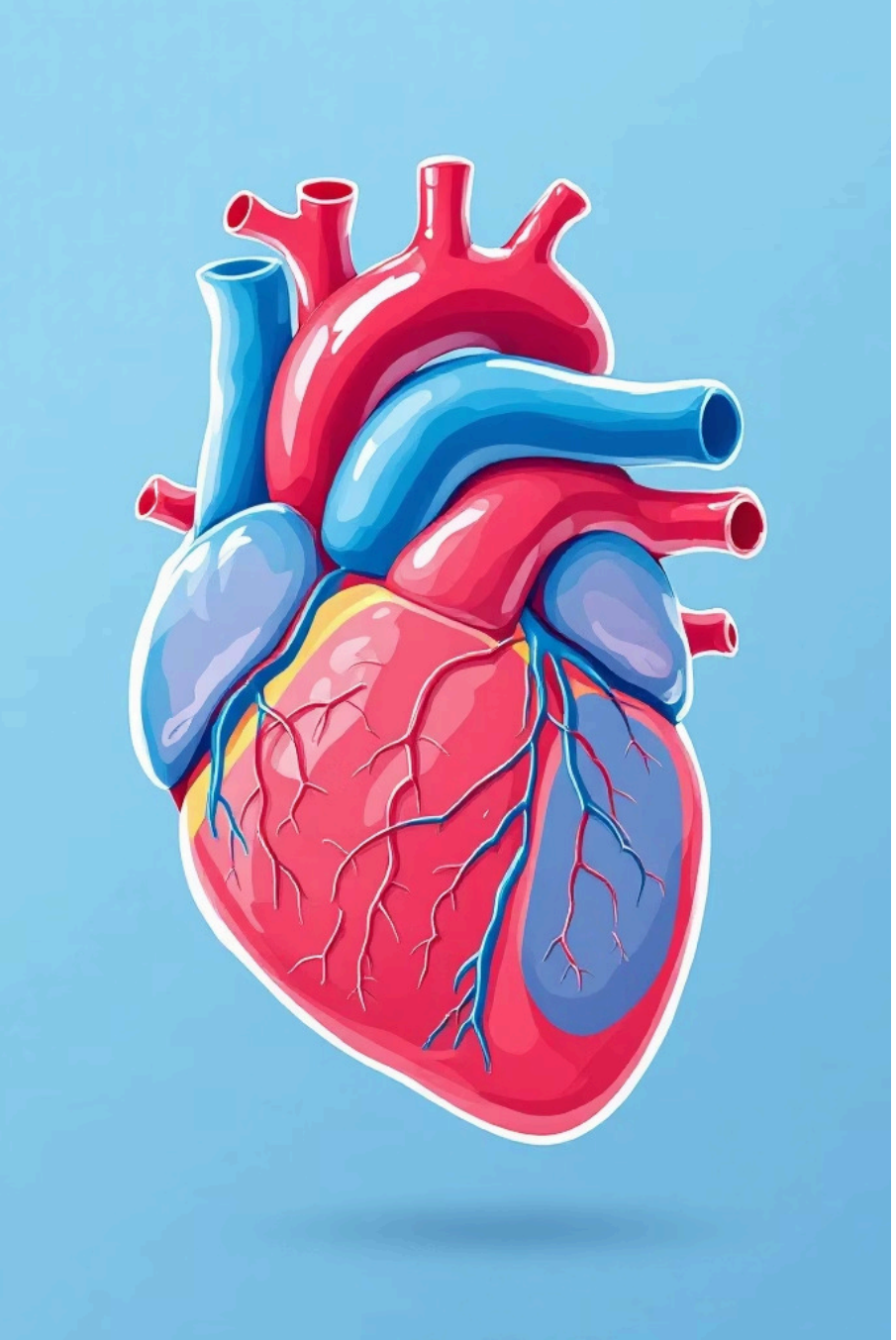


Projet de Prédiction du Risque de Maladie Cardiaque (CHD)

Table des matières

Table des matières

1. Introduction

1. Introduction

Les maladies cardiaques figurent parmi les premières causes de mortalité à l'échelle mondiale, entraînant chaque année la perte de millions de vies, d'après l'Organisation mondiale de la santé. Dans les pays développés comme les États-Unis, elles sont à l'origine d'environ un décès sur deux.

1.1. Contexte du Projet

Ce projet vise à développer un système de prédiction du risque de maladie cardiaque à 10 ans en utilisant des techniques de machine learning. L'objectif est d'analyser des données médicales pour identifier les facteurs de risque et prédire la probabilité de développer une maladie cardiaque.

1.2. Objectifs

- Analyser les données médicales pour identifier les facteurs de risque
- Développer un modèle prédictif précis
- Visualiser les résultats pour une meilleure compréhension
- Créer une solution scalable utilisant Apache Spark

2. Jeu de données

2. Jeu de données

Nous disposons de deux fichiers CSV pour notre analyse : train.csv et test.csv.

Ces fichiers contiennent les variables suivantes:

id	Identifiant unique pour chaque entrée dans les données.
age	Âge du patient, une variable continue représentant un facteur de risque potentiel.
education	Niveau d'éducation du patient, une variable nominale.
sex	Sexe du patient, représenté par "M" pour masculin et "F" pour féminin
is_smoking	Indique si le patient est un fumeur actuel, noté "YES" ou "NO".
cigsPerDay	Nombre moyen de cigarettes fumées par jour, une variable continue
BPMeds	Indique si le patient prend des médicaments contre l'hypertension, une variable nominale.
prevalentHyp	Indique si le patient a déjà eu un AVC, une variable nominale..
diabetes	Indique si le patient est diabétique, une variable nominale.
totChol	Taux de cholestérol total du patient, une variable continue.
sysBP	Tension artérielle systolique du patient, une variable continue
diaBP	Tension artérielle diastolique du patient, une variable continue.
BMI	Indice de masse corporelle du patient, une variable continue.
heartRate	Fréquence cardiaque du patient, une variable continue.
glucose	Niveau de glucose dans le sang du patient, une variable continue.

La variable cible TenYearCHD, qui représente le risque de maladie coronarienne sur 10 ans. C'est une variable binaire où "1" signifie "Oui" et "0" signifie "Non".

3. Architecture et Technologies

3. Architecture et Technologies

3.1. Stack Technologique

- **Langage de Programmation** : Java 8
- **Framework Principal** : Apache Spark 3.2.0
  - Spark Core
  - Spark SQL
  - Spark MLlib
- **Visualisation** : JFreeChart 1.5.3
- **Gestion de Projet** : Maven
- **Logging** : SLF4J avec Log4j

3.2. Structure du Projet

```
Projet/
├── src/
│   └── main/
│       ├── java/
│       │   └── spark/
│       │       └── batch/
│       │           ├── Main.java
│       │           ├── Data.java
│       │           └── Model.java
│       └── resources/
├── pom.xml
└── output/
```

4. Méthodologie

4. Méthodologie

Préparation des Données

1. **Chargement** :
    - Lecture des fichiers CSV
    - Inférence automatique du schéma
    - Vérification des en-têtes
2. **Prétraitement** :
    - Encodage des variables catégorielles
    - Gestion des valeurs manquantes
    - Normalisation des données
3. **Analyse Exploratoire** :
    - Distribution des variables
    - Corrélations
    - Visualisations

Modélisation

1. **Construction du Pipeline** :
    - Assemblage des caractéristiques
    - Normalisation
    - Régression logistique
2. **Validation Croisée** :
    - 5 plis
    - Optimisation des hyperparamètres
    - Évaluation par accuracy
3. **Prédiction** :
    - Application sur données de test
    - Export des résultats
    - Analyse des prédictions

5. Implémentation et Résultats

5. Implémentation et Résultats

5.1. Préparation des données

La préparation des données comprend un ensemble d'étapes essentielles visant à assurer leur qualité et leur cohérence avant d'entreprendre toute analyse ou modélisation. Ce processus englobe diverses tâches telles que :

5.1.1. Exploration du schéma des données:

La première étape consiste à analyser le schéma des données afin de bien comprendre la structure du jeu de données. Cela implique d'examiner les types de données associés à chaque colonne, les noms des colonnes, ainsi que toute information complémentaire sur les variables incluses.

```
root
 |-- id: integer (nullable = true)
 |-- age: integer (nullable = true)
 |-- education: double (nullable = true)
 |-- sex: string (nullable = true)
 |-- is_smoking: string (nullable = true)
 |-- cigsPerDay: double (nullable = true)
 |-- BPMeds: double (nullable = true)
 |-- prevalentStroke: integer (nullable = true)
 |-- prevalentHyp: integer (nullable = true)
 |-- diabetes: integer (nullable = true)
 |-- totChol: double (nullable = true)
 |-- sysBP: double (nullable = true)
 |-- diaBP: double (nullable = true)
 |-- BMI: double (nullable = true)
 |-- heartRate: double (nullable = true)
 |-- glucose: double (nullable = true)
 |-- TenYearCHD: integer (nullable = true)
```

5.1.2. Analyse de la distribution des données:

Une analyse de la distribution des données est réalisée afin d'extraire des statistiques descriptives telles que la moyenne, l'écart-type, les valeurs minimale et maximale, ainsi que les quartiles. Cette étape permet d'obtenir des informations précieuses sur la manière dont les valeurs sont réparties dans chaque colonne.

summary	id	age	education	sex	is_smoking	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp
count	3390	3390	3303	3390	3390	3368	3346	3390	3390
mean	1694.5	49.54218289085546	1.9709355131698456	null	null	9.069477434679335	0.029886431560071727	0.006489675516224189	0.3153392330383481
stddev	978.7530332009194	8.59287805192691	1.019080506396504	null	null	11.87907768177863	0.17029944333964098	0.08030854328226925	0.46471938562050297
min	0	32	1.0	F	NO	0.0	0.0	0	0
max	3389	70	4.0	M	YES	70.0	1.0	1	1

	diabetes	totChol	sysBP	diaBP	BMI	heartRate	glucose	TenYearCHD
	3390	3352	3390	3390	3376	3389	3086	3390
	0.02566371681415929	237.07428400954655	132.60117994100295	82.8830383480826	25.794964454976267	75.97727943346119	82.08651976668827	0.15073746312684366
	0.1581533077833387	45.24742978889774	22.292029541520748	12.023581073495865	4.1154487553601475	11.971868473429424	24.244753474941987	0.3578455731418609
	0	107.0	83.5	48.0	15.96	45.0	40.0	0
	1	696.0	295.0	142.5	56.8	143.0	394.0	1

