Rida Adam
Dioukhane Bara
Doussau Guillaume
Wang Xindi

Mai 2020

PROJET GM-MI SEMESTRE 2
Application COVID-19

ING1-GMI02

EISTI

Table des matières

1 Remerciements **2**

2 Introduction **3**

3 Rapport technique **4**
3.1 Fonctionnalités de l'application 4
3.1.1 Mode d'emploi 4
3.1.2 Réalisation technique 10
3.1.3 Problématiques, limites et choix 13

4 Data Mining **15**
4.1 Statistiques descriptives 15
4.2 Prédictions des survivants 17
4.3 Importance des variables et ACP 18
4.4 Évolution du nombre de malades dans un région données 20

5 Organisation du travail durant le projet **21**
5.1 Répartitions des tâches 22
5.2 Point sur le travail en équipe 22
5.3 Améliorations possibles du travail 22

6 Conclusion **23**

1 Remerciements

Nous tenons à exprimer notre gratitude pour Mme Zaouche et Mme Oudinot, pour leur écoute attentive, soutien, conseils et encadrement tout au long de la réalisation de ce projet.

Nous tenons aussi à souligner la capacité d'adaptation de l'équipe pédagogique de l'EISTI qui a su malgré les contraintes et les imprévus gérer avec brio la crise et l'enseignement a distance. Cela nous a permis de continuer a bénéficier d'un enseignement de qualité a distance.

2 Introduction

Dans le cadre de notre formation d'ingénieurs en génie mathématiques et en informatique nous avons été amenés a développer une application concernant un sujet qui n'en fini plus de faire les unes des médias : un outil d'aide a la décision sur la crise du coronavirus. En effet dans cette période de crise sanitaire les restrictions dues au COVID19 imposent à certaines communes de France de rester confinées afin d'enrailler la propagation du virus. De ce fait le déplacement entre communes non-confiné est devenu plus complexe, notre objectif est donc de faciliter cette tâche aux habitants et de fournir un outil d'aide a la décision aux professionnels de santé. Nous proposons donc d'une part un logiciel (programme en Java) qui permet de gérer et de renseigner des données médicales puis d'en tirer des statistiques. Nous avons d'autre part réalisé une étude prédictive sur l'évolution de l'épidémie grâce a des modèles mathématiques et des techniques de Data Mining

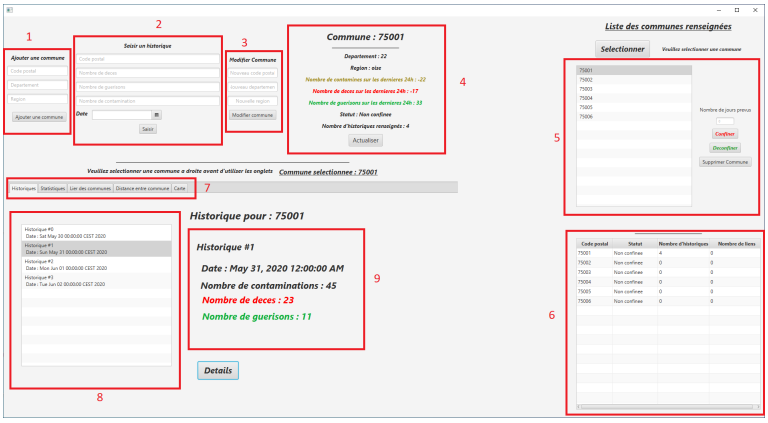
3 Rapport technique

3.1 Fonctionnalités de l'application

3.1.1 Mode d'emploi

Lors du lancement de l'application, l'écran A s'affiche en premier. Nous avons volontairement implémenté 6 communes depuis le code source pour gagner du temps de saisie pour les tests.

Écran A :



1. Ajout commune
Il est possible d'ajouter une commune en saisissant ses attributs. Il suffit de cliquer sur le bouton ajouter pour l'intégrer au modèle.

2. Saisie d'un historique
Il est possible de saisir un historique pour une commune précise. Modification d'une commune Cette section sert comme son nom l'indique a redéfinir les attributs d'une commune.

3. Modification d'une commune
Cette section sert comme son nom l'indique a redéfinir les attributs d'une commune.

4. Détails d'une commune
Cette section affiche les attributs détaillés d'une commune. Pour mettre a jour cette section, il faut sélectionner dans la section 5 une commune puis mettre a jour en cliquant sur Actualiser ou Sélectionner.

5. Liste des communes et confinement/suppression
C'est dans cette partie que l'on peut confiner/déconfiner des communes ou les supprimer. Il suffit de cliquer sur une commune puis sur le bouton de l'action a effectuer.

