تشخیص کنایه در متن با استفاده از یادگیری ماشین

دانشگاه صنعتی شریف

احمد آقاپورباب، علیرضا محمدیان، کیان باختری
bakhtari.kian@gmail.com - alireza.am1379@gmail.com - alivar6@gmail.com

۲۵ تیر ۱۴۰۰

چکیده: در این پروژه، مسئله‌ی دسته‌بندی یک قطعه نوشته (مانند نظرات در شبکه‌های اجتماعی) به دو دسته‌ی «معمولی» و «کنایه‌آمیز» مورد بررسی قرار گرفته و سعی شده است تا با استفاده از روش‌های مبتنی بر یادگیری ماشین به حل و انجام آن پرداخته شود. مجموعه دادگان استفاده شده در این پروژه، شامل حدود یک میلیون نظر (comment) در شبکه‌ی اجتماعی Reddit می‌باشد که همراه با اطلاعاتی از قبیل نام کاربری نویسنده، موضوع کلی نظر و... توسط برچسب‌گذاری به دو دسته‌ی معمولی و کنایه‌آمیز تقسیم شده‌اند. نتایج به دست آمده:

۱ مقدمه

یکی از مسائل معروف در حوزه‌ی پردازش زبان طبیعی [1] تشخیص کنایه است. این موضوع برای خود انسان نیز می‌تواند چالش‌برانگیز باشد و برای ماشین هم به سادگی امکان‌پذیر نیست. در این پروژه به حل این مسئله با استفاده از یادگیری ماشین پرداخته شده و چندین مدل و الگوریتم متفاوت آزموده شده‌اند. برای پیش‌پردازش از دو یا (یا سه یا چند؟) روش تبدیل کلمات و متن‌ها به بردارها [2] استفاده شده و کارایی مدل‌های آزموده شده مورد بررسی و تفسیر قرار گرفته‌اند. پس از تحلیل اکتشافی داده و بررسی ویژگی‌های مجموعه داده‌ها در بخش دوم، به مهندسی ویژگی‌ها و پیش‌پردازش پرداخته شده است و پس از آن در بخش چهارم، تعلیم و آزمون چند مدل ابتدایی‌تر به عنوان base line انجام شده است. و بعد چه و چه و چه

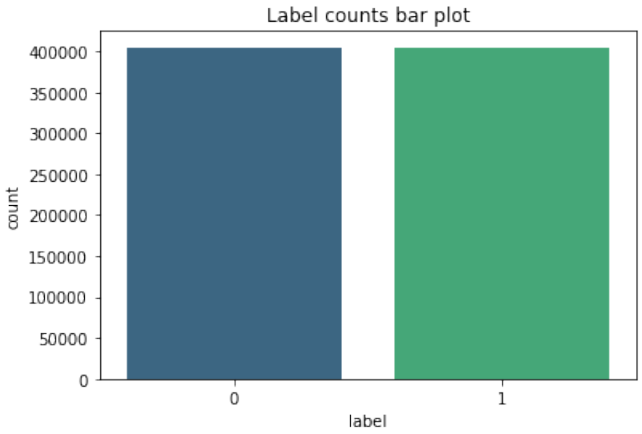
۲ تحلیل اکتشافی داده

مجموعه دادگانی که در اختیار داریم ۸۰۸۹۶۰ سطر و ۱۰ ستون را در خود جای داده است. هر سطر شامل متن یک نظر، برچسب کلاس مربوطه، نام کاربری نویسنده، امتیازات آن نظر، تاریخ و زمان ثبت نظر، موضوع و متن نظری می‌باشد که در ساختار درختی نظرات، در جایگاه پدر این نظر قرار دارد. منظور از موضوع، subreddit ای است که این نظر در آن ثبت شده است. مثلاً بخش سیاست، ورزش، یا موضوعات بامزه. ۳۸ سطر هستند که متن نظر مربوطه در آن‌ها ناموجود است و این سطرها حذف می‌شوند. به غیر از این، مقادیر گم‌شده‌ی دیگری در مجموعه وجود ندارد و همه‌ی ستون‌ها از نظم و تمیزی اولیه برخوردار هستند. موردی که در این بخش بیشتر به آن خواهیم پرداخت، توازن مجموعه و چگونگی تقسیم توزیع ویژگی‌ها برحسب برچسب کلاس است. در شکل ۱ نمودار تعداد نظرات برحسب برچسب کلاس را مشاهده می کنید. ۴۰۴۲۲۵ نظر معمولی و ۴۰۴۷۳۵ نظر کنایه‌آمیز در اختیار داریم که توازن خوبی را به دست می‌دهد.

۲-۱ Subreddits

در این مجموعه داده ۱۴۷۸۷ گروه موضوعی به نام subreddit وجود دارد. در شکل ۲، ده subreddit ای را مشاهده می‌کنید که بیشترین تعداد نظرات در آن‌ها ثبت شده و تعداد آن‌ها به شکل نمودار ستونی نمایش داده شده است. از حالا به بعد، هر subreddit را یک «گروه» می‌نامیم. در شکل ۳، نمودار ستونی تعداد نظرات در همان گروه‌های پرطرفدار را مشاهده می‌کنید که برحسب برچسب کلاس تفکیک شده است. علی‌رغم این که در مجموعه گروه‌هایی وجود دارند که همه‌ی نظرات‌شان معمولی و یا کنایه‌آمیز باشد، اما عموماً از تعداد بسیار کمی نظر تشکیل شده‌اند و به نظر می‌رسد که در گروه‌هایی که تعداد زیادی نظر را شامل می‌شوند، عموماً از توازن نسبی میان تعداد نظرات معمولی و کنایه‌آمیز برخوردارند. در شکل ۴ درصد نظرات کنایه‌آمیز در گروه‌های پرطرفدار به نمایش درآمده است.

[1] Natural Language Processing (NLP)
[2] word embedding



شکل ۱: نمودار تعداد نظرات برحسب برچسب کلاس (صفر به معنای معمولی و یک به معنای کنایه‌آمیز است)

۲-۲ طول نظرات

شکل ۵ نمودار جعبه‌ای طول نظرات را نمایش می‌دهد. مشاهده می‌شود که تعداد اندکی نظر با طول بسیار زیاد (نزدیک به ۱۰۰۰۰حرف) وجود دارد اما اکثر نظرات با طولی کم‌تر از ۵۰۰ حرف نوشته شده‌اند. در شکل ۶ و ۷ به ترتیب توزیع طول نظرات و همین توزیع به تفکیک برچسب کلاس قابل مشاهده است. توزیع‌ها با تقریب قابل قبولی یکسان هستند و در نتیجه طول نظر یک ویژگی مهم برای تشخیص کنایه‌آمیز بودن آن به حساب نمی‌آید.

۲-۳ تاریخ و ساعت

توزیع تاریخ و ساعت ثبت نظرات هم به تفکیک کلاس نظرات بررسی شده و در شکل‌های ۸، ۹، و ۱۰ قابل مشاهده است. مشخصاً مجموعه از سال‌های ۲۰۱۶ و سال‌های نزدیک‌تر به ۲۰۱۶ تعداد نظرات بیشتری را در خود جای داده، اما باز هم به نظر نمی‌رسد که تاریخ یا ساعت ثبت یک نظر تاثیر به خصوصی در کنایه‌آمیز بودن یا نبودن آن داشته باشد.

۲-۴ امتیازات

برای هر نظر، سه امتیاز با نام‌های score, ups, downs ثبت شده است. توزیع امتیازات به تفکیک کلاس نظرات در شکل‌های ۱۱، ۱۲، و ۱۳ به نمایش درآمده است. باز هم می‌توان دید که بالا یا پایین بودن امتیازات یک نظر اطلاعات چندانی از کنایه‌آمیز بودن یا نبودن آن به دست نمی‌دهد.

۲-۵ ابر کلمات

با استفاده از کتابخانه‌ی wordcloud یک نمایش بصری از کلمات پر بسامد در نظرات تشکیل می‌شود که اندازه‌ی هر کلمه متناسب با تکرار آن کلمه است و می‌تواند در مورد واژه‌های پرتکرار شهود قابل قبولی به دست دهد. در شکل ۱۴ نمایش ابر کلمات برای نظرات معمولی و در شکل ۱۵ برای نظرات کنایه‌آمیز قابل مشاهده است. مشاهده‌ی این تصاویر اثر به خصوصی در تصمیمات گروه برای پیش‌پردازش و مهندسی ویژگی‌ها به دنبال نداشت.

