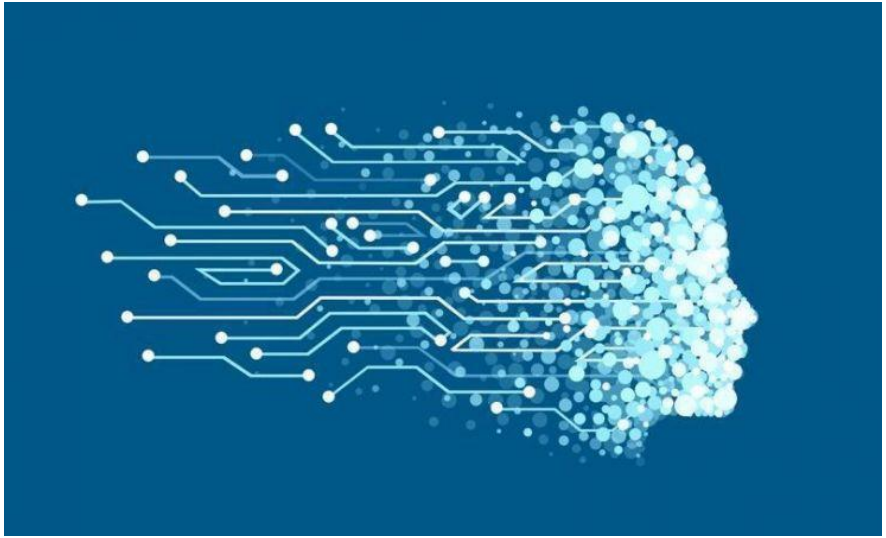
DA 250 / MDA 710 FINAL: NETFLIX SENTIMENT ANALYSIS

ARI BAZIGOS, CARTER EKBERG,
HUNTER LUCKOW, TYLER WELSH

Overview

- Significance and Objectives
- Data Preprocessing
- Exploratory Analysis
- Topic Modeling
- Machine Learning



Project Objectives

- Use tweets regarding Netflix to calculate consumer sentiment
- Observe whether stock price during the dip of 4/19/22 had an effect on tweet sentiment
- Utilize Topic Modeling to group tweets into different categories

Project Significance

- Netflix recently had the worst quarter in its history
- First time ever they lost more subscribers than gained
- Company stakeholders may begin to predict company sentiment from external sources.
- This may assist in public relations, customer satisfaction, and shareholder comfort.
    - If the company can gain a sense of what the masses are saying about the company via social media, they can better structure their business and applications to satisfy various parties.
- Not only applicable to Twitter data, may also be applicable to other social media platforms.

Data Sources

DATA PRE-PROCESSING AND DATA FRAME CONSTRUCTION

Loading Tweets

- snscrape using keyword “Netflix”
- 2000 tweets per day from April 11 - April 22
- Created a .csv file for each day
- Loaded all files into DataFrame below

	id	date	user	content	likes	retweets	quotes	replies	day
0	1513668213938307074	2022-04-11 23:59:59+00:00	https://twitter.com/simesanony2	@xtinethomas01 @KEdge23 So they have never watched the plethora of cooking programmes or seen the range of food on the internet.\n\nI accept they have probably never read a book and rely on a diet of Netflix and Facebook but it's all there for the finding.\n\nI don't buy the lack of knowledge, it's laziness	1	0	0	1	11
1	1513668179855220743	2022-04-11 23:59:51+00:00	https://twitter.com/IanMcGraw1000	@netflix Netflix you really have me staring at a clock the whole day	7	0	0	0	11

