«تمرین سوم داده‌کاوی»

محمدحسین کمیلی

۹۹۴۲۲۱۶۳

ارشد علوم کامپیوتر

دانشگاه شهید بهشتی

خرداد ۱۴۰۰

۱. کرنل‌ها درواقع راه و روشی بهینه‌تر و باهزینه محاسباتی خیلی کم‌تر برای انتقال داده‌ها به ابعاد بالاتر ارائه می‌کنند که در آن بعد بالاتر، داده‌ها با وسیله یک ابرصفحه به صورت خطی جدایی پذیرند اما در ابعاد اصلی مسأله داده‌ها به صورت خطی جدایی پذیر نبوده‌اند. کرنل‌های زیاد و متفاوتی ارائه شده‌اند که معروف‌ترین آن‌ها کرنل چندجمله‌ای و کرنل RBF می‌باشد. در ادامه تعدادی دیگر از کرنل‌ها معرفی میشوند:

- کرنل چندجمله‌ای (Polynomial Kernel)
- کرنل گاوسی (Gaussian Kernel)
- کرنل پایه شعاعی ((Gaussina Radial Basis Function (RBF)
- کرنل تانژانت هایپربولیک (Hyperbolic Tangent Kernel)
- کرنل سیگموید (Sigmoid Kernel)
- کرنل آنووا (Anova Radial Basis Kernel)
- کرنل اسپلاین خطی (Linear Spline Kernel)
- کرنل لاپلاس (Laplace RBF Kernel)
- کرنل تابع بسل نوع اول (Bessel function of the first kind Kernel)

در مورد یک قانون کلی برای استفاده از کرنل‌ها میتوان گفت استفاده از کرنل‌ها خیلی به داده بستگی دارد. اگر بدانیم مجموعه داده به صورت خطی جدایی پذیر است، تقریباً کم هزینه‌ترین و معقول‌ترین کرنل، کرنل خطی است. اگر دانش پیشینی از داده موجود نباشد، می‌توان از کرنل‌های همه‌منظوره مثل RBF یا کرنل گاوسی یا کرنل لاپلاس استفاده کرد. همچنین در مسائل پردازش تصویر، کرنل چندجمله‌ای از کرنل‌های معروف به شمار می‌آید.

۲. در کد پیاده‌سازی‌شده، ۲۰ مدل SVM با استفاده از مقادیر مختلف برای دو پارامتر kernel و C ارائه‌شده است

۳. برای مدل‌های ارائه‌شده، ۴ کرنل استفاده‌شده عبارت‌اند از: کرنل خطی، کرنل چندجمله‌ای، کرنل گوسی و کرنل RBF و در میان آن‌ها بهترین نتیجه را کرنل خطی با دقت ۹۹/۲۵ روی مجموعه اعتبارسنجی داراست. همچنین در بین کل ۲۰ مدل ارائه‌شده، مدل دارای کرنل خطی با مقدار ۱۰ و ۱ برای دارامتر C با دقت ۹۹/۲۵ روی مجموعه اعتبارسنجی بهترین مدل می‌باشد.

۴. با استفاده از پارامتر C می‌توان تعیین کرد که حاشیه یا margin نرم است یا سخت. هر چقدر C مقدار بزرگتری داشته‌باشد، مدل از حاشیه سخت‌تر برای جداسازی استفاده می‌کند. با کاهش پارامتر C تا ۱، دقت افزایش یافته‌است و پس از آن دقت کاهش یافته است و بهترین نتیجه با استفاده از کرنل خطی هنگامی که پارامتر C برابر با ۱ است به دست آمده‌است. برای اینکه بتوانیم دقیق‌تر میزان تأثیر سختی یا نرمی حاشیه را مشاهده کنیم می‌توان برای یک کرنل ثابت تعداد زیادی مدل تعریف کرد که همگی با هم فقط در پارامتر C تفاوت دارند و سپس دقت همه مدل‌ها را به تصویر کشید. هرچه تعداد مدل‌هایی که تعریف میکنیم بیشتر باشد، می‌توانیم تفسیر دقیق‌تری بدست‌آوریم.

