



Republique du Senegal
Un Peuple - Un But - Une Foi
Ministère l'enseignement supérieur de la recherche et
de l'innovation
École Polytechnique de Thiès
B.P.A 10 Thiès
Tel: (221) 76 223 61 74 - Fax: (221) 33 951 14 67

Département Génie Informatique et Télécoms
Deuxième année cycle DIC
2022-2023

Rapport projet Modélisation Stochastique

Prédicteurs de délai pour le centre d'appel du VANAD

Travail réalisé par Mouhamadou Naby DIA & Abdou Sakho

Professeur: M Thiongane

I. Présentation des données

Nous disposons de données qui nous proviennent du centre d'appel du **VANAD Laboratories** (Rotterdam, aux Pays-Bas). Les données sont collectées sur la période d'une année, de **janvier 2014 à décembre 2014**.

Ce centre d'appels est multi-compétences (ie. reçoit plusieurs types de service), fonctionne de **8H à 20H** du **lundi au vendredi**, reçoit au total **27 types d'appels** (types de service), et compte **312 agents** au total. Chaque agent possède un ensemble de compétences, qui correspond à l'ensemble des types d'appels qu'il peut servir. Nous disposons de deux jeux de données distincts: des **données sur les appels entrants**, et des **données sur les activités des agents**.

Les données sur les activités des agents comprennent les informations suivantes : l'indice de l'activité, l'heure de début de l'activité, l'heure de fin de l'activité et l'identification de l'agent. La durée des activités est calculée en soustrayant l'heure de début de l'activité de l'heure de fin de l'activité.

Mais ces données sur les activités des agents ne concernent pas ce présent travail. Nous nous concentrerons surtout sur les données des appels entrants.

Le jeu de données des d'appels qui contient les informations suivantes : l'heure d'arrivée de l'appel, l'heure à laquelle l'agent commence à traiter l'appel, l'heure de départ, le type de service demandé par l'appelant, l'identité de l'agent qui traite l'appel, etc.

date_received	queue_name	agent_number	answered	consult	transfer	hangup	call_types	date
2014-01-02 08:03:21	30175	6935	2014-01-02 08:03:22	NULL	NULL	2014-01-02 08:05:54	0	2014-01-02
2014-01-02 08:04:37	30560	6940	2014-01-02 08:04:38	NULL	NULL	2014-01-02 08:09:49	1	2014-01-02
2014-01-02 08:06:26	30172	6968	2014-01-02 08:06:27	NULL	NULL	2014-01-02 08:15:17	2	2014-01-02
2014-01-02 08:08:07	30175	9424	2014-01-02 08:08:08	NULL	NULL	2014-01-02 08:10:37	0	2014-01-02
2014-01-02 08:08:26	30172	8076	2014-01-02 08:08:27	NULL	NULL	2014-01-02 08:14:59	2	2014-01-02
2014-01-02 08:09:10	30175	9423	2014-01-02 08:09:11	2014-01-02 08:12:56	NULL	2014-01-02 08:14:14	0	2014-01-02

Figure 1: Exemple de journal d'appel

Les colonnes qui nous intéressent le plus sont:

- ❖ **date_received**: date et heure d'arrivée de l'appel;
- ❖ **queue_name**: l'ID du type de service demandé par l'appelant;
- ❖ **agent_number**: l'ID de l'agent qui traite l'appel
- ❖ **answered**: date et heure à laquelle l'appel a été pris(**NULL** sinon);
- ❖ **hangup**: date et heure à laquelle l'appel a pris fin;
- ❖ etc.

Le mécanisme de routage fonctionne comme suit. Lorsqu'un client appelle, il interagit avec l'IVR (unité de réponse vocale interactive) en utilisant son clavier pour choisir le type d'appel. S'il y a un agent disponible ayant les compétences nécessaires pour traiter ce type d'appel, le client (l'appel) est alors dirigé vers l'agent le plus longtemps inactif parmi les agents disponibles; sinon, il attend dans une file d'attente invisible. Les appels de cette file d'attente sont traités dans l'ordre FCFS (premier arrivé, premier servi).

