Text Classification of Milwaukee Bucks's Fan Comments on Reddit

John Collopy

April 30 2024

Abstract

This project consisted of designing a Bernoulli Naive Bayes model, a Logistic Regression model, a Convolutional Neural Network, and a BERT model to classify Reddit comments in the r/MkeBucks subreddit according to thread label. The models were evaluated according to their F1 score, while precision and recall were analyzed as well. These metrics indicated that the Bernoulli Naive Bayes model performed best on both the validation and testing data; however, the model's relatively high number of false positive classifications speak both to the difficulties of classifying social media data as well as definitively determining the superiority of any model concerning this text classification problem.

1 Introduction

The goal of this project was to evaluate the performances of different text classification methods on domain-specific social media data. The data used are comments from Milwaukee Bucks's fans in post-game Reddit threads from the subreddit r/MkeBucks, and models were used to predict whether a comment in a post-game thread followed a win or a loss. The data consisted of over 9,000 comments following 64 games, at which point the Milwaukee Bucks had a record of 41 wins and 23 losses. Classifying social media content can be difficult due to the use of informal language, lack of context, and ambiguity, and in this specific domain, there was concern that these problems might be exacerbated in a space of passionate fans. There are questions as to which methods best handle such difficulties, and so the project will consist of implementing and evaluating different text classification methods on the Reddit comments mentioned above. For this project, Bernoulli Naive Bayes, Logistic Regression, a Convolutional Neural Network, and a BERT model were used for the classifications. The respective performances of these different methods might be of use to others searching to perform text classification on domain-specific social media content.

2 Background/Related Work

2.1 Paper 1

Kanish Shah and other contributors in their paper, "A comparative analysis of logistic regression, random forest and KNN models for the text classification." analyze the performances of different machine learning approaches to text classification problems. The data used in their project is BBC news articles. The project in this analysis also compares different approaches to text classification, but specifically in regards to social media data, rather than news articles.

2.2 Paper 2

Santiago González-Carvajal and Eduardo C. Garrido-Merchán in their paper, "Comparing BERT against traditional machine learning text classification." explore the efficacy of BERT models in text classification. They argue that BERT offers a flexibility that other models cannot offer in this sort of problem. The authors used an IMBD movie review dataset for their study. While this project will explore the performances of BERT on text classification, the project will evaluate the performance of BERT on a very different type of text data, and thus could yield drastically different results.

2.3 Paper 3

Shaomin Zheng and Meng Yang in their paper "A new method of improving bert for text classification." argue for the superiority for BERT models in text classification. That being said, they claim that shortcomings exists in BERT models, particularly in their inability to recognize context in longer text data. It is to be seen if this shortcoming exists when classifying social media data.

2.4 Paper 4

David Rogers and contributors in their paper, "Real-time text classification of user-generated content on social media: Systematic review." speak to the superior performance of neural networks over traditional machine learning techniques in text classification. In their reasearch, they studied social media data from a multitude of sites. It is in this respect that my project differs from theirs, as

I analyze social media data from a specific site. It is to be seen whether neural networks outperform traditional machine learning techniques in my project.

2.5 Paper 5

Hemant Purohit and contributors in their paper, "Intent classification of short-text on social media." discuss the difficulties discerning intent in social media text data. This task will be a challenge in my analysis, as understanding intent will allow for more accurate classifications. However, this project is not necessary classifying with respect to intent, but with respect to a thread label that might reflect intent.

3 Model Building Approach

3.1 Naive Bayes

The Bernoulli Naive Bayes model was built using Python's `scikit-learn` library. The comments were tokenized using the `CountVectorizer()`, adhering to the Bag of Words model framework. For this model, the Reddit comments were stemmed, stopwords were removed, and non-alphanumeric characters were removed. A simple modification was made to the default settings of `CountVectorizer()` and Bernoulli Naive Bayes in `scikit-learn` by allowing the model to capture bigrams, as while the goal was to keep the Naive Bayes model simple, accounting for bi-grams made the model more able to capture language patterns within the data.

3.2 Logistic Regression

The Logistic Regression model was built using Python's `scikit-learn` library. The comments were tokenized using the `CountVectorizer()`, adhering to the Bag of Words model framework. For this model, the Reddit comments were stemmed, stopwords were removed, and non alphanumeric characters were removed. A simple modification was made to the default settings of `CountVectorizer()` and Logistic Regression in `scikit-learn` by allowing the model to capture bigrams, as while the goal was to keep the Logistic Regression model simple, accounting for bi-grams made the model more able to capture language patterns within the data.

