جامعة مولاي إسماعيل
UNIVERSITÉ MOULAY ISMAÏL

كلية العلوم
FACULTÉ DES SCIENCES

Bachelor in
Computer Engineering

Année Universitaire 2021-2022
BACHELOR EN GENIE INFORMATIQUE
Semestre S5

Rapport de projet sur l'intelligence artificielle

ÉVALUATION DE PLUSIEUR MODÈLE DE MACHINE LEARNING

Réalisé par : ELKAAM Hiba

Présenté le 04/02/2022 devant le jury :

BenhlimaSaid de la faculté des sciences-Meknès
Oubelkacem Ali de la faculté des sciences-Meknès
Bekri Alide la faculté des sciences-Meknès
Ba-Ichou Ayoubde la faculté des sciences-Meknès

Dédicaces

A ***Notre chers parents***

Pour leurs sacrifices, leurs prières et leurs soutiens.

A ***Notre amis et mes collègues***
Nous vous souhaitons la prospérité et le succès.

A ***tous nos enseignants avec notre considération***
Qui n'ont épargné aucun effort pour nous offrir un bon enseignement.

Et à tous ceux et celles qui nous ont aidés dans la réalisation et le bon déroulement de ce projet.

Remerciements

Tout d'abord, nous remercions à notre Dieu de nous avoir donné la force de faire ce travail modeste.

Nous tenons à exprimer notre profonde gratitude ainsi que toute notre reconnaissance à notre cher encadrant Monsieur Ali Bekri, Professeur dans la faculté des sciences de Meknès, qui nous a fait avantage de son savoir-faire, de ses conseils, de ses directives, de sa disponibilité et pour l'intérêt manifeste qu'il a porté à ce projet.

Nous adressons nos vifs remerciements à tous les enseignants, et précisément ceux de la filière Mathématiques et Informatiques pour leurs contributions à notre formation académique, ainsi qu'aux membres de jury pour l'honneur qu'ils nous ont fait en acceptant de siéger à notre soutenance.

Que tous ceux et celles qui ont aidé à l'achèvement de ce projet trouvent l'expression de nos remerciements.

Nos remerciements à nos chers parents, pour leur assistant à travers de nos parcours pédagogique.

Abstract

Le diabète est une maladie chronique qui pourrait causer une crise mondiale des soins de santé. Selon 382 millions de personnes vivent avec le diabète dans le monde entier. D'ici 2035, le diabète sucré ou tout simplement le diabète est une maladie causée par l'augmentation glycémie. Diverses méthodes traditionnelles, basées sur des tests physiques et chimiques, sont disponibles pour diagnostiquer le diabète. Cependant, la prévision précoce du diabète est une tâche assez difficile pour les médecins en raison de l'interdépendance complexe de divers facteurs.

Les méthodes de science des données ont le potentiel de bénéficier à d'autres domaines scientifiques en apportant un nouvel éclairage sur des questions courantes. Une de ces tâches est d'aider à faire des prédictions sur les données médicales. L'apprentissage automatique est un domaine scientifique émergent dans la science des données traitant de la façon dont les machines apprennent de l'expérience

Introduction Générale

L’objectif de ce projet est de développer un système qui peut effectuer la prévision précoce du diabète pour un patient avec un plus grande précision en combinant les résultats de différentes techniques d’apprentissage automatique.

Ce projet vise à diabète via trois différentes méthodes d’apprentissage machine supervisées, y compris : SVM, régression logistique, KNN, DTC,RF.

Ce projet vise également à proposer une technique efficace de détection précoce de la maladie diabétique.

Chapitre 1:contexte générale et description du Projet

Machine learning:

Le but de machine learning est la construction de l'ordinateur systèmes qui peuvent s'adapter et apprendre de leur expérience. Une définition plus détaillée et plus formelle de l'apprentissage automatique est donné par Mitchel : Un ordinateur programme est dit d'apprendre de l'expérience E avec respect de certaines catégories de tâches T et rendement mesure P, si son exécution aux tâches dans T, comme mesuré par P, s'améliore avec l'expérience E.

