تحليل البيانات الضخمة
Big data analytics

Dr Ziad Abdallah

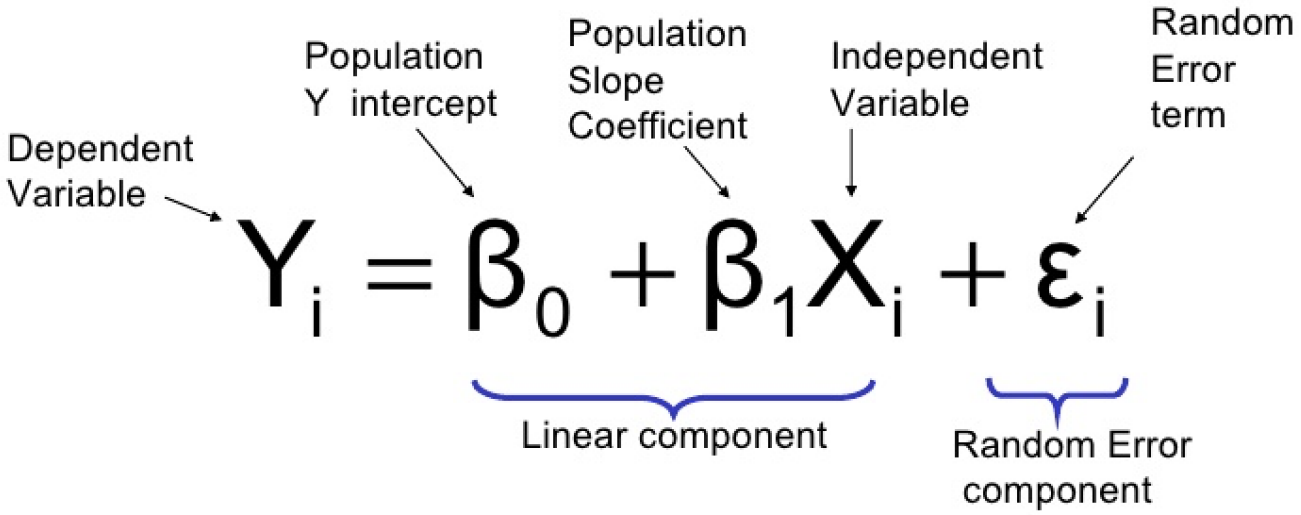
تحليل البيانات الكبيرة في الإحصاءات الرسمية

يتمحور العرض حول الانحدار، التجميع، والتصنيف.

العرض التقديمي يستعرض كيف يمكن استخدام تحليل البيانات الكبيرة في مجال الإحصاءات الرسمية.

الانحدار (Regression)

- يُعتبر تحليل الانحدار مجموعة من الأساليب الإحصائية المستخدمة لتقدير العلاقات بين **متغير المستقل وواحد** أو أكثر من المتغيرات .
- يُستخدم الانحدار الخطي للعثور على العلاقة بين المتغيرات.
- في التعلم الآلي والنمذجة الإحصائية