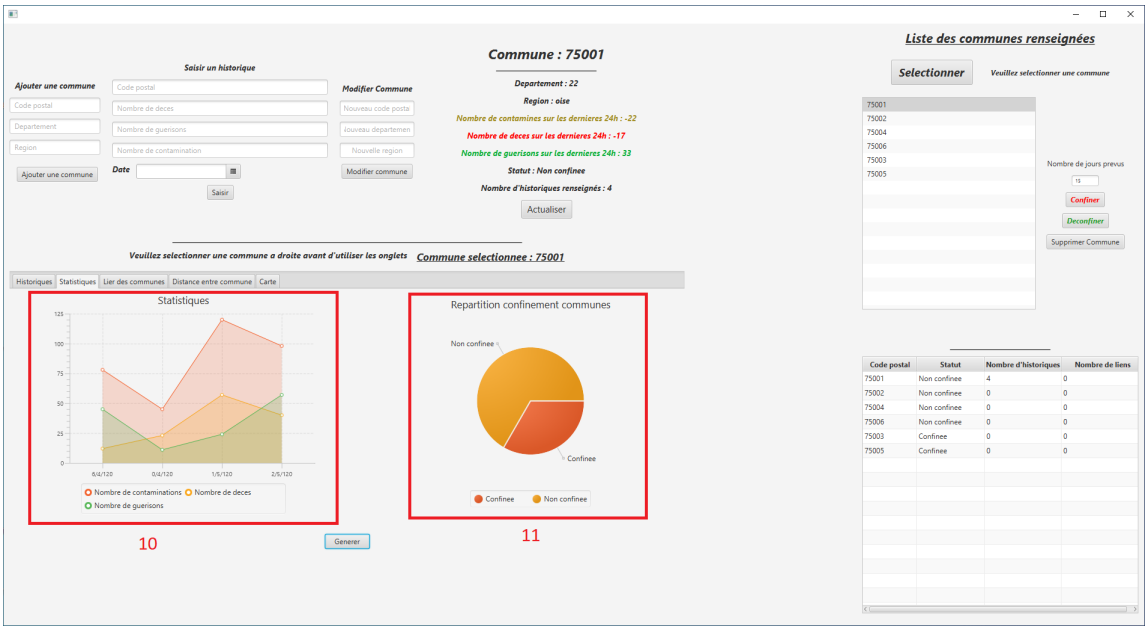
6. Liste détaillées des information renseignées par commune
Cette liste donne des informations sur le nombre de communes liées a chaque commune ainsi que le nombre d'historique.

7. Onglets de navigation
Il s'agit des onglets qui permettent de naviguer entre les différentes parties du projet.

8. Liste des historique pour une commune
Il s'agit de la liste des historiques renseignes pour une Commune précise. Pour l'afficher il faut sélectionner une commune puis appuyer sur le bouton Sélectionner.

9. Détails de l'historique
Il est possible d'accéder au statistiques détaillées de chaque historique en cliquant dessus puis sur le bouton Details.

Écran B :



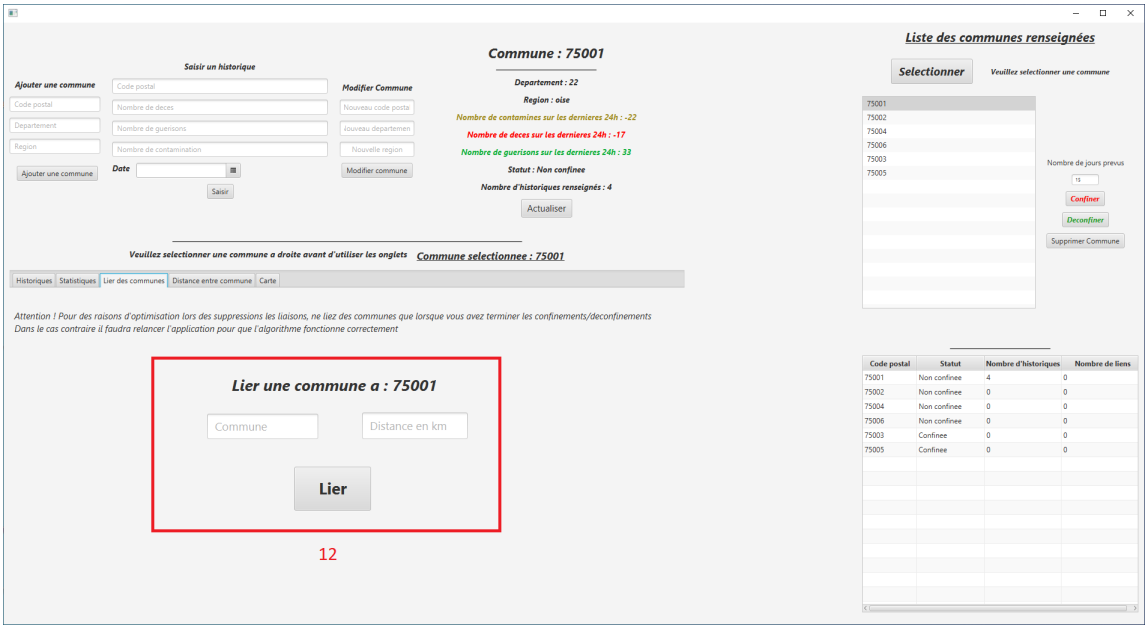
10. Statistiques locales

Les statistiques locales sont générées uniquement si une commune est sélectionnée. Il suffit ensuite d'appuyer sur le bouton Générer. Nb : Il faut régénérer le graphe en cas d'ajout de données.

11. Statistiques globales

Ces statistiques s'affichent indépendamment de toute sélection, il suffit d'appuyer sur Générer pour l'afficher nb : Il faut régénérer le graphe en cas de confinement ou de déconfinement.

