القسم العاشر: مكتبة SKlearn

A. Data Preparation

1. Data files from SKlearn
2. Data cleaning
3. Metrics module
4. Feature selection
5. Data Scaling
6. Data Split

B. ML Algorithms

1. Linear Regression
2. Logistic Regression
3. Neural Network
4. SVR
5. SVC
6. K-means
7. PCA
8. Decision Tree
9. Ensemble Regression
10. Ensemble Classifier
11. K Nearest Neighbors
12. Naïve Bayes
13. LDA , QDA
14. Hierarchical Clusters
15. DbScan
16. NLP
17. Apriori

C. Algorithm Evaluation :

1. Model Check
2. Grid Search
3. Pipeline
4. Model Save

D. Time Series

1.3) Metrics Module

و هي بالغة الأهمية في عمل حساب لكمية الأخطاء اثناء الاختبار , وتنقسم إلي نوعين , نوع للتوقع , ونوع للتصنيف :

أدوات التوقع :

1.3.1 metrics.mean_absolute_error
1.3.2 metrics.mean_squared_error
1.3.3 metrics.median_absolute_error

أدوات التصنيف :

1.3.4 metrics.confusion_matrix
1.3.5 metrics.accuracy_score
1.3.6 metrics.f1_score
1.3.7 metrics.recall_score
1.3.8 metrics.precision_score
1.3.9 metrics.precision_recall_fscore_support
1.3.10 metrics.precision_recall_curve
1.3.11 metrics.classification_report
1.3.12 metrics.roc_curve
1.3.13 metrics.auc
1.3.14 metrics.roc_auc_score
1.3.15 metrics.zero_one_loss

1.3.1) Mean Absolute Error

متوسط الخطأ القياسي mean absolute error

$$\frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

الصيغة :

```
# Import Libraries
from sklearn.metrics import mean_absolute_error
#-------------------------------------------------

#Calculating Mean Absolute Error
MAEValue = mean_absolute_error(y_test, y_pred, multioutput='uniform_average') # it can be raw_values
#print('Mean Absolute Error Value is : ', MAEValue)
```

مثال :

```
from sklearn.metrics import mean_absolute_error
y_true = [3, -0.5, 2, 7]
y_pred = [2.5, 0.0, 2, 8]
```

هنا يحسب القيمة بالشكل الطبيعي

```
mean_absolute_error(y_true, y_pred)
```

واذا كانت القيم هي مصفوفات

```
y_true = [[0.5, 1], [-1, 1], [7, -6]]
y_pred = [[0, 2], [-1, 2], [8, -5]]
```

هنا ياتي بالمتوسط لهم كلهم

```
mean_absolute_error(y_true, y_pred) # 0.75
mean_absolute_error(y_true, y_pred, multioutput='uniform_average') # 0.75
```

و هنا لكل صف علي حدة

```
mean_absolute_error(y_true, y_pred, multioutput='raw_values') # array([0.5, 1. ])
```

1.3.2) Mean Squared Error

متوسط مربع الخطأ Mean Squared Error

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

الصيغة

```
#Import Libraries
from sklearn.metrics import mean_squared_error
#------------------------------------------------

#Calculating Mean Squared Error
MSEValue = mean_squared_error(y_test, y_pred, multioutput='uniform_average') # it can be raw_values
#print('Mean Squared Error Value is : ', MSEValue)
```

مثال

```
from sklearn.metrics import mean_squared_error
y_true = [3, -0.5, 2, 7]
y_pred = [2.5, 0.0, 2, 8]
mean_squared_error(y_true, y_pred)

y_true = [[0.5, 1],[-1, 1],[7, -6]]
y_pred = [[0, 2],[-1, 2],[8, -5]]
```

هنا ياتي بالمتوسط لهم كلهم

```
mean_squared_error(y_true, y_pred)
mean_squared_error(y_true, y_pred, multioutput='uniform_average')
```