Avec la montée du Machine Learning approches que nous avons la capacité de trouver une solution pour ce problème, nous avons développé un système utilisant des données qui a la capacité de prédire si diabétique ou non. De plus, prévoir la maladie mène tôt au traitement des patients avant il devient critique. L'exploration de données a la capacité de extraire la connaissance cachée d'une énorme quantité de données liées au diabète. Pour cette raison, il a un rôle important dans la recherche sur le diabète, maintenant plus que jamais. Le but de cette recherche est de développer un système qui peut prédire le niveau de risque de diabète d'un patient avec une plus grande précision. Cette recherche a mis l'accent sur élaborer un système fondé sur trois classifications méthodes à savoir, Support Vector Machine, Logistique algorithmes de régression et de réseau neuronal artificiel

Outils de développement & environnement de travail

Dans ce projet, nous utiliserons python3 et Jupyter notebook ainsi que pyqt5 . Nous examinerons le projet en important l'ensemble de données, en effectuant une analyse exploratoire des données pour obtenir des renseignements et comprendre à quoi ressemble l'ensemble de données, puis nous établirons le modèle. Nous utiliserons également les arbres de décision, les forêts aléatoires, les machines vectorielles de soutien …

Les 7 étapes de l'apprentissage automatique, fournit le cadre général suivant des étapes dans l'apprentissage automatique:

- ➔Collecte de données
- ➔Préparation des données
- ➔Choisir un modèle
- ➔Formation du modèle
- ➔Évaluation du modèle
- ➔Réglage des paramètres
- ➔Faire des prévisions

Matériaux et méthodes:

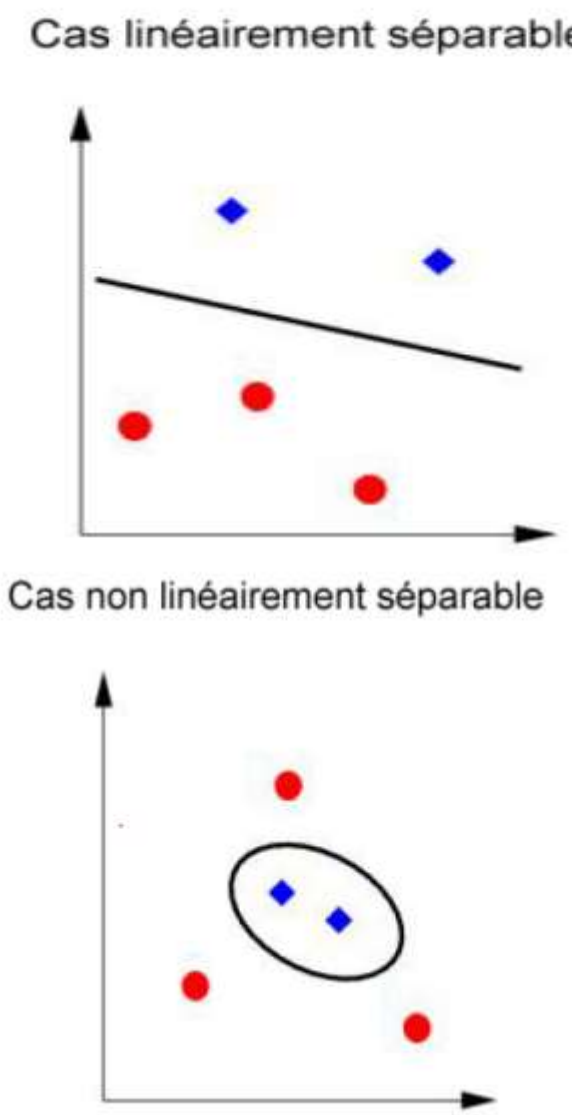
Classificateur d'arbre de décision :

- **Les arbres de décision** sont une méthode d'apprentissage utilisée pour la classification et la régression. L'objectif est de créer un modèle qui prédit la valeur d'une variable cible en apprenant des règles de décision simples déduites des caractéristiques des données. Un arbre peut être vu comme une approximation constante par morceaux.