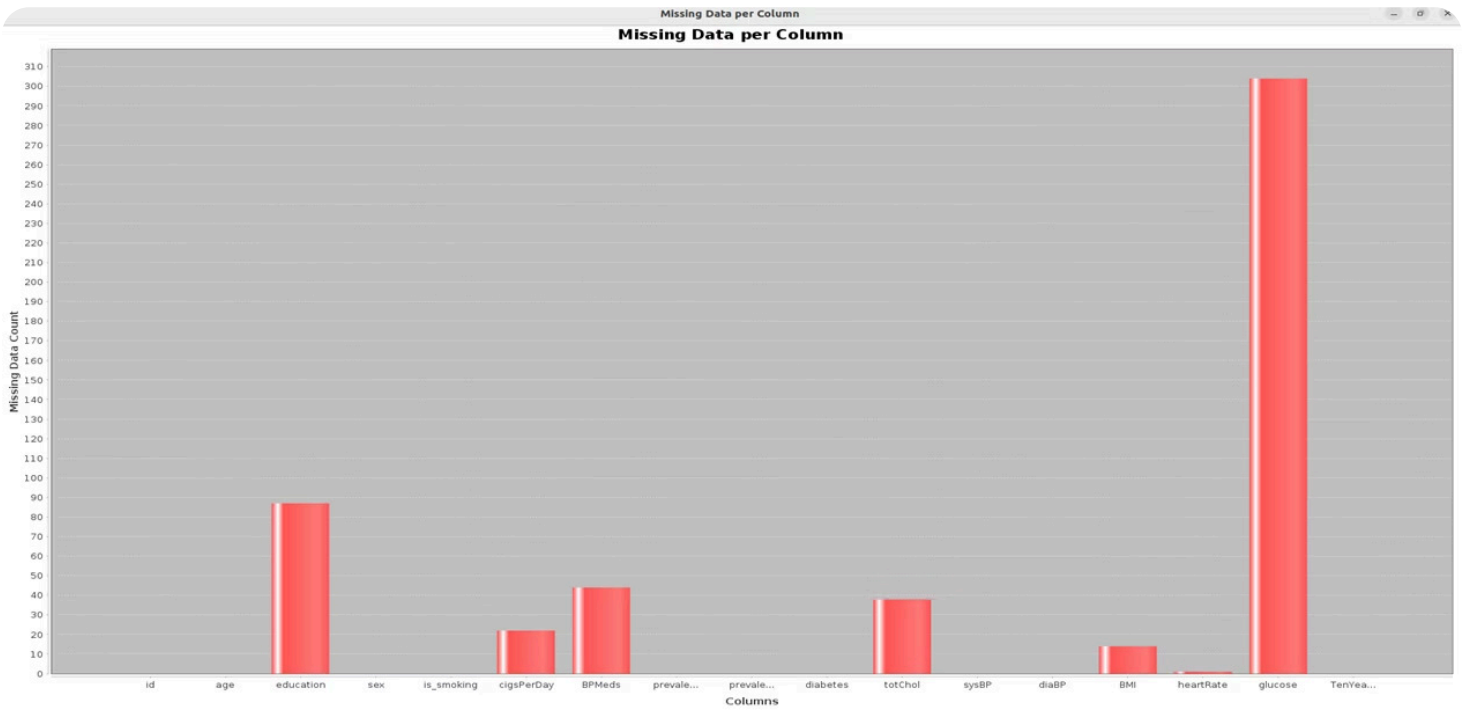
5. Implémentation et Résultats

5.1. Préparation des données

5.1.3 Gestion des données manquantes :

Les données manquantes sont fréquentes dans les jeux de données réels. Plusieurs méthodes peuvent être employées pour y remédier, notamment le remplacement des valeurs absentes par la moyenne ou la médiane des valeurs existantes dans la colonne concernée.

```
Nombre de données manquantes par colonne :
----------------------------------------
id                : 0
age               : 0
education         : 87
sex               : 0
is_smoking        : 0
cigsPerDay        : 22
BPMeds            : 44
prevalentStroke   : 0
prevalentHyp      : 0
diabetes          : 0
totChol           : 38
sysBP             : 0
diaBP             : 0
BMI               : 14
heartRate         : 1
glucose           : 304
TenYearCHD        : 0
```



5. Implémentation et Résultats

5.1. Préparation des données

5.1.4. Code:

Le code Java fourni réalise le prétraitement des données en vue de leur analyse. Il débute par l'affichage de la structure du jeu de données, puis effectue un nettoyage en convertissant les valeurs textuelles en données numériques et en identifiant les valeurs manquantes. Enfin, ces dernières sont remplacées par la médiane propre à chaque colonne concernée.

```java
public class Data {
    public static Dataset<Row> prepare(Dataset<Row> Data) {
        // Afficher le schéma des données de train
        Data.printSchema();
        // Visualiser la distribution des données de chaque colonne
        Data.describe().show();

        // Remplacer les valeurs "M" par 1 et "F" par 0 dans la colonne "sex"
        Data = Data.withColumn("sex", when(col("sex").equalTo("M"), 1).otherwise(0));
        // Remplacer les valeurs "Yes" par 1 et "No" par 0 dans la colonne "is_smoking"
        Data = Data.withColumn("is_smoking", when(col("is_smoking").equalTo("YES"), 1).otherwise(0));

        // Liste des noms de colonnes dans Data
        String[] columnNames = Data.columns();
        // Liste pour stocker le nombre de données manquantes pour chaque colonne
        long[] missingCount = new long[columnNames.length];

        // Calculer le nombre de données manquantes pour chaque colonne
        for (int i = 0; i < columnNames.length; i++) {
            missingCount[i] = Data.filter(col(columnNames[i]).isNull()).count();
        }

        // Afficher chaque colonne et le nombre de données manquantes
        System.out.println("Nombre de données manquantes par colonne :");
        System.out.println("----------------------------------------");
        for (int i = 0; i < columnNames.length; i++) {
            String columnName = columnNames[i];
            long count = missingCount[i];
            System.out.printf("%-20s : %-10d\n", columnName, count);
        }

// Remplacer les valeurs manquantes par la médiane de chaque colonne
        for (String colName : Data.columns()) {
            double median = Data.stat().approxQuantile(colName, new double[]{0.5}, 0.0)[0];
            Data = Data.na().fill(median, new String[]{colName});
        }
```

5.1.5. Résultats:

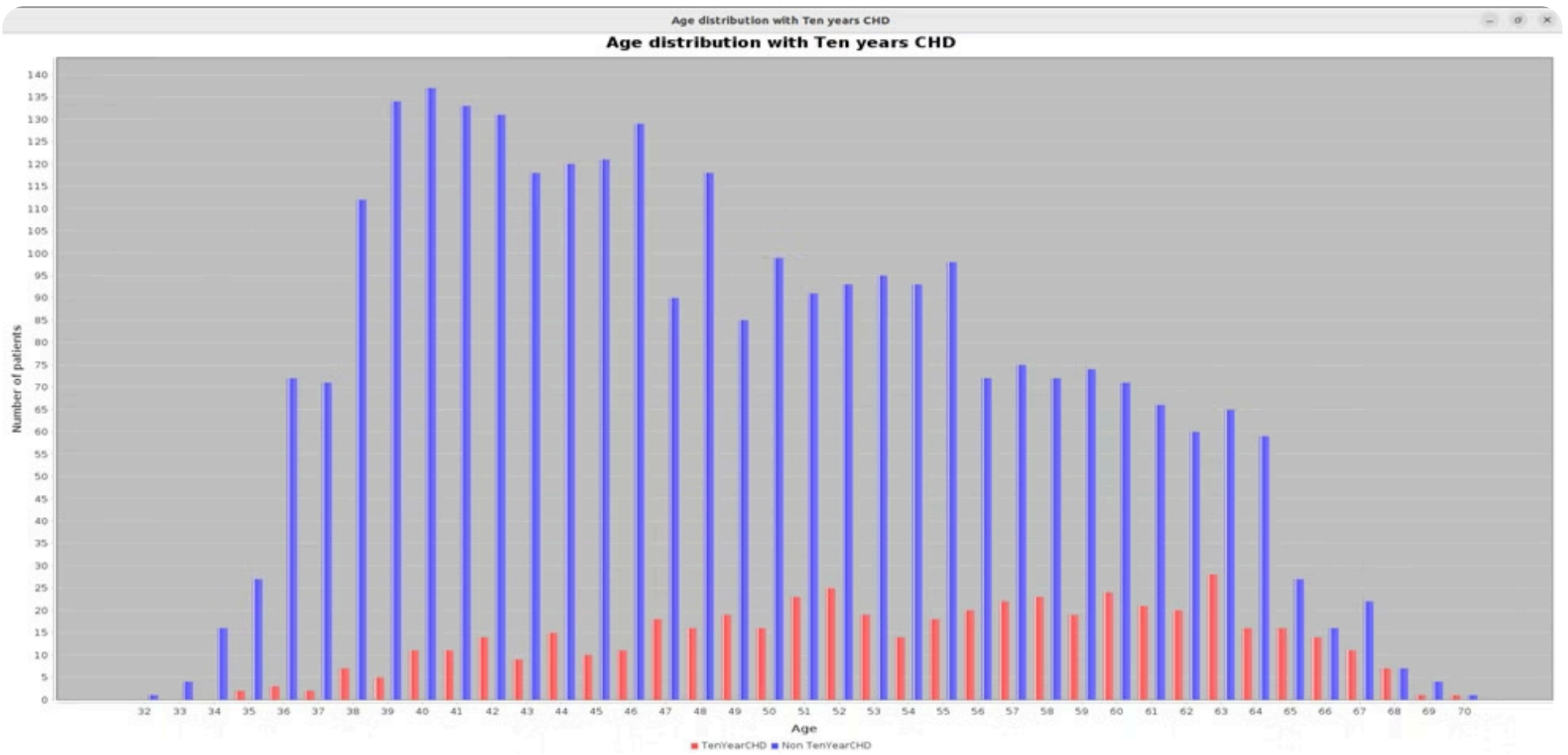
Voici les données après la préparation

```
+---+---+---------+---+----------+----------+------+--------------+-----------+--------+-------+-----+-----+-----+---------+-------+----------+
| id|age|education|sex|is_smoking|cigsPerDay|BPMeds|prevalentStroke|prevalentHyp|diabetes|totChol|sysBP|diaBP|  BMI|heartRate|glucose|TenYearCHD|
+---+---+---------+---+----------+----------+------+--------------+-----------+--------+-------+-----+-----+-----+---------+-------+----------+
|  0| 64|      2.0|  0|         1|       3.0|   0.0|             0|          0|       0|  221.0|148.0| 85.0|25.38|     90.0|   80.0|         1|
|  1| 36|      4.0|  1|         0|       0.0|   0.0|             0|          1|       0|  212.0|168.0| 98.0|29.77|     72.0|   75.0|         0|
|  2| 46|      1.0|  0|         1|      10.0|   0.0|             0|          0|       0|  250.0|116.0| 71.0|20.35|     88.0|   94.0|         0|
|  3| 50|      1.0|  1|         1|      20.0|   0.0|             0|          1|       0|  233.0|158.0| 88.0|28.26|     68.0|   94.0|         1|
|  4| 64|      1.0|  0|         1|      30.0|   0.0|             0|          0|       0|  241.0|136.5| 85.0|26.42|     70.0|   77.0|         0|
|  5| 61|      3.0|  0|         0|       0.0|   0.0|             0|          1|       0|  272.0|182.0|121.0| 32.8|     85.0|   65.0|         1|
|  6| 61|      1.0|  1|         0|       0.0|   0.0|             0|          1|       0|  238.0|232.0|136.0|24.83|     75.0|   79.0|         0|
|  7| 36|      4.0|  1|         1|      35.0|   0.0|             0|          0|       0|  295.0|102.0| 68.0|28.15|     60.0|   63.0|         0|
|  8| 41|      2.0|  0|         1|      20.0|   0.0|             0|          0|       0|  220.0|126.0| 78.0| 20.7|     86.0|   79.0|         0|
|  9| 55|      2.0|  0|         0|       0.0|   0.0|             0|          1|       0|  326.0|144.0| 81.0|25.71|     85.0|   78.0|         0|
| 10| 61|      1.0|  0|         0|       0.0|   0.0|             0|          1|       0|  234.0|185.0|121.0|35.22|     80.0|   78.0|         0|
| 11| 53|      2.0|  0|         0|       0.0|   0.0|             0|          0|       0|  210.0|138.0| 86.5|22.49|     88.0|   87.0|         0|
| 12| 43|      2.0|  0|         0|       0.0|   0.0|             0|          0|       0|  213.0| 96.0| 62.0|19.38|     74.0|   80.0|         0|
| 13| 44|      1.0|  1|         1|      40.0|   0.0|             0|          0|       0|  227.0|146.5| 97.0|26.92|     80.0|   67.0|         0|
| 14| 58|      3.0|  0|         0|       0.0|   0.0|             0|          1|       0|  188.0|160.0|120.0|35.58|     88.0|   85.0|         0|
| 15| 51|      1.0|  1|         1|      15.0|   0.0|             0|          0|       0|  212.0|146.0| 89.0|24.49|    100.0|  132.0|         1|
| 16| 50|      1.0|  0|         0|       0.0|   0.0|             0|          1|       0|  240.0|163.0|105.0|31.37|     89.0|   75.0|         0|
| 17| 44|      3.0|  0|         0|       0.0|   0.0|             0|          0|       0|  257.0|129.0| 93.0|27.56|     75.0|   76.0|         0|
| 18| 56|      3.0|  0|         0|       0.0|   0.0|             0|          0|       0|  267.0|122.5| 85.0|24.22|     92.0|  100.0|         0|
| 19| 42|      1.0|  1|         1|      30.0|   0.0|             0|          0|       0|  232.0|130.0| 91.0|25.77|     72.0|   70.0|         0|
+---+---+---------+---+----------+----------+------+--------------+-----------+--------+-------+-----+-----+-----+---------+-------+----------+
only showing top 20 rows
```

