Department of Data Processing
Machine Intelligence and Society (in Python)
SS 2023
Course Lecturer: Prof.Diepold

Group Project – Machine Intelligence and Society 2023

Report

Machine Learning Methods and Fairness:
Analysis of Binary Classifiers on the North Carolina Policing Dataset

Main authors:
Claudia Visintin
Julia Strohmenger
With help from:
Xiangyu Jin
Chongjin Ren

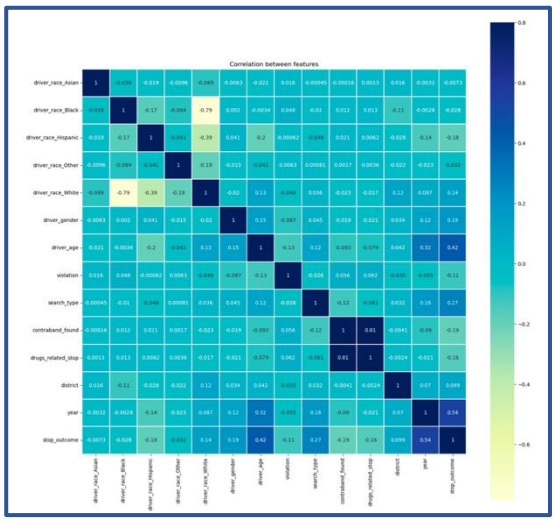
1. Introduction & Preprocessing

The project focuses on the analysis of a dataset containing information about traffic stops in North Carolina (NC). The main goal of our study was to preprocess the dataset and subsequently develop a predictive model to determine whether a given traffic stop in North Carolina results in an arrest.
We started out by implementing several data preprocessing steps.
While the code in the notebook showcases our iterative approach to understanding, cleansing, and transforming the dataset, here is a summary of the most important preprocessing steps we took in order to later be able to derive actionable and fair insights and build a reliable predictive model.

Preprocessing Step	**Python function(s) used**	**Justification**
Handling missing data	dataset.isnull0.sum fillna() dropna()	Addressing these gaps is paramount; overlooked missing values might skew analysis or compromise predictive model accuracy.
Dropping irrelevant/less relevant features	drop()	Based on our analysis of the dataset, certain features like 'state', 'officer_id', and 'driver_race_raw' were dropped as they didn't contribute significant value to the predictive model. Trimming these features can reduce complexity, potentially enhancing model efficacy. ○ **state**: Since it only contains the state "NC", it doesn't add much value for a predictive model. ○ **officer_id**: While individual officers might have their own biases or tendencies, this data might be too granular for a generalized model. ○ **driver_race_raw**: **driver_race** seems to be a cleaner representation and captures the same information as **driver_race_raw**.
Scaling	StandardScaler scaler.fit.transform	**driver_age** and **year** are scaled using **StandardScaler()**. Scaling these features means that they'll have a mean of 0 and a standard deviation of 1, which can be beneficial for certain machine learning algorithms.
Date Transformation	pd.to_datetime() dt.year drop()	The **stop_date** feature is transformed to only capture the 'year', as the yearly trend is more relevant for the upcoming analysis.
Label Encoding/ One-hot encoding	LabelEncoder() encoder.fittransform pd.to_datetime pd.get_dummies	The outcome variable **y** is not binary and needs to be converted to numerical form. Most other features are converted as well. The **driver_race** column is one-hot encoded. This creates separate columns for each race, where a '1' or '0' indicates the presence or absence of that race.

2. Choice of Methods and Features, Training, Validation and Testing

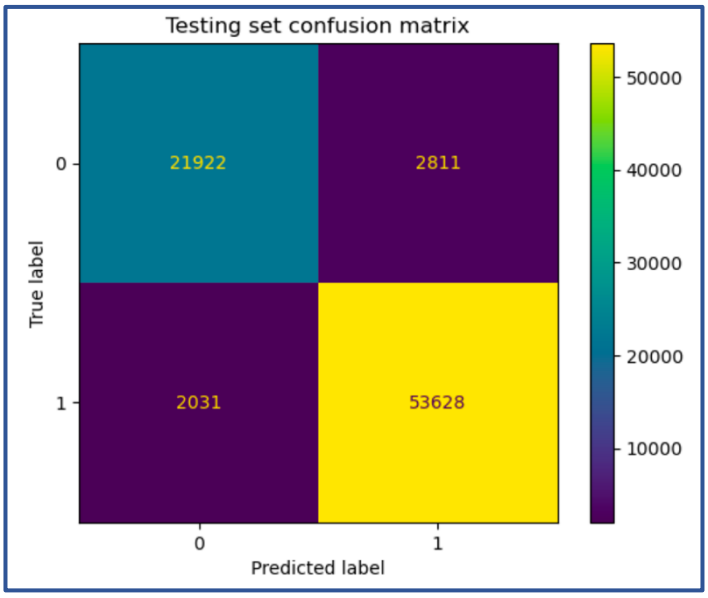
As our choice of methods for binary classifiers, we picked KNN and Neuronal Networks.
K-Nearest Neighbors (KNN) is a leaner and more interpretable baseline model for initial analysis and fairness assessment. It can help identify features that are relevant for classification, as it provides feature importance based on neighbor contributions.
Neural Networks can automatically learn and rank feature importance during training,
helping identify essential features. They have the ability to handle complex patterns and automatically learn relevant features, making them suitable for tasks involving intricate relationships. Comparing the two models will provide a comprehensive perspective on your data and help you address fairness concerns effectively.
Having ruled out multicollinearity with the help of a correlation matrix, we did not drop any more features and proceeded to prepare the data by converting it to ensure capability with the learning algorithm, splitting and scaling it.

