
بسم الله الرحمن الرحیم

وزارت علوم، تحقیقات و فناوری

پژوهشگاه علوم و فناوری اطلاعات ایران (ایرانداک)

طرح پژوهشی

طراحی روشی هوشمند برای دسته‌بندی نوشتارهای علمی فارسی

مجری

جلال‌الدین نصیری

همکار

نصراله پاک‌نیت

فروردین ۱۳۹۹

حقوق: پژوهشگاه علوم و فناوری اطلاعات ایران

طرح پژوهشی طراحی روشی هوشمند برای دسته‌بندی نوشتارهای علمی فارسی پیرو قرارداد شماره ۳۳/۹۸۵۰ مورخ ۱۳۹۷/۸/۲۷ میان پژوهشگاه علوم و فناوری اطلاعات ایران(کارفرما) و آقای دکتر جلال‌الدین نصیری اجرا شده است. گزارش حاضر یک جلد از مستندات این طرح است.

این گزارش و تمامی حقوق مادی آن بر اساس قانون حمایت حقوق مولفان و مصنفان و هنرمندان، مصوب ۱۳۴۸ و اصلاحیه‌های بعدی آن و همچنین آیین‌نامه‌های اجرایی این قانون متعلق به پژوهشگاه علوم و فناوری اطلاعات ایران و هرگونه استفاده از تمامی یا پاره‌ای از آن، شامل: نقل قول، تکثیر، انتشار، کاربرد نتایج، تکمیل و مانند آنها به صورت چاپی، الکترونیکی یا وسایل دیگر فقط با اجازه کتبی پژوهشگاه امکان‌پذیر است. نقل قول در حد هزار واژه در انتشارات علمی مانند کتاب و مقاله با درج اطلاعات کامل کتابشناختی، نیازی به مجوز پژوهشگاه ندارد.

صحت مندرجات گزارش بر عهده مجری طرح پژوهشی است.

طراحی روشی هوشمند برای دسته‌بندی نوشتارهای علمی فارسی

مدیر طرح پژوهشی: دکتر جلال‌الدین نصیری، پژوهشگاه علوم و فناوری اطلاعات ایران.

نشانی: تهران، خیابان انقلاب، چهارراه فلسطین، شماره ۱۰۹۰، طبقه ۴، اتاق ۴۲۰

صندوق پستی: ۱۳۱۵۷۷۳۳۱۴ تلفن: ۶۶۴۹۴۹۸۰-۰۲۱(داخلی ۴۲۳)

وبگاه: http://irandoc.ac.ir/nasiri/nasiri.html

رایانامه: j.nasiri@irandoc.ac.ir

همکار طرح: دکتر نصراله پاک‌نیت

چکیده

با توجه به حجم بالای متون علمی و سرعت پایین کنترل اقلام اطلاعاتی در سامانه ثبت و در نتیجه عدم نمایش به روز درگنج، افزایش سرعت کنترل اقلام اطلاعاتی از موارد اولویت‌دار سامانه ثبت است. به عنوان مثال حدود ۱۴۰ هزار متون علمی همچنان نمایه نشده و با توجه به سیل پایان‌نامه های جدید، استفاده از روش هایی که می‌تواند کمکی به افزایش سرعت نمایه‌سازی کنند از اولویت های سامانه ثبت می‌باشد.

پایان‌نامه و رساله‌هایی که در پایگاه داده ثبت پژوهشگاه علوم و فناوری اطلاعات ایران(ایرانداک) ثبت می‌گردد دارای فراداده موضوع اصلی مانند علوم انسانی، فنی و مهندسی، علوم پایه، هنر ومعماری، علوم پزشکی و غیره می‌باشند. استخراج هوشمند حوزه پارساها بوسیله الگوریتم‌های یادگیری ماشین باعث افزایش سرعت فرایند نمایه‌سازی و افزایش کیفیت داده‌های پایگاده داده گنج شود.

در این طرح پژوهشی هدف دسته‌بندی هوشمند محتوایی متون به یکی از ۵ دسته موضوع اصلی علوم انسانی، فنی و مهندسی، علوم پایه، هنر ومعماری، علوم پزشکی می‌باشد. به عبارت دیگر افزایش سرعت نمایه‌سازی بوسیله خودکارسازی تایید یا عدم تایید موضوع اصلی ثبت شده از طرف پژوهشگر می‌باشد. در این پژوهش ابتدا پیش‌پردازش معمول در پردازش زبان طبیعی انجام شده و ویژگی‌ها متمایز کننده استخراج می‌گردد. در ادامه با استفاده از دسته‌بندهای متنوع محتوای پایان‌نامه‌ها به پنج دسته آموزش داده می‌شود. نتایج ارزیابی‌ها بر روی متون پایان‌نامه و رساله‌های فارسی نشان می‌دهد این روش با ارزیابی با متون بیشتر می‌تواند با دقت و سرعت خیلی خوبی موضوع اصلی پارساها را مشخص کند.

کلیدواژه‌ها: دسته‌بندی چندکلاسه؛ نمایه‌سازی خودکار؛ دادگان نامتوازن؛ پردازش الگو

فهرست مطالب

عنوان **شمارهٔ صفحه**

فهرست جدول‌ها

عنوان **شمارهٔ صفحه**

فهرست شکل‌ها

عنوان **شمارهٔ صفحه**

قدردانی

اکنون که به یاری خدا طراحی روشی هوشمند برای دسته‌بندی نوشتارهای علمی فارسی را به پایان رسانده‌ام جا دارد از زحمات همکاران خوبم علل‌الخصوص جناب آقای دکتر علی‌دوستی رئیس محترم پژوهشگاه که در انتخاب طرح جاری کمک شایانی نموده‌اند و همکارانم در پژوهشکده علوم اطلاعات تشکر و قدردانی نمایم. همچنین از ناظر محترم طرح جناب آقای دکتر هاشمی‌نژادکه با نظرات ارزشمند خود بهبود این طرح را تضمین نموده‌اند تشکر می‌نمایم.

فصل اول

کلیات

۱-۱. مقدمه

«پژوهشگاه علوم و فناوری اطلاعات ایران» (ایرانداک) که از سال ۱۳۸۸ به این نام خوانده می‌شود، بر پایه گاه‌شمار و همچنین نخستین سند در این زمینه، در یکم مهر ماه سال ۱۳۴۷ با نام «مرکز اسناد ایران» تأسیس شد. مأموریت بنیادین ایرانداک بر پایه اساس‌نامه و برنامه استراتژیک آن؛ پژوهش، مدیریت دانش، آموزش، همکاری‌های پژوهشی و اطلاع‌رسانی، و پشتیبانی از سیاست‌گذاری علم و فناوری است.

پژوهشگاه علوم و فناوری اطلاعات با جمع‌آوری و پردازش تمام پایان‌نامه‌ها و رساله‌های وزارت علوم، تحقیقات و فناوری از جایگاه ممتاز و روبه رشدی برای دانشجویان، اساتید و مدیران برخوردار شده است. دستاوردهای مدیریت دانش و اطلاعات علمی و فنی کشور در سامانه‌ها و پایگاه‌های اطلاعات ارائه می‌شوند. در این میان، پایگاه اطلاعات گنج با صدها هزار رکورد؛ از بزرگ‌ترین، دیرین‌ترین، و پربازدیدترین پایگاه‌های اطلاعات علمی و فنی کشور است. اطلاعات پایان‌نامه‌ها و رساله‌ها و پیشنهادیه آنها که در ایرانداک به ثبت می‌رسند، بی‌همتاست. ارائه سرویس های ارزش افزوده برای سیاست‌گذاران و تصمیم‌گیرندگان حوزه علم و فناوری و امکان تجاری‌سازی دانش در جهت تثبیت نقش مرجعیت پژوهشگاه می‌تواند مفید باشد.

با توجه به حجم بالای متون علمی و سرعت پایین کنترل اقلام اطلاعاتی در سامانه ثبت و در نتیجه عدم نمایش به روز درگنج، افزایش سرعت کنترل اقلام اطلاعاتی از موارد اولویت‌دار سامانه ثبت

است. به‌عنوان مثال حدود ۱۴۰ هزار متون علمی همچنان نمایه نشده و با توجه به سیل پایان‌نامه‌های جدید، استفاده از روش‌هایی که می‌تواند کمکی به افزایش سرعت نمایه‌سازی کنند از اولویت‌های سامانه ثبت می‌باشد.

استخراج و تایید اقلام اطلاعاتی پایان‌نامه‌ها که به آن فراداده[۱] نیز گفته می‌شود از وظایف نمایه‌ساز می‌باشد. در صورتی‌که بتوان کنترل، ورود یا تایید فراداده را بوسیله ماشین انجام داد سرعت کنترل اقلام اطلاعاتی بهتر خواهد گردید. پایان‌نامه و رساله‌هایی که در پایگاه داده ثبت پژوهشگاه علوم و فناوری اطلاعات ایران(ایرانداک) ثبت می‌گردد دارای فراداده موضوع اصلی مانند علوم انسانی، فنی و مهندسی، علوم پایه، هنر ومعماری، علوم پزشکی و غیره می‌باشند. استخراج هوشمند حوزه پارساها بوسیله الگوریتم‌های یادگیری ماشین باعث افزایش سرعت فرایند نمایه‌سازی و افزایش کیفیت داده‌های پایگاده داده گنج شود.

در این طرح پژوهشی هدف دسته‌بندی هوشمند محتوایی متون به یکی از ۵ دسته موضوع اصلی علوم انسانی، فنی و مهندسی، علوم پایه، هنر ومعماری، علوم پزشکی می‌باشد. به‌عبارت دیگر افزایش سرعت نمایه‌سازی بوسیله خودکارسازی تایید یا عدم تایید موضوع اصلی ثبت شده از طرف پژوهشگر می‌باشد.

۱-۲. بیان مساله

سرعت پایین کنترل اقلام اطلاعاتی و حجم روز افزون پایان‌نامه‌ها به علت به روز نبودن متون علمی در پایگاه داده گنج می‌باشد. ثبت و بررسی صحت اقلام داده وارد شده توسط پژوهشگر و ویرایش اشتباهات احتمالی یکی از وظایف نمایه‌سازان می‌باشد. اقلام اطلاعاتی شامل نام و نام پژوهشگران، دانشگاه یا پژوهشگاه متبوع، نام پایان‌نامه و دسته پایان‌نامه از مواردی است که یک نمایه‌ساز به آنها دقت می‌کند و در صورت وجود اختلاف باید تصحیح کند.

---

[۱] Metadata

به عبارت دیگر استخراج و تایید اقلام اطلاعاتی پایان‌نامه‌ها که به آن فراداده گفته می‌شود از وظایف نمایه‌ساز می‌باشد.یک از راه های افزایش سرعت کنترل اقلام اطلاعاتی، ماشینی کردن کنترل بعضی از این فرادادهها می‌باشد. پایان‌نامه و رساله‌هایی که در پایگاه داده ثبت پژوهشگاه علوم و فناوری اطلاعات ایران(ایرانداک) ثبت می‌گردد دارای فراداده "موضوع اصلی" مانند علوم انسانی، فنی و مهندسی، علوم پایه، هنر ومعماری، علوم پزشکی و غیره می‌باشند. استخراج هوشمند حوزه پارساها بوسیله الگوریتم‌های یادگیری ماشین باعث افزایش سرعت فرایند کنترل اقلام اطلاعاتی و افزایش کیفیت داده‌های پایگاده داده گنج شود.

استفاده از روش‌های خودکار برای تصدیق اقلام داده‌ای که پژوهشگر وارد نموده است می‌تواند در افزایش سرعت فرآیند نمایه‌سازی نیمه‌خودکار کنونی کارا باشد. در واقع جایگاه این طرح پژوهشی کمک به نمایه‌ساز در جهت افزایش سرعت تایید یا عدم تایید صحت اقلام اطلاعاتی وارد شده می‌باشد. در این طرح پژوهشی "موضوع اصلی" متون علمی به صورت خودکار بوسیله استخراج محتوای متون علمی به صورت هوشمند سنجیده می‌گردد.

پژوهشگاه نیازمند ابزارهایی برای پیدا کردن الگوها و نظم و قاعده‌ها در میان متون علمی در جهت افزایش کارایی می‌باشد. دسته‌بندی یکی از تکنیک‌های پرکاربرد و از مهمترین روش‌های متن‌کاوی است که با استفاده از آن می توان حجم زیادی از داده‌ها را برحسب نظم‌هایی که درون آن‌ها وجود دارد در گروه‌های منظم و از پیش تعریف شده دسته‌بندی کند. روش‌های دسته‌بندی متعددی برای استخراج دانش از دل داده‌ها وجود دارد و الگوریتم‌های مختلفی از روی این روش‌ها بدست آمده است.

روش های یادگیری ماشین معمولا در سه گروه خوشه‌بندی[۲]، دسته‌بندی[۳] و درونیابی[۴] تقسیم می‌گردند. هدف دسته‌بندی داده‌ها، سازماندهی و تخصیص داده‌ها به کلاس‌های مجزا می‌باشد. در

[۲] Clustering

[۳] Classification

[۴] Regression

این فرآیند براساس داده‌های توزیع شده، مدل اولیه‌ای آموزش داده می‌شود. سپس از این مدل برای پیشگویی کلاس داده‌ی جدید استفاده می‌شود. مدل بدست آمده در اشکال مختلف مانند قوانین طبقه‌بندی (If-Then)، درخت‌های تصمیم، فرمول‌های ریاضی و شبکه‌های عصبی قابل نمایش هستند. هدف در دسته‌بندی داده‌ها این است که یک مدل پیشگویی کننده بدست آوریم که این مدل اولاً توانایی دسته‌بندی داده‌های ورودی را داشته باشد و ثانیاً بتوان از آن جهت پیشگویی برای تعیین دسته‌ی یک داده که تازه به سیستم اضافه شده، استفاده نمود. در این پژوهش با استفاده چکیده و نام متون فارسی، روشی هوشمند برای دسته‌بندی نوشتارهای علمی فارسی طراحی شده و در مقیاس آزمایشگاهی پیاده‌سازی خواهد شد. دسته‌های مورد نظر در این پژوهش شامل ۵ دسته علوم انسانی، فنی و مهندسی، علوم پایه، هنر و علوم پزشکی خواهد بود.

استخراج و حذف ایست‌واژه‌ها، ریشه‌یابی کلمات و در نهایت طراحی الگوریتم هوشمند و یادگیری آن از مراحل این طرح پژوهشی خواهد بود. استفاده از تعداد تکرار کلمات درمتن، تعداد تکرار کلمات در همه مستندات آزمایشگاهی، موقعیت کلمات در جمله و همچنین نام پایان‌نامه و رساله در الگوریتم پیشنهادی مورد توجه قرار خواهد گرفت. لازم به ذکر است که خرورجی این طرح پژوهشی مدل آموزش دیده می‌باشد و نرم افزار ارائه نمی‌گردد.

۱-۳. محدودیت‌های پژوهش

از آنجاییکه روش‌های دسته‌بندی متون برای زبان‌های لاتین طراحی و توسعه یافته‌اند، استفاده از الگوریتم‌های موجود با محدودیت‌هایی روبرو است. ولی اگر بخواهیم به طور کلی محدودیت های پژوهش را نام ببریم، می‌توان به موارد ذیل اشاره نمود:

- کمبود ابزارهای پیش‌پردازش کارا در زبان فارسی
- پیچیدگی زمانی بالای الگوریتم های دسته‌بندی هوشمند متون فارسی.

فصل دوم

مروری بر مفاهیم و روش‌های استخراج ویژگی

۲-۱. مقدمه

برای پردازش نوشتارها، مجموعه پردازش‌هایی که فارغ از کاربرد مورد نظر و به منظور بهبود نوشتار در اغلب کاربردها صورت می‌گیرد، پیش پردازش اطلاق می‌شود. در این فصل، ابتدا بیان مختصری از مجموعه روش‌های پیش‌پردازش ارائه می‌گردد. همچنین روش‌های مهم استخراج ویژگی نیز در قسمت دیگر این فصل بیان می‌گردد.

۲-۲. تعاریف رسمی

تعریف کلمه (لغت یا واژه): کلمه (w) کوچکترین واحد گسسته با معنی در متن است که با یک شماره اندیس از مجموعه لغات واژه‌نامه $\{1,2,...,V\}$ نمایش داده می‌شود.

تعریف عبارت[۱]: هر توالی از کلمات کنار هم $ph=w_{i+1}w_{i+2}...w_{i+l}$ یک عبارت به طول l نامیده می‌شود که مقدار اطلاعات متقابل نقطه‌ای[۲] (PMI) آن بزرگتر از حد آستانه مشخصی باشد. به بیان دیگر، عبارت یک چند کلمه‌ای (n-گرم در واحد کلمه) با هم‌رخدادی بالا است. به عنوان مثال "محیط آموزشی" یک عبارت و " محیط ابر" با توجه به هم رخدادی کم عبارت محسوب نمی‌شود.

مقدار اطلاعات متقابل نقطه‌ای برای عبارت ph بصورت زیر محاسبه می‌گردد (Liu 2012):

$$PMI(ph) = \log \frac{freq(w_{i+1}, w_{i+2}, ..., w_{i+l})}{\prod_{j=i+1}^{i+l} freq(w_j)} \qquad (۱-۲)$$

---

[۱] Phrase

[۲] Point-wise Mutual Exclusion (PMI)

تعریف جمله : مجموعه‌ای از کلمات کنار هم که از نظر گرامری با هم در ارتباط بوده و معمولاً معنی مستقل و کامل دارد، جمله خوانده می‌شود.

تعریف متن (سند) : توالی از n کلمه که بصورت $d = \{w_1, w_2, \ldots, w_n\}$ نمایش داده می‌شود که در آن w_i نشان دهنده i–اُمین کلمه متن است.

تعریف مجموعه مستندات (پیکره) : مجموعه‌ای از m متن که بصورت $C = \{d_1, d_2, \ldots, d_m\}$ نمایش داده می‌شود که در آن d_i نشان دهنده متن i–اُم است.

۲-۳. پیش پردازش متن فارسی

استانداردسازی، تفکیک متن به جملات، عبارات و کلمات و برچسب‌گذاری و حاشیه‌نویسی آنها تاثیر بسزایی بر روی پردازش و استخراج اطلاعات، دسته‌بندی یا دیگر کاربردهای پردازش زبان طبیعی دارد. زبان فارسی یکی از زبان‌های هندو–اروپایی[۱] است که توسط بیش از صد میلیون نفر از مردم جهان صحبت شده و بعنوان زبان رسمی سه کشور ایران، افغانستان (فارسی دری) و تاجیکستان (فارسی تاجیکی) است. بدلیل پیچیدگی‌های زبانی، کمبود منابع و مطالعات انجام شده در این زبان از دیدگاه محاسباتی کمتر مورد توجه پژوهشگران قرار گرفته است (Shamsfard 2011 و Feely et al. 2014).

متاسفانه ابزارهای استاندارد پیش‌پردازش ایجاد شده برای متون زبان فارسی از قبیل (Shamsfard 2010)، (Sarabi et al. 2013) و (Seraji et al. 2012) بصورت رایگان منتشر نشده‌اند. برخی از ابزارهای کد باز[۲] پیش‌پردازش موجود از قبیل (Khallash et al. 2014)، (Jadidinejad et al. 2010) و (Manshadi 2013) نیز از دقت مناسبی برخوردار نیستند. در این بخش به بیان توضیحات ابزارهای تولید شده و مورد نیاز در نظرکاوی در زبان فارسی می‌پردازیم.

