Méthodes d'imputation des données manquantes sur R

DIOP_Ousseynou

Année 2022-2023

Contents

Introduction

Malgré la quantité croissante de données disponibles et l'émergence du BigData, les problématiques de données manquantes restent très répandues dansles problèmes statistiques et nécessitent une approche particulière. Ignorer les données manquantes peut entraîner, outre une perte de précision, de forts biais dans les modèles d'analyse,des erreurs dans les résultats de l'analyse pouvant conduire ainsi a des conclusions erronées.Pour résoudre ce problème, les méthodes d'imputation ont été développées pour estimer les valeurs manquantes en utilisant des informations disponibles dans la base.

Dans cette présentation , il sera question de definir les différents types de données manquantes et illustration de leurs répartitions possibles, de decrire les principales stratégies de gestion des données manquantes par suppression de données ou par d'autre méthode stochastique, sans souci d'exhaustivité .

I. Généralité

1. Définitions

1.1 Données manquantes

Les données manquantes sont des valeurs qui ne sont pas présentes dans un ensemble de données. Elles peuvent être manquantes pour différentes raisons, comme une erreur de saisie, un échec de mesure, ou un refus de réponse.

1.2 Imputation

L'imputation est un processus qui consiste à remplacer les valeurs manquantes dans un ensemble de données par des valeurs estimées afin de maintenir l'intégrité des données et d'obtenir des résultats plus précis et plus fiables.

2. Typologie des données manquantes

Afin d'aborder correctement l'imputation des données manquantes il faut en distinguer les causes, surtout si elles ne sont pas le simple fruit du hasard.Il existe trois catégories de données manquantes:

2.1 MCAR (missing completely at random)

Une donnée est MCAR, c'est-à dire manquante de façon complètement aléatoire si la probabilité d'absence est la même pour toutes les observations. Cette probabilité ne dépend donc que de paramètres extérieurs indépendants de cette variable.

Par exemple : si chaque participant à un sondage décide de répondre à la question du revenu en lançant un dé et en refusant de répondre si la face 6 apparaît [1]. À noter que si la quantité de données MCAR n'est pas trop importante, ignorer les cas avec des données manquantes ne biaisera pas l'analyse. Une perte de précision dans les résultats est toutefois à prévoir

2.2 MAR (Missing at random)

Le cas des données MCAR est peu courant. Il arrive lorsque les données ne manquent pas de façon complètement aléatoire; si la probabilité d'absence est liée à une ou plusieurs autres variables observées, on parle de missingness at random (MAR). Il existe des méthodes statistiques appropriées qui permettrons d'éviter de biaiser l'analyse

2.3 MNAR (Missing not at random)

La donnée est manquante de façon non aléatoire (MNAR) si la probabilité d'absence dépend de la variable en question. Un exemple répandu est le cas où des personnes avec un revenu important refusent de le dévoiler. Les données MNAR induisent une perte de précision (inhérente à tout cas de données manquantes) mais aussi un biais qui nécessite le recours à une analyse desensibilité.

II. bibliothéques

Il existe plusieurs bibliothèques pour imputer les données manquantes dans R.

- **mice**: permet d'imputer les données manquantes à l'aide de différents modèles d'imputation, notamment la régression linéaire, la régression logistique, l'ACP, etc.
- **Amelia**: une méthode qui utilise des algorithmes d'EM et de Bootstrap pour estimer les données manquantes.
- **missForest**: une méthode basée sur les forêts aléatoires pour imputer les données manquantes.
- **imputeTS**: une méthode pour imputer les données manquantes dans les séries temporelles.
- **Hmisc**: propose plusieurs méthodes d'imputation telles que la moyenne, la médiane, le mode, la régression, etc.

- **zoo**: est une bibliothèque en R qui fournit des fonctions pour l'analyse de séries temporelles avec des données manquantes. Elle propose notamment des méthodes d'imputation de données manquantes comme la méthode de lissage locf (last observation carried forward) qui remplace les valeurs manquantes par la dernière observation valide connue. Elle propose également d'autres méthodes comme la méthode de lissage nocb (next observation carried backward) ou la méthode de lissage na.approx (linear interpolation)

Ceci ne représente qu'une liste non exhaustive de bibliothèques pour l'imputation des données manquantes dans R.

