



پردیس علوم
دانشکده ریاضی، آمار و علوم کامپیوتر

بررسی رابطه بین متن نوشته شده توسط یک فرد و خودکشی کردن او با استفاده از متدهای کاوش متن و یادگیری ماشین

نگارنده

نرگس بابااحمدی

استاد راهنما: باقر باباعلی

پایاننامه برای دریافت درجه کارشناسی
در رشته علوم کامپیوتر

تیر ۱۴۰۲

چکیده

خودکشی یکی از چالش‌های اصلی سلامت عمومی است و شناسایی زودهنگام افراد در معرض خطر خودکشی می‌تواند گامی مهم در پیشگیری از آن باشد. با این حال، شناسایی افراد در معرض خطر خودکشی یک وظیفه چالش برانگیز است زیرا رفتار این افراد معمولاً پدیده پیچیده و چند وجهی است. در سال‌های اخیر، تکنیک‌های داده کاوی متن^۱ و یادگیری ماشین^۲ برای تجزیه و تحلیل داده‌های متنی و شناسایی الگوهای نشان‌دهنده رفتار خودکشی استفاده شده است. هدف این پایان‌نامه، توسعه یک مدل با دقت بالا برای شناسایی متنی که نشانگر رفتار منجر به خودکشی هستند با استفاده از تکنیک‌های داده کاوی متن و یادگیری ماشین است. به منظور دستیابی به این هدف، از داده‌های متنی موجود در رسانه اجتماعی Reddit^۳ استفاده شده است، زیرا این رسانه یک منبع غنی از داده‌های متنی است که می‌تواند برای تحلیل زبان افرادی که ممکن است در معرض خطر خودکشی باشند، استفاده شود. ترکیبی از تکنیک‌های یادگیری ماشین و یادگیری عمیق شامل ماشین بردار پشتیبان، جنگل تصادفی و شبکه‌های عصبی برای توسعه مدل استفاده شده است. این پایان‌نامه می‌تواند به عنوان پایه‌ای برای تحقیقات بیشتر در این حوزه و ارائه راهکارهای بهبود سلامت عمومی افراد مورد استفاده قرار گیرد.

¹Text Mining

²Machine Learning

³Reddit

سپاسگزاری

از استاد راهنمای گرانقدر جناب آفای دکتر بابا علی که با راهنمایی های خود مرا در انجام این پژوهه
یاری دادند کمال تشکر را دارم.

پیشگفتار

موضوع خودکشی یکی از مهمترین موضوعاتی است که در حوزه سلامت عمومی بررسی می‌شود. خودکشی یکی از عوامل مهم در کاهش کیفیت زندگی افراد و حتی ایجاد مشکلات جدی برای جامعه است. با توجه به اینکه خودکشی به صورت غیرمستقیم و بدون هشدار قبلی اتفاق می‌افتد، شناسایی زودهنگام آن بسیار حائز اهمیت است.

امروزه، با پیشرفت فناوری و کاهش هزینه‌های داده کاوی و یادگیری ماشین، استفاده از این روش‌ها به عنوان یک راهکار جدید در شناسایی افراد در معرض خطر خودکشی مطرح شده است. در این راستا، تحلیل داده‌های متّنی می‌تواند به عنوان یکی از روش‌های مؤثر در شناسایی افراد در معرض خطر خودکشی مورد استفاده قرار گیرد.

در این پایان نامه، با بررسی ادبیات و تحلیل داده‌های موجود در پلتفرم‌های آنلاین، مدلی با دقت بالا برای شناسایی افراد در معرض خطر خودکشی پیشنهاد شده است. برای این منظور، داده‌های متّنی مختلف از جمله متون نوشته شده توسط افراد در شبکه‌های اجتماعی، ایمیل‌های ارسال شده و پیام‌های متّنی ارسال شده به دیگران، جمع‌آوری و تحلیل شده‌اند. سپس با استفاده از متدهای یادگیری ماشین و همچنین شبکه‌های عصبی^۴، الگوهای مشترک در این متون شناسایی شده و مدلی برای شناسایی افراد در معرض خطر توسعه داده شده است.

⁴Neural Networks

فهرست مطالب

۱	۱	۱	۱ مفاهیم مقدماتی
۱	۱	۱.۱	۱.۱ مقدمه ای بر یادگیری ماشین
۲	۱.۱.۱	۱.۱.۱	۱.۱.۱ الگوریتم های یادگیری ماشین
۵	۲.۱.۱	۲.۱.۱	۲.۱.۱ الگوریتم های طبقه بندی
۱۳	۲.۱	۲.۱	شبکه های عصبی
۱۴	۱.۲.۱	۱.۲.۱	۱.۲.۱ ساختار شبکه های عصبی
۱۵	۳.۱	۳.۱	شبکه های عصبی از پیش آموزش داده شده
۱۷	۴.۱	۴.۱	تنظیم پارامتر
۱۷	۱.۴.۱	۱.۴.۱	۱.۴.۱ جستجوی خطی
۱۸	۲.۴.۱	۲.۴.۱	۲.۴.۱ جستجوی تصادفی
۱۹		۲	دادگان
۱۹	۱.۲	۱.۲	۱.۲ توصیف دادگان
۲۱	۲.۲	۲.۲	۲.۲ پیش پردازش دادگان
۲۶	۳.۲	۳.۲	۳.۲ تقسیم بندی دادگان
۲۷		۳	آزمایش ها و نتیجه ها
۲۹	۱.۳	۱.۳	۱.۳ رگرسیون
۲۹	۱.۱.۳	۱.۱.۳	۱.۱.۳ رگرسیون خطی ساده
۳۳	۲.۱.۳	۲.۱.۳	۲.۱.۳ رگرسیون لجستیک
۳۷	۲.۳	۲.۳	۲.۳ طبقه بند بردار پشتیبان خطی
۳۸	۳.۳	۳.۳	۳.۳ الگوریتم جنگل تصادفی
۴۲	۴.۳	۴.۳	۴.۳ دسته بند بیز ساده
۴۵	۵.۳	۵.۳	۵.۳ شبکه های عصبی
۴۵	۱.۵.۳	۱.۵.۳	۱.۵.۳ شبکه عصبی ساده
۴۶	۲.۵.۳	۲.۵.۳	۲.۵.۳ شبکه های از پیش آموزش داده شده

٤ جمع بندی

٤٧

ث

فصل ۱

مفاهیم مقدماتی

۱.۱ مقدمه‌ای بر یادگیری ماشین

یادگیری ماشین به عنوان یکی از حوزه‌های مهم هوش مصنوعی^۱، امکانات بسیاری را برای ما فراهم می‌کند. این روزها، بسیاری از کارها و اتفاقات در اطراف ما، توسط ماشین‌ها و بدون دخالت انسان‌ها انجام می‌شود و بعضاً بازدهی بیشتری و عملکرد دقیق‌تری نسبت به انسان‌ها دارند [۱]. آیا این به معنای این است که ماشین‌ها فکر می‌کنند؟ آلن تورینگ، با مطرح کردن این سوال در یکی از مقالات خود در سال ۱۹۵۰، نقطه شروعی برای پژوهش‌ها و فعالیت‌های در زمینه هوش مصنوعی قرار داد.

در دهه‌ی ۱۹۶۰، پیدایش شبکه‌های عصبی، باعث شد که یادگیری ماشین به یک سطح جدید از پیچیدگی و کاربرد برسد. در این دهه، الگوریتم‌های جدیدی نیز مانند الگوریتم‌های بیزی^۲ و الگوریتم‌های خوشبندی^۳، ارائه شدند.

در دهه‌ی ۱۹۹۰، با پیدایش اینترنت و تولید داده‌های بزرگ، اهمیت یادگیری ماشین به طور چشمگیری افزایش یافت. در این دهه، الگوریتم‌های جدیدی نیز مانند شبکه‌های عصبی عمیق^۴، الگوریتم‌های ماشین بردار پشتیبان^۵ و الگوریتم‌های شبکه‌های مولد^۶، ارائه شدند [۲]. در دهه‌ی اخیر، با پیشرفت تکنولوژی‌های پردازشی و تولید داده‌های بیشتر، یادگیری ماشین به یکی از حیاتی‌ترین حوزه‌های علمی و کاربردی تبدیل شده است. به طور کلی، این حوزه به دلیل کاربردهای فراوانی که در حوزه‌های مختلف دارد، همچنان در حال توسعه و پیشرفت است.

¹Artificial Intelligence

²Bayesian Algorithms

³Clustering Algorithms

⁴Deep Neural Networks

⁵Support Vector Machine

⁶Generative Adversarial Networks

این پروژه با استفاده از متدهای یادگیری ماشین و یادگیری عمیق انجام شده است. در ادامه به تعدادی از این مفاهیم می‌پردازیم.

۱۰.۱ الگوریتم‌های یادگیری ماشین

یادگیری نظارت شده

الگوریتم یادگیری نظارت شده^۷ یکی از رایج‌ترین الگوریتم‌های یادگیری ماشین است. در این الگوریتم، مدل یادگیری ماشین از داده‌های آموزشی با برچسب استفاده می‌کند تا بتواند روی داده‌های جدیدی که برچسب ندارند، پیش‌بینی کند^[۳].

در این الگوریتم، داده‌ها به دو بخش تقسیم می‌شوند: داده‌های آموزشی و داده‌های تست. داده‌های آموزشی شامل ورودی‌هایی است که برچسب یا پاسخ صحیح دارند، به عنوان مثال، تصاویری از حیوانات با برچسب‌های "گربه" و "سگ". در این الگوریتم، مدل یادگیری ماشین از داده‌های آموزشی استفاده می‌کند تا یاد بگیرد که هر نوع تصویر، چه برچسبی دارد^[۴]. سپس با استفاده از این دانش، می‌تواند برای تصاویر جدیدی که برچسب ندارند، پیش‌بینی کند که برچسب آن‌ها چیست^[۵].

برای آموزش مدل یادگیری نظارت شده، ابتدا باید یک تابع هدف^۸ انتخاب کرد. این تابع باید درستی پیش‌بینی مدل را به عنوان یک عدد تخمینی برای هر داده آموزشی محاسبه کند. سپس مدل با استفاده از روش‌های بهینه‌سازی، مانند کاوش گرادیان^۹، سعی می‌کند تا این تابع هدف را بیشینه کند. با این کار، دقت پیش‌بینی مدل برای داده‌های آموزشی افزایش می‌یابد.

بعد از آموزش مدل با داده‌های آموزشی، می‌توان از آن برای پیش‌بینی برچسب داده‌های تست استفاده کرد. با این کار، می‌توان دقت پیش‌بینی مدل را برای داده‌های جدید ارزیابی کرد. در صورتی که دقت پیش‌بینی مدل برای داده‌های تست نیز به اندازه کافی بالا باشد، می‌توان آن را برای پیش‌بینی برچسب داده‌های جدید استفاده کرد.

⁷Supervised Learning

⁸Objective Function

⁹Gradient Descent

چند نمونه از الگوریتم‌های یادگیری نظارت شده عبارتند از [۶] :

- الگوریتم درخت تصمیم^{۱۰}
- الگوریتم نزدیک‌ترین همسایه^{۱۱}
- الگوریتم ماشین بردار پشتیبان
- الگوریتم رگرسیون خطی^{۱۲}
- الگوریتم رگرسیون لجستیک^{۱۳}

یادگیری نظارت نشده

الگوریتم یادگیری نظارت نشده^{۱۴} یک الگوریتم یادگیری ماشین است که در آن داده‌هایی بدون برچسب به مدل یادگیری داده می‌شوند و مدل باید الگوهایی که در داده‌ها وجود دارد را شناسایی کند. در این الگوریتم، مدل به صورت خودکار الگوهایی را کشف می‌کند که در داده‌ها وجود دارد و این الگوها را برای تکرار در آینده استفاده می‌کند [۷].

در الگوریتم یادگیری نظارت نشده، داده‌ها به دو بخش تقسیم می‌شوند: داده‌های آموزشی و داده‌های تست. در بخش آموزشی، مدل با استفاده از داده‌های بدون برچسب، سعی می‌کند الگوهایی را که در داده‌ها وجود دارد، شناسایی کند. به عنوان مثال، در صورت داشتن تصاویری از طبیعت بدون برچسب، مدل باید بتواند الگوهایی مانند درختان، آسمان، آفتاب، آسیاب‌های بادی و غیره را شناسایی کند. برای این منظور، از الگوریتم‌های یادگیری نظارت نشده مانند مدل‌های خوشبندی استفاده می‌شود.

یکی از روش‌های مهم یادگیری نظارت نشده در داده‌های متنی، استفاده از مدل‌های مبتنی بر ورودی است. در این روش، برای هر کلمه در داده‌های متنی، یک بردار ویژگی مشخص می‌شود که شامل ویژگی‌هایی مانند تعداد تکرار کلمه در متن، تعداد کلمات هم معنی، تعداد کلمات مشابه در متن و غیره است. با استفاده از این بردار ویژگی، می‌توان با الگوریتم‌های خوشبندی، مدل‌های یادگیری ماشین را آموزش داد و به شناسایی الگوها و الگوهایی که در داده‌ها وجود دارد، پرداخت.

¹⁰Decision Tree

¹¹k-Nearest Neighbors

¹²Linear Regression

¹³Logistic Regression

¹⁴Unsupervised Learning

یادگیری نیمه نظارتی

یادگیری نیمه نظارتی^{۱۵} یک شیوه یادگیری است که در آن، برخلاف یادگیری نظارتی که دارای برچسب برای داده‌ها است، برخی از داده‌ها بدون برچسب و برخی دیگر دارای برچسب هستند. هدف در یادگیری نیمه نظارتی، تخمین برچسب داده‌های بدون برچسب با استفاده از داده‌های دارای برچسب است.

یادگیری نیمه نظارتی در بسیاری از موارد، به دلیل اینکه بدون نیاز به برچسب داده‌های بیشتر، قابل اجرا است، بسیار مفید است. به عنوان مثال، در بسیاری از مسائل دسته‌بندی، بعضی از داده‌ها دارای برچسب هستند و بعضی دیگر ندارند. با استفاده از داده‌های دارای برچسب، می‌توان یک مدل یادگیری نظارتی ایجاد کرد و با استفاده از آن، برای داده‌های بدون برچسب، برچسب تخمین زد.

برای یادگیری نیمه نظارتی، روش‌های استفاده از داده‌های دارای برچسب و بدون برچسب وجود دارد. یکی از روش‌های معمول، استفاده از شبکه‌های عصبی با نوع خاصی از لایه‌ها است که به نام لایه‌های نیمه نظارتی^{۱۶} معروفند. این لایه‌ها به گونه‌ای طراحی شده‌اند که بتوانند از داده‌های دارای برچسب و بدون برچسب به طور همزمان استفاده کنند و مدلی را ایجاد کنند که بتواند برای داده‌های بدون برچسب، برچسب تخمینی تولید کند.

یکی دیگر از روش‌های معمول در یادگیری نیمه نظارتی، استفاده از الگوریتم‌های خوشه‌بندی است. در این روش، ابتدا داده‌های دارای برچسب به گروه‌های خاصی تقسیم می‌شوند و سپس داده‌های بدون برچسب با استفاده از خوشه‌بندی، در گروه‌هایی با شباهت بالا قرار می‌گیرند. سپس برای هر گروه، یک مدل یادگیری نظارتی ایجاد می‌شود و با استفاده از این مدل‌ها، برای داده‌های بدون برچسب، برچسب تخمینی تولید می‌شود.

در کل، یادگیری نیمه نظارتی یک شیوه یادگیری مفید برای مسائلی است که دارای داده‌های بدون برچسب هستند و یا به دلیل هزینه بالای برچسب‌گذاری داده‌ها، امکان استفاده از داده‌های بیشتر با برچسب نیست. با استفاده از این شیوه یادگیری، می‌توان به دقت و کارایی بهتری رسید و در بسیاری از موارد، نیاز به داده‌های بیشتر با برچسب را کاهش داد.