---

[۱] Indo-European

[۲] Open Source

۲-۳-۱. نرمال‌سازی متن و جداسازی جملات و کلمات متن

قبل از پردازش متون جهت استانداردسازی حروف و فاصله‌ها بایستی پیش‌پردازش‌هایی روی آن‌ها انجام شود. در واقع در این مرحله بایستی همه‌ی نویسه‌های (حروف) متن با جایگزینی با معادل استاندارد آن‌ها، یکسان‌سازی گردند. در پردازش رسم‌الخط زبان فارسی، با توجه به شباهتی که با رسم‌الخط عربی دارد، همواره در نگارش تعدادی از حرف‌ها مشکل استفاده از کاراکترهای عربی معادل وجود دارد؛ که از جمله‌ی آنها می‌توان به حروف "ک"، "ی" و همزه اشاره نمود. در اولین گام باید مشکلات مربوط به این حروف را با یکسان‌سازی آنها برطرف کرد. علاوه بر این، اصلاح و یکسان‌سازی نویسه‌ی نیم‌فاصله و فاصله در کاربردهای مختلف آن و همچنین حذف نویسه‌های اعراب، تشدید، تنوین و «ـ» که برای کشش نویسه‌های چسبان مورد استفاده قرار می‌گیرد و مواردی مشابه برای یکسان‌سازی متون (مشابه موارد اصلاح شده در ابزار PrePer) (Seraj 2013)، از اقدامات لازم قبل از شروع پردازش متن می‌باشد.

در این فاز مطابق با یک سری قاعده دقیق و مشخص، فاصله‌ها و نیم‌فاصله‌های موجود در متن برای وندهایی نظیر "ها"، "تر" و "ی" غیرچسبان (در انتهای لغات) و همچنین پیشوندها و پسوندهای فعل ساز نظیر "نمی"، "می"، "ام"، "ایم"، "اید" و موارد مشابه نیز اصلاح می‌گردند.

پس از پایان مرحله‌ی پیش پردازش متون، ابزار تشخیص‌دهنده‌ی جملات با استفاده از علامت‌های "."، "؛"، "!"، "؟"، "?"، ":" و بکارگیری برخی دستورات گرامری زبان فارسی و در نظرگرفتن حروف ربط و برخی لغات آغاز کننده‌ی جملات (از قبیل حروف ربط مانند "که"، "اساسا"، "البته"، "تا"، "اما"، "اگر"، "ولی"، "زیرا"، "سپس"، "همچنین"، "و"و "یا")، مرز جمله‌ها را تعیین می‌نماید. تشخیص‌دهنده‌ی لغات (توکن) نیز با استفاده از علامت‌های فضای خالی، "،"، ","، "-" و غیره و در نظر گرفتن اصلاحات اعمال شده در مورد پیشوندها و پسوندها در فاز قبلی، واژه‌ها را شناسایی می‌نماید. همچنین پردازش ویژه‌ای برای درنظر گرفتن توکن واحد برای کلمات اختصاری (از قبیل

A.T.R یا بی.بی.سی)، تاریخ و زمان (از قبیل 5:35 یا 2015/2/25)، اعداد اعشاری (از قبیل 5/17 یا 5.17)، آدرس صفحات وب، ایمیل و سایر عبارات و علائم خاص (جایگزینی کلمه " ا... " با کلمه اصلی آن) انجام می‌شود.

۲-۳-۲. ریشه‌یابی

ریشه‌یابی کلمات یکی از مهمترین عملیات پیش‌پردازش متون در بازیابی اطلاعات و پردازش زبان‌های طبیعی است. هدف الگوریتم‌های ریشه‌یابی، حذف پیشوند و پسوندهای کلمات و تعیین ریشه اصلی کلمه هستند. توضیحات بیشتر درباره اهمیت ریشه‌یابی و تحلیل‌های ریخت‌شناسی کلمات در استخراج دانش از متن در (Abdul-Mageed et al. 2014) وجود دارد. برخلاف زبان انگلیسی، چالش‌های مختلفی هنگام ریشه‌یابی کلمات زبان فارسی وجود دارد از جمله اینکه ضمایر می‌توانند به دو صورت جدا و متصل در جمله ظاهر شوند. البته در مورد افعال مسئله کمی پیچیده‌تر است، بطوری که علاوه بر وندهای فعلی، شخص (فاعل) و زمان جمله نیز بر روی حالت فعل تاثیرگذار هستند.
در روش‌های ریشه‌یابی فعلی در زبان فارسی، بعد از حذف وندها ممکن است معنای کلمه تغییر یابد.

۲-۳-۳. تصحیح‌کننده خطاهای املایی (خطا در تایپ کلمات)

با بررسی اولیه متون نظرات زبان فارسی می‌توان پی‌برد که علاوه بر مشکلات مربوط به شکل اختصاری یا محاوره‌ای کلمات، غلط‌های املایی زیادی سهواً یا عمداً (برای راحتی در تایپ) در متن نظرات وجود دارد. هدف از تولید این ابزار تصحیح خودکار خطاهای املایی ناشی از تایپ اشتباه کلمات در متون نظرات می‌باشد. بدین منظور به ازای کلمات بدون مفهوم (که کلمه یا ریشه‌ی آن در لیست زبان فارسی رسمی یا محاوره‌ای وجود ندارند) ، شبیه‌ترین کلمه از نظر املایی جایگزین آن می‌شود. معیار استفاده شده برای آستانه شباهت املایی تبدیل کلمات به شکل ممکن است:

- جداسازی کلمات بهم چسبیده (فزودن فاصله در وسط کلمه)

- اصلاح/تغییر تنها یک حرف با یکی از کاراکترهای مجاور آن در صفحه کلید کامپیوتر (زبان فارسی) (Kay et al. 2012)
- اصلاح حروف هم صدا از نظر تلفظ (مانند حروف : "ز"، "ذ"، "ض" و "ظ" که همگی به صدای "z" اشاره دارند).

۲-۳-۴. برچسب‌زنی ادات سخن

برچسب‌زنی نقش ادات سخن[۱] عمل انتساب برچسب‌های نحوی به واژه‌ها و نشانه‌های تشکیل دهنده یک متن است به صورتی که این برچسب‌ها نشان دهنده نقش کلمات و نشانه‌ها در جمله باشند. در زبان فارسی اغلب کلمات (حدود ۹۰٪ در پیکره بیجن‌خان (Oroumchian et al. 2006) و (Bijankhan 2004)) دارای نقشی واحد در جملات مختلف هستند. سایر واژگان از نقطه نظر برچسب‌زن نحوی دارای ابهام هستند، زیرا ممکن است کلمات در جایگاه‌های مختلف در جمله، برچسب‌های نحوی متفاوتی داشته باشند. بنابراین برچسب‌زنی نحوی، عمل ابهام‌زدایی از برچسب‌ها با توجه به زمینه (ساختار جمله) مورد نظر است.

۲-٤. روش‌های معمول استخراج بردار ویژگی‌های متن

استخراج و انتخاب ویژگی‌های مناسب[۲] از یک مجموعه داده نقش حیاتی در بهبود کیفیت و کارایی روش‌های یادگیری ماشین دارد. خصوصاً در داده‌های با تعداد ابعاد بالا مانند متون، داده‌های بیان ژنی[۳]، تصویر، صوت، ویدئو و غیره انتخاب ویژگی امری ضروری است (Saeys et al. 2007) و (Forman 2003). هدف از انتخاب ویژگی، استخراج مجموعه‌ای از ویژگی‌های مناسب از متون زبان طبیعی است که در مرحله بعد بوسیله روش‌های دسته‌بندی احساسات (مشابه روش‌های دسته‌بندی

۱- Part of Speech tagging

۲ Relevant features

۳ Gene expression data

متون) استفاده می‌شوند. بطور کلی، استخراج ویژگی با دو هدف انجام می‌شود: ۱- افزایش کارایی و سرعت روش‌های دسته‌بندی با کاهش ابعاد و اندازه داده‌ها. بخصوص جهت بکارگیری برخی روش‌های دسته‌بندی که فاز آموزش آنها هزینه و سربار زمانی یا حافظه‌ای بالایی دارند (مانند SVM)، این امر ضروری است. ۲- افزایش دقت روش‌های دسته‌بندی با حذف ویژگی‌های نویزی (که وجود آنها باعث افزایش خطای دسته‌بندی برای داده‌های جدید می‌شوند) و استخراج ویژگی‌های مناسب (که باعث نزدیک شدن داده‌های درون دسته‌ها و تمایز بیشتر بین داده‌های دسته‌های مختلف می‌شوند).

در این بخش، انواع ویژگی‌های و روش‌های انتخاب ویژگی برای تبدیل متن به بردارهای عددی قبل از دسته‌بندی توضیح داده شده است.

۲-۴-۱. بسامد کلمه – معکوس بسامد سند [۱]

این روش فرکانس کلمه را در یک سند به ما می‌دهد. معیار بهتر این است که فرکانس کلمه در سند را با مقایسه ارتباط آن با اسناد دیگر بدست بیاوریم. این معیار بسامد کلمه – معکوس بسامد سند یا به اختصار تی.اف – آی.دی.اف نام دارد که میزان خاص بودن کلمه در متن مورد نظر را نشان می‌دهد. برای محاسبه آن از فرمول زیر استفاده می‌شود (Brian 2012).

$$TFIDF(t,d,n,N) = TF(t,d) \times IDF(n,N) \qquad (۲-۲)$$

تی- اف از رابطه زیر و از تقسیم فرکانس کلمه بر تعداد کل کلمات سند بدست می‌آید. ای-دی-اف نیز از (معادله ۲-۳) محاسبه می‌شود که در آن، n تعداد اسنادی از پیکره است که دارای کلمه مورد نظر می‌باشد و N تعداد کل اسناد پیکره است.

$$TF(t,d) = \frac{m_{td}}{\sum_{k} m_{kd}} \qquad (۲-۳)$$

$$IDF(n,N) = \log \frac{(N+1)}{(n+1)} \qquad (۲-٤)$$

[۱] TF-IDF

با استفاده از این معیار کلماتی از یک سند که در اسناد دیگر کمتر استفاده شده‌اند امتیاز بالاتری دارند.

علاوه بر حالت ساده دودویی کلمات به عنوان ویژگی (بر اساس حضور کلمات در متن)، برای وزن‌دهی به کلمات انواع نسخه‌های مبتنی بر Delta-TFIDF (Paltoglou and Thelwall 2010) (از قبیل Augmented TF، LogAve TF ، BM25 TF ، BM25+ TF، Delta Smoothed IDF ، Delta Prob. IDF، Delta Smoothed Prob. IDF و Delta BM25 IDF) پیاده‌سازی و آزمایش شده است. در روش Delta-TFIDF برای هر کلاس (مثبت و منفی) بطور جداگانه از روش وزن‌دهی TFIDF ساده استفاده می‌شود (Martineau and Finin 2009).

۲-۴-۲ . بسامد کلمه- معکوس بسامد جمله[۱]

بسامد کلمه- معکوس بسامد جمله یک اندازه آماری است، که سازگاری با (TF-IDF) را به جملات متن می‌دهد. در اصطلاح TF-ISF ، استخراج کلمات کلیدی بر اساس فرکانس هر جمله از متن سند به عنوان یک بردار از وزن در نظر گرفته شده است. در اندازه گیری TF-ISF، نرخ t_k از طریق فراوانی معکوس جمله s به وسیله فراونی محاسبه می‌شود (Fiori 2014):

$$\text{TF-ISF(S)} = \sum_{t \in S} freq(t_k) \times isf\ (t_k) \qquad (۵-۲)$$

$$isf\ (t_k) = 1 - \frac{\log(n(t_k))}{\log(n)} \qquad (۶-۲)$$

جایی که n نشان دهنده‌ی تعداد جملات در مجموعه اسناد است.

[۱] TF-ISF

۲-۴-۳. ویژگی‌های چندکلمه‌ای

در اغلب کارهای مرتبط (دسته‌بندی متون) از این ویژگی استفاده شده است (Agarwal and Mittal 2016) و (Habernal et al. 2015). در این پژوهش از حضور تک‌کلمه‌ای‌ها[۱]، دوکلمه‌ای‌ها[۲] و سه‌کلمه‌ای‌های[۳] مختلف به عنوان یک گروه از ویژگی‌ها استفاده شده است. البته برای کاهش ابعاد از ابزارهای تبدیل کلمات محاوره‌ای به رسمی و ریشه‌یاب استفاده شده (Manning et al. 2008) و بصورت تجربی (با توجه به اندازه پیکره) حداقل رخداد چندکلمه‌ای‌ها[۴]، سه در نظر گرفته شده است.

۲-۴-۴. ویژگی چندکاراکتری

مشابه با ویژگی‌های چندکلمه‌ای، از ویژگی‌های چندکاراکتری[۵] نیز مطابق با رویکرد (Blamey et al. 2012) استفاده می‌شود. در این پژوهش از تعداد ۳ تا ۶-کاراکتری‌های مختلف موجود در متون نظرات به عنوان یک دسته از ویژگی‌ها استفاده شده است. همچنین بصورت تجربی (با توجه به اندازه پیکره) حداقل رخداد چندکاراکتری‌ها، پنج در نظر گرفته شده است.

۲-۴-۵. ویژگی برچسب نقش ادات سخن (**POS Tags**)

با توجه به معنی و کاربرد متفاوت کلمات در نقش‌های مختلف، در بسیاری از کارهای مرتبط از ویژگی نقش کلمات نیز برای دسته‌بندی احساسات استفاده می‌شود. در این پژوهش تنها از کلمات با برچسب‌های اسم، فعل، صفت، قید و سایر استفاده می‌شود. علاوه بر این تعداد هر یک از برچسب‌های

---

[۱] unigram

[۲] bigram

[۳] trigram

[۴] n-Gram

[۵] Character n-gram

نقش ادات سخن[۱] برای کلمات نیز به عنوان ویژگی استفاده می‌شود (مشابه (Mohammad et al. 2013)).

فصل سوم

مروری بر روش‌های دسته‌بندی تفکیکی

[۱] part-of-speech tag

۳-۱. مقدمه

با توجه به موفقیت دسته‌بندهای تفکیکی در کاربردهای گوناگون، گونه‌های مختلفی از این نوع دسته-بند گسترش یافته‌اند. رایج‌ترین دسته‌بند تفکیکی را روش ماشین بردار پشتیبان باید دانست که توسط دکتر Vapnik در سال ۱۹۹۵ ارائه گردید(V. Vapnik,1995) و خیلی زود توانست به یکی از ابزارهای قدرتمند در دسته‌بندی داده‌ها تبدیل شود. این روش برای جداسازی داده‌های دو کلاس از یکدیگر ابداع شد. از ویژگی‌های مهم این روش می‌توان به پایه‌ی قوی ریاضی آن اشاره کرد. در این روش دسته‌بند در نتیجه‌ی حل یک مسأله‌ی بهینه‌سازی با تابع هدف محدب[۱] با تعدادی قید[۲] بدست می‌آید. تعداد این قیود مسأله، به اندازه‌ی تعداد نمونه‌های یادگیری است. نتیجه‌ی حل این مسأله، یافتن ابرسطح جداکننده‌ی بهینه با حداکثر حاشیه[۳] است. یعنی تا آنجا که ممکن است دسته‌بند دور از نقاط داده‌ای هر دو کلاس باشد. در این فصل مزایا و معایب گونه های مختلف مدل‌های تفکیکی قدیمی به طور کامل بررسی می‌شوند.

---

[۱] Convex

[۲] Constraint

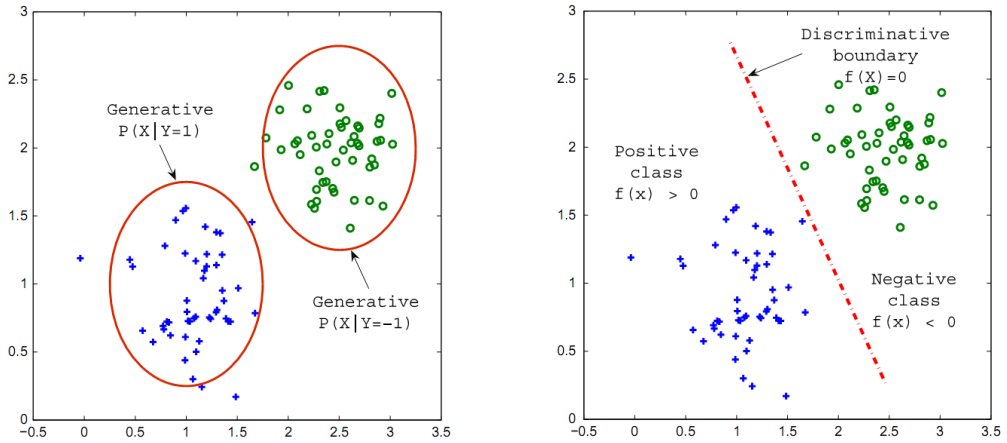
[۳] Margin

۳-۲. مقایسه دسته‌بندهای تولیدی و تفکیکی

همانطور که اشاره شد، دسته‌بندها به دو گروه تولیدی و تفکیکی تقسیم می‌شوند.دسته‌بندهای تولیدی بر پایه‌ی تخمین احتمال تعلق نمونه به کلاس می‌باشند در حالی که در دسته‌بندهای تفکیکی بر پایه مرز بین دو کلاس تصمیم گیری انجام می‌شود. در (شکل ۱) تفاوت این دو مدل نشان داده شده است. هر کدام از مدل‌ها دارای مزایای خاص خود می‌باشند. دسته‌بندهای تفکیکی به طور کلی دارای دقت بالاتری نسبت به دسته‌بندهای تولیدی دارند ولی در صورتی که داده‌ها دارای نویز باشند مدل‌های تولیدی بهتر از مدل‌های تفکیکی خواهند بود. مدل‌های تولیدی برای داده‌های نیمه برچسب خورده می‌تواند استفاده شود ولی سرعت تخمین در مدل تفکیکی بالاتر است (Adankon and Cheriet 2011). در جدول ۳-۱ به طور خلاصه دسته‌بندهای تولیدی و تفکیکی مقایسه شده‌اند (Ng and Jordan 2002). توانایی استفاده مستقیم از دانش اولیه در مدل تولیدی وجود دارد که در مدل تفکیکی به آسانی امکانپذیر نیست. مدل‌های تفکیکی از عمومیت بالاتری نسبت به مدل دیگر برخوردار هستند. مدل تفکیکی به رویکردهای حاشیه موازی و ناموازی تقسیم می‌گردند که در ادامه هریک مورد مطالعه قرار می‌گیرند.