و هنا لكل صف علي حدة

```
mean_squared_error(y_true, y_pred, multioutput='raw_values')
```

1.3.3) Media Absolute Error

نفس فكرة mean لكنه يقوم برص قيم الاخطاء و اختيار القيمة الـmedian

الصيغة :

```
#Import Libraries
from sklearn.metrics import median_absolute_error
#-----------------------------------------------
#Calculating Median Squared Error
MdSEValue = median_absolute_error(y_test, y_pred)
#print('Median Squared Error Value is : ', MdSEValue )
```

مثال

```
from sklearn.metrics import median_absolute_error
y_true = [3, -0.5, 2, 7]
y_pred = [2.5, 0.0, 2, 8]
median_absolute_error(y_true, y_pred)
```

1.3.4) Confusion Matrix

TP	FP
FN	TN

مصفوفة التشتت

```
#Import Libraries
from sklearn.metrics import confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt
#-----------------------------------------------
#Calculating Confusion Matrix
CM = confusion_matrix(y_test, y_pred)
#print('Confusion Matrix is : \n', CM)

# drawing confusion matrix
sns.heatmap(CM, center = True)
plt.show()
```

مثال

```
from sklearn.metrics import confusion_matrix
y_pred = ['a','a','b','b','a','b','a','a','a','a']
y_true  = ['a','b','b','a','b','a','a','b','a','b']
confusion_matrix(y_true, y_pred)
```

-	pred a	pred b
actual a	3	3
actual b	4	1

```
from sklearn.metrics import confusion_matrix
y_pred = ['a','b','c','a','b','c','a','b','c','a']
y_true =  ['a','a','b','b','a','b','c','c','b','b']
confusion_matrix(y_true, y_pred)
```

-	pred a	pred b	pred c
actual a	1	2	0
actual b	2	0	3

actual c	1	1	0

```
from sklearn.metrics import confusion_matrix
y_pred = [5,8,9,9,8,5,5,9,8,5,9,8]
y_true =  [9,9,8,8,5,5,9,5,8,9,8,5]
confusion_matrix(y_true, y_pred)
```

-	pred 5	pred 8	pred 9
actual 5	1	2	1
actual 8	0	1	3
actual 9	3	1	0

1.3.5) Accuracy Score

وهي تساوي

((TP + TN) / float(TP + TN + FP + FN))

الصيغة

```
#Import Libraries
from sklearn.metrics import accuracy_score
#-----------------------------------------------

#Calculating Accuracy Score  : ((TP + TN) / float(TP + TN + FP + FN))
AccScore = accuracy_score(y_test, y_pred, normalize=False)
#print('Accuracy Score is : ', AccScore)
```

مثال

```
from sklearn.metrics import accuracy_score
y_pred = [0, 2, 1, 3,5,3]
y_true = [0, 1, 2, 3,5,3]
print(accuracy_score(y_true, y_pred)) # fraction of all Trues over everything
print(accuracy_score(y_true, y_pred, normalize=False)) #number of all Trues
```

1.3.6) F1 Score

وهي تساوي

الصيغة

F1 = 2 * (precision * recall) / (precision + recall)

```
#Import Libraries
from sklearn.metrics import f1_score
#-----------------------------------------------

#Calculating F1 Score  : 2 * (precision * recall) / (precision + recall)
# f1_score(y_true, y_pred, labels=None, pos_label=1, average='binary', sample_weight=None)

F1Score = f1_score(y_test, y_pred, average='micro') #it can be : binary,macro,weighted,samples
#print('F1 Score is : ', F1Score)
```

مثال

```
from sklearn.metrics import f1_score
y_pred = [0, 2, 1, 0, 0, 1]
y_true = [0, 1, 2, 0, 1, 2]
f1_score(y_true, y_pred, average='micro')
```

1.3.7) Recall Score (Sensitivity)