https://cdn-images-1.medium.com/max/1436/1*_TqRJ9SmwFzRigJhMiN2uw.png

$$\beta_1 = \frac{\sum (X_i - \bar{X})(Y_i - \bar{Y})}{\sum (X_i - \bar{X})^2}$$

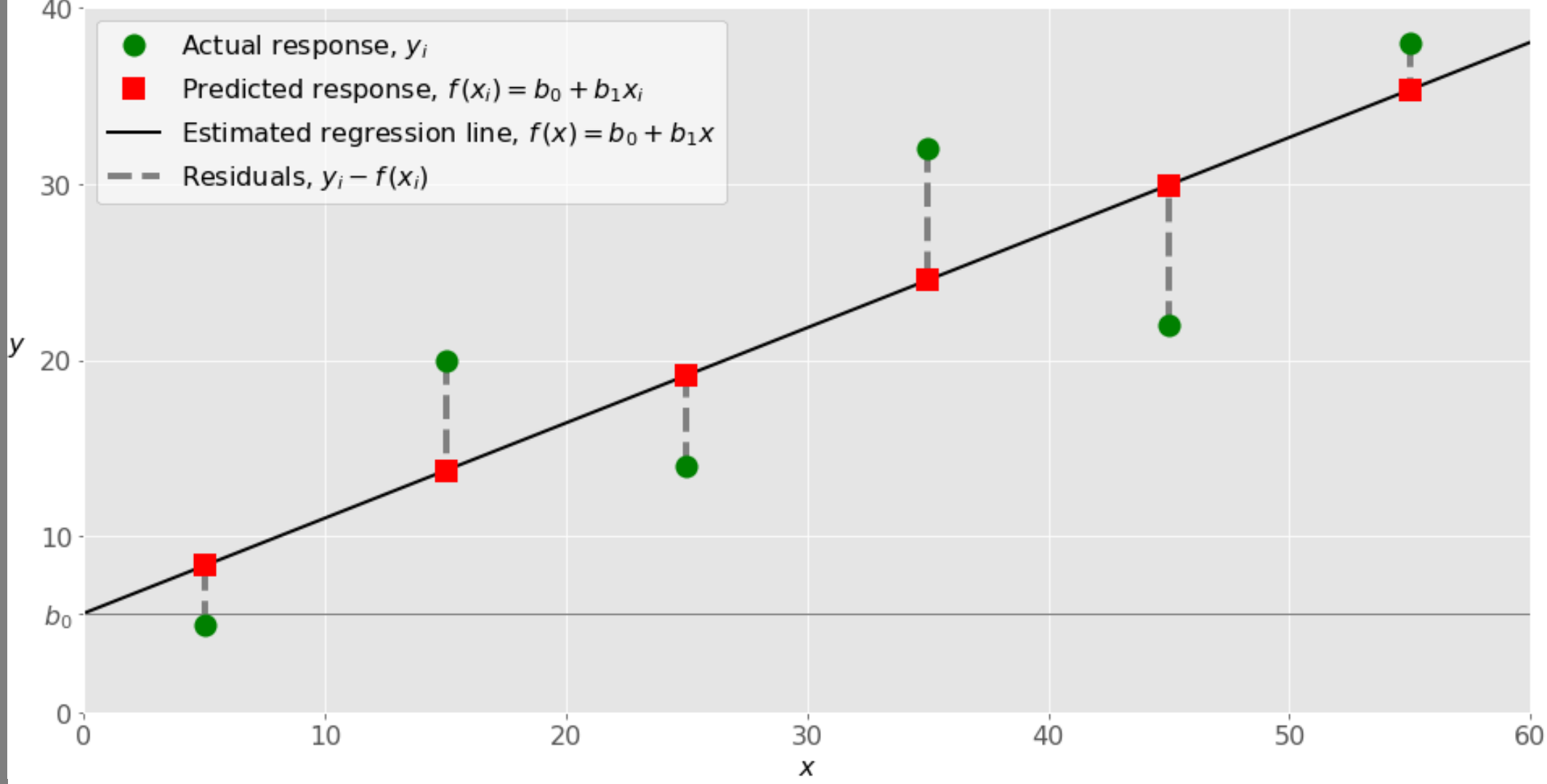
X_i = The feature X

$\bar{X}$ = Mean of X

Y_i = The target Y

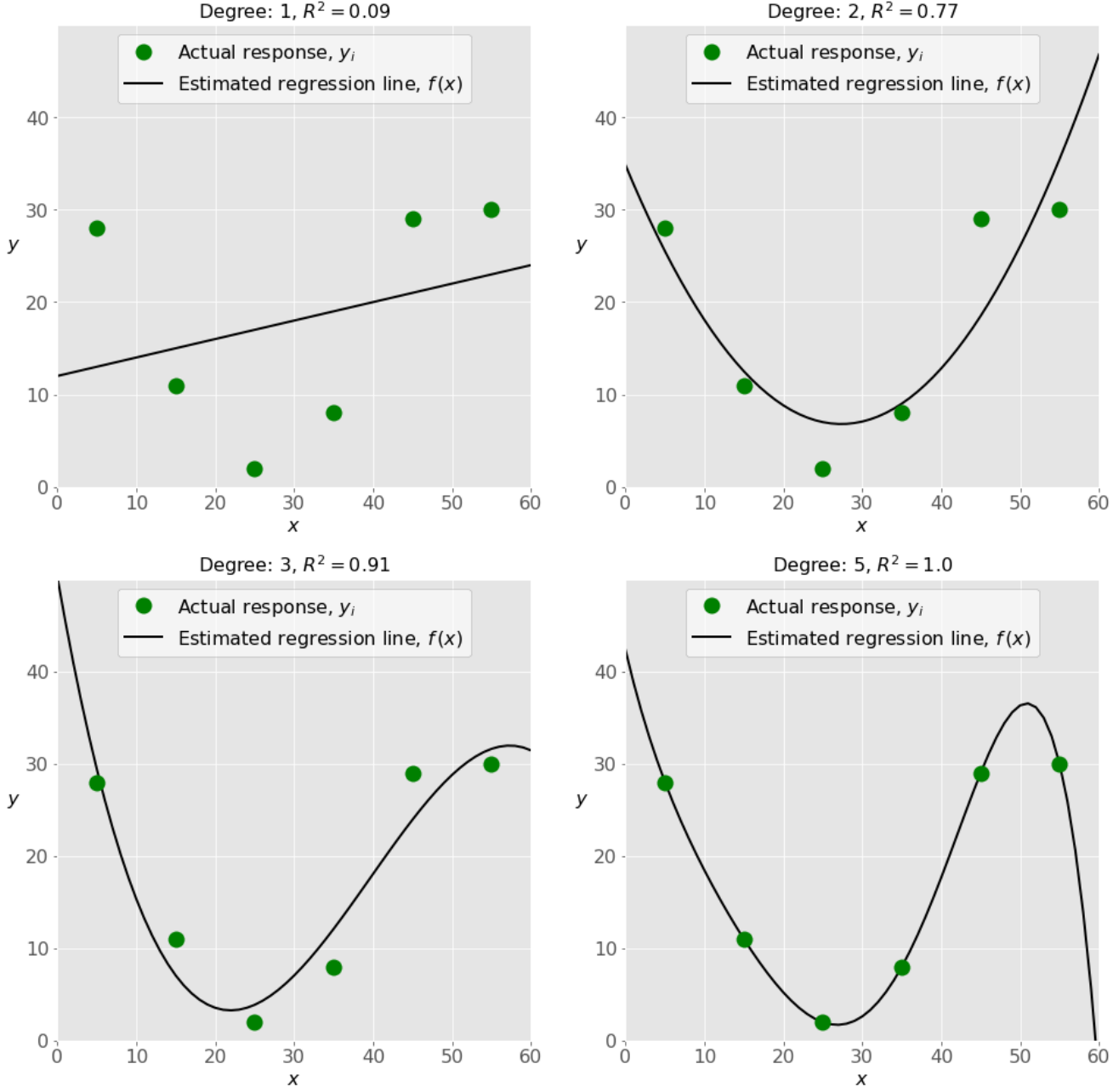
$\bar{Y}$ = Mean of Y

الانحدار



https://realpython.com/linear-regression-in-python/

الانحدار



https://realpython.com/linear-regression-in-python/

كيف نعمل؟

Python لديه دالة للعثور على علاقة بين نقاط البيانات ورسم خط للانحدار الخطي.

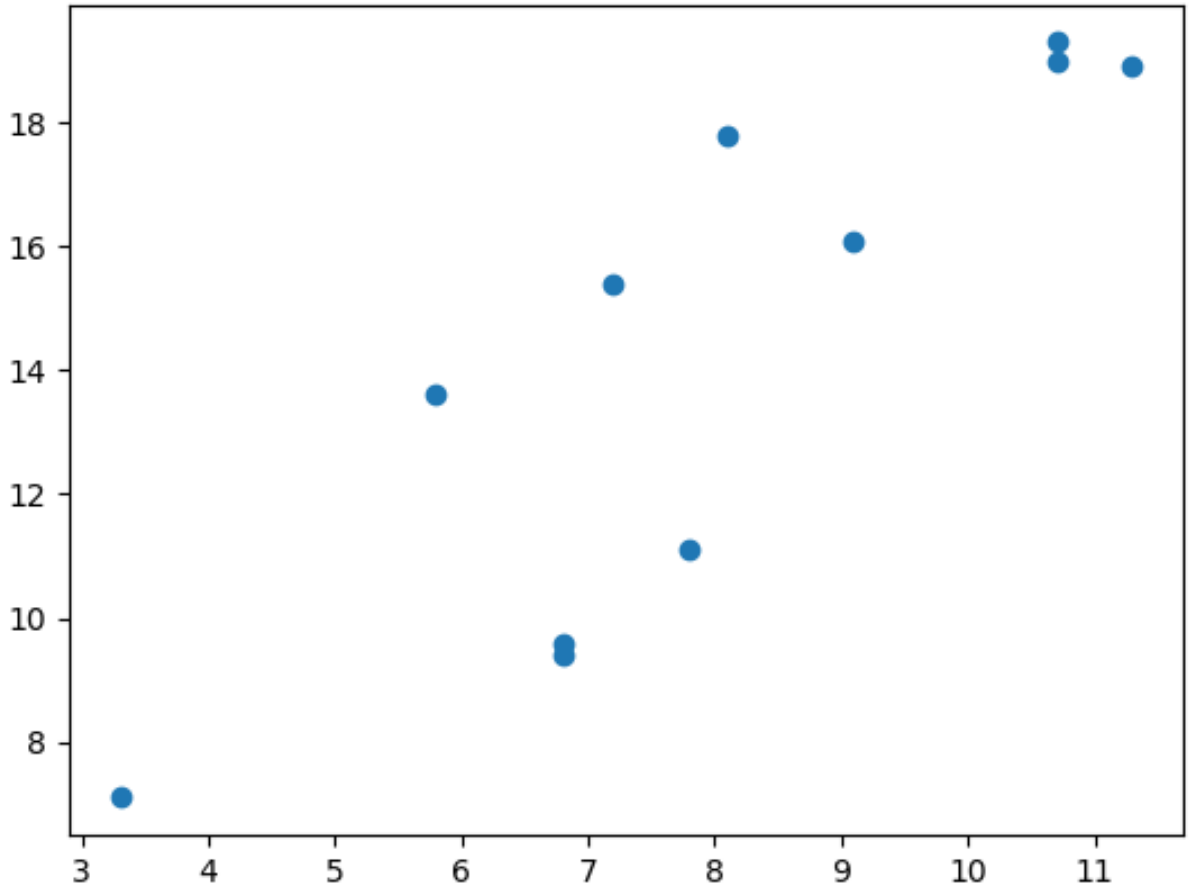
يتم رسم البيانات باستخدام رسم scatter في matplotlib