جدول ۱. مقایسه دسته‌بند تولیدی و تفکیکی

نوع مدل	دقت (بطورکلی)	دقت (داده نادقیق)	تعداد کلاس‌های زیاد	استفاده مستقیم از دانش اولیه	قابلیت افزایش کلاس	سرعت آموزش
دسته‌بند تولیدی	متوسط	متوسط	بله	بله	بله	کم
دسته‌بندتفکیکی	زیاد	کم	خیر	خیر	خیر	متوسط



شکل ۱. مصورسازی دسته‌بند تفکیکی در مقابل دسته‌بند تولیدی الف)مدل تفکیکی. ب) مدل تولیدی (Adankon and Cheriet 2011)

۳-۳. رویکردهای حاشیه موازی

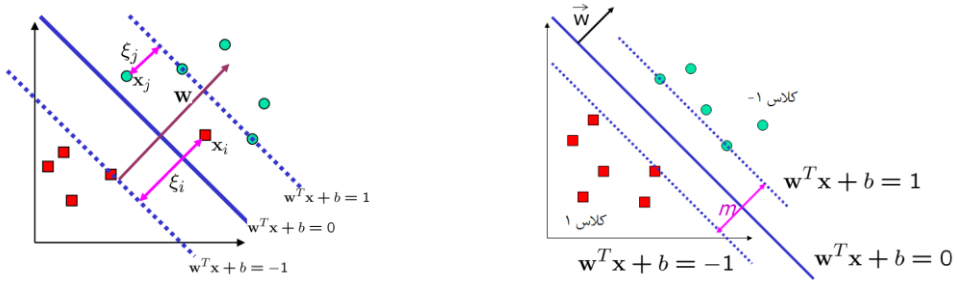
۳-۳-۱. روش Classic SVM

روش ماشین بردار پشتیبان برای جداسازی داده‌های دو کلاس از داده‌ها ابداع شده است. فرض کنید مجموعه‌ی $X = \{(x_i, y_i)\}_{i=1}^{n} c$ را در اختیار داشته باشیم که n تعداد نمونه‌ی آموزش سیستم است. هر $x_i \in \mathcal{R}^m$ یک نمونه‌ی یادگیری با m ویژگی در فضای ورودی[۱] است و $y_i \in \{-1,1\}$ برچسب کلاس داده‌ی x_i است. یعنی هر نمونه متعلق به کلاس ۱ یا ۱- است. در ساده ترین حالت نقاط داده‌ای دو کلاس مجموعه ی X، کاملاً جدای از هم قرار گرفته‌اند طوری که می‌توان از یک دسته‌بند خطی برای جدا کردن آنها از هم استفاده کرد. (شکل ۱) این وضعیت را نشان می‌دهد. برای بدست آوردن بهترین خط مسئله بهینه‌سازی زیر در ساده‌ترین حالت باید حل گردد.

$$Minimize \ \frac{1}{2}\|w\|^2$$
$$subject\ to\ y_i(w^T.x_i + b) \geq 1 \quad i = 1, \dots, n \qquad (۱-۳)$$

[۱] Input space

هدف بدست آوردن یک سطح جداکننده برای دو کلاس از داده‌های جداپذیر خطی است که بیشترین فاصله‌ی ممکن از داده‌های دو کلاس را داشته باشد. ماکزیمم کردن m (شکل ۱ الف: ناحیه‌ای که بین خطوط نقطه‌چین قرار دارد) ما را به این هدف می‌رساند. درنتیجه خطوطی که با نقطه‌چین نمایش داده شده‌اند، به محض رسیدن به نقاطی از دو کلاس از پیشروی باز می‌مانند. نقاط داده‌ای قرار گرفته بر روی خطوطی که با نقطه‌چین نمایش داده شده‌اند را *بردار پشتیبان*[1] می‌نامند. همان‌طور که گفته شد، هدف SVM ماکزیمم کردن حاشیه‌ی m است. به این نوع مسأله‌ی حاشیه‌ی سخت گفته می‌شود.



الف: مسأله‌ی حاشیه‌ی سخت

ب: مسأله‌ی حاشیه‌ی نرم و خطای در دسته بندی

شکل ۲. نمایش هندسی ماشین بردار پشتیبان

در حالت هایی که داده‌ها به صورت خطی جداپذیر نباشند از مسئله حاشیه نرم و سطح تصمیم غیرخطی استفاده می‌شود.

۱) مسأله‌ی حاشیه‌ی نرم

در گونه دیگر که به *مسأله‌ی حاشیه‌ی نرم* معروف است اجازه‌ی خطا در دسته‌بندی داده می‌شود. در حالتی که تعداد اندکی از نقاط داده‌ای یک کلاس، طوری داخل نقاط داده‌ای کلاس دیگر قرار گرفته‌اند که به راحتی حالت قبل نمی‌توان یک صفحه بهینه پیدا کرد که به صورت کامل دو کلاس را از هم جدا کند. در این مورد همان طور که (در شکل ۱ب) نشان داده شده است، با استفاده از متغیر کمکی $\xi = (\xi_1, \xi_2, \dots, \xi_n)$ که در واقع به هر نقطه‌ی داده‌ای متناظر می‌شود، خطا در دسته‌بندی،

[1] Support Vector (SV)

اجازه داده می‌شود. ξ_i یک متغیر کمکی در بهینه‌سازی است. بدیهی است که اگر $\xi_i = 0$ یعنی داده ی x_i درست دسته‌بندی شده و در کلاس خودش قرار دارد. مسأله‌ی پیدا کردن صفحه بهینه در این حالت، در گرو حل مسأله‌ی بهینه‌سازی زیر است:

$$Minimize \ \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i \qquad (۲-۳)$$
$$subject\ to \ \ y_i(w^T x_i + b) \geq 1 - \xi_i$$

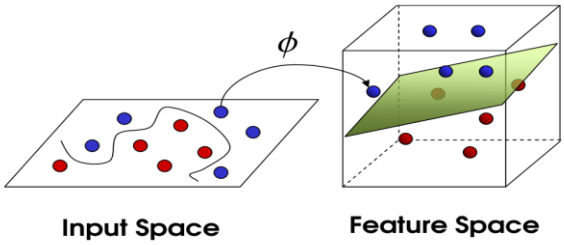
هدف از قرار دادن عبارت $C\sum_{i=1}^{n}\xi_i$ این است که تعداد نقاطی که اشتباه دسته‌بندی می‌شوند را تحت کنترل داشته باشیم و به آن ضریب پنالتی می‌گویند. درواقع C یک پارامتر تنظیم است که در ابتدا آن را مقداردهی می‌کنیم. تنظیم این پارامتر می‌تواند یک تعادل را بین بیشینه کردن ناحیه‌ی بین دو کلاس و خطای در طبقه‌بندی را سبب شود.

۲) سطح تصمیم غیرخطی

اگر داده ها به صورت غیر خطی جداپذیر باشند توسط روش‌های گفته شده طبقه بندی نمی‌شوند. یک راه حل برای این کار این است که نقاط x_i را به یک فضای با ابعاد بیشتر تبدیل کنیم و در آنجا یک سطح تصمیم‌گیری خطی پیدا کنیم. پس دو تعریف زیر را ارائه می دهیم:

فضای ورودی: فضایی که نقاط x_i در آن واقع شده‌اند.

فضای ویژگی: فضایی که x_i ها بعد از اعمال تابعی مثل $\varphi(x_i)$ به آنها، ایجاد می‌شود و معمولاً با ابعاد بیشتر است. (شکل ۳) اصول این کار را نشان می‌دهد.



شکل ۳. تبدیل فضا برای یافتن یک جداکننده‌ی خطی

با توجه به اینکه عملیات خطی در فضای تصویر معادل عملیات غیرخطی در فضای ورودی است، پس عمل تبدیل باعث می‌شود که نوع عملیات راحت‌تر شود. از طرف دیگر عمل تبدیل باعث می‌شود تا عمل دسته‌بندی در فضای تصویر به راحتی انجام شود در حالی که این عمل در فضای ورودی به سختی انجام می‌شد.

باید به این نکته توجه داشت که فضای تصویر در عمل ابعاد بزرگتری نسبت به فضای ورودی دارد. مثلاً ممکن است دارای ابعاد بی‌نهایت باشد. برای حل این مسأله و فرار از مشکلی که به آن اشاره شد، از روشی به نام حقه‌ی هسته[۱]، استفاده می‌شود. همچنین یکی از نقاط ضعف روش،حساسیت بسیار زیاد نسبت به نمومه‌های نادقیق و پرت میباشد که برای حل آن گسترش‌های گوناگونی بوجود آمده است.

۳-۳-۲. روش FSVM[۲]

این روش سعی در برطرف کردن حساسیت SVM دارد. در بسیاری از کاربردهای واقعی، تأثیر نقاط مختلف برای یک سیستم یادگیری ماشین متفاوت است. غالب اوقات تعدادی از داده‌های آموزش اهمیت بیشتری نسبت به سایرین دارند. داده‌هایی که اهمیت بیشتری دارند باید بصورت کاملاً درست دسته بندی شوند و برای داده‌های با اهمیت کم، مثلاً داده‌های نویزی، می‌توانند بصورت اشتباه هم

[۱] Kernel trick

[۲] Fuzzy Support vVector Machines

دسته‌بندی شوند. هر داده ممکن است کاملاً به یک کلاس تعلق نداشته باشد. مثلاً ممکن است ۹۰٪ به یک کلاس تعلق داشته باشد و ۱۰٪ از دید آن کلاس، بی‌اهمیت باشد. در این روش یک عضویت فازی $0 < s_i \leq 1$ به هر نقطه‌ی داده‌ای x_i اختصاص می‌یابد (Lin and Wang 2002). این تعلق فازی در واقع میزان گرایش داده به یک کلاس با s_i و میزان بی‌اهمیت بودن داده از دید آن کلاس با $1 - s_i$ است. حال می‌بایست SVM را با تعریف تعلق‌های فازی بازنویسی کرد. فرض کنید مجموعه‌ی X شامل داده‌های برچسب خورده با تعلق فازی باشند:

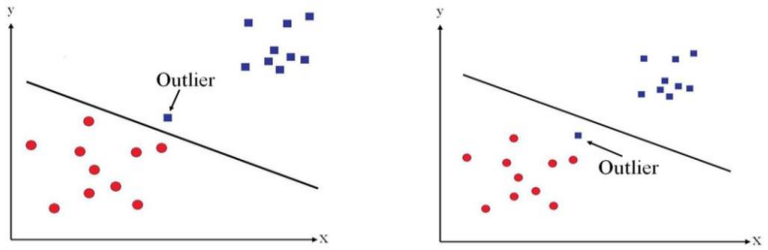
$$(y_1, x_1, s_1), \dots, (y_n, x_n, s_n)$$

هر داده‌ی آموزش $x_i \in \mathcal{R}^N$ دارای یک برچسب $y_i \in \{-1,1\}$ و یک تعلق فازی $\sigma < s_i \leq 1$ با σ به اندازه‌ی کافی کوچک، می‌باشد. همچنین $z = \varphi(x)$ تابع نگاشت فضای ورودی $\mathcal{R}^N$ به فضای ویژگی Z است.

چون s_i میزان گرایش داده‌ی x_i به یک کلاس و ξ_i میزان خطای SVM است، $s_i\xi_i$ یک میزان خطا با وزن‌های مختلف (برای داده‌های مختلف) است. ابرسطح بهینه نیز بصورت:

$$\begin{aligned} &Minimize\ \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n} s_i\xi_i \\ &subject\ to\ y_i(w^T.z_i + b) \geq 1 - \xi_i,\ \ i = 1, \dots, n \\ &\xi_i \geq 0, \quad i = 1, \dots, n \end{aligned} \qquad (۳\text{-}۳)$$

بازنویسی می‌شود که C یک مقدار ثابت است. قابل تذکر است که s_i کوچک‌تر، تأثیر پارامتر ξ_i را بیشتر کرده در نتیجه داده‌ی متناظر x_i اهمیت کمتری می‌یابد. مسأله‌ای که باقی می‌ماند، بدست آوردن تابعی است که تعلق هر داده به کلاسش (s_i) را مشخص کند. در SVM تنها پارامتر آزاد ثابت C است ولی در FSVM این تعداد برابر نمونه‌های آموزش است. روش‌های زیادی برای بدست آوردن مقدار اهمیت آمده است.



شکل ٤. نحوه‌ی برخورد روش‌های FSVM (الف) و SVM (ب) در حضور یک داده‌ی پرت[۲۸].

(ب) **(الف)**

برای محاسبه درجه تعلق فازی راه های گوناگونی وابسته به کاربرد ارائه شده است. در روش پایه که در (Lin and Wang 2005) و (Tang and Qu 2008) به‌کار گرفته شده بدین صورت است که تعلق فازی تابعی از فاصله‌ی بین هر داده تا مرکز دسته‌اش است. فرض می‌کنیم نقاط ورودی بصورت $(y_1, x_1, s_1), \ldots, (y_n, x_n, s_n)$ و مرکز دسته‌ی کلاس ۱، x_+ و مرکز دسته‌ی کلاس ۱-، x_- باشد. اگر شعاع کلاس ۱ را $r_+ = \max_{\{x_i:\, y=1\}} |x_+ - x_i|$ و شعاع کلاس ۱- را $r_- = \max_{\{x_i:\, y=-1\}} |x_- - x_i|$ در نظر بگیریم، می‌توان تعلق فازی را به عنوان تابعی از میانگین و شعاع هر کلاس تعریف نمود:

$$s_i = \begin{cases} 1 - |x_+ - x_i|/(r_+ + \delta) & if\ y_i = 1 \\ 1 - |x_- - x_i|/(r_- + \delta) & if\ y_i = -1 \end{cases} \qquad (۴-۳)$$

که $\delta > 0$ برای جلوگیری از حالت $s_i = 0$ بکار می‌رود. (شکل ۴) نحوه‌ی برخورد روش‌های FSVM و SVM را در حضور یک داده‌ی پرت، نشان می‌دهد. برتری روش FSVM در کاربردهای گوناگون نشان داده شده است (Heo and Gader 2009)، (Hsu et al. 2009) و (Zhao et al. 2010).

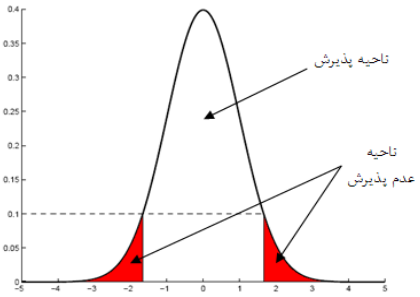
۳-۳-۳. PC-SVM

در این روش ابتدا برای هر کلاس یک توزیع در نظر می‌گیرد (Yazdi et al. 2007). در صورتی که دانش قبلی برای توزیع داده‌های یک کلاس وجود نداشته باشد از توزیع نرمال استفاده می‌شود. مقدار

اهمیت هر داده بر اساس توزیع در نظر گرفته شده محاسبه می‌گردد. قیدهای موجود در SVM کلاسیک به حالت قیدهای احتمالاتی تبدیل می‌شود. مسئله پیدا نمودن ابر سطح بهینه با حل مسئله بهینه‌سازی زیر بدست خواهد آمد.

$$Minimize \ \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i \qquad (۳,۵)$$
$$subject\ to \ \Pr(y_i(w^T x_i + b) \geq u_i - \xi_i) \geq \delta_i$$

در فرمول بالا u_i یک متغیر تصادفی مستقل با تابع توزیع مشخص و $0 \leq \delta_i \leq 1$ مقدار تاثیر نمونه iام در مشخص شدن ابر سطح بهینه را نشان می‌دهد. در این حالت داده‌هایی که از مرکز دسته دور شوند ارزش کمتری می‌گیرند و تاثیر کمتری در تصمیم گیری خواهند داشت. در (شکل ۵) مفهوم احتمال و دور بودن از مرکز نشان داده شده است. در این روش می‌توان نمونه‌هایی که از حدی مشخصی از میانگین دور باشند را به طور کلی حذف نمود.



شکل ۵. مدل گوسی یک بعدی

کاربرد این روش در تشخیص اعداد دست‌نویس و برتری آن بر ماشین بردار پشتیبان کلاسیک در (Yazdi et al. 2007) نشان داده شده است.

۳-۳-۴. روش RSVM[۱]

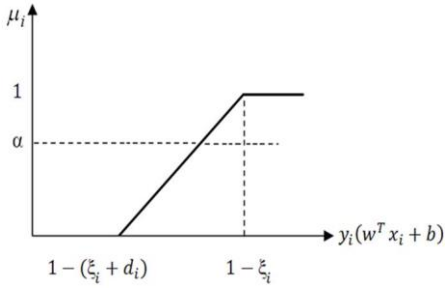
در روش RSVM به قیدهای مسأله کمی انعطاف بیشتری داده می‌شود (Sabzekar et al. 2011). برای این منظور قیدهای مساله کلاسیک را با نامساوی جایگزین کرده‌اند. علامت $\gtrsim$ به این معنی است

[۱] Relaxed Support Vector Machines

که برای برخی نمونه‌های یادگیری اجازه‌ی تخطی از حدود هم داده می‌شود. برای قیود کم اهمیت‌تر، که به معنی نمونه‌های کم اهمیت‌تر نیز می‌باشد، تخطی بیشتر از حدی که در فرمول اصلی مشخص شده انجام خواهد شد اما برای داده‌های با اهمیت بالا میزان این تخطی کم و ناچیز می‌باشد و جواب‌های مسأله می‌بایست در فضای همان قید جستجو شود. مساله بهینه سازی RSVM در زیر آمده است:

$$Minimize\ Q(w,b,\xi) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i$$
$$subject\ to\ \ y_i(w^T x_i + b) \gtrsim 1 - \xi_i,\ \ i = 1,\dots,n \qquad (۶-۳)$$
$$\xi_i \geq 0, i = 1,\dots,n$$

یک تابع عضویت[۱] برای نامساوی فازی موجود در قیود مسأله در نظر گرفته شده است. این تابع را μ_i نامیده و آن را بصورت خطی و با توجه به (شکل ۶) تعیین می‌شود.



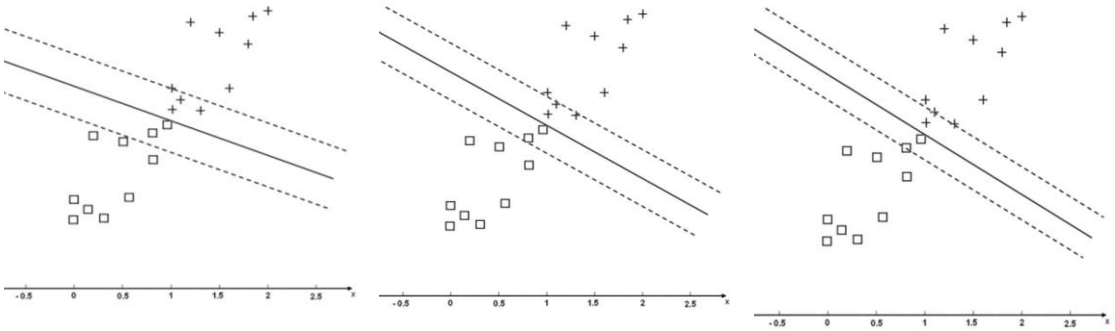
شکل ٦. تابع عضویت μ_i.

در نهایت مدل RSVM با نامساوی غیر فازی به صورت زیر بدست می‌آید:

$$Minimize\ \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i \qquad (۷-۳)$$
$$subject\ to\ y_i(w^T x_i + b) \geq 1 - \xi_i - d_i(1-\alpha), i = 1,\dots,n$$
$$\xi_i \geq 0, \qquad i = 1,\dots,n.$$

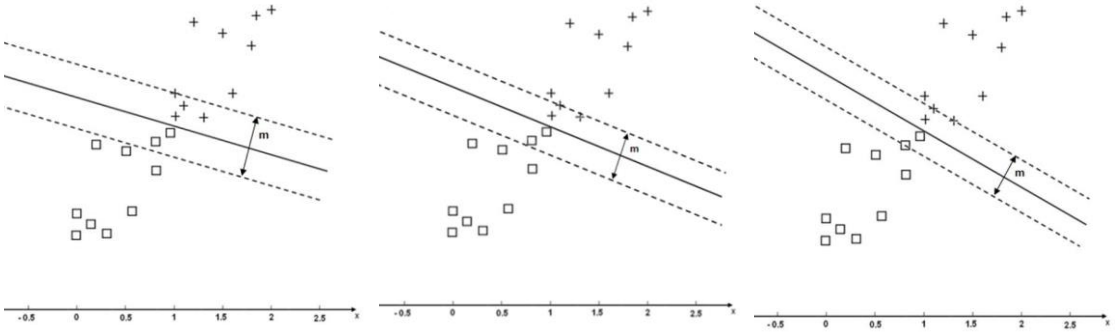
[۱] Membership function

در این روش پارامتر α میزان اطمینان از داده‌های هر کلاس و پارامترd بیان کننده میزان تحملی است که به ازای هر داده در نظر گرفته می‌شود. بررسی تاثیر مقادیر مختلف d وα در شکل‌های زیر مشخص شده است: (شکل ۷) و (شکل ۸) تأثیر تحمل را برای مجموعه‌ی ساختگی از داده‌ها نشان می‌دهد.