5. Implémentation et Résultats

5.2. Analyse des données de Train

5.2.1. Analyse par âge



Ce graphique indique une tendance à la hausse du risque de maladie coronarienne (CHD) avec l'âge, comme en témoigne la présence accrue de barres rouges dans les tranches d'âge plus élevées.

À l'inverse, la majorité des individus sans risque apparent de CHD appartiennent aux groupes d'âge les plus jeunes.

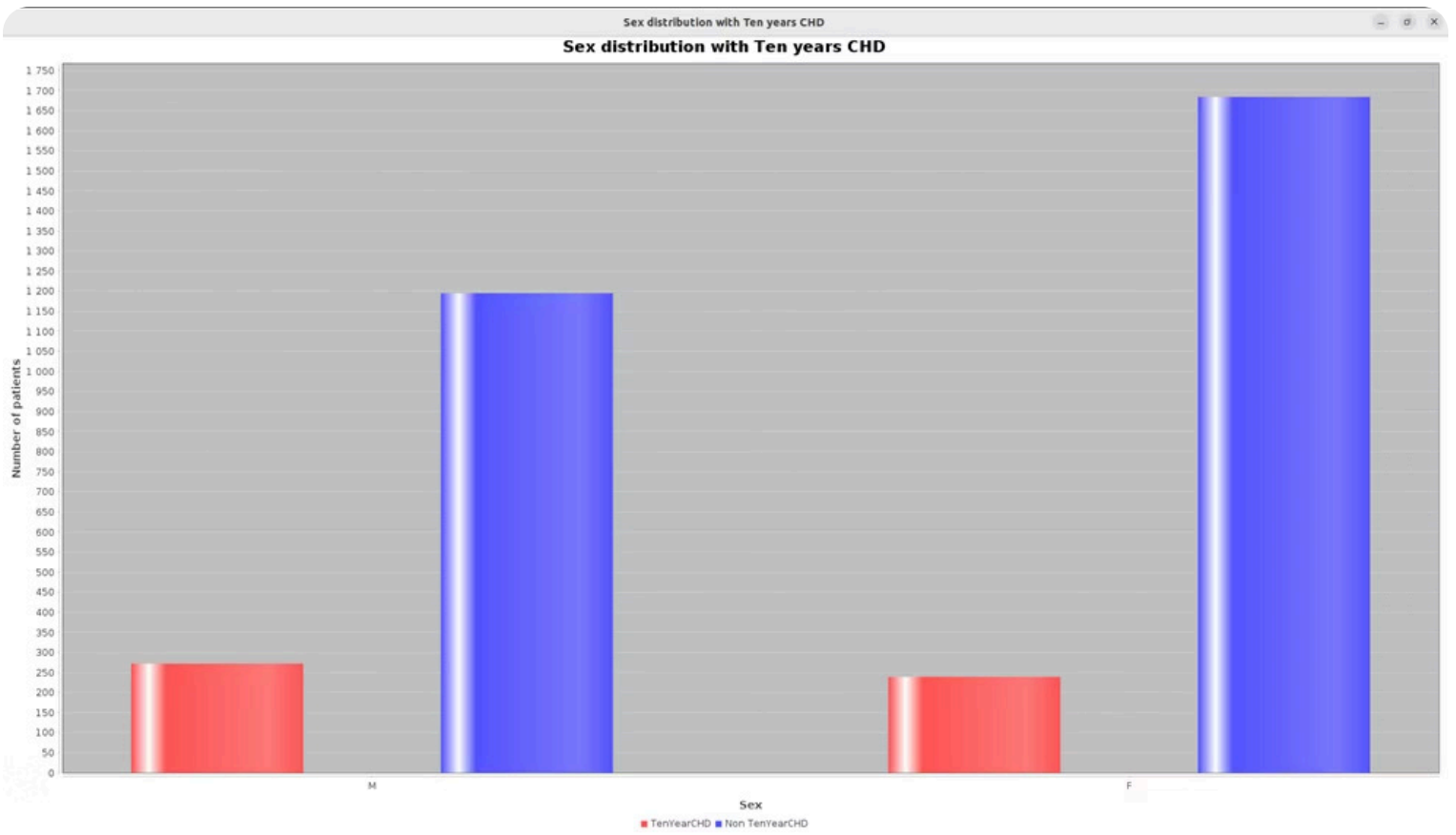
```java
public static void analyse(Dataset<Row> Data) {
    // Supprimer la colonne "id"
    Data = Data.drop("id");

    // Création du DataFrame avec le nombre de 'TenYearCHD' et 'Non TenYearCHD' par âge
    Dataset<Row> ageCounts = Data.groupBy("age").agg(
            sum(when(col("TenYearCHD").equalTo(1), 1).otherwise(0)).as("TenYearCHD"),
            count("age").as("Total")
    );

    ageCounts = ageCounts.orderBy("age"); // Tri du DataFrame par âge
    ageCounts.show(30); // Affichage du DataFrame

    // Création du dataset pour le graphique
    DefaultCategoryDataset ageDataset = new DefaultCategoryDataset();
    List<Row> ageRows = ageCounts.collectAsList();
    for (Row row : ageRows) {
        ageDataset.addValue(row.getLong(1), "TenYearCHD", row.get(0).toString());
        ageDataset.addValue(row.getLong(2) - row.getLong(1), "Non TenYearCHD", row.get(0).toString());
    }

    // Création du graphique
    JFreeChart ageChart = ChartFactory.createBarChart(
            "Age distribution with Ten years CHD", "Age", "Number of patients", ageDataset
    );

    // Affichage du graphique dans une fenêtre Swing
    ChartPanel ageChartPanel = new ChartPanel(ageChart);
    JFrame ageFrame = new JFrame("Age distribution with Ten years CHD");
    ageFrame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    ageFrame.add(ageChartPanel);
    ageFrame.pack();
    ageFrame.setVisible(true);
```

Ce code Java est conçu pour l'analyse de données médicales. Il commence par supprimer la colonne *id*, jugée non pertinente pour l'étude, puis regroupe les données selon l'âge. Ensuite, il calcule la répartition des cas de maladie et génère un graphique à barres afin de visualiser les résultats.

5. Implémentation et Résultats

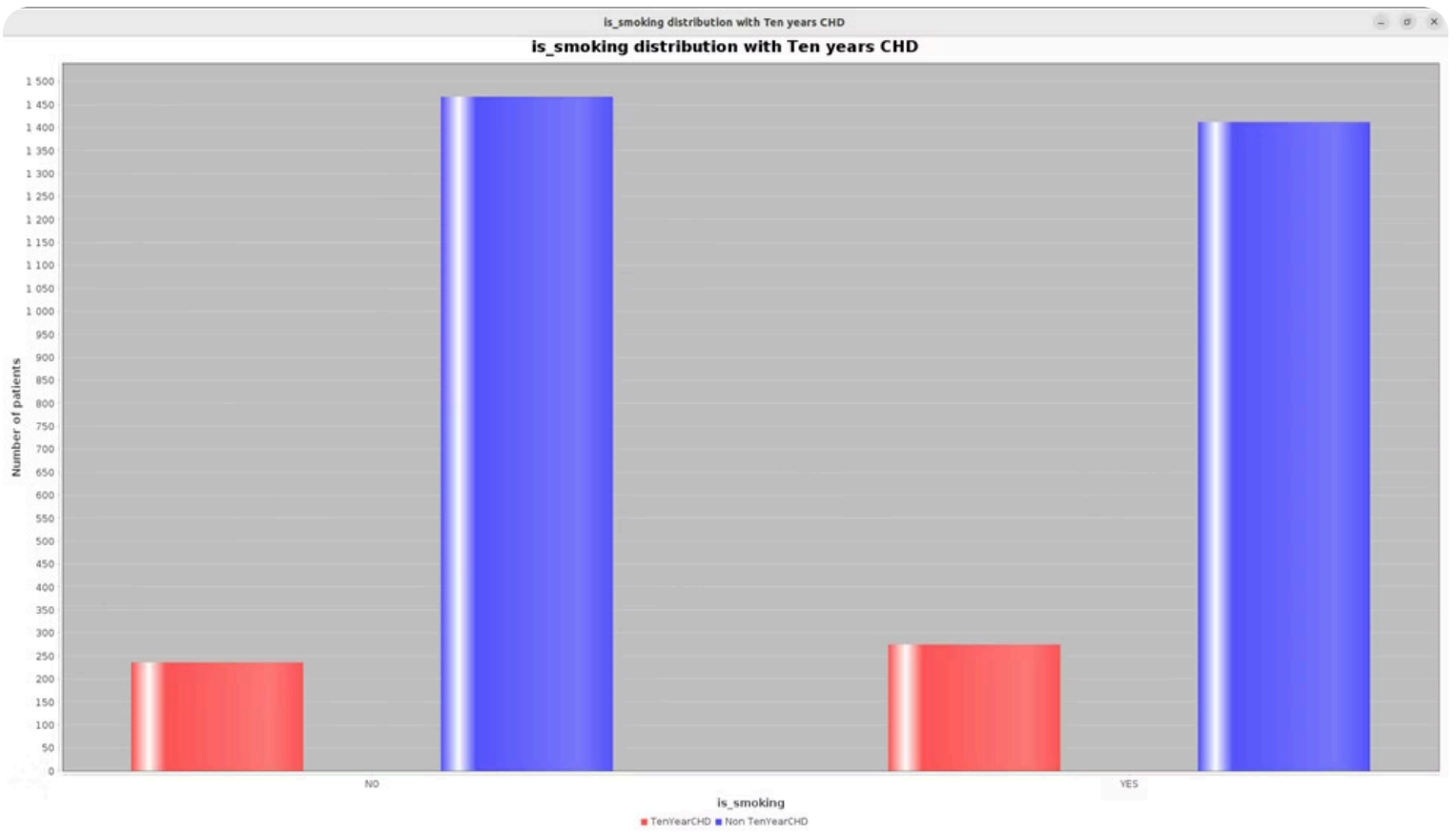
5.2. Analyse des données de Train

5.2.2. Analyse par sexe :



Ce graphique représente la répartition du risque prévu de maladie coronarienne (CHD) sur une période de dix ans selon le sexe.
On y observe une majorité de femmes dans la cohorte étudiée, mais les hommes présentent une proportion relativement plus élevée de risque prévu de CHD, comme l'indiquent les barres rouges.