كيف يعمل؟

يستخدم المثال التالي بيانات عن درجات الحرارة في الليل والنهار:

```
x = [8.1, 11.3, 9.1, 7.8, 6.8, 6.8, 7.2, 10.7, 10.7, 5.8, 3.3]

y = [17.8, 18.9, 16.1, 11.1, 9.4, 9.6, 15.4, 19.3, 19.0, 13.6, 7.1]
```

```
import matplotlib.pyplot as plt

x = [8.1, 11.3, 9.1, 7.8, 6.8, 6.8, 7.2, 10.7, 10.7, 5.8, 3.3]

y = [17.8, 18.9, 16.1, 11.1, 9.4, 9.6, 15.4, 19.3, 19.0, 13.6, 7.1]
plt.scatter(x, y)
plt.show()
```



تنفيذ الانحدار الخطي

رسم الخط الأفضل ملاءمة باستخدام scatter plot في matplotlib

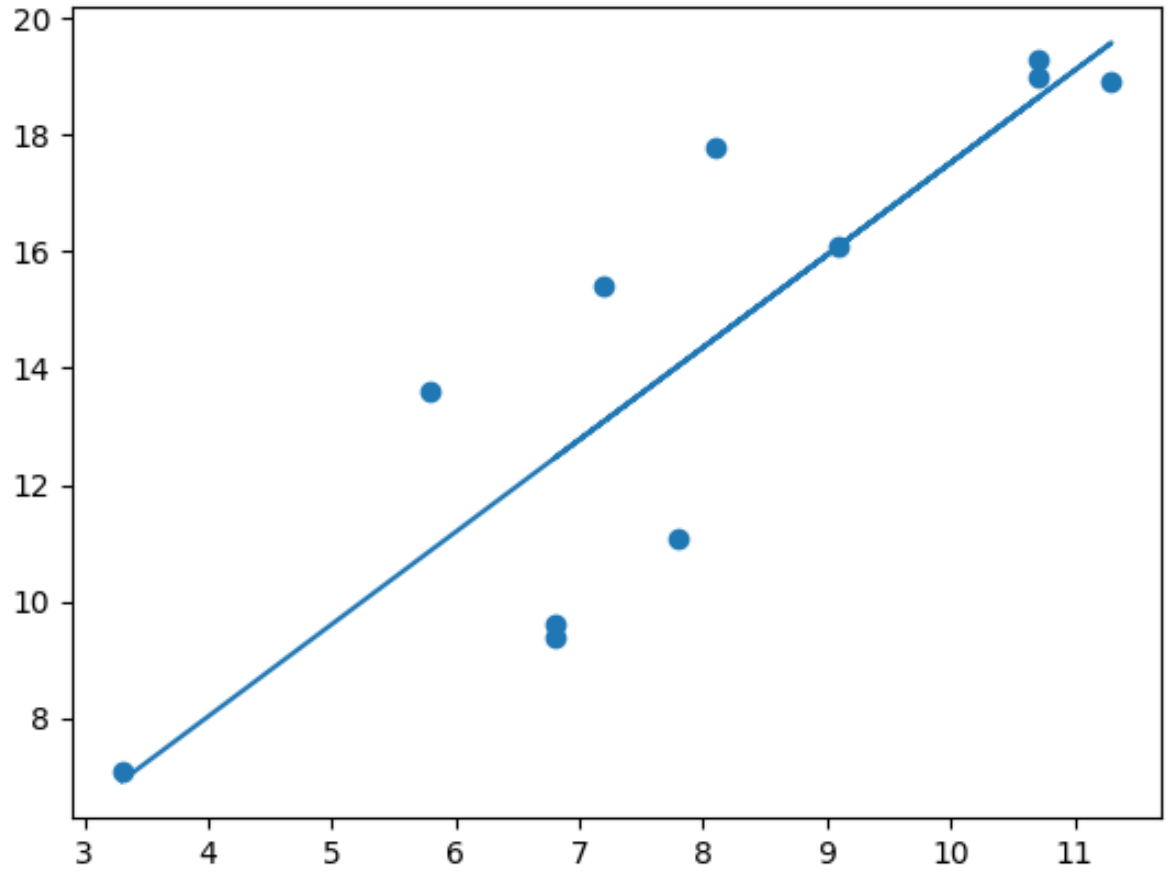
إنشاء دالة باستخدام slope and intercept لحساب قيم y لخط أفضل ملاءمة.

يُستخدم scipy لحساب الميل، والتقاطع، والقيمة r ، والقيمة p ، والخطأ القياسي للبيانات.

تنفيذ الانحدار الخطي

- import matplotlib.pyplot as plt
- from scipy import stats
- x = [8.1, 11.3, 9.1, 7.8, 6.8, 6.8, 7.2, 10.7, 10.7, 5.8, 3.3]
- y = [17.8, 18.9, 16.1, 11.1, 9.4, 9.6, 15.4, 19.3, 19.0, 13.6, 7.1]

- slope, intercept, r, p, std_err = stats.linregress(x, y)
- def myfunc(x):
- return slope * x + intercept
- mymodel = list(map(myfunc, x))
- plt.scatter(x, y)
- plt.plot(x, mymodel)
- plt.show()



فهم معامل الارتباط

- يصف معامل الارتباط R العلاقة بين قيم x و y
- قيمة R تساوي 0 تعني عدم وجود علاقة، بينما قيمة r تساوي 1 أو 1- تعني ارتباطًا بنسبة 100%.

تنفيذ الانحدار الخطي

```
import matplotlib.pyplot as plt
from scipy import stats
from sklearn import linear_model
import pandas as pd
A = [6.46, 6.4, 7.37, 6.94, 7.19, 5.61, 1.47, -1.94, -1.82, 3.86, 1.48, -0.63, 2.19]
B = [8, 8.53, 9.55, 9.45, 9.51, 8.54, 6.62, 4.15, 4.26, 7.57, 6.58, 5.81, 6.91]
C = [9.78, 10.58, 11.82, 11.92, 11.94, 11.47, 11.82, 10.23, 10.2, 10.95, 11.28, 11.81, 11.01]
data = {'A': A, 'B': B}
target= {'C': C}
X = pd.DataFrame(data)
Y = pd.DataFrame(target)
regr = linear_model.LinearRegression()
regr.fit(X, Y)
```

التصنيف (Classification)

- تصنيف البيانات هو عملية تنظيم البيانات إلى مجموعات فرعية وفقًا لسمات مشتركة.
- تساعد عملية التصنيف في فهم البيانات والبحث عن الأنماط والتنبؤ بالتصرفات المستقبلية.

- في مجال البيانات الضخمة، يلعب تصنيف البيانات دورًا حاسمًا في تنظيم وفهم كميات هائلة من البيانات .يمكن استخدامه لتصنيف مختلف أنواع البيانات مثل الصور والنصوص والفيديوهات والمزيد.

التصنيف (Classification)

- أمثلة على تقنيات التصنيف
- الجيران الأقرب k-NN
- الشجرة القرارية

تصنيف(KNN Classification)

- تصنيف KNN (K-Nearest Neighbors Classification)هو أحد أساليب التعلم الآلي المستخدمة لتصنيف البيانات إلى فئات مختلفة.

- يعتمد تصنيف KNNعلى مبدأ الجاورة القريبة، حيث يتم تصنيف نقطة جديدة على أساس النقاط القريبة منها.
- لتصنيف نقطة جديدة باستخدام KNN، يتم حساب المسافة بين النقطة الجديدة وجميع النقاط في مجموعة التدريب.