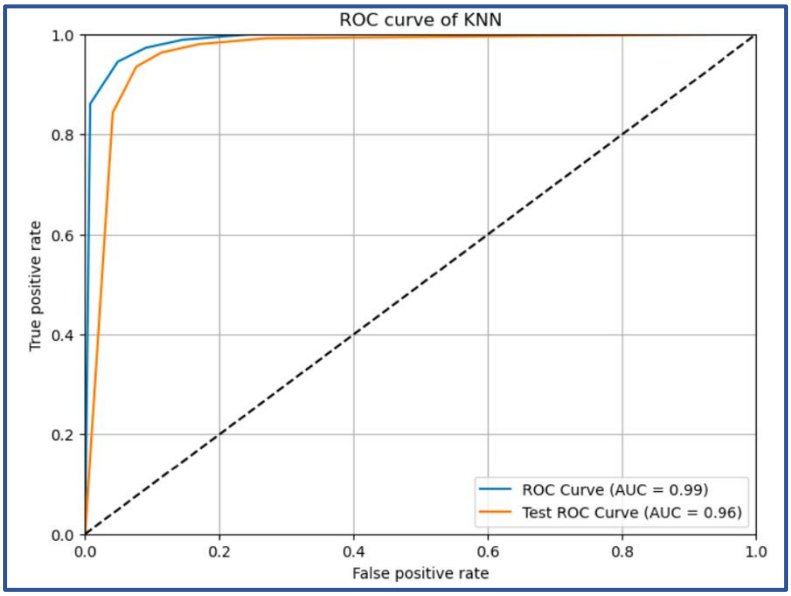


The KNN Model

We built a K-Nearest Neighbors (KNN) classifier with 5 neighbors (n_neighbors = 5). We chose 5 neighbors for our KNN classifier to strike a balance between bias reduction and computation efficiency. This decision minimizes bias, critical for our fairness-focused project while ensuring computational feasibility by avoiding excessive calculations associated with larger 'k' values.
The results from the test set predictions (**y_test_pred**) have been visualized using a confusion matrix.

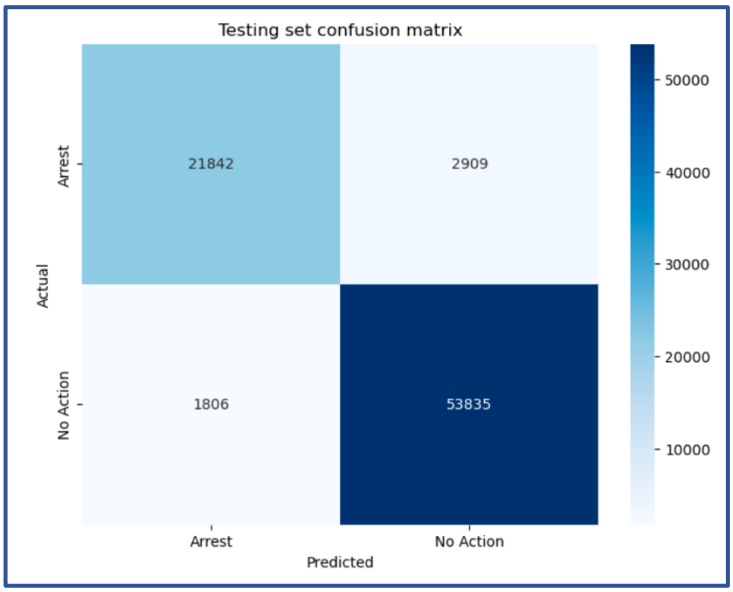
In evaluating the model's performance, several key metrics were considered.
The KNN model demonstrates strong performance, as indicated by its high accuracy (0.94), recall (0.96), precision (0.95), and F1-score (0.96). The Precision-Recall curve supports the model's robustness. With its high AUC (0.96), the ROC curve indicates the model distinguishes well between the positive and negative classes across varying thresholds.

