Result

Maxime Daigle

maxime.daigle.1@umontreal.ca

2 Mai 2018

Contents

[Description du programme d’extraction de caractéristiques 1](#_Toc512608943)

[Naives Bayes 1](#_Toc512608944)

[Expérience 1 1](#_Toc512608945)

[Expérience 2 1](#_Toc512608946)

[Expérience 3 1](#_Toc512608947)

[Expérience 4 2](#_Toc512608948)

[Expérience 5 2](#_Toc512608949)

[Expérience 6 3](#_Toc512608950)

[Expérience 7 3](#_Toc512608951)

[Expérience 8 3](#_Toc512608952)

[Expérience 9 4](#_Toc512608953)

[Analyse 4](#_Toc512608954)

[Arbre de décision (J48) 5](#_Toc512608955)

[Expérience 1 5](#_Toc512608956)

[Expérience 2 5](#_Toc512608957)

[Expérience 3 5](#_Toc512608958)

[Expérience 4 6](#_Toc512608959)

[Expérience 5 6](#_Toc512608960)

[Expérience 6 7](#_Toc512608961)

[Expérience 7 7](#_Toc512608962)

[Analyse 7](#_Toc512608963)

[SVM (SMO) 8](#_Toc512608964)

[Expérience 1 8](#_Toc512608965)

[Expérience 2 8](#_Toc512608966)

[Expérience 3 8](#_Toc512608967)

[Expérience 4 9](#_Toc512608968)

[Expérience 5 9](#_Toc512608969)

[Expérience 6 10](#_Toc512608970)

[Expérience 7 10](#_Toc512608971)

[Analyse 10](#_Toc512608972)

[MultiLayerPerceptron 11](#_Toc512608973)

[Expérience 1 11](#_Toc512608974)

[Expérience 2 11](#_Toc512608975)

[Expérience 3 11](#_Toc512608976)

[Expérience 4 12](#_Toc512608977)

[Expérience 5 12](#_Toc512608978)

[Analyse 13](#_Toc512608979)

[Comparaison entre algorithmes 13](#_Toc512608980)

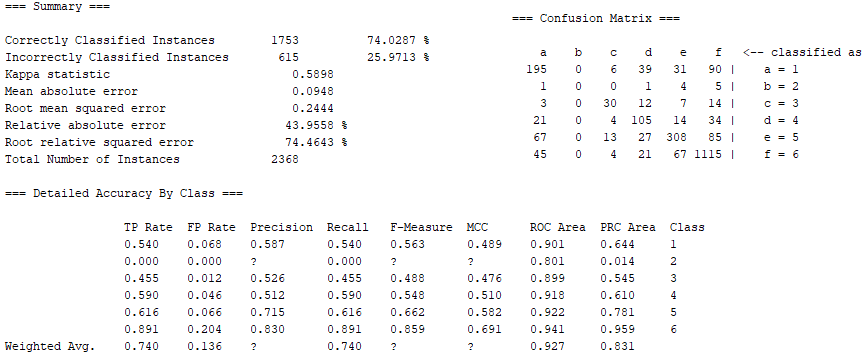
# Description du programme d’extraction de caractéristiques

Le programme enlève les ponctuations et, pour chaque instance, prend en note le sens du mot intérêt, les mots dans la fenêtre de contexte et leur catégorie. De plus, le programme a la variable window pour modifier la grandeur de la fenêtre et la variable booléenne use\_stoplist pour activer ou désactiver l’élimination des stopwords à l’aide du fichier stoplist-english.txt.

# Naives Bayes

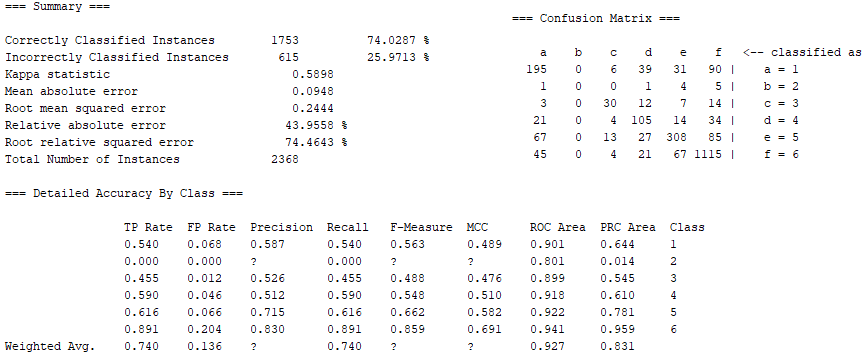
## Expérience 1

La taille de la fenêtre de contexte est de 2 mots et le fichier stoplist-english.txt est utiliser pour enlever les mot outils. Dans le preprocess de Weka, IDFTransform, TFTransform et lowerCaseToken sont à True. De plus, SnowballStemmer et MultiStopwords sont utilisés.