5.2.3. Analyse par is_smoking:

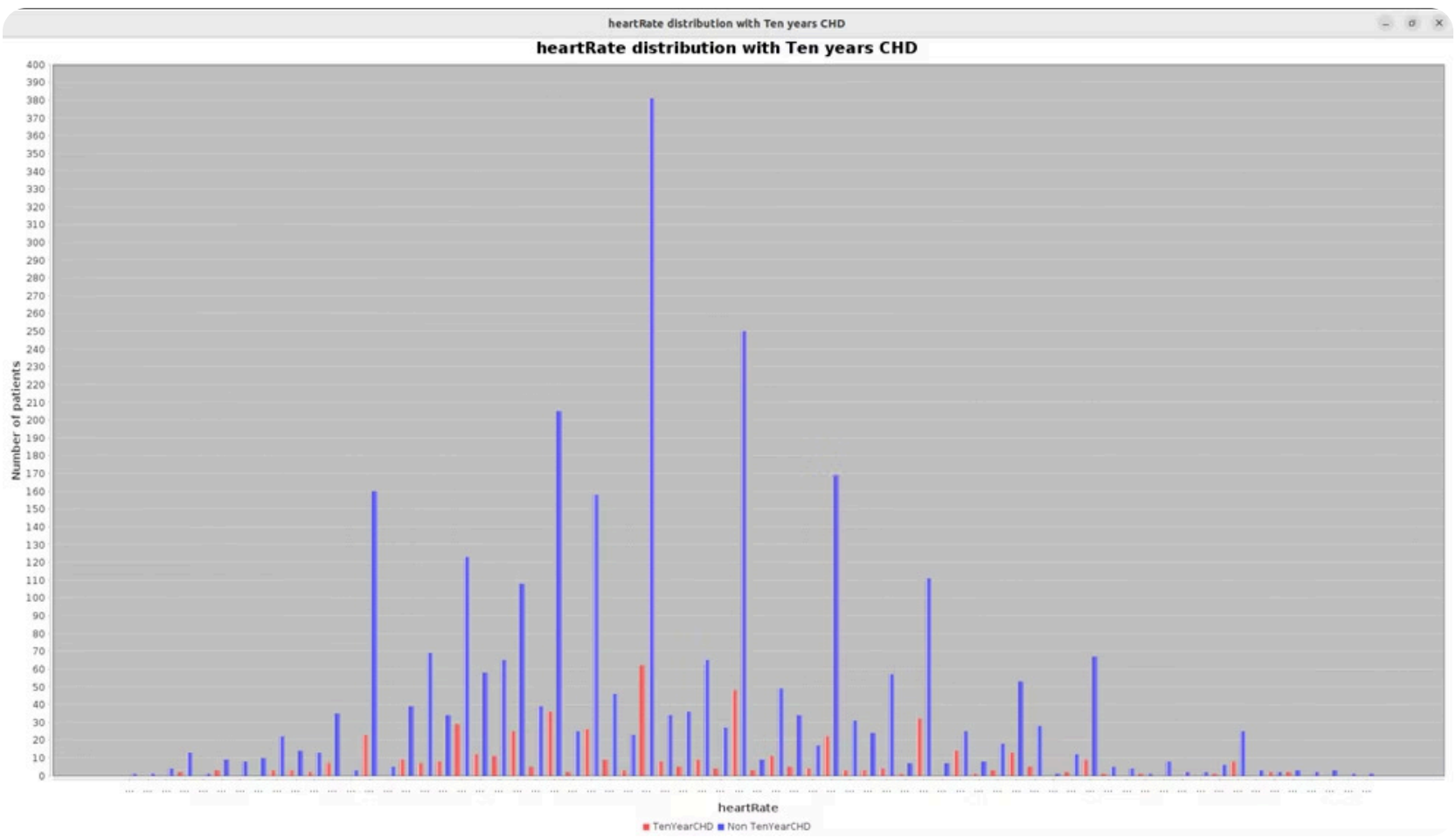


Ce graphique met en évidence qu'une part importante des fumeurs présente un risque élevé de maladie coronarienne (CHD), comme le montrent les barres rouges. En revanche, cette proportion est nettement plus faible chez les non-fumeurs.

5. Implémentation et Résultats

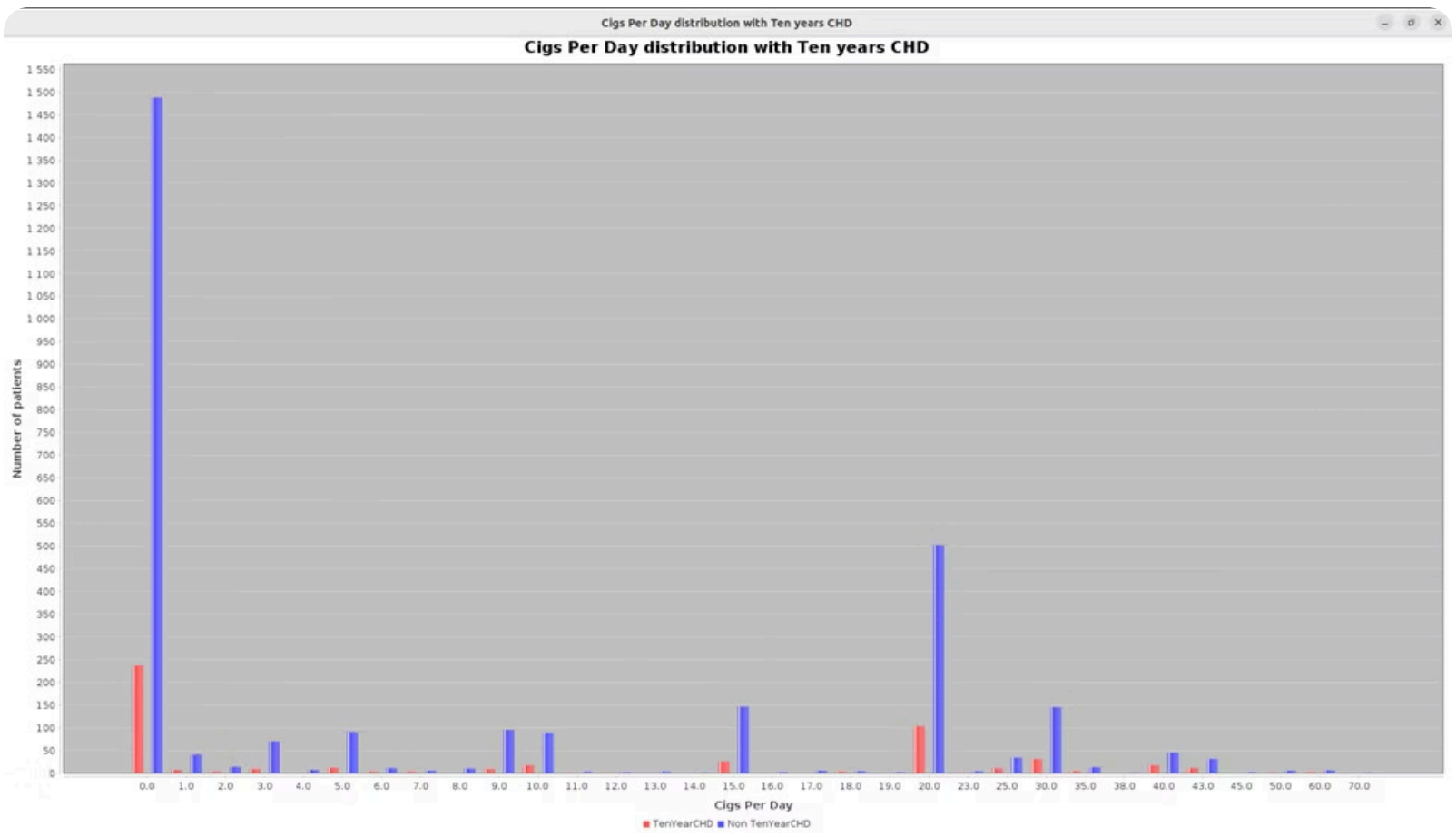
5.2. Analyse des données de Train

5.2.4. Analyse par heartRate:



Sur ce graphique, on remarque que la fréquence cardiaque des participants est très variable. Cependant, la plupart d'entre eux, quelle que soit leur fréquence cardiaque, ne sont pas considérés comme présentant un risque de maladie coronarienne (barres bleues). On observe néanmoins quelques participants avec des fréquences cardiaques particulièrement hautes ou basses qui sont identifiés comme étant à risque (barres rouges).

5.2.5. Analyse par cigsPerDay :



Ce graphique à barres illustre la relation entre le nombre de cigarettes fumées quotidiennement et le risque estimé de maladie cardiaque coronaire (CHD) sur dix ans.

