فهرست

فهرست اشکال

قسمت اول

مقدمه

بعضی اوقات، موضوعات مورد بحث در شبکه‌های اجتماعی مختلف، به صورتی ویروسی یا اصطلاحا viral گسترش پیدا می‌کنند و حتی ممکن است برای این اتفاق روند منطقی نیز به دست نیاید؛ اما به صورت کلی بررسی‌های عمقی و جزئی‌تر عمدتاً نتایج قابل توجهی را نشان می‌دهند. مشخصه‌ی این وقایع، محبوب بودن آن‌ها در سطوح مختلف است. به این معنی که بعضی به سرعت گسترش پیدا می‌کنند و سریعاً نیز فراموش می‌شوند در حالی که بعضی از آن‌ها ماندگار خواهند بود. در منظر رسانه معاصر، شبکه‌های اجتماعی در شکل‌گیری رویدادها یا محتوای viral نقش مهمی ایفا می‌کنند. توییتر و سایر سرویس‌های مایکرو بلاگینگ، منابع مهم اطلاعاتی در دنیای امروز وب محسوب می‌شوند. شناخت عواملی که بیش‌ترین نقش را در سرعت پخش شدن پیام‌ها ایفا می‌کنند، در تحلیل *Opinion Formation* تاثیر بسزایی دارند. بنابراین شناسایی عواملی که باعث می‌شود کاربری یک توئیت را اصطلاحاً ریتوئیت کند می‌تواند در پیش‌بینی‌های مرتبط با آن اثرگذار و حتی در مواردی سودده باشد. در واقع توئیتر، فضایی را برای ارائه‌ی نظرات روزانه و به روزرسانی‌های زندگی در قالب توییت ارائه می‌دهد. بنابراین، شناسایی مکانیسم viral شدن توییتر منتشر شده، می‌تواند برای سازمان‌هایی که در زمینه برندسازی و تأثیرگذاری با استفاده از رسانه‌های اجتماعی فعالیت می کنند، سودمند باشد. محبوبیت یا viral بودن یک توییت را می توان از نظر تعداد دفعات بازتوییت توییت اندازه‌گیری کرد که در این پروژه مورد بررسی قرار گرفته است.

مرور ادبیات

به طور کلی تحقیقاتی که در این زمینه انجام شده‌اند به سه دسته تقسیم می‌شوند: تحلیل ساختاری[۱]، تحلیل محتوا[۲] و تحلیل احساسات[۳]. در تحلیل ساختاری، هدف شناسایی کاربران تاثیرگذار است که می‌تواند به کمک محاسبه‌ی رتبه‌ی صفحه[۴]ی کاربر (با توجه به تعداد فالوورها و ...) [۱] و با استفاده از تعداد retweetها و منشن‌ها [۲] انجام شود. در تحلیل محتوا به این سوال پاسخ داده می‌شود که کدام توییت‌ها با احتمال بیش‌تری Viral می‌شوند. طبق تحقیق [۳]، تعداد منشن‌ها و URLهای یک توییت، جزو عوامل مهم در این زمینه هستند. در این حوزه ویژگی‌های استفاده شده در پیش‌بینی، می‌تواند تعداد فعل‌ها، اسم‌ها و یا صفت‌های یک توییت نیز باشد [۴]. تحلیل احساسات نیز، به دنبال کلاسه‌بندی توییت‌ها بر حسب احساسات منعکس شده در آن است (برای مثال مثبت، منفی یا خنثی).

در تحقیق [۵]، از مدل بیز و یک مدل ساده‌ی خطی، برای پیش‌بینی احتمال Viral شدن توییت با استفاده از ویژگی‌هایی مثل تعداد فالوورها و توییت‌های کاربر، متن توییت، تاریخ توییت و ... استفاده شده است و فرض شده اگر تعداد retweetها بیش‌تر از یک مقدار مشخص باشند (T)، توییت Viral محسوب می‌شود. نتایج نشان می‌دهد هر چه مقدار T بزرگ‌تر در نظر گرفته شود، ویژگی‌ها بهتر می‌توانند تمایز توییت‌ها را منعکس کنند. به علاوه، تعداد فالوورهای کاربر، طول توییت، تعداد هشتگ‌ها و منشن‌ها و نوع احساسات توییت جزو فاکتورهای مهم در تحلیل محتوا معرفی شده‌اند؛ برای مثال توییت‌ها با احساسات منفی با احتمال بیش‌تری Viral می‌شوند!

