Fairness of Predictive Modelling Based on Traffic Check Data

Taeyoung Kim, Mingi Kang
Technische Universität München
Fakultät für Elektrotechnik und Informationstechnik

Abstract

This research aims to design a prediction model and evaluate its fairness. We designed a model to predict arrest decisions based on the North Carolina Policing Dataset. First, we compared logistic regression, KNN, and fine-tuned KNN. A fine-tuned KNN with an optimal accuracy of 0.95 was adopted as the prediction algorithm. We considered that gender, age, and race were appropriate factors in assessing the fairness of the arrest decision. Therefore, the independence of these characteristics and arrest decision was evaluated. As a result, the p-value of all characteristics was lower than the set alpha value of 0.05, so the null hypothesis was rejected. In other words, it was recognised that there was a correlation between gender, age, race, and arrest decision. To evaluate the impact of excluding these features on the accuracy, we compared the prediction model, including gender, age, and race, with the model excluding them. As a result, a decrease in model accuracy was confirmed, excluding these characteristics. To ensure high independence and accuracy, we adjusted the thresholds from 0 to 1 in increments of 0.01 to find the optimal thresholds. Finally, applying these to the model presented a "fair" model with high accuracy and independence.

1 Data Preprocessing

We examined unique values and missing values for all data columns. In particular, the column '*drugs related stop*' had numerous missing values, totalling 397,708. We attempted two approaches to address this problem: including NaN as '*False*' once and excluding it once while training. However, we observed no significant differences. Moreover, after detecting multi-correlation with '*contraband found*', we dropped the '*drugs related stop*' column.

Additionally, since the information in the '*district*' column was already included in the '*officer ID*' and did not seem to have a clear relationship with the '*stop outcome*' (i.e., arrest decision), we ignored it. The '*state*' column had the same values across all data entries, so we did not include it in our training. We converted the '*stop date*' column into '*date-time*' format and separated it into year, month, day, and day of the week, encoding Monday to Sunday as 0 - 6. Notably, we observed no significant trends in the monthly([1]**Figure 1**) and daily data(**Figure 2**), so we dropped them. Nevertheless, when considering the ratio of arrests to the number of stopped vehicles by year, values for 2000-2009 were at least approximately 40%p to 80%p higher than those for 2010-2015(**Figure 0**). Similarly, when examining the ratio by day of the week, the values for Friday through Sunday were at least 10%p to 50%p higher than those for Monday through Thursday(**Figure 4**). Nevertheless, we ignored this data due to the difficulty in identifying a clear correlation in arrest trends over the years and the limitations of the provided year range. In summary, we dropped data related to years, months, and days that did not show significant patterns. We calculated the arithmetic mean of the '*driver age*' values and replaced missing values in the '*driver age*' column with this mean. Regarding the data columns related to '*driver race*', we observed that all '*driver race*' values were mapped to unique '*driver race raw*' values(**Figure 7**). Therefore, '*driver race*' labels indicate the unique '*driver race raw*' values. Furthermore, while theoretically, it is possible to recognise the extent to which different racial characteristics are quantitatively mixed when observing someone's racial characteristics, people categorise racial characteristics in practice broadly based on criteria such as Asian, White, or Black. Therefore, replacing each '*driver race raw*' with the guaranteed uniqueness of the name '*driver race*' is not problematic([2]**Table 3**). For the '*violation*' column in the given data, when the violations were '*Equipment*', '*Registration/plates*' or '*Seat belt*' the probability of arrest was

[1] All figures can be found in the ipynb file, and the figure number was marked in the plot title.

[2] All tables can be found in the ipynb file, and the table number was marked at the bottom of the table.

close to 0.5(**Figure 6**). This convergence to a probability of 0.5 suggests that the type of violation did not significantly impact arrest decisions. However, since there was no clear criterion for neglecting any violation category due to significant differences from 0.5, we chose to include all violation categories in our training(**Table 6** and **Figure 8**).

See Table 1 in **APPENDIX** on the last page of this paper for detailed preprocessing steps.