البسط قيمة TP و المقام TP + FN وهي 2 وهي المرتين التي كانت a في الحقيقية ولم تكن في المتوقعة

```
# (TP / float(TP + FN))   1 / 1+2
```

الصيغة

```
#Import Libraries
from sklearn.metrics import recall_score
#-----------------------------------------------------
#Calculating Recall Score : (Sensitivity) (TP / float(TP + FN))   1 / 1+2
# recall_score(y_true, y_pred, labels=None, pos_label=1, average='binary', sample_weight=None)

RecallScore = recall_score(y_test, y_pred, average='micro') #it can be : binary,macro,weighted,samples
#print('Recall Score is : ', RecallScore)
```

مثال

```
from sklearn.metrics import recall_score
y_pred =  ['a','b','c','a','b','c','a','b','c','a']
y_true =   ['a','a','b','b','a','b','c','c','b','b']
recall_score(y_true, y_pred, average=None)
```

1.3.8) Precision Score (Specificity)

البسط قيمة TP و المقام TP + FN وهي 3 وهي الثلاث مرات التي كانت a في المتوقعة ولم تكن في الحقيقية

from

الصيغة

```
#(TP / float(TP + FP))

#Import Libraries
from sklearn.metrics import precision_score
#-----------------------------------------------

#Calculating Precision Score : (Specificity) #(TP / float(TP + FP))
# precision_score(y_true, y_pred, labels=None, pos_label=1, average='binary',sample_weight=None)

PrecisionScore = precision_score(y_test, y_pred, average='micro') #it can be : binary,macro,weighted,samples
#print('Precision Score is : ', PrecisionScore)
```

مثال

```
from sklearn.metrics import precision_score

y_pred =  ['a','b','c','a','b','c','a','b','c','a']
y_true =   ['a','a','b','b','a','b','c','c','b','b']

precision_score(y_true, y_pred, average=None)
```

1.3.9) Precision Recall Fscore Support

الصيغة

```
#Import Libraries
from sklearn.metrics import precision_recall_fscore_support
#------------------------------------------------

#Calculating Precision recall Score :
#metrics.precision_recall_fscore_support(y_true, y_pred, beta=1.0, labels=None, pos_label=1, average=
#                                        None, warn_for = ('precision','recall', 'f-score'), sample_weight=None)

PrecisionRecallScore = precision_recall_fscore_support(y_test, y_pred, average='micro') #it can be :
binary,macro,weighted,samples
#print('Precision Recall Score is : ', PrecisionRecallScore)
```

مثال

```
import numpy as np
from sklearn.metrics import precision_recall_fscore_support
y_pred = np.array(['cat', 'pig', 'dog', 'cat', 'cat', 'dog'])
y_true =  np.array(['cat', 'dog', 'pig', 'cat', 'dog', 'pig'])

precision_recall_fscore_support(y_true, y_pred, average=None)
```

1.3.10) Precision Recall Curve

لحساب قيم threshold ,recall , precision معا , لكن فقط للقيم البيناري

from

الصيغة

```
#Import Libraries
from sklearn.metrics import precision_recall_curve
#-----------------------------------------------

#Calculating Precision recall Curve :
# precision_recall_curve(y_true, probas_pred, pos_label=None, sample_weight=None)

PrecisionValue, RecallValue, ThresholdsValue = precision_recall_curve(y_test,y_pred)
#print('Precision Value is : ', PrecisionValue)
#print('Recall Value is : ', RecallValue)
#print('Thresholds Value is : ', ThresholdsValue)
```

مثال

```
import numpy as np
from sklearn.metrics import precision_recall_curve
y_pred =  np.array([0, 0, 1, 1])
y_true =   np.array([0.1, 0.4, 0.35, 0.8])

precision, recall, thresholds = precision_recall_curve(y_pred,y_true)

print(precision)
print(recall)
print(thresholds)
```

1.3.11) Classification Report

وهي تقوم بحساب كلا من : precision , recall , f1score , support لكل قيمة من القيم , سواء ارقام او نصوص , كما تقوم بإظهار المتوسطات بأنواعها macro , micro , support

