Rapport TP3

IFT-3919
Qualité du logiciel et métriques
A2022

Jenny Diep (20036864)
Charbel Machaalani (20204556)

Université de Montréal

T1. (30%) Visualisez chacune des métriques de l'échantillon en créant les boites à moustaches. Calculez les informations pertinentes selon les définitions du cours (diapositives 6,7) et décrivez les distributions.

En utilisant Seaborn, une bibliothèque Python de visualisation de données basée sur matplotlib, pour créer le diagramme de boîte (boîte à moustache).

On observe les résultats suivants (les images suivantes se retrouvent dans le fichier nommé "png"):

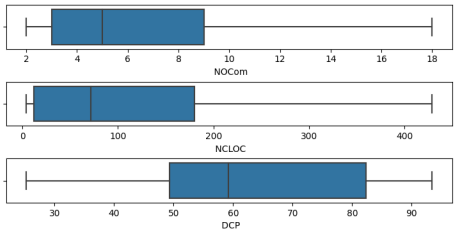


Figure 1. Graphe à mustache du fichier jfreechart-stats.csv

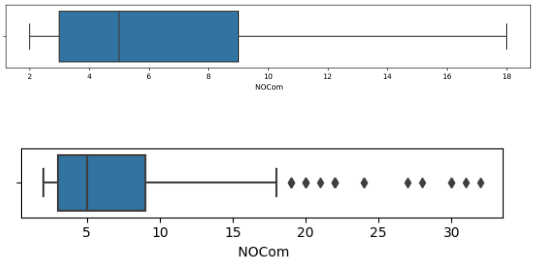


Figure 2. Graphe à moustache de la colonne NOCom (version sans fliers and avec)

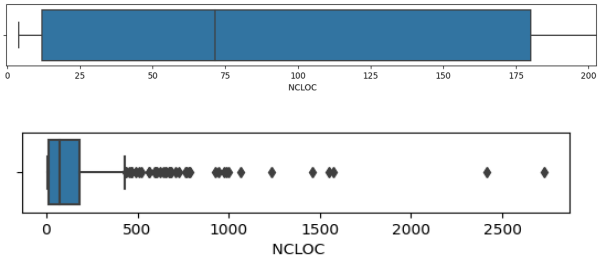


Figure 3. Graphe à moustache de la colonne NCLOC (version agrandi & avec les fliers)

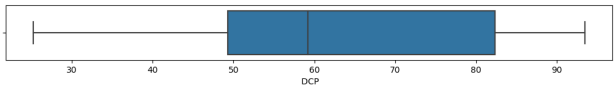


Figure 4. Graphe à moustache de la colonne DCP

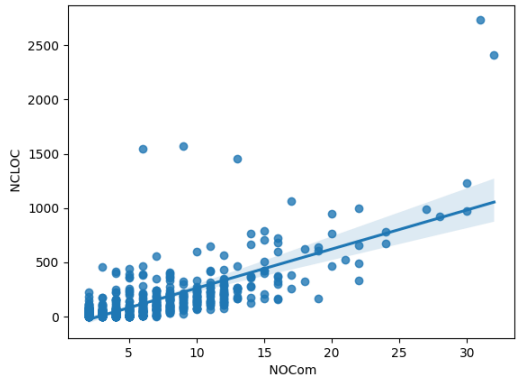
Après avoir calculé les distributions (voir dans le fichier csv_results), nous observons que le graphe à moustache pour NOCom n'est pas très significatif puisque la taille du graphe (sample size) n'est que de 18. Si la taille de l'échantillon est trop petite, les quartiles et les valeurs aberrantes (outliers) montrées par le boxplot peuvent ne pas être significatives. On devrait utiliser un autre type de graphe dans ce cas afin d'étudier les données.

Dans la figure 3, on observe qu'il y a 39 valeurs aberrantes qui vont jusqu'à la valeur 2500+. Pareillement dans la figure 2, on détermine qu'il y a 19 valeurs aberrantes.Puisque les valeurs aberrantes sont ceux qui sont trop loin du restant des données peuvent affecter nos résultats, on peut enlever les datas abnormal et/ou des cas spéciaux afin de retrouver le graphe dans la figure 1.

