# Chapitre V

# Réalisation

# V.1 Introduction

Après avoir présenté l'architecture de notre système dans le chapitre précédent et l'illustration que ce système fait des opérations d'apprentissage à partir des données des clients et produit deux modèles. Un pour l'acquisition et le second pour la rétention, et produit également un ensemble de catégories obtenues par la segmentation.

Dans ce chapitre nous allons d'abord expliquer les outils utilisés dans la réalisation de notre système, l'environnement de développement de, le langage de programmation utilisé dans le développement. Sans oublier la structure de données nécessaire pour appliquer se système. A la fin, nous allons analyser les résultats obtenus par notre système.

# V.2 Outils utilisés

## V.2.1 Packages

### V.2.1.1 LIBSVM

LIBSVM est une bibliothèque développée par Chih-Chung Chang et Chih-Jen Lin, deux chercheurs du département informatique de l'université national de Taiwan, depui l'année 2000. Cette bibliothèque est développée pour implémenter dans la résolution des problèmes lieé a la classification avec l'utilisation des machine a vecteur support les machine a vecteur que ce soit la classification mono-class, binaire, multi-classes et même pour la régression.

Elle est disponible en plusieurs langages de programmation comme C++, java, PHP, C#. Et compatible avec diverses plateformes logicielles comme R, MATLAB, Perl, Ruby, Weka, CLISP, CUDA). Elle peut utiliser sur l'environnement de Windows ou linux. Disponible sur le site http://www.csie.ntu.edu.tw/~cjlin/libsvm/

### V.2.1.2 Fonctionnalité de LIBSVM

Nous pouvons faire un appelle aux fonctionnalités de cette bibliothèque en ligne de commande, comme ci-dessous :

**« svm-scale »** pour la normalisation de données.

**« svm-train »** pour l'apprentissage.

**« svm-predict »** génère les classes prédites avec l'utilisation du modèle obtenu.

LIBSVM donne la possibilité d'entrer et de modifier les paramètres nécessaires pour obtenir un résultat diffusante. Dons ci-dessous (tableau 5.1) et (tableau 5.2) une liste de ces paramètres.

| **Noyau** | |
|---|---|
| **Type de noyau à utiliser** | **-t kernel_type**<br>**0** => fonction linéaire = u'*v<br>**1** => fonction polynomiale = (gamma*u'*v + coef0)^degré<br>**2** => fonction Radiale de formule = exp(-gamma*\|u-v\|^2)<br>**3** => une fonction sigmoïde = tanh(gamma*u'*v + coef0)<br>**4** => création d'un noyau. |
| **Paramètres des fonctions noyau** | **-d degree** => degré de la fonction noyau, 3 par défaut<br>**-g gamma** => gamma de la fonction noyau, 1 par défaut<br>**-r coef0** => coef0 de la fonction noyau, 0 par défaut |

Tableau 5.1 : **Paramètres de noyau**

| **SVM** | |
|---|---|
| **Type de SVM à utiliser** | **-s svm_type**<br>**0** => classification de type C-SVM (par défaut)<br>**1** => Pour une classification de type nu-SVM<br>**2** => Pour une classification de type one-class-SVM<br>**3** => Régression de type epsilon-SVM<br>**4** => Régression de type nu-SVM |
| **Paramètres dépendants du type SVM** | **-c cost** => paramètre C, pour la pénalisation de l'erreur.<br>**-wi weight** => changer C à weight*C<br>**-n nu** => nu du type nu-SVC, One-class-SVM et nu-SVR, 0.5 par défaut<br>**-p epsilon** => epsilon de la fonction de perte (Loss Function) pour le type epsilon-SVR, 0.1par défaut |
| **Paramètres SVM généraux** | **-m cachesize** => la taille mémoire (Mbits) allouée au noyau, 100 par défaut<br>**-e epsilon** => critère d'arrêt de l'apprenant, 0.001 par défaut<br>**-h shrinking** => activer/désactiver l'heuristique de shrinking, 1 par défaut<br>**-b probability-estimates** => activer/désactiver l'apprentissage avec probabilité, 0 par défaut à<br>**-v n** => spécifique au mode validation-croisée le n est le nombre de sous-divisions de l'ensemble d'apprentissage (n-fold), désactivé par défaut.<br>**-q** => Mode silencieux, pour omettre l'affichage des messages, 0 par défaut |

Tableau 5.2 : **Paramètres de SVM**

## V.2.2 Environnement de programmation

### V.2.2.1 Eclipse

Eclipse IDE est un environnement de développement intégré libre et extensible, permettant potentiellement de créer des projets de développement mettant en plusieurs langages de programmation.

