به نام خدا

کارگاه علم داده با پایتون پیشرفته

جلسه نهم:کشف داده های پرت(Outlier detection)

مدرس :

مهرناز جلیلی

دانشجو کارشناسی ارشد علم داده ها

دانشگاه شهید بهشتی

داده‌های پَرت (OUTLIERS)

داده‌های پَرت همه‌جا هستند. آن‌ها به دلایل مختلفی تولید می‌شوند و معمولاً در میان انواع داده‌ها دیده می‌شوند. این نوع داده‌ها را که معمولاً غیرعادی هستند و از الگوهای عمومی در یک مجموعه‌ی داده پیروی نمی‌کنند، می‌توان توسط الگوریتم‌های مختلفِ تشخیصِ داده‌های پرت شناسایی کرد. با شناسایی داده‌های پَرت می‌توان آن‌ها را از مجموعه‌ی داده کنار گذاشت تا مجموعه‌ی داده، کمی تمیزتر و مناسب‌تر جهت تزریق به الگوریتم‌هایی مانند طبقه‌بندی و خوشه‌بندی باشد.

. البته در برخی از مواقع خودِ داده‌های پَرت هستند که صورت مسئله می‌باشند. مثلاً در بین بیماران و علائمِ آن‌ها ممکن است به دنبال بیمارانی بگردیم که علائمشان با دیگر بیماران همخوانی ندارد و به نوعی در آن مجموعه‌ی داده، غیر طبیعی هستند.

فرض کنید می‌گویند میانگین حقوق در یک شرکت ۴میلیون تومان است. آیا این بدان معنی است که اکثر افراد حاضر در آن شرکت ۴میلیون تومان (یا نزدیک به آن) حقوق می‌گیرند. در نگاه یک غیرمتخصص بلی ولی در نگاه یک متخصص آمار و داده‌کاوی قطعا جواب خیر است. ممکن است ۹۵درصد افراد حاضر در آن شرکت حقوق ۱میلیون تومان بگیرند و ۵درصد بقیه حوقشان ۲۰میلیون تومان باشد. در واقع این ۵درصد نوعی داده پرت هستند که میانگین Mean را به نفع خود جا به جا کرده‌اند!

تشخیص داده‌های پرت و دارای نویز (Noise)

داده‌های پرت و داده‌هایی که دارای نویز (noise) هستند، در بسیاری از مجموعه‌ها، دیده می‌شوند. فرض کنید شما مدیرِ یک وب‌سایتِ فروشگاهی هستید و می‌خواهید سنِ کاربران خود را تحلیل کنید. مثلاً این‌که افراد در بازه‌ی سنیِ مختلف، بیشتر به کدام محصولات تمایل نشان می‌دهند. برای این‌کار در هنگام خرید، سنِ خریدار را از او دریافت می‌کنید. آیا مطمئن هستید که افراد معمولاً سنِ خود را در بازه‌ی ۰ تا ۱۰۰ سال وارد می‌کنند؟ برای مثال شخصی ممکن است سهواً سنِ خود را به جای **۲۵ سال**، **۲۵۰ سال** درج کند و یا شخصی به جای این‌که سن خود را درج کند، سهواً سالِ تولد خود را وارد نماید! به این دست از داده‌ها که معمولاً با بقیه‌ی داده‌ها ناسازگار هستند داده‌های پرت (outliers) می‌گویند و مجموعه‌ی داده را دارای نویز (noise)می‌دانند.

اما چرا بایستی داده‌های پرت را مورد بررسی قرار دهیم؟

فرض کنید مجموعه‌ی داده‌ای از بیمارانِ مختلف دارید. این مجموعه‌ی داده می‌تواند شاملِ *ویژگی‌ها (ابعاد)* مختلف باشد. مثلا *سنِ شخص، تعداد دفعات مراجعه به بیمارستان در سال گذشته، سابقه‌ی بیماری مشابه در والدین* و... حال فرض کنید تعداد ۱۰۰هزار بیمار دارید که برای هر کدام از آن‌ها این اطلاعات را جمع‌آوری کرده‌اید. یک متخصص با کمک این اطلاعات احتمالاً می‌تواند روند بیماری‌ها و الگوهای مشخص را تشخیص دهد. اما ممکن است برخی از افراد، از الگوها و یا گروه‌های خاصی تبعیت نکنند. برای مثال، ممکن است برخی از افرادْ بیماری‌های خاصی داشته باشند که هنوز توسط متخصص به عنوان یک الگو درک نشده باشد. حتی ممکن است برخی افراد در بعضی از ویژگی‌ها به عنوان داده‌ی پَرت در نظر گرفته شوند. برای مثال یک شخص در بین مجموعه‌ی کلیِ داده عادی باشد ولی در بینِ هم سن و سال‌های خود به عنوان نمونه‌ای پَرت در نظر گرفته شود. مثلاً ممکن است فردی که ۱۲ سال دارد با میزان قندِ خونِ ۱۰۰، در میانِ تمامِ افرادِ مجموعه، طبیعی به نظر برسد ولی در میانِ افرادی در بازه‌ی سنیِ ۱۰ تا ۱۵ سال (هم سن و سال‌های خودش)، به عنوانِ یک داده‌ی پَرت و غیرعادی باشد (البته این صرفاً یک مثال بود و پایه‌ی پزشکی نداشت). پس به دست آوردنِ داده‌های پَرت در حوزه‌ای مانندِ پزشکی نیز به این صورت می‌تواند کمک کننده باشد.

مثالِ دیگری که می‌تواند به خوبی بیان‌گرِ کاربرد این حوزه باشد، تشخیص دزدیده شدنِ کارت‌های بانکی است. فرض کنید یک کارتِ بانکی دارید و به صورت معمول و عادی از این کارت استفاده‌هایی می‌کنید. مثلاً حقوقِ ماهیانه‌ی شما به این کارت واریز می‌شود و شما در طولِ ماه آرام آرام آن مبالغ را توسط دستگاه‌های POSدریافت کرده و یا به صورت اینترنتی از فروشگاه‌های مشخص خرید می‌کنید. حال کارتِ شما دزدیده می‌شود و این شخص سریعاً به محلِ دیگری رفته و با دانستن رمزِ کارت، سریعاً درخواست دریافتِ مبلغی نامتعارف را از یک دستگاهِ POSدر یک زمان نامتعارف انجام می‌دهد. این کار یک عملِ غیر طبیعی (برای کارتِ شما) است، و اگر یک الگوریتمِ تشخیص داده‌های (یا همان فرآیندهای) پَرت در شبکه‌ی شتاب موجود باشد، احتمالاً می‌تواند این عملیات را شناسایی کرده و کارتِ بانکی را به عنوان دزدیده شده ضبط نماید (و یا درخواست رمزی مانند رمزِ دوم انجام شود).

در حوزه‌هایی مانندِ ورزشِ فوتبال نیز می‌توان از داده‌کاوی و فرآیندهای تشخیصِ داده‌های پَرت استفاده کرد. برای مثال از طریقِ سنسورهایی که به بازیکنان متصل است و با کمکِ تحلیلِ آن‌ها در شرایطِ مختلف، می‌توان بازیکنانی که توانایی‌های بالاتری (با توجه به شرایط) دارند را کشف کرد. برای مثال، برخی از بازیکنان در شرایطِ جویِ بارانی، عملکردِ بهتری از خود به نمایش می‌گذارند و در واقع به عنوان یک داده‌ی پَرت، از سایر بازیکنان جدا شده و شناسایی می‌شوند.

این‌ها نمونه‌هایی از کاربردهای مختلفِ تشخیصِ داده‌های پَرت بود. همان‌طور که متوجه شدید، داده‌های پَرت لزوماً یک عنصر نامطلوب نیستند و در بسیاری از مواقع ما به دنبال داده‌های پَرت می‌گردیم تا از آن‌ها استفاده کنیم.

نویزها که به داده‌های غیرطبیعی (anomalies) نیز شهرت دارند، باعث خراب شدنِ آمارها و داده‌های مجموعه‌ی داده می‌شوند. برای مثال فرض کنید سنِ افرادِ مختلف از آن‌ها دریافت کرده و در جدولی مانندِ جدولِ زیر قرار داده‌اید:

کاربر	سن
#1	17
#2	15
#3	25
#4	250
#5	71
#6	62
#7	33
#8	44
#9	1363

به سادگی می‌توان تشخیص داد که این مجموعه‌ی داده برای مقدارِ سن دارای داده‌های پرت است.

