به نام خدا

گزارش فاز اول پروژه پایانی درس یادگیری ماشین

استاد درس: دکتر محمدحسین رهبان

پویا خانی

تابستان ۱۴۰۰

بخش ۱.۳ : در این قسمت برای بردار سازی به روش BOW از CountVectorizer کتابخانه Scikit learn استفاده کرده ام و بخاطر محدودیت رم سرویس کولب، از ۱۰۰۰ توکن با تکرار بیشتر برای ساخت vocabulary استفاده کردم. نتایج به صورت زیر شد(توجه شود من در گزارش فقط accuracy را نمایش میدهم و مابقی metric ها در نوتبوک آورده شده است):

حالت اول) بدون پیش پردازش متن :

Model and Parameters	Accuracy
LR with Solver=Saga & C=1	0.8584
LR with Solver=Saga & C=1.1	0.8584
KNN with default parameters	0.6182
KNN with n_neighbors=100 & weights = uniform	0.6565
SVM with default parameters	0.8517
SVM with C=0.01	0.6764

حالت دوم) با پیش پردازش ابتدایی :

Model and Parameters	Accuracy
LR with Solver=Saga & C=1	0.8584
LR with Solver=Saga & C=0.08	0.8584
KNN with default parameters	0.62
KNN with n_neighbors=100 & weights = uniform	0.6658
SVM with default parameters	0.8531

حالت سوم) با پیش پردازش سطح بالا :

Model and Parameters	Accuracy
LR with Solver=Saga & C=1	0.8592
LR with Solver=Saga & C=0.7	0.8593
KNN with default parameters	0.635
KNN with n_neighbors=100 & weights = uniform	0.698
SVM with default parameters	0.8594
SVM with C=0.01	0.7194

لذا در حالت بدون پیش پردازش داده ها و با بردارسازی BOW ، بهترین مدل مدل Logistic Regression با پارامتر های پیشفرض شد(دقت 0.8584) و در حالت با پیش پردازش ابتدایی نیز همین نتیجه حاصل شد(دقت 0.8584) و پیش پردازش تاثیری در دقت نگذاشت، اما در حالت با پیش پردازش سطح بالا، مدل SVM با پارامتر های پیشفرض بهترین دقت را خروجی داد و دقت از حالت بدون پیش پردازش، کمی بیشتر شد(دقت 0.8594) .

طبق خواسته صورت پروژه ، گفته شده که شرح کوتاهی از روش های استفاده شده در پیش پردازش سطح بالا بیاوریم. در پیش پردازش سطح بالا، ابتدا تمامی حروف انگلیسی به صورت lower case تبدیل میشود چون upper case یا lower case بودن حروف در مفهوم جمله و نتیجه ما بی تاثیر است و بخاطر دقت بهتر ابتدا همه حروف را lower case میکنیم که فرضا کلمه Bad و bad متفاوت نباشند. سپس اعداد را به روش Regular expressions حذف میکنیم(جایگذاری با جای خالی) زیرا اعداد اغلب مفهوم خاصی را نمیرسانند و بیشتر چیزی را به صورت کمی بررسی میکنند و بخاطر دقت بهتر، آن هارا حذف میکنیم. سپس کاراکتر هایی که کاراکتر انگلیسی نیستند و در متن وجود دارند را(مثل ‘ یا ’’ یا ٪ یا ...) به روش Regular expressions حذف میکنیم. حال یک متن حاوی کلمات انگلیسی داریم و نوبت به جداسازی کلمات یا به عبارتی، tokenization میرسد. برای اینکار از word_tokenize خود NLTK استفاده میکنیم.حال یک سری از توکن های جدا شده، احتمالا stop word ها مثل کاراکتر رفتن به خط بعدی یا شروع جمله یا... است که برای حذف آن ها از corpus هه NLTK به نام stopwords.words استفاده میکنیم که چون جملات انگلیسی هستند، از مجموعه انگلیسی آن استفاده میشود. برای این کار چک میکنیم که هر توکن اگر در این مجموعه وجود داشت، یعنی stop word است و باید حذف شود. سپس نوبت به stemming میرسد. این روش یعنی جایگزین کردن کلمات با ریشه شان. مثلا در متن مفهوم کلمه played و playing نباید متفاوت باشد و با روش Stemming هر دو کلمه به play تبدیل میشود. البته روش بعدی یعنی lemmatization هم کار مشابهی انجام میدهد اما کیفیت استخراج ریشه و شباهت یابی کلمات اش بهتر است و کلمه بی معنی تولید نمیکند اما روش Stemming ممکن است فرضا کلمه scooter را به scoot تبدیل کند چون er ته کلمه را مثل er کلمات supporter و player میبیند و فکر میکند با حذف آن ها ریشه کلمه یعنی support و play استخراج میشود اما برای کلمه scooter ، er به این معنی نیست و خودش یک اسم است و نباید er حذف شود که خب در روش Lemmatization این تشخیص صورت میگیرد و همه کلمات ، بامفهوم هستند.

بخش ۲.۳ : بردار سازی BOW که در قسمت قبل تحلیل شد و باز در جدول های زیر همان نتایج را کپی میکنیم. اما در این قسمت بردارسازی W2V هم انجام میشود و نتایج حاصل شده را در جدول زیر میبینیم:

Model and Parameter and Vectorize method	Accuracy
LR with Solver=Saga & C=0.7(BOW)	0.8593
KNN with n_neighbors=100 & weights = uniform(BOW)	0.698
SVM with default parameters(BOW)	0.8594
LR with Solver=Saga & C=1(W2V)	0.8668
KNN with n_neighbors=50	0.8332
SVM with C=1.28	0.8842

حال میبینیم که بهترین LR و KNN و SVM با بردارسازی W2V است و در کل بهترین مدل SVM است اما بخاطر اینکه در فاز دوم نیاز به fine-tuning آن داریم و طبق مطالعه ای که کردم،SVM.SVC در scikit-learn امکان fine-tune شدن ندارد، لذا بهترین مدل را دومین بهترین مدل یعنی LR با بردارسازی W2V ذخیره کردم.

بخش ۳.۳ : در این بخش برای پیش پردازش، از همان پیش پردازش سطح بالای سابق استفاده کردم و MLP با ۴ لایه مخفی به ترتیب شامل ۵۰۰ و ۲۰۰ و ۵۰ و ۱۰ نورون با تابع فعالساز ReLu استفاده شد که دقتی برابر با 0.862 به ما داد.