

Text Classification on NBA Tweets
Text As Data | Isaac Tabb | 09-03-2023

Dataset Discussion

(1a) My dataset is from Kaggle¹ and contains NBA tweets ranging from September 15th, 2020 to October 11th, 2020. The tweets are about the four NBA teams that made it to the Conference Finals and NBA Finals during that period (Boston Celtics, Denver Nuggets, Miami Heat, Los Angeles Lakers). The original dataset ranged from September 13th, 2020 to October 13th, 2020 and included all 30 NBA teams, but since the dataset was quite large (~18000 samples), I reduced the size by focusing on the dates of the Conference finals and NBA finals. The tweets were pulled using search queries with the format “[insert NBA team name]”. For example, many of the tweets were pulled using the query “#BostonCeltics”, and each of those tweets contains that hashtag. The team label of each tweet corresponds with the search query. I selected this dataset because of my interest in the relationship between social media and the NBA, particularly, how NBA teams are affected by social media sentiments. One possible application of automatic labeling would be collecting tweets about a certain team to see how they perform depending on general social media sentiments. The role of automatic labeling would be in the collection of data for teams.

(1b) The input text that will be used for classification is the content of each tweet. Since the tweets were pulled using a hashtag search query, each tweet originally included a #[insert team name] in the content of the tweet. To prevent these hashtags from affecting the model, I removed all hashtags from the tweets. This will make it easier to find other interesting patterns. The labels that will be used are the team names. Since the original data was about all 30 NBA teams, some preprocessing was employed to reduce these labels. I wanted to focus specifically on the four teams that were a part of the Conference Finals and NBA Finals so the labels are as follows: “BostonCeltics”, “DenverNuggets”, “MiamiHeat”, “LosAngelesLakers”.

(1c) The dataset was not already split into training, validation, and test so random sampling was employed to do so. Figure 1 shows that the distribution leans very heavily towards the Miami Heat (~72% of all cases). The remaining labels, LosAngelesLakers (~10%), DenverNuggets (~9%), and BostonCeltics (~9%) all have relatively similar percentages of the distribution. This distribution holds across all three datasets.

Figure 1: Label Counts

	Training (60%)	Validation (20%)	Test (20%)
MiamiHeat	4364	1446	1449
BostonCeltics	513	169	190
DenverNuggets	520	163	167
LosAngelesLakers	603	222	194

K-Means Clustering

(2a)

Cluster 1: Top 5 Tokens: ‘game’: .0743, ‘heat’: .0525, ‘laker’: .0439, ‘final’: .0432, ‘nba’: .0377


Examples:


1. Damn Denver I know that 1 hurt it reminds me of og hitting that 3 on the celtics last series
2. I’m rooting for the [hashtag] but if the [hashtag] win I won’t be mad, what @JimmyButler is doing...

¹ <https://www.kaggle.com/datasets/wjia26/nba-tweets?select=NBADataset+-+13-09-2020+till+13-10-2020.csv>

Cluster 2: Top 5 Tokens: ‘good’: .0880, ‘player’: .0292, ‘time’: .0285, ‘great’: .0188, ‘bam’: .0186

Examples:

1. Andre Iguodala has had a great year . <https://t.co/GYjlot3IL8>
2. Great defence on AD.


Cluster 3: Top 5 Tokens: ‘herro’: .0531, ‘jimmy’: .0509, ‘tyler’: .0392, ‘’: .0355, ‘butler’: .0331



Examples:

1. Idk what’s going on with all these missed 3s
2. TYLER HERRO!!!!


Cluster 4: Top 5 Tokens: ‘team’: .0712, ‘love’: .0321, ‘podcast’: .0216, ‘murray’: .0208, ‘nugget’: .0183

Examples:

1. First the Bucks, Now the Celtics.... that’s my top 2/3 teams out. take it away boys Miami in 7 
2. I’m always for the underdog, @MiamiHEAT have been my team...