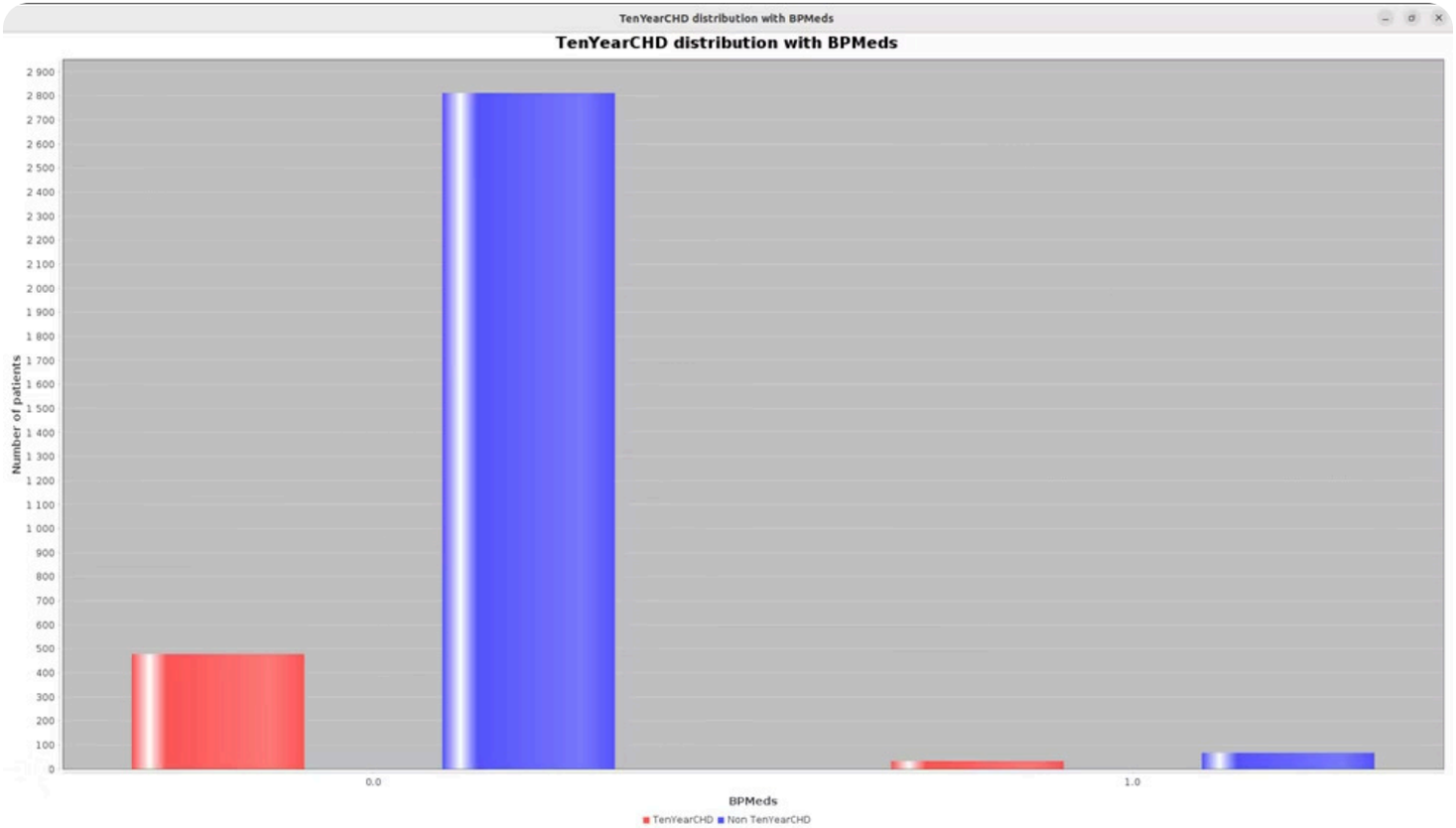
On constate que la majorité des participants, représentés par les barres bleues élevées, ne fument pas et présentent un faible risque de CHD.

Cependant, une minorité, signalée par les barres rouges, montre une consommation de cigarettes liée à un risque accru de CHD.

5. Implémentation et Résultats

5.2. Analyse des données de Train

5.2.6. Analyse par BPMeds :



Ce graphique à barres montre la relation entre la prise de médicaments contre la tension artérielle (BPMeds) et le risque de maladie cardiaque coronaire (CHD) sur dix ans.

6. Modèle

6. Modèle

6.1. Création du modèle

La régression logistique est une technique statistique couramment utilisée pour prédire des résultats binaires à partir de variables indépendantes. Elle permet notamment d'estimer la probabilité qu'un événement survienne en fonction de différentes caractéristiques.

Dans le cadre de notre projet, nous avons employé la régression logistique afin d'évaluer le risque de maladie coronarienne sur une période de 10 ans, en nous appuyant sur des facteurs démographiques, comportementaux et médicaux.

Cette méthode nous donne la possibilité de quantifier l'impact de chaque variable sur le risque de développer une maladie coronarienne, tout en identifiant les facteurs de risque les plus déterminants. Par ailleurs, nous avons intégré une validation croisée dans notre modèle afin de tester sa robustesse et sa capacité à produire des prédictions fiables sur de nouvelles données.

```java
public static CrossValidatorModel InitialiseyModel(Dataset<Row> trainData) {
    StandardScaler scaler = new StandardScaler()
    // Création du modèle de régression logistique
    LogisticRegression lr = new LogisticRegression()
            .setLabelCol("TenYearCHD")
            .setFeaturesCol("scaledFeatures")
            .setMaxIter(10000);

    // Création d'un pipeline pour enchaîner les étapes de prétraitement et le modèle de régression logistique
    Pipeline pipeline = new Pipeline()
            .setStages(new PipelineStage[]{assembler, scaler, lr});

    // Paramètres de la validation croisée
    ParamMap[] paramGrid = new ParamGridBuilder()
            .addGrid(lr.regParam(), new double[]{0.1, 0.01})
            .addGrid(lr.elasticNetParam(), new double[]{0.0, 0.5, 1.0})
            .build();

    // Création de l'évaluateur pour la validation croisée
    MulticlassClassificationEvaluator evaluator = new MulticlassClassificationEvaluator()
            .setLabelCol("TenYearCHD")
            .setPredictionCol("prediction")
            .setMetricName("accuracy");

    // Initialisation de la validation croisée
    CrossValidator cv = new CrossValidator()
            .setEstimator(pipeline)
            .setEvaluator(evaluator)
            .setEstimatorParamMaps(paramGrid)
            .setNumFolds(5); // Nombre de plis pour la validation croisée

    // Entraînement du modèle avec la validation croisée
    CrossValidatorModel cvModel = cv.fit(trainData);

    return cvModel;
}
```

6. Modèle

6.2. Test du modèle

Lors de la phase de test du modèle de régression logistique, plusieurs étapes clés sont effectuées pour évaluer sa performance :

1. **Division des données** : Les données d'entraînement (TrainData) sont séparées en deux ensembles distincts : 70 % sont réservés à l'entraînement (trainingData) et 30 % à la validation (testingData).
2. **Initialisation du modèle** : Le modèle de régression logistique, préalablement défini, est configuré. Cette étape comprend la préparation des caractéristiques (feature engineering), la mise à l'échelle des données, puis l'entraînement du modèle sur l'ensemble trainingData.
3. **Prédiction des résultats** : Une fois entraîné, le modèle est appliqué aux données de test (testingData) pour générer des prédictions, qui sont ensuite enregistrées dans un DataFrame nommé predictions.
4. **Évaluation de la performance** : Un évaluateur de classification binaire est mis en place afin de mesurer la performance du modèle à l'aide de la métrique d'exactitude (accuracy).
5. **Calcul de l'exactitude** : L'exactitude du modèle est ensuite calculée grâce à cet évaluateur, en se basant sur les prédictions réalisées sur l'ensemble de test.

Cette méthode d'évaluation permet de juger de manière objective la capacité du modèle à généraliser et à fournir des prédictions fiables sur des données inédites, un aspect crucial pour son application en contexte clinique ou décisionnel.

Accuracy de 86%:

On a évalué la performance du modèle avec la métrique (accuracy) avec un évaluateur.

le résultats a donné un taux de plus de quatre-vingts six pourcent

```
Accuracy: 0.8660194174757282
```

```java
public static void TestModel(Dataset<Row> trainData) {

    // Division des donnees en ensembles d'entrainement et de test
    Dataset<Row>[] splits = trainData.randomSplit(new double[]{0.7, 0.3}, 520);
    Dataset<Row> trainingData = splits[0];
    Dataset<Row> testingData = splits[1];

    //Initialisation du Model
    CrossValidatorModel model = InitialiseyModel(trainingData);

    // Predire les resultats sur les donnees de test
    Dataset<Row> predictions = model.transform(testingData);

    // Creation d'un evaluateur pour la classification
    MulticlassClassificationEvaluator evaluator = new MulticlassClassificationEvaluator()
            .setLabelCol("TenYearCHD")
            .setPredictionCol("prediction")
            .setMetricName("accuracy");

    // Calcul de l'exactitude
    double accuracy = evaluator.evaluate(predictions);

    // Affichage de l'exactitude
    System.out.println("Accuracy: " + accuracy);

}
```

6. Modèle

6.2. Résultats du modèle

Voici la distribution des caractéristiques de chaque patient avec leur prédiction.

id	age	education	sex	is_smoking	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp	diabetes	totChol	sysBP	diaBP	BMI	heartRate	glucose	prediction
3390	43	2.0	1	1	35.0	0.0	0	0	0	207.0	117.0	65.0	24.42	60.0	100.0	0.0
3391	56	3.0	0	0	0.0	0.0	0	0	0	192.0	122.0	82.5	28.61	68.0	58.0	0.0
3392	58	1.0	0	1	20.0	0.0	0	1	0	260.0	180.0	100.0	25.56	100.0	77.0	0.0
3393	47	3.0	0	0	0.0	0.0	0	0	0	231.0	102.5	66.0	23.4	70.0	78.0	0.0
3394	44	1.0	1	0	0.0	0.0	0	0	0	160.0	118.5	87.0	25.81	54.0	77.0	0.0
3395	41	2.0	0	1	7.0	0.0	0	0	0	260.0	101.0	68.0	22.49	80.0	77.0	0.0
3396	59	1.0	1	0	0.0	0.0	0	0	0	229.0	100.5	66.0	25.18	44.0	81.0	0.0
3397	41	1.0	0	1	15.0	0.0	0	0	0	242.0	139.0	80.0	19.68	72.0	60.0	0.0
3398	39	3.0	1	1	20.0	0.0	0	0	0	148.0	101.0	62.0	24.47	70.0	81.0	0.0
3399	38	2.0	0	1	3.0	0.0	0	0	0	180.0	115.0	86.0	24.91	70.0	77.0	0.0
3400	46	2.0	0	0	0.0	0.0	0	0	0	229.0	125.0	80.0	27.27	66.0	80.0	0.0
3401	37	2.0	1	1	20.0	0.0	0	0	0	232.0	129.0	74.0	24.46	86.0	88.0	0.0
3402	67	1.0	0	0	0.0	1.0	0	1	0	263.0	201.0	93.0	30.04	75.0	78.0	0.0
3403	61	1.0	1	0	0.0	0.0	0	1	0	239.0	122.0	83.0	28.85	62.0	94.0	0.0
3404	64	3.0	0	0	0.0	0.0	0	1	0	196.0	150.0	84.0	25.98	60.0	93.0	0.0
3405	47	1.0	0	0	0.0	1.0	0	1	0	277.0	138.5	99.0	39.64	85.0	81.0	0.0
3406	62	2.0	1	0	0.0	0.0	0	0	0	193.0	132.5	80.0	27.2	70.0	78.0	0.0
3407	44	2.0	0	1	3.0	0.0	0	0	0	239.0	103.0	67.0	26.58	66.0	73.0	0.0
3408	42	1.0	1	1	35.0	0.0	0	0	0	218.0	116.0	86.0	17.81	85.0	69.0	0.0
3409	63	1.0	0	0	0.0	0.0	0	1	0	306.0	195.0	105.0	27.96	75.0	87.0	0.0

only showing top 20 rows

Les résultats des prédictions:

prediction	prediction_count
0.0	840
1.0	8

On a obtenu 840 cas avec la prédiction 0 et 8 cas avec la prédiction 1.

