Ecole nationale de la statistique
et de l'analyse de l'information

Projet de fin d'études

Recommandation dynamique contextuelle pour le site e-commerce de Louis Vuitton

Etudiantes :
Léa Fortat
Solène Venezia
Oriane Vidamant

Tutrice :
Camille-Sovanneary Gauthier

17 Mars 2021

Table of contents

Remerciements **5**

Introduction **6**

Présentation des données disponibles **9**

1 Des différences de comportement de navigation selon les zones géographiques et le contexte du produit **10**
1.1 Une homogénéité dans les produits disponibles en ligne dans les quatre zones 10
1.2 Des différences de comportement de navigation 12
1.3 L'impact du contexte produit sur la probabilité de clic 14
1.3.1 Les variables de contextes qualitatives 14
1.3.2 Les variables de contextes quantitatives 15
1.3.3 Le pouvoir explicatif du contexte produit 16
1.3.4 Synthèse des caractéristiques mises en évidence 17

2 Les algorithmes de bandits standards à l'épreuve de leurs limites **19**
2.1 La limite des bandits non contextuels 19
2.1.1 3 bandits standards : ϵ-Greedy, Thompson Sampling, Upper Confidence Bound 19
2.1.2 La performance des bandits indépendants limitée 20
2.2 L'apport du contexte aux algorithmes de bandit 22
2.2.1 La construction du contexte 22
2.2.2 Les performances des bandits nettement améliorées 22
2.2.3 Un accroissement des performances au détriment du temps de calcul 23

3 NeuralUCB, un algorithme de recommandation adapté aux contextes complexes **25**
3.1 Une meilleure interprétation du contexte grâce à un algorithme innovant 25
3.2 Des résultats prometteurs pour l'algorithme NeuralUCB mais qui restent à approfondir 26
3.2.1 Une structure de réseau de neurones simple adaptée au contexte des produits 26
3.2.2 NeuralUCB, des résultats prometteurs mais limités par les temps de calcul . 27
3.3 La capacité d'adaptation de NeuralUCB grâce à la malléabilité des réseaux de neurones 28
3.3.1 Les améliorations envisageables de notre architecture 28
3.3.2 La malléabilité des réseaux de neurones permet l'adaptation de NeuralUCB aux contextes complexes 29

Conclusion **30**

Annexes **32**

A Présentation des données **32**

B Répartition des modalités de la variable USAGE **34**

C Moyenne des probabilités de clic selon la variable MADE_IN **35**

D Distribution des prix en dollar des produits disponibles en ligne **36**

E Nuages de points de la probabilité de clic en fonction de la variable de prix zz_ZZ_Price **37**

F Représentation des bandits non contextuels en échelle log-log **38**

G Les bandits non contextuels limités à 4000 itérations **39**

H Représentation des bandits contextuels en échelle log-log **41**

I L'algorithme NeuralUCB **42**

J Les réglages du réseau de neurones pour NeuralUCB **44**

Bibliography **46**

Liste des tableaux

1 Indicateurs sur les probabilités de clic en fonction des zones 9

1.1 V de Cramer et p-valeurs associées au test du khi deux entre chaque variable de contexte et la zone géographique 11
1.2 Répartition des appareils de navigation selon les zones 12
1.3 Moyenne des probabilités de clic selon les zones en fonction de la variable productGroup 14
1.4 P-valeurs du test F de l'ANOVA entre la probabilité de clic et chacune des variables de contexte selon les zones 15
1.5 Coefficients de corrélation et tests d'indépendance entre la probabilité de click et le prix d'un produit pour chacune des 4 zones géographiques 16

A.1 Présentation des variables disponibles dans la table de navigation 32
A.2 Présentation des variables disponibles dans la table des produits 33

C.1 Moyenne des probabilités de clic selon les zones en fonction de la variable MADE_IN 35

Table des figures

1.2 Répartition des modalités de la variable *trafficSource* dans les 4 zones géographiques 13

2.1 Pseudo regret cumulé en fonction du nombre d'itérations pour chacune des quatre zones . . . 21
2.2 Articles recommandés par l'algorithme Thompson Sampling . . . 21
2.3 Pseudo regret cumulé en fonction du nombre d'itérations pour chacune des quatre zones . . . 23

3.1 Schéma du réseau de neurones . . . 26
3.2 Perte des échantillons d'apprentissage et de validation au cours des epochs, pour les États-Unis . . . 27
3.3 Pseudo regret cumulé de l'algorithme NeuralUCB en fonction du nombre d'itérations pour la zone des USA . . . 27
3.4 Résultats obtenus à partir de la base de données Mnist avec NeuralUCB et d'autres algorithmes de recommandation . . . 27
3.5 Perte et accuracy des échantillons d'apprentissage et de validation au cours des epochs avec l'optimizer *Adam*, pour les États-Unis . . . 28

B.2 Répartition des modalités de la variable USAGE dans les 4 zones géographiques . . . 34

D.1 Distribution des prix en dollar des produits disponibles en ligne selon les zones géographiques . . . 36

E.2 Nuages de points de la probabilité de clic en fonction de la variable de prix zz_ZZ_Price dans chacune des 4 zones géographiques . . . 37

F.1 Pseudo regret cumulé en fonction du nombre d'itérations pour chacune des quatre zones . . . 38

G.1 Regret cumulé en fonction du nombre d'itérations pour chacune des quatre zones . . . 39

H.1 Regret cumulé en fonction du nombre d'itérations pour chacune des quatre zones . . . 41

I.1 Algorithme principal de NeuralUCB . . . 42
I.2 Schéma du réseau de neurones . . . 43
I.3 Algorithme de descente de gradient de NeuralUCB . . . 43

J.1 Performances du réseau de neurones avec 20 neurones (graphiques supérieurs) et 50 neurones (graphiques inférieurs) sur la couche cachée, pour les États-Unis . . . 44

Remerciements

Nous tenions à remercier Camille Sovanneary Gauthier pour l'encadrement et le suivi de ce projet grâce auquel nous avons pu développer nos compétences autour des systèmes de recommandation. Nous la remercions également pour les conseils et l'aide qu'elle a su nous apporter tout au long du projet.

Nous remercions également Arthur Katossky qui a veillé à la bonne organisation générale de ce projet de fin d'études.

Introduction

Ces dernières années, l'utilisation des systèmes de recommandation est devenue de plus en plus courante, jusqu'à devenir presque omni-présente lors des navigations sur le web. Sur les plateformes de streaming comme Netflix, les recommandations se basent sur l'historique de visionnage de l'utilisateur mais égalemement sur les préférences qu'il exprime à travers des notes. De nombreux secteurs ont recours à ce type de système et le e-commerce ne fait pas exception puisque leur utilisation permet notamment l'amélioration de l'expérience utilisateur mais aussi l'optimisation des performances (nombre de clics, nombre d'achats, panier moyen etc).

Pour les sites de e-commerces, plusieurs méthodes existent à ce jour pour mettre en place ces systèmes. En particulier, les algorithmes de bandits offrent l'avantage d'un juste équilibre entre exploitation (recommander des produits qui ont très bien fonctionné jusqu'à présent) et exploration (recommander des produits qui ont été peu recommandés jusqu'à présent), ce qui est particulièrement adapté dans un contexte où les changements d'utilisateurs sont fréquents. Ils permettent ainsi de présenter à l'utilisateur à la fois des produits qui sont fortement susceptibles de l'intéresser, mais également des produits nouveaux ou peu populaires.

Chez Louis Vuitton, des systèmes de recommandation sont déjà mis en place. Ces derniers consistent notamment à proposer le produit le plus vus ou le plus acheté, ou peuvent être basé sur la similarité visuelle des produits déjà consultés. D'autre part, la recommandation peut aussi être construite à l'aide de modèles de scoring qui ont permis d'identifier les produits pour lesquels les clients sont les plus appétents. Cependant, la recommandation n'intègre pas encore à ce jour l'utilisation de bandits contextuels. L'intégration du contexte (produit ou utilisateur) permet de mieux personnaliser les recommandations. Ce type d'approche est particulièrement intéressant pour la recommandation de nouveaux produits puisque ces derniers, grâce à l'analyse de leur contexte, peuvent être recommandés même s'ils n'ont jamais été vus et cliqués au préalable. Ce projet de fin d'études a pour objectif d'évaluer les approches de recommandations contextuelles qui peuvent être utilisées sur le site de Louis Vuitton. Plus précisément, nous nous sommes posées la problématique suivante :

Pouvons nous mettre en place un système de recommandation dynamique pour le site e-commerce de Louis Vuitton basé sur le contexte ?

Pour répondre à cette problématique nous utiliserons les données des produits Louis Vuitton ainsi que les historiques de navigation sur la période allant du 23 au 27 novembre 2020. Notre étude débutera par une analyse des différents comportements de navigation et de l'impact du contexte produit sur les probabilités de clic. Par la suite, nous démontrerons l'intérêt de l'utilisation du contexte dans la recommandation grâce à l'implémentation de méthodes simples de bandits non contextuels et contextuels. Pour finir, nous mettrons en place une technique de recommandation contextuelle plus élaborée et novatrice.

Présentation des données

Algorithme de bandit : cadre et notations

Comme énoncé précédemment, les algorithmes de bandit sont particulièrement adaptés à la recommandation puisqu'ils mettent en place un compromis entre exploration et exploitation. En particulier, à chaque tour t, $t \in \{1, ..., T\}$, un agent choisit de jouer un bras parmi les K bras disponibles et une récompense, ou un *reward*, est observée. Selon les applications, la récompense peut être de différentes natures : une note donnée à un film, un temps d'écoute d'une chanson, un clic sur un produit, etc. Le but de l'agent est alors de maximiser la récompense cumulée à la fin des T tours afin de trouver le bras optimal. Le compromis entre exploration et exploitation intervient dans l'élaboration de la stratégie à adopter puisque l'agent alterne entre le choix d'un bras qui a bien marché jusque là et le choix d'un bras qui a été peu choisi jusque là. En particulier, la performance de l'agent ou l'intérêt de la recommandation est déterminée à partir de la notion de *regret*. Celui-ci se définit comme la différence entre la meilleure récompense possible et la récompense réellement obtenue par l'algorithme. L'objectif des algorithmes de bandit est alors de minimiser ce regret.

Ici, à chaque tour t, l'agent choisi un produit à recommander à l'utilisateur. Puisque nous nous intéressons aux clics lors des visites sur le site internet, la récompense est alors égale à 1 si l'utilisateur a cliqué sur le produit recommandé. Le but des algorithmes de bandit construits par la suite sera donc de trouver le bon produit à recommander aux utilisateurs à chaque tour afin de maximiser le nombre de clics. Nous travaillons ici avec des données *offline*, nous disposons des probabilités de clics des produits du catalogue Louis Vuitton (définies dans la suite de la présentation des données). Nous utilisons alors le pseudo-regret R_T afin d'évaluer plus clairement nos bandits.

$$R_T = T.r^* - E[\sum_{t=1}^{T} r_t]$$

où r_t est la fonction de récompense et r^* est la meilleure récompense possible (la probabilité de clic la plus élevée).

Description des données disponibles

Dans cette étude, les données mises à disposition sont réparties en deux bases de données. La première contient les données de navigation des utilisateurs du site de Louis Vuitton sur une période allant du 23 au 27 novembre 2020. Les utilisateurs peuvent venir de 4 zones géographiques : le Japon, les USA, la Grande-Bretagne et la Chine. Cette première base est constituée de 6 928 943 observations et de 11 variables. La description de ces variables est disponible à la Table A.1 en annexe.

La deuxième contient les caractéristiques liées aux produits disponibles sur le site internet. Cette deuxième base est constituée de 23941 produits et de 27 variables. La description de ces variables est disponible à la Table A.2 en annexe. Parmi ces 27 variables, 7 ne seront pas utilisées dans la suite. En effet, certaines variables comme l'identifiant de la fonction du produit (FunctionId) ou sa description (presentation_description) possèdent presque exclusivement des valeurs manquantes. D'autres variables ne sont pas conservées car il existe dans la table une autre variable fournissant une information similaire. Par exemple, la variable marketingColorID donne l'identifiant de la couleur du produit présente sur le site alors que macroColorID donne la couleur générale du produit. D'autre part, la variable marketingColorID possède 1200 modalités, contre seulement 32 pour la variable macroColorID, c'est donc cette dernière variable qui est conservée.

Traitement des valeurs manquantes et aberrantes

Seules les valeurs manquantes de la table des produits seront traitées ici. En effet, la table navigation possède des valeurs manquantes uniquement pour la variable indiquant le temps de la session en seconde. Cependant, cette variable ne sera pas utilisée lors de la construction des modèles de bandit.

La première étape dans le traitement des valeurs manquantes de la table des produits est de supprimer au préalable les observations qui contiennent des valeurs manquantes pour la totalité ou la très large majorité des variables : c'est le cas ici pour 281 observations. Ces observations sont supprimées car lorsque l'identifiant sku n'est pas renseigné, l'exploitation des produits devient impossible. Après cette première étape, la table des produits est constituée de 23660 observations et de 20 variables. Elle contient des valeurs manquantes sur plusieurs variables, les traitements appliqués sont les suivants :

— variables collectionId (2338 valeurs manquantes), MADE_IN (18138 valeurs manquantes), SEASON (3998 valeurs manquantes), USAGE (13727 valeurs manquantes), handleId (20159 valeurs manquantes) et sizeId (339 valeurs manquantes) : les valeurs manquantes sont remplacées par une modalité à part entière notée "Missing".

— variable macroColorId (341 valeurs manquantes) : Pour chaque produit qui possède une valeur non manquante sur la variable marketingColorId, la valeur manquante de la variable macroColorId est remplacée par le mode de la variable macroColorId des produits qui possèdent la même valeur pour la variable marketingColorId. Pour les autres produits, les valeurs manquantes restantes sont remplacées par une modalité à part entière notée "Missing".

— variable zz_ZZ_Price (6 valeurs manquantes) : pour chaque produit, la valeur manquante est remplacée par la médiane des produits du même type (variable productGroup) et du même groupe de matière (variable materialGroupId)

D'autre part, aucune valeur aberrante ou extrêmes n'ont été identifiées.