Cluster 5: Top 5 Tokens: ‘’: .1104, ‘let’: .0998, ‘win’: .0488, ‘@miamiheat’: .0488, ‘’: .0435

Examples:

1. @Lakers @KingJames @AntDavis23 Thanks, @bakersfieldnow & @Jon_Singh19.

2. I don’t think The heat can win one more game. Well keep on thinking and let the Heat...

(2b) Based on question (2a), it is clear that these clusters make sense. The **first cluster** seems to often discuss the NBA finals matchup between the Miami Heat and the Los Angeles Lakers which can be seen in the magnitude of ‘laker’, ‘heat’, and ‘final’. Some of the other clusters, like the fourth cluster, have no top-5 tokens about the Miami Heat or Los Angeles Lakers. The **second cluster** includes tweets that talk about specific player’s success (or lack thereof) noting that ‘good’, ‘player’, and ‘great’ are all in the top-5 tokens. The examples of tweets in the second cluster also show users discussing gameplay in regards to specific players (ex. Andre Iguodala). For the **third cluster**, it is apparent that the tweets are about the Miami Heat. Four of the top-5 terms come from names of Heat players, Jimmy Butler and Tyler Herro. It seems that the **fourth cluster** discusses a mixture of the Denver Nuggets (‘nuggets’, ‘murray’) and sports media (‘podcast’), two very different topics. The terms ‘team’ and ‘love’ also have high magnitudes. It is possible that this cluster includes a mixed bag of tweets praising user’s favorite teams and sports media posts about teams. The **fifth cluster** appears to include tweets about the success of the Miami Heat. The fire emoji has the highest magnitude for this cluster, along with the terms ‘let’ and ‘win’. The Miami Heat’s twitter username also has a high magnitude.

(2c) Figure 2: Clusters Confusion Matrix

	Miami Heat	Boston Celtics	Denver Nuggets	LA Lakers	Total
Cluster 1	1381	210	188	328	2107
Cluster 2	416	58	60	85	619
Cluster 3	853	66	79	48	1046
Cluster 4	419	85	124	39	667
Cluster 5	1295	94	69	103	1561
Total	4364	513	520	603	6000

(2d) The confusion matrix in Figure 2 unearths many clear trends in the clusters. In the **first cluster**, the percentage of tweets about the Miami Heat (65.5%) is slightly less than it is for the entire dataset (72.7%) while the percentage of tweets about the Los Angeles Lakers (15.6%) is noticeably higher

than it is for the entire dataset (10%). As noted in question (a), the tweets in the first cluster seem to be about the Miami Heat vs. Los Angeles Lakers NBA Finals. The reason for the uptick in percentage for the Los Angeles Lakers and slight downtick for the Miami Heat may have to do with the Los Angeles Lakers winning the NBA finals. The **second cluster** doesn't show many substantial changes in distribution of labels in comparison to the distribution in the entire dataset. The Los Angeles Lakers have a slightly higher percentage (from 10% to 13.7%) and the Miami Heat have a slightly lower percentage (from 72.7% to 67.2%). In question (a) it was noted that this cluster discusses the players themselves. The uptick for the Los Angeles Lakers may be a result of their big-name players (ex. LeBron James & Anthony Davis). The downtick for the Miami Heat could be a result of the team's lack of big-name players (their second best player was rookie Tyler Herro). The **third cluster** picked up on tweets about the Miami Heat with 81.5% of the tweets in the cluster being labeled as such. As noted in question (a), four of the top five tokens in this cluster were names of Miami Heat players. The **fourth cluster** seems to have been able to pick up on the Denver Nuggets tweets. Although only 18.6% of the tweets in the cluster were about the Nuggets, the percentage of tweets about the Nuggets in the entire dataset is only 8.7%. The vast majority of this cluster though was still Miami Heat. Finally, the **fifth cluster** was able to pick up on tweets about the Miami Heat much like the third cluster did, with 82.9% of the tweets in the cluster being labeled as Miami Heat. This makes sense when looking at the top tokens from question (a) where the username of the Miami Heat's twitter account was one of the top tokens. Overall, the K-means clustering algorithm was not terrible at picking up on tweets about the majority class, the Miami Heat, but when it came to the minority classes, the algorithm did not fare as well.