Un classificateur d'arbre de décision est une approche systématique pour la classification multi classe. Il pose un ensemble de questions à l'ensemble de données. L'algorithme de classification de l'arbre de décision peut être visualisé sur un arbre binaire. Sur la racine et chacun des nœuds internes, une question est posée et les données sur ce nœud sont ensuite divisées en enregistrements distincts qui ont des caractéristiques différentes. Les feuilles de l'arbre font référence aux classes dans lesquelles le jeu de données est divisé. Dans ce TP8, nous entraînons un classificateur

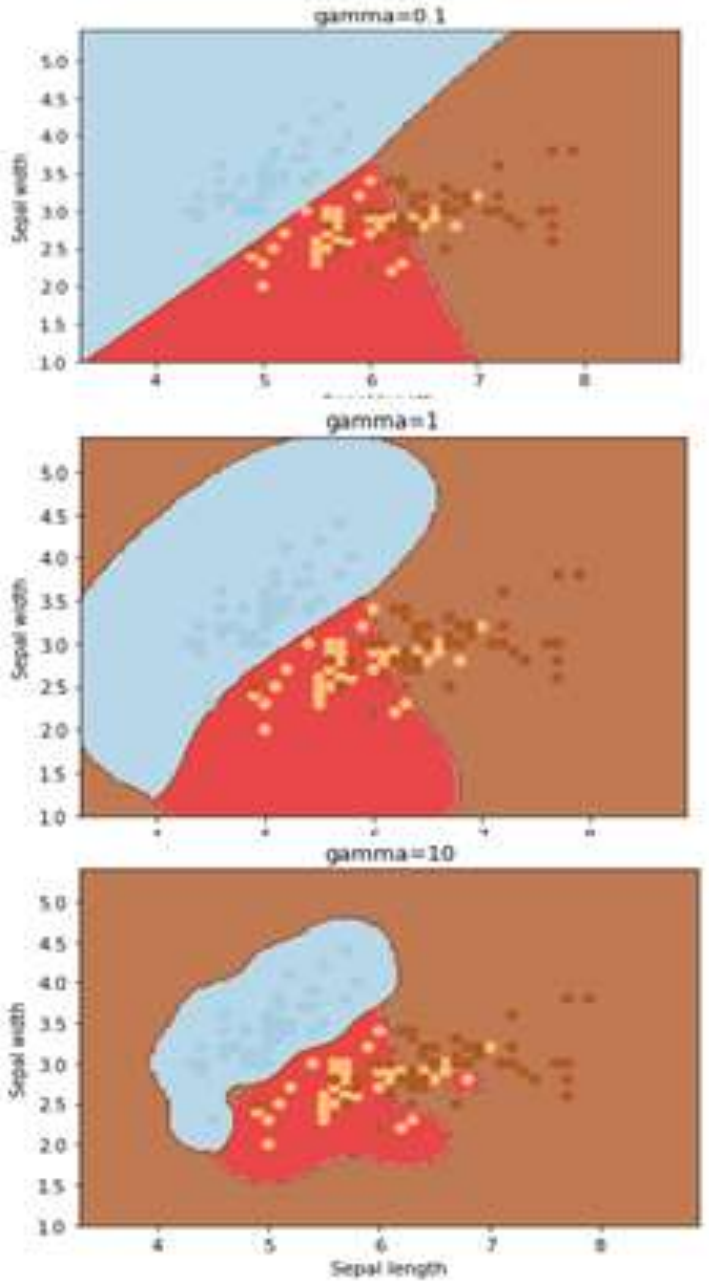
SVM (Support vector machine)

- Les **SVM** sont un ensemble de méthodes d'apprentissage supervisé utilisées pour la classification, la régression et la détection des valeurs aberrantes.



SVM paramètres

- Kernel paramètres sélectionne le type d'hyperplan utilisé pour séparer les données. L'utilisation de 'linéaire' utilisera un hyperplan linéaire (une ligne dans le cas de données 2D). 'Rbf' et 'poly' utilisent un hyper-plan non linéaire.
- gamma est un paramètre pour les hyperplans non linéaires. Plus la valeur gamma qu'il essaie de s'adapter exactement à l'ensemble de données d'entraînement est élevée



L'augmentation du gamma conduit à un sur ajustement lorsque le classificateur essaie de s'adapter parfaitement aux données d'entraînement.

k-neighrest neighbors

- Classificateur KNN. Cet algorithme de classification ne dépend pas de la structure des données. Chaque fois qu'un nouvel exemple est rencontré, ses k voisins les plus proches à partir des données d'entraînement sont examinés. La distance entre deux exemples peut être la distance euclidienne entre leurs vecteurs caractéristiques.

