


Microbes Classification

دفتر ملاحظات المبتدئين ل Data Sprint 71 - تصنيف الميكروبات. تحديد الفئة التي تنتمي إليها الكائنات الحية الدقيقة

 sayan_saha

📅 2022/4/14 👁 45 مشاهدة

#التعلم الآلي #سباق البيانات #مبتدئ #تصنيف #بايثون العلامات:



تحميل المكتبات

لا يتم تحميل جميع إمكانات Python في بيئة العمل الخاصة بنا بشكل افتراضي (حتى تلك المثبتة بالفعل في نظامك). لذلك ، نقوم باستيراد كل مكتبة نريد استخدامها.

في علم البيانات ، تعد numpy و pandas المكتبات الأكثر استخداما. Numpy مطلوب للحسابات مثل المتوسط ، الوسيط ، الجذور التربيعية ، إلخ. تستخدم الباندا لمعالجة البيانات وإطارات البيانات. نختار أسماء مستعارة لمكتباتنا من أجل راحتنا (pd <-- numpy والباندا np <-- pd).

ملاحظة: يمكنك استيراد جميع المكتبات التي تعتقد أنها ستكون مطلوبة أو يمكنك استيرادها أثناء التنقل.

هنا سنقوم باستيراد المكتبات التالية

```
في [ ] :
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.tree import DecisionTreeClassifier
```

تحميل مجموعة البيانات

يتم استخدام وحدة الباندا لقراءة الملفات.

يمكنك معرفة المزيد عن الباندا هنا

```
في [ ] : df = pd.read_csv(r"/content/train.csv")
```

ماذا عليك أن تفعل الآن؟

- أداء EDA وتصور البيانات ، لفهم البيانات. تعرف على المزيد حول أكاديمية الإمارات الدبلوماسية هنا. تعرف على المزيد حول تصور البيانات هنا
- قم بتنظيف البيانات إذا لزم الأمر (مثل إزالة القيم المفقودة أو تعبئتها ، ومعالجة القيم المتطرفة ، وما إلى ذلك). تعرف على المزيد حول التعامل مع القيم المفقودة هنا
- قم بإجراء المعالجة المسبقة للبيانات إذا كنت تشعر أنها مطلوبة. تعلم ترميز ساخن واحد هنا.

أكاديمية الإمارات الدبلوماسية الأساسية

```
في [ ] : df.head()
```

Major	كونفيكسهول4	...	بوندينجوكس1	رقم أويلر	اتجاه	مدى	منطقة ممتلئة	إكستريما	القطر المتساوي	الانحراف	صلابه	الرقم التسلسلي
0	6.98	...	6.98	22.9	2.15	5.00	0.0726	6.95	2.000	22.5	10.30	13126
1	20.40	...	20.40	22.9	13.00	4.57	0.0207	20.60	v10.7	20.3	7.41	12936
2	9.82	...	9.81	22.6	6.08	6.67	0.2990	9.84	3.810	19.5	12.60	28006
3	5.60	...	4.54	22.5	8.67	3.34	0.1940	4.79	3.090	21.9	5.81	24884
4	15.50	...	15.50	23.0	22.30	5.36	0.0110	15.40	0.751	17.3	7.51	10680

في [] : df.shape

[] خارج : (21368, 26)

في [] : df.describe()

[] خارج :

مدى	منطقة ممتلئة	إكستريما	القطر المتساوي	الانحراف	صلابه	الرقم التسلسلي	عدد
213	21368.000000	21368.000000	21368.000000	21368.000000	21368.000000	21368.000000	21368.000000
5.846141	0.420587	11.829647	3.637855	19.489904	9.700074	15257.325908	دني
3.256171	0.873546	6.045689	2.212756	3.463016	4.058715	8808.215772	الامراض المنقوله جنسيا
0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	دقيقه
3.300000	0.093675	6.720000	2.170000	17.400000	6.590000	7619.500000	25%
5.260000	0.230000	12.000000	3.380000	20.700000	9.360000	15247.500000	50%
7.850000	0.438000	17.100000	4.602500	22.200000	12.600000	22905.250000	75%
23.000000	23.000000	23.000000	23.000000	23.000000	23.000000	30525.000000	ماكس