-syntaxe:

```
# installation: install.packages("Nom_library")
# charger : library("Nom_library")
```

III. Méthodes d'imputation

Il existe plusieurs méthodes d'imputation de données dans RStudio. Les méthodes d'imputation les plus courantes sont:

- Imputation par la suppression;
- Imputation par la moyenne;
- Imputation par la médiane;
- Imputation par régression;
- Imputation par la méthode LOCF;
- Imputation par la méthode « hot-deck »;
- Imputation par la méthode de Monte carlo;
- Imputation par la forêt aléatoire;
- imputation par la méthode du plus proche voisin ;
- Imputation par l'analyse en composantes principales (ACP).

1. Méthode d'imputation par suppression

L'imputation par suppression consiste à supprimer les individus avec des valeurs manquantes de l'analyse. Cela peut se faire de différentes manières :

- Suppression listwise : elle consiste à supprimer tous les individus qui ont au moins une valeur manquante.
- Suppression pairwise : elle consiste à supprimer uniquement les observations manquantes pour une variable spécifique. -Suppression par blocs : elle consiste à supprimer des blocs d'individus avec des valeurs manquantes. Cette méthode est utile lorsque les valeurs manquantes sont groupées, par exemple lorsque des questions d'un questionnaire ont été omises par erreur.

L'imputation par suppression est simple à mettre en œuvre mais peut entraîner une perte importante d'informations et introduire des biais dans l'analyse.

- **Exemple**:

```
# Création d'un data.frame avec des valeurs manquantes
df <- data.frame(var1 = c(1, 2, NA, 4, 5), var2 = c(NA, 7, 8, NA, 10))

# Suppression des observations avec des données manquantes
df_complete <- df[complete.cases(df), ]

# Affichage du data.frame complet
df_complete
```

2. Méthode d'imputation par la moyenne

La méthode d'imputation par la moyenne, également connue sous le nom de méthode d'imputation par la valeur moyenne, est une méthode simple d'imputation de données manquantes. Elle consiste à remplacer les valeurs manquantes dans une variable par la moyenne des valeurs non manquantes de cette variable.

Cette méthode peut être utilisée pour des variables numériques continues, telles que l'âge, le poids, la taille, etc. Elle est rapide et facile à implémenter, mais elle peut entraîner une baisse de la variance et de la corrélation entre les variables. Elle est également sensible aux valeurs aberrantes qui peuvent fausser la moyenne.

la méthode d'imputation par la moyenne est généralement utilisée comme méthode de référence ou comme méthode de base pour les comparaisons avec d'autres méthodes d'imputation plus sophistiquées.

- **Exemple**:

```r
# Créer un data.frame avec des valeurs manquantes
df <- data.frame(x = c(1, 2, NA, 4, NA, 6, 7, NA, 9))

# Calculer la moyenne des valeurs non manquantes
mean_value <- mean(df$x, na.rm = TRUE)

# Remplacer les valeurs manquantes par la moyenne
df[is.na(df$x), "x"] <- mean_value
```

- **Syntaxe**: nous avons calculé la moyenne des valeurs non manquantes à l'aide de la fonction *mean()* en spécifiant *na.rm = TRUE* pour exclure les valeurs manquantes. Nous avons ensuite remplacé les valeurs manquantes dans la colonne "x" du *data.frame* par la moyenne en utilisant la syntaxe *df[is.na(df$x), "x"] <- mean_value*.

3. Méthode d'imputation par la mediane

La méthode d'imputation par la médiane est une méthode statistique permettant de remplacer les valeurs manquantes d'une variable numerique par sa médiane. La médiane est une mesure de tendance centrale qui se situe au milieu des valeurs observées et qui sépare la distribution en deux parties égales.

Lorsqu'une valeur manquante est détectée dans une variable continue, on calcule la médiane de cette variable sur l'ensemble des observations non manquantes. Cette médiane est ensuite utilisée pour remplacer la valeur manquante.

Cette méthode est assez simple à mettre en oeuvre et peut être utile lorsque la distribution de la variable n'est pas normale ou lorsque la variable contient des valeurs aberrantes qui pourraient biaiser l'estimation de la moyenne. Cependant, elle ne permet pas de prendre en compte les relations entre les différentes variables et peut être moins précise que d'autres méthodes plus sophistiquées en termes de performance de prédiction.