Écran C :

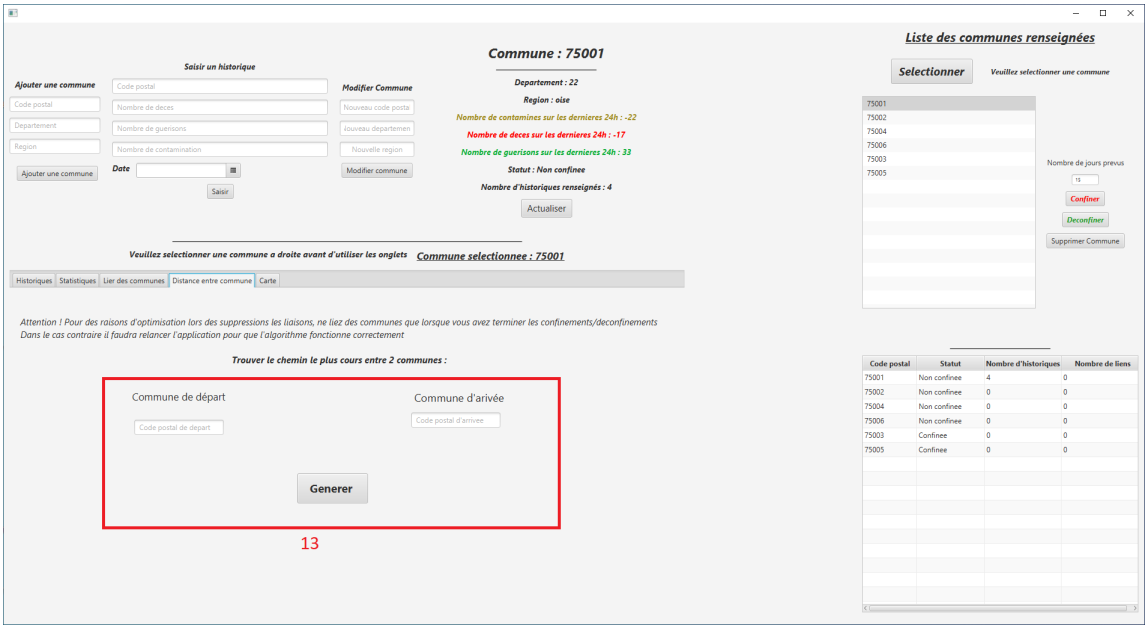


12

12. Liaison de commune

Une fois qu'une commune est sélectionné, il est possible de lui lier une commune. Attention il ne faut plus confiner ou déconfiner après avoir commencé a lier des communes.

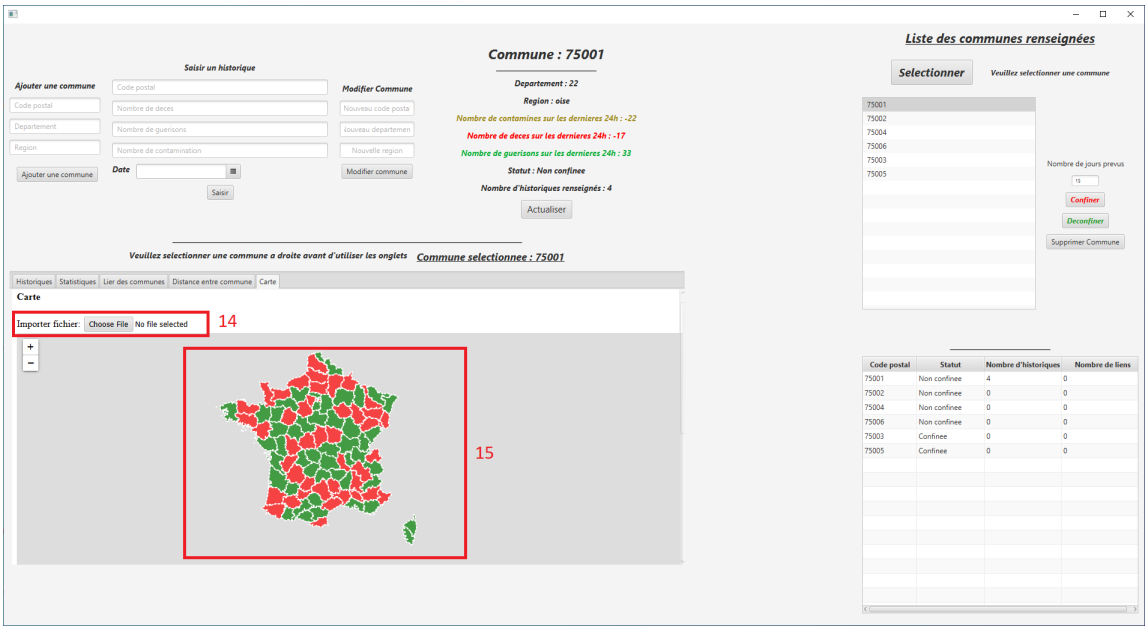
Écran D :



13. Chemin optimal

Permet de générer le chemin optimal entre deux communes. Attention pour des raisons d'optimisation, une génération est possible par instance. Il faut donc relancer l'application après une génération.

Écran E :



14. Carte

La Carte (facultative) affiche des couleurs aléatoirement pour le moment. Un champ est présent pour importer un fichier (généré par le code Java) contenant les coloration pour la carte. Ce fichier prend en compte l'instance courante.