2 Binary Classifier Selection

In our group project, we explored two classification techniques: logistic regression and k-Nearest-Neighbors (referred to as 'KNN'). We adopted an 80:20 split for the training and testing data sets. We focused on the logistic regression's general configuration since our fine-tuning experiments did not produce significantly different outcomes. In contrast, with KNN, there was a distinct difference between the results from the default configuration and those from the fine-tuned version. Consequently, we analysed and compared the performance of both KNN configurations.

2.1 Logistic Regression

First, we examined logistic regression analysis results using a Confusion Matrix. True Positives (TP) and True Negatives (TN) were 20,480 and 53,969, respectively. False Negatives (FN) and False Positives (FP) were 4,197 and 1,772, respectively(**Figure 9**). Since FN and FP are relatively small in number compared to TP and TN, it suggests a high level of prediction accuracy.
Secondly, we looked at the 'ROC(receiver operating characteristic) curve', representing the False Positive Rate (FPR) and True Positive Rate (TPR) changes. The Area Under the Curve (referred to as 'AUC') was 0.97(**Figure 10**). The logistic regression classifier had an AUC greater than 0.9, indicating its reliability.
Thirdly, we examined the 'PR(Precision-Recall) curve'. Precision represents the ratio of True Positives among those classified as True. In contrast, Recall represents the True ratio among those predicted as True by the model.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{1}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2}$$

The PR curve we obtained shows that when Recall is 1.0, Precision is approximately 0.95(**Figure 11**). In other words, both Precision and Recall values are high simultaneously, confirming that our logistic regression model is well-trained. Lastly, we calculated Precision, Recall, and F1 scores by changing thresholds. First, the F1 score is the harmonic mean of Precision and Recall, and it tends to be higher when Precision and Recall values are similar. Looking at the graph(**Figure 12**), even when the threshold increased to around 0.9, we can see that Precision and Recall values are similar. Therefore, it was evident that the F1 Score was high, around 0.9. Considering that the logistic regression model, based on the sigmoid function, has a default threshold of 0.5, increasing the threshold to 0.9 still results in a high F1 Score, which indicates that this model is indeed a adequately trained one.

2.2 K-Nearest Neighbours (KNN)

Using K-nearest neighbours (KNN), we calculated each feature's arrest/no-action level. Some of the data obtained from preprocessing were in string type, so we encoded them to integer type for ease of use. The provided graph represents arrest/no-action based on ten features using Kernel Density Estimation.
Firstly, we plotted a Kernel Density Estimation (referred to as 'KDE') graph to see if there is a difference between True and Predicted Values. In this graph, we set arrest as 0 and no-action as 1. The results showed a substantial match between True and Predicted Values in the arrest and no-action sections(**Figure 13-22, 23**). When represented as a Confusion Matrix, TP and TN were 17,004 and 42,498, respectively, while FN and FP were 2,790 and 2,042, respectively(**Figure 24**). Similar to the logistic regression model mentioned earlier, since the values of FN and FP are relatively small compared to TP and TN, we can conclude that the arrest/no-action prediction is accurate.
Secondly, we determined the accuracy of KNN through the ROC Curve. We observed that the graph approached the top-left corner and the AUC was 0.91(**Figure 25**). While this AUC value is lower than the AUC from the ROC Curve of the logistic regression model, 0.97, it still indicates a high level of accuracy.
Thirdly, in the Precision-Recall Curve, we found that as Recall values increased, Precision values also increased. Similar to the logistic regression model, we noted that when Recall was 1.0, Precision was approximately 0.95(**Figure 26**).
Finally, we calculated Precision, Recall, and F1 Score for different threshold values in KNN(**Figure 27**). The Precision and Recall values were similar up to a threshold of approximately 0.8. Additionally, the F1 Score consistently increased to a maximum value of approximately 0.93, which occurred at a threshold of 0.8. When we also considered accuracy, we found significant accuracy up to a threshold of 0.7, with the highest accuracy of 0.92 at a threshold of 0.4.