۲-۶ علائم خاص

در زیربخش ابر کلمات مشاهده کردیم که چه کلماتی بیشترین بسامد را در نظرات هر کلاس داشتند. اما موارد دیگری نیز قابل بررسی هستند، مانند میزان استفاده از علائم خاص مثل نقطه، ویرگول، پرانتز و غیره. این موضوع از این جهت ارزش بررسی دارد که می دانیم خیلی اوقات کنایه خودش را به شکل علائم اختصاری و صورتک‌ها نشان می‌دهد، مانند (: و !!! و یا سایر موارد مشابه. نمودار تعداد علائم خاص استفاده شده در نظرات به تفکیک کلاس را می‌توانید در شکل ۱۶ مشاهده کنید. به نظر می‌آید از نظر تعداد علائم خاص تفاوت چشم‌گیری میان نظرات کنایه‌آمیز و معمولی وجود ندارد.

۲-۷ اصطلاحات

برای بررسی تاثیر حضور اصطلاحات و کلمات پرتکرار در نظرات، از مدل logistic regression استفاده شده است. با استفاده از امکان نمایش ویژگی‌هایی که خود مدل پر اهمیت دانسته است، می‌توان دریافت که حضور یا عدم حضور چه اطلاعاتی در متن روی احتمال کنایه‌آمیز بودن آن تاثیر می‌گذارد. در شکل ۱۷ می‌توان ۴۰ کلمه و اصطلاحی که به زعم مدل نقش پررنگی در کنایه‌آمیز بودن یا نبودن نظر ایفا می‌کنند و نمی‌کنند را مشاهده کرد. بعدها در طراحی معماری مدل‌های CNN برای تعیین ابعاد فیلترها از نتیجه‌ی این بخش کمک خواهیم گرفت.

۲-۸ نظرات پدر

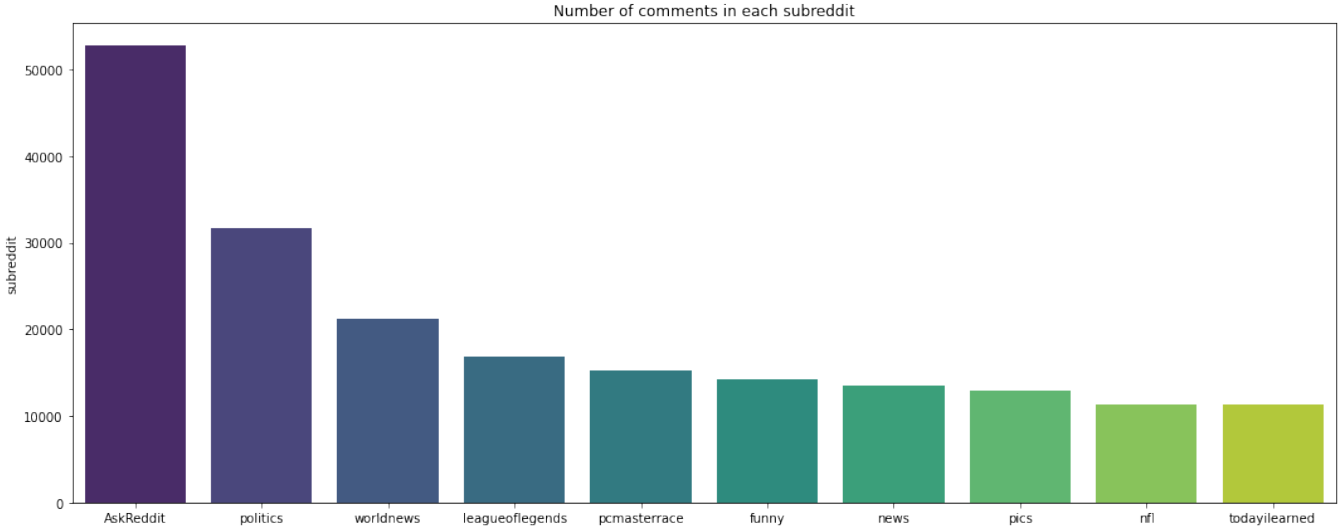
به احتمال زیادی اگر پدر یک نظر کنایه‌آمیز باشد، خود آن نظر نیز کنایه‌آمیز است. این جمله حداقل در ظاهر درست به نظر می‌رسد. اما در این مجموعه‌ی داده ما چندان امکان بررسی این موضوع را نداریم چرا که اشتراک نظرات اصلی و نظرات پدر در مقایسه با تعداد نظرات کل مجموعه بسیار کم است و برچسب کلاس نظرات پدر در اختیار ما قرار نگرفته است.

۲-۹ نتیجه‌گیری

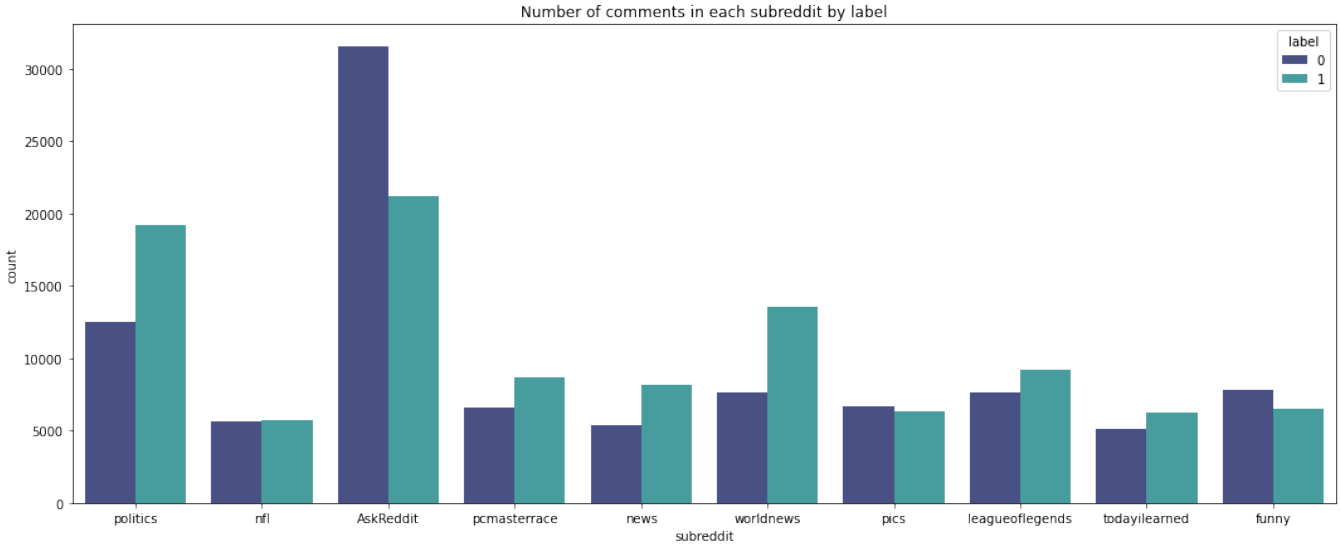
پس از مشاهده‌ی ارقام و نمودارهای توضیح داده شده در این بخش، می‌توان گفت مجموعه‌ی داده‌ها از توازن خوبی در تمامی ویژگی‌ها برخوردار است و کنایه‌آمیز بودن یا نبودن نظرات، تاثیر معنی‌داری بر سایر ویژگی‌های نظر مانند طول یا امتیاز آن ندارد. می‌توانیم به بخش پیش‌پردازش و مهندسی ویژگی‌ها بپردازیم.