- **Exemple**: Avec la bibliothèque zoo

```
# Charger la bibliothèque
library(zoo)

# Créer un exemple de dataframe avec des données manquantes
df <- data.frame(a = c(1, 2, NA, 4, 5),
                 b = c(NA, 2, 3, 4, 5))
df
```

```
##    a  b
## 1  1 NA
## 2  2  2
## 3 NA  3
## 4  4  4
## 5  5  5
```

```
# Imputation par la médiane
df_med <- na.aggregate(df, FUN = median)

# Afficher le résultat
df_med
```

```
##   a   b
## 1 1 3.5
## 2 2 2.0
## 3 3 3.0
## 4 4 4.0
## 5 5 5.0
```

- **Syntaxe**:la fonction *na.aggregate()* est utilisée pour remplacer les valeurs manquantes par la médiane de chaque colonne. La fonction prend deux arguments principaux : le premier est le dataframe à imputer, le deuxième est la fonction de résumé(une fonction qui prend en entrée plusieurs valeurs et retourne une seule valeur qui résume les données) à utiliser pour remplacer les valeurs manquantes. Dans ce cas, la fonction de résumé est *median()*.

4. Méthode d'imputation par la regression

La méthode d'imputation par la régression est une technique d'imputation des données manquantes qui utilise un modèle de régression pour prédire les valeurs manquantes en se basant sur les valeurs des variables explicatives connues.

La méthode consiste à :

- Identifier les variables explicatives qui sont corrélées avec la variable à imputer;
- estimer un modèle de régression linéaire ou non linéaire entre la variable à imputer et les variables explicatives connues;
- utiliser le modèle pour prédire les valeurs manquantes de la variable à imputer.

Cette méthode peut être efficace lorsque la variable à imputer est corrélée avec d'autres variables dans le jeu de données. Toutefois, elle peut être sensible aux valeurs aberrantes et à la non-linéarité des relations entre les variables.

Pour imputer des données manquantes par la méthode de la régression, nous pouvons utiliser la fonction predict() de R en spécifiant un modèle linéaire.

- **exemple**: Supposons que vous avez un dataframe df avec une variable numérique var1 et une variable catégorielle var2. Les données manquantes dans var1 seront imputées en utilisant var2 comme prédicteur :

```
## Chargement de la bibliothèque
# library(dplyr)

## Création d'un modèle linéaire

# model <- lm(var1 ~ var2, data = df)

## Imputation des données manquantes

# df %>%
  # mutate(var1_imputed = ifelse(is.na(var1), predict(model, newdata = .), var1))
```

- **Syntaxe**:la fonction *predict()* est utilisée pour imputer les valeurs manquantes de *var1* en utilisant les valeurs de *var2* comme entrée pour le modèle. La fonction *mutate()* est utilisée pour créer une nouvelle colonne dans *df* appelée *var1_imputed* qui contient les valeurs imputées.

5. Méthode d'imputation par LOCF (last observation carried forward)

L'imputation par LOCF (Last Observation Carried Forward) consiste à remplacer les valeurs manquantes par la dernière observation valide précédant la valeur manquante dans la série chronologique. Cette méthode est souvent utilisée dans les données longitudinales où les observations sont prises à des moments réguliers dans le temps. Elle est particulièrement utile lorsque les données manquantes sont de courtes durées et que la variable a peu de changements brusques. Cependant, cette méthode peut conduire à une surestimation de la corrélation et de la variance, ainsi qu'à une baisse de la précision de l'estimation.

