等滲回歸和PAVA演算法

發佈於 2021-04-21 10：46：35 1.9K 1 舉報

磐創AI分享

來源 | analyticsvidhya

作者 | NOOBMASTER21

編譯 | Flin

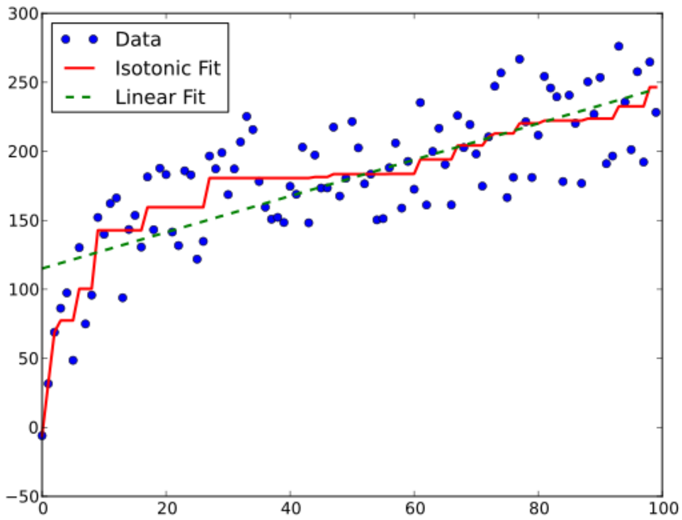
介紹

等滲回歸是很少被談論但肯定是最酷的回歸技術之一。 我之所以說“很少談論”，是因為與線性回歸不同，它不經常被講授或使用。 等滲回歸做出一個更籠統的假設，即最能代表數據的函數是單調的，而不是線性的（是的，線性也是單調的，反之亦然）。

“Isotonic（等滲）”一詞源自兩個希臘詞根：“ *iso* ”和“ *tonos* ”;” *iso* “的字面意思是相等，” tonos “的意思是伸展。

因此，等滲回歸（也稱為等長回歸），對數據擬合一個分段常數非遞減（階梯狀）函數，因此提供了線性回歸的替代方法，線性回歸本質上對數據擬合一條直線。

與線性回歸相比，這是等滲回歸的樣子。



如果你像我一樣，那麼你可能想知道演算法背後的數學知識，然後才能開始使用它們。 對於任何希望使用該演算法獲得最佳結果的人來說，確切地知道是什麼原因導致了演算法的行為方式是一個好習慣。

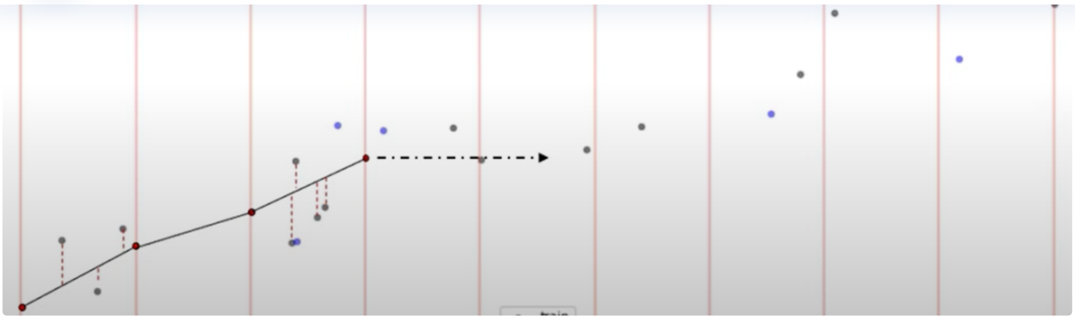
所以這是我們要做的：

1. 首先，我們將闡述等滲回歸所解決的問題。
2. 然後，我們將看一些數學並嘗試了解解決方案。 在這一部分中，我們將學習PAVA演算法。
3. 最後，我們將研究它在python中的實現及其主要應用程式。

問題表述

等滲回歸解決了以下問題：給定的順序 n 個數據點 *y1，y2....yn*， 我們如何才能通過β*1,....*βn 單調序列總結這一序列?