3.3 Convolutional Neural Network

The Convolutional Neural Network was constructed using `TensorFlow` and `Keras` libraries. For this model, nonalphanumeric characters were removed, but the comments were not stemmed, and stopwords were left in, as it was anticipated that the Neural Network would better be able to navigate these features better than the Naive Bayes and Logistic Regression models. The model was designed in a function with customizable hyperparameters such as number of filters, kernel size, number of dense layer neurons, learning rate, and dropout rate. The model's architecture consists of an embedding layer that maps text to dense vectors, a convolutional layer for feature extraction with a ReLU activation function, a max pooling layer to reduce dimensionality, a flattening step, and dense layers with L2 regularization targeted at binary classification through a sigmoid activation function. A major concern when designing this CNN was overfitting, and so an early stopping callback was implemented in the model. Then, a grid of the hyperparameters mentioned above was iterated over in order to determine the optimal model design. The hyperparameter-tuning indicated that the best model had hyperparameters: `filters = 48`, `kernel size = 4`, `number of dense layers = 40`, `learning rate = 0.001`, and `dropout rate = 0.6`.

Below are the formulas for the activation functions:

$$\text{ReLU: } f(x) = \max(0, x)$$

$$\text{Sigmoid: } f(x) = (1 + e^{-x})^{-1}$$

As mentioned, and early stopping callback was implemented, and below is a visualization of the training and validation accuracy over the number of epochs for which it was trained.

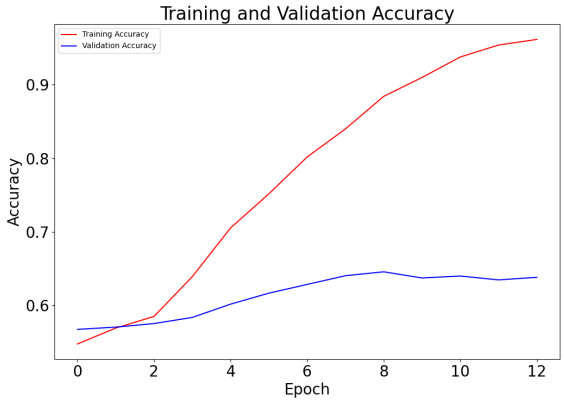

Figure 1: Training vs. Validation Accuracy CNN

As seen in the figure, the validation accuracy stopped growing at around 8 epochs, at which point the model stopped its training. Thus, overfitting concerns were addressed.

3.4 BERT Model

The pre-trained BERT model selected for this task was the MiniBERT model from Python's HuggingFace library. This model was chosen largely due to the limited computational resources available for this project,

as MiniBERT is smaller and faster than other BERT models.

The optimizer selected was Adam with a standard learning rate of '5e-5'. The training loop for this model consisted of a forward pass, in which a batch of size 30 was passed into the model, and a backward pass, in which the gradients were computed and model parameters updated according to the magnitude of the Cross Entropy loss. Below is a visual showing the training vs. validation accuracy over 6 epochs.

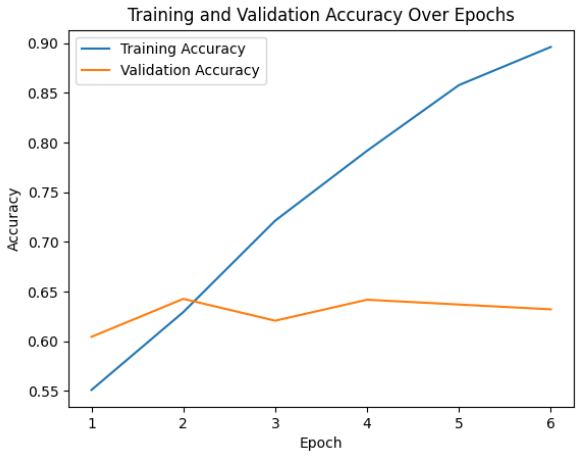

Figure 2: Training vs. Validation Accuracy BERT

Interestingly, the validation accuracy starts how higher than the training accuracy, but this changes quite quickly. This plot illustrates that extreme overfitting is avoided in training, as the training accuracy is not dramatically higher than validation accuracy.

4 Experiments

4.1 Model Performances on Training Data

The following subsections outline the respective performances of the different models on the training data.

