

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ



دانشگاه فنی و حرفه ای

دانشکده فنی و حرفه ای دختران تهران – دکتر شریعتی

پایان نامه کارشناسی ناپیوسته

رشته کامپیوتر گرایش مهندسی حرفه ای نرم افزار

عنوان:

مدل سازی جوامع عاطفی و تشخیص احساسات

استاد راهنما:

سرکار خانم دکتر زهرا ولدخانی

نگارش:

زهرا رحیمیان

پاییز ۱۴۰۲

تقديم به:

آنان كه آرزو داشتند وطن جايي براي ماندن بود...

تشکر و قدردانی

سپاسگزارم از استاد گرامی‌ام که من را به سمت یکی از مباحث روز و ارزشمند راهنمایی کردند و همچنین سپاسگزارم از خانواده عزیزم که همواره حامی و پشتیبان من بوده‌اند.

چکیده

این پایان‌نامه به هدف شناسایی و تحلیل جوامع عاطفی در شبکه‌های اجتماعی از طریق تکنیک‌های پیشرفته علم داده و ماشین لرنینگ می‌پردازد. با استفاده از دیتاست‌های معتبر و روش‌های نوین پردازش متن، رویکردی نوآورانه برای تحلیل احساسات کاربران توسعه یافته است. یافته‌های تحقیق حاکی از آن است که مدل‌های پیاده‌سازی شده قادر به تشخیص دقیق و اتوماتیک حالات عاطفی متن‌های فارسی هستند. نتایج حاصل از این پژوهش، دلالت‌های مهمی برای طراحی سیستم‌های توصیه‌گر و تجزیه و تحلیل رفتار کاربران در پلتفرم‌های دیجیتال دارد و پیشنهادهایی برای بهبود سیستم‌های موجود و توسعه روش‌های تحلیل متن در زبان فارسی ارائه می‌دهد.

کلیدواژه: علم داده، ماشین لرنینگ، تشخیص احساسات، تشخیص جامعه

فهرست نوشتار

فصل اول: «مقدمه»

- ۱-۱ انگیزه ۱
- ۱-۲ هدف ۱
- ۱-۳ رئوس مطالب سایر فصل‌ها ۱

فصل دوم: «تجزیه و تحلیل نیازمندی‌ها»

- ۲-۱ مقدمه ۳
- ۲-۲ نتیجه‌گیری ۳

فصل سوم: «تشخیص جامعه عاطفی در شبکه های اجتماعی»

- ۳-۱ چکیده ۵
- ۳-۲ مقدمه ۵
- ۳-۳ تعریف اجتماعات مشتق شده ۶
- ۳-۴ تحلیل احساسات و شناسایی جوامع ۶
- ۳-۵ آشنایی با مقیاس اکمن ۷
- ۳-۶ استفاده از الگوریتم‌های تشخیص جامعه برای تحلیل رفتار عاطفی کاربران ۷
- ۳-۶-۱ فرآیند تشخیص احساس پست ۸
- ۳-۶-۲ تحلیل پروفایل کاربر و سنج تاثیرگذاری ۸
- ۳-۶-۳ رفتار احساسی کاربر ۱۰
- ۳-۶-۴ تشخیص جامعه ۱۱
- ۳-۶-۵ نمایش نمودار احساسی ۱۲
- ۳-۷ نتیجه‌گیری ۱۳

فصل چهارم: «پیاده سازی»

- ۴-۱ مقدمه ۱۵
- ۴-۲ معرفی دیتاست استفاده شده ۱۵
- ۴-۳ کدنویسی ۱۶
- ۴-۳-۱ نصب نیازمندی‌ها و اضافه کردن کتابخانه‌های لازم ۱۶

۱۸	۴-۳-۲ وارد کردن دیتاست و خواندن محتوای آن
۲۱	۴-۳-۳ پیش پردازش
۲۸	۴-۳-۴ بردارسازی متن
۲۹	۴-۳-۵ ساخت و ارزیابی مدل
۳۴	۴-۳-۶ مصورسازی
۳۵	۴-۳-۷ ذخیره سازی مدل آموزش داده شده
۳۶	۴-۳-۸ استقرار مدل با پکیج Flask
۳۷	۴-۳-۹ کدهای HTML, CSS, js
۳۷	۴-۴ اجرای نرم افزار
۳۹	۴-۵ نتیجه گیری

فصل پنجم: «جمع بندی و پیشنهادها»

۴۱	۵-۱ نتیجه گیری
۴۱	۵-۲ پیشنهادهایی برای کارهای آتی
۴۲	فهرست منابع
۴۳	چکیده انگلیسی

فهرست شکل‌ها

شکل ۳-۱ معماری سیستم.....	۷
شکل ۳-۲ معماری ابزار تشخیص احساس.....	۸
شکل ۳-۳ تبدیل به گراف خطی.....	۱۲
شکل ۴-۱ دیتاست کامنت‌های اسنپ فود.....	۱۵
شکل ۴-۲ نصب Hazm روی سیستم لوکال.....	۱۶
شکل ۴-۳ نصب و وارد کردن کتابخانه‌های ضروری.....	۱۷
شکل ۴-۴ وارد کردن دیتاست.....	۱۸
شکل ۴-۵ بررسی ساختار دیتافریم.....	۱۹
شکل ۴-۶ نمایش آمار توصیفی دیتاست.....	۲۰
شکل ۴-۷ بررسی مقادیر یکتای دیتاست.....	۲۱
شکل ۴-۸ حذف ستون Unnamed.....	۲۲
شکل ۴-۹ فیلتر کردن سطرها.....	۲۲
شکل ۴-۱۰ تبدیل نوع داده.....	۲۳
شکل ۴-۱۱ جداسازی نظرات و برچسب‌ها.....	۲۴
شکل ۴-۱۲ تقسیم داده‌ها به دو مجموعه آموزشی و تست.....	۲۴
شکل ۴-۱۳ نرمال‌سازی داده.....	۲۵
شکل ۴-۱۴ توکن‌سازی کلمات.....	۲۶
شکل ۴-۱۵ ریشه‌یابی کلمات.....	۲۶
شکل ۴-۱۶ واژه‌یابی.....	۲۷
شکل ۴-۱۷ تبدیل کلمات به رشته متنی.....	۲۷
شکل ۴-۱۸ بردارسازی متن.....	۲۸
شکل ۴-۲۲ مدل‌های ساخته شده.....	۳۰
شکل ۴-۲۳ ویژگی‌های XGBoost.....	۳۲
شکل ۴-۲۴ ساخت Pipeline.....	۳۳
شکل ۴-۲۵ کدهای مصورسازی.....	۳۴
شکل ۴-۲۶ نتیجه مصورسازی.....	۳۵
شکل ۴-۲۷ ساخت فایل pickle.....	۳۵
شکل ۴-۲۸ کد پکیج فلسک.....	۳۶
شکل ۴-۲۹ بخشی از کد رابط کاربری.....	۳۷
شکل ۴-۳۰ وب‌سایت تشخیص احساس کامنت.....	۳۸
شکل ۴-۳۱ وارد کردن نظر مثبت و دیدن نتیجه.....	۳۸

شکل ۳۲-۴ وارد کردن نظر منفی و دیدن نتیجه ۳۹

فصل اول

«مقدمه»

۱-۱ انگیزه

اهمیت و ضرورت تحقیق در زمینه تحلیل احساسات در داده‌های فارسی زبان از این واقعیت ناشی می‌شود که فضای مجازی به عرصه‌ای برای بیان نظرات و احساسات کاربران تبدیل شده است. با توجه به حجم عظیم داده‌های تولیدی در شبکه‌های اجتماعی، استفاده از تکنیک‌های پیشرفته یادگیری ماشین و پردازش زبان طبیعی برای شناسایی و تحلیل احساسات، نه تنها برای سازمان‌ها و کسب‌وکارها بلکه برای مطالعات اجتماعی و روانشناختی حیاتی است. مطالعات مشابه، اگرچه در سایر زبان‌ها به وفور انجام شده‌اند، اما در حوزه زبان فارسی هنوز جای کار بسیار دارند و با چالش‌هایی مانند کمبود منابع داده‌ای معتبر و ابزار تحلیل مناسب روبرو هستند.

۱-۲ هدف

این تحقیق به دنبال پر کردن خلأ پردازش زبان طبیعی با زبان فارسی است و با ارائه یک مدل دقیق و کارآمد، در راستای بهبود درک ما از جامعه دیجیتال فارسی‌زبان گام برمی‌دارد.

۱-۳ رئوس مطالب سایر فصل‌ها

این تحقیق دارای فصل‌های مقدمه، تجزیه و تحلیل نیازمندی‌ها، تشخیص جامعه عاطفی در شبکه‌های اجتماعی، پیاده‌سازی و جمع‌بندی و پیشنهادات است.

فصل دوم

«تجزیه و تحلیل

نیازمندی‌ها»

۲-۱ مقدمه

در این پروژه، از ابزارها و محیط‌های توسعه متنوعی برای اجرای بخش‌های مختلف کار استفاده شده. برای اجرای کدهای پکیج فلسک، از محیط توسعه اسپایدر استفاده شده است. برای نوشتن کدهای وب پیچ، از محیط توسعه vscode استفاده شده است. به کارگیری نوت‌بوک کولب برای پیاده‌سازی و تحلیل داده‌ها نشان‌دهنده تمایل به استفاده از ابزارهای تحلیل داده در محیط ابری است. استفاده از مقالات دسترسی‌پذیر از گوگل اسکولار برای ترجمه و تحلیل محتوا نیز بیانگر تمرکز بر تحقیق و استفاده از منابع علمی معتبر است. در نهایت، از یک دیتاست معتبر موجود در کگل برای تأمین داده‌های لازم برای تحقیق یک استفاده شده است.

۲-۲ نتیجه‌گیری

در این فصل ابزارهای استفاده شده برای نگارش این پایان نامه و انجام پروژه عملی معرفی شدند. در فصل بعد به بررسی دیتاست و داده‌های استفاده شده پرداخته شده است.

فصل سوم

«تشخیص جامعه عاطفی

در شبکه‌های اجتماعی»

۳-۱ چکیده

بررسی و تحلیل شبکه‌های اجتماعی به‌طور گسترده به عنوان یک حوزه تحقیقی پیچیده و چالش‌برانگیز شناخته شده است. نمونه بارز این موضوع، سازماندهی رأس‌ها در خوشه‌ها است که بسیاری از یال‌ها رأس‌های همان خوشه را به هم متصل می‌کنند و تعداد کمتری یال رأس‌های خوشه‌های مختلف را به هم می‌رسانند. این موضوع شامل جنبه‌ای اساسی است که مربوط به تشخیص اجتماعات کاربران می‌شود. در حوزه‌هایی نظیر جامعه‌شناسی و علوم کامپیوتر که تعاملات و ارتباطات اغلب به صورت گراف‌ها نمایش داده می‌شوند، تشخیص اجتماعات از اهمیت ویژه‌ای برخوردار است. مقیاس عاطفی اکمن نقطه کلیدی است که با آن رفتار عاطفی کاربران در توییت‌هایشان تحلیل می‌شود. در نتیجه، اجتماعات مشتق شده با استفاده از سه معیار مختلف ارزیابی می‌شوند، در حالی که نسخه وزن دار یک الگوریتم تشخیص اجتماعات مدولاریتی مورد استفاده قرار می‌گیرد. شواهد کافی وجود دارد که نشان می‌دهد روش پیشنهادی ما اجتماعات تأثیرگذاری ایجاد می‌کند.