- **Exemple**: Avec la bibliothèque zoo

```
## Chargement des données

#data <- read.csv("nom_du_fichier.csv")

## Conversion des variables en série temporelle

#data$Date <- as.Date(data$Date, format="%Y-%m-%d")
#data_ts <- zoo(data[,-1], data$Date)

## Imputation des valeurs manquantes par LOCF

#data_imputed <- na.locf(data_ts)

## Conversion de la série temporelle en data.frame

#data_imputed <- data.frame(Date=index(data_imputed), coredata(data_imputed))

## Sauvegarde des données imputées

#write.csv(data_imputed, "nom_du_fichier_imputed.csv", row.names = FALSE)
```

- **Syntaxe**: Supposons que les données sont stockées dans un fichier CSV avec une colonne Date contenant les dates d'observation et les variables à imputer dans les autres colonnes. On commence par charger les données et les convertir en une série temporelle à l'aide de la fonction *zoo()*.la fonction *index()* permet d'extraire l'index de la série temporelle comme la colonne « Date », et la fonction *coredata()* pour extraire les données de la série temporelle. Ensuite, on impute les valeurs manquantes à l'aide de la fonction *na.locf()* et on convertit la série temporelle imputée en un *data.frame*. Finalement, on sauvegarde les données imputées dans un fichier CSV.

6. Méthode d'imputation de Monte carlo

L'imputation par Monte Carlo est une méthode probabiliste d'imputation des données manquantes qui consiste à simuler plusieurs valeurs possibles pour chaque donnée manquante à partir d'une distribution de probabilité, puis à choisir une valeur aléatoire parmi ces simulations pour remplacer la donnée manquante. Cette méthode permet de prendre en compte l'incertitude liée à l'imputation et de générer des données imputées qui respectent la distribution de probabilité de la variable.

- **Exemple**: Avec la bibliothèque mice

```
## Chargement des données

#data <- read.csv("data.csv")

## Chargement de la bibliothèque mice

#library(mice)

## Création de l'objet imputation

#imp <- mice(data, method = "norm", m = 5, maxit = 50)

## Imputation des données manquantes

#data_imputed <- complete(imp)
```

- **Syntaxe**:la méthode de Monte Carlo est utilisée avec une distribution normale pour imputer les données manquantes *(method = "norm")*. L'argument *m*spécifie le nombre de simulations à effectuer pour chaque donnée manquante, et *maxit* spécifie le nombre d'itérations de l'algorithme. Enfin, la fonction *complete()* permet de récupérer les données imputées dans un objet *data.frame*.

7. Méthode d'imputation par la forêt aléatoire

La méthode d'imputation par la forêt aléatoire est une technique d'imputation de données manquantes basée sur l'utilisation d'un modèle de forêt aléatoire

Le principe de la méthode est le suivant : pour chaque variable avec des valeurs manquantes, on utilise les autres variables disponibles pour construire un modèle de forêt aléatoire. Ce modèle est ensuite utilisé pour prédire les valeurs manquantes pour cette variable en se basant sur les variables explicatives. Cette opération est répétée plusieurs fois, en utilisant des échantillons différents des données originales à chaque fois, afin d'obtenir des prédictions plus robustes.

La méthode de la forêt aléatoire est particulièrement utile pour les ensembles de données complexes, avec un grand nombre de variables et des relations complexes entre les variables. Elle est également adaptée aux données avec des valeurs manquantes à des endroits aléatoires et peut traiter des données de différents types, y compris les variables catégorielles.

- **Exemple**:Avec la bibliothèque miceforest

```
## Chargement de la bibliothèque missForest

#library(missForest)

## Imputation des données manquantes avec missForest

#imputed_data <- missForest(data)
```

8. Méthode d'imputation par le plus proche voisin

La méthode d'imputation par le plus proche voisin (ou k-nearest neighbor imputation) consiste à imputer les valeurs manquantes en se basant sur les valeurs des voisins les plus proches. Pour chaque valeur manquante, les k observations les plus proches sont identifiées à partir des valeurs des variables disponibles, et la valeur manquante est imputée en utilisant la valeur moyenne ou médiane de ces k voisins. Cette méthode est basée sur l'hypothèse que les valeurs manquantes sont similaires à celles des observations les plus proches dans l'espace de données.

- **Exemple**:

```
# library(imputeTS)
# imputed_ts <- knn.impute(my_ts_object, k=5)
```

9. Méthode d'imputation parl'analyse en composantes principales (ACP)

La méthode d'imputation par l'analyse en composantes principales (ACP) est utilisée pour imputer les valeurs manquantes dans un ensemble de données multivariées. Elle consiste à réduire la dimension de l'espace des données en utilisant une transformation linéaire pour trouver les principales composantes qui expliquent la variation dans les données.