6. Modèle

6.2. Résultats du modèle

Voila le fichier CSV de sortie qui correspond aux résultats présentés.

```
id,age,education,sex,is_smoking,cigsPerDay,BPMeds,prevalentStroke,prevalentHyp,diabetes,totChol,sysBP,diaBP,BMI,heartRate,glucose,prediction
3390,43,2.0,1,1,35.0,0.0,0,0,0,207.0,117.0,65.0,24.42,60.0,100.0,0.0
3391,56,3.0,0,0,0.0,0.0,0,0,0,192.0,122.0,82.5,28.61,68.0,58.0,0.0
3392,58,1.0,0,1,20.0,0.0,0,1,0,260.0,180.0,100.0,25.56,100.0,77.0,0.0
3393,47,3.0,0,0,0.0,0.0,0,0,0,231.0,102.5,66.0,23.4,70.0,78.0,0.0
3394,44,1.0,1,0,0.0,0.0,0,0,0,160.0,118.5,87.0,25.81,54.0,77.0,0.0
3395,41,2.0,0,1,7.0,0.0,0,0,0,260.0,101.0,68.0,22.49,80.0,77.0,0.0
3396,59,1.0,1,0,0.0,0.0,0,0,0,229.0,100.5,66.0,25.18,44.0,81.0,0.0
3397,41,1.0,0,1,15.0,0.0,0,0,0,242.0,139.0,80.0,19.68,72.0,60.0,0.0
3398,39,3.0,1,1,20.0,0.0,0,0,0,148.0,101.0,62.0,24.47,70.0,81.0,0.0
3399,38,2.0,0,1,3.0,0.0,0,0,0,180.0,115.0,86.0,24.91,70.0,77.0,0.0
3400,46,2.0,0,0,0.0,0.0,0,0,0,229.0,125.0,80.0,27.27,66.0,80.0,0.0
3401,37,2.0,1,1,20.0,0.0,0,0,0,232.0,129.0,74.0,24.46,86.0,88.0,0.0
3402,67,1.0,0,0,0.0,1.0,0,1,0,263.0,201.0,93.0,30.04,75.0,78.0,0.0
3403,61,1.0,1,0,0.0,0.0,0,1,0,239.0,122.0,83.0,28.85,62.0,94.0,0.0
3404,64,3.0,0,0,0.0,0.0,0,1,0,196.0,150.0,84.0,25.98,60.0,93.0,0.0
3405,47,1.0,0,0,0.0,1.0,0,1,0,277.0,138.5,99.0,39.64,85.0,81.0,0.0
3406,62,2.0,1,0,0.0,0.0,0,0,0,193.0,132.5,80.0,27.2,70.0,78.0,0.0
3407,44,2.0,0,1,3.0,0.0,0,0,0,239.0,103.0,67.0,26.58,66.0,73.0,0.0
3408,42,1.0,1,1,35.0,0.0,0,0,0,218.0,116.0,86.0,17.81,85.0,69.0,0.0
3409,63,1.0,0,0,0.0,0.0,0,1,0,306.0,195.0,105.0,27.96,75.0,87.0,0.0
3410,53,2.0,1,0,0.0,0.0,0,0,1,234.0,113.0,68.0,24.8,76.0,108.0,0.0
3411,56,1.0,0,1,15.0,0.0,0,0,0,269.0,121.0,75.0,22.36,50.0,66.0,0.0
3412,53,1.0,0,1,20.0,0.0,0,0,0,222.0,123.0,82.0,25.52,72.0,67.0,0.0
3413,39,4.0,0,1,9.0,0.0,0,0,0,180.0,113.0,73.0,17.65,70.0,73.0,0.0
3414,43,1.0,1,1,30.0,0.0,0,0,0,252.0,112.0,78.0,24.25,90.0,65.0,0.0
3415,64,1.0,0,0,0.0,0.0,0,0,0,295.0,127.0,78.0,22.89,67.0,73.0,0.0
3416,63,4.0,0,0,0.0,0.0,0,0,0,248.0,164.5,76.0,29.35,70.0,77.0,0.0
3417,51,1.0,1,1,20.0,0.0,0,0,0,232.0,112.5,74.0,26.37,70.0,77.0,0.0
3418,52,1.0,1,1,25.0,0.0,0,1,0,206.0,173.0,117.0,29.63,75.0,77.0,0.0
3419,64,2.0,1,0,0.0,0.0,0,0,0,193.0,114.0,79.0,16.59,75.0,64.0,0.0
3420,56,1.0,1,1,43.0,0.0,0,1,0,240.0,128.5,87.5,31.5,80.0,77.0,0.0
3421,53,2.0,0,1,30.0,0.0,0,0,0,250.0,149.5,95.0,28.02,68.0,77.0,0.0
3422,42,2.0,0,1,20.0,0.0,0,0,0,200.0,95.0,55.0,23.68,60.0,83.0,0.0
3423,50,1.0,1,1,20.0,0.0,0,0,0,259.0,108.0,81.0,22.81,80.0,72.0,0.0
3424,48,1.0,0,1,10.0,0.0,0,0,0,195.0,121.0,78.0,26.27,75.0,80.0,0.0
```

Code:

L'implémentation de la fonction qui affiche les résultats des prédictions , les enregistrer après dans un fichier csv et retourne à la fin les prédictions.

```java
public static Dataset<Row> ResultatsModel(Dataset<Row> trainData,Dataset<Row> testData,String outputFilePath) {

    // Model
    CrossValidatorModel model = InitialiseyModel(trainData);
    // Predire les resultats sur les donnees de test
    Dataset<Row> predictions = model.transform(testData);

    // Selectionner toutes les colonnes sauf "features" et "scaledFeatures"
    Dataset<Row> selectedPredictions = predictions
            .select("id", "age", "education", "sex", "is_smoking", "cigsPerDay", "BPMeds", "prevalentStroke",
                    "prevalentHyp", "diabetes", "totChol", "sysBP", "diaBP", "BMI", "heartRate", "glucose", "prediction");

    // Enregistrer les predictions dans un fichier de sortie au format CSV
    selectedPredictions.coalesce(1)  // Pour n'avoir qu'un seul fichier de sortie
            .write()
            .option("header", true)
            .csv(outputFilePath);

    // Afficher les predictions
    selectedPredictions.show();

    // Grouper par la colonne de prediction et compter le nombre d'occurrences de chaque valeur
    Dataset<Row> predictionCounts = predictions.groupBy("prediction")
            .count()
            .withColumnRenamed("count", "prediction_count");

    // Afficher les resultats
    predictionCounts.show();

    return predictions;
}
```

6. Modèle

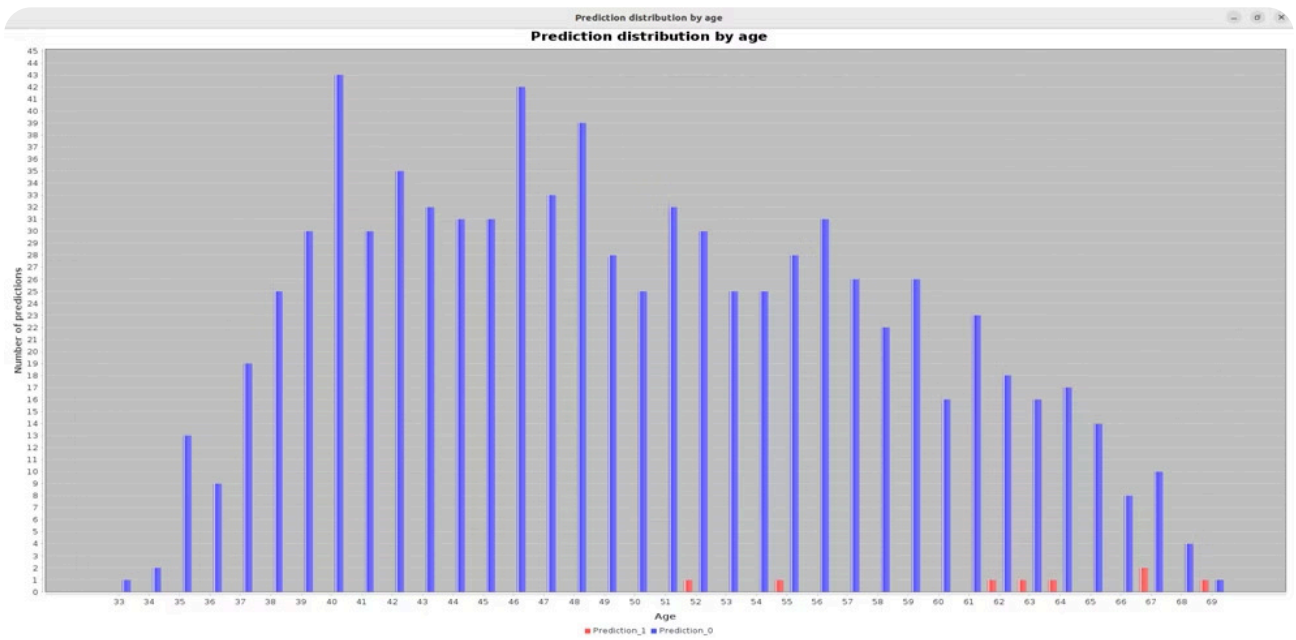
6.2. Analyse des prédictions

```java
public  void AnalysePredictions(Dataset<Row> predictions){
    // Création du DataFrame avec le nombre de prédictions 0 et 1 par âge
    Dataset<Row> ageCounts = predictions.groupBy("age").agg(
            sum(when(col("prediction").equalTo(1), 1).otherwise(0)).as("prediction_1"),
            sum(when(col("prediction").equalTo(0), 1).otherwise(0)).as("prediction_0"),
            count("age").as("Total")
    );
        // Tri du DataFrame par âge
        ageCounts = ageCounts.orderBy("age");
        // Affichage du DataFrame
        ageCounts.show(30);

        // Création du dataset pour le graphique
        DefaultCategoryDataset ageDataset = new DefaultCategoryDataset();
        List<Row> ageRows = ageCounts.collectAsList();
        for (Row row : ageRows) {
                ageDataset.addValue(row.getLong(1), "Prediction_1", row.get(0).toString());
                ageDataset.addValue(row.getLong(2), "Prediction_0", row.get(0).toString());
        }

        // Création du graphique
        JFreeChart ageChart = ChartFactory.createBarChart(
                "Prediction distribution by age", "Age", "Number of predictions", ageDataset
        );

        // Affichage du graphique dans une fenêtre Swing
        ChartPanel ageChartPanel = new ChartPanel(ageChart);
        JFrame ageFrame = new JFrame("Prediction distribution by age");
        ageFrame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        ageFrame.add(ageChartPanel);
        ageFrame.pack();
        ageFrame.setVisible(true);
```

Analyse par âge:

Les barres bleues représentent les prédictions 0, et les barres rouges représentent les prédictions 1

On voit que La tendance générale indique que le risque prévu de maladie avec (barres rouges) augmente avec l'âge.