2.3 Fine-tuned K-Nearest Neighbours (KNN)

We needed to find the optimal value of K, and since there was a significant difference between the results before and

after adjustment, as mentioned earlier, we conducted fine-tuning. We selected a range from 1 to 50 for fine-tuning as our candidates for K values. The optimal K value obtained within this range was 34. At K = 34, we observed an Accuracy of approximately 0.95(**Figure 28**).
After fine-tuning, we still saw a rough match between True and Predicted Values in both the arrest and no-action areas in the KDE(**Figure 29**). In the Confusion Matrix, TP and TN were 17,218 and 42,768, respectively, while FN and FP were 2,576 and 1,772, respectively(**Figure 30**). Compared to the values before tuning, TP and TN increased, while FN and FP decreased, indicating more accurate prediction results.
The ROC Curve and Precision-Recall Curve showed similar trends to those before tuning. The AUC in the ROC Curve increased by 0.01 to 0.92, indicating high accuracy(**Figure 31**). In the Precision-Recall Curve, when Recall was 1.0, Precision was approximately 0.95(**Figure 32**). Precision, Recall, and F1 Score also revealed significant differences with changes in the threshold. Precision and Recall values were similar up to a threshold of approximately 0.9. Additionally, the F1 Score remained high at around 0.9 until a threshold of 0.9. The maximum value of the F1 Score was approximately 0.95(**Figure 33**).

2.4 Dummy Classifier: Performance Comparison

Upon checking the confusion matrix, we found that FN and FP had very high values, precisely 17,112 and 17,157, respectively. The FN and FP values were substantial(**Figure 34**). We observed that the AUC value of the ROC curve was the lowest at 0.5, which signifies the lowest tool accuracy. We can find a similar value in the ROC curve of the dummy classifier. The ROC curve of the dummy classifier took a linear shape, and the AUC was confirmed to be 0.5(**Figure 35**). Looking at the Precision-Recall Curve, we can also determine that this classifier could be a better model(**Figure 36**). It can be observed that when Recall is around 0.7, Precision remains at 0.7. Even when calculated with different thresholds, Precision consistently shows a low value of 0.7, and as the threshold increases, Recall decreases sharply. The accuracy of the dummy classifier was approximately 0.57(**Figure 37**).

2.5 Model Selection

Ultimately, comparing the accuracy of the previously created Fine-tuned KNN model, which had an accuracy of 0.95, to the dummy classifier's accuracy of 0.5, we can conclude that the Fine-tuned KNN model significantly outperforms the dummy classifier. Therefore, we have chosen the Fine-tuned KNN model as the final selection.

3 Features for Fairness Check

Fairness can be seen as having rational reasons for the outcomes, meaning that information without rational reasons should not be used in making decisions. To evaluate fairness, we checked if there was information in the '*stop outcome*' output that had no rational reason for use. In other words, we assessed whether driver information that should not be used in deciding arrest status was used for fairness evaluation. Such information is in the data columns '*driver gender*', '*driver age*', and '*driver race*'. Therefore, we included these three features in the group for fairness evaluation. We then created a contingency table for each feature with '*stop outcome*'.

3.1 Independence

Based on the generated contingency tables, we performed a chi-squared test to calculate p-values for each feature, which we used to measure independence. We set alpha to 0.05 and evaluated whether to accept the null hypothesis. The results showed that the p-values for all features were less than the set alpha value of 0.05, leading us to reject the null hypothesis(**Table 11**). We found that driver gender, age, and racial characteristics are not independent of arrest status; in other words, there is a correlation between these characteristics and arrest status.

3.2 Separation

We used the chi-squared statistic from the same contingency tables as a measure of separation. A higher chi-squared statistic indicates a stronger correlation, so a more minor chi-squared statistic has a greater separation. The results showed that '*driver race Asian*' had the highest separation, followed by '*driver race Black*', '*driver race Other*', '*driver race White*', '*driver race Hispanic*', '*driver gender*', and '*driver age*' in descending order(**Table 12**).