(الف) بدون تحمل ($d = 0$) (ب) $d = 0.1$ (ج) $d = 0.3$

شکل ۷. بررسی تأثیر تحمل در (α=0.7) RSVM.



(الف) $\alpha = 0.9$ (ب) $\alpha = 0.7$ (ج) $\alpha = 0.5$

شکل ۸. بررسی تأثیر مقادیر مختلف فاکتور اطمینان.

همانطور که ملاحظه می‌شودکم‌کردن فاکتور اطمینان برای داده‌های نامطمئن سبب کاهش تأثیر نویز در بدست آوردن صفحه تصمیم بهینه خواهد شد. در کاربرد تشخیص آریتمی‌های قلبی مزیت این روش بر حالت کلاسیک نشان داده شده است (Sabzekar et al. 2011).

۳-٤. رویکردهای حاشیه ناموازی

۳-۴-۱. روش Twin SVM

ماشین بردار پشتیبان دوقلو یک رویکرد جدید از روش های مبتنی بر مرز است که در سال ۲۰۰۷ ارایه شد (Khemchandani and Chandra 2007). در این روش به جای اینکه یک خط بهینه بدست آید، دو خط مجزا بدست می‌آید.این خطوط با یکدیگر موازی نیستند. در واقع هر خط بیانگر معیار برای همان کلاس است. در این روش برای بدست آوردن خط مسئله بهینه سازی به صورت زیر تغییر می‌کند. در این حالت هدف پیدا نمودن خطی برای هر کلاس است که کمترین فاصله از داده‌های کلاس خود و حداکثر فاصله از داده های کلاس مخالف باشد. فاصله داده‌های کلاس مخالف اگر از خط کمتر از یک باشد به عنوان خطا در نظر گرفته می‌شود. به زبان دیگر مسئله بهینه سازی یک تعادل بین کمینه کردن فاصله نقاط کلاس خود و کمینه کردن مقدار خطا (فاصله داده‌های کلاس مخالف اگر از خط کمتر از یک باشد) خواهد بود که این تعادل با پارامتر C_2و C_1 مشخص می‌شود. در زیر دو مسئله بهینه‌سازی با داده های کلاس + و با داده‌های کلاس − نشان داده شده است.

$$min_{w^+,b^+}\frac{1}{2}(Aw^+ + eb^+)^T(Aw^+ + eb^+) + C_1 e^T y$$
$$subject\ to\ -(Bw^+ + eb^+) + y \geq e\ , y \geq 0e. \qquad (۸-۳)$$

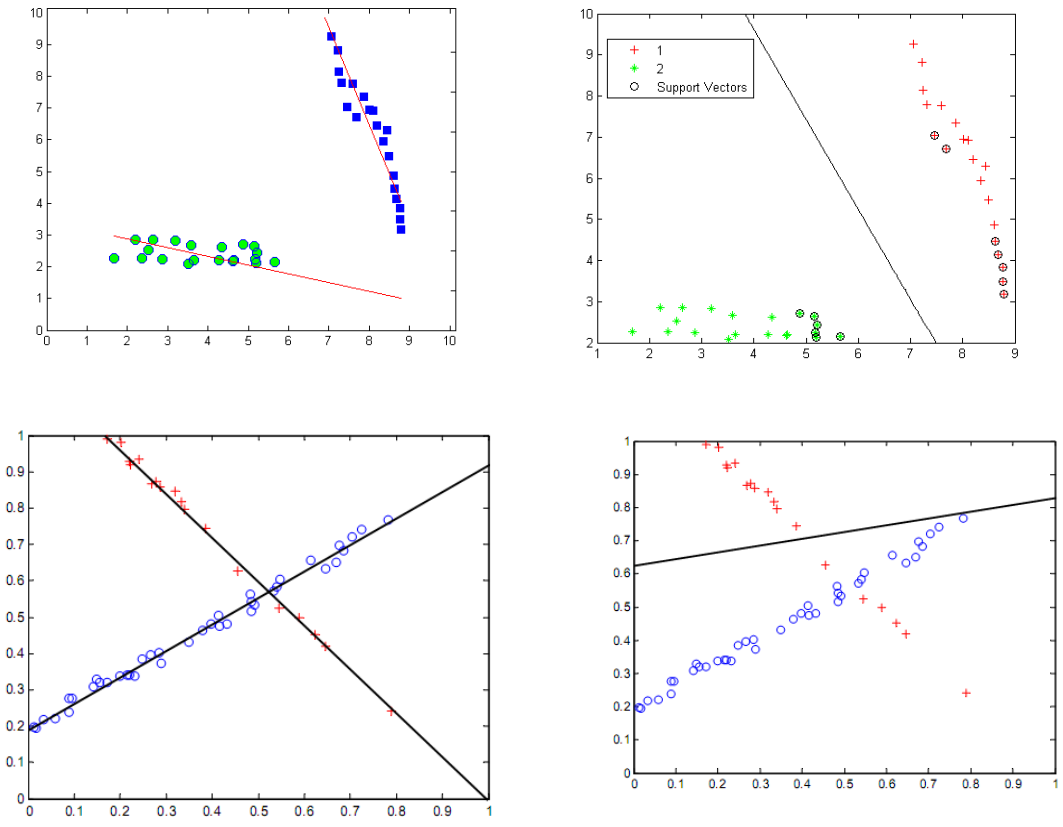
$$min_{w^-,b^-}\frac{1}{2}(Bw^- + eb^-)^T(Bw^- + eb^-) + C_2 e^T y$$
$$subject\ to\ \ (Aw^- + eb^-) + y \geq e\ , y \geq 0e. \qquad (۹-۳)$$

ماترس A و B داده های کلاس + و -، w^+و b^+ ماتریس وزن برای خط مربوط به دادهای + است. نتایج تحقیقات نشان داده است که سرعت حل مسئله بهینه سازی TWSVM نسبت SVM در حدود

چهار برابر سریعتر است در عبارت زیر تعداد داده ها m فرض شده است در نتیجه مرتبه زمانی SVM، m^3 و مرتبه زمانی TWSVM با توجه به نصف شدن قیود برای هر مسئله $O\left(\frac{m}{2}\right)^3$ است در نتیجه داریم:

$$\left[m^3/\left[2\times\left(\frac{m}{2}\right)^3\right]\right]=4$$

دقت این روش طبقه بندی نسبت به روش SVM مناسب بوده و این روش به داده‌های نویز حساسیت کمتری دارد. در شکل زیر تفاوت هندسی بین دو روش نشان داده شده است. اخیراً استفاده از این روش در کاربردهای متنوع در حال گسترش می‌باشد (Naik et al. 2011)، (Mozafari et al. 2011)، (Mozafari et al. 2011) و (Nie and He 2010).



الف: ماشین بردار پشتیبان ب : ماشین بردار پشتیبان دوقلو

شکل ۹. بررسی تفاوت SVM با TWSVM (Naik et al. 2011)

۳-۴-۲. روش LS-TSVM

گونه‌ی جدیدی از ماشین بردار پشتیبان دوقلو در سال ۲۰۰۹ به نام LS-TWSVM ارائه شده است (Kumar and Gopal 2009). این روش که مانند TWSVM بر اساس حاشیه ناموازی استوار است بسیار سریعتر ازآن است. LS-TWSVM دو مسئله بهینه‌سازی به مسائل خطی تبدیل می‌شوند به زبان دیگر در این روش اساساً بهینه سازی وجود ندارد. برای این منظور با مساوی قرار دادن قید در حالت TWSVM، متغیر y بدست آورده و قید حذف می شود. فرمول بندی در زیر آمده است.

$$min_{w^+,b^+}\frac{1}{2}(Aw^+ + eb^+)^T(Aw^+ + eb^+) + \frac{C_1}{2}y^Ty \qquad (۱۰-۳)$$
$$subject\ to\ -(Bw^+ + eb^+) + y = e$$

$$min_{w^-,b^-}\frac{1}{2}(Bw^- + eb^-)^T(Bw^- + eb^-) + \frac{C_1}{2}y^Ty \qquad (۱۱-۳)$$
$$subject\ to\ \ (Aw^- + eb^-) + y = e$$

بعد از تبدیل کردن نامساوی به مساوی، عمل جایگزینی انجام می‌شود. با استفاده از مشتق‌گیری نسبت به w و b و مساوی صفر قرار دادن مقادیر بهینه w,b بدست می‌آید.

$$+\frac{C_1}{2}\|Bw^- + eb^- + e\|^2 min_{w^+,b^+}\frac{1}{2}\|Aw^+ + eb^+\|^2 \qquad (۱۲-۳)$$

$$+\frac{C_1}{2}\|Aw^+ + eb^+ + e\|^2 min_{w^-,b^-}\frac{1}{2}\|Bw^- + eb^-\|^2 \qquad (۱۳-۳)$$

$$\begin{bmatrix} w^+ \\ b^+ \end{bmatrix} = -(F^TF + \frac{1}{C_1}E^TE)^{-1}F^Te_2 \qquad (۱۴-۳)$$

$$\begin{bmatrix} w^- \\ b^- \end{bmatrix} = -(E^TE + \frac{1}{C_1}F^TF)^{-1}E^Te_1 \qquad (۱۵-۳)$$

که در عبارات بالا $E=[Ae_1]$ و $F=[Be_2]$ می‌باشد. در کاربرد شناسایی سرطان سینه و شناسایی اعمال انسان نتایج چشمگیری نسبت به دیگر مدل‌های تفکیکی دست یافته است (Naik et al. 2011) و (Kumar and Gopal 2009).

۳-۴-۳. روش TBSVM

نسخه جدید دیگر اتز دسته‌بندهای تفکیکی بر پایه حاشیه ناموازی TBSVM می‌باشد این روش در سال ۲۰۱۱ ارائه شده است (Shao et al. 2011). هدف ازآن ارائه روشی است که مزایای TWSVM وSVM را داشته باشد. برای این منظور با دقت در روش TWSVM به وضوح مشخص می شود که مزیت اصلی این روش سرعت بالا و انعطاف آن در برخورد با چیدمان داده های متفاوت می‌باشد. همچنین حساسیت کمتر به داده‌های نویز از مزایای دیگر این روش است. روش SVM با استفاده از مفهوم ماکزیمم ناحیه به عنوان یک روش قدرتمند برای داده‌های تست معرفی می‌شود. برای این منظور در مسئله بهینه‌سازی یک عبارت جدید وارد شده که بیانگر فاصله داده‌های + تا داده‌های − است که به نوعی همان ماکزیمم ناحیه را تداعی می کند. فرمول‌های بهینه‌سازی این روش در زیر آمده است.

(۱۶-۳)

$$\begin{aligned} &min_{w^+,b^+}\frac{1}{2}(Aw^+ + eb^+)^T(Aw^+ + \\ &(\|w^+\|^2 + \|b^+\|^2)\, eb^+) + C_1 e^T y + \frac{1}{2}C_3 \\ &subject\ to\ -(Bw^+ + eb^+) + y \geq e\ , y \\ &\geq 0e. \end{aligned}$$

(۱۷-۳)

$$\begin{aligned} &min_{w^-,b^-}\frac{1}{2}(Bw^- + eb^-)^T(Bw^- + \\ &(\|w^-\|^2 + \|b^-\|^2)\, eb^-) + C_2 e^T y\ + \frac{1}{2}C_4 \\ &subject\ to\ \ (Aw^- + eb^-) + y \geq e\ , y \\ &\geq 0e. \end{aligned}$$

۳-۴-۴. Knowledge Based LS-TWSVM

طبقه‌بندی با دانش قبلی یکی از زمینه‌های رو به افزایش می‌باشد (Kumar et al. 2010) به طور مثال داده‌های پزشکی که فقط دانش متخصص وجود دارد و داده‌های آموزش وجود ندارد نمونه‌ای از کاربرد های این زمینه می‌باشد. اخیرا روش KB-SVM ارایه شده است. براساس روش استفاده شده KB-SVM روش KBLS-TWSVM پدید آمده است. معمولا دانش را به صورت زیر نشان می‌دهند:

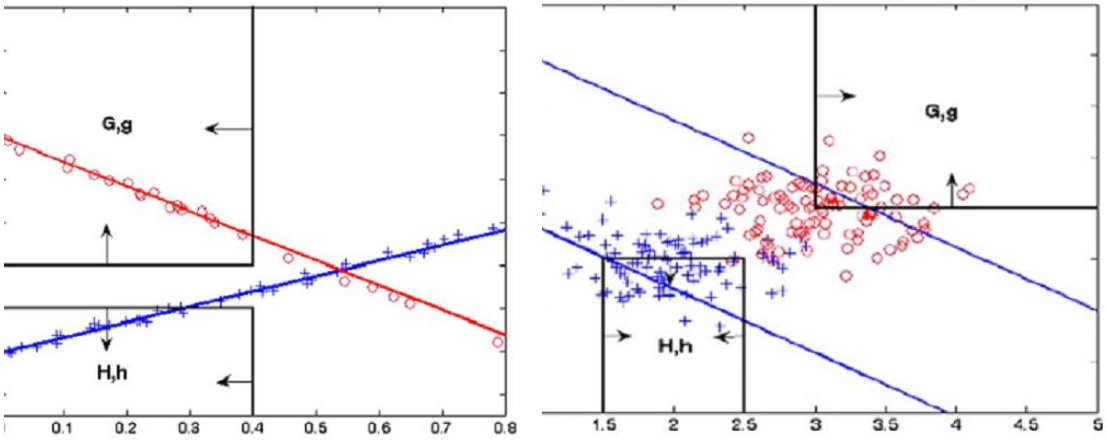
$$\begin{aligned} &l_t \quad \text{sets belonging to class } +1 : \{x|H^i x \leqslant h^i\}, \quad i = 1, 2, \ldots, l_t, \\ &k_t \quad \text{sets belonging to class } -1 : \{x|G^j x \leqslant g^j\}, \quad j = 1, 2, \ldots, k_t. \end{aligned} \tag{۱۸-۳}$$

به زبان دیگر مجموعه‌ای از ویژگی‌ها اگر دارای محدودیت خاصی باشند این نمونه طبق دانش پیشین به کلاسی خاص تعلق دارد. مثلا اگر ما بدانیم کسی فشار خونش بالای ۲۰ باشد، این فرد حتما مریض است. حال اگر ما بخواهیم این دانش را در پیدا نمودن صفحه تصمیم بهینه دخیل کنیم از روش KBLS-TWSVM استفاده می‌شود.

فرمول بندی این روش به صورت زیر است :

$$\begin{aligned} \underset{w^{(1)},b^{(1)}}{\text{Min}} \quad & \frac{1}{2}\|Aw^{(1)} - eb^{(1)}\|_2^2 + c_1 e' q_1 + d_1(e' r_1 + p_1) \\ \text{subject to} \quad & -(Bw^{(1)} - eb^{(1)}) + q_1 \geqslant e, \quad q_1 \geqslant 0e, \\ & -r_1 \leqslant G' u_1 - w^{(1)} \leqslant r_1, \quad u_1 \geqslant 0e, \; r_1 \geqslant 0e, \\ & g' u_1 - b^{(1)} + 1 \leqslant p_1, \quad p_1 \geqslant 0. \end{aligned} \tag{۱۹-۳}$$

نتایج بدست آمده در شکل های زیر مشخص شده است. در شکل زیر فلش‌ها دانش متخصص را نشان می‌دهد.



الف : نمایش گرافیکی با اعمال KBLS-TWSVM

ب : نتایج برای پایگاه داده Cross با اعمال KBLS-TWSVM

شکل ۱۰. نمایش هندسی KBLS-TWSVM (Kumar et al. 2010)

۳-۵. مقایسه رویکردهای حاشیه موازی با ناموازی

به طور کلی هر دو روش حاشیه موازی نسبت به داده‌های نویزی و پرت حساس می‌باشند (Sabzekar et al. 2011). گسترش روش SVM جهت مقاوم‌سازی آن نسبت به نمونه‌های پرت یک رویکرد مورد توجه برای محققان بوده است (Yazdi et al. 2007)، (Nasiri et al. 2009)، (Sabzekar et al. 2011) و (Khemchandani and Chandra 2007). مزیت اصلی روش‌های حاشیه ناموازی سرعت و دقت نسبتا بالاتر آن در مقایسه با روش‌های بر پایه حاشیه موازی می‌باشد. ولی همچنان این روش‌ها نیز از داده‌های نویزی تاثیرات شدیدی را متحمل می‌شوند. سرعت بالا در روش‌های ناحیه ناموازی از آنجا حاصل می‌شود که به جای اینکه تمام نمونه‌ها در قید مساله بهینه‌سازی وجود داشته باشند تنها داده‌های یک کلاس حضور خواهند داشت. همچنین براساس اینکه در این روش‌ها محدودیت موازی بودن حاشیه وجود ندارد سطح تصمیم مناسبتری می‌تواند بدست آید که نتیجه آن دقت بالاتر این‌گونه از روش‌ها خواهد بود.

۳-۶. جمع‌بندی

یکی از مشکلات اساسی در دسته‌بندهای SVM این است که تمام داده‌هایی که در آموزش SVM دخالت دارند، تأثیر یکسانی در فاز آموزش سطح تصمیم دارند. از طرفی روش کلاسیک که قبلاً اشاره شد، بصورت صفر و یک به داده‌های آموزش می‌نگریستند. یعنی یک داده یا نرمال است و یا نویزی. یک راه برای حل این مشکل این است که به داده‌های آموزش درجه‌ی اهمیت اختصاص بدهیم. به این ترتیب داده‌ی با اهمیت کمتر نقش کمتری در مشخص کردن دسته‌بند دارد ولی تأثیر آن نادیده گرفته نمی‌شود. در چنین سیستمی می‌توان به داده‌های پرت و نویزی اهمیت کمتری اختصاص داد. به این ترتیب مشکلاتی که روش‌های حذف داده‌های غیر نرمال داشتند را نیز برطرف کرده‌ایم. در این فصل روش‌های گسترش و مقاوم‌سازی SVM نسبت به داده‌های نویز و پرت بررسی شد.

مشکل دیگر روش SVM متکی بودن به یک حاشیه موازی است. به زبان دیگر محدودیت حاشیه موازی باعث کمتر شدن انعطاف این روش شده است. گونه جدیدی از این نوع دسته‌بند که بر پایه حاشیه ناموازی استوار هستند جهت فائق آمدن به این محدودیت در حال گسترش است. سرعت، مزیت دیگر دسته‌بندهای حاشیه ناموازی نسبت به دسته‌بند های حاشیه موازی است. ولی رویکردهای حاشیه ناموازی نیز نسبت به نمونه‌های نادقیق حساس می‌باشند. به نظر می‌رسد گسترش و مقاوم‌سازی ماشین بردار پشتیبان دوقلو برای بسیاری از کاربردها از جمله شناسایی اعمال انسان می‌تواند مفید باشد.