Eclipse est destiné principalement aux applications Java. Il utilise plusieurs modules appelés « plug-ins » dans son architecture. Comme « WindowBuilder » qui conçu par Google et permet de créer des applications graphique sous Java sans

dépenser beaucoup de temps. Permet de basculer facilement entre l'interface graphique et son code Java. Nous pouvons facilement ajouter des contrôles en utilisant le glisser-déposer, ajouter des gestionnaires d'événements, modifier diverses propriétés des contrôles à l'aide d'un éditeur de propriété et bien plus encore.

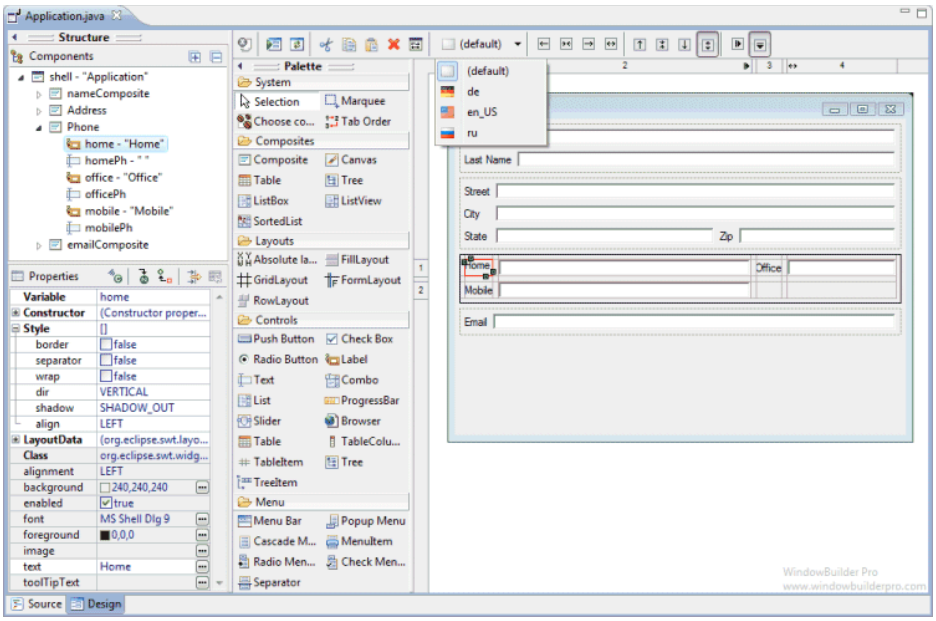


Figure 5.1 : Interface de WindowBuilder

### V.2.2.2 Langage java

Java est un langage de programmation orienté objet. Il permet de créer des logiciels compatibles avec Windows, Linux, Macintosh, Solaris. Java est un langage indépendant de l'architecture parce qu'il Le bytecode généré n'est pas lié à un système d'exploitation en particulier. De ce fait, il peut être interprété très facilement sur n'importe quel environnement disposant d'une JVM. Enfin est un langage simple à prendre en main et nous pouvons intégrer la bibliothèque LIBSVM dans notre application.

### V.2.2.3 Plateforme Weka

Weka est un logiciel open source publié sous la licence GNU, se compose d'une collection d'algorithmes d'apprentissage pour les tâches data mining. Les algorithmes peuvent être soit appliquées directement à un ensemble de données ou appelés à partir d'un code propre de Java. Weka contient des outils pour le prétraitement des données, classification, régression, clustering, règles d'association, et la visualisation. Il est aussi bien adapté pour le développement de nouveaux programmes d'apprentissage.