Construction de la probabilité de clic d'un produit

Lors de la construction de nos algorithmes de bandit, la probabilité de clic servira à construire l'environnement. Ici, nous considérons que les clics sont indépendants. Soit p_k la probabilité que l'utilisateur clique sur le produit k, alors le fait qu'un utilisateur clique ou non sur un produit suit

une loi $Beta$ de paramètre p_k.

Pour calculer les probabilités de clic de chaque produit, nous utilisons les variables dict_skuidlists et dict_product_click qui sont respectivement les dictionnaires des produits vus et des produits cliqués par l'utilisateur. Chacune des clés des ces dictionnaires est un identifiant produit et la valeur associée est le nombre de vues (respectivement de clics) de ce produit. La première étape est de construire un dictionnaire de vues global. Pour cela, nous regroupons l'ensemble des dictionnaires de vues de chaque utilisateur. Le dictionnaire obtenu a pour clés l'ensemble des produits qui ont été vus au moins une fois, et la valeur associée à chaque clé correspond au nombre de fois où le produit a été vu par l'ensemble des utilisateurs.

Par la suite nous créons un dictionnaire de clics. Afin d'être sûr que nous ne comptabilisons pas les cas où un clic est enregistré sans que le produit ait été vu au préalable (comportement jugé anormal), le dictionnaire de clics a pour clé l'ensemble des clés du dictionnaire de vues. La valeur associée à chaque clé correspond au nombre de fois où le produit a été cliqué par l'ensemble des utilisateurs.

Pour finir, nous créons le dictionnaire de probabilité de clic qui a pour clé l'ensemble des produits du dictionnaire de vues. La valeur associée à chaque clé est égale au nombre de fois où le produit a été cliqué divisé par le nombre de fois où ce produit a été vu. Certaines probabilités sont strictement supérieures à 1, il s'agit de cas où par exemple, le produit a été vu une seule fois mais 2 clics ou plus ont été enregistrés, ce qui peut arriver par exemple lors du rechargement de la page web par l'utilisateur. Nous remplaçons par la valeur 1 les probabilités strictement supérieures à 1.

Les calculs des probabilités de clic ont été fait pour l'ensemble des utilisateurs sans distinction des zones géographiques, mais également pour chacune des quatre zones prises individuellement. D'autre part, les produit présents dans le dictionnaire d'une zone correspondent à des produits qui sont disponibles en ligne dans cette même zone. Les résultats sont les suivants :

Zone	**Nombre de produits pour lesquels une probabilité est calculée**	**Nombre de produits pour lesquels la probabilité est nulle**	**Moyenne des probabilités**	**Quartile 3**	**Max**
Global	14412	5770	0.015	0.015	1
Japon	10225	3788	0.014	0.014	1
Chine	9852	3648	0.014	0.014	1
Grande-Bretagne	10553	4413	0.017	0.017	1
USA	12873	5782	0.016	0.016	1

TABLE 1 – Indicateurs sur les probabilités de clic en fonction des zones

Nous observons dans la table 1 ci-dessus qu'il y a 14412 produits pour lesquels une probabilité de clic est calculé. Cela signifie que 14412 produits ont été vus au moins une fois au cours de la période allant du 23 au 27 novembre 2020, sur toutes les zones géographiques confondues. D'autre part, nous observons que la probabilité de clic, toute zone confondue, est en moyenne de 0.015 et que 75% des produits ont une probabilité de clic inférieure ou égale à 0.011.

Chapitre 1

Des différences de comportement de navigation selon les zones géographiques et le contexte du produit

1.1 Une homogénéité dans les produits disponibles en ligne dans les quatre zones

Avant de mettre en place des méthodes de recommandation de produit, il est intéressant de comprendre quels types de produits sont disponibles en ligne selon les différentes zones géographiques. Les produits disponibles sur le site e-commerce de Louis Vuitton sont-ils identiques selon les zones géographiques ou des sélections sont elles déjà mises en place ?

La variable zz_ZZ_viewOnline permet de préciser pour chaque produit si ce dernier est disponible sur le site internet dans la zone géographique ZZ. Au Japon, 13 984 produits sont visibles sur le site internet, en Chine 15 886, aux USA 14 427 et en Grande-Bretagne 14 673. 11 447 produits sont disponibles simultanément au Japon et en Chine, contre 11 254 aux USA et en Grande-Bretagne. Enfin, seuls 7 958 produits sont disponibles dans les 4 zones géographiques simultanément. Nous observons ainsi qu'il y a déjà quelques différences dans le catalogue de produits entre les 4 zones.

Comment se traduisent ces différences de catalogue en termes de caractéristiques de produit ? Nous comparons alors les caractéristiques des produits disponibles en ligne selon les zones afin d'identifier si les contextes produits sont différents.

Nous prenons l'exemple de la variable USAGE. La répartition des modalités de la variable USAGE parmi les produits disponibles en ligne dans chacune des 4 zones géographiques est donnée par les diagrammes en barre disponibles à l'annexe B.

La figure B.2 représente la répartition des modalités de la variable USAGE des produits disponibles en ligne selon les 4 zones. La modalité "Missing", qui représente plus de 55% des effectifs pour chaque zone n'est pas représentée afin de rendre plus visible la part des autres modalités. Nous observons que la répartition est très similaire entre les 4 zones : la modalité la plus représentée (or "Missing") parmi les produits disponibles sur le site e-commerce de Louis Vuitton dans les 4 zones est la modalité "Casual" suivie de "City Relax". Malgré tout des différences dans les effectifs de

certaines modalités apparaissent. De plus, si nous effectuons un test d'indépendance du khi-deux entre la variable USAGE et la variable de zone géographique, alors la p-valeur du test obtenue est de 1e-41. Puisque cette p-valeur est inférieure à 0.05, l'hypothèse nulle d'indépendance des deux variables est rejetée au niveau 5%. De plus le V de Cramer, qui mesure l'intensité de la liaison entre les deux variables est de 0.04. L'étude de la variable USAGE permet donc d'en déduire une différence significative entre les 4 zones dans les modalités que prennent les produits en ligne pour la variable USAGE. Cependant ces différences restent moindres étant donné la très faible valeur du V de Cramer associé.

L'étude des autres variables de contexte permet d'en ressortir les résultats suivants :

	p-valeur du test du khi-deux	**V de Cramer**
collectionID	10e-198	0.11
MADE_IN	7e-30	0.03
SEASON	10e-212	0.09
handleId	5e-24	0.03
macroColorId	1e-11	0.04
materialGroupId	4e-65	0.05
productGroup	4e-65	0.05
sizeId	1	0.17

TABLE 1.1 – V de Cramer et p-valeurs associées au test du khi deux entre chaque variable de contexte et la zone géographique

La table 1.1, qui présente les valeurs des V de Cramer et des p-valeurs des tests du khi-deux entre chacune des variables de contexte et la zone géographique, amène aux mêmes conclusions. En effet, mis à part pour la variable sizeId dont la liaison avec la zone géographique n'est pas significative, les caractéristiques des produits disponibles en ligne sont significativement différentes selon la zone géographique. Cependant, les faibles valeurs des V de Cramer qui sont obtenues sont le témoin de différences peu prononcées.

Pour finir, nous nous intéressons à la distribution des prix des produits disponibles en ligne dans chacune des 4 zones. Afin de pouvoir comparer les prix entre les différentes zones, les prix donnés par les variables zz_ZZ_Price sont tous convertis en dollars. La distribution des prix sur chacune des zones géographiques est disponible à l'annexe D. La figure D.1 ne permet pas d'en déduire visuellement de différences significatives entre les zones quant au distributions des prix des produits du catalogue. Pour quantifier plus précisément les différences, nous calculons les moyennes des prix en dollar pour les 4 zones : la moyenne est de 3355$ pour le Japon, 3918$ pour la Chine, 2768$ pour les USA et de 3038$ pour la Grande-Bretagne.

Afin d'identifier si les différences entre les moyennes sont significatives, nous procédons à une analyse de la variance (ANOVA) à un facteur. L'ANOVA permet de mesurer si la zone géographique considérée a un effet sur le prix en dollar. En particulier, un test F est réalisé dont l'hypothèse nulle a vérifier est l'égalité des moyennes entre les zones géographiques. La p-valeur du test est de 2e-22. Puisque cette dernière est inférieure à 0.05, nous rejetons l'hypothèse nulle d'égalité des moyennes. Ainsi, les produits disponibles en Chine et au Japon ont en moyenne des prix significativement plus élevés que ceux disponibles aux USA ou en Grande-Bretagne. Ces différences restent tout de même

faibles puisque les distributions du prix en dollar en fonction des zones sont très similaires.

Cette première étude, qui consistait à comparer les caractéristiques du catalogue de produits entre les 4 zones permet de mettre en évidence que les produits mis en ligne sur le site e-commerce de Louis Vuitton sont globalement homogènes entre chaque zone. Louis Vuitton est donc consistant dans la proposition de produits qu'ils font à travers les zones.

1.2 Des différences de comportement de navigation

Nous nous intéressons maintenant aux caractéristiques liées au comportement de navigation des utilisateurs selon les différentes zones. En effet, plusieurs comportements existent pour parcourir le site de Louis Vuitton : un utilisateur peut utiliser son téléphone, son ordinateur, sa tablette, il peut faire des recherches ciblées, utiliser des filtres, etc.

Un utilisateur peut avoir un comportement différents selon qu'il effectue des recherches à partir de son téléphone portable ou de son ordinateur. La Table 1.2 montre la répartition des modalités de la variable *Device* selon les 4 zones.

	Smartphone	Desktop	Tablet
Grande-Bretagne	78.75%	19.64%	1.61 %
États-Unis	81.40%	17.2%	1.4%
Chine	87.85%	11.75%	0.4%
Japon	90.6%	8.25%	1.15%

TABLE 1.2 – Répartition des appareils de navigation selon les zones

Nous observons ainsi que pour l'ensemble des zones, le téléphone est l'appareil le plus utilisé pour accéder au site, puis vient en seconde position l'ordinateur. L'ordinateur est alors plus utilisé en Grande-Bretagne et aux Etats-Unis avec respectivement 19,65% et 17,2% des navigations, contre 8,25% et 11,75% au Japon et en Chine. Nous remarquons également que les appareils de type console ou smart TV ne sont jamais utilisés dans les 4 zones.

La source de trafic est l'origine des visiteurs sur un site internet. L'utilisateur peut venir par exemple à partir d'une recherche directe dans la barre de recherche internet, depuis un autre site ou encore avec accès par email. La figure 1.2 représente la répartition des modalités de la variable *trafficSource* pour chacune des 4 zones géographiques.

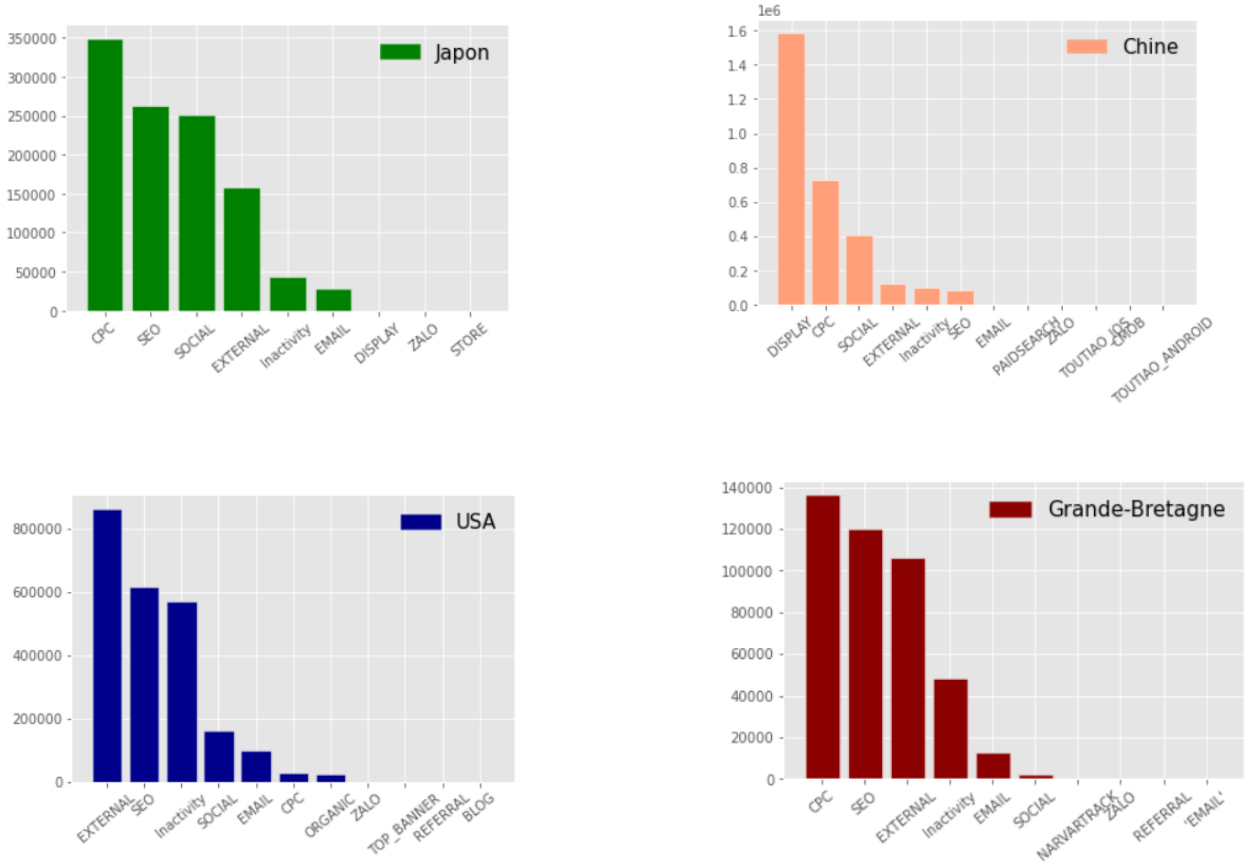


FIGURE 1.2 – Répartition des modalités de la variable *trafficSource* dans les 4 zones géographiques

Nous observons ainsi que la répartition des modalités de la variable trafficSource est très hétérogène entre les 4 zones. En effet, en Chine, la source de trafic la plus représentée est "DISPLAY" alors que cette modalité est quasiment inexistante pour les autres zones. Aux Etats-Unis, la modalité la plus représentée est "EXTERNAL" alors que celle-ci arrive respectivement en 3ème et 4ème position en Grande-Bretagne et au Japon.