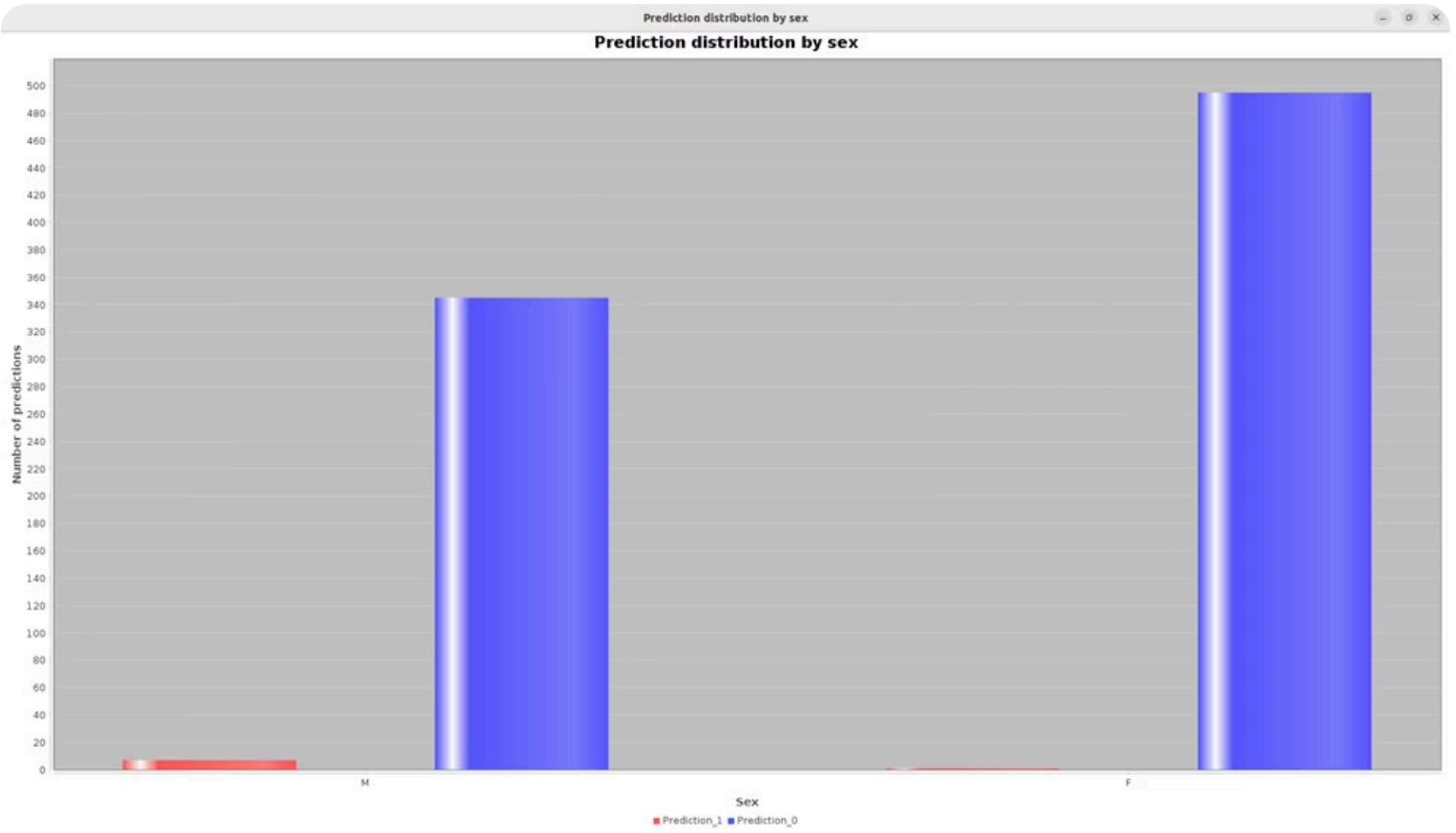
6. Modèle

6.2. Analyse des prédictions

Analyse par sexe:

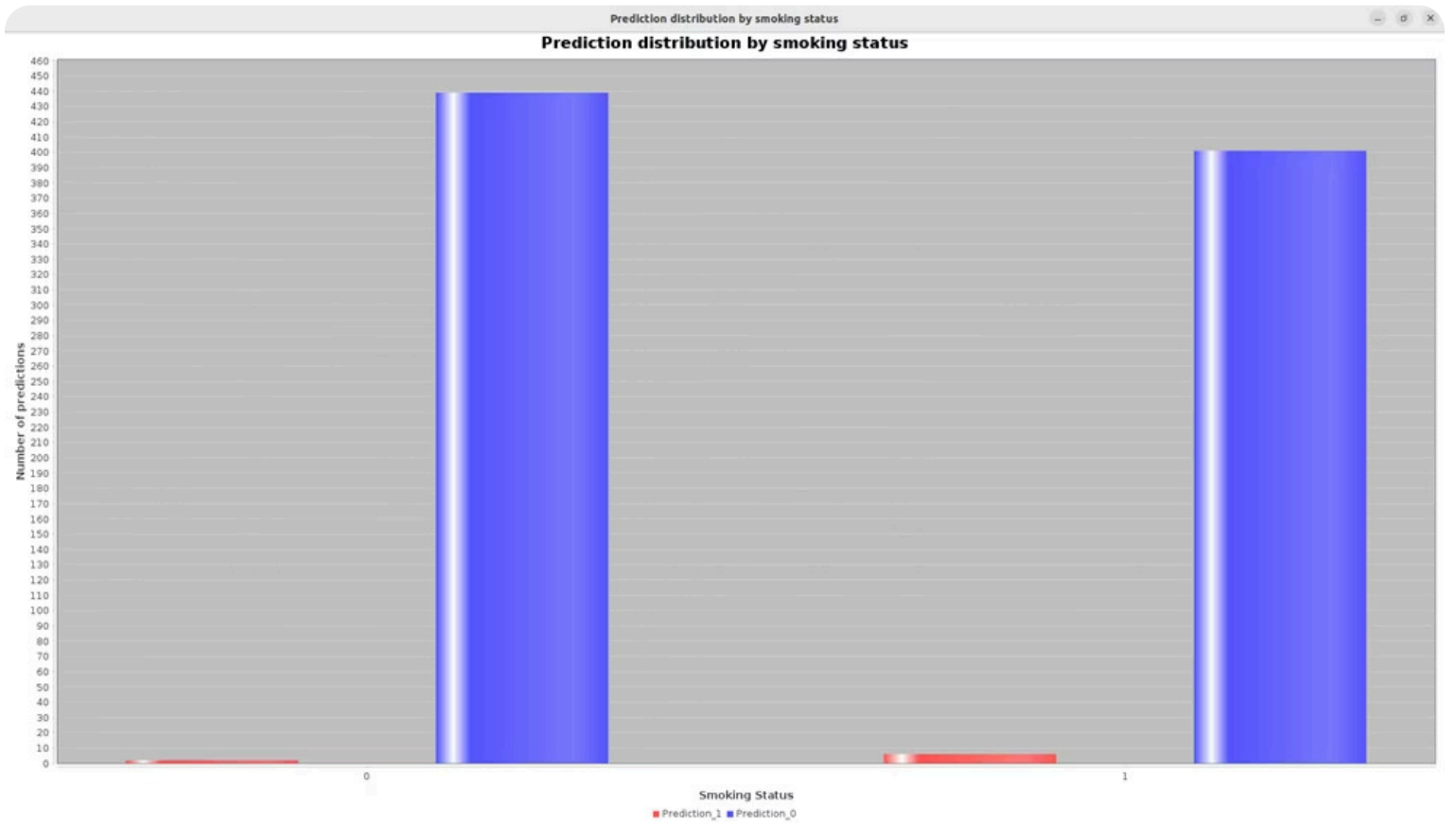
Les grandes barres bleues indiquent un grand nombre d'hommes et de femmes sans risque prévu de maladie.

Les petites barres rouges suggèrent qu'il y a très peu d'hommes plus que femmes pour lesquels le modèle prédit un risque avec (Prédiction 1).



Analyse par is_smoking:

Ce graphique à barres représente la prédiction de maladie selon le statut tabagique. Les barres rouges montrent qu'il y a très peu de personnes dans les deux catégories avec un taux plus élevé pour les personnes qui fument.