En regardant la figure 1, on peut voir qu'il y a une asymétrie dans le graphe NCLOC et NOCom. Il en existe même un peu dans le graphe DCP. L'asymétrie indique que les données ne sont peut-être pas distribuées normalement.

T2. (30%) Étudiez les corrélations entre NoCom et NCLOC, NoCom et DCP. Visualisez les données, calculez les droits de régression, et les coefficients de corrélation qui ont du sens.

Observons les graphiques suivants:



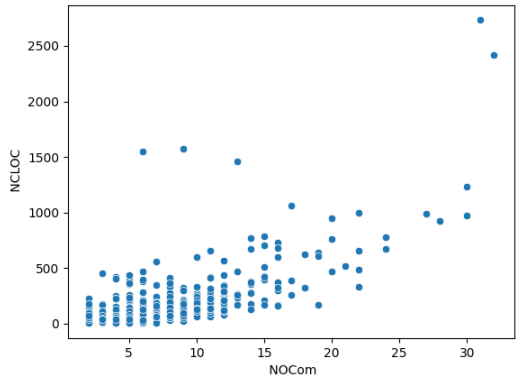
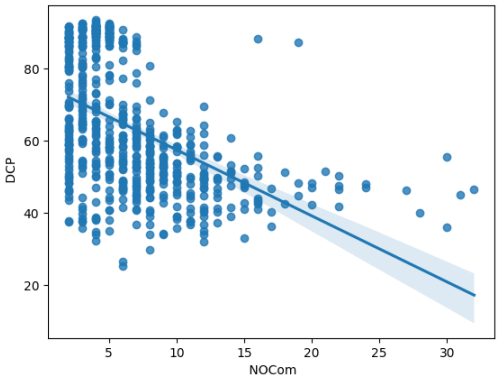


Figure 5. Corrélation entre NOCom et NCLOC



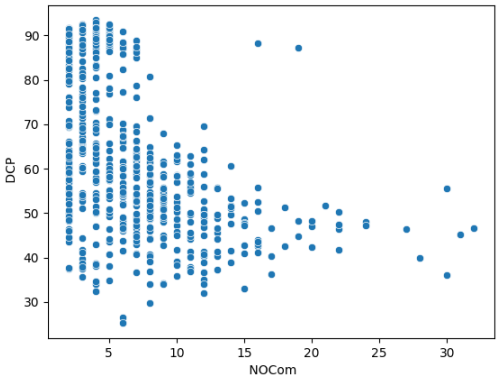


Figure 6. Corrélation entre NOCom et DCP

Avec l'outil Scipy et Seaborn, on peut déterminer visuellement et mathématiquement la corrélation entre les données. Selon les résultats de calcul, on peut dire que les variables sont presques normalement distribuées avec une corrélation presque parfaite positive pour NOCom et NCLOC et une corrélation presque parfaite négative pour NOCom et DCP.

Voici les résultats des calculs:

NOCom et NCLOC:

slope: 36.046998 intercept: -97.992499

r-squared: 0.510613

PearsonRResult(statistic=0.7145716586953953, pvalue=7.726928022388957e-100)

NOCom et DCP:

slope: -1.830022 intercept: 75.809529

r-squared: 0.237903

PearsonRResult(statistic=-0.4877529974573942, pvalue=4.41524835880053e-39)

T3. (40%) Nous voulons évaluer l'hypothèse : « les classes qui ont été modifiées plus de 10 fois sont mieux commentées que celles qui ont été modifiées moins de 10 fois ». Décrivez la conception d'un quasi-expérience qui vous permettra de le faire. Ensuite, évaluez l'hypothèse, discutez les résultats et décrivez vos conclusions. Rappelez-vous que selon les diapositives du cours, les étapes à suivre sont : choix d'étude, énoncé des hypothèses, définition des variables, interprétation et généralisation des résultats, discussion des menaces à la validité.

L'étude qu'on envisage faire ressemble beaucoup à la collecte de données effectuée dans les tâches précédentes. Nous prendrons des classes aléatoires durant le processus de développement d'une application et nous évaluerons les valeurs de NCLOC, NOCom et DCP. Nous utilisons ensuite ces valeurs pour juger si les classes ayant plus de commits ou plus que 10 commits peuvent être considérées comme des classes mieux documentées.

