Rapport d'Analyse

OpenClassrooms – Projet n°8

Kickstarter : Peut-on prédire si un projet va réussir ou échouer avant qu'il soit lancé ?

J. Vangansberg

Résumé

Le financement participatif ou *crowdfunding* connaît un essor important depuis ces 10 dernières années[1]. C'est devenu une alternative aux moyens de financement classique comme les banques. Cette analyse porte sur l'une des plateformes les plus connues : Kickstarter[2]. Cette structure existe depuis 2009 et plus de 500 000 projets y ont été proposés. 192 000 ont atteint leur objectif[3]. L'entreprise s'est ouverte à l'international en 2014 mais elle reste principalement utilisée par des états-uniens.

Dans cette analyse, j'utilise le *Stochastic Gradient Boosting* sur un jeu de données extrait de *Kaggle* pour définir si un projet va réussir ou échouer avant qu'il soit lancé. Le modèle a un taux d'*accuracy* de 68.65%. Les variables les plus importantes pour la réussite d'un projet sont : un objectif de financement raisonnable et une période de financement courte. De plus, il existe des écarts importants entre les taux de réussite des différentes catégories. En effet, moins de 10% des *Apps* réussissent à atteindre leur objectif tandis que ce chiffre atteint plus de 65% pour les jeux de société. Cette plateforme est plutôt adaptée aux petits projets qualitatifs qu'aux projets qui nécessitent un apport en capital important.

[1] *Crowdfunding: Mapping EU markets and events study -European Commission*
[2] https://www.kickstarter.com/
[3] Chiffres en 2020

Table des matières

1. Introduction

Ma démarche est de tenter de prévoir si un projet proposé sur Kickstarter va réussir ou échouer avant qu'il soit lancé. Le jeu de données vient de la plateforme *Kaggle* et il contient environ 370 000 projets lancés entre 2009 et 2017. Afin d'effectuer des prédictions sur l'échec ou le succès d'un projet, je vais utiliser différents algorithmes de classification dont je comparerai la précision. Dans ce but, j'utiliserai un mélange entre des données catégorielles et des données numériques. Ces données seront préparées afin que les algorithmes puissent les utiliser dans un cadre optimal. Lorsque les différents modèles seront entraînés et testés, je sélectionnerai le meilleur d'entre eux pour approfondir son analyse.

2. Nettoyage des données

Le dataset contient 378661 entrées et 14 variables. Chaque entrée concerne un projet :

Nom de la variable	Description de la variable	Type	Données non-nulles
`ID`	Numéro d'identifiant	Int64	378661
`name`	Nom	Object	**378657**
`main_category`	La catégorie principale	Object	378661
`category`	La sous-catégorie	Object	378661
`currency`	La devise locale	Object	378661
`country`	Pays d'origine	Object	378661
`goal`	Objectif de financement fixé par le créateur dans sa devise	Float64	378661
`usd_goal_real`	Objectif de financement en dollars fixé par le créateur	Float64	378661
`launched`	Date de lancement	Object	378661
`deadline`	Date butoir	Object	378661
`pledged`	Promesse de financement dans la devise locale	Float64	378661
`usd_pledged`	Promesse de financement en dollars (conversion faite par Kickstarter)	Float64	**372041**
`usd_pledged_real`	Promesse de financement en dollars (conversion faite par Fixer.io API)	Float64	378661
`backers`	Nombre de personnes qui soutiennent financièrement le projet	Float64	378661
`state`	L'état du projet (*successful, failed, canceled, suspending ou still live*[4])	Object	378661

TABLEAU 1 – Description du jeu de données

Un projet est défini comme *successful* si `usd_goal_real` est inférieur à `usd_pledged_real`. Par conséquent les variables : `pledged`, `usd_pledged`, `usd_pledged_real` mais également `backers` sont des *data leakage*[5], c'est-à-dire que ces informations ne sont pas accessibles au lancement du projet et donc au moment où l'on souhaite faire la prédiction. Ces informations ne seront donc pas utilisées dans les algorithmes.

2.1. Données manquantes

Le tableau n°1 nous informe que deux variables contiennent des *NaN*[6]. Cependant certains *datasets* peuvent enregistrer les valeurs manquantes sous d'autres formes. C'est le cas ici pour

[4] Réussi, échoué, annulé, suspendu ou en cours
[5] Littéralement « Fuite de données »
[6] *Not a Number*

la variable `country` qui enregistre ses *NaN* sous la valeur `'N,0 »'`. Ces données sont donc remplacées par des *NaN*.

Le TABLEAU 1 nous indique que la variable `usd_pledged` contient également des *NaN*. En les inspectant, il s'avère que ces enregistrements ont également des *NaN* dans la variable `country` et ont un statut *undefined*. Environ 1% des données ont ces caractéristiques. Je choisis de supprimer ces enregistrements car ils sont peu nombreux et d'une mauvaise qualité.

1.1. Données dupliquées

Aucun enregistrement n'a de données identiques sur toutes les colonnes. C'est également le cas, pour la variable `ID` qui ne comporte que des entrées uniques. Cette variable sera donc utilisée en tant qu'index.

Cependant, 0.75% données ont des noms identiques. Comme ils ont des identifiants différents, je décide de les conserver. En effet, certaines données ont des noms identiques mais sont rattachées à des catégories différentes.

2.2. Type de données

Le TABLEAU n°1 nous indique que les variables `launched` et `deadline` sont encodées en `object` alors qu'elles représentent des dates. Ces données sont transformées en `datetime64[ns]`. Grâce à ces modifications, je peux créer de nouvelles variables qui me seront utiles pour la phase exploratoire et la modélisation :

`month` : le mois de lancement

`day_of_the_week` : le jour de la semaine du lancement

`launch_hour`[7] : l'heure de lancement

`time` : nombre de jours entre le lancement et la date butoir

3. Analyse exploratoire

Le premier indicateur que je choisis d'analyser est le nombre de projets lancés par mois depuis la création de Kickstarter. On peut constater que le nombre de projets lancés croit lentement jusqu'en 2012 date à laquelle Kickstarter s'est implanté dans d'autre pays[8]. En 2014, on peut noter une très forte hausse mais qui s'est corrigé rapidement. Enfin sur cette FIGURE 1, nous pouvons noter une saisonnalité annuelle avec une baisse du nombre de projets lancés en fin d'année.

[7] Cette donnée a été découpée en tranche de deux heures pour faciliter l'analyse de l'importance des variables.
[8] https://www.kickstarter.com/