۳-۲ مقدمه

در چند سال گذشته، توییت به همراه سایر رسانه‌های اجتماعی محبوبیت فزاینده‌ای پیدا کرده است. این امر منجر به ایجاد دغدغه‌های پژوهشی عظیم و فرصت‌های جدیدی برای مطالعه تعاملات متقابل گروه‌های مختلف افراد شده است. شناسایی جوامع و تحلیل احساسات دو نمونه از این موارد هستند که به عنوان موضوعات محبوب در بررسی و درک بهتر شبکه‌های اجتماعی شناخته می‌شوند. از یک طرف، شناسایی جوامع در تلاش است تا شبکه‌های اجتماعی را با هدف اصلی شناسایی خوشه‌هایی از کاربران مرتبط و وابسته به یکدیگر تحلیل کند، و از طرف دیگر، تحلیل احساسات سعی دارد تا به رفتار کاربران در سطح احساسی پی ببرد و در نتیجه نگرش آن‌ها را نسبت به موضوعات متنوعی مانند نحوه احساس افراد تعیین کند. تعیین رفتار کاربران در هر یک از جوامع ظهور یافته و همچنین در کل شبکه، یکی از اساسی‌ترین جنبه‌های تحلیل شبکه‌های اجتماعی است و مفهومی پویا برای تجزیه و تحلیل دقیق روش‌های ارتباط کاربران برای ایجاد جوامع اجتماعی می‌باشد. برای توضیح دینامیک‌های اجتماعی تعامل بین گروه‌های افراد، لازم است که ما ساختار جامعه شبکه را مطالعه کنیم.

۳-۳ تعریف اجتماعات مشتق شده

اجتماعات مشتق شده به گروه‌های یا خوشه‌هایی اشاره دارد که از طریق تجزیه و تحلیل داده‌های شبکه‌های اجتماعی به دست آمده‌اند. این اجتماعات می‌توانند مجموعه‌هایی از کاربران باشند که بر اساس رفتار یا ویژگی‌های مشابهشان در شبکه‌های اجتماعی شناسایی شده‌اند. برای سنجش و بررسی دقت و کارایی این اجتماعات، سه روش مختلف ارزیابی به کار رفته است. این معیارها ممکن است شامل مواردی مانند تراکم ارتباطات درون اجتماع، میزان همپوشانی با اجتماعات واقعی، و یا معیارهای دیگری باشند که برای ارزیابی کیفیت تشخیص اجتماعات در نظر گرفته شده‌اند.

در تحلیل اجتماعات، از یک الگوریتم خاص که بر مبنای مدولاریتی (یک مفهوم در شبکه‌های اجتماعی که برای اندازه‌گیری قدرت یک تقسیم‌بندی شبکه به اجتماعات استفاده می‌شود) کار می‌کند، استفاده شده است. عبارت وزن‌دار به این معنی است که در حین تحلیل داده‌ها، برخی از عوامل یا مشخصات خاص با اهمیت بیشتری در نظر گرفته شده‌اند.

۳-۴ تحلیل احساسات و شناسایی جوامع

روش‌های شناسایی جامعه کاربران در شبکه‌های اجتماعی نقش مهمی در ارتقاء استراتژی‌های اقتصادی و بازاریابی دارند. این روش‌ها با شناسایی دقیق گروه‌های کاربران مرتبط و تحلیل رفتار و ترجیحات آن‌ها، امکان هدف‌گیری و اجرای طرح‌های تبلیغاتی موثرتر در شبکه‌های خاص را فراهم می‌کنند. بدین ترتیب، ارائه توصیه‌ها و محتواهای سازگار با علایق و نیازهای کاربران میسر می‌شود. علاوه بر این، تحلیل نگرش‌ها و تصمیمات مردم بر اساس محتوای موجود در شبکه‌های اجتماعی اهمیت ویژه‌ای دارد، زیرا این نگرش‌ها می‌توانند تحت تأثیر جو عمومی شکل گرفته و بر تمام جنبه‌های زندگی تأثیرگذار باشند. شناسایی وضعیت احساسی کاربران از طریق تحلیل محتوای تولید شده توسط آن‌ها در شبکه‌های اجتماعی نه تنها یک چالش است بلکه می‌تواند بینش‌های مهمی در مورد وضعیت احساسی یک جامعه یا منطقه ارائه دهد. در نهایت، درک این جنبه‌های احساسی رفتار کاربران به ما کمک می‌کند تا جوامع را با دقت بیشتری شناسایی کرده و ساختارهای اجتماعی متراکم‌تر و معنادارتری را توسعه دهیم.

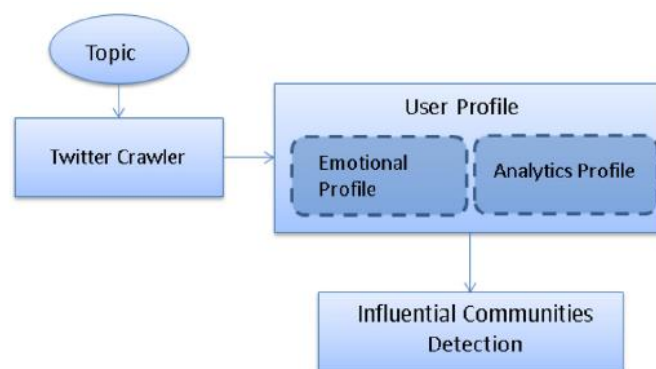
۵-۳ آشنایی با مقیاس اکمن

مقیاس Ekman، که توسط پل اکمن، روان‌شناس برجسته، توسعه یافته است، به شناسایی شش هیجان اصلی انسانی می‌پردازد: خشم، تنفر، ترس، شادی، غم، و تعجب. این مقیاس بر این فرض استوار است که این هیجان‌ها در فرهنگ‌های متفاوت سراسر جهان به صورت بنیادین وجود دارند و در بیانات چهره و زبان بدن به طور مشابهی تجلی می‌یابند.

استفاده از مقیاس اکمن تنها به مطالعات روان‌شناختی محدود نمی‌شود. در عرصه‌هایی چون بازاریابی، تحلیل رفتار مصرف‌کننده، و حتی در حوزه‌های پیشرفته‌تری مانند هوش مصنوعی و تحلیل داده‌های بزرگ، از این مقیاس بهره گرفته می‌شود. به خصوص در هوش مصنوعی، این مقیاس برای شناسایی و تحلیل احساسات در متون، صداها، و تصاویر کاربرد دارد.

۶-۳ استفاده از الگوریتم‌های تشخیص جامعه برای تحلیل رفتار عاطفی کاربران

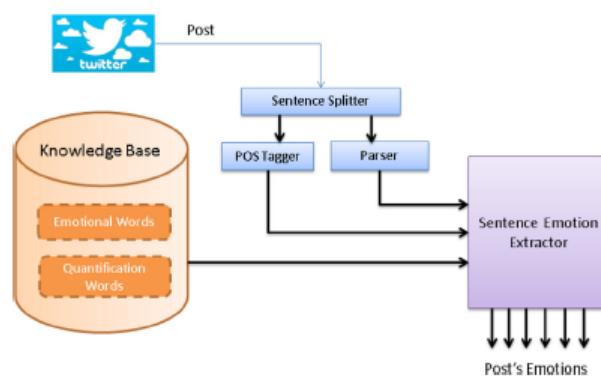
در این بخش، روش معرفی شده برای تجزیه و تحلیل و مدل‌سازی گفتگوها در موضوعات خاص در شبکه توییتر ارائه می‌شود. این روش توییت‌های کاربران را تجزیه و تحلیل کرده و محتوای احساسی آن‌ها را بر اساس مقیاس اکمن ششگانه مشخص می‌کند که شامل شش احساس انسانی اساسی: «خشم، انزجار، ترس، شادی، غم، تعجب» می‌شود. پس از آن، نفوذ کاربر در شبکه برای شناسایی جوامع تاثیرگذارتر در شبکه مشخص شده تعیین می‌گردد. جوامع تاثیرگذار استخراج شده بر اساس محتوای احساسی پست‌ها و همچنین نفوذ کاربران استفاده می‌شوند و پس از آن به عنوان نمایندگی تعاملات احساسی در شبکه مربوطه در نظر گرفته می‌شوند. معماری سیستم پیشنهادی در شکل زیر نشان داده شده است.



شکل ۱-۳ معماری سیستم

۱-۶-۳ فرآیند تشخیص احساس پست

در ابتدا، پست‌های تویتر وارد سیستم شده و سپس توسط یک جداکننده جمله به جملات مجزا تقسیم می‌شوند. هر جمله به نوبه خود توسط یک برچسب‌گذار اجزای گفتار مورد تجزیه و تحلیل قرار می‌گیرد که به هر کلمه نقش دستوری می‌دهد. سپس، توسط یک تجزیه‌کننده، ساختار دستوری جملات تجزیه شده و نمودار وابستگی ساخته می‌شود که روابط معنایی و دستوری کلمات را در جمله نشان می‌دهد. این اطلاعات همراه با دانش موجود در پایگاه دانش که شامل کلمات احساسی و کلمات کمیت‌سنجی است، به یک استخراج‌کننده احساسات جمله ارسال می‌شود که قدرت احساسی هر کلمه و وضعیت احساسی کل جمله را تعیین می‌کند. در نهایت، احساسات شناسایی شده از هر جمله جمع‌آوری شده و به صورت فهرستی از احساسات مرتبط با کل پست ارائه می‌شود.



شکل ۲-۳ معماری ابزار تشخیص احساس

۲-۶-۳ تحلیل پروفایل کاربر و سنجه تاثیرگذاری

در این بخش از تحقیق، روشی برای تخمین نفوذ یک کاربر در شبکه تویتر توضیح داده شده است. این روش که قبلاً تحت عنوان "دنبال‌کننده به دنبال شونده" معرفی شده، شبکه‌ای را تصور می‌کند که در آن کاربران تویتر به عنوان گره‌ها و ارتباطات بین آن‌ها به عنوان لبه‌ها محسوب می‌شوند. معیار نفوذ نباید تنها بر اساس تعداد "دنبال‌کنندگان" یک کاربر باشد، زیرا این اطلاعات می‌توانند به طور بالقوه دنبال‌کنندگانی باشند که محتوای کاربر را دریافت نمی‌کنند. در عوض، نسبت "دنبال‌کنندگان به دنبال شونده‌گان" (Ff ratio) می‌تواند یک شاخص مناسب‌تری باشد. معیار دیگری که می‌توان در نظر

گرفت تعداد توییت‌های یک کاربر است که نشان‌دهنده نفوذ بیشتری بر شبکه مرتبط می‌باشد. علاوه بر این، فرکانس توییت‌های کاربر که نشان‌دهنده کاربرانی است که بیشتر از دیگران توییت می‌کنند نیز مورد استفاده قرار گرفته است.

علاوه بر این، برخی ویژگی‌های اضافی که تعامل بین کاربران مختلف را در توییت‌اندازه‌گیری می‌کنند نیز در نظر گرفته شده است. به طور دقیق، تعداد بازتوییت‌ها و پاسخ‌ها نشان می‌دهد که آیا یک کاربر از محتوای دیگر کاربران لذت می‌برد و آن را بازتوییت و یا بحث می‌کند یا خیر. میزان علاقه‌مندی دنبال‌کنندگان یا دیگر کاربران به محتوای یک کاربر خاص می‌تواند از طریق تعداد محبوبیت‌ها و اشاره‌ها مشخص شود؛ این ویژگی‌ها نشان می‌دهند که ممکن است برخی از پست‌های کاربران برای برخی دیگر جالب‌تر باشد. به طور خاص، تعداد کلیک‌ها نشان‌دهنده تعداد دفعاتی است که کاربران یک پست خاص را مشاهده کرده‌اند و نشان‌دهنده نوعی از "پیوند" بین کاربران، بدون استفاده از اشاره‌ها، است. تمام معیارهای ذکر شده ترکیب شده‌اند تا فرمول زیر را تشکیل دهند که با ویژگی‌های پست‌ها سروکار دارد:

$$Post\ Impact = \frac{(Clicks + 1) * (Favorites + 1) + (Mentions + 1)}{Tweets} * \frac{(Replies + 1) + (Retweets + 1)}{Tweets}$$

در کار حاضر، آخرین توییت‌های کاربر برای محاسبه معیارهای فوق مطابق با API توییت‌ر پردازش شده‌اند. تجربیات ما نشان می‌دهد که ارزش‌های k مورد نظر بر اساس تمام ویژگی‌های ذکر شده و معیارهای ارزیابی شده است و به صورت زیر ارائه شده است:

$$Influence\ Metric = Post\ Impact \times Frequency \times \log(Ff + 1)$$

در مورد نسبت Ff ، یک نکته مهم این است که نسبت Ff در یک پایه لگاریتم ۱۰ قرار داده شده است تا از مقادیر بیش از حد جلوگیری شود. علاوه بر این، مقدار ۱ به نسبت اضافه شده است تا از معیار بودن آن به صفر در مواردی که کلیک‌ها، محبوبیت‌ها، اشاره‌ها، پاسخ‌ها یا بازتوییت‌ها صفر هستند جلوگیری شود.