$$\begin{aligned} &\text{minimize}_{\beta_1,\dots,\beta_n} \quad \sum_{i=1}^{n}(y_i-\beta_i)^2 \\ &\text{subject to} \quad \beta_1 \le \cdots \le \beta_n. \end{aligned}$$



要制定單調序列及其含義，我們必須做一個假設。

同方差的正態誤差

像其他線性模型一樣，我們可以假定這種回歸類型中的誤差是同方差的。 換句話說，所有誤差將具有有限的方差。 由於誤差不依賴於預測值 xi，我們可以制定一個可以有效擬合數據的線性模型。 同樣，我們可以假設因變數是隨機的，並且它們服從正態分佈。

$$Y_i \text{ is Normal}(\mu_i, \sigma^2)$$

和

$$x_i \leq x_j \quad \text{implies} \quad \mu_i \leq \mu_j$$

以上條件是由於單調性約束。

處理重複的預測變數值

我們可以看到，當 xi = xj 意味著 μi = μj （相同的x值在y的分佈中有相同的平均值）。

因此，對於所有重複的預測器或x值，我們可以有一個均值參數，換句話說，對於每個唯一的預測器或x值，我們在y中只能有一個均值參數。

它甚至意味著什麼？ 為什麼我們要首先考慮這一點？ 我的意思是我們怎麼會有重複的x值？

我將在這裡解釋問題的一部分，另一部分需要更多解釋。 那麼，重複或相同的x值意味著什麼？

正如我們在定義本身中看到的，等滲回歸以單調方式擬合數據。 因此，在擬合數據時，如果演算法發現違反此單調性約束的點，則該點將與相鄰的x值合併在一起，以形成我們之前考慮的塊或**單調序列**。

很酷的是，**單調序列或塊**中的所有x值都將具有相同的y值。 你可以在上面的等滲曲線中看到這一點，等滲曲線的平坦部分或拉伸部分（沒錯，等拉伸回歸就是從這裡得名）。

這些顯示為繪圖中的步驟。 現在，我們要在這些所謂的塊中找到y的值。 你現在可能已經可以看到PAVA演算法是如何解決的。 現在，讓我們確切地瞭解y值的計算方式。

令 z1,....,zk 為按排序順序的唯一x值（以後將具有相同的y值）。

然後，我們可以將所有唯一x值的**權重**定義為：

$$w_j = \sum_{\substack{i=1 \\ x_i = z_j}}^{n} 1$$

因此，現在y值變為：

$$V_j = \frac{1}{w_j} \sum_{\substack{i=1 \\ x_i = z_j}}^{n} Y_i$$

$$V_j \sim \mathrm{Normal}(\nu_j, \sigma^2/w_j)$$

現在，此分佈表示將要預測的唯一y值，其分量為 v1,……,vk。

這些元件將是單調的，因為這是我們必須遵循的約束。

$$\nu_1 \leq \nu_2 \leq \cdots \leq \nu_k$$

還，

$$\nu_j = \mu_i \quad \text{whenever} \quad z_j = x_i$$

現在我們需要回到數學描述。

負對數似然

你一定聽說過線性回歸中的最大似然估計及其最終如何給出最佳擬合線。 通常，我們嘗試使似然函數最大化，但是如果我們取似然函數的對數並將整個表達式乘以–1，則得到的是負對數似然，它會最小化而不是因為–1而最大化。 因此，基本上，我們通過最小化來最大化。

$$f(\mu, \sigma^2) = n\log(\sigma) + \frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - \mu_i)^2$$

我們通過取正態分佈（假設y值來自這個分佈）的對數乘以–1來獲得。

拉格朗日

如果你對約束優化略知一二，那麼你很可能聽說過拉格朗日函數。 每當我們面臨優化（在此處最小化）上述成本函數的任務時，在一些約束條件（這裡是單調性）下，我們使用拉格朗日乘子。

> Lagrange乘數：https://en.wikipedia.org/wiki/Lagrange_multiplier

$$L(\mu, \sigma^2, \lambda) = n\log(\sigma) + \frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - \mu_i)^2 + \sum_{j=1}^{k-1}\lambda_j(\nu_j - \nu_{j+1})$$