Comparing Classifiers

(3a) As seen in Figure 3a, The Dummy Classifier with strategy "most frequent" was not well fit to the dataset (as expected). The classifier performed well accuracy-wise but this was due to class imbalance. The MiamiHeat label covers just over 72% of the documents in the dataset which explains the accuracy score. Disregarding accuracy, the macro-averaged precision, recall, and F1 scores were all far less than optimal. The Dummy Classifier with strategy "stratified" was also not well fit. The use of stratified random sampling caused the accuracy for the second Dummy Classifier to fall to 0.547. Additionally, the macro-averaged precision, recall, and F1 scores remained low as the classifier randomly chooses labels based on the distribution in the training set.

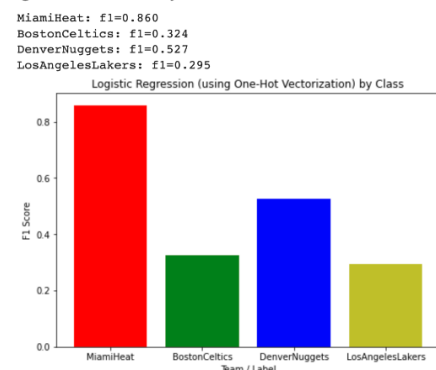
Figure 3a: Evaluation Metrics by Classifier

	Accuracy	Precision	Recall	F1
Dummy (Most Freq)	0.723	0.181	0.250	0.210
Dummy (Stratified)	0.547	0.254	0.253	0.253
LogisReg (One-Hot)	0.756	0.602	0.457	0.501
LogisReg (TFIDF)	0.760	0.662	0.419	0.473
SVC (One-Hot)	0.750	0.623	0.388	0.433

*Best performing classifier by macro-averaged F1 score highlighted in green.

Of the three non-baseline classifiers, the model with the best performance by macro-averaged F1 score was the Logistic Regression classifier using One-Hot Vectorization (**LR-OH**). Although LR-OH had the best performance, it was still not very effective. LR-OH yielded a relatively good accuracy score (75.6%) and an alright precision (60.2%) score but LR-OH's macro-averaged recall was quite subpar at 45.7%. Additionally, LR-OH yielded a poor F1 score of .501. One can likely attribute LR-OH's poor performance to the sparsity of the dataset. A

Figure 3b: LR-OH by Class



common flaw in Logistic Regression models is that they can overfit on sparse, high-dimensional datasets. Tuning the C parameter to apply stronger regularization could help with this issue. Additionally, the solver parameter could be adjusted to ‘saga’ which according to SciKit Learn performs well on sparse multinomial logistic regression as it provides L1 regularization².

The second best classifier in terms of F1 score was Logistic Regression using TF-IDF (**LR-TFIDF**). Although LR-TFIDF yielded good accuracy (76.0%) and precision (66.2%) scores, LR-TFIDF was worse than LR-OH in terms of both recall (41.9%) and F1 score (47.3%). The general reasons for LR-TFIDF’s poor performance are similar to that of LR-OH as Logistic Regression models perform poorly on sparse data. One might have expected the model using TF-IDF to perform better than its One-Hot counterpart though. One instance where One-Hot Vectorization can outperform TFIDF is when the documents are short, like tweets. TFIDF gives very little weight to stop words. If a tweet is only a few tokens long and the majority are stop words, with TF-IDF the tweet will only end up having two or three significant terms. Additionally, an advantage of TF-IDF is that repeated terms in a document can be weighted more heavily. This advantage does not apply with tweets since they do not often contain repeated terms.