Dans ce qui suite nous allons utiliser weka pour faire des expérimentations sur les donnés d'apprentissage afin d'obtenir les modèles qui nous allons intégrer dans notre application.


Figure 5.2 : Interface de Weka

## V.2.3 Tests et résultats

### V.2.3.1 Tests

Les données des clients sont tirés du site « http://archive.ics.uci.edu/ml/datasets/Bank+Marketing » , c'est un site qui contient une collection de bases de données pour plusieurs domaine, destiné pour utilisé par la communauté de l'apprentissage automatique et l'analyse empirique des algorithmes d'apprentissage.

**Pour l'acquisition des clients**

- Nous avons utilisé la base de données nommé « bank.csv ».

- Cette base contient 4521 enregistrement avec 5 attribut (âge, job, marital, éducation et la classe client).

- Nous avons converti ce fichier de type csv à un fichier arff supporté par weka.

- Nous avons fait l'opération de l'apprentissage sur ces données avec les paramètres suivants.

- Type d'SVM et (one-classe SVM).

- La fonction noyau utilisé est la fonction polynomial son degré est 3.

**Pour la rétention des clients**

- Nous avons utilisé la base de données nommé « indices.csv ».

- Cette base contient 576 enregistrements avec 4 attributs (le retrait moyenne par jour, moyenne par semaine, moyenne par mois et la classe client ou ex-client).

- Type d'SVM et (C-SVC).

- La fonction noyau utilisé est la fonction RBF son degré est 3.

**Pour la segmentation des clients**

- Nous avons utilisé la même base de données qui utilisé pour l'acquisition.

- Nous avons utilisé l'algorithme SimpleKmeans de weka pour faire la segmentation.

- Nombre de clusters est 2.

- La fonction de distance utilisé est la fonction euclidienne $D_n(x_i, x_j) = \sqrt{\sum_{k=1}^{n}(x_{ik} - x_{jk})^2}$.

### V.2.3.2 Résultats

**Pour l'acquisition des clients**

- Nous avons trouvé les résultats suivant :

- Les enregistrements correctement classé est 50.14% => donc le modèle de classification est incapable d'identifier 49.86% des clients.

- Sachant que nous avons changé plusieurs fois les paramètres d'apprentissage mais cette résultat est la bonnes que nous la trouvé.

**Pour la rétention des clients**

- Nous avons trouvé les résultats suivant :

- Les enregistrements correctement classé est 52.43%.

- Les enregistrements incorrectement classé est 47.57%.

- Comme l'acquisition des que nous avons changé plusieurs fois les paramètres mais cette résultat ne change pas beaucoup.

**Pour la segmentation des clients**

- Le résultat de la segmentation c'est une deux cluster. Le premier cluster (44%) consiste les clients qui sont âge entre 33 et 43, travaillent en secteur de management, mariés et sont niveaux est un niveau universitaire.

- Le deuxième cluster (56%) consiste les clients qui sont âge entre 31 et 32, des ouvriers, mariés et sont niveaux est un niveau secondaire.

## V.3 Conclusion

Dans ce chapitre, nous avons appliqué les techniques de data mining pour la GRC dans les banques. Malgré que les taux de reconnaissances obtenus n'aient pas été élevés, nous croyons qu'ils peuvent aider considérablement la banque et la débarrasser d'un énorme effort manuel. Néanmoins, ces résultats peuvent être améliorés en utilisant des bases de données plus importantes et en multipliant le nombre de caractéristiques des clients et leurs comportements puisque l'âge, la fonction, la situation familiale et le niveau d'éducation uniquement ne suffisent pas pour caractériser un client pour une banque.

Pour la segmentation des clients nous avons trouvé que la technique de k-means est appropriée pour catégoriser les clients en se basant sur l'âge, la fonction, la situation familiale et le niveau d'éducation. L'administrateur peut déterminer à partir de ces résultats des politiques spécifiques pour traiter les catégories de ses clients, ainsi que d'attirer de nouvelles catégories de clients à la banque.