Rapport_Ivanhoe_partie_prediction

November 17, 2023

1 Projet sous Spark : accidents de la circulation en France entre 2012 et 2018

Auteur : Ivanhoé Botcazou

Date : 6 novembre 2023

1.1 Prédiction des données avec les outils de Spark

Dans cette seconde partie nous aimerions expliquer à l'aide des données étudiées précédemment les risques de mort, blessure graves et autres pour les utilitaires de la route en France Métropolitaine.

Grandes étapes de notre travail :

- Sélection des données et choix des variables : concaténation d'un tableau source.
- Gestion des valeurs manquantes.
- Choix d'un modèle et entraînement.
- Test sur un échantillon final

[1]:
```
#Modules

import os
from pyspark.sql import SparkSession
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sn
import copy

from pyspark.sql.types import StringType #Type pour une colonne
from pyspark.sql.types import FloatType

from pyspark.sql.functions import *

from pyspark.ml.feature import Imputer

from pyspark.ml.classification import RandomForestClassifier

from pyspark.ml.feature import VectorAssembler
```

```
from pyspark.ml.evaluation import MulticlassClassificationEvaluator
```

[2]:
```
#Chargement des données

spark = SparkSession.builder.master("local").appName('Botcazou').getOrCreate()
 ↪#initialiser l'environement Spark


path = '/home/ibotcazou/Bureau/Master_data_science/DATAS_M2/
 ↪Informatique_charbonel_Marie/DATA_Marie/projet_spark'

Annees = range(2012,2019)

car,usa,lieux,vehi = {},{},{},{} #Dico qui vont contenir les DataFrames Spark

for a in Annees:
    car[f'car_{a}'] = spark.read.load(path + f"/caracteristiques_{a}.
 ↪csv",format="csv", sep=",", inferSchema="true", header="true")
    usa[f'usa_{a}'] = spark.read.load(path + f"/usagers_{a}.csv",format="csv",
 ↪sep=",", inferSchema="true", header="true")
    lieux[f'lieux_{a}'] = spark.read.load(path + f"/lieux_{a}.
 ↪csv",format="csv", sep=",", inferSchema="true", header="true")
    vehi[f'vehi_{a}'] = spark.read.load(path + f"/vehicules_{a}.
 ↪csv",format="csv", sep=",", inferSchema="true", header="true")
```

```
23/11/17 19:14:32 WARN Utils: Your hostname, ibotcazou-Latitude-7480 resolves to
a loopback address: 127.0.1.1; using 192.168.1.15 instead (on interface wlp2s0)
23/11/17 19:14:32 WARN Utils: Set SPARK_LOCAL_IP if you need to bind to another
address
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use
setLogLevel(newLevel).
23/11/17 19:14:33 WARN NativeCodeLoader: Unable to load native-hadoop library
for your platform… using builtin-java classes where applicable
```

[3]:
```
# Concat data in a large data set

car_12_18 = car['car_2012']
usa_12_18 = usa['usa_2012'].withColumn("âge", 2012 - col("an_nais")) # add a
 ↪column age
lieux_12_18 = lieux['lieux_2012']
vehi_12_18 = vehi['vehi_2012']

for a in Annees[1:]:  # Commencer à partir du deuxième élément, car le premier
 ↪est déjà dans df_12_18
    car_12_18 = car_12_18.unionByName(car[f'car_{a}'])
```

```
    usa_12_18 = usa_12_18.unionByName(usa[f'usa_{a}'].withColumn("âge", a -␣
 ↪col("an_nais")))
    lieux_12_18 = lieux_12_18.unionByName(lieux[f'lieux_{a}'] )
    vehi_12_18 = vehi_12_18.unionByName(vehi[f'vehi_{a}'])

car_12_18 = car_12_18.withColumnRenamed("col","coli") #change name of the column
```