۳ پیش‌پردازش

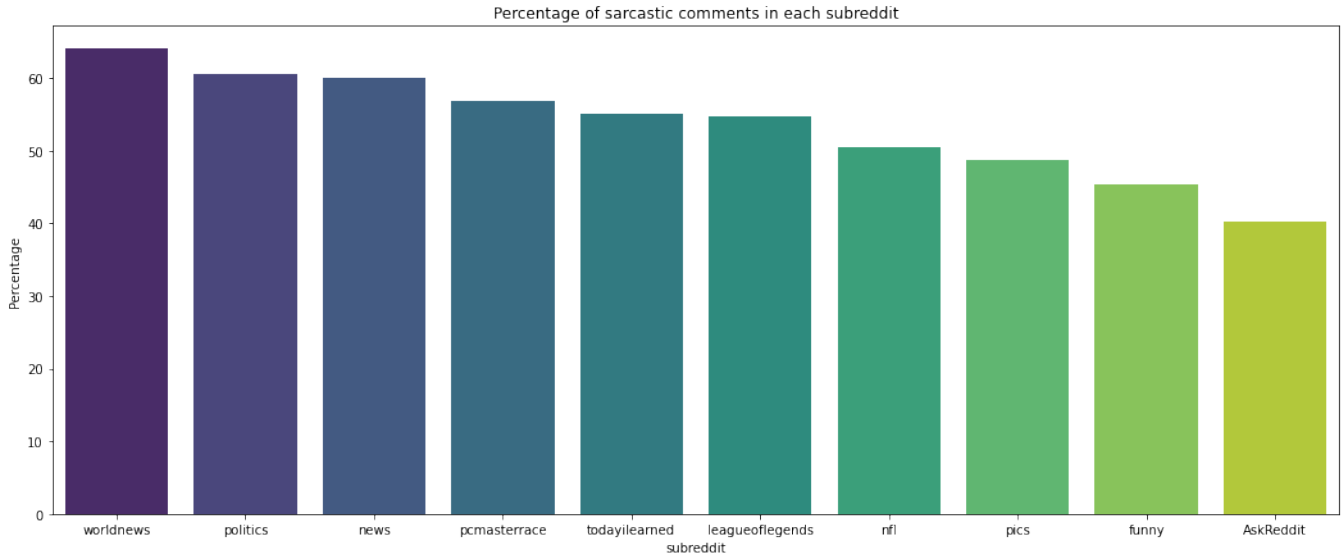
بدون تردید نظرات شامل واژه‌ها و نشانه‌هایی هستند که در تشخیص کنایی بودن یا نبودن کمک شایانی نمی‌کنند و به اصطلاح informa-tive نیستند. ابتدا سعی شده است تا با چند پیش‌پردازش روی نظرات، آن‌ها را به شکلی تبدیل کنیم که برای ماشین قابل هضم‌تر باشد. ذکر



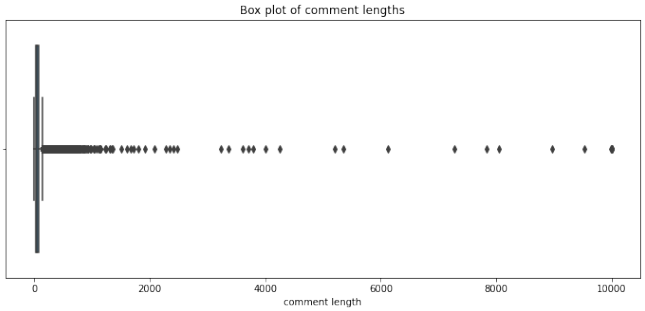
شکل ۲: نمودار ستونی تعداد نظرات در ده گروه (subreddit) حجیم‌تر مجموعه



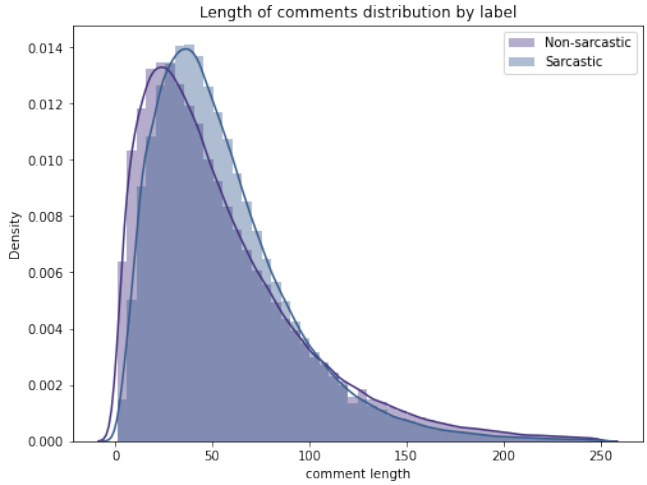
شکل ۳: نمودار ستونی تعداد نظرات در ده گروه (subreddit) حجیم‌تر مجموعه به تفکیک برچسب کلاس نظرات



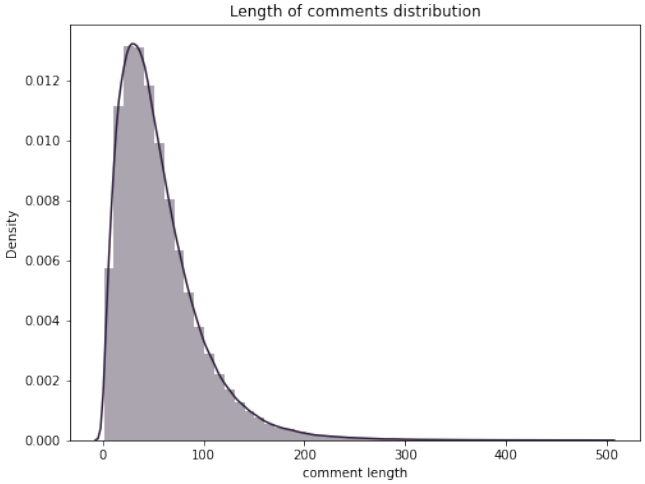
شکل ۴: نمودار ستونی درصد نظرات کنایه‌آمیز در ده گروه (subreddit) حجیم‌تر مجموعه



شکل ۵: نمودار جعبه‌ای طول نظرات



شکل ۷: توزیع طول نظرات به تفکیک برچسب کلاس



شکل ۶: توزیع طول نظرات

این مورد حائز اهمیت است که در بخش آزمایش مدل‌ها، عملکرد الگوریتم‌های یادگیری ماشین را در هر دو حالتی که این پیش‌پردازش‌ها انجام شده باشند یا نشده باشند مورد بررسی قرار خواهیم داد.

ابتدا بررسی شد که اگر نظراتی شامل کاراکترهای غیر ascii هستند، این کاراکترها حذف شوند. به نظر مجموعه‌ی داده از پیش مقداری تمیز شده و این مورد انجام شده است. سپس کلمات اختصاری [۳] به حالت غیر اختصاری تبدیل شدند، مانند تبدیل don't به do not . در مرحله‌ی بعد، نشانه‌ها و علائم نگارشی از نظرات حذف شدند. مجدداً شایان ذکر است که این موارد به صورت آزمایشی انجام شده‌اند و در ادامه حتما نظراتی که این تبدیل‌ها بر آن‌ها اعمال نشده باشد نیز آزموده خواهند شد.

[3] contraction

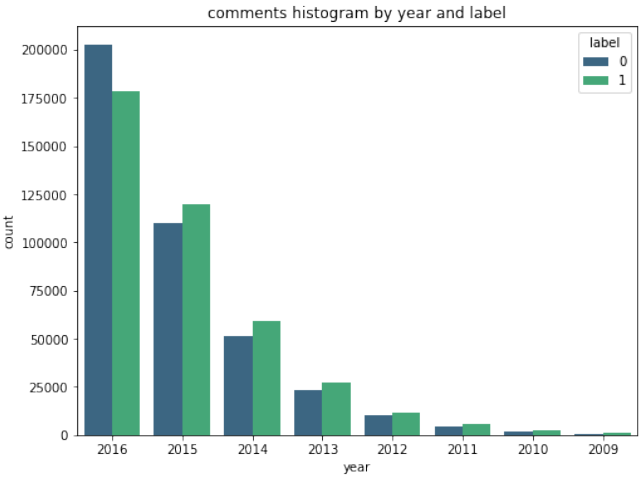
به عنوان پیش‌پردازش بعدی، افعال به حالت مصدری تبدیل شدند [۴] که این موضوع از پیچیدگی داده‌ها می‌کاهد و ممکن است کار آموزش و پیش‌بینی را برای مدل آسان‌تر کند. در آخرین مرحله، کلماتی مانند حروف اضافه و... که بسیار در زبان پر استفاده هستند به عنوان stop words شناخته شده و از نظرات حذف شدند.