3.1.2 Réalisation technique

Une documentation Javadoc est disponible avec les sources

a. Conception à partir du modèle de donnée

Inspiration : Le modèle de donnée et les instructions fournies nous a été grandement utile pour poser un cadre à notre travail et guider notre réflexion. Elles nous ont fait gagner énormément de temps de conception en nous permettant de bâtir un modèle cohérent rapidement.
Compte tenu de nos compétences respectives, nous avions tous les éléments en main pour mener à bien ce projet (sauf la partie Carte qui s'est avérée hors de notre portée en terme de compétence).
Les classes principales se résument donc à :

- Modèle : Cela pourrait s'apparenter au pays, on peut ajouter ou supprimer des communes puis effectuer des actions propres à un ensemble de commune.

- Commune : Cette classe se divise en deux sous classes (Confinée et non Confinée), il s'agit de caractériser une commune ainsi que toutes les actions qui lui sont propre (Par exemple les historiques).

Principales fonctionnalités retenues : Les fonctionnalités que nous avons décidé de retenir sont les suivantes :

- Modèle
  - Ajout/Suppression de commune
  - Génération d'un fichier "couleur" contenant sur chaque ligne une commune puis une couleur corrélée à son nombre de contamination (3 niveaux de coloration avec des seuils modifiables - green/orange)
  - Confinement/Déconfinement d'une commune : Lors de sa création, une commune est considérée comme non Confinée par défaut.

- Commune
  - Génération de statistiques

Fonctionnalités supplémentaires implémentées : La majorité des fonctionnalités supplémentaires se trouvent dans la classe Commune. En effet, les fonctionnalités ajoutées dans la classe Modèle ont pour but de faciliter la gestion de l'interface graphique. Concernant la classe Commune, nous avons ajouté en attribut de celle-ci : une liste d'historique. Cet attribut donne la possibilité de créer un historique a une date précise pour une commune et de le sauvegarder dans l'objectif de les exploiter statistiquement. Nous avons entre autre implémenté la possibilité de lier une Commune à l'instance courante pour pouvoir utiliser l'algorithme de Dijkstra.

Fonctionnement du modèle : Le fonctionnement est relativement simple. Une fois le modèle initialisé, il est possible d'ajouter des communes puis de réaliser toutes les actions décrites ci-dessus. Le modèle permet de stocker les différentes communes.

b. Module historique / Statistiques

Format de données des historiques : Comme dit précédemment, chaque commune possède une liste d'historique. Un historique est composé des nombres de décès/guérisons/contamination de la journée en question. Les ajouts se font grâce à une méthode précise. Celle ci écrase l'historique si jamais un historique a déjà été saisi à cette date.

Attributs "24h" de la classe Commune : Vous l'avez sûrement déjà remarqué, mais notre classe Commune comporte des attributs ayant pour fin "24h". Il s'agit des données statistiques des dernières 24h. Pour être plus rigoureux et pour conserver de la consistance dans ces attributs nous avons fait le choix d'élargir ces statistiques. Autrement dit, ces attributs sont actualisés grâce aux données recueillies dans les deux derniers historique et ce quel que soit l'écart temporel entre les deux (toujours supérieur à 24h cependant). L'intérêt est de toujours avoir des données à afficher même si les saisies ne se sont pas toutes faites à un intervalle rigoureux de 24h.

Statistiques représentées : Après plusieurs débats et pour ne pas chevaucher avec la partie Data Mining du projet, nous avons décidé de représenter deux graphes.

Le premier est un diagramme circulaire représentant la proportion de communes confinées par rapport aux non confinées. La récupération de cette information s'est faite grâce à la taille des tableaux du modèle. (Un des intérêts du triple tableau, cf : Problématiques, limites et choix)

Le deuxième choix était un graphe "simple" avec en abscisse les date et en ordonnées les nombres de décès/guérisons/contaminations. Nous avons superposé les 3 courbes pour mieux les faire contraster et avoir une vision direct des 3 indicateurs les plus pertinents. Les données sont récupérées en parcourant les tableaux d'historique pour une commune donnée.

Remarque : Le diagramme circulaire traite des données à l'échelle globale (Modèle) tandis que le diagramme "simple" exploite des données locales (Commune).

c. Implémentation de l'algorithme de Dijkstra

Liens entre communes et vertex : Après plusieurs essais non fructueux, nous avons implémenté et adapté avec succès une version de l'algorithme de Dijkstra assez simple à prendre en main. Chaque Commune possède un attribut "vertex" qui correspond à un noeud dans le graphe de Dijkstra. Par conséquent chaque commune créée, génère dans sa création un vertex.
Ces vertex possèdent plusieurs attributs notamment un attribut "Edge" qui correspond aux liens du vertex.

Nous avons donc pu créer une méthode "LieerCommune2" (car 2eme version d'une première non fonctionnelle) qui ajoute un vertex d'une commune au vertex d'une autre.

La particularité fut cependant de devoir s'assurer d'une liaisons dans les deux sens. En effet, cette liaison ne se fait que dans un sens. Il a donc fallu prendre en compte cet élément dans l'interface graphique a chaque création de lien par l'utilisateur.

Prise en compte de la contrainte : La contrainte imposée était que le chemin emprunté ne doit passer que par des communes non confinées. Nous avons donc simplement autoriser la liaison de commune que si elles étaient non confinées. En faisant cela, nous sommes sûrs que quelque soit le chemin généré, il ne passera que par des communes non confinées.