上式稱為拉格朗日方程。 求解該方程式將為我們提供負對數似然函數的最小值，從而最終使可能性最大化，從而確保與數據的最佳擬合。

請注意，除了對數似然函數中兩個已經存在的術語之外，又增加了一項。 這個額外的術語是由於我們正在考慮的約束條件而產生的結果。 因此，我們的問題是一個約束優化問題。

KKT條件

對於無約束凸優化問題，我們知道當一個點的梯度為零時可以達到最小值。 KKT條件是我們現在面臨的約束凸優化問題的全域最小值的等價條件。

這些條件是：

1. 拉格朗日最小化
2. 原問題的可行性
3. 對偶問題的可行性
4. 互補鬆弛性

對於第一個條件（**拉格朗日極小化**），

$$\frac{\partial L}{\partial \nu_m} = \lambda_m - \lambda_{m-1} - \frac{1}{\sigma^2}\sum_{i=1}(y_i - \mu_i)\frac{\partial \mu_i}{\partial \nu_m}$$
$$= \lambda_m - \lambda_{m-1} - \frac{w_m(v_m - \nu_m)}{\sigma^2}$$

請注意，關於y值或預測的值，我們如何獲取Lagrangian的偏導數。

需要注意的是，如果*xi=zm*，∂μi/∂vm是等於1，否則等於0。 因此，僅當**xi = zm** （其中z m代表唯一的x值或我們將擁有唯一y值的x值）時，第一行總和中的項才為非零。

可以看出，該公式僅對**1<m** <k<=“” strong=“” style=“font-size： inherit; color: inherit; line-height: inherit; “>有效。 以使其適用於1 <= M <= K，我們定義 λ0 =λķ = 0。 </k<="">

將上述方程式設置為零將為我們提供第一個 Karush–Kuhn–Tucker 條件。 讓我們把它設為零，乘以 **σ2**，

$$1.\ -w_m(v_m - \nu_m) + \kappa_m - \kappa_{m-1} = 0,\ m = 1, \ldots, k$$

這裡 *κm = σ2λm*已被引入。

可以看出，當 *κm = σ2λm*時，κm 為零、正或負，因此我們可以在其餘的KKT條件下使用 κm 。

現在第二個條件（原問題的可行性）：

$$2.\ \nu_j \leq \nu_{j+1},\ j = 1, \ldots, k-1.$$

對於第三個條件（對偶問題的可行性）：

$$3.\ \kappa_j \geq 0,\ j = 1, \ldots, k-1.$$

最後，（**互補鬆弛性 **）條件：

$$4.\ \kappa_j(\nu_j - \nu_{j+1}) = 0,\ j = 1, \ldots, k-1.$$

現在你可能想知道為什麼我們要引入 *κm = σ2λm* 。 原因是，當我們估計平均參數或y值時，換句話說，我們通常不考慮方差。 為了實現不含方差的最大似然估計，我們引入了一個只直接依賴於拉格朗日乘子**λm**的新變數。

現在我們有了KKT條件，我們準備好計算y值。

定義塊
首先，我們僅應用第一個（拉格朗日導數等於零）和第四個（互補鬆弛性）條件。

我們可以將y值的空間劃分為等量連續的塊，如果該塊中的值不等於任一側的平均值，則這些塊的長度將為1。

此時，我們將唯一的y值稱為 v，它是特定塊的平均值。

因此，讓我們考慮一個塊。

一個塊只有唯一的y值，在該塊中我們將其稱為*v （均值參數）* 。

假設，

$$\nu_{j^*-1} \neq \nu_{j^*} = \cdots = \nu_{j^{**}-1} \neq \nu_{j^{**}}$$

上標的星號用於區分兩個不同的塊。 因此，序列， vj⁎ = · · = vj⁎⁎−1 表示的塊。

現在該邊緣的情況下，我們定義 v0 =−∞和v K + 1 = +∞。

互補鬆弛性條件表明，

$$\kappa_{j^*-1} = \kappa_{j^{**}-1} = 0.$$

因此，現在使用第一個條件，

$$0 = \sum_{j=j^*} \left[-w_j(v_j - \nu_j) + \kappa_j - \kappa_{j-1} \right]$$
$$= -\sum_{j=j^*}^{j^{**}-1} w_j(v_j - \nu)$$

