Racial Bias Detection

Aarushi Parashar and **Kabir Walia**

1 Introduction

Implicit racial biases in sports commentary have predominantly been studied from a social sciences perspective - on small-scale datasets and subjective annotation of "racial" language. However, more recently, this area of research has garnered attention due to harmful impacts of implicit racial biases including involuntarily profiling and segregating marginalized groups. Iyyer et al conducted a major study into football commentary from 1960-2019, collected from YouTube transcripts to produce the dataset titled FOOTBALL [1]. Although their work affirms the conclusions of previous social science analyses (that point to the fact that commentators tend to compliment caucausian players for their strategy and intelligence while providing applause to colored players for their strength and other physical attributes), the simplistic NLP approaches applied lack the sophistication to handle the temporal as well as confounding characteristics of the data.

Our project focuses on using more nuanced NLP and Machine Learning methods to improve upon some of the limitations highlighted by Iyyer et al and hence provide deeper insight into implicit racial bias in sports commentary.[1]

2 Data

The dataset is collected from transcripts of 1,455 full game broadcasts from the U.S. NFL and National Collegiate Athletic Association (NCAA) recorded between 1960 and 2019. mentions of players within these transcripts to information about their race (white or nonwhite) and position (e.g., quarterback). In total, FOOTBALL contains 267,778 mentions of 4,668 unique players, 65.7% of whom are nonwhite. The white and non

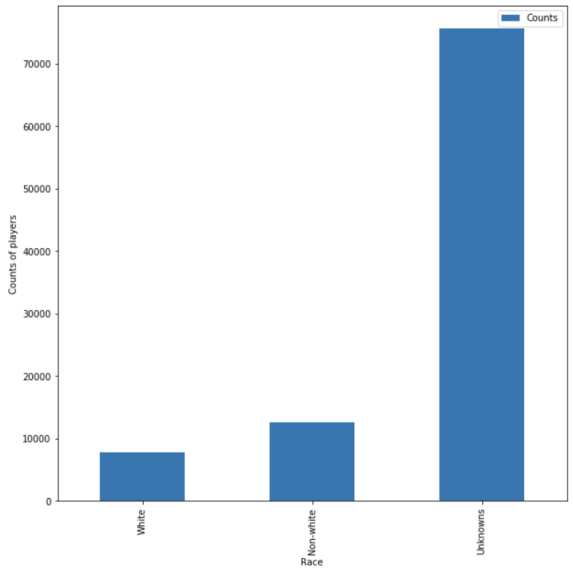

Figure 1: Frequency of each race in the mentions.

white denominations are used instead of specifically separating every race because a more balanced dataset will help in the presence of other confounding variables and the nonwhite race category is disproportionately black and many races are not represented. There is no missing data as the dataset has been preprocessed and curated by its authors, however, as illustrated in Figure 2 when investigating the total counts of each race in the dataset there is a significant amount of unknown races which are omitted for the analysis. Visually, in Figure 3 there is no clear difference when plotting the difference in frequency between common words in a word cloud indicating the need for deeper analysis.

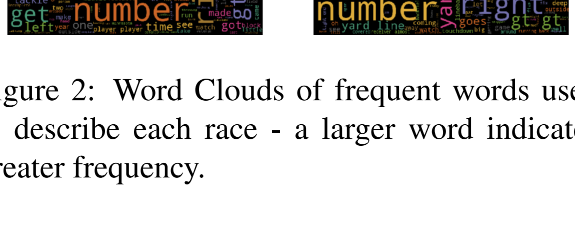

Figure 2: Word Clouds of frequent words used to describe each race - a larger word indicates greater frequency.

[1]While our project aims to detect racism we want to make it explicitly clear the the inability for any following methodology to detect racism does **not** indicate the absence of racism.

2.1 Confounding Factors

Through the exploratory data analysis it becomes evident that certain confounding variables need to be accounted for. The first confounding factor is that certain positions are disproportionately white or non white as seen in Figure 1. This affects as the mentions of those players many players' actions ("passes the ball downfield") depend on the position they play. In considering the "brain vs. brawn" debate some positions inherently require more brawn. Hence, to combat this performance will be compared within the Quarterback Position and within all positions to see the effect of the factor.