اگر تعداد دنبال‌کنندگان دقیقاً برابر با تعداد دنبال‌شوندگان باشد، نسبت Ff (دنبال‌کنندگان به دنبال‌شوندگان) صفر خواهد بود. برای اجتناب از این موضوع، به نسبت کلیک‌ها، محبوبیت‌ها، اشاره‌ها،

پاسخ‌ها و بازتوییت‌ها به تعداد کل توییت‌ها، یک واحد اضافه می‌شود. این افزودن یک واحد به نسبت‌های پیشنهادی به منظور جلوگیری از صفر شدن معیار در مواقعی است که مقادیر کلیک‌ها، محبوبیت‌ها، اشاره‌ها، پاسخ‌ها یا بازتوییت‌ها صفر باشند. این رویکرد به اطمینان از آنکه معیار تأثیرگذاری همیشه یک مقدار مثبت دارد و هرگز به صفر نمی‌رسد کمک می‌کند، که برای تجزیه و تحلیل دقیق‌تر تأثیر کاربران در شبکه‌های اجتماعی حیاتی است.

۳-۶-۳ رفتار احساسی کاربر

در این بخش از تحقیق، رفتار احساسی کاربران به صورت دقیق تشریح شده است که شامل دو بخش اصلی است؛ اولین بخش به تجزیه و تحلیل وضعیت احساسی هر پست خاص می‌پردازد و دومین بخش وضعیت احساسی کلی کاربر را مشخص می‌کند.

برای بخش اول، روش تحلیل توییت‌های کاربران را در بازه زمانی سه هفته‌ای مورد بررسی قرار می‌دهد تا وضعیت احساسی هر پست را بر اساس شش احساس اساسی اکمن تعیین کند. این روش به دنبال شناسایی الگوهای احساسی در زمان‌های مختلف است و از بازه زمانی سه هفته‌ای بهره می‌برد تا تغییرات در وضعیت احساسی کاربران را به دست آورد. به علاوه، این روش از یک ابزار خاص استفاده می‌کند که توییت‌ها را به صورت احساسی تفسیر می‌کند و اگر هر توییتی پس از تجزیه و تحلیل احساسی شده باشد، وضعیت احساسی کلی کاربر بر اساس وضعیت احساسی توییت‌هایش تعیین می‌گردد.

در بخش دوم، وضعیت احساسی کلی کاربران بر اساس تفسیر احساسی هر توییت در بازه زمانی مشخص محاسبه می‌شود. این وضعیت احساسی کلی از طریق مشخص کردن اینکه آیا کاربران دارای یک پالس احساسی زنده هستند یا نه تعیین می‌گردد؛ اگر دست کم ۱۰٪ از پست‌های آن‌ها به عنوان احساسی شناخته شوند، آن‌ها دارای وضعیت حساسیتی تلقی می‌شوند، در غیر این صورت به عنوان بی‌طرف تعریف می‌گردند.

به طور کلی، وضعیت احساسی کاربران بر اساس تجزیه و تحلیل توییت‌هایشان در بازه‌ای سه هفته‌ای و با توجه به آستانه حداقل ۱۰٪ از پست‌هایشان که به عنوان احساسی شناخته می‌شوند، تعیین می‌گردد. این روش به منظور فراهم آوردن تعادل بین پست‌های احساسی و غیر احساسی کاربران در نظر گرفته شده و در موارد استثنا، مانند کاربرانی که احساسات شدیدی را نشان می‌دهند یا احساساتی

را در بیش از ۵۰٪ از توییت‌هایشان به نمایش می‌گذارند، این آستانه می‌تواند تا ۱۰۰٪ افزایش یابد. این روش به ما امکان می‌دهد تا یک تصویر دقیق و کامل از رفتار و ویژگی‌های کاربران را از طریق بررسی کل فعالیت‌های پست‌های آن‌ها فراهم آوریم.

۴-۶-۳ تشخیص جامعه

در این بخش از تحقیق، روشی برای شناسایی جوامع تأثیرگذار در شبکه کاربران تویتر شرح داده شده است. انگیزه از این کار ناشی از علاقه به شناسایی جوامع مهم در گراف کاربران تویتر است و این فرآیند با اضافه کردن یک مرحله تبدیل ترانسفورماسیون به عنوان مرحله پیش‌پردازش آغاز می‌شود. برای رسیدن به این هدف، الگوریتم بهینه‌سازی مدولاریتی که جامعه‌های تأثیرگذار را در شبکه تویتر می‌یابد، به کار گرفته شده است. این اطلاعات به صورت گراف دوگانه استخراج شده و در نظر گرفته می‌شود، که به عنوان گراف خطی شناخته می‌شود. در نهایت، الگوریتم مدولاریتی وزن‌دار برای استخراج جوامع مهم به کار برده می‌شود که با گراف اولیه مطابقت داده می‌شود.

روش به صورت جامع در مراحل زیر خلاصه شده است:

۱. تبدیل به گراف خطی، که در آن گراف خطی دوگانه یک گراف اولیه است؛ دوگانه به معنای معکوس کردن یال‌ها و گره‌ها است. تبدیل متناظر در شکل ۳ نشان داده شده است. برای دقت بیشتر، کاربران (یعنی گره‌ها) توسط بردار وضعیت احساسی توییت‌هایشان بر اساس مقیاس اکمن نمایش داده می‌شوند، که در آن مقدار ۱ به معنای حضور احساس متناظر و مقدار ۰ به معنای غیاب آن است. از سوی دیگر، یال‌ها بین این گره‌ها رابطه "دنبال کردن" را نشان می‌دهند که می‌توان مشاهده کرد آن‌ها برچسب‌های متفاوتی دارند و به گره‌های مختلف متصل می‌شوند.

۲. استفاده از الگوریتم تشخیص جامعه وزن‌دار، که یک روش بهینه‌سازی مدولاریتی است تا جوامع در شبکه تویتر شناسایی شوند.

۳. تبدیل به گراف دوگانه اولیه، که برعکس مرحله اولیه تبدیل است.

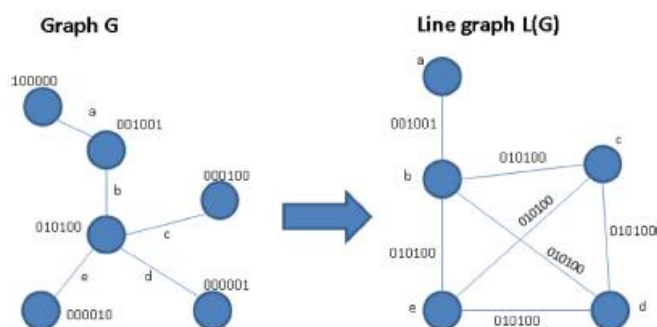


Fig. 3. Conversion to line graph.

شکل ۳-۳ تبدیل به گراف خطی

۵-۶-۳ نمایش نمودار احساسی

در این بخش از تحقیق، نمایش گرافی احساسی و وزندهی به تعاملات احساسی بین دو گره در شبکه مورد بحث قرار گرفته است. هر یک از شش احساس اکمن به وسیله یک یال احساسی بین دو نود محاسبه می شود که توسط یال متناظر نشان داده می شود. در ابتدا، هر موضوع توییت بر اساس نحوه بیان اطلاعات متناظر در توییت ها مشخص می شود. بنابراین، گراف پیشنهادی احساسات ارتباطی بین کاربران را که در قسمت موضوعی از ابزار کلی استفاده می کنند، نشان می دهد. بیشتر دقیق، گراف احساسی یک گراف جهت دار وزن دار $G = (V, E)$ است که در آن گره ها شش احساس انسانی اساسی را که توسط اکمن تعریف شده اند، نمایش می دهند. مدل خاصی برای تعریف این وزن وجود دارد که در هر یال نشان دهنده شدت این دو تعامل احساسی است. وزن به صورت زیر محاسبه می شود:

$$W_{emotion1 \rightarrow emotion2} = \frac{\sum_i^m Inf - Metric_i * Post_i}{\sum_j^k Inf - Metric_j * Post_j}$$

که در آن k تعداد پست هایی است که بر روی یک موضوع خاص با احساس $emotion1$ به لحاظ احساسی نشانه گذاری شده اند. پارامتر n تعداد پست هایی با احساس $emotion1$ را نشان می دهد که در پاسخ به پستی با احساس $emotion2$ ارسال شده اند. پست i پست احساسی را که کاربر ارسال کرده است و پارامتر Inf_Metric نفوذ کاربری را که پست را ارسال کرده است نشان می دهد. این فرمول نشان دهنده بخش وزن دار از توییت های کاربران است که احساس $emotion1$ را برای پاسخ دادن یا کامنت گذاشتن بر روی پست هایی که احساس $emotion2$ را نشان می دهند، حمل می کند.

بنابراین، مقدار نزدیک به صفر نشان می‌دهد که پست‌های نشانه‌گذاری شده با emotion2 توسط کاربرانی که پست‌هایی با emotion1 دارند، کامنت نشده‌اند. در مقابل، مقدار نزدیک به ۱۰۰ نشان می‌دهد که پست اولیه نشانه‌گذاری شده با emotion2 به طور گسترده‌ای توسط کاربرانی که emotion1 دارند، کامنت شده است.

۷-۳ نتیجه‌گیری

بر اساس مطالعه و تحلیل انجام شده در این تحقیق، می‌توان نتیجه گرفت که الگوریتم‌ها و رویکردهای پیشرفته معرفی شده در این مقاله، قادر به شناسایی و تجزیه و تحلیل دقیق جوامع موجود در شبکه‌های اجتماعی مانند توییتر هستند. با استفاده از مقیاس اکمن و الگوریتم‌های تشخیص جامعه وزن دار، ما توانستیم نه تنها جوامع متأثر از تعاملات احساسی کاربران را شناسایی کنیم، بلکه به درک عمیق‌تری از دینامیک‌های احساسی موجود در این جوامع دست یابیم. نتایج حاصل از این تحقیق نشان داد که تعاملات عاطفی کاربران می‌تواند تأثیر بسزایی در شکل‌گیری و پایداری اجتماعات داشته باشد و به ما امکان می‌دهد که با دقت بیشتری به تحلیل رفتارهای کاربران در فضای مجازی بپردازیم. این دستاوردها می‌توانند در زمینه‌های مختلفی مانند بازاریابی دیجیتال، مطالعات اجتماعی و توسعه سیاست‌های اجتماعی کاربردی و مؤثر باشند.


فصل چهارم

«پیاده‌سازی»

۴-۱ مقدمه


در فصل قبل روش های تحلیل احساس پست و پروفایل کاربر و در نهایت تشخیص جامعه بررسی شد. کامنت یا نظر بخشی از پروفایل کاربر است که می تواند حاوی اطلاعات مهمی باشد از این رو در فصل یک پروژه عملی برای تحلیل احساس کامنت ها توسعه داده شده است.

۴-۲ معرفی دیتاست استفاده شده

 Search

Snappfood - Persian Sentiment Analysis

70,000 comments with two labels



[Data Card](#)
[Code \(6\)](#)
[Discussion \(2\)](#)

About Dataset

Snappfood (an online food delivery company) user comments containing 70,000 comments with two labels (i.e. polarity classification):

- Happy
- Sad

Label	Number
Negative	35000
Positive	35000

Usability 8.24

License CC0: Public Domain

Expected update frequency Never

Tags Text NLP Text Mining

شکل ۴-۱ دیتاست کامنت های اسنپ فود

Snappfood - Persian Sentiment Analysis یک دیتاست مربوط به تحلیل احساسات به زبان فارسی است. دیتاست شامل ۷۰,۰۰۰ نظر است که با دو برچسب تقسیم شده اند که این برچسب ها نشان دهنده نوع احساس (مثبت یا منفی) در هر نظر هستند. دو برچسب اصلی داده ها Happy و Sad هستند که برای دسته بندی نظرات به نظرات مثبت و منفی به کار رفته اند. هر دو برچسب Happy و Sad دارای تعداد مساوی نظرات هستند، یعنی هر کدام ۳۵,۰۰۰ نظر، که نشان دهنده ی توازن داده ها در دیتاست است. این دیتاست امتیاز ۸,۲۴ را دارد که نشان دهنده خوانایی بالای آن است. همچنین این

دیتاست دارای مجوز CC0: Public Domain است که است نشان می‌دهد می‌توان به صورت آزاد و بدون محدودیت از آن استفاده کرد.