فصل چهارم

چارچوب پیشنهادی

٤-١. مقدمه

در این فصل به بررسی چارچوب و روش پیشنهادی پرداخت می‌شود. همان‌طور که در (شکل ۱۱) مشاهده می‌کنید، تمامی مراحل اصلی انجام شده در ۵ گام می‌باشد. در گام اول به دریافت داده‌های ورودی پرداخته می‌شود. در گام دو پیش‌پردازش داده‌های ورودی را انجام می‌دهیم. در گام سوم از تمامی کلمات به وجود آمده از اسناد ورودی، ویژگی‌های آماری را استخراج می‌کنیم. در گام چهارم با استفاده از دسته‌بندهای یادگیری ماشین به آموزش داده‌های ورودی و ساخت مدل می‌پردازیم. در گام پنجم و آخر نیز برای بهتر کردن دقت عملکرد برنامه به پس‌پردازش گرایش متون استخراج شده پرداخته خواهد شد.

شکل ۱۱. چارچوب پیشنهادی برای استخراج گرایش متون

در ادامه به شرح کامل هریک از مراحل گفته شده پرداخته خواهد شد.

٤-٢. دریافت داده‌های ورودی

داده‌های مورد استفاده در این پژوهش، پایان‌نامه‌های به زبان فارسی هستند که از مجموعه ایرانداک[1] (پژوهشگاه علم و فناوری اطلاعات ایران) جمع‌آوری شده است. همگی این اسناد به شکل فایل ورد[1]

[1] Irandoc.ac.ir

و با فرمت Docx می‌باشند. اطلاعات مربوط به تعداد اسناد، گرایش پایان‌نامه‌ها، مجموع تعداد کلمات اسناد هر گرایش و تعداد گرایش متون مربوط به هر گرایش که توسط نویسندگان اسناد نوشته شده است در (جدول ۲) آمده است.

جدول ۲. اطلاعات داده‌های ورودی

گرایش	تعداد اسناد	تعداد کلمات	تعداد کلیدواژگان
انسانی	۷۹	۸۵۷۱۶۲	۲۷۰
فنی	۶۶	۷۹۴۹۸۱	۲۵۹
هنر	۴۶	۶۳۴۹۰۳	۱۴۶
پایه	۷۶	۷۵۲۰۰۳	۱۸۰
پزشکی	۴۵	۳۹۴۹۰۹	۱۳۹
جمع کل	۳۱۲	۳۴۳۱۹۸۵	۹۹۴

٤-۳. پیش‌پردازش

یکی از مراحل مهم در پردازش زبان‌های طبیعی، پیش‌پردازش متن می‌باشد. مراحل انجام پیش‌پردازش‌های انجام شده برروی اسناد ورودی را در (شکل ۱۲) مشاهده می‌کنید.

شکل ۱۲. مراحل پیش‌پردازش متن

۴-۳-۱. حذف صفحات زائد

در این بخش تمامی صفحات زائد پایان‌نامه‌ها حذف می‌شوند. این صفحات که شامل بخش‌هایی از صفحات ابتدایی و انتهایی پایان‌نامه‌ها، همانند (تقدیرنامه، تعهدنامه‌ها، اسناد مربوط به بخش‌های داری دانشگاه‌ها، منابع، چکیده به زبان انگلیسی و ...) می‌باشند. این صفحات دارای کلماتی هستند که از اهمیت زیادی جهت استخراج گرایش متون برخوردار نیستند و صرفاً بار محاسباتی برنامه را افزایش می‌دهند.

[1] Word

تفاوت فرمت‌های DOC و DOCX

برای مدتی طولانی Word شرکت مایکروسافت برای ذخیره سازی فایل‌ها از فرمت DOC استفاده می‌کرد. اما این رویه در سال ۲۰۰۷ دست خوش تغییر شد و فرمت ذخیره‌سازی Word جدید از DOCبه DOCX تغییر نمود. اما اضافه شدن یک X ساده به انتهای نام این فرمت، چیزی بی‌معنی و ساده نبود. در واقع X اضافه شده نشان از استاندارد Office Open XML بود.

تقریبا ۳۰ سالی از عمر فرمت DOC می‌گذرد. مایکروسافت برای اولین بار از این فرمت در نسخه ابتدایی- Word که روی MSDOS عرضه شده بود - استفاده کرد. در آن زمان، Word تنها برنامه‌ای بود که از این فرمت پشتیبانی می‌کرد. البته مایکروسافت در سال ۲۰۰۶ این رویه را تغییر داد و مشخصات این فرمت را در اختیار همگان قرار داد.

اگر چه در دهه نود میلادی و خصوصا اوایل سال ۲۰۰۰، واژه‌پردازهای مختلف بسیاری قادر به شناسایی فایل های DOC بودند، اما تقریبا هیچکدام از آن‌ها به طور کامل از فرمت عجیب و استثنایی Wordپشتیبانی نمی‌کردند. همین فرمت خاص و متفاوت مایکروسافت این امکان را در اختیار ردموندی‌ها گذاشت تا بر بسیاری از محصولات خوب آن دوره، نظیر WordPerfect شرکت Corel فائق آید و باعث شد که آن‌ها سلطنت نرم‌افزارهای اداری و واژه‌پردازها را به واسطه آفیس و Word به دست بگیرند.

این روند تا سال ۲۰۰۸ نیز ادامه پیدا کرد و فرمت DOC همچنان به عنوان محبوب‌ترین فرمت واژه‌پرداز شناخته می‌شد. فایل‌های ذخیره شده به صورت فرمت DOC روی ورژن‌های قدیمی نرم‌افزار Word نیز قابل شناسایی بودند و این امکان را برای کاربران فراهم می‌کرد که به راحتی فایل‌های خود را بدون هیچ نگرانی خاصی در اختیار دیگران قرار دهند.

در اوایل سال ۲۰۰۰ و بالا گرفتن رقابت میان دو فرمت متن‌باز Open Office و Open Document Format، مایکروسافت تصمیم گرفت تا مرزهای سلطنت خود را گسترده‌تر کند. به همین منظور

ردموندی ها در آن دوران از توسعه فایل‌های DOCX و همراهانش نظیر XLSX و PPTX سخن به میان آوردند.

استانداردهای جدید مایکروسافت نیز «Office Open XML» نام گرفت که برای توسعه آن‌ها از زبان های سطح بالا به جای فرمت‌های سطح پایین دو دویی استفاده شده بود. زبان های برنامه‌نویسی قدرتمندتر مزایای بسیاری در اختیار کاربران قرار می‌دادند.

فایل‌های کم حجم‌تر، شانس خراب‌شدن پایین‌تر و ظاهر بهتر تصاویر، همگی از مزیت‌های استانداردهای تازه مایکروسافت بودند. به همین ترتیت در سال ۲۰۰۷، تغییرات گسترده‌ای هم در برنامه Word ایجاد شدند. فرمت قدیمی DOC از حالت پیش فرض ذخیره سازی Word فاصله گرفت و DOCX جایگزین آن شد. اما این تغییر رویه ساده و بی‌دردسر هم نبود. بسیاری از کاربران در آن دوره فکر می‌کردند که با آمدن فرمت جدید DOCX و انتشار ورژن تازه مایکروسافت آفیس، نسخه‌های قدیمی دیگر به کار هیچ فردی نمی‌آیند. چرا که ورژن‌های قدیمی دیگر قادر به شناسایی فرمت های جدید XML نبودند.

البته این گمانه‌زنی‌ها تقریبا غلط از آب درآمدند. حتی Word 2003 نیز توانایی خواندن فایل XML را داشت. پس از آن هم مایکروسافت با انتشار به روز رسانی‌های مختلفی این امکان را برای ورژن‌های دیگر نیز فراهم نمود. البته بسیاری از کاربران همچنان ترجیح می‌دادند که از فرمت DOC به جای DOCX استفاده کنند.

دلیل‌شان هم این بود که ورژن‌های قدیمی‌تر با این فرمت سازگاری بهتری دارند و در همه جا می توان از آن‌ها استفاده کرد. با این حال چند سال بعد فرمت DOCX دنیای واژه‌پردازها را در دستان خود گرفت و تبدیل به فرمت استاندارد فایل‌های متنی شد. البته باید خاطر نشان کرد که DOCX دیگر رقیب قدرتمندی هم نداشت که بتواند در مقابلش قد علم کند. مایکروسافت هم این فرصت را غنیمت شمرد و خیلی راحت این فرمت را به استانداردی جهانی تبدیل کرد. در حال حاضر DOCX از هر نظر بهترین فرمت برای ذخیره‌سازی فایل‌هایWORD است. سبک و کم حجم است و برای انتقال

زحمتی روی دوش کاربر نمی‌گذارد. حتی می‌توان با ابزارهای آنلاینی مانند Google Doc ، آن‌ها را خواند و اجرا کرد.

استانداردسازی

ابتدا تمامی اسناد را با استفاده از یونیکد[۱] (UTF-8) استاندارد می‌کنیم، وقتی شما، کاراکتری را در یک برنامه ویرایش متن یا اپلیکیشن وب قرار می‌دهید، این کاراکتر با استفاده از مجموعه‌ای از اعداد، کدگذاری می‌شود. زمانی که مرورگر، محتوای اپلیکیشن وب را دریافت می‌کند، این اعداد رمزگشایی شده و بر روی نمایشگر، نشان داده می‌شوند.

حروف، اعداد و علائمی که در اپلیکیشن‌های وب استفاده می‌شوند، به همان شکلی که شما آنها را می‌بینید، در کامپیوتر مدیریت نمی‌شوند. کامپیوترها فقط با اعداد سروکار دارند. پس این حروف و کاراکترها، باید به مجموعه‌ای از اعداد ۰ و ۱ تبدیل شوند تا مدیریت آنها آسان باشد. لذا استاندارد واحدی باید وجود داشته باشد. بر همین اساس، مشخص می‌شود که هرکدام از این اعداد چه کاراکترهایی را نمایش دهند و چگونه بر روی دیسک ذخیره شوند.

برای نیل به این هدف، انجمن استانداردهای آمریکا در سال ۱۹۶۰ یک روش کدگذاری ۷ بیتی، با نام (ASCII) ”American Standard Code for Information Interchange“ را معرفی کرد. در آن زمان، مجموعه کاراکترهای ASCII، ۱۲۸ کاراکتر (۷ بیت) داشتند که بیشتر مخصوص زبان‌های لاتین بود.

در دهه ۱۹۸۰، تصمیم بر این شد که در مجموعه کاراکتر ASCII به جای ۷ بیت، از یک بایت کامل یعنی ۸ بیت، برای کدگذاری استفاده شود. لذا تعداد کاراکترها به ۲۵۶ عدد می‌رسید. بر این اساس، کاراکترهای بعد از ۱۲۷ تا ۲۵۵ نیز، به عنوان کدهای رزرو شده در نظر گرفته شدند و زبان‌های دیگر، عموما در این بازه قرار می‌گرفتند. اما در این محدوده بین زبان‌های مختلف، استاندارد واحدی وجود نداشت و هر زبانی، کد مختصِ الفبایِ خودش را نشان می‌داد. به عبارت دیگر کد ۲۰۰ در یک زبان،

---

[۱] Unicode

حرف متفاوتی را در زبان دیگر برمی‌گرداند. بنابراین، نیاز به استاندارد واحدی بود تا ضمن سازگاری با تمامی‌زبان‌ها، کدهای منحصر به فردی را برای هر کاراکتر در نظر بگیرد.

در ابتدا دو تلاش مستقل برای ایجاد مجموعه کاراکترهای واحد صورت گرفت. اولین آنها-ISO'' 10646'' پروژه سازمان بین‌المللی استاندارد بود، و پروژه بعدی نیز''Unicode'' نام داشت که توسط کنسرسیومی به نام کنسرسیوم یونیکد سازماندهی می‌شد. داشتن دو نوع استاندارد مطمئنا چیزی نبود که بتوان آن را استاندارد واحدی نامید. ISO و Unicode این مطلب را دریافتند و تصمیم گرفتند در سال ۱۹۹۱ به یکدیگر بپیوندند.

یونیکد، مجموعه‌ای از کاراکترها با اعداد منحصر به فرد است، که به آن در اصطلاح پوینت کد (Point Code) گفته می‌شود. هر پوینت کد، کارکتر واحدی را نمایش می‌دهد. بر این اساس، استاندارد یونیکد سه نوع روش کدگذاری را تعیین می‌کند، و به یک کاراکتر اجازه می‌دهد در داخل یک یا چند بایت کدگذاری شود (یعنی در ۸ یا ۱۶ یا ۳۲ بیت). این سه نوع روش کدگذاری به ترتیب UTF-8 , UTF-16 و UTF-32 نامیده می‌شوند ''UTF'' ، مخفف فرمت انتقال یونیکد است.

تفاوت این روش‌های کدگذاری، در نحوه ارایه حروف، اعداد و علائم، بین زبان‌های کشورهای مختلف است. به طوری که نحوه ارایه کاراکترها در یک کشور با کشور دیگر متفاوت است.

(UTF-8) ، اولین بار بطور رسمی در کنفرانس USENIX در سال ۱۹۹۳ معرفی شد. در حال حاضر (UTF-8)، غالب‌ترین روش کدگذاری کاراکتر در میان وب‌سایت‌ها است. (UTF-8)، روشی است که قابلیت کدگذاری تمامی‌کاراکترهای موجود و یا به عبارتی تمامی کد پوینت‌های موجود در یونیکد را دارد.

(UTF-8)، همانطور که گفته شد الگوریتمی است که اعداد مربوط به پوینت‌کد را به باینری تبدیل می‌کند، بطوری که بتوان آنها را بر روی دیسک ذخیره کرد.

(UTF-8) طول متغیری دارد و می‌تواند تا ۴ بایت افزایش یابد، ولی کاراکترهای اصلی (ASCII) را می‌تواند با یک بایت نمایش دهد. چون طول متغیری دارد باید روشی وجود داشته باشد که مشخص شود، کاراکتر از یک بایت یا چند بایت ساخته شده است. لذا، (UTF-8)، در بایت اول تنها از ۷ بیت آن استفاده می‌کند و بیت اول آن برای این هدف کنار گذاشته شده است.

مزایایUTF-8

UTF-8تنها الگوریتم موجود برای XML است که نیازی به BOM یا شاخص کدگذاری ندارد.

UTF-8و UTF-16 روش‌های کدگذاری استاندارد برای متون یونیکد در فایلهای HTML هستند، و UTF-8پرکاربردترین آنها است.

رشته کد UTF-8 می‌تواند همانند یک الگوریتم اکتشافی ساده به نظر برسد. این ویژگی که بیشتر روش‌های کدگذاری آن را ندارند، به UTF-8 اجازه می‌دهد نوع کدگذاری را تشخیص دهد. با این روش، بدون اینکه نیازی به افزودن بیت به آن داشته باشد، از خطاهای معمولی که هنگام تغییر یک سیستم به یک انکدینگ پیش‌فرض روی می‌دهد ، اجتناب خواهد کرد.

UTF-8می‌تواند هر نوع کارکتر یونیکد را کدگذاری کند. فایلها را، بدون اینکه مجبور باشند فونت درستی را انتخاب کنند، با اسکریپت‌های متفاوت به درستی نمایش دهد.

UTF-8، از کدهای ۰–۱۲۷ برای کاراکترهای اسکی استفاده می‌کند. این کد بر خلاف دیگر سیستم‌ها، نیازی به افزایش حجم برای نشان دادن کدهای اسکی ندارد. این بدین معنی است که در تمامی نرم‌افزارهایی که از کاراکترهای ۷ بیتی پشتیبانی می‌کنند، قابل پردازش است.

(UTF-8) قابلیت خود هماهنگی دارد: اگر بایت‌ها به دلیل خطا یا مشکلی از بین بروند ، می‌توان شروع کاراکتر معتبر بعدی را پیدا کرد و پردازش را ادامه داد.

کدگذاری در(UTF-8)، نیازی به عملیات ریاضی مانند ضرب و تقسیم ندارد و از عملیات ساده بیتی استفاده می‌کند.

معایب

به کاراکترهایی که در روش‌های کدگذاری دیگر مانند ISO-8859 و WINDOWS-1252 می‌توانند با یک بایت نشان داده شوند، در UTF-8 باید با دو بایت نمایش داده شوند.

یک مبدلUTF-8 ، که با نسخه‌های کنونی استاندارد، سازگار نیست. ممکن است یک عدد شبیه به UTF-8 متفاوت را دریافت کند و آن را به خروجی یونیکد تبدیل کند.

متون کدگذاری شده توسط UTF-8، به جز برای کاراکترهای ASCII، حجم بیشتری نسبت به سیستم‌های دیگر اشغال می‌کند.

در UTF-8، این امکان وجود دارد که یک کاراکتر را از وسط یک رشته کد بشکافید. اگر دو قطعه جدا شده نتوانند بعدا در توالی هم قرار بگیرند، این امر ممکن است باعث شود آن رشته کد، نامعتبر شود.

بسیاری از نرم‌افزارها مانند ویرایشگر متن، UTF-8را نمی‌توانند نمایش دهند یا ترجمه کنند، مگر اینکه آن متن با یک BOM شروع شود.

UTF-8، نسبت به یک انکدینگ چند بایته، که تنها برای یک زبان خاص طراحی شده است، حجم بیشتری میگیرد. کدگذاری زبان‌های آسیای شرقی نیاز به دو بایت برای هر کاراکتر دارند، در صورتی که درUTF-8 ، ۳ بایت نیاز است.

۴-۳-۲. وضع قوانین نگارشی

از جمله اقداماتی که در این حوزه توسط پژوهشگران انجام می‌شود ایجاد شروطی است که به تمیز شدن متن کمک شایانی می‌کند. در این پژوهش سه شرط زیر را در نظر گرفته‌ایم:

۱) حذف علائم نگارشی

۲) حذف اعداد

۳) عذف کلمات کمتر از سه حرف

علائم نگارشی مانند (":"، "؟"، "("، ")" و ...) می‌باشند. با حذف آن‌ها از اسناد به روند پاکسازی متن کمک می‌کنیم.

همان گونه که می‌دانیم اعداد در اکثر مواقع جزئی از کلیدواژگان متن نبوده است و به همین دلیل به پاکسازی آن‌ها از متن اقدام می‌نماییم. حذف اعداد فارسی به طور مستقیم ممکن نیست. به دلیل پشتیبانی نشدن زبان فارسی در توابع برنامه‌نویسی موجود در این حوزه، سیستم قادر به شناسایی اعداد فارسی به طور مستقیم نیست و هرکدام از آن‌ها را به عنوان یک کلمه واحد در نظر می‌گیرد. به عنوان مثال هیچ تفاوتی بین کلمه "مدیریت" و عدد "۱۹۲۳" قائل نمی‌شود. برای همین ابتدا با نوشتن برنامه‌ای تمامی اعداد را به زبان انگلیسی تبدیل می‌کنیم و سپس به راحتی تمامی آن‌ها را از اسناد حذف می‌کنیم.

برای قسمت حذف کلمات کمتر از سه حرف ابتدا توجه شما را به (جدول ۳) جلب می‌نماییم.

جدول ۲. نمونه‌ای از عبارت‌های دو حرفی

می	من	ما	در	یک	دو	سه	نه
از	به	او	آن	را	با	که	بر

همان گونه که مشاهده می‌کنیم تمامی این عبارات کلمات زائدی هستند که باعث زیاد شدن حجم محاسبات ما می‌شوند. با توجه به اینکه نباید کلیدواژه ها حذف گردند، بررسی ها نشان می‌دهد که در تمامی ۳۱۲ سند مورد بررسی فقط یک کلیدواژه کمتر از سه حرف هست و آن هم کلمه "سگ" می‌باشد که مربوط به یکی از اسناد گرایش پزشکی می‌باشد.

