توثيق مشروع (diabetes prediction)

الطالب : عبدالرحمن السليماني cs4/pm

مادة : تنقيب البيانات

اشراف الدكتورة : هبة

اشراف الأستاذة: نورا

فكرة المشروع : تصميم مودل يتوقع ما إذا كان لدى المريض مرض السكري، بناء على قياسات تشخيصية معينة مضمنة في البيانات.

شرح الداتا سيت : تحتوي على متغير واحد (Dependent تابع) وهو عمود (Outcome) ومجموعة متغيرات (مستقلة independent) وهم :

• 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 skin fold thickness (mm)

• Insulin: 2-Hour serum insulin (mu U/ml)

• BMI: Body mass index (weight in kg/(height in m)^2)

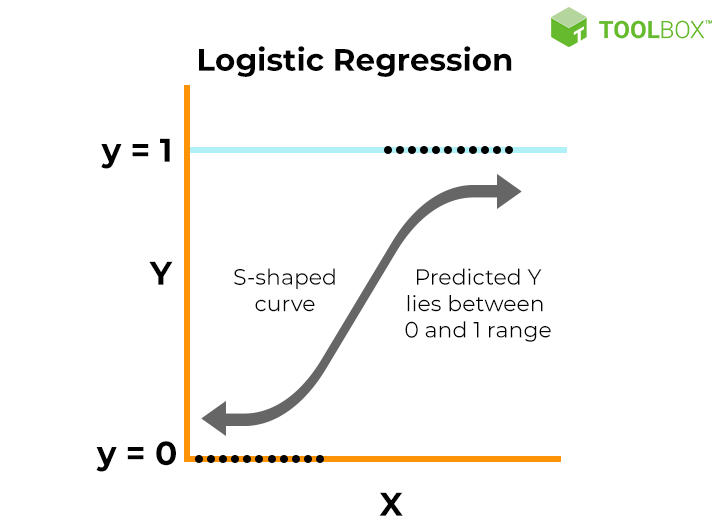
• DiabetesPedigreeFunction: Diabetes pedigree function

• Age: Age (years)

الخوارزميات المستخدمة في المشروع :

استخدام خوارزمية Logistic Regression

الانحدار اللوجستي هو تقنية لتحليل البيانات تستخدم الرياضيات في إيجاد العلاقات بين عامِلَين من عوامل البيانات. ثم يستخدم هذه العلاقة للتنبؤ بقيمة أحد هذين العامِلَين بناءً على الآخر. وعادة ما يكون للتنبؤ عددًا محدودًا (قيمة نهائية) من النتائج، مثل نعم أو لا.



أسباب اختيار الخوازمية:

ان لها تصنيف ثنائي (binary Classification) :

اما ان الشخص مصاب بالسكري او لا

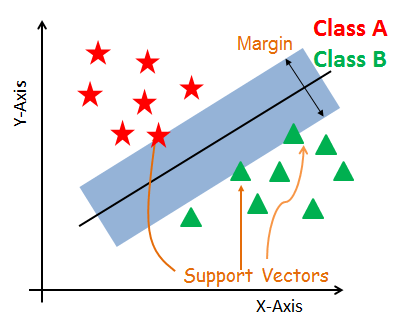
فا Logistic Regression مناسب لهذا النوع من المشاكل

• الكفاءة والبساطة (Efficiency and Simplicity): الانحدار اللوجستي له كفاءة حسابية وبسيط نسبياً مقارنةً بالخوارزميات المعقدة.

استخدام خوارزمية SVC ( Support Vector Classification)