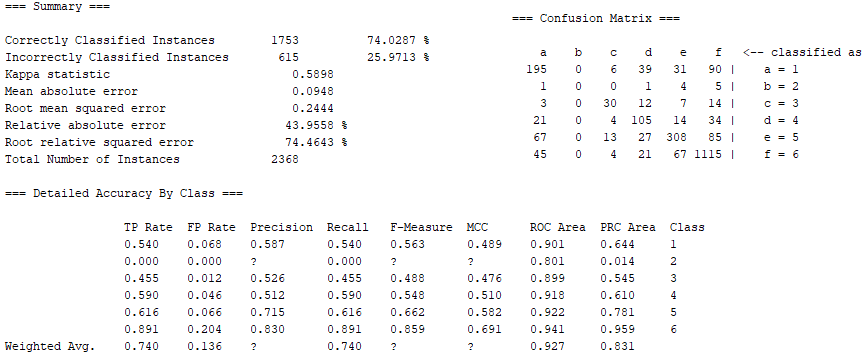
## Expérience 2

Comme l’expérience 1, sauf que IDFTransform est mis à Faux.



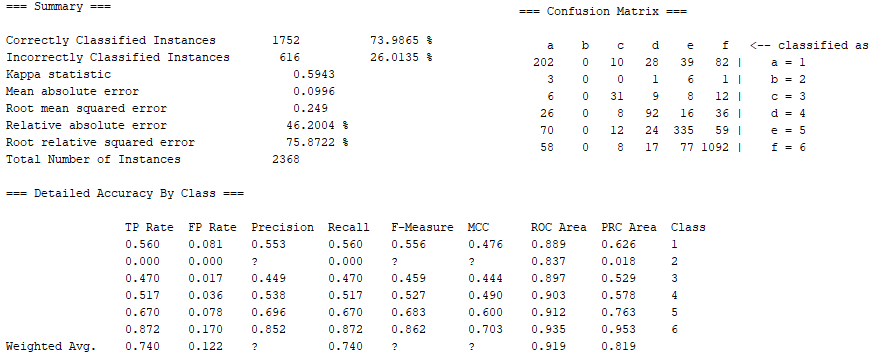
## Expérience 3

Comme l’expérience 1, sauf que MultiStopwords n’est pas utilisé.



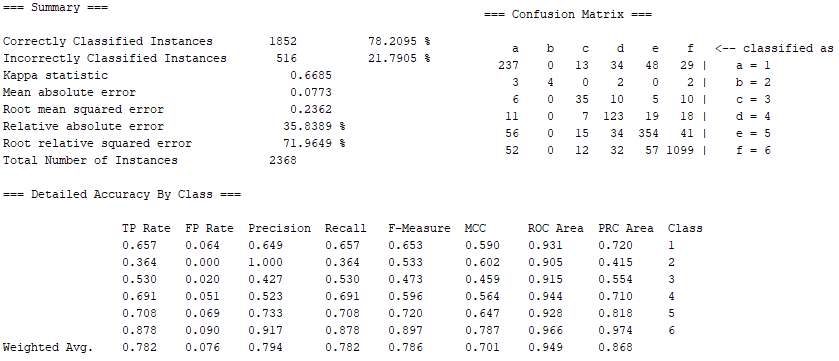
## Expérience 4

Comme l’expérience 1, sauf que SnowballStemmer n’est pas utilisé.



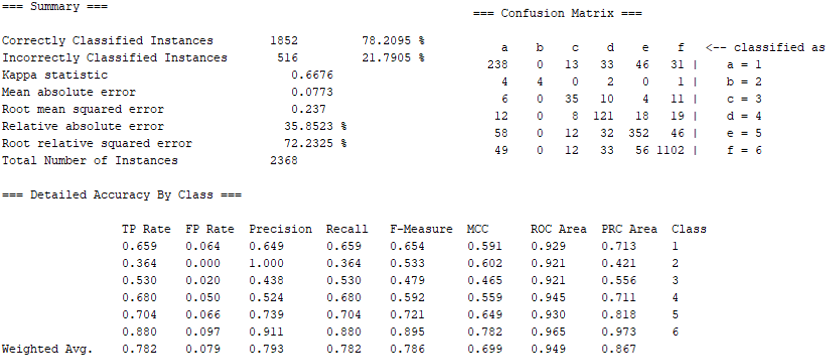
## Expérience 5

Comme l’expérience 1, sauf que le fichier stoplist-english.txt n’est pas utilisé.



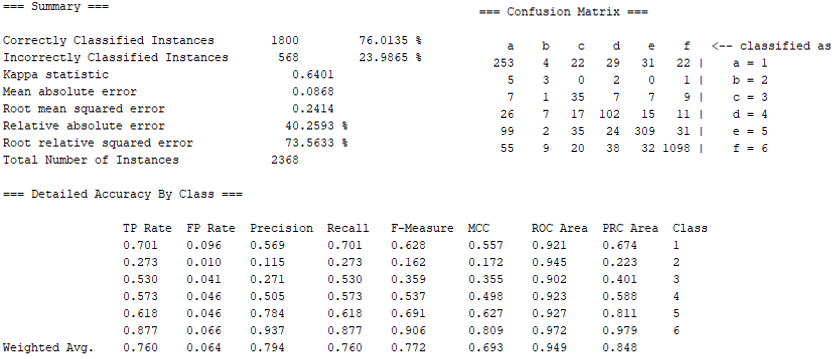
## Expérience 6

Comme l’expérience 1, sauf que le fichier stoplist-english.txt et le MultiStopwords ne sont pas utilisés.