۴ بازنمایی عددی (embedding)

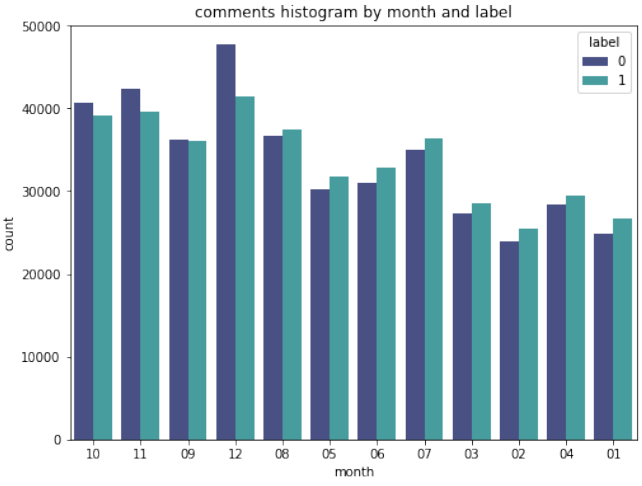
برای بازنمایی عددی ابتدا از دو روش معروف و توصیه شده در مستند پروژه استفاده شد که w2v و tf-idf نام دارند. هم برای داده‌هایی که پیش‌پردازش شده بودند و هم برای آن‌هایی که نشده بودند از این دو روش استفاده شده و مدل‌های ابتدایی آزموده شدند. در ادامه از روش glove هم استفاده شد که نتایج آن در بخش آزمایش مدل‌ها شرح داده خواهد شد. نکته‌ای که شایان ذکر می‌باشد این است که استفاده از روش w2v برای همه‌ی دادگان حجم بسیار زیادی از ram را اشغال می‌کرد و در تمامی اجراها از میزان ram در دسترس در سرویس google colab و Kaggle فراتر رفته و باعث می‌شد عملا استفاده از این روش فقط برای بخشی از دادگان مجموعه امکان‌پذیر باشد. در این روش، پس از تبدیل کلمات حاضر در مجموعه‌ی داده‌ها به یک لغت‌نامه [۵] که کلمه‌ها را به بردارها تبدیل می‌کند، کلمات حاضر در هر نظر به بردارهایی تبدیل شدند و به شکل سطرهای یک ماتریس دو بعدی کنار هم قرار گرفتند تا هر نظر تبدیل به یک ماتریس دو بعدی شود؛ ماتریسی که تعداد سطرهایش تعداد کلمات حاضر در نظر است و تعداد ستون‌هایش، ابعاد فضای برداری‌ای که کلمات در آن حاضر هستند. در ادامه برای یک‌شکل شدن ماتریس‌ها از zero padding استفاده شد.

اولین راه حل برای مشکل مذکور درباره‌ی پر شدن ram این بود که سطرهای ماتریس مربوط به هر نظر را با هم میانگین گرفته تا به ازای هر نظر یک برادر با طول بعد فضای برداری کلمات حاصل شود. با توجه به خواص روش w2v که روابط جبری میان کلمات را به معانی آن‌ها مربوط می‌سازد، این کار عملا معادل این است که کلمات حاضر در یک نظر را با هم میانگین بگیریم. حسن این روش کاهش حجم داده‌ی تولید شده است و ضعف آن در این است که با این کار اطلاعاتی که مربوط به ترتیب کلمات باشد از بین می‌رود و می‌توان گفت که اگر از این زاویه به آن نگاه کنیم، روش w2v به روش tf-idf تقلیل می‌یابد.

اما راه حل اصلی استفاده از معماری خط لوله [۶] است که از داده‌ها بخش به بخش بازنمایی عددی می‌سازد و به مدل تحویل می‌دهد. با این روش مشکل مذکور حل شده و برای آموزش دادن اکثر مدل‌ها با بازنمایی w2v از این روش استفاده شد.

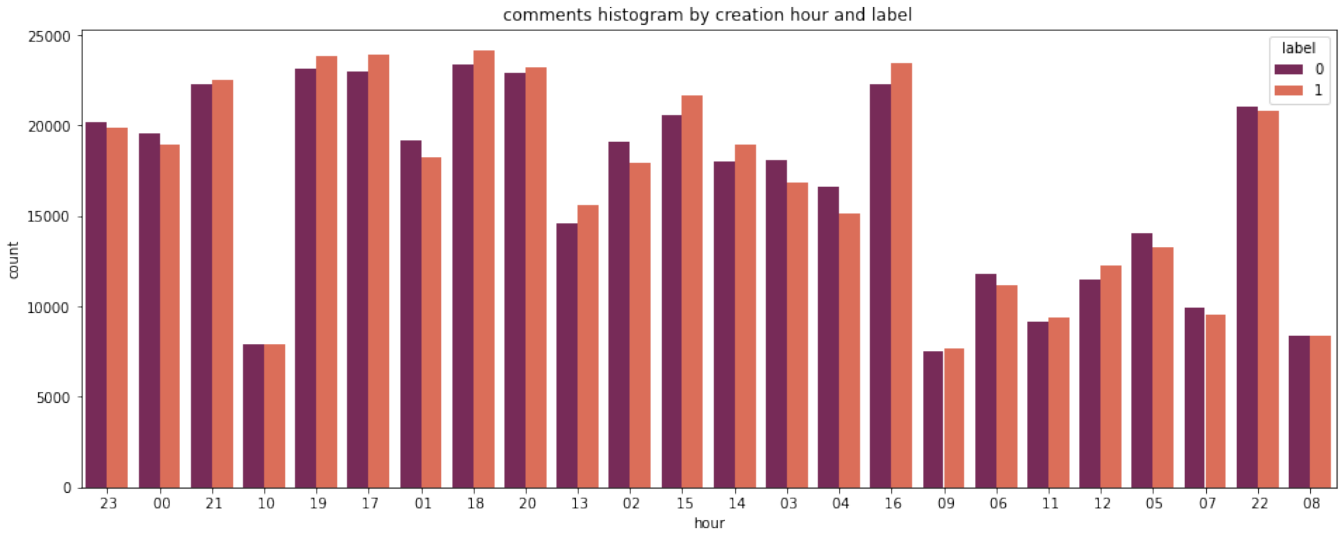


شکل ۸: هیستوگرام نظرات بر حسب سال ثبت نظر و به تفکیک کلاس

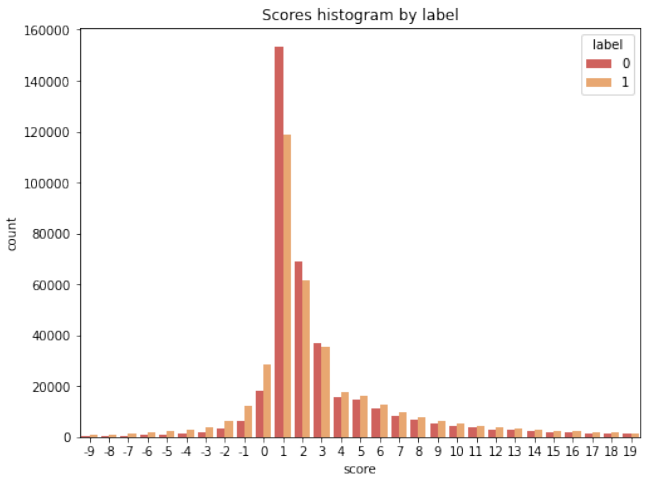


شکل ۹: هیستوگرام نظرات بر حسب ماه ثبت نظر و به تفکیک کلاس

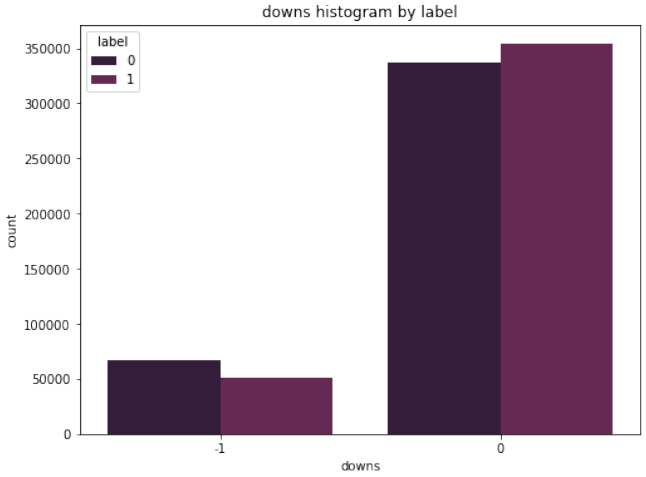
[4] lemmatization
[5] dictionary
[6] pipe line



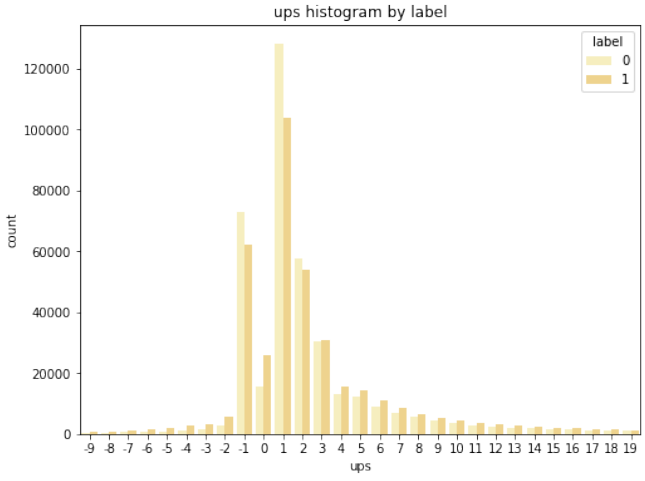
شکل ۱۰: هیستوگرام نظرات بر حسب ساعت ثبت نظر و به تفکیک کلاس