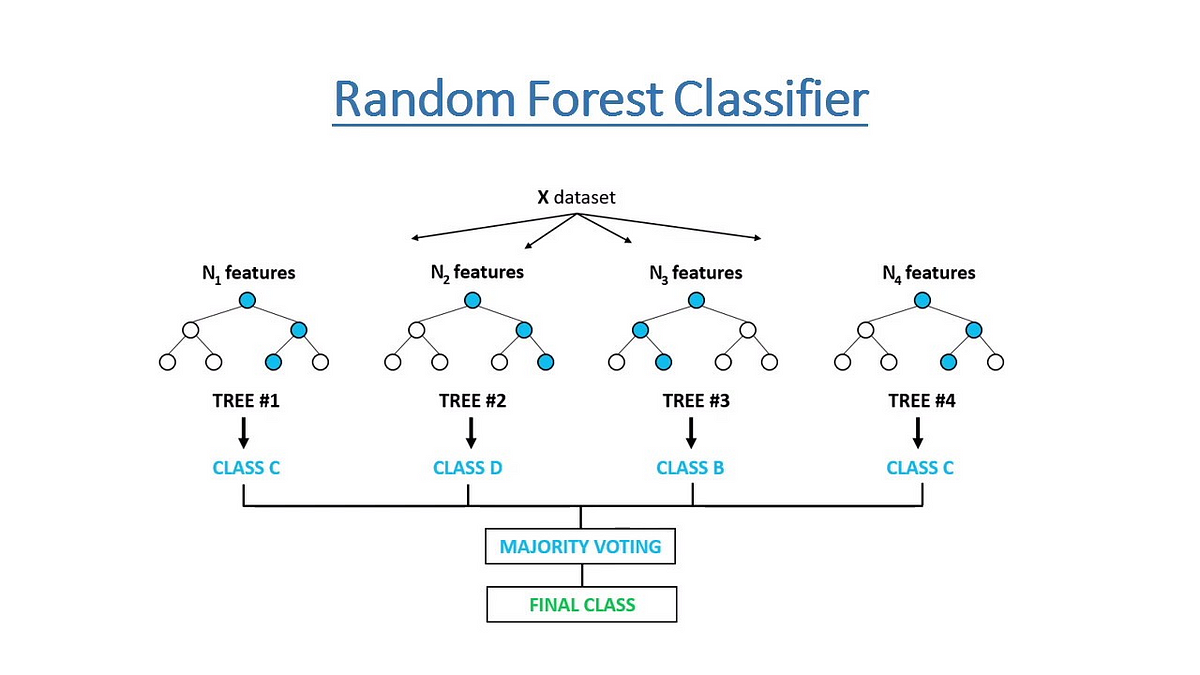
خوارزمية آلة المتجه الداعم SVM هي خوارزمية تعلم آلي خاضع للإشراف (supervised learning) يمكن استخدامها  في مسائل التصنيف Classification أو التنبؤ Regression.

تم استخدام SVC لبناء نموذج يتعلم من بيانات المرضى ويقوم بتصنيف مريض جديد إما كمصاب بالسكري أو غير مصاب.



استخدام خوارزمية random forest classifier

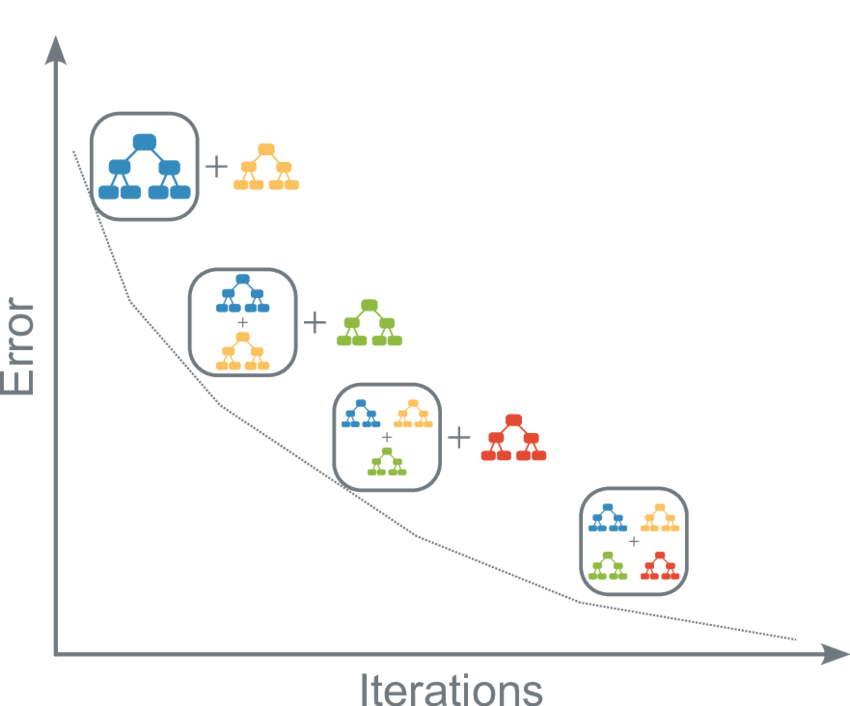
الغابات العشوائية random forest هي خوارزمية تعلم خاضعة للإشراف supervised learning. يمكن استخدامها لحل مشاكل التصنيف والانحدار (او التنبؤ). وهي أيضًا الخوارزمية الأكثر مرونة وسهولة في الاستخدام.



