## گسسته‌سازی با استفاده از برچسب‌گذاری مقادیر صفت خاصه

این الگوریتم یک مجموعه‌داده که شامل صفات خاصه و برچسب می‌باشد را به‌عنوان ورودی دریافت کرده و مقادیر صفات خاصه به‌صورت گسسته را به همراه نقاط برش، برمی‌گرداند. نوع این الگوریتم، بالابه‌پایین است. الگوریتم‌های بالابه‌پایین با نقاط برش خالی شروع می‌شوند و در هر مرحله یک نقطه برش به نقاط برش اضافه می‌شود. در مقابل الگوریتم‌های پایین‌به‌بالا با تمامی نقاط برش ممکن کار خود را آغاز کرده و در هر مرحله یک نقطه برش را حذف می‌کنند. یک روش بالابه‌پایین کلاسیک، MDLP و یک روش پایین‌به‌بالای شناخته‌شده، ChiMerge است.

مراحل این الگوریتم به شرح زیر می‌باشد. (مراحل برای یک مجموعه‌داده دارای دو برچسب Y و N توضیح داده شده است اما با اندکی تغییر می‌توان برای مجموعه‌داده‌های با بیش از دو برچسب نیز آن را پیاده‌سازی کرد.)

1. به‌ازای هر ستون از ستون‌های صفات خاصه مراحل 2 تا 6 را تکرار کن.
2. مقادیر صفت خاصه را به‌صورت صعودی مرتب کن.
3. برای هر مقدار درصد برچسب Y و N را محاسبه کن.
4. درصورتی‌که درصد برچسب Y برای یک مقدار، بزرگ‌تر مساوی m(درصد Y ها در کل مجموعه‌داده) باشد، به آن مقدار، برچسب Y، درغیراین­صورت برچسب N را نسبت بده.
5. صفت خاصه مرتب‌شده را پیمایش کرده و درصورتی‌که برچسب دو مقدار متوالی با یکدیگر متفاوت‌اند یک نقطه برش بین آن دو، آن‌گونه که در مرحله 6 توضیح داده شده، لحاظ کن.
6. فرض می‌کنیم دو مقدار x1 و x2 دو مقدار متوالی از صفت خاصه بوده که به ترتیب برچسب آلفا و بتا به آن‌ها نسبت‌داده‌شده و نقطه برش cp بین این دو مقدار باید محاسبه شود. این مقدار به‌صورت زیر محاسبه می‌شود.

a: تعداد رکوردهایی که برچسب آلفا و مقدار x1 را دارند.

A: تعداد رکوردهایی که برچسب آلفا و مقدار x1 را دارند تقسیم‌بر تعداد رکوردهایی که مقدار x1 را دارند.

b: تعداد رکوردهایی که برچسب بتا و مقدار x2 را دارند.

B: تعداد رکوردهایی که برچسب بتا و مقدار x2 را دارند تقسیم‌بر تعداد رکوردهایی که مقدار x2 را دارند.

روش پیشنهادی ما با 5 گسسته ساز[[1]](#footnote-1) زیر مقایسه شده است.

* mdlp
* Chi2
* ChiMerge(chiM)
* Extended Chi2(eChi2)
* Modofied Chi2(mChi2)

روش‌های ذکرشده از طریق زبان برنامه‌نویسی R، در پکیج discretization قابل‌دسترسی می‌باشند.

عملکرد الگوریتم پیشنهادی ما برای گسسته‌سازی، با استفاده از 10 مجموعه‌داده از مخزن keel data set، موردبررسی قرارگرفته است. ویژگی‌های اصلی این مجموعه‌داده‌ها در جدول 1 آمده است.

|  |  |  |
| --- | --- | --- |
| **#Atts.** | **#Ex.** | **Data Set** |
| 7 | 106 | Appendicities |
| 19 | 539 | Bands |
| 3 | 306 | Haberman |
| 13 | 270 | Heart |
| 19 | 155 | hepatitis |
| 5 | 961 | Mammographic |
| 5 | 5472 | Phoneme |
| 8 | 768 | Pima |
| 9 | 699 | Wisconsin |
| 57 | 4597 | Spambase |

جدول 1 – ویژگی‌های مجموعه‌داده‌های مورداستفاده

برای مقایسه روش پیشنهادی با روش‌های گسسته‌سازی ذکرشده، از طبقه‌بندهای زیر استفاده‌شده است.