۵. چهار بخش این سؤال به ترتیب در زیر ارائه می‌شوند:

1. در این قسمت روی ویژگی battery_power از روش binning استفاده کردیم. این روش را ۴ بار با استفاده از مقادیر متفاوت برای سطل‌ها(Buckets) تست کردیم. در دو بار اول عرض سطل‌ها یکسان و در مرحله سوم غیریکسان و در مرحله چهارم هم تنها یک سطل برای تمامی مقادیر این ویژگی در نظر گرفتیم تا به نوعی میزان اهمیت ویژگی battery_power را مشاهده کنیم. از نظر معیارهای تعریف‌شده نظیر f1 ،recall، precision، و ... هر چهار تقسیم‌بندی به نتایج یکسانی منجر شدند.

2. در این قسمت روی ویژگی‌های دسته‌ای (categorical) موجود در دیتاست روش onehot encoding را اعمال کردیم. ۶ ویژگی categorical دیتاست عبارت‌اند از: ،four_g، dual_sim، blue wifi، touch_screen، three_g، . با اعمال این روش، به دقتی ۹۸ درصدی روی مجموعه اعتبارسنجی دست‌یافتیم. در مورد چرایی استفاده از این روش می‌توان گفت در هنگام کار کردن با داده‌های کتگوریکال، تعدادی از الگوریتم‌های یادگیری نمی‌توانند به صورت مستقیم از این نوع ویژگی‌ها دانش استخراج کنند. پس لازم است روی مقادیر این ویژگی‌ها تبدیلی اعمال شود که آن‌ها را به حالت عددی در بیاورد. حال ممکن است بگوییم کافیست به هر یک از مقادیر ممکن آن ویژگی یک عدد نسبت دهیم و برای مثال برای ویژگی رنگ که شامل مقادیر سبز و آبی و قرمز است تبدیلی به صورت سبز : ۱ و قرمز : ۲ و آبی : ۳ را در نظر بگیریم. مشکلی که با استفاده از این تبدیل پیش می‌اید این است که در بسیاری از ویژگی‌های کتگوریکال، بین مقادیر آن ویژگی هیچ ترتیبی وجود ندارد اما با استفاده از این تبدیل درواقع ما نوعی ترتیب را بین مقادیر ایجاد کردیم که اصلاً وجود ندارد. در این مواقع onehot encoding گزینه‌ای مطلوب محسوب می‌شود.

3. در مورد تبدیلات لگاریتمی و استفاده آن‌ها به طور کلی می‌توان گفت: هنگامی که داده‌های پیوسته ما از فرم زنگوله‌ای پیروی نمی‌کنند و به سمت چپ یا راست کجی یا به صورت دقیق‌تر چولگی دارند، با اعمال این تبدیل، میتوانیم آن داده‌ها را به فرم نرمال نزدیک‌تر کنیم تا در نتیجه این تغییر، بتوانیم تحلیل آماری معتبرتری از داده‌ها استخراج کنیم. با استفاده از رسم کردن چندین ویژگی دارای مقادیر پیوسته وهمچنین مقادیر آن‌ها پس از اعمال تبدیل لگاریتمی می‌توان متوجه شد که چولگی برطرف نمی‌شود و با اعمال این تبدیل تنها کمی پیچیدگی به مدل اضافه خواهیم‌کرد.
همچنین در این قسمت با استفاده از standardScaler داده‌ها را اسکیل کردیم که در نتیجه آن دقت به ۹۸ درصد رسید که حدوداً یک درصد کاهش را شامل می‌شود اما در سرعت برازش مدل svm تأثیر به‌سزایی را شاهد هستیم. به طور کلی نرمال یا اسکیل کردن داده‌ها پیش از یادگیری، در سرعت برازش یا همگرایی مدل بسیار کارساز خواهدبود.

4. در این قسمت، یک ویژگی جدید با نام sc_surface به داده‌ها اضافه کردیم. این ویژگی از ضرب دو ویژگی دیگر یعنی sc_h و sc_w به دست می‌آید. با توجه به رابطه خطی موجود بین دو ویژگی قبل و ویژگی اضافه‌شده، می‌توان فهمید که این ویژگی با ویژگی‌های موجود کورلیشن دارد و همانطور که می‌توان حدس زد، در مدل svm پیاده‌سازی شده بر دادگان جدید، دقت افزایش نمی‌یابد.

