Les Pima sont un groupe d'Amérindiens vivant en Arizona. Une prédisposition génétique a permis à ce groupe de survivre normalement à un régime pauvre en glucides pendant des années. Au cours des dernières années, en raison d'un passage soudain des cultures agricoles traditionnelles aux aliments transformés, ainsi que d'un déclin de l'activité physique, ils ont développé la prévalence la plus élevée de diabète de type 2 et pour cette raison ils ont fait l'objet de nombreuses études.

L'ensemble de données comprend des données provenant de 768 femmes présentant 8 caractéristiques, en particulier :

- Nombre de grossesses
- Concentration de glucose plasmatique a 2 heures dans un test de tolérance au glucose par voie orale
- Tension artérielle diastolique (mm Hg)
- Epaisseur du pli cutané du triceps (mm)
- 2 heures d'insuline sérique (mu U/ml)
- Indice de masse corporelle (poids en kg/(taille en m)^2)
- Le diabète pedigree fonction
- Âge (années) La dernière colonne de l'ensemble de données indique si la personne a reçu un diagnostic de diabète (1) ou non (0).

L'objectif est de déterminer quelles sont les caractéristiques (features) pour identifier les personnes qui ont un diabète de type 2

L'objectif est de poursuivre le TP précédente où de nombreux prétraitements ont été effecutés en ingénierie des données pour pouvoir faire un modèle de prédiction des survivants ou non.

Installation

Avant de commencer, il est nécessaire de déjà posséder dans son environnement toutes les librairies utiles. Dans la seconde cellule nous importons toutes les librairies qui seront utiles à ce notebook. Il se peut que, lorsque vous lanciez l'éxecution de cette cellule, une soit absente. Dans ce cas il est nécessaire de l'installer. Pour cela dans la cellule suivante utiliser la commande :

! pip install nom_librairie

Attention : il est fortement conseillé lorsque l'une des librairies doit être installer de relancer le kernel de votre notebook.

Remarque : même si toutes les librairies sont importées dès le début, les librairies utiles pour des fonctions présentées au cours de ce notebook sont ré-importées de manière à indiquer d'où elles viennent et ainsi faciliter la réutilisation de la fonction dans un autre projet.

```
# utiliser cette cellule pour installer les librairies manquantes
# pour cela il suffit de taper dans cette cellule : !pip install nom_librairie_m
# d'exécuter la cellule et de relancer la cellule suivante pour voir si tout se
# recommencer tant que toutes les librairies ne sont pas installées ...


#!pip install ..

# ne pas oublier de relancer le kernel du notebook
```

```
# Importation des différentes librairies utiles pour le notebook

#Sickit learn met régulièrement à jour des versions et
#indique des futurs warnings.
#ces deux lignes permettent de ne pas les afficher.
import warnings
warnings.filterwarnings("ignore", category=FutureWarning)

import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import sys
import pandas as pd
import numpy as np
import sklearn
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.metrics import precision_recall_fscore_support as score
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import RandomForestClassifier

#Sickit learn met régulièrement à jour des versions et indique des futurs warnin
#ces deux lignes permettent de ne pas les afficher.
import warnings
warnings.filterwarnings("ignore", category=FutureWarning)
```

Pour pouvoir lire et sauvegarder sur votre répertoire Google Drive, il est nécessaire de fournir une autorisation. Pour cela il suffit d'éxecuter la ligne suivante et de saisir le code donné par Google.

```
from google.colab import drive
drive.mount('/content/gdrive/')
```

```
Mounted at /content/gdrive/
```

Corriger éventuellement la ligne ci-dessous pour mettre le chemin vers un répertoire spécifique dans votre répertoire google drive :

```
my_local_drive='/content/gdrive/My Drive/Colab Notebooks/ML_FDS'
# Ajout du path pour les librairies, fonctions et données
sys.path.append(my_local_drive)
# Se positionner sur le répertoire associé
%cd $my_local_drive

%pwd
```

```
/content/gdrive/My Drive/Colab Notebooks/ML_FDS
'/content/gdrive/My Drive/Colab Notebooks/ML_FDS'
```

Travaux pratiques

Récupérer le fichier pima-indians-diabetes.csv et le mettre dans un dataframe. Attention les premières lignes correspondent à la description des données. Il est possible de ne pas les lire en mettant skiprows=9 dans la fonction read_csv.

```

```

Afficher le nombre de ligne et de colonnes du dataframe ainsi que les 5 premières lignes

```

```

Afficher la matrice de corrélation. Rappel il faut utiliser la fonction corr().

```

```

Afficher, à l'aide de seaborn, la matrice de correlation

Il est important d'analyser les histogrammes de chaque variable pour mieux comprendre comment les données sont réparties.
A l'aide du code suivant, afficher les différents histogrammes.

```
import matplotlib.pyplot as plt
df.hist(bins=50, figsize=(20, 15))
plt.show()
```

Existe-t'il des valeurs nulles ? Existe-til des valeurs manquantes ? Rappel vous pouvez le voir avec des histogrammes mais aussi avec une heatmap.

En fait on peut constater qu'il n'y a pas de valeurs manquantes avec le heatmap mais par contre il y a des valeurs nulles. Il faut toujours faire attention à la manière dont sont codées les valeurs manquantes. Ici nous voyons dans les histogrammes que pour BMI, BloodP, PlGlcConc, SkinThick, TwoHourSerIns il existe des valeurs manquantes. Le nombre de grossesses n'est pas considéré comme une valeur manquante bien sûr.

Transformer les valeurs nulles par la médiane de la série.

Les données sont, à présent, transformées et nous allons pouvoir créer un jeu de données de test et d'apprentissage. Faire une copie du dataframe en df2. Sur df appliquer un scaling pour normaliser les valeurs par rapport à la moyenne et l'écart type (utilisation de StandardScaler (). Nous conservons la copie df2 sans transformation.
L'objectif à présent est d'appliquer différents classifieurs pour voir celui qui est le plus performant. Pour le ou les meilleurs rechercher les hyperparamètres et créer un pipeline à sauvegarder. Il faut ensuite pouvoir traiter de nouvelles données pour prédire si il y a diabète ou pas.

Tester les résultats sur df et sur df2.