其中

$$\nu = \nu_{j^*} = \cdots = \nu_{j^{**}-1}.$$

解出上面的式子，

$$\nu_{j^*} = \cdots = \nu_{j^{**}-1} = \frac{\sum_{j=j^*}^{j^{**}-1} w_j v_j}{\sum_{j=j^*}^{j^{**}-1} w_j}$$

現在，這向我們揭示了一條非常重要且很酷的資訊。 僅通過應用第一和第四條件，我們發現，

等量塊中的平均值是vj值的加權平均值，vj值是塊的yi值的未加權平均值。

PAVA演算法（The Pool Adjacent Violators Algorithm）

PAVA演算法基本上可以執行其名稱所暗示的功能。 它檢查這些點，如果發現違反約束的點，則將其值與連續成員合併，最終形成一個塊。

具體而言，PAVA會執行以下操作：

1. **[初始化步驟] **將v和κ設置為滿足條件1，3和4的任何值。
2. [終止步驟]如果滿足條件2（原始條件），則終止。
3. [**池相鄰的違背者]選擇任意的j，使得 **vj > vj+1.。 然後「合併」包含j和j + 1的塊，使
 其成為一個塊（該合併塊的nu或μ值將再次成為該合併塊的值的加權平均值）。[此步驟保持條件1、3和4。]
4. 再次執行步驟2。

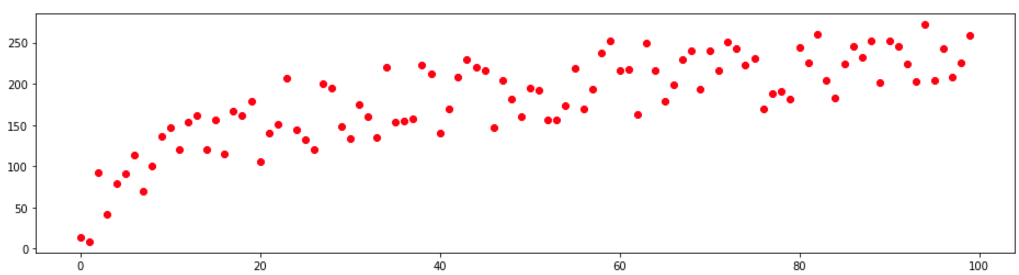
現在，我們對這種演算法的工作原理有了基本瞭解，讓我們在python中實現它：

首先，我們需要導入所需的庫：

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.isotonic import IsotonicRegression
from matplotlib.collections import LineCollection
import random
```


然後讓我們為其生成一些點和一些隨機雜訊：

```
n=100 #no. of points
x=np.arange(n)
y= 50*np.log(1+x) + np.random.randint(-50,50,size=(n,))
```

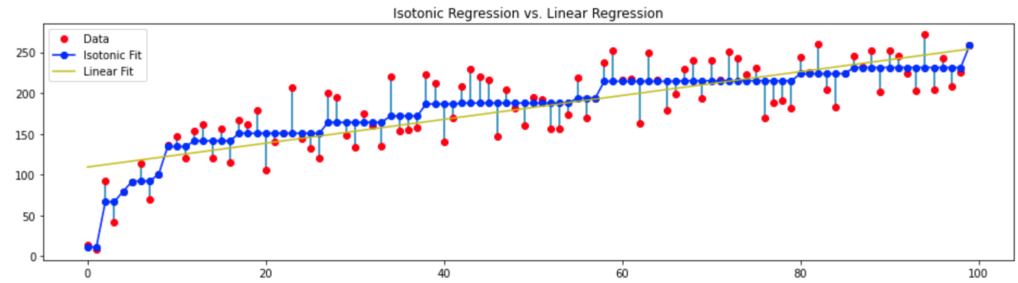


```
lr=LinearRegression()
y_ir=ir.fit_transform(x,y)
lr.fit(x[:,np.newaxis],y)
```

產生一些線段並繪製結果：