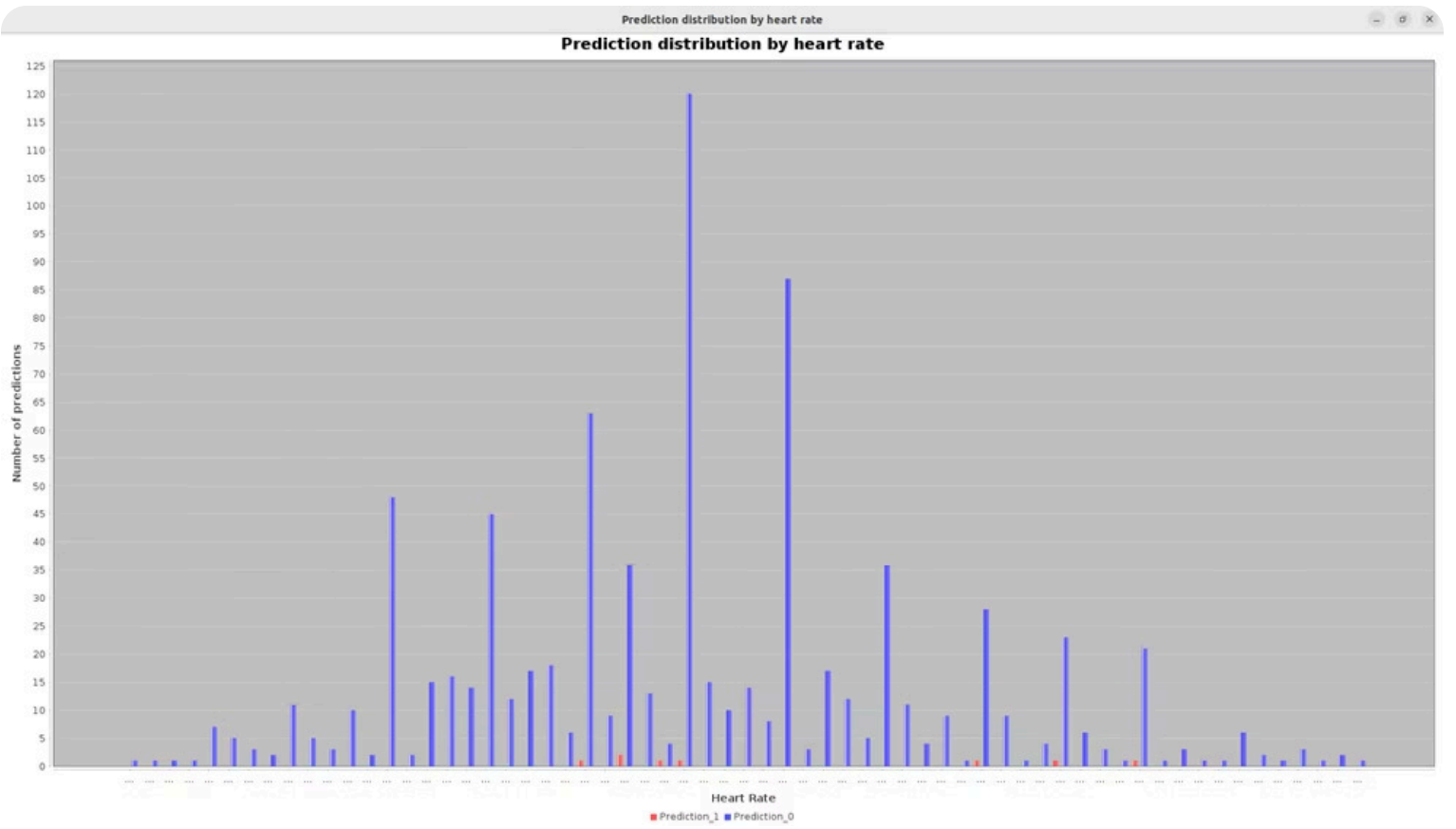


6. Modèle

6.2. Analyse des prédictions

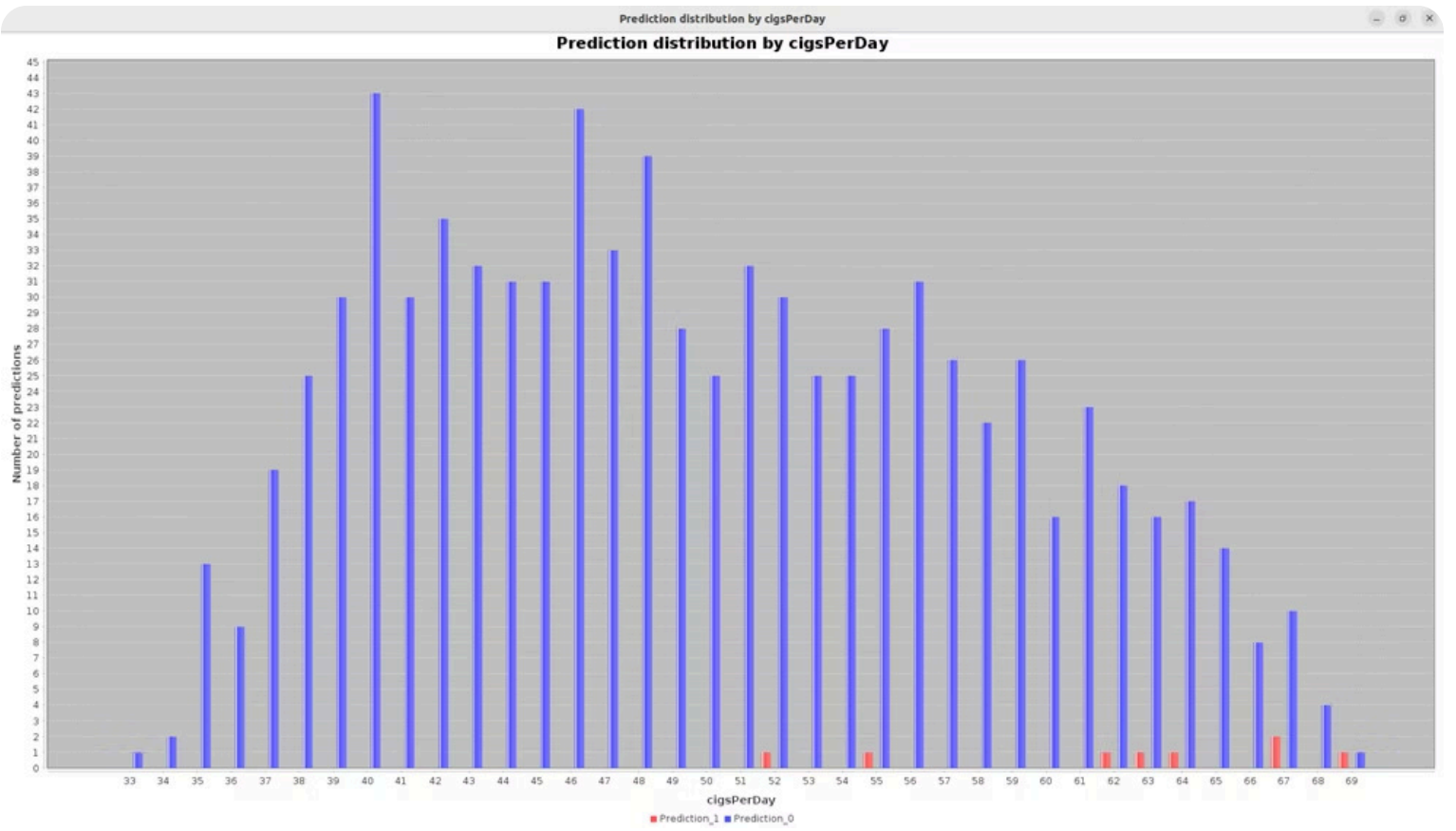
Analyse par heartRate:

En ce qui concerne la fréquence cardiaque la majorité des prédictions indiquent aucun risque prévu de la maladie étudié pour la majorité des fréquence avec quelques exceptions où le modèle prédit un risque de maladie à différentes fréquences cardiaques.



Analyse par cigsPerDay:

Pour le nombre de cigarettes fumées par jour les cas ayant une prédiction de 1 se trouvent parmi ceux avec des nombres élevés de cigarettes.

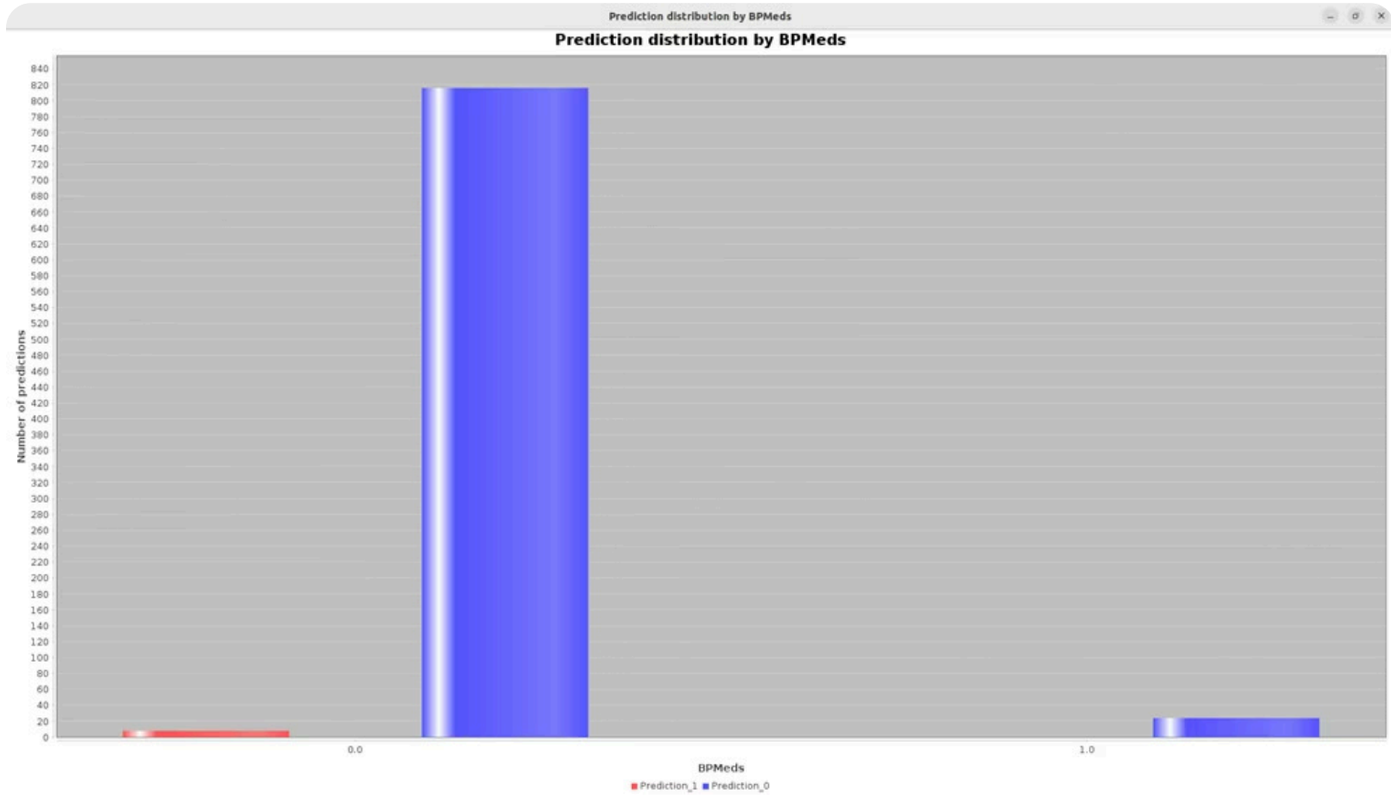


6. Modèle

6.2. Analyse des prédictions

Analyse par BPMeds:

Finalisant par analyse selon les individus prenant des médicaments contre l'hypertension. La distribution montre que les cas prévus avec une maladie concernent principalement les personnes qui n'utilisent pas ce type de médicaments.



7. Points Forts et Améliorations

7. Points Forts et Améliorations

7.1. Points Forts

1. **Architecture** :
   - Utilisation de Spark pour le traitement distribué
   - Pipeline ML complet et reproductible
   - Visualisations intégrées
2. **Techniques** :
   - Validation croisée robuste
   - Gestion avancée des valeurs manquantes
   - Visualisations interactives

7.2. Suggestions d'Amélioration

1. **Techniques** :
   - Implémentation d'autres modèles (Random Forest, SVM)
   - Ajout de métriques supplémentaires (AUC, F1-score)
   - Feature engineering plus avancé
2. **Architecture** :
   - Amélioration de la gestion des logs
   - Support du mode batch

8. Conclusion

8. Conclusion

Ce projet démontre l'application réussie des techniques de machine learning pour la prédiction du risque de maladie cardiaque. L'utilisation d'Apache Spark permet une solution scalable et performante, tandis que les visualisations facilitent l'interprétation des résultats.

L’emploi de la régression logistique, combiné à une préparation rigoureuse des données, a été déterminant pour développer des modèles prédictifs précis. Par ailleurs, des technologies telles qu’Apache Spark ont grandement contribué en offrant un traitement rapide et efficace de volumes importants de données, facilitant ainsi des prises de décision éclairées dans le domaine de la santé.

Merci pour votre attention !

MAHDA Kaoutar **&** ***MARRAH Douae***