[13]:
```
joindata = car_12_18.alias('c').join(vehi_12_18.alias('v'),col("c.Num_Acc") ==␣
 ↪col("v.Num_Acc")).join(usa_12_18.alias('u'),col("c.Num_Acc") == col("u.
 ↪Num_Acc")).join(lieux_12_18.alias('l'),col("c.Num_Acc") == col("l.Num_Acc"))

features_cols = ["an","mois","jour","hrmn","lum","agg","int",␣
 ↪"atm","coli","com",
    "catv","obs","obsm","choc","catu","sexe","trajet","âge","catr",
   ␣
 ↪"circ","nbv","vosp","prof","plan","lartpc","larrout","surf","infra","situ","env1"]

target_col = "grav"

data = joindata.select(features_cols + ["grav"]).filter(col('gps')=='M')

#Permet de gérer le valeur manquantes en mettant la moyenne à la place ou␣
 ↪encore la médiane
imputer = Imputer(inputCols=features_cols, outputCols=features_cols)
data = imputer.fit(data).transform(data)

# Créez un assembleur de vecteurs
vector_assembler = VectorAssembler(inputCols=features_cols,␣
 ↪outputCol="features")

# Transformez les données en utilisant l'assembleur de vecteurs
data = vector_assembler.transform(data)

data.show()
```

```
[Stage 109:================================================>        (6 + 1) / 7]
+---+----+----+----+---+---+---+---+----+---+----+---+----+----+----+----+------
+---+----+----+---+----+----+----+------+-------+----+-----+----+----+----+-----
---------------+
| an|mois|jour|hrmn|lum|agg|int|atm|coli|com|catv|obs|obsm|choc|catu|sexe|trajet
|âge|catr|circ|nbv|vosp|prof|plan|lartpc|larrout|surf|infra|situ|env1|grav|
features|
+---+----+----+----+---+---+---+---+----+---+----+---+----+----+----+----+------
+---+----+----+---+----+----+----+------+-------+----+-----+----+----+----+-----
---------------+
| 12|   3|  16|1930|  5|  2|  1|  1|   6| 11|   7|  0|   1|   7|   1|   2|
```

```
5| 73|   3|   2|  0|   0|   1|   1|     0|     72|   1|    0|   1|   0|
1|[12.0,3.0,16.0,19…|
| 12|   3|  16|1930|  5|  2|  1|  1|   6| 11|   7|  0|   1|   7|   3|   1|
5|  4|   3|   2|  0|   0|   1|   1|     0|     72|   1|    0|   1|   0|
4|[12.0,3.0,16.0,19…|
| 12|   9|   1|2145|  5|  2|  1|  1|   3| 11|   7|  0|   2|   3|   1|   1|
5| 18|   3|   2|  2|   0|   1|   1|     0|     65|   1|    0|   1|  99|
1|[12.0,9.0,1.0,214…|
| 12|   9|   1|2145|  5|  2|  1|  1|   3| 11|   7|  0|   2|   3|   2|   1|
0| 20|   3|   2|  2|   0|   1|   1|     0|     65|   1|    0|   1|  99|
1|[12.0,9.0,1.0,214…|
| 12|   9|   1|2145|  5|  2|  1|  1|   3| 11|   7|  0|   2|   3|   1|   1|
5| 36|   3|   2|  2|   0|   1|   1|     0|     65|   1|    0|   1|  99|
1|[12.0,9.0,1.0,214…|
| 12|   9|   1|2145|  5|  2|  1|  1|   3| 11|   7|  0|   2|   3|   2|   2|
0| 40|   3|   2|  2|   0|   1|   1|     0|     65|   1|    0|   1|  99|