3.3 Sufficiency

We calculated the target and conditional information entropy from the same contingency tables and defined the difference as sufficiency. High sufficiency means that each feature significantly reduces the uncertainty of arrest status when it is determined, indicating a strong correlation. Given the above results, we concluded that our designed model correlates with '*driver gender*', '*driver age*' and '*driver race*' in predicting '*stop outcome*'(**Figure 39**). However, more is needed to assess whether our model is wholly fair. If '*driver gender*', '*driver age*' and '*driver race*' were considered in determining arrest status, it would be considered unfair. However, this model makes predictions based on actual data. Therefore, it is still unclear whether this model

result, when applying the optimal threshold to the model's predictions, the model showed an approximately 9.71% FP rate, an approximately 6.24% FN rate(**Figure 76**), an ROC curve AUC of 0.97(**Figure 77**), and a Precision-Recall curve AUC of 0.98(**Figure 78**). This result indicates that our designed model predicts arrest outcomes like reality. However, we adjusted the thresholds to make '*driver gender*', '*driver age*' and '*driver race*' as independent as possible from '*stop outcome*'. In other words, this model remains as independent as possible from information that should not be used in determining arrest outcomes and can still predict arrest outcomes significantly effectively, even under those conditions.

6 Conclusion

We have designed a model to make meaningful predictions while ensuring fairness using the given data. The model trained on the data exhibited correlations with age, gender, and race, which were considered unfair. We aimed to mitigate the correlation between age, gender, and race attributes and the decision to arrest by finding the optimal thresholds, thus addressing accuracy issues. Statistics can provide a macro-level perspective beyond individual dimensions, revealing societal issues that may not be readily apparent. However, using statistics can also distort reality depending on who utilises them and for what purpose. Therefore, it is insufficient to evaluate its fairness solely using the current given data. To confirm the fairness of the model we created through data, it is necessary to rigorously examine whether arrest decisions were made based on unfair grounds, not just simple correlations. Hence, additional research is needed to investigate the causal relationship between sensitive features and arrest decisions using other data.

References

[1] A. L. et al. *Machine Learning: A First Course for Engineers and Scientists*. Cambridge University Press, 2022.

[2] J. Z. et al. Is chatgpt fair for recommendation? evaluating fairness in large language model recommendation. *arXiv*, 2305.07609v2, July 2023.

[3] M. Kuhn. *Applied predictive modeling*. Springer, 2013.

A Appendix

Preprocessing step	Python function(s) used	Justification
unique values	`.nunique( )`	To show the number of unique values.
	`.unique( )`	To show unique values.
missing values in column	`.mean( )`	To calculate the mean value.
	`.fillnad(int(np.rint(...)))`	If there's any missing value in the column, fill it with the value '...'.
	`df.dropna(subset=['...'], inplace=True)`	To drop missing values in '...' column. Changes saved with 'inplace=True'.
	`df_encoded.isna().sum()`	To check for other missing values.
columns to drop	`.drop(['...'], axis=1)`	To drop the column '...'.
'stop_date' to dataframe format	`.to_datetime( )`	To convert to dataframe format.
	`.dt.year`	To extract year from the data.
	`.dt.month`	To extract month from the data.
	`.dt.day`	To extract day from the data.
	`.dt.dayofweek`	To extract day of week from data.
drop the original 'stop_date'	`.drop('...', axis = 1)`	To drop the original '...' column. Attempted to identify tendency by subdividing 'stop_outcome'.
normalisation and cross tabulation	`.crosstab(..., normalise='index') .plot( ) .title( ) .ylabel( ) .show( )`	To create normalised cross tabulation. "normalise='index'" makes sum equal to 1. Aimed to determine features to drop.
label encoding	`LabelEncoder( )`	Analyses data and learns how to encode.
	`.fit_transform( )`	Transforms data; original data is encoded as numbers.
	`.loc( )`	Selects columns by name.
one-hot encoding	`pd.concat( )`	Concatenates data frames.
	`pd.get_dummies( )`	One-hot encodes categorical data.
compare 'driver_race_raw' and 'driver_race'	`.groupby( )`	Groups dataframe by a column.
	`.size( )`	Returns the size of each group.
	`sns.heatmap(..., annot=True, fmt='d', cmap="YlGnBu", cbar=True)`	Displays numeric values, formatted as integers, with a color map. Helps decide which column to delete.
correlation	`.corr( )`	Calculates the correlation of a dataframe or series.

Table 1: A detailed table of preprocessing steps.