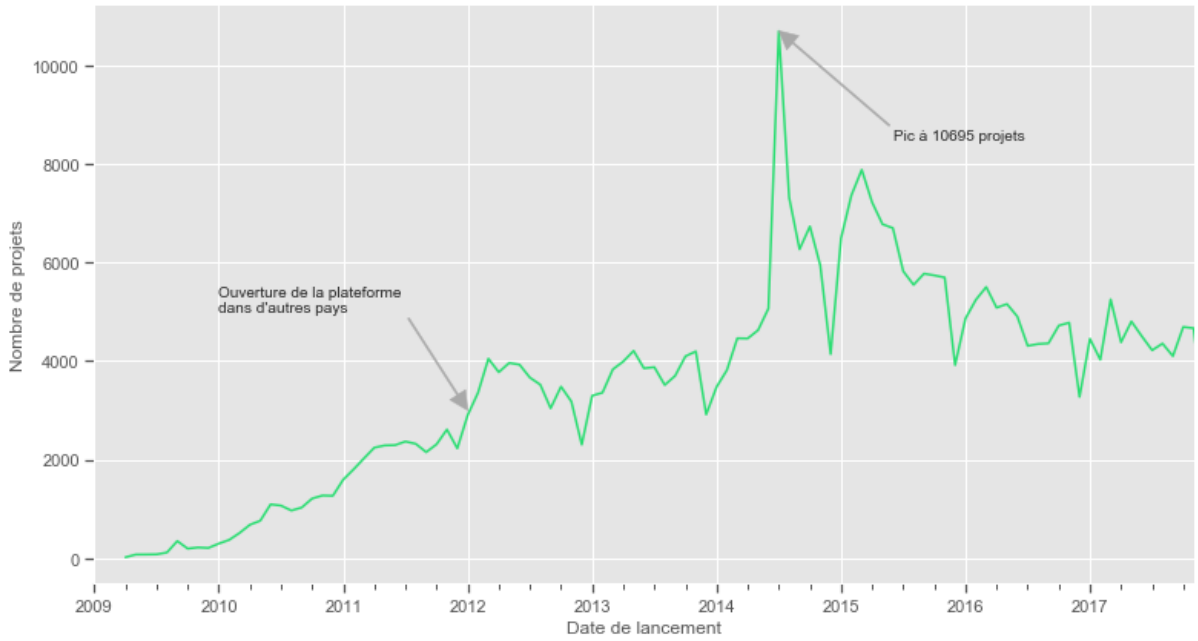

FIGURE 1 – Nombre de projets lancés sur Kickstarter entre le 28 avril 2009 et le 30 novembre 2017

3.1. La cardinalité des variables catégorielles

Le TABLEAU n°2 nous indique que la variable `category` contient 159 valeurs uniques.

Variable	Cardinalité
`main_category`	15
`category`	159
`currency`	14
`country`	22
`state`	5

TABLEAU 2 – Cardinalité par variable catégorielle

3.2. Etats des projets

Plus de 50% des projets échouent, environ 10% sont annulés et 35% réussissent à atteindre leur objectif. Le but de cette analyse étant de prévoir si un projet échoue ou réussi, je vais restreindre le reste de cette analyse à ces deux statuts (*successful* et *failed*).

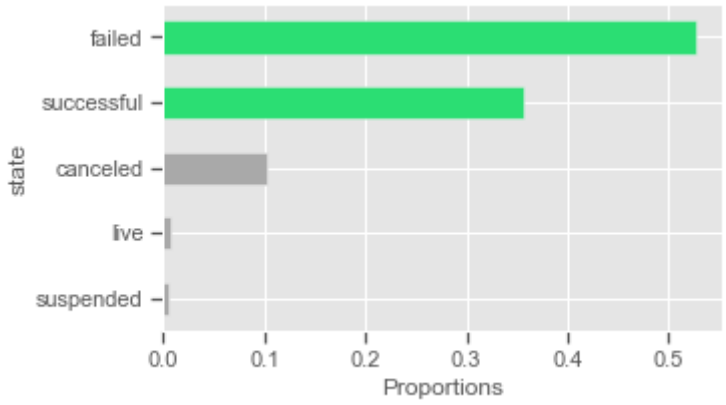

FIGURE 2 – Diagramme à barres : Proportion des projets par état

3.3. Les catégories et les sous-catégories

Les catégories les plus représentées sont *Film & Video*, *Music* et *Publishing* (FIGURE 3). On peut noter que la catégorie *Music* a également un taux de succès élevé (FIGURE 4). Enfin, la catégorie avec le moins de projet, *Dance*, a le taux de succès le plus élevé avec plus de 60% réussite.

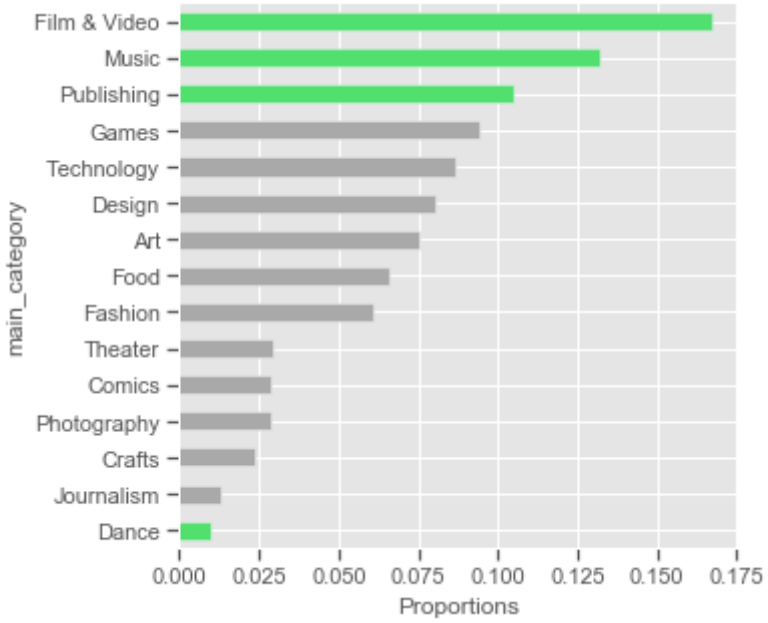

FIGURE 3 – Diagramme à barres : Proportions du nombre de projets par catégorie principale

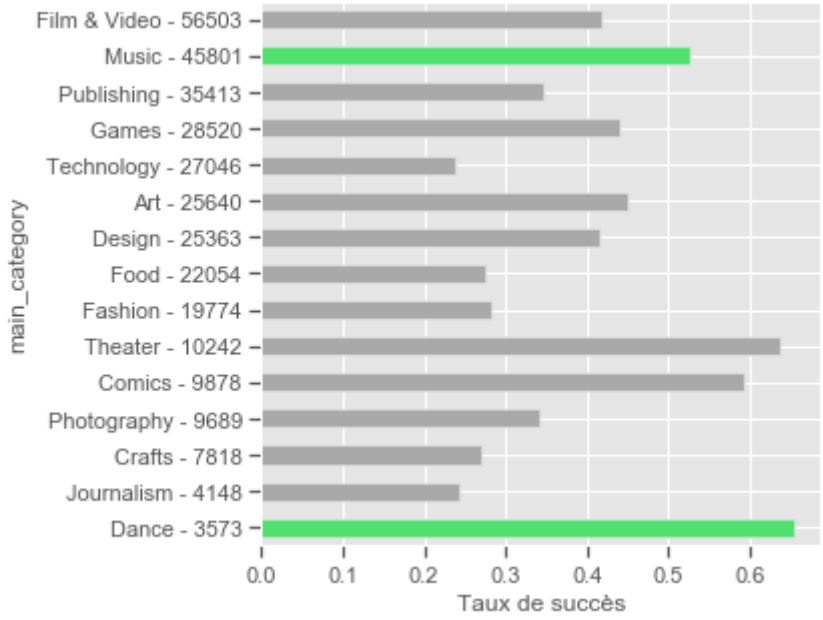

FIGURE 4 – Diagramme à barres : Taux de succès par catégorie principale

Quand on s'intéresse à *Technology*, on peut voir que des sous-catégories populaires comme les applications mobiles et web ont un taux de succès de moins de 10%. Cependant, on constate

que certains projets moins fréquents ont un taux de succès qui dépasse souvent les 40%[9]. (FIGURE 5)