Another confounding variable is description of personality/personal attributes vs. a play (an action made by a player in the game). Negatively referencing a bad play does not indicate racial bias, however referring to a non white player in a derogatory manner when describing personal traits and not referring to a white player with the same criticism would indeed be an indication of racial bias[1]. (Henceforth, this problem will be referred to as the **Player/Play confound**).To account for this fine line in semantic difference, we will attempt to predict race from player mentions as opposed to predicting the probability that a word describes a player given the race and position as done in Iyyer et. al.[1]. In reformulating the problem to be a classification problem as well as working under the assumption that on average white and nonwhite players have the same number of good and bad plays, the focus will be primarily on the descriptive phrases that are not based on action but rather based on personality.

2.2 Feature Representation

In terms of transforming the text to extract the most insights, two common practice Natural Language Processing methods of feature representation - embeddings and TFIDF- were evaluated.

2.2.1 TFIDF

The textual information in the mentions is converted to numeric data using an approach called the Term Frequency - Inverse Document Frequency (TF-IDF). This approach produces a matrix of weights - corresponding to the importance of words across the several documents (or in this case mentions). TF-IDF works on the assumption that more important words occur less frequently. Therefore, instead of simply using word frequencies as features, using the TF-IDF weights allow our models to better learn the impact of important words. This is aligned with the reasoning that racial commentary won't involve only a few specific words. These weights are computed as $tfidf(t,d,D) = tdf(t,d)\dot{i}df(t,D)$ where the TFIDF weight of a term t in a document d is given by the product of the term frequency (tf) of t in d and inverse term frequency of t in the collection of all documents D is given as $idf(t,D) = log\frac{N}{|d \in D : t \in d|}$[5] The outcome is a large matrix of weights per word in each document/mention that are served as features to the models mentioned below. The TfidfTransformer in sklearn was used to perform this transformation. These features yielded the best results in terms of generalization so this is the transformation that was used in the following model sections.

2.2.2 Embeddings

Word Embedding converts a word to an n-dimensional vector. Words which are related such as 'house' and 'home' map to similar n-dimensional vectors, while dissimilar words such as 'house' and 'airplane' have dissimilar vectors.

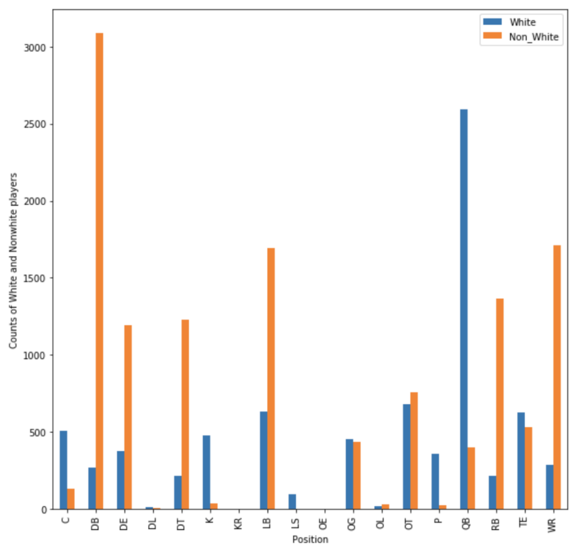

Figure 3: Breakdown of race per position.

In this way the 'meaning' of a word can be reflected in its embedding, a model is then able to use this information to learn the relationship between words. FastText and GloVe, standard packages that use different methods to generate vector representations for words, were used to generate embeddings.[4] This method resulted in comparable but slightly worse results to TF IDF and was computationally significantly more expensive, hence the results in context of embeddings are not discussed.