الصيغة

```
#Import Libraries
from sklearn.metrics import classification_report
#-----------------------------------------------

#Calculating classification Report :
#classification_report(y_true, y_pred, labels=None, target_names=None,sample_weight=None, digits=2,
output_dict=False)

ClassificationReport = classification_report(y_test,y_pred)
#print('Classification Report is : ', ClassificationReport )
```

مثال

```
from sklearn.metrics import classification_report
y_true = [0, 1, 2, 2, 2,5]
y_pred = [0, 0, 2, 2, 1,0]
print(classification_report(y_true, y_pred ))
```

```
from sklearn.metrics import classification_report
y_true = ['a','d','a','g','a','d']
y_pred = ['a','a','g','g','d','g']
print(classification_report(y_true, y_pred ))
```

1.3.12) ROC Curve

اداة الـ ROC و هي اختصار Receiver operating characteristic , و هي فقط تستخدم مع التصنيف الثنائي binary classification

هي اداة لتحديد القيمة المناسبة للـ sensitivity & specificity , واختيار الـ threshold المناسبة , مع ملاحظة اننا نعطيها y_pred_prob فهي تاخذ احتمالية و ليست قيم متوقعة

تعطي 3 قيم , fpr و هي false positive rate و تساوي 1- specificity و tpr و هي true positive rate و تساوي sensitivity و قيمة الثريشهولد المناسبة

الصيغة

```
#Import Libraries
from sklearn.metrics import roc_curve
#-----------------------------------------------

#Calculating Receiver Operating Characteristic :
#roc_curve(y_true, y_score, pos_label=None, sample_weight=None,drop_intermediate=True)
```

```
fprValue, tprValue, thresholdsValue = roc_curve(y_test,y_pred)
#print('fpr Value  : ', fprValue)
#print('tpr Value  : ', tprValue)
#print('thresholds Value  : ', thresholdsValue)
```

هذا المثال

```
import numpy as np
from sklearn import metrics
y =      np.array([1    , 1     , 2     , 2])
scores = np.array([0.1  , 0.4   ,   0.35, 0.8])
fpr, tpr, thresholds = metrics.roc_curve(y, scores, pos_label=2)
```

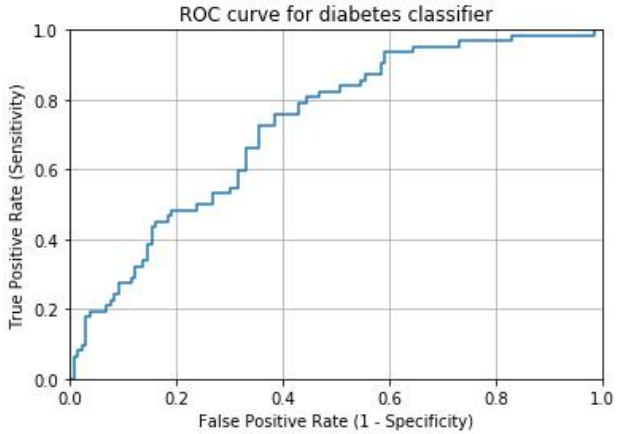
سيكون له القيم :

```
fpr          :       array([0. , 0. , 0.5, 0.5, 1. ])
tpr          :       Out[3]: array([0. , 0.5, 0.5, 1. , 1. ])
thresholds   :       Out[4]: array([1.8 , 0.8 , 0.4 , 0.35, 0.1 ])
```

أي أنه حينما كانت الثريشهولد تساوي 1.8 , كانت tpr وهي sensitivity تساوي 0 , بينما كانت fpr تساوي 0 اي ان ال specificity تساوي 1
كذلك حينما كانت الثريشهولد تساوي 0.35 , كانت tpr وهي sensitivity تساوي 1 , بينما كانت fpr تساوي 0.5 اي ان ال specificity تساوي 0.5