در تحقیق [۶] بررسی viral بودن به کمک الگوریتم ViralWatch انجام می‌شود که در سال ۲۰۱۵ منتشر شده‌است. به صورت خلاصه به هر موضوع در این روش امتیازی داده می‌شود و در همه‌ی شبکه‌های اجتماعی، عبارت‌های کاندید بررسی می‌شوند و هشتگ‌ها یا کلمه‌های کلیدی مرتبط شناسایی می‌شوند. (در مورد توییتر این روش تنها هشتگ‌ها را بررسی می‌کند). در نهایت امتیازات، معیار بررسی خواهد بود و با یک مقدار از پیش تعیین شده سنجیده می‌شوند.

در کل عوامل بسیاری می‌توانند در viral شدن یک محتوا در شبکه‌های اجتماعی موثر باشند. یکی از این عوامل که در کسب و کارها نیز بررسی می‌شود، به اشتراک‌گذاری آن محتوا در سایر شبکه‌های اجتماعی است. بعضی از منابع مواردی از جمله پیدا کردن حوزه‌ی niche برای محتوا، محتوای خلاقانه، زمان مناسب به اشتراک گذاری و تاثیرگذاری آن بر احساسات افراد (مانند یک فاجعه یا یک رویداد بسیار خوشحال کننده) را در viral شدن محتوا بسیار موثر می‌دانند. در کنار الگوریتم‌ها و مدل‌هایی که برای پیش‌بینی این موضوع در شبکه‌های اجتماعی مختلف استفاده می‌شود، تعدادی ابزار نیز به این منظور شکل گرفته‌اند که از بین آن‌ها می‌توان به Talkwalker Analytics، Display Purposes (بر مبنای هشتگ)، TweetDeck، OneMillionTweetMap و حتی در بعضی بررسی‌های سطحی Google trend اشاره کرد.

در نهایت می‌توان گفت که برای بررسی viral شدن یا نشدن یک توئیت، از دو دسته ویژگی واضح[۵] و نهفته[۶] استفاده می‌شود.

---

[۱] Structural Analysis
[۲] Content Analysis
[۳] Sentiment Analysis
[۴] PageRank
[۵] Obvious
[۶] Latent

قسمت دوم

قدم صفر

در ابتدا لازم است بعد از فراخوانی دیتاست، آشنایی اولیه و نسبی با آن حاصل شود. به این منظور از دستورات shape، columns، dtypes، value_counts استفاده شده است تا شکل کلی و تعداد سطرها و ستون‌ها، اسامی ستون‌ها و مقادیر منحصر به فرد آن‌ها شناسایی شود. همچنین دیکشنری‌هایی که می‌توان در مراحل بعدی از آن‌ها داده‌های مفیدی استخراج کرد، شناسایی شدند.

مرحله‌ی اول: پیش‌پردازش داده‌ها

در این بخش missing valueهای هر feature شناسایی و بر اساس تعداد missing valueها مرتب شده‌اند. اطلاعات این قسمت در دیتافریم data_info قابل مشاهده است. برای مثال کلیه‌ی داده‌های ستون contributors، نامشخص هستند.

در ادامه متن هر توئیت را به شکلی منسجم‌تر تبدیل می‌کنیم و موارد اضافی را از آن حذف می‌کنیم که برای این کار تابع clean_tweet تعریف شده است که متن توئیت را به عنوان ورودی می‌گیرد و اصلاحات لازم را بر روی آن اعمال می‌کند. در این تابع از کدهای regex برای کار با رشته‌ها استفاده شده‌است.

