بسمعه تعالی

پروژه درس ساختمان گسسته

تهیه کنندگان:مهدی مردانی و

احمدرضا یزدانی

الگوریتمk-means

وimage compression

مقدمه

روش‌ها و الگوریتم‌های متعددی برای تبدیل اشیاء به گروه‌های همشکل یا مشابه وجود دارد. الگوریتم-k میانگین یکی از ساده‌ترین و محبوب‌ترین الگوریتم‌هایی است که در «داده‌کاوی (Data Mining) «بخصوص در حوزه «یادگیری نظارت نشده « (Unsupervised Learning)به کار می‌رود.

معمولا در حالت چند متغیره، باید از ویژگی‌های مختلف اشیا به منظور طبقه‌بندی و خوشه کردن آن‌ها استفاده کرد. به این ترتیب با داده‌های چند بعدی سروکار داریم که معمولا به هر بعد از آن، ویژگی یا خصوصیت گفته می‌شود. با توجه به این موضوع، استفاده از توابع فاصله مختلف در این جا مطرح می‌شود. آنچه اهمیت دارد روشی برای اندازه‌گیری میزان شباهت یا عدم شباهت بین اشیاء است که باید در روش‌های خوشه‌بندی لحاظ شود.

در خوشه‌بندی-k میانگین از بهینه‌سازی یک تابع هدف (Object Function) استفاده می‌شود. پاسخ‌های حاصل از خوشه‌بندی در این روش، ممکن است به کمک کمینه‌سازی (Minimization)یا بیشینه‌سازی (Maximization) تابع هدف صورت گیرد. به این معنی که اگر ملاک «میزان فاصله Distance Measure))بین اشیاء باشد، تابع هدف براساس کمینه‌سازی خواهد بود پاسخ عملیات خوشه‌بندی، پیدا کردن خوشه‌هایی است که فاصله بین اشیاء هر خوشه کمینه باشد. در مقابل، اگر از تابع مشابهت (Dissimilarity Function) برای اندازه‌گیری مشابهت اشیاء استفاده شود، تابع هدف را طوری انتخاب می‌کنند که پاسخ خوشه‌بندی مقدار آن را در هر خوشه بیشینه کند.

خوشه‌بندی k- میانگین(K-means)

الگوریتم-k میانگین یکی از پرکاربردترین الگوریتم‌های خوشه‌بندی می‌باشد. حرف k که در این الگوریتم وجود دارد به این واقعیت برمی‌گردد که این الگوریتم به دنبال تعداد ثابتی از خوشه‌هاست که بر اساس نزدیکی نقاط داده‌ای به هم تعریف شده‌اند. این الگوریتم ابتدا در سال ۱۹۸۷توسط جی.بی مک کوئین به وجود آمد. برای درک ساده‌تر این الگوریتم یک مثال ساده دو بعدی زده می‌شود (x1, x2)ولی روند کار برای بیش از دو متغیر (, ...x1,x2) نیز همین‌گونه است.

مراحل k- میانگین(K-means)

مرحله یک:

در اولین مرحله ، الگوریتم به طور تصادفی k نقطه داده‌ای را انتخاب می‌کند تا مثل هسته روی آن‌ها کار شود.

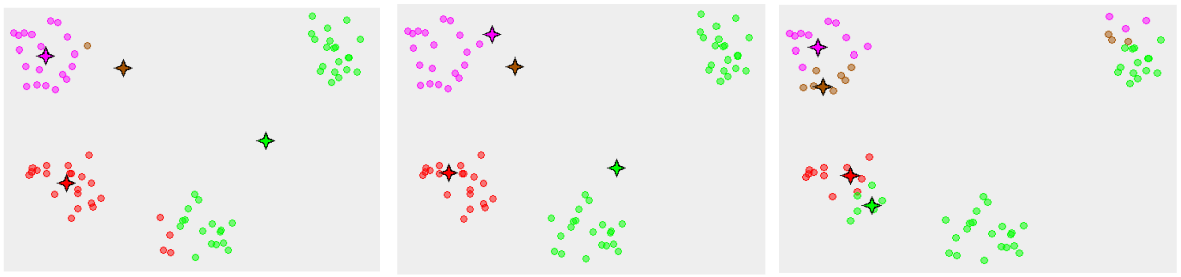
مرحله دوم:

در این مرحله هر داده را به نزدیک ترین هسته اختصاص می‌دهد. یک راه برای این‌کار یافتن مرز بین خوشه‌هاست. مرز بین دو خوشه به نقاطی گفته می‌شود که با هر خوشه فاصله یکسانی داشته باشند. (یادآوری: دو نقطه A و B را در نظر بگیرید، همه نقاطی که دارای فاصله مساوی با دو نقطه A و B باشند، بر روی یک خط قرار می‌گیرند که این عمود منصفِ خط واصل A و B می‌باشد.) مرز خوشه‌های حاصل هم با خط‌های پررنگ‌تر نشان داده شده است که با خطوط منقطع، زاویه قائمه دارند. اگر از این خطوط به عنوان راهنما استفاده شود مشخص است که کدام اطلاعات به کدام هسته‌ها نزدیک‌تر هستند. در سه بعد این مرزها صفحات صاف خواهند بود و در n بُعد، صفحات مرزی n-1 بُعدی هستند. خوشبختانه، الگوریتم‌های کامپیوتری به راحتی می‌توانند مسائل چند بُعدی را حل کنند. بر اساس هسته‌های اولیه، این داده به خوشه‌ای که توسط هسته شماره۲ کنترل می‌شود اختصاص یافته است، چرا که به آن هسته بیش از دو هسته دیگر نزدیک‌تر است.

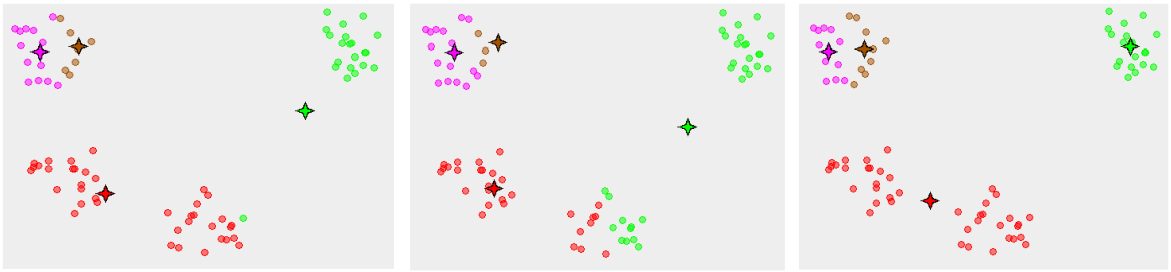
مرحله سوم:

در این مرحله مرکز ثقل هر خوشه محاسبه می‌شود. این مراکز ثقل بهتر از هسته‌های اولیه برای مشخص کردن خوشه‌ها، عمل می‌کنند. یافتن مرکز ثقل تنها با به‌دست آوردن میانگین اندازه هر بُعد برای همه اطلاعات در خوشه انجام می‌شود.

حال این مراکز برای تکرار بعدی الگوریتم، نقش هسته را بازی می‌کنند. مرحله دوم تکرار می‌شود و هر نقطه بار دیگر به یک خوشه با نزدیک‌ترین مرکز، اختصاص می‌یابد. این حلقه همین‌طور ادامه پیدا می‌کند تا دیگر هیچ تغییری در خوشه‌ها ایجاد نشود.