4.1.1 Naive Bayes

The initial model built was a Bernoulli Naive Bayes model. After fitting it to the training data, below are some performance metrics on the validation data:

Table 1: **Naive Bayes on Validation Data**

Precision	Recall	F1
0.574	0.943	0.714

As seen in the table above, there is a stark difference between the model's Precision and Recall scores, which indicates that the model yields a high number of False Positive classifications. This indicates that the model frequently predicts losing comments as winning ones, while rarely misclassifying winning comments as losing ones. This is evident in the confusion matrix below.

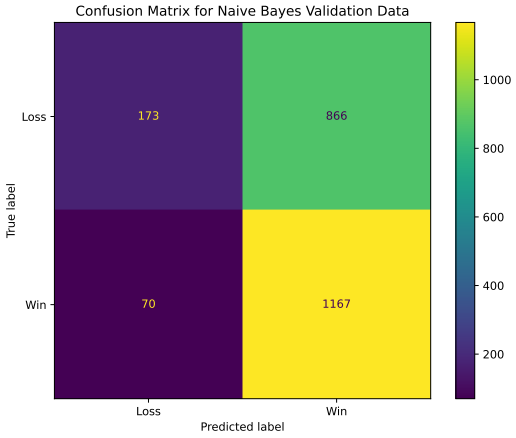

Figure 3: Confusion Matrix For N.B. Validation Data

The high number of False Positives in the confusion matrix above was concerning, but the low number of False Negatives helped offset this shortcoming, and this is reflected in the F1 score.

4.1.2 Logistic Regression

The next model built was a Logistic Regression model. After training it on the training data, below are some performance metrics on the validation data:

Table 2: **Logistic Regression on Validation Data**

Precision	Recall	F1
0.665	0.733	0.698

Interestingly, while there is still a difference between Precision and Recall, the gap is not as wide as that in the Naive Bayes model. The precision is higher, while the recall is much lower, which indicates that Logistic Regression better handles predicting losses while mishandling wins more frequently. This can be seen in the confusion matrix below.

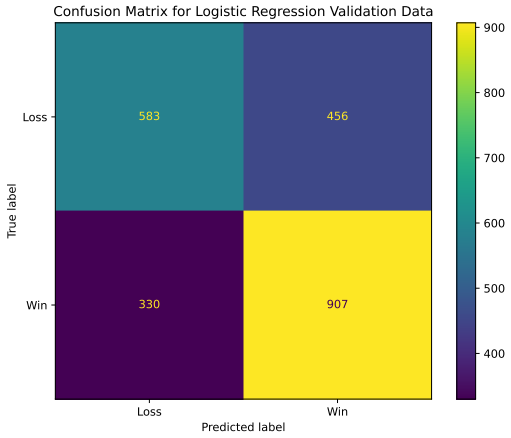

Figure 4: Confusion Matrix for L.R. on Validation Data

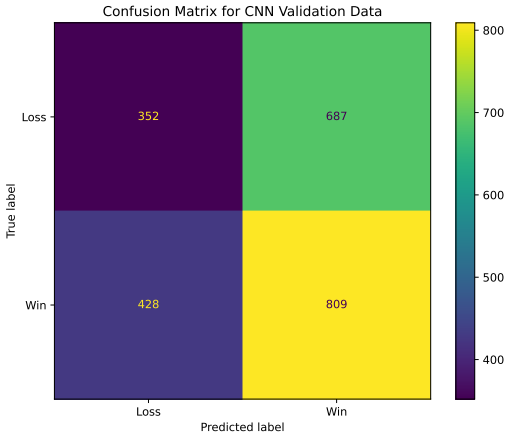

Figure 5: Confusion Matrix For CNN Validation Data

Although it handles certain classifications better than the Naive Bayes (as seen above in the number of True Negatives), due to its lower F1 score, it is so far the worst performing model.

Essentially, similar to the Logistic Regression, it does handle some classifications better than the Naive Bayes (as seen above in the number of True Negatives), yet it still has a lower F1 score.

4.1.3 Convolutional Neural Network

Next, a Convolutional Neural Network was designed. After training it on the training data, below are some performance metrics on the validation data.

Table 3: **CNN on Training Data**

Precision	Recall	F1
0.641	0.771	0.7

This model performs quite similarly to the Logistic Regression model, although the F1 score is slightly higher. Their respective precision and recall scores are quite close, indicating that the Convolutional Neural Network handles predicting losses better than the Naive Bayes model, while failing to predict wins as favorably. This can also be seen in the Confusion Matrix below.