۴-۳ کدنویسی

در این بخش، هر بخش از نوت‌بوک گوگل کولب این پروژه، با جزئیات توضیح داده شده است.

۴-۳-۱ نصب نیازمندی‌ها و اضافه کردن کتابخانه‌های لازم

استفاده از دستور `pip install hazm` برای نصب کتابخانه hazm در محیط پایتون است. Hazm یک کتابخانه متن باز برای پردازش زبان طبیعی فارسی است. این کتابخانه امکانات متنوعی را برای کار با متن‌های فارسی فراهم می‌کند، از جمله تجزیه و تحلیل دستوری، استخراج ویژگی‌ها، و تقطیع کلمات. استفاده از Hazm برای پروژه‌هایی که نیاز به تحلیل داده‌های متنی فارسی دارند بسیار مفید است.

```
C:\Windows\System32>pip show hazm
Name: hazm
Version: 0.9.4
Summary: Persian NLP Toolkit
Home-page: https://roshan-ai.ir/hazm/
Author: Roshan
Author-email: salam@roshan-ai.com
License: MIT
Location: C:\Users\Zehra\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages
Requires: fasttext-wheel, flashtext, gensim, nltk, numpy, python-crfsuite, scikit-learn
Required-by:
```

شکل ۴-۲ نصب Hazm روی سیستم لوکال

برای مثال، اگر شما بخواهید احساسات موجود در نظرات فارسی را تحلیل کنید یا نیاز به تقسیم کردن جملات به کلمات و عبارات داشته باشید، 'Hazm' ابزار مناسبی برای این کارها است. این کتابخانه به شما کمک می‌کند تا با استفاده از تکنیک‌های پردازش زبان طبیعی، داده‌های متنی فارسی را به راحتی مدیریت و تحلیل کنید.

```
import numpy as np
import pandas as pd
import re
import matplotlib.pyplot as plt

from hazm import Normalizer, Stemmer, word_tokenize, Lemmatizer, stopwords_list
from gensim.models import Word2Vec, FastText
from sklearn.metrics import classification_report
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.ensemble import RandomForestClassifier
from sklearn.neural_network import MLPClassifier
from sklearn.linear_model import LogisticRegression
from xgboost import XGBClassifier
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import MinMaxScaler
from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
from wordcloud import WordCloud
```

شکل ۳-۴ نصب و وارد کردن کتابخانه‌های ضروری

در این بخش از کد، مجموعه‌ای از کتابخانه‌های مختلف وارد شده‌اند که هر کدام کاربرد خاصی در پروژه‌های پردازش زبان طبیعی و یادگیری ماشین دارند.

۱. numpy (np): کتابخانه‌ای برای محاسبات علمی در پایتون. این کتابخانه برای کار با آرایه‌های

بزرگ و چندبعدی مناسب است و توابع متنوعی برای انجام عملیات ریاضی و آماری دارد.

۲. pandas (pd): ابزاری برای تجزیه و تحلیل داده‌ها. این کتابخانه برای خواندن، نوشتن و مدیریت داده‌های ساختاریافته (مثل جداول) به کار می‌رود.

۳. re: کتابخانه‌ای برای کار با عبارتهای منظم (Regular Expressions). این کتابخانه امکان جستجو، ویرایش و مدیریت متن‌ها بر اساس الگوهای خاص را فراهم می‌کند.

۴. matplotlib.pyplot (plt): کتابخانه‌ای برای ترسیم نمودارها و گراف‌ها. این ابزار برای نمایش داده‌ها به صورت بصری استفاده می‌شود.

۵. hazm: این کتابخانه مجموعه‌ای از ابزارها برای پردازش زبان فارسی است، شامل تبدیل‌کننده‌ها (Normalizer, Stemmer, Lemmatizer) و ابزارهایی برای تقسیم جمله به کلمات (word_tokenize) و لیستی از کلماتی که احساس ندارند و حس خاصی به جمله نمی‌دهند (stopwords_list).

۶. gensim: کتابخانه‌ای برای مدل‌سازی موضوعات و کار با مدل‌هایی مانند Word2Vec و FastText که برای تبدیل کلمات به بردارهای عددی استفاده می‌شوند.

۷. scikit-learn: این کتابخانه شامل ابزارهای مختلف برای یادگیری ماشین است، از جمله توابعی

برای تقسیم داده ها (train_test_split)، مدل های یادگیری (MultinomialNB, RandomForestClassifier, MLPClassifier, LogisticRegression, XGBClassifier)، پایپ لاین ها (Pipeline) و ابزارهای پیش پردازش مانند MinMaxScaler و TfidfVectorizer. ۸. wordcloud: کتابخانه ای برای تولید تصاویر WordCloud که در آن ها اندازه کلمات بر اساس فراوانی ظاهر شدن شان در متن تعیین می شود.

۲-۳-۴ وارد کردن دیتاست و خواندن محتوای آن

```
!gdown --id '1HZ8HDQXnI_8A0wP9SnSwxTalEtv6ECM-'

df = pd.read_csv('/content/Snappfood - Sentiment Analysis.csv')
df.head()
```

Unnamed: 0	comment	label	label_id
0	و افعا حیف وقت که بنویسم سرویس دهیتون شده افتضاح	SAD	1
1	...قرار بود ۱ ساعته برسه ولی نیم ساعت زودتر از مو	HAPPY	0
2	...قیمت این مدل اصلا با کیفیتش سازگاری نداره، فقط	SAD	1
3	...عاللی بود همه چه درست و به اندازه و کیفیت خوب	HAPPY	0
4	شیرینی وانیلی فقط یک مدل بود	HAPPY	0

شکل ۴-۴ وارد کردن دیتاست

دستور !gdown URL در پایتون برای دانلود فایل از Google Drive با استفاده از شناسه مخصوص آن فایل استفاده می شود. gdown یک ابزار کمکی است که امکان دانلود مستقیم فایل ها از Google Drive را در محیط کدنویسی (مانند جویپتر نوت بوک یا اسکریپت پایتون) فراهم می کند. برای استفاده از این دستور، ابتدا باید gdown را نصب کنید. این می تواند با اجرای دستور !pip install gdown در محیط پایتون انجام شود. سپس، با استفاده از دستور gdown به همراه id-- و شناسه فایل مورد نظر، می توانید فایل را مستقیماً دانلود کنید. این روش برای دسترسی سریع و آسان به فایل های ذخیره شده در Google Drive در پروژه های کدنویسی بسیار مفید است. با استفاده از دستور df = pd.read_csv(URL)، داده های موجود در فایل CSV با نام

Snappfood - Sentiment Analysis.csv توسط کتابخانه pandas خوانده می‌شوند. این فایل حاوی داده‌هایی است که برای تحلیل احساسات در نظرات مشتریان اسنپ‌فود استفاده می‌شوند. با استفاده از دستور `df.head()`، پنج رکورد اول از دیتافریم `df` (که حاوی داده‌های خوانده شده از فایل CSV است) نمایش داده می‌شوند. این کار کمک می‌کند تا یک نگاه کلی به ساختار داده‌ها و نوع اطلاعاتی که در دیتافریم وجود دارند، داشته باشیم.

```
df.shape

(70000, 4)

df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 70000 entries, 0 to 69999
Data columns (total 4 columns):
#   Column      Non-Null Count  Dtype  
---  -
0   Unnamed: 0    1 non-null      object  
1   comment      70000 non-null  object  
2   label        70000 non-null  object  
3   label_id     70000 non-null  int64   
dtypes: int64(1), object(3)
memory usage: 2.1+ MB
```

شکل ۴-۵ بررسی ساختار دیتافریم

توسط این کدها، ساختار DataFrame را بررسی می‌کنیم. دستور `df.shape` ابعاد DataFrame را نشان می‌دهد، که در این دیتاست نشان دهنده‌ی وجود ۷۰,۰۰۰ ردیف و ۴ ستون است. دیتافریم چهار ویژگی یا مشخصه دارد و داده‌هایی برای ۷۰,۰۰۰ مورد نظرات کاربران را در خود جای داده است. دستور `df.info` یک خلاصه‌ای از DataFrame را فراهم می‌کند، از جمله اطلاعات در مورد هر ستون، تعداد داده‌های غیر تهی (non-null) در هر ستون، و نوع داده‌های هر ستون (Dtype). بر اساس این خروجی، DataFrame ما دارای چهار ستون است:

- ستون اول Unnamed نام دارم.
- ستون دوم با نام comment که شامل ۷۰,۰۰۰ مقدار غیر تهی (non-null) است و از نوع object است، که معمولاً به متن یا مقادیر مخلوط در pandas اشاره دارد.

- ستون سوم با نام label دارای ۷۰,۰۰۰ مقدار غیر تهی است و از نوع object است، که برچسب‌های مربوط به حالت احساسی نظرات را نشان می‌دهد.
- ستون چهارم با نام label_id دارای ۷۰,۰۰۰ مقدار غیر تهی و از نوع int64 است، که نشان دهنده‌ی یک شناسه عددی برای برچسب‌های احساسی است.

```
df.describe(include='all')
```

	Unnamed: 0	comment	label	label_id
count	1	70000	70000	70000.000000
unique	1	70000	2	NaN
top	۳۰! ساعت طول کشید تا برسه:	واقعا حیف وقت که بنویسم سرویس دهیتون شده افتضاح	SAD	NaN
freq	1	1	35000	NaN
mean	NaN	NaN	NaN	0.500000
std	NaN	NaN	NaN	0.500004
min	NaN	NaN	NaN	0.000000
25%	NaN	NaN	NaN	0.000000
50%	NaN	NaN	NaN	0.500000
75%	NaN	NaN	NaN	1.000000
max	NaN	NaN	NaN	1.000000

شکل ۴-۶ نمایش آمار توصیفی دیتاست

- دستور `df.describe(include='all')` به منظور نمایش آمار توصیفی از همه ستون‌های DataFrame به کار می‌رود، از جمله ستون‌های عددی و غیر عددی.
- ستون Unnamed: 0: این ستون فقط یک مقدار غیر تهی دارد.
 - ستون comment: دارای ۷۰,۰۰۰ مقدار غیر تهی است که همه منحصر به فرد هستند. این نشان می‌دهد که هر نظر در دیتاست منحصر به فرد است. ستون top یک نمونه نظر را نشان می‌دهد که بیشترین بار تکرار شده است، اما از آنجا که تکرار freq تنها ۱ است، می‌توان فهمید که این نمونه نظر منحصر به فرد است.
 - ستون label: دارای دو مقدار منحصر به فرد (unique: 2) است که احتمالاً نشان دهنده‌ی دو نوع برچسب (مثبت و منفی) است. مقدار top نشان می‌دهد که برچسب SAD بیشترین تکرار را دارد،

ولی از آنجایی که تعداد freq برابر با ۳۵,۰۰۰ است، این نشان می‌دهد که برچسب‌ها به طور مساوی تقسیم شده‌اند.

- ستون label_id: این ستون شامل اعداد صحیح است که با توجه به میانگین mean: 0.5 و انحراف معیار std: 0.500004 می‌توان گفت که اعداد صحیح برای نمایش دو برچسب مختلف به کار رفته‌اند. مقادیر کمینه min: 0.0 و بیشینه max: 1.0 این گفته را تایید می‌کنند.

```
df.nunique()

Unnamed: 0      1
comment      70000
label          2
label_id       2
dtype: int64
```

شکل ۷-۴ بررسی مقادیر یکتای دیتاست

دستور df.nunique() در محیط پایتون، تعداد مقادیر منحصر به فرد در هر ستون از DataFrame را نشان می‌دهد. این تابع می‌تواند برای ارزیابی تنوع داده‌ها و انتخاب روش‌های مناسب برای پیش‌پردازش و تحلیل داده مفید باشد.