Plusieurs types de page d'entrée existent lorsqu'un utilisateur arrive sur un site. En Grande-Bretagne et aux États-Unis, la page d'entrée la plus utilisée, pour près de 50% des navigations, est la page d'accueil du site de Louis Vuitton. En revanche, en Chine et au Japon ces comportements diffèrent puisque les utilisateurs arrivent principalement sur le site par le biais d'une liste de produit. Cela est le cas pour 77,7% des navigations en Chine et pour 51,5% des navigations au Japon.

La variable hasSearch indique si la session de navigation contient ou non une ou plusieurs recherches. Ainsi, en Grande-Bretagne, 21,9% des sessions de navigation contiennent une recherche, contre 20,9% aux États-Unis. Ces deux zones se démarquent de la Chine et du Japon, qui contiennent respectivement 7,9% et 12,2% de recherche.
La variable hasFilterBeenUsed est également une indicatrice qui permet de savoir si le visiteur a utiliser un ou plusieurs filtres. Pour ce comportement de navigation, le Chine se distingue des trois autres zones puisque seulement 2,7% des recherches utilisent un filtre contre 11% pour la Grande-Bretagne, 9,2% pour les États-Unis et 10,8% pour le Japon.

Aux vues de ces différences de comportement de navigation dans les 4 zones étudiées, nous choisissons d'effectuer pour la suite des recommandations différentes selon les zones géographiques.

1.3 L'impact du contexte produit sur la probabilité de clic

Les analyses précédentes ont montré que les comportements de navigation étaient différents selon la zone géographique considérée. En complément, il est intéressant de s'interroger sur l'impact du contexte du produit sur les tendances de navigation en fonction des zones, en particulier sur les probabilités de clic. Un produit d'une certaine origine ou d'un certains groupe aura t-il plus de chance d'être cliqué par l'utilisateur ? Les contextes plus favorables aux clics sont ils les mêmes en fonction des zones géographiques ? Pour finir, les variables de contextes sont elles de bons prédicteurs de la probabilité de clic ?

1.3.1 Les variables de contextes qualitatives

Pour répondre à ces interrogations, nous allons comparer les valeurs des probabilités de clic en fonction des modalités de la variable productGroup et MADE_IN. Nous calculons dans un premier temps les moyennes des probabilités de clic en fonction des modalités que prennent les produits sur ces deux variables. Les moyennes sont calculées pour les navigations prises au global mais également en différenciant les zones. Les résultats sont les suivants pour la variable productGroup :

	Moyenne des probabilités de clic				
Modalité	**Global**	**Japon**	**Chine**	**USA**	**Grande-Bretagne**
Accessoires	0,016	0,014	0,021	0,017	0,016
Autres	0,020	0,012	0,012	0,026	0,017
Jewelry	0,005	0,012	0,005	0,003	0,005
Leather Goods	0,013	0,015	0,012	0,016	0,017
PeakDeret2019	0,050	0,030	0,067	0,055	0,079
PeakEuropa2019	0,112	0,022	0,075	0,191	0,158
Perfume	0,025	0,023	0,017	0,023	0,022
Ready to wear	0,015	0,014	0,013	0,015	0,016
Souliers	0,016	0,012	0,009	0,017	0,019
Watches	0,008	0,012	0,004	0,005	0,005

TABLE 1.3 – Moyenne des probabilités de clic selon les zones en fonction de la variable productGroup

Nous observons sur la table 1.3 représentant les moyennes des probabilités de clic selon les zones en fonction des modalités de la variable productGroup que les parfums ont en moyenne une probabilité de clic plus importante que les bijoux pour toutes les zones considérées. Par exemple, en Chine, les parfums ont en moyenne une probabilité de clic de 0.017 contre 0.005 pour les bijoux. D'autre part, nous observons que les groupes de produits qui ont en moyenne la plus forte probabilité d'être cliqués ne sont pas les mêmes selon les zones. En effet, si les produits du groupe PeakEuropa2019 ont la plus forte probabilité de clic en moyenne en Chine, aux USA, en Grande-Bretagne et sur les 4 zones au global, ce n'est pas le cas au Japon puisque ce sont les produits du groupe PeakDeret2019 qui ont en moyenne la plus forte probabilité d'être cliqués. De plus, la préférence pour les produits PeakEuropa2019 est plus marquée aux USA et en Grande-Bretagne puisque les moyennes des probabilités de clic sont supérieures à 0.15, alors que la probabilité moyenne est de 0.075 en Chine. Les

résultats pour la variable MADE_IN sont disponibles à l'annexe C.

Nous observons sur la table C.1 en Annexe, représentant les moyennes des probabilités de clic selon les zones en fonction des modalités de la variable MADE_IN que les produits fabriqués en France ont en moyenne une plus faible probabilité d'être cliqué pour toutes les zones considérées mis à part la Chine. Par exemple, au Japon, les produits fabriqués en France ont en moyenne une probabilité de clic de 0.007 contre 0.015 pour les produits fabriqués en Italie.

Si des différences de moyennes apparaissent en fonction du contexte produit à travers les zones, il est important de s'interroger sur la significativité de ces différences. Pour cela, nous procédons à une analyse de la variance (ANOVA) à un facteur qui permet de mesurer si une variable de contexte produit a un effet significatif sur la probabilité de clic grâce au test F d'égalité des moyennes parmi toutes les sous-population de produits, où chaque sous-population est caractérisé par une modalité prise sur la variable de contexte. Les p-valeurs des tests F réalisés entre la probabilité de clic et chacune des variables de contexte sont les suivantes :

Variable	**Global**	**Japon**	**Chine**	**USA**	**Grande-Bretagne**
SEASON	6e-10	2e-38	6e-24	0.00004	2e-08
USAGE	7e-08	0.004	6e-7	1e-13	0.038
materialGroupId	4e-17	0.011	2e-30	1e-25	2e-35
productGroup	4e-17	0.011	2e-30	1e-25	2e-35
macroColorId	0.007	0.614	0.004	0.00001	0.154
collectionId	0.084	1e-07	0.837	0.268	0.968
MADE_IN	0.309	0.137	0.847	0.185	0.176
handleId	0.716	0.486	0.651	0.224	0.093
sizeId	nan	1	1	1	1

TABLE 1.4 – P-valeurs du test F de l'ANOVA entre la probabilité de clic et chacune des variables de contexte selon les zones

Nous observons dans la table 1.4 que les p-valeurs sont inférieures à 0.05 pour toutes les zones considérées pour les variables SEASON, USAGE, materialGroupID et productGroup. Cela signifie que les moyennes des probabilités de clic sont significativement différentes au niveau 5% selon les modalités de ces variables dans toutes les zones, l'effet de ces variables sur la probabilité de clic est donc significatif. De plus, les variables collectionId et macroColorID ont des p-valeurs inférieures à 0.05 seulement pour certaines zones : l'effet de ces variables sur la probabilité de clic n'est donc significatif que dans ces zones. Pour finir, les variables MADE_IN, handleId et sizeId ne possèdent aucune p-valeurs inférieure à 5%, il n'y a pas de différence significative dans les moyennes des probabilités de clic selon les modalités prises sur ces variables. Ces trois variables n'ont pas d'effet significatif sur la probabilité de clic, elles ne seront pas utilisée lors de la construction des bandits contextuels.

1.3.2 Les variables de contextes quantitatives

Nous nous intéressons à présent au lien entre la probabilité de clic et les variables de prix zz_ZZ_Price. Dans un premier temps, pour chacune des 4 zones géographiques, nous traçons les nuages de points de la probabilité de clic en fonction de la variable zz_ZZ_Price. Les nuages sont les disponibles à l'annexe E.

Les quatre nuages de points ne permettent pas d'en déduire une relation apparente, cependant il apparaît que les valeurs hautes des probabilités sont plutôt associées à des valeurs faibles du prix. Pour compléter l'étude des nuages de points, nous nous intéressons tout d'abord au calcul de la covariance afin de quantifier la liaison entre les deux variables et en particulier de mettre en évidence le sens de la liaison. D'autre part nous nous intéressons au calcul de deux coefficients de corrélation à savoir le coefficient de corrélation linéaire de Pearson et le coefficient de corrélation de Spearman. Le coefficient de corrélation de Pearson permet de mesurer l'intensité de la liaison linéaire entre les 2 variables. Le coefficient de Spearman quant à lui permet de mesurer l'intensité de la liaison entre les 2 variables même lorsque cette liaison est non-linéaire monotone. Enfin, nous complétons l'analyse de ces deux coefficients par deux tests d'indépendance d'hypothèse nulle "il n'y a pas de corrélation entre les deux variables" : le test de Pearson et le test de Spearman. Les résultats sont les suivants :

	Japon ja__JP__Price	**Chine zh__CN__Price**	**USA en__US__Price**	**Grande-Bretagne en__GB__Price**
Covariance	- 1509	- 186	-11	- 2
Coefficient de corrélation de Pearson	- 0.023	-0,030	-0,019	-0,004
p-valeur du test d'indépendance de Pearson	0.022	0,003	0,032	0,693
Coefficient de corrélation de Spearman	- 0.058	-0,061	-0,096	-0,164
p-valeur du test d'indépendance de Spearman	4e-09	1e-09	1e-27	1e-64

TABLE 1.5 – Coefficients de corrélation et tests d'indépendance entre la probabilité de click et le prix d'un produit pour chacune des 4 zones géographiques

Nous observons tout d'abord à l'aide de la table 1.5 que les covariances sont toutes strictement inférieures à 0, ce qui indique que dans les 4 zones géographiques, la liaison entre la probabilité de clic et le prix du produit est négative. De plus, les coefficients de corrélations de Pearson et de Spearman sont tous négatifs avec des valeurs faibles en valeur absolue. La p-valeur du test de Pearson est supérieure à 0.05 pour la Grande-Bretagne, l'hypothèse nulle est acceptée, il n'y a donc pas de corrélation linéaire entre la probabilité de clic et le prix dans cette zone. En revanche, la p-valeur du test de Spearman est inférieure à 0.05 pour les 4 zones géographiques, l'hypothèse nulle est rejetée. Tous ces résultats montrent donc qu'il existe une relation monotone négative et significative entre la probabilité de clic et le prix du produit dans les 4 zones, l'intensité de cette relation reste malgré tout faible puisque les coefficients de corrélation associés le sont.

1.3.3 Le pouvoir explicatif du contexte produit

Afin d'évaluer le pouvoir explicatif de l'ensemble des variables de contexte sur la probabilité de clic, nous construisons des modèles de régressions linéaires multiples pour chacune des 4 zones

géographiques. Plusieurs modèles de régressions sont construits avec ajout ou suppression d'une ou plusieurs variables de contexte. Puisque les modèles ne possèdent pas le même nombre de variables, leurs performances sont comparées par le coefficient de détermination R^2 ajusté. Pour la Grande-Bretagne par exemple, le meilleur modèle obtenu, c'est à dire celui dont le R^2 ajusté est le plus élevé, contient les variables SEASON, USAGE, materialGroupId, productGroup et en_GB_Price. Afin d'étudier la corrélation entre la probabilité de clic et l'ensemble de ces variables explicatives, nous utilisons le coefficient de détermination R^2 : plus ce coefficient est proche de 0 et moins les variables explicatives n'ont d'intérêt dans l'explication de la probabilité de clic. Ici, le R^2 est de 0.029, ce qui illustre le faible impact que ces variables explicatives ont sur la probabilité de clic. Les observations sont similaires pour la Chine, le Japon et les USA puisque les R^2 des meilleurs modèles obtenus sont respectivement de 0.042, 0.049 et 0.027.

1.3.4 Synthèse des caractéristiques mises en évidence

Cette première analyse de l'impact du contexte produit sur la probabilité de clic permet d'en conclure dans un premier temps que certains contextes produits sont plus favorables au clic. En Grande-Bretagne par exemple, les montres (modalité "Watches" pour la variable productGroup) sont un exemple de contexte moins favorable puisque leur moyenne des probabilités de clic est plus faible (ici elle est de 0.008). En revanche, les parfums (modalité "Perfume" pour la variable productGroup) sont au contraire un exemple de contexte plus favorable puisque la moyenne des probabilités est de 0.025. D'autre part des prix plus élevés correspondent généralement à de faibles probabilités de clic.

Dans un second temps, les contextes favorables aux clics peuvent être plus marqués dans certaines zones. En effet, le groupe de produit PeakEuropa2019 par exemple est le groupe avec la plus forte probabilité de clic en Chine, aux USA et en Grande-Bretagne, cependant l'intérêt pour les produits de ce groupe est plus marqué aux USA et en Grande-Bretagne car la moyenne des probabilité de clic y est nettement plus importante en comparaison de la Chine.

Dans un troisième temps, les contextes plus favorables peuvent également être différents selon les zones. Par exemple, contrairement à la Chine, aux USA et à la Grande-Bretagne, le groupe de produits avec la probabilité moyenne la plus forte au Japon n'est pas le groupe PeakEuropa2019 mais PeakDeret2019.

D'autre part, il a été démontré que les différences de moyennes sont bien significatives au niveau 5% pour une grande partie des variables de contexte. Ce dernier point confirme donc que le contexte du produit a un effet significatif sur la probabilité de clic. Ces conclusions démontrent ainsi l'intérêt d'intégrer le contexte produit lors de la construction des algorithmes de bandits. C'est ce que nous mettrons en oeuvre par la suite.

Pour finir, la construction de régressions linéaires multiples entre la probabilité de clic et les variables de contextes a permis de mettre en évidence le faible pouvoir explicatif de ces régresseurs. En effet, les performances des modèles se sont avérées peu concluantes puisque les coefficients de détermination R^2 associés étaient faibles et très proche de 0. Ces observations suggèrent ainsi que la probabilité de clic s'explique d'avantage par d'autres facteurs inobservés notamment ceux liés au style du produit (forme, présence d'un logo, monogramme, etc) et aux préférences mêmes de l'utilisateur (appétence pour une couleur en particulier, pour un style de vêtements, etc). Cela met également en évidence que l'utilisation de ce contexte dans la construction des bandits possédera

certainement des limites et ne permettra pas d'atteindre une recommandation optimale et personnalisée à chaque utilisateur.