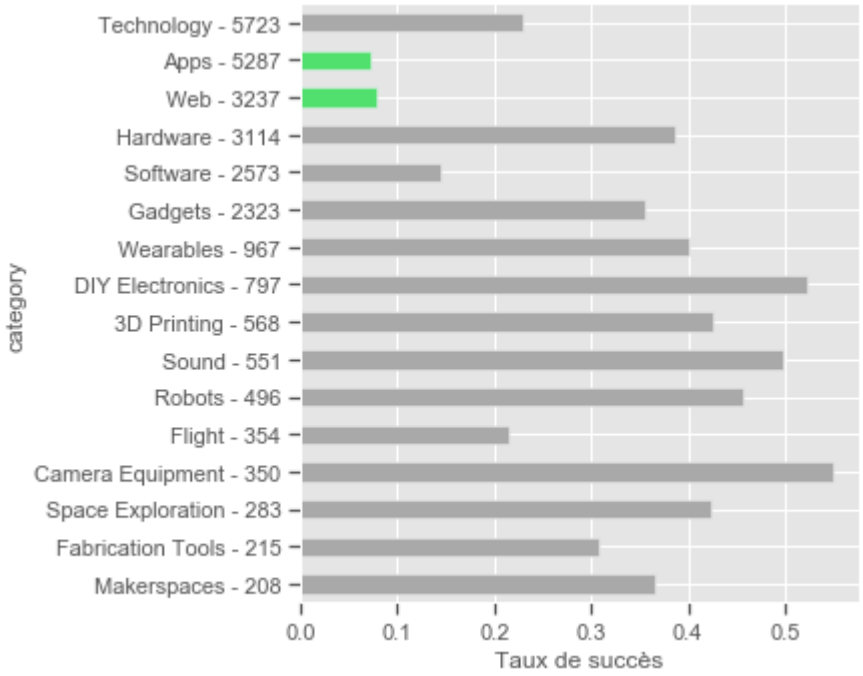

FIGURE 5 – Diagramme à barres : Taux de succès des sous-catégories de la catégorie *Technology*

Cette relation entre la fréquence et le taux de succès n'est pas systématique comme nous montre la FIGURE 6, où le taux de succès de la sous-catégorie la plus populaire *Tabletop Games* est largement supérieur aux autres catégories.

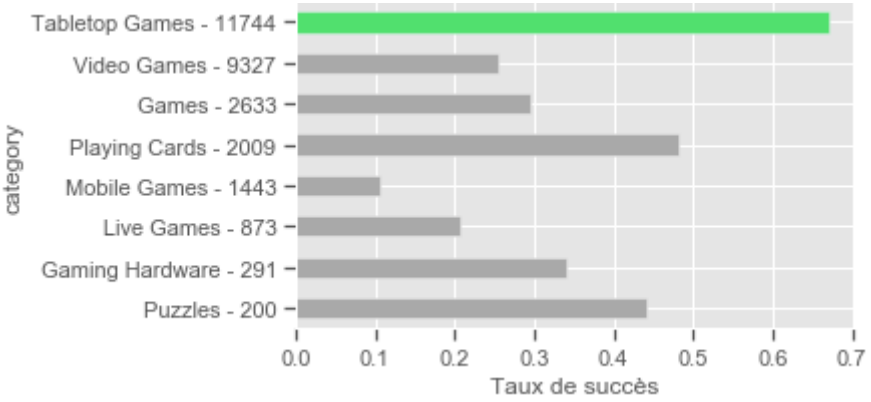

FIGURE 6 – Diagramme à barres : Taux de succès des sous-catégories de la catégorie *Games*

Ces informations nous indiquent que les chances de réussites d'un projet dépendent en partie de la catégorie/sous-catégorie auquel il appartient.

[9] *DIY Electronics, 3D Printing, Sound, Robots, Camera Equipment, Space Exploration*

3.4. Les pays

La FIGURE 7 nous rappelle que Kickstarter est avant tout une entreprise états-unienne. En effet, presque 80% des projets proviennent de ce pays et cela même si la plateforme s'est exportée à l'international en 2012.

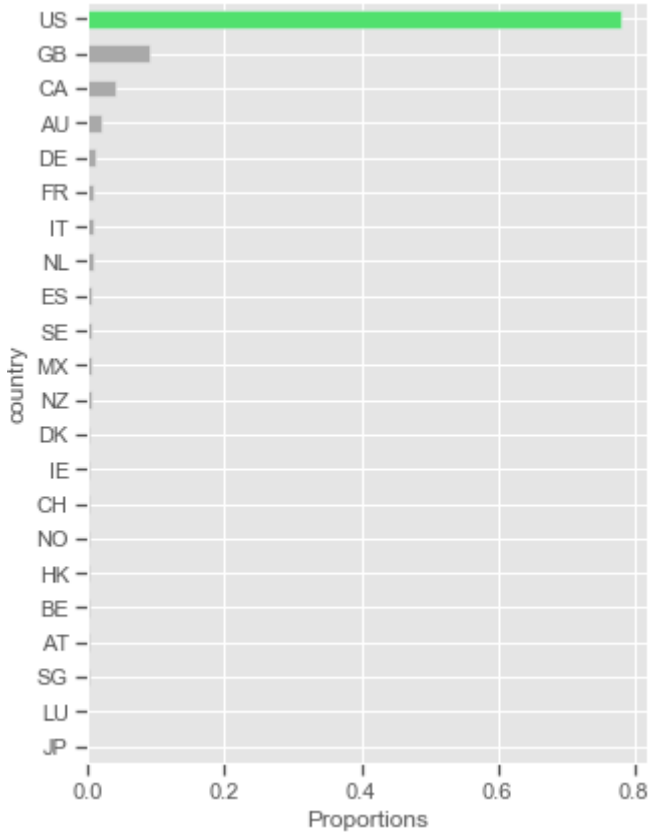

FIGURE 7 – Proportion des pays d'origine

Outre la proportion, le taux de succès y semble aussi supérieur comme on peut le voir sur la FIGURE 8. Le seul pays qui dépasse le taux de succès des Etats-Unis est Hong-Kong. Le Royaume-Uni est le deuxième pays d'origine le plus important, et le troisième en taux de succès. Ce trio de pays anglophones[10] en tête du classement, nous laisse suggérer qu'avoir l'anglais pour langue maternelle, facilite l'introduction d'un projet sur Kickstarter et augmente ses chances de réussite.

[10] Hong-Kong a pour langue officielle le cantonais mais également l'anglais depuis 1974.

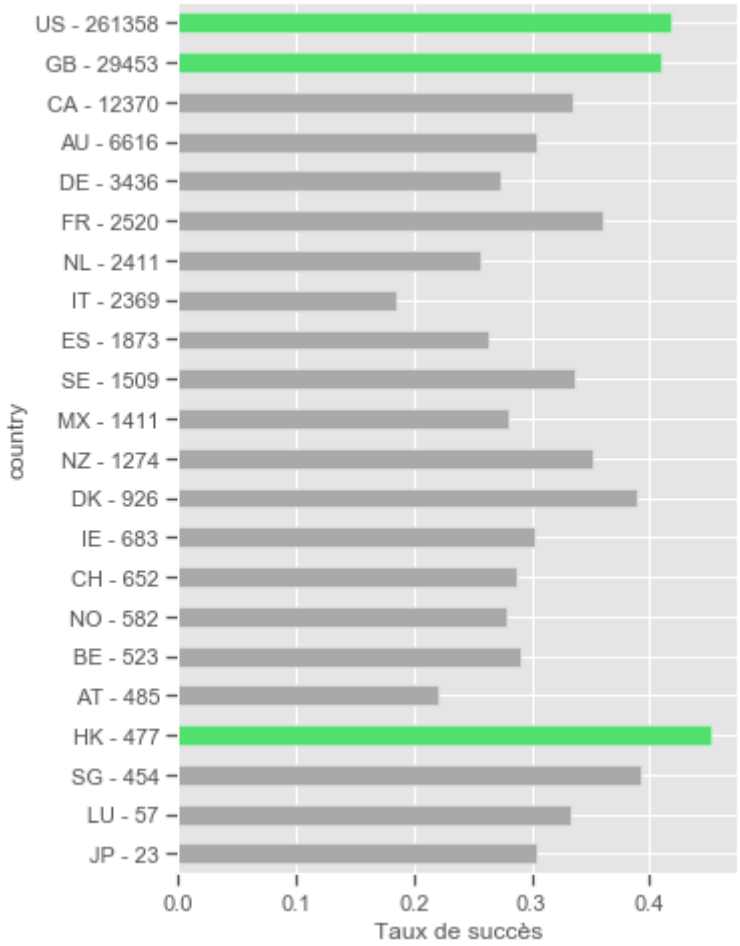

FIGURE 8 – Taux de succès par pays d'origine

3.5. Variables numériques :

3.5.1. `usd_goal_real`

Les FIGURE 9 et 10 nous indiquent que les projets qui réussissent ont des objectifs de financement inférieurs aux projets qui échouent. Néanmoins, demander moins de 1000$ ne semble pas porter ses fruits.