یادگیری تقویتی

یادگیری تقویتی^{۱۷} در واقع یکی از شاخه‌های مهم یادگیری ماشین است که به شیوه تعاملی و با دنبال کردن یک رویه عملکرد، عامل یادگیری ماشین با محیط تعامل دارد و با هدف حصول بیشینه پاداش از محیط، به یادگیری می‌پردازد. در واقع، یادگیری تقویتی به عنوان یک فرآیند تعاملی است که در آن، عامل باید با اعمال یک عمل به محیط، بیشینه پاداش را به دست آورد.

در یادگیری تقویتی، عامل با دیدن وضعیت فعلی محیط، با توجه به سیاستی که دارد، یک عمل

¹⁵Semi-Supervised Learning

¹⁶Semi-Supervised Layers

¹⁷Reinforcement Learning

را انتخاب می‌کند و برای اعمال آن به محیط، پاداشی از محیط دریافت می‌کند. با گذشت زمان و انجام اعمال مختلف، عامل می‌تواند با تجربه، یاد بگیرد که در وضعیت‌های مختلف، کدام عمل بهترین پاداش را با خود به همراه دارد.

در یادگیری تقویتی، محیط می‌تواند به صورت یک بازی، یک ربات صنعتی، یا حتی یک محیط مجازی باشد. وضعیت، شرایط فعلی محیط را توصیف می‌کند. برای مثال، در یک بازی، وضعیت شامل موقعیت بازیکن، تعداد جان بازیکن، امتیاز بازیکن و غیره است. عمل، هرگونه اقدام یا تصمیمی است که عامل در محیط انجام می‌دهد. برای مثال، در یک بازی، عمل می‌تواند شامل حرکت کردن بازیکن، حمله به حریف و غیره باشد.

در یادگیری تقویتی، سیاست، مجموعه‌ای از قوانین و روابه‌هایی است که عامل با توجه به وضعیت جاری، یک عمل را انتخاب می‌کند. پاداش، امتیازی است که عامل به دست می‌آورد و اندازه‌گیری می‌کند که عملی که انجام داده است، مفید واقع شده است یا خیر. تابع ارزش، یک تخمین از میزان پاداشی است که می‌توان با انجام یک عمل در یک وضعیت خاص، به دست آورد. این تابع برای هر وضعیت و برای هر عمل، یک ارزش تخمینی را محاسبه می‌کند که برای یادگیری تقویتی بسیار مهم است.^[۸]

۲۰.۱ الگوریتم‌های طبقه‌بندی

الگوریتم‌های طبقه‌بندی^{۱۸} هستند که برای مسئله دسته‌بندی داده‌ها به کلاس‌های مختلف استفاده می‌شوند. در ادامه، به برخی از این الگوریتم‌ها اشاره می‌شود.

رگرسیون خطی ساده

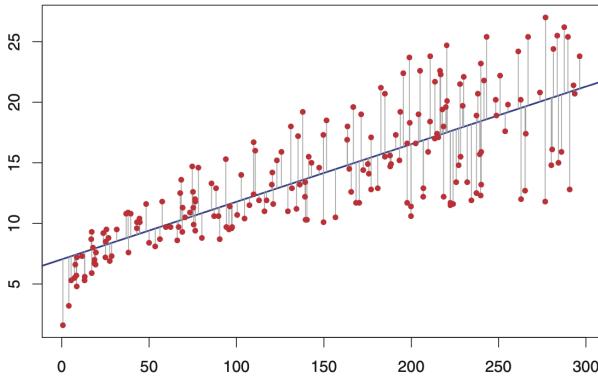
رگرسیون^{۱۹} روشی برای مدل‌سازی یک مقدار هدف بر اساس پیش‌بینی‌کننده‌های مستقل است. این روش بیشتر برای پیش‌بینی و یافتن رابطه علت و معلولی بین متغیرها استفاده می‌شود. تکنیک‌های رگرسیون عمدهاً بر اساس تعداد متغیرهای مستقل و نوع رابطه بین متغیرهای مستقل و وابسته متفاوت است.

رگرسیون خطی ساده نوعی تحلیل رگرسیونی است که در آن تعداد متغیرهای مستقل یک است و بین متغیر مستقل (x) و متغیر وابسته (y) رابطه خطی وجود دارد. خط قرمز در نمودار ۱.۱ به عنوان بهترین برازش خط مستقیم نامیده می‌شود. بر اساس نقاط داده شده، سعی می‌کنیم خطی را ترسیم کنیم که نقاط را به بهترین شکل برازش کند. خط را می‌توان بر اساس معادله خطی نشان داده شده در زیر مدل کرد.^[۹]

$$y = \alpha + \beta x \quad (1.1)$$

¹⁸Classification Algorithms

¹⁹Regression



شکل ۱.۱: بهترین برازش خطی [۳۲]

انگیزه الگوریتم رگرسیون خطی، یافتن بهترین مقادیر برای α و β است. برای درک بهتر رگرسیون خطی، به دو مفهوم مهم، نگاهی می‌اندازیم.

- تابع هزینه به ما کمک می‌کند تا بهترین مقادیر ممکن را برای α و β پیدا کنیم که بهترین خط را برای نقاط داده برازش می‌کند. از آنجایی که ما بهترین مقادیر را برای α و β می‌خواهیم، این مساله جستجو را به یک مساله کمینه‌سازی تبدیل می‌کنیم که در آن می‌خواهیم خطای بین مقدار پیش‌بینی شده و مقدار واقعی را به حداقل برسانیم.

$$\text{minimize} \frac{1}{n} \sum_{i=1}^n (\text{pred}_i - y_i)^2 \quad (۲.۱)$$

$$J = \frac{1}{n} \sum_{i=1}^n (\text{pred}_i - y_i)^2 \quad (۳.۱)$$

تابع بالا را برای کمینه‌سازی انتخاب می‌کنیم. تفاوت بین مقادیر پیش‌بینی شده و واقعی، خطای را اندازه‌گیری می‌کند. خطای را مربع می‌کنیم و روی تمام نقاط داده جمع می‌کنیم و آن مقدار را بر تعداد کل نقاط داده تقسیم می‌کنیم. این میانگین، مربعات خطای را در تمام نقاط داده ارائه می‌دهد. بنابراین، این تابع هزینه به عنوان تابع میانگین مربعات خطای^{۲۰} نیز شناخته می‌شود. حال با استفاده از این تابع می‌خواهیم مقادیر α و β را طوری تغییر دهیم که مقدار مربعات خطای در مینیمم قرار گیرد [۱۰].

^{۲۰}MSE

- مفهوم مهم دیگر برای درک رگرسیون خطی، کاهش گرادیان است. کاهش گرادیان روشی برای به روز رسانی α و β برای کاهش تابع هزینه است. ایده این است که ما با مقادیری برای α و β شروع می کنیم و سپس این مقادیر را به طور مکرر تغییر می دهیم تا هزینه را کاهش دهیم. کاهش گرادیان به ما در چگونگی تغییر مقادیر کمک می کند [۱۱].
- برای ترسیم قیاس، گودالی را به شکل U تصور می کنیم که در بالاترین نقطه گودال ایستاده ایم و هدف ما رسیدن به انتهای گودال است. مساله ای وجود دارد که ما فقط می توانیم تعداد محدودی قدم برداریم تا به پایین گودال برسیم. اگر تصمیم بگیریم هر بار یک قدم برداریم، در نهایت به انتهای گودال خواهیم رسید، اما این کار زمان زیادی می برد. اگر تصمیم بگیریم هر بار قدم های بلندتری برداریم، زودتر به آن می رسیم، اما این احتمال وجود دارد که از پایین گودال عبور کنیم و دقیقاً در انتهای گودال توقف نکنیم. در الگوریتم کاهش گرادیان، تعداد قدم هایی که برمی داریم، نرخ یادگیری است. این نرخ تصمیم می گیرد که الگوریتم با چه سرعتی به حداقل همگرا شود.
- برای به روز رسانی مقادیر α و β ، گرادیان تابع هزینه را محاسبه می کنیم. برای یافتن این گرادیان ها، مشتقات جزئی را با توجه به α و β محاسبه می کنیم [۱۲].

$$\alpha := \alpha - l \frac{1}{n} \sum_{i=1}^n (pred_i - y_i) \quad (4.1)$$

$$\beta := \beta - l \frac{1}{n} \sum_{i=1}^n (pred_i - y_i).x_i \quad (5.1)$$

مشتقات جزئی گرادیان هستند و برای به روز رسانی مقادیر α و β استفاده می شوند. نرخ یادگیری است که یک ابرپارامتر است که باید آن را مشخص کنیم. نرخ یادگیری کمتر می تواند ما را به حداقل نزدیک تر کند، اما زمان بیشتری برای رسیدن به حداقل طول می کشد، نرخ یادگیری بزرگ تر زودتر همگرا می شود، اما این احتمال وجود دارد که از حداقل ها عبور کنیم.

رگرسیون لجستیک

رگرسیون لجستیک یک الگوریتم یادگیری ماشین است که برای مسائل طبقه‌بندی استفاده می‌شود، یک الگوریتم تحلیل پیش‌بینی و مبتنی بر مفهوم احتمال است.

می‌توانیم رگرسیون لجستیک را یک مدل رگرسیون خطی بنامیم، اما رگرسیون لجستیک از یک تابع هزینه پیچیده‌تر استفاده می‌کند، این تابع هزینه را می‌توان به عنوان «تابع سیگموید»^{۲۱} یا به جای تابع خطی به عنوان «تابع لجستیک» تعریف کرد.

فرضیه رگرسیون لجستیک این است که تابع هزینه را بین 0 و 1 محدود کند. بنابراین تابع خطی نمی‌تواند مورد استفاده قرار گیرند، زیرا این تابع می‌تواند مقداری بیشتر از 1 یا کمتر از 0 داشته باشد که طبق فرضیه رگرسیون لجستیک امکان پذیر نیست.

$$0 \leq h_{\theta}(x) \leq 1 \quad (6.1)$$

برای نگاشت مقادیر پیش‌بینی شده به احتمالات، از تابع سیگموید استفاده می‌کنیم. تابع هر مقدار واقعی را به مقدار دیگری بین 0 و 1 نگاشت می‌کند. در یادگیری ماشین، از تابع سیگموید برای ترسیم پیش‌بینی‌ها به احتمالات استفاده می‌کنیم.

همانطور که در مورد تابع هزینه (J_{θ}) در رگرسیون خطی بیان شد، این تابع نشان دهنده هدف بهینه‌سازی است، یعنی یک تابع هزینه ایجاد می‌کنیم و آن را به حداقل می‌رسانیم تا بتوانیم یک مدل دقیق با حداقل خطا ایجاد کنیم.

$$J(\theta) = \frac{1}{2} \sum_{i=1}^m (h_{\theta}(x^i) - y^i)^2 \quad (7.1)$$

اگر بخواهیم از تابع هزینه رگرسیون خطی در رگرسیون لجستیک استفاده کنیم، فایده‌ای نخواهد داشت زیرا در نهایت یک تابع غیرمحدب با کمینه‌های موضعی^{۲۲} متعدد است، که در آن به حداقل رساندن مقدار تابع هزینه و پیدا کردن کمینه کلی^{۲۳}، بسیار دشوار است.

²¹Sigmoid function

²²Local Minimums

²³Global Minimum

تابع هزینه در رگرسیون لجستیک به این صورت تعریف می‌شود:

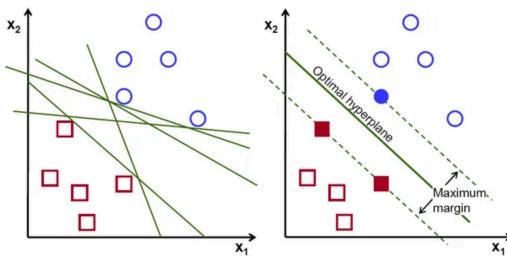
$$Cost(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)), & \text{اگر } y = 1, \\ -\log(1 - h_\theta(x)), & \text{اگر } y = 0. \end{cases} \quad (8.1)$$

تابع فوق را می‌توان در فرم بسته نیز نوشت:

$$J(\theta) = \frac{-1}{m} \sum [y^i \log(h_\theta(x(i))) + (1 - y^i) \log(1 - h_\theta(x(i)))] \quad (9.1)$$

ماشین بردار پشتیبان

هدف از الگوریتم ماشین بردار پشتیبان، یافتن یک ابر صفحه^{۲۴} در یک فضای N بعدی است که به طور مشخص نقاط داده را طبقه‌بندی می‌کند. برای جدا کردن دو دسته از نقاط داده، ابر صفحه‌های زیادی وجود دارد که می‌توان انتخاب کرد. هدف ما این است که صفحه‌ای را پیدا کنیم که حداکثر حاشیه، یعنی حداکثر فاصله بین نقاط داده دو کلاس را داشته باشد. حداکثر کردن فاصله حاشیه، تقویتی بوجود می‌آورد که می‌توان نقاط داده آینده را با اطمینان بیشتری طبقه‌بندی کرد [۱۴].

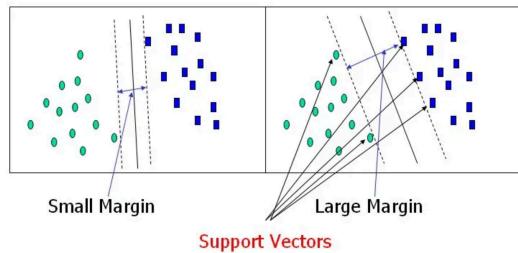


شکل ۲۰.۱: ابرصفحه‌های ممکن [۱۵]

²⁴Hyperplane

²⁵Margin Distance

ابرصفحه‌ها، مرزهای تصمیم‌گیری هستند که به طبقه‌بندی نقاط داده کمک می‌کنند. نقاط داده‌ای که در هر دو طرف ابرصفحه قرار می‌گیرند را می‌توان به کلاس‌های مختلف نسبت داد. همچنین، ابعاد ابرصفحه به تعداد ویژگی‌ها بستگی دارد. اگر تعداد ویژگی‌های ورودی ۲ باشد، آنگاه ابرصفحه فقط یک خط است. اگر تعداد ویژگی‌های ورودی ۳ باشد، ابرصفحه به یک صفحه دو بعدی تبدیل می‌شود. تصور زمانی که تعداد ویژگی‌ها از ۳ بیشتر شود دشوار می‌شود.



شکل ۳.۱: بردار پشتیبان [۲۹]

بردارهای پشتیبان نقاط داده‌ای هستند که به ابرصفحه نزدیکتر می‌باشند و بر موقعیت و جهت ابرصفحه تأثیر می‌گذارند. با استفاده از این بردارهای پشتیبان، حاشیه طبقه بندی کننده را به حد اکثر می‌رسانیم. حذف بردارهای پشتیبان موقعیت ابرصفحه را تغییر می‌دهد. این‌ها نکاتی هستند که به ما در ساخت ماشین بردار پشتیبان کمک می‌کنند.