لازم به ذکر است برای آنکه هرکدام از این سه شرط را اجرا کنیم ابتدا باید هر سند را به شکل لیستی از کاراکترها تبدیل کنیم و شرط کنیم که اگر هر کدام از کاراکترهای متن مورد نظر جزو علائم نگارشی، اعداد و یا کمتر از دو حرف باشند را حذف نماید.

۴-۳-۳. حذف ایست‌واژه‌ها

همان گونه که در قسمت پیشینه پژوهش نیز گفته شد، ایست‌واژه‌ها لغاتی هستند که علی‌رغم تکرار فراوان در متن، از لحاظ معنایی دارای اهمیت کمی هستتند. تعدادی از این ایست‌واژه‌ها همانند ("از"،"به"،"را"، ...) در قسمت قبل حذف می‌شوند. اما این عبارات کافی نیستند و بسیاری از کلمات نیز هستند که هیچگاه بیانگر محتوای متن مورد نظر ما نمی‌باشند. نمونه از ایست‌واژه‌های مورد استفاده در این پژوهش را در (جدول ۴) مشاهده می‌کنید.

جدول ۳. نمونه ایست‌واژه‌ها

است	بیشتر	هست	شکل	تصویر	جدول
آمد	بود	رفت	می‌توان	آیا	تنها
چون	زیرا	برای	آنکه	شما	سپس
همچنین	ولی	اگر	اتفاق	خواهش	همان
چیزی	آنان	خواست	نمودار	حتی	قرار
گرفته	طور	کلی	یکی	بدون	لذا

برای آنکه متن مورد نظر را از ایست‌واژه‌ها پاک کنیم، ابتدا لیستی از کلمات هر متن تشکیل می‌دهیم. سپس با شرطی که ایجاد می‌کنیم، قید می‌کنیم که اگر کلماتی از متن موجود باشند که در لیست ایست‌واژه‌های ما باشند آنگاه کلمه مورد نظر را از متن حذف کند.

۴-۳-۴. نرمال‌سازی

جهت یکسان‌سازی متن از قسمت نرمال‌سازی استفاده می‌کنیم. در این پژوهش جهت نرمال‌سازی از دو برنامه هضم[1] و فارسی‌یار[2] استفاده شده است.

[1] Sobhe.ir

[2] Text-mining.ir

۴-۳-۵. مفهوم نرمال‌سازی

با توجه به اینکه ممکن است اسناد منتشر شده، از کدگذاری‌های متفاوتی استفاده کنند لذا برای بهبود تحلیل‌های متنی نیازمند یکسان‌سازی این متون هستیم. درواقع حذف کدحروف غیر مناسب (مانند اعراب حروف) یا تبدیل کدهای مختلف حروف (مانند کد /ی/ عربی و فارسی) به یک کد واحد در این مرحله انجام می‌شود. همچنین کلماتی که دارای ساختار نوشتاری مختلف هستند به یک کلمه تبدیل می‌شود.

در اولین گام باید متون برای استفاده در گام‌های بعدی به شکلی استاندارد درآیند. از آنجایی که متون مختلف ممکن است بسیار به هم شبیه باشند اما به دلیل تفاوت‌های ساده ظاهری از نظر ماشین متفاوت باشند؛ به همین دلیل سعی شده است این تفاوت‌های ساده‌ی ظاهری برطرف گردد. برای رسیدن به این هدف، قبل از مقایسه متون، پیش‌پردازش‌هایی روی آنها آنجام می‌شود. طبیعتاً هر چه این پیش‌پردازش‌ها قوی‌تر باشد، نتایج حاصل ازمقایسه متون قابل اطمینان‌تر خواهد بود. لازم به ذکر است که از آنجایی که زبان فارسی جزو زبان‌های غیر ساخت‌یافته است با مشکلات بسیار بیشتری نسبت به سایر زبان‌ها مواجه خواهیم شد. متون غیرساخت‌یافته، متونی هستند که پیش‌فرض خاصی در مورد قالب آنها نداریم و آن‌ها را به صورت مجموعه‌ای مرتب از جملات در نظر می‌گیریم.

به طور معمول ناسازگاری‌های موجود در متن شامل موارد زیر است:

وجود encoding های مختلف برای بعضی از کاراکترها مانند «ی» و «ک»، فاصله‌های اضافه، تب‌های اضافه، تنوع نوشتار با حروف بزرگ و کوچک، روش‌های مختلف چسبیدن وندها به کلمات اصلی، روش‌های مختلف اتصال اجزای کلمات مرکب، کلمات چندامالیی، غلط‌های املایی، کلمات محاوره‌ای، اختصارات، تمایز بین مجامع به منظور لیست کردن اسامی افراد و گروه ها، استفاده از اصطلاحات غیر استاندارد و مختصرسازی‌های غیر رایج (Colo → Colorado) .

پردازش زبان فارسی از جهاتی با پردازش زبان انگلیسی تفاوت دارد. در زبان انگلیسی تمامی حروف و تمامی کلمات جدا از هم و با قانونی مشخص نوشته می‌شوند و این در حالی است که در زبان فارسی بعضی از حروف به هم چسبیده هستند، برخی از حروف جدا ازهم نوشته می‌شوند، بعضی از کلمات یکپارچه‌اند، بعضی از کلمات با فاصله یا نیم‌فاصله به دو یا چند بخش تقسیم می‌شوند. تمامی حوزه‌های مرتبط با پردازش زبان طبیعی به نحوی با متون واقعی سروکار دارند. صورت‌های غیراستاندارد نویسه‌ها و کلمات به وفور در این نوع متون نوشته دیده می‌شوند. قبل از این که بتوان از این متون به منظور استفاده در سیستم‌های تبدیل متن به گفتار، ترجمه ماشینی، بازشناسی حروف فارسی، خلاصه‌ساز فارسی، جستجو در متون فارسی و غیره استفاده کرد و یا در پایگاه داده ذخیره نمود، باید ابتدا پیش‌پردازشی روی آن‌ها انجام گیرد تا صورت‌های غیر استاندارد به شکل استاندارد تبدیل گردند. اگر حروف، نشانه‌های نگارشی و کلمات فارسی به شکل یکسانی نوشته نشوند، متون مورد استفاده قابل تحلیل توسط سامانه‌های رایانه‌ای نخواهند بود. طی فرآیند نرمال‌سازی، علایم نگارشی، حروف، فاصله‌های بین کلمات، اختصارات و غیره بدون ایجاد تغییرات معنایی در متن به شکل استاندارد تبدیل می‌گردند. بنابراین، بایستی از یک استاندارد مشترک برای پیش‌پردازش و پردازش متون استفاده کرد. در حالت ساده عملیات نرمال‌سازی متن با مراحل زیر انجام پذیر است:

اصلاح انواع حرف «ک» به معادل فارسی آنان.

اصلاح انواع حرف «ی» به معادل فارسی آنان.

بررسی همزه و انواع مختلف املاهای موجود و اصلاح هر کدام (به عنوان مثال تبدیل ؤ به و ، ئ به ی ، أ به ا ، إ به او، ...)

حذف شناسه‌ی همزه از انتهای واژه‌هایی مثل شهداء

حذف شناسه «آ» به «ا» مانند : آب به اب

اصلاح نویسه‌ی «طور» در واژه‌هایی مانند به طور، آن طور، این طور و...

بررسی وجود حرف «ی» در انتهای لغاتی مانند خانه‌ی ما و اصلاح آنان

حذف تشدید از واژه‌ها

اصلاح نویسه‌ی نیم‌فاصله

اصلاح اعراب و حذف فتحه، کسره و ضمه و همچنین تنوین‌ها

حذف نیم‌فاصله‌های تکراری

حذف نویسه‌ی «ـ» که برای کشش نویسه‌های چسبان مورد استفاده قرار می‌گیرد. مانند تبدیل«بـــــــر» و «بـــر» به «بر»

چسباندن پسوندهای «تر»، «ترین» و ... به آخر واژه‌ها

اصلاح فاصله‌گذاری «ها» در انتهای واژه‌ها و همچنین پسوندهای «های»، «هایت»، و...

اصلاح فاصله‌گذاری «می»، «نمی»، «درمی»، «برمی»، «بی» در ابتدای واژه‌ها

تبدیل «ه» به «ه‌ی»

تبدیل «ب» متصل به ابتدای واژه‌ها به «به»

اصلاح فاصله‌گذاری پسوندها

حذف فاصله‌ها و نیم‌فاصله‌های اضافه بکار رفته در متن

تصحیح فاصله‌گذاری در مورد علائم سجاوندی بدین صورت که علائم سجاوندی به لغات قبل از خود می‌چسبند و با لغت بعد از خود فاصله خواهند داشت.

۴-۳-۶. نرمال‌سازی با هضم

هضم، برنامه رایگانی است که جهت امور پردازش زبان‌های فارسی به وجود آمده است. یکی از مزیت‌های هضم آن است که به صورت یک بسته[1] در زبان برنامه‌نویسی پایتون[2] قابل استفاده است. برای استفاده از برنامه هضم ابتدا می‌بایستی هر متن را به شکل لیستی از کلمات دربیاوریم که در اصطلاح به آن نشان‌دار[3] کردن نیز گفته می‌شود. سپس هر کلمه از متن را با استفاده از این برنامه نرمال می‌کنیم.

۴-۳-۷. نرمال‌سازی با فارسی‌یار

برنامه فارسی‌یار، برنامه‌ای وب محور است که جهت پردازش زبان فارسی ساخته شده است. برای استفاده رایگان از قسمت‌های مختلف این برنامه باید دارای کلید API باشید. استفاده‌کنندگان رایگان از API حداکثر مجاز به ارسال روزی ۱۰۰ درخواست (فراخوانی) و در هر ثانیه یک درخواست و هر فراخوانی (درخواست) با طول ورودی متنی کمتر از ۳۰۰۰ کاراکتر یا ۶۰۰۰ هزاربایت می‌باشند. دقت عملکرد این برنامه نسبت به برنامه هضم مزیت آن می‌باشد. در این برنامه برعکس برنامه هضم، برای نرمال‌سازی هر متن را به صورت یک متن کامل به آن می‌دهیم.

۴-۳-۸. ریشه‌یابی

ریشه‌یابی کلمات به معنی کاهش هر کلمه به پایه‌ای‌ترین شکل آن کلمه می‌باشد. در این بخش نیز از دو برنامه هضم و فارسی‌یار استفاده کرده‌ایم. در (جدول ۵) و(جدول ۶) مثال‌هایی از آن‌ها را ملاحظه می‌کنید. اما یکی از مواردی که باید به آن توجه شود آن است که داده‌های ورودی ما اسناد علمی می‌باشند، پس دارای عبارات تخصصی مربوط به هر کدام از گرایش‌ها نیز می‌باشند. برخی از این کلمات هستند که قابل تشخیص برای هر کدام از این دو برنامه نیستند و وقتی که جهت ریشه‌یابی از آن‌ها استفاده می‌کنیم، به طور کلی ماهیت آن کلمه دچار تغییرات می‌شود. به طور مثال کلمه "شرون

[1] Package

[2] Python

[3] Tokenize

لاسون" را در نظر بگیرید. این کلمه یک اسم لاتین است که با زبان فارسی نوشته شده است. این کلمه با ریشه‌یابی توسط برنامه هضم به عبارت "ساروس" و با برنامه فارسی‌یار به عبارت "لاس" تبدیل می‌شود.

جدول ٤.تغییرات کلمات با ریشه‌یابی فارسی‌یار

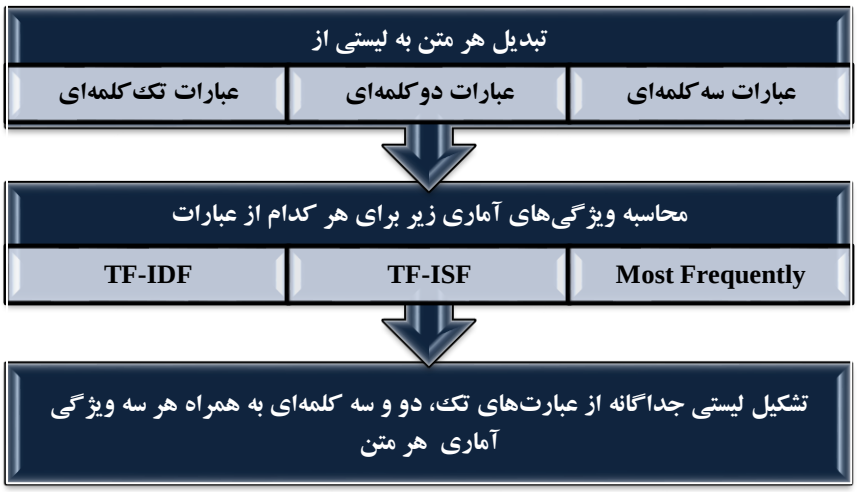
بعد از ریشه‌یابی	قبل از ریشه‌یابی
کارشناس	کارشناسان
مدیریت	مدیریت
مالیات	مالیاتی
نوآور	نوآوری

جدول ۵. تغییرات کلمات با ریشه‌یابی هضم

بعد از ریشه‌یابی	قبل از ریشه‌یابی
کارشناس	کارشناسان
مدیر	مدیریت
مالیات	مالیاتی
نوآور	نوآوری

٤-٤. استخراج ویژگی

پس از آن که تمامی اسناد ورودی را پیش‌پردازش کردیم، وارد بخش استخراج ویژگی از کلمات شده‌ایم. مراحل عملیات استخراج ویژگی را در (شکل ۱۳) مشاهده می‌کنیم.



شکل ۱۳. مراحل بخش استخراج ویژگی از کلمات

۴-۴-۱. استخراج عبارات تک‌کلمه‌ای، دوکلمه‌ای و سه‌کلمه‌ای

اولین گام در بخش استخراج کلمات این است تصمیم بگیریم که کلمات خروجی ما دارای چند عبارت یا (Grams) باشد. با توجه به بررسی‌های انجام شده برروی کلیدواژگان نوشته شده توسط

نویسندگان پایان‌نامه‌ها، تنها ۳۰ عدد از مجموع کلیدواژه‌ها دارای عبارتی با ۴ و یا بیش از آن کلمه می‌باشند، پس تصمیم بر آن شد که کلمات را به صورت تک‌کلمه‌ای[1]، دو کلمه‌ای[2] و سه‌کلمه‌ای[3] از اسنادی که مورد پیش‌پردازش قرار داده‌ایم، استخراج شوند. در(جدول ۷) مثالی از این کلمات را ملاحظه می‌کنیم.

جدول ٦. نمونه‌ای از عبارات تک، دو و سه کلمه‌ای

عبارات سه‌کلمه‌ای	عبارات دوکلمه‌ای	عبارات تک‌کلمه‌ای
استقرار مدیریت دانش	جامعه آماری	رگرسیون
سازمان امر مالیات	روش تصادف	آزمون
تعداد جامعه آماری	ابزار پژوهش	مستقل
مولفه فرهنگ سازمان	عبارت پرسشنامه	تحلیل

۴-۴-۲. محاسبه ویژگی‌های آماری

در این قسمت برای هر کدام از عبارت‌های تک، دو و سه‌کلمه‌ای ویژگی‌های آماری را محاسبه می‌کنیم. بدین معنی که وقتی در حال محاسبه ویژگی آماری یک عبارت تک‌کلمه‌ای هستیم، فقط عبارت‌های تک‌کلمه‌ای آن متن و متون دیگر مورد بررسی قرار می‌گیرند و لیست تشکیل شده از عبارت‌های دو و سه‌کلمه‌ای اسناد مورد بررسی قرار نمی‌گیرند. در ابتدا به محاسبه ویژگی (بسامد کلمه – معکوس بسامد سند) سپس ویژگی (بیشترین فرکانس) و بعد از آن‌ها ویژگی (بسامد کلمه – معکوس بسامد جمله) پرداخته‌ایم.

بسامد کلمه – معکوس بسامد سند

در این قسمت ابتدا میزان فراوانی هر عبارت را در سند مربوطه‌اش می‌سنجیم. بدین معنی که، تعداد تکرار هر عبارت را در هر متن می‌شماریم و به تعداد کل کلمات همان متن تقسیم می‌کنیم و بدین

[1] Unigram

[2] Bigram

[3] Trigram

صورت مقدار بسامد کلمه یا همان (TF) آن بدست می‌آید. سپس برای محاسبه مقدار معکوس بسامد سند (IDF) مقدار تعداد کل اسناد را، که در این پژوهش ۳۱۲ می‌باشد را بر تعداد اسنادی که آن عبارت در آن‌ها دیده شده است تقسیم می‌کنیم و از آن‌ها لگاریتم می‌گیریم.

$$IDF(n,N) = \log \frac{N}{n} \quad (۱-٤)$$

سپس مقدار TF را در مقدار IDF ضرب می‌نماییم. این مراحل را برای تمامی عبارات انجام می‌دهیم و در آخر دارای لیستی از عبارات تک، دو و سه‌کلمه‌ای به صورت جداگانه هستیم که دارای مقادیر (TF-IDF) متناظرشان می‌باشد.

بیشترین تکرار کلمه

برای محاسبه بیشترین تکرار کلمه تنها کافی است تا تعداد تکرار هر عبارت را در هر سند محاسبه کنیم. بدین صورت میزان فراوانی هر کلمه در هر متن بدست می‌آید.

بسامد کلمه- مسامد معکوس جمله

در این قسمت عملیات محاسبه مقدار (TF) همانند بخش (TF-IDF) می‌باشد. اما برای محاسبه مقدار (ISF)، تعداد کل اسناد را به تعداد تکرارهایی که آن عبارت در کل ۳۱۲ سند ما آمده است تقسیم می‌کنیم. اگرچه با توجه به تعریف دقیق (ISF) باید مقدار (TF) را به تعداد جملاتی که آن عبارت در کل اسناد آمده است تقسیم کنیم، اما در این پژوهش برای محاسبه دقیق‌تر میزان پراکندکی کلمات در اسناد مختلف به تعداد کلمات آن‌ها تقسیم شده است. در آخر نیز مفدار (TF) بدست آمده را در مقدار (ISF) آن ضرب می‌کنیم.

۴-۴-۳. ساخت لیست

حال که برای تمامی عبارات ساخته شده از اسناد، مقادیر ویژگی آن‌ها را محاسبه کرده‌ایم، لیستی برای عبارت‌های تک‌کلمه‌ای تشکیل می‌دهیم که در آن، عبارت‌های هر سند دارای مقادیر ویژگی (TF-IDF، TF-ISF، MF) متناظرشان می‌باشد. یعنی در این مرحله ما دارای لیستی شامل ۳۱۲ سند هستیم، که هرکدام از این اسناد دارای عبارات تک‌کلمه‌ای تشکیل دهنده‌شان هستند و هرکدام از این عبارات دارای سه مقدار ویژگی‌شان می‌باشند. نمونه‌ای از این ماتریس را برای یکی از اسناد در (جدول ۸) قابل مشاهده می‌باشد. تمامی این مراحل را برای کلمات دو و سه‌کلمه‌ای نیز تکرار می‌کنیم و در کل دارای سه لیست کلی می‌باشیم.