II. Que cherchons nous à faire

Des études réalisées sur les centres d'appels (les systèmes de service en général) ont montré que le *fait d'informer les clients sur leur temps d'attente dès leur arrivée permet d'augmenter la satisfaction des clients sur la qualité de service perçue, et réduit considérablement le nombre d'abandons*. Ceci combiné avec la proposition de rappel si le temps d'attente estimé est très long réduit davantage les abandons et augmente la satisfaction des clients.

Dans ce travail, nous voulons tester l'efficacité de certains prédicteurs présenté dans *Thiongane et al. (2023) "A New Delay History Predictor for Multi-skill Call Center"*. en utilisant les données du VANAD pour l'année

2014. Nous allons utiliser les prédicteurs LES, Avg-LES, AvgC-LES et WAvgC-LES et ANN (Artificial Neural Network). LES, Avg-LES, AvgC-LES et WAvgC-LES sont des prédicteurs qui utilisent l'historique des attentes des clients passés pour prédire les temps d'attente des nouveaux clients alors que ANN est un prédicteur qui utilise l'apprentissage machine.

Pour prédire le temps d'attente W d'un nouveau client de type T , le prédicteur ANN peut utiliser toutes les informations dont pensons avoir une influence sur le temps d'attente d'un client. Bien sûr il faut que ces informations soient observables au niveau système à l'arrivée d'un client. Nous pouvons considérer un ensemble d'informations sur chaque client qui arrive sur le call center. Toutes ces informations forment un vecteur x qui décrit l'état du système à l'arrivée d'un nouveau client. En d'autres termes, à l'arrivée d'un nouveau client, on observe l'état du système x qui est passé en entrée à la fonction F pour le calcul du temps d'attente prédit pour ce client ($F(x)$ représente alors la prédiction du temps d'attente W).

Nous souhaitons trouver la fonction F qui minimise les erreurs de prédictions.

Dans les données toutes les informations qui décrivent le vecteur x ne sont pas observées. Le premier challenge est de recouvrir ces données manquantes. Pour ce faire, nous allons faire un replay des journées du centre d'appels (simulation par retraitage). Nous allons utiliser le package simevents de la librairie SSJ. A la fin de la simulation, nous allons avoir un jeu de données $D = \{z_1, \dots, z_n\}$ avec $z_i = (x_i, w_i)$ où $x_i \equiv$ "état du système" et $w_i \equiv$ "temps d'attente" du client i . Nous allons utiliser 80% des données pour apprendre la fonction de prédiction F avec des méthodes d'apprentissage machine et mesurer les performances de ce dernier avec les 20% des données restantes.

Mais avant tout cela, il est nécessaire de faire un petit preprocessing sur les données du call center.

III. Le Preprocessing

Dans cette partie, il sera question:

- ❖ **d'importer le dataset sur R:** dans ce travail, nous nous intéressons aux 12 fichiers [calls-2014-01.csv](#) à [calls-2014-12.csv](#) qui contiennent les informations sur les appels entrants.
- ❖ **faire le mapping entre les 27 types de services** dans les données et les chiffres de 0 à 26 dans une nouvelle colonne `call_types`. Ce sera cette colonne qui sera utilisé comme Type sur le dataset final généré pour les customers.

30175	30560	30172	30066	30176	30179	30173	30511	30181
30519	30241	30174	30325	30363	30334	30180	30236	30177
30178	30584	30598	30518	30170	30694	30729	30747	30764

Figure 2: Les 27 types de services existants

- ❖ **suppression des entrées:** supprimer toutes les entrées qui ont un hangup à NULL.
- ❖ **génération des données:** générer pour chaque mois, les données des appels par jour
Exemple: pour le fichier [calls-2014-01.csv](#), nous allons générer les fichiers [2014-01-01.csv](#) à [2014-01-31.csv](#) avec des jours manquants pour les weekend ou les jours non ouvrables.

IV. La génération du dataset

Cette partie a été faite avec Java en utilisant la librairie SSJ. Donc comme dit tantôt, nous allons faire une replay de toutes les journées de l'année et créer un dataset où pour chaque entrée nous aurons un customer avec toutes les informations pertinentes sur lui, les valeurs de prédicteurs LES, Avg-

LES, AvgC-LES et WAvgC-LES et son vrai waiting time calculé sur la base des données.