A la suite de cette étude descriptive qui permet de détecter des caractéristiques favorables au clic, nous allons mettre en place des algorithmes de bandits non contextuels et contextuels basés sur les probabilités de clic et permettant de recommander un produit à un utilisateur d'une zone géographique en particulier.

Chapitre 2

Les algorithmes de bandits standards à l'épreuve de leurs limites

L'étude du contexte réalisée à la partie précédente nous a permis d'analyser la relation entre les caractéristiques du produit et la probabilité qu'un utilisateur clique sur le produit depuis le site e-commerce de Louis Vuitton. Elle nous a permis d'identifier les caractéristiques les plus explicatives de la probabilité de clic mais aussi de voir que même si le contexte du produit a une influence, elle reste malgré tout relative à la zone géographique dans laquelle l'utilisateur se trouve. A l'aune de ces résultats, nous pouvons construire nos algorithmes de recommandation pour le site e-commerce de Louis Vuitton en fonction des zones géographiques (Chine, Japon, USA, Grande-Bretagne).

Nous commencerons par implémenter plusieurs algorithmes standards, tel que ϵ-Greedy, sans contexte, puis nous introduirons le contexte des produits.

2.1 La limite des bandits non contextuels

2.1.1 3 bandits standards : ϵ-Greedy, Thompson Sampling, Upper Confidence Bound

Nous testons 3 algorithmes de bandits standards non contextuels : ϵ-Greedy, Thompson Sampling et Linear UCB. Cela signifie que seules les probabilités de clic sont prises en compte pour construire les bandits.

ϵ-Greedy est un algorithme qui consiste à choisir aléatoirement entre l'exploitation (recommandation de la meilleure action actuelle) et l'exploration (recommandation des autres actions possibles). Dans cet algorithme, le ϵ correspond à la probabilité de choisir l'exploration, afin de permettre d'obtenir le meilleur compromis possible entre l'exploitation et l'exploration. A chaque itération t, l'action proposée est alors définie par :

$$a_t = \begin{cases} \text{Meilleure action actuelle } a_t^* & \text{avec la probabilité } 1-\epsilon \\ \text{Autre action } a & \text{avec la probabilité } \frac{\epsilon}{K} \text{ où K est le nombre de bras} \end{cases}$$

Upper Confidence Bound (UCB) est un algorithme pour lequel le compromis exploration-exploitation réside dans le calcul de la borne supérieure de l'intervalle de confiance de la récompense estimée de la recommandation. A chaque itération t, l'algorithme choisit l'action a_t qui maximise la quantité $\left\{\hat{\mu}_{t-1}(i) + \sqrt{\frac{2\log t}{T_i(t-1)}}\right\}$, où $\hat{\mu}_{t-1}(i)$ est la récompense moyenne de l'action a_t sur les t

itérations précédentes et où $T_i(t-1)$ est le nombre de fois où l'action i a déjà été jouée.
Ainsi, au fil des itérations, l'algorithme passe de l'exploration à l'exploitation. En effet, lorsque l'algorithme a une faible connaissance des actions disponibles et des meilleures actions à prendre, la partie exploration de l'équation, soit $\sqrt{\frac{2\log t}{T_i(t-1)}}$, permet à UCB de rechercher l'ensemble de toutes les actions possibles. En revanche, lorsque l'exploration progresse, de meilleures estimations sont formées pour les récompenses données par chaque action. On peut introduire un paramètre dans le calcul afin d'orienter l'algorithme vers plus d'exploration ou au contraire vers plus d'exploitation.

Thompson Sampling construit un modèle de probabilité à partir de la loi a priori des récompenses. Ici, nous supposons que les récompenses suivent une loi Bernoulli. A chaque itération t, l'action choisie a_t est celle qui maximise la quantité $\{\hat{\mu}_i\}$. $\hat{\mu}_i$ est un échantillon de $Beta(S_{i,t}+1, F_{i,t}+1)$ pour chaque action i, et nous avons respectivement $S_{i,t}$ et $F_{i,t}$, le nombre de succès et le nombre d'échecs après t itérations.

Nous rappelons que pour évaluer chacun de ces algorithmes, nous utilisons le pseudo-regret cumulé suivant :

$$R_T = T.r^* - E[\sum_{t=1}^{T} r_t]$$

où r_t est la récompense à l'itération t (la probabilité de clic de l'article recommandé par l'algorithme à l'itération t) et r^* est la meilleure récompense possible (la probabilité de clic la plus élevée). Nous avions vu dans la Table 1 que pour nos 4 zones d'études $r^* = 1$, mais les probabilités de clic sont très concentrées en dessous de 0.015 [1]. De ce fait, le pseudo-regret risque d'être élevé tant que l'algorithme n'a pas trouvé un des meilleurs articles.

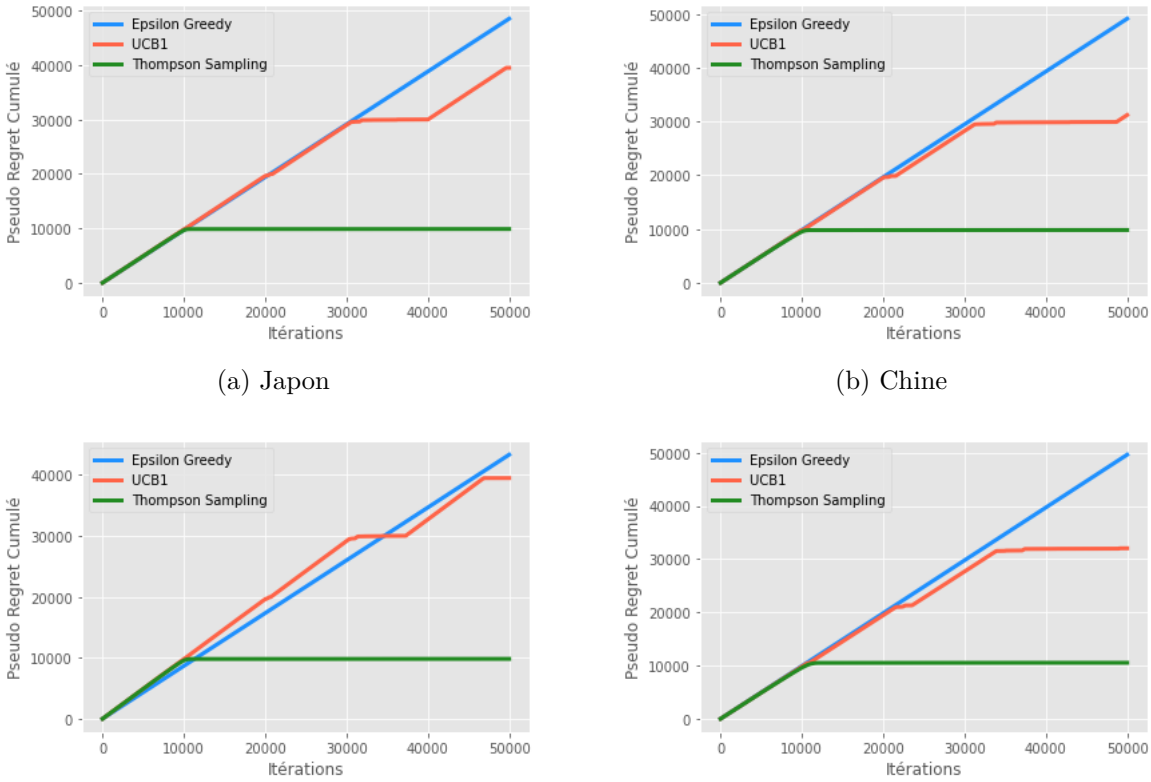
2.1.2 La performance des bandits indépendants limitée

Nous représentons dans les graphiques ci-dessous (Figure 2.1), le pseudo-regret cumulé au fil des itérations. Ici, les graphiques sont représentés en level-level, mais une autre manière de présenter ce pseudo-regret cumulé est une représentation en log-log, comme il est possible de voir en Annexe F.1.
Nous observons qu'un algorithme se détache : Thompson Sampling. On observe que ce bandit explore les articles jusqu'à la 10 000ème itération environ puis semble trouver le meilleur article, puisque que le pseudo-regret est constant sur les itérations restantes. On observe un escalier également pour UCB1, mais cet algorithme semble avoir besoin de plus de temps pour trouver le meilleur article, voir ne semble pas le trouver pour le Japon et la Chine car le pseudo-regret cumulé ne se stabilise pas. Enfin, ϵ-Greedy semble en exploration pure, c'est-à-dire qu'il parcours les articles sans en trouver un suffisamment intéressant pour l'exploiter. Néanmoins, pour les USA, cet algorithme se démarque cependant car sa pente est plus faible, ce qui montre bien qu'il y a un apprentissage pour ϵ-Greedy.

Les bandits standards non contextuels ont une performance faible mais permettent d'identifier tout de même quelques articles intéressants à recommander. En annexe H, nous pouvons voir que la performance des bandits indépendants dépend avant tout du nombre d'itérations. Or, le site de e-commerce de Louis Vuitton répertorie plusieurs milliers d'articles ce qui implique d'appliquer un nombre d'itérations très important pour obtenir une recommandation pertinente. Nous allons

1. La distribution des probabilités de clic est détaillée dans la table 1, dans le préambule.

introduire le contexte dans la section suivante afin d'observer s'il permet d'accroître significativement la performance des algorithmes de recommandation.



(a) Japon (b) Chine

(c) USA (d) Grande-Bretagne

FIGURE 2.1 – Pseudo regret cumulé en fonction du nombre d'itérations pour chacune des quatre zones

La Figure 2.2 permet d'observer les articles recommandés par l'algorithme Thompson Sampling pour chacune des zones géographiques. Ces articles ont une probabilité de clic de 1, soit la probabilité de clic la plus élevée possible. Cela confirme que l'algorithme Thompson Sampling trouve bien le meilleur article à recommander à partir de la 10000ème itération (ou 12000ème).

(a) Japon (b) Chine (c) USA (d) Grande-Bretagne

FIGURE 2.2 – Articles recommandés par l'algorithme Thompson Sampling

2.2 L'apport du contexte aux algorithmes de bandit

Afin d'observer l'apport du contexte, nous reprenons les algorithmes standards utilisés précédemment. Les bandits non contextuels sont également nommés des bandits indépendants, ils cherchent à retrouver la loi des probabilités de clics. Les bandits contextuels, quant à eux, cherchent à trouver la régression qui permet de déterminer les probabilités de clic en fonction du contexte. Ici, le contexte représente l'ensemble des caractéristiques que nous disposons sur les produits. Nous allons voir si l'algorithme ϵ-greedy, utilisé pour les bandits indépendants, permet d'atteindre cet objectif.

2.2.1 La construction du contexte

Nous avons vu dans la section 1.3.1 que plusieurs variables qui composent le contexte sont des variables qualitatives. Cependant, pour construire des algorithmes contextuels, il est nécessaire d'avoir des variables quantitatives.

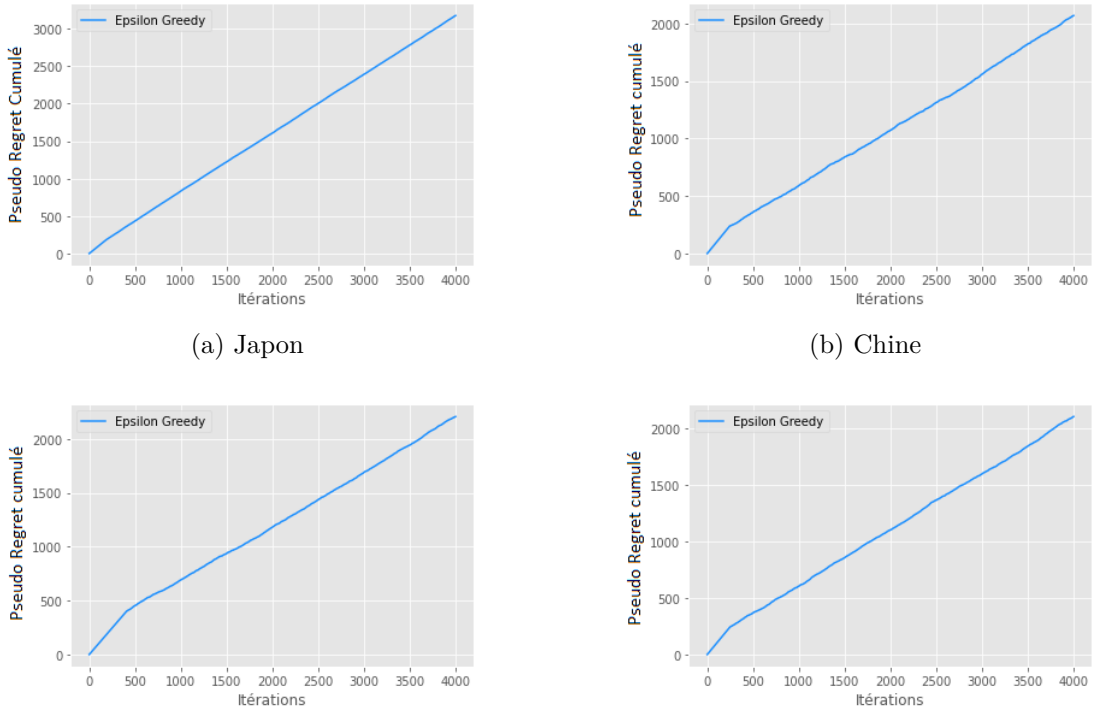
Une procédure de One Hot Encoding a été mise en place afin de transformer ces variables qualitatives en variables binaires (ou *dummies*). Chaque modalité devient alors une variable à part entière, qui prend la valeur 1 si la modalité est présente pour l'article et 0 sinon. Cette méthode conduit évidemment à une augmentation importante du nombre de variables, puisque nous sommes passées de 11 variables à 2783 variables pour la Chine, 2501 pour la Grande-Bretagne, 2603 pour le Japon et 2418 pour les USA. Enfin, une fois les variables qualitatives encodées, l'ensemble des variables sont mises sous la forme d'un tableau (ou *array*).