در رگرسیون لجستیک، خروجی تابع خطی را می‌گیریم و با استفاده ازتابع سیگموید مقدار را در بازه ۰ تا ۱ قرار می‌دهیم. اگر مقدار به دست آمده بزرگتر از مقدار آستانه $\frac{1}{\theta}$ باشد، به آن برچسب ۱ و در غیر این صورت برچسب ۰ به آن اختصاص می‌دهیم. در ماشین بردار پشتیبان، خروجی تابع خطی را می‌گیریم و اگر آن خروجی بزرگتر از ۱ باشد، شناسایی می‌کنیم. آن را با یک کلاس و اگر خروجی منفی ۱ باشد، با کلاس دیگری شناسایی می‌کنیم. حال با توجه به مطالب ذکر شده در بالا، تابع ضرر و گرادیان ماشین بردار پشتیبان به صورت زیر بدست می‌آیند [۱۶]:

$$\min_w \lambda \|w\|^2 + \sum_{i=1}^n (1 - y_i \langle x_i, w \rangle)_+ \quad (10.1)$$

$$w := w + l \cdot (y_i \cdot x_i - 2\lambda w) \quad (11.1)$$

الگوریتم جنگل تصادفی

الگوریتم جنگل تصادفی^{۲۶} یکی از الگوریتم‌های یادگیری ماشین است که به تکنیک یادگیری ناظارت شده تعلق دارد. این الگوریتم برای مسائل طبقه‌بندی و رگرسیون در یادگیری ماشین مورد استفاده قرار می‌گیرد. اصولاً این الگوریتم بر مبنای مفهوم یادگیری گروهی عمل می‌کند، به این معنا که چندین طبقه‌بندی کننده را با هم ترکیب می‌کند تا یک مسئله پیچیده را حل کرده و عملکرد مدل را بهبود بخشد.

در الگوریتم جنگل تصادفی، چندین درخت تصمیم در زیرمجموعه‌های مختلف از مجموعه داده قرار می‌گیرند و به صورت میانگین، خروجی طبقه‌بندی را بهبود می‌بخشد. به عبارت دیگر، این الگوریتم به جای اعتماد به یک درخت تصمیم، از پیش‌بینی‌های هر درخت و بر اساس اکثربیت آرا، پیش‌بینی نهایی را انجام می‌دهد. تعداد بیشتر درختان در جنگل، دقت پیش‌بینی را افزایش می‌دهد و از بروز مشکل پیش‌پردازش^{۲۷} جلوگیری می‌کند [۱۷].

این الگوریتم در دو فاز قابل بررسی است. فاز اول ایجاد جنگل تصادفی با ترکیب N درخت تصمیم و فاز دوم پیش‌بینی برای هر درخت ایجاد شده در فاز اول است. کارکرد این الگوریتم را می‌توان در پنج مرحله توصیح داد: مرحله ۱: نقاط تصادفی K داده را از مجموعه آموزش انتخاب می‌کنیم. مرحله ۲: درختان تصمیم مرتبط با نقاط داده انتخاب شده (زیرمجموعه‌ها) را می‌سازیم.

مرحله ۳: تعداد N را برای درختان تصمیمی که می‌خواهیم بسازیم انتخاب می‌کنیم.

مرحله ۴: مرحله ۱ و ۲ را تکرار می‌کنیم.

مرحله ۵: برای نقاط داده جدید، پیش‌بینی‌های هر درخت تصمیم‌گیری را پیدا می‌کنیم و نقاط داده جدید را به دسته‌ای که آرای اکثربیت را کسب می‌کند، اختصاص می‌دهیم.

دسته‌بند بیز

دسته‌بند بیز^{۲۸} مدل یادگیری ماشین بر پایه احتمال است که برای طبقه‌بندی استفاده می‌شود. اصل طبقه‌بندی کننده بر اساس قضیه بیز است.

قضیه بیز: با استفاده از قضیه بیز، با توجه به اینکه B رخ داده است، می‌توانیم احتمال وقوع A را پیدا کنیم. در اینجا، B شرط^{۲۹} و A فرضیه^{۳۰} است. فرضی که در اینجا مطرح می‌شود این است که پیش‌بینی کننده‌ها ویژگی‌ها مستقل هستند. یعنی وجود یک ویژگی خاص بر دیگری تأثیر نمی‌گذارد. از این رو به آن ساده^{۳۱} گفته می‌شود [۱۸].

²⁶Random Forest Algorithm

²⁷Overfitting

²⁸Naive Bayes Classifier

²⁹Evidence

³⁰Hypothesis

³¹Naive

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)} \quad (12.1)$$

پس اگر A را پارامتر هدف و B را بردار ویژگی در نظر بگیریم، داریم:

$$P(y|x_1, \dots, x_n) = \frac{P(x_1|y) \cdots P(x_n|y)P(y)}{P(x_1) \cdots p(x_n)} \quad (13.1)$$

اکنون، می‌توانیم با مشاهده مجموعه داده‌ها، مقادیر هر یک را به دست آوریم و آن‌ها را در معادله جایگزین کنیم. برای همه ورودی‌های مجموعه داده، مخرج تغییر نمی‌کند. بنابراین می‌توان مخرج را حذف کرد و تناسبی را معرفی کرد.

$$P(y|x_1, \dots, x_n) \propto P(y) \prod_{i=1}^n P(x_i|y) \quad (14.1)$$

$$y = argmax_y P(y) \prod_{i=1}^n P(x_i|y) \quad (15.1)$$

۲۰.۱ شبکه‌های عصبی

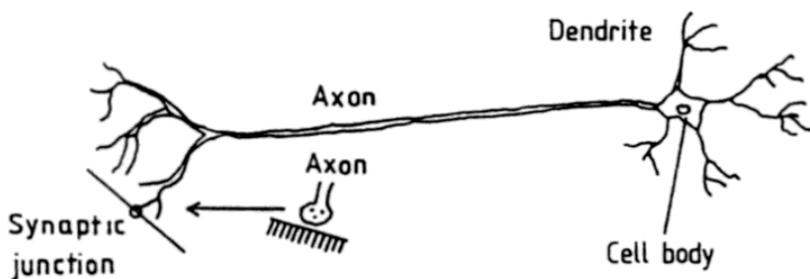
شبکه‌های عصبی مصنوعی^{۳۲} در حقیقت توسعه‌ای از مدل‌های شبکه عصبی هستند که در دهه ۱۹۴۰ توسط دانشمندانی مانند وارن مک‌کالوک و والتر پیتس و برای حل مسائلی مانند تشخیص الگو و پردازش تصویر استفاده می‌شدند.

در دهه ۱۹۵۰، دانشمندانی مانند جان مک‌کارتی و فرانک روزنبلات، شروع به توسعه شبکه‌های عصبی بر اساس ایده‌های مدل‌های شبکه عصبی کردند. اما در دهه ۱۹۶۰، با معرفی الگوریتم پساننتشار خطای^{۳۳} توسط دانشمندانی مانند دیوید رملهارت و پائول ورز، شبکه‌های عصبی به مرحله بعدی توسعه و استفاده خود رسیدند.

در دهه‌های بعدی، با پیشرفت تکنولوژی و افزایش توان محاسباتی، شبکه‌های عصبی به عنوان یکی از قدرتمندترین روش‌های یادگیری ماشین شناخته شدند و در بسیاری از زمینه‌های کاربردی مانند تشخیص تصویر، ترجمه مашینی، تشخیص گفتار و غیره، استفاده شدند.

بنابراین، شبکه‌های عصبی مصنوعی می‌توانند به عنوان یک نتیجه از پیشرفت در تحقیقات مدل‌های شبکه عصبی در دهه‌های گذشته در نظر گرفته شوند.

هر شبکه عصبی مصنوعی ساخته شده از نورون‌هایی مصنوعی (مشابه نورون‌ها در سیستم عصبی انسان) است که از طریق اتصالاتی (مشابه آکسون‌ها^{۳۴} و دندریت‌ها^{۳۵}) با نورون‌های دیگر در ارتباط هستند. مانند سیناپس‌ها^{۳۶} در سیستم عصبی انسان، هر ارتباط بین نورون‌ها سیگنال‌هایی را منتقل می‌کند که قدرت آن‌ها می‌تواند توسط وزنی که به طور مداوم در طول فرآیند یادگیری تنظیم می‌شود، افزایش یا کاهش یابد [۱۹].



شکل ۴.۱: نورون طبیعی [۳۳]

³²Artificial Neural Network

³³Backpropagation

³⁴Axon

³⁵Dendrite

³⁶Synapse

۱۰۲۱ ساختار شبکه های عصبی

شبکه های عصبی مصنوعی می توانند به اشکال مختلفی ساخته شوند، اما بیشتر شامل چندین لایه از نورون ها هستند که به طور خاص به هم متصل شده اند. در ادامه، ساختار یک شبکه عصبی معمولی را توضیح می دهیم [۲۰] :

- **لایه ورودی^{۳۷}** : این لایه شامل ورودی های شبکه است که به شکل عددی وارد شبکه می شوند. به طور معمول، هر نورون، یکی از ویژگی های مهم ورودی ها مانند رنگ، اندازه یا شکل را نشان می دهد.
- **لایه مخفی^{۳۸}** : این لایه بین لایه ورودی و خروجی قرار دارد و شامل نورون هایی است که به طور مستقیم با لایه ورودی و خروجی ارتباط برقرار می کنند. هر نورون در این لایه، با ورودی های خود که از لایه قبلی وارد می شوند، محاسباتی را انجام می دهد و خروجی خود را به نورون های لایه بعدی می فرستد.
- **لایه خروجی^{۳۹}** : این لایه شامل خروجی های شبکه است که به عنوان نتیجه شبکه به کاربر ارائه می شود. هر خروجی معمولاً یکی از کلاس های ممکن برای مسئله را نشان می دهد. برای مثال، در یک مسئله تشخیص تصویر، خروجی می تواند نشان دهنده یک شئ خاص در تصویر باشد.
- **وزن ها^{۴۰}** : وزن ها نشان دهنده قدرت ارتباط بین نورون های مختلف در لایه های مختلف شبکه هستند. این وزن ها به طور مداوم در طول فرآیند یادگیری تنظیم می شوند تا بهترین نتیجه برای ورودی های جدید به دست آید.
- **تابع فعال سازی^{۴۱}** : این تابع برای محاسبه خروجی نورون ها استفاده می شود. این تابع به عنوان تابعی غیرخطی عمل می کند تا شبکه بتواند الگوها و ویژگی های پیچیده تر را بیاموزد.
- **الگوریتم های یادگیری^{۴۲}** : به منظور آموزش شبکه های عصبی، الگوریتم های یادگیری مانند الگوریتم پسانشان خطا استفاده می شوند. این الگوریتم با استفاده از داده های آموزشی، وزن های شبکه را به طور مداوم تنظیم می کند تا بهترین نتیجه برای ورودی های جدید حاصل شود.

³⁷Input Layer

³⁸Hidden Layer

³⁹Output Layer

⁴⁰Weights

⁴¹Activation Function

⁴²Learning Algorithms

- اریبی‌ها^{۴۳} : اریبی‌ها به طور مشابه با وزن‌ها، پارامترهایی هستند که توسط شبکه به طور مداوم تنظیم می‌شوند. این پارامترها به عنوان مقداری ثابت در محاسبات هر نرون استفاده می‌شوند و به شبکه کمک می‌کنند تا بهترین نتیجه را به دست آورد.

۳۰۱ شبکه‌های عصبی از پیش آموزش داده شده

شبکه‌های عصبی از پیش آموزش داده شده^{۴۴} ، یکی از متدهای مهم در حوزه هوش مصنوعی هستند که در سال‌های اخیر بسیار مورد توجه قرار گرفته‌اند. این شبکه‌ها در ابتدا بر روی یک مجموعه داده آموزش داده می‌شوند و پس از آموزش، می‌توانند برای حل مسائل مشابه در زمینه‌های مختلف مورد استفاده قرار گیرند.

یکی از مزایای استفاده از شبکه‌های عصبی از پیش آموزش داده شده، کاهش زمان و هزینه آموزش است. آموزش یک شبکه عصبی بر روی یک مجموعه داده بزرگ و پیچیده، ممکن است مدت زمان زیادی را به خود اختصاص دهد و هزینه‌های بالایی را برای سازمان یا فرد آموزش دهنده به همراه داشته باشد. با استفاده از شبکه‌های عصبی از پیش آموزش داده شده، می‌توان زمان و هزینه آموزش را به شدت کاهش داد و در حل مسئله بهتر عمل کرد.

علاوه بر این، شبکه‌های عصبی از پیش آموزش داده شده، دارای قابلیت انتقال پذیری هستند. این به این معنی است که می‌توان از یک شبکه عصبی که برای حل یک مسئله خاص آموزش دیده است، برای حل مسائل مشابه در زمینه‌های مختلف استفاده کرد. به عنوان مثال، شبکه‌هایی که برای تشخیص تصویر آموزش دیده‌اند، می‌توانند برای حل مسائل مشابهی مانند تشخیص بخش‌های صورت، تشخیص اشیاء در تصاویر پزشکی و غیره استفاده شوند. به دلیل ویژگی‌های مشترکی که شبکه‌های عصبی از پیش آموزش داده شده با مسائل جدید دارند، می‌تواند عملکرد بهتری نسبت به شبکه‌های جدید داشته باشند.

برت^{۴۵} یکی از مدل‌های شبکه‌های عصبی از پیش آموزش داده است که در حوزه پردازش زبان طبیعی^{۴۶} استفاده می‌شود. این مدل توسط گوگل در سال ۲۰۱۸ معرفی شد و به سرعت محبوبیت بسیاری در میان پژوهشگران و صنعت‌گران به دست آورد.

⁴³Biases

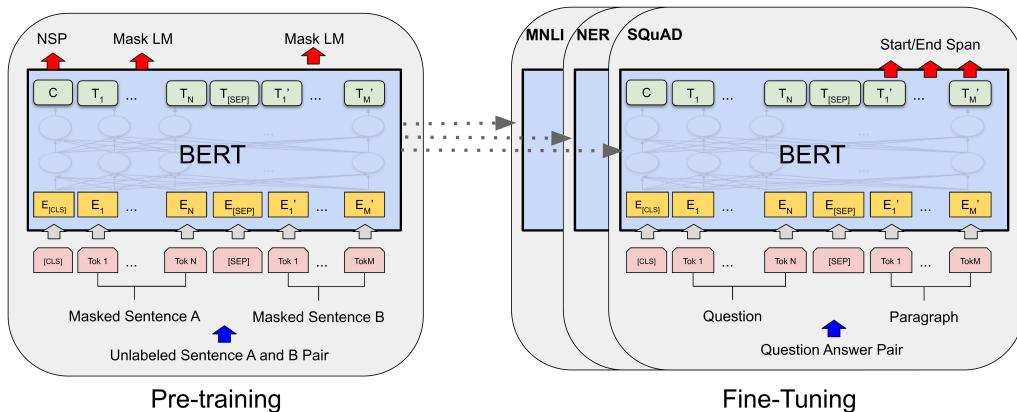
⁴⁴Pre-trained Neural Networks

⁴⁵BERT

⁴⁶Natural Language Processing

برت در هسته‌ی خود دارای یک مدل زبانی مبتنی بر تبدیل‌کننده^{۴۷} با تعداد زیادی از کدگذارها^{۴۸} و لایه‌های self-attention است. برتر از پیش روی دو مساله^{۴۹} آموزش داده شده است. اولین مساله مدل کردن زبانی^{۵۰} است که ۱۵ درصد از توکن‌ها را به صورت ماسک شده قرار داده بوده‌اند که برتر آموزش داده شده است تا بر اساس متن آن‌ها را پیش‌بینی نماید. مساله دوم هم مرتبط با پیش‌بینی عبارت بعدی^{۵۱} است. در این مساله برتر آموزش داده شده است تا اگر یه جمله بعدی به صورت احتمالاتی داده شده باشد و یا اصلاً جمله‌ی بعدی را نداشته باشیم چطور بتوانیم از روی یک جمله، جمله‌ی بعدی را پیش‌بینی نماییم. هردوی این مسئله‌ها، مسائل پایه‌ای و رایج در پردازش زبان‌های طبیعی هستند و در مسائل زیادی کاربرد دارند.