L'ACP consiste à :

- Centrer et réduire les variables de l'ensemble de données ;
- Calculer la matrice de covariance ou la matrice de corrélation de l'ensemble de données ;
- Calculer les valeurs et vecteurs propres de la matrice de covariance ou de corrélation ;
- Sélectionner les composantes principales (CP) en fonction de la quantité de variance expliquée ou de la valeur propre ;
- Utiliser les CP sélectionnées pour imputer les valeurs manquantes dans les variables correspondantes.

L'avantage de cette méthode est qu'elle peut être appliquée à des ensembles de données de grande taille et de nombreuses variables. Cependant, elle ne fonctionne pas bien avec des données très asymétriques ou des données qui ont des distributions de probabilité très différentes.

- **Exemple**: Avec mice

```
#library(mice)

## Charger les données

# data <- read.csv("donnees.csv")

## Convertir les variables catégorielles en facteurs

#for (i in 1:ncol(data)) {
#   if (is.character(data[,i])) {
#    data[,i] <- as.factor(data[,i])
#  }
#}

## Imputation par ACP

# imp <- mice(data, method = "pca", m = 5, maxit = 50)

## Obtenir les données imputées

# imputed_data <- complete(imp)
```

- **Syntaxe**: nous avons d'abord chargé les données à partir d'un fichier CSV. Ensuite, nous avons converti toutes les variables catégorielles en facteurs. Enfin, nous avons appliqué la méthode d'imputation par ACP en utilisant la fonction *mice* avec les options *method = "pca"*, $m = 5$ (nombre de voisins pour la méthode KNN) et *maxit = 50* (nombre maximal d'itérations). Enfin, nous avons obtenu les données imputées en utilisant la fonction *complete*.

Iv. Limites d'imputations des données manquantes

L'imputation des données manquantes est une pratique efficace couramment utilisée dans le domaine de la statistique et de l'apprentissage automatique , mais elle présente des limites notamment:

- Biais : l'imputation des données manquantes peut entraîner un biais dans les résultats de l'analyse. Le biais peut être dû à des hypothèses incorrectes ou à une modélisation inappropriée de la distribution des données manquantes.
- Perte d'information : l'imputation des données manquantes peut entraîner une perte d'information si les valeurs imputées ne sont pas exactes.
- Incertitude : l'imputation des données manquantes introduit de l'incertitude dans les résultats de l'analyse, car les valeurs imputées ne sont pas exactes. Cette incertitude peut se propager dans les analyses ultérieures.
- Complexité : l'imputation des données manquantes peut être un processus complexe, en particulier pour les ensembles de données volumineux ou avec des modèles statistiques complexes.
- Fiabilité : la fiabilité de l'imputation des données manquantes dépend de la qualité des données disponibles et des méthodes d'imputation utilisées. Il est important de valider la qualité des données imputées avant d'utiliser les résultats pour des analyses ultérieures.

V. Applications

Cette section vise à appliquer les méthodes d'imputation des données manquantes présentées précédemment sur une base de données.

1. Présentation de la base de données

Pour l'application , nous utiliserons une partie de la base de données AmesHousing . AmesHousing est une base de données de référence souvent utilisée en apprentissage automatique pour la prédiction des prix de l'immobilier. Elle contient des informations sur les ventes de maisons à Ames, dans l'Iowa, aux États-Unis, entre 2006 et 2010. La base de données contient 2930 observations et 82 variables, dont 23 variables catégorielles et 58 variables numériques.

Les variables de cette base de données comprennent des informations sur la taille, l'âge, le type, la qualité et les caractéristiques de la maison, ainsi que des informations sur le quartier et l'emplacement géographique. La variable cible de cette base de données est le prix de vente de la maison.

Les variables catégorielles incluent des informations telles que le type de revêtement de sol, la qualité de la cuisine et les matériaux de la toiture, tandis que les variables numériques comprennent des informations telles que la surface habitable, la surface du terrain et le nombre de chambres et salles de bain. La plupart des variables ont quelques valeurs manquantes.