Détermination du chemin optimal : L'algorithme commence par générer les chemins optimaux d'une source vers toutes les communes accessibles via les liens créés par l'utilisateur. Lorsque c'est fait, nous pouvons sélectionner une commune cible et une méthode se charge de nous retourner le chemin optimal. Un élément est cependant à prendre en compte, dès que la source est instanciée il faut relancer l'application pour définir une source.

d. Interface graphique

Outils choisis : Parmis tous les outils permettant de créer une interface graphique, nous avons fait le choix de JavaFX. Ce choix est justifié d'une part par le fait que l'outil est récent et fréquemment mis à jour et d'autre part qu'il s'utilise en tandem avec un autre outil : Scene builder. Ce dernier nous a permis de formaliser l'interface graphique plus facilement de gérer plusieurs éléments de style et de disposition des éléments plus rapidement qu'avec une édition manuelle du code FXML.
La montée en compétences pour maîtriser ces outils a en outre été particulièrement chronophage.

Structure logiciel JavaFX : Le contrôleur sert principalement à gérer les actions utilisateurs et récupérer les données saisie. Il y a néanmoins l'initialisation du modèle ainsi quelques élément fluidifiant l'exécution de l'application

3.1.3 Problématiques, limites et choix

a. Vidéos d'explications tardives

Nous avons commencé le développement relativement tôt, la principal conséquence de cette décision fut que nous avons décidé (après consultation et accord de Mme Zaouche) de considérer les attributs "Departement" et "Region" comme tel et non comme des classes à part entière. En effet, réadapter notre modèle de donnée ainsi que toutes les méthodes que nous avions code jusqu'alors nous aurait fait perdre une quantité significative de temps qui dans le contexte actuel s'est avéré plus que précieux.

b. Choix d'optimisation

Triple tableau modèle : Afin de réduire au maximum les parcours inutiles de tableaux nous avons décidé d'opter pour un format de 3 tableaux.

- Un tableau de communes non confinées
- Un tableau de communes confinées
- Un tableau contenant toutes les communes

L'intérêt était dans un premier temps de pouvoir gérer facilement le confinement et le déconfinement. Par la suite il s'est avéré particulièrement utile d'avoir un tableau regroupant toutes les communes pour éviter les multiples parcours lorsque des méthodes non spécifiques au statut (confinée ou non) de la commune. Comme par exemple Dijkstra ou la gestion des historiques.

Tableaux de String : Nous avons ajouté pour chaque tableau d'objet un tableau de String correspondant ainsi que des méthodes actualisant automatiquement les tableaux de String en cas de modification dans les tableaux. Il s'agit d'une décision à but purement ergonomique. En effet, la quasi-totalité des éléments JavaFX exploitent des String, en particulier les ListView qui nous permettent d'afficher des éléments sous forme de liste. Le fait d'avoir ces tableaux de String a facilité le développement de la synchronisation des données avec l'interface.

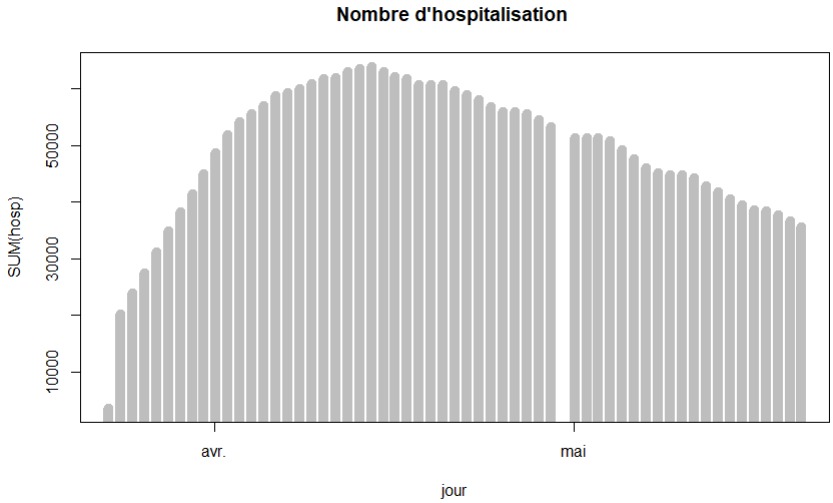
Utilisation de Dijkstra : Nous avons rencontré une difficulté inattendue lors de l'implémentation de cet algorithme. Lorsqu'une commune est confinée, ses liens ne sont pas mis à jour. Les mettre à jour entraînerait une complexité beaucoup trop élevée. Il faudrait parcourir toutes les communes pour voir si elles pointent vers cette commune ou si elles sont pointée par celle ci. Par conséquent, dès que les liens sont créés il ne faut plus confiner ou de confiner de communes sous peine d'obtenir un résultat erroné. Nous avons donc fait le choix de faire confiance à l'utilisateur en l'avertissant qu'il ne doit utiliser Dijkstra et lier des communes qu'à la fin de son utilisation.

Carte : A l'heure où nous rédigeons ce rapport, nous avons réussi à afficher une carte colorée mais elle n'est pas encore interactive. Nous continuons à travailler dessus et espérons avoir une version fonctionnelle pour le jour de la soutenance même si cette fonctionnalité reste facultative.