- يتم اختيار أقرب Kنقاط للنقطة الجديدة، ويتم تصنيفها بناءً على الغالبية من فئات النقاط القريبة.

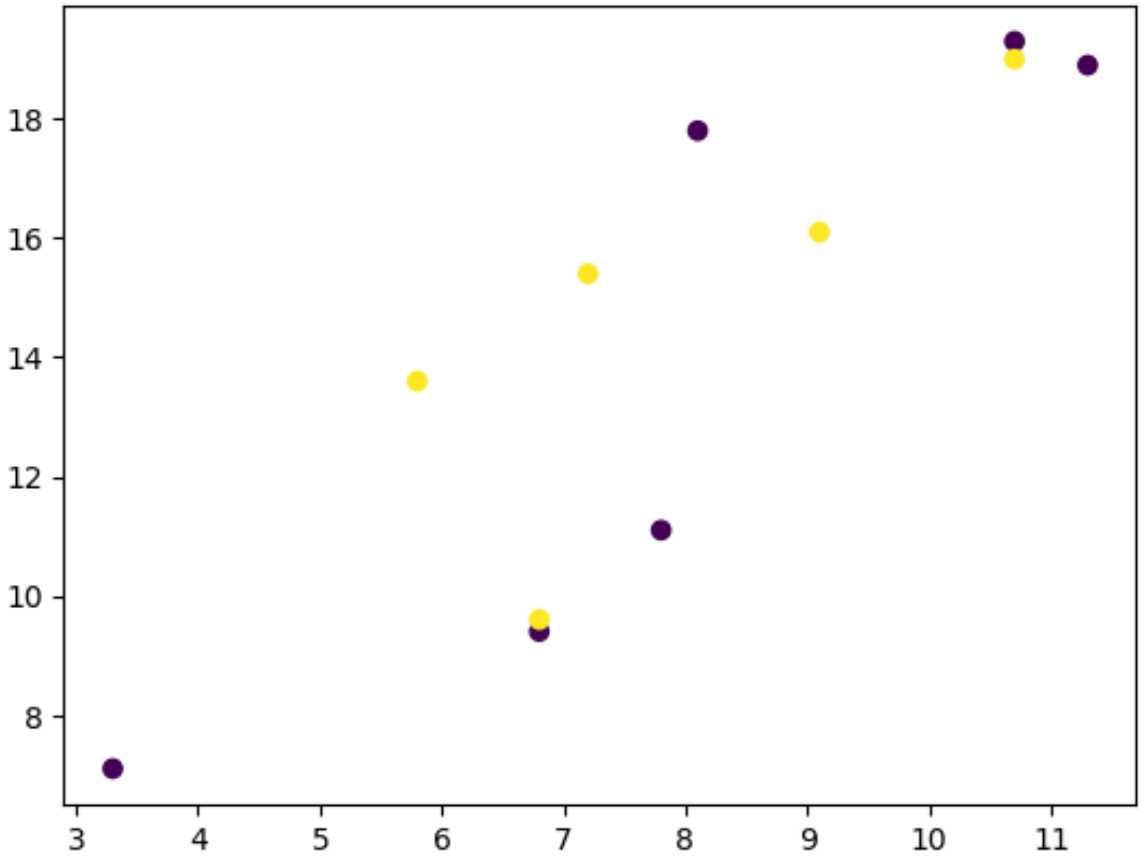
كيف يعمل؟

يستخدم المثال التالي بيانات عن درجات الحرارة في الليل والنهار:

```
x = [8.1, 11.3, 9.1, 7.8, 6.8, 6.8, 7.2, 10.7, 10.7, 5.8, 3.3]

y = [17.8, 18.9, 16.1, 11.1, 9.4, 9.6, 15.4, 19.3, 19.0, 13.6, 7.1]
```

```
classes = [N, N, Y, N, N, Y, Y, N, Y, Y, N]
```

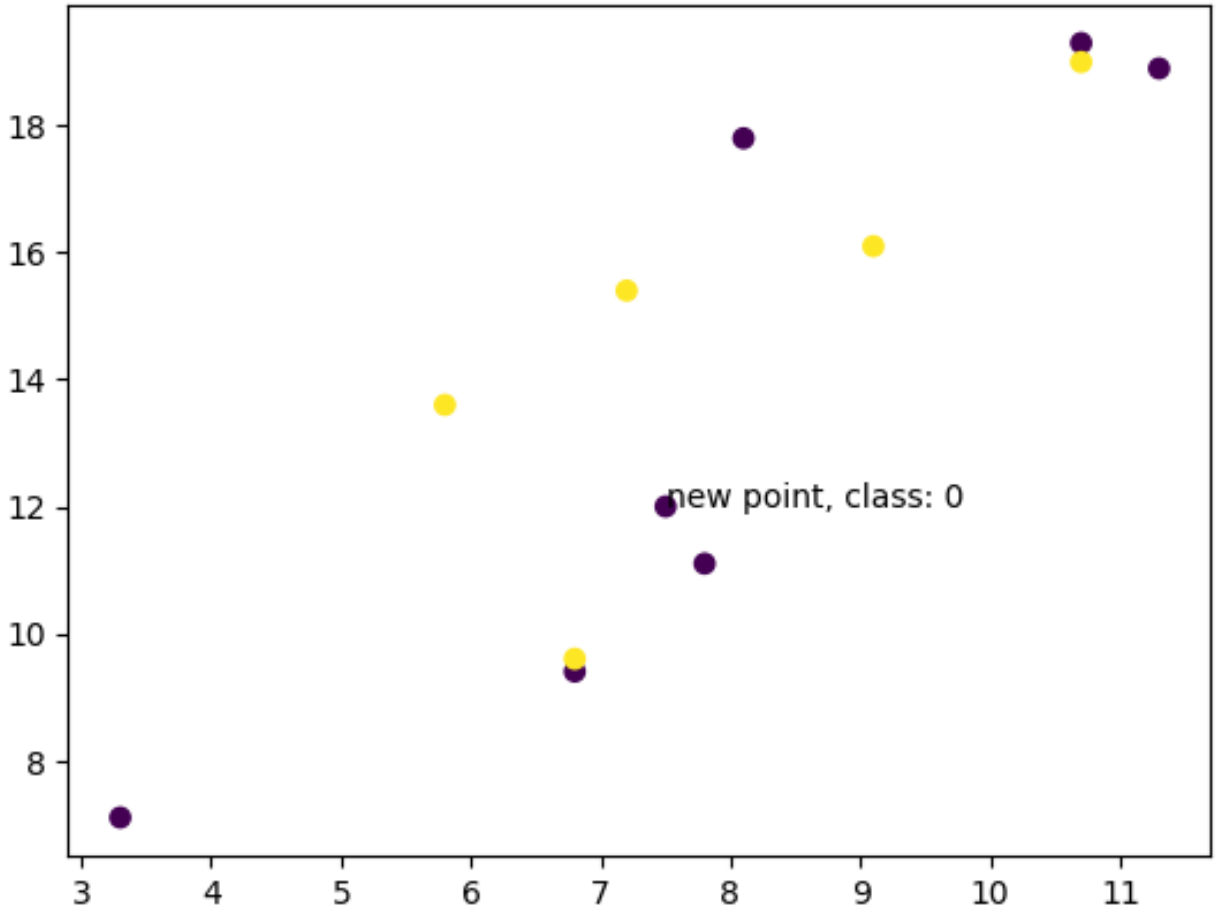
```
import matplotlib.pyplot as plt
x = [8.1, 11.3, 9.1, 7.8, 6.8, 6.8, 7.2, 10.7, 10.7, 5.8, 3.3]

y = [17.8, 18.9, 16.1, 11.1, 9.4, 9.6, 15.4, 19.3, 19.0, 13.6, 7.1]
classes = [0, 0, 1, 0, 0, 1, 1, 0, 1, 1,0]

pt.scatter(x, y, c=classes)

pt.show()
```