1. La classe Customer

Cette classe va contenir toutes les informations sur le customer: le type de service auquel il a accédé, les tailles des différentes files d'attentes, son heure d'arrivée, sa durée de service, le nombre de serveurs, une variable booléenne qui indique s'il est servi ou non, les valeurs de différents prédicteurs LES, Avg-LES, AvgC-LES et WAvgC-LES et son temps d'attente réel. Et ce sont ces informations qui vont être écrit sur chaque entrée du dataset.

2. La classe OneDaySimulation

La classe OneDaySimulation contient le code pour faire la retraçage d'une journée. Elle contient une méthode createCustomers qui va parcourir le fichier csv et créer les customers pour chaque entrée, prévoir leur événement arrivée(Arrival: arrivé dans la file d'attente) qui aboutit aux événements départ(Departure: départ de la file d'attente) et fin de service (EndOfService: fin de service du client). Ce différents événements vont permettre de définir les différentes informations sur le customer. Et à la fin de service de chaque customer on écrit les informations sur un fichier csv.

3. La classe Simulation

La classe Simulation va permettre tout simplement d'automatiser le processus de création du dataset. Elle va permettre de créer le fichier csv de base, parcourir tous les mois et pour chaque mois, faire la simulation de toutes ses journées et donner un fichier csv de sortie contenant tout les customer.

A la fin, on obtient un fichier csv pour chaque mois et donc de [1.csv](#) à [12.csv](#) que nous allons utiliser pour estimer les prédicteurs basés sur les historiques et aussi entraîner un prédicteur basé sur les ANN.

V. Le modèle ANN

Pour cette partie, nous allons importer les 12 fichiers csv et les combiner en un seul dataset complet. Ce dernier contient au total 1,542,662 entrées avec 37 colonnes.

Et après avoir évaluer la distribution des types de services, on s'est rendu compte du fait que les 8 services les plus demandés constituent 98,8% du dataset complet.

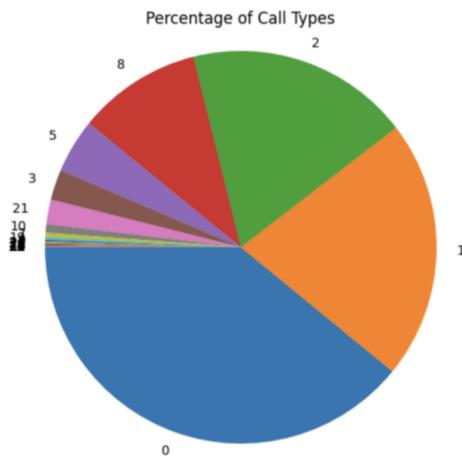


Figure 3: Répartition des types de services demandées

Et comme demandé, nous nous concentrerons sur les 5 types les plus demandés qui constituent 93,589% du dataset soit 1,443,763 entrées et 15 colonnes.

Puis nous avons regardé la distribution des données. Et nous avons fait un Min-Max feature scaling pour ramener la distribution de tous les features entre 0 et 1.