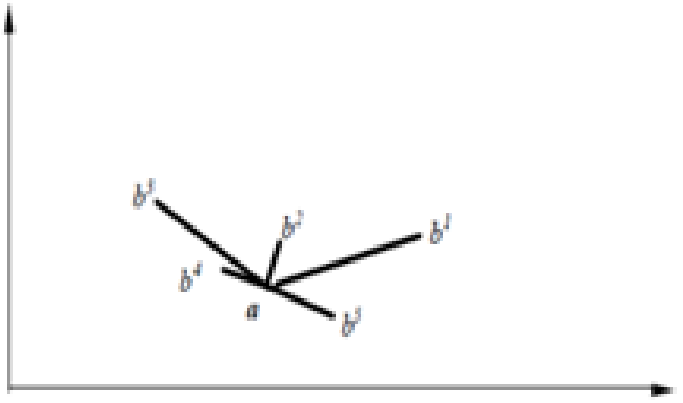


Figure: Méthode des 3 plus proche voisins

Dans l'exemple de la figure, les trois plus proches voisins de a sont b4, b2 et b5, donc a sera affecté à la classe majoritaire parmi ces trois points

Chapitre 2 : Analyse de l'ensemble de données

Discussion:

Description et exploration des données:

- Cet ensemble de données provient de l'Institut national du diabète et des maladies digestives et rénales. Il est fourni avec l'aimable autorisation de la Pima Indians Diabetes Database et est disponible sur Kaggle. Il comprend plusieurs variables prédictives médicales et une variable cible, Résultat. Les variables prédictives comprennent le nombre de grossesses du patient, son IMC, son niveau d'insuline, son âge et ainsi de suite. L'ensemble de données comporte 9 colonnes, comme indiqué ci-dessous:

- `Pregnancies` – Number of times pregnant
- `Glucose` – Plasma glucose concentration a 2 hours in an oral glucose tolerance test
- `BloodPressure` – Diastolic blood pressure (mm Hg)
- `SkinThickness` – Triceps skinfold thickness (mm)
- `Insulin` – 2-Hour serum insulin (mu U/ml)
- `BMI` – Body mass index (weight in kg/(height in m)^2)
- `DiabetesPedigreeFunction` – Diabetes pedigree function
- `Age` – Age (years)
- `Outcome` – Class variable (0 or 1) 268 of 768 are 1, the others are 0

Visualisation et affichage des données:

➔Info sur les données et leur type

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Pregnancies               768 non-null    int64
 1   Glucose                   768 non-null    int64
 2   BloodPressure             768 non-null    int64
 3   SkinThickness             768 non-null    int64
 4   Insulin                   768 non-null    int64
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64
 8   Outcome                   768 non-null    int64
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

➔Les colonnes

```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
      dtype='object')
```

Chapitre 3 : Exploitation des modèles

Divisez l'ensemble de données en données « d'entraînement » et « de test ».

- Une fois que nous avons séparé les fonctionnalités de la cible, nous pouvons créer un train et une classe de test. Comme leur nom l'indique, nous allons entraîner notre modèle sur le train et tester le modèle sur le jeu de test. J'ai sélectionnai au hasard 80% des données à être dans notre formation, et 20% comme test. Mais pour avoir un champs de test plus grand , je l'ai changé par 40% pour

```
Entrée [28]: x_train,x_test,y_train,y_test = train_test_split(x,y,test_size = 300,random_state=0)
             print(x_train.shape,x_test.shape,y_train.shape,y_test.shape)

             (468, 8) (300, 8) (468,) (300,)
```

Entraînement et utilisation des classificateurs.

