تمرین ۳
درس داده کاوی

استاد: جناب آقای دکتر فراهانی

دستیاران آموزشی: جناب آقای نوید کاشی، جناب آقای علی شریفی

دانشجو: عمران اسدالهی کیا

تیر ۱۴۰۰

۱- توابع کرنل، ضرب داخلی بین دو نقطه در یک فضای ویژگی مناسب را برمی گردانند. بنابراین، با هزینه محاسباتی کم، حتی در فضاهای با ابعاد بالا، مفهومی از شباهت را تعریف می کنند.

توابع کرنل، برای داده های ترتیبی، نمودار ها، متن ها، تصاویر و همچنین بردار ها معرفی می شوند. پرکاربردترین نوع تابع کرنل، RBF است. زیرا دارای پاسخ محلی و متناهی در کل بازه محور x است.

کرنلهای پرکاربرد SVM:

- کرنل چند جمله ای

این کرنل در پردازش تصویر پرکاربرد است. معادله آن به صورت زیر است:

$$k(\mathbf{x_i}, \mathbf{x_j}) = (\mathbf{x_i} \cdot \mathbf{x_j} + 1)^d$$

که در آن d درجه چند جمله ای است.

- کرنل گاوسی

این یک کرنل برای اهداف عمومی است. و هنگامی که هیچ دانش پیشینی در مورد داده ها وجود ندارد استفاده می شود. معادله آن به صورت زیر است:

$$k(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right)$$

- تابع پایه شعاعی گاوسی (RBF)

این کرنلی برای اهداف عمومی کلربرد دارد. و هنگامی که هیچ دانش پیشینی در مورد داده ها وجود نداشته باشد، مورد استفاده قرار می گیرد. معادله آن به صورت زیر است:

$$k(\mathbf{x_i}, \mathbf{x_j}) = \exp(-\gamma\|\mathbf{x_i} - \mathbf{x_j}\|^2)$$

و برای $\gamma > 0$ گاهی اوقات با استفاده از پارامتر زیر استفاده می شود:

$$\gamma = 1/2\sigma^2$$

- کرنل RBF لاپلاس

این هم یک کرنل برای اهداف عمومی است. و هنگامی که هیچ دانش پیشینی در مورد داده ها وجود ندارد استفاده می شود. معادله آن به صورت زیر است:

$$k(x, y) = \exp\left(-\frac{\|x - y\|}{\sigma}\right)$$

- کرنل تانژانت هیپربولیک(tanh)

می توانیم از آن در شبکه های عصبی استفاده کنیم. معادله مربوط به آن عبارت است از:

$$k(\mathbf{x_i}, \mathbf{x_j}) = \tanh(\kappa\mathbf{x_i} \cdot \mathbf{x_j} + c)$$

در برخی موارد (نه همیشه) $k > 0$ و $c < 0$.

- کرنل سیگموئید

می توان این کرنل را در شبکه های عصبی مورد استفاده قرار داد. معادله مربوط به آن عبارت است از:

$$k(x, y) = \tanh(\alpha x^T y + c)$$

- کرنل تابع بِسِل (Bessel) از نوع اول

ما می توانیم از آن برای حذف مقطع عرضی در توابع ریاضی استفاده کنیم. معادله آن عبارت است از:

$$k(x, y) = \frac{J_{v+1}(\sigma\|x - y\|)}{\|x - y\|^{-n(v+1)}}$$

که J تابع بسل از نوع اول است.

- کرنل پایه شعاعی ANOVA

ما می توانیم از آن در مسائل رگرسیون استفاده کنیم. معادله مربوط به آن عبارت است از:

$$k(x,y) = \sum_{k=1}^{n} \exp(-\sigma(x^k - y^k)^2)^d$$

- کرنل spline خطی بصورت یک بعدی

این کرنل، هنگام کار با بردارهای بزرگ داده پراکنده ، کاربرد زیادی دارد. این کرنل اغلب در دسته بندی متن مورد استفاده قرار می گیرد. کرنل spline همچنین در مسائل رگرسیون عملکرد خوبی دارد. معادله آن عبارت است از:

$$k(x,y) = 1 + xy + xy\ min(x,y) - \frac{x+y}{2}\ min(x,y)^2 + \frac{1}{3}\min(x,y)^3$$

اگر در مورد توابع کرنل SVM سوالی دارید ، در صورت تمایل با ما در میان بگذارید. ما از این که پاسخگوی سوالات شما باشیم خوشحال خواهیم شد.