4 Data Mining

Grâce aux informations mises en sur différents sites internet nous avons pu effectuer une analyse des données à l'aide de R studio qui nous a permit d'obtenir les différents tableau et graphiques ci-dessous.

4.1 Statistiques descriptives



Pour commencer nous allons appliquer divers outils de statistiques pour résumer au mieux les données
On voit bien que le nombre de malade par pour jour suit une courbe en cloche avec un pic au alentours du 15 avril. Certain modèle épidémiologiques comme le modèle SIR (Individus Sain, Infectées, Rétablies) mettent en évidence à l'aide de système d'équation différentielles, l'allure que peut prendre la courbe du nombre de personnes infectées dans une région données, dans certain cas, si la maladie est suffisamment infectieuse cette courbe peut avoir un forme de cloche comme dans notre situation.

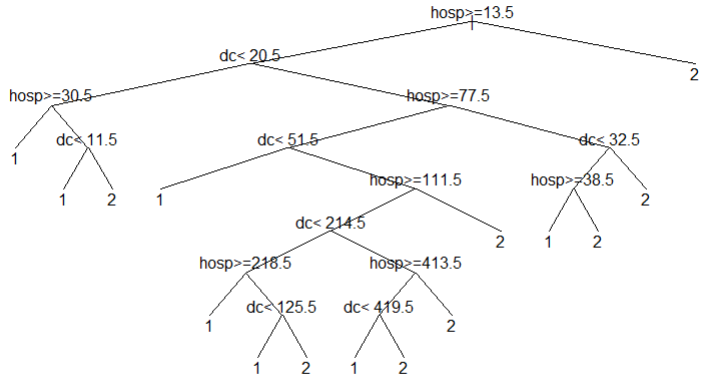
```
      reg          cl_age90        jour                          hosp            rea               rad
 Min.   : 1.0   Min.   : 9.0   Min.   :2020-03-23 00:00:00   Min.   :    0   Min.   :   0.00   Min.   :    0.0
 1st Qu.: 6.0   1st Qu.:29.0   1st Qu.:2020-04-07 00:00:00   1st Qu.:    2   1st Qu.:   0.00   1st Qu.:    8.0
 Median :32.0   Median :54.0   Median :2020-04-21 00:00:00   Median :   16   Median :   2.00   Median :   44.0
 Mean   :39.6   Mean   :53.1   Mean   :2020-04-21 05:04:00   Mean   :  251   Mean   :  45.36   Mean   :  364.3
 3rd Qu.:75.0   3rd Qu.:79.0   3rd Qu.:2020-05-06 00:00:00   3rd Qu.:  152   3rd Qu.:  23.00   3rd Qu.:  233.0
 Max.   :94.0   Max.   :90.0   Max.   :2020-05-20 00:00:00   Max.   :13209   Max.   :2668.00   Max.   :23033.0
       dc
 Min.   :   0.0
 1st Qu.:   0.0
 Median :   2.0
 Mean   : 113.9
 3rd Qu.:  44.0
 Max.   :6844.0
```

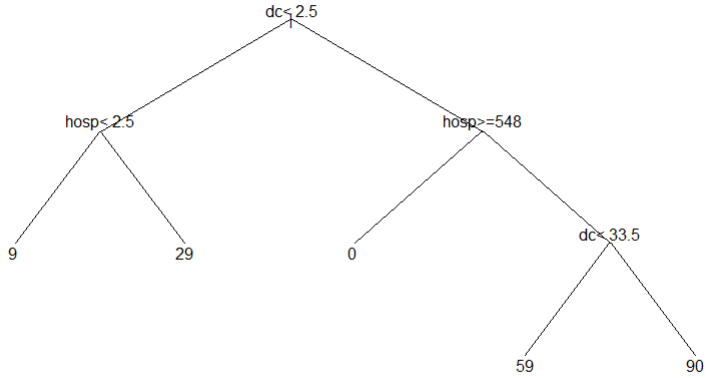
FIGURE 1 –

FIGURE 2 –

Au regard de la moyennes d'âge des individus hospitalisés (fig1) ainsi que le nombre de décès en fonction de l'âge (fig2), on constante que le patients les plus vieux semblent plus vulnérable au virus.