EXPLORATORY ANALYSIS

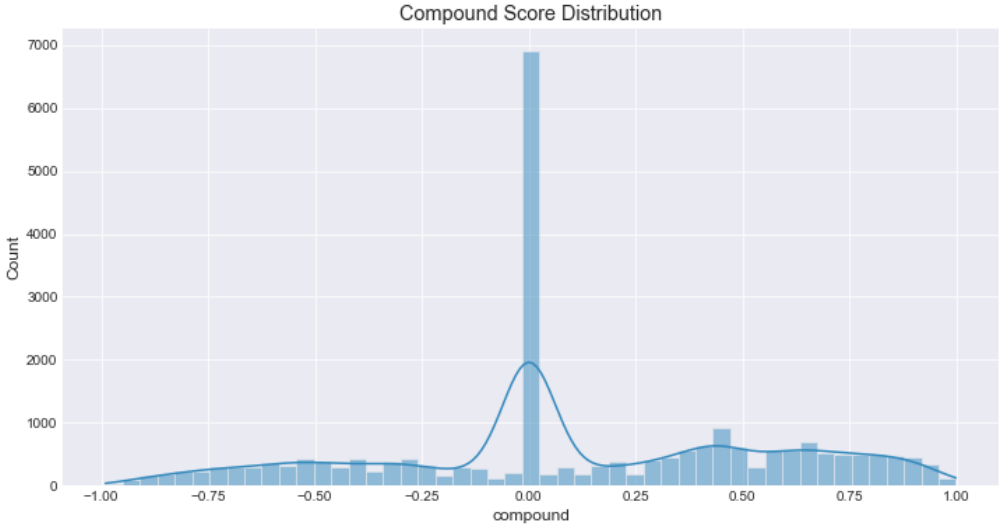
Calculating Sentiment Scores

- Applied VaderSentiment to calculate sentiment scores
- Each tweet given neg, pos, neu, and compound score
- Added scores to our DataFrame

	id	date	user	content	likes	retweets	quotes	replies	day	neg	neu	pos	compound
2	1513668177506574342	2022-04-11 23:59:50+00:00	https://twitter.com/King_Zulair	Netflix finally brought back Blade 😔	0	0	0	0	11	0.000	0.698	0.302	0.3818
3	1513668167662460930	2022-04-11 23:59:48+00:00	https://twitter.com/TheLastDiety32	Netflix shows be half-way thru production and get canceled 😞 like gahhhhh damn bitch	0	0	0	0	11	0.331	0.420	0.249	-0.3182

Removing Compound Scores of 0

- Removed because no net positive or negative with a compound score = 0.
- Felt that these tweets would not be beneficial to our algorithms
- Originally 24,000 tweets
- After removal, approximately 17,000 tweets



Sentiment Scores by Day

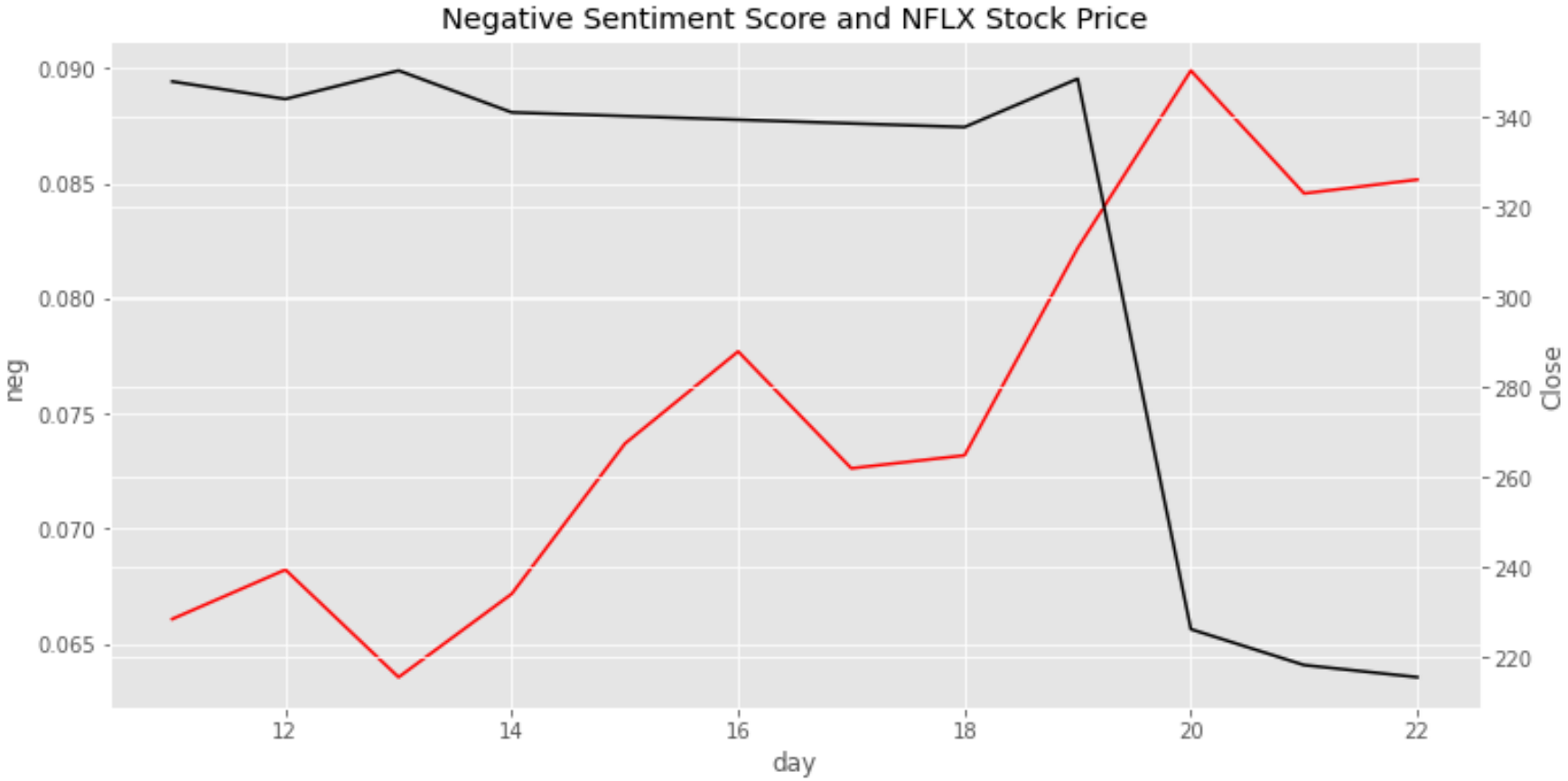
- Grouped by “day” to calculate mean scores for each day
- Now we can see a trend in scores over time
- Negative score stands out as we see an increasing trend.

day	neg	neu	pos	compound
11	0.066070	0.811103	0.122826	0.134012
12	0.068213	0.816968	0.114818	0.111923
13	0.063543	0.802457	0.133998	0.180205
14	0.067166	0.810852	0.121978	0.142359
15	0.073695	0.814461	0.111842	0.109852
16	0.077697	0.812575	0.109735	0.098675
17	0.072620	0.808624	0.118749	0.116685
18	0.073180	0.803624	0.123195	0.135079
19	0.082189	0.815173	0.102644	0.078409
20	0.089902	0.806432	0.103663	0.054481
21	0.084562	0.804063	0.111375	0.083004
22	0.085164	0.795063	0.119780	0.093277

Netflix Stock

- We then imported Netflix stock data from yahoo finance for the same date range

Date	Open	High	Low	Close	Volume	Dividends	Stock Splits	day
2022-04-11	350.000000	354.779999	345.200012	348.000000	3777100	0	0	11
2022-04-12	355.910004	359.410004	342.250000	344.100006	3824300	0	0	12
2022-04-13	343.920013	352.000000	341.160004	350.429993	3231000	0	0	13
2022-04-14	350.950012	352.000000	339.859985	341.130005	4338100	0	0	14
2022-04-18	340.000000	342.359985	331.619995	337.859985	5105000	0	0	18
2022-04-19	333.220001	351.679993	333.220001	348.609985	20906900	0	0	19
2022-04-20	245.199997	248.699997	212.509995	226.190002	133387500	0	0	20
2022-04-21	220.000000	227.679993	211.520004	218.220001	53501600	0	0	21
2022-04-22	220.179993	226.270004	210.050003	215.520004	37458500	0	0	22


Negative Sentiment Score and NFLX Stock Price
neg
Close
day
0.090
0.085
0.080
0.075
0.070
0.065
340
320
300
280
260
240
220
12
14
16
18
20
22

TOPIC MODELING

Cleaning Tweets

- Before Topic Modeling, need to “clean” text
- Remove hyperlinks, emojis, usernames, and punctuation.
- We utilized the nltk and re libraries
- Also some written functions found online

content	clean_tweet
@xtinethomas01 @KEdge23 So they have never watched the plethora of cooking programmes or seen the range of food on the internet.\n\nI accept they have probably never read a book and rely on a diet of Netflix and Facebook but it's all there for the finding.\n\nI don't buy the lack of knowledge, it's laziness	never watch plethora cook programm seen rang food internet accept probabl never read book reli diet netflix facebook find buy lack knowledg lazi
@netflix Netflix you really have me staring at a clock the whole day	netflix realli stare clock whole day
Netflix finally brought back Blade 😌	netflix final brought back blade 😌

Implementing Topic Modeling

- Imported Topic Modeling Algorithm “LatentDirichletAllocation”
- Clusters words frequently found in individual tweets.
- Made “Netflix” a stop word (cannot be a word in a topic)
- Set cluster frequency minimum to 20 instances
- Set number of Topics to 4, and cluster size to 6.

	Topic 0 words	Topic 0 weights	Topic 1 words	Topic 1 weights	Topic 2 words	Topic 2 weights	Topic 3 words	Topic 3 weights
0	watch	508.6	subscrib	364.6	ad	469.9	share	342.7
1	show	313.7	time	256.6	price	255.4	subscrib	207.9
2	like	256.2	first	235.9	plan	234.1	content	205.6
3	movi	228.1	stock	175.6	cancel	227.3	password	202.2
4	get	191.8	lose	149.5	stream	212.4	price	190.0
5	good	179.9	year	148.8	support	206.2	disney	175.6

MACHINE LEARNING PREDICTIVE MODELS

Constructing “neg_tweet” Variable

- Used chosen words from our Topic Modeling to flag in tweets
- If these words are in a tweet, we flag it as a “negative tweet”

```
days_no0 = days_no0.assign(neg_tweet=np.where((days_no0['content'].str.contains('lose') |
                                               days_no0['content'].str.contains('cancel') |
                                               days_no0['content'].str.contains('password') |
                                               days_no0['content'].str.contains('price') |
                                               days_no0['content'].str.contains('plan') |
                                               days_no0['content'].str.contains('stock') |
                                               days_no0['content'].str.contains('subscrib')), 1, 0)
```

id	date	user	content	likes	retweets	quotes	replies	day	neg	neu	pos	compound	score	neg_tweet
56	2022-04-11 23:58:36+00:00	https://twitter.com/SlimJenkins9	@netflix , its you or groceries at this point. These prices getting out of hand.	0	0	0	0	11	0.000	0.814	0.186	0.4939	0	1

Further Processing

- Chose 4 features and target column.
- 4 features include “neg”, “neu”, “pos”, and “neg_tweet” columns.
- Target is “score” column
- Target column is based off of negative and positive compound scores.
- Since scores are based on compound values, we removed the compound feature column as it would add too much bias to the models.
- Now we are ready to start developing our models.

Developing Train and Test Sets

In [94]:
```
df = days_no0[['neg', 'neu', 'pos', 'neg_tweet', 'score']]
```

In [95]:
```
df.head()
```

Out[95]:

	neg	neu	pos	neg_tweet	score
0	0.000	0.926	0.074	0	0
2	0.000	0.698	0.302	0	0
3	0.331	0.420	0.249	1	1
4	0.161	0.621	0.218	1	0
7	0.026	0.809	0.165	1	0

In [59]:
```
X=df.iloc[:,0:-1].values
y=df.iloc[:,-1].values
le = LabelEncoder()
y=le.fit_transform(y)
```

In [96]:
```
# Creating train & test sets.
X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    test_size=0.20,
                                                    stratify=y,
                                                    random_state=1)
```

- Created dataframe with features and target.
- Assigning feature columns to X and target column to y.
- Using sklearn train test split to develop train and test sets.

Perceptron

```
In [97]: ppn = Perceptron(max_iter=50, eta0=0.05, random_state=1)

In [98]: ppn.fit(X_train, y_train)

Out[98]: Perceptron(eta0=0.05, max_iter=50, random_state=1)

In [99]: y_pred = ppn.predict(X_test)

In [100]: accuracy_score(y_test, y_pred)

Out[100]: 0.9136774569845436
```

	Predicted Negative	Predicted Positive
Actual Negative	1829	304
Actual Positive	0	1296

- Perceptron model with 50 iterations through the data.
- Accurately predicted score 91.4% of the time.
- Confusion Matrix

Logistic Regression

```
In [101]: lr = LogisticRegression(C=100.0, random_state=1)

In [102]: lr.fit(X_train, y_train)

Out[102]: LogisticRegression(C=100.0, random_state=1)

In [103]: y_pred = lr.predict(X_test)

In [104]: accuracy_score(y_test, y_pred)

Out[104]: 0.9833770778652668
```

- Obtained accuracy of 98.3% using Logistic Regression

	Predicted Negative	Predicted Positive
Actual Negative	2104	29
Actual Positive	28	1268

- Confusion Matrix

SVM with Cross Validation

In [105]:
```
svc = SVC(kernel='linear', C=1.0, random_state=1)
```

In [106]:
```
svc.fit(X_train, y_train)
y_pred = svc.predict(X_test)
print(svc.score(X_test, y_test))
```

```
0.9813356663750364
```

In [107]:
```
kfold = StratifiedKFold(n_splits=10).split(X_train, y_train)
scores=[]

for k, (train,test) in enumerate(kfold):
    svc.fit(X_train[train], y_train[train])
    score=svc.score(X_train[test], y_train[test])
    scores.append(score)
    print('Fold: %2d, Class dis.: %s, Acc: %.3f' % (k+1, np.bincount(y_train[train]), score))

print('\nCV accuracy:%.3f +/- %.3f' % (np.mean(scores), np.std(scores)))
```

```
Fold:  1, Class dis.: [7675 4665], Acc: 0.988
Fold:  2, Class dis.: [7675 4665], Acc: 0.982
Fold:  3, Class dis.: [7676 4665], Acc: 0.985
Fold:  4, Class dis.: [7676 4665], Acc: 0.981
Fold:  5, Class dis.: [7675 4666], Acc: 0.985
Fold:  6, Class dis.: [7675 4666], Acc: 0.985
Fold:  7, Class dis.: [7675 4666], Acc: 0.987
Fold:  8, Class dis.: [7675 4666], Acc: 0.989
Fold:  9, Class dis.: [7675 4666], Acc: 0.977
Fold: 10, Class dis.: [7675 4666], Acc: 0.978

CV accuracy:0.984 +/- 0.004
```

In [108]:
```
from sklearn.model_selection import cross_val_score

scores = cross_val_score(estimator=svc,
                         X=X_train,
                         y=y_train,
                         cv=10,
                         n_jobs=1)
print(np.mean(scores))
```

```
0.9837366800424453
```

- Mean cross validation score of 98.4% with a standard deviation of 0.4%.

Conclusions

- Using Sentiment Analysis and Topic Modeling, we were able to effectively predict whether tweets would have a positive or negative score.
- Topic Modeling allowed us to choose relevant keywords to utilize in our Machine Learning models.
- The combination of chosen keywords by humans and calculated sentiment scores allowed our models to predict accurately.
- With help from Machine Learning, companies could enhance their customer and stakeholder satisfaction, if able to effectively predict overall market sentiment.

REFERENCES

- Investigate.ai
  - https://investigate.ai/bloomberg-tweet-topics/scrape-tweets-from-presidential-primary-candidates/
- Snscrape
  - https://urldefense.com/v3/__https://medium.datadriveninvestor.com/how-to-build-a-twitter-scraping-app-with-python-b3fc069a19c6*5Cn__;JQ!!DeIc-uvKXH9G!5T9CbaEqjUZdNgpKYalRx5XoU3s8gU2y6eKQSVr-6WS_2Q1CTrH0mP8asxTfF6kBC5BOfhqtUlGIcOeiDS_g2RciMuCe4iTT$
- Topic Modeling
  - https://urldefense.com/v3/__https://ourcodingclub.github.io/tutorials/topic-modelling-python/*5Cn__;JQ!!DeIc-uvKXH9G!5T9CbaEqjUZdNgpKYalRx5XoU3s8gU2y6eKQSVr-6WS_2Q1CTrH0mP8asxTfF6kBC5BOfhqtUlGIcOeiDS_g2RciMjseTSiN$
- Python Machine Learning (Sebastian Raschka)

THANK YOU