* J48
* Naïve Bayes
* Random Forest
* Linear Regression
* Baging

معیاری که برای محاسبه دقت روش‌های ذکرشده در بالا استفاده‌شده accuracy می‌باشد که به‌صورت زیر محاسبه می‌شود.

|  |  |  |  |
| --- | --- | --- | --- |
| **True** | |  | |
| **Negative** | **Positive** |
| False Positive (FP) | True Positive (TP) | **Positive** | **Predicted** |
| True Negative (TN) | False Negative (FN) | **Negative** |

برای به‌دست‌آوردن دقت روش‌ها به‌صورت زیر عمل می‌کنیم.

1. 10 fold cross validation را بر روی تمامی مجموعه‌داده‌ها اعمال کن.
2. به‌ازای هر مجموعه‌داده (D) مراحل 3 تا 7 را تکرار کن.
3. به‌ازای هر گسسته ساز (S) مراحل 4 تا را تکرار کن.
4. به‌ازای هر طبقه‌بند (C) مراحل 5 تا را تکرار کن.
5. باتوجه‌به fold‌های به‌دست‌آمده برای مجموعه‌داده D، 10 fold CV را روی این مجموعه‌داده با استفاده از طبقه‌بند C اعمال کن. برای گسسته‌سازی، هر بار که یکی از fold‌ها به‌عنوان داده مقصد و بقیه به‌عنوان داده آموزشی انتخاب می‌شوند، گسسته ساز را روی داده آموزشی اعمال کرده و باتوجه‌به نقاط برش به‌دست‌آمده، داده مقصد را نیز گسسته‌سازی کن.
6. دقت مرحله 5 را از طریق میانگین گرفتن بین 10 دقت به‌دست‌آمده برای هر fold محاسبه کن.
7. دقت به‌دست‌آمده را به‌عنوان دقت گسسته ساز S، با استفاده از طبقه‌بند C، روی مجموعه‌داده D لحاظ کن.

جدول 2 نتایج میانگین مرتبط با accuracy برای هر طبقه‌بند را نشان می‌دهد. برای هر طبقه‌بند، گسسته سازها از بهترین به بدترین مرتب شده‌اند.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| bag | | lr | | j48 | | rf | | nb | | mean | |
| avl  chi2  chiM  mdlp  eChi2  mChi2 | |  | | --- | | 0.8369 | | 0.8308 | | 0.8306 | | 0.8252 | | 0.8222 | | 0.8135 | | mdlp  avl  chi2  chiM  eChi2  mChi2 | |  | | --- | | 0.8209 | | 0.8174 | | 0.8165 | | 0.8155 | | 0.8100 | | 0.8079 | | chiM  chi2  eChi2  mdlp  avl  mChi2 | |  | | --- | | 0.8222 | | 0.8206 | | 0.8198 | | 0.8194 | | 0.8189 | | 0.8117 | | avl  chiM  chi2  eChi2  mdlp  mChi2 | |  | | --- | | 0.8466 | | 0.8416 | | 0.8379 | | 0.8306 | | 0.8239 | | 0.8106 | | eChi2  mdlp  chiM  chi2  avl  mChi2 | |  |  |  | | --- | --- | --- | | 0.8115 |  |  | | 0.8100 |  |  | | 0.8094 |  |  | | 0.8090 |  |  | | 0.8072 |  |  | | 0.7996 |  |  | | avl  chim  chi2  mdlp  echi2  mchi2 | |  | | --- | | 0.8254 | | 0.8239 | | 0.8230 | | 0.8199 | | 0.8188 | | 0.8087 | |

جدول 2 – نتایج میانگین accuracy مرتبط با شش طبقه‌بند.

روش پیشنهادی ما، در دو مورد بهترین نتیجه را به دست آورده است. درصورتی‌که میانگین دقت‌های به‌دست‌آمده برای هر روش گسسته‌سازی را محاسبه کنیم، مقادیر به‌طوری‌که در ستون آخر جدول 2 نشان داده ‌شده است خواهد بود که روش پیشنهادی ما از این لحاظ بهترین دقت را بین 6 گسسته ساز داراست.