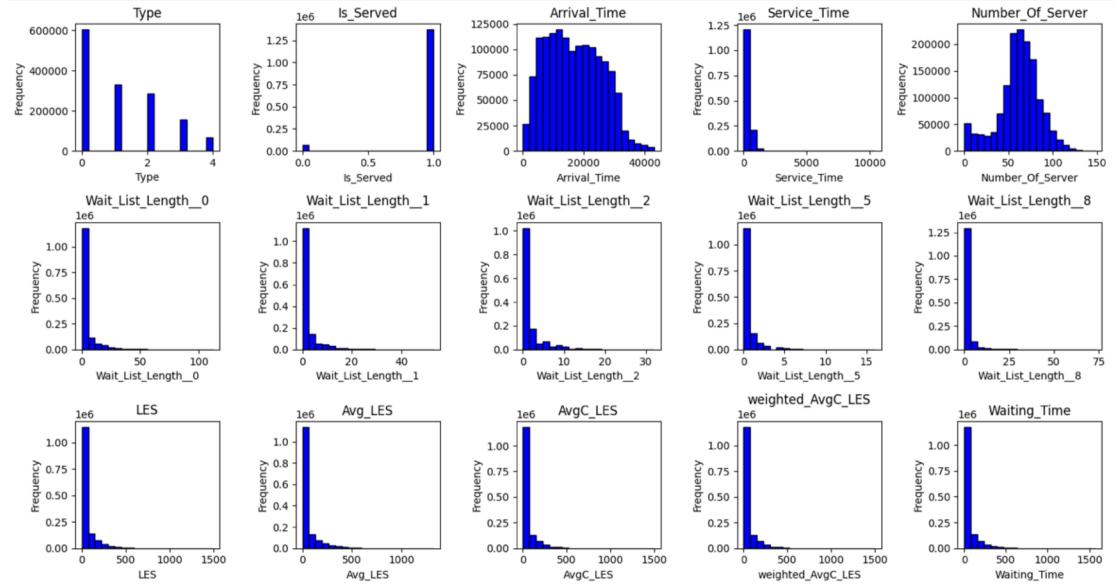


Figure 4-a: Distribution before feature scaling

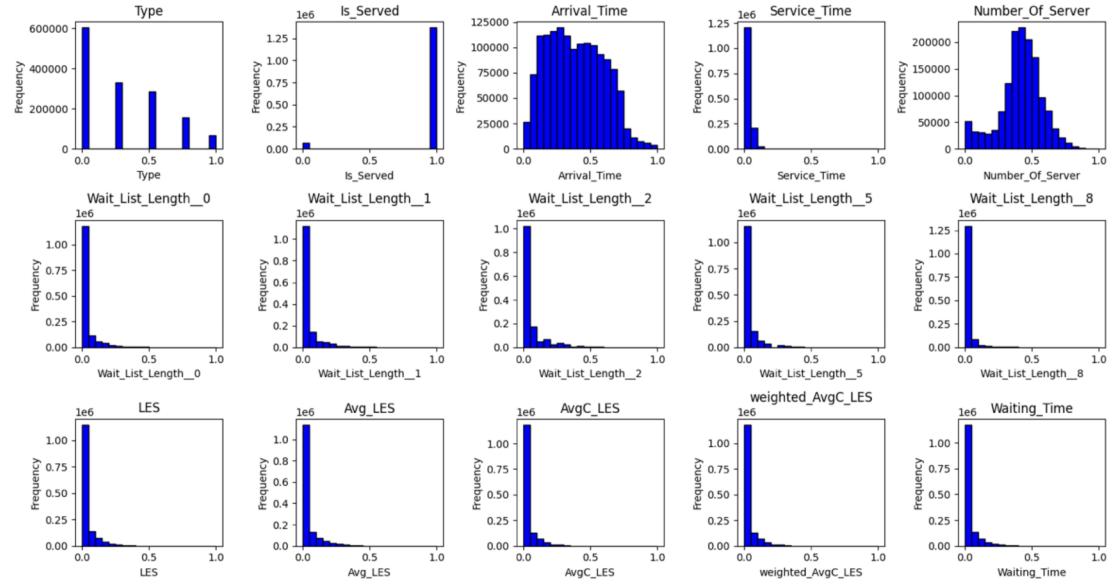


Figure 4-b: Distribution after feature scaling

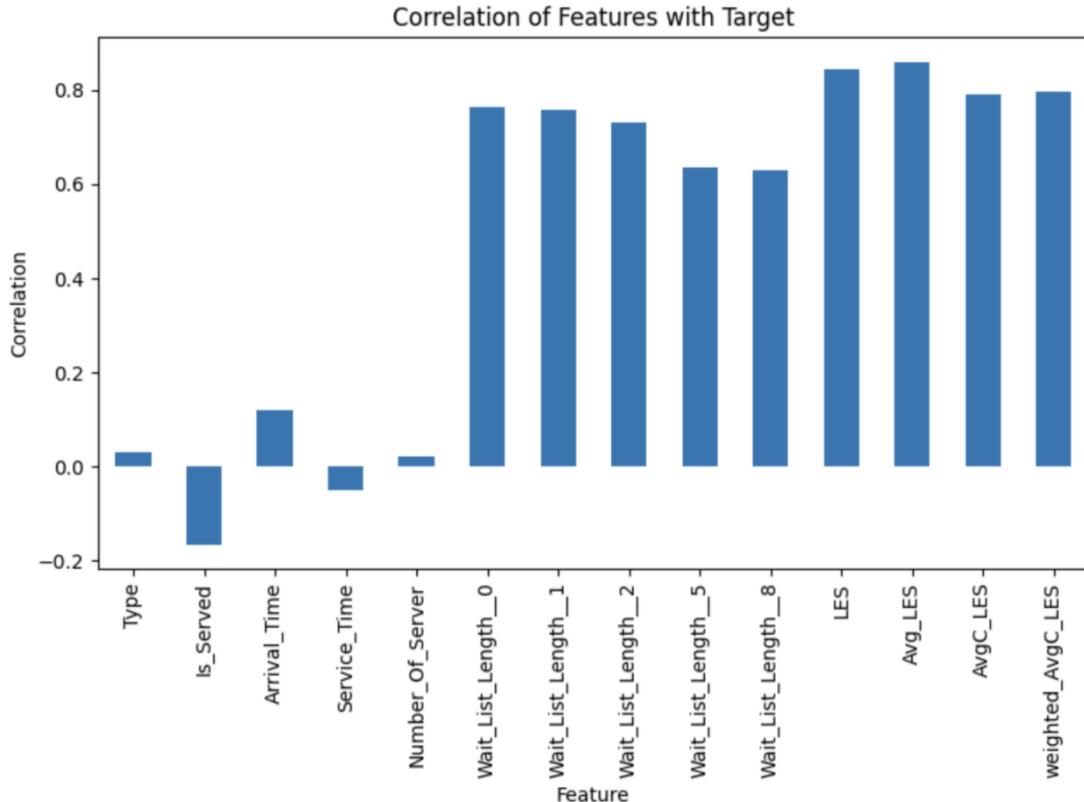


Figure 5: Correlation entre les features et les target

Après tout ce preprocessing sur les données, nous avons séparé les données en 80% pour le train set et 20% pour le test set.

Notre modèle de réseaux neurones contient 3 couches, une couche Dense avec 64 units avec une fonction d'activation ReLU, une autre couche Dense avec 32 units avec une fonction d'activation ReLU également et une dernière couche Dense avec un seul neurone et une fonction d'activation linéaire. Nous avons utilisé l'Adam optimizer, MSE comme fonction de perte et pour les metrics nous avons opté pour le RMSE et RRASE. Le modèle a été entrainé sur 10 epochs.

VI. Les résultats finaux