La médiane de l'objectif en dollars est de 5500$. Ce chiffre varie en fonction de sa catégorie principale (FIGURE 11). La catégorie qui a les objectifs les plus élevés est *Technology* avec une médiane de 20 000$ contre, par exemple, seulement 2345$ pour la catégorie *Crafts*.

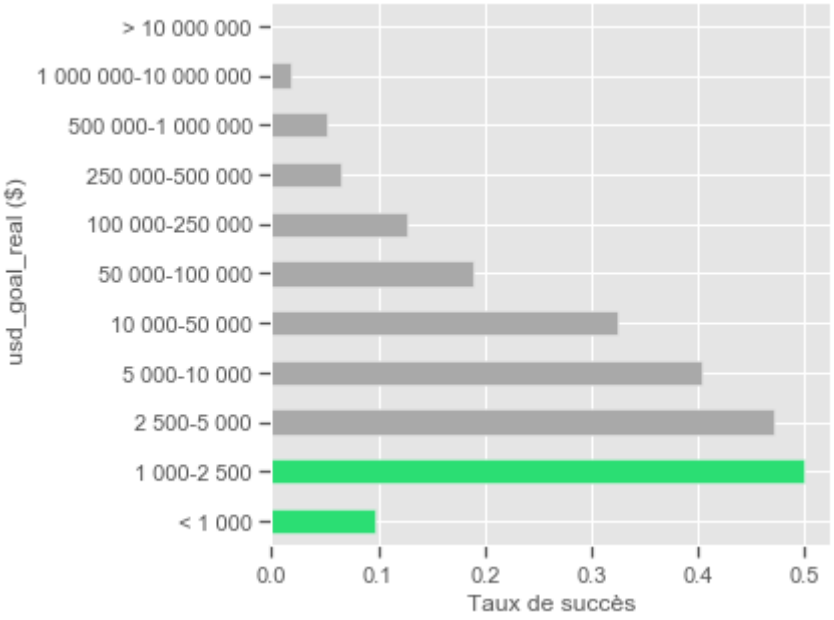

FIGURE 9 – Diagramme à barres : taux de succès par tranche de `usd_goal_real`

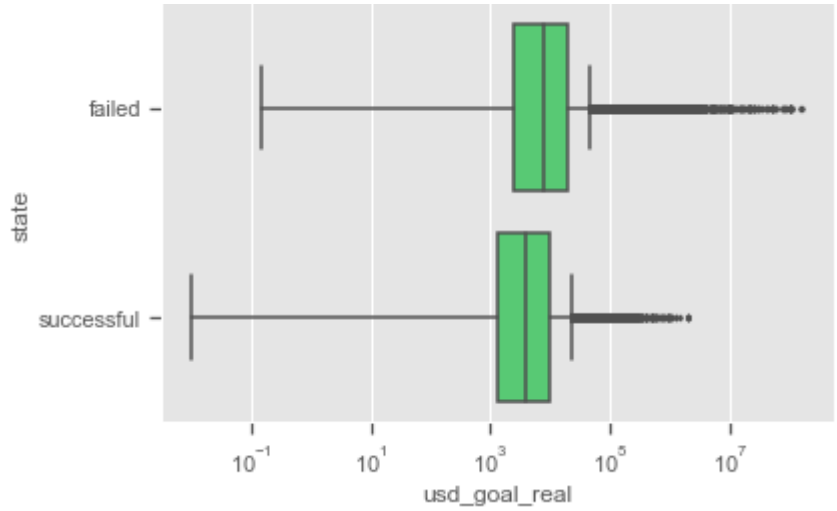

FIGURE 10 – Boîte à moustaches : Distribution de `usd_goal_real` en fonction de l'état du projet (échelle logarithmique)

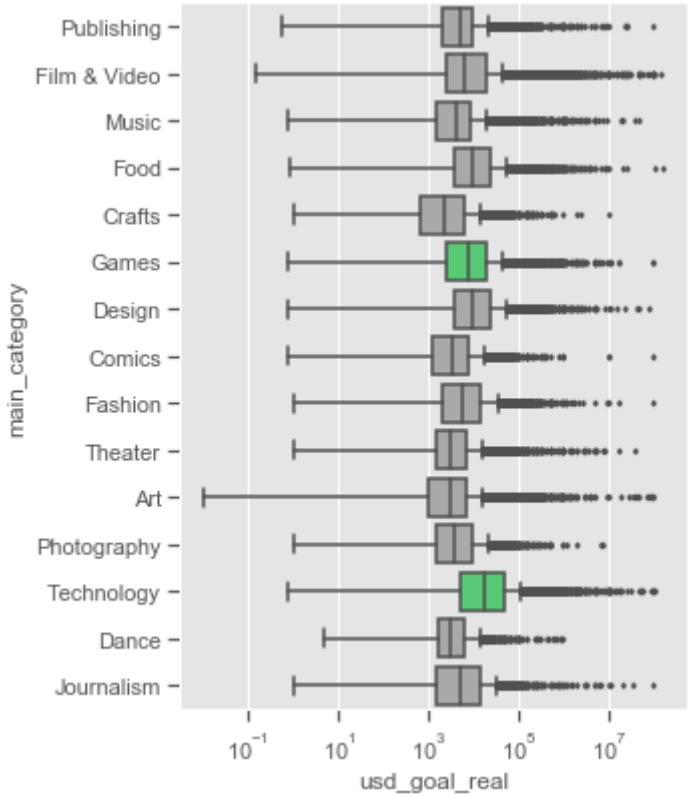

FIGURE 11 – Boîtes à moustaches : Distribution de `usd_goal_real` par `main_category` (échelle logarithmique)

`usd_goal_real` est représentée sur une échelle logarithmique car il y a de forts écarts. Notamment les projets avec les objectifs de financement les plus élevés s'élèvent à 1 x 10^8 $.