3 Methodology

3.1 Sub-sampling and Voting

In our midterm report we identified the significant impact of the class imbalance on our classification models[9]. The full data has 65.7% non-white players. When only looking at QBs, 75.9% of players are white. This was swaying our models to predict more of the majority class. To tackle this imbalance, we used a technique called sub-sampling. In this method, we isolate the examples of the minority class and randomly sample an equal number of examples from the majority class to gain a 50-50 split in the training data. This is a popular technique to handle class imbalance - especially in a case like ours where we have a large dataset. However, the drawbacks for this method are 1) loss of useful data and 2) since the data is collected over several decades, the commentary would also vary and if random samples have only specific time periods, the model could be biased. As a solution to this problem, we trained 5 models (used each of the following models 5 times each) on different randomly sampled datasets. To combine these models, we used a method called Hard Voting, wherein we use each of the 5 models to make predictions on the test dataset and then choose the class label which was predicted by a majority (3 or more) of the classifiers. This helps us tackle imbalance while preventing loss of information and biasing our model.

The chosen models are all discriminative in nature due to previous experimentation with generative vs. discriminative models[9]. All models are tuned based on subsampled data (from the full dataset) with TF-IDF feature representation.

3.2 Hyperparameter Tuning via Grid Search and Random Search

The base models of LogisticRegression were tuned on a 5-fold cross-validation scheme using the Grid Search approach, whereas Decision Trees and XGBoost were initially tuned using Random Search and then Grid Search (See RandomSearchCV and GridSearchCV in sklearn)[10]. They were evaluated using their voting classifier. First random search was used to identify an approximate range of values for different parameters (since tree-based models have a large number of hyperparameters). This was then followed by a fine-grained grid search process. In case of LogisticRegression, we had employed 2 grid searches - with decreasing range and step sizes. For example, when training Logistic Regression, the C parameter was first tuned over a range of 1-30 (in steps of 5). When we saw better performance in the 1-10 range, we then did another grid search in that range (in steps of 1). We looked at F1-scores as our evaluation metric for tuning the models.

3.3 Logistic Regression

Logistic regression is a linear classification algorithm that assigns probabilities to all potential classes (RACE) and outputs the one with a higher probability. The logistic function is defined as $\sigma(t) = \frac{1}{1+e^{-(0+1^{x})}}$ Where t = -B0 + B1x and there is only a single feature x. In our case, the features are the TFIDF representations of the words.

3.3.1 Model Training

The LogisticRegression class in sklearn is used. An L2 regularization with log loss function was used to avoid overfitting because the data is dense. The default optimization solver is the liblinear solver that uses a variant of gradient descent called coordinate descent. Instead of updating all parameters together, this algorithm updates one parameter at a time. The C parameter is set to 5 (higher the value, less regularized the model). We experimented with other regularizers (L1, ElasticNet - a linear combination of L1 and

L2) but our model performance was the best with only L2 - a stronger penalty on mistakes.

3.4 Decision Trees

Decision trees (discriminative models) build classification or regression models in the form of a tree structure. They break down a data set into smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes. Random Forests were also considered, which consist of a large number of individual decision trees that operate as an ensemble. Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes our model's prediction[6] However, they were overfitting the data so ultimately the decision tree algorithm was evaluated.

3.4.1 Model Training

The DecisionTreeClassifier class in sklearn was used. The parameters that were tuned are the splitting_criterion, max_depth and the minimum_impurity_decrease. The rest were kept at default values. We identified the later two hyperparameters as strong indicators of model fit through our Random Search. Low values of max_depth or high values of minimum_impurity_split led to a very low training accuracy - a symptom of underfitting. We set our grid search ranges to account for this. As a result, we identified a max_depth of 100 alongside a minimum_impurity_decrease of 0.000007 as optimal values for learning. Our grid search also identified entropy to be a better splitting_criterion than the default gini-impurity measure.

3.5 XGBoost

Due to the success of Decision Trees, XGBoost was the next iteration to try. XGBoost is a decision-tree-based ensemble Machine Learning algorithm that uses a gradient boosting framework, a special case of boosting where errors are minimized by gradient descent algorithm[8]. This algorithm trains one decision tree after another - placing higher weights on the mistakes made by the previous decision tree. The intuition is that models learn from mistakes made by previous models - hence producing better results.