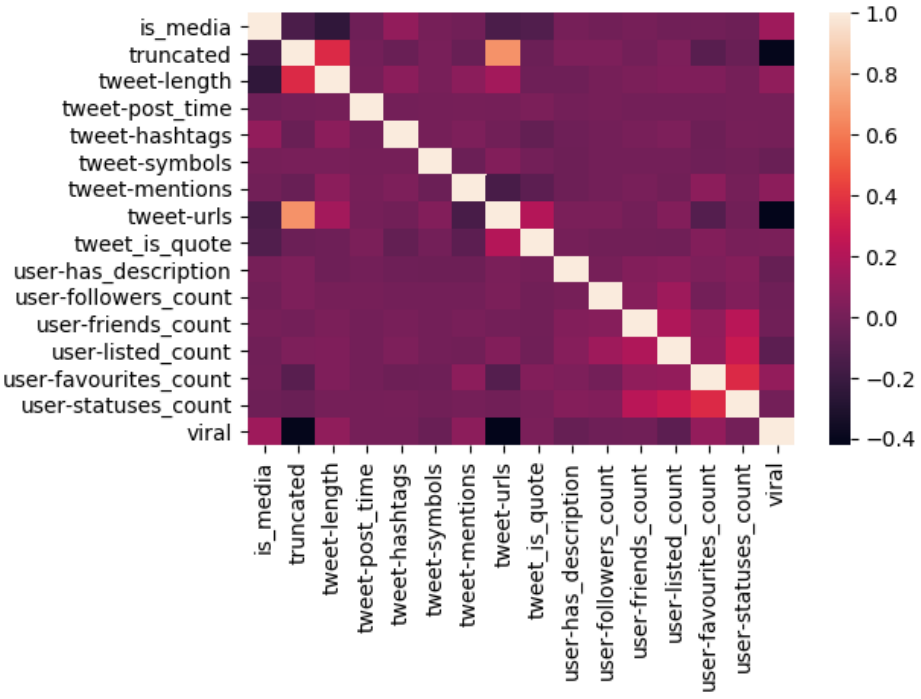
لازم به ذکر است در این پروژه، شناسایی داده‌های پرت پس از feature engineering انجام شده است. همچنین استاندارد کردن داده‌ها تاثیر مثبتی در مدل‌های ما نخواهد داشت و به همین دلیل از آن استفاده نشده است.

مرحله دوم: شناسایی و استخراج داده‌های موردنظر جهت داده‌کاوی (Feature Engineering)

در این مرحله داده‌هایی که از اهمیت بیشتری در پیش‌بینی ما برخوردارند شناسایی شده‌اند. مبنای این کار، حذف featureهایی با تعداد زیاد missing value، یا تعداد مقادیر منحصر به فرد[۷] بسیار کم و بررسی دقیق‌تر داده‌ها بوده است. برای این کار و همچنین استخراج featureهای قابل استفاده در دیکشنری‌هایی مانند "user" تابع clean_dataset تعریف شده است که دیتاست (test یا train) را به عنوان ورودی دریافت و تغییرات لازم را بر روی آن اعمال می‌کند. در این تابع ستون "viral" بر اساس تعداد ریتوئیت‌ها ساخته می‌شود و در صورتی که تعداد ریتوئیت‌های یک توئیت از میانه‌ی retweet count بیشتر شود آن را به عنوان viral شناسایی خواهد کرد.

همبستگی میان داده‌های نهایی در این بخش به صورت شکل ۱ خواهد بود. همان طور که دیده می‌شود همبستگی قابل توجه یا معناداری بین featureهای مختلف شناسایی نشده است و امکان حذف featureها به دلیل correlation بالا، نیست.