✓le classificateur SVM :

J ’ai choisit ici d’utiliser un hyper-plan non linéaire et un gamma minimale.

```
svm_model=svm.SVC(kernel='rbf', gamma=0.01,probability=True)
```

✓le classificateur LR :

```
knn=neighbors.KNeighborsClassifier(n_neighbors=9)
```

✓le classificateur RF :

```
from sklearn.ensemble import RandomForestClassifier

# Create a RandomForestClassifier object
rf_model = RandomForestClassifier(random_state=42)

rf_model.fit(x_train, y_train.ravel())
```

✓le classificateur LR :

```
#création d'une instance de la classe
lr = LogisticRegression(solver="liblinear")
```

✓le classificateur RR :

```
ridge_reg = Ridge(alpha=50, max_iter=100, tol =0.1)
ridge_reg.fit(x_train,y_train)
```

✓le classificateur DTC :

```
# Create Decision Tree classifer object
DTC = DecisionTreeClassifier(criterion="entropy", max_depth=3)
```

Validation croisée pour chaque modèle:

pour le DTC

```
#évaluation en validation croisée : 10 cross-validation
succes = model_selection.cross_val_score(DTC,x,y,cv=10,scoring='accuracy')
#détail des itérations
print(succes)
#moyenne des taux de succès = estimation du taux de succès en CV
print(succes.mean())
```

```
[0.72727273 0.72727273 0.72727273 0.67532468 0.71428571 0.79220779
 0.75324675 0.80519481 0.71052632 0.75       ]
0.7382604237867396
```

pour la regression logistique

```
from sklearn import model_selection
#évaluation en validation croisée : 10 cross-validation
succes = model_selection.cross_val_score(lr,x,y,cv=10,scoring='accuracy')
#détail des itérations
print(succes)
#moyenne des taux de succès = estimation du taux de succès en CV
print(succes.mean())
```

```
[0.74025974 0.75324675 0.79220779 0.72727273 0.74025974 0.74025974
 0.81818182 0.79220779 0.73684211 0.82894737]
0.7669685577580314
```

pour le svm

```
#évaluation en validation croisée : 10 cross-validation
succes = model_selection.cross_val_score(svm_model,x,y,cv=10,scoring='accuracy')
#détail des itérations
print(succes)
#moyenne des taux de succès = estimation du taux de succès en CV
print(succes.mean())
```

```
[0.61038961 0.67532468 0.61038961 0.67532468 0.67532468 0.7012987
 0.68831169 0.62337662 0.65789474 0.64473684]
0.6562371838687627
```

pour le knn:

```
#évaluation en validation croisée : 10 cross-validation
succes = model_selection.cross_val_score(knn,x,y,cv=10,scoring='accuracy')
#détail des itérations
print(succes)
#moyenne des taux de succès = estimation du taux de succès en CV
print(succes.mean())
```

```
[0.67532468 0.75324675 0.74025974 0.67532468 0.7012987  0.80519481
 0.72727273 0.77922078 0.77631579 0.75       ]
0.7383458646616542
```

Mesures de performances

✓Performance Metrics : Confusion Matrix, F1 Score, Precision Score, Recall Score

➔**Matrice de confusion:** Il s'agit d'une visualisation tabulaire des prédictions du modèle par rapport aux étiquettes de vérité sur le terrain.

Confusion Matrix		Predicted	
		Negative	Positive
Actual	Negative	True Negative	False Positive
	Positive	False Negative	True Positive

➔**F1 Score :** C'est la moyenne harmonique entre précision et rappel.

$$F_1 = \left(\frac{\text{recall}^{-1} + \text{precision}^{-1}}{2} \right)^{-1} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}.$$

➔**Score de précision:** La précision est la fraction des événements positifs/négatifs prédits qui sont en fait positifs/négatifs.

$$Precision\ (positive\ class) = \frac{TP}{TP+FP} = \frac{True\ Positive}{Number\ of\ cases\ predicted\ as\ positive}$$

$$Precision\ (negative\ class) = \frac{TN}{TN+FN} = \frac{True\ Negative}{Number\ of\ cases\ predicted\ as\ negative}$$

➔**Score de rappel:** C'est la fraction d'événements positifs/négatifs que vous avez prédite correctement.

$$Recall = \frac{True\ Positive}{True\ Positive+False\ Negative}$$

$$= \frac{True\ Positive}{Total\ Actual\ Positive}$$

RL

	precision	recall	f1-score	support
0	0.80	0.92	0.86	201
1	0.76	0.55	0.64	99
accuracy			0.79	300
macro avg	0.78	0.73	0.75	300
weighted avg	0.79	0.79	0.78	300