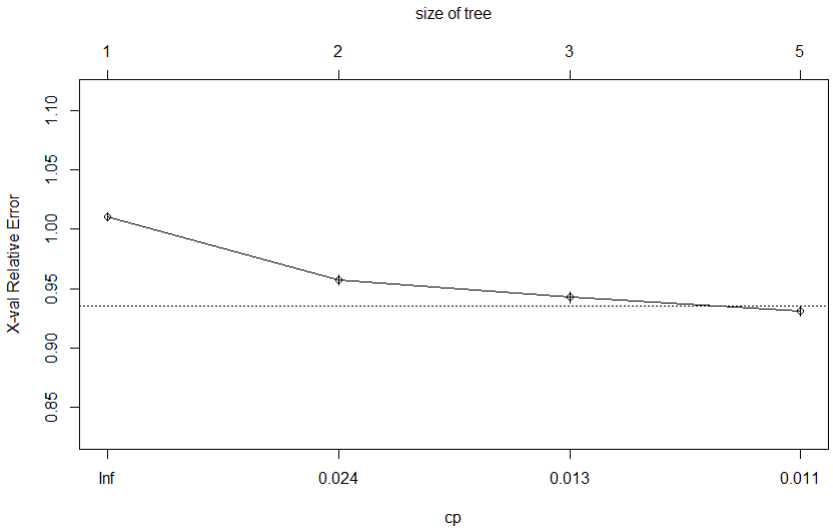
4.2 Prédictions des survivants

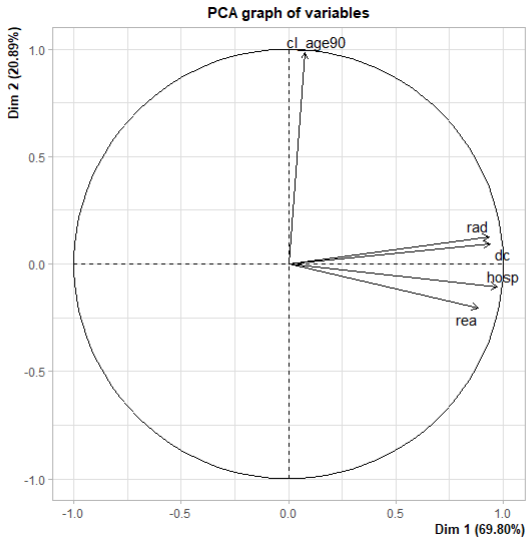


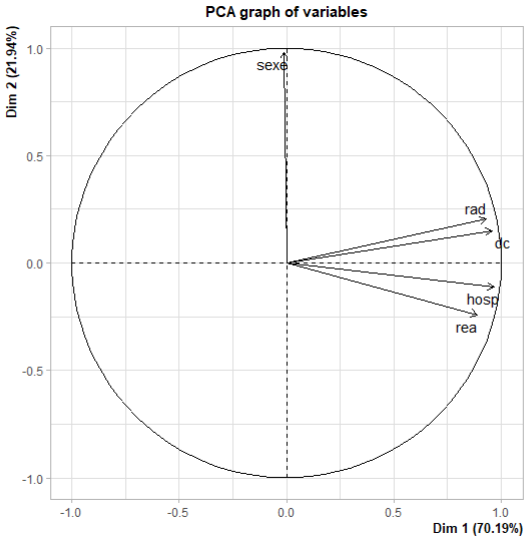


Un arbre de décision est le résultat d'un algorithme qui permet d'affecter à chaque individu une catégorie, en utilisant des règles de décision. Dans notre premier exemple, pour un jour données, et en ayant connaissance du nombre d'hospitalisation et du nombre de décès, on est capable de prédire le sexe du patients, ici correspond à un homme et 2 correspond à une femme. Dans notre second exemple on applique le même méthode pour prévoir cette fois-ci la classe d'âge du patient.

4.3 Importance des variables et ACP



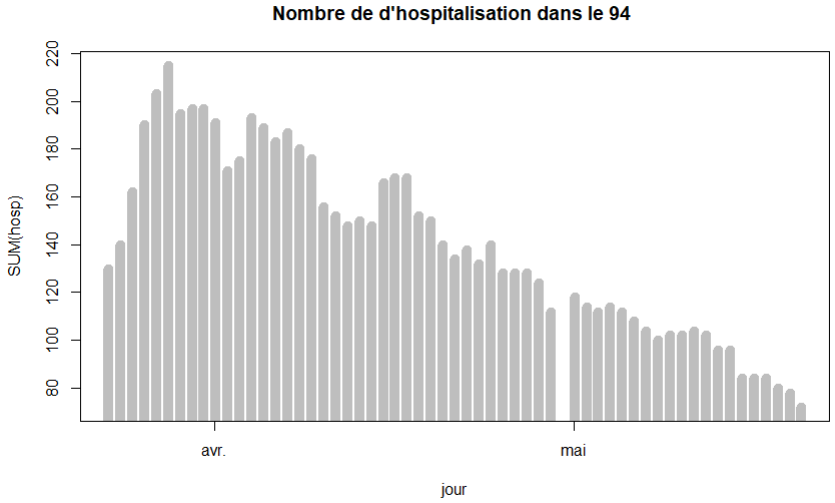




L' ACP est une méthode descriptive permettant de représenter graphiquement l'essentielle de l'information contenu dans un tableau de données quantitatives.
Tout d'abord on cherche le nombre d'axes minimal sur lesquels la projection minimise la déformation de la réalité, la première figure nous montre que l'on peut se limiter à seulement deux dimension, en effet on observe une coupure juste après le premier axe.

Ensuite sur notre première ACP (classe d'âge) on voit que les variable Hospitalisation, décès, retour à domicile et réanimation sont corrélées positivement et participe à la formation du première axe, la classe d'âge quant à elle participe à la formation du second axe mais on ne peut pas la relier avec le premier groupe de variable.