2.2.2 Les performances des bandits nettement améliorées

La Figure 2.3 représente le pseudo-regret cumulé au fil des itérations pour l'algorithme ϵ-greedy, pour les 4 zones géographiques. En Annexe H.1 se trouve une représentation graphique en échelle log-log.
Lorsque nous ajoutons le contexte, la complexité de l'algorithme évolue et cela a un impact sur le temps de calcul. Afin d'avoir un temps de calcul raisonnable, le nombre d'itérations a été fixé à 4000, pour un nombre de bras compris entre 8000 et 12000.

Dans le but de pouvoir comparer les algorithmes sans et avec contexte, la Figure G.1 en Annexe H représente le pseudo-regret cumulé des bandits non contextuels pour 4000 itérations. Nous observons alors que le pseudo-regret cumulé est bien plus faible qu'avec les algorithmes sans contexte, ce qui montre bien que l'algorithme est en apprentissage.
Les performances de l'algorithme ϵ-greedy avec contexte sont similaires pour chacune des 4 zones géographiques. Par exemple, pour la Chine nous remarquons que ce bandit a un pseudo-regret cumulé d'un peu plus de 2000 à la 4000ème itération. Cela est alors bien meilleur que pour ϵ-greedy sans contexte puisque ce pseudo-regret cumulé était à environ 4000 pour la 4000ème itération. Les performances de l'algorithme ϵ-Greedy sont accrues alors même que nous n'utilisons que le contexte produit.

Nous remarquons néanmoins que même si l'algorithme est en apprentissage, le pseudo-regret cumulé ne se stabilise pour aucune zone. Il serait donc intéressant de pouvoir augmenter le nombre d'itérations afin d'étudier comment évolue le regret cumulé pour un très grand nombre d'itérations.



(a) Japon

(b) Chine

(c) USA

(d) Grande-Bretagne

FIGURE 2.3 – Pseudo regret cumulé en fonction du nombre d'itérations pour chacune des quatre zones

2.2.3 Un accroissement des performances au détriment du temps de calcul

Introduire le contexte dans les algorithmes permet d'accroître significativement les performances, mais cela a un coût. Nous remarquons en effet que le pseudo-regret cumulé ne se stabilise pas, or nous avons dû nous restreindre à 4000 itérations du fait des longs temps de calcul (entre 8h à 11h selon les zones géographiques). Des temps de calculs aussi importants ne sont pas viables pour le site de e-commerce de Louis Vuitton puisque la recommandation fait partie de l'expérience client. Elle doit alors être rapide, voire immédiate, et pertinente.

L'utilisation de serveurs plus puissants peuvent permettre de réduire ces temps de calcul, tel que le cluster de l'ENSAI, nous n'y avons cependant pas eu accès pour ce projet.

De plus, les contextes que nous utilisons peuvent être considérés comme simples, puisqu'ils ne sont pas personnalisés aux utilisateurs, ne comprennent pas d'analyses d'images ou de textes. Or, ajouter de la complexité au contexte avec des images ou de l'analyse de textes pourrait être très intéressant pour une entreprise telle que Louis Vuitton. En effet, les images transmettent des informations visuelles essentielles dans l'industrie de la mode, les descriptions, quant à elles, apportent une spécification supplémentaires par zone géographique. Un même article ne sera pas décrit de la même manière sur le site chinois que sur le site anglais, ce qui introduit des différences dans le contexte des produits communs entre les zones, ce qu'il n'y a pas dans le contexte de ce projet.

En plus des temps de calcul accrût, ces contextes plus complexes rajouteraient un coût d'implémentation important. La question qui se pose alors est : Comment introduire un contexte complexe, aussi personnalisé que possible dans un algorithme de bandit tout en gardant un temps de calcul

raisonnable ?

Des chercheurs se sont penchés sur ce problème en proposant de nouveaux algorithmes qui prennent en compte cette complexité de calcul. C'est le cas de l'algorithme NeuralUCB, présenté dans le chapitre suivant.

Chapitre 3

NeuralUCB, un algorithme de recommandation adapté aux contextes complexes

Prendre en compte le contexte permet d'accroître la performance des algorithmes de recommandation, nous venons de le voir à travers l'exemple du bandit standard ϵ-Greedy. Néanmoins, nous observons que le pseudo-regret est loin d'être aussi faible que nous pouvions l'attendre. La principale raison de cette performance limitée est que les algorithmes de bandits standard ne sont pas toujours adaptés aux contextes.

En effet, certaines situations de recommandation sont plus complexes que d'autres, c'est pourquoi de nouveaux algorithmes ont vu le jour afin de prendre en compte cette complexité. Nous choisissons d'observer les performances de l'algorithme NeuralUCB, présenté par Dongruo Zhou, Lihong Li, Quanquan Gu dans leur article *Neural Contextual Bandits with UCB-based exploration* publié en 2020.

3.1 Une meilleure interprétation du contexte grâce à un algorithme innovant

L'algorithme NeuralUCB utilise l'architecture de l'algorithme standard UCB, il calcule la récompense espérée et la borne supérieure de son intervalle de confiance afin de choisir le meilleur article à recommander. La nouveauté réside dans l'estimation de la récompense. NeuralUCB estime la récompense grâce à un réseau de neurones, qui peut être plus ou moins profond en fonction de la complexité du contexte, ce qui permet d'interpréter de manière optimale les informations contenues dans le contexte. L'algorithme développé par Dongruo Zhou, Lihong Li, Quanquan Gu n'est pas le premier algorithme à utiliser des réseaux de neurones mais leur approche est innovante en termes de contraintes. NeuralUCB a uniquement besoin de vérifier une hypothèse selon laquelle la fonction de récompense est bornée pour fonctionner. NeuralUCB est alors tout à fait adapté à notre situation car nous utilisons les probabilités de clic comme fonction de récompense, ses valeurs sont donc bien évidemment bornées car comprises entre 0 et 1. L'autre innovation apportée par l'algorithme NeuralUCB concerne l'utilisation du réseau de neurones. En effet, contrairement aux autres algorithmes neuronaux, NeuralUCB utilise son réseau de neurones uniquement pour estimer la fonction de récompense. L'intégralité du réseau est utilisé pour interpréter le contexte, ce qui est très intéressant pour Louis Vuitton, qui peut avoir des contextes très complexes. En annexe 2 I nous

présentons le détail de l'algorithme NeuralUCB.

Nous avons fait le choix d'implémenter cet algorithme car NeuralUCB a présenté des résultats prometteurs sur des bases de données réelles, telle que MNIST par exemple [1]. Plusieurs algorithmes avaient été comparés, dont des algorithmes qui n'utilisent aucun réseau neuronal (LinUCB, GLMUCB ou encore KernelUCB) et des algorithmes qui utilisent une combinaison de réseau neuronal et d'autres méthodes, telles que Upper Confidence Bound. L'algorithme NeuralUCB avait alors présenté le regret cumulé le plus faible, ce qui montre les bonnes performances de cet algorithme en pratique.

De plus, choisir un algorithme qui utilise des réseaux de neurones est intéressant car ils peuvent s'adapter à différents types de données (images, textes, etc).

Dongruo Zhou, Lihong Li, Quanquan Gu ont utilisé un réseau de neurones simple pour leur test sur la base de données MNIST, nous reprenons cette architecture simple.

Le schéma ci-contre représente la structure du réseau de neurones utilisé par les trois auteurs. C'est un réseau de neurones linéaire composé de 2 couches, la première comporte 100 neurones et la seconde permet de sortir la probabilité. La fonction d'activation ReLU est utilisée et une descente de gradient stochastique (*SGD*) pour optimiser les paramètres du réseau.
Nous proposons d'utiliser cette même structure en l'adaptant à notre contexte. En effet, le contexte issu de la base de données MNIST est composé de pixels ce qui n'est pas le cas de notre contexte, il est donc vraisemblable que d'autres réglages soient nécessaires.

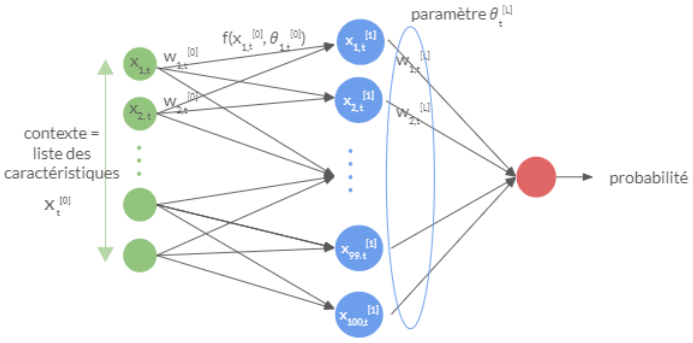


FIGURE 3.1 – Schéma du réseau de neurones

3.2 Des résultats prometteurs pour l'algorithme NeuralUCB mais qui restent à approfondir

3.2.1 Une structure de réseau de neurones simple adaptée au contexte des produits

Nous reprenons alors l'architecture du réseau de neurones décrite ci-dessus, les différentes adaptations réalisées pour notre contexte sont décrites en annexe J. Avec cette architecture plutôt simple, nous avons déjà entre 48 000 et 54 000 paramètres à estimer, en fonction des zones géographiques. Néanmoins, l'algorithme de descente de gradient stochastique (avec un taux d'apprentissage de 0.01) converge rapidement, moins de 50 epochs suffisent et les performances sont bonnes. Nous utilisons deux mesures pour les évaluer : l'accuracy et la perte (calculée à partir de l'erreur quadratique

1. MNIST est une base de données de classification, elle est composée d'images de chiffres et de leur label (de 0 à 9). Les auteurs ont transformé cette base de données en une base de données pour bandits contextuels à 10 bras. Les classes de chiffres sont alors les bras du bandit et les pixels des images le contexte.

moyenne). Nous comparons ces mesures pour l'échantillon d'entraînement des produits du catalogue de la zone et l'échantillon de validation. Nous obtenons les résultats suivants pour les États-Unis (les résultats sont similaires pour les quatre zones).

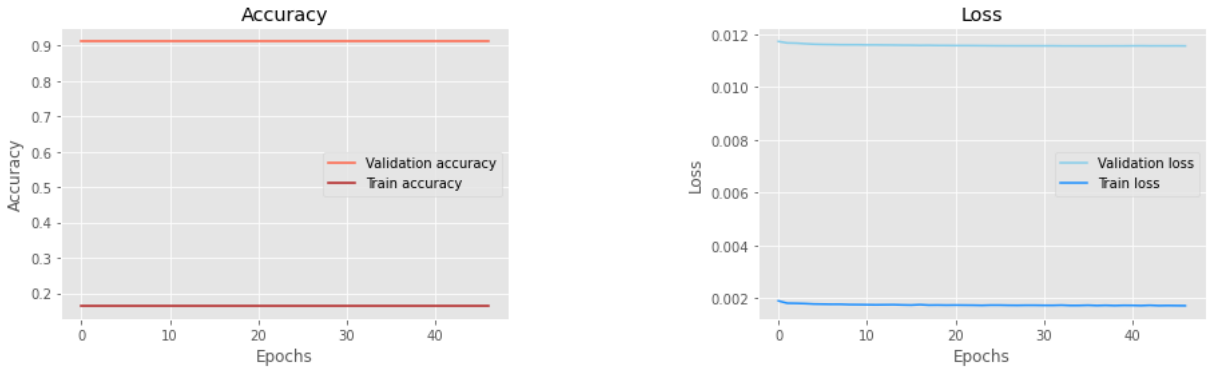


FIGURE 3.2 – Perte des échantillons d'apprentissage et de validation au cours des epochs, pour les États-Unis

Nous observons sur les graphiques ci-dessus la rapide convergence de l'algorithme de descente de gradient stochastique. La perte de l'échantillon de validation est plus élevée que celle de l'échantillon d'apprentissage, néanmoins cette différence est minime (de l'ordre de 0.01), elle ne traduit donc pas un important problème de sur-apprentissage. En revanche, la différence entre l'accuracy de l'échantillon de validation et d'apprentissage est beaucoup plus importante et surprenante. Notre but n'est pas d'obtenir un réseau qui correspond parfaitement au contexte mais un réseau qui offre une bonne prédiction de la probabilité de clic. En ce sens, il est plutôt encourageant d'avoir une accuracy aussi élevée pour l'échantillon de validation.

3.2.2 NeuralUCB, des résultats prometteurs mais limités par les temps de calcul

La Figure 3.3 représente le pseudo-regret cumulé de l'algorithme NeuralUCB au fil des itérations pour les États-Unis. Le nombre d'itérations a été fixé à 500 afin d'avoir un temps de calcul convenable (plus de 12h de calcul).

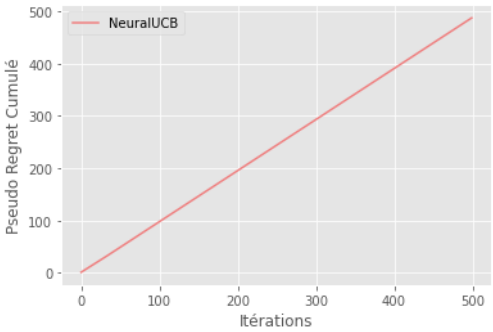


FIGURE 3.3 – Pseudo regret cumulé de l'algorithme NeuralUCB en fonction du nombre d'itérations pour la zone des USA

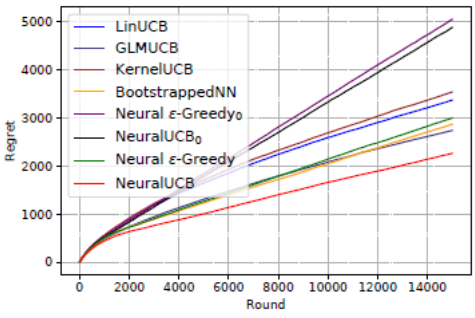


FIGURE 3.4 – Résultats obtenus à partir de la base de données Mnist avec NeuralUCB et d'autres algorithmes de recommandation

Nous observons un pseudo-regret cumulé est linéaire. Néanmoins, Dongruo Zhou,Lihong Li et

Quanquan Gu avaient obtenu des résultats prometteurs sur la base MINST avec 15000 itérations, or nous n'en avons que 500 ici. De plus, nous voyons sur la figure 3.4 que NeuralUCB est linéaire pour les 1500 premières itérations.