به عنوان نتیجه می‌توان گفت که برتر می‌تواند به صورت محتوایی کلمات را در جملات و متن‌های خاص یاد بگیرد. مزیت این مدل این است که پس از اینکه به صورت از پیش آموزش داده شده روی مساله‌های فوق آماده شد، می‌توان آن را روی هر مساله دلخواهی از پردازش زبان‌های طبیعی تنظیم کرد و امروزه با مجموعه داده‌های به نسبت کوچک روی مساله‌های دلخواه پردازش زبان‌های طبیعی هم به نتایج خوب با دقت‌های بالایی رسید و به صورت محاسباتی هم هزینه‌ی کمتری دارد. چراکه در گذشته برتر روی داده‌های زیادی آموزش داده شده است.



[۲۱]: ساختار برتر [۵.۱]

⁴⁷Transformer

⁴⁸Encoders

⁴⁹Task

⁵⁰Language Modeling

⁵¹Next Sequence Prediction

⁵²Fine-tune

٤٠١ تنظیم پارامتر

در مدل‌های یادگیری ماشین، پارامترها مقادیری هستند که توسط کاربر یا الگوریتم به مدل داده می‌شوند و می‌توانند بر روی عملکرد مدل تأثیر بگذارند. به عنوان مثال، در مدل‌های خطی مانند رگرسیون خطی و ماشین بردار پشتیبانی، پارامترهایی مانند ضرایب وزنی وجود دارند که تعیین می‌کنند که مدل چقدر قوی و عمومی خواهد بود. در مدل‌های دیگر مانند شبکه‌های عصبی، پارامترهایی مانند تعداد لایه‌ها، تعداد واحدهای هر لایه، نرخ یادگیری و نرخ کاهش گرادیان نیز وجود دارند که تعیین می‌کنند که مدل چقدر عمیق و پیچیده خواهد بود.

پیدا کردن پارامترهای بهینه می‌تواند عملکرد مدل را بهبود بخشد و تعیین پارامترهای نادرست می‌تواند منجر به بیش‌برازش^{۵۳} یا کم‌برازش^{۵۴} مدل شود که هر دو می‌توانند منجر به پایین آمدن دقت مدل بر روی داده‌های تست شوند [۲۲].

برای تعیین پارامترهای مناسب در مدل‌های یادگیری ماشین، می‌توان از روش‌هایی مانند جستجوی خطی^{۵۵} یا جستجوی تصادفی^{۵۶} استفاده کرد. در هر یک از این روش‌ها، پارامترهای مختلفی برای مدل تعیین می‌شوند و برای هر یک از آن‌ها، مدل با پارامترهای داده شده روی داده‌های آموزشی، آموزش داده می‌شود و عملکرد آن روی داده‌های اعتبارسنجی بررسی می‌شود. سپس پارامترهایی که عملکرد بهتری را دارند، برای آزمون نهایی بر روی داده‌های تست استفاده می‌شوند.

٤٠٢ جستجوی خطی

در جستجوی خطی، یک مجموعه مشخص از مقادیر برای هر ابپارامتر انتخاب می‌شود و برای هر ترکیب ممکن از مقادیر ابپارامترها، یک مدل آموزش داده می‌شود و عملکرد آن با استفاده از معیارهای ارزیابی مانند دقت^{۵۶}، فراخوانی^{۵۷} و دقت پیش‌بینی^{۵۸}، ارزیابی می‌شود. سپس، مدلی که بهترین عملکرد را داشته باشد، به عنوان مدل نهایی انتخاب می‌شود. استفاده از جستجوی خطی می‌تواند به عنوان یک روش سریع و موثر برای پیدا کردن بهترین مقادیر ابپارامترها در الگوریتم‌های یادگیری ماشین مورد استفاده قرار گیرد [۲۳].

^{۵۳}Underfitting

^{۵۴}Grid Search

^{۵۵}Random Search

^{۵۶}Accuracy

^{۵۷}Recall

^{۵۸}Precision

۲۰۴۱ جستجوی تصادفی

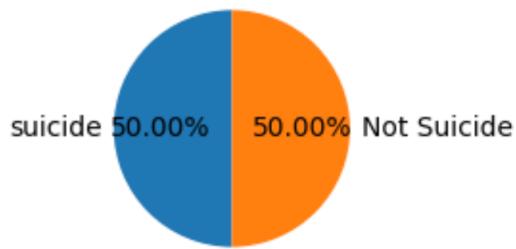
جستجوی تصادفی نیز همانند جستجوی خطی، یک روش برای پیدا کردن بهترین مقادیر برای ابرپارامترهای یک الگوریتم یادگیری ماشین است. در این روش، به جای تعیین یک مجموعه مقادیر ثابت برای هر ابرپارامتر، از یک توزیع احتمالی برای هر پارامتر استفاده می‌شود. سپس، تعدادی ترکیب تصادفی از مقادیر ابرپارامترها انتخاب می‌شود و برای هر یک، یک مدل با استفاده از آن پارامترها آموزش داده می‌شود و عملکرد آن با استفاده از معیارهای ارزیابی، ارزیابی می‌شود. در نهایت، مدلی که بهترین عملکرد را داشته باشد، به عنوان مدل نهایی انتخاب می‌شود.

فصل ۲

دادگان

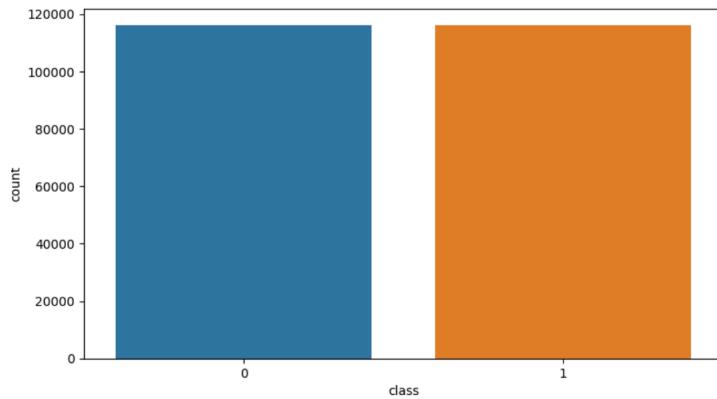
۱۰۲ توصیف دادگان

این مجموعه داده دارای متونی است که توسط افراد در شبکه‌های اجتماعی، ایمیل‌های ارسال شده و پیام‌های متنی ارسال شده به دیگران نوشته شده است. این داده‌ها از سایت ردیت توسط API Pushshift جمع آوری شده‌اند. مجموعه داده شامل داده‌های سال‌های ۲۰۰۷ تا ۲۰۲۱ است. این داده‌ها می‌توانند برای تحلیل و پیش‌بینی رفتارهای آینده افراد مورد استفاده قرار گیرند و در کاهش نرخ خودکشی و بهبود سلامت روانی جامعه موثر باشند.



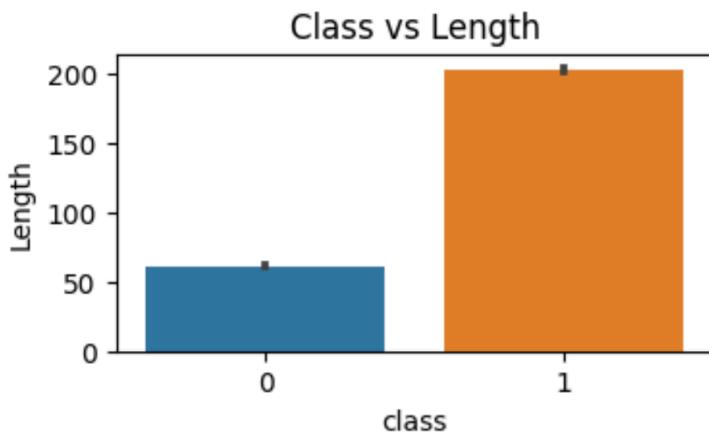
شكل ۱.۲ : توزیع دادگان

در این مجموعه داده، داده‌ها در دو کلاس suicidal و non-suicidal طبقه‌بندی شده‌اند که به ترتیب متونی را دارا می‌باشند که شامل افرادی هستند که به نحوی درگیر خودکشی شده‌اند و افرادی که خودکشی نکرده‌اند. تعداد داده‌ها در هر کلاس با هم برابر است و این باعث می‌شود که داده‌ها همگن باشند.



شکل ۲.۲: تعداد داده‌های هر کلاس

این مجموعه داده شامل انواع مختلف متون است که در صورت استفاده مناسب می‌توانند به تحلیل‌های مفیدی درباره رفتارهای آینده افراد کمک کنند. همچنین طول متون متغیر است و شامل طول‌های مختلف از ۱ تا ۱۵۶۳۲ کاراکتر می‌باشد که اکثریت آن در بازه ۲۶ تا ۱۵۴ کاراکتر هستند.



شکل ۳.۲: طول داده‌های هر کلاس

۲۰۲ پیش‌پردازش دادگان

تبديل تمام حروف متن به حروف کوچک: این مرحله به منظور جلوگیری از در نظر گرفتن دو کلمه متفاوت به دلیل تفاوت حروف بزرگ و کوچک انجام شده است. برای مثال، کلمه Apple و apple به دلیل تفاوت حروف بزرگ و کوچک، به عنوان دو کلمه متفاوت در نظر گرفته خواهند شد. با تبدیل تمام حروف به حروف کوچک، این مشکل برطرف می‌شود و هر دو کلمه به عنوان یک کلمه در نظر گرفته می‌شوند.

حذف فاصله ابتدایی و انتهایی متن: در برخی موارد، ممکن است که دریافت داده‌ها از منابع مختلف و با کیفیت‌های مختلف انجام شود و این باعث می‌شود که در ابتدایا یا انتهایی متن، فاصله‌های اضافی وجود داشته باشد. این فاصله‌ها در پردازش متن ممکن است باعث ایجاد مشکلاتی شوند. به عنوان مثال، در برنامه‌های مربوط به پردازش زبان طبیعی، فاصله اضافی در ابتدایا و انتهایی متن ممکن است باعث ایجاد مشکلات در تشخیص یا جستجوی الگوهای شود.

حذف تگ‌های HTML: در این مرحله، تمام تگ‌های HTML از متن حذف می‌شوند. حذف کد HTML و فرمت‌بندی از متن می‌تواند به کاهش حجم متن و بهبود قابلیت خواندن آن کمک کند.

حذف همه علائم نگارشی از متن: حذف علائم نگارشی می‌تواند بهبود قابلیت خواندن و پردازش متن را فراهم کند. علائم نگارشی ممکن است با کلمات دیگری به هم متصل شده و باعث ایجاد کلمات تکراری یا بدون معنی شوند. همچنین، در مواردی که متن باید برای پردازش به یک الگوریتم داده شود، حذف علائم نگارشی می‌تواند بهبود قابل توجهی در نتیجه نهایی ایجاد کند.

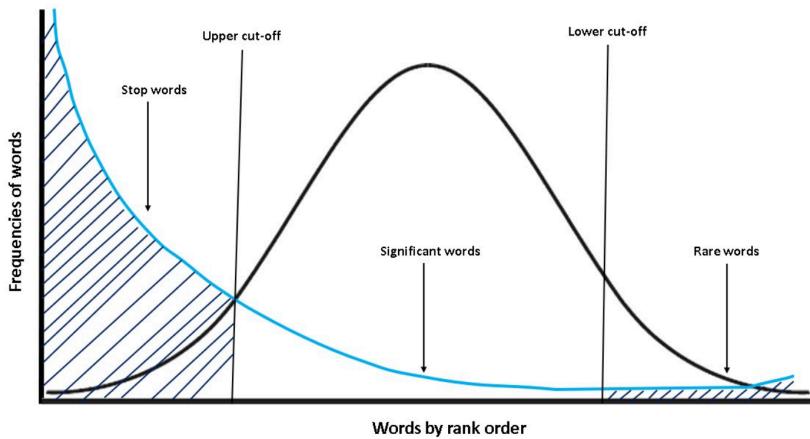
جایگزینی هر نوع فاصله‌ای که بیش از یکی است : در این مرحله، تمام فاصله‌های اضافی بین کلمات را حذف کرده و یک فاصله به عنوان جایگزین آنها قرار می‌دهیم. این کار به منظور بهبود کیفیت پردازش متن و حذف هرگونه فاصله اضافی بین کلمات است.

جایگزینی کردن هر کاراکتر غیر الفبایی : در این مرحله، تمام کاراکترهای غیر الفبایی مانند اعداد، نمادهای خاص، علائم ریاضی و... با یک فاصله جایگزین می‌شوند.

حذف ایست واژه: طبق قانون Zipf، اگر نمودار بین فراوانی توزیع کلمه و رتبه یک کلمه را در یک متن رسم کنیم، متوجه خواهیم شد که بسامد کلمات با رتبه یک کلمه نسبت معکوس دارد، یعنی رتبه ۱ به پر تکرار ترین کلمه، رتبه ۲ به دومین کلمه پرتکرار و ... داده می‌شود.

از نمودار ۴.۲ می‌توان نتیجه گرفت که به طور کلی سه نوع کلمه وجود دارد:

- کلمات توقف: این دسته کلمات درواقع کلمات پر تکرار مانند a ، an ، the ، is و ... هستند.
- کلمات مهم: آن دسته از کلمات هستند که دارای فراوانی متوسطی هستند و معنای واقعی را به متن اضافه می‌کنند. این کلمات تر از کلمات توقف هستند.
- کلمات نادر: کلماتی هستند که با فراوانی بسیار کمتری به وجود می‌آیند و اهمیت نسبتاً کمتری نسبت به کلمات مهم دارند. کلماتی که به ندرت وجود دارند ممکن است برای درک متن مفید باشند یا نباشند.



شکل ۴.۲: نمودار بسامد - رتبه [۳۰]

کلمات توقف به دو دلیل از متن حذف می‌شوند:

- اطلاعات مهمی به متن اضافه نمی‌کنند.
- فراوانی کلمات توقف در یک متن بسیار زیاد است. حذف این کلمات از متن، باعث کوچک شدن حجم داده‌ها و افزایش سرعت محاسبات می‌شود.

ریشه‌یابی کلمات: ^۱ فرآیند الگوریتمی یافتن لم یک کلمه بسته به معنی کلمه و محل قرارگیری آن در جمله است. برای مثال ریشه‌ی کلمه‌ی کلمه‌ی cats ، cat است و ریشه‌ی کلمه‌ی running ، run است. ریشه‌یابی کلمات مزایای زیادی دارد. از جمله آن‌ها عبارتند از:

- دقت تحلیل متن را بهبود می‌بخشد: برگرداندن کلمات، باعث می‌شود کلمات هم معنی راحت‌تر شناسایی شوند.
- اندازه داده را کاهش می‌دهد: از آنجایی که کلمات را به شکل پایه‌شان تبدیل می‌کند، باعث کاهش اندازه داده متن می‌شود
- نتایج جستجوی بهتر: از آنجایی که اشکال مختلف یک کلمه همه به یک فرم تبدیل می‌شوند، جست و جو راحت‌تر و با دقت بهتری انجام می‌شود.

تعیین نقش کلمات: برای ریشه‌یابی کلمات، معنی آن‌ها مورد نیاز است و در پیدا کردن معنی کلمات، نقش آن‌ها در جمله پارامتر مهمی است که تابع ریشه‌یاب از آن استفاده می‌کند. از آنجایی که ما برای ریشه‌یابی کلمات از تابع WordNetLemmatizer استفاده کردیم، نیازمند این هستیم که نقش کلمات را در فرم WordNet داشته باشیم. پس با استفاده از دستور wordnet.POS کار را انجام می‌دهیم [۲۴].