تنفيذ KNN

- import matplotlib.pyplot as ptfrom
- sklearn.neighbors import KNeighborsClassifier
- x = [8.1, 11.3, 9.1, 7.8, 6.8, 6.8, 7.2, 10.7, 10.7, 5.8, 3.3]
- y = [17.8, 18.9, 16.1, 11.1, 9.4, 9.6, 15.4, 19.3, 19.0, 13.6, 7.1]
- classes = [0, 0, 1, 0, 0, 1, 1, 0, 1, 1,0]
- data = list(zip(x, y))
- knn = KNeighborsClassifier(n_neighbors=1)
- knn.fit(data, classes)

- new_x = 7.5
- new_y = 12
- new_point = [(new_x, new_y)]
- prediction = knn.predict(new_point)
- print(prediction)
- plt.scatter(x + [new_x], y + [new_y], c=classes + [prediction[0]])
- plt.text(x=new_x, y=new_y, s=f"new point, class: {prediction[0]}")
- plt.show()



الشجرة القرارية (Decision Tree)

- الشجرة القرارية ((Decision Treeهي أحد الخوارزميات في مجال تعلم الآلة وتحليل البيانات تُستخدم للتصنيف والتنبؤ. تعتمد الشجرة القرارية على مبدأ تقسيم البيانات إلى أجزاء أصغر استنادًا إلى سلوك متغيرات مختلفة في البيانات. وفيما يلي شرح موجز لكيفية عمل الشجرة القرارية:
- المبدأ العام: تهدف الشجرة القرارية إلى تقسيم البيانات إلى مجموعات فرعية صغيرة قابلة للتصنيف. يتم ذلك عبر تسلسل من القرارات التي تعتمد على متغيرات معينة.
- القمة (الجذر): تمثل القمة الجذر في الشجرة السؤال الأول الذي يجب البحث عن إجابته. ويتعين على القمة تقديم سؤال يساعد في تصنيف البيانات.
- الأفرع: بناءً على إجابة السؤال في القمة، يتم التحول إلى الأفرع الفرعية في الشجرة. كل فرع يمثل اختيار متغير معين ويحتوي على سؤال جديد.
- الأوراق (الأهداف): تمثل الأوراق في نهاية كل فرع في الشجرة. تحتوي الأوراق عادة على التصنيف النهائي للبيانات.

الشجرة القرارية (Decision Tree)

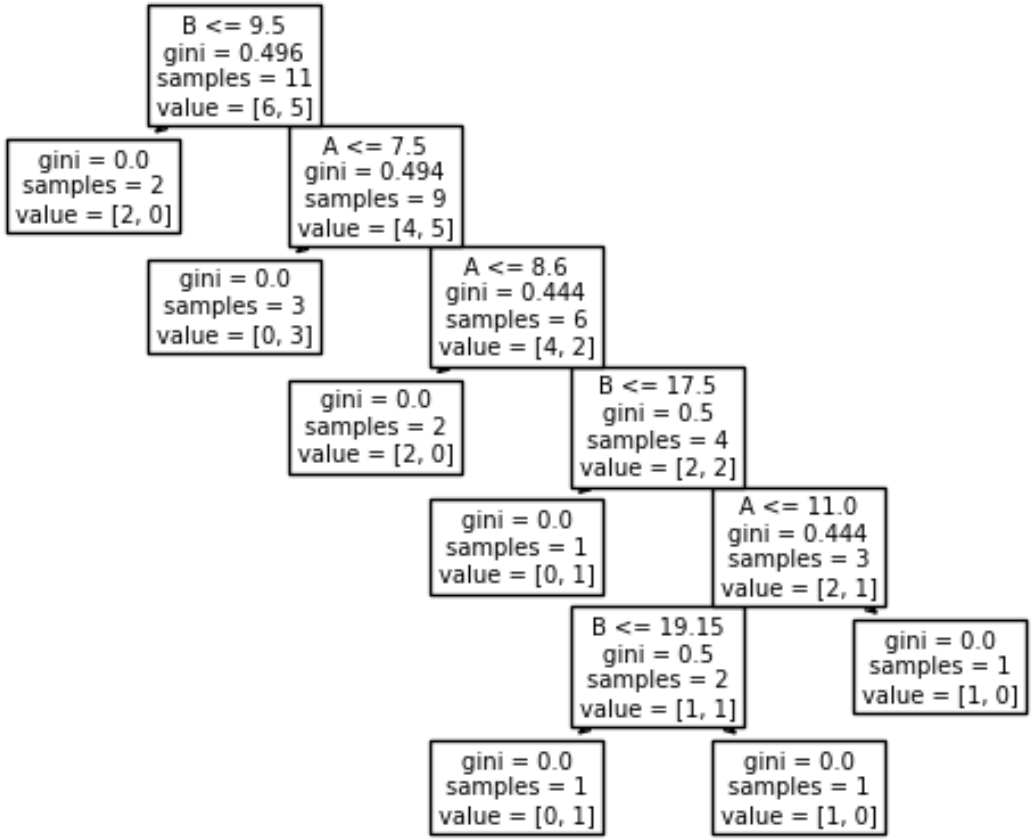
- التصنيف والتنبؤ: عندما تكون البيانات جاهزة للتصنيف، يتم البدء من القمة ومتابعة السؤال في كل فرع حتى الوصول إلى ورقة. الوصول إلى ورقة يعني التصنيف النهائي للبيانات.

- تحسين الشجرة: يمكن تحسين الشجرة بمعالجة وتقليل العمق، وهذا يساعد في تقليل الفقدان وزيادة دقة التصنيف.

- الشجرة القرارية تُستخدم في العديد من المجالات، مثل تصنيف البريد الإلكتروني كـ"سبام" أو "غير سبام" وأيضًا في تنبؤ الأحداث مثل التنبؤ بالطقس. تمتاز بالسهولة في التفسير والتطبيق وتوفير نتائج سريعة.