بخش مقدار دهی اولیه تکرار ۱ تکرار۲

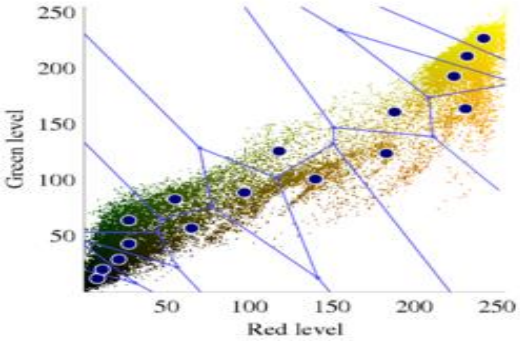


تکرار۵ تکرار۴ تکرار۳

کاربردها

از الگوریتم خوشه‌بندی-k میانگین در بخش‌بندی بازار کسب و کار (market Segmentation)، دسته‌بندی مشتریان(Customer Segmentation) ،بینایی رایانه‌ای (Computer Vision)و زمین‌آمار (Geostatistics) استفاده می شود. برای مثال در تشخیص تعداد رنگ و یا فشرده سازی تصاویر برحسب رنگ‌ها می‌توان از این الگوریتم‌ها استفاده کرد.

در تصویر بالا گل رز زرد رنگی دیده می‌شود که در یک محیط سبز قرار گرفته است. با استفاده از الگوریتم‌های خوشه‌بندی می‌توان تعداد رنگ‌ها را کاهش داده و از حجم تصاویر کاست. در تصویر زیر دسته بندی رنگ‌های گل رز دیده می‌شود.



در این تصویر، هر طیف رنگ براساس میزان رنگ قرمز و سبز، بوسیله سلول‌های ورونوی (Voronoi Cell)تقسیم‌بندی شده است. این تقسیم‌بندی می‌تواند توسط الگوریتم‌ها خوشه‌بندی-k میانگین صورت گرفته باشد. در کل تصویر نیز، طیف رنگ‌های مختلف برای تصویر گل رز در یک نمودار ورونوی (Voronoi diagram) نمایش داده شده است که خوشه‌ها را بیان می‌کند.

معایب و مزایای خوشه‌بندی-k میانگین

از آنجایی که در این روش خوشه‌بندی، محاسبه فاصله بین نقاط توسط تابع فاصله اقلیدسی انجام می‌شود، از این الگوریتم‌ها به صورت استاندارد، فقط برای مقدارهای عددی (و نه ویژگی‌های کیفی) می‌توان استفاده کرد. از طرف دیگر با توجه به محاسبات ساده و سریع آن‌ها،

پرکاربرد و موثر است. از طرف دیگر نسخه‌های تعمیم یافته از روش خوشه بندی-k میانگین نیز وجود دارد که با توابع فاصله دیگر مانند فاصله منهتن و یا فاصله‌هایی که برای داده‌های باینری قابل استفاده است، مراحل خوشه‌بندی را انجام می‌دهد.

به منظور ارزیابی نتایج خوشه‌بندی از معیارهای متفاوتی کمک گرفته می‌شود. ممکن است از قبل برچسب خوشه‌ها مشخص باشد و بخواهیم کارایی الگوریتم را با توجه به مقایسه برچسب‌های واقعی و حاصل از خوشه‌بندی، اندازه‌گیری کنیم. در این حالت، شاخص‌های ارزیابی بیرونی، بهترین راهنما و معیار برای سنجش صحت نتایج خوشه‌بندی محسوب می‌شوند. معمولا به این برچسب‌ها، استاندارد طلایی (Golden Standard) و در کل چنین عملی را ارزیابی Benchmarkمی‌گویند. برای مثال شاخص رَند (Rand Index) یکی از این معیارها و شاخص‌های بیرونی است.

از طرف دیگر اگر هیچ اطلاعات اولیه از ساختار و دسته‌بندی مشاهدات وجود نداشته باشد، فقط ملاک ارزیابی، می‌تواند اندازه‌هایی باشد که میزان شباهت درون خوشه‌ها و یا عدم شباهت یا فاصله بین خوشه‌ها را اندازه می‌گیرند. بنابراین برای انتخاب بهتر و موثرترین روش خوشه‌بندی از میزان شباهت درون خوشه‌ها و شباهت بین خوشه‌ها استفاده می‌شود. روشی که دارای میزان شباهت بین خوشه‌ای کم و شباهت درون خوشه‌ای زیاد باشد مناسب‌ترین روش خواهد بود. این معیارها را به نام **شاخص‌های ارزیابی درونی** می‌شناسیم. به عنوان مثال شاخص نیم‌رخ **(**silhouette**)** یکی از این معیارها است که شاخصی برای سنجش مناسب بودن تعلق هر مشاهده به خوشه‌اش ارائه می‌دهد. به این ترتیب معیاری برای اندازه‌گیری کارایی الگوریتم خوشه‌بندی بدست می‌آید.