۲- انجام شد.

۳- ترکیبات مختلف درخواست شده ی پارامترها بررسی و در جدول ۱ گزارش شده است.

Kernel	C	Gamma (RBF)	Degree(Poly)	Test Accuracy	Train Accuracy
linear	1	-	-	95.89	97.8
linear	100	-	-	97.05	99.09
linear	1000	-	-	97.05	99.69
rbf	1	0.1	-	80.97	99.89
rbf	1	0.001	-	72.77	75.63
rbf	1	scale	-	88.14	98.25
rbf	100	0.1	-	81.58	100
rbf	100	0.001	-	94.78	97.27
rbf	100	scale	-	88.84	100
rbf	1000	0.1	-	81.58	100
rbf	1000	0.001	-	95.39	99.4
rbf	1000	scale	-	88.84	100
poly	1	-	2	43.29	63.53
poly	1	-	3	76.08	95.15
poly	100	-	2	52.65	83.05
poly	100	-	3	74.42	100
poly	1000	-	2	55.76	87.55
poly	1000	-	3	74.42	100

جدول ۱: مقایسهٔ پارامترهای محتلف SVM

۴- با توجه به اینکه زمانی که C خیلی بزرگ است Hard margin معادل Soft margin است، تمام C=1000 را Hard margin در نظر می‌گیریم.

۵- آ) در تمرین قبل روی battery_power از binning با سایز مساوی استفاده کردیم.

ب) روش کدبندیOne-Hot ، یکی از پرکاربردترین روشهای کدبندی است و عملکرد آن به جز در مواردی که متغیر دسته‌ای مقادیر خیلی زیادی بگیرد، بسیار خوب است (معمولا از این روش برای متغیرهایی که بیش‌تر از ۱۵ مقدار متفاوت بگیرند، مناسب نیست. در برخی از مواردی که تعداد متغیرها کمتر است نیز امکان دارد گزینه مناسبی نباشد). کدبندی One-Hot ستون‌های دودویی (binary) جدیدی می‌سازد که هر یک مربوط به یکی از مقادیری هستند که متغیر به خود می‌گیرد.
این روش در تمرین قبل روی blue، dual_sim، touch_screen و wifi تست شد. که همگی دارای بیش از دو دسته نبودند.

ج) تبدیل لگاریتمی یک روش تبدیل داده است که در آن هر متغیر x را با یک log (x) جایگزین می کند. انتخاب پایه لگاریتم معمولاً به عهده تحلیلگر است و این به اهداف مدل سازی آماری بستگی دارد.
هنگامی که داده های مداوم اصلی ما از bell curve پیروی نمی‌کنند، می توانیم این داده ها را تبدیل کرده تا تجزیه و تحلیل آماری حاصل از این داده‌ها معتبرتر شود. به عبارت دیگر ، تغییر شکل ورود به سیستم داده های اصلی ما را کاهش می دهد یا از بین می برد. نکته مهم این است که داده های اصلی باید یک توزیع نرمال را دنبال کنند یا تقریباً دنبال کنند. در غیر این صورت ، تبدیل لگاریتمی کار نخواهد کرد.
د) در تمرین قبل که روی دیتاست موبایل کار شده بود انجام شد.

۶- در شماره ۲ همین تمیرین انجام شد.

۷- الگوریتم درخت تصمیم قادر است علاوه بر متغیرهای کمی، متغیرهای کیفی را نیز پیش بینی کند. نتیجه پیاده سازی الگوریتم درخت تصمیم مجموعه ای از شرط های منطقی(conditions then-if) با ساختار درختی است که برای پیش بینی یک ویژگی به کار می رود. طوری که داده هایی که در برگ های انتهایی این درخت قرار می گیرند توسط یکی از مقادیر ویژگی هدف برچسب می خورند. این مدل به دلیل سهولت در تفسیر نتایج و ناپارامتری و غیر خطی بودن، نیاز به پیش فرض رابطه خطی بین متغیر های مستقل و وابسته ندارد.
الگوریتم درخت تصمیم به گونه ای عمل می کند که سعی دارد گوناگونی و یا تنوع (از نظر ویژگی هدف) را در گره ها به حداقل ممکن برساند. این عدم یکنواختی در گره ها با استفاده از معیارهای عدم خلوص (measure Impurity) قابل اندازه گیری است که مهمترین و پرکاربرد ترین آن شاخص جینی می باشد.اغلب تفاوت انواع درخت های تصمیم در همین معیار اندازه گیری عدم خلوص، شیوه شاخه بندی (Splitting) و هرس کردن گره های درخت می باشد.
انواع درخت تصمیم:

- ID3

یکی از الگوریتم‌های بسیار ساده درخت تصمیم است. اطلاعات به دست آمده به عنوان معیار تفکیک به کار می‌رود. این الگوریتم هیچ فرایند هرس کردن را به کار نمی‌برد و مقادیر اسمی و مفقوده را مورد توجه قرار نمی‌دهد. الگوریتم C4 تکامل یافتهٔ ID3 است، Gain Ratio به عنوان معیار تفکیک در نظر گرفته می‌شود. عمل تفکیک زمانی که تمامی نمونه‌ها پایین آستانه مشخصی واقع می‌شوند، متوقف می‌شود. پس از فاز رشد درخت عمل هرس کردن بر اساس خطا اعمال می‌شود. این الگوریتم مشخصه-های اسمی را نیز در نظر می‌گیرد.

- CART

برای برقراری درخت‌های رگرسیون و دسته‌بندی از این الگوریتم استفاده می‌شود. نکته حائز اهمیت این است که این الگوریتم درخت‌های باینری ایجاد می‌کند به طوری که از هر گره داخلی دو لبه از آن خارج می‌شود و درخت‌های بدست آمده توسط روش اثربخشی هزینه، هرس می‌شوند. از ویژگی‌های این الگوریتم، توانایی در تولید درخت‌های رگرسیون است. در این نوع از درخت‌ها برگ‌ها به جای کلاس مقدار واقعی را پیش‌بینی می‌کنند. الگوریتم برای تفکیک کننده‌ها، میزان مینیمم مربع خطا را جستجو می‌کند. در هر برگ، مقدار پیش‌بینی بر اساس میانگین خطای گره‌ها می‌باشد.

- CHID

این الگوریتم درخت تصمیم به جهت در نظرگرفتن مشخصه‌های اسمی طراحی شده است. الگوریتم برای هر مشخصه ورودی یک جفت مقدار که حداقل تفاوت را با مشخصه هدف داشته باشد، پیدا می‌کند.

محققان آمار کاربردی، الگوریتم‌هایی را جهت تولید و ساخت درخت تصمیم توسعه دادند. الگوریتم CHAIDدر ابتدا برای متغیرهای اسمی طراحی شده بود. این الگوریتم با توجه به نوع برچسب کلاس از آزمون‌های مختلف آماری استفاده می‌کند. این الگوریتم هرگاه به حداکثر عمق تعریف شده‌ای برسد و یا تعداد نمونه‌ها در گره جاری از مقدار تعریف شده‌ای کمتر باشد، متوقف می‌شود. الگوریتم CHAIDهیچگونه روش هرسی را اجرا نمی‌کند.

دیگر الگوریتمها:

ID4
ds CART: DempsterShafer Classification and Regression Tree
ID5R
EC4.5: Efficient Classifier 4.5
CHAID: Chi square Automatic Interaction Detection
RF: Random Forest
RT: Random Tree
DS: Decision Stump
QUEST: Quick Unbiased Efficient Statistical Tree

۸- تست ما در جدول ۲ با ترکیبات مختلف آزمون و آموزش نشان میدهد درخت تصمیم برای این نوع داده مفید نیست.

۹- همانطور که در جدول ۲ مشخص است افزایش عمق در بیشتر موارد باعث بیش برازش می شود.

Criterion	Max depth	Splitter	Test Accuracy	Train Accuracy
gini	5	random	72.47	74.91
gini	5	best	82.17	88.11
gini	10	random	81.58	95.55
gini	10	best	83.28	99.32
gini	15	random	81.43	99.95
gini	15	best	81.93	99.99
gini	20	random	81.38	100
gini	20	best	82.33	100
entropy	5	random	75.83	76.53
entropy	5	best	81.98	87.57
entropy	10	random	83.38	97.16
entropy	10	best	84.98	99.86
entropy	15	random	81.63	99.95
entropy	15	best	84.68	100
entropy	20	random	82.43	100
entropy	20	best	84.08	100

جدول ۲: مقایسه پارامترهای مختلف درخت تصمیم

۱۰- هرس کردن درخت تصمیم مقابل عمل تقسیم کردن است و با هرس کردن زیر گره‌هایی در درخت تصمیم حذف می‌گردد. زمانی که یک درخت تصمیم ساخته می‌شود، تعدادی از شاخه‌ها ناهنجاری‌هایی در داده‌های آموزش منعکس می‌کنند که ناشی از داده‌های پرت و یا نویز است.