```
lines=[[[i,y[i]],[i,y_ir[i]]] for i in range(n)]
lc=LineCollection(lines)
plt.figure(figsize=(16,4))
plt.plot(x,y,'r.',markersize=12)
plt.plot(x,y_ir,'b.-',markersize=12)
plt.plot(x,lr.predict(x[:, np.newaxis]), 'y-')
plt.gca().add_collection(lc)
plt.legend(('Data','Isotonic Fit','Linear Fit'))
plt.title("Isotonic Regression")
```

最後結果：



人們還可以看到該演算法即時工作

應用領域

現在的問題是，等滲回歸在哪裡使用?

等滲回歸通常用於分類器的概率校準。 分類器的概率校准處理優化分類器輸出的過程，以便可以將分類器模型的輸出直接解釋為置信度。 例如：如果分類器以90%的概率將一定數量的樣本歸為某個特定類別，則這些樣本中大約90%實際上應該屬於該特定類別。

主要有兩種方法：

1. Platt's Scaling：將邏輯回歸模型擬合到分類器模型的輸出。
2. 等滲回歸：使等滲或階梯狀曲線適合模型的輸出。

現在，我們將看到等滲回歸如何用於概率校準：

注意：儘管等滲回歸在校正單調失真方面比Platt Scaling更強大，但它很容易使數據過擬合。 線性回歸的靈活性帶來了需要更多數據的代價。

讓我們導入庫：

```
from sklearn.datasets import make_classification
from sklearn.svm import SVC
from sklearn.calibration import CalibratedClassifierCV
from sklearn.model_selection import train_test_split
from sklearn.calibration import calibration_curve
```

產生資料並將其分為訓練集和測試集：

```
X, y = make_classification(n_samples=10000, n_classes=2, random_state=42)
trainX, testX, trainy, testy = train_test_split(X, y, test_size=0.2, random_state=42)
```

擬合兩種模型：一種不帶校準，一種帶校準：

```
model1=SVC()
model1.fit(trainX,trainy)
preds1=model1.decision_function(testX)
```

```
model2=SVC()
cal=CalibratedClassifierCV(model2,method='isotonic',cv=5)

```

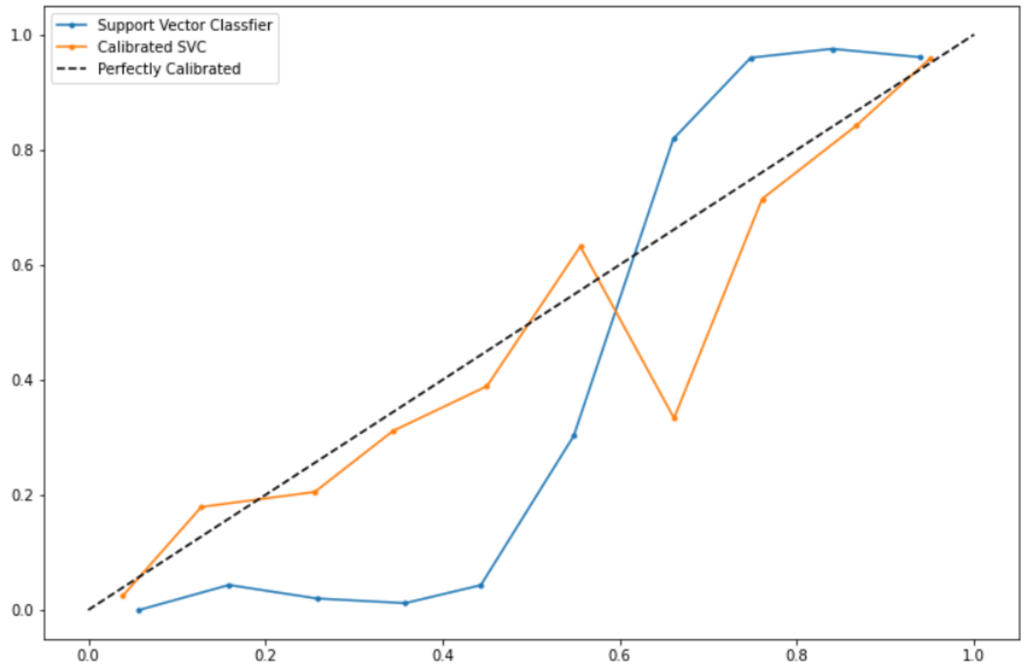
尋找校準曲線：

```
fop1, mpp1= calibration_curve(testy, preds1, n_bins=10, normalize=True)
fop2, mpp2= calibration_curve(testy, preds2, n_bins=10)
```

繪製曲線：