Il serait alors intéressant de pouvoir augmenter ce nombre d'itérations afin d'observer l'évolution du pseudo-regret cumulé pour un très grand nombre d'itérations, ou au moins 2000 itérations pour évaluer si notre NeuralUCB prend la même direction que celui construit par les auteurs.

Enfin, améliorer et optimiser l'algorithme pourrait également avoir une influence sur le pseudo-regret cumulé.

3.3 La capacité d'adaptation de NeuralUCB grâce à la malléabilité des réseaux de neurones

Nous avons testé l'algorithme de recommandation NeuralUCB avec une seule architecture neuronale, des améliorations sont alors envisageables. De plus, nous avons adapté notre réseau de neurones à notre contexte, nous pouvons néanmoins imaginer les structures adaptées aux différents contextes que nous pouvons rencontrer chez Louis Vuitton.

3.3.1 Les améliorations envisageables de notre architecture

Nous avons réalisé différents réglages pour notre réseau de neurones, détaillé en annexe J. Nous pouvions également utiliser un autre algorithme d'optimisation des paramètres. La vitesse d'optimisation est importante pour les algorithmes de recommandation, car nous cherchons à produire les recommandations les plus rapides possibles. Modifier l'algorithme d'optimisation peut être un moyen de réduire la vitesse de calcul de la recommandation. Nous proposons par exemple de comparer les résultats obtenus précédemment avec ceux obtenus avec l'algorithme *Adam*.

L'algorithme d'optimisation des paramètres *Adam* (Adaptative Moment Estimation) est un algorithme adaptatif, contrairement à la descente de gradient stochastique, qui utilise un taux d'apprentissage fixe. *Adam* est justement un algorithme qui converge plus rapidement mais la descente de gradient stochastique est meilleure en généralisation. Cela se retrouve dans les performances de notre réseau de neurones, pour les États-Unis, présenté dans la figure 3.5 (nous obtenons des résultats similaires pour les trois autres zones).

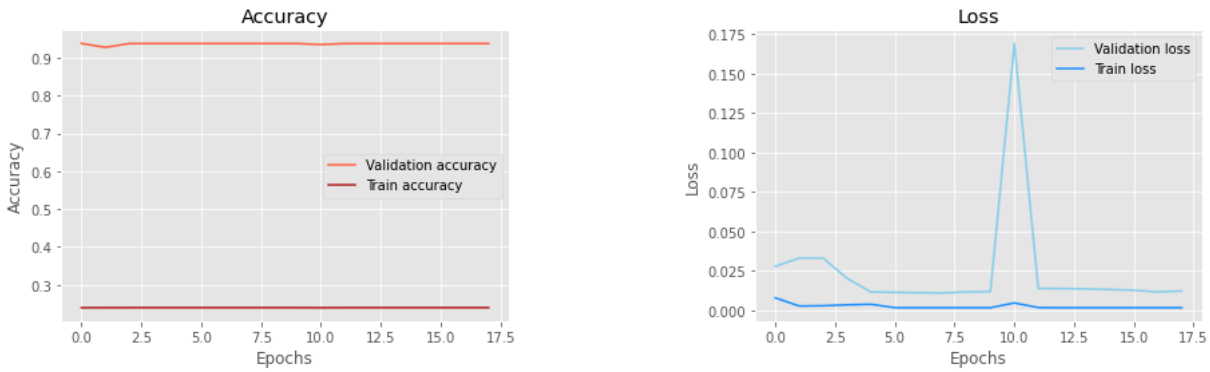


FIGURE 3.5 – Perte et accuracy des échantillons d'apprentissage et de validation au cours des epochs avec l'optimizer *Adam*, pour les États-Unis

En effet, nous observons que moins de 20 epochs suffisent pour la convergence, néanmoins la perte est moins stable que celle avec l'algorithme *SGD*. Pour un contexte plus complexe, l'utilisation de l'algorithme *Adam* peut se révéler intéressante pour réduire les temps de calcul. Zhuang *et alii* ont développé un algorithme, *AdaBelief* [6], qui permet de combiner les avantages des algorithmes adaptatifs comme *Adam*, c'est-à-dire la rapidité de convergence, et des algorithmes de descente de gradient comme *SGD*. Cet algorithme n'est pas encore disponible dans les packages Python tel que *tensorflow* mais il peut être intéressant de le mettre en place pour le site de e-commerce de Louis Vuitton, afin d'obtenir des recommandations pertinentes et rapides.

3.3.2 La malléabilité des réseaux de neurones permet l'adaptation de NeuralUCB aux contextes complexes

Pour des contextes plus complexes, il aurait certainement fallu modifier plus que les réglages du réseau de neurones et l'algorithme d'optimisation des paramètres. C'est dans ce cas, de contextes qui peuvent évoluer, que le choix d'un algorithme neuronal prend tout son sens. La malléabilité des réseaux de neurones permet d'interpréter des contextes très différents, de complexité également différente.

Nous pouvons identifier plusieurs types de contexte pour le site de e-commerce de Louis Vuitton. Tout d'abord, nous n'avons pas pu le mettre en oeuvre, mais il est évident que les images des produits sont des éléments important à prendre en compte dans le contexte. Inclure des images au contexte nécessite de modifier la structure de notre réseau de neurones. Nous pouvons alors utiliser un réseau de neurones convolutionnel, c'est un type de réseau particulièrement adapté aux images. De plus, ce type de réseau est largement utilisé dans l'état de l'art, étudié par Zhang, Yao, Sun dans leur article *Deep Learning based Recommender System : A Survey and New Perpectives* [3], et offre des résultats intéressants.

Un autre élément qui serait intéressant d'intégrer au contexte des produits Louis Vuitton est la description des produits. Cet élément permet d'ajouter de la variabilité entre les produits communs à toutes les zones, car un produit n'est pas décrit de la même façon dans les quatre zones géographiques. Pour intégrer les descriptions, un réseau de neurones récurrents peut être nécessaire. Les réseaux de neurones récurrents permettent de prendre en compte l'ordre des mots. Il existe différent types de réseaux de neurones récurrents (les réseaux de neurones récurrents dits naïf, le *Long Short Term Memory*, le réseau avec mécanisme d'attention, etc). Dans le cas des descriptions, il serait sûrement plus judicieux de partir sur un *LSTM*, de façon à ce que le réseau considère la phrase entière et non pas seulement le mot, pour chaque mot.

Comme nous l'avons vu dans la partie 2.2.3, introduire un contexte à un coût, introduire un contexte complexe a un coût encore plus élevé. En effet, plus le réseau de neurones est complexe plus il va coûter en termes d'implémentation et de réglages. Néanmoins, implémenter ses réseaux de neurones complexes peut permettre d'intégrer toute la richesse des informations possédées par Louis Vuitton sur ses produits et clients, et donc produire des recommandations plus fines, plus dynamiques et enfin, plus personnalisées.

Conclusion

Lors de ce projet de fin d'études, nous avons montré l'intérêt de l'utilisation du contexte dans la recommandation grâce à l'implémentation et la comparaison de méthodes simples de bandits non contextuels et contextuels puis d'une méthode de recommandation adaptée aux contextes complexes. En particulier, nous avons observé que l'introduction du contexte produit dans les algorithmes de bandits permettait de réduire le regret cumulé.

Un système de recommandation dynamique basé sur le contexte pourrait donc être pertinent pour le site e-commerce de Louis Vuitton, cependant ces méthodes disposent d'un coût de développement élevé notamment en terme de temps de calcul.

Pour ce projet, nous étions limitées d'une part par la confidentialité des données mais également par nos capacités de calcul ce qui n'a pas permis de prendre en compte le contexte des utilisateurs. Malgré tout, pour un site tel que celui de Louis Vuitton, il serait tout à fait pertinent et possible de personnaliser les recommandations grâce aux caractéristiques des utilisateurs en intégrant dans les algorithmes d'autres informations telles que l'âge de l'utilisateur, l'historique de ses achats, etc. D'autre part, inclure une analyse des images des produits proposés ainsi qu'une analyse textuelles de leur description fournie, sont également des moyens qui permettraient d'améliorer et de personnaliser les recommandations.

Nous avons également fait le choix de ne recommander qu'un seul produit mais il pourrait être tout aussi intéressant de recommander une liste de produits, afin de proposer plusieurs choix aux utilisateurs et d'augmenter la probabilité d'obtenir un clic. Ce changement dans la recommandation entraîne également une redéfinition de la récompense puisque cette dernière peut être par exemple égale à 1 dès lors qu'un des produits recommandés a été cliqué par l'utilisateur, mais peut également être plus complexe (nécessité d'avoir 2 produits cliqués ou plus, etc)

NeuralUCB est un algorithme innovant, prometteur sur des contextes plus complexes. Cependant, nous n'avons pas réussi à obtenir des résultats satisfaisants. Nos capacités de calculs étaient en effet limitées mais il serait intéressant d'augmenter le nombre d'itérations de l'algorithme pour réellement observer les résultats. Dans cette même optique, d'autres algorithmes auraient pu être testés notamment l'algorithme SquareCB, développé par Dylan J. Foster et Alexander Rakhlin dans leur étude *Beyond UCB : Optimal and Efficient Contextual Bandits with Regression Oracles* [1]. Cet algorithme est la première réduction d'une régression oracle online à des bandits contextuels.

Annexes

Annexe A

Présentation des données

Variable	Description
visitorId	Identifiant contenu dans le cookie
sessionStartTime	Date et heure du début de la session en format timestamp
sessionDurationSeconds	Temps de la session
device	Appareil utilisé pour la navigation
trafficSource	Source du traffic
hasSearch	Variable binaire qui vaut 1 si la session continent une recherche
language	Langage
firstPageTypology	Type de page d'entrée
hasFilterBeenUsed	Variable binaire qui vaut 1 si le visiteur a utilisé un ou plusieurs filtres
dict_skuidlists	Dictionnaire des produits vus pendant la session {skuId : nombre de vues}
dict_product_click	Dictionnaire des produits cliqués pendant la session {skuId : nombre de clics}

Table A.1 – Présentation des variables disponibles dans la table de navigation

Variable	Description	Conservée pour la suite
skuId	Identifiant sku du produit. Il est définit pour un modèle et une taille précise	Oui
ProductId	Identifiant du produit	Oui
collectionId	Identifiant de la collection	Oui
MADE_IN	Pays de fabrication	Oui
SEASON	Saison	Oui
USAGE	Usage	Oui
FunctionId	Identifiant fonction	Non : plus de 99% de valeurs manquantes
handleId	Type de porté	Oui
sizeId	Taille	Oui
itemSkuSize	Taille	Non : conservation de la variable équivalente sizeID
macroColorId	Couleur générale	Oui
marketingColorId	Couleur sur le site	Non : conservation de la variable équivalente macroColorId
materialGroupId	Groupe de la matière	Oui
materialId	Identifiant de la matière	Non : conservation de la variable équivalente materialGroupId
productGroup	Type de produit	Oui
presentation_description	Description générique	Non : plus de 99% de valeurs manquantes
fr_FR_detailedDescription	Description présente sur le site français	Non
fr_FR_commercialName	Nom sur le site français	Oui
fr_FR_image_url	Url de l'image du site	Non
zz_ZZ_viewOnline	Variable binaire qui vaut 1 si le produit est disponible sur le site de la zone ZZ	Oui
zz_ZZ_Price	Prix sur le site de la zone ZZ	Oui

TABLE A.2 – Présentation des variables disponibles dans la table des produits

Annexe B

Répartition des modalités de la variable USAGE

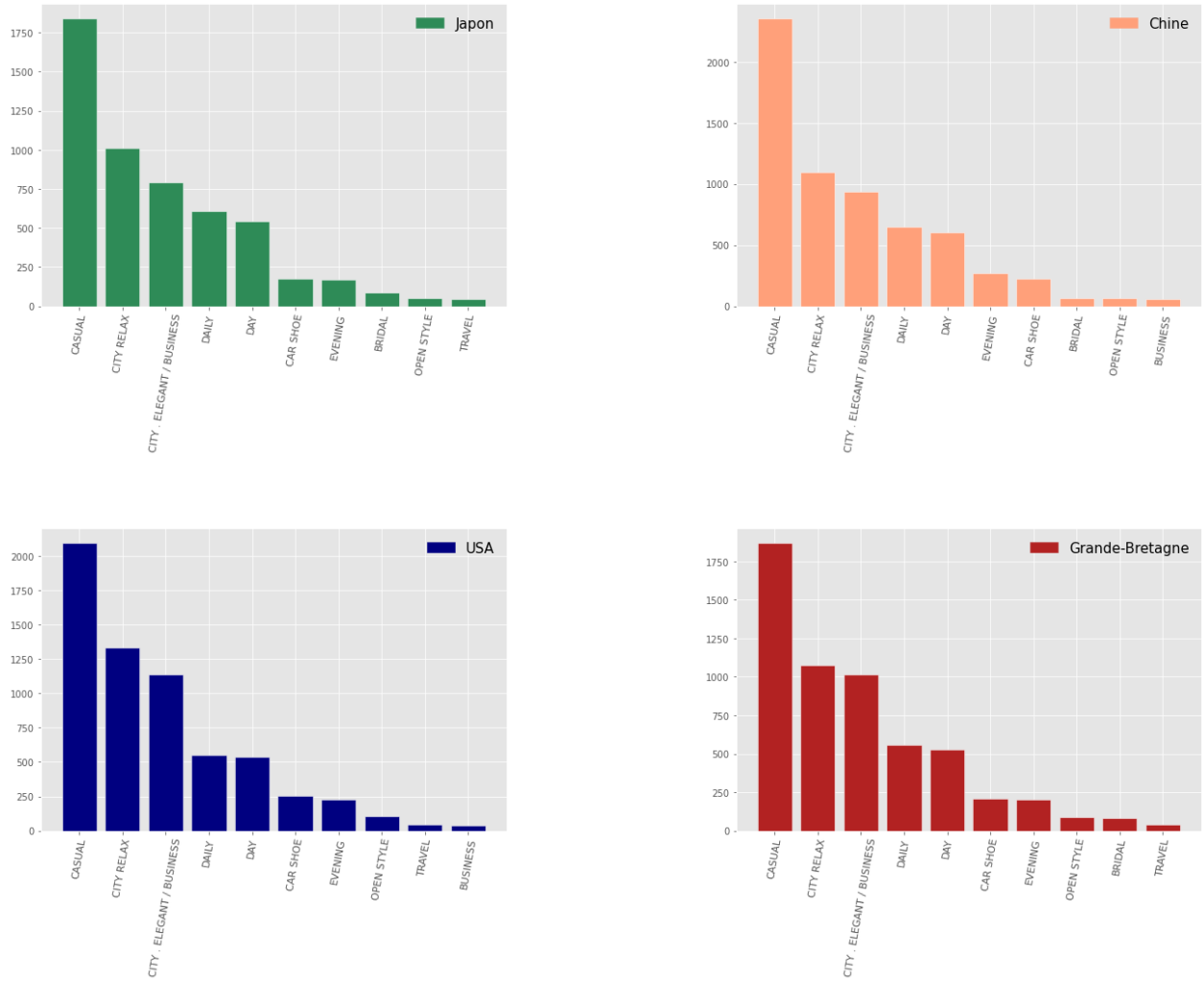


FIGURE B.2 – Répartition des modalités de la variable USAGE dans les 4 zones géographiques