تبدیل کلمات به بردار: ^۲ راه‌های متفاوتی برای بردار سازی کلمات وجود دارد. در این مقاله از روش‌های زیر استفاده شده است:

- CountVectorizer: این تابع توسط کتابخانه scikit-learn پیاده سازی شده است. این تابع مراحل زیر را طی می‌کند:
 - لیستی از رشته‌ها دریافت می‌کند.
 - هر کدام از رشته‌ها را به توکن‌هایشان تقسیم‌بندی می‌کند.
 - یک لیست از توکن‌های یکتای کل ورودی‌ها پیدا می‌کند. عضوهای این لیست، اعضای بردار ویژگی هستند.
 - با استفاده از یک حلقه، روی تمام ورودی‌ها حرکت می‌کند و تعداد تکرار هر کدام از ویژگی‌ها در متن را پیدا می‌کند.
 - یک ماتریس اسپارس از ورودی می‌سازد و خروجی تابع همین ماتریس است.

¹Lemmatization

²Vectorization

	about	bird	heard	is	the	word	you
About the bird , the bird , bird bird bird	1	5	0	0	2	0	0
You heard about the bird	1	1	1	0	1	0	1
The bird is the word	0	1	0	1	2	1	0

شکل ۵.۲: تبدیل متن به بردارهای عددی

• TfIdfVectorizer: این تابع نیز توسط کتابخانه scikit-learn پیاده سازی شده و مراحل آن به شرح زیر است:

- رشته‌های ورودی را به توکن تقسیم می‌کند.
- تعداد تکرار هر توکن در هر رشته را پیدا می‌کند (TF)
- برای هر کلمه فرکانس معکوس (IDF) را محاسبه می‌کند. فرکانس معکوس معیاریست از میزان رایج یا نادر بودن کلمه در ورودی.
- سپس امتیاز TF-IDF را برای هر کلمه در هر ورودی محاسبه می‌کند. این امتیاز به شکل زیر محاسبه می‌شود.

$$TF(i, j) = \frac{\text{Term } i \text{ frequency in document } j}{\text{Total words in document } j} \quad (2.1)$$

$$IDF(i) = \log_2 \frac{\text{Total documents}}{\text{documents with term } j} \quad (2.2)$$

$$score = TF \cdot IDF \quad (2.3)$$

- با استفاده از امتیازهای بدست آمده، ماتریس TF-IDF را می‌سازد [۲۵].

در تابع پیاده‌سازی شده در scikit-learn پارامترهایی برای ایجاد تغییر در ماتریس خروجی وجود دارد. در این مقاله، از دو مورد از این پارامترها استفاده شده است. این پارامترها عبارتند از max features و ngram range. پارامتر اول مربوط به تعداد سطون‌های ماتریس خروجی است. در این پژوهه این پارامتر برابر با ۱۰۰۰ قرار داده شده به این معنا که ۱۰۰۰ کلمه‌ی پر تکرار مورد بررسی قرار می‌گیرند. پارامتر دوم مربوط به تعداد بخش‌های کلمه است. در این پژوهه این پارامتر را با دو مقدار استفاده کردیم.

- مقدار پیش فرض آن که فقط کلمات تک بخشی مانند machine و subfield را به عنوان ویژگی در نظر میگیرد.
- آن را برابر با (۱, ۲) قرار دادیم که باعث می شود کلمات دو بخشی مانند machine و blue bottle learning هم در نظر گرفته می شوند.
- تخصیص نمرات احساسات به متن با استفاده از مدل تحلیل احساسات VADER. مدل تحلیل احساسات VADER با توجه ویژگی های هر کلمه، به آن نمره ای بین یک (معنای مثبت) و منفی یک (معنای منفی) نسبت می دهد. این ویژگی ها عبارتند از [۲۶] :

- کلمات:

$$\text{Good} = +0.8, \text{Bad} = -0.8$$

- کلمات خاص نشان دهنده وضعیت ^۳ :

$$\text{amazing} = +1.0, \text{horrible} = -0.9$$

- اموجی ها ^۴ :

باتوجه به بار معنایی هر اموجی به آن یک عدد نسبت می دهد.

- ترکیب ها ^۵ :

$$\text{sort of} = -0.3$$

همچنین روابطی مبتنی بر احساسات ^۶ نیز دارد. مانند:

- با حروف بزرگ نوشتن:

$$\text{SO GOOD} = +0.9$$

- کلماتی که نشان دهنده میزان احساس است هستند:

$$\text{very good} = +0.9, \text{insanely good} = +1.0$$

- کلمات منفی یا مثبت کننده ^۷ :

$$\text{not good} = -0.8$$

با استفاده از این مدل، به هر سطر از داده چهار عدد نسبت داده می شود.

³Amplifiers

⁴Emojis

⁵Combos

⁶Sentiment-Based

⁷Negation Reversal

۳۰۲ تقسیم‌بندی دادگان

برای آماده‌سازی داده‌ها برای آموزش، الگوریتم‌های یادگیری ماشین، نخست باید داده‌ها را به بخش‌های آموزش، تست و اعتبارسنجی تقسیم کنیم. در بخش آموزش الگوریتم داده‌ها را می‌بیند و مدل را می‌آموزد، سپس با استفاده از داده‌های اعتبارسنجی پارامترهای مدل را تنظیم و در بخش تست مدل را بر روی داده‌ای که هیچ‌گاه ندیده برسی می‌کند.

۳۰ درصد از داده‌ها به عنوان داده تست از مجموعه داده کنار گذاشته شده و از داده‌های باقی‌مانده ۲۰ درصد به عنوان داده اعتبارسنجی و ۸۰ درصد به عنوان داده آموزشی در نظر گرفته شده‌اند. برای تقسیم‌بندی داده‌ها از تابع train test split از کتابخانه scikit-learn استفاده شده است. مهم‌تر از همه اینکه قبل از تقسیم، تمامی داده‌ها را بر می‌زنیم^۸. این کار جلوی اثرات احتمالی بعضی از سطرهای داده را می‌گیرد و عملکرد مدل را بهبود می‌بخشد.

⁸Shuffle

فصل ۳

آزمایش‌ها و نتیجه‌ها

در این بخش به بررسی نتیجه‌های بدست آمده توسط هر یک از الگوریتم‌های ذکر شده در فصل یک می‌پردازیم.

می‌دانیم که داده‌های پروژه را به چهار روش مختلف تبدیل به بردار کردیم (بخش ۲.۲). در هر الگوریتم ابتدا برای هر کدام از این چهار روش، با استفاده از جست و جوی خطی، بهترین پارامترها را پیدا کرده و سپس برای هر روش، مدل را با پارامترهای ایده‌آل آموزش می‌دهیم. سپس دقت مدل را با معیارهای متفاوتی حساب می‌کنیم. برای توضیح معیارها ابتدا نیاز به نمادگذاری داریم.

اعضای کلاس ۱ که درست برآورد شده‌اند: True Positive

اعضای کلاس صفر که درست برآورد شده‌اند: True Negative

اعضای کلاس ۱ که اشتباه برآورد شده‌اند: False Negative

اعضای کلاس صفر که اشتباه برآورد شده‌اند: False Positive

حال با استفاده از این مقادیر، ماتریس درهم‌ریختگی^۱ را رسم می‌کنیم. این ماتریس به شکل زیر بدست می‌آید:

$$\begin{bmatrix} \text{TruePositive} & \text{FalsePositive} \\ \text{FalseNegative} & \text{TrueNegative} \end{bmatrix}$$

سپس معیارهای دقت را بدست می‌آوریم. این معیارها عبارتند از [۲۷]:

- معیار صحت:

$$Precision = \frac{TP}{TP + FP} \quad (1.3)$$

¹Confusion Matrix

• معیار پوشش :

$$Recall = \frac{TP}{TP + FN} \quad (2.3)$$

• امتیاز F1 :

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \quad (3.3)$$

• دقت :

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \quad (4.3)$$

پس از بدست آوردن معیارها، آنها را در نموداری میله‌ای نمایش می‌دهیم. نمودار بعدی که رسم می‌کنیم، نمودار **مشخصه عملکرد**^۲ است [۲۸]. این نمودار ترسیم نسبت نرخ مثبت صحیح^۳ بر حسب نرخ مثبت کاذب^۴ است. بهترین عملکرد دسته‌بندی در این نمودار در نقطه‌ای با مختصات (۰، ۱) رخ خواهد داد. در این نقطه کمترین نرخ اشتباه و بیشترین نرخ صحیح را داریم. هر چه داده‌های دو کلاس به هم شبیه‌تر باشند و تمایز آن‌ها سخت‌تر باشد، منحنی این نمودار به خط $y = x$ نزدیک‌تر است.

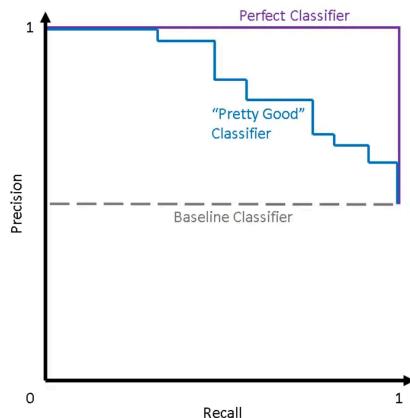
آخرین نموداری که رسم می‌کنیم، نمودار **صحت-پوشش**^۵ است. الگوریتم طبقه‌بندی خوب است که هم صحت و هم پوشش بالایی داشته باشد. پس با استفاده از منحنی این نمودار می‌توانیم میزان خوب بودن یک الگوریتم طبقه‌بندی را بفهمیم. هترین عملکرد دسته‌بندی در این نمودار در نقطه‌ای با مختصات (۱، ۱) رخ خواهد داد.

²Receiver Operating Characteristic

³True Positive Rate

⁴False Positive Rate

⁵Precision-Recall Curve



شکل ۱.۳: نمودار صحت-پوشش [۳۱]

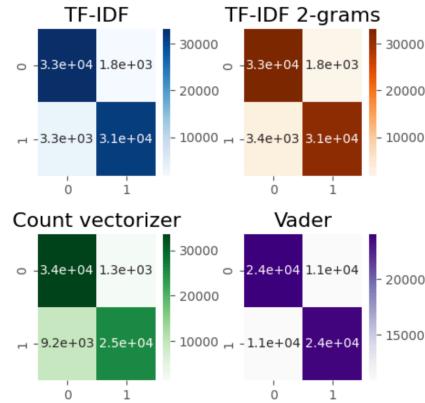
۱.۰۳ رگرسیون

۱.۰۱.۳ رگرسیون خطی ساده

نتایج آزمایش رگرسیون خطی در شکل ۳.۳ نمایش داده شده است. با توجه به ماتریس درهم ریختگی، مدل‌هایی که با داده‌های TF-IDF($n=(1,2)$) آموزش داده شده‌اند، شبیه به هم بوده و تعداد زیادی از داده‌ها را درست پیش‌بینی کرده‌اند.

مدلی که با داده‌های Count Vectorized آموزش دیده، در پیش‌بینی داده‌های کلاس صفر بهتر از دو مدل قبلی عمل کرده اما عملکرد کلی آن به خوبی آن‌ها نیست.

مدل آموزش داده شده با داده‌های Vader کم دقت‌ترین مدل بوده و نتایج خوبی نداشته است.

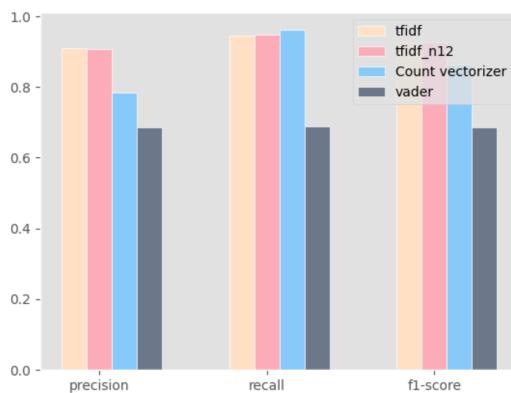


شکل ۲.۳ : ماتریس درهم ریختگی الگوریتم رگرسیون خطی

معیارهای دقت را در جدول زیر می‌توانید مشاهده کنید:

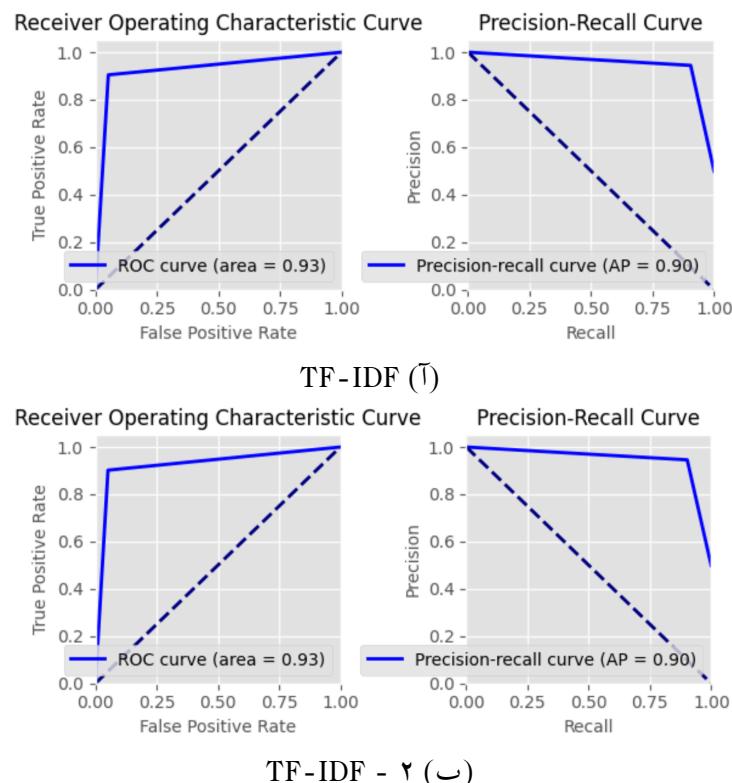
جدول ۱.۳ : جدول نتایج رگرسیون خطی

metric	TF-IDF	TF-IDF 2	CountVec	Vader
precision	0.908	0.906	0.784	0.685
recall	0.947	0.948	0.962	0.688
F1-score	0.927	0.927	0.864	0.687
accuracy	0.925	0.925	0.848	0.685

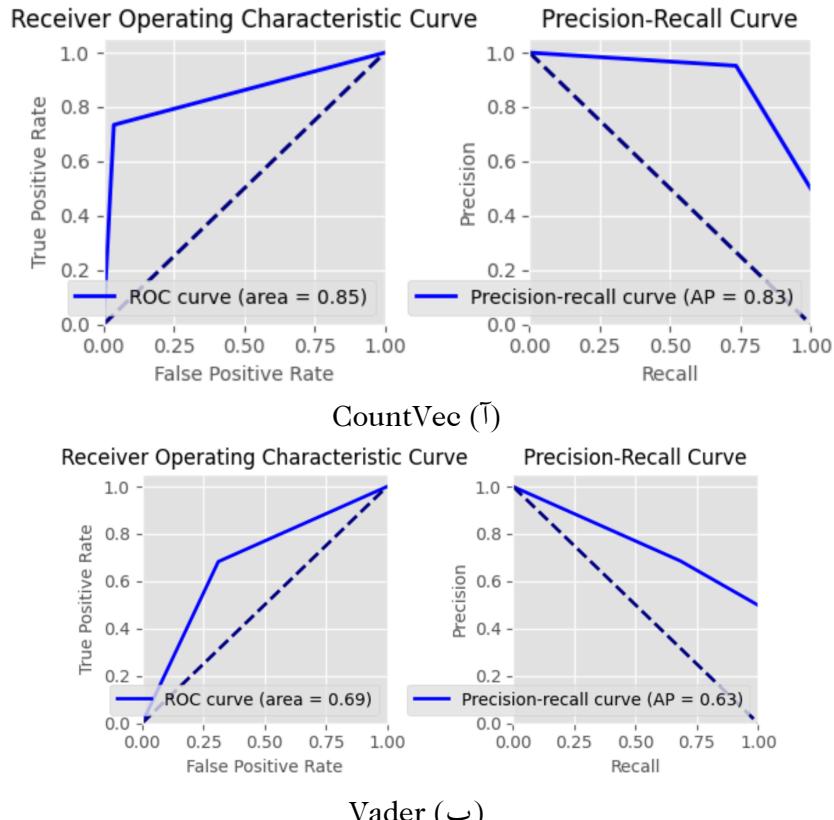


شکل ۳.۳ : نتایج الگوریتم رگرسیون خطی

حال به رسم نمودارهای مشخصه عملکرد و صحت-پوشش می‌پردازیم.