4.1.4 BERT Model

Next, a BERT was designed. After training it on the training data, below are some performance metrics on the validation data.

Table 4: **BERT on Training Data**

Precision	Recall	F1
0.685	0.637	0.66

These metrics stand out when compared to all other models, as the precision is higher than the recall. Thus, this is the first model that handles losing comments better than wins. This can be seen in the confusion matrix below.

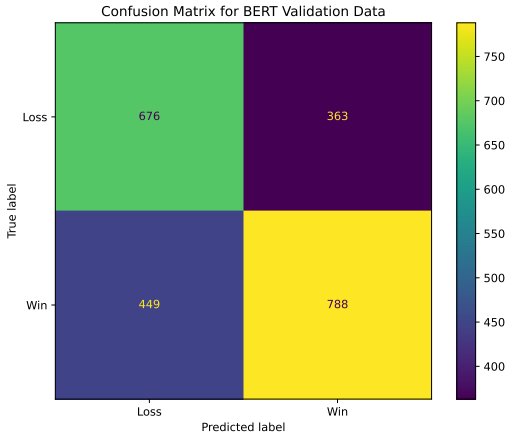

Figure 6: Confusion Matrix for BERT Training Data

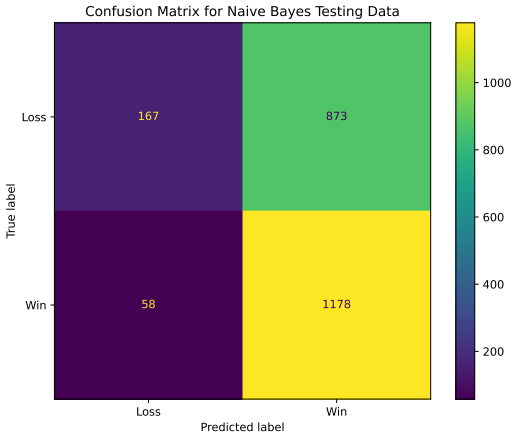

Figure 7: Confusion Matrix for N.B. Testing Data

Although this model stands out, given its low F1 score, it the worst performing model on the validation data.

In this initial analysis, the Naive Bayes model performed best on the validation data, although some concerns about its performance were raised. An exploration of the testing data was necessary to determine if trends in validation evaluation persisted.

4.2 Model Performances on Test Data

The following subsections outline the respective performances of the different models on the testing data.

4.2.1 Naive Bayes

Below are some performance metrics from the Naive Bayes model on the test data:

Table 5: **Naive Bayes on Testing Data**

Precision	Recall	F1
0.574	0.953	0.717

Below is the confusion matrix to supplement the metrics above.

Many of the same problems exist in the Naive Bayes' respective performances on the validation and testing data. The stark difference between precision and recall remains, although there are slight improvements in precision, recall, and F1. Essentially, it is clear that the Naive Bayes yields a high F1 score, but it struggles to correctly classify comments from losing post-game threads.

4.2.2 Logistic Regression

Below are some performance metrics from the Logistic Regression model on the test data:

Table 6: **Logistic Regression on Testing Data**

Precision	Recall	F1
0.661	0.749	0.703

Many of the same trends appear in the Logistic Regression's respective performances on the validation and testing data. Given the higher recall score, the model seems to handle predicting wins than losses. However, when compared to the Naive Bayes model, it predicts Losses better and Wins more poorly. The F1 score is slightly lower. The magnitude of this trend can be seen in the confusion matrix below.

Figure 8: Confusion Matrix for L.R Testing Data

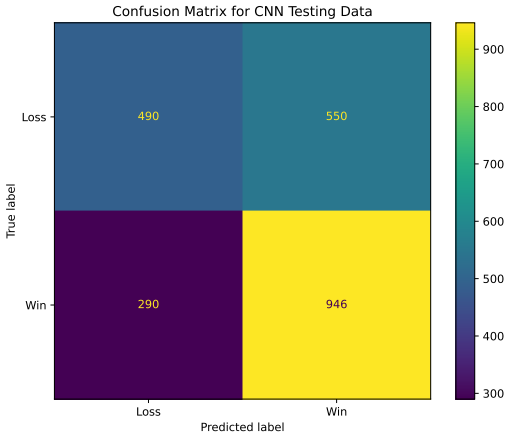

Figure 9: Confusion Matrix for CNN Testing Data

Just as in the validation set, the Logistic Regression does not earn as high of performance metrics as the Naive Bayes. However, it does clearly perform more strongly in different areas of the data than the Naive Bayes, and the lower number of false positives in the confusion matrix highlights this notion..