3.5.2. `usd_pledged_real`

Contrairement aux objectifs, les promesses sont plus mesurées, la médiane est de 624.7$. De plus, les catégories ayant les objectifs les plus hauts ne sont pas ceux avec les plus hautes promesses. En effet 50% des promesses de financement de la catégorie *Technology* sont inférieurs à 449 $ tandis que ce chiffre s'élève à 2065 $ pour la catégorie *Dance*. (FIGURE 12)

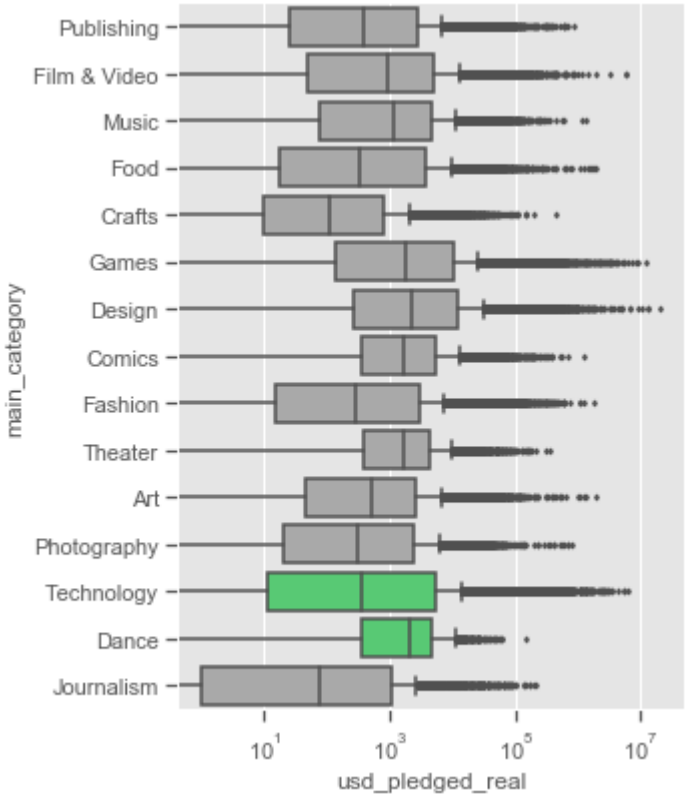

FIGURE 12 - Boîtes à moustaches : Distribution de `usd_pledged_real` par `main_category` (échelle logarithmique)

3.6. Les variables temporelles

3.6.1. Le nombre de jours

Les projets restent en moyennent 33 jours sur la plateforme et au maximum 3 mois. Sur la FIGURE 13 la médiane et le premier quartile des distributions sont confondus et égales à 29. C'est-à-dire qu'un nombre important de projets ont une durée identique : 29 jours.

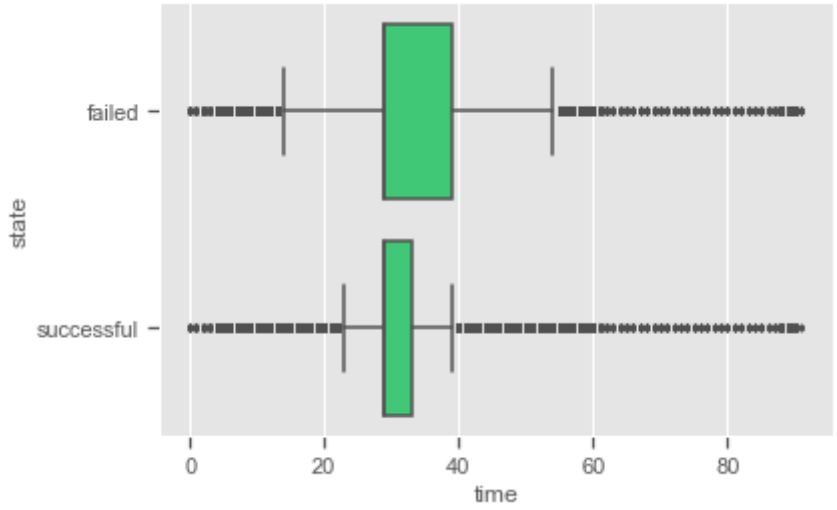

FIGURE 13 – Boîtes à moustaches : Distribution de `time` par `state` (échelle linéaire)

Les projets qui ont une période de financement de moins de 15 jours ont le plus de réussite. Le taux de réussite a tendance à s'effriter avec l'allongement de la période de financement.

Toutefois, la période entre 2 mois et 2 mois et demi atteint environ 45% de taux de succès. (FIGURE 14)

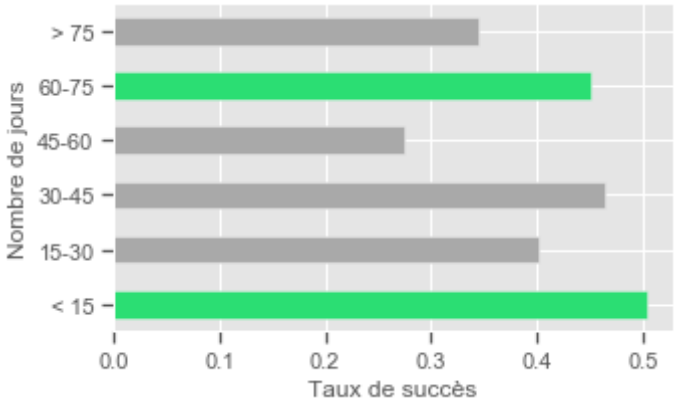

FIGURE 14 – Diagrammes à bars : taux de succès par tranche de `time`

3.6.2. Les jours de la semaine

C'est le mardi qui a le plus de projets lancés. C'est également ce jour que les promesses de dons et le taux de succès sont les plus importantes. A l'inverse, le week-end semble être la période la moins propice pour lancer un projet. (FIGURE 15)

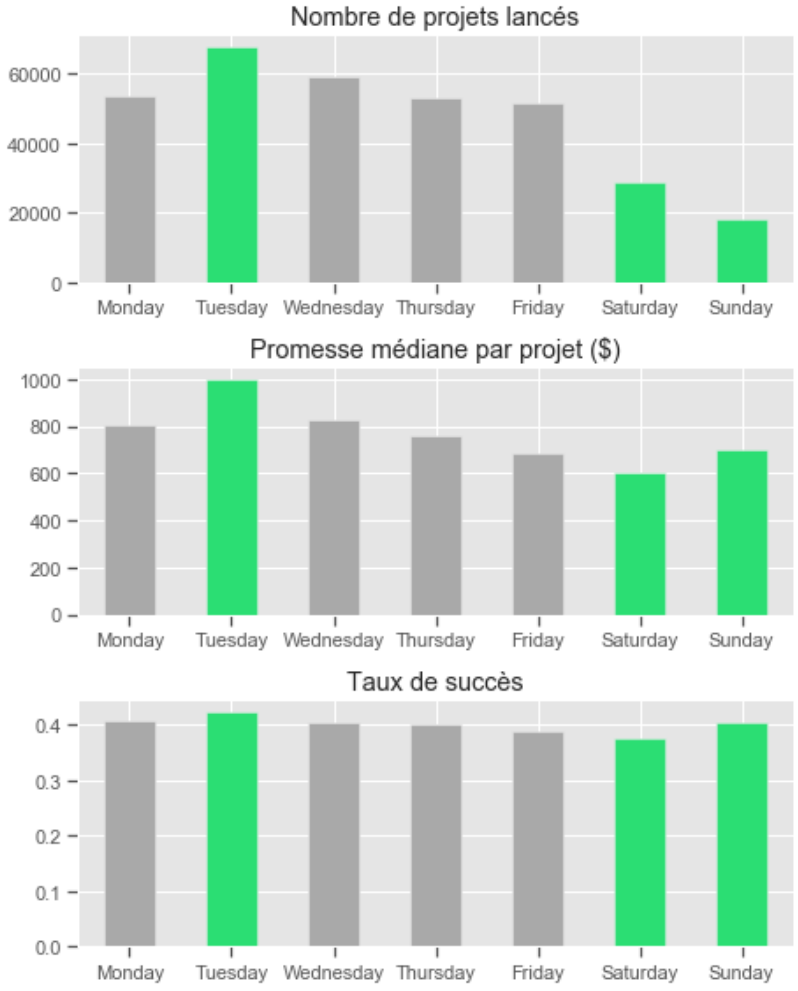

FIGURE 15 – Nombre de projets lancés, montant de la médiane de la variable `usd_pledged_real` et taux de succès par jour de la semaine