Decision Tree تنفيذ

- from sklearn import treefrom sklearn.tree
- import DecisionTreeClassifier
- import matplotlib.pyplot as plt
- import pandas as pd
- A = [8.1, 11.3, 9.1, 7.8, 6.8, 6.8, 7.2, 10.7, 10.7, 5.8, 3.3]
- B = [17.8, 18.9, 16.1, 11.1, 9.4, 9.6, 15.4, 19.3, 19.0, 13.6, 7.1]
- classes = [0, 0, 1, 0, 0, 1, 1, 0, 1, 1,0]
- features = ['A', 'B’]
- # Use the classes list directly as the target variable
- Y = classes
- # Initialize and fit the decision tree
- classifierdtree = DecisionTreeClassifier()
- dtree = dtree.fit(X, Y)
- # Plot the decision treetree.plot_tree(dtree)
- plt.show()

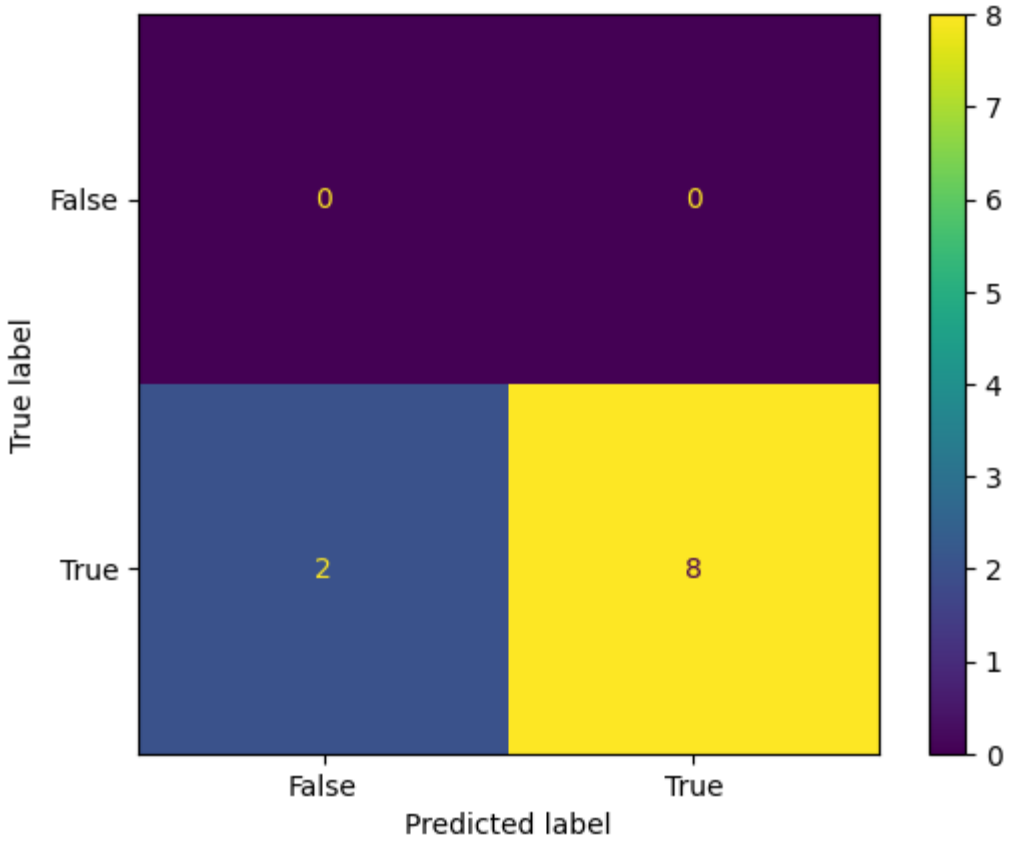


- print(dtree.predict([[4, 1], [7, 1]]))
- [0 0]

مصفوفة الارباك(Confusion matrix)

```
import matplotlib.pyplot as plt
import numpy
from sklearn import metrics
actual = numpy.random.binomial(1,.9,size = 10)
predicted = numpy.random.binomial(1,.9,size = 10)
confusion_matrix = metrics.confusion_matrix(actua
predicted)
cm_display =
metrics.ConfusionMatrixDisplay(confusion_matrix =
confusion_matrix, display_labels = [False, True])
cm_display.plot()
plt.show()
```

True Negative (Top-Left Quadrant)
False Positive (Top-Right Quadrant)
False Negative (Bottom-Left Quadrant)
True Positive (Bottom-Right Quadrant)



تقييم نتائج التصنيف

- الدقة Accuracy
- تقيس الدقة كم مرة كان النموذج صحيحًا.

- الضبط Precision
- من الإيجابيات المتوقعة، ما هي النسبة المئوية التي هي حقا إيجابية؟ الدقة لا تقيم الحالات السلبية التي تم التنبؤ بها بشكل صحيح.

- (True Positive + True Negative) / Total Predictions
- Accuracy = metrics.accuracy_score(actual, predicted)
- True Positive / (True Positive + False Positive)
- Precision = metrics.precision_score(actual, predicted)

تقييم نتائج التصنيف

- الحساسية Sensitivity أو Recall من جميع الحالات الإيجابية، ما هي النسبة المئوية التي تم التنبؤ بها بأنها إيجابية؟
- تقيس الحساسية جودة النموذج في التنبؤ بالنتائج الإيجابية.
- ينظر إلى الإيجابيات الحقيقية والسلبيات الكاذبة (التي هي إيجابية تم التنبؤ بها بشكل غير صحيح على أنها سلبية).

- True Positive / (True Positive + False Negative)
- Sensitivity_recall = metrics.recall_score(actual, predicted)

- التحديد Specificity
- كم هي جودة النموذج في التنبؤ بالنتائج السلبية؟
- التحديد مشابه للحساسية، ولكنه ينظر إليه من منظور النتائج السلبية.

- True Negative / (True Negative + False Positive)
- Specificity = metrics.recall_score(actual, predicted, pos_label=0)

تقييم نتائج التصنيف

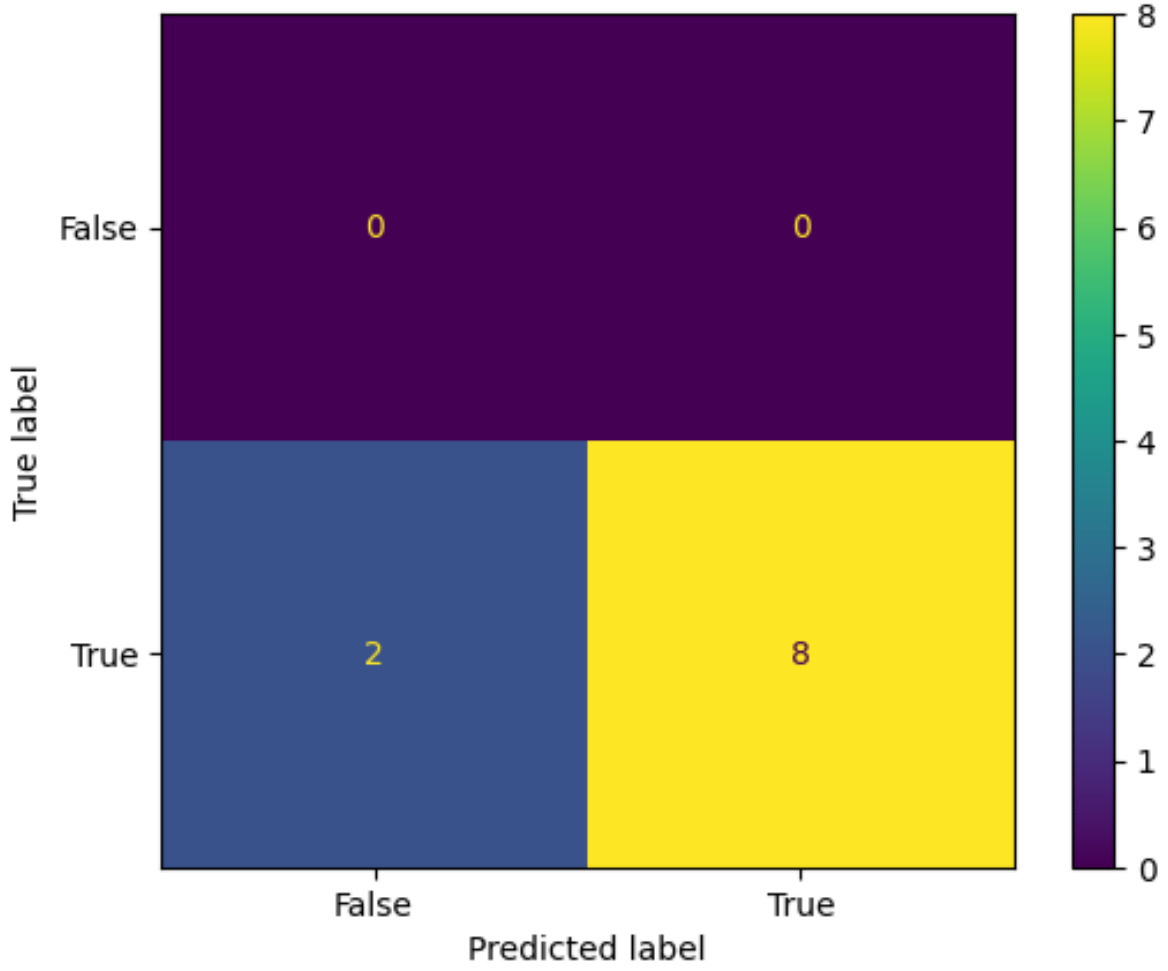
- F-score هو "المتوسط التوافقي" للدقة والحساسية.
- إنه يأخذ في الاعتبار كلاً من حالات الإيجابية الكاذبة والسلبية الكاذبة وهو جيد لمجموعات البيانات غير المتوازنة.
- {'Accuracy': 0.8, 'Precision': 1.0, 'Sensitivity_recall': 0.8, 'Specificity': 0.0, 'F1_score': 0.888888888888889}