صفوف 25 × عمودا 8

في [] : df.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21368 entries, 0 to 21367
Data columns (total 26 columns):
#   Column                Non-Null Count  Dtype
---  -
0   Serial No             21368 non-null  int64
1   Solidity              21368 non-null  float64
2   Eccentricity          21368 non-null  float64
3   EquivDiameter         21368 non-null  float64
4   Extrema               21368 non-null  float64
5   FilledArea            21368 non-null  float64
6   Extent                21368 non-null  float64
7   Orientation           21368 non-null  float64
8   EulerNumber           21368 non-null  float64
9   BoundingBox1          21368 non-null  float64
10  BoundingBox2          21368 non-null  float64
11  BoundingBox3          21368 non-null  float64
12  BoundingBox4          21368 non-null  float64
13  ConvexHull1           21368 non-null  float64
14  ConvexHull2           21368 non-null  float64
15  ConvexHull3           21368 non-null  float64
16  ConvexHull4           21368 non-null  float64
17  MajorAxisLength       21368 non-null  float64
18  MinorAxisLength       21368 non-null  float64
19  Perimeter             21368 non-null  float64
20  ConvexArea            21368 non-null  float64
21  Centroid1             21368 non-null  float64
22  Centroid2             21368 non-null  float64
23  Area                  21368 non-null  float64
24  raddi                 21368 non-null  float64
25  microorganisms        21368 non-null  object
dtypes: float64(24), int64(1), object(1)
memory usage: 4.2+ MB
```

فصل ميزات الإدخال وميزات الإخراج

قبل بناء أي نموذج للتعلم الآلي ، نقوم دائما بفصل متغيرات الإدخال ومتغيرات الإخراج. متغيرات الإدخال هي تلك الكميات التي تتغير قيمها بشكل طبيعي في التجربة ، في حين أن متغير الإخراج هو الذي تعتمد قيمه على متغيرات الإدخال. لذلك ، تعرف متغيرات الإدخال أيضا باسم المتغيرات المستقلة لأن قيمها لا تعتمد على أي كمية أخرى ، وتعرف متغيرات / متغيرات الإخراج أيضا بالمتغيرات التابعة لأن قيمها تعتمد على متغير آخر أي متغيرات الإدخال. كما هو الحال هنا في هذه البيانات ، نريد التنبؤ بما إذا كان النيزك يشكل تهديدا للأرض أم لا ، وبالتالي فإن المتغير **الخطر** هو متغيرنا المستهدف والميزات المتبقية هي متغير الإدخال.

حسب الاصطلاح ، يتم تمثيل متغيرات الإدخال ب "X" ويتم تمثيل متغيرات الإخراج ب "y".

]:

في

```
# Input/independent variables
X = df.drop('microorganisms', axis = 1) # here we are dropping the Target feature as this is the target and 'X' is input features, the changes are not made inplace as we have not used 'inplace = True'

y = df['microorganisms'] # Output/Dependent variable
```

تقسيم البيانات إلى مجموعة التدريب والاختبار

نريد التحقق من أداء النموذج الذي قمنا ببنائه. لهذا الغرض ، نقوم دائما بتقسيم (بيانات الإدخال والإخراج) البيانات المعطاة إلى مجموعة تدريب سيتم استخدامها لتدريب النموذج ، ومجموعة اختبار سيتم استخدامها للتحقق من مدى دقة النموذج في التنبؤ بالنتائج.

لهذا الغرض لدينا فئة تسمى "train_test_split" في وحدة "sklearn.model_selection".

]:

في

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3)
```

]:

في

```
X_train.shape
```

]:

خارج

```
(14957, 25)
```

]:

في

```
scaler = StandardScaler().fit(X_train)
X_train = scaler.transform(X_train)
X_test = scaler.transform(X_test)
```

]:

في

```
X_train
```

]:

خارج

```
array([[ 0.32634627,  1.39684129,  0.00589208, ..., -0.27320905,
        -0.65161765, -1.43915238],
       [-0.15345051,  1.05306538, -0.45460585, ...,  1.20098974,
        -0.48929823, -0.3784857 ],
       [ 1.19337128,  0.04629309, -0.88632266, ..., -1.6666779 ,
        -0.4708028 ,  0.14109764],
       ...,
       [ 0.83180004, -1.2330444 ,  0.38004664, ..., -0.49609387,
         0.65213402,  1.06201432],
       [ 1.25372734,  0.9548437 , -1.28925835, ..., -0.80497361,
         0.10519773,  0.55676431],
       [ 1.30131143, -1.00222343,  0.81176345, ...,  0.23574053,
        -0.31315128, -0.69381904]])
```

نموذج البناء

الآن نحن مستعدون أخيرا ، ويمكننا تدريب النموذج.