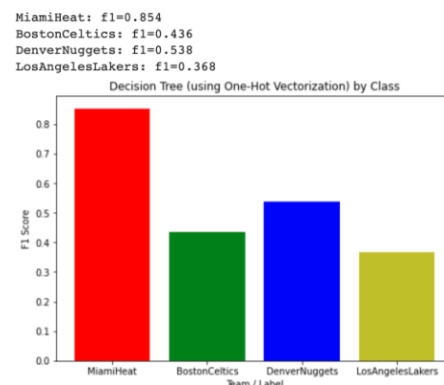
The third best classifier in regard to F1 score was the SVM classifier with One-Hot Vectorization (**SVM-OH**). Like the other two models the SVM-OH performed ineffectively overall. SVM-OH returned good accuracy (75.0%) and alright precision (62.3%), but performed the worst of all three classifiers in terms of recall (38.8%) and F1 score (43.3%). Although SVMs typically outperform Logistic Regression models, a common downfall of SVMs is that they do not work well with noisy datasets. Since the NBA is a game of matchups, many of the common features under each label overlap. SVMs draw optimized hyperplanes as boundaries between labels but this can prove challenging in a situation where the labels overlap often. Additionally, SVMs tend to struggle with imbalanced datasets. Tuning the C parameter would work as a regularization technique to combat the noise in the dataset while tuning gamma could prevent both underfitting and overfitting, respectively³.

(3b) The classifier and vectorization approach that was chosen to compare against the baselines was a Decision Tree using One-Hot Vectorization. A Decision Tree is a simple classifier which predicts labels based on decision rules generated from the training data. One parameter for Decision Trees is ‘min_samples_leaf’, which defines the minimum number of samples necessary at any leaf node. Decision Trees can struggle with sparse data but tuning parameters like ‘min_samples_leaf’ can help to avoid overfitting as leaf nodes cannot be highly specific (i.e. apply to only one or two samples)⁴. One-Hot vectorization was used as it can outperform TF-IDF when applied to shorter texts, like tweets (see 3a). As seen in Figure 3c, the Decision Tree outperformed all five of the baseline classifiers in terms of macro-averaged F1-score, with a

Figure 3c: Evaluation Metrics for Decision Tree (One-Hot)

Accuracy	Precision	Recall	F1
0.753	0.604	0.508	0.545

Figure 3d: Decision Tree (One-Hot) by Class



² https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression

³ <https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html>

⁴ <https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html#sklearn.tree.DecisionTreeClassifier>

score of .545. Additionally, the Decision Tree yielded a relatively good accuracy score of 75.3% but less than optimal precision at 60.4%. Notably, the classifier had the highest recall score of all of the classifiers at 50.8%. Although the Decision Tree performed better than the baselines, an F1-score of .545 is still subpar. One flaw of Decision Trees is that they do not always perform well with imbalanced data. As seen in Figure 3d, the Decision Tree yielded a very strong F1-score for the majority class Miami Heat but struggled with classifying the minority classes.

Parameter Tuning

(4) To tune the parameters of the LR-TFIDF classifier, two stages were employed. The first stage sequentially ran through each of the parameters, tuning them one by one. For each parameter, the settings which yielded the best results were noted for the second stage. In the second stage, a parameter grid was employed to test all possible combinations of each parameter's **best** settings.