۳-۳-۴ پیش‌پردازش

انجام پیش‌پردازش و تمیزسازی داده‌ها یک مرحله مهم در فرآیند یادگیری ماشین است، زیرا کیفیت و دقت مدل‌های آموزش داده شده به طور مستقیم به کیفیت داده‌های ورودی وابسته است. پیش‌پردازش شامل حذف ویژگی‌های اضافه، حذف یا جایگزینی داده‌های گمشده، تبدیل داده‌ها به فرمت مناسب برای مدل‌سازی، نرمال‌سازی و استانداردسازی، و حذف نویز از داده‌ها می‌شود. این اقدامات به حذف تاثیر داده‌های پرت یا غیر مرتبط کمک کرده و اطمینان می‌دهند که مدل بر روی الگوهای واقعی در داده‌ها تمرکز کند و نه بر روی خطاها یا استثنائات. بدون پیش‌پردازش مناسب، مدل‌های یادگیری ماشین ممکن است دچار overfitting شده یا الگوهای نادرستی را یاد بگیرند، که به نتایج نامطلوب و عملکرد ضعیف در هنگام برخورد با داده‌های جدید منجر می‌شود.

```
df = df[['comment' , 'label' , 'label_id']]
df.dropna(inplace=True)
df.head()
```

	comment	label	label_id
0	واقعا حیف وقت که بنویسم سرویس دهیتون شده افتضاح	SAD	1
1	...قرار بود ۱ ساعته برسه ولی نیم ساعت زودتر از مو	HAPPY	0
2	...قیمت این مدل اصلا با کیفیتش سازگاری نداره، فقط	SAD	1
3	...عاللی بود همه چه درست و به اندازه و کیفیت خوب	HAPPY	0
4	شیرینی وانیلی فقط یک مدل بود	HAPPY	0

شکل ۸-۴ حذف ستون Unnamed

در این قسمت از پیش پردازش به دلیل مفید نبودن ستون Unnamed، سه ستون comment، label و label_id از DataFrame اصلی انتخاب می‌شوند و DataFrame جدیدی با این سه ستون ایجاد می‌شود. یعنی به نوعی این سه ستون را حذف می‌کنیم.

سپس، دستور df.dropna(inplace=True) فراخوانی می‌شود که هر ردیفی را که شامل مقادیر تهی (NaN) است، حذف می‌کند. پارامتر inplace=True باعث می‌شود که تغییرات مستقیماً روی DataFrame اعمال شود بدون اینکه نیاز به اختصاص دادن تغییرات به یک متغیر جدید باشد.

در نهایت، df.head() فراخوانی می‌شود که پنج رکورد اول از DataFrame را برای بررسی نشان می‌دهد. این کار به ما اجازه می‌دهد تا محتوای DataFrame پس از اعمال تغییرات بالا را بررسی کنیم.

```
df.query('label == "HAPPY" and label_id == 1.0')
```

```
comment label label_id
```

```
df.query('label == "SAD" and label_id == 0')
```

```
comment label label_id
```

شکل ۹-۴ فیلتر کردن سطرها

از تابع `df.query()` در کتابخانه `pandas` پایتون برای فیلتر کردن سطرها از یک `DataFrame` بر اساس شرایط خاص استفاده می‌شود و در اینجا برای چک ناسازگاری در برچسب‌گذاری استفاده شده است.

هدف از این کوئری‌ها بررسی صحت داده‌ها و اطمینان از این است که برچسب‌ها به درستی به `label_id` مربوطه اختصاص داده شده‌اند. اگر نتایجی از اولین `query` برگردانده شوند، نشان دهنده ناهماهنگی در مجموعه داده است که باید برطرف شود، چرا که نظرات `HAPPY` نباید `label_id 1.0` داشته باشند. دومین `query` برای بررسی تطابق نظرات `SAD` با `label_id 0` است، که اگر برچسب‌ها درست باشند نباید استثناء‌ای بازگردانده شود.

```
df['label_id'] = df['label_id'].astype(int)
df.head()
```

	comment	label	label_id
0	واقعا حیف وقت که بنویسم سرویس دهیتون شده افتضاح	SAD	1
1	...قرار بود ۱ ساعته برسه ولی نیم ساعت زودتر از مو	HAPPY	0
2	...قیمت این مدل اصلا با کیفیتش سازگاری نداره، فقط	SAD	1
3	...حالتی بود همه چه درست و به اندازه و کیفیت خوب	HAPPY	0
4	شیرینی وانیلی فقط یک مدل بود	HAPPY	0

شکل ۱۰-۴ تبدیل نوع داده

با این کد ستون `label_id` از `DataFrame` به نوع داده‌ای عدد صحیح تبدیل شده و سپس پنج رکورد اول `DataFrame` نمایش داده می‌شوند. این تغییر نوع داده برای آماده‌سازی داده‌ها جهت استفاده در الگوریتم‌های یادگیری ماشین که نیاز به ورودی‌های عددی دارند، صورت گرفته.

در ادامه داده‌ها به دو بخش تقسیم می‌شوند. X که حاوی نظرات است و y که شامل برچسب‌ها است. این کار قبل از آموزش مدل‌های یادگیری ماشین انجام می‌شود تا مدل هنگام یادگیری برچسب‌ها را نبیند و در ادامه بیش‌برازش اتفاق نیافتد. $X.shape$ نشان می‌دهد بعد از جداسازی متغیر X دارای ۷۰,۰۰۰ سطر است.

سپس، علائم نگارشی هم در زبان انگلیسی و هم در فارسی، به همراه حروف الفبای انگلیسی، از متن حذف می‌شوند. هدف این است که تنها متن فارسی باقی بماند و هیچ عنصر دیگری در داده‌ها نباشد. این کار با استفاده از یک تابع `lambda` و متد `re.sub()` انجام می‌شود که یک عبارت منظم (regular expression) را برای شناسایی و حذف کاراکترهای مورد نظر استفاده می‌کند. حذف علائم نگارشی و حروف غیرضروری به کاهش نویز و تمرکز بر محتوای مفید متن کمک کند.

توابع لامبدا در پایتون، که اغلب به عنوان توابع ناشناس شناخته می‌شوند، یک روش مختصر برای تعریف توابع کوچک هستند. این توابع بی‌نام بوده و تنها شامل یک عبارت هستند، که نتیجه آن بلافاصله برگردانده می‌شود. توابع لامبدا معمولاً در مواردی استفاده می‌شوند که نیاز به یک تابع موقتی برای انجام عملیات‌های کوچک است، مانند اعمال یک تابع بر روی هر عنصر یک لیست یا هر سطر یا ستونی در یک `DataFrame`. به دلیل ساختار مختصر و قابلیت استفاده مجدد، استفاده از توابع لامبدا می‌تواند به کدنویسی تمیزتر و خوانایی بالاتر کمک کند.

```
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
```

۲۴

تابع `train_test_split` از کتابخانه `sklearn.model_selection` برای تقسیم داده ها به دو مجموعه آموزشی و تست استفاده می شود. در اینجا داده ها به دو بخش تقسیم شده اند: ۸۰٪ برای آموزش و ۲۰٪ برای تست، که با پارامتر `test_size=0.2` تعیین شده است. چهار متغیر `y_train`, `X_train`, `X_test` و `y_test` به ترتیب شامل ویژگی های آموزشی، ویژگی های تست، برچسب های آموزشی و برچسب های تست هستند. این تقسیم بندی امکان ارزیابی عملکرد مدل یادگیری ماشین را بر روی داده هایی که در فرآیند آموزش دیده نشده اند، فراهم می کند و یک اقدام استاندارد در توسعه مدل های پیش بینی است.

```
normalizer = Normalizer()
X_train = X_train.apply(lambda x: normalizer.normalize(x))
X_test = X_test.apply(lambda x: normalizer.normalize(x))

X_train.head(10)
```

47734	...سلام هزار تومن خرید کردم حیثون اومده بود تا س
9114	...شیرینی های مکرکی رو من نیم کلو برای مهمان سف
12549	...خیلی عالی بود سفارش من یک ربع طول کشید آوردن خ
10475	...همه روغنش ریخته بود و طعمش اصلا خوب نبود ولی ا
60989	...واقعا تا بحال اینقدر بوی زوخم مرغو تجربه نکرده
49917	...بیتزای بیرونی عالی بود اگر فلفل هالوپن هم اضاف
22952	...با سلام با وجود ثبت توضیحات در خصوص نریختن پنی
31188	...فاصله با رستوران حدود متریه ولی یک ساعت و چهل و
54998	...سیگار و اشتباه ارسال کرده بودند
16466	...من کمتر نظر میدم اما واقعا عالی بود اگر نگم به

Name: comment, dtype: object

شکل ۱۳-۴ نرمال سازی داده

در ادامه از کتابخانه `hazm` برای نرمال سازی داده های متنی فارسی استفاده می شود. `hazm` یک کتابخانه پردازش زبان طبیعی برای زبان فارسی است که امکانات مختلفی مانند نرمال سازی، توکن سازی، استخراج ویژگی ها، و تجزیه نحوی را ارائه می دهد. در این کد، ابتدا یک شیء `Normalizer` ایجاد می شود. سپس، این نرمال ساز برای پیش پردازش و نرمال سازی متن نظرات در هر دو مجموعه داده های آموزشی و تست استفاده می شود. عملیات نرمال سازی به وسیله تابع `apply()` و یک تابع `lambda` که متد `normalize` را برای هر متن فراخوانی می کند، اجرا می شود. در نهایت با نمایش ده سطر اول از `X_train` بعد از اعمال نرمال سازی، نتیجه نرمال سازی را بررسی می کنیم.

نرمال سازی متن ممکن است شامل مواردی مانند حذف فاصله های اضافی، تبدیل حروف به شکل استاندارد و یکنواخت سازی استفاده از حروف کوچک و بزرگ باشد. این مرحله برای کاهش تنوع در نحوه نوشتاری کلمات و بهبود کارایی مدل های یادگیری ماشین بسیار مهم است.

```

X_train = X_train.apply(lambda x: word_tokenize(x))
X_test = X_test.apply(lambda x: word_tokenize(x))

X_test.head()

```

67753 [شیلر، متل، همیشه، عالی]
26768 ...متل، همیشه، عالی، بود، ممنون، از، مدیریت، مج
27107 ...و تغییر، در، سفارش، ارسالی، به، ناچار، ...بدلیل
27819 ...غذایش، عالی، بود، داغ، بود، خوشمزه، بود، و، ب
27744 ...لایک، دارید، انصافاً، ...، عالی، بود، ...، یکم، ق
Name: comment, dtype: object

شکل ۱۴-۴ توکن سازی کلمات

از تابع `word_tokenize` کتابخانه `hazm` برای توکن سازی کلمات در داده های متنی استفاده می شود. توکن سازی یکی از مراحل اولیه در پردازش زبان طبیعی (NLP) است و شامل تقسیم کردن متن به واحدهای کوچک تری مانند کلمات یا عبارات می شود. این فرآیند به مدل اجازه می دهد تا بتواند اطلاعات متنی را به طور موثرتری پردازش کند.