3.5.1 Model Training

After an initial random search, we identified the most important parameter to tune as max_depth to control for overfitting. We first used a grid search from 0-100 (in steps of 10) and later from 20-30 (in steps of 1). This method resulted in a max_depth value of 24. As expected, this was lower than the Decision tree model as here we are ensembling 100 trees (num_estimators - default was working well) and so the max depth for any tree would not be as much as a single tree in the earlier classifier. The subsample parameter was changed from the default 1 to 0.5 because XGBoost would randomly sample half of the training data prior to growing trees - further preventing overfitting. Subsampling will occur once in every boosting iteration. All other parameters were kept at default values (based on results random search).

4 Results and Evaluation

As explained before, by subsampling our data and using voting classifiers we are able to appropriately handle the class imbalance issue. Earlier, when looking at QB data, a classifier that always predicts 'white' would achieve a 75.9% accuracy - which would be misleading. Now, with a 50-50 split, the best performance such a classifier could produce would be 50%. And so, we want models to achieve higher accuracy than this naive baseline as accuracy is now an important indicator of model performance. To compute different evaluation metrics, we separate out 20% of the total data (chosen randomly) before training our models.

4.1 Evaluation Metrics

We analyze the performance of our models using mainly accuracy, precision and recall. We look at precision and recall values per class White, Non-White as well. Precision is defined as $\frac{TP}{TP+FP}$ where TP=true positives and FP=false positives. Intuitively, it says that out of all instances that were predicted as the given class, how many actually belonged to that class. Recall is defined as

$\frac{TP}{TP+FN}$ where FN=false negatives. Recall tells us what fraction of the instances that actually belong to the given class were predicted correctly. The F1-score is a harmonic mean of these two values - it lays more weight on smaller values.

4.2 Results

Our evaluation on the test sets produce some very unique outcomes. For each model, we achieve the same values for accuracy, precision, recall and F1. This is shown in the table one.

Full Data- Sub Sampling				
Model	Precision (macro avg)	Recall (macro avg)	F1 (macro avg)	Test Accuracy
Logistic Regression	0.61	0.61	0.61	0.61
Decision Trees	0.58	0.58	0.58	0.58
XGBoost	0.54	0.54	0.54	0.54

Table 1: Precision, Recall (micro avg), F1 and Accuracy when training the data on subsampled data.

Full Data - Sub Sampling				
Model	Precision (white)	Precision (nonwhite)	Recall (white)	Recall (nonwhite)
Logistic Regression	0.61	0.62	0.62	0.61
Decision Trees	0.54	0.54	0.51	0.57
XGBoost	0.58	0.58	0.56	0.59

Table 2: Precision and Recall when training the data on subsampled data.

This is neither impossible nor incorrect. This is an outcome of balancing our training data such that both classes have the same number of instances. Furthermore, we have shown the Precision and Recall values per class in the table two.

Quarterback Position - Sub Sampling				
Model	Precision (white)	Precision (nonwhite)	Recall (white)	Recall (nonwhite)
Logistic Regression	0.66	0.66	0.66	0.66
Decision Trees	0.58	0.55	0.48	0.65
XGBoost	0.59	0.58	0.58	0.59

Table 3: Precision and Recall when training the data on the Quarterback subset of subsampled data.

Table two tells us that each model when trained using the subsampling and voting method turns out to be extremely balanced, i.e. it has a similar prediction power for both the White and Non-White classes. More theoretically, when precision and recall are equal, it would be because the FP(False Positives) = FN(False Negatives) for a class, i.e. the number of White players that the model predicts as Non-White is the same as the number of Non-White players that model predicts as White. We thus ensure that our models aren't overfitting on any particular class. For both the full data as well as QB data, Logistic Regression outperforms XGBoost, which in turn outperforms the Decision Tree classifier. Logistic Regression achieves an accuracy of 61% compared to 58% of XGBoost and 54% of Decision-Tree classifier.