جدول ۷. نمونه‌ای از ماتریس تشکیل شده از عبارات تک‌کلمه‌ای و ویژگی آن‌ها

کلمات	TF-IDF	TF-ISF	Most frequent
چکیده	۱,۹۴۵*۱۰$^{-۵}$	۵,۳۷*۱۰$^{-۵}$	۱
منظور	۱,۲۶۹*۱۰$^{-۳}$	۶,۹*۱۰$^{-۳}$	۸
بررسی	۰	۱,۳۰۴*۱۰$^{-۲}$	۴۰
رابطه	۱,۲۵۷*۱۰$^{-۲}$	۳,۳۲۲*۱۰$^{-۲}$	۷
عامل	۶,۴۹۰*۱۰$^{-۵}$	۱,۰۷*۱۰$^{-۳}$	۱۵۰
فرهنگ	۳,۳۶۹*۱۰$^{-۲}$	۹,۴۵*۱۰$^{-۳}$	۲

٤-٥. انتخاب الگوریتم یادگیرنده

حال که لیست ورودی ما آماده شده است نوبت به انتخاب الگوریتم یادگیرنده می‌رسد. ما در این مرحله ابتدا داده‌های آموزش را به هرکدام از الگوریتم‌های یادگیری (بیز ساده، درخت تصمیم و ماشین‌های بردار پشتیبان) داده‌ایم و نتایج بدست آمده برای داده‌های تست را بدست آورده‌ایم.

۴-۵-۱. دسته‌بند بیز ساده گاوسی[1]

[1] Gaussian Naive Bayes Classifier

بیز ساده گاوسی یکی از مشهورترین نوع طبقه‌کننده‌های بیز ساده است. نحوه عملکرد آن در اینجا بدین گونه است که برای طبقه‌بندی داده‌های تست از ساختار زیر استفاده می‌کند. در این ساختار احتمال تعلق داده به کلاس کلیدواژگان بررسی شده است.

(۲-۴)

$$posterior(KW) = \frac{p(KW) * p(TF-IDF \mid KW) * p(TF-ISF \mid KW) * p(MF \mid KW)}{MP}$$

در (معادله ۴-۲) P(KW) همان احتمال کلیدواژه بودن است که از تقسیم تعداد کلیدواژه‌ها بر تعداد کل کلمات بدست می‌آید. P(TF-IDF | KW)، P(TF-ISF | KW) و P(MF | KW) که همان مقادیر همسایگی[1] ما هستند، به دلیل اینکه در این قسمت از تابع گاوسی استفاده کرده‌ایم از (معادله ۴-۳) بدست می‌آید. X که همان ویژگی‌های مربوط به داده تست ما می‌باشد. μ میانگین ویژگی‌هایی در داده‌های آموزش ما هستند که متعلق به کلاس کلیدواژگان می‌باشند. σ^2، مقدار واریانس ما می‌باشد که مقدار آن از (معادله ۴-۴) بدست می‌آید.

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right) \quad (۳-٤)$$

$$\sigma^2 = \frac{\sum (x_i - \overline{x})^2}{n-1} \quad (۴-۴)$$

MP یا همان احتمال حاشیه‌ای[2] به معنی مجموع ویژگی‌های داده تست ما می‌باشند.

[1] Likelihood

[2] marginal probability

حال تمامی این مراحل را برای حالتی که این داده تست برای کلاس غیرکلیدواژگان می‌باشد را نیز محاسبه می‌کنیم. در انتها مقادیر بدست آمده برای هر دو کلاس را با یکدیگر مقایسه می‌کنیم و هرکدام که بیشتر بود، داده تست به آن کلاس تعلق می‌گیرد.

۴-۵-۲. دسته‌بند ماشین بردارهای پشتیبان

هدف کلی این دسته‌بند پیدا کردن خطی برای جداسازی داده‌ها از یکدیگر می‌باشد. در اینجا به تعداد ویژگی‌هایی که داریم دارای ابعاد هستیم. به طور فرض اگر فقط دو ویژگی بیشترین تکرار و بیشترین بسامد سند را در نظر بگیریم، داده‌های آموزش ما در فضای دوبعدی ترسیم می‌شوند. هسته اصلی[1] استفاده شده در این پژوهش برای ماشین‌های بردار پشتیبان خطی[2] می‌باشد. پس با یک خط داده‌های آموزش ما از یکدیگر جدا می‌کند (شکل ۱۴) و با استفاده از همین خط اقدام به جداسازی داده‌های تست می‌کند. این دسته‌بند پارامترهایی نیزدارد که ما از پارامتر C استفاده کرده‌ایم. وجود پارامتر C مقدار حاشیه و خطای دسته‌بند را کنترل می‌کند. در حالت کلی دو حالت برای مقدار C وجود دارد:

۱) مقدار C را زیاد در نظر بگیریم و حاشیه را باریک کنیم و خطای آموزش را کمتر کنیم.

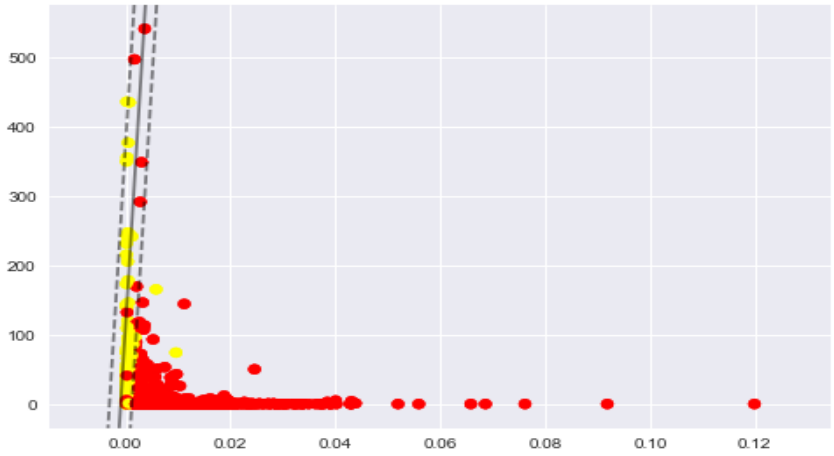
۲) مقدار C را کوچک درنظر بگیریم و خطای داده‌های آموزشی را افزایش دهیم.

مقدار C قبل آموزش داده‌ها و با جستجوی حریصانه[3] پیدا می‌شود. در این پژوهش مقدار C برابر با (2^5) می‌باشد. نتایج بدست آمده از این دسته‌بند نیز در فصل چهارم مورد بررسی قرار می‌گیرد.

[1] Kernel

[2] Linear

[3] Grid Search



شکل ۱٤. نحوه جداسازی داده‌های آموزش توسط دسته‌بند ماشین‌های بردار پشتیبان با دو ویژگی بیشترین تکرار و بیشترین بسامد سند

فصل پنجم

ارزیابی روش پیشنهادی

۵-۱. معیارهای ارزیابی

برای ارزیابی عملکرد پیش‌بینی شده در روش‌های استخراج کلمات کلیدی آماری، الگوریتم‌های طبقه‌بندی و روش‌های ترکیبی از دقت طبقه بندی، اندازه گیری F و منطقه تحت منحنی(AUC) به عنوان معیارهای ارزیابی استفاده می‌شود. دقت طبقه بندی (ACC) یکی از معیارهای به کار رفته در بررسی طبقه‌بندی‌ها است. این نسبت، نسبت مثبت واقعی و منفی واقعی در کل تعداد موارد است که توسط معادله داده نشان شده است:

$$ACC = \frac{TN + TP}{TP + FP + FN + TN} \quad (۱-۵)$$

جایی که TN، TP، FP و FN نشان دهنده تعداد منفی واقعی، تعداد مثبت واقعی، تعداد مثبت کاذب و تعداد منفی‌های کاذب است. دقت (PRE) درصد مثبت واقعی نسبت به مجموع مثبت واقعی و مثبت کاذب است که توسط معادله زیر ارائه شده است:

$$PRE = \frac{TP}{TP + FP} \quad (۲-۵)$$

در (REC)، نسبت مثبت‌های واقعی نسبت به مثبت‌های واقعی و منفی‌های کاذب است که توسط معادله زیر داده می‌شود:

$$REC = \frac{TP}{TP + FN} \quad (۳-۵)$$

مقدار F مقداری بین (۰ و ۱) می‌باشد. این میانگین هارمونیک دقیق و فراخوانی است که توسط معادله زیر تعیین می‌شود:

$$F\_measure = \frac{2 \times PRE \times REC}{PRE + REC} \quad (۴-۵)$$

محدوده تحت منحنی (AUC) یکی دیگر از شاخص‌های معمول برای ارزیابی طبقه‌بندی‌ها است. این برابر با احتمالی است که یک طبقه‌بندی یک نمونه مثبت تصادفی انتخاب شده را بالاتر از یک

انتخاب منفی انتخاب می‌کند. مقدار آن از (۰ به ۱) طول می‌کشد. مقادیر بالاتر AUC نشان دهنده عملکرد بهتر الگوریتم‌های طبقه بندی است.

۵-۲ نتایج با دسته‌بند بردارهای ماشین پشتیبان

حال با توجه به معیارهای ارزیابی گفته شده به تحلیل نتایج مربوط به دسته‌بند بردارهای ماشین پشتیبان می‌پردازیم. در جریان عملیات یادگیری مقدار Cross validation برابر با ۵ می‌باشد. بدین معنی که تعداد داده‌های آموزش و تست ۵ مرحله به صورت تصادفی انتخاب شده‌اند و آموزش داده می‌شوند. میانگین میزان دقت عملکرد دسته‌بند در مجموع ۵ مرحله در (جدول ۹) آمده است.

جدول ۹. میانگین میزان دقت با استفاده از دسته‌بند بردارهای ماشین پشتیبان

Cross validation	۱	۲	۳	٤	۵	میانگین
نتایج	۰,۸۷۶۹۲	۰,۸۳۸۷۰	۰,۸۵۴۸۳	۰,۸۷۰۹۶	۰,۹۱۸۰۳	۰,۸۹۲

میانگین مجموع معیارهای ارزیابی مدل ساخته شده با Cross Validation برابر با مقدار ۵ را برای هرکدام از گرایش‌ها را در جدول ۱۰ شاهد می‌باشیم.

جدول۱۰. میانگین مجموع معیارهای ارزیابی مدل ساخته شده برای گرایش‌های مختلف با استفاده از دسته‌بند بردارهای ماشین پشتیبان

گرایش	Precision	Recall	F1-Score
انسانی	۱	۰,۷۹	۰,۸۸
فنی	۱	۰,۸۲	۰,۹۰
هنر	۰,۷۸	۰,۸۸	۰,۸۲
پایه	۰,۸۱	۰,۹۵	۰,۸۸
پزشکی	۰,۷۵	۱	۰,۸۶

همان طور که مشاهده می‌کنید، بیشترین میزان Precision مربوط به گرایش انسانی وفنی است، یعنی مدل یادگیرنده هرگاه سندی را انسانی و یا فنی تشخیص داده است، درست بوده. اما معیار Recall گرایش انسانی از سایر گرایش‌ها کمتر است. بدین معنی که اسنادی مربوط به گرایش انسانی در داده‌های تست بوده اما اشتباه تشخیص داده است. بیشترین معیار Recall مربوط گرایش پزشکی می‌باشد. با توجه به معیار F1-Score نیز که ترکیبی از دو معیار Precision و Recall می‌باشد، بهترین عملکرد نیز مربوط به گرایش فنی می‌باشد.

۵-۲ نتایج دسته‌بند بیز ساده

میزان دقت دسته‌بند بیز ساده برای Cross Validation برابر با مقدار ۵ را در جدول ۱۱ مشاهده می‌کنید.

جدول ۱۱. میانگین میزان دقت با استفاده از دسته‌بند بیز ساده

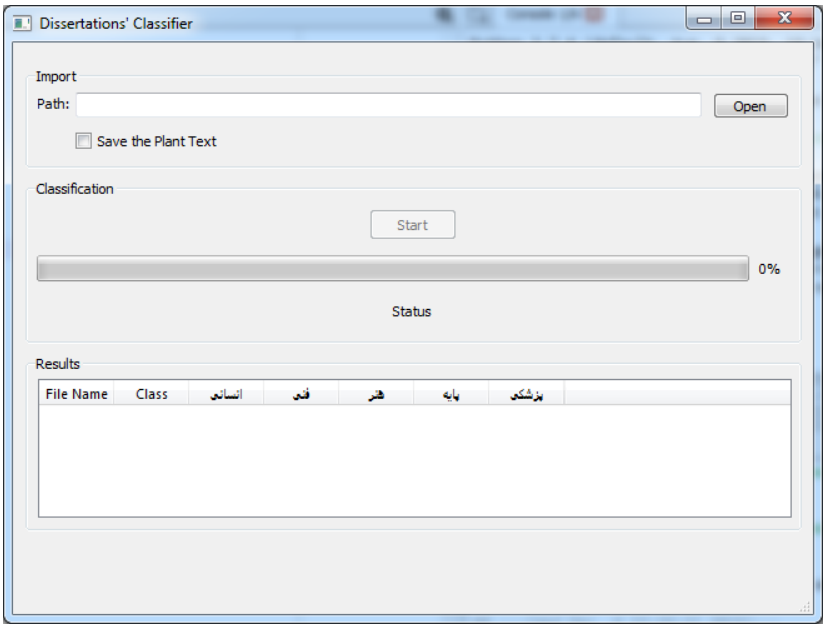
Cross validation	۱	۲	۳	٤	٥	میانگین
نتایج	۰,۶۱۹۰۴	۰,۵۷۱۴۲	۰,۶۴۵۱۶	۰,۶۱۲۹۰	۰,۶۷۷۴۱	۰,۶۲۵

همان گونه که از نتایج پیدا است، دسته‌بند بیز ساده در مقایسه با دسته‌بند ماشین بردارهای پشتیبان عملکرد ضعیف‌تری دارد و نتیج بدست آمده از آن قایل قبول نیز نمی‌باشد. در خاتمه، در (شکل‌های ۱۵ و ۱۶) نمونه‌هایی از رابط کاربری ایجاد شده برای استفاده از دسته‌بند پیشنهادی نمایش داده شده است.

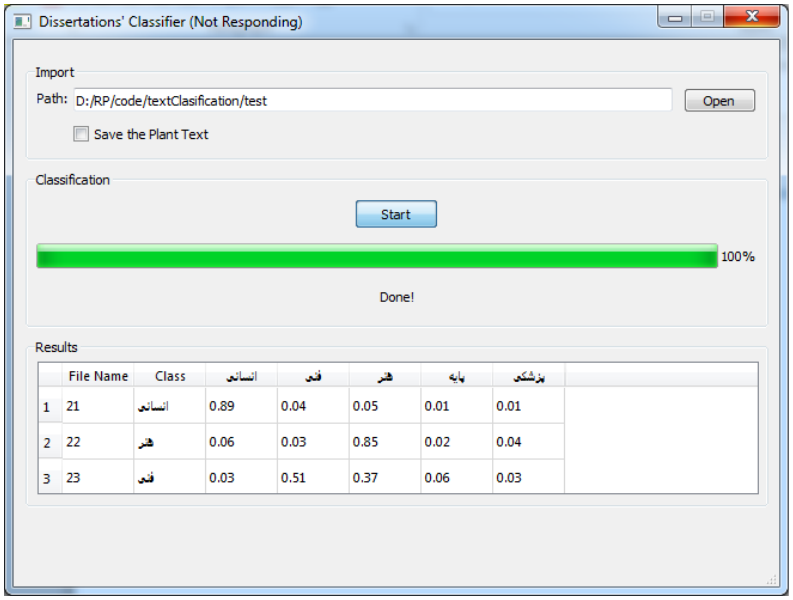
۵-۳ توصیه‌های کاربردی

با توجه به دقت کسب شده در دسته‌بندی متون علمی به نظر می رسد با گسترش الگوریتم توسعه یافته این امکان وجود دارد در سیستم‌هایی که بر مبنای محتوای متون علمی فارسی کار می کنند می تواند تاثیر مثبتی داشته باشد. به عنوان مثال در سامانه همانند جویی الگوریتم گسترش یافته می‌تواند به عنوان پیش‌پردازش انجام شود و بدینوسیله سرعت و دقت استخراج سرقت ادبی را بالا ببرد.

پیشنهاد کاربردی دیگر استفاده این الگوریتم در سامانه ثبت است. استفاده از روش‌های خودکار برای تصدیق اقلام داده‌ای که پژوهشگر وارد نموده است می‌تواند در افزایش سرعت فرآیند نمایه‌سازی نیمه‌خودکار کنونی کارا باشد. در واقع جایگاه این طرح پژوهشی کمک به نمایه‌ساز در جهت افزایش سرعت تایید یا عدم تایید صحت اقلام اطلاعاتی وارد شده می‌باشد. در این طرح پژوهشی "موضوع اصلی" متون علمی به صورت خودکار بوسیله استخراج محتوای متون علمی به صورت هوشمند سنجیده شد. این الگوریتم با استفاده از تشخیص خودکار فراداده که می تواند به عنوان طرح پژوهشی بعدی باشد گامی موثر برای افزایش سرعت نمایه سازی می‌باشد.



شکل ۱۵. تصویری از رابط کاربری ایجاد شده برای دسته‌بند پیشنهادی



شکل ۱۶. تصویری از رابط کاربری ایجاد شده برای دسته‌بند پیشنهادی

```
# -*- coding: utf-8 -*-
"""
Created on Mon Oct 21 13:47:48 2019

@author: jalal nasiri
"""
from PyQt5 import QtCore, QtGui, QtWidgets
from app import main
import sys

if __name__ == "__main__":
    main()