شکل ۱۱: توزیع امتیاز (score) نظرات به تفکیک کلاس



شکل ۱۳: توزیع امتیاز downs به تفکیک کلاس



شکل ۱۲: توزیع امتیاز ups نظرات به تفکیک کلاس

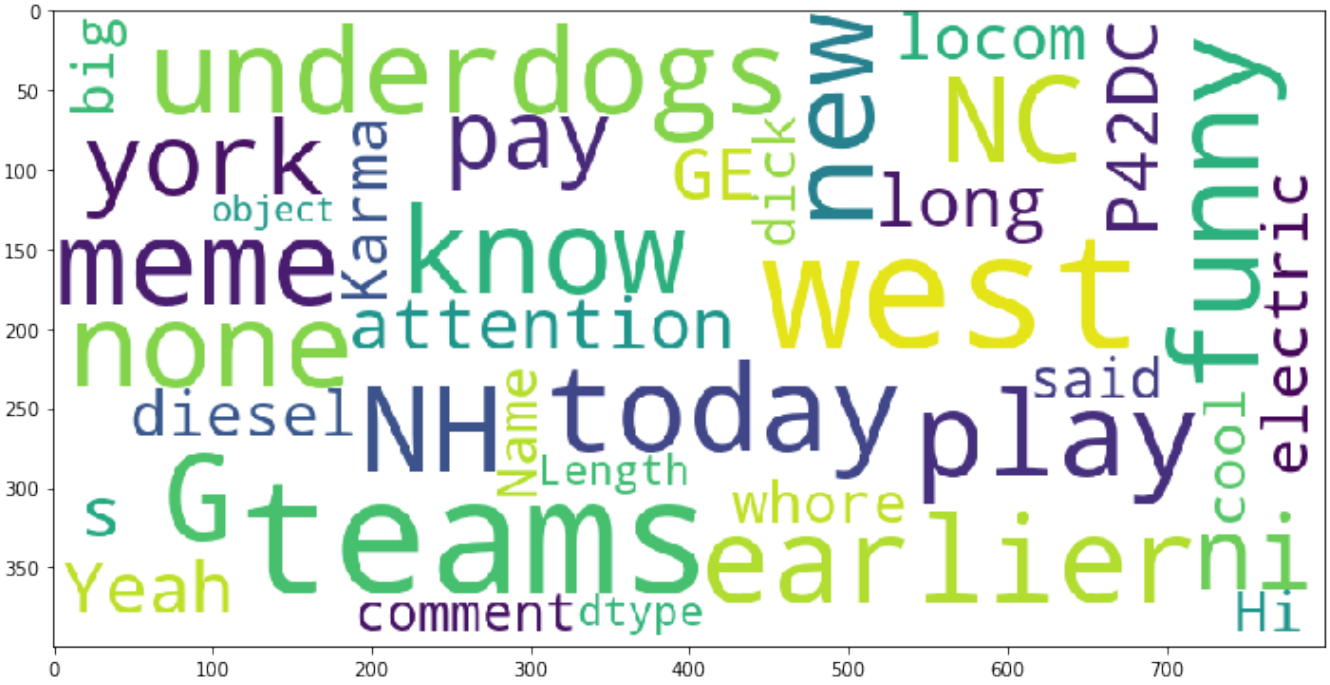
۵ مدل‌های اولیه

پس از تحلیل اکتشافی داده، پیش‌پردازش و بازنمایی عددی نظرات، در این بخش چند مدل ساده مورد آزمون قرار داده شد و تلاش شد تا عملکرد پیش‌پردازش‌ها و بازنمایی‌های متفاوت آزمایش شود.

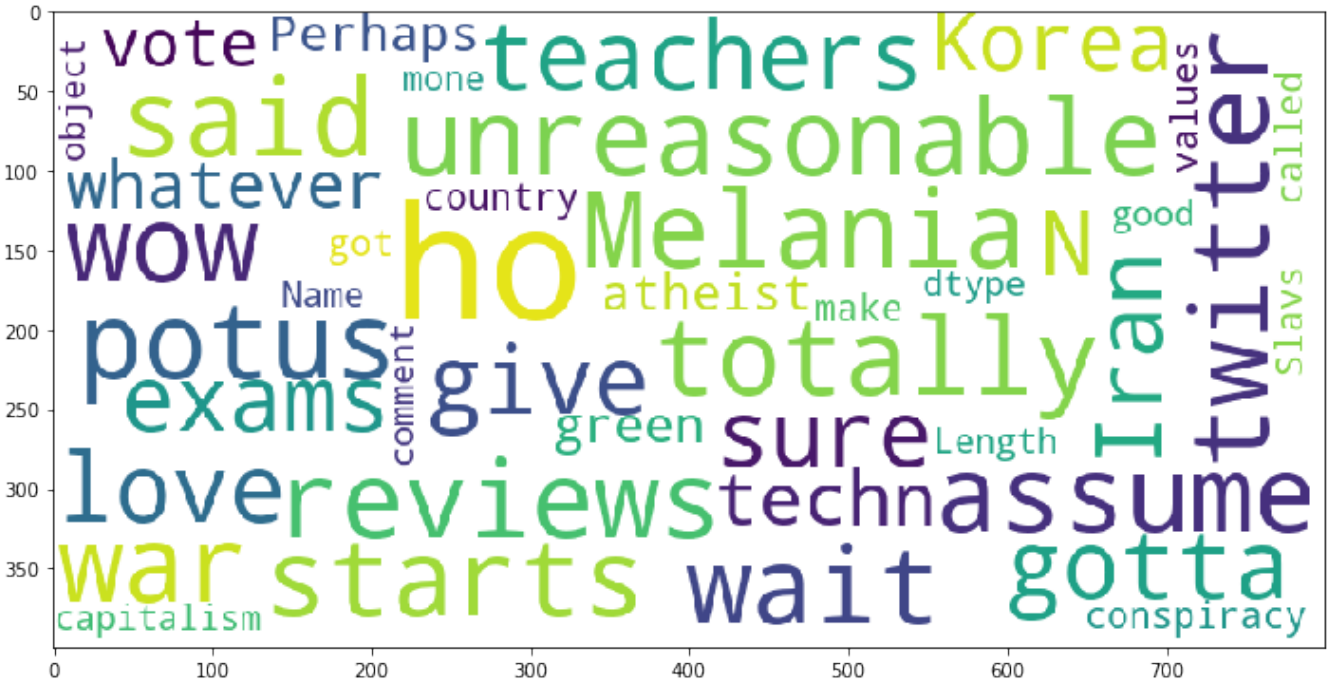
۱-۵ Logistic Regression and tf-idf

در بسیاری از مسائل دسته‌بندی در یادگیری ماشین، از این مدل به عنوان مدل پایه [۷] استفاده می‌شود. ابتدا از بازنمایی tf-idf استفاده شد و پس از استفاده از جست‌وجوی جدولی [۸] برای یافتن بهترین ابرپارامترها، نتایج accuracy:72 و F1score:71 به دست آمد. نتیجه‌ی سه بار آزموده شدن این مدل بر داده‌های ارزشیابی در شکل ۱۸ به نمایش درآمده است.

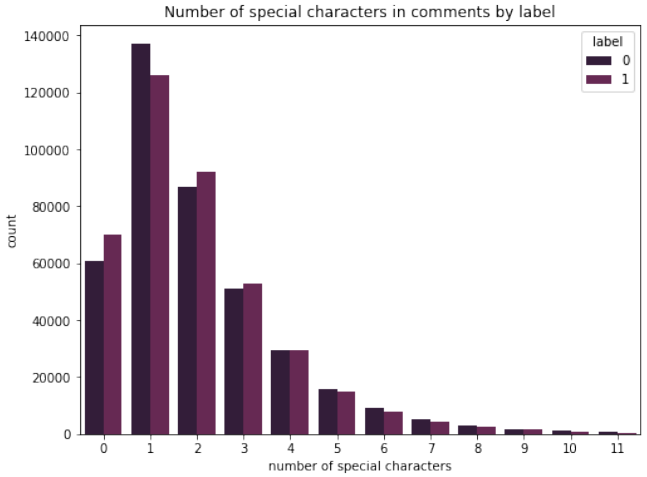
[7] base model
[8] grid search



شکل ۱۴: ابر کلمات پر بسامد در نظرات معمولی



شکل ۱۵: ابر کلمات پر بسامد در نظرات کنایه‌آمیز



شکل ۱۶: نمودار تعداد علائم خاص استفاده شده در نظرات به تفکیک کلاس

+9.093	yes because	-3.150	necessarily
+7.595	obviously	-3.186	so yeah
+7.540	clearly	-3.216	never said
+7.301	totally	-3.285	them but
+6.844	because	-3.347	not sure
+6.739	yeah because	-3.421	as well
+5.728	how dare	-3.426	right now
+5.635	good thing	-3.450	not really
+5.161	gee	-3.457	but yeah
+5.105	duh	-3.462	sadly
+5.028	yeah	-3.618	unfortunately
+4.857	fault	-3.619	doesn mean
+4.657	not like	-3.631	afaik
+4.617	but thought	-3.715	right but
+4.532	right because	-3.935	true but
+4.494	everyone knows	-4.147	that but
+4.483	forgot the	-4.228	it but
+4.469	dropped this	-4.244	imo
+4.421	therefore	-4.257	fair enough
+4.420	sexist	-4.841	iirc

شکل ۱۷: بخشی از کلمات تاثیرگذار در تصمیم‌گیری مدل logisticRegression .ستون سبز رنگ کلمات و اصطلاحات با وزن بیشتر در نظرات کنایه‌آمیز را نشان می‌دهد و ستون قرمز رنگ کلمات و اصطلاحات مربوط به نظرات معمولی. اعداد نوشته شده وزن هر اصطلاح را نشان می‌دهد.