۶. در قسمت قبل، حاصل اعمال هر یک از روش‌های مهندسی داده به صورت جداگانه روی دادگان بررسی شد. در این قسمت به بررسی نتیجه حاصل از اعمال تمامی روش‌ها به صورت همزمان روی دادگان می‌پردازیم. با اعمال همه روش‌ها، دقت مدل svm افزایش یا کاهش خاصی نداشت و در نهایت به دقت ۹۸ درصد دست‌یافتیم. در میان تمامی ۵ حالت قبل، بهترین نتایج را در مرحله binning و همچنین افزودن ویژگی جدید دریافت کردیم.

۷. الگوریتم‌های مختلفی در مبحث درخت‌های تصمیم ارائه می‌شوند که به طور کلی در دو مورد با یکدیگر تفاوت دارند: الگوریتم‌های مختلف، رویکردهای مختلفی را به هنگام انتخاب یک گره و شکستن آن به دو یا چند گره دیگر دنبال می‌کنند. همچنین انتخاب الگوریتم برای درخت تصمیم به نوع متغیرهای هدف نیز بستگی دارد. تعدادی از انواع الگوریتم‌های درخت‌های تصمیم عبارت‌اند از:

- ID3
- C4.5
- CART (Classification And Regression Tree)
- CHAID (Chi_square Automatic Interaction Detection)
- MARS (Multivariate Adaptive Regression Splines)

در ادامه الگوریتم ID3 دقیق‌تر مورد بررسی قرار می‌گیرد:

الگوریتم ID3 از رویکرد بالا به پایین حریصانه برای ساخت درخت تصمیم از میان فضای تمام شاخه‌های ممکن استفاده میکند. گام‌های این الگوریتم در زیر آورده شده‌است:

1. آغاز کار با مجموعه اصلی S به عنوان ریشه
2. در هر تکرار، الگوریتم ویژگی با کمترین استفاده در S را در نظر گرفته و برای آن آنتروپی (Entropy) و بهره اطلاعات (Information Gain) را محاسبه می‌کند.
3. در این مرحله، ویژگی با کمترین آنتروپی یا بیشترین بهره اطلاعات را انتخاب می‌کند.
4. سپس مجموعه S با استفاده از ویژگی انتخاب‌شده شکسته می‌شود تا زیرمجموعه‌ای از داده تولید شود.

5. الگوریتم به دور زدن روی هر زیرمجموعه ادامه می‌دهد و ویژگی‌هایی که قبلاً انتخاب نشده‌اند را در نظر می‌گیرد.

۸. در این مرحله یک درخت تصمیم ساده روی دیتا ایجاد شد که در اولین تلاش و بدون تنظیم پارامترهای مختلف و قابل تنظیم درخت تصمیم به دقت ۸۸ دست‌یافت

۹. در این مرحله، تأثیر عمق درخت (max_depth) , تعداد نمونه‌های موجود در هر گره (min_sample_split) مورد بررسی قرار گرفت. همچنین در خلال بررسی این دو پارامتر، پارامتر دیگری به نام criterion که درواقع تابعی برای اندازه‌گیری کیفیت تقسیم‌بندی است نیز مورد بررسی قرار گرفت. در مورد عمق درخت میتوان به صورت کلی گفت با افزایش عمق تا حدی مشخص، دقت بهتر می‌شود و پس از آن با افزایش بیشتر روندی تناوبی به وجود می‌آید. در مورد افزایش عمق درخت، نکت حائز اهمیت این است که با افزایش عمق درخت، احتمال بیش‌برازش یا overfitting هم افزایش می‌یابد که البته روش‌هایی برای جلوگیری از بیش‌برازش درخت تصمیم وجود دارد. همچنین برای در مورد تعداد نمونه‌های موجود در هر گره می‌توان گفت با توجه به وجود وضعیت تصادفی در مدل، بهترین نتایج هنگامی به‌دست می‌آید که تعداد نمونه‌ها بین ۲ تا ۵ قرار گیرد و با افزایش داده‌ها دقت به مراتب کاهش می‌یابد. در هنگام بررسی هر دو پارامتر، پارامتر criterion نیز مورد بررسی قرار گرفت و در مورد این دیتاست که در حال حاضر ما با آن کار می‌کنیم، entropy با اختلاف بسیار بسیار کمی نتایج بهتری نسبت به gini ارائه می‌دهد.