```

```
# -*- coding: utf-8 -*-
"""
Created on Tue Oct 22 11:28:50 2019

@author: jalal nasiri
"""

from PyQt5.QtWidgets import (QMainWindow, QApplication, QFileDialog,
                             QMessageBox, QGridLayout, QTableWidgetItem,
                             QDialog)
from PyQt5 import QtCore, QtGui, QtWidgets
from tc import ImportData, Preprocessor, classifier
from PyQt5 import QtCore, QtGui, QtWidgets
from PyQt5.QtCore import QThread, pyqtSlot
from GUI import gui
import sys
from os.path import expanduser

class TCApp(gui.Ui_MainWindow, QMainWindow):
    """ The main class to connect
    different parts of module: tc
    to GUI.

    functions
    -------------------
    Init_GUI
    get_data_path
    Classifier_trggr
    Display_Table
    Progress_Bar_hndle
    Display_Status
    set_pbar_range
    """
    def __init__(self):
        super(TCApp ,self).__init__()

        self.setupUi(self)

        self.user_in = None  # Stores user's data and input
```

```
40.         self.data_info = None
41.         self.init_GUI()
42.
43.         self.Classification_Start_Button.setEnabled(False)
44.
45.         self.prog = 0
46.         self.Classification_Progress_Bar.setRange(0, 100)
47.         self.Classification_Progress_Bar.setValue(self.prog)
48.         self.status ='status'
49.         self.imported_data_number=0
50.     def init_GUI(self):
51.         """
52.         Initialize the GUI of application
53.         """
54.
55.         # Threads
56.         self.__threads = []
57.
58.         # Buttons
59.         self.Path_Open_Button.clicked.connect(self.get_data_path)
60.         self.Classification_Start_Button.clicked.connect(self.Classifier_tr
    ggr)
61.         self.Classification_Start_Button.clicked.connect(self.Progress_Bar_
    hndle)
62.
63.
64.
65.     def get_data_path(self):
66.         """
67.         Gets the dataset path from a user.
68.         """
69.
70.         data_filename = QFileDialog.getExistingDirectory(self, "Open a fold
    er",
71.                                                          "",
72.                                                          QFileDialog.Sho
    wDirsOnly)
73.         if data_filename:
74.
75.             self.Address_Bar.setText(str(data_filename))
76.             self.Classification_Start_Button.setEnabled(True)
77.
78.             # Enable widgets in Read group box
79.             for w in self.Classification_Start_Button.children():
80.
81.                 if not isinstance(w, QGridLayout):
82.
83.                     w.setEnabled(True)
84.
85.     def Classifier_trggr(self):
86.         """
87.         Loads a dataset.
88.         """
89.
90.         self.data_reader = ImportData.Data_Reader(self.Address_Bar.text(),
    self.Address_Bar.text() , True) # , True if self.header_check.isChecked() e
    lse False
91.
92.         self.data_reader.Text_Maker_Save()
93.
94.         self.user_in , self.doc_names= self.data_reader.Corpus_Reader()
95.         self.prog =0
96.         self.Progress_Bar_hndle()
```

```
97.
98.         self.set_pbar_range(100)
99.         if (len(self.user_in))!=0:
100.                    self.Stop_Word_List = ''
101.                    self.PreProc = Preprocessor.PreProcessing(self.user_in,
    self.Stop_Word_List)
102.                    self.prog=10
103.                    self.Progress_Bar_hndle()
104.
105.                    self.status ='Pre-processing...'
106.                    self.Display_Status()
107.                    self.PreProc.String_Splitter()
108.                    self.prog =30
109.                    self.Progress_Bar_hndle()
110.
111.                    self.status ='Pre-processing...'
112.                    self.Display_Status()
113.                    self.PreProc.TextCleaner()
114.                    self.prog=70
115.                    self.Progress_Bar_hndle()
116.
117.                    self.status ='Pre-processing...'
118.                    self.Display_Status()
119.                    self.PreProc.wordToString()
120.                    self.prog=80
121.                    self.Progress_Bar_hndle()
122.
123.                    self.status ='Classification...'
124.                    self.Display_Status()
125.                    self.Preprocessed_Data = self.PreProc.Data_Vectorizer()

126.                    self.prog = 90
127.                    self.Progress_Bar_hndle()
128.
129.                    self.clf = classifier.InputClassifier(self.Preprocessed_
    Data)
130.                    self.status ='Done!'
131.                    self.Display_Status()
132.                    self.prog = 100
133.                    self.Progress_Bar_hndle()
134.
135.                    self.clf.SVCClassifier()
136.
137.                    self.Display_Table()
138.                else:
139.                    self.Classification_Start_Button.setEnabled(False)
140.                    self.status ='No valid file in the directory!'
141.                    self.Display_Status()
142.                    self.Result_Table.setRowCount(0)
143.
144.            def Display_Table(self):
145.                self.label_list = self.clf.Show_Label()
146.                self.no_samples = len(self.label_list)
147.                self.Proba_Matrix = self.clf.SVC_Proba()
148.                self.result_text = ''
149.                no_counter = 0
150.                _, col = self.Proba_Matrix.shape
151.                self.Result_Table.setRowCount(self.no_samples)
152.                for label in self.label_list:
153.                    no_counter += 1
154.                    self.Result_Table.setItem(no_counter-
    1, 0, QTableWidgetItem(str(no_counter)))
155.
```

```
                    if label== "fa":
                        self.result_text = "فنی"
                    if label== "pa":
                        self.result_text =  "پایه"
                    if label== "pe":
                        self.result_text = "پزشکی"
                    if label== "ho":
                        self.result_text = "هنر"
                    if label== "en":
                        self.result_text =  "انسانی"
                    self.Result_Table.setItem(no_counter-
    1, 1, QTableWidgetItem( self.result_text))
                    self.Result_Table.setItem(no_counter-
    1, 0, QTableWidgetItem(self.doc_names[no_counter-1]))

                    for i in range(col):
                        self.Result_Table.setItem(no_counter-
    1, i+2, QTableWidgetItem("%.2f"%self.Proba_Matrix[no_counter-1,i]))


            def Progress_Bar_hndle(self):

                self.Classification_Progress_Bar.setValue(self.prog)


            def Display_Status(self):
                self.Process_Status_Label.setText(self.status)


            @pyqtSlot(int)
            def set_pbar_range(self, pbar_rng):
                """
                Sets range of the progress bar.

                Parameters
                ----------

                """

                self.Classification_Progress_Bar.setRange(0, pbar_rng)

            @pyqtSlot(int, str, str, str)
            def update_gs_info(self, pbar_val): #  ,curr_acc, best_acc, elap
    sed_t
                """
                Updates current value of the progress bar.

                Parameters
                ----------

                """

                self.Classification_Progress_Bar.setValue(pbar_val)


        def main():

            app = QApplication(sys.argv)
            tcapp = TCApp()
            tcapp.show()
            sys.exit(app.exec_())
```

```
215.
216.       if __name__ == "__main__":
217.           main()
```

```
1.  # -*- coding: utf-8 -*-
2.  """
3.  Created on Sat Oct 19 20:34:07 2019
4.
5.  @author: jalal nasiri
6.  """
7.  import ntpath
8.  import docx
9.  import os
10. import re
11. from os.path import splitext, split
12.
13.
14. class Data_Reader:
15.     '''''
16.     Data_Reader Class:
17.     '''
18.     def __init__(self, path, save_path, keep_text):
19.         self.path = path
20.         self.save_path = save_path
21.         self.keep_text = keep_text
22.         self.doc_names =  []
23.
24.
25.     def Corpus_Reader(self):
26.         string_corpus = []
27.         set_class = ''
28.
29.         for filename in os.listdir(self.path):
30.             f_name , f_ext = splitext(filename)
31.             if f_ext==".txt":
32.                 self.doc_names.append(f_name)
33.                 file_path = self.path+'/'+filename
34.                 loaded_file = open(file_path,'r',encoding = "utf-8")
35.                 line_seperated_data = loaded_file.readlines()
36.                 set_class = str(filename[:2])
37.                 string_corpus.append([line_seperated_data[0],set_class])
38.
39.         return string_corpus, self.doc_names
40.
41.
42.
43.
```

```
    def Docx_Text_Convertor(self,file_path):

        saving_path = self.path+'/'   # it must change in future.

        preFix= ''
        seperator=' '
        temp=''
        file_name_ = ntpath.basename(file_path)
        file_name_wF, f_ext = splitext(file_name_)
        # print(file_path)
        try:
            doc = docx.Document(file_path)
            temp_doc = []
            for i in doc.paragraphs:
                temp = i.text.split()
                for j in temp:
                    temp_doc.append(j)
            temp_doc = seperator.join(temp_doc)
            textfile_name=str(saving_path)+preFix+ file_name_wF + ".txt"
            with open(textfile_name, 'w',encoding='utf8') as f:
                for item in temp_doc:
                    f.write("%s" % item)
        except:
            raise

    def Text_Maker_Save(self):
        path_string= self.path

        for filename in os.listdir(path_string):
            _ , f_ext = splitext(filename)
            if (f_ext == '.txt'):
                pass

            elif(f_ext == '.docx'):
                self.Docx_Text_Convertor(path_string+"/"+filename)
            elif(f_ext!=''):
                print(filename," file will not be processed because its for
mat is not accepted by this software. ( only accepts .txt and .docx )")
            else:
                pass


    def Save_Text_File(self):
        if self.keep_text:
            pass
```

```
# -*- coding: utf-8 -*-
"""
Created on Sun Oct 20 10:31:16 2019

@author: jalal nasiri
```

```
"""
import numpy as np
import joblib
from sklearn import svm

class InputClassifier:
    """InputClassifier
    Functions
    -------------
    Show_Label
    SVCClassifier
    SVC_Proba
    Display_Proba
    """

    model = joblib.load('./tc/model/SVCmodel_Proba.sav')
    labels = joblib.load('./tc/model/label_encoder.joblib')
    def __init__(self, imported_data):
        self.imported_data = imported_data
        self.Fin_Results = []
        self.clf_labels = ['انسانی','فنی','هنر','پایه','پزشکی']

    def Show_Label(self):
        le = InputClassifier.labels
        return list(le.inverse_transform(self.Fin_Results))

    def SVCClassifier(self):
        self.Fin_Results = InputClassifier.model.predict(self.imported_data
)
        return self.Fin_Results

    def SVC_Proba(self):
        return InputClassifier.model.predict_proba(self.imported_data)

    def Display_Proba(self):
        Proba_text = ''
        Proba_M = self.SVC_Proba()
        I , J = Proba_M.shape
        for i in range(I):
            Proba_text +='%d :  |  '%(i+1)
            for j in range(J):
                Proba_text = Proba_text + self.clf_labels[j] + ": %.2f " %
Proba_M[i,j] + '  |  '
            Proba_text += '\n'
        return Proba_text
```

```
# -*- coding: utf-8 -*-
"""
Created on Mon Oct 21 11:41:54 2019

@author: jalal nasiri
"""

import numpy as numpy
import pandas as pd
from hazm import Stemmer
```

```
from hazm import Lemmatizer
import nltk
import pickle
import joblib
import re
from string import punctuation
from collections import OrderedDict
from collections import Counter, defaultdict
import collections
from sklearn import preprocessing
from sklearn.preprocessing import LabelEncoder
from sklearn.feature_extraction.text import CountVectorizer,\
                            TfidfTransformer, TfidfVectorizer


class PreProcessing:
    """Preprocessing Class

    Functions
    --------------
    TextCleaner
    String_Splitter
    wordToString
    Data_Vectorizer
    """
    def __init__(self, imported_data,stopwords_list):
        self.imported_data = imported_data
        self.stopwords_list = stopwords_list
        self.stringList=[]

    def TextCleaner(self):
        self.stopwordsList= ''
        Data = self.imported_data
        stemmer = Stemmer()
        lemmatizer = Lemmatizer()
        dataList = Data
        table = str.maketrans('', '', punctuation)

        for i in range(0, len(dataList)):
            for j in range(0, len(dataList[i][0])):
                dataList[i][0][j] = stemmer.stem(dataList[i][0][j])

                dataList[i][0][j] = lemmatizer.lemmatize(dataList[i][0][j])

            dataList[i][0] = [word for word in dataList[i][0] if word.isalp
ha()]
            dataList[i][0]= [w.translate(table) for w in dataList[i][0]]
            dataList[i][0] = [word for word in dataList[i][0] if len(word)
> 3]
        self.imported_data = dataList
        return self.imported_data

    def String_Splitter(self):
        for i in range(len(self.imported_data)):
            self.imported_data[i][0]=self.imported_data[i][0].split()
        return self.imported_data

    def wordToString(self):
        self.stringList = []
        for i in range(0,len(self.imported_data)):
            self.stringList.append(' '.join(word for word in self.imported_
data[i][0]))
        self.imported_data = self.stringList
```

```
69.         return self.imported_data
70.
71.     def Data_Vectorizer(self):
72.         count_vect = CountVectorizer(decode_error="replace",\
73.                                      vocabulary=pickle.load(open("./tc/model
    /count_vect.pkl", "rb")))  # count_vect.pkl
74.         X_train_counts = count_vect.fit_transform(self.imported_data)
75.         tf_transformer = TfidfTransformer().fit(X_train_counts) # use_idf=F
    alse
76.         X_train_tf = tf_transformer.transform(X_train_counts)
77.
78.         return X_train_tf
79.
80.
```

فهرست منابع

A.Y. Ng and M.I. Jordan, “On Discriminative vs. Generative classifiers: A comparison of logistic regression and naive Bayes,” Neural Information Processing Systems (NIPS), vol. 12, 2002.

B. Liu, "Sentiment analysis and opinion mining," *Synthesis Lectures on Human Language Technologies*, vol. 5, pp. 1-167, 2012.

B. Agarwal and N. Mittal, *Prominent Feature Extraction for Sentiment Analysis*. Berlin: Springer International Publishing, 2016.

B. Blamey, T. Crick, and G. Oatley word-gram feature selection for sentiment classification of OSN corpora," in *Research and Development in Intelligent Systems XXIX*, ed: Springer, 2012, pp. 207-212.

C. Cortes and V. Vapnik, "Support-vector networks," Machine learning, vol. 20, pp. 273-297, 1995.

C. F. Lin and S. D. Wang, "Fuzzy support vector machines," Neural Networks, IEEE Transactions on, vol. 13, pp. 464-471, 2002.

C. C. Hsu, et al., "Fuzzy support vector machines with the uncertainty of parameter C," Expert Systems with Applications, vol. 36, pp. 6654-6658, 2009.

C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to information retrieval* vol. 1: Cambridge University Press, 2008.

C. Lin and S. Wang, "Fuzzy support vector machines with automatic membership setting," Support Vector Machines: Theory and Applications, pp. 629-629, 2005.

D. Kay, E. Unruh, and G. Tandon, "Spell-check for a keyboard system with automatic correction," ed: Google Patents, 2012.

Fiori, Alessandro, ed. Innovative document summarization techniques: Revolutionizing knowledge understanding: Revolutionizing knowledge understanding. IGI Global, 2014.

F. Oroumchian, S. Tasharofi, H. Amiri, H. Hojjat, and F. Raja, "Creating a feasible corpus for Persian POS tagging," *Department of Electrical and Computer Engineering, University of Tehran*, 2006.

G. Paltoglou and M. Thelwall, "A study of information retrieval weighting schemes for sentiment analysis," in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, 2010, pp. 1386-1395.

G. Heo and P. Gader, "Fuzzy SVM for noisy data: A robust membership calculation method," Fuzzy Systems (FUZZ), pp. 431-436,2009.

G. Forman, "An extensive empirical study of feature selection metrics for text classification," *The Journal of machine learning research*, vol. 3, pp. 1289-1305, 2003.

G. Naik, et al., "Twin SVM for gesture classification using the surface electromyogram test addition," IEEE Transactions on Information Technology in Biomedicine, vol. 14, pp. 301-308, 2011.

H. S. Yazdi, et al., "The probabilistic constraints in the support vector machine," Applied Mathematics and Computation, vol. 194, pp. 467-479, 2007.

H. Tang and L. Qu, "Fuzzy support vector machine with a new fuzzy membership function for pattern classification," pp. 768-773, 2008.

H. Jadidinejad, F. Mahmoudi, and J. Dehdari, "Evaluation of PerStem: a simple and efficient stemming algorithm for Persian," presented at the Workshop of the Cross-Language Evaluation Forum for European Languages, 2010.

I. Habernal, T. Ptáček, and J. Steinberger, "Reprint of "Supervised sentiment analysis in Czech social media"," *Information Processing & Management,* vol. 51, pp. 532-546, 2015.

J. A. Nasiri, et al., "Intelligent Arrhythmia Detection Using Genetic Algorithm and Emphatic SVM (ESVM)," Third UKSim European Symposium, pp. 112-117, 2009.

J. Martineau and T. Finin, "Delta TFIDF: An Improved Feature Space for Sentiment Analysis," presented at the Proceedings of the Third International ICWSM Conference, 2009.

K. Mozafari, et al., "Action Recognition by Local Space-Time Features and Least Square Twin SVM (LS-TSVM)," in Informatics and Computational Intelligence (ICI), First International Conference onpp. 287-292, , 2011.

K. Mozafari, et al., "Action Recognition by Local Space-Time Features and Least Square Twin SVM (LS-TSVM)," in Informatics and Computational Intelligence (ICI), First International Conference onpp. 287-292, , 2011.

K. Mozafari, et al., "Hierarchical Least Square Twin Support Vector Machines Based Framework for Human Action Recognition," in Machine Vision and Image Processing (MVIP), 7th Iranian, 2011, pp. 1-5, 2011.

Lott, Brian. "Survey of keyword extraction techniques." UNM Education 50 (2012).

M. Arun Kumar and M. Gopal, "Least squares twin support vector machines for pattern classification," Expert Systems with Applications, vol. 36, pp. 7535-7543, 2009.

M. Abdul-Mageed, M. Diab, and S. Kübler, "SAMAR: Subjectivity and sentiment analysis for Arabic social media," *Computer Speech & Language,* vol. 28, pp. 20-37, 2014.

M. Arun Kumar, et al., "Knowledge based Least Squares Twin support vector machines," Information Sciences, vol. 180, pp. 4606-4618, 2010.

M. Bijankhan, "The role of the corpus in writing a grammar: An introduction to a software," *Iranian Journal of Linguistics,* vol. 19, 2004.

M. Shamsfard, "Challenges and open problems in Persian text processing," *Proceedings of LTC,* vol. 11, 2011.

M. Shamsfard, H. S. Jafari, and M. Ilbeygi, "STeP-1: A Set of Fundamental Tools for Persian Text Processing," in *LREC*, 2010.

M. Seraji, B. Megyesi, and J. Nivre, "A basic language resource kit for Persian," in *Eight International Conference on Language Resources and Evaluation (LREC 2012), 23-25 May 2012, Istanbul, Turkey*, 2012, pp. 2245-2252.

M. Khallash and M. Imany. (2014). *Hazm: Python library for digesting Persian text*. Available: https://github.com/sobhe/hazm

M. Manshadi. (2013). *Farsi Verb Tokenizer*. Available: https://github.com/mehdi-manshadi/Farsi-Verb-Tokenizer

M. Seraji, "PrePer: A Pre-processor for Persian," in Proceedings of The Fifth International Conference on Iranian Linguistics (ICIL5), Bamberg, Germany, 2013.

M. Sabzekar, et al., "Relaxed constraints support vector machine," Expert Systems, 2011.

M. M. Adankon and M. Cheriet, "Help-Training for semi-supervised support vector machines," Pattern Recognition, 2011.

R. Khemchandani and S. Chandra, "Twin support vector machines for pattern classification," Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 29, pp. 905-910, 2007.

S. Zhao, et al., "A New Fuzzy Support Vector Regression Method," Journal of Qingdao University(Natural Science Edition), vol. 23, pp. 47-51, 2010.

S. M. Mohammad, S. Kiritchenko, and X. Zhu, "NRC-Canada: Building the state-of-the-art in sentiment analysis of tweets," presented at the 7th International Workshop on Semantic Evaluation (SemEval 2013), 2013.

W. Feely, M. Manshadi, R. Frederking, and L. Levin, "The CMU METAL Farsi NLP Approach," in *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, 2014, pp. 4052-4055.

W. Nie and D. He, "A probability approach to anomaly detection with twin support vector machines," Journal of Shanghai Jiaotong University (Science), vol. 15, pp. 385-391, 2010.

Y. Saeys, I. Inza, and P. Larrañaga, "A review of feature selection techniques in bioinformatics," *bioinformatics*, vol. 23, pp. 2507-2517, 2007.

Y. H. Shao, et al., "Improvements on Twin Support Vector Machines," Neural Networks, IEEE Transactions on, pp. 1-1, 2011.

Z. Sarabi, H. Mahyar ,and M. Farhoodi, "ParsiPardaz: Persian Language Processing Toolkit," in *Computer and Knowledge Engineering (ICCKE), 2013 3th International eConference on*, 2013, pp. 73-79.

Abstract

Due to the high volume of scientific texts and the low speed of control of information items in the SABT system, increasing the speed of controlling information items is one of the priorities of the SABT system. For example, about 140,000 scientific texts are still not indexed, and the use of methods that can help increase the speed of indexing is one of the priorities of the SABT system.

The dissertations that are registered in the database of the Iranian Institute of Information Science and Technology (IranDoc) have the main subject such as humanities, technical and engineering, basic sciences, arts and architecture, medical sciences, etc. Automatic extracting of main subject of scientific texts by machine learning algorithms speeds up the indexing process and increases the quality of GANJ system.

In this research project, the goal is classification of scientific texts in to five class of Humanities, Engineering, Basic Sciences, Arts and Architecture, Medical Sciences. In this research, first, the usual pre-processing in natural language processing has been done and Proper features are extracted. The results of the evaluations on the Persian scientific texts show this method, by evaluating with more texts, can determine the main subject of the Persian scientific texts with great accuracy and speed.

Keywords: Multi-Class Classification; Automated Indexing; Unbalanced Data; Pattern Recognitio

Iranian Research Institute
for Information Science and Technology (Irandoc)

Research project

Designing an intelligent method for classification of Persian scientific documents

By:

Jalal A. Nasiri

Nasrollah Pakniat

Spring 2020