استخدام خوارزميةGradient Boosting Classifier

هو نموذج تعلم آلي يقوم ببناء نماذج فعّالة للتصنيف عبر تكوين سلسلة من النماذج الضعيفة وتعديل الأخطاء بين التوقعات والقيم الفعلية. يستخدم في حل مشاكل التصنيف مع تميزه بالقدرة

على التعامل مع البيانات غير المتوازنة



شرح الاكواد :

import pandas as pd

: pandasوهي مكتبة لمعالجة البيانات وتحليلها.

import numpy as np

: Numpyتستخدم للتعامل مع المصفوفات الكبيرة والحقول متعددة المستوى، وكذلك توفر مكتبة كبيرة من الاقترانات الرياضية عالية المستوى للعمل على هذه الحقول والمصفوفات.

import matplotlib.pyplot as plt

Matplotlib : مكتبة تصميم رسوميات خاصة بلغة البرمجة ببايثون ومكتبتها الرياضية نمباي.

import seaborn as sns

Seaborn :هي مكتبة متخصصة بتحليل البيانات ودراستها وتم استخدامها في المشروع لتحسين النتائج

exploring data :

data=pd.read\_csv("diabetes.csv")

data.head()

عرض أولى البيانات

data.info()

والناتج 768 يمثل عدد البيانات ولا يوجد null

data.describe()

وصف للبيانات مثل العمر min هو 21 و max هو 81 و المتوسط هو 33

data.duplicated.sum()

للتحقق من البيانات المتكررة والناتج false و 0 لذلك لا يوجد تكرار في البيانات

Data analysis and Visualization:

في هذا الجزء نقوم بتحليل البيانات وعلاقتها ببعضها

data.corr()

قياس نسبة العلاقة بين الاعمدة

اعلى رقم 1 (اقوى علاقة)الى 0 (لاتوجد علاقة)

sns.heatmap(data.corr(),annot=true,fmt='0.2f', linewidth=.5)

نسبة العلاقة بين الاعمدة كصورة بيانية

annot=trueيضيف قيم النسبة بالارقام بداخل مربعات الشكل الاحداثي

fmt='0.2f' يقلص الاعداد العشرية لرقمين فقط بعد الفاصلة

linewidth=.5 التباعد بين مربعات الشكل الاحداثي لتظهر مخرجات الشكل الاحداثي بشكل أوضح

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

data.hist(figsize = (20,20))

يقوم بعرض اشكال بيانية احصائية لكامل اعمدة الداتاسيت

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

sns.countplot(x='Outcome',data=data,palette['g','r'])

تحليل لنتائج البيانات اما ان المريض مصاب بالسكري ونرمز له بالرقم(1) او غير مصاب بالمرض (0)

palette['g','r'] عرض لون اعمدة الشكل الاحداثي

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

plt.figure(figsize=(20,6))

plt.subplot(1, 3, 1)

plt.title("Counter Plot")

sns.countplot(x = 'Pregnancies',data = data)

plt.subplot (1, 3, 2)

plt.title('Distribution Plot')

sns.distplot(data["Pregnancies " ])

plt.subplot(1, 3, 3)

plt.title('Box Plot')

sns.boxplot(y=data["Pregnancies " ] )

plt.show()

هذا الكود بشكل عام يمثل رسوم بيانية مختلفة واحصائيات لعمود (عدد مرات

الحمل)

plt.figure(figsize=(20,6)) عرض وارتفاع الشكل البياني

plt.subplot(1, 3, 1)

رسوم بيانية فرعية نحدد فيها عدد الصفوف في الرقم الاول (1)

اما الرقم الثاني يمثل عدد الاعمدة (3)

الرقم الثالث يمثل موقع الرسم البياني (1)

plt.title("Counter Plot")

تحديد نوع الرسم البياني

sns.countplot(x = 'Pregnancies',data = data)

اختيار عمود (Pregnancies) لتحليل بيانات العمود

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

sns.boxplot(x = 'Age',data = data)

تمثيل بياني للاعمار من نوع boxplot

شرح المخرجات للشكل boxplot: النتيجة تظهر ان العمر ما بين 20 الى 65 والنقاط في الشكل تسمى (outlier) اي انها بعيدة عن الوضع الطبيعي للبيانات الموجودة .