	Train: 0	Train: 1	Train: 2	Mean	Std
Accuracy	0.722267	0.720468	0.723046	0.721927	0.001079
F1_score	0.711787	0.709788	0.712259	0.711278	0.001071
Precision	0.738173	0.738487	0.739197	0.738619	0.000428
Recall	0.687222	0.683237	0.687216	0.685892	0.001877
ROC_AUC	0.722200	0.720492	0.722959	0.721884	0.001032

شکل ۱۸: نتایج سه بار آموزش و آزمون مدل logisticRegression به همراه بازنمایی tf-idf بر داده‌های ارزشیابی

در پایان پروژه و پس از آزمودن مدل‌های پیشرفته‌تر، مدل logisticRe-gression رودی داده‌های آزمون نیز سنجیده شد و نتایج به دست آمده شباهتی با امتیازات داده‌های ارزشیابی نداشت. F1score نزدیک به ۵۰ به دست آمد.

۵-۲ Logistic Regression and w2v

با استفاده یک جریان داده در خط لوله‌ای که ابتدا بازنمایی را ایجاد می‌کند و سپس مدل را آموزش می‌دهد عملکرد logistic regression را مورد آزمایش قرار دادیم و نتیجه اصلاً رضایت‌بخش نبود. امتیازات در بهترین حالت به ۶۵ نزدیک می‌شدند.

۵-۳ Random forest and tf-idf

در این بخش از جنگل تصادفی استفاده شده است. به علت حجم بالای ماتریس داده (حدود ۸۰۰۰۰۰ سطر و ۱۰۰۰۰ ستون) آموزش این مدل مدت زمان زیادی به طول انجامید و در نهایت هم نتایج قابل قبولی حاصل نشد. چهار مرتبه با بیشینه عمق depth max های متفاوت مورد آزمایش قرار گرفت اما امتیازی که به ۷۰ برسد به دست نیامد و در اجراهای مختلف بین ۵۵ تا ۶۵ نوسان می‌کرد.

۵-۴ Random forest and w2v

مانند زیر بخش پیشین، اما این بار با استفاده از بازنمایی w2v یک جنگل تصادفی آموزش داده شد. نتایج نچندان مطلوب مانند قسمت قبلی حاصل شد و امتیازات از ۶۵ فراتر نرفتند.

۵-۵ Linear SVM and tf-idf

در این بخش از مدل خطی svm و بازنمایی tf-idf استفاده شد. ابرپارامترهای مدل با استفاده از جست‌وجوی جدولی تنظیم شدند و نتایجی بسیار شبیه به logisticRegression به دست آمد. دلیل استفاده از مدل خطی svm این است که زمان اجرای مدل غیر خطی آن بسیار حساس به تعداد نمونه‌های ورودی است و آموزش آن برای این حجم از داده عملی نمی‌باشد. نتایج accuracy:72 و F1score:70 به دست آمد.

۶-۵ Linear SVM and mean w2v

برای آموزش دادن یک مدل خطی svm با داده‌های بازنمایی شده به روش w2v و بدون تولید شدن حجم بسیار زیاد داده‌ی بازنمایی شده، از روش میانگین‌گیری ماتریس هر نمونه که در ابتدای قسمت اصلی این بخش توضیح داده شد استفاده شد. نتیجه به صورت مقابل به دست آمد که پیشرفتی در بر ندارد: accuracy:64 و F1score:63

۶ مدل‌های مبتنی بر یادگیری عمیق

پس از آزمون و خطا با مدل‌های پایه، چندین مدل مبتنی بر یادگیری عمیق مورد آموزش و آزمون قرار داده شدند تا امکان دسته‌بندی نظرات به دو کلاس کنایه‌آمیز و معمولی با استفاده از شبکه‌های عصبی نیز مورد بررسی قرار گیرد.

۶-۱ شبکه‌ی تمام متصل

برای شروع، یک شبکه‌ی تمام متصل [۹] دو لایه با داده‌های بازنمایی شده به روش w2v آموزش داده شد و نتیجه‌ی به دست آمده بهتر از مدل‌های پایه نبود. چندین حالت دیگر شبکه‌های تمام متصل نیز آزموده شدند و هیچ‌یک نتیجه‌ی بهتری در بر نداشتند، با افزایش پیچیدگی مدل مشکل بیش‌برازش پیش می‌آمد و با کاهش آن، مشکل کاهش دقت به وجود می‌آمد.

۶-۲ CNNs

علی‌رغم تعلق CNN [۱۰] ها به دنیای بینایی ماشین، [۱۱] در مسائل مرتبط به پردازش زبان طبیعی نیز استفاده‌های فراوانی از آن‌ها می‌شود. فیلترهایی که در مسائل بینایی ماشین روی عکس‌ها و تصاویر حرکت می‌کنند، در مسئله‌ی فعلی ما می‌توانند روی جملات حرکت کرده و کلمات، اصطلاحات، و الگوهایی که به تشخیص کنایه‌آمیز بودن یا نبودن جملات کمک می‌کنند را شناسایی کنند.

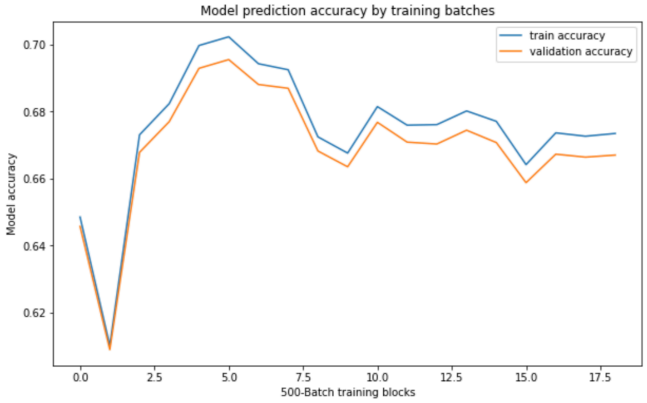
ورودی‌های اولین مدل CNN که آموزش داده شد، ماتریس‌هایی با ابعاد ۱۰۰ سطر در ۱۰۰ ستون بودند که همان بازنمایی w2v است. سه لایه‌ی convolutional پیاپی به همراه لایه‌ی maxPooling میان آن ها ساختار اولین مدل CNN را تشکیل می‌دهد. ضمن استفاده از batchNormalization و مقداردهی اولیه به وزن‌ها به روش kaiming- ، در نهایت از یک لایه‌ی تمام متصل برای کاهش بعد خروجی به دو کلاس استفاده شده است. ساختار کلی این مدل را در شکل ۱۹ مشاهده می‌کنید. نمودار دقت مدل بر داده‌های آموزش و ارزشیابی نیز در شکل ۲۰ به نمایش در آمده است.

همان‌طور که در نمودار نیز پیداست، دقت مدل به ۷۰ نزدیک شده اما بعداً کم‌تر شده است و می‌توان برداشت کرد که مدل از پیچیدگی کافی برخوردار نبوده است.