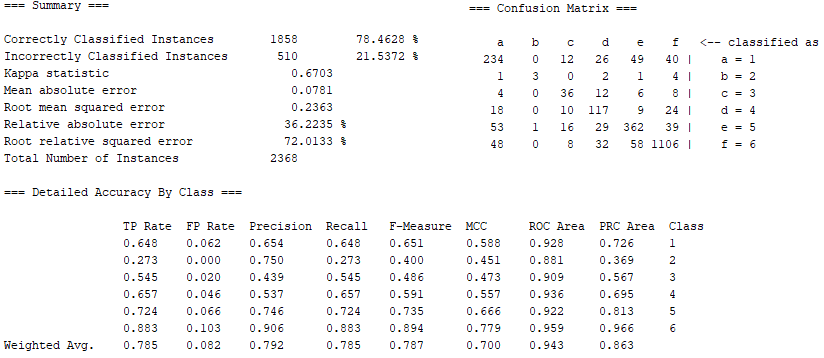
## Expérience 7

Comme l’expérience 1, sauf que le fichier stoplist-english.txt et le MultiStopwords ne sont pas utilisés et la taille de la fenêtre est 1.



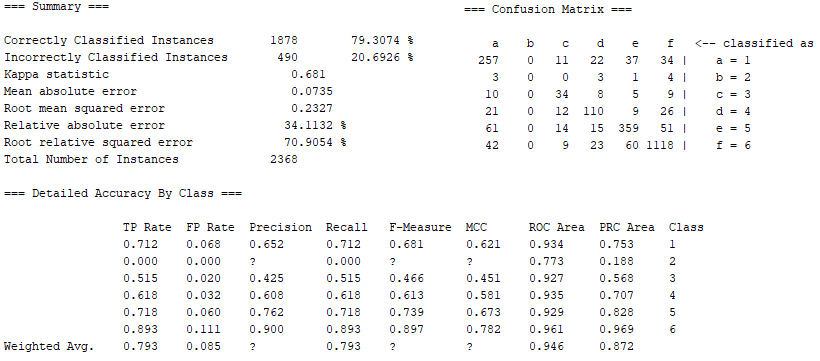
## Expérience 8

Comme l’expérience 1, sauf que le fichier stoplist-english.txt et le MultiStopwords ne sont pas utilisés et la taille de la fenêtre est 3.



## Expérience 9

Comme l’expérience 1, sauf que le fichier stoplist-english.txt et le MultiStopwords ne sont pas utilisés et la taille de la fenêtre est 5.



## Analyse

IDFTransform n’a pas d’impact et, quand stoplist-english.txt est utilisé, MultiStopwords n’a aussi pas d’impact.

Quand SnowballStemmer n’est pas utilisé, le taux d’instances correctement classifiées diminue de 0.0422%. Pour environ la moitié des classes, la précision augmente légèrement tandis que, pour l’autre moitié, elle diminue légèrement. C’est la même chose pour le rappel. Alors, le taux d’instances correctement classifiées pour chaque classe varie un peu, mais, globalement, le taux diminue faiblement.

Ne pas utiliser le fichier stoplist-english.txt augmente le taux d’instances correctement classifiées de 4.18%. Il permet, en autre, de classifier correctement quelques instances avec le deuxième sens tandis qu’avec le stoplist-english.txt, aucune instance de cette catégorie n’était identifiée. De plus, si stoplist-english.txt et MultiStopwords ne sont pas utilisés, les classifications changent très légèrement, mais le taux reste exactement le même.

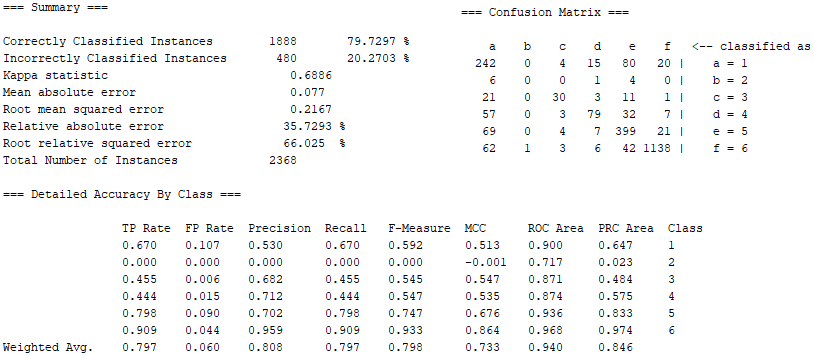
Diminuer la taille de la fenêtre à 1 diminue le taux d’instances correctement classifiées de 2.196%. Augmenter la taille à 3 augmente le taux de 0.2533% et Augmenter la taille à 5 augmente le taux de 1.098%. Une fenêtre plus grande donne, donc, de meilleurs résultats.

La meilleure performance obtenue est, alors, un taux d’instances correctement classifiées de 79.3074%.