جدول 3 میانگین زمان اجرای گسسته سازها را به ثانیه نشان می‌دهد. مقدارهای مشخص‌شده، میانگین زمان گسسته‌سازی یک مجموعه‌داده را نشان می‌دهد. از مجموعه‌داده‌های جدول 1 برای محاسبه این زمان استفاده‌شده است. گسسته سازها، از کمترین به بیشترین ازنظر زمان اجرا مرتب شده‌اند.

|  |  |
| --- | --- |
| Time(s) | Discretier |
| 0.172 | avl |
| 1.862 | mdlp |
| 88.862 | chiM |
| 91.108 | mChi2 |
| 174.947 | Chi2 |
| 305.651 | eChi2 |

فرایند گسسته‌سازی تنها یک‌بار بر روی داده آموزشی اجرا می‌شود درنتیجه به نظر می‌رسد که زمان اجرا معیار ارزیابی مهمی به‌حساب نیاید. بااین‌وجود درصورتی‌که فاز گسسته‌سازی بیش‌ازحد طول بکشد، می‌تواند برای کاربردهای عملی غیرممکن شود. همان‌طور که در جدول مشخص‌شده است، روش پیشنهادی ما، با اختلاف زیادی سریع‌تر از بقیه روش‌ها عمل می‌کند. درصورتی‌که اندازه مجموعه‌داده بزرگ باشد، زمان اجرا یک معیار مهم خواهد بود. برای مثال درصورتی‌که مجموعه‌داده‌ی مشابه spambase با ده میلیون رکورد داشته باشیم، الگوریتم پیشنهادی ما حدود نیم ساعت، الگوریتم mdlp، 5 ساعت و الگوریتم echi2،50 روز زمان نیاز خواهد داشت.

جدول 4 میانگین تعداد نقطه برش برای هر گسسته ساز را نشان می‌دهد. مقدارهای مشخص‌شده، میانگین تعداد نقاط برش برای یک صفت خاصه، بین صفات خاصه مجموعه‌داده‌های جدول 1 را نشان می‌دهد. گسسته سازها، از کمترین به بیشترین ازنظر تعداد نقطه برش، مرتب شده‌اند. آخرین ردیف جدول 4، میانگین تعداد نقاط برش، بدون اعمال گسسته ساز را نشان می‌دهد.

|  |  |
| --- | --- |
| #Intervals | Discretier |
| 1.4170 | mdlp |
| 1.5893 | mChi2 |
| 2.0811 | eChi2 |
| 28.3551 | chiM |
| 30.2684 | Chi2 |
| 98.0847 | avl |
| 286.316289 | nothing |

یک ویژگی مطلوب برای گسسته‌سازی این است که صفت خاصه گسسته‌سازی شده تا جای ممکن تعداد مقادیر کمتری را داشته باشد زیرا تعداد زیاد نقاط برش فرایند یادگیری را کندتر و غیرمؤثرتر می‌کند. از طرف دیگر تعداد کم نقاط برش نیز باعث ازدست‌رفتن اطلاعات می‌شود؛ اما ازآنجایی‌که می‌توان داده اصلی را در کنار داده گسسته‌سازی شده نگه داشت، این مسئله قابل‌چشم‌پوشی است و هرچه تعداد نقاط برش کمتر باشد، گسسته ساز بهتر است. همان‌طور که در جدول بالا مشخص است، روش پیشنهادی ما، بین روش‌های بررسی‌شده بیشترین تعداد نقطه برش را داراست که یک ویژگی منفی به‌حساب می‌آید.

### بررسی شباهت روش گسسته­سازی ارائه‌شده با روش‌های موجود

با استفاده از 30 روش گسسته‌سازی موجود در نرم‌افزار KEEL، مجموعه‌داده haberman گسسته‌سازی شده و نقاط برش آن با نقاط برش روش ارائه‌شده مقایسه شد. هیچ‌کدام از روش‌ها، نقاط برش دقیقه برابر با روش ارائه‌شده نداشتند، تنها 2 روش chi2 و extended chi2 مقادیری تقریباً مشابه با روش ما داشتند اما الگوریتم آن‌ها با روش ما متفاوت بود.

برای حل مشکل تعداد اینتروال‌ها، از کد sp.del که در قسمت ضمیمه آورده شده است استفاده شد. تعداد نقاط برش از میانگین 96 عدد به 2 عدد برای هر ویژگی رسید اما دقت نیز 4 درصد کاهش پیدا کرد. ارائه روشی برای کاهش تعداد اینتروال‌ها به‌گونه‌ای که تعداد اینتروال‌ها کاهش کمتری داشته و در مقابل دقت نیز کاهش پیدا نکند می­تواند راه­گشا باشد.

1. discretizer [↑](#footnote-ref-1)