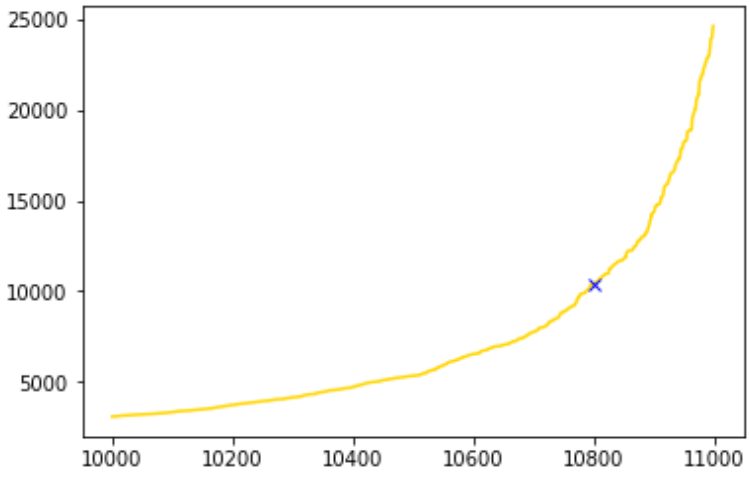
[۷] Unique values



شکل ۱ همبستگی میان ویژگی‌ها

در ادامه برای شناسایی داده‌های پرت، ابتدا از الگوریتم IQR استفاده شد که در پیوست آمده است، اما این الگوریتم مناسب ساختار داده‌های ما نیست. بنابراین شناسایی و حذف داده‌های پرت با روش DBSCAN انجام شده است. بهینه‌سازی مقادیر eps (۱۰۸۰۰) و min_samples (۳) انجام شد و در نهایت حدود ۳,۶ درصد از داده‌ها به عنوان داده‌های پرت شناسایی شدند.

برای بهینه‌سازی از متد ارائه شده در [۷] استفاده شد. در این روش فاصله‌ی نزدیک‌ترین نقطه از همه‌ی نقاط را پیدا می‌کنیم، فواصل به دست آمده را به شکل صعودی مرتب و در یک نمودار رسم می‌کنیم. جایی که این فاصله تغییر بزرگی داشته باشد، به عنوان مقدار بهینه‌ی eps انتخاب می‌شود. قسمتی از این نمودار برای داده‌های این پروژه که تغییر ناگهانی در آن رخ داده است، در شکل ۲ قابل مشاهده است.



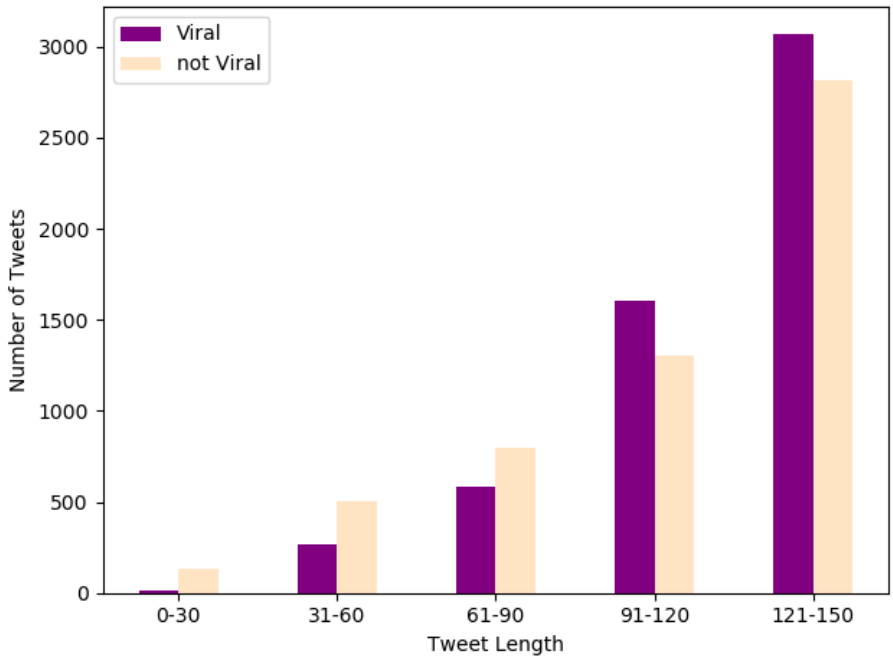
شکل۲ قسمتی از فواصل نزدیک ترین نقطه برای تمام نقاط

کل نمودار به این دلیل رسم نشده که به علت اختلاف زیاد مینیمم و ماکسیمم فواصل محاسبه شده، تغییر ناگهانی آن در نمودار کلی به درستی نشان داده نمی‌شود.

لازم به ذکر است که با کمک بردار predict، label تخصیص داده شده به هر رکورد قابل شناسایی است. در نهایت با حذف داده‌های پرت به دیتافریم X می‌رسیم که با استفاده از آن x و y را برای مدل‌های پیش‌بینی خود تعریف می‌کنیم.