روش‌های حذف داده‌های دارای نویز زیاد است و در این درس به چند روشِ ساده و کاربردی در شناسایی و حذف نویز خواهیم پرداخت. یکی از این روش‌ها حذفِ مقادیر بالا و پایین داده‌ها به تعداد مشخص است. برای مثال در همین جدول بالا، می‌توانیم مقادیری که کمتر از ۱۰ و یا بیش‌تر از ۱۰۰ هستند را حذف کنیم و یا مقادیری که در بازه‌ی بین ۱۰ تا ۱۰۰ قرار ندارد را با میانگینِ سن‌های باقی‌مانده جایگزین کنیم. با این‌کار داده‌ها در یک بازه‌ی مشخص و معقول قرار می‌گیرند. پس در مثال بالا، می‌توانیم کاربران ۴ و ۹ را حذف کنیم و یا مقدار سن را برای آن‌ها برابر ۳۸ که میانگین سن‌های باقی‌مانده افراد است، قرار می‌دهیم.

البته در بعضی از مواقع ما به دنبال پیدا کردنِ نویزها هستیم تا داده‌ها را با توجه به مقادیرِ غیرطبیعی (anomalies)تحلیل کنیم. مثلاً می‌خواهیم در یک سری تراکنش‌های بانکی، آن دسته از تراکنش‌هایی که رفتارِ غیرِ عادی داشتند را کشف کرده و به تخلف‌های یک فرد در بانک رسیدگی کنیم.در این موارد از الگوریتم DBSCAN استفاده می کنیم که یکی از الگوریتم‌هایی است که می‌تواند داده‌های پَرت را تشخیص دهد. در واقع DBSCANرا هم می‌توان برای خوشه‌بندی مورد استفاده قرار داد و هم می‌توان از آن به عنوانِ یک الگوریتمْ جهتِ تشخیص داده‌های پرت استفاده کرد. همچنین روشی به عنوان **SVMتک کلاسه (one class SVM)**موجود است که می‌تواند داده‌های پرت را تشخیص دهد.

اما ممکن است در بعضی از مواقع، کلاً یک ویژگی (یک بُعد) پَرت باشد. جدول زیر را در نظر بگیرید

	معدل کل	تعداد مقالات	مدرک IELTS زبان	سنوات تحصیل	دکتری قبول شده؟
#1	19,5	3	1	3	بلی
#2	16,5	0	1	4	خیر
#3	15	0	0	3	خیر
#4	17	2	1	2,5	بلی
#5	18,5	2	0	2,5	بلی
#6	15,5	1	1	2,5	خیر
#7	19	3	1	3	بلی

در این مجموعه‌ی داده، یک رئیس دانشکده می‌خواهد بر اساسِ ویژگی‌های دانشجویان و سابقه‌ی دانشجویانِ گذشته، به این نتیجه برسد که کدام یک از دانشجوهای جدید، می‌توانند در آزمون دکتر قبول شوند. همان‌طور که مشاهده می‌کنید، ویژگی‌هایی مانندِ **معدلِ کل، تعداد مقالات، مدرک زبان ELTاو سنوات تحصیلی** در تشخیص و ساختِ مدل جهت پیش‌بینی داده‌های آینده، کاربرد دارند. حال فرض کنید یک ویژگیِ دیگر مانندِ **جنسیت** به ویژگی‌های بالا اضافه شده باشد. آیا این ویژگی می‌تواند در تشخیصِ این‌که شخصی دکتری قبول شود یا خیر موثر باشد؟ فرض کنیم جواب **منفیست**، یعنی **ویژگیِ جنسیت با توجه به معیارهای آماری و از روی دانشجویان گذشته، تاثیری بر قبول شدن یا نشدنِ افراد در مقطع دکتری ندارد**. پس این ویژگی یک ویژگیِ نویز به حساب می‌آید. یعنی برخی اوقات یک ویژگی یا بُعد نیز می‌تواند نویز باشد به این صورت که در تصمیم‌گیریِ نهایی تاثیرِ چندانی نداشته باشد. روش‌های تشخیص ویژگی‌های نویز بسیار هستند. یکی از آن‌ها که در دوره‌ی جبرخطی در موردِ آن صحبت کردیم، الگوریتمِ **PCA**است که ویژگی‌هایی با تاثیرِ کم را از میان ویژگی‌های خود -تقریبا- حذف می‌کند. روش‌های دیگری مانند **chi2**نیز وجود دارند که قادر به تشخیصِ ویژگی‌های کم‌اهمیت هستند.

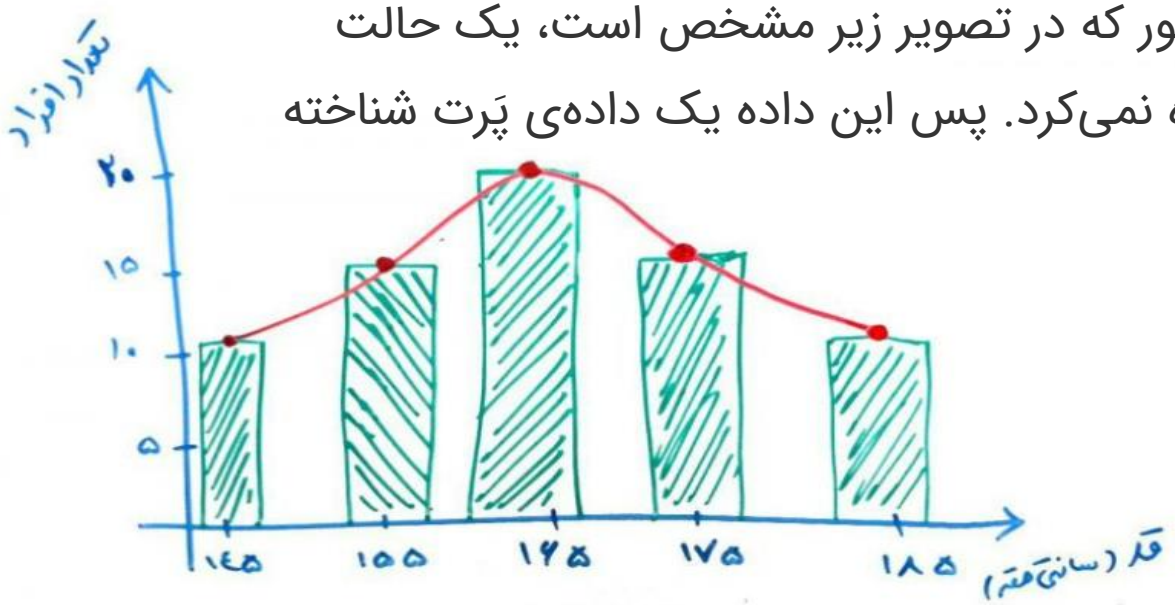
تست‌های آماری (Statistical Test)جهت تشخیص داده‌های پرت

تست‌های آماری یا همان Statistical Testsنیز برای تشخیص داده ی پرت هستند. آن‌ها یک فرض بر روی داده‌ها دارند و همچنین انتظار دارند که داده‌ها از یک **توزیع احتمالی (Probability Distribution)** پیروی کنند و هر داده ای از این توزیع احتمالی پیروی نکرد، داده‌ی پَرت شناخته می‌شود.
برای فهمِ بهتر، یکی از توزیع‌ها به نامِ **توزیع گوسی یا همان توزیع نرمال** را شرح می‌دهیم.

در آمار و احتمالات، وقتی یک مجموعه‌ی داده در اختیار دارید، بعضاً فرض بر این است که این مجموعه‌ی داده از یک توزیعِ آماری پیروی می‌کند. مثلا فرض کنید در یک کلاسِ ۴۰نفره هستید، که هر کدام از دانشجویانِ این کلاس، یک قدِ مشخص (به سانتی‌متر) دارند. نمودار زیر نشان می‌دهد که از نظرِ قد در بازه‌های مختلف، چند نفر (تعداد) وجود دارند.