3.6.3. Les mois

Juillet est un mois particulier, c'est le moment où le plus de projets sont lancés. Cependant c'est également pendant ce mois que les financements et que le taux de réussite sont au plus bas. Octobre est le moment où les financements sont au plus haut. Enfin, c'est en décembre qu'il y a le moins de projets lancés. (FIGURE 16)

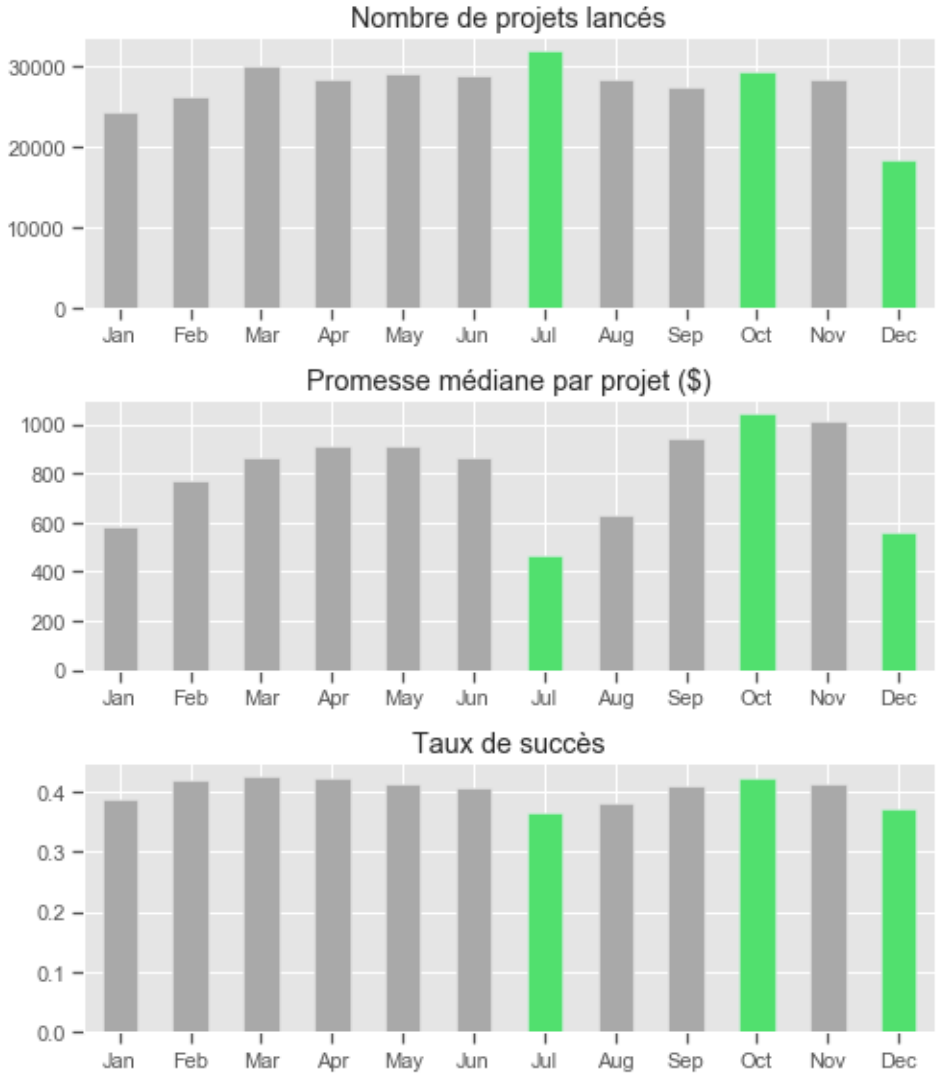

FIGURE 16 – Nombre de projets lancés, montant de la médiane de la variable `usd_pledged_real` et taux de succès par mois

3.6.4. Les heures

Il y a plus de projets qui lancés l'après-midi et en soirée que le matin. Les projets lancés entre 14h à 16h sont ceux qui récoltent le plus d'argent et qui ont le plus de chance de réussir. (FIGURE 17)

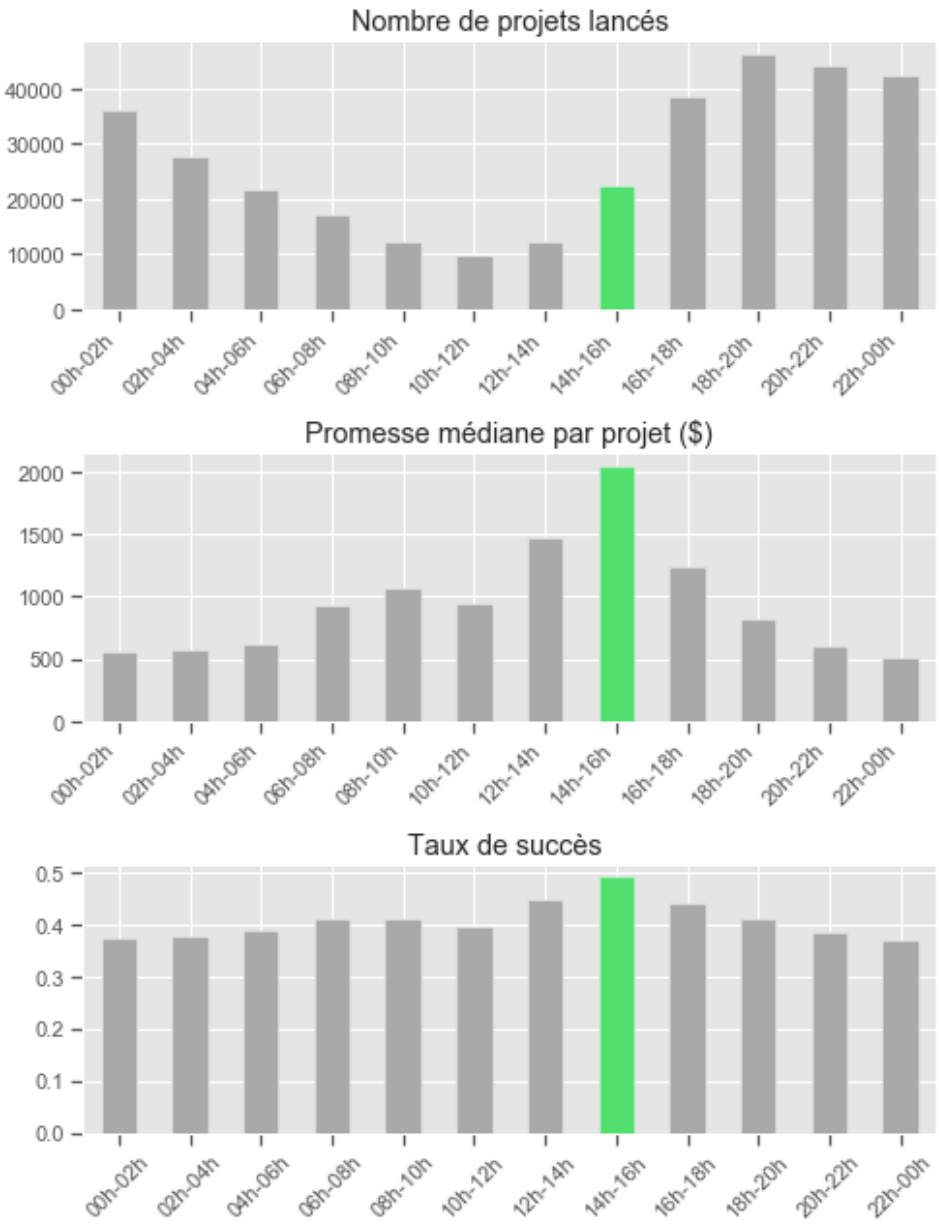

FIGURE 17 – Nombre de projets lancés, montant de la médiane de la variable `usd_pledged_real` et taux de succès par créneau horaire de deux heures

4. Préparation des données

4.1. Mise à l'échelle des variables numériques

La variable `usd_goal_real` prend des valeurs très étendues, il faut donc la transformer en logarithme pour atténuer ces écarts. Ensuite, afin d'utiliser au mieux les différents algorithmes, il faut mettre à l'échelle les variables numériques : `usd_goal_real` et `time`. Pour cela, je choisis d'appliquer une standardisation :

$$z = \frac{x - \mu}{\sigma}$$

Je sélectionne cette transformation car elle est polyvalente et donne de bons résultats avec les différents algorithmes que je vais utiliser.