4.4 Évolution du nombre de malades dans un région données



5 Organisation du travail durant le projet

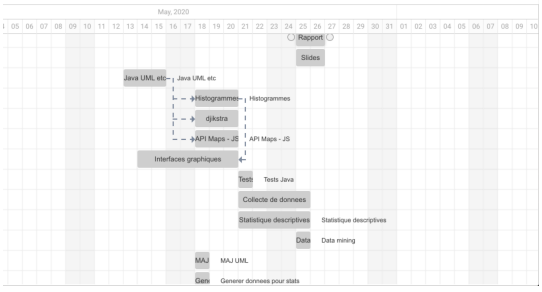
Durant les 3 semaines de projet, nous nous sommes répartis le travail de façon à être le plus efficace possible tout en réalisant l'ensemble du travail demandé dans le temps imparti. La méthode de travail qui nous paraissait la meilleure était le choix d'objectifs intermédiaires à atteindre chaque semaine dans l'avancement du travail.
Le but de cette méthode était principalement de ne pas nous retrouver avec une charge de travail excessive à la fin de la période de projet.
Pour effectuer de la meilleure façon la répartition du travail, nous avons utilisé l'outil Trello. Ce dernier nous a permis de réaliser un calendrier regroupant l'ensemble des tâches à effectuer avec les dates clés associées.
Au sein du groupe, nous avons fait le choix unanime de travailler pendant les jours de week-ends, afin d'être le plus confortable possible au niveau du planning. Nous avons donc décidé de découper le travail à faire de la manière suivante :

- 1ère semaine *(du 13 Mai au 17 Mai)* : Réalisation de l'UML et implémentation Java du problème. Commencement des interfaces graphiques de l'application.

- 2eme semaine *(du 18 Mai au 24 Mai)* : En début de semaine, Début en parallèle des histogrammes, de l'implémentation dijkstra et de API Maps. Avancement des interfaces graphiques . En fin de semaine, premiers tests java de l'applications et début de l'étude des statistiques dans l'optique de la prédiction : Recherche de statistiques sur les décès et guérisons du Coronavirus en France, et réalisation des statistiques descriptives.

- 3eme semaine *(du 25 Mai au 31 Mai)* : Finition de la partie d'études des données et du travail plus généralement.En début de semaine, réalisation de la partie Datamining(Arbres de décision) Sur le reste de la semaine, modification et corrections mineures du travail. Réalisation du rapport et du diaporama.

Voici une ébauche du planning tel qu'il l'est sur Trello



5.1 Répartitions des tâches

Suite à une discussion initiale pour établir les tâches à réaliser du projet, nous avons déterminé que notre objectif était d'obtenir un résultat aussi élevé que possible et de renforcer nos capacité de programmation en Java et nos connaissances en statistiques. Nous avons décidé de ne pas élire de chef d'équipe, d'utiliser WhatsApp pour nous contacter, de partager des fichiers à l'aide de Google Drive et de Trello pour gérer le planning des tâches comme dit précédemment. Nous avons en outre utilise un diagramme de Gantt (synchronisé avec Trello) pour repartir au mieux dans le temps les taches a réaliser. L'objectif était aussi d'avoir une vision globale rapide et synthétique des différentes phases du projet.
Après avoir recensé les capacités de chacun, nous avons décidé de choisir Adam comme responsable principale au niveau de la partie Java et mise en oeuvre de l'application, et Bara et Guillaume comme assistants pour l'aider quand il le fallait. Ensuite, Bara, Xindi et Guillaume furent responsables de la partie Datamining, Adam a également apporté son aide dans la réalisation de cette partie. Le rapport a été rédigé conjointement par les membres de l'équipe.

5.2 Point sur le travail en équipe

Avec les efforts de tous, le plan a été strictement mis en œuvre. En raison du travail à distance, Nous avons cependant connu quelques problèmes de communication. Plusieurs fois par semaine, nous rendions compte des derniers progrès et assignions les tâches dans le groupe WhatsApp. Selon la situation et les besoins, une vidéo-conférence par Teams était convenue à l'avance pour s'assurer que tout le monde reçoive les informations importantes et nécessaires au bon avancement du projet.

5.3 Améliorations possibles du travail

Nous avons recensé des améliorations possibles à notre travail actuel.
Tout d'abord, dans la partie Java,
- La finalisation de la carte de la France afin de visualiser les différentes communes étudiées et leur état par rapport à l'épidémie du Coronavirus.
- La possibilité de supprimer n'importe quels liens ou communes
Dans la partie Datamining,
- Réalisation d'autres analyses notamment sur les conditions sociales ou les problèmes de santé des patients (pas de données pertinentes trouvées)
- Faire des analyses sur d'autres pays que la France

6 Conclusion

Pour conclure, il s'agit du deuxième projet d'ingénieur auquel nous avons affaire. Un projet concret car pleinement intègre a l'actualité, qui implique des contraintes concrètes. Cette expérience fut très formatrice tant sur le plan technique que relationnel. En effet, nous avons dû apprendre à maîtriser de nombreux outils et technologies que nous n'avions pas l'habitude d'utiliser. Un autre challenge fut de gérer le travail en groupe et de maintenir la cohésion au sein de celui-ci.
Malgré de nombreuses obstacles et imprévus nous avons mené à bien ce projet et après un travail de fond soutenu nous avons réussi à cerner les besoins du client et à y répondre.

Au-delà de l'aspect formateur, cette aventure nous a permis de faire un bilan et en état des lieux sur nos compétences en soulignant nos forces et nos faiblesses. Nous avons donc pu mieux cerner nos méthodes de travail et cela nous servira pour des projets futurs, au sein de l'école et dans notre cursus professionnel.