Quarterback Position - Sub Sampling				
Model	Precision (macro avg)	Recall (macro avg)	F1 (macro avg)	Test Accuracy
Logistic Regression	0.66	0.66	0.66	0.66
Decision Trees	0.57	0.56	0.56	0.56
XGBoost	0.58	0.58	0.58	0.58

Table 4: Precision, Recall (micro avg), F1 and Accuracy when training the data on the Quarterback subset of subsampled data.

Logistic Regression- Without Sub Sampling				
Data	Precision (white)	Precision (nonwhite)	Recall (white)	Recall (nonwhite)
All Positions	0.55	0.71	0.29	0.88
QB	0.80	0.64	0.96	0.26

Table 5: Precision and Recall of Logistic Regression when training the data without subsampling (with the class imbalance).

Logistic Regression- Without Sub Sampling					
Data	Precision (macro avg)	Recall (macro avg)	F1 (macro avg)	Test Accuracy	Weighted Test Accuracy
All Positions	0.63	0.58	0.58	0.68	0.58
QB	0.72	0.61	0.62	0.79	0.61

Table 6: Precision, Recall (micro avg), F1 Accuracy and Weighted Accuracy of Logistic Regression when training the data without subsampling (with the class imbalance).

As shown by figures 4-6, Logistic regression produces a higher number of True Positive and True Negatives than the other two models. (bottom right and top left). Furthermore, our hypothesis about the False positives and False negatives being equal is true - the models are indeed balanced and hence produce very similar precision and recall scores. This is true for the full data as well [2] F1-scores are hence useful indicators of model performance as well. Essentially, we want our model to be able to learn the difference in commentary between the two classes and hence want to minimize the FP, FN values for both classes. Therefore, F1 combining both

[2]See Appendix.

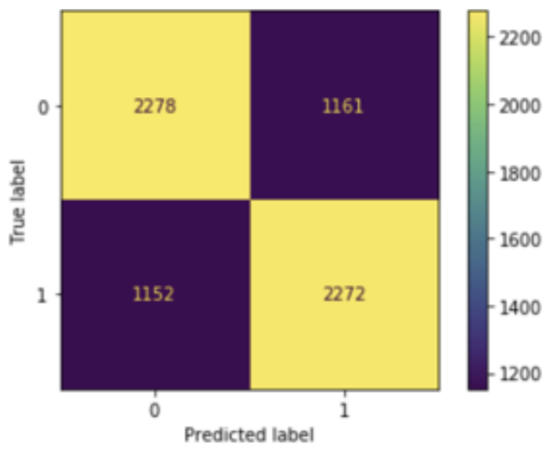

Figure 4: Confusion Matrix for Logistic Regression.

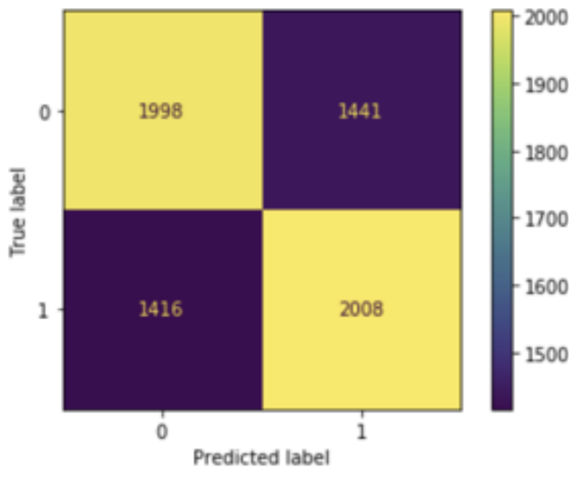

Figure 5: Confusion Matrix for XGBoost.

gives us a good indication of model performance.