The first parameter used was TFIDF vectorizer's **sublinear_tf**. The two possible values for `sublinear_tf` are True or False. Both settings yielded the same F1-score of 0.473. The purpose of `sublinear_tf` is to make it so that term occurrence in a document is not equal to term importance. In tweets, terms do not repeat very often, so this setting does not heavily affect the classifier performance. The second parameter was TFIDF vectorizer's **max_features**. `Max_features` was tested on values ranging from 315 to 10395 (i.e. the vocabulary size). Many of the F1-scores hovered around 4.7 but the lower `max_features` values (between 630 and 2520) yielded scores in the 4.8s and 4.9s. The tweets dataset is quite sparse and many terms only occur one or a couple of times. Logistic Regression tends to overfit on sparse data so removing these infrequent terms could combat this issue. The third parameter used was the **C regularization** parameter. Values ranging from 10^{-3} to 10^5 were used (incrementing the exponent). The values 10^1 and 10^2 yielded F1-scores of 0.515 and 0.500, respectively. At first, this seems counterintuitive since the data is sparse and thus should need more regularization. It is important to note though that less regularization means that we are giving more weight to the training set (i.e. allowing it freedom to overfit). Since the training and validation sets are two disjoint subsets of the same NBA tweets dataset, one can assume that the validation set is somewhat similar in nature to the training set. A classifier that gives more weight to the training set will perform well on a validation set that is quite similar. The final parameter used was the Logistic Regression **solver**. The solvers used were 'lbfgs', 'liblinear', 'newton-cg', 'sag', and 'saga' and all of them performed equally except 'liblinear'. The 'liblinear' solver likely performed poorly because it is the only one of the five that cannot handle multinomial loss. Being able to handle multinomial loss is vital to multi-class classification performance.

When running the parameter grid in the second stage, the top result yielded an F1-score of 0.5172 (Figure 4), an increase from the score of 0.473 yielded by the baseline Logistic Regression classifier. The settings for the top result were `sublinear_tf=True`, `max_features=945`, `C=10^1`, and `solver='lbfgs'`. Additionally, the second top result yielded a nearly identical F1-score of 0.5169 with the same settings except with `sublinear_tf=False`. The reduction of `max_features` to 945 shows that reducing the sparsity of the dataset improves the classifier performance. Additionally, with less features the dataset is less likely to overfit, therefore there is less need for regularization. This may explain why a higher value of C was effective in improving the classifier performance.

Figure 4: Logistic Regression (w/ TFIDF)

Accuracy	Precision	Recall	F1
0.754	0.599	0.475	0.517

`sublinear_tf=True`, `max_features=945`,
`C=10^1`, `solver='lbfgs'`

Context Vectors using BERT

Figure 5a: Feature Extraction Pipeline w/ RoBERTa-Base

Accuracy	Precision	Recall	F1
0.726	0.505	0.273	0.256

Figure 5b: Fine-Tuned Model w/ RoBERTa-Base

Accuracy	Precision	Recall	F1
0.723	0.181	0.250	0.210

Figure 5c: DistilBERT-Base-Uncased Fine-Tuned on SST2

Accuracy	Precision	Recall	F1
0.752	0.517	0.374	0.398

Figure 5d: RoBERTa-Base w/ learning_rate=1e-5, batch_size=32, epochs=3

Accuracy	Precision	Recall	F1
0.723	0.181	0.250	0.210

Figure 5e: DistilBERT-Base-Uncased Model Fine-Tuned on SST2 w/ epochs=10

Accuracy	Precision	Recall	F1
0.770	0.633	0.538	0.576

(5a-c) The **first custom model** was the DistilBERT-Base-Uncased model fine tuned on the SST2 dataset. The motivation for this choice came from the HuggingFace documentation which stated that the model performed well on topic classification⁵. The model uses the parameters: model=distilbert-base-uncased, learning_rate=1e-5, batch_size=32, and epochs=3. The **second model** was the RoBERTa-Base model using the same parameter settings as the DistilBERT-Base-Uncased model (learning_rate=1e-5, batch_size=32, epochs=3). The motivation behind this choice was to see if using the DistilBERT parameters on the RoBERTa-Base model would improve the classifier's performance. The **third model** was the DistilBERT-Base-Uncased model with the same parameters, except with epochs increased to 10. The motivation was to see if the model continued to improve as epochs increased or if the classifier performance would plateau.