```
plt.figure(figsize=(12,8))
plt.plot(mpp1,fop1,marker='.')
plt.plot(mpp2,fop2,marker='.')
plt.plot([0,1],[0,1],linestyle='--',color='black')
plt.legend(("Support Vector Classfier","Calibrated SVC","Perfectly Calibrated"))
```

結果：



結論

因此，現在我們知道了等滲回歸的基礎知識（相信我還有很多其他知識）以及它與線性回歸的比較。 我們還看到了如何在python中使用它以及在哪裡應用它。 最後，我們知道了等滲回歸在擬合單調函數方面比線性回歸更有靈活性是有代價的，就是更多的數據。

我希望我能夠說明想要更深入地學習該演算法細節的任何人。

感謝閱讀！

尾注

PAVA演算法的動畫參考：

https://www.r-bloggers.com/2020/05/what-is-isotonic-regression/

請遵循以下說明以取得更多詳細資訊：

http://www.stat.umn.edu/geyer/8054/notes/isotonic.pdf

此外，你還可以從以下位置了解有關概率校準的更多資訊：

https://machinelearningmastery.com/calibrated-classification-model-in-scikit-learn/A

縮圖的積分：

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

文章分享自微信公眾號：

程式設計演算法 https python 網路安全

登錄 后參與評論

1評論

熱度 最新

使用者8786198 2021-06-30

請問等滲回歸屬於線性回歸還是非線性回歸呢?

回復 點讚 舉報

相關文章

人臉對齊介紹
2017-08-29 4.9K

機器學習概念總結筆記（一）
2021-08-03 2.7K

主流機器學習演算法簡介與其優缺點分析
2017-12-14 3.6K

邏輯回歸演算法學習與思考

機器學習 程式設計演算法

本文是作者對於邏輯回歸演算法的學習和思考，主要介紹：邏輯回歸的演算法介紹、邏輯回歸的數學原理、邏輯回歸的總結以及網路安全場景預測，歡迎大家參考討論。 邏輯回歸的演算法介紹 邏輯回歸...

企鵝號小編 2017-12-27 496 0

机器学习十大算法系列（一）——逻辑回归

机器学习

本系列博文整理了常见的机器学习算法，大部分数据问题都可以通过它们解决： 1.线性回归 (Linear Regression) 2.逻辑回归 (Logistic Regression) 3.决策树 (Decision Tree) 4.支持向量机（SVM） 5.朴素贝叶斯 (Naive Bayes) 6.K邻近算法（KNN） 7....

深度学习思考者 2018-01-03 626 0

关于Cewu Lu等的《Combining Sketch and Tone for Pencil Drawing Pro...

编程算法

相关论文的链接：Combining Sketch and Tone for Pencil Drawing Production 第一次看《Combining Sketch and Tone for Pencil Drawing Production》一文是在两年前，随意看了一下...

用户1138785 2018-01-03 941 0

sklearn API 文档 - 0.18 中文翻译

apache scikit-learn

所有函数和类的确切API，由docstrings给出。API会为所有预期类型和允许的功能，以及可用于算法的所有参数。 原文链接 : http://scikit-learn.org/stable/modules/classes.html 译文链接 : http://cwiki.apachecn.org/pages/viewpage.action?...

片刻 2018-01-05 2K 0

机器学习　人工智能　scikit-learn

中文文档: http://sklearn.apachecn.org/cn/0.19.0/modules/calibration.html 英文文档: http://sklearn.apachecn.org/en/0.19.0/modules/calibration.html 官方文档: http://scikit-...

片刻　2018-01-05　496　0