# Arbre de décision (J48)

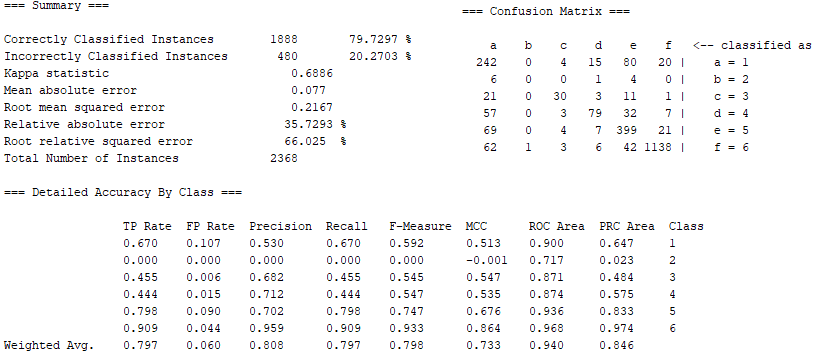
## Expérience 1

La taille de la fenêtre de contexte est de 2 mots et le fichier stoplist-english.txt est utiliser pour enlever les mot outils. Dans le preprocess de Weka, IDFTransform, TFTransform et lowerCaseToken sont à True. De plus, SnowballStemmer et MultiStopwords sont utilisés.



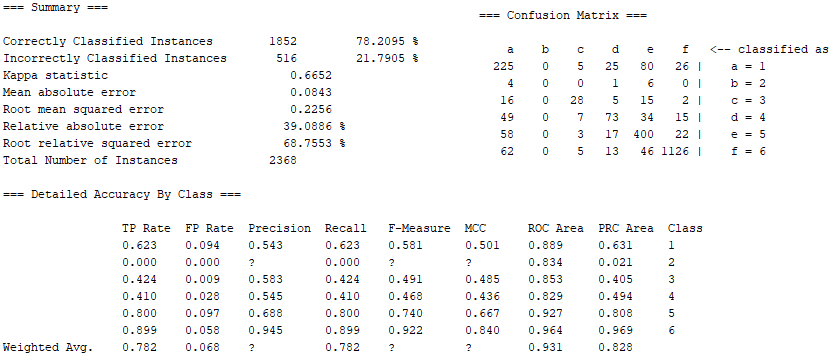
## Expérience 2

Comme l’expérience 1, sauf que IDFTransform est mis à Faux.



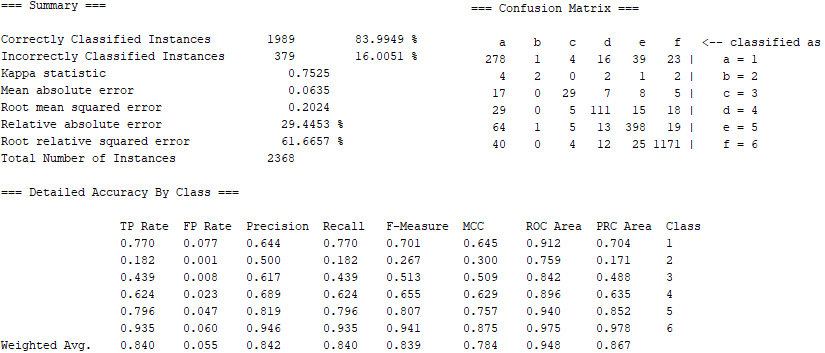
## Expérience 3

Comme l’expérience 1, sauf que SnowballStemmer n’est pas utilisé.



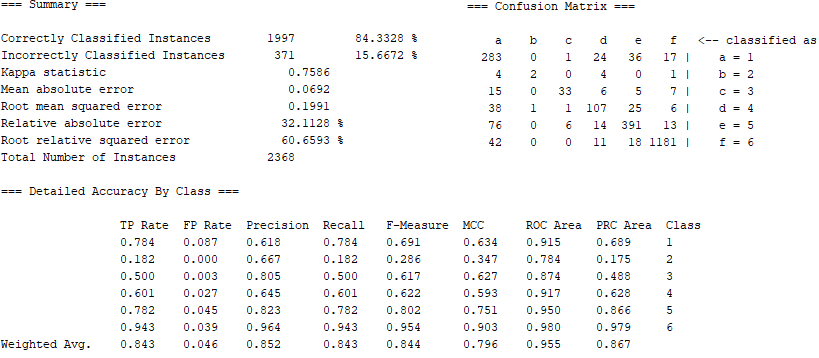
## Expérience 4

Comme l’expérience 1, sauf que le fichier stoplist-english.txt et le MultiStopwords ne sont pas utilisés.



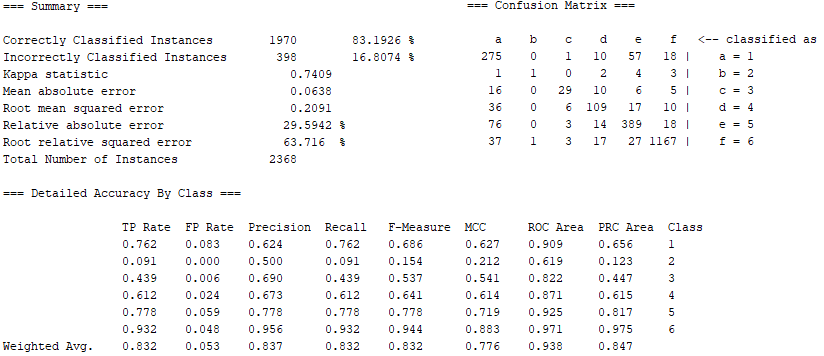
## Expérience 5

Comme l’expérience 1, sauf que le fichier stoplist-english.txt et le MultiStopwords ne sont pas utilisés et la taille de la fenêtre est 1.