و يمكن رسمها عبر هذا الكود هكذا :

```
fpr, tpr, thresholds = metrics.roc_curve(y_test, y_pred_prob)
plt.plot(fpr, tpr)
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.0])
plt.title('ROC curve for diabetes classifier')
plt.xlabel('False Positive Rate (1 - Specificity)')
plt.ylabel('True Positive Rate (Sensitivity)')
plt.grid(True)
```



فالخط الازرق يظهر العلاقة بين القيمتين , لاحظ ان محور اكس هو 1 ناقص الـ specificity , فالنقطة اعلي يمين المنحي , هي بقيمة 0.9 علي sensitivity لكن ب 0.1 علي specificity

ويمكن عبر استخدام الدالة :

```
def evaluate_threshold(threshold):
    print('Sensitivity:', tpr[thresholds > threshold][-1])
    print('Specificity:', 1 - fpr[thresholds > threshold][-1])
```

الدالة تعطيها قيمة مقترحة للعتبة , تقوم هي بحساب كلا من sensitivity & specificity

1.3.13) AUC

أداة AUC و هي اختصار area under curve

و هي التي تقوم بحساب المساحة تحت المنحني السابق شرحه , ولاحظ ان كلما زادت المساحة تحت المنحني كلما دل هذا علي دقة الخوارزم , وذلك لانه يتيح قيم عالية الـ sensitivity & specificity معا

الصيغة

```
#Import Libraries
from sklearn.metrics import roc_curve
from sklearn.metrics import auc
#----------------------------------------------------

#Calculating Area Under the Curve :

fprValue2, tprValue2, thresholdsValue2 = roc_curve(y_test,y_pred)
AUCValue = auc(fprValue2, tprValue2)
#print('AUC Value  : ', AUCValue)
```

مثال

```
import numpy as np
from sklearn import metrics
y =      np.array([1    , 1     , 2     , 2])
scores = np.array([0.1  , 0.4   ,   0.35, 0.8])
fpr, tpr, thresholds = metrics.roc_curve(y, scores, pos_label=2)

metrics.auc(fpr, tpr)
```

1.3.14) ROC AUC Score

و هذا الأمر يجمع الأمرين السابقين معا , اذ اننا نقوم بحساب auc مباشرة من القيم دون تطبيق roc اولا

الصيغة

```
#Import Libraries
from sklearn.metrics import roc_auc_score
#-----------------------------------------------

#Calculating ROC AUC Score:
#roc_auc_score(y_true, y_score, average='macro', sample_weight=None,max_fpr=None)

ROCAUCScore = roc_auc_score(y_test,y_pred, average='micro') #it can be : macro,weighted,samples
#print('ROCAUC Score : ', ROCAUCScore)
```

مثال

```
import numpy as np
from sklearn import metrics
y =      np.array([1    , 1     , 2     , 2])
scores = np.array([0.1  , 0.4   ,   0.35, 0.8])
metrics.roc_auc_score(y, scores)
```

1.3.15) Zero One Loss

و هي تقوم بحساب عدد مرات اللا تطابق . .

الصيغة

```
#Import Libraries
from sklearn.metrics import zero_one_loss
#-----------------------------------------------

#Calculating Zero One Loss:
#zero_one_loss(y_true, y_pred, normalize = True, sample_weight = None)

ZeroOneLossValue = zero_one_loss(y_test,y_pred,normalize=False)
#print('Zero One Loss Value : ', ZeroOneLossValue )
```

مثال

```
from sklearn.metrics import zero_one_loss
y_pred = [1, 2, 3, 4]
y_true = [2, 2, 3, 4]
```

هنا تكون نسبة كم مرة لا تطابق علي العدد الكلي (0.25)

```
zero_one_loss(y_true, y_pred)
```

و هنا يأتي بالرقم نفسه , اي كم مرة لا تطابق

```
zero_one_loss(y_true, y_pred, normalize=False)
```