في الشكل منطقة زرقاء فيها ثلاثة خطوط سود تسمى (quartile) الربع

اول خط فيها فيها يسمى الربع الاول (first quartile) وهلم جرا لبقية الخطوط في الشكل، وتمثل احصائيا اغلبية الاعمار في البيانات

: create model

x=data.drop('Outcome', axis=1)

y=data['Outcome']

x: هو المتغير المسؤول عن التدرب نقوم بتحديد كامل اعمدة الداتاسيت الخاصة بالمتغيرات ما عدا (drop) عمود (outcome) ومن ثم تحديد العمود كامل(axis=1) لحذفه

y: المتغير المسؤول عن الاختبار ويسمى بال (target)او (label)

وتكون فيه مخرجات المتغيرات للداتاسيت

/////////\_\_\_\\\\\\\\\\\\\\\\\\

rm=RandomOverSampler(random\_state=41)

x\_res,y\_res=rm.fit \_resample(x,y)

يقوم هذا الكود بموازنة البيانات .

يستخدم RandomOverSampler لزيادة عينات الفئة الأقل حجماً.

fit\_resampleيقوم بتدريب النموذج وزيادة العينات بشكل عشوائي

/////////////\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

print('old data set shape{}'.format(Counter(y)))

print('old data set shape{}'.format(Counter(y\_res)))

لمقارنة البيانات القديمة في المتغير (y) بالجديدة في المتغير (y\_res) ونلاحظ ان البيانات اصبحت متزنة في المتغير الجديد.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

x\_train, x\_test, y\_train, y\_test=train\_test\_split (x, y, test\_size=.2,random\_state=41))

تقسيم الداتاسيت الى اربعة اجزاء :

جزئين للاختبار ويساوي (2.) ويساوي بالقيمة المئوية (%20)

جزئين للتدريب ويساوي بالقيمة المئوية (%80)

تم تقسيم هذه النسب لتحقيق توازن بين التدريب والاختبار لضمان تعلم فعّال

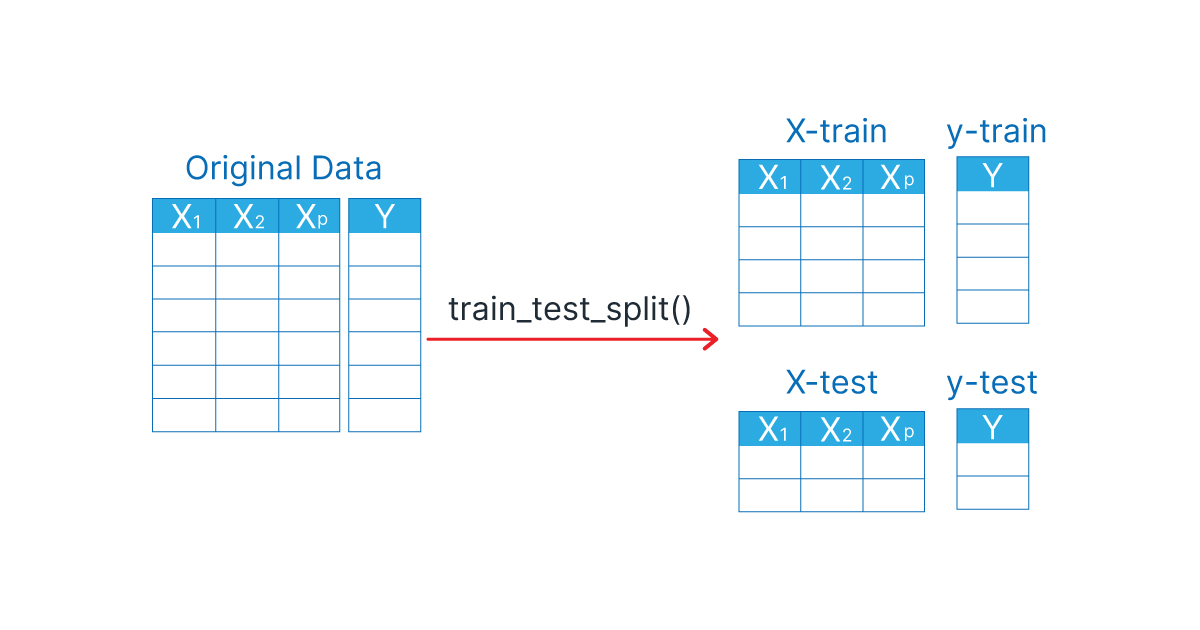
random\_state=41

اخترنا اي رقم عشوائي لضمان أنه في كل مرة يتم فيها تشغيل الكود، سنحصل على نفس تقسيم البيانات لاختبارها والحصول على نفس دقة البيانات في كل مرة.