- 2 * ((Precision * Sensitivity) / (Precision + Sensitivity))
- F1_score = metrics.f1_score(actual, predicted)
- print({"Accuracy":Accuracy,"Precision":Precision,"Sensitivity_recall":Sensitivity_recall,"Specificity":Specificity,"F1_score":F1_score})

تقييم نتائج التصنيف

- print({"Accuracy":Accuracy,"Precision": Precision,"Sensitivity_recall":Sensitivity_recall,"Specificity":Specificity,"F1_score":F1_score})

- {'Accuracy': 0.8, 'Precision': 1.0, 'Sensitivity_recall': 0.8, 'Specificity': 0.0, 'F1_score': 0.888888888888889}

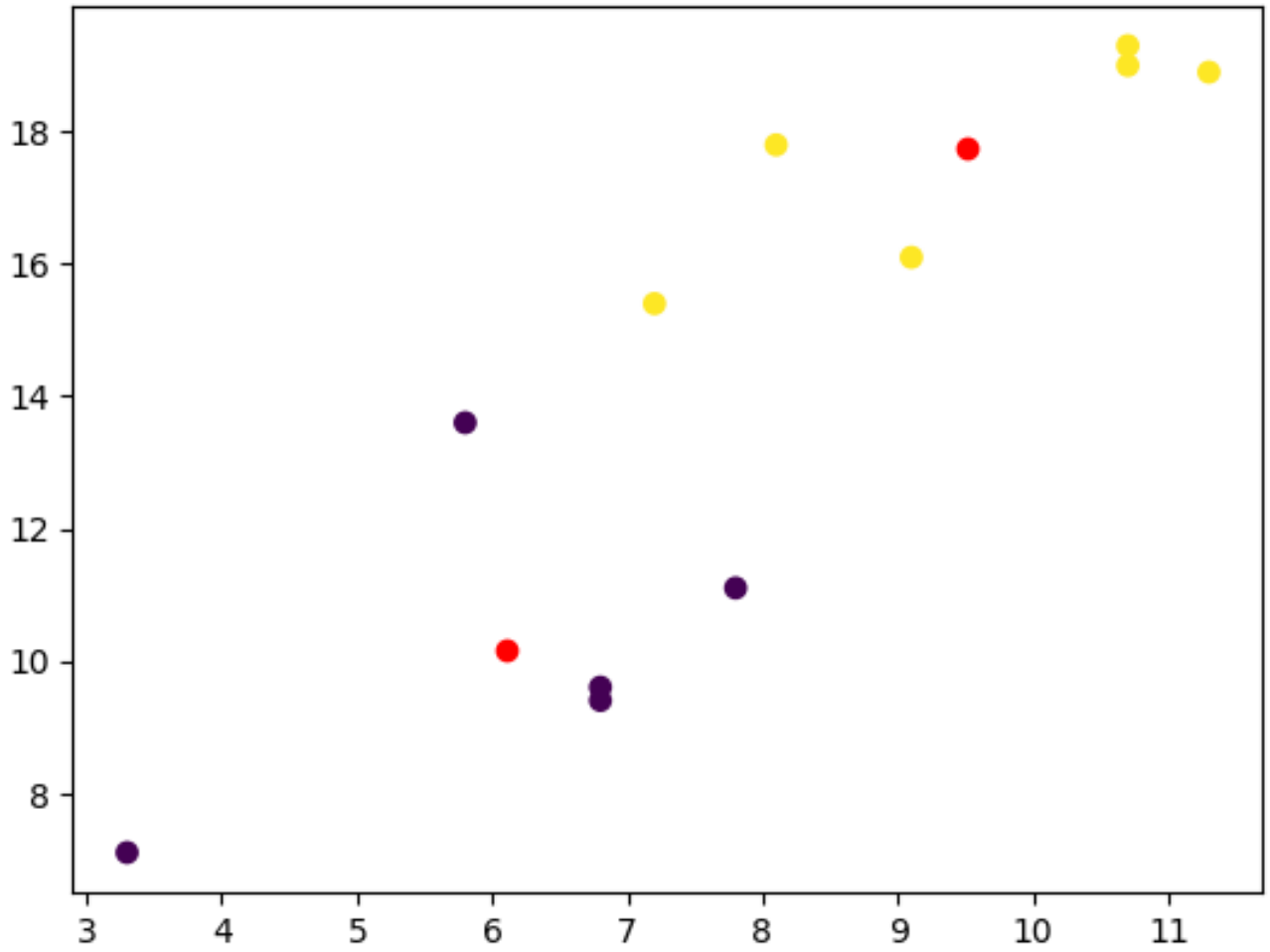


التجميع باستخدام(K-means Clustering) K-means

- خوارزمية K-Means هي خوارزمية للتعلم غير المشرف تستخدم لتصنيف مجموعة بيانات إلى عناقيد.
- تختار الخوارزمية عددًا من النقاط k لكل عنقودة تُعرف بالنقاط الوسطى.
- تشكل كل نقطة بيانات عنقودة مع النقاط الوسطى الأقرب إليها.
- يتم إيجاد النقطة الوسطى لكل عنقودة بناءً على أعضاء العنقودة الحاليين.
- تكرر الخوارزمية هذه العملية حتى يحدث التقارب، أي أن النقاط الوسطى لا تتغير.
- يتم تحديد قيمة k بحيث يكون مجموع تربيع الفرق بين النقطة الوسطى والنقاط داخل العنقودة هو الأدنى.