Furthermore, Logistic Regression and Decision Tree classifier produce a higher accuracy on the QB data than they do on the full dataset. The logistic regression classifier usually performs better than tree based approaches when the two classes aren't well-separated. This is because decision trees develop several cuts in the feature space in their effort to separate the two classes and hence tend to overfit when there is some overlap. Although the boosting method produces better results than an individual, deeper decision tree, it wasn't able to match the logistic regression performance. We hypothesize that the TFIDF feature transformation is particularly useful for logistic regression as it looks at counts of words and furthermore their importance to documents. As a result, the logistic regression model would be identifying these phrases that

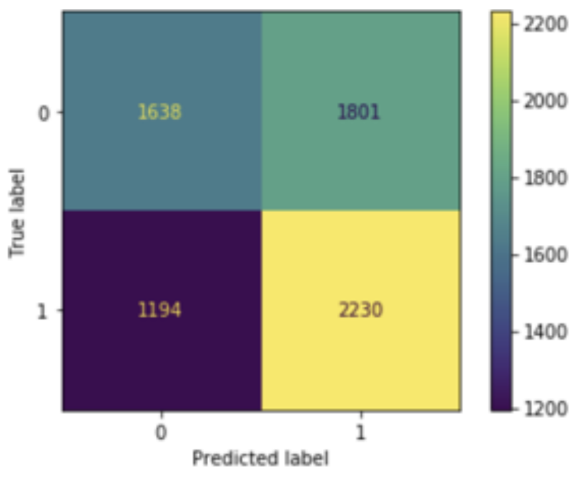

Figure 6: Confusion Matrix for Decision Trees.

maybe commonly used in reference to a White or Non-White player.

Although the numbers we get aren't very large (which makes sense - commentators in more recent times would probably not be openly racist), there is still some cause of concern. In an ideal world, we shouldn't be able to predict whether a player is White or Non-White simply based on the commentary (excluding his name). However, achieving 66% on QB data and 61% on the full data demonstrates that there is some inequality in the commentary about the two classes. There is some information that can be garnered from the text itself.

4.2.1 Impact of Subsampling

To evaluate the impact of subsampling, we can compare the results of our best classifier Logistic Regression with and without subsampling. With subsampling, we produce F1-scores of 0.61 on the full data and 0.66 on the QB data. Without subsampling, as seen in Table 5 and Table 6, we get 0.58 on full data and 0.62 on the QB data. We see a clear rise in F1-scores with subsampling - indicating our models are better identifying instances of respective classes. Comparing accuracies here isn't appropriate due to the imbalance data without subsampling. We saw higher accuracies without sampling but that was because our models were overfitting on the majority class. Furthermore, when looking at weighted accuracy, a metric that penalizes more for mistakes on minority classes, we get similar results to the sub-

sampling method.
Further, looking at the confusion matrix for Logistic Regression when trained without subsampling (and voting) on the full data. We see that there is larger tilt towards the Non-White class [0: White, 1: Non-White]. The confusion matrix for the subsampling case was more balanced.

4.2.2 Error Analysis

We looked at a few examples in the QB data where the predictions were incorrect. For example, a couple instances where Non-White players were labeled as White by the logistic regression model had phrases like ‘strategy worked’ and ‘nice work from in the first down’. These are not just positive phrases but also refer to intelligence (‘strategy’). Comparing this to examples where the model predicted White players as Non-White, we saw phrases like ‘win in a footrace’ and ‘muscles his way’. There were several such instances where such errors saw descriptions of more physical characteristics (‘muscles’). This relates to the brain vs brawn argument identified by Iyyer et al [1] and can be confirmed by our best performing model.

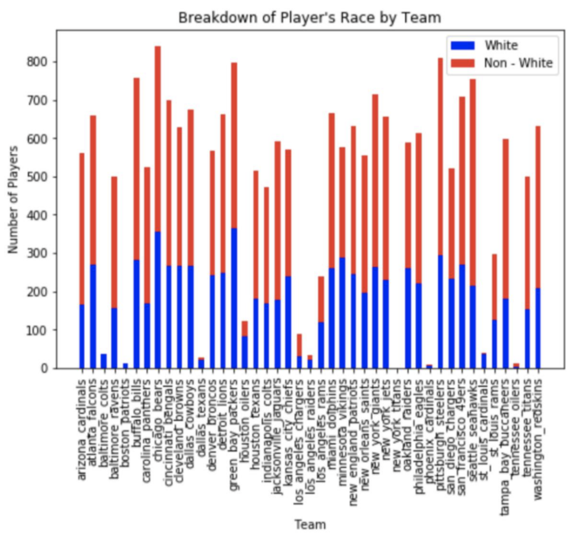

Figure 7: Break down of player’s race by team in the NFL.