در برخی الگوریتم‌های ایجاد درخت، هرس کردن جزئی از الگوریتم محسوب می‌شود. در حالی که در برخی دیگر، تنها برای رفع مشکل بیش برازش از هرس کردن استفاده می‌شود.

چندین روش، معیارهای آماری را برای حذف کمتر شاخه‌های قابل اطمینان به کار می‌برند. درخت‌های هرس شده تمایل به کوچک‌تر بودن و پیچیدگی کم‌تر دارند و بنابراین به راحتی قابل فهم می‌باشند. آن‌ها معمولا در طبقه‌بندی صحیح داده‌های تست سریع‌تر و بهتر از درخت‌های هرس نشده عمل می‌کنند.

دو رویکرد رایج برای هرس درخت به شرح ذیل وجود دارد:

- **پیش‌هرس(Pre pruning)**

در این رویکرد یک درخت به وسیله توقف‌های مکرر در مراحل اولیه ساخت درخت، هرس می‌شود. به محض ایجاد یک توقف گره به برگ تبدیل می‌شود.

- **هرس پسین(Post pruning)**

رویکرد هرس پسین درخت تصمیم رایج‌تر است به این صورت که زیر درخت‌ها از یک درخت رشد یافته کامل را حذف می‌کند. یک زیر درخت در یک گره به وسیله حذف کردن شاخه‌ها و جایگزینی آن‌ها با یک برگ، هرس می‌شود.

۱۱- بخش امتیازی.

۱۲- جدول ۳ مقایسه دقیق random forest با درخت تصمیم را نشان می‌دهد.

Criterion	Max depth	# of estimators	Test Accuracy	Train Accuracy
gini	5	10	70.27	82.93
gini	5	100	83.48	92.3
gini	10	10	77.58	98.54
gini	10	100	87.39	99.96
gini	15	10	79.73	99.7
gini	15	100	88.19	100
gini	20	10	79.73	99.7
gini	20	100	88.53	100
entropy	5	10	75.48	86.32
entropy	5	100	83.13	90.97
entropy	10	10	81.17	98.82
entropy	10	100	88.29	100
entropy	15	10	79.93	99.52
entropy	15	100	88.09	100
entropy	20	10	80.03	99.64
entropy	20	100	88.94	100

جدول ۳: مقایسه پارامترهای مختلف Random Forest

۱۳- مزایای درخت تصمیم:

- درخت تصمیم بدیهی است و نیاز به توصیف ندارد.
- هر دو مشخصه اسمی و عددی را می‌تواند مورد توجه قرار دهد.
- نمایش درخت تصمیم به اندازه کافی برای نشان دادن هرگونه طبقه‌بندی غنی است.
- مجموعه داده‌هایی که ممکن است دارای خطا باشند را در نظر می‌گیرد.
- مجموعه داده‌هایی که دارای مقادیر مفقوده هستند را شامل می‌شود.
- درخت‌های تصمیم روش‌های ناپارامتری را نیز مورد توجه قرار می‌دهد.

علاوه بر نقاط قوت اشاره شده، درخت تصمیم نیاز به محاسبات پیچیده برای دسته‌بندی داده‌ها ندارد. همچنین درخت تصمیم نشان می‌دهد که کدام مشخصه تاثیر بیشتری در دسته‌بندی دارند.

۱۴-

بر پایه درخت تصمیم مانند C4.5

الگوریتمهای شاخه و حرص مانند RIPPER، PART، CAMUR، BIGBIOCL

بر پایه فازی کردن مانند FURIA

بر پایه برآورد احتمال مانند MLRules، Rough set theory، LERS (LEM1, LEM2, MLEM2)