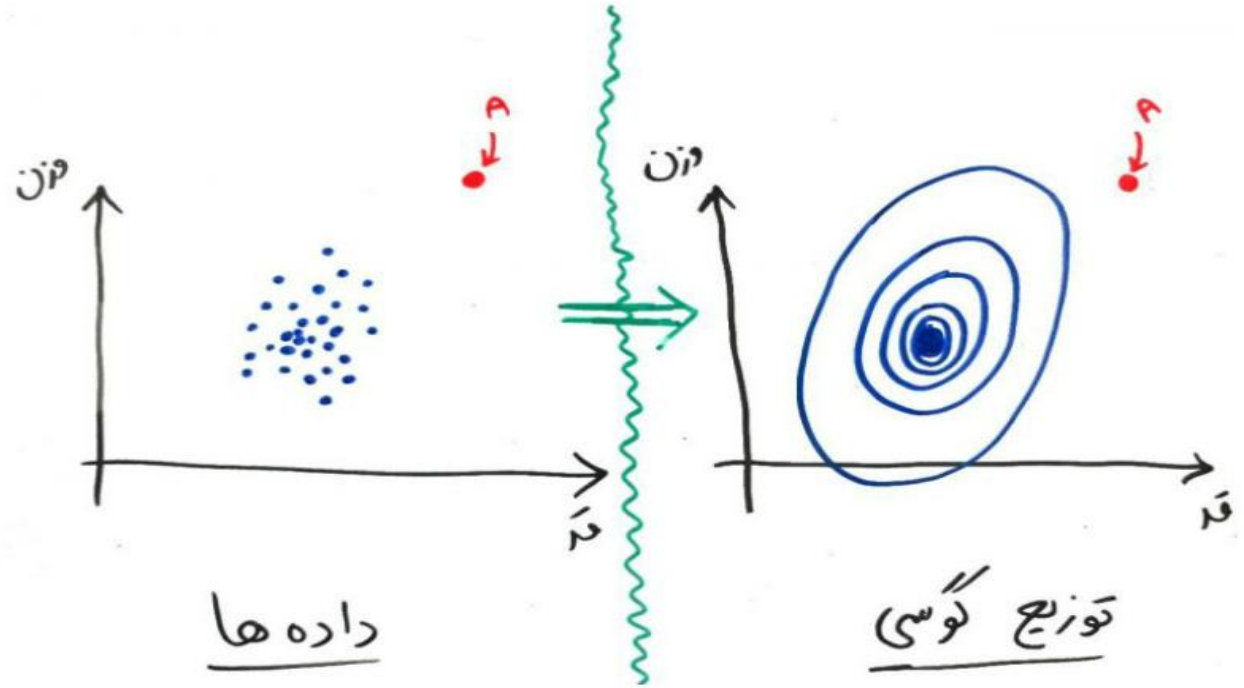
شکل زیر را نگاه کنید:

برای مثال، تعدادِ ۲۰نفر، قدی در محدوده‌ی ۱۶۵سانتی‌متر دارند و به همین صورت برای بقیه‌ی قدها می‌توانید تعداد مشخص را مشاهده کنید. این یک نوع توزیع گوسی (Gaussian Distribution) است. به این معنی که یک عدد مانند ۱۶۵وجود دارد که بیشترین تعداد از آن قد در بین داده‌های ما موجود است و هر چه از این قدِ ۱۶۵سانتی‌متری فاصله بگیریم، تعدادِ افراد در بازه‌های دیگر کم و کم‌تر می‌شود (تا جایی که به صفر برسد-مثلاً تعدادِ افرادی که قد ۳۰۰سانتی‌متر یا ۱۰ سانتی‌متر داشته باشند **صفر** است). اگر توزیعِ گوسی را در نظر داشته باشیم، این توزیع انتظار دارد که داده‌های موجود در مجموعه‌ی داده، از این قانون پیروی کنند. دوباره شکلِ بالا را نگاه کنید، اگر شخصی (داده‌ای) وجود داشت که مثلاً ۲۳۰سانتی‌متر بود، همان‌طور که در تصویر زیر مشخص است، یک حالت غیرِطبیعی به وجود می‌آمد و در واقع انتظارِ توزیعِ گوسی را برآورده نمی‌کرد. پس این داده یک داده‌ی پَرت شناخته می‌شد.



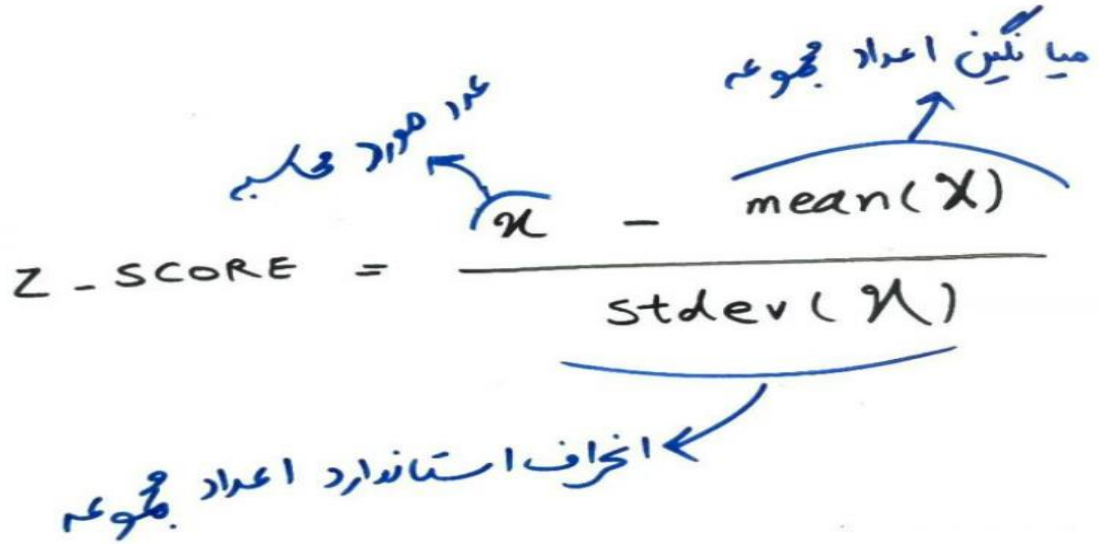
البته توجه داشته باشید که توزیع گوسی (Gussian) فقط یک نوع-و شاید معروف‌ترین نوع- توزیع‌های احتمالی باشد. توزیع‌های احتمالیِ بسیار زیادِ دیگری هم وجود دارند که در دوره‌ای جدا به آن‌ها خواهیم پرداخت.

حالا احتمالاً متوجه شدید که در تست‌های آماری نیز به همین صورت رفتار می‌شود. این تست‌ها **ابتدا یک فرض برای یک توزیع احتمالی در نظر می‌گیرد و بعد از آن، داده‌هایی را که از این فرض تبعیت نکنند، به عنوانِ یک داده‌ی پَرت حساب می‌کنند**. برای روشن‌تر شدن موضوع، تصویر زیر را مشاهده کنید:

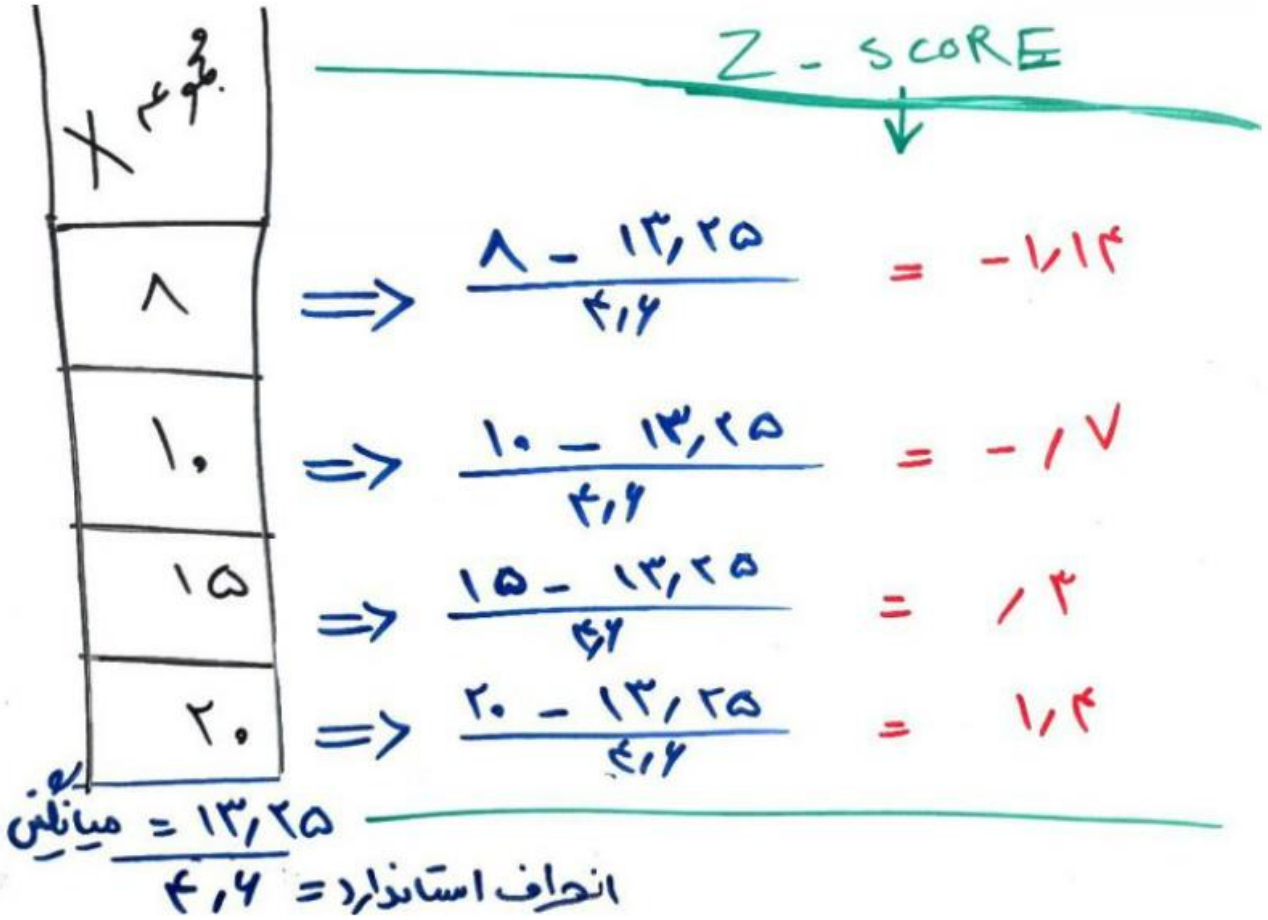


محاسبه‌ی داده‌های پَرت با استفاده از z-score

برای این‌که z-scoreرا برای یک عدد در یک مجموعه محاسبه کنیم، باید آن عدد را منهای میانگینِ آن مجموعه کرده و سپس بر انحراف استاندار تقسیم کنیم. در واقع z-scoreباعث می‌شود که هر کدام از عناصرِ مجموعه‌ی داده، به یک عددِ دیگر تبدیل شوند که میانگینِ آن اعدادِ تبدیل شده صفر (۰) و انحراف استاندارد آن‌ها یک (۱) است.
اجازه بدهید دوباره بگوییم: **اعدادِ مجموعه‌ی قبلی به اعدادی تبدیل می‌شوند که میانگینِ آن‌ها ۰ است و انحرافِ معیارِ آن ۱.**

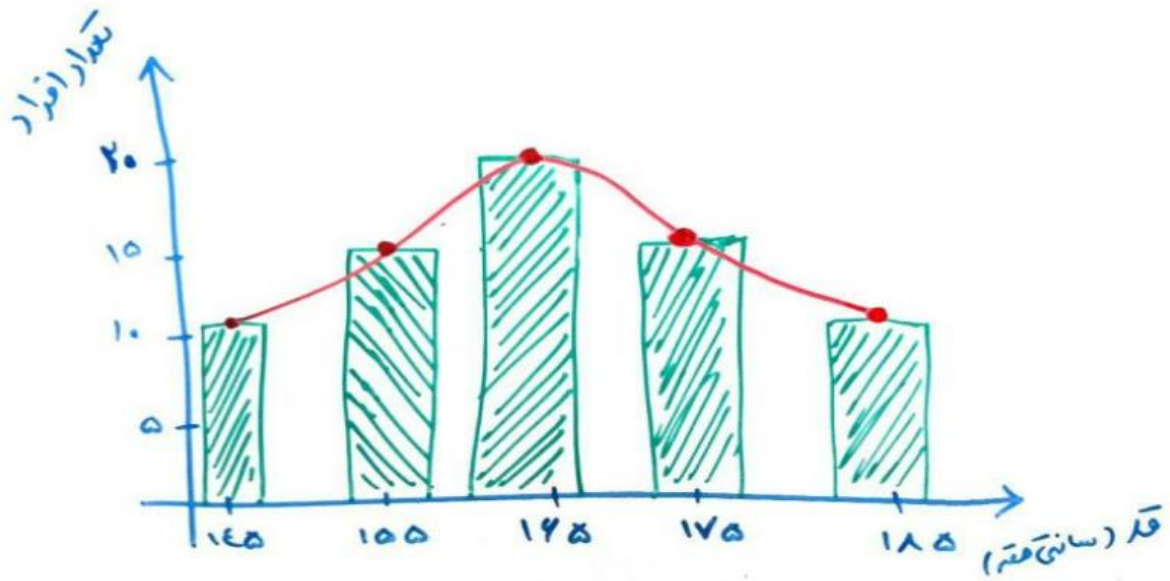


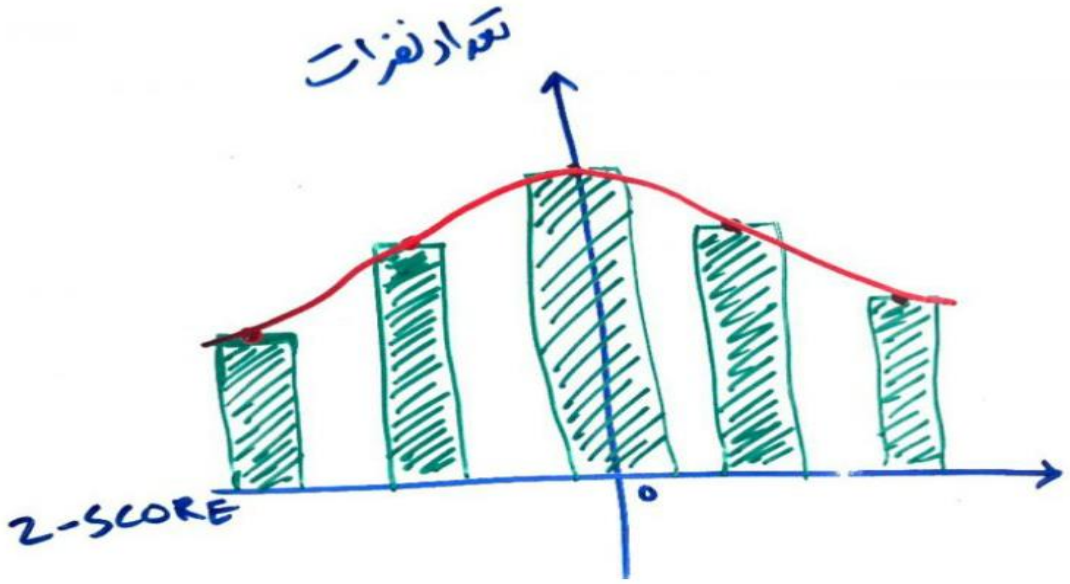
برای درکِ بهتر، شکل زیر که شامل یک مجموعه هست را در نظر بگیرید:



حال چگونه داده‌های پَرت را با استفاده از خروجیِ z-scoreمحاسبه کنیم؟ در اسلاید قبل گفتیم که مدل‌های تستِ آماری یک فرض در موردِ داده‌ها دارند. z-scoreهم فرض می‌کند که داده‌ها یک توزیع گوسی دارند. z-scoreبا تبدیلِ داده‌ها و فرضِ این‌که داده‌ها یک توزیع گوسی یا همان نرمال با میانگینِ ۰ و انحرافِ استانداردِ ۱ دارند، آن‌ها را می‌شناسد. باز هم مثالِ اسلایدِ قبل را این‌جا می‌آوریم.

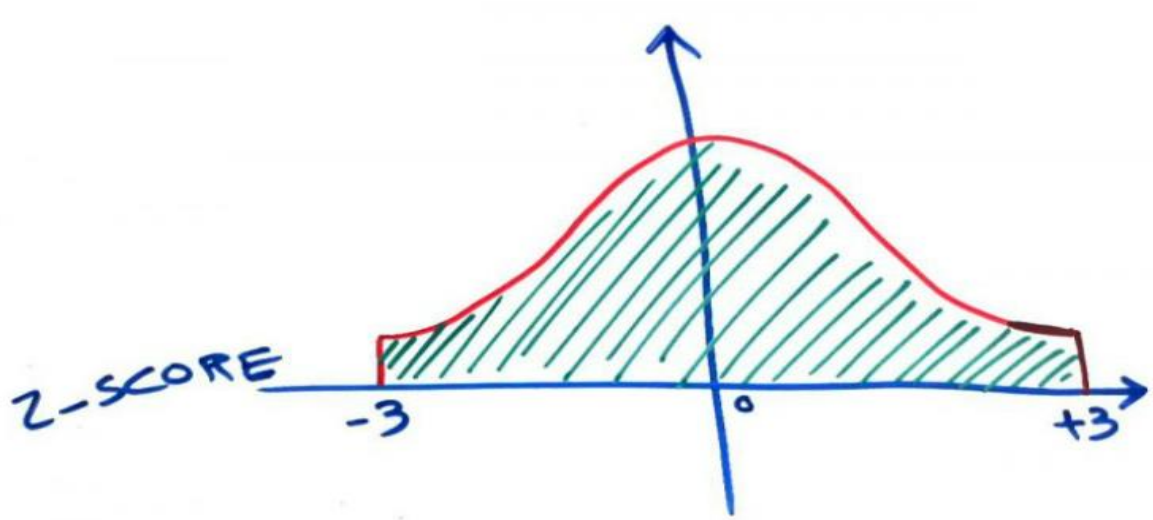
برای مثال، تعدادِ ۲۰نفر، قدی در محدوده‌ی ۱۶۵سانتی‌متر دارند و به همین صورت برای بقیه‌ی قدها می‌توانید تعداد مشخص را مشاهده کنید. این یک نوع توزیع گوسی(Gaussian Distribution) است. اگر اعدادِ مجموعه‌ی ۴۰نفره‌ی کلاس را با z-scoreبه بازه‌ای دیگر تغییر دهیم چیزی مانندِ شکل زیر می‌شود:





همان‌طور که می‌بینید میانگینِ ۰ و انحراف استانداردِ **۱** در این نمودار مشخص است. حال برای این‌که داده‌های پَرت را تشخیص بدهیم می‌توانیم از مجموعه‌ی داده، آن‌هایی که امتیازِ z-scoreآن‌ها بیشتر از ۳ و کمتر از ۳- باشد را از بینِ داده‌ها حذف کنیم. معمولاً برای تشخیص داده‌های پَرت از طریق z-scoreعددِ ۳ و ۳- یا چیزی در همین بازه را قرار می‌دهند (که این نیز پایه‌ی آماری در توزیعِ گوسی دارد). مثلا اگر قدِ شخصی ۲۵۰ بود، احتمالاً با تبدیل z-scoreاین عدد به **عددی مانندِ ۴** تبدیل می‌شود و چون بزرگ‌تر از ۳ بود، شخص با قدِ ۲۵۰سانتی‌متر از بین داده‌ها حذف می‌شد. اگر بخواهیم از روی شکل توضیح دهیم، یک سری داده‌ها که در گوشه‌ی توزیع گوسی قرار می‌گیرند، حذف می‌شوند:

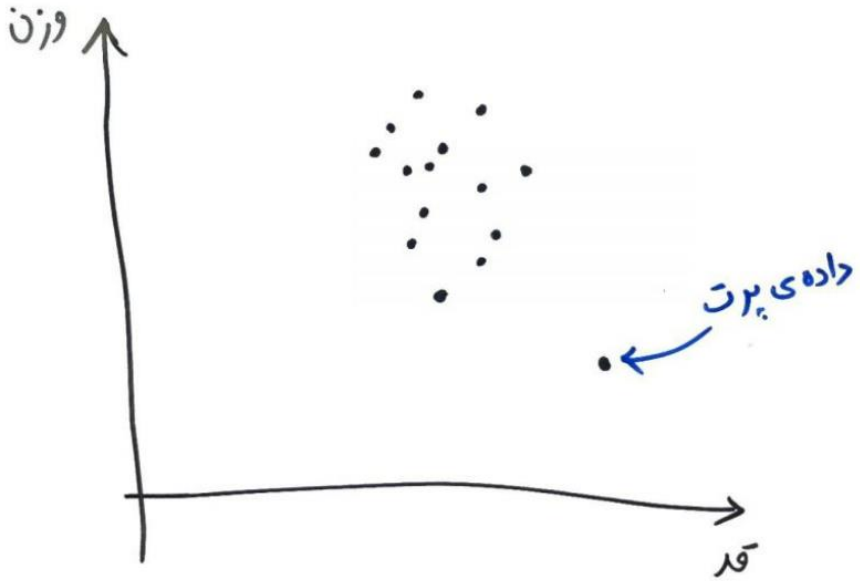
در تصویرِ زیر مشاهده می‌کنید که اعدادِ در بازه‌ی ۳- و ۳+ نگهداشته شده اند و آن‌هایی که بیشتر یا کمتر از این بازه بوده‌اند، از بین رفته‌اند. به این ترتیب z-scoreمی‌تواند داده‌های پَرت یا همان outliersرا شناسایی و حذف کند.



الگوریتم جنگل ایزوله (Isolation Forest) جهت تشخیص داده‌های پرت

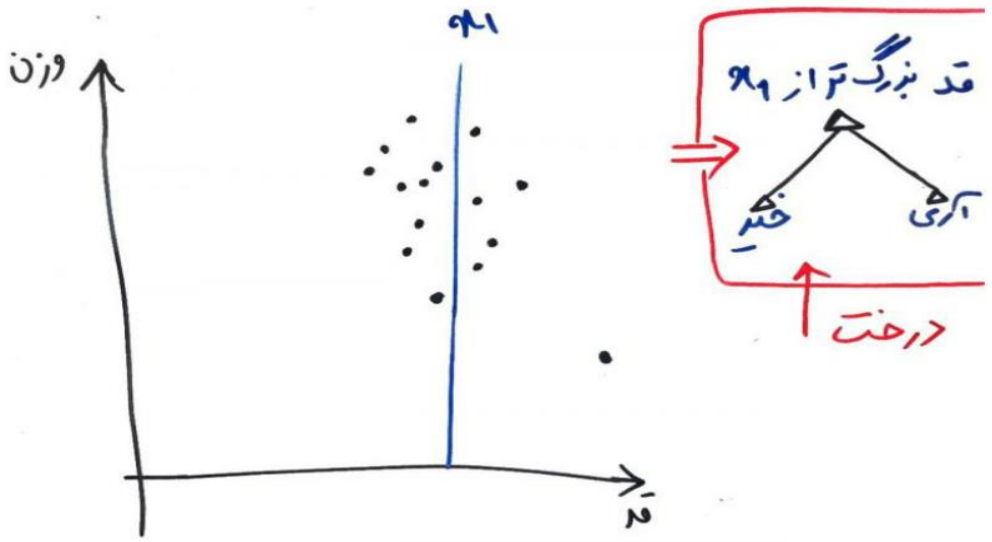
ین الگوریتم که برای به دست آوردنِ داده‌های پَرت به وجود آمده است می‌تواند داده‌هایی را که از داده‌های دیگر جدا (و تنها) هستند شناسایی کرده و به عنوان داده‌های پَرت (Outliers) علامت بزنند.

برای شروع، شکل زیر را نگاه کنید (این شکل، داده‌های افراد مختلف است که بر اساس قد و وزن بر روی یک تصویرِ دو بُعدی نگاشت شده اند)