Furthermore, as Logistic Regression is a fairly interpretable model, we were keen to identify some of the higher weighted features (words). After sorting words based on weights in the logistic regression model, we identified words like ’pittsburgh’, ’bengals’, etc as the top-most weighted words. This led us to think about whether different teams from different places had varying ratios of White-Non-White players and so we looked at Figure 7. The Pittsburgh Steelers and Cincinnati Bengals have a larger non-white player ratio than white. This could’ve accounted to model identifying these team names more commonly in non-white player references.

5 Discussion

5.1 Usability in Production

This model can be used in production to indicate instances of potential racism. This model cannot be used as a definitive indicator of racism and no extreme action can result from the results of the model. However, if the model is able to accurately predict the race with relatively high confidence it could indicate directions of further investigation (i.e. if the same commentator has a significant amount of these instances and it has been human verified talking to the commentator could be necessary). What is important to note is that the model cannot be used on its own but rather as a part of a process or system as we are dealing with an extremely sensitive topic.

5.2 Fairness

Evaluating fairness for this model is quite interesting because the features are just words so technically the protected attribute is the y itself. Our project is actually developing a methodology to detect fairness. Hence, there is not much to say in this regard except for the model should not be used as a definitive indicator as described above.

5.3 Weapon of Math Destruction

A weapon of math destruction is a predictive model whose outcome is not easily measured, whose predictions can have negative consequences and that creates self fulfilling feedback loops. For any of our models, the train and test sets are rigorously separated. Also, considering the problem is reformulated to be a classification task, outcome is easily measurable, the race of the player. All of the models use a significant amount of data and more data from recent transcripts can easily be obtained. Finally the most important point is that all models only detect racism (i.e. different comments being used for white vs non white players). It is explicitly stated in the introduction of this paper that none of these models can be used to predict the absence of racism. If it wasn't then negative feedback would definitely be an issue because then racism would persist as the model would predict there is no racism when there is and nothing would be done to remedy this. However, this is explicitly stated hence this negative feedback cycle doesn't exist. Finally, can the produced model have negative consequences? While such a model would be used to detect racist commentary and hence work towards establishing norms of equality and respect in sport. However, if the model makes errors and learns from another pattern that isn't racism and uses that to predict race with a high accuracy, then commentary can be deemed as racist when in reality another difference between the races. Hence, it is apparent that no extreme consequences can be taken from the results of these models. These models should be used as a means to point in the direction of potential racism not definitively classify racism. As is the model is not a Weapon of Math Destruction but as with anything it depends on what actions its results translate to.

6 Conclusion

Overall, this project enabled us to run experiments using computational methods for a very relevant issue both in Football and in the world. We believe that our experiments will present a significant contribution to the study of language bias within the field of NLP. By extending the work of Iyyer et al, we represent the first of efforts to apply discriminative methodologies in sports commentary bias - a crucial part of mass media. Given that any model we use is able to predict the race of a player with greater than 50% accuracy (in the subsampled case) indicates that there is an inequality in the text being used between the two races. Further analysis into what that inequality is and if it constitutes racism would be very helpful in making. both the NCAA and the NFL a more diverse and welcoming atmosphere.

7 Future Work

While this project is fairly comprehensive in terms of discriminative machine learning methods, there are definitely other avenues of exploration to better detect inequalities in classes.

7.1 Time Period Analysis

Another useful technique is to create subsets of data by time period. Then we could see how model performance varied throughout time and see if racism or inequality towards non whites has decreased as time has gone on (has the prediction accuracy gotten closer to 50%/baseline majority).