4.2. Encodage des variables catégorielles

Afin d'utiliser des variables catégorielles dans des algorithmes, il faut les encoder pour qu'elles soient compréhensibles par ces derniers. La transformation *One hot Encoder* consiste à créer une nouvelle colonne pour chaque valeur que peut prendre une variable. Lorsqu'une entrée prend une certaine valeur, on indique le nombre 1 dans cette colonne et 0 dans les autres.

Les variables à transformer sont : `main_category`, `category`, `currency`, `country`, `month`, `day_of_week`, `launch_time`. En appliquant cette méthode, le *dataset* passe de 10 colonnes à 244. L'augmentation du nombre de colonnes est principalement causée par la variable *category* qui a une cardinalité de 159.

Enfin je modifie, la variable indépendante `state` : *failed* devient 0 et *successful* 1.

4.3. Séparation des données

Afin d'éviter *l'overfitting*[11], il faut séparer les données en plusieurs instances. Une pour entrainer les données et une autre pour tester les algorithmes. Cette méthode permet de tester les modèles sur des données qu'ils n'ont jamais rencontrées. Ces instances ont pour nom : `X_train`, `y_train`, `X_test`, `y_test`. 75% des données sont utilisées pour entraîner les algorithmes et 25% pour les tester.

5. Modèles

La librairie *scikit-le*arn est très utile pour créer des modélisations sur *Python*. Elle a l'avantage d'avoir un fonctionnement identique sur la plupart des algorithmes.

Le jeu de données est relativement volumineux : 331462 lignes et 243 colonnes. Afin d'accélérer le temps d'exécution de certains algorithmes, il existe certaines méthodes.

5.1. Méthodes pour réduire le temps d'exécution

5.1.1. Approximation par noyau

La fonction `Nystroem()` est une méthode de *low-rank approximation* du noyau. Pour cela, elle utilise un sous-ensemble du jeu de données. La fonction de noyau retenu est le noyau de base radiale ou *RBF*[12]. En utilisant cette méthode le nombre de variables passe de 243 à 100.[13]

5.1.2. Stochastic Gradient Descent

La descente du gradient est une méthode qui permet de trouver les paramètres optimaux d'une fonction coût. Cependant cette méthode peut être gourmande en capacité de calcul quand le jeu de données est important. Pour remédier à ce problème, il existe une autre variante : la descente du gradient stochastique. Cette dernière va utiliser un sous-ensemble[14] du *dataset* au lieu de sa totalité à chaque étape des calculs. Cette méthode peut être utilisée en combinaison avec la

[11] Sur-interprétation
[12] *Radial Basis Function*
[13] Valeur par défaut
[14] Cela peut-être une observation ou groupe de quelques observations

régression logistique ou le *Support Vector Classifier* grâce à la méthode `SGDClassifier()` de *Scikit-learn*.

5.2. Les modèles

5.2.1. Le choix des hyperparamètres

Certains algorithmes peuvent être configurés avec des hyperparamètres. Ces paramètres permettent d'optimiser les performances d'un algorithme. Deux fonctions sont disponibles dans `sklearn` pour trouver les meilleurs paramètres : `GridSearchCV()` et `RandomizedSearchCV()`. Ces méthodes vont tester des combinaisons de paramètres sur un *cross validation set* afin d'en extraire la combinaison qui donne le meilleur résultat. Ce processus peut être long mais grâce à la combinaison de l'approximation par noyau et du *Stochastic Gradient Descent*, le temps de traitement peut être réduit jusqu'à un facteur dix[15].

5.2.2. Régression logistique

L'implémentation d'une régression logistique peut se faire en utilisant la fonction `SGDClassifier()` de `sklearn`. Il suffit d'attribuer la valeur `log` au paramètre `loss`.

Paramètres fixes :

- `penalty: 'elasticnet'`
- `class_weight: 'balanced'`

Paramètre à optimiser grâce à la fonction `GridSearchCV()` :

- `alpha: np.logspace(-6, 1, 6)`

Meilleur paramètre :

- `alpha: 2.51 * 10-5`

`accuracy_score` de la régression logistique : **65.1%**

5.2.3. Support Vector Classifier

L'implémentation du SVC avec `sklearn` est très proche de la régression logistique, il suffit de changer le paramètre de la fonction coût (`loss`) par `hinge` et d'ajuster quelques paramètres.

Paramètres fixes :

- `penalty: 'elasticnet'`
- `class_weight: 'balanced'`

Paramètres à optimiser grâce à la fonction `GridSearchCV()` :

- `alpha: np.logspace(-6, 1, 6)`
- `l1_ratio : [0, 0.05, 0.1, 0.2, 0.5, 0.8, 0.9, 0.95, 1]`

Meilleurs paramètres :

- `l1_ratio : 1`
- `alpha : 1 x` 10^{-6}

[15] Gain de temps sur le *Support Vector Classifier*

Evaluation du modèle :

`accuracy_score` du *Support Vector Classifier* : **64.38%**

5.2.4. Stochastic Gradient Boosting

La librairie `sklearn.ensemble` propose la fonction `GradientBoostingClassifier()`. Cette fois ci, afin d'optimiser les hyperparamètres, j'opte pour la méthode `RandomizedSearchCV()`. Cette dernière ne teste pas tous les paramètres mais une combinaison aléatoire de certains d'entre eux. Ce choix me permet d'accélérer le temps d'exécution de la fonction et de tester plus de paramètres de façon simultanée. Contrairement aux deux algorithmes précédents, je n'entraîne pas le modèle sur les données modifiées par `Nystroem()`. En effet, cela me permet d'utiliser la méthode `feature_importances` qui n'a de sens que si les données ne sont pas modifiées. De plus l'algorithme est plus performant sur les données originales.

Paramètres fixes :

- `Learning rate : 0.1`

Paramètres à optimiser grâce à la fonction `RandomizedSearchCV()`:

- `max_depth:[6,7,8,9,10]`
- `min_samples_leaf : [150, 200, 250, 300]`
- `min_samples_split: [1250, 1500, 1750, 2000]`
- `n_estimators: [80, 90, 100, 110]`
- `subsample: [0.6,0.7,0.8]`

Meilleurs paramètres :

- `max_depth: 10`
- `min_samples_leaf : 200`
- `min_samples_split: 1500`
- `n_estimators: 110`
- `subsample: 0.7`

Evaluation du modèle :

- `accuracy_score` du *Stochastic Gradient Boosting* : **68.65%**

6. Evaluation du meilleur modèle : *Stochastic Gradient Boosting*

6.1. Performances

Voici différents indicateurs permettant d'évaluer le modèle :

- Taux de précision : 0.65
- Taux de rappel : 0.49
- F1-score : 0.56

Les scores et la matrice de confusion nous indiquent que le modèle est bien plus performant pour évaluer les projets qui vont échouer. En effet, le taux de rappel de 0.49 nous indique que parmi les projets qui ont réussi plus de la moitié ont été défini comme un échec.