در اینجا متن های موجود در ستون `comment` از هر دو مجموعه داده های آموزشی و تست به وسیله تابع `apply` و یک تابع `lambda` که `word_tokenize` را فراخوانی می کند، به لیستی از کلمات تبدیل شده اند. نتیجه این است که هر ردیف از ستون `comment` اکنون شامل یک لیست از کلمات است به جای یک رشته متنی یکپارچه.

```

stemmer = Stemmer()
X_train = X_train.apply(lambda words: [stemmer.stem(word) for word in words])
X_test = X_test.apply(lambda words: [stemmer.stem(word) for word in words])

X_test.head()

```

67753 [شیلر، متل، همیشه، عالی]
26768 ...متل، همیشه، عال، بود، ممنون، از، مدیر، مجموعه
27107 ...تغییر، در، سفار، ارسال، به، ناچار، ...بدلیل، تم
27819 ...غذا، عال، بود، داغ، بود، خوشمزه، بود، و، بسته
27744 ...لایک، دارید، انصافاً، ...، عال، بود، ...، یکم، قارچ
Name: comment, dtype: object

شکل ۱۵-۴ ریشه یابی کلمات

تابع `Stemmer` در کتابخانه `hazm` برای ریشه یابی (Stemming) فرآیندی است که در آن کلمات به شکل ریشه یا فرم اصلی خود که معمولاً نقش دستوری دارند (مانند فعل یا اسم) تبدیل می شوند. این کار به کاهش تنوع و تبدیل کلمات مختلف با همان ریشه به یک شکل استاندارد کمک می کند تا مدل های یادگیری ماشین بهتر بتوانند اطلاعات را یاد بگیرند و تعمیم دهند.

ابتدا یک شیء `Stemmer` ایجاد شده است و سپس این ریشه یاب برای پردازش لیست های کلمات در هر سطر از ستون های `comment` در `X_train` و `X_test` با استفاده از تابع `apply` و یک تابع `lambda` به کار رفته است. در نتیجه هر کلمه در لیست به ریشه خود تبدیل شده است.

```

lemmatizer = Lemmatizer()
X_train = X_train.apply(lambda words: [lemmatizer.lemmatize(word) for word in words])
X_test = X_test.apply(lambda words: [lemmatizer.lemmatize(word) for word in words])

X_train.head()

```

47734	...سلار هزارو نومن خریدو کرد##کنو حیفشون و اومده
9114	...شیرینی، مکزیک، روز من، #هست، کیلو، برا، مه، س
12549	...خیل، حل، بود##ش، سفار، من، یکریم، طول، کشید
10475	...همه، روغن، ریخت##ریزو، و، طعم، اصلا، خوب، بود##ب
60989	...واقعا، نار، بحال، اینقر، بور، زوخ، مرغور، تجربه

Name: comment, dtype: object

شکل ۱۶-۴ واژه یابی

تابع Lemmatizer در کتابخانه hazm برای انجام واژه یابی (lemmatization) استفاده می شود. واژه یابی فرآیندی است که در آن کلمات به شکل دیکشنری یا واژه ی اصلی خود تبدیل می شوند، که می تواند به شکل جمع یا زمان های مختلف فعل باشد. این تکنیک اغلب نسبت به ریشه یابی (stemming) دقیق تر است زیرا در تلاش است تا واژه ها را به شکلی که در زبان مورد استفاده قرار می گیرند، تبدیل کند، و نه فقط به یک پایه خام. این مرحله پیش پردازش برای کاهش تنوع کلمات و تمرکز بر معنای اصلی آن ها به منظور بهبود عملکرد مدل مفید است.

```

X_train = X_train.apply(lambda x: ' '.join(x)).to_list()
X_test = X_test.apply(lambda x: ' '.join(x)).to_list()

```

شکل ۱۷-۴ تبدیل کلمات به رشته متنی

در آخرین مرحله پیش پردازش لیست کلمات پردازش شده در مراحل قبلی را به رشته های متنی تبدیل می کنیم. با استفاده از تابع lambda هر لیست از کلمات (x) را به یک جمله با استفاده از متد join() تبدیل کنیم. این متد کلمات را با فاصله (space character) که در اینجا به عنوان آرگومان به join() داده شده به هم متصل می کند و جملاتی را ایجاد می کند که شبیه به جملات اصلی هستند. این کار برای مدل هایی که نیاز به ورودی متنی پیوسته دارند، مانند برخی مدل های یادگیری ماشین برای تحلیل احساسات، مفید است. با تبدیل لیست کلمات به جملات، می توان ورودی هایی را که بیشتر شبیه به داده های واقعی هستند به مدل ها ارائه کرد.

۴-۳-۴ بردارسازی متن

```
vectorizer = TfidfVectorizer()
x_train = vectorizer.fit_transform(X_train)
x_test = vectorizer.transform(X_test)
```

شکل ۴-۱۸ بردارسازی متن

قبل از دادن داده‌ها به مدل باید آنها را به شکل بردار درآورد. برای این کار از یک بردارسازی متنی، به نام `TfidfVectorizer` از کتابخانه `scikit-learn` استفاده شده است. این بردارساز برای تبدیل متن‌های خام به ماتریسی از ویژگی‌های `TF-IDF` استفاده می‌شود. `TF-IDF` مخفف `Term Frequency-Inverse Document Frequency` است و به کلمات متن بر اساس اهمیت آن‌ها وزن‌دهی می‌کند.

ابتدا یک شی از `TfidfVectorizer` ایجاد شده است. سپس، این بردارساز با استفاده از داده‌های آموزشی آموزش دیده و سپس داده‌های آموزشی را به بردارهای ویژگی `TF-IDF` تبدیل کرده است. این کار با استفاده از متد `fit_transform` انجام شده.

برای داده‌های تست، فقط تبدیل انجام می‌شود چون مدل باید از واژگانی که قبلاً در مرحله آموزش دیده شده‌اند استفاده کند. این کار با استفاده از متد `transform` انجام می‌شود.

این گام ضروری است تا مدل‌های یادگیری ماشین بتوانند متن‌ها را به عنوان ورودی قابل فهمی در قالب اعداد به جای متن خام دریافت کنند. استفاده از `TF-IDF` به مدل کمک می‌کند تا بر کلماتی که اطلاعات بیشتری نسبت به سایر کلمات دارند تمرکز کند و از ویژگی‌هایی که به طور مکرر در تمام متون تکرار می‌شوند و اطلاعات کمتری دارند، چشم‌پوشی کند.

در یادگیری ماشین و پردازش داده‌ها، `fit_transform` و `transform` دو متد مرتبط با تبدیل داده‌ها هستند که در کتابخانه‌های مانند `scikit-learn` استفاده می‌شوند. `fit_transform` دو عملیات `fit` و `transform` را به طور همزمان انجام می‌دهد. متد `fit` الگوهای کلیدی (مانند میانگین، انحراف استاندارد، حداکثر، و غیره) را از داده‌های آموزشی یاد می‌گیرد. به عبارت دیگر، متد `fit` برای محاسبه و ذخیره پارامترهایی است که برای تبدیل داده‌ها لازم هستند. پس از یادگیری الگوها، `transform` داده‌ها را مطابق با پارامترهای یادگرفته شده تبدیل می‌کند. `fit_transform` برای داده‌های آموزشی استفاده می‌شود تا هم الگوها را یاد بگیرد و هم داده‌های آموزشی را تبدیل کند.

متد `transform` فقط عملیات تبدیل را با استفاده از پارامترهایی که از قبل توسط `fit` یاد گرفته شده‌اند، انجام می‌دهد. `transform` برای داده‌های تست (یا هر داده‌ای جدید) استفاده می‌شود و این اطمینان را می‌دهد که داده‌های تست مطابق با همان الگوها و پارامترهایی که از داده‌های آموزشی

یاد گرفته شده اند، تبدیل می شوند و هیچ گونه اطلاعاتی از داده های تست در محاسبه پارامترها استفاده نشده است.

به طور خلاصه، `fit_transform` را برای آموزش مدل بر روی داده های آموزشی و سپس `transform` را برای تبدیل داده های تست یا جدید بدون دگرگونی پارامترهای تبدیل استفاده می شوند. این امر به حفظ یکپارچگی مدل و جلوگیری از نشت اطلاعات کمک می کند، که می تواند منجر به ارزیابی نادرست عملکرد مدل شود.

۵-۳-۴ ساخت و ارزیابی مدل

```
rfc = RandomForestClassifier()
rfc.fit(x_train, y_train)
y_pred = rfc.predict(x_test)
print(classification_report(y_test, y_pred))
```

	precision	recall	f1-score	support
0	0.88	0.79	0.83	6993
1	0.81	0.89	0.85	7007
accuracy			0.84	14000
macro avg	0.84	0.84	0.84	14000
weighted avg	0.84	0.84	0.84	14000

شکل ۴-۱۹ مدل های ساخته شده

```
xgb = XGBClassifier()
xgb.fit(x_train, y_train)
y_pred = xgb.predict(x_test)
print(classification_report(y_test, y_pred))
```

	precision	recall	f1-score	support
0	0.88	0.79	0.83	6993
1	0.81	0.90	0.85	7007
accuracy			0.84	14000
macro avg	0.84	0.84	0.84	14000
weighted avg	0.84	0.84	0.84	14000

شکل ۴-۲۰ مدل های ساخته شده

```
mnb = MultinomialNB()
mnb.fit(x_train, y_train)
y_pred = mnb.predict(x_test)
print(classification_report(y_test, y_pred))
```

	precision	recall	f1-score	support
0	0.88	0.76	0.81	6993
1	0.79	0.90	0.84	7007
accuracy			0.83	14000
macro avg	0.83	0.83	0.83	14000
weighted avg	0.83	0.83	0.83	14000

شکل ۴-۲۱ مدل های ساخته شده

```
lr = LogisticRegression(max_iter=1000)
lr.fit(x_train, y_train)
y_pred = lr.predict(x_test)
print(classification_report(y_test, y_pred))
```

	precision	recall	f1-score	support
0	0.88	0.81	0.84	6993
1	0.83	0.89	0.86	7007
accuracy			0.85	14000
macro avg	0.85	0.85	0.85	14000
weighted avg	0.85	0.85	0.85	14000

شکل ۴-۲۲ مدل های ساخته شده

در این بخش از الگوریتم های مختلف مدل سازی شده و داده هایی که در مرحله پیش پردازش آماده شده بودند، به عنوان ورودی به مدل ها داده شده. در نهایت گزارشی از نحوه عملکرد مدل چاپ شده که شامل اطلاعات بسیار مفیدی است و با استفاده از اطلاعات بهترین مدل انتخاب شد. این گزارش شامل چندین متریک استاندارد است که هر کدام بخشی از عملکرد مدل را در تشخیص کلاس های مختلف ارزیابی می کنند:

۱. دقت (Precision): این متریک نشان می دهد که از تمام پیش بینی های مثبت، چند درصد واقعاً مثبت هستند. دقت بالا نشان دهنده این است که تعداد کمی از مثبت های کاذب (false positives) وجود دارد.

۲. بازیابی (Recall): بازیابی نشان می دهد که از تمام موارد واقعاً مثبت، چند درصد توسط مدل به درستی به عنوان مثبت شناسایی شده اند. بازیابی بالا نشان می دهد که مدل بیشتر موارد مثبت را پوشش می دهد و تعداد کمی از منفی های کاذب (false negatives) دارد.