K-means تنفيذ

- import matplotlib.pyplot as pltfrom sklearn.cluster
- import Kmeans
- x = [8.1, 11.3, 9.1, 7.8, 6.8, 6.8, 7.2, 10.7, 10.7, 5.8, 3.3]
- y = [17.8, 18.9, 16.1, 11.1, 9.4, 9.6, 15.4, 19.3, 19.0, 13.6, 7.1]
- X = list(zip(x,y))
- kmeans = KMeans(n_clusters=2)
- kmeans.fit(X)
- print(kmeans.cluster_centers_)
- plt.scatter(x,y,c=kmeans.labels_, cmap='viridis’)
- plt.scatter(kmeans.cluster_centers_[:,0] ,kmeans.cluster_centers_[:,1], color='red’)
- plt.show()

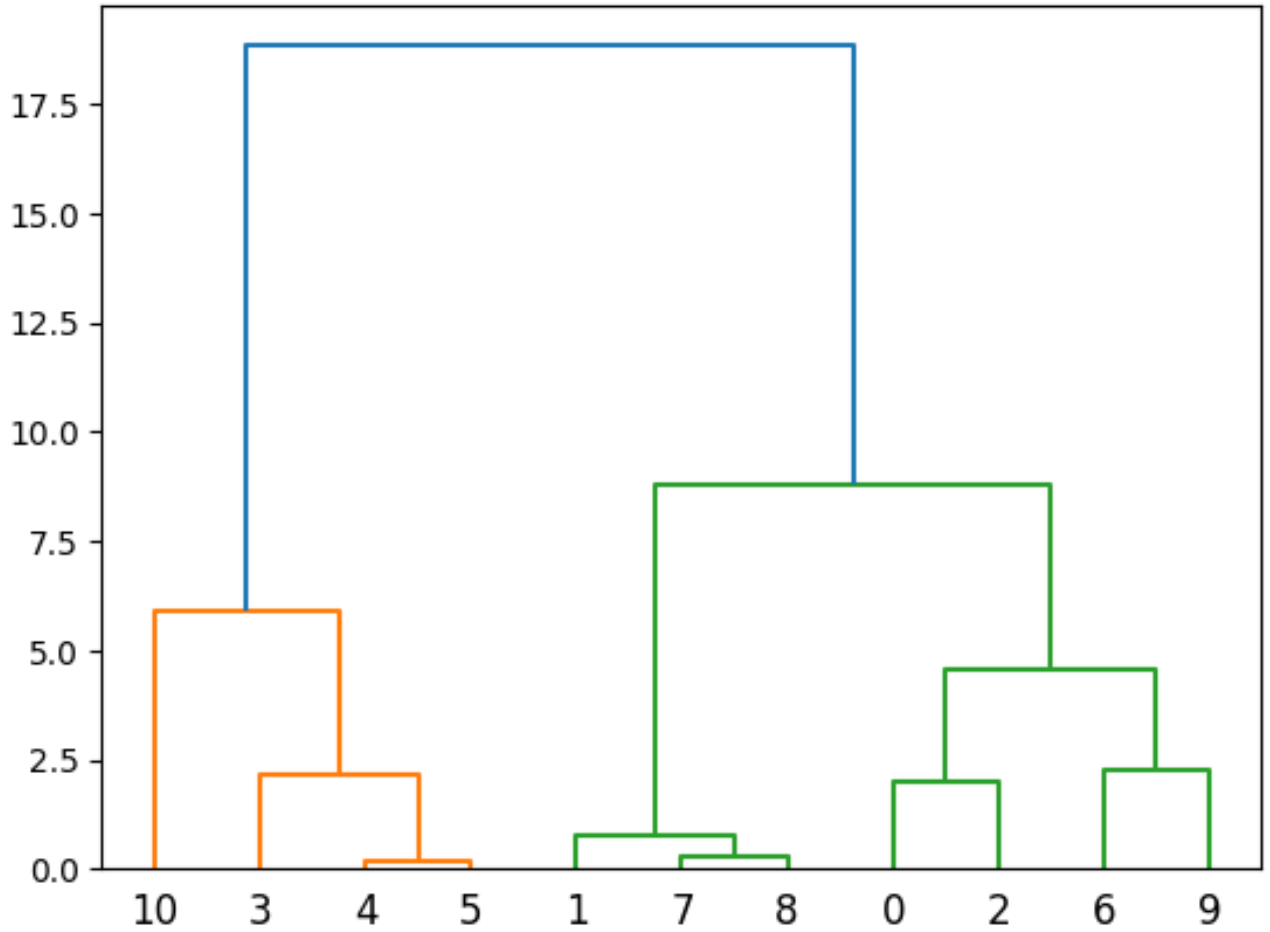


التجميع التسلسلي(Hierarchical Clustering)

- تعمل هذه الخوارزمية عن طريق دمج العناصر المشابهة معًا لتشكيل تجمعات أكبر.
- تبدأ الخوارزمية بتعيين كل عنصر بتجمع فردي ثم تدمج التجمعات بناءً على المشابهة بينها.
- يتم تكرار هذه العملية حتى يتم تشكيل تجمعات كبيرة تحتوي على جميع العناصر المشابهة.
- مثال: تجميع أبراج الهواتف المحمولة بناءً على قوة الإشارة.

Hierarchical Clusterings تنفيذ

- import numpy as np
- import matplotlib.pyplot as pltfrom scipy.cluster.hierarchy
- import dendrogram, linkage
- x = [8.1, 11.3, 9.1, 7.8, 6.8, 6.8, 7.2, 10.7, 10.7, 5.8, 3.3]
- y = [17.8, 18.9, 16.1, 11.1, 9.4, 9.6, 15.4, 19.3, 19.0, 13.6, 7.1]
- X = list(zip(x,y))
- linkage_data = linkage(X, method='ward', metric='euclidean’)
- dendrogram(linkage_data)
- plt.show()



خلاصة

- في ختام هذا العرض التقديمي، نستعرض النقاط الرئيسية التي تم مناقشتها:
  - الانحدار
  - التصنيف
  - مقاييس التقييم
- يمكن استخدام خوارزميات التعلم الآلي لحل مجموعة واسعة من المشكلات، ولكن من المهم دائمًا اختيار الخوارزمية المناسبة وفقًا لطبيعة المشكلة والبيانات المتاحة.

خلاصة

- تقنيات تحليل البيانات تلعب دوراً حاسماً في الإحصاءات الرسمية لأسباب عديدة:

- 1. الفهم العميق للبيانات: تقنيات تحليل البيانات تساعد في فهم البيانات بشكل أعمق وتوفير رؤى قيمة من البيانات التي قد تكون مخفية في البداية.

- 2. تحسين الدقة: يمكن أن تساعد هذه التقنيات في تحسين دقة التوقعات والتقديرات، مما يجعل الإحصاءات الرسمية أكثر موثوقية.

- 3. الكفاءة في التحليل: يمكن لتقنيات تحليل البيانات معالجة كميات كبيرة من البيانات بكفاءة، مما يوفر الوقت والجهد في التحليل.

خلاصة

- 4. التعامل مع بيانات معقدة: هذه التقنيات قادرة على التعامل مع بيانات معقدة وغير منظمة، والتي قد تكون صعبة في التحليل باستخدام الطرق التقليدية.

- 5. التنبؤ بالتوجهات المستقبلية: تستخدم تقنيات تحليل البيانات لتوقع التوجهات والأنماط المستقبلية، مما يساعد في التخطيط وصنع القرار.

- 6. التحديث المستمر للإحصاءات: بفضل التحديث المستمر للبيانات، يمكن لتقنيات تحليل البيانات أن توفر إحصاءات حديثة ومحدثة باستمرار.

- يمكن أن تكون تقنيات تحليل البيانات أداة قوية جداً في إثراء وتحسين جودة الإحصاءات الرسمية.

الأسئلة

شكرا