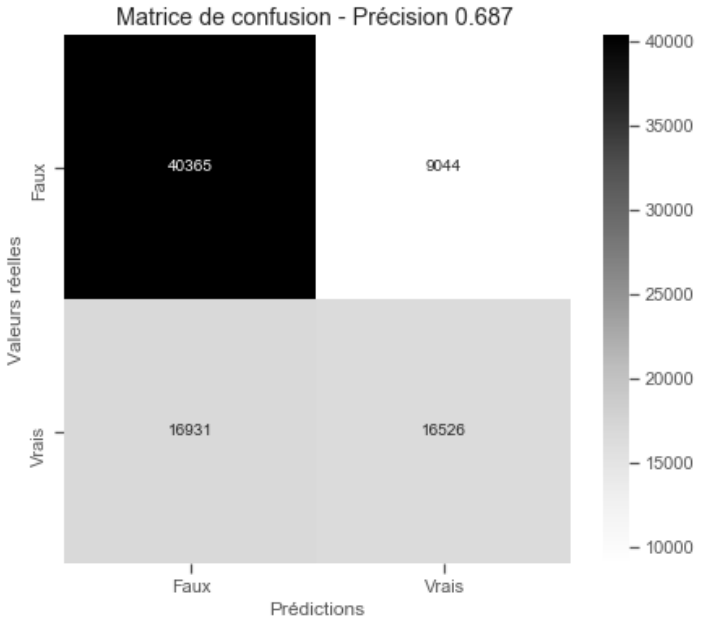

FIGURE 18 – Matrice de confusion

6.2. Importance des variables

Les deux variables numériques `usd_goal_real` et `time`, sont les deux plus importantes du modèle (FIGURE 17). Elles sont suivies par des variables issues de `category` et `main_category`. On peut noter la catégorie *Tabletop Games* à la 3e place. Pour rappel cette dernière est caractérisée par un taux de réussite élevé. A l'inverse les catégories *Apps* et *Web* qui ont un taux de réussite très bas se situent à la 5 et 10e place. En 12e position, on constate la présence d'une variable temporelle : `time_14h_16h`.

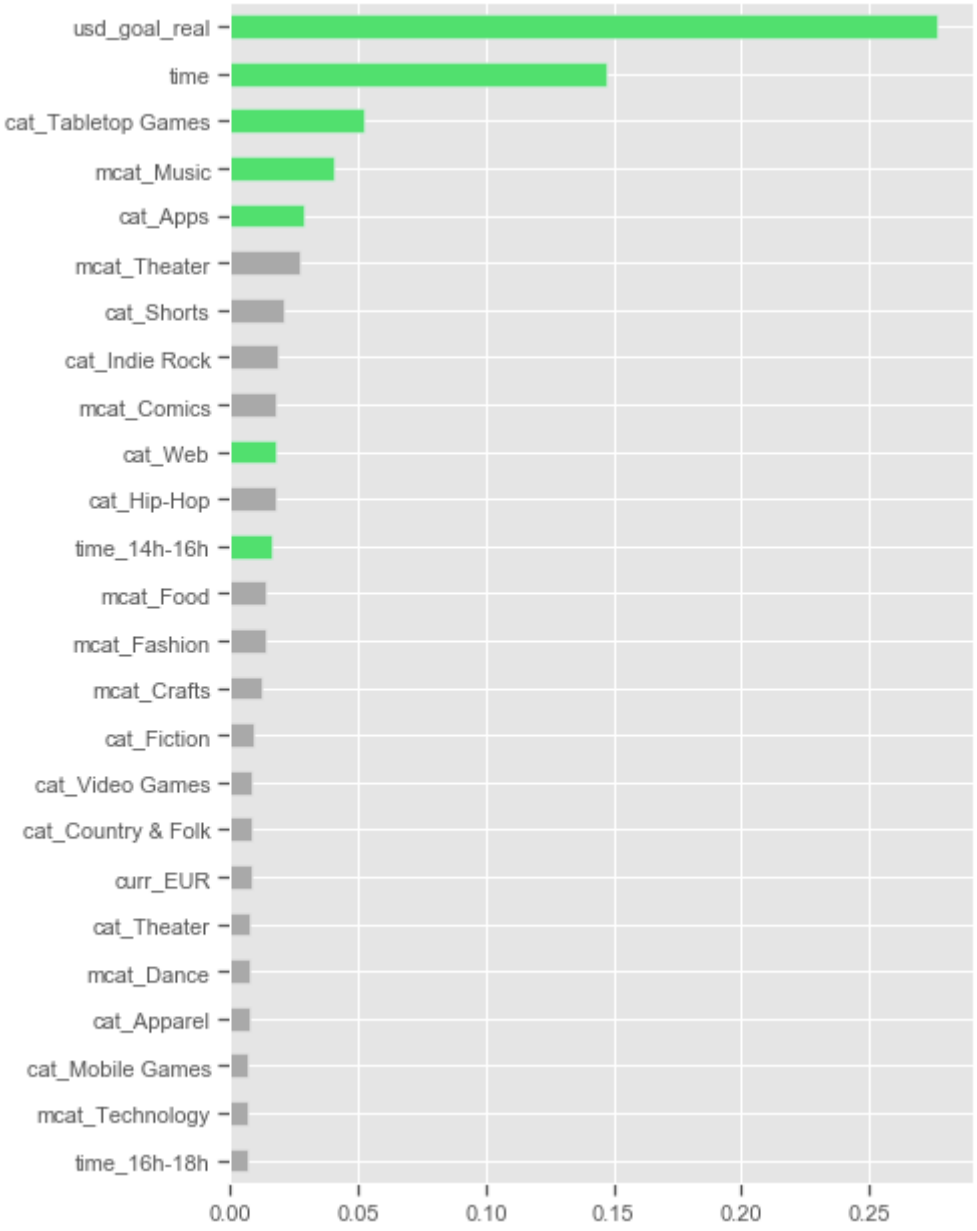

FIGURE 19 – Diagramme à barres : l'importance des variables utilisées par le modèle *Stochastic Gradient Boosting* triées par ordre décroissant

7. Conclusion

En combinant l'analyse exploratoire et l'analyse des variables du modèle *Stochastic Gradient Boosting*, je peux émettre certaines recommandations.

Recommandations importantes :

- L'objectif de financement ne doit pas être trop élevé.
- La durée entre le lancement et la date butoir doit rester courte
- *Tabletop_games, Music, Comics* sont des catégories à favoriser
- *Apps, Web* sont des catégories à éviter

Recommandations secondaires :

- Les meilleurs moments pour lancer un projet sont : entre 14 et 16 heures, le mardi et en octobre.
- Les pires moments sont : le matin et la nuit, le week-end et en juillet ou en décembre.

Les projets qui nécessitent des financements importants comme la création d'une application ou d'un site web ont peu de chance d'aboutir sur Kickstarter. Au contraire les projets qui ont des objectifs de financement plus modestes comme pour la création d'un jeu de société ou d'une bande-dessinée sont plus adaptés à cette plateforme de *crowdfunding*.

La réussite des catégories comme *Tabletop games* et *Comics*, nous laisse suggérer que les financeurs sont intéressés par des projets tangibles. En effet, à l'issu du projet, ils pourront posséder le jeu ou la bande dessinée. De plus, il se peut que les projets des catégories *Apps* et *Web* ont peu de succès pour les raisons opposées.

La catégorie *Music* est très populaire et à un taux de succès élevé. Il se peut que les artistes réussissent grâce à leur communauté en dehors de Kickstarter. A cet effet, il serait intéressant d'analyser le chemin d'accès des soutiens. Ont-ils exploré la catégorie *Music* ou ont-ils suivi un lien URL ?