مرحله سوم :تشریح داده‌ها

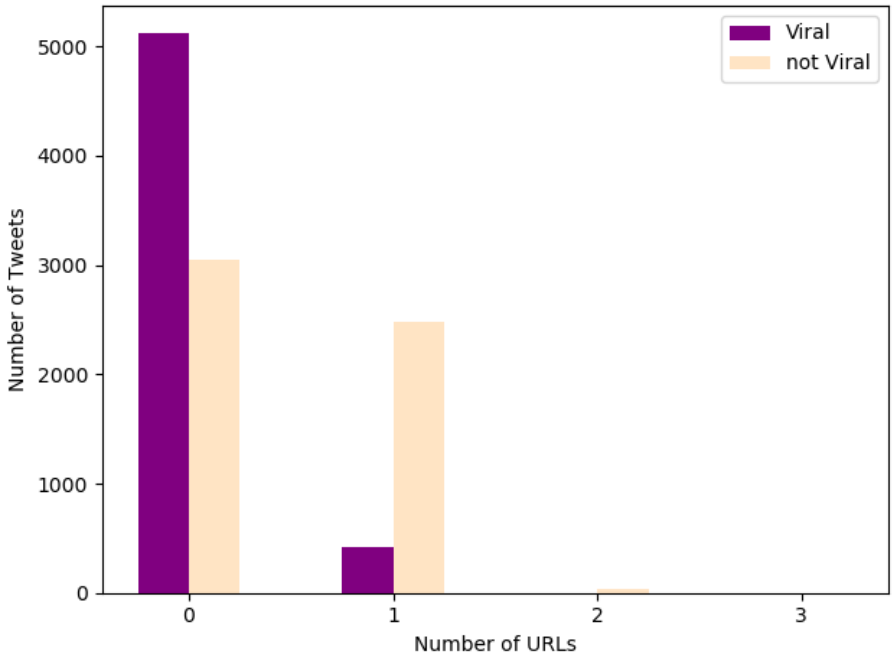
علاوه بر مواردی که در شناسایی داده‌ها در مراحل پیشین استفاده شده (مثل describe()، Heatmap و ...)، شکل‌های ۳ و ۴ نیز اطلاعات جالبی از داده‌ها در اختیار می‌گذارند.



شکل ۳ تعداد توییت‌های وایرال شده و نشده بر حسب طول توییت

در این نمودار، تعداد توئیت‌ها بر اساس طول آن‌ها و viral بودن یا نبودن آن دسته‌بندی شده‌اند. می‌توان مشاهده کرد برای پیام‌هایی با طول کمتر از ۹۰، تعداد توئیت‌هایی که viral نیستند بیشتر از پیام‌های viral است اما این روند برای پیام‌هایی با طول بیشتر از ۹۱ برعکس می‌شود. این نمودار تاثیرات طول توئیت در viral بودن را تا حدودی نمایش می‌دهد. این نمودار تحت عنوان Length در کد مربوطه آورده شده است.

در شکل ۴ به تاثیر URLها می‌پردازیم:



شکل ۴ تعداد توییت‌های وایرال شده و نشده بر حسب تعداد URLهای توییت

همان طور که در شکل ۴ دیده می‌شود، با تغییر از ۰ به ۱ در تعداد URL ها، شاهد کاهش شدید تعداد توئیت‌های viral هستیم. با تعداد ۲ URL تعداد توئیت‌های viral تقریبا به ۰ رسیده است. بنابراین به صورت کلی، URL ها به نظر تاثیرگذار می‌آیند.

مرحله چهارم :ارائه مدل جهت پیش‌بینی و ارزیابی آن

مدل‌های انتخابی برای پیش‌بینی viral بودن یا نبودن یک توئیت در این پروژه عبارتند از:

۱. Random forest
۲. Bagging
۳. Logistic regression

برای انتخاب مجموعه‌ای از ویژگی‌ها از بین ویژگی موجود، که بهترین عملکرد را داشته باشند، از روش RFE استفاده شد که یک متد Backward Selection است. در این روش با همه‌ی ویژگی‌ها مدل ساخته می‌شود و به هر ویژگی یک ضریب اختصاص داده می‌شود. سپس ویژگی با کم‌ترین ضریب حذف می‌شود. در مرحله‌ی بعد با ویژگی‌های باقی‌مانده مدلی ساخته می‌شود و رویکرد مرحله‌ی پیشین تکرار می‌شود. این کار آن قدر ادامه پیدا می‌کند تا به تعداد ویژگی‌های مورد نظر برسیم .