Par exemple dans la base nous avons les variables comme:

- SalePrice : prix de vente de la maison.
- OverallQual : qualité générale de la maison.
- GrLivArea : surface habitable au-dessus du niveau du sol.
- GarageType : type d'emplacement de stationnement.
- YearBuilt : année de construction de la maison.
- MasVnrType : type de revêtement en pierre.
- Lot_Frontage: représente la distance en pieds linéaires de la propriété par rapport à la rue. Il est important de comprendre les variables de la base de données et leur signification afin de pouvoir appliquer les méthodes d'imputation appropriées pour les données manquantes.
- **Notez bien**:*Les bibliothèques et syntaxes utilisées pour l'application des méthodes d'imputation peuvent différer de celles présentées dans la partie théorique en raison des particularités et de la structure des données*

2.Prérequis

```
##installation de la bibliothèque AmesHousing et charger la base (Ames_raw)
# install.packages("AmesHousing") ##J'ai déja la base
# Ames_raw
##Importation de la base
base_midm<- read.csv2("C:/Users/dell/Desktop/ENSAE/ISEP2/Semestre_2/Programmation R/Expo
View(base_midm)

##Information de la base

summary(base_midm)
```

voir les données manquantes avec sapply qui applique une fonction donnée à chaque élément d'une liste et renvoie les résultats sous forme de vecteur. Dans ce cas, nous appliquons la fonction sum(is.na(x)) à chaque variable de la base de base_midm .

```r
sapply(base_midm, function(x) sum(is.na(x)))
```

```
##               X           Order             PID     MS_SubClass    Lot_Frontage
##               0               0               0               0             490
##        Lot_Area    Overall_Qual    Overall_Cond      Year_Built    Mas_Vnr_Area
##               0               0               0               0              23
##    BsmtFin.SF.1    BsmtFin.SF.2     Bsmt_Unf_SF   Total_Bsmt_SF     X1st.Flr.SF
##               1               1               1               1               0
##     X2nd.Flr.SF Low.Qual.Fin.SF     Gr_Liv_Area  Bsmt.Full.Bath  Bsmt.Half.Bath
##               0               0               0               2               2
##       Full_Bath       Half_Bath   Bedroom_AbvGr   Kitchen.AbvGr   Garage_Yr_Blt
##               0               0               0               0             159
##     Garage_Cars     Garage_Area    Wood.Deck.SF   Open.Porch.SF  Enclosed.Porch
##               1               1               0               0               0
##     X3Ssn.Porch    Screen.Porch       Pool.Area        Misc.Val         Mo.Sold
##               0               0               0               0               0
##         Yr.Sold      Sale_Price           DEDED
##               0             390             801
```

3.Imputation par la moyenne

```r
##Sans aucune bibiliothèque

# copie de la base

library(data.table)
midm1<-copy(base_midm)
View(midm1)

# Calculer la moyenne de LotFrontage sans les valeurs manquantes

mean_Lot_Frontage <- mean(midm1$Lot_Frontage, na.rm = TRUE)

# Remplacer les valeurs manquantes par la moyenne

midm1$Lot_Frontage[is.na(midm1$Lot_Frontage)] <- mean_Lot_Frontage

#Verification des valeurs manquantes

sum(is.na(midm1$Lot_Frontage))
View(midm1)
```

```r
##Avec la bibliothèque Hmisc

library(Hmisc)

midm2<-copy(base_midm)

# Imputation des valeurs manquantes dans la variable "LotFrontage"

midm2$Lot_Frontage <- impute(midm2$Lot_Frontage, mean)

# Vérification des valeurs manquantes après l'imputation

sapply(midm2, function(x) sum(is.na(x)))
View(midm2)
```

4.Imputation par la mediane

```r
##Avec la bibliothéque Hmisc


library(Hmisc)

midm_med<-copy(base_midm)

# Imputation des valeurs manquantes par la médiane pour la variable LotFrontage

midm_med$Lot_Frontage <- with(midm_med, impute(Lot_Frontage, median))


# Vérification des valeurs manquantes après imputation

sapply(midm_med, function(x) sum(is.na(x)))
```