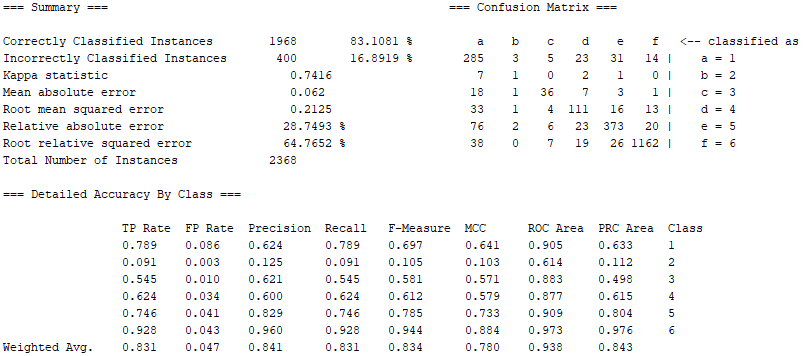
## Expérience 6

Comme l’expérience 1, sauf que le fichier stoplist-english.txt et le MultiStopwords ne sont pas utilisés et la taille de la fenêtre est 3.



## Expérience 7

Comme l’expérience 1, sauf que le fichier stoplist-english.txt et le MultiStopwords ne sont pas utilisés et la taille de la fenêtre est 5.



## Analyse

IDFTransform n’a pas d’impact. Quand SnowballStemmer n’est pas utilisé, le taux d’instances correctement classifiées diminue de 1.5202%.

Ne pas utiliser stoplist-english.txt et MultiStopwords augmente le taux de 4.2652%. De plus, diminuer la taille de la fenêtre à 1 permet d’augmenter le taux de 0.3379%.

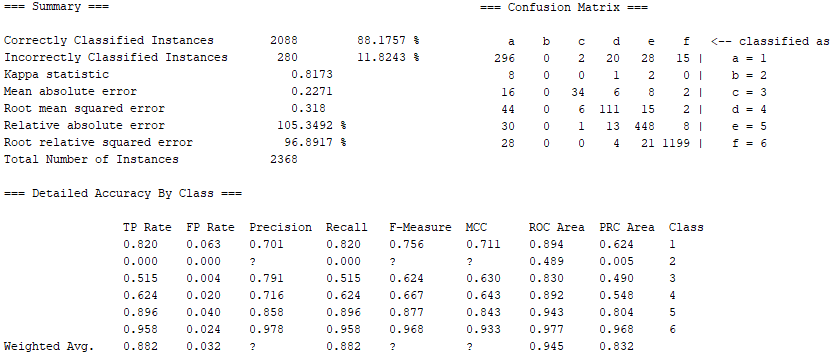
Augmenter la fenêtre à 3 mots descend le taux de 0.8023% et augmenter la fenêtre à 5 mots fait descendre le taux de 0.8868%. Donc, l’arbre de décision performe le mieux avec une fenêtre de 1 mot et performe de moins en moins bien en augmentant la taille de la fenêtre.

La meilleure performance obtenue est, alors, un taux d’instances correctement classifiées de 84.3328%.

# SVM (SMO)

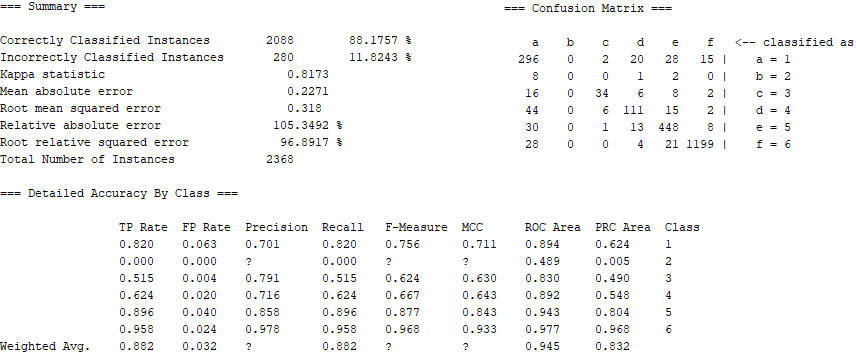
## Expérience 1

La taille de la fenêtre de contexte est de 2 mots et le fichier stoplist-english.txt est utiliser pour enlever les mot outils. Dans le preprocess de Weka, IDFTransform, TFTransform et lowerCaseToken sont à True. De plus, SnowballStemmer et MultiStopwords sont utilisés.