۱۰. Pruning یا هرس کردن درواقع یک روش فشرده‌سازی داده است که در مبحث درخت‌های تصمیم با حذف قسمت‌های غیربحرانی و اضافه درخت تصمیم، باعث کاهش سایز درخت تصمیم می‌شود. هرس کردن، پیچیدگی نهایی کلاس‌بند را کاهش داده و باعث کاهش بیش‌برازش و در نتیجه افزایش دقت پیش‌بینی می‌شود. یک استراتژی معمول برای هرس کردن این است که درخت را تا آن‌جا که هر گره شامل تعداد مشخصی نمونه شود رشد دهیم و سپس گره‌هایی که اطلاعاتی اضافه نمی‌کنند را حذف کنیم. نکته مهم این است که هرس کردن باید سایز درخت تصمیم را بدون کاستن دقت کاهش دهد.

۱۱. برای اعمال هرس روس درخت تصمیم، از متد cost_complexity_pruning_path که در کتابخانه sklearn موجود است استفاده می‌کنیم. با استفاده از این متد، می‌توانیم مقادیر alpha و impurity یا ناخالصی را در هر برگ بدست آوریم. سپس با استفاده از تصویرسازی مقادیر آلفا نسبت بهه عمق و دقت، میتوانیم بهترین مقدار آلفا را برای مسأله پیدا کنیم.در مورد مسئله‌ای که در حال حاضر روی آن

کار می‌کنیم حتی با هرس کردن نیز به دقت بیش از تقریباً ۸۹ درصد نمی‌رسیم اما برای درخت‌های پرعمق، هرس کردن می‌تواند مفید واقع شود.

۱۲. با استفاده از جنگل تصادفی (random forest) به دقت بالاتری در مقایسه با درخت تصمیم رسیدیم. دلیل این مشاهده این است که جنگل تصادفی تعداد زیادی درخت تصمیم را با استفاده از انتخاب ویژگی تصادفی می‌سازد که باعث دست‌یابی به دقت بالاتر و البته در زمان بیشتری می‌شود. درواقع با استفاده از یک مدل جنگل تصادفی، ما تعداد زیادی مدل درخت تصمیم را با هم در یک مدل ترکیب می‌کنیم.

۱۳. روش درخت تصمیم به دلیل داشتن ویژگی‌های مثبت زیادی که در ادامه به آن‌ها اشاره می‌کنیم، حتی با وجود روش‌های قدرتمندی نظیر شبکه‌های عصبی عمیق باز هم کاربردهای زیادی در زمینه یادگیری دارند. از ویژگی‌های مثبت درخت‌های تصمیم میتوان به موارد زیر اشاره کرد:

- سادگی فهم و تفسیر: با کمی توضیح یا نمایش گرافیکی درخت‌های تصمیم، به راحتی برای مبتدیان و افراد غیرمتخصص قابل درک و فهم است.
- توانایی کار با انواع داده‌های عددی و کتگوریکال: روش‌هایی مانند شبکه‌های عصبی مصنوعی نیازمند تبدیل همه داده‌ها به نوع عددی هستند. اما الگوریتم‌های جدید درخت تصمیم نظیر C4.5 چنین محدودیتی ندارند.
- آماده‌سازی داده کم: این روش‌ها اغلب به آماده‌سازی داده کمی نیاز دارند و تقریباً هیچ نیازی به ساخت متغیرهای یه اصطلاح dummy ندارد.
- استفاده از مدل جعبه سفید یا جعبه باز: اگر وضعیت داده‌شده توسط مدل قابل دسترسی یا مشاهده باشد معمولاً توضیح آن به سادگی توسط منطق بولی قابل ارائه است.
- عملکرد مناسب با دادگان زیاد
- شباهت بسیار زیاد به نوع تصمیم‌گیری انسان
- ...

۱۴. الگوریتم‌های مختلفی برای یادگیری قوانین موجود در دادگان و کمک به تصمیم گیری ارائه شده‌اند. الگوریتم‌هایی نظیر Agrawal، RULEX، Progol، C4.5، CN2، IREP، Ripper و ... که در ادامه،

تعدادی از این الگوریتم‌های یادگیری مبتنی بر قوانین (rule-based learning algorithms) مورد بررسی قرار می‌گیرند:

1. Ripper: الگوریتم Ripper یا Repeated Incremental Pruning to Produce Error Reduction یک الگوریتم استقرای قانون (rule induction) است که شامل سه مرحله فرآیند قانون است:
   1) ساخت: در این مرحله، مجموعه آموزشی به دو مجموعه رشد(growing) و هرس(pruning) تقسیم می‌شود. قوانین بر پایه مجموعه رشد ساخته می‌شوند و بعداً از مجموعه هرس برای کاهش این قوانین ساخته‌شده استفاده می‌شود.
   2) بهینه‌سازی: در این مرحله، تمامی قوانین دوباره مورد بررسی قرار می‌گیرند تا میزان خطا در کل مجموعه داده کاهش یابد.
   3) پاک‌سازی: در این مرحله، اینکه آیا هر قانون Description Length(DL) مجموعه قوانین و داده‌ها را افزایش می‌دهد یا خیر.محاسبه DL این امکان را به ما می‌دهد که پیچیدگی هر مجموعه قوانین را به صورت کمی در بیاوریم. قانونی که باعث افزایش DL شود، حذف می‌شود.
2. CN2: این الگوریتم نیز یک الگوریتم یادگیری برای rule induction است که برای مواقعی که حتی داده کامل نیست طراحی شده است. این الگوریتم، بر پایه الگوریتم‌های AQ و ID3 است که در نتیجه، مجموعه قوانینی شبیه به الگوریتم AQ می‌سازد اما شبیه به ID3 نیز قادر به مدیریت داده‌های نویزی است.

۱۵. از درخت‌های تصمیم برای حل مسائل سری زمانی هم می‌توان استفاده کرد. اگرچه که برای استفاده از درخت‌های تصمیم روی سری‌های زمانی، باید پیش‌پردازش‌هایی روی داده صورت گیرد تا داده به فرم مسأله‌های یادگیری نظارت‌شده تبدیل شود. همچنین برای ارزیابی مدل نیاز به روشی به نام walk-forward validation می‌باشد. برای تبدیل سری زمانی به مسأله یادگیری نظارت‌شده نیز می‌توان از روش پنجره متحرک (sliding window) استفاده کرد.

۱۶. داده را از لینک داده‌شده دریافت و به همانگونه خواسته‌شده به دو قسمت آموزش و تست تقسیم کردیم. پیش‌پردازش‌هایی روی داده انجام دادیم که در ادامه آن‌ها را به همراه دلیل بیان می‌کنم:

1. ستون Date نیاز به تغییر فرمت تاریخ داشت که با استفاده از یک تابع کمکی، این تغییر صورت گرفت.
2. در ستون‌های Low، High، Open، Price، و Vol. برای جدا کردن هر سه رقم از کاراکتر کاما (,) استفاده شده‌بود که برای تبدیل این ستون‌ها به نوع float ابتدا نیاز بود این کاراکتر از همه مقادیر حذف شود.
3. در ستون Change پس از هر مقدار از کاراکتر درصد (%) وجود داشت که باز هم مثل قسمت قبل نیاز بود این کاراکتر از همه مقادیر حذف شود.
4. در ستون Vol. برای نشان دادن هزار یا میلیون از دو حرف K و M پس از هر مقدار استفاده شده‌بود که نیاز بود با استفاده از یک تابع کمکی ابتدا این حرف از مقدار برداشته شده و با توجه به نوع حرف، در یکی از اعداد ۱۰۰۰ و یا ۱۰۰۰۰۰۰ ضرب شود.
5. همه ستون‌های شامل اعداد دارای نوع object بودند که به float64 تبدیل شدند.
6. در آخر ستون Date نیز به نوع مناسب datetime تبدیل شد

۱۷. در این قسمت از مدل‌های VAR، XGBoost، و LSTM استفاده کردیم. مشکلی در هنگام آموزش LSTM بود که فکر می‌کنم به دلیل صحیح نبودن فرم داده‌ها به همگرایی نرسیدیم. در مورد خطاها، مدل XGBoost دارای RMSE برابر با ۱۵۴۲۳ و VAR برابر با ۵/۱۸ بود.

۱۸.

۱۹. با استفاده از AdaBoostRegressor و اعمال آن روی دیتاست میزان RMSE برابر با ۱۵۷۶۷ بدست آمد

۲۰. با استفاده از RandomForestRegressor و اعمال آن روی دیتاست میزان RMSE برابر با ۱۵۵۲۷ بدست آمد

۲۱. با صرف مظر کردن از ۵ درصد خطا و اعلام آن به عنوان کارایی مدل دقت‌های دریافت‌شده اصلاً خوب نیست. احتمالاً در فرآیند آماده‌سازی داده اشتباهی انجام داده‌ام که در صدد رفع آن هستم