Annexe C

Moyenne des probabilités de clic selon la variable MADE_IN

	Moyenne des probabilités de clic				
Modalité	**Global**	**Japon**	**Chine**	**USA**	**Grande-Bretagne**
France	0,010	0,007	0,013	0,009	0,010
Italy	0,016	0,015	0,013	0,015	0,018
Japan	0,017	0,010	0,013	0,019	0,005
Missing	0,015	0,014	0,015	0,017	0,018

TABLE C.1 – Moyenne des probabilités de clic selon les zones en fonction de la variable MADE_IN

Annexe D

Distribution des prix en dollar des produits disponibles en ligne

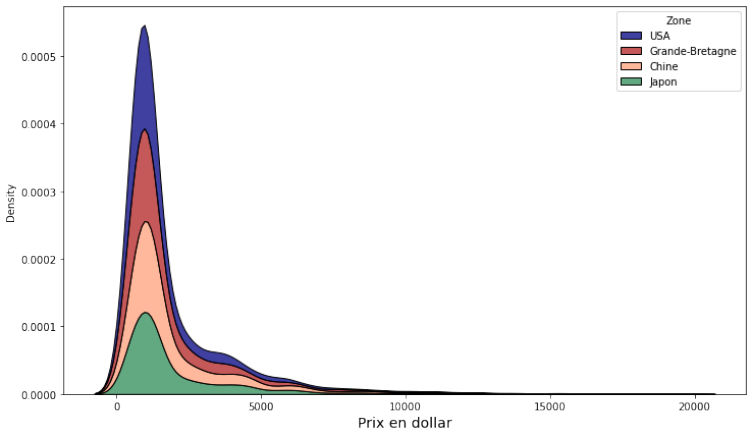


FIGURE D.1 – Distribution des prix en dollar des produits disponibles en ligne selon les zones géographiques

Annexe E

Nuages de points de la probabilité de clic en fonction de la variable de prix zz_ZZ_Price

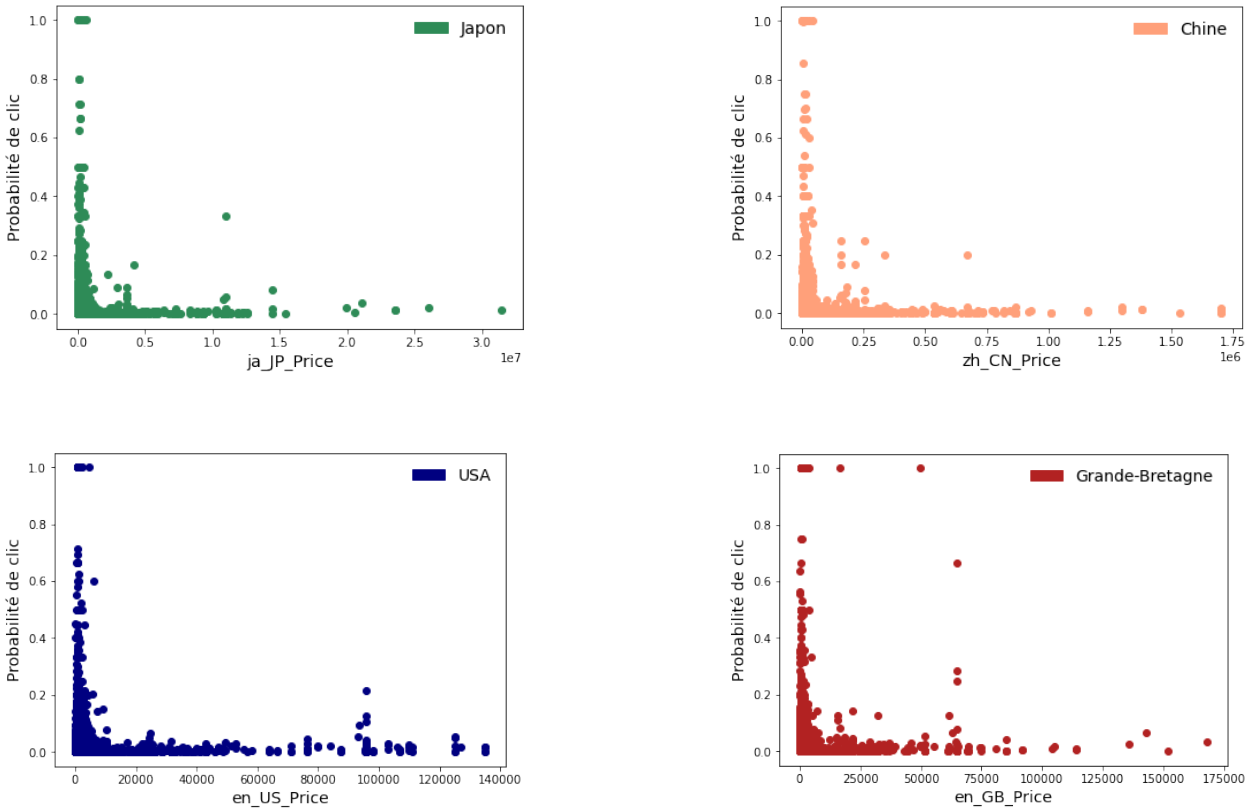
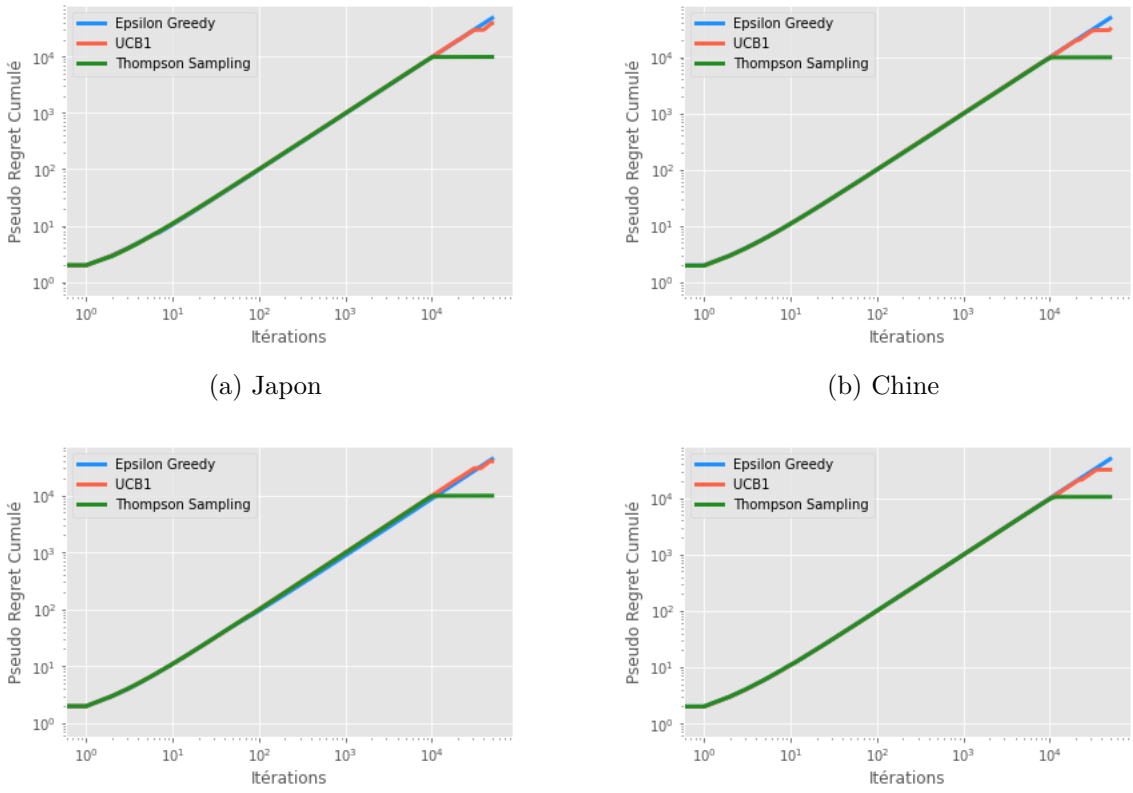


FIGURE E.2 – Nuages de points de la probabilité de clic en fonction de la variable de prix zz_ZZ_Price dans chacune des 4 zones géographiques

Annexe F

Représentation des bandits non contextuels en échelle log-log



(a) Japon

(b) Chine

(c) USA

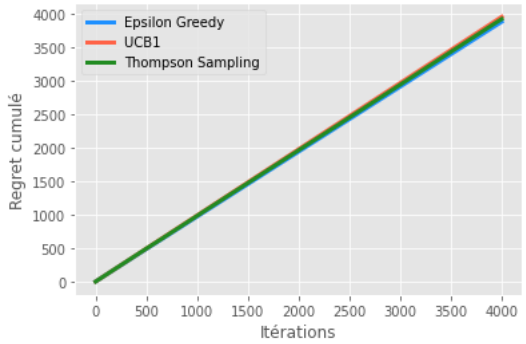
(d) Grande-Bretagne

FIGURE F.1 – Pseudo regret cumulé en fonction du nombre d'itérations pour chacune des quatre zones

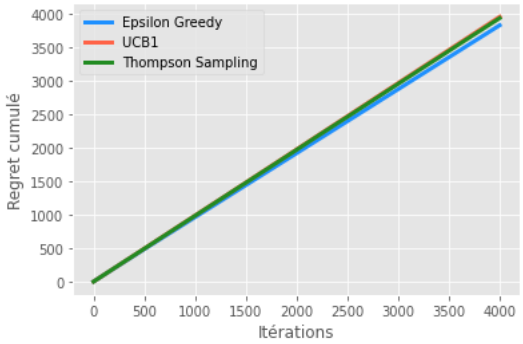
Nous remarquons que pour chaque zone, les courbes du pseudo-regret cumulé se superposent. Néanmoins, Thompson Sampling se détache des autres bandits puisqu'il devient stable après l'itération 10^4 pour chacune des zones. L'algorithme UCB1 se distingue également de ϵ-Greedy pour le Japon, la Chine et la Grande-Bretagne.

Annexe G

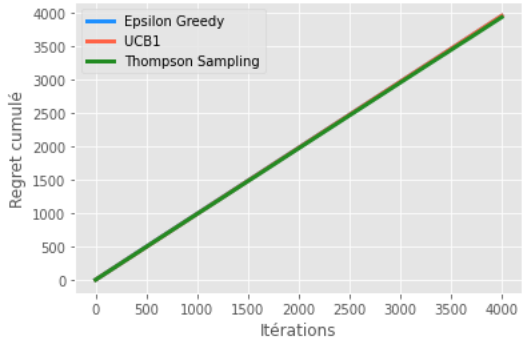
Les bandits non contextuels limités à 4000 itérations



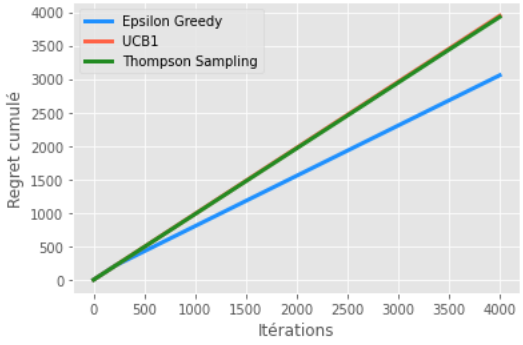
(a) Japon



(b) Chine



(c) USA



(d) Grande-Bretagne

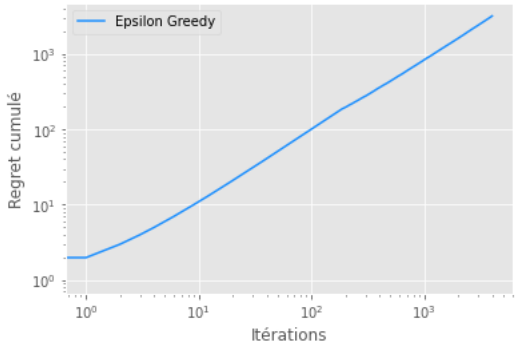
FIGURE G.1 – Regret cumulé en fonction du nombre d'itérations pour chacune des quatre zones

Nous observons qu'avec seulement 4000 itérations, le pseudo-regret cumulé est presque linéaire pour les 4 zones à l'exception de ϵ-Greedy pour la Grande-Bretagne. Il est très difficile de distinguer un algorithme parmi les trois. De plus, si l'on se fie à ces graphique, nous serions conduit à sélectionner l'algorithme ϵ-Greedy pour la Grande-Bretagne, or lorsque le nombre d'itérations augmente à 50 000, Thompson-Sampling est le seul algorithme à se stabiliser à un pseudo-regret cumulé de 10 000. Cela montre que le nombre d'itérations joue un rôle très important dans la construction et

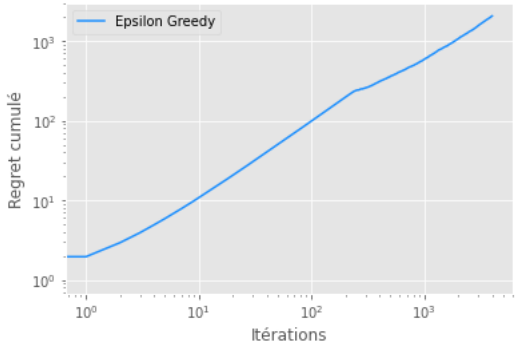
la sélection des bandits, et notamment des bandits indépendants. En effet, un nombre d'itérations inférieur au nombre de bras conduit le bandit à ne pas explorer tous les possibles et donc à potentiellement loupé le meilleur article. En revanche, nous remarquons que malgré le faible nombre d'itérations nous obtenons déjà de bien meilleure performance grâce au contexte.