SVM

	precision	recall	f1-score	support
0	0.69	0.97	0.80	201
1	0.62	0.10	0.17	99
accuracy			0.68	300
macro avg	0.66	0.54	0.49	300
weighted avg	0.67	0.68	0.60	300

DTC

	precision	recall	f1-score	support
0	0.72	0.97	0.83	201
1	0.77	0.24	0.37	99
accuracy			0.73	300
macro avg	0.75	0.60	0.60	300
weighted avg	0.74	0.73	0.67	300

KNN

	precision	recall	f1-score	support
0	0.79	0.89	0.84	201
1	0.70	0.54	0.61	99
accuracy			0.77	300
macro avg	0.75	0.71	0.72	300
weighted avg	0.76	0.77	0.76	300

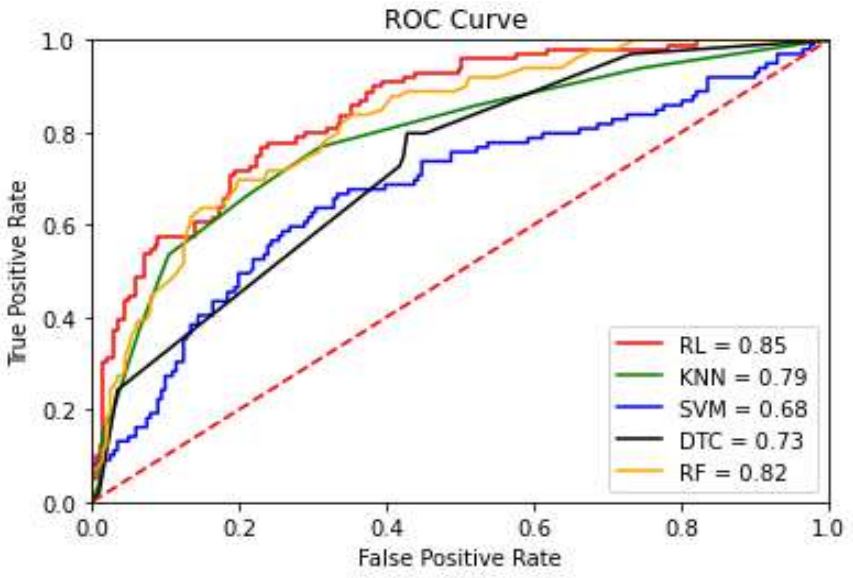
RF

	precision	recall	f1-score	support
0	0.78	0.89	0.83	201
1	0.68	0.49	0.57	99
accuracy			0.76	300
macro avg	0.73	0.69	0.70	300
weighted avg	0.75	0.76	0.75	300

✓Courbe de Roc :

➔Le graphique ROC trace **la sensibilité sur l'axe des y** et **(spécificité 1) sur l'axe des x.** Nous plaçons un point sur le graphique pour chaque valeur seuil du test, traçant la sensibilité et la spécificité du test à cette valeur. La connexion de ces points crée une courbe - la courbe ROC.

➔Le **coin supérieur gauche** de la boîte ROC est le point où sensibilité = 100% et spécificité = 100% (1-spécificité = 0%). Cela représente un test parfait.



➔Nous pouvons quantifier les performances de chaque test en trouvant **l'aire** sous la courbe ROC .Cela nous permet de comparer quantitativement différents .

➔Notez que cette comparaison ne nous oblige pas à déterminer un seuil à l'avance - nous comparons en fait l'ensemble du test, pas seulement un seuil spécifique.

➔**Plus une courbe est proche du coin supérieur gauche, plus la zone en dessous est grande. Ainsi, une plus grande surface sous la courbe ROC implique un meilleur test**.

✓Le meilleur modèle est donc RL (Logistic regression)

Chapitre 4:Deep learning

Keras est l'une des bibliothèques Python les plus puissantes et les plus faciles à utiliser pour les modèles d'apprentissage profond et qui permet l'utilisation des réseaux de neurones de manière simple. Pour définir un modèle de deep learning, on définit les caractéristiques suivantes :

Nombre de couches : ici j'ai utilisé 3 couches

Types des couches : dense

Nombre de neurones dans chaque couche : 9 neurones

Fonctions d'activation de chaque couche :relu et sigmoid

Taille des entrées et des sorties.