(5d) The approach that performed best was the end-to-end DistilBERT-Base-Uncased model. As seen in Figure 5e, the DistilBERT model which used 10 epochs yielded the best F1 score of any classifier so far at 0.576. The DistilBERT which used 3 epochs yielded an F1 score of 0.398 (Figure 5c), which while that score is very poor, it was significantly better than the pipeline approach or the RoBERTa end-to-end model. The pipeline approach yielded a very poor F1 score of 0.256 (Figure 5a). The pipeline approach makes inferences about the NBA tweets based on the pre-trained roberta-base model, which is trained on Wikipedia and book text. The end-to-end models expand upon the pre-trained roberta-base model by training on the NBA tweets dataset as well. Wikipedia and book text are often not comparable to tweet text, thus additional training on the NBA tweets is necessary for classifier performance. The DistilBERT end-to-end model strongly outperformed the RoBERTa end-to-end model as well. According to HuggingFace, DistilBERT-Base-Uncased was pretrained on raw text only, meaning that humans did not label the text. As a result, DistilBERT can function very well on publicly available data⁶, explaining why DistilBERT performed so well on the tweet dataset. Additionally, the parameter values used for the DistilBERT models were from a fine-tuned model that was meant to be used for topic classification.

⁵<https://huggingface.co/distilbert-base-uncased-finetuned-sst-2-english>

⁶<https://huggingface.co/distilbert-base-uncased>

Conclusion & Further Work

(6b) Many patterns and trends arise when viewing the confusion matrix in Figure 6b. Since the training dataset contained so many tweets about the Miami Heat, the model was able to understand many of the common **Miami Heat** tweet features. Some Miami Heat tweets were predicted incorrectly as Boston

Figure 6b: Test-Set Classification Confusion Matrix

	Predicted Label			
	Miami Heat	Boston Celtics	Denver Nuggets	LA Lakers
Miami Heat	1308	47	14	80
Boston Celtics	99	78	1	12
Denver Nuggets	62	0	92	13
LA Lakers	108	0	19	67

Celtics and Los Angeles Lakers tweets. This is likely because the Miami Heat played against these two teams in the playoffs. In terms of **Boston Celtics** tweets, the model predicted the majority of the tweets as Miami Heat. These misclassifications can likely be attributed to the Heat vs. Celtics matchup in the Conference Finals. Since there are so many more tweets about the Miami Heat, when the model sees a Boston Celtics tweet about the Heat vs. Celtics matchup, it likely latches on to the Miami Heat label since that label is more common overall. In regards to **Denver Nuggets** tweets, the model performed pretty well, predicting more Nuggets tweets correctly than as Miami Heat tweets. Since the Nuggets never played the Heat in the playoffs, Nuggets tweets were somewhat distinguishable from tweets about the Miami Heat. A few Denver Nuggets tweets were classified as Los Angeles Lakers as well, which can probably be attributed to the Nuggets vs. Lakers matchup during the playoffs. The model struggled with **Los Angeles Lakers** tweets, predicting many of them to be about the Miami Heat. Similarly to the Boston Celtics, this is likely a result of the Lakers vs. Heat NBA Finals matchup. The Lakers had a bit of crossover with the Denver Nuggets as well which makes sense since these teams also played each other.

To analyze these errors, a few specific tweets corresponding to each common mislabeling were printed. For the **Miami Heat**, tweets labeled as Boston Celtics appeared to be about the Heat vs. Celtics matchup, with one example reading “This guy..... I mean he’s not wrong ! @celtics @MiamiHEAT”. Miami Heat tweets that were mislabeled as Los Angeles Lakers also appeared to be about the matchup between the two teams. One example read, “1 remaining from Lakers to the championship 🏆👇 Click the link for details👇🏆 https://t.co/vZ9rPE0Ejm”. For the **Boston Celtics**, tweets labeled as Miami Heat seemed to be about the Celtics vs. Heat matchup. One tweet read, “Let’s Go !!!! https://t.co/Ix1WW6MSsh”, where the link led to a picture of a man in a Celtics cap. In regards to the **Denver Nuggets**, tweets mislabeled as Miami Heat may have just been too ambiguous for the model to understand they were about the Nuggets. For example, one tweet read, “The [hashtag] have taking control of the first half leaving the [hashtag] dumbfounded. It's Halftime. Score 63 - 53 https://t.co/wSUGftjMLz”. Since tweets like these vaguely discuss the NBA, the model chooses the Miami Heat since that is the label it knows best. Some Nuggets tweets were also mislabeled as Lakers because of the matchup between the two teams, for example, “What is your prediction for the Game 4? Comment your prediction! https://t.co/MV9N0HEpVD”. For the **Los Angeles Lakers** tweets, it was clear that Miami Heat mislabels were a result of the Heat vs. Lakers Finals matchup. One example read, “Chris Paul explains why he picks the Lakers to win the NBA Finals vs the Heat https://t.co/Ea6i0zqswx”. Some Lakers tweets were also misclassified as Denver Nuggets as well. The examples appeared to be about the Lakers winning the championship though, so the reasoning behind the mislabeling is likely because Nuggets tweets often also discussed the Lakers.