این ضریب اهمیت در مدل رگرسیون لجستیک همان ضرایب محاسبه شده برای هر ویژگی هنگام ساخت مدل است. در مدل رندم فارست نحوه‌ی محاسبه‌ی ضرایب ویژگی‌ها به این شکل است که در هر انشعاب هر درخت، بهبودی که در معیار ارزیابی انشعاب رخ می‌دهد محاسبه شده و مجموع این بهبودها برای همه‌ی انشعاب‌ها در همه‌ی درخت‌ها به عنوان اهمیت آن ویژگی در نظر گرفته می‌شود.

روش RFE تنها برای متدهایی قابل استفاده است که مقدار coef_ (مربوط به مدل‌های رگروسیونی) یا feature_importances_ (مربوط به مدل‌هایی که پایه‌ی آن‌ها درخت تصمیم است) برای آن‌ها محاسبه شود؛ لذا این روش برای bagging قابل استفاده نیست. از آن جایی که ویژگی‌های استفاده شده در مدل رندم فارست می‌تواند به عنوان پایه‌ای برای سایر مدل‌ها قرار گیرد، برای bagging از همین متد استفاده شد. به این صورت که بهترین featureها در نتیجه‌ی پیاده‌سازی رندم فارست به عنوان ورودی bagging استفاده و در نهایت با استفاده از cross validation، argumentهای مناسب این مدل شناسایی می‌شوند.

Random forest:

مقدار بهینه‌ی پارامترهای ورودی آن با استفاده از forهای تو در تو تعیین می‌شود (Gridy Search). در نهایت نتایج به دست آمده از این مدل در فایلی به نام RandomForest_df ذخیره می‌شوند که شامل آرگومان‌های تنظیم شده، featureهای انتخابی و میانگین accuracy و recall برای سنجش دقت مدل است.

در این مدل تعداد درخت‌ها (۱۰،۳۰) معیار ارزیابی (gini و entropy)، ماکسیمم عمق درخت (۵،۱۵) و مینیمم نمونه لازم برای split جدید ۵ و ۲۰، مینیمم تعداد برگ‌ها (۲،۱۰) را تغییر دادیم و با ساختن مدل برای تمام حالات ممکن بهترین مدل به شکل زیر تعریف می‌شود:

- تعداد درخت‌ها: ۳۰
- معیار ارزیابی: gini
- ماکسیمم عمق درخت: ۱۵
- مینیمم نمونه لازم برای split جدید: ۲۰
- مینیمم تعداد برگ‌ها: ۲

و ۱۴ feature نیز انتخاب می‌شوند:

۱. is_media
۲. truncated
۳. tweet-length
۴. tweet-post_time
۵. tweet-hashtags
۶. tweet-symbols
۷. tweet-urls
۸. tweet_is_quote
۹. user-followers_count
۱۰. user-friends_count
۱۱. user-listed_count
۱۲. user-favourites_count
۱۳. user-statuses_count
۱۴. user-created_at

Bagging:

از خروجی‌های موجود در فایل RandomForest_df استفاده می‌کنیم و در ادامه featureهایی که نیازی به آن‌ها نیست حذف می‌شوند. پارامترهای ورودی مدل n_estimators با مقادیر ۱۰ تا ۴۰ (با گام ۱۰) و bootstrap با مقادیر true/false با استفاده از for تعیین می‌شوند و در مدل قرار می‌گیرند. لازم به ذکر است که ۰,۲ داده‌ها به عنوان داده‌های تست تعیین شده‌اند. با توجه به این که bagging، مواردی مانند coef یا feature_importances ندارد، قابلیت استفاده از RFE برای آن وجود نخواهد داشت و به همین علت در این مدل استفاده نشده‌است. در نهایت نتایج به دست آمده مشابه مدل قبلی، در Bagging_df ذخیره می‌شوند. بهترین حالت مدل پارامترهای زیر را دارد:

- bootstrap: True
- n_estimator: ۴۰

و همان ۱۴ feature رندم فارست در این مدل استفاده می‌شوند.