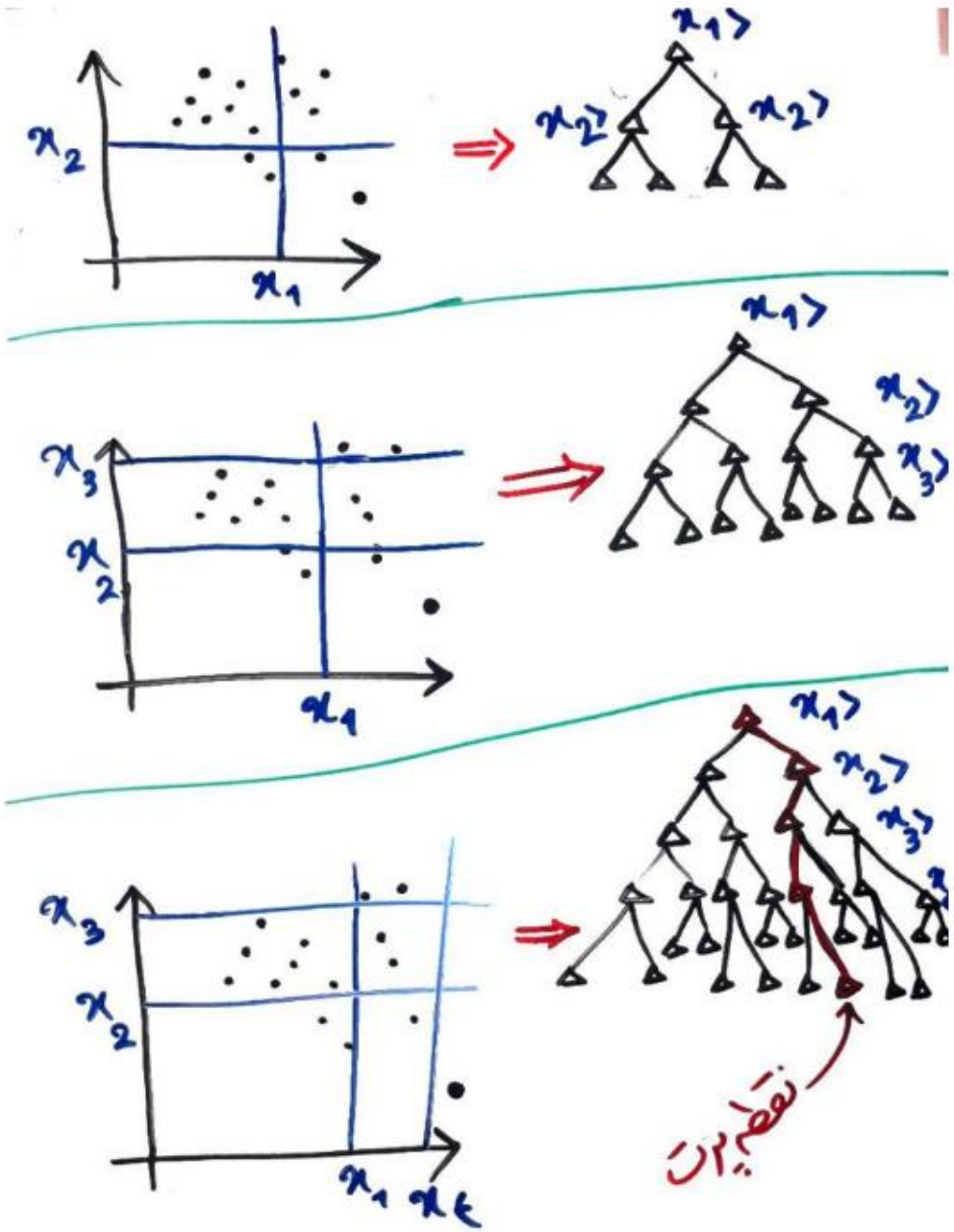


الگوریتمِ جنگلِ ایزوله (Isolation Forest) یک ویژگی (یک بُعد) را به صورت تصادفی انتخاب می‌کند و سپس یک مقدارِ تصادفی بین کمینه (Minimum) و بیشینه‌ی (Maximum) آن انتخاب کرده و با یک خطِ جداساز آن بُعد را جدا می‌کند. چیزی مانندِ شکل زیر:

همان‌طور که می‌بینید این جداساز به صورتِ تصادفی، ویژگیِ قد را انتخاب کرده و با یک خطِ تصادفی (در اینجا خط (x1، این ویژگی را به دو قسمت تبدیل کرد. با این کار می‌توان یک درختِ دودویی ساخت که برای جداکردنِ داده‌ها مطابقِ خطِ x1کار می‌کند. درخت ایجاد شده‌ی فرضی را در سمت راست تصویر می‌بینید. این درخت در ریشه، داده‌ها را به دو قسمت تقسیم می‌کند، داده‌هایی که از نظرِ قد بزرگ‌تر از x1هستند و داده‌هایی که از نظرِ ویژگیِ قد کوچک‌تر از x1هستند.



حال دوباره الگوریتم، یک ویژگی (بُعد) تصادفی را انتخاب می‌کند.
و دوباره خطی برای جداسازیِ آن ویژگی به صورت تصادفی می‌کشد.
در شکلِ زیر ۳بار این‌کار را انجام دادیم تا درخت کمی توسعه یابد:



در شکل‌های بالا هر بار، یک ویژگی به صورتِ تصادفی انتخاب شد و آن ویژگی با یک خطِ جداکننده، به قسمت‌های مختلفی تقسیم شد تا درختِ نظیر به صورت درخت‌های سمت راستِ آن تولید شود. در آخرین شکل مشاهده می‌کنید که نقطه‌ی پرت (پایین سمت چپ) به صورت یک نقطه‌ی جدا (ایزوله) تنها مانده است. در واقع ما آنقدر این تقسیم‌بندی را انجام دادیم تا بلاخره یک نقطه، تنهای تنها، در یکی از محوطه‌ها پیدا شد. درختِ متناظرِ سمت راست را نگاه کنید. این درخت درباره‌ی نقطه‌ی پرت به پایان (برگ) رسیده است و دیگر نمی‌توان آن را ادامه داد. البته یک درخت را می‌توان اینقدر ادامه داد که تمامی نقاط بلاخره در یک محوطه‌ی ایزوله (تنها) شوند. این نقطه‌ی پرت همان‌طور که می‌بینید از x1بزرگ‌تر، از x2کوچکتر، از x3کوچکتر و از x4بزرگ‌تر است (با خطِ قرمز رنگ بر روی درختِ آخر نمایان است). در واقع ما این‌قدر ویژگیِ تصادفی انتخاب و تقسیم می‌کنیم که نقاط به صورتِ تنها در یک خانه قرار بگیرند . پس برای به دست آوردن نقاط تنها بایستی درخت را خیلی بیشتر از این ادامه دهیم. در این‌جاست که به ایده‌ی اصلیِ جنگلِ ایزوله ((Isolation Forest می‌رسیم.

با روشِ تقسیم‌بندیِ درخت‌ها در جنگلِ ایزوله، داده‌های پَرت، سریع‌تر از سایرِ نقاط (داده‌ها) تنها (ایزوله) می‌شوند.

اگر بخواهیم از منظرِ درخت بگوییم، داده‌های پَرت به ریشه (بالای) درخت نزدیک‌تر هستند زیرا این درخت‌ها، نقاطِ ایزوله را زودتر از سایرین پیدا می‌کنند و آن‌ها را ایزوله (تنها) می‌کنند. در واقع هر چقدر یک داده، بیشتر پَرت باشد، زودتر توسطِ درخت‌ها پیدا می‌شود.

برای تولیدِ جنگل بایستی چند درخت (مثلا ۱۰۰۰درخت) ساخته شود و هر درخت مشخص می‌کند که کدام نقاط زودتر ایزوله شدند (داده‌ی پَرت هستند)، و جنگلِ ایزوله هم با توجه به این درخت‌ها و تصمیماتشان، تصمیمِ نهایی را گرفته و به هر نقطه (به صورت میانگین) یک امتیاز بین ۰ تا ۱ می‌دهد. هر چقدر این امتیاز به ۱نزدیک‌تر باشد، این نقطه به احتمال بیشتری، داده‌ی پَرت است. این امتیاز بر اساس فرمول زیر محاسبه می‌شود:

$$S = 2^{-\frac{E(h(x))}{c(n)}}$$

ارتفاع نقطه

میانگین ارتفاع نقاط

در واقع هر چقدر یک نقطه در درخت‌های مختلف، زودتر ایزوله (تنها) شود، صورتِ کسر کوچک‌تر شده و در نتیجه امتیازِ آن به یک (۱) نزدیک‌تر می‌شود.

الگوریتم DBSCAN جهت تشخیص داده های پرت

خوشه بندی مکانی داده های دارای نویز بر مبنای چگالی به اختصار(DBSCAN) همانند k-میانگین، یک الگوریتم خوشه‌بندی بدون نظارت است. از این الگوریتم می‌توان برای تشخیص داده‌های پرت استفاده کرد.

یکی از دلایل محبوبیت DBSCANاین است که می‌تواند خوشه‌های غیرخطی قابل تفکیک را پیدا کند، در حالی‌که الگوریتم k-میانگین و مدل مخلوط گوسی قادر به انجام این کار نیستند. زمانی‌که خوشه‌ها به اندازه کافی متراکم هستند و به وسیله نواحی با چگالی پایین از یکدیگر جدا شده‌اند، الگوریتم DBSCANعملکرد خوبی دارد.

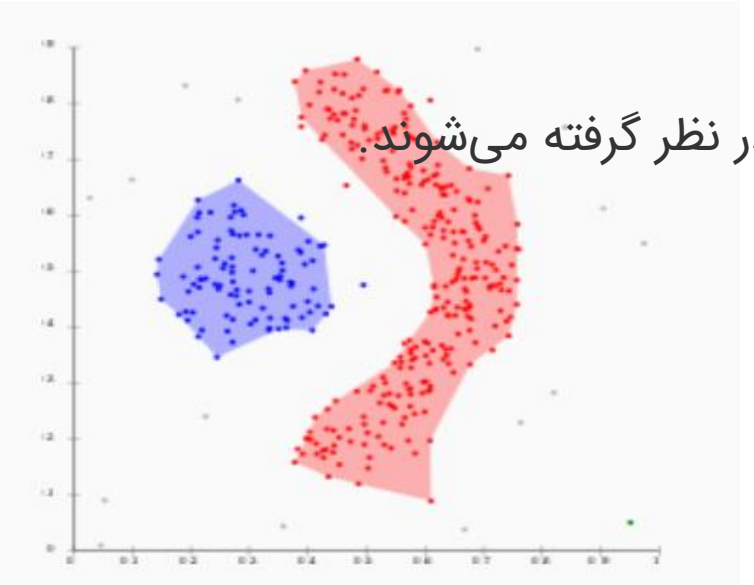
الگوریتم DBSCAN، خوشه‌ها را نواحی‌ای پیوسته با چگالی بالا در نظر می‌گیرد. نحوه عملکرد این الگوریتم بسیار ساده است:

۱- برای هر یک از نمونه‌ها، تعداد نمونه‌هایی که در فاصله εاپسیلون از آن قرار دارند را محاسبه می‌کند. این ناحیه، همسایگی εنمونه نامیده می‌شود.