3|[12.0,9.0,1.0,214…|
| 12|   9|   1|2145|  5|  2|  1|  1|   3| 11|  33|  0|   2|   8|   1|   1|
5| 18|   3|   2|  2|   0|   1|   1|     0|     65|   1|    0|   1|  99|
1|[12.0,9.0,1.0,214…|
| 12|   9|   1|2145|  5|  2|  1|  1|   3| 11|  33|  0|   2|   8|   2|   1|
0| 20|   3|   2|  2|   0|   1|   1|     0|     65|   1|    0|   1|  99|
1|[12.0,9.0,1.0,214…|
| 12|   9|   1|2145|  5|  2|  1|  1|   3| 11|  33|  0|   2|   8|   1|   1|
5| 36|   3|   2|  2|   0|   1|   1|     0|     65|   1|    0|   1|  99|
1|[12.0,9.0,1.0,214…|
| 12|   9|   1|2145|  5|  2|  1|  1|   3| 11|  33|  0|   2|   8|   2|   2|
0| 40|   3|   2|  2|   0|   1|   1|     0|     65|   1|    0|   1|  99|
3|[12.0,9.0,1.0,214…|
| 12|  11|  20|1815|  5|  2|  1|  1|   5|670|   7|  0|   2|   1|   1|   2|
5| 50|   3|   2|  2|   0|   1|   3|     0|     77|   1|    0|   1|   0|
1|[12.0,11.0,20.0,1…|
| 12|  11|  20|1815|  5|  2|  1|  1|   5|670|   7|  0|   2|   1|   1|   2|
5| 57|   3|   2|  2|   0|   1|   3|     0|     77|   1|    0|   1|   0|
1|[12.0,11.0,20.0,1…|
| 12|  11|  20|1815|  5|  2|  1|  1|   5|670|   7|  0|   2|   1|   2|   1|
0| 56|   3|   2|  2|   0|   1|   3|     0|     77|   1|    0|   1|   0|
1|[12.0,11.0,20.0,1…|
| 12|  11|  20|1815|  5|  2|  1|  1|   5|670|   7|  0|   2|   1|   1|   1|
5| 22|   3|   2|  2|   0|   1|   3|     0|     77|   1|    0|   1|   0|
1|[12.0,11.0,20.0,1…|
| 12|  11|  20|1815|  5|  2|  1|  1|   5|670|   7|  0|   2|   1|   2|   1|
0| 30|   3|   2|  2|   0|   1|   3|     0|     77|   1|    0|   1|   0|
3|[12.0,11.0,20.0,1…|
| 12|  11|  20|1815|  5|  2|  1|  1|   5|670|   7|  0|   2|   8|   1|   2|
5| 50|   3|   2|  2|   0|   1|   3|     0|     77|   1|    0|   1|   0|
1|[12.0,11.0,20.0,1…|
| 12|  11|  20|1815|  5|  2|  1|  1|   5|670|   7|  0|   2|   8|   1|   2|
```

```
5| 57|   3|   2|  2|   0|   1|   3|     0|     77|   1|    0|   1|   0|
1|[12.0,11.0,20.0,1…|
| 12|  11|  20|1815|  5|  2|  1|  1|   5|670|   7|  0|   2|   8|   2|   1|
0| 56|   3|   2|  2|   0|   1|   3|     0|     77|   1|    0|   1|   0|
1|[12.0,11.0,20.0,1…|
| 12|  11|  20|1815|  5|  2|  1|  1|   5|670|   7|  0|   2|   8|   1|   1|
5| 22|   3|   2|  2|   0|   1|   3|     0|     77|   1|    0|   1|   0|
1|[12.0,11.0,20.0,1…|
| 12|  11|  20|1815|  5|  2|  1|  1|   5|670|   7|  0|   2|   8|   2|   1|
0| 30|   3|   2|  2|   0|   1|   3|     0|     77|   1|    0|   1|   0|
3|[12.0,11.0,20.0,1…|
+---+----+----+----+---+---+---+---+----+---+----+---+----+----+----+------
+---+----+----+---+----+----+----+------+-------+----+-----+----+----+----+-----
---------------+
only showing top 20 rows
```