机器学习常见算法简介及其优缺点总结

机器学习　深度学习　人工智能　神经网络

机器学习常见算法的一种合理分类：生成/识别，参数/非参数，监督/无监督等。例如，Scikit-Learn文档页面通过学习机制对算法进行分组，产生类别如：1，广义线性模型，2，支持向量机，3，最近...

企鹅号小编　2018-01-11　780　0

【Scikit-Learn 中文文档】概率校准 - 监督学习 - 用户指南 | ApacheCN

机器学习　scikit-learn

1.16. 概率校准 执行分类时, 您经常希望不仅可以预测类标签, 还要获得相应标签的概率. 这个概率给你一些预测的信心. 一些模型可以给你贫乏的概率估计, 有些甚至不支持概率预测. 校准模块可以让您更...

片刻　2018-01-15　1.1K　0

转行数据挖掘和机器学习（四）

数据挖掘　机器学习

目前从纯数学专业转行到机器学习领域已经有两年半了，又到了该总结转行经验和个人成长的时候。笔者在公司里面已经做过智能推荐系统，智能安全系统和智能运维系统。除此之外，笔者对量子计算等...

企鹅号小编　2018-01-15　649　0

十种深度学习算法要点及代码解析

人工智能　深度学习　编程算法

谷歌董事长施密特曾说过：虽然谷歌的无人驾驶汽车和机器人受到了许多媒体关注，但是这家公司真正的未来在于机器学习，一种让计算机更聪明、更个性化的技术。 也许我们生活在人类历史上最关键...

企鹅号小编　2018-01-22　646　0

机器学习中常见4种学习方法、13种算法和27张速查表！

机器学习　神经网络　人工智能　深度学习　大数据

-免费加入AI技术专家社群>> 机器学习的算法很多。很多时候困惑人们都是，很多算法是一类算法，而有些算法又是从其他算法中延伸出来的。这里，我们从两个方面来给大家介绍，第一个方面是学习的方式，第二个方面是算法的分类。 一、4大主要学习方...

企鹅号小编　2018-01-25　15.3K　0

一文图解机器学习的基本算法！

机器学习　编程算法　人工智能

来源：软件定义世界 本文长度为2877字，建议阅读6分钟 本文为你分析如何选择机器学习的各种方法。 每当提到机器学习，大家总是被其中的各种各样的算法和方法搞晕，觉得无从下手。确实，机器学习的各种套路确实不少，但是如果掌握了正确的路径和...

数据派THU　2018-01-26　511　0

无人驾驶机器学习算法大全（决策矩阵、聚类、回归……）

人工智能　机器学习　编程算法

来源：机器人圈 作者：多啦A亮 本文长度为4600字，建议阅读6分钟 本文全面概述了无人驾驶现阶段使用的机器学习技术。 ［导读］无人驾驶被认为是未来人工智能技术应用的最大市场规模和影响力...

数据派THU　2018-01-29　2K　0

机器学习 编程算法 决策树

專 欄 ❉PytLab，Python 中文社区专栏作者。主要从事科学计算与高性能计算领域的应用，主要语言为Python，C，C++。熟悉数值算法(最优化方法，蒙特卡洛算法等）与并行化 算法（MPI,OpenMP…

Python中文社区 2018-02-01 1.2K 0

Kaggle机器学习实战总结

其他

專 欄 ❉王勇，Python中文社区专栏作者，目前感兴趣项目商业分析、Python、机器学习、Kaggle。17年项目管理，通信业干了11年项目经理管合同交付，制造业干了6年项目管理：PMO,变革，生产转移…

Python中文社区 2018-02-01 965 0

机器学习新手必看10大算法

机器学习 人工智能 编程算法 线性回归

选自TowardsDataScience 作者：James Le 机器之心编译 参与：程耀彤、路雪 本文介绍了机器学习新手需要了解的 10 大算法，包括线性回归、Logistic 回归、朴素贝叶斯、K 近邻算法等。 在机器学…

企鹅号小编 2018-02-02 474 0

机器学习算法一览（附python和R代码）

python r 语言 机器学习

“谷歌的无人车和机器人得到了很多关注，但我们真正的未来却在于能够使电脑变得更聪明，更人性化的技术，机器学习。 ”—— 埃里克 施密特（谷歌首席执行官） 当计算从大型计算机转移至个人电…

CDA数据分析师 2018-02-05 967 0

14种机器学习常见算法分类汇总！

机器学习 人工智能 神经网络 深度学习

机器学习无疑是当前数据分析领域的一个热点内容。很多人在平时的工作中都或多或少会用到机器学习的算法。这里总结一下常见的机器学习算法，以供您在工作和学习中参考。 机器学习的算法很多。…

CDA数据分析师 2018-02-05 1.7K 0

点击加载更多