Logistic regression:

با توجه به ماهیت ۰ و ۱ بودن viral ، روش رگرسیون لاجستیک با کمک RFE می‌تواند نتایج مناسبی داشته باشد. مقدار پارامتر penalty با استفاده از for، برای دو حالت l۱ و l۲ بررسی می‌شود. در نهایت نتایج به دست آمده مشابه دو روش قبلی در فایل LogisticRegression_df ذخیره شده است.

بهترین حالت این مدل دارای ۱۲ feature به صورت زیر و penalty = l۲ است.

۱. is_media
۲. truncated
۳. tweet-length
۴. tweet-post_time
۵. tweet-hashtags
۶. tweet-symbols
۷. tweet-mentions
۸. tweet-urls
۹. tweet_is_quote
۱۰. user-has_description
۱۱. user-listed_count

(تمامی فایل‌های ذخیره شده داخل پوشه‌ی محتویات پروژه قرار داده شده‌اند.)

مرحله پنجم: نتیجه گیری و تحلیل

در نهایت از دیتافریم info برای تجمیع نتایج بهینه‌ی به دست آمده از سه روش بالا استفاده می‌کنیم. این دیتافریم بهترین نتایج (بر اساس accuracy و recall) هر کدام از سه روش استفاده شده را در خود ذخیره می‌کند (شکل ۵).

	classifier	tuned_arguments	selected_features	Avg. Accuracy	Avg. Recall
0	LogisticRegression	{'penalty': 'l2'}	[Number of Features 11.0\r\nis_media ...	79.123711	88.439306
1	RandomForest	{'n_estimators': 30, 'criterion': 'gini', 'max...	[Number of Features 14.0\r\nis_media ...	85.567010	86.705202
2	Bagging	{'bootstrap': True, 'n_estimator': 40}	[Number of Features 14.0\r\nis_media ...	78.660436	82.325581

شکل ۵ مدل‌های نهایی

بنابراین مدل رندم فارست بهترین نتیجه را با مقدار متوسط accuracy، ۸۵٫۶٪ و متوسط recall ۸۶٫۷٪ به ما می‌دهد. که از ۱۴ feature زیر استفاده می‌کند. بعد از آن به ترتیب روش‌های bagging و در انتها logistic regression قرار می‌گیرند.

۱. is_media
۲. truncated
۳. tweet-length
۴. tweet-post_time
۵. tweet-hashtags
۶. tweet-symbols
۷. tweet-urls
۸. tweet_is_quote
۹. user-followers_count
۱۰. user-friends_count
۱۱. user-listed_count
۱۲. user-favourites_count
۱۳. user-statuses_count
۱۴. user-created_at

همان‌طور که دیده می‌شود، featureهای انتخاب شده، مجموعه‌ای از featureهای واضح مانند user-followers_count و featureهای پنهان مانند tweet-urls هستند.

پارامترهای بهینه مدل به صورت زیر هستند:

- 'n_estimators': ۳۰
- 'criterion': 'gini'
- 'max_depth': ۱۵
- 'min_samples_split': ۲۰
- 'min_samples_leaf': ۲

برای مشاهده‌ی دقیق‌تر مدل رندم فارست، ۴ درخت از این الگوریتم کشیده شده‌اند که عکس آن‌ها در فایل پروژه قرار داده شده است. در این۴ درخت تعداد URLها در سطح اول یا دوم به عنوان یکی از splitها انتخاب شده است. هم‌چنین دو ویژگی طول توییت (تعداد کاراکترهای آن) و truncated که آن هم به نوعی به طول توییت مرتبط است در سطح دوم یا سوم به عنوان یکی از splitها انتخاب شده است. این موارد نشان‌دهنده‌ی اهمیت تعداد URLهای توییت و طول آن در پیش‌بینی است که با توجه به شکل‌های ۲ و ۳ انتظار آن را داشتیم. نکته‌ی جالب دیگر این است که ویژگی user-friends_count در ۳ درخت از ۴ درخت

رسم شده، در سطح سوم یکی از splitها را تشکیل می‌دهد! این مورد را این‌طور می‌توان تفسیر کرد که هرچه تعداد دوستان کاربر بیش‌تر باشد، افراد بیش‌تری توییت را می‌بینند و احتمال وایرال شدن افزایش می‌یابد.