۲- چنان‌چه تعداد نمونه‌هایی که در همسایگی εنمونه قرار دارند بیشتر از min-samplesباشد، این نمونه، نمونه مرکزی در نظر گرفته می‌شود. نمونه مرکزی به نمونه‌ای اطلاق می‌شود که در ناحیه‌ای با چگالی بالا قرار دارد (ناحیه‌ای که نمونه‌های زیادی در آن قرار دارند).

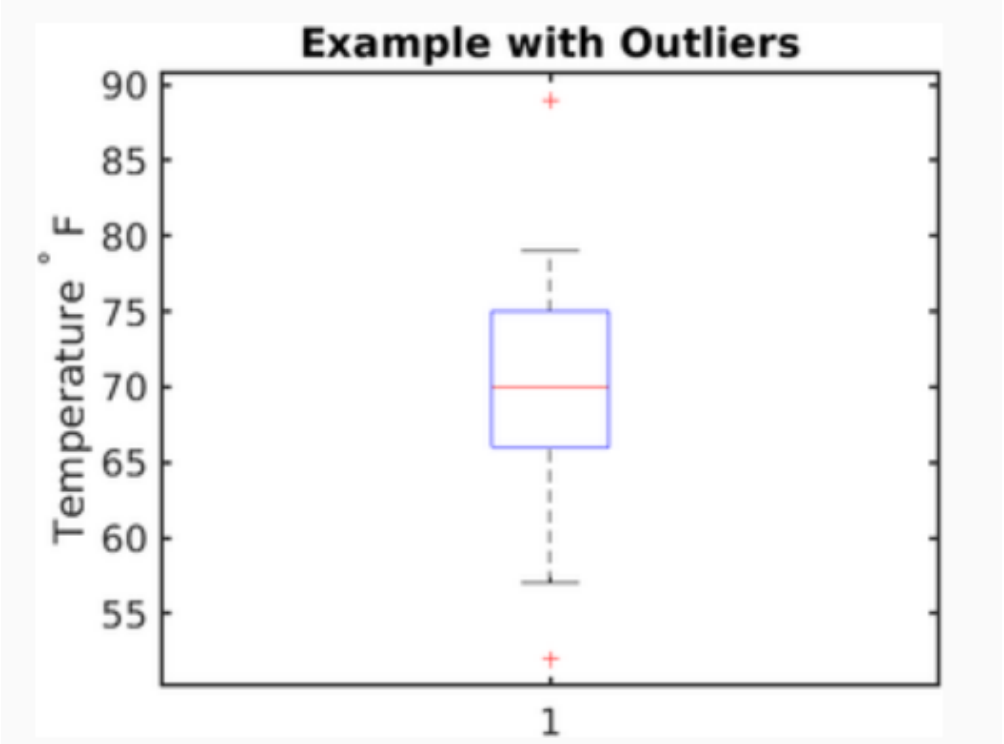
۳- تمامی نمونه‌هایی که در همسایگی εنمونه مرکزی قرار دارند، به همان خوشه تعلق می‌گیرند. ممکن است در همسایگی ε نقطه مرکزی، بیش از یک نقطه مرکزی وجود داشته باشد، بنابراین، یک توالی طولانی از نمونه‌های مرکزی همسایه، یک خوشه واحد را تشکیل می‌دهد.

۴- نمونه‌هایی که نمونه مرکزی نیستند و یا در همسایگی εنقطه مرکزی قرار ندارند، داده‌پرت در نظر گرفته می‌شوند.



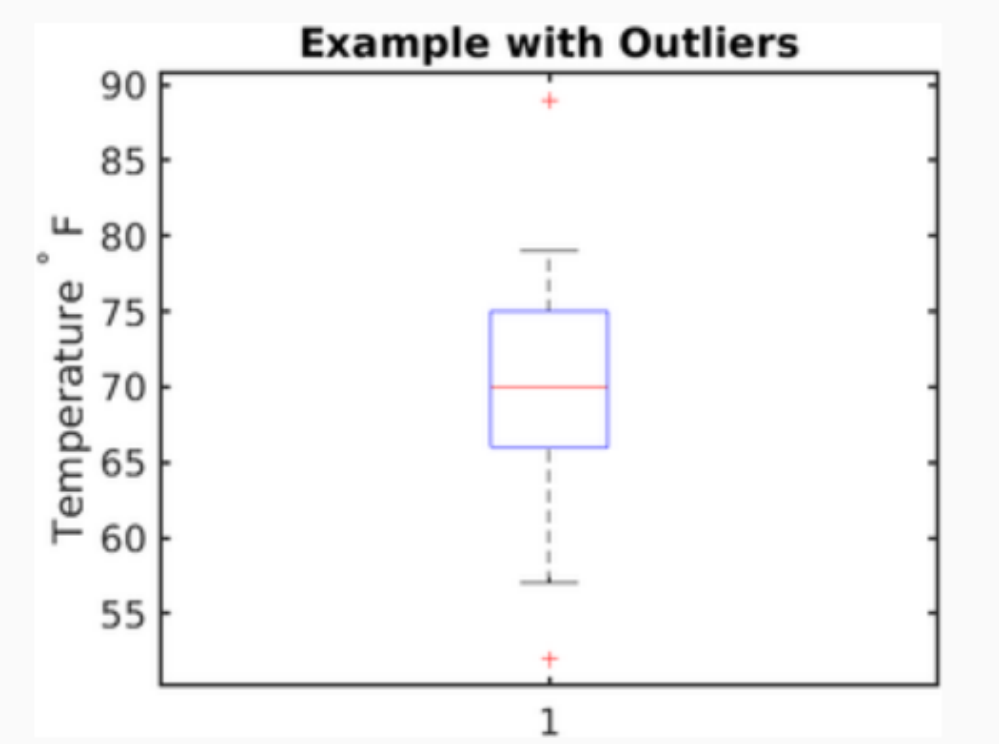
نمودار جعبه‌ای و روش توکی

هرچند Boxplotsروش رایج و محبوبی برای تشخیص ناهنجاری‌ها است، اما آن‌چنان که باید به روش توکی برای تشخیص داده‌های پرت توجه نشده است. پیش از معرفی روش توکی، مروری برBoxplots خواهیم داشت :