۳. نمره F1 (F1-Score): این متریک میانگین هارمونیک دقت و بازیابی است. نمره F1 در مواقعی که توازن بین دقت و بازیابی مهم است، مفید است. این متریک برای مواردی که توزیع کلاس ها

- متوازن نیست یا هزینه های متفاوتی برای خطاهای مختلف وجود دارد، بسیار مهم است.
۴. پشتیبانی (Support): این عدد نشان می دهد که چه تعداد نمونه واقعی برای هر کلاس در مجموعه داده های تست وجود دارد. این متریک برای تعیین نمایندگی هر کلاس در داده های تست مفید است.
۵. دقت کلی (Accuracy): این متریک نشان دهنده کلی ترین ارزیابی از عملکرد مدل است و درصد کلی نمونه هایی را که به درستی طبقه بندی شده اند نشان می دهد.
۶. میانگین های کلی (Macro Avg, Weighted Avg): این ارزیابی ها میانگین گیری متریک ها را بر اساس وزن دهی متفاوت انجام می دهند. میانگین ماکرو بدون توجه به تعداد نمونه های هر کلاس میانگین گیری می کند، در حالی که میانگین وزن دار بر اساس تعداد نمونه ها در هر کلاس وزن دهی می کند.
- این متریک ها با هم استفاده می شوند تا یک تصویر کامل و متعادل از عملکرد مدل فراهم کنند، به ویژه در مواردی که ممکن است بعضی کلاس ها بیش از بقیه نمایندگی شده باشند یا برخی از انواع خطاها مهم تر از دیگران باشند.
- Random Forest یک الگوریتم یادگیری نظارتی است که به طبقه بندی های درخت تصمیم (Decision Trees) متکی است. این مدل با ایجاد مجموعه ای از درختان تصمیم گیری در هنگام آموزش کار می کند، که هر کدام روی زیرمجموعه ای تصادفی از داده ها و ویژگی ها آموزش می بینند. پیش بینی نهایی از طریق رأی گیری اکثریت در میان درختان انجام می شود. ویژگی های برجسته RFC عبارتند از کاهش بیش برآزش (overfitting) نسبت به درختان تصمیم مجزا و قدرت بالا در مدل سازی مسائل پیچیده.
- XGBoost، که مخفف eXtreme Gradient Boosting است، یک پیاده سازی بهینه شده از الگوریتم گرادیان تقویتی است. این مدل عملکرد برجسته ای در مسابقات داده کاوی به خود گرفته و به دلیل سرعت بالا، قابلیت مقیاس پذیری و بهبود عملکرد به شهرت رسیده است. XGB با استفاده از تکنیک های پیچیده مانند Pruning (هرس کردن) درختان، بهینه سازی موازی و روش های منظم سازی برای کاهش بیش برآزش، قادر است مدل های دقیق و کارآمدی را تولید کند.



شکل ۲۳-۴ ویژگی های XGBoost

Multinomial Naive Bayes یک الگوریتم طبقه بندی احتمالاتی است که بر اساس قضیه بیز و با فرض استقلال شرطی بیز ساده بین ویژگی ها کار می کند. این مدل به ویژه در زمینه هایی مانند تحلیل احساسات و دسته بندی متن کاربرد دارد و برای داده هایی با توزیع های چندجمله ای مناسب است. MNB برای مسائلی که تعداد دفعات وقوع ویژگی ها (مانند کلمات در یک متن) مهم است، موثر عمل می کند.

Logistic Regression یک الگوریتم طبقه بندی است که برای پیش بینی احتمال وقوع یک رویداد بر اساس یک یا چند متغیر مستقل استفاده می شود. این مدل یک مورد خاص از مدل های خطی است که از تابع لجستیک برای مدل سازی احتمال ها استفاده می کند و می تواند احتمال ها را به دو کلاس ۰ یا ۱ تقسیم کند. Logistic Regression به دلیل سادگی، شفافیت مدل و قابلیت تفسیر پذیری بالا، در بین پژوهشگران و صنعتگران محبوبیت دارد.

```
clf_pipeline = Pipeline([
    ('tfidf', vectorizer),
    ('xgb', xgb)
])

y_pred = clf_pipeline.predict(X_test)

print(classification_report(y_test, y_pred))
```

	precision	recall	f1-score	support
0	0.88	0.79	0.83	6993
1	0.81	0.90	0.85	7007
accuracy			0.84	14000
macro avg	0.84	0.84	0.84	14000
weighted avg	0.84	0.84	0.84	14000

```
clf.predict(['!!حیف پولییی که به این غذا دادم'])

array([1])
```

شکل ۲۴-۴ ساخت Pipeline

در آخرین مرحله مدلسازی با استفاده از الگوریتم XGBoost که یکی از بهترین دقت ها را داشت، یک Pipeline تعریف شده است. از پایپلاین ها به منظور ساده سازی کد و اطمینان از اینکه همان مراحل پیش پردازش و طبقه بندی به طور متوالی بر روی داده های جدید اعمال شود، استفاده می شود. هم TfIdfVectorizer و هم مدل (XGBClassifier) باید به Pipeline داده شود تا خط لوله به درستی عمل کند. در پایان، یک نمونه متن به مدل داده شده است تا کلاس آن پیش بینی شود و مدل هم به درستی پیش بینی را انجام داده است.

۶-۳-۴ مصورسازی

```

persian_stopwords = stopwords_list()

font_path = '/content/Peyda-Black.ttf'

plt.figure(figsize=(20, 10))

unique_sentiments = df['label'].unique()
num_unique_sentiments = len(unique_sentiments)
rows = num_unique_sentiments // 2 + num_unique_sentiments % 2
cols = 2 if num_unique_sentiments > 1 else 1

for index, label in enumerate(unique_sentiments):
    plt.subplot(rows, cols, index + 1)
    # Filter the dataframe by sentiment
    df_filtered = df[df['label'] == label]
    # Join all the text items in a single string
    text_data = " ".join(comment for comment in df_filtered['comment'])

    wordcloud = WordCloud(
        background_color='white',
        font_path=font_path,
        stopwords=persian_stopwords,
        width=800,
        height=400,
        max_words=50,
        max_font_size=100,
        scale=5
    ).generate(text_data)

    # Disable axis ticks
    plt.xticks([])
    plt.yticks([])
    plt.imshow(wordcloud, interpolation='bilinear')
    plt.title(label, fontsize=20)

plt.tight_layout()
plt.show()

```

شکل ۴-۲۵ کدهای مصورسازی

برای مصورسازی این پروژه از تکنیک ابر کلمات (word clouds) استفاده شده است. این تکنیک یک روش بصری برای نشان دادن کلمات بیشتر استفاده شده در یک مجموعه متن است، که کلمات بزرگ‌تر نشان‌دهنده فراوانی بیشتر آن‌ها هستند. این روش معمولاً برای درک بهتر و تحلیل بصری داده‌های متنی و بررسی تفاوت‌های احساسی بین دسته‌های مختلف استفاده می‌شود.

از کتابخانه‌های matplotlib برای نمایش و wordcloud برای ایجاد ابر کلمات استفاده شده است. برای نمایش صحیح کاراکترهای فارسی در ابر کلمات باید فونت فارسی به پروژه اضافه و آدرس آن در کد آورده شود.



شکل ۴-۲۶ نتیجه مصورسازی

۷-۳-۴ ذخیره سازی مدل آموزش داده شده

```
import pickle

with open('xgb.pkl', 'wb') as file:
    pickle.dump(xgb, file)

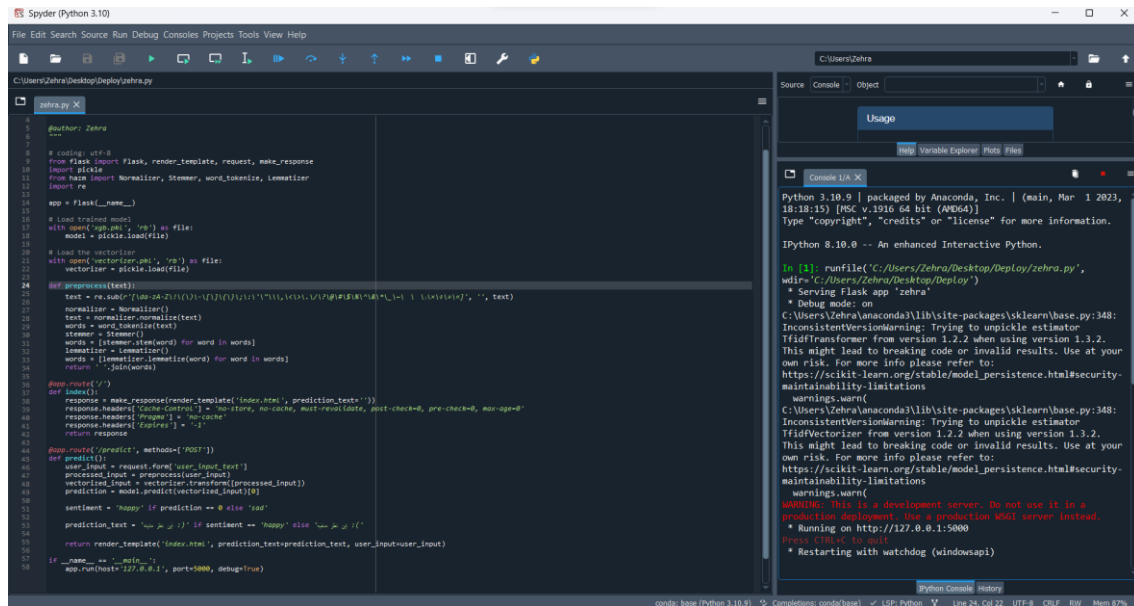
with open('vectorizer.pkl', 'wb') as file:
    pickle.dump(vectorizer, file)
```

شکل ۴-۲۷ ساخت فایل pickle

در نهایت برای ذخیره سازی مدل و Deploy کردن آن از کتابخانه pickle در پایتون استفاده شده است که اجازه می دهد اشیاء پایتون را به فرمت فایل باینری ذخیره کنیم. این شیوه اجازه می دهد که مدل ها و بردار سازها را بدون نیاز به آموزش مجدد در آینده، بتوان مجدداً بارگذاری و استفاده کرد. این رویکرد زمان بندی و منابع را صرفه جویی می کند و به سرعت بخشیدن به پروسه استقرار مدل در تولید (production) کمک می کند.

فایل های pickle به دست آمده از مدل و فرآیند بردار سازی را دانلود می کنیم تا در ادامه بتوانیم به کمک پکیج Flask و کدهای html – css یک وب پیچ ساده که به مدل متصل (End to End Pipeline) است ایجاد کنیم.

۸-۳-۴ استقرار مدل با پکیج Flask



شکل ۲۸-۴ کد پکیج فلسک

این کد مربوط به اپلیکیشن وب این پروژه است که با استفاده از فریم‌ورک Flask در زبان برنامه‌نویسی پایتون نوشته شده است. این اسکریپت یک مدل یادگیری ماشین آموزش دیده را به عنوان یک سرویس وب قابل دسترس قرار می‌دهد. در این کد، دو مدل پیکل شده که یکی مربوط به مدل XGBoost و دیگری مربوط به Vectorizer هستند، بارگذاری شده‌اند. همچنین تابعی به نام preprocess تعریف شده که وظیفه پیش‌پردازش متن ورودی را دارد. این پیش‌پردازش شامل نرمال سازی، توکن سازی، ریشه یابی و اشتقاق می‌شود. دو مسیر HTTP تعریف شده‌اند: یکی برای نمایش صفحه اصلی و دیگری برای پیش‌بینی احساسات متن ورودی که از طریق یک درخواست POST به سرور فرستاده می‌شود. پس از دریافت متن، این اسکریپت آن را پیش‌پردازش کرده و با استفاده از مدل یادگیری ماشین، احساس آن را پیش‌بینی می‌کند.