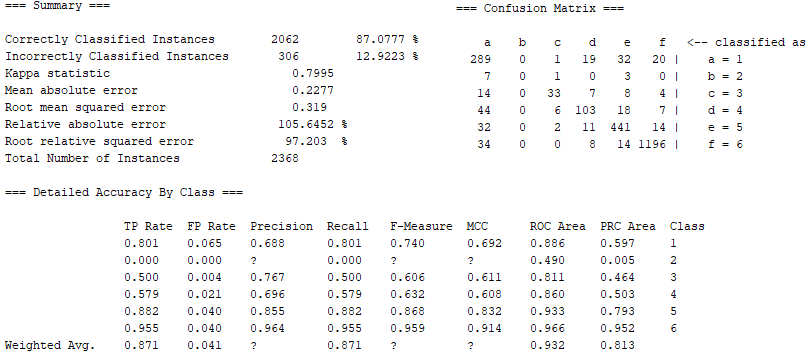
## Expérience 2

Comme l’expérience 1, sauf que TFTransform est mis à Faux.



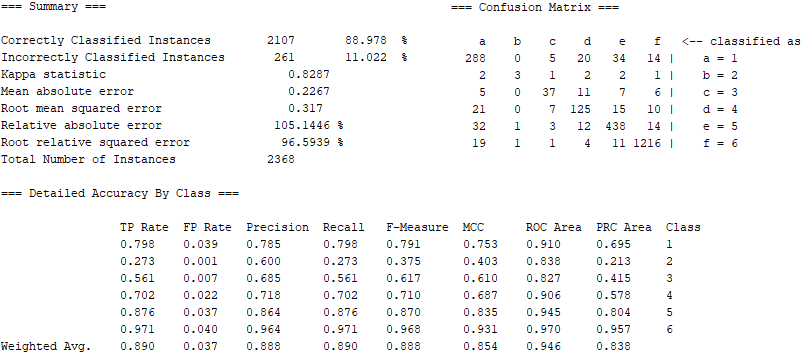
## Expérience 3

Comme l’expérience 1, sauf que SnowballStemmer n’est pas utilisé.



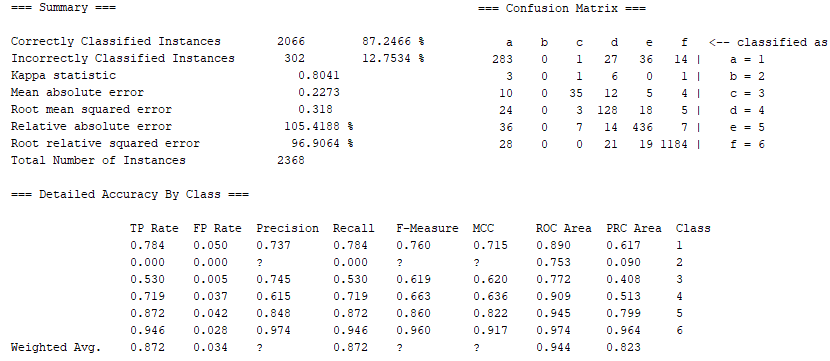
## Expérience 4

Comme l’expérience 1, sauf que le fichier stoplist-english.txt et le MultiStopwords ne sont pas utilisés.



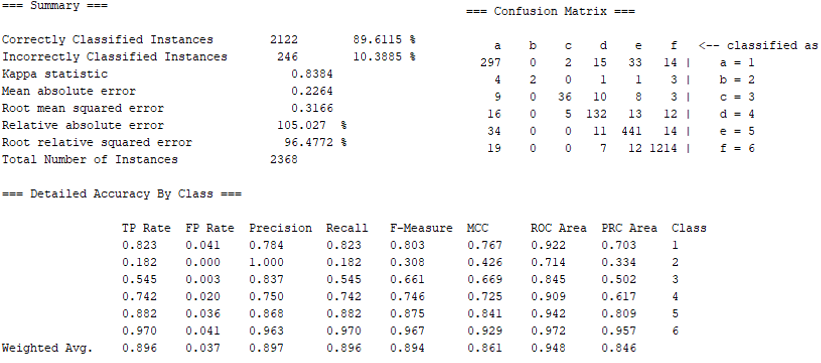
## Expérience 5

Comme l’expérience 1, sauf que le fichier stoplist-english.txt et le MultiStopwords ne sont pas utilisés et la taille de la fenêtre est 1.



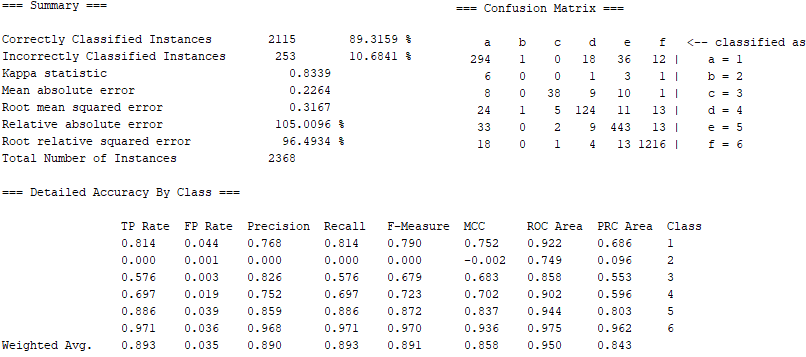
## Expérience 6

Comme l’expérience 1, sauf que le fichier stoplist-english.txt et le MultiStopwords ne sont pas utilisés et la taille de la fenêtre est 3.