[9] fully connected
[10] convolutional neural network
[11] computer vision

```
N_net(
  (conv1): Conv2d(1, 16, kernel_size=(5, 100), stride=(1, 1))
  (bnorm): BatchNorm2d(16, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (pool1): MaxPool2d(kernel_size=(2, 1), stride=2, padding=0, dilation=1, ceil_mode=False)
  (conv2): Conv2d(16, 16, kernel_size=(3, 1), stride=(1, 1))
  (pool2): MaxPool2d(kernel_size=(2, 1), stride=2, padding=0, dilation=1, ceil_mode=False)
  (conv3): Conv2d(16, 8, kernel_size=(3, 1), stride=(1, 1))
  (fc): Linear(in_features=168, out_features=2, bias=True)
)
```

شکل ۱۹: معماری اولین مدل CNN



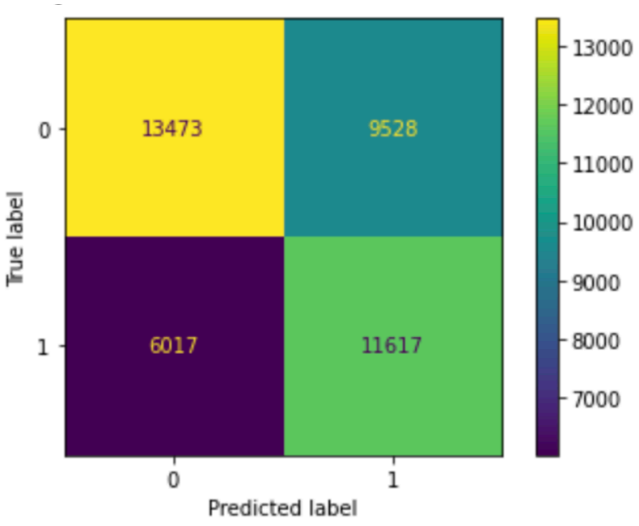
شکل ۲۰: دقت اولین مدل CNN بر داده‌های آموزش و ارزشیابی

```
N_net(
  (conv1): Conv2d(1, 32, kernel_size=(5, 100), stride=(1, 1))
  (convAug): Conv2d(1, 32, kernel_size=(1, 100), stride=(1, 1))
  (bnorm): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (pool1): MaxPool2d(kernel_size=(2, 1), stride=2, padding=0, dilation=1, ceil_mode=False)
  (conv2): Conv2d(32, 32, kernel_size=(3, 1), stride=(1, 1))
  (pool2): MaxPool2d(kernel_size=(2, 1), stride=2, padding=0, dilation=1, ceil_mode=False)
  (conv3): Conv2d(32, 32, kernel_size=(3, 1), stride=(1, 1))
  (fc): Linear(in_features=672, out_features=2, bias=True)
  (conv4): Conv2d(32, 32, kernel_size=(2, 1), stride=(1, 1))
  (fc2): Linear(in_features=704, out_features=2, bias=True)
)
```

شکل ۲۱: معماری بهبود یافته‌ی اولین مدل CNN

```
Accuracy:   0.61744801279685
F1_score:   0.5991387090951288
Precision:  0.5493970205722393
Recall:     0.6587841669502098
```

شکل ۲۲: نتایج اولین مدل CNN روی داده‌ی آزمون (امتیازات حدود ۱۰ درصد کم‌تر از اعدادی هستند که بر داده‌ی ارزشیابی به دست آمده بود)



شکل ۲۳: جدول سردرگمی (confusion matrix) برای مدل CNN

برای بهبود این مدل، به پیچیدگی آن اضافه کردیم که معماری جدید به دست آمده را در شکل ۲۱ می‌توانید مشاهده کنید. تعداد کانال‌های لایه‌های میانی دو برابر شده و همچنین یک لایه‌ی residual به شبکه اضافه شده است. به تبع، تعداد پارامترهای لایه‌های تمام متصل نیز افزایش قابل توجهی داشتند. این مدل بهبود معنی‌داری نسبت به مدل قبلی پیدا کرد و پس از ۱۵ پیمایش یادگیری روی داده‌های آموزش، به دقت و F1score بیش از ۷۰ اما کم‌تر از ۷۳ رسیده بود. متاسفانه به علت قطعی اینترنت و مشکلات ناشی از آن، نمودار پیشرفت این مدل ذخیره نشده اما پارامترهای مدل بازیابی شده و نتایج آن بر داده‌ی آزمون (داده‌ی test) مطابق با شکل ۲۲ به دست آمد. امتیازها حدود ۱۰ درصد کم‌تر از اعدادی هستند که روی داده‌ی ارزشیابی به دست آمده بود. همچنین جدول سردرگمی [۱۲] این مدل در شکل ۲۳ به نمایش در آمده است.

در ادامه مدل‌های convolutional را به حالت‌های دیگری نیز تغییر دادیم اما بهبودی حاصل نشد و نتایج بهتر از قبل نمی‌شدند. به عنوان نمونه، نمودار دقت یکی از همین مدل‌ها در شکل ۲۷ قابل مشاهده است که دیگر یادگیری معنی‌داری ندارد.

سپس یک شبکه‌ی CNN منتها این بار با استفاده از بازنمایی Glove آموزش داده شد. این شبکه با استفاده از ساختار نمایش داده شده در شکل ۲۴ بنا شد و اما نتایج بهتری نسبت به شکل به دست نداد. نمودار دقت مدل در شکل ۲۵ و امتیازات چهارگانه‌ی آن بر داده‌ی ارزشیابی در شکل ۲۶ قابل مشاهده است.

۳-۶ RNNs: LSTM

در مسائلی که سری زمانی در آن‌ها اهمیت دارد استفاده از مدل‌های RNN [۱۳] توصیه می‌شود. در این پروژه از این مدل‌ها نیز استفاده شده و سعی شد تا مسئله‌ی تشخیص کنایه در متن به نتایج بهتری برسد.

همان‌گونه که در شکل ۲۸ دیدیده می‌شود مدل LSTM دارای سلول‌هایی است و هر سلول آن دارای سه فاز می‌باشد و دو مسیر که یکی حالت سلول هست و دیگری حافظه. در فاز اول شبکه تصمیم میگیرد که چه مقدار از حالت قبلی را باید نگه دارد. برای مثال در مدل زبان اینگونه است که پس از خواندن یک متن، بعضی از اطلاعات گذشته فراموش شده و برای مثال اگر موضوع متن عوض شود فاز دوم سلول تصمیم میگیرد که در این سلول چقدر به حالتی که داشته‌ایم باید اضافه شود. برای مثال اگر ورودی این سلول یک صفت مانند جنسیت باشد چقدر آن را باید تاثیر دهیم. در فاز سوم نیز از حالتی که داشته‌ایم تصمیم‌گیری می‌شود که چقدر در کلمات بعدی تاثیر بگذارد.

این مدل را با استفاده از بازنمایی w2v آموزش دادیم و بر خلاف انتظارات نتایج مطلوبی حاصل نشد. نمودار دقت مدل بر داده‌های آموزش و ارزشیابی در شکل ۲۹ قابل مشاهده است.

برای آزمایش، از بازنمایی فراهم شده توسط کتابخانه‌ی tensor-flow استفاده شد و نتایج بهتری به دست آمد. دقت پیش‌بینی مدل بر داده‌های ارزشیابی تا ۷۱ بالا رفت اما هنگام آزمون با استفاده از داده‌های آزمون، نتایج بسیار نامطلوبی حاصل شد. تمامی امتیازات در پیش‌بینی داده‌های آزمون از ۵۵ کم‌تر بودند. در هر دوی این مدل‌ها از لایه‌های dropOut نیز استفاده شد.

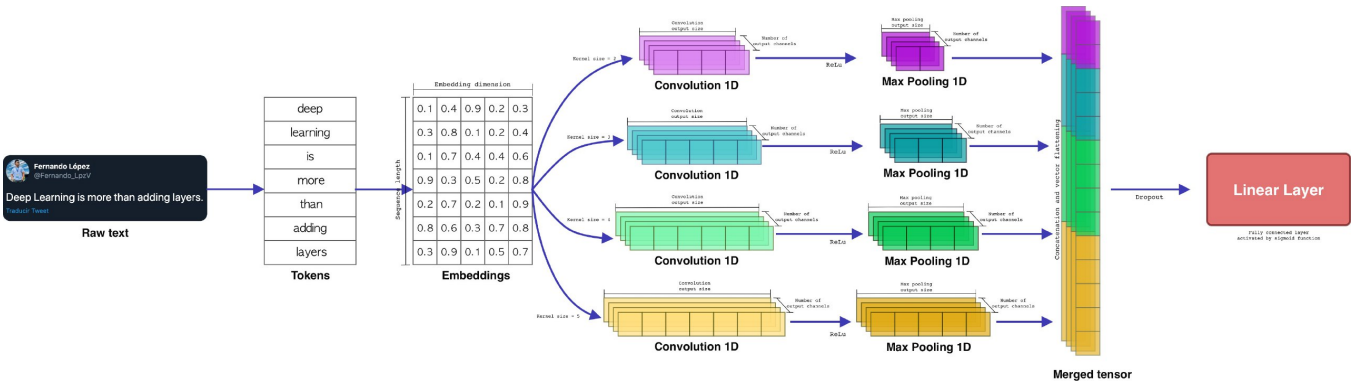
۴-۶ RNNs: GRU

این مدل فرایند یادگیری سریع‌تری نسبت به مدل LSTM دارد و از این جهت نسبت به آن مزیت دارد. تفاوت این مدل با مدل قبل در این است که به جای لایه‌های LSTM از لایه‌های GRU استفاده شده است. نمودار دقت مدل در شکل ۳۱ ترسیم شده و همان‌طور که پیداست این مدل نیز پیشرفت محسوسی نسبت به مدل‌های پیشین به دست نداده است.