در نهایت از مدل رندم فارست با پارامترها و featureهای معرفی شده در قسمت‌های قبل برای پیش‌بینی viral بودن یا نبودن داده‌های تست استفاده شد که نتایج آن در فایلی به نام test_data قرار دارد. در این فایل ۵۴۴ پیام به عنوان پیام‌های viral پیش‌بینی شده‌اند.

منابع

[۱] Kwak, Haewoon & Lee, Changhyun & Park, Hosung & Moon, Sue. (۲۰۱۰). What Is Twitter, a Social Network or a News Media?. Proceedings of the ۱۹th International Conference on World Wide Web, WWW '۱۰. ۱۹. ۱۰,۱۱۴۵/۱۷۷۲۶۹۰,۱۷۷۲۷۵۱.

[۲] Cha, Meeyoung & Haddadi, Hamed & Benevenuto, Fabrício & Gummadi, Krishna P.. (۲۰۱۰). Measuring User Influence in Twitter: The Million Follower Fallacy.. AAAI Conference on Weblogs and Social Media. ۱۴.

[۳] Suh, Bongwon & Hong, Lichan & Pirolli, Peter & Chi, Ed. (۲۰۱۰). Want to be Retweeted? Large Scale Analytics on Factors Impacting Retweet in Twitter Network. IEEE. ۱۸۴-۱۷۷. ۱۰,۱۱۰۹/SocialCom.۲۰۱۰,۳۳.

[۴] Rowe, Mathew & Angeletou, Sofia & Alani, Harith. (۲۰۱۱). Predicting Discussions on the Social Semantic Web. ۴۲۰-۴۰۵. ۸-۲۱۰۶۴-۶۴۲-۳-۹۷۸/۱۰,۱۰۰۷_۲۸.

[۵] Jenders, Maximilian & Kasneci, Gjergji & Naumann, Felix. (۲۰۱۳). Analyzing and predicting viral tweets. WWW ۲۰۱۳ Companion - Proceedings of the ۲۲nd International Conference on World Wide Web. ۶۶۴-۶۵۷. ۲۴۸۷۷۸۸,۲۴۸۸۰۱۷/۱۰,۱۱۴۵.

[۶] Pöyry, Essi & Laaksonen, Salla-Maaria & Kekkonen, Arto & Pääkkönen, Juho. (۲۰۱۸). Anatomy of Viral Social Media Events. ۱۰,۲۴۲۵۱/HICSS.۲۰۱۸,۲۷۲.

[۷] Rahmah, Nadia & Sitanggang, Imas. (۲۰۱۶). Determination of Optimal Epsilon (Eps) Value on DBSCAN Algorithm to Clustering Data on Peatland Hotspots in Sumatra. IOP Conference Series: Earth and Environmental Science. ۳۱. ۰۱۲۰۱۲. ۱۰,۱۰۸۸/۱۷۵۵-۱۳۱۵/۳۱/۱/۰۱۲۰۱۲.

پیوست

کد مربوط به روش IQR:

```
col = x[['tweet-length','tweet-post_time','tweet-hashtags','tweet-symbols','tweet-mentions',
       'tweet-urls','user-followers_count','user-friends_count','user-listed_count','user-favourites_count',
       'user-statuses_count','user-created_at']]
col['retweet_count'] = data[['retweet_count']]

def outlier_detect(df):
    for i in df.describe().columns:
        Q١=df.describe().at['٢٥%',i]
        Q٣=df.describe().at['٧٥%',i]
        IQR=Q٣ - Q١
        LTV=Q١ - ١,٥ * IQR
        UTV=Q٣ + ١,٥ * IQR
        x=np.array(df[i])
        p=[]
        o=[]
        for j in x:
            if j < LTV or j>UTV:
                p.append('outlier')
                #p.append(df[i].median())
            else:
                p.append(j)
        df[i]=p
    return df

outlier_detect(col)
```