## Expérience 7

Comme l’expérience 1, sauf que le fichier stoplist-english.txt et le MultiStopwords ne sont pas utilisés et la taille de la fenêtre est 5.



## Analyse

TFTransform n’a pas d’impact. Quand SnowballStemmer n’est pas utilisé, le taux d’instances correctement classifiées diminue de 1.098%.

Ne pas utiliser le fichier stoplist-english.txt et MultiStopwords augmente le taux d’instances correctement classifiées de 0.8023%. Il permet, en autre, de classifier correctement quelques instances avec le deuxième sens tandis que sans les stopwords, aucune instance de cette catégorie n’était identifiée.

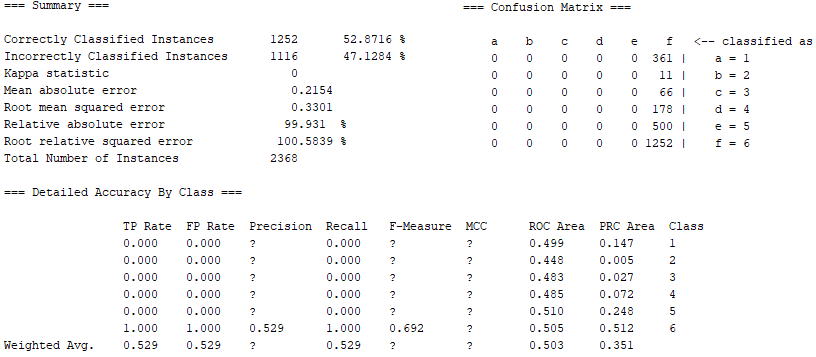
Diminuer la taille de la fenêtre de contexte à 1 mots diminue le taux de 1.7314%. Augmenter la taille à 3 mots augmente le taux de 0.6335% et augmenter la fenêtre à 5 mots donne une augmentation du taux de 0.3379%. Donc, le SVM a sa meilleure performance avec une fenêtre de 3 mots.

La meilleure performance obtenue est, alors, un taux d’instances correctement classifiées de 89.6115%.

# MultiLayerPerceptron

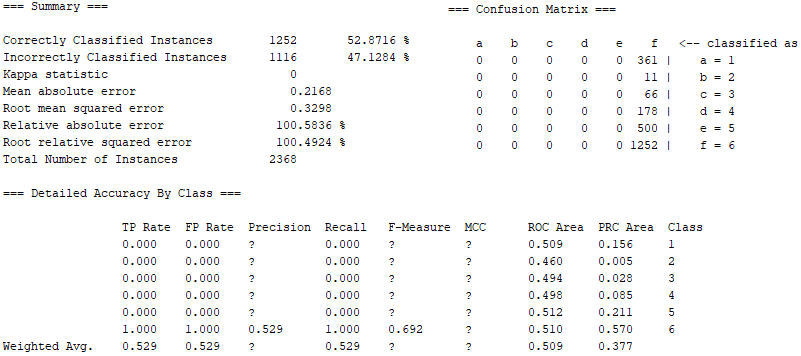
## Expérience 1

La taille de la fenêtre de contexte est de 2 mots et le fichier stoplist-english.txt est utiliser pour enlever les mot outils. Dans le preprocess de Weka, IDFTransform, TFTransform et lowerCaseToken sont à True. De plus, SnowballStemmer et MultiStopwords sont utilisés. Pour ce qui est des caractéristiques particulières au MultiLayerPerceptron, il y a 5 neurones cachés et reset est à False.



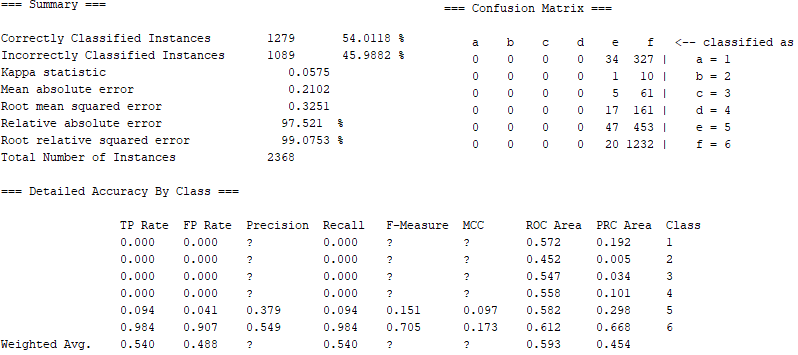
## Expérience 2

Comme l’expérience 1, sauf que le fichier stoplist-english.txt et le MultiStopwords ne sont pas utilisés, la taille de la fenêtre est 5 et reset est mis à True. (avec 10 et 30 neurones cachés, ça donne le même résultat)



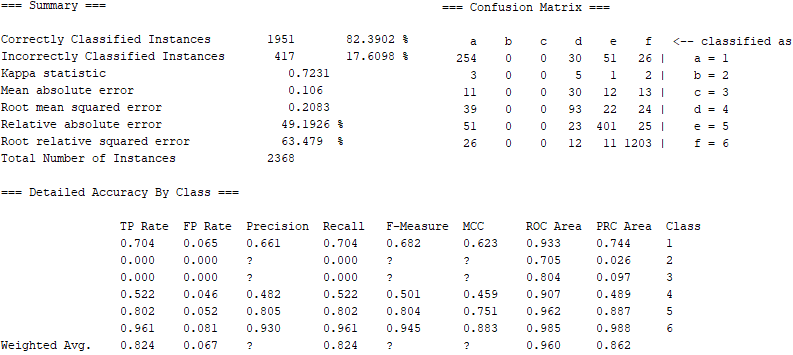
## Expérience 3

Comme l’expérience 1, sauf que le fichier stoplist-english.txt et le MultiStopwords ne sont pas utilisés, il y a 30 neurones cachés, reset est mis à False et le learning rate est à 0.1 au lieu de 0.3.



