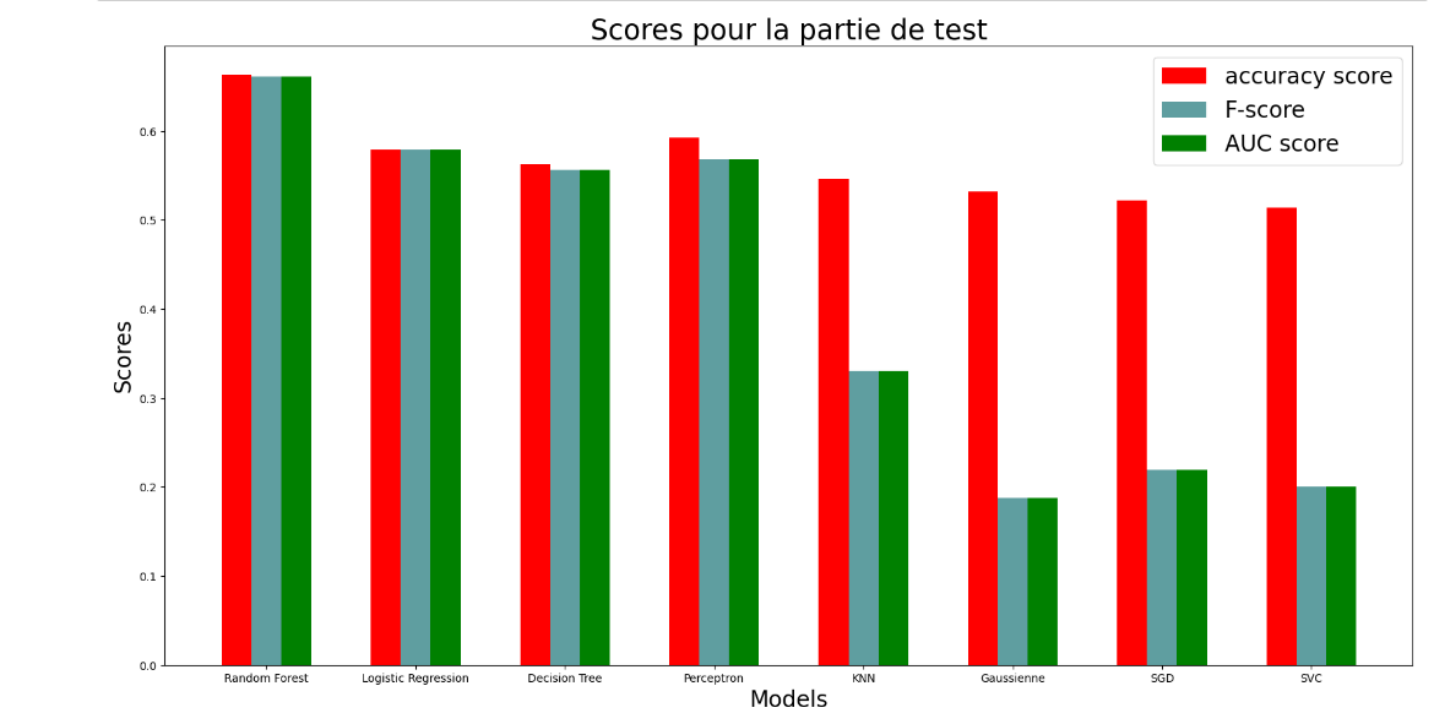
Out[57]:

	Model	train Acc Score	train F-Score	train AUC Score	test Acc Score	test F-Score	test AUC Score
3	Random Forest	1.000000	1.000000	1.000000	0.660991	0.655738	0.660903
7	Decision Tree	1.000000	1.000000	1.000000	0.579014	0.578748	0.579083
1	KNN	0.775437	0.770451	0.775285	0.563123	0.556466	0.563025
2	Logistic Regression	0.591581	0.560781	0.590741	0.592130	0.567994	0.591567
0	Support Vector Classification.	0.555825	0.337051	0.551462	0.546349	0.330043	0.542715
4	Gaussienne	0.532240	0.184218	0.526566	0.531971	0.187432	0.527174
5	Perceptron	0.523695	0.217143	0.518493	0.522386	0.219658	0.518002
6	Stochastic Gradient Decent	0.516286	0.195247	0.510983	0.514315	0.200540	0.509877

MODELING


Scores pour la partie de train
accuracy score
F-score
AUC score
Scores
1.0
0.8
0.6
0.4
0.2
0.0
Random Forest
Decision Tree
KNN
Logistic Regression
Perceptron
Gaussienne
SGD
SVC.
Models

MODELING


Scores pour la partie de test
accuracy score
F-score
AUC score
Scores
0.0
0.1
0.2
0.3
0.4
0.5
0.6
Random Forest
Logistic Regression
Decision Tree
Perceptron
KNN
Gaussienne
SGD
SVC
Models

MODELING

```
rf=RandomForestClassifier(random_state=1)

params={"n_estimators": [25,30],
              "max_depth": [3,10,15],
              "max_features": [3, 4,6],
              "min_samples_split": [3, 4,6],
              "min_samples_leaf": [3, 4,6],
             }

rand_search_rfc=RandomizedSearchCV(rf,param_distributions=params,cv=4)

rand_search_rfc.fit(X_train,Y_train)

print(rand_search_rfc.best_params_)
```

```
{'n_estimators': 30, 'min_samples_split': 3, 'min_samples_leaf': 6, 'max_features': 4, 'max_depth': 10}
```

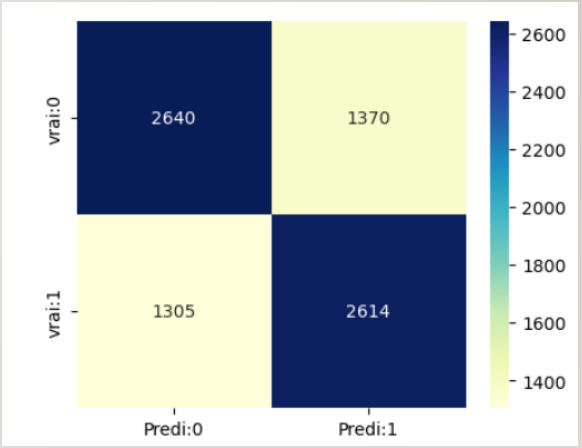
```
rf=RandomForestClassifier(**rand_search_rfc.best_params_)
rf.fit(X_train,Y_train)


y_pred=rf.predict(X_test)
y_prob=rf.predict_proba(X_test)[:,1]

from sklearn.metrics import accuracy_score,roc_curve,roc_auc_score


print('Accuracy of random forest test :',accuracy_score(y_pred,Y_test))
print('AUC of random forest test :',roc_auc_score(Y_test,y_prob))
print('Improvement of Acc {:0.2f}%.'.format( 100 * (accuracy_score(y_pred,Y_test) - accuracy_score(pred_R,Y_test)) / accurac
print('Improvement of AUC {:0.2f}%.'.format( 100 * (roc_auc_score(Y_test,y_prob) - roc_auc_score(pred_R,Y_test)) / roc_auc_s
```

```
Accuracy of random forest test : 0.6626308487829486
AUC of random forest test : 0.7218144992201813
Improvement of Acc 0.25%.
Improvement of AUC 9.21%.
```



API

NOUS AVONS FAIT LE CHOIX DE NE PAS TRAITER CETTE PARTIE DU TRAVAIL.

CONCLUSION