تابع الگوریتم k-means بر روی عکس

```
def kmeans(the_data,k=3,q=20):
    import random
    ks = []
    for i in range(k):
        ks +=
[[random.randint(0,255),random.randint(0,255),random.
randint(0,255)]]
    kpixels = {}
    for qqq in range(q):
        for i in range(k):
            kpixels[i] = []
        for i in the_data:
            dist = []
            for j in range(k):
                dist += [int(((i[0] - ks[j][0])**2 +
(i[1] - ks[j][1])**2 + (i[2] - ks[j][2])**2 )** 0.5)]
            minn = 0
            for a in range(len(dist)):
                if dist[a] < dist[minn]:
                    minn = a
            kpixels[minn] += [i]
        for i in range(k):
            sum0 = 0
            sum1 = 0
            sum2 = 0
            nums = len(kpixels[i])
            for j in range(nums):
                sum0 += kpixels[i][j][0]
                sum1 += kpixels[i][j][1]
                sum2 += kpixels[i][j][2]
            if nums != 0:
                ks[i][0] = sum0 // nums
                ks[i][1] = sum1 // nums
                ks[i][2] = sum2 // nums
    new_data = []
    for i in range(len(the_data)):
        for j in range(k):
            if the_data[i] in kpixels[j]:
                new_data += [ks[j]]
                break
    return new_data
```

تابع نهایی الگوریتم k-means

```
from PIL import Image
def kmeans(the_data,k=3,q=20):
    import random
    ks = []
    for i in range(k):
        ks += [[random.randint(0,255),random.randint(0,255),ran-
dom.randint(0,255)]]
    kpixels = {}
    for qqq in range(q):
        for i in range(k):
            kpixels[i] = []
        for i in the_data:
            dist = []
            for j in range(k):
                dist += [int(((i[0] - ks[j][0])**2 + (i[1] -
ks[j][1])**2 + (i[2] - ks[j][2])**2 )** 0.5)]
            minn = 0
            for a in range(len(dist)):
                if dist[a] < dist[minn]:
                    minn = a
            kpixels[minn] += [i]
        for i in range(k):
            sum0 = 0
            sum1 = 0
            sum2 = 0
            nums = len(kpixels[i])
            for j in range(nums):
                sum0 += kpixels[i][j][0]
                sum1 += kpixels[i][j][1]
                sum2 += kpixels[i][j][2]
            if nums != 0:
                ks[i][0] = sum0 // nums
                ks[i][1] = sum1 // nums
                ks[i][2] = sum2 // nums
    new_data = []
    for i in range(len(the_data)):
        for j in range(k):
            if the_data[i] in kpixels[j]:
                new_data += [tuple(ks[j])]
                break
    return new_data

name = input("enter your image name  ")
colors = int(input("how many colors do you want? "))
img = Image.open(name)
img = img.convert("RGB")
f_data = img.getdata()
the_data = []
for i in f_data:
    the_data += [i]
new_data = kmeans(the_data,colors)
img.putdata(new_data)
img.save("new_image.jpg")
```

نمونه عملکرد تابع:

قبل از اجرای الگوریتم:

بعد از اجرای الگوریتم:

Sources

1.https://avalform.com/%D8%A7%D9%84%DA%AF%D9%88%D8%B1%DB%8C%D8%AA%D9%85-k-means-%D8%A2%D9%85%D9%88%D8%B2%D8%B4-%D8%AA%D8%B5%D9%88%DB%8C%D8%B1%DB%8C/.

2.https://blog.faradars.org/k-means-clustering-algorithm/.