[14]:
```
# Fraction de données à utiliser pour l'ensemble d'entraînement
train_ratio = 0.8
test_ratio = 1 - train_ratio

# Divisez les données en ensembles d'entraînement et de test
train_data, test_data = data.randomSplit([train_ratio, test_ratio], seed=42)

train_data.show()
```

```
[Stage 123:>                                                        (0 + 1) / 1]
+---+----+----+----+---+---+---+---+----+---+----+---+----+----+----+------
+---+----+----+---+----+----+----+------+-------+----+-----+----+----+----+-----
---------------+
| an|mois|jour|hrmn|lum|agg|int|atm|coli|com|catv|obs|obsm|choc|catu|sexe|trajet
|âge|catr|circ|nbv|vosp|prof|plan|lartpc|larrout|surf|infra|situ|env1|grav|
features|
+---+----+----+----+---+---+---+---+----+---+----+---+----+----+----+------
+---+----+----+---+----+----+----+------+-------+----+-----+----+----+----+-----
---------------+
| 12|   1|   1|  15|  5|  2|  1|  2|   7|487|   7|  2|   0|   8|   1|   1|
5| 20|   4|   2|  2|   0|   1|   1|     0|     60|   2|    0|   4|  99|
3|[12.0,1.0,1.0,15…|
| 12|   1|   1|  15|  5|  2|  1|  2|   7|487|   7|  2|   0|   8|   2|   1|
0| 20|   4|   2|  2|   0|   1|   1|     0|     60|   2|    0|   4|  99|
1|[12.0,1.0,1.0,15…|
| 12|   1|   1|  45|  3|  1|  1|  1|   2|166|   7|  0|   2|   5|   1|   1|
4| 41|   1|   3|  3|   3|   1|   1|    20|    134|   9|    0|   1|  99|
3|[12.0,1.0,1.0,45…|
| 12|   1|   1|  45|  3|  1|  1|  1|   2|166|   7|  0|   2|   5|   1|   1|
```

```
5| 57|   1|   3|  3|   3|   1|   1|    20|    134|   9|    0|   1|  99|
3|[12.0,1.0,1.0,45…|
| 12|   1|   1|  45|  3|  1|  1|  1|   2|166|   7|  0|   2|   5|   2|   1|
0| 57|   1|   3|  3|   3|   1|   1|    20|    134|   9|    0|   1|  99|
2|[12.0,1.0,1.0,45…|
| 12|   1|   1|  45|  3|  1|  1|  1|   2|166|  10|  0|   2|   3|   1|   1|
5| 57|   1|   3|  3|   3|   1|   1|    20|    134|   9|    0|   1|  99|
3|[12.0,1.0,1.0,45…|
| 12|   1|   1|  45|  5|  2|  1|  1|   2| 55|   7|  0|   2|   1|   1|   1|
9| 29|   4|   1|  2|   0|   1|   2|     0|     74|   1|    1|   1|   0|
1|[12.0,1.0,1.0,45…|
| 12|   1|   1|  45|  5|  2|  1|  1|   2| 55|   7|  0|   2|   1|   1|   1|
9| 53|   4|   1|  2|   0|   1|   2|     0|     74|   1|    1|   1|   0|
4|[12.0,1.0,1.0,45…|
| 12|   1|   1|  45|  5|  2|  1|  1|   2| 55|   7|  0|   2|   1|   2|   1|
0| 79|   4|   1|  2|   0|   1|   2|     0|     74|   1|    1|   1|   0|
4|[12.0,1.0,1.0,45…|
| 12|   1|   1|  45|  5|  2|  1|  1|   2| 55|   7|  0|   2|   1|   2|   1|
0| 94|   4|   1|  2|   0|   1|   2|     0|     74|   1|    1|   1|   0|
4|[12.0,1.0,1.0,45…|
| 12|   1|   1|  45|  5|  2|  1|  1|   2| 55|   7|  0|   2|   4|   1|   1|
9| 53|   4|   1|  2|   0|   1|   2|     0|     74|   1|    1|   1|   0|
4|[12.0,1.0,1.0,45…|
| 12|   1|   1|  45|  5|  2|  1|  1|   2| 55|   7|  0|   2|   4|   2|   1|
0| 79|   4|   1|  2|   0|   1|   2|     0|     74|   1|    1|   1|   0|
4|[12.0,1.0,1.0,45…|
| 12|   1|   1|  45|  5|  2|  1|  1|   2| 55|   7|  0|   2|   4|   2|   1|
0| 94|   4|   1|  2|   0|   1|   2|     0|     74|   1|    1|   1|   0|
4|[12.0,1.0,1.0,45…|
| 12|   1|   1| 125|  5|  2|  1|  2|   6|557|   7|  2|   0|   9|   1|   1|
5| 34|   3|   2|  2|   0|   1|   1|     0|     85|   2|    0|   1|  99|
4|[12.0,1.0,1.0,125…|
| 12|   1|   1| 125|  5|  2|  1|  2|   6|557|   7|  2|   0|   9|   2|   2|
5| 32|   3|   2|  2|   0|   1|   1|     0|     85|   2|    0|   1|  99|
4|[12.0,1.0,1.0,125…|
| 12|   1|   1| 215|  5|  2|  1|  3|   1| 18|   7|  0|   2|   1|   1|   2|
0| 29|   3|   2|  2|   0|   1|   1|    31|     61|   2|    0|   1|   0|
1|[12.0,1.0,1.0,215…|
| 12|   1|   1| 215|  5|  2|  1|  3|   1| 18|   7|  0|   2|   1|   2|   1|
0| 21|   3|   2|  2|   0|   1|   1|    31|     61|   2|    0|   1|   0|
1|[12.0,1.0,1.0,215…|
| 12|   1|   1| 215|  5|  2|  1|  3|   1| 18|   7|  0|   2|   1|   2|   1|
0| 28|   3|   2|  2|   0|   1|   1|    31|     61|   2|    0|   1|   0|
1|[12.0,1.0,1.0,215…|
| 12|   1|   1| 215|  5|  2|  1|  3|   1| 18|  14|  0|   2|   3|   1|   2|
0| 29|   3|   2|  2|   0|   1|   1|    31|     61|   2|    0|   1|   0|
1|[12.0,1.0,1.0,215…|
| 12|   1|   1| 215|  5|  2|  1|  3|   1| 18|  14|  0|   2|   3|   2|   1|
```

```
0| 21|   3|   2|  2|   0|   1|   1|    31|     61|   2|    0|   1|   0|
1|[12.0,1.0,1.0,215…|
+---+----+----+----+---+---+---+---+----+---+----+---+----+----+----+----+------
+---+----+----+---+----+----+----+------+-------+----+-----+----+----+----+-----
---------------+
only showing top 20 rows
```