شکل ۴.۳: نمودارهای مشخصه عملکرد و صحت پوشش رگرسیون خطی



شکل ۵.۳: نمودارهای مشخصه عملکرد و صحت پوشش رگرسیون خطی

همانطور که از مساحت زیر نمودارها مشخص است، داده‌های TF-IDF با دارا بودن 0.93 و 0.90 مساحت زیر نمودارهای مشخصه عملکرد و صحت پوشش، بهترین نتایج و بعد از آن‌ها داده‌های Count vectorization با 0.85 و 0.83 در رتبه بعدی و درنهایت هم داده‌های Vader با 0.69 و 0.63 در جایگاه آخر قرار دارند.

۲۰۱۰۳ رگرسیون لجستیک

پارامترهایی که در این مدل تلاش به بهینه کردن آنها می‌کنیم عبارتند از:

- c : معکوس قدرت منظم سازی^۶ است. برای کنترل مبادله بین دستیابی به یک خطای آموزشی کم و یک خطای تست کم استفاده می‌شود. هرچه مقدار این پارامتر کمتر باشد، قدرت منظم سازی بیشتر و در نتیجه مدل ساده‌تر است. مدل ساده‌تر قدرت تعیین بیشتری دارد اما خطای بیشتری روی داده‌ی آموزشی دارد. همچنین زیاد بودن مقدار این پارامتر می‌تواند مدل پیچیده‌ای تولید کند که داده آموزشی را حفظ می‌کند.
- Penalty: مدل تابع منظم ساز را مشخص می‌کند. می‌تواند L1 یا L2 باشد.

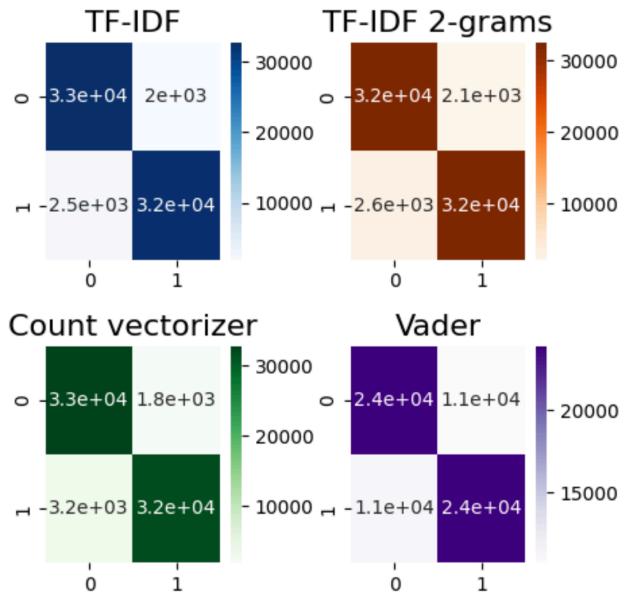
پارامترهای بدست آمده توسط جست و جوی خطی برای رگرسیون لجستیک به شرح زیر است:

جدول ۲.۳: پارامترهای رگرسیون لوجستیک

Method	c	penalty
TF-IDF	10	l2
TF-IDF($n=(1,2)$)	10	l2
CountVectorizer	1	l2
Vader(emotion)	0.1	l2

با استفاده از پارامترهای بدست آمده مدل را آموزش دادیم. ماتریس درهم ریختگی در شکل ۶.۳ رسم شده است.

⁶Regularization

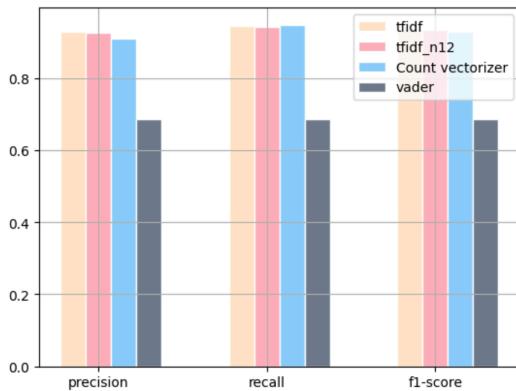


شکل ۶.۳: ماتریس درهم ریختگی الگوریتم رگرسیون لجستیک

با توجه به ماتریس، تعداد TP و TN به طور کلی بیشتر از تعداد FP و FN است که نشانه خوبی است. این نشان می‌دهد که رگرسیون لجستیک موارد مثبت و منفی را با دقیقی معقول شناسایی می‌کند. با مقایسه چهار ماتریس درهم ریختگی، به نظر می‌رسد که سه ماتریس اول (TF-IDF, TF-IDF-2, Count Vectorize) عملکرد مشابهی دارند، با تعداد بالایی مثبت و منفی واقعی، و تعداد نسبتاً کم مثبت و منفی نادرست. این نتایج نشان می‌دهد که مدل رگرسیون لجستیک هنگام استفاده از این سه روش بردارسازی، عملکرد خوبی دارد. از سوی دیگر، چهارمین ماتریس درهم ریختگی (vader) در مقایسه با سه ماتریس دیگر، تعداد کمتری از مثبتها و منفی‌های درست را نشان می‌دهد. این نشان می‌دهد که این روش بردارسازی به اندازه سه روش دیگر برای این کلاس‌بندی مؤثر نبوده است. معیارهای دقت را در جدول زیر می‌توانید مشاهده کنید.

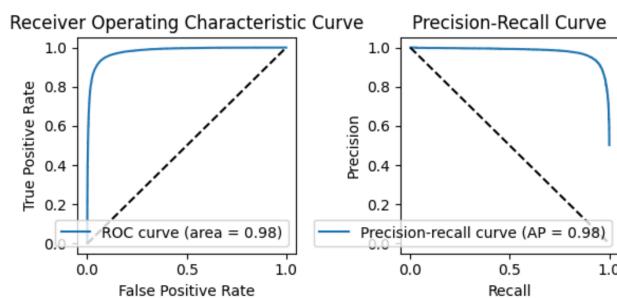
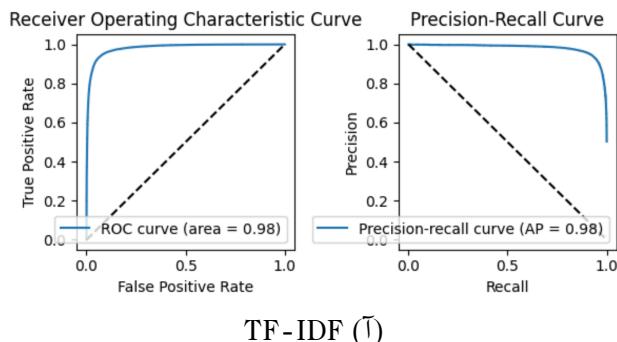
جدول ۳.۳: جدول نتایج رگرسیون لوجستیک

metric	TF-IDF	TF-IDF 2	CountVec	Vader
precision	0.929	0.926	0.910	0.684
recall	0.944	0.942	0.948	0.686
F1-score	0.936	0.934	0.928	0.685
accuracy	0.936	0.933	0.927	0.685

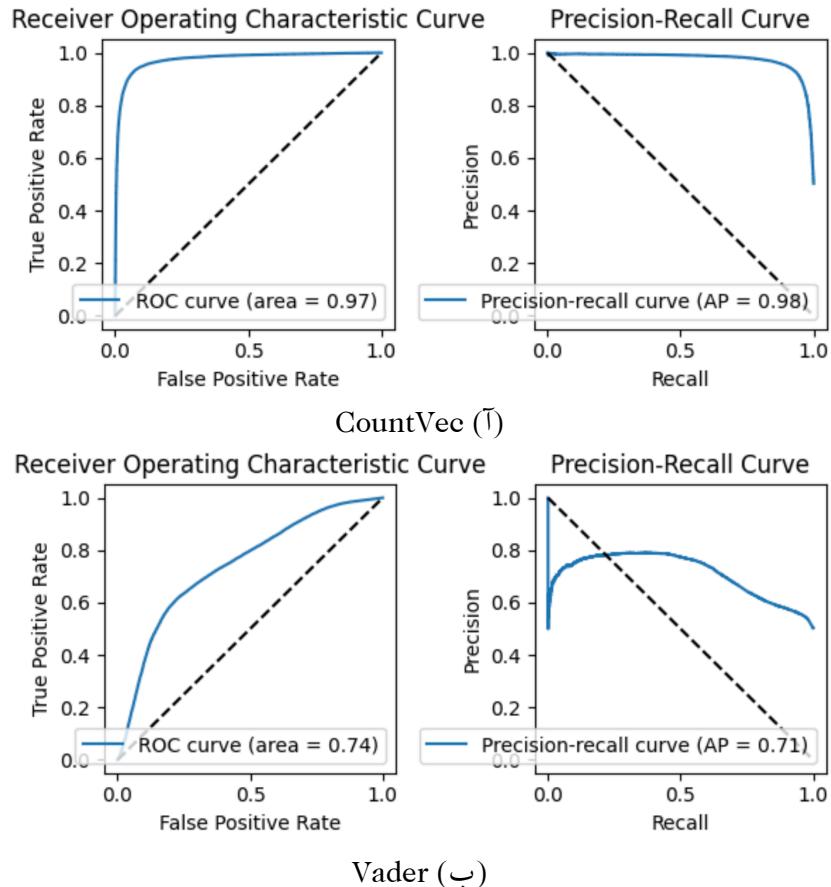


شکل ۷.۳: نمودار نتایج رگرسیون لجستیک

این نتایج نشان می‌دهد که با الگوریتم رگرسیون لجستیک، روش‌های بردار سازی TF-IDF و TF-IDF-2 موثرتر از روش‌های CountVec و Vader در این مسئله عمل کردند. حال به رسم نمودارهای مشخصه عملکرد و صحت-پوشش می‌پردازیم.



شکل ۸.۳: نمودارهای مشخصه عملکرد و صحت پوشش رگرسیون لجستیک



شکل ۹.۳: نمودارهای مشخصه عملکرد و صحت پوشش رگرسیون لجستیک

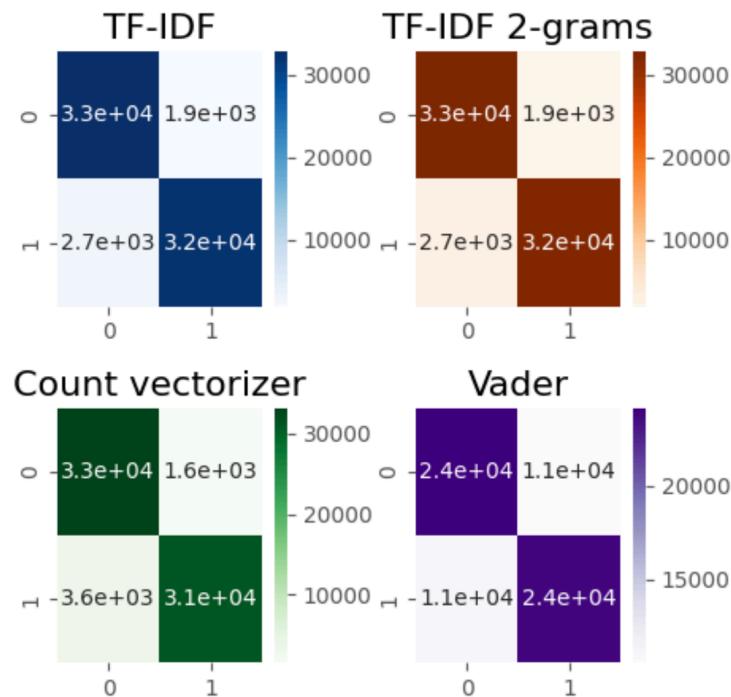
همانطور که از شکل ۶.۳ هم مشخص بود، با توجه به اینکه در جفت نمودارهای شکل ۸.۳ و بخش (آ) شکل ۹.۳ نمودار به نقطه‌ی بهینه بسیار نزدیک است، می‌توان نتیجه گرفت که الگوریتم نتیجه خوبی داشته. اما در نمودار (ب) شکل ۹.۳ می‌توان دید که نمودار از نقاط بهینه دور است و خوب عمل نکرده است.

۲۰۳ طبقه بند بردار پشتیبان خطی

با توجه به مجموعه داده های پروژه و همچنین توضیحات ذکر شده در فصل اول، به این نتیجه می رسیم که در این مسئله الگوریتم های رگرسیون لجستیک و بردار پشتیبان مثل یکدیگر عمل کرده و در یک صفحه های دو بعدی (صفحه $y-x$) منحنی های خطی و غیر خطی را بازش می کنند. پس نتایج این دو الگوریتم یکی خواهد بود. این موضوع از نتایج بردار پشتیبان خطی که در جدول زیر نمایش داده شده است نیز مبرهن است:

جدول ۴.۳: جدول نتایج ماشین بردار پشتیبان

metric	TF-IDF	TF-IDF 2	CountVec	Vader
precision	0.926	0.926	0.891	0.681
recall	0.944	0.902	0.956	0.692
F1-score	0.935	0.934	0.923	0.687
accuracy	0.935	0.934	0.920	0.686



شکل ۱۰.۳: بردار درهم ریختگی الگوریتم بردار پشتیبان

۳.۰.۳ الگوریتم جنگل تصادفی

پارامترهای تنظیم شده برای این الگوریتم عبارتند از:

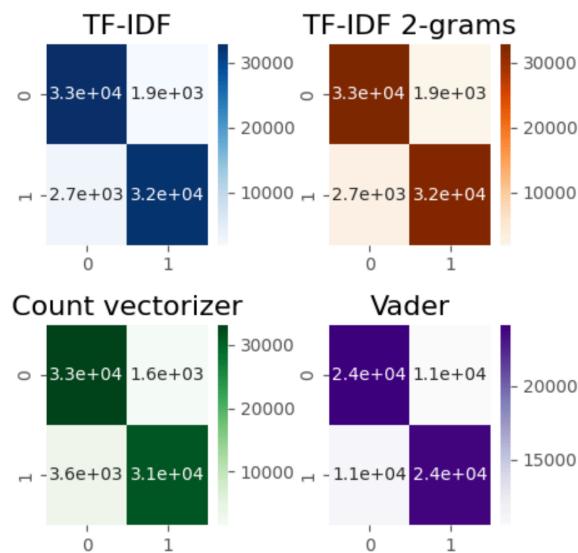
- max depth : این پارامتر نشان دهندهٔ حداکثر عمق هر کدام از درخت‌های تصمیم در جنگل تصادفی است. بیشتر بودن این پارامتر به معنای درخت‌های عمیق‌تر و در نتیجه مدل پیچیده‌تر است.
- min sample split : حداقل تعداد نمونه‌های لازم برای تقسیم کردن یک راس از درخت تصمیم را نشان می‌دهد. بیشتر کردن مقدار این پارامتر باعث می‌شود درخت با تعداد کم داده تقسیم نشود و می‌تواند جلوی بیش برازش مدل را بگیرد.
- min sample leaf : این پارامتر مینیمم تعداد داده لازم برای تشکیل یک راس از درخت را نشان می‌دهد. هر چه مقدار این پارامتر بیشتر باشد، داده بیشتری برای تشکیل یک راس لازم بوده و احتمال بیش برازش کمتر است.
- n estimators : تعداد درخت‌های تصمیم موجود در جنگل تصادفی را نشان می‌دهد. تعداد بیشتر درخت‌ها باعث می‌شود زمان بیشتری برای آموزش صرف شود ولی می‌تواند دقت مدل را افزایش بدهد.