هناك الكثير من نماذج التعلم الآلي مثل الانحدار اللوجستي ، الغابة العشوائية ، شجرة القرار ، وما إلى ذلك لأقول لك بعضها. ومع ذلك ، فإننا هنا نستخدم RandomForest Classifier (باستخدام مكتبة sklearn).

ثم نقوم بتغذية النموذج بكل من البيانات (X_train) والإجابات لتلك البيانات (y_train)

تدريب النموذج

]:

في

```
from sklearn.linear_model import LogisticRegression
```

]:

في

```
model = LogisticRegression()
```

]:

في

```
model.fit(X_train,y_train)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/linear_model/_logistic.py:818: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG,
```

```
[ ]: LogisticRegression()

[ ]: print(X_train.shape,y_train.shape)

(14957, 25) (14957,)
```

التحقق من صحة النموذج

أتساءل عن مدى جودة تعلم نموذجك! دعونا نتحقق من ذلك.

التنبؤ ببيانات الاختبار (X_test)

الآن نحن نتنبأ باستخدام نموذجنا المدرب على مجموعة الاختبار التي أنشأناها ، أي X_test وتقييم نموذجنا على بيانات غير متوقعة.

```
[ ]: pred = model.predict(X_test)
```

تقييم النموذج

يعد تقييم أداء نموذج التعلم الآلي الذي قمنا ببنائه جزءاً أساسياً من أي مشروع للتعلم الآلي. يتم أداء نموذجنا باستخدام بعض مقاييس التقييم.

هناك الكثير من مقاييس التقييم لاستخدامها في مشكلة الانحدار ، وتسمية بعضها - درجة الدقة ، درجة F1 ، الدقة ، الاستدعاء ، إلخ. ومع ذلك ، فإن **درجة الدقة** هي المقياس لهذا التحدي.

```
[ ]: from sklearn.metrics import accuracy_score
```

التحقق من دقة مجموعة بيانات التحقق من الصحة

طباعة ("درجة الدقة",accuracy_score(X_test,y_test))

```
[ ]: print('Accuracy score',accuracy_score(pred,y_test))
```

Accuracy score 0.9273124317579161

التنبؤ بالمخرجات لاختبار مجموعة 😊 البيانات قمنا بتدريب نموذجنا وتقييمه والآن سنتنبأ أخيراً بالمخرجات / الهدف لبيانات الاختبار (أي testing_set_label.csv) الواردة في قسم "البيانات" في صفحة المشكلة.

مجموعة اختبار التحميل

قم بتحميل بيانات الاختبار التي سيتم تقديم التقديم النهائي عليها.

```
[ ]: test_data = pd.read_csv(r'/content/test.csv')
```

ملاحظه:

- استخدم نفس التقنيات للتعامل مع القيم المفقودة كما هو الحال مع مجموعة بيانات التدريب.
- لا تقم بإزالة أي ملاحظة / سجل من مجموعة بيانات الاختبار وإلا فستحصل على إجابة خاطئة. يجب أن يكون عدد العناصر في التنبؤ الخاص بك هو نفسه عدد السجلات الموجودة في مجموعة بيانات الاختبار.
- استخدم نفس التقنيات لمعالجة البيانات مسبقاً كما هو الحال مع مجموعة بيانات التدريب.

لماذا نحتاج إلى القيام بنفس الإجراء المتمثل في ملء القيم المفقودة وتنظيف البيانات والمعالجة المسبقة للبيانات على بيانات الاختبار الجديدة كما تم القيام به لبيانات التدريب والتحقق من الصحة؟