Afin d'eviter le test aléatoire ,j'ai fait une boucle me donnant comme résultat le meilleur nombre de couches nécessaires pour avoir la meilleure accuracy.

```
from keras.utils import np_utils

#Defining and desining neural network model
for i in range(1,80):
    model = Sequential()
    model.add(Dense(i, input_dim=8, activation='relu'))
    model.add(Dense(i, activation='relu'))
    model.add(Dense(1, activation='sigmoid'))
    model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
    scores = model.evaluate(x_test,y_test)
    print("\n  %s: %.2f  %%" %  (model.metrics_names[1], scores[1]*100))
    print("i" , i)
```

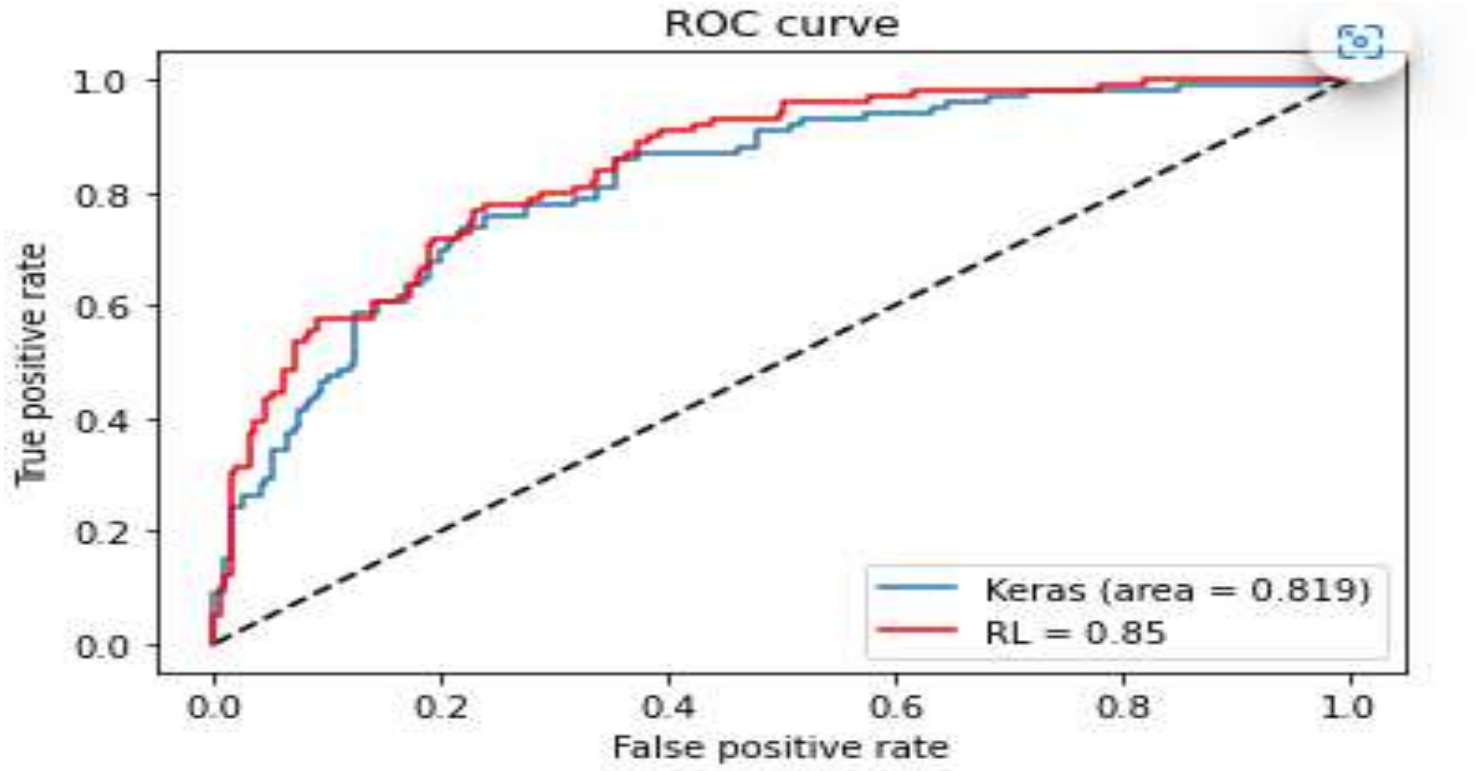
➔*Accuracy: 81,92%*

```
scores = model.evaluate(x_test,y_test)
print("\n%s: %.2f%%" % (model.metrics_names[1], scores[1]*100))
```

```
10/10 [==============================] - 0s 2ms/step - loss: 0.4331 - accuracy: 0.8192

accuracy: 81.92%
```