[15]:
```
# Créez le modèle de forêt aléatoire
rf_classifier = RandomForestClassifier(featuresCol="features",␣
 ↪labelCol=target_col, numTrees=200, maxDepth=10, seed=4)

# Entraînez le modèle sur l'ensemble d'entraînement
model = rf_classifier.fit(train_data)
```

```
23/11/17 19:26:41 WARN MemoryStore: Not enough space to cache rdd_1273_0 in
memory! (computed 105.9 MiB so far)
23/11/17 19:26:41 WARN BlockManager: Persisting block rdd_1273_0 to disk
instead.
23/11/17 19:26:57 WARN MemoryStore: Not enough space to cache rdd_1273_0 in
memory! (computed 362.6 MiB so far)
23/11/17 19:27:15 WARN MemoryStore: Not enough space to cache rdd_1273_0 in
memory! (computed 362.6 MiB so far)
23/11/17 19:27:39 WARN MemoryStore: Not enough space to cache rdd_1273_0 in
memory! (computed 362.6 MiB so far)
23/11/17 19:28:11 WARN MemoryStore: Not enough space to cache rdd_1273_0 in
memory! (computed 362.6 MiB so far)
23/11/17 19:28:55 WARN DAGScheduler: Broadcasting large task binary with size
1431.8 KiB
23/11/17 19:28:56 WARN MemoryStore: Not enough space to cache rdd_1273_0 in
memory! (computed 362.6 MiB so far)
23/11/17 19:29:54 WARN DAGScheduler: Broadcasting large task binary with size
2.5 MiB
23/11/17 19:29:56 WARN MemoryStore: Not enough space to cache rdd_1273_0 in
memory! (computed 362.6 MiB so far)
23/11/17 19:31:07 WARN DAGScheduler: Broadcasting large task binary with size
4.7 MiB
23/11/17 19:31:08 WARN MemoryStore: Not enough space to cache rdd_1273_0 in
memory! (computed 362.6 MiB so far)
23/11/17 19:32:45 WARN DAGScheduler: Broadcasting large task binary with size
1432.2 KiB
23/11/17 19:32:46 WARN DAGScheduler: Broadcasting large task binary with size
9.1 MiB
23/11/17 19:32:48 WARN MemoryStore: Not enough space to cache rdd_1273_0 in
memory! (computed 362.6 MiB so far)
23/11/17 19:34:44 WARN DAGScheduler: Broadcasting large task binary with size
```

```
2.7 MiB
23/11/17 19:34:47 WARN DAGScheduler: Broadcasting large task binary with size
17.7 MiB
23/11/17 19:34:48 WARN MemoryStore: Not enough space to cache rdd_1273_0 in
memory! (computed 362.6 MiB so far)
23/11/17 19:37:20 WARN DAGScheduler: Broadcasting large task binary with size
5.2 MiB
23/11/17 19:37:26 WARN DAGScheduler: Broadcasting large task binary with size
33.7 MiB
23/11/17 19:37:28 WARN MemoryStore: Not enough space to cache rdd_1273_0 in
memory! (computed 362.6 MiB so far)
23/11/17 19:40:22 WARN DAGScheduler: Broadcasting large task binary with size
9.6 MiB
23/11/17 19:40:29 WARN MemoryStore: Not enough space to cache rdd_1273_0 in
memory! (computed 362.6 MiB so far)
[Stage 232:=========================================>                (5 + 1) / 7]
```

[18]:
```
# Faites des prédictions sur l'ensemble de test
predictions_train = model.transform(train_data)

# Évaluez les performances du modèle
evaluator = MulticlassClassificationEvaluator(labelCol=target_col,␣
 ↪predictionCol="prediction", metricName="accuracy")
accuracy = evaluator.evaluate(predictions_train)
print(f"Accuracy train: {accuracy}")
```

```
23/11/17 19:44:05 WARN DAGScheduler: Broadcasting large task binary with size
26.6 MiB
[Stage 261:>                                                        (0 + 1) / 1]

Accuracy train: 0.5403528376177399
```

[16]:
```
# Faites des prédictions sur l'ensemble de test
predictions = model.transform(test_data)

# Évaluez les performances du modèle
evaluator = MulticlassClassificationEvaluator(labelCol=target_col,␣
 ↪predictionCol="prediction", metricName="accuracy")
accuracy = evaluator.evaluate(predictions)
print(f"Accuracy: {accuracy}")
```

```
23/11/17 19:41:28 WARN DAGScheduler: Broadcasting large task binary with size
26.6 MiB
[Stage 241:>                                                        (0 + 1) / 1]

Accuracy: 0.5371797093241936
```