Annexe H

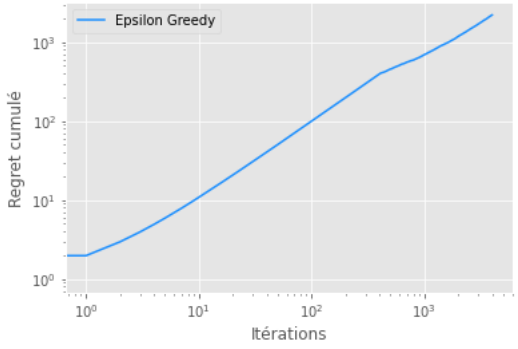
Représentation des bandits contextuels en échelle log-log



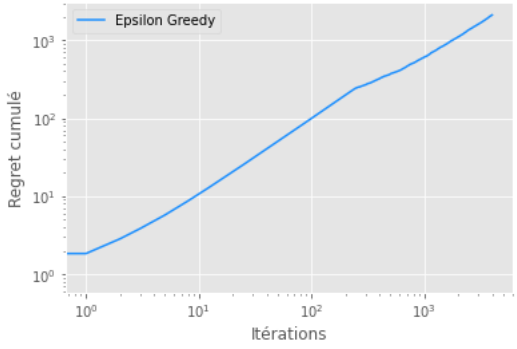
(a) Japon



(b) Chine



(c) USA



(d) Grande-Bretagne

FIGURE H.1 – Regret cumulé en fonction du nombre d'itérations pour chacune des quatre zones

Nous observons une nette amélioration de la performance du bandit ϵ-Greedy. Néanmoins, le pseudo-regret cumulé n'atteint pas une forme logarithmique, qui est la forme de regret espéré, pour un algorithme de recommandation.

Annexe I

L'algorithme NeuralUCB

L'algorithme NeuralUCB, présenté par Dongruo Zhou, Lihong Li, Quanquan Gu dans leur article *Neural Contextual Bandits with UCB-based exploration*, a une architecture similaire aux algorithmes UCB (Upper Confidence Bound).

Algorithm 1 NeuralUCB

Input: Number of rounds T, regularization parameter λ, exploration parameter ν, confidence parameter δ, norm parameter S, step size η, number of gradient descent steps J, network width m, network depth L.
Initialization: Randomly initialize $\boldsymbol{\theta}_0$ as described in the text
Initialize $\mathbf{Z}_0 = \lambda \mathbf{I}$
for $t = 1, \ldots, T$ **do**
  Observe $\{\mathbf{x}_{t,a}\}_{a=1}^{K}$
  for $a = 1, \ldots, K$ **do**
    Compute $U_{t,a} = f(\mathbf{x}_{t,a}; \boldsymbol{\theta}_{t-1}) + \gamma_{t-1} \sqrt{\mathbf{g}(\mathbf{x}_{t,a}; \boldsymbol{\theta}_{t-1})^\top \mathbf{Z}_{t-1}^{-1} \mathbf{g}(\mathbf{x}_{t,a}; \boldsymbol{\theta}_{t-1})/m}$
    Let $a_t = \operatorname{argmax}_{a \in [K]} U_{t,a}$
  end for
  Play a_t and observe reward r_{t,a_t}
  Compute $\mathbf{Z}_t = \mathbf{Z}_{t-1} + \mathbf{g}(\mathbf{x}_{t,a_t}; \boldsymbol{\theta}_{t-1}) \mathbf{g}(\mathbf{x}_{t,a_t}; \boldsymbol{\theta}_{t-1})^\top / m$
  Let $\boldsymbol{\theta}_t = \text{TrainNN}(\lambda, \eta, J, m, \{\mathbf{x}_{i,a_i}\}_{i=1}^{t}, \{r_{i,a_i}\}_{i=1}^{t}, \boldsymbol{\theta}_0)$
  Compute

$$\gamma_t = \sqrt{1 + C_1 m^{-1/6} \sqrt{\log m} L^4 t^{7/6} \lambda^{-7/6}} \cdot \left(\nu \sqrt{\log \frac{\det \mathbf{Z}_t}{\det \lambda \mathbf{I}} + C_2 m^{-1/6} \sqrt{\log m} L^4 t^{5/3} \lambda^{-1/6} - 2 \log \delta} + \sqrt{\lambda} S \right)$$
$$+ (\lambda + C_3 t L) \Big[(1 - \eta m \lambda)^{J/2} \sqrt{t/\lambda} + m^{-1/6} \sqrt{\log m} L^{7/2} t^{5/3} \lambda^{-5/3} (1 + \sqrt{t/\lambda}) \Big].$$

end for

FIGURE I.1 – Algorithme principal de NeuralUCB

Nous observons que le mécanisme de choix de l'article à recommander est identique à celui de UCB, l'algorithme choisit l'article qui maximise l'Upper confidence bound. De même que pour les algorithmes de bandit standards, NeuralUCB met à jour ses paramètres une fois le choix de l'article effectué. En revanche, nous observons que l'estimation du *reward* n'est pas standard, le paramètre θ, qui est utilisé dans l'estimation du *reward*, est mis à jour grâce à un réseau de neurones.

Les auteurs donnent quelques indications sur le réseau de neurones, le schéma ci-dessous résume ces indications. Ils donnent également l'algorithme de descente de gradient pour minimiser la fonction de coût.

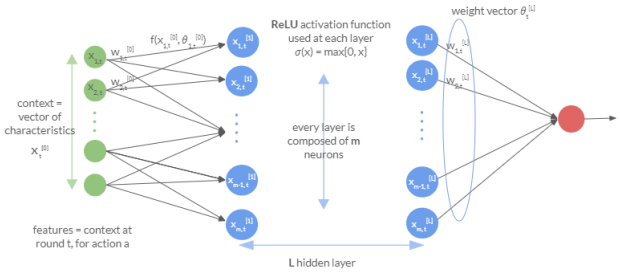


FIGURE I.2 – Schéma du réseau de neurones

Algorithm 2 TrainNN($\lambda, \eta, U, m, \{\mathbf{x}_{i,a_i}\}_{i=1}^t, \{r_{i,a_i}\}_{i=1}^t, \boldsymbol{\theta}^{(0)}$)

1: **Input:** Regularization parameter λ, step size η, number of gradient descent steps U, network width m, contexts $\{\mathbf{x}_{i,a_i}\}_{i=1}^t$, rewards $\{r_{i,a_i}\}_{i=1}^t$, initial parameter $\boldsymbol{\theta}^{(0)}$.
2: Define $\mathcal{L}(\boldsymbol{\theta}) = \sum_{i=1}^t (f(\mathbf{x}_{i,a_i}; \boldsymbol{\theta}) - r_{i,a_i})^2/2 + m\lambda\|\boldsymbol{\theta} - \boldsymbol{\theta}^{(0)}\|_2^2/2$.
3: **for** $j = 0, \ldots, J-1$ **do**
4: $\quad \boldsymbol{\theta}^{(j+1)} = \boldsymbol{\theta}^{(j)} - \eta \nabla \mathcal{L}(\boldsymbol{\theta}^{(j)})$
5: **end for**
6: **Return** $\boldsymbol{\theta}^{(J)}$.

FIGURE I.3 – Algorithme de descente de gradient de NeuralUCB

La fonction de *reward* est estimée par la fonction suivante : $f(x;\theta) = \sqrt{m}(W_L\sigma(W_{L-1}\sigma(...\sigma(W_1 x)))$, avec la fonction ReLU (Rectified Linear Unit) comme fonction d'activation, $\sigma(x) = \max\{0, x\}$. La fonction de *reward* prend en entrée le contexte de l'article préalablement choisit par la partie UCB de l'algorithme et sort une probabilité. Les auteurs préconisent d'utiliser la même largeur pour toutes les couches du réseau de neurones (le même nombre de neurones) et d'ajouter au moins 2 couches au réseau.

Quant à l'algorithme de descente de gradient, la fonction *TrainNN()*, une descente de gradient classique est utilisée, mais une descente de gradient stochastique peut être également utilisée.

Annexe J

Les réglages du réseau de neurones pour NeuralUCB

Nous avons décidé de nous inspirer de l'architecture du réseau de neurones utilisés par les auteurs de *Neural Contextual Bandits with UCB-based Exploration* pour notre premier algorithme NeuralUCB. Cette architecture, comme présenté Figure 3.1, est relativement simple du fait qu'elle ne possède que 2 couches linéaires. Néanmoins, plusieurs ajustements sont nécessaires pour que le réseau soit adapté à notre contexte, qui est très différent de celui utilisé par les auteurs. Pour réaliser nos tests pour définir la structure du réseau de neurones, nous divisons notre contexte en deux échantillons, un échantillon d'entraînement (70 % du contexte) et un échantillon de validation (30 %). Nous rappelons que le contexte correspond au caractéristiques des produits du catalogue de chaque zone géographique d'intérêt.

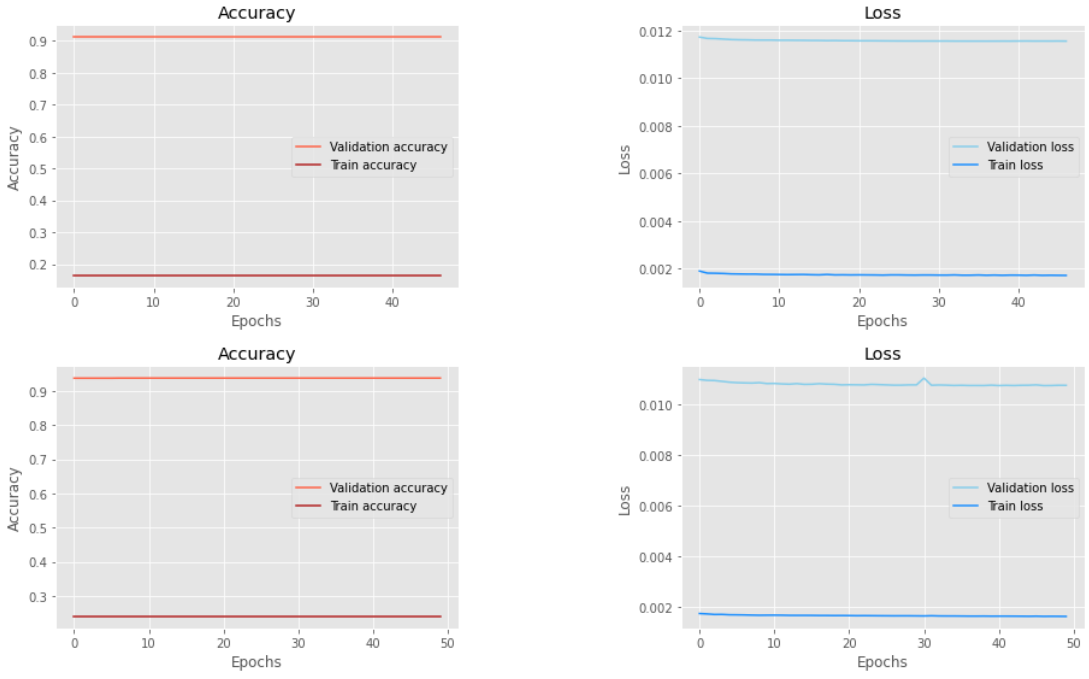


FIGURE J.1 – Performances du réseau de neurones avec 20 neurones (graphiques supérieurs) et 50 neurones (graphiques inférieurs) sur la couche cachée, pour les États-Unis

Nous commençons par expérimenter différents nombres de neurones sur la couche cachée. Nous observons qu'il y a peu de différences ici, on détecte une légère plus rapide convergence pour la structure avec 20 neurones (40 epochs suffisent, contre 50 pour le réseau à 50 neurones sur la couche cachée), même si l'accuracy et la perte semblent stable dès la première epoch. Nous avons appliqué un *early stopping* avec une patience de 10 epochs, ce qui explique la différence dans le nombre de d'epochs. Nous avons testé d'autres largeur de couche cachée, les résultats sont similaires. Nous choisissons de conserver la structure avec 20 neurones car elle permet d'avoir le meilleur compromis performance/nombre de paramètres. En effet, plus le nombre de neurones est important, plus l'algorithme a de paramètres à estimer. Avec 20 neurones, le réseau doit déjà estimer 48 000 à 54 000 neurones (en fonction des zones géographiques).

Nous observons que la perte de l'échantillon de validation est très légèrement supérieure à celle de l'échantillon d'entraînement. Nous avons testé différentes tailles de batch, différents taux d'apprentissage et avons ajouté une couche de dropout pour réduire cet écart. Ces différents réglages n'ont pas modifié la valeur des pertes, ce à quoi nous nous attendions car la différence est très petite.

Notre réseau de neurones final a alors la structure et les réglages suivants :

- 2 couches, 1 couche cachée
- 20 neurones sur la couche cachée
- Une couche de dropout
- Un taux d'apprentissage de 0.01
- Une taille de batch de 50

Bibliographie

[1] Foster, D. J. and Rakhlin, A. *Beyond UCB: Optimal and Efficient Contextual Bandits with Regression Oracles*, 2020.

[2] Rakotomalala, R. *Analyse de corrélation : Étude des dépendances - Variables quantitatives*, 2017.

[3] Zhang, Yao, Sun. *Deep Learning based Recommender System : A Survey and New Perpectives, arXiv:1707.07435v7 [cs.IR], 2019.*

[4] Zhou, D., Li, L. and Gu, Q. *Neural Contextual Bandits with UCB-based Exploration*, 2020.

[5] Zhou, D., Li, L. and Gu, Q. *Github de l'algorithme NeuralUCB.*

[6] Zhuang, Tang, Ding, Tatikonda, Dvornek, Papademetris, S. Duncan *AdaBelief Optimizer: Adapting Stepsizes by the Belief in Observed Gradients.*