Machine learning VS Deep learning



➔La comparaison entre les deux modèles en utilisant la courbe de ROC m ’a confirmé que le Deep learning et exactement la RL est la meilleure pour utiliser ce genre d’analyse.

Chapitre5:Interface graphique - Qt5

PyQt est un module libre qui permet de lier le langage Python avec la bibliothèque Qt distribué sous deux licences :
unecommerciale et la GNU GPL.

Il permet ainsi de créer des interfaces graphiques en Python. Une extension de Qt Creator (utilitairegraphique de créationd'interfaces Qt) permet de générer le code Python d'interfacesgraphiques.

Diabetes Prediction

We have token all the related factors in its tests and implementing using machine learning techniques by extracting knowledge from our real health care medical dataset to predict diabetic patients.

Please enter values to predict diabetes!

*Pregnancies:	1	*Insulin:	1
*Glucose:	20	*BMI:	80
*Blood Pressure:	72	*Diabets pedigree:	0.627
*Skin Thickness:	70	*Age:	50

Result | Clear | deep learning | machine learning

Our diagnosis suggests patient does suffer from diabetes.
Please get checked soon.

➔Using Machine learning.

Diabetes Prediction

We have token all the related factors in its tests and implementing using machine learning techniques by extracting knowledge from our real health care medical dataset to predict diabetic patients.

Please enter values to predict diabetes!

*Pregnancies:	1	*Insulin:	1
*Glucose:	20	*BMI:	80
*Blood Pressure:	72	*Diabets pedigree:	0.627
*Skin Thickness:	70	*Age:	50

Result | Clear | deep learning | machine learning

indicates that you are [82.00484] % suspected to be diabetic

➔Using Deep learning.

Conclusion

L'apprentissage automatique a la grande capacité de révolutionner la prédiction du risque de diabète à l'aide de méthodes de calcul avancées et de la disponibilité d'une grande quantité d'ensembles de données épidémiologiques et génétiques sur le risque de diabète. La détection du diabète à ses premiers stades est la clé du traitement. Ce travail a décrit une approche d'apprentissage automatique pour prédire les niveaux de diabète. La technique peut également aider les chercheurs à développer un outil précis et efficace qui atteindra la table des cliniciens pour aider les chercheurs à prendre de meilleures décisions sur l'état de la maladie.

Comment prédire et diagnostiquer exactement cette maladie en utilisant l'apprentissage automatique mérite d'être étudié. Selon les expériences ci-dessus, nous avons constaté que la précision de l'utilisation de PCA n'est pas bonne. Le résultat, qui n'utilisait que du glucose à jeun, a une meilleure performance. Cela signifie que le glucose à jeun est l'indice le plus important pour prédire, mais seule l'utilisation du glucose à jeun ne peut pas obtenir le meilleur résultat, donc si vous voulez prédire avec précision, nous avons besoin de plus d'indices. De plus, en comparant les résultats de trois classifications, nous pouvons constater qu'il n'y a pas beaucoup de différence entre la forêt aléatoire, l'arbre de décision , régression logistique et le réseau neuronal. Le meilleur résultat pour l'ensemble de données est de 0,85, et la meilleure performance est de 0,8021, ce qui peut indiquer que l'apprentissage automatique peut être utilisé pour prédire le diabète, mais il est très important de trouver des attributs appropriés, un classificateur et une méthode d'exploration de données. En raison des données, nous ne pouvons pas prédire le type de diabète, donc à l'avenir, nous visons à prédire le type de diabète et à explorer la proportion de chaque indicateur, ce qui peut améliorer la précision de la prédiction du diabète.