As seen in the confusion matrix above, the CNN does perform better in some areas of the data than the Naive Bayes model, but since the scoring criterion for this analysis is F1 score, we cannot say that it is the superior model in this task.

4.2.3 Convolutional Neural Network

Below are some performance metrics from the Convolutional Neural Network on the test data:

Table 7: **CNN on Testing Data**

Precision	Recall	F1
0.665	0.733	0.698

Many of the same trends appear in the CNN's respective performances on the validation and testing data. Given the higher recall score, the model seems to handle predicting wins than losses, just as the Logistic Regression model does. However, when compared to the Naive Bayes model, it predicts losses better and wins more poorly, just as the Logistic Regression model does. The F1 score is slightly lower than it is in the training section. The magnitude of this trend can be seen in the confusion matrix below.

4.2.4 BERT Model

Below are some performance metrics from the BERT model on the testing data:

Table 8: **BERT on Testing Data**

Precision	Recall	F1
0.674	0.633	0.653

These metrics still stand out when compared to all other models, as the precision is higher than the recall. Even what evaluated on the testing data, the precision is still higher than recall, thus it is the only model whose relative strength is avoiding False Positive classifications. This is also seen clearly in the confusion matrix below.

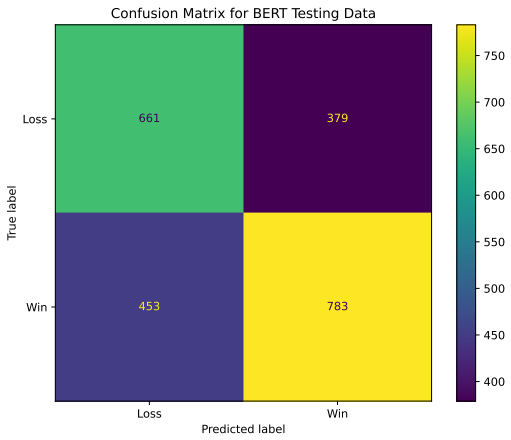

Figure 10: Confusion Matrix for BERT Testing Data

The metrics and confusion matrix above indicate that the BERT model's relative strength is correctly identifying comments that come from losing threads, unlike every other model.

4.3 Error Analysis

Perhaps the most striking result in the model evaluations was the low Precision score of the Naive Bayes, as it struggled to correctly predict comments that followed losses. Thus, it is worth looking at some comments that the Naive Bayes model incorrectly predicted as a winning comment and the Logistic Regression, CNN, and BERT model correctly classified. Below is one of these comments:

Dame is checked out and is perpetually lazy, apathetic, and downright stupid on the court. It's miserable to watch. I miss Jrue, and Khris is better than Dame right now

This comment clearly is expressing some negative sentiment towards the Damian Lillard, one of their highest performing players, yet the Naive Bayes model predicted that it would be in a winning thread. Let's see some of the associated probabilities with each word following a loss:

Word	Probability
dame	0.4727
check	0.3103
perpetu	0.5000
lazi	0.5000
stupid	0.6087
court	0.4726
miser	0.5556
watch	0.5420
miss	0.5689
jrue	0.5800
khri	0.4045
better	0.4494
right	0.5504

Table 9: Word Probabilities

As seen in the table above, while there are more words associated with losing, the word "check" as a very low association with losing, which is likely why Naive Bayes classified it as a win. Now, here are the coefficients for the Logistic Regression:

Word	Coefficient
dame	-0.1701
check	0.2474
perpetu	-0.0280
lazi	-0.2384
stupid	-0.6631
court	0.1460
miser	0.1129
watch	-0.2534
miss	-0.4716
jrue	-0.3370
khri	0.3443
better	0.1552
right	-0.2622

Table 10: Word Coefficients

Unlike in the Naive Bayes model, more of the words have losing associations, and words like "stupid" and "miss" have quite large negative coefficients, which explains the classification of the Logistic Regression model, which happens to be correct.