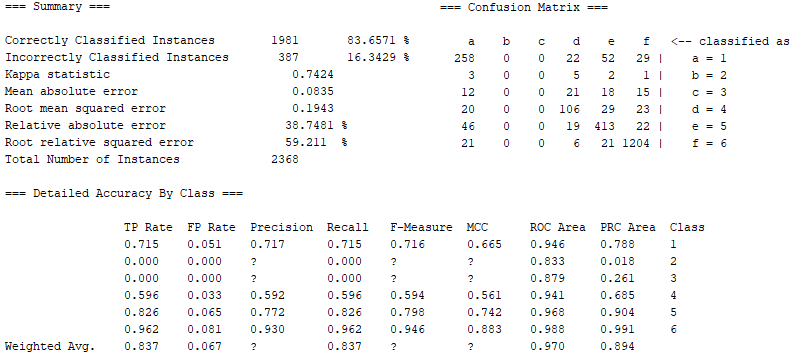
## Expérience 4

Comme l’expérience 1, sauf que le fichier stoplist-english.txt et le MultiStopwords ne sont pas utilisés, il y a 10 neurones cachés, reset est mis à False, TrainingTime est à 100 au lieu de 500 et le learning rate est à 0.01 au lieu de 0.3.



## Expérience 5

Comme l’expérience 1, sauf que le fichier stoplist-english.txt et le MultiStopwords ne sont pas utilisés, il y a 30 neurones cachés, reset est mis à False, TrainingTime est à 100 au lieu de 500 et le learning rate est à 0.01 au lieu de 0.3.



## Analyse

Dans les premières expériences, le MultiLayerPerceptron n’a pas très bien performé. L’utilisation de stopwords, le nombre de neurones cachés et l’utilisation de reset n’ont pas eu l’air d’avoir d’impact. Il a eu tendance à seulement tout classifier les instances dans la sixième catégorie. Cela pourrait être causé par le déséquilibre dans la représentation des classes dans les instances (53% des instances sont dans la sixième catégorie).

Diminuer le learning rate à 0.1 a semblé aider le MultiLayerPerceptron, mais il a seulement classifié toutes les instances dans 2 des 6 catégories (qui représente 53% et 21% de toutes les instances) au lieu de toutes les classifiés dans une seule catégorie. Diminuer le learning rate à 0.01 permet de classifier les instances dans 4 des 6 catégories (en ignorant les catégories représentant 1% et 3% des instances). De plus, avec ce learning rate, augmenter le nombre de neurones cachés de 10 à 30 a augmenté le taux d’instances correctement classifiés de 1.2669%.

La meilleure performance obtenue est, alors, un taux d’instances correctement classifiées de 83.6571%.

# Comparaison entre algorithmes

Dans tous les cas, IDFTransform et TFTransform n’ont pas d’impact. De plus, utiliser stoplist-english.txt et MultiStopwords donne étonnamment de moins bons résultats. Quand les stopwords ne sont pas enlevés, le taux d’instances correctement classifiées augmente de 4.18% pour Naives Bayes, 4.2652% pour l’arbre de décision et seulement 0.8023% pour le SVM. SnowballStemmer permet d’améliorer légèrement les résultats (augmente le taux de 0.0422% pour Naives Bayes, 1.5202% pour l’arbre et 1.098% pour le SVM).

L’arbre de décision performe mieux que Naive Bayes en obtenant un taux d’instances correctement classifiés plus grand de 5.0254% (84.3328% pour l’arbre contre 79.3074% pour Naives Bayes). L’arbre performe le mieux avec une fenêtre de 1 mot et performe de moins en moins bien en grandissant la fenêtre tandis que c’est exactement le contraire pour Naive Bayes.

Le MultiLayerPerceptron a mieux performé (avec un taux de 83.6571%) que Naives Bayes, mais légèrement moins que l’arbre de décision qui a eu un taux plus grand de 0.6757%. De plus, il n’a pas réussi à identifier une seule instance appartenant à la troisième catégorie tandis que l’arbre de décision a correctement identifié 33 des 66 instances dans cette catégorie et Naives Bayes 34 des 66.

SVM est la méthode qui a obtenu les meilleurs résultats. Le SVM performe mieux que l’arbre de décision en obtenant un taux d’instances correctement classifiés plus grand de 5.2787% (89.6115% pour le SVM contre 84.3328% pour l’arbre). Contrairement à l’arbre et à Naives Bayes, le SVM a performé le mieux avec une fenêtre de 3 mots.