5.Imputation par la regression

```r
# Exemple pour la variable "LotFrontage"

midm_rl<-copy(base_midm)
model <- lm(Sale_Price ~ ., data = midm_rl[, -1])
```

```
midm_rl$Sale_Price <- ifelse(is.na(midm_rl$Sale_Price),
                         predict(model, newdata = midm_rl), midm_rl$Sale_Price)
View(midm_rl)
sum(is.na(midm_rl))
```

6.Imputation par Last Observation Carried Forward

```
library(zoo)

midm_locf<-copy(base_midm)

#Imputer les valeurs manquantes de la variable « Lot_Frontage »

midm_locf$Lot_Frontage <- na.locf(midm_locf$Lot_Frontage)

#Imputer les valeurs manquantes de la variable « Garage_Yr_Blt »

midm_locf$Garage_Yr_Blt <- na.locf(midm_locf$Garage_Yr_Blt)

#Vérifier que toutes les valeurs manquantes ont été remplacées

sum(is.na(midm_locf)) #Vérifier que toutes les valeurs manquantes ont été remplacées
summary(midm_locf)
```

7.Imputation par Monte carlo

```
midm_mc<-copy(base_midm)
library(mice)

## Convertir les variables qualitatives en variables facteur

midm_mc <- data.frame(lapply(midm_mc, factor))

## Convertir les variables qualitatives en variables facteur

m <- 1 # nombre d'itérations

# imp <- mice(midm_mc, m = m, method = "norm.predict") #ça prend du temps avec ma mach

#summary(imp)
```

8.Imputation par les K plus proche voisins

```
#Imputation par les K plus proche voisins


library(VIM)
midm_k<-copy(base_midm)

#dat.kNN=kNN(midm_k, k=5, imp_var=FALSE) #ça prend du temps avec ma machine
```

9.Imputation par ACP

```
midm_acp<-copy(base_midm)

# Supprimer les observations contenant des valeurs manquantes
data_complete <- na.omit(midm_acp)

# Effectuer une ACP sur les données complètes
pca <- princomp(data_complete)

# Imputer les valeurs manquantes à l'aide des composantes principales
data_imputed <- predict(pca, newdata = midm_acp, na.action = na.pass)
View(data_complete)
View(data_imputed)
```

10.Imputation par La Foret aleatoire

```
library(missForest)
midm_fa<-copy(base_midm)

# mifa.imp <- missForest(midm_fa) #ça prend du temps avec ma machine
# mifa.complete <- mifa.imp$ximp
# View(mifa.complete)
```

Conclusion

Il est important de prendre en compte les données manquantes lors de l'analyse des données, car cela peut affecter les résultats et les conclusions tirées à partir des données. Des méthodes d'imputation appropriées peuvent être utilisées pour remplacer les valeurs manquantes et améliorer la qualité des données pour l'analyse.Ces méthodes d'imputation peuvent être basées sur des approches statistiques ou non statistiques et peuvent varier en fonction de la complexité des données, du niveau de précision souhaité et des objectifs de l'analyse.Il faut noter que l'imputation peut introduire des erreurs dans l'ensemble de données, en particulier si les méthodes utilisées ne sont pas adaptées ou si les données manquantes sont mal interprétées. Par conséquent, il est important de comprendre les méthodes d'imputation et de les appliquer avec précaution pour assurer l'intégrité des données et l'exactitude des résultats d'analyse

- **Notez bien**: *Ce document ne présente qu'une partie des méthodes d'imputation disponibles pour traiter les données manquantes. De ce fait, il est possible d'approfondir les recherches pour découvrir d'autres méthodes plus complexes pouvant inclure des algorithmes d'apprentissage automatique tels que les réseaux de neurones, les machines à vecteurs de support, etc .*

Références bibliograpiques

- *ICHO-PRO : Imputation des données manquantes avec R*
- *K. Bache et M. Lichman, UCI Machine Learning Repository, 2013, http://archive.ics.uci.edu/ml.*
- *Mohamed Ali KHOUAJA :Exploitation et traitement des données manquantes sous R*
- *Missing Data: Analysis and Design » par John W. Graham et Jaejoon Song: https://www.missingdata.lshtm.ac.uk/*
- *WikiStat:https://www.math.univ-toulouse.fr/~besse/Wikistat/pdf*

https://ichi.pro/fr/imputation-des-donnees-manquantes-avec-r-184650418014357