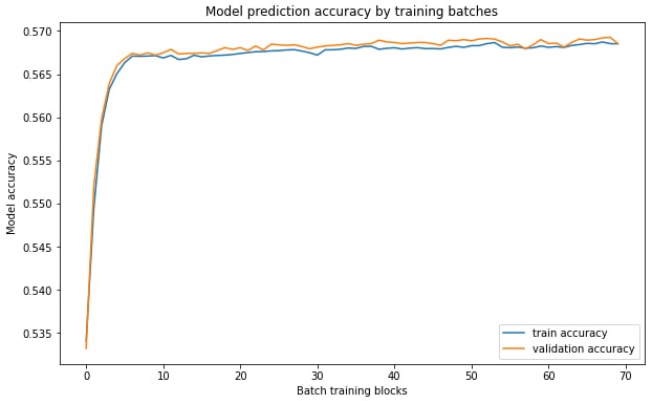
۷ نتایج نهایی

متاسفانه به علت در دسترس نبودن زمان بیشتر برای عیب‌یابی و بهبود مدل‌ها، نتایج بهتری حاصل نشد و بهترین F1score ای که می‌توانیم بر روی داده‌ی آزمون اعلام کنیم متعلق به نسخه‌ی بهبود یافته‌ی اولین مدل CNN است که به ۶۰ می‌رسد.

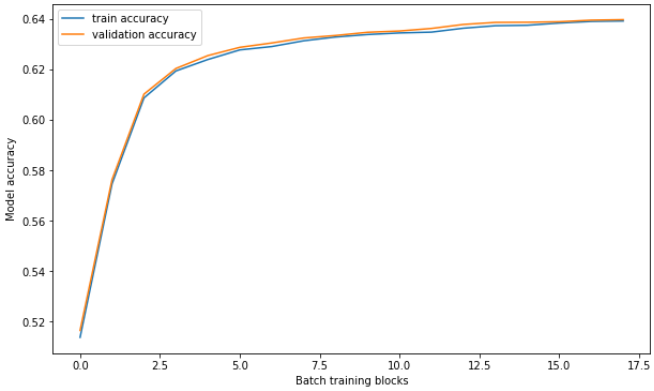
[12] confusion matrix
[13] Recurrent neural network



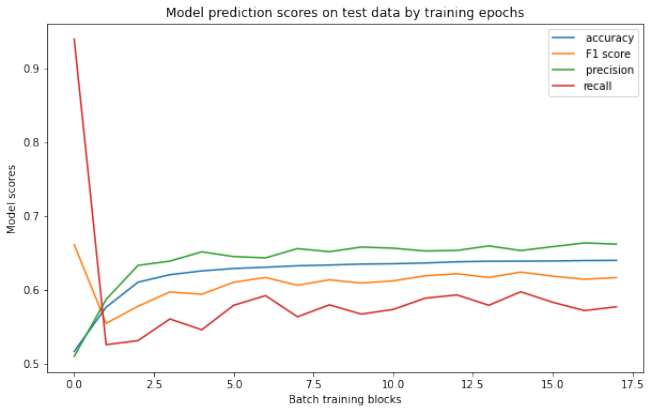
شکل ۲۴: معماری مدل CNN آموزش داده شده با بازنمایی Glove



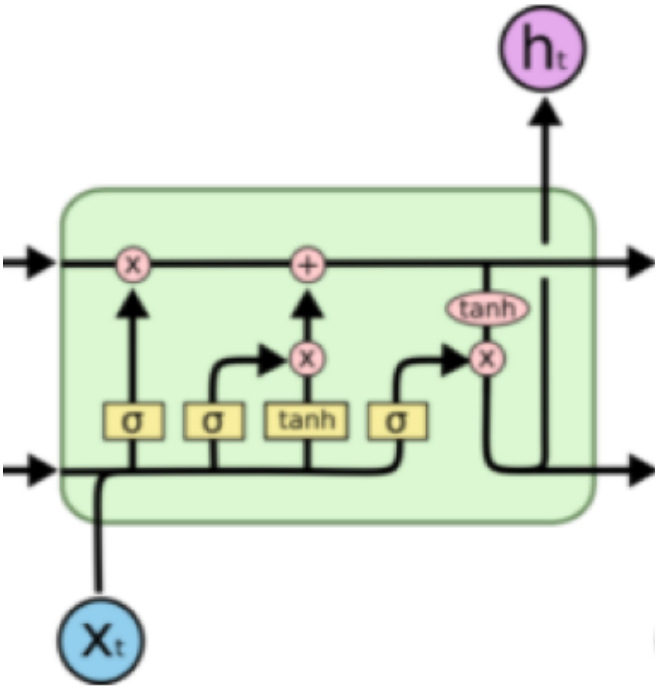
شکل ۲۷: نمودار دقت یک مدل CNN پس از تغییر ساختار اولیه



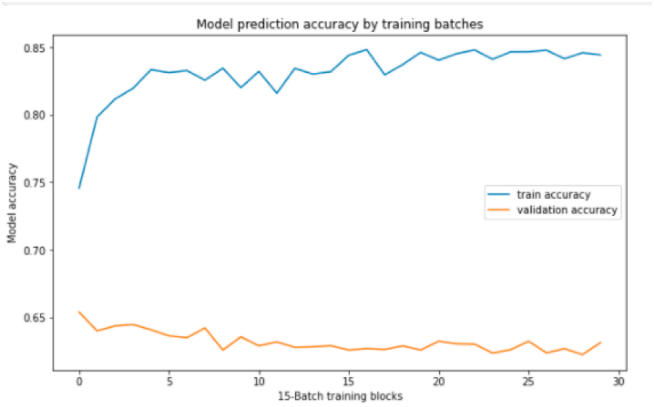
شکل ۲۵: نمودار دقت مدل CNN آموزش داده شده با بازنمایی Glove



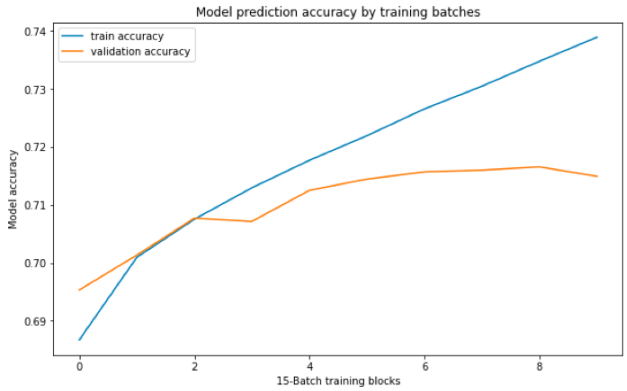
شکل ۲۶: نمودار امتیازات چهارگانه مدل CNN آموزش داده شده با بازنمایی Glove



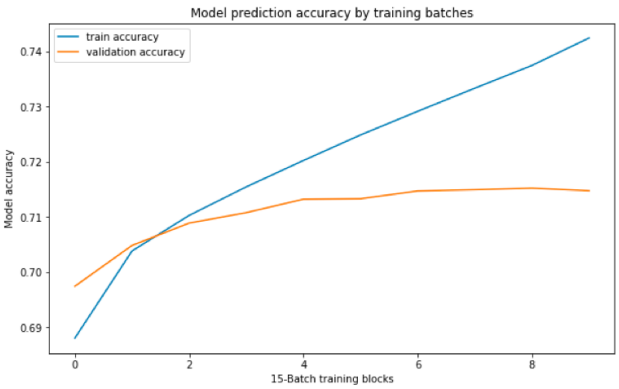
شکل ۲۸: نمونه‌ای از ساختار LSTM



شکل ۲۹: نمودار دقت مدل LSTM بر داده‌های یادگیری و ارزشیابی



شکل ۳۰: نمودار دقت مدل LSTM بر داده‌های یادگیری و ارزشیابی پس از استفاده از بازنمایی w2v مختص به کتابخانه‌ی tensorflow



شکل ۳۱: نمودار دقت مدل GRU