الجواب: نظرا لأن نموذجنا قد تم تدريبه على تنسيق معين من البيانات وإذا لم نقدم بيانات الاختبار بتنسيق مماثل ، فسيعطي النموذج تنبؤات خاطئة وسيزداد RMSE للنموذج. أيضا ، إذا كان النموذج مبنيا على عدد "n" من الميزات ، أثناء التنبؤ ببيانات الاختبار الجديدة ، فيجب عليك دائما إعطاء نفس العدد من الميزات للنموذج. في هذه الحالة ، إذا قمت بتوفير عدد مختلف من الميزات أثناء التنبؤ بالإخراج ، فسيقوم نموذج ML الخاص بك بإلقاء ValueError يقول شيئا مثل "عدد الميزات المعطاة x : نتوقع ن'. لست واثقا من هذه التصريحات؟ حسنا ، بصفتك عالم بيانات ، يجب عليك دائما إجراء بعض التجارب ومراقبة النتائج.

```
[ ]: test_data.head()
```

[] خارج

بهول4	كونفيكسهول3	...	بوندينجيوكس1	رقم أولير	اتجاه	مدى	منطقة ممثلة	إكستريما	القطر المتساوي	الانحراف	صلابه	الرقم التسلسلي
4.79	...	4.79	22.2	5.44	3.66	0.591	5.14	5.16	20.2	6.70	22885	0
18.20	...	18.20	22.2	17.60	1.91	0.403	21.90	4.74	20.2	3.20	19703	1
3.12	...	3.09	22.6	0.75	10.40	0.389	3.98	4.12	17.4	13.40	27194	2
16.70	...	16.20	21.8	20.50	5.89	0.360	17.30	4.35	17.0	8.74	4687	3
21.50	...	20.80	22.6	4.39	4.04	0.221	20.60	3.34	18.9	7.59	17886	4

صفوف 25 × عمودا 5



إجراء التنبؤ على مجموعة بيانات الاختبار

حان الوقت لتقديم التقديم!!!

```
[ ]: target = model.predict(test_data)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/base.py:444: UserWarning: X has feature name
s, but LogisticRegression was fitted without feature names
  f"X has feature names, but {self.__class__.__name__} was fitted without"
```

ملاحظة: اتبع إرشادات التقديم الواردة في قسم "كيفية الإرسال".

كيفية حفظ نتائج prediciton محليا عبر دفتر ملاحظات jupyter؟

إذا كنت تعمل على دفتر ملاحظات Jupyter ، فقم بتنفيذ مجموعة الرموز أدناه. سيتم إنشاء ملف باسم "prediction_results.csv" في دليل العمل الحالي.

```
[ ]: #target = pd.read_csv(r'test_ans.csv')
res = pd.DataFrame(target) #target is nothing but the final predictions of your model on in
put features of your new unseen test data
res.columns = ["prediction"]
res.to_csv("submission.csv", index = False) # the csv file will be saved locally on the sam
e location where this notebook is located.
```

,OR

إذا كنت تعمل على Google Colab ، فاستخدم مجموعة التعليمات البرمجية أدناه لحفظ نتائج التنبؤ محليا

كيفية حفظ نتائج التنبؤ محليا عبر دفتر ملاحظات colab؟

إذا كنت تعمل على دفتر ملاحظات Google Colab ، فقم بتنفيذ مجموعة الرموز أدناه. سيتم تنزيل ملف باسم "prediction_results" في نظامك.

```
[ ]: # To create Dataframe of predicted value with particular respective index
#target = pd.read_csv(r'/content/test_ans.csv')
res = pd.DataFrame(target) # target are nothing but the final predictions of your model on
input features of your new unseen test data
res.columns = ["prediction"]

# To download the csv file locally
from google.colab import files
res.to_csv('submission.csv', index = False)
files.download('submission.csv')
```

أحسننت! 👍

أنت مستعد تماما لتقديم طلب. دعنا نتوجه إلى صفحة التحدي لتقديم التقديم.

: [] في

[الإبلاغ عن دفتر الملاحظات هذا](#)



أكثر	الشروع	تعلم
استضافة تحديات الذكاء الاصطناعي والتعلم الآلي	علم البيانات	دورات علوم البيانات الذكاء الاصطناعي
المدونة:	التعلم الآلي	ممارسة علوم البيانات والتحديات الذكاء الاصطناعي
قصص المتعلمين	التعلم العميق	منتدى المناقشة
	تصور البيانات 101	علوم البيانات والجلسات المباشرة الذكاء الاصطناعي
	معالجة اللغات الطبيعية 101	دفاتر ملاحظات المجتمع (مشاركة التعليمات البرمجية)
	السلسلة الزمنية 101	

منظمة

من نحن

اتصل بنا

وسائل التواصل الاجتماعي