As mentioned earlier, the Naive Bayes model does a very good job compared to the other models as correctly classifying comments from winning post-game threads. Thus, it is worth looking at an example that only it got correct. Here is such an example:

Malik is playing well, sending him to the bench now will throw him off

This is a very interesting example, as it compliments a player, but Logistic Regression, the Neural Network, and BERT model all incorrectly classified it as a losing comment. Here is a simple breakdown as to why

by comparing some measurements from the Naive Bayes and Logistic Regression. The Logistic Regression model assigns negative coefficients to every word in the sentence except "sending", and thus it classifies the comment as a losing one. Below are the coefficients:

Word	*Coefficient*
malik	-0.5407
play	-0.0569
well	-0.2482
send	0.4619
bench	-0.3270
throw	-0.0845

Table 11: Word Coefficients

On the other hand, the Naive Bayes model weakly associates "Malik", "playing", and "well" with losing, while it associates "sending", "bench", and "throwing" all more strongly with winning. Below are the probabilities of each word given a win:

Word	*Probability*
malik	0.4348
play	0.4826
well	0.4767
send	0.6429
bench	0.5251
throw	0.5133

Table 12: Word Coefficients

Thus, the different word assessments explain the different classifications. These different results speak to the difficulties in this task and how slightly different word evaluations lead to different results. Both comments above are relatively short, and perhaps more of an in-depth assessment on the part of the user would improve model performances and fewer differences between model results.

5 Conclusions

The main insight drawn from this analysis is the wide performance scope of the models. The Naive Bayes model had a very high recall score and a shockingly low precision score, while the BERT model had a very low recall score. While the Naive Bayes seemingly handled the winning labels very well, it would be interesting to see if this would hold across different NBA subreddits. As the scope of this project only pertains to a single subreddit, it would be interesting to evaluate the performances of these models across different subreddits for different NBA teams. As the Milwaukee Bucks won significantly more games than they lost, the nature of the comments in a subreddit of a losing team would likely be considerably different. Thus, the relative performances would likely change when used on different data, such as comments from the San Antonio Spurs subreddit.

Essentially, this project does not yield definite insights regarding the general performances of these models on text classification, as this project's scope is rather niche. However, it was found that the traditional Bernoulli Naive Bayes method performed best, although only slightly and with legitimate shortcomings. On a different dataset, it is entirely possible that a different model would yield better metrics.

6 References

1. Shah, Kanish, et al. "A comparative analysis of logistic regression, random forest and KNN models for the text classification." Augmented Human Research 5 (2020): 1-16. Link

2. González-Carvajal, Santiago, and Eduardo C. Garrido-Merchán. "Comparing BERT against traditional machine learning text classification." arXiv preprint arXiv:2005.13012(2020). Link

3. Zheng, Shaomin, and Meng Yang. "A new method of improving bert for text classification." Intelligence Science and Big Data Engineering. Big Data and Machine Learning: 9th International Conference, IScIDE 2019, Nanjing, China, October 17–20, 2019, Proceedings, Part II 9. Springer International Publishing, 2019. Link

4. Rogers, David, et al. "Real-time text classification of user-generated content on social media: Systematic review." IEEE Transactions on Computational Social Systems 9.4 (2021): 1154-1166. Link

5. Purohit, Hemant, et al. "Intent classification of short-text on social media." 2015 ieee international conference on smart city/socialcom/sustaincom (smartcity). IEEE, 2015. Link

7 Screenshots

Below are screenshots from my demo. The demo takes in a comment as input, and predicts whether the comment would be posted following a win or a loss according to the selected model. I attempted to draft a comment with some ambiguity. Here is the selected comment: "This team extremely is talented, but I am worried about the three point shooting, and I am skeptical going forward."

The Naive Bayes model classifies this comment as a "Win" with "Confidence" = 68.28

Figure 11: Naive Bayes Demo

The Logistic Regression model classifies this comment as a "Loss" with "Confidence" = 51.92

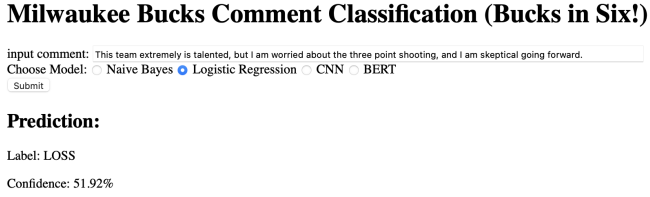

Figure 12: Logistic Regression Demo

The CNN model classifies this comment as a "Win" with "Confidence" = 53.72

Figure 13: CNN Demo

The BERT model classifies this comment as a "Win" with "Confidence" = 73.91

Figure 14: BERT Demo

Interestingly, only the Logistic Regression classifies this comment as a loss, but only slightly. The BERT model most strongly asserts that this comment follows a win, with the Naive Bayes close behind. The CNN only slightly predicts a win.