پارامترهای بهینه بدست آمده برای این الگوریتم عبارتند از:

جدول ۵.۳: پارامترهای الگوریتم جنگل تصادفی

Method	min samples leaf	min samples split	n estimators
TF-IDF	1	5	200
TF-IDF(n=(1,2))	1	5	100
CountVectorizer	1	2	200
Vader(emotion)	2	5	50

با اعمال پارامترهای فوق و آموزش دادن مدل به نتایج زیر می‌رسیم:

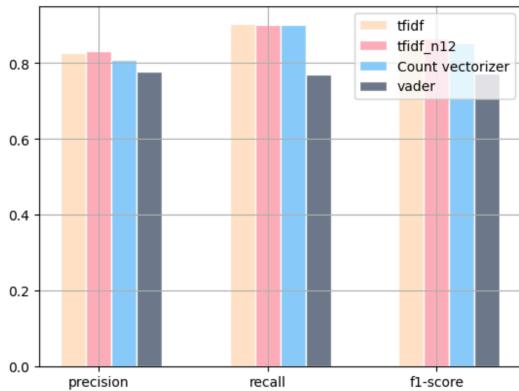


شکل ۱۱.۳: بردار درهم ریختگی الگوریتم جنگل تصادفی

با توجه به بردار درهم ریختگی شکل ۱۱.۳ متوجه می‌شویم که این الگوریتم نیز مانند الگوریتم‌های قبلی، روی داده‌های TF-IDF-2 و TF-IDF و Count Vectorizer نتیجه بهتری نسبت به داده‌های Vader دارد.

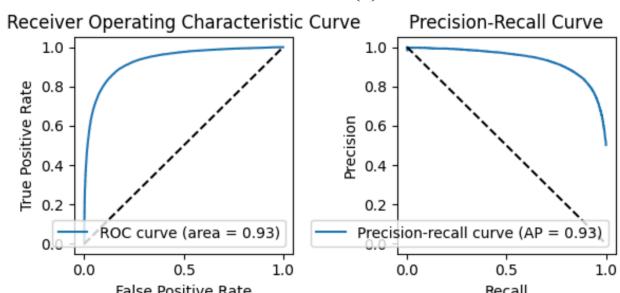
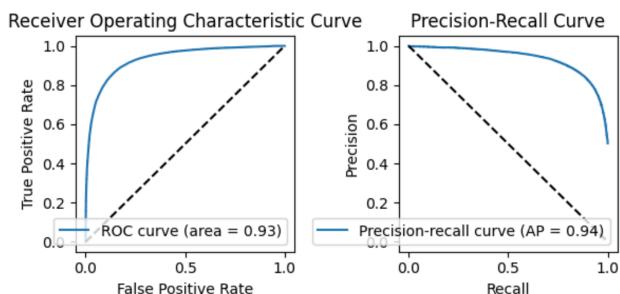
جدول ۶.۳: جدول نتایج الگوریتم جنگل تصادفی

metric	TF-IDF	TF-IDF 2	CountVec	Vader
precision	0.827	0.831	0.807	0.777
recall	0.904	0.902	0.900	0.769
F1-score	0.864	0.865	0.851	0.773
accuracy	0.857	0.859	0.843	0.774

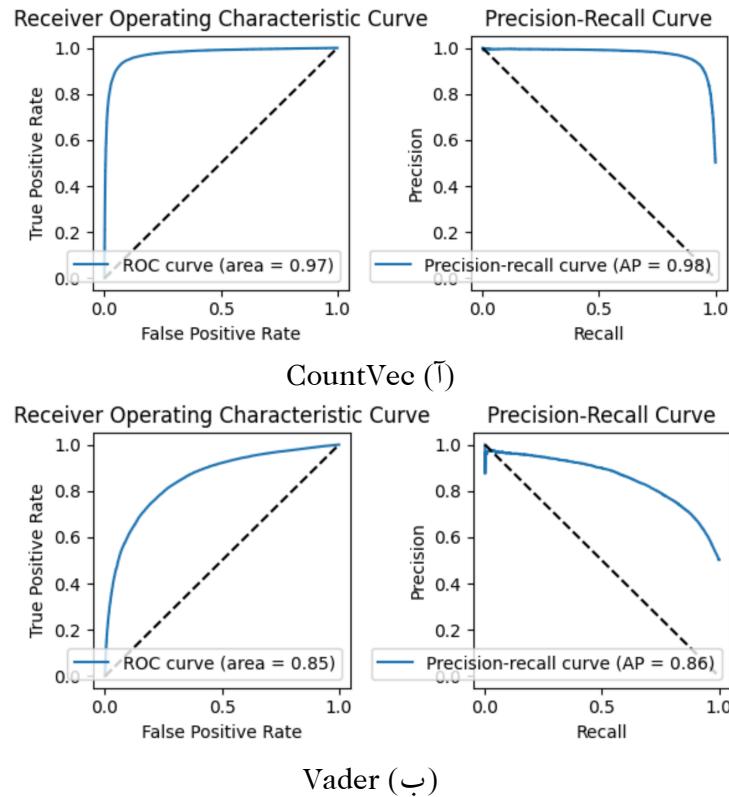


شکل ۱۲.۳: نمودار نتایج جنگل تصادفی

با توجه به نتایج بردار درهم ریختگی و نتایج بدست آمده، انتظار می‌رود نمودارهای مشخصه عملکرد و صحت-پوشش نیز نتایج مشابهی نشان دهند. در ادامه به رسم آن‌ها می‌پردازیم:



شکل ۱۳.۳: نمودارهای مشخصه عملکرد و صحت پوشش جنگل تصادفی



Vader (ب)

شکل ۱۴.۳ : نمودارهای مشخصه عملکرد و صحت پوشش جنگل تصادفی

همانطور که پیش‌بینی می‌شد، سه نمودار اول به نقاط بهینه نزدیک‌تر بوده و نتایج بهتری را نشان می‌دهند.
 لازم به ذکر است که در الگوریتم قبل مساحت زیر نمودار ROC برابر با 0.74 و مساحت زیر نمودار Precision-Recall برابر با 0.71 بود (شکل ۹.۳)، ولی در این الگوریتم این مساحت‌ها به ترتیب برابر با 0.85 و 0.86 هستند (شکل ۱۴.۳) که این نشان می‌دهد روش بردار سازی Vader در این الگوریتم نسبت به الگوریتم قبل نتایج بهتری داشته است.

۴۰۳ دسته بند بیز ساده

پارامترهایی که در این مدل تلاش برای بهبود آنها می‌کنیم عبارتند از:

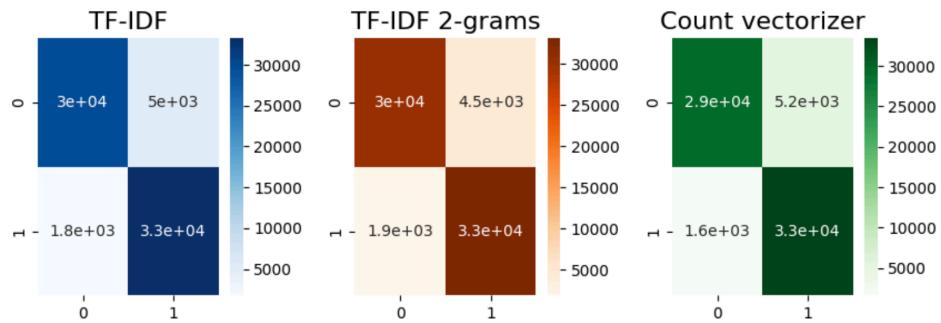
- alpha: در الگوریتم Naive Bayes اگر یکی از ویژگی‌ها در مجموعه داده آموخته شده باشد، احتمال محاسبه شده برای آن توسط الگوریتم صفر می‌شود. برای جلوگیری از این اتفاق، از پارامتر آلفا استفاده می‌شود. در واقع با این کار پس از محاسبه هر احتمال، مقدار مثبتی به آن اضافه می‌کنیم که این مقدار را اندازه آلفا تعیین می‌کند.
- prior: دامنه این پارامتر مقادیر صفر و یک است. اگر مقدار آن را برابر با ۱ قرار دهیم، مدل احتمال هر کلاس را از داده آموخته یاد می‌گیرد. در غیر این صورت، مدل در نظر می‌گیرد که احتمال وقوع کلاس‌ها با هم برابر است.

این الگوریتم را نمی‌توان روی داده‌ای که از روش Vader Sentiment Score بدست آمده آموخت داد زیرا در این نوع بردارسازی، ممکن است داده‌های ستون‌هایمان منفی باشد و این موضوع با فرض دسته‌بند بیز مبنی بر اینکه ویژگی منفی نداریم، همخوانی ندارد.
بهترین پارامترهای بدست آمده عبارتند از:

جدول ۷.۳: پارامترهای الگوریتم دسته‌بند بیز ساده

Method	alpha	fit prior
TF-IDF	0.1	False
TF-IDF(n=(1,2))	1.0	False
CountVectorizer	0.1	False

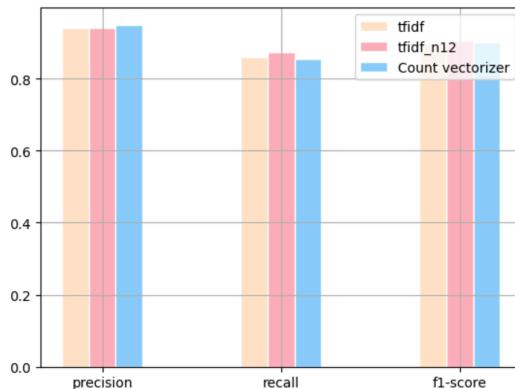
حال با استفاده از پارامترهای بهینه بدست آمده، مدل را آموخت می‌دهیم. نتایج در شکل ۱۵.۳ نشان داده شده است.



شکل ۱۵.۳: بردار درهم ریختگی الگوریتم دسته‌بند بیز

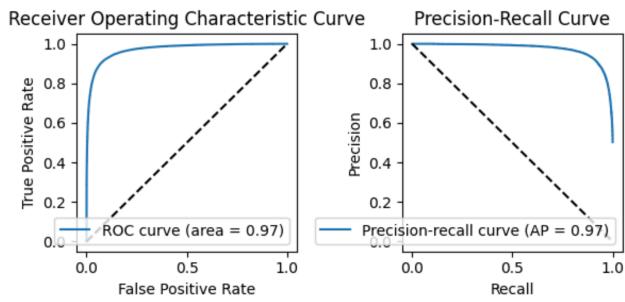
جدول ۸.۳: نتایج الگوریتم دسته‌بند بیز

metric	TF-IDF	TF-IDF 2	CountVec
precision	0.942	0.939	0.949
recall	0.858	0.873	0.854
F1-score	0.898	0.905	0.899
accuracy	0.902	0.908	0.904

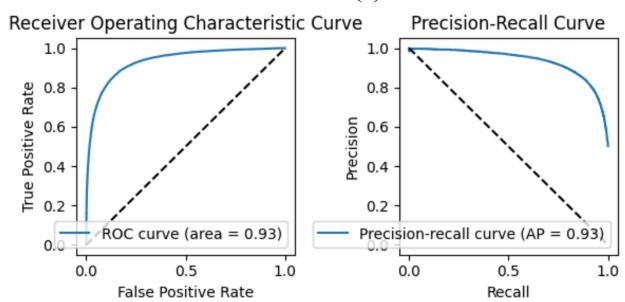


شکل ۱۶.۳: نمودار نتایج دسته‌بند بیز

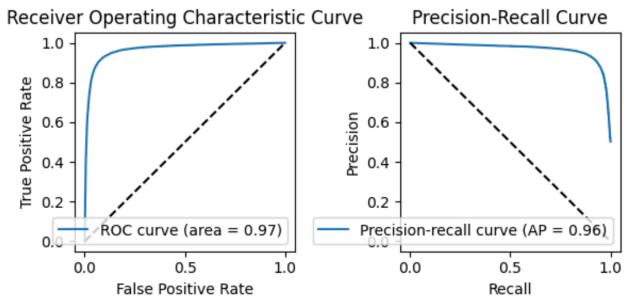
همانطور که مشاهده می‌کنید، هر سه روش برداری متن توانستند نتایج خوبی به دست آورند. نتایج نشان می‌دهد که معیارهای Accuracy و F1-score برای TF-IDF-2 و Count Vec. بالاتر از TF-IDF هستند، در حالی که معیارهای Recall و Precision برای TF-IDF و Count Vec. بالاتر از TF-IDF-2 هستند. بنابراین، برای این مجموعه داده، به نظر می‌رسد که Vec. روش برداری مناسب‌تری برای الگوریتم باشد.



TF-IDF (τ)



TF-IDF - ۲ (β)



CountVec (ج)

شکل ۱۷.۳: نمودارهای مشخصه عملکرد و صحت پوشش دسته‌بند بیز

با توجه به نمودارهای کشیده شده و مساحت زیر آن‌ها نیز به این نتیجه می‌رسیم که داده مدل بهترین نتیجه Count Vectorization را داده است.

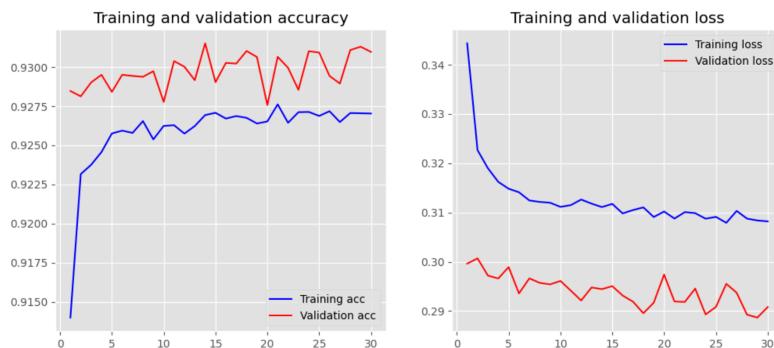
۵.۳ شبکه‌های عصبی

۱.۰۵.۳ شبکه عصبی ساده

ساختار شبکه عصبی آموزش داده شده در شکل ۱۸.۳ قابل مشاهده است. شبکه عصبی فوق را به تعداد ۳۰ دور آموزش می‌دهیم. سپس دقت آن را روی داده‌های تست حساب می‌کنیم. دقت حاصل برابر با ۰.۹۳۱۰٪ بست می‌آید. نتایج آموزش شبکه عصبی در شکل ۱۹.۳ نمایش داده شده است.

Layer (type)	Output Shape	Param #
=====		
dense_4 (Dense)	(None, 64)	631616
dropout_3 (Dropout)	(None, 64)	0
dense_5 (Dense)	(None, 32)	2080
dropout_4 (Dropout)	(None, 32)	0
dense_6 (Dense)	(None, 16)	528
dropout_5 (Dropout)	(None, 16)	0
dense_7 (Dense)	(None, 1)	17
=====		
Total params: 634241 (2.42 MB)		
Trainable params: 634241 (2.42 MB)		
Non-trainable params: 0 (0.00 Byte)		

شکل ۱۸.۳: ساختار شبکه عصبی



شکل ۱۹.۳: آموزش شبکه عصبی

۲.۵.۳ شبکه‌های از پیش آموزش داده شده

در این بخش از شبکه برتر استفاده شده. در استفاده از این نوع شبکه، نیاز به بردار سازی داده‌های ورودی نیست. ابتدا متن ورودی را توکن‌بندی کرده و به فرمت ورودی BERT تبدیل می‌کنیم. مدل با استفاده از بهینه‌ساز Adam بر روی مجموعه داده آموزشی دارای برچسب، آموزش می‌یابد و وزن‌های قبلی آن به روز رسانی می‌شوند. مدل را ۳ بار روی داده آموزشی آموزش می‌دهیم و پس از بهبود وزن‌ها، مدل بر روی یک مجموعه آزمون ارزیابی می‌شود و دقت آن حساب می‌شود. دقت داده آزمایشی پس از طی کردن این مراحل برابر با 0.9744 می‌شود که از نتایج تمامی مدل‌های یادگیری ماشین بهتر است. هرچند که برای ۳ درصد پیشرفت نسبت به بهترین مدل قبلی، حدود ۳۰ ساعت تلاش به بهبود وزن‌های این مدل کردیم.