بر اساس رتبه مانند TSP، k-TSP

بر پایه الگوریتم ژنتیک مانند BIOHEL

کشف زیر گروه مانند CN2-SD، SDEFSR

RIPPER

هرس افزایشی مکرر برای تولید خطا (Repeated Incremental Pruning to Produce Error Reduction) یکی از کارآمدترین و مورد استفاده ترین الگوریتم های یادگیری قاعده است. یک استراتژی تقسیم و تسخیر (divide-and-conquer) را برای rule induction اجرا می کند. Ripper اصطلاحاً هرس خطای کاهش یافته افزایشی را برای تدوین مجموعه ای از قوانین اولیه برای هر کلاس اعمال می کند. سپس، یک مرحله بهینه سازی اضافی هر قانون را در مجموعه فعلی به نوبت در نظر می گیرد و دو قانون جایگزین از آنها ایجاد می کند: یک قانون جایگزینی و یک قانون تجدید نظر. پس از آن، در مورد اینکه آیا مدل باید قاعده اصلی، جایگزینی یا قانون تجدید نظر را بر اساس معیار حداقل طول توصیف حفظ کند، تصمیم گیری می شود.

TSP

جفت امتیاز دهی برتر (Top Scoring Pair) یک روش القای قاعده است که بر اساس مقادیر نسبی بین جفت ویژگی ها بنا شده است. TSP برای داده های ریز آرایه و ایجاد قوانین در فضای مشخصه ای که با مقایسه دو به دو سطح بیان ژن تشکیل شده است، ایجاد شده است. مزیت اصلی رویکرد TSP این است که، بر اساس مقادیر نسبی، از مشکل یکپارچه سازی داده ها از منبع مختلف استفاده می کند که به طور بالقوه در مقیاس های مختلف نشان داده می شود و می تواند از اثرات دسته ای رنج ببرد. علاوه بر این، طبقه بندی TSP قوانین تصمیم گیری را ارائه می دهد که تفسیر آنها آسان است زیرا شامل مقادیر نسبی بین جفت ویژگی ها (ژن ها در مورد آن).

SDEFSR

کشف زیرگروه با سیستم فازی تکاملی (Subgroup Discovery with Evolutionary Fuzzy System) مجموعه ای از الگوریتم های القایی قاعده مبتنی بر کشف زیرگروه است که با استفاده از منطق فازی تفسیرپذیری نتایج را بهبود می بخشد. الگوریتم های SDEFSR قادر به تکامل قوانین فازی و استفاده از تعاریف مجموعه فازی هستند.

FURIA

الگوریتم القای قاعده نامنظم فازی (Fuzzy Unordered Rule Induction Algorithm) نسخه بهبود یافته الگوریتم RIPPER است. FURIA از الگوریتم اصلاح شده RIPPER به عنوان مبنا استفاده می کند و قوانین فازی و مجموعه قوانین غیر مرتب را می آموزد. نقطه قوت اصلی این الگوریتم روش کشش قانون است، که مشکل فشرده رکوردهای جدید را حل می کند که در صورت طبقه بندی می توانند خارج از فضای تحت پوشش قوانین قبلی باشند. نمایش قوانین فازی نیز پیشرفته است، اساساً یک قانون فازی از طریق جایگزینی فواصل با فواصل فازی، یعنی مجموعه های فازی با عملکرد عضویت ذوزنقه ای به دست می آید.

۱۵- به طور کلی داده های سری زمانی به داده‌های وابسته به زمان مربوط می‌شود. تحلیل سری زمانی نیز یک از روش‌های تحلیل چنین داده‌هایی است. برای مثال تشخیص روند تغییرات ارزش سهام با توجه به داده‌های جمع‌آوری شده در طول یک سال می‌تواند تحلیل سری زمانی نامیده شود.

به صورت پیش فرض شاید به نظر برسد که از درخت تصمیم برای حل مسائل وابسته به زمان نمیتوان استفاده کرد، ولی نهایتاً از درخت تصمیم در آنالیز تصمیم، برای مشخص کردن استراتژی که با بیشترین احتمال به هدف برسد بکار می‌رود.

پس از آنجا که نهایتاً در نتیجه تحلیل های سری زمانی نیز نهایتاً میخواهیم تصمیمی مبنی بر موضوعی برای آینده اتخاذ کنیم، میتوان از درخت تصمیم کمک گرفت.

۱۶- طبق درخواست سوال فایل قیمت بیت کوین از ۲۰۱۰/۰۱/۰۱ تا ۲۰۲۱/۰۵/۰۱ دریافت شد. البته علی رغم تصور، داده ها از ۲۰۱۰/۰۷/۱۸ شروع می شوند. دیتا طبق درخواست سوال به دو قسمت تست و آموزش از

۲۰۱۰/۰۱/۰۱ تا ۲۰۲۰/۰۱/۰۱ برای آموزش و ۲۰۲۰/۰۱/۰۲ تا ۲۰۲۱/۰۵/۰۱ برای تست تقسیم و در نظر گرفته شدند و تصحیحات لازم روی داده انجام شد.