7.2 Neural Methods

Neural Networks are becoming more and more commonplace for textual data. Recurrent Neural Networks (RNN) especially have been proven to better account for meaning and context in a text in a computational manner. Exploring various RNN architectures may be helpful in better accounting for the difference in meaning of what is said to the two classes. More specifically, attention mechanisms could be used to better take in to account the context.

7.3 InterpretML

We focused on the interpretation of the logistic regression model only as that provided significant numbers compared to the other two models. But that was also because Logistic regression is a very interpretable model and it works well to identify feature importance when looking at word counts and word relevance through TFIDF. However, other models, developed with more sophisticated feature transformations might be useful. For example, using Explainable Boosting Machines explained by Caruana et al[11] could be utilized here. Of course, developing graphs for each word would not be suitable, however, utilizing categories like the brain vs brawn argument. We can also tag words based on whether they are game-dependent (throw, foul), etc or performance dependent ('fast', 'poor'), etc and evaluate model performance based on which of these tags do the words indicate more about the class.

8 Appendix

8.1 References

[1]https://www.aclweb.org/anthology/D19-1666.pdf

[2]https://medium.com/better-programming/understanding-racial-bias-in-machine-learning-algorithms-1c5afe76f8b

[3]https://theundefeated.com/features/tony-dungy-some-announcers-biased-language-perpetuates-black-qb-stereotypes/

[4]https://theundefeated.com/features/tony-dungy-some-announcers-biased-language-perpetuates-black-qb-stereotypes/

[5]https://nlp.stanford.edu/IR-book/html/htmledition/tf-idf-weighting-1.html

[6]https://www.analyticsvidhya.com/blog/2020/05/decision-tree-vs-random-forest-algorithm/

[7]https://medium.com/greyatom/a-quick-guide-to-boosting-in-ml-acf7c1585cb5

[8]https://towardsdatascience.com/a-beginners-guide-to-xgboost-87f5d4c30ed7

[9]https://github.com/aaparashar/racial-bias-detection/blob/master/Midterm_Report.pdf

[10]https://machinelearningmastery.com/hyperparameter-optimization-with-random-search-and-grid-search/

[11]https://arxiv.org/pdf/2006.06466.pdf

8.2 Diagrams

Full Data - Without Sub Sampling				
Model	Precision (white)	Precision (nonwhite)	Recall (white)	Recall (nonwhite)
Logistic Regression	0.55	0.71	0.29	0.88
Decision Trees	0.42	0.68	0.23	0.84
XGBoost	0.50	0.69	0.20	0.90

Table 7: Precision and Recall when training the data when training the data on subsampled data (with the class imbalance).

Full Data - Without Sub Sampling					
Model	Precision (macro avg)	Recall (macro avg)	F1 (macro avg)	Test Accuracy	Weighted Test Accuracy
Logistic Regression	0.63	0.58	0.58	0.68	0.58
Decision Trees	0.55	0.30	0.53	0.63	0.53
XGBoost	0.60	0.55	0.54	0.66	0.55

Table 8: Precision, Recall (micro avg), F1 and Accuracy when training the data on subsampled data (with the class imbalance).

Quarterback Position - Without Sub Sampling				
Model	Precision (white)	Precision (nonwhite)	Recall (white)	Recall (nonwhite)
Logistic Regression	0.80	0.64	0.96	0.26
Decision Trees	0.79	0.45	0.92	0.22
XGBoost	0.80	0.52	0.93	0.25

Table 9: Precision and Recall when training the data when training the data on subsampled data (with the class imbalance).

Quarterback Position - Without Sub Sampling					
Model	Precision (macro avg)	Recall (macro avg)	F1 (macro avg)	Test Accuracy	Weighted Test Accuracy
Logistic Regression	0.72	0.61	0.62	0.79	0.61
Decision Trees	0.62	0.57	0.57	0.75	0.56
XGBoost	0.66	0.59	0.60	0.76	0.59

Table 10: Precision and Recall when training the data on the Quarterback subset of not subsampled data (with the class imbalance).