La figure 7, présente le RRMSE pour les cinq types d'appels et pour divers prédicteurs. Nous avons pris $N = 10$ pour Avg-LES, $N_k = 100$ pour AvgC-LES, et $\alpha = 0,2$ pour WAvgC- LES. En plus des prédicteurs DH, nous utilisons également un prédicteur ANN. Le prédicteur ANN fonctionne mieux, mais il nécessite une phase d'apprentissage très coûteuse et comportant de nombreux paramètres.

$$\text{RRMSE}(\%) = 100 \times \sqrt{\frac{\sum_{i=1}^I (c_i - \hat{c}_i)^2}{\sum_{i=1}^I c_i^2}}$$

Figure 6: Relative Root Mean Squared Error

	L_{ES}	Avg_L_{ES}	AvgC_L_{ES}	W_AvgC_L_{ES}	ANN
0	44.502269	51.609690	50.548024	50.122990	24.079708
1	45.593803	54.424435	53.330139	53.185670	26.366525
2	53.253891	61.611710	61.086576	60.666761	28.100604
3	49.715647	56.453244	54.785192	53.984475	25.507648
4	63.993984	68.378078	66.910578	66.953618	35.187699

Figure 7: RRMSE de prédicteurs de délai

Un lien d'invitation en tant que collaborateur vous sera envoyé sur le repo github pour que vous puissiez accéder aux détails de l'implémentation de ce travail: <https://github.com/dnaby/VANAD-Call-Center.git>