1. **Choix d'étude**

Nous faisons recours à une quasi-expérience.

2. **Énoncé des hypothèses**

H0: NOCom et NCLOC sont directement corrélés

H1: Les classes qui ont été modifiées plus de 10 fois sont mieux commentées que celles qui ont été modifiées moins de 10 fois.

3. **Définition des variables**

Nous identifions le NOCom comme étant la variable indépendante. DCP et NCLOC seraient les variables dépendantes. Nous voulons trouver s'il y a corrélation entre NOCom et DCP. Pour confirmer que le code est mieux commenté lorsqu'il y a plus de 10 commits, nous devons prouver que le nombre de NCLOC augmente avec plus de commits (NCLOC et NOCom sont directement proportionnels). Avec cela établi, nous devons aussi confirmer que DCP augmente de façon directement proportionnelle avec NOCom pour pouvoir dire que le code est mieux commenté avec plus de commits, puisque cela voudrait dire que le code grandi et reste bien commenté.

4. **Interprétation et généralisation des résultats**

Les résultats observés dans les figures 5 et 6 sont hautement significatifs car le p-value est plus petit que 0.005. (pvalue = 7.726928022388957e-100 pour la figure 5 et pvalue=4.41524835880053e-39 pour la figure 6).

Selon la figure 5, nous pouvons observer qu'il y a un changement directement proportionnel entre NCLOC et NOCom. Cela confirme l'hypothèse H0 qu'il existe une corrélation directement proportionnelle entre NCLOC et NOCom et que la quantité de code augmente avec le nombre de

commits. Avec la figure 6, nous observons que la densité de commentaire diminue lorsque le nombre de commits augmente. Cela voudrait dire que plus le nombre de commentaires n'augmente pas proportionnellement avec la taille du code, puisque la taille du code augmente avec le nombre de commits mais la densité de commentaire diminue. Nous pouvons alors infirmer H1 puisque le code ne serait pas mieux commenté si le nombre de commits dépasse 10.

En effet, nous avons pus trouver que le coefficient de spearman calculé à partir des valeurs de NOCom et DCP est de -0.5335180851612711 ce qui est en lien avec notre analyse. DCP diminue lorsque NOCom augmente. Il y a une corrélation inversement proportionnelle puisque le coefficient est négatif. Cette corrélation est aussi significative car le coefficient est plus petit que -0.5.

Il faut garder en tête que l'exactitude de ce modèle pourrait être menacée par le 'gaming effect' si les participants étaient au courant de l'expérimentation. Ils seraient plus incité à mieux documenter le code s'ils sont au courant qu'ils sont observés. Pour éviter cela, il faut s'assurer que les participants ne savent pas qu'il font partie de cette étude.

5. Discussion des menaces à la validité

Il peut y avoir des variables confondantes pouvant menacer la validité de construction de notre étude. Par exemple, nous ne prenons pas en considération le nombre de méthodes dans les classes. Cela pourrait être utile pour observer si l'on ajoute des méthodes avec plus de commits puisque nous savons que si les commits ne rajoutent pas des méthodes, les modifications amenées au code avec le commit pourraient être mineurs ne nécessitant pas plus de documentation. Par exemple, les modifications pourraient être de simples bug fix. Nous ne prenons pas en considération aussi la nature de chacune des classes qui pourrait nécessiter moins de commentaire mais quand même être bien documentés.

Quant à la validité interne, il faut mener l'expérimentation sur une courte période de temps puisque les participants pourraient apprendre à mieux documenter le code avec le temps et fausser les résultats à travers leur apprentissage.

Pour améliorer la précision et la validité des résultats, nous pouvons aussi aller chercher des plus de classes venant d'autre application pour obtenir une meilleure représentation des classes codés en générale pour divers types d'applications.

Une autre menace à la validité serait le fait que seulement les classes ayant survécu le processus de développement jusqu'au point de la collecte des données sont prises en considération. Cela pourrait être problématique puisque les classes moins bien documentées pourraient ne pas survivre au processus de développement puisqu'elles peuvent être laborieuses à maintenir. Les développeurs pourraient alors s'en débarrasser et considérer des alternatives.