فصل ۴

جمع بندی

در این پایان نامه، با بررسی ادبیات و تحلیل داده‌های موجود در پلتفرم‌های آنلاین، مدلی با دقت بالا برای شناسایی افراد در معرض خطر خودکشی پیشنهاد دادیم. تحلیل داده‌های متنی از جمله متون نوشته شده توسط افراد در شبکه‌های اجتماعی، ایمیل‌های ارسال شده و پیام‌های متنی ارسال شده به دیگران، به عنوان یکی از روش‌های مؤثر در شناسایی افراد در معرض خطر خودکشی، مورد استفاده قرار گرفت. با استفاده از مدل‌های یادگیری ماشین و شبکه‌های عصبی، الگوهای مشترک در متون مورد بررسی شناسایی شده و مدلی برای شناسایی افراد در معرض خطر توسعه داده شده است.

نتایج حاصل از پایان نامه نشان می‌دهد که استفاده از تحلیل داده‌های متنی به عنوان یک روش نوین در شناسایی افراد در معرض خطر خودکشی، قابلیت‌های بسیاری را برای بهبود خدمات بهداشتی و درمانی فراهم می‌کند. با استفاده از مدل‌های پیشنهاد شده در این پایان نامه، می‌توان به صورت زودهنگام افرادی را شناسایی کرد که در معرض خطر خودکشی هستند و به آن‌ها کمک کرد تا به موقع دریافت خدمات ارزیابی و درمانی شوند.

نتایج آزمایش مدل‌ها نشان داد که بیشترین دقت با استفاده از مدل BERT به عنوان یک شبکه‌ی عصبی پیش‌آموزش دیده و فاین‌تیون شده، ۹۷ درصد به دست آمده است. همچنین، مدل‌های یادگیری ماشین ذکر شده که از روش‌های TF-IDF و CountVectorization برای بردارسازی استفاده کردند، دارای دقت‌هایی بین ۸۶ تا ۹۳ درصد بودند. بالاترین دقت نیز از مدل رگرسیون لوجستیک بدست آمد و نتایج این مدل تقریباً برابر با شبکه عصبی استفاده شده در این پژوهه است.

واژه‌نامه انگلیسی به فارسی

Accuracy	دقت
Activation Function.....	تابع فعال‌سازی
Artificial Intelligence.....	هوش مصنوعی
Artificial Neural Network	شبکه‌های عصبی مصنوعی
Axon	آکسون
Backpropagation.....	پس‌انتشار خطای
Bayesian Algorithms.....	الگوریتم‌های بیزی
BERT	برت
Bias	اریبی
Clustering Algorithms.....	الگوریتم‌های خوشه‌بندی
Classification Algorithms	الگوریتم‌های طبقه‌بندی
Confusion Matrix.....	ماتریس درهم‌ریختگی
Deep Neural Networks.....	شبکه‌های عصبی عمیق
Dendrite	دندربیت
Evidence	شرط
F1-score	بازخوانی
False Positive Rate	نرخ مثبت کاذب
Fine-tune	تنظیم
Generative Adversarial Networks	شبکه‌های مولد
Global Minimums	کمینه‌های کلی
Gradient Descent	کاهش گرادیان
Grid Search	جستجوی خطی
Hidden Layer	لایه مخفی
Hypothesis	فرضیه
Hyperplane	ابر صفحه
Input Layer	لایه ورودی
Language Modeling	مدل کردن زبانی

Lemmatization	ریشه‌یابی کلمات
Local Minimums	کمینه‌های موضعی
Machine Learning	یادگیری ماشین
Margin Distance.....	فاصله حاشیه
MSE	تابع میانگین مربعات خط
Naive	ساده
Naive Bayes Classifier.....	دسته‌بند بیز
Natural Language Processing.....	پردازش زبان طبیعی
Neural Networks	شبکه‌های عصبی
Next Sequence Prediction	پیش‌بینی عبارت بعدی
Objective Function	تابع هدف
Overfitting.....	بیش‌برازش
Output Layer	لایه خروجی
Precision	دقت پیش‌بینی
Precision-Recall Curve	منحنی صحت-پوشش
Pre-trained Neural Networks	شبکه‌های عصبی از پیش آموزش داده شده
Random Forest	جنگل تصادفی
Random Search	جستجوی تصادفی
Receiver Operating Characteristic plot	نمودار مشخصه عملکرد فراخوانی
Recall	رجرسیون
Regression	نظم‌سازی
Regularization	یادگیری تقویتی
Reinforcement Learning	تحلیل احساسات
Sentiment Analysis	تابع سیگموید
Sigmoid Function	ماشین بردار پشتیبان
Support Vector Machine (SVM)	یادگیری با ناظارت
Supervised Learning	طبقه‌بندی متن
Text Classification	مجموعه آموزشی
Training Set	یادگیری بدون ناظارت
Unsupervised Learning	مجموعه اعتبارسنجی
Validation Set	بردار
Vector	کد گذاری کلمات
Word Embedding	

واژه‌نامه فارسی به انگلیسی

Axon	آکسون
Hyperplane	ابر صفحه
Bias	اربیسی
Bayesian Algorithms	الگوریتم‌های بیزی
Clustering Algorithms	الگوریتم‌های خوشه‌بندی
Classification Algorithms	الگوریتم‌های طبقه‌بندی
F1-score	بازخوانی
BERT	برت
Vector	بردار
Overfitting	بیش‌برازش
Natural Language Processing	پردازش زبان طبیعی
Backpropagation	پساننتشار خطای
Next Sequence Prediction	پیش‌بینی عبارت بعدی
Sigmoid Function	تابع سیگموید
Activation Function	تابع فعال‌سازی
MSE	تابع میانگین مربعات خطای
Objective Function	تابع هدف
Sentiment Analysis	تحلیل احساسات
Fine-tune	تنظیم
Random Search	جستجوی تصادفی
Grid Search	جستجوی خطی
Random Forest	جنگل تصادفی
Accuracy	دقت
Precision	دقت پیش‌بینی
Dendrite	دندریت
Regression	رگرسیون
Lemmatization	ریشه‌یابی کلمات

Naive	ساده
Neural Networks	شبکه‌های عصبی
Pre-trained Neural Networks	شبکه‌های عصبی از پیش آموزش داده شده
Deep Neural Networks	شبکه‌های عصبی عمیق
Artificial Neural Network	شبکه‌های عصبی مصنوعی
Generative Adversarial Networks	شبکه‌های مولد
Evidence	شرط
Text Classification	طبقه‌بندی متن
Margin Distance	فاصله حاشیه
Recall	فراخوانی
Hypothesis	فرضیه
Gradient Descent	کاهش گرادیان
Word Embedding	کد گذاری کلمات
Input Layer	لایه ورودی
Global Minimums	کمینه‌های کلی
Local Minimums	کمینه‌های موضعی
Output Layer	لایه خروجی
Hidden Layer	لایه مخفی
Input Layer	لایه ورودی
Confusion Matrix	ماتریس درهم‌ریختگی
Support Vector Machine (SVM)	ماشین بردار پشتیبان
Training Set	مجموعه آموزشی
Validation Set	مجموعه اعتبارسنجی
Language Modeling	مدل کردن زبانی
Precision-Recall Curve	منحنی صحت-پوشش
Regularization	نظم‌سازی
False Positive Rate	نرخ مثبت کاذب
Receiver Operating Characteristic plot	نمودار مشخصه عملکرد
Artificial Intelligence	هوش مصنوعی
Supervised Learning	یادگیری با نظارت
Unsupervised Learning	یادگیری بدون نظارت
Reinforcement Learning	یادگیری تقویتی
Machine Learning	یادگیری ماشین

كتاب نامه

- [1] Ethem Alpaydin (2020). Introduction to Machine Learning (Fourth ed.).
- [2] Samuel, Arthur (1959). "Some Studies in Machine Learning Using the Game of Checkers". IBM Journal of Research and Development
- [3] Russell, Stuart J.; Norvig, Peter (2010). Artificial Intelligence: A Modern Approach (Third ed.).
- [4] Mohri, Mehryar; Rostamizadeh, Afshin; Talwalkar, Ameet (2012). Foundations of Machine Learning.
- [5] Mitchell, T. (1997). Machine Learning. McGraw Hill.
- [6] Alpaydin, Ethem (2010). Introduction to Machine Learning. MIT Press.
- [7] Zhang, Bosen; Huang, Haiyan; Tibbs-Cortes, Laura E.; Vanous, Adam; Zhang, Zhiwu; Sanguinet, Karen; Garland-Campbell, Kimberly A.; Yu, Jianming; Li, Xianran (2023). "Streamline unsupervised machine learning to survey and graph indel-based haplotypes from pan-genomes"
- [8] Van Otterlo, M.; Wiering, M. (2012). Reinforcement learning and Markov decision processes. Reinforcement Learning. Adaptation, Learning, and Optimization
- [9] David A. Freedman (2009). Statistical Models: Theory and Practice. Cambridge University Press. p.26. "A simple regression equation has on the right hand side an intercept and an explanatory variable with a slope coefficient. A multiple regression e right hand side, each with its own slope coefficient"

- [10] Rencher, Alvin C.; Christensen, William F. (2012), "Chapter 10, Multivariate regression – Section 10.1, Introduction", Methods of Multivariate Analysis, Wiley Series in Probability and Statistics, vol.709 (3rd ed.), John Wiley and Sons
- [11] Barzilai, Jonathan; Borwein, Jonathan M. (1988). "Two-Point Step Size Gradient Methods". IMA Journal of Numerical Analysis. 8 (1): 141–148. doi:10.1093/imanum/8.1.141.
- [12] Fletcher, R. (2005). "On the Barzilai–Borwein Method". In Qi, L.; Teo, K.; Yang, X. (eds.). Optimization and Control with Applications. Applied Optimization. Vol. 96. Boston: Springer. pp.235–256. ISBN 0-387-24254-6.
- [13] Hosmer, David W.; Lemeshow, Stanley (2000). Applied Logistic Regression (2nd ed.). Wiley. ISBN 978-0-471-35632-5
- [14] Cortes, Corinna; Vapnik, Vladimir N. (1995). "Support-vector networks". Machine Learning. 20 (3): 273–297. doi:10.1007/BF00994018.
- [15] Agarap, A. F. M. (2018, February). A neural network architecture combining gated recurrent unit (GRU) and support vector machine (SVM) for intrusion detection in network traffic data. In Proceedings of the 2018 10th international conference on machine learning and computing (pp. 26-30)
- [16] Ben-Hur, Asa; Horn, David; Siegelmann, Hava; Vapnik, Vladimir N. "Support vector clustering" (2001);. Journal of Machine Learning Research. 2: 125–137
- [17] Ho, Tin Kam (1995). Random Decision Forests (PDF). Proceedings of the 3rd International Conference on Document Analysis and Recognition, Montreal, QC, 14–16 August 1995. pp. 278–282
- [18] Joyce, James (2003), "Bayes' Theorem", in Zalta, Edward N. (ed.), The Stanford Encyclopedia of Philosophy (Spring 2019 ed.), Metaphysics Research Lab, Stanford University, retrieved 2020-01-17
- [19] B. Kroese and V. D P. Smagt (1996). An Introduction to Neural Networks, University of Amsterdam, Amsterdam, Switzer-land.

- [20] Russell, Ingrid(2014). "Neural Networks Module". Archived from the original on May 29.
- [21] Devlin, Jacob; Chang, Ming-Wei; Lee, Kenton; Toutanova, Kristina (11 October 2018). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding"
- [22] Claesen, Marc; Bart De Moor (2015). "Hyperparameter Search in Machine Learning".
- [23] Chicco D (December 2017). "Ten quick tips for machine learning in computational biology".
- [24] Liu, H.; Christiansen, T.; Baumgartner, W. A.; Verspoor, K. (2012). "BioLemmatizer: A lemmatization tool for morphological processing of biomedical text". Journal of Biomedical Semantics.
- [25] Rajaraman, A.; Ullman, J.D. (2011). "Data Mining" (PDF). Mining of Massive Datasets. pp. 1–17
- [26] Vong Anh Ho, Duong Huynh-Cong Nguyen, Danh Hoang Nguyen, Linh Thi-Van Pham, Duc-Vu Nguyen, Kiet Van Nguyen, Ngan Luu-Thuy Nguyen(2019). "Emotion Recognition for Vietnamese Social Media Text". In Proceedings of the 2019 International Conference of the Pacific Association for Computational Linguistics (PACLING 2019), Hanoi, Vietnam.
- [27] Powers, David M. W. (2011). "Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness and Correlation". Journal of Machine Learning Technologies.
- [28] Fawcett, Tom (2006). "An Introduction to ROC Analysis" (PDF). Pattern Recognition Letters.
- [29] garap, A. F. M. (2018, February). A neural network architecture combining gated recurrent unit (GRU) and support vector machine (SVM) for intrusion detection in network traffic data. In Proceedings of the 2018 10th international conference on machine learning and computing (pp. 26-30).

- [30] mith, J. D., and Johnson, A. B. (2022). A Comparative Study of Stop Words Removal Techniques in Natural Language Processing. *Journal of Computational Linguistics*, 15(3), 45-62.
- [31] onzalez, J., Martinez, R., and Lopez, A. (2019). Evaluating Classification Models Using Precision-Recall Curves. *Journal of Machine Learning Research*, 20(3), 112-130.
- [32] ames, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). *An Introduction to Statistical Learning: With Applications in R*. Springer.
- [33] üller, B., Reinhardt, J., and Strickland, M. T. (1995). *Neural networks: an introduction*. Springer Science and Business Media.

Abstract

Suicide is a major public health concern, and early identification of individuals at risk of suicidal behavior can be a critical step in preventing suicide. However, identifying individuals at risk of suicide can be a challenging task, as suicidal behavior is often a complex and multifaceted phenomenon that can be difficult to detect. In recent years, text mining techniques and machine learning have been increasingly used to analyze text data and identify patterns that can be indicative of suicidal behavior.

The aim of this thesis is to develop a high-precision model that can identify texts that indicate suicidal behavior using text mining techniques and machine learning. To achieve this aim, text data from the social media platform Reddit was used, as it provides a rich source of text data that can be used to analyze the language of individuals who may be at risk of suicidal behavior. A combination of machine learning and deep learning techniques were used to develop the model, including Support Vector Machines (SVM), Random Forests, and Neural Networks.

The developed model achieved high accuracy in identifying texts that indicate suicidal behavior, demonstrating the potential of text mining and machine learning in identifying individuals at risk of suicide. This thesis can serve as a basis for further research in this field and provide solutions for improving public health by enabling early identification and intervention for individuals at risk of suicidal behavior.



College of Science
School of Mathematics, Statistics, and Computer Science

Investigating the relationship between a person's written text and their suicide using text mining and machine learning methods

Narges Baba Ahmadi

Supervisor: Bagher Babaali

A thesis submitted in partial fulfillment of the requirements for
the degree of B.Sc. in Computer Science

Summer 2023