ملاحظة1: الجزئين هما لاعمدة المتغيرات للداتاسيت والجزء

الاخر لعمود المخرجات (Outcome)

الشكل التوضيحي للعملية :



ملاحظة2:مجموعة التدريب تُستخدم لتعلم المودل، بينما تُستخدم مجموعة الاختبار لتقييم أدائه.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

model1=LogisticRegression()

model2=SVC()

model3=RandomForestClassifier(n \_estimators=100,class\_weight='balanced')

model4=GradientBoostingClassifier (n\_estimators=1000)

استخدام خوارزمية Logistic Regression للتدرب على الموديل

RandomForestClassifier(n \_estimators=100,class\_weight='balanced')

نموذج تصنيف يعتمد على فكرة تجميع عدة أشجار قرارية (Decision Trees) وتم استخدام 100 شجرة قرارية. وتم جعل البيانات متزنة (balanced)

GradientBoostingClassifier (n\_estimators=1000)

هي تقنية تقوم ببناء تسلسل من المودلز الضعيفة وتجميعها للحصول على مودل قوي.

وتم تحديد 1000 من النماذج الضعيفة للحصول على مودل

افضل.

///\_\_\\\\\\\\\\\\_\_\_\_\_///////

columns=[' LogisticRegression', 'SVC', 'RandomForestClassifier', 'GradientBoostingClassifier']

result1=[]

result2=[]

result3=[]

الشرح: انشانا قائمة باسماء الخوارزميات

وعرفنا result لاضافة نتائج مقاييس صحة التوقع

////////

def cal(model):

model.fit(x\_train,y\_train)

pre=model.predict(x\_test)

accuracy=accuracy\_score(pre,y\_test)

recall=recall\_score(pre, y\_test)

f1=f1\_score(pre,y\_test)

result1.append(accuracy)

result2.append(recall)

result3.append(f1)

sns.heatmap(confusion\_matrix (pre,y\_test) , annot=True)

print (model)

print('accuracy is :',accuracy, 'recall is :',recall, 'f1 is :', f1)

cal(model1)

تقوم الدالة بصفة عامة على تقييم أداء المودل

model1.fit(x\_train,y\_train)

التدرب على البيانات

pre=model1.predict(x\_test)

توقع نتيجة المخرجات لكي نقارنها لاحقا

accuracy\_score(pre,y\_test)

recall\_score(pre,y\_test)

f1\_score(pre,y\_test)

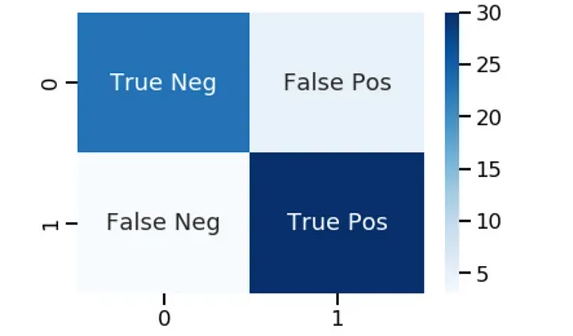
مقاييس لقياس صحة التوقع مع البيانات الحقيقية

result1.append(accuracy)

result2.append(recall)

result3.append(f1)

اضافة نتائج البيانات الى result (انظر للخلية السابقة)

sns.heatmap(confusion\_matrix (pre,y\_test) 

• True Positive (TP): النموذج توقع بشكل صحيح حالات تنتمي إلى الفئة الإيجابية(توقع صحيح للسكري)

• True Negative (TN): النموذج توقع بشكل صحيح حالات تنتمي إلى الفئة السلبية(توقع صحيح بعدم وجود السكري)

• False Positive (FP): النموذج توقع بشكل خاطئ حالات إيجابية عندما تكون فعليًا سلبية

(توقع بشكل خاطئ بوجود السكري)

• False Negative (FN): النموذج توقع بشكل خاطئ

حالات سلبية عندما تكون فعليًا إيجابية (توقع بشكل خاطئ بعدم وجود السكري)

///////////

final\_result=pd.DataFrame({'Algorisms' : col , 'Accuracies' :resulti, 'Recal' :result2, 'F1\_Score' :result3})

عرض مقاييس صحة البيانات على شكل( DataFrame )

//\_\_\_\_\_\_////

fig,ax=plt.subplots(figsize=(15, 5))

plt.plot(final\_result.Algorisms, result1, label='Accuracies')

plt.plot (final\_result.Algorisms, result2, label='Recall ')

plt.plot (final\_result.Algorisms , result3 , label=' Fi\_Score')

plt.legend()

plt.show()

انشاء شكل بياني يحتوي على مقاييس صحة التوقع