۹-۳-۴ کدهای HTML, CSS, js

```

    <p id="modalPredictionText"></p>
  </div>
</div>
</div>
</div>

{% if prediction_text %}
  <h2 id="predictionText">پیش بینی احساس: {{ prediction_text }}</h2>
{% endif %}
<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script>
<script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.4.8/dist/umd/popper.min.js"></script>
<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>
</div>
<script type="text/javascript">
  document.addEventListener('DOMContentLoaded', function () {
    var form = document.querySelector('form');
    var loader = document.getElementById('loader');
    var overlay = document.getElementById('overlay');
    var resultModal = document.getElementById('resultModal');
    var modalPredictionText = document.getElementById('modalPredictionText');

    form.onsubmit = function() {
      loader.style.display = 'block';
      overlay.style.display = 'block';
    };
  });

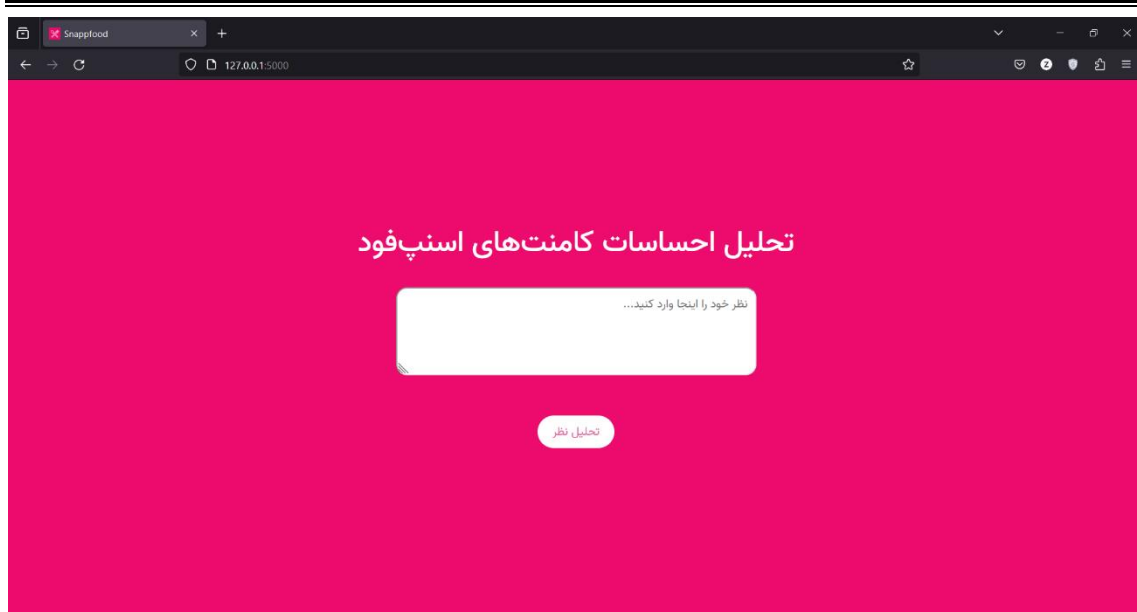
```

شکل ۲۹-۴ بخشی از کد رابط کاربری

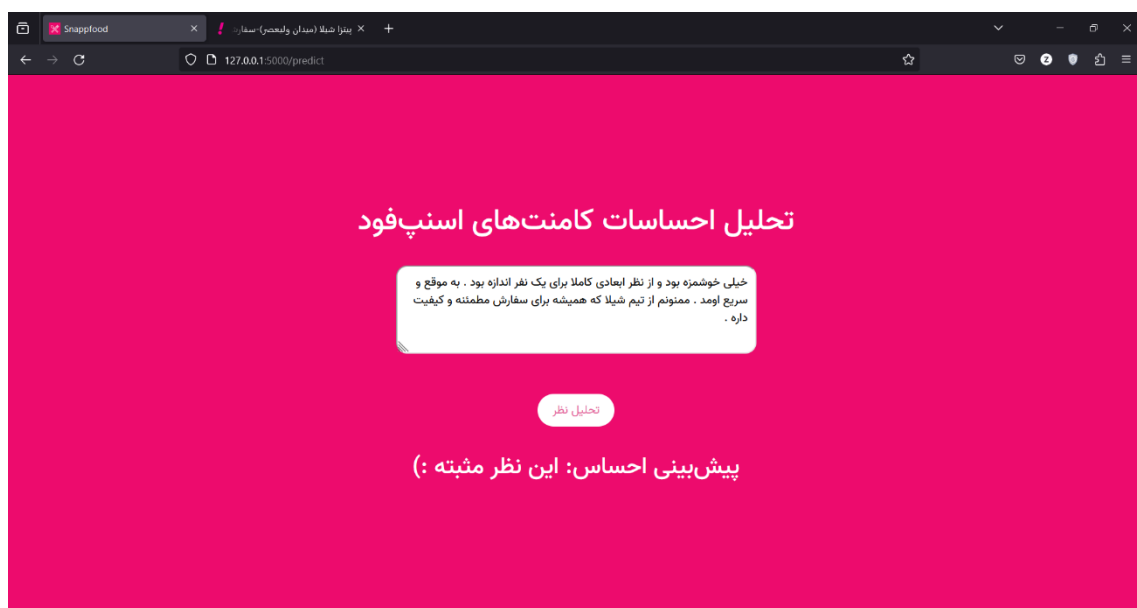
فایل index.html به عنوان رابط کاربری اصلی اپلیکیشن وب برای تحلیل احساسات متون فارسی طراحی شده است. این صفحه دارای فرمی است که کاربران می توانند متن خود را برای تحلیل وارد کنند. پس از ارسال متن، این درخواست به سروری که مدل یادگیری ماشین را اجرا می کند فرستاده می شود و نتایج تحلیل به صورت دینامیک روی همین صفحه نمایش داده می شود. طراحی صفحه با المان های زیبا و کاربردی، از جمله تصاویر، دکمه ها، فونت ها و رنگ بندی، تمرکز بر تجربه کاربری و دسترسی آسان و واکنش گرا را نشان می دهد. این فایل با استانداردهای وب و دسترسی، مطابقت با انواع دستگاه ها و مرورگرها را تضمین می کند و تجربه ای روان و مؤثر را برای کاربران فراهم می آورد.

۴-۴ اجرای نرم افزار

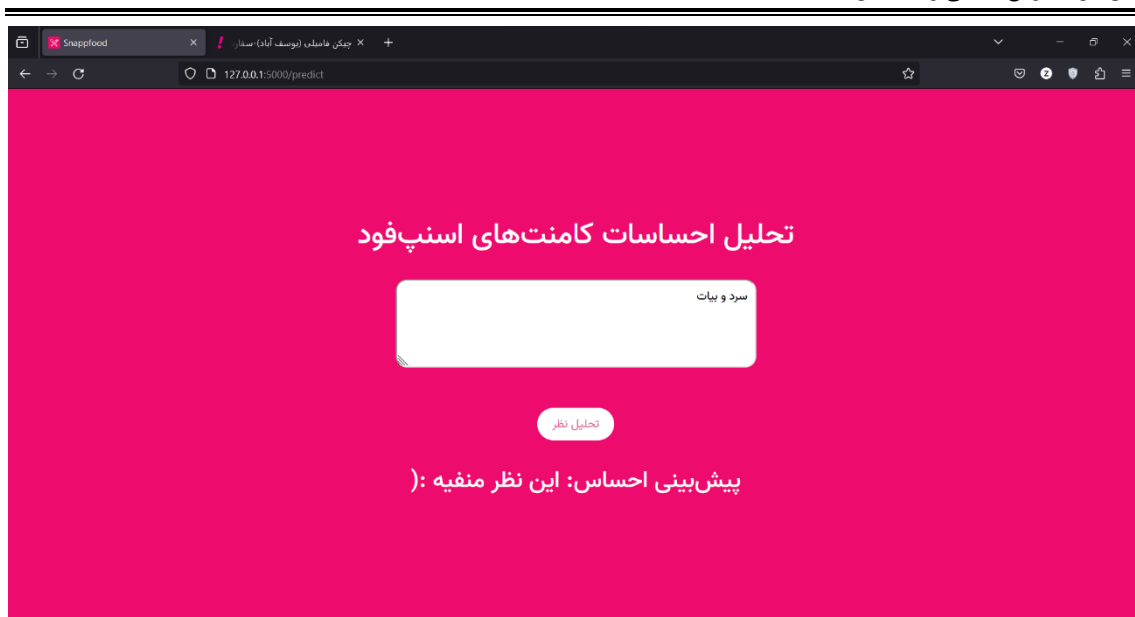
بعد از اجرای کد پایتون حاوی پکیج فلسک، بر روی localhost (۱۲۷,۰,۰,۱) و پورت ۵۰۰۰ وبسایت تحلیل نظر بالا می آید. در ادامه کامنت های واقعی از پلتفرم اسنپ فود کپی می شوند و به عنوان ورودی به مدل داده می شوند و بعد از پردازش مدل خروجی را مشخص می کند.



شکل ۳۰-۴ وبسایت تشخیص احساس کامنت



شکل ۳۱-۴ وارد کردن نظر مثبت و دیدن نتیجه



شکل ۳۲-۴ وارد کردن نظر منفی و دیدن نتیجه

۵-۴ نتیجه‌گیری

این پروژه با استفاده از ابزارهای مدرن پیاده‌سازی شده و شامل بخش‌های مختلفی است. از جمله کارهای انجام شده می‌توان به نصب کتابخانه‌ها، وارد کردن دیتاست، پیش‌پردازش و بردارسازی متن، ساخت و ارزیابی مدل‌ها، و مصورسازی داده‌ها اشاره کرد. در این پروژه، تمرکز بر روی تحلیل احساسات نظرات فارسی است و از داده‌های اسنپ‌فود استفاده شده است. برای پیاده‌سازی، ابزارهایی چون Pandas، Hazm، و Scikit-learn به کار رفته‌اند. این پروژه شامل مراحل مختلفی از جمله پیش‌پردازش داده‌ها، تقسیم داده‌ها به دو بخش آموزشی و تست، نرمال‌سازی، و استفاده از الگوریتم‌های یادگیری ماشین مانند Random Forest و XGBoost است. در نهایت، مدل‌ها با استفاده از معیارهایی مانند دقت و نمره F1 ارزیابی شده‌اند. در فصل بعد جمع‌بندی و پیشنهادات ارائه می‌شوند.

فصل پنجم

«جمع‌بندی و پیشنهادها»

۵-۱ نتیجه‌گیری

این پژوهش به تحلیل دقیق و عمیق احساسات در شبکه‌های اجتماعی پرداخته و با به‌کارگیری الگوریتم‌های نوین و معتبری چون XGBoost و Random Forest، توانسته است دینامیک‌های پیچیده‌ی جوامع مجازی را شناسایی کند. مطالعه بر روی داده‌های اسنپ‌فود و توییتر نشان داد که تعاملات عاطفی کاربران نقش بسزایی در شکل‌گیری و استحکام اجتماعات دارند. به‌کارگیری روش‌های پیشرفته‌ی تحلیل داده‌ها، از جمله استفاده از کتابخانه‌هایی نظیر Hazm و Scikit-learn، به تفسیر و درک بهتر رفتارهای کاربری منجر شده است. نتایج این تحقیق می‌تواند در حوزه‌هایی مانند بازاریابی دیجیتال، سیاست‌گذاری‌های اجتماعی و مطالعات رفتاری مورد استفاده قرار گیرند و زمینه‌ساز توسعه‌ی راهکارهای نوآورانه در این عرصه‌ها شوند.

۵-۲ پیشنهادهایی برای کارهای آتی

برای پروژه‌های آینده می‌توان بر روی توسعه و بهینه‌سازی الگوریتم‌های یادگیری ماشین در زمینه شبکه‌های عصبی عمیق کار کرد که می‌توانند در تحلیل احساسات و داده‌های شبکه‌های اجتماعی کاربردی‌تر باشند. همچنین، افزایش حجم و تنوع داده‌هایی که برای تحلیل مورد استفاده قرار می‌گیرند، می‌تواند به بهبود کیفیت و دقت مدل‌های موجود کمک کند. بررسی اثربخشی الگوریتم‌های جدید در زمینه‌های مختلفی نظیر بازاریابی دیجیتال و سیاست‌گذاری‌های اجتماعی نیز می‌تواند بینش‌های جدیدی در این حوزه‌ها ارائه دهد.

فهرست منابع

نمونه مقاله

- [1] Emotional community detection in social networks,
Andreas Kanavos, Isidoros Perikos, Ioannis Hatzilygeroudis
- [2]

نمونه صفحات وب

- [3] <https://www.kaggle.com/datasets/soheiltehranipour/snappfood-persian-sentiment-analysis>
- [4] <https://www.kaggle.com/code/nimanta/snappfood-sentiment-analysis>
- [5]

Abstract

This thesis aims to identify and analyze emotional communities in social networks through advanced data science and machine learning techniques. Utilizing credible datasets and innovative text processing methods, an innovative approach for analyzing user sentiments has been developed. The research findings indicate that the implemented models are capable of accurately and automatically detecting emotional states in Persian texts. The results of this study have significant implications for designing recommendation systems and analyzing user behavior on digital platforms, offering suggestions for improving existing systems and developing text analysis methods in Persian.

Keywords: Data Science, Machine Learning, Sentiment Detection, Community Detection



Technical and Vocational University
Dr. Shariaty- Tehran Girls Technical and
Vocational College

B.S Thesis on Software Engineering

Title:

Modeling Emotional Communities and Emotion
Recognition

Supervisor:

Dr. Zahra Valadkhani

By:

Zahra Rahimian

Fall 1402