**Twitter Topic Modelling with Databricks Pyspark ML**

**Objective:** Our goal was to build a model to identify topics in a collection of Tweets. We wanted to see how well the model could infer a set of topics from an unlabelled collection of Tweets, and how coherently it could cluster the Tweets into groups based on their content.

**Data Source:** Our dataset was a collection of approximately 45,000 tweets obtained June 18, 2021 through Twitter’s Recent Search API, searching on: "Toronto", English language, retweets excluded.

**Key Challenge #1:** We were working with unlabelled data. There was no ‘ground truth’ to train the model on. It’s not practical to manually identify topics and labels for thousands of Tweets.

**Key Challenge #2:** Tweets are very short texts, full of misspelled words and hashtags that mash a group of words together. Tweets are written by many different authors in many different writing styles and often incoherent. So there is lots of noise and comparatively little content.

**Models:** We tried two models: Latent Dirichlet Allocation (LDA), and Word2Vec with K-Means Clustering. Implementations for both models are available in Pyspark ML.

**Data Preparation:** We cleaned the data by lowercasing and removing hyperlinks, twitter handles and non-alphabetical characters. We used PySpark ML feature extraction functions to tokenize (transform texts into lists of words) and remove stopwords (common words that are likely to occur frequently across all topics, e.g. ‘and’, ‘the’). We also replaced common 2-word phrases with a single token (e.g. replacing 'blue', 'jays' with blue\_jays').

**Model 1 - LDA:** This popular algorithm for topic modelling came out in 2003. Each topic is modelled as a probability distribution over words. For each document, LDA assigns a weight for every topic, which is the strength of that topic in the document, based on the proportion of words in the document associated with that topic. To train the model, the algorithm starts by randomly assigning a topic to each instance of each word in each document. It uses heuristic optimization methods (e.g., Gibbs sampling) to modify the topic assignments word by word to try to find a better set of assignments. An optimal set of assignments is one where most documents and most words are predominantly a single topic. The input to LDA is a matrix of the word frequencies in each document, which we obtained using Pyspark ML’s CountVectorizer transformer.

**Model 2 - Word2Vec with K-Means Clustering:** The intuition behind Word2Vec (published in 2013) is that if you train a binary classifier to predict whether any word in the vocabulary is likely to appear within a given window of another word, you can use the weights from the trained classifier as vector representations of the words. These are dense vectors, typically much shorter than the sparse word-frequency vectors fed into LDA. For Word2Vec’s vectors, the closer two vectors are to each other, the more likely the words are related (i.e., have similar meanings and/or are often used in the same context). So the distance can be used as a measure of relatedness. Pyspark ML's Word2VecModel transforms each document into a vector that is the average of all the word weights in the document. To identify ‘topics’ (i.e., groups of similar Tweets), we fed the vectorized Tweets into Pyspark ML’s K-Means clustering module. We normalized the vectors to unit length before running K-Means.

**Tuning the Models:** One key parameter in both models is K, the number of topics/clusters. For LDA, we used coherence scoring to evaluate different values of K. Coherence scoring is not available in Pyspark ML, so we used Gensim’s CoherenceModel for that. However, we found that the coherence scores were not all that helpful. We suspect that the existing topic coherence scoring models are less helpful with very short documents like Tweets. For LDA, we chose the value K=12 based on manual inspection of topic top words, even though the coherence score was highest for K=9. For K-means, we chose K=15 based on the silhouette score, which is available in Pyspark ML’s ClusteringEvaluator module.

**Evaluating the results:** Beyond coherence scores, topic models can be evaluated by manual inspection of the topic top words and the resulting groupings of Tweets into topics. We used pyLDAvis to produce visualizations, which can be seen at: <https://rkatzin.github.io> The topic top words for LDA are shown below:

**Topic Top Words – LDA**

Topic 1: police, community, man, toronto\_police, pride, canada, black, canadian, mohamed, video, yussuf, break, men, people, two, jewish, white, united, scarborough, case

Topic 2: ontario, toronto\_ontario, lockdown, canada, days, stay, covid, ontario\_canada, euro, toronto\_ontario\_canada, mentalhealth, drug, business, last, health, last\_year, home, order, year, family

Topic 3: ontario, toronto\_ontario, jays, new, blue, blue\_jays, posted, photo, posted\_photo, toronto\_blue, toronto\_blue\_jays, vaccine, york, photo\_toronto, posted\_photo\_toronto, yankees, photo\_toronto\_ontario, dose, second, gospel

Topic 4: leafs, maple, maple\_leafs, toronto\_maple, raptors, team, toronto\_maple\_leafs, sun, toronto\_sun, toronto\_raptors, season, spezza, nba, fan, first, fans, article, hockey, history, nhl

Topic 5: playing, amp, per, feat, june, company, fc, new, stock, one, love, inc, girl, cent, exchange, drake, cannabis, rock, cnw, globe

Topic 6: people, like, city, amp, us, right, june, hwy, please, via, need, st, many, get, one, public, rd, th, help, like\_toronto

Topic 7: im, area, canada, like, get, dont, back, one, general, edt, stn, dispatched, edt\_stn, dispatched\_edt, general\_area, dispatched\_edt\_stn, good, go, see, pumper

Topic 8: vs, boston, chicago, vegas, nyc, la, vs\_toronto, miami, ny, houston, detroit, tampa, ml, seattle, sox, cleveland, london, mlb, dallas, lowry

Topic 9: news, covid, time, first, hospital, cbc, care, says, today, since, free, patients, icu, months, health, torontos, via, tv, march, general

Topic 10: housing, new, amp, centre, torontos, home, plan, realestate, market, gta, k, buy, homes, b, building, condo, win, price, rent, downtown

Topic 11: summer, citynewstoronto, restaurant, art, opening, reopen, artist, perfect, patios, patio, outdoor, excited, congratulations, best, borders, world, announce, restaurants, quebec, manitoba

Topic 12: canada, ontario, cases, business, covid, real, latest, russia, home, toronto\_canada, lockdown, health, estate, new, real\_estate, covid\_cases, family, sales, open, reporting

In the above, some topics seem to make sense, e.g., Topic 3 looks like a baseball topic, although there are also baseball words in Topic 8 (sox, mlb). Topics 2, 9 and 12 look like COVID topics with slightly different flavours. Topic 10 looks like a real estate topic, but for some reason the actual 2-word phrase real\_estate appears in Topic 12.

**Findings and Conclusions:** Our impression was that Word2Vec with K-Means produced slightly more sensible groupings of Tweets, even though the coherence score was lower than with LDA. Word2Vec is designed to capture the context where words appear, so it could be better at identifying words that belong to the same topic even if they don’t occur in the same Tweet. The topics were far from perfect. In both models (but perhaps slightly more with LDA than Word2Vec with K-Means), lots of tweets seemed unrelated to the top words of the topic they were assigned to. Still, these models show that machine learning can be helpful to get a more structured sense of the content of a very large body of noisy texts. We found that Databricks Pyspark ML provides good functionality for many standard natural language processing tasks.