۱۷- Adaboost و Random Forest پیاده سازی شد.

۱۸- Adaboost از روش boosting استفاده می کند.

۱۹- پیاده سازی Adaboost انجام شد. مشاهده می شود هرچه نرخ یادگیری بیشتر میشود دقت مقداری بهتر میشود. البته این مقدار توام با تعداد مدلهای استفاده شده یا extimator تغییر میکند که افزایش آن تاثیر مثبتی بر دقت دارد این بهبود بیشتر از مقدار ۲ نمیتواند باشد و بهترین مقدار learning rate ۲ می باشد.

```
parameters:
random_state=0, learning_rate=0.5, n_estimators=100, loss='linear'
RMSE: 20401.82377393996
accuracy: 0.0
------------------------------
parameters:
random_state=0, learning_rate=1, n_estimators=150, loss='linear'
RMSE: 20983.3568315924
accuracy: 0.0
------------------------------
parameters:
random_state=0, learning_rate=1.5, n_estimators=200, loss='linear'
RMSE: 21218.19262818136
accuracy: 0.0
------------------------------
parameters:
random_state=0, learning_rate=2, n_estimators=300, loss='linear'
RMSE: 21581.53311152942
accuracy: 0.0
------------------------------
parameters:
random_state=0, learning_rate=2.5, n_estimators=400, loss='linear'
RMSE: 21266.15792744618
accuracy: 0.0
```

۲۰- پیاده سازی Random Forest انجام شد. مشاهده می شود هرچه عمق بیشتر میشود دقت مقداری بهتر میشود بعلاوه اگر تعداد sample کم شود باعث over fit میشود و همچنین اگر زیاد شود باعث کم شدن دقت می شود، این مورد عمق نیز مصداق دارد اگر خیلی زیاد شود میتواند باعث over fit شود و اگر خیلی کم باشد تقریباً اصلاً بکار نمی آید.

```
parameters:
max_depth=16, random_state=0, criterion='mse', min_samples_leaf=1
RMSE: 14276.161083618244
accuracy: 0.045
------------------------------
parameters:
max_depth=8, random_state=0, criterion='mse', min_samples_leaf=10
RMSE: 2048.2149623525047
accuracy: 0.0033333333333333335
------------------------------
parameters:
max_depth=4, random_state=0, criterion='mse', min_samples_leaf=50
RMSE: 328.0626660437526
accuracy: 0.0
------------------------------
parameters:
max_depth=2, random_state=0, criterion='mse', min_samples_leaf=100
RMSE: 401.63513650607325
accuracy: 0.0
```

۲۱- در سوالات قبل پاسخ داده شد.

۲۲- بخش امتیازی

۲۳- بخش امتیازی

۲۴- بخش امتیازی

۲۵- بخش امتیازی

۲۶- برای این سوال، یک دیتاست بزرگ داده شده است. از آنجا که ارزش واقعی داده های آزمون داده شده در دیتاست (tournament data)، این فایل را به دو قسمت مجزای test و train تقسیم میکنیم. داده ها دارای ۵ کلاس مختلف است: ۰، ۰,۲۵، ۰,۵، ۰,۷۵، ۱. سوال مطرح شده انتظار را مطرح نکرده است. اگر مقادیر مرتب شده یا مقادیر منطقی مانند: ۰,۲۴ یا ۰,۷۸ و غیره باشد معنی دار هستند و باید از مدل های رگرسیون استفاده کنیم. در غیر این صورت بهتر است از روش های کلاسیفیکیشن استفاده کنیم.

نمی توان از مدلهای یادگیری ماشین دلخواه استفاده کرد. به عنوان مثال Support Vector Machines (SVMs) یک ماتریس ۵۰۰۰۰۰ در ۵۰۰۰۰۰ ایجاد میکنند که پردازش آن در RAM غیر ممکن است. بعلاوه، حل یک برنامه درجه دو با این اندازه از نظر محاسباتی کارا نیست. بنابراین بهتر است از روش های کارآمد مانند درختان تصمیم یا مدل های خطی رگرسیون استفاده کنیم.