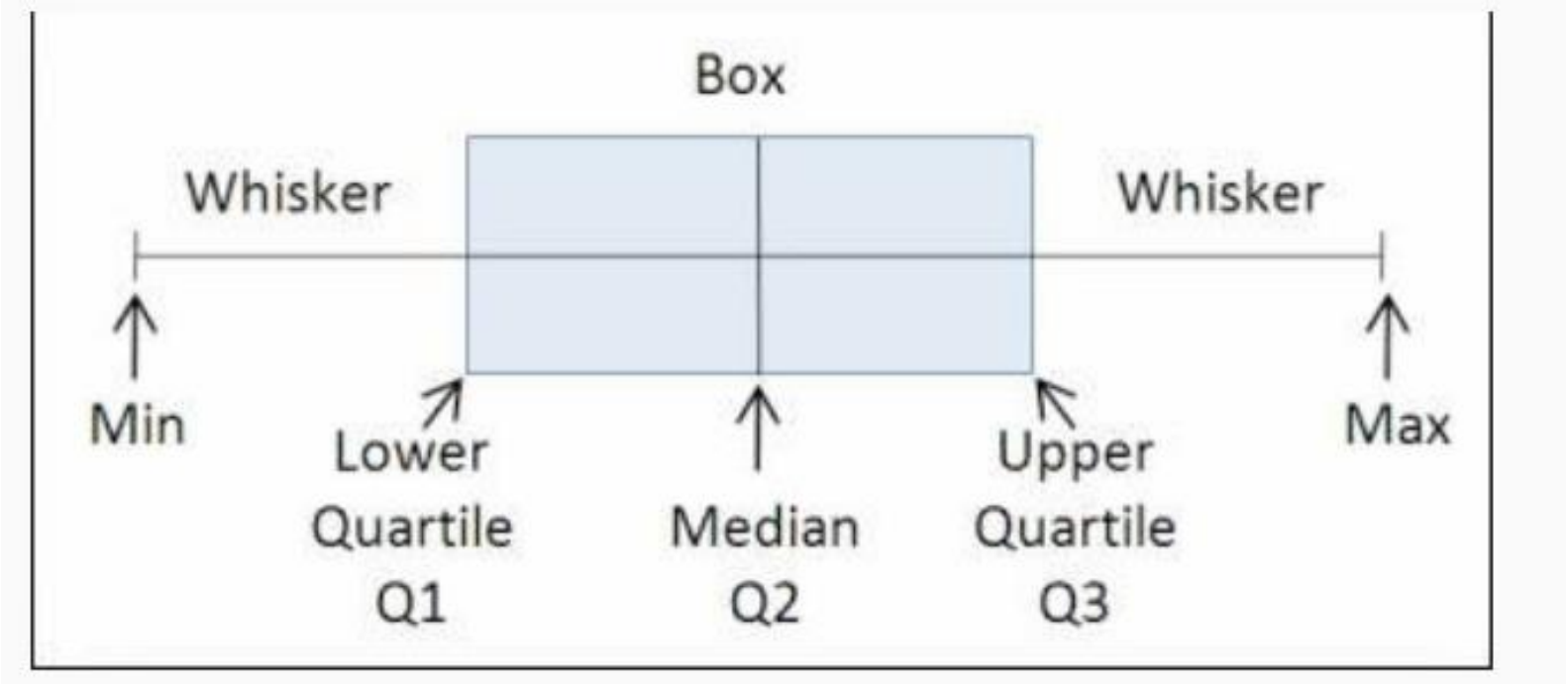


Boxplotها برای نمایش داده‌های عددی در چارک‌ها، جعبه‌ای با خطوط ترسیم می‌کند. این روش بسیار ساده و در همان حال برای نشان دادن داده‌های پرت بسیار مؤثر است.

خطوط بالا و پایین محدوده یک توزیع را مشخص می‌کنند و هر چیزی که در بالا و یا پایین این خطوط قرار بگیرد، داده پرت در نظر گرفته می‌شود. برای درک بهتر تشخیص داده پرت در نظر داشته باشید که در تصویر بالا، تمامی داده‌هایی که بالای (به طور تقریبی) 80 و پایین (به طور تقریبی) 58 قرار بگیرد، داده پرت است.



در قدم اول، نمودار جعبه‌ای، دیتاست را به ۵ بخش تقسیم می‌کنید:



- **مینیمم:** پایین‌ترین نقطه داده‌های موجود در توزیع به استثنای داده‌های پرت.
- **ماکسیمم:** بالاترین نقطه داده‌های موجود در توزیع به استثنای داده‌های پرت.
- **میانه (چارک دوم/پنجاه‌مین صدک):** مقدار میانی دیتاست.
- **صدک اول (چارک اول/بیست‌وپنجمین صدک):** میانه نیمه پایین دیتاست.
- **صدک سوم (چارک سوم/هفتادوپنجمین صدک):** میانه نیمه بالایی دیتاست.

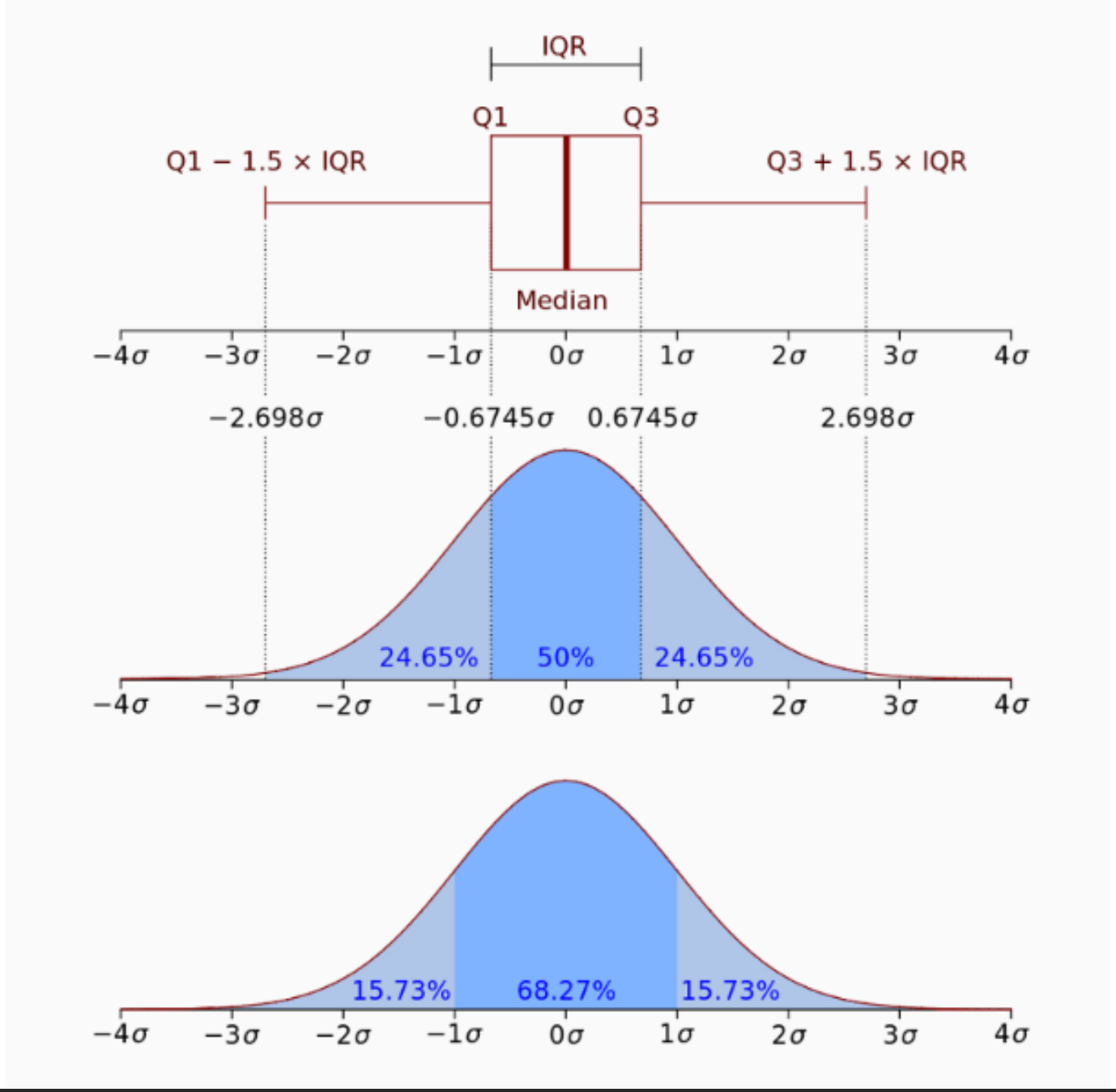
- اهمیت دامنه بین چارکی در این است که داده‌های پرت را مشخص می‌کند. IQRبه شکل زیر است:

```
IQR = Q3 - Q1

Q3: third quartile
Q1: first quartile
```

در نمودارهای جعبه‌ای، فاصله ۱.۵ برابری IQRمحاسبه می‌شود و نقطه‌داده‌هایی که در قسمت فوقانی دیتاست قرار دارند، را در بر می‌گیرد. به همین ترتیب، فاصله ۱.۵ برابری IQRنقطه‌داده‌هایی که در قسمت پایین دیتاست مشاهده می‌شوند، محاسبه می‌شود. به بیان دقیق‌تر:

- اگر نقاط مشاهده شده پایین از (Q1 – 1.5 * IQR)یا خط پایینی نمودار جعبه‌ای باشند، داده پرت به حساب می‌آیند.
- اگر نقاط مشاهده‌شده بالای (Q3 + 1.5 * IQR)یا خط فوقانی نمودار جعبه‌ای باشند، داده پرت به حساب می‌آیند.


IQR
Q1
Q3
Q1 − 1.5 × IQR
Q3 + 1.5 × IQR
Median
−4σ
−3σ
−2σ
−1σ
0σ
1σ
2σ
3σ
4σ
−2.698σ
−0.6745σ
0.6745σ
2.698σ
24.65%
50%
24.65%
15.73%
68.27%
15.73%