(6c) At the beginning of the report, it was discussed how a classifier like this could be used to unearth relationships between the NBA and social media. Possible uses would be automatic labeling of tweets to collect data for analyzing social media sentiments towards certain teams. With how the model currently performs though, questions like these could not be accurately answered. The confusion matrix shows that the model often struggles to distinguish between teams that are commonly mentioned hand-in-hand. If someone were to analyze sentiments about specific teams, both false positives and false negatives could impact the accuracy of the analysis. For example, with the Miami Heat, a false positive would be one of the 108 Los Angeles Lakers tweets classified as Miami Heat. Mislabeling these tweets would mean analyzing sentiments about the Los Angeles Lakers and applying them to analysis of the Miami Heat. A false negative would be one of the 80 Miami Heat tweets that were labeled as Los Angeles Lakers tweets. While this mislabelling would not add noise to the Miami Heat analysis, it would discard helpful data for analyzing sentiments about the Miami Heat. In short, for this classifier to be deployed, it would need to more accurately distinguish which team is the focus of a tweet, especially in the very common case where multiple teams are mentioned together.

Figure 6a: DistilBERT-Base-Uncased Fine-Tuned on SST2 w/ epochs=10

Accuracy	Precision	Recall	F1
0.769	0.628	0.545	0.577

(6d) It is quite unlikely that the deployment of this system could have any negative societal effects. The most negative effects that could come from the deployment of this system would relate more to the players rather than society as a whole. For instance, say that the Los Angeles Lakers analyzed sentiments in tweets about their team and found that their star player Anthony Davis performs worse when social media is sending hate towards him. The Lakers might try to hide Davis from the media, but it would be far-fetched to expect the team to attempt to trade Davis or reduce his playing time. If the use of a classification model like this were to become popular in sports though, one possible negative effect could be a minor environmental impact. Sports teams are businesses and often they are willing to pay a large sum of money to improve their team. If every major sports team were to start analyzing social media sentiments about their team to understand how social media is affecting their performance, the combination of all of those systems could take up a lot of energy. The amount of energy that these systems would take up though in comparison to much larger neural network systems would be miniscule.

(6e) The main step to improve the classification effectiveness of this system would be improving the dataset. A lot of the reason that the classifier struggled was due to the class imbalance. Expanding the dataset to a uniform distribution of labels would likely improve the classifier's performance. For example, this would provide more data for the classifier to understand what a Boston Celtics tweet looks like. Additionally, expanding the range of time that the dataset covers would likely improve the classifier. The dataset only includes tweets from the NBA Finals and Conference Finals of the 2020 season. If this classifier were to ever be deployed it would need to correctly classify tweets about teams no matter the time frame. Including tweets ranging across multiple seasons would provide a more comprehensive depiction of what a tweet looks like for a certain NBA team. Finally, another improvement would be changing the labeling method of the dataset. The labels of tweets are simply based on the query used to pull the tweet from Twitter. To improve the dataset, manual labeling would need to be utilized to provide more accurate labeling on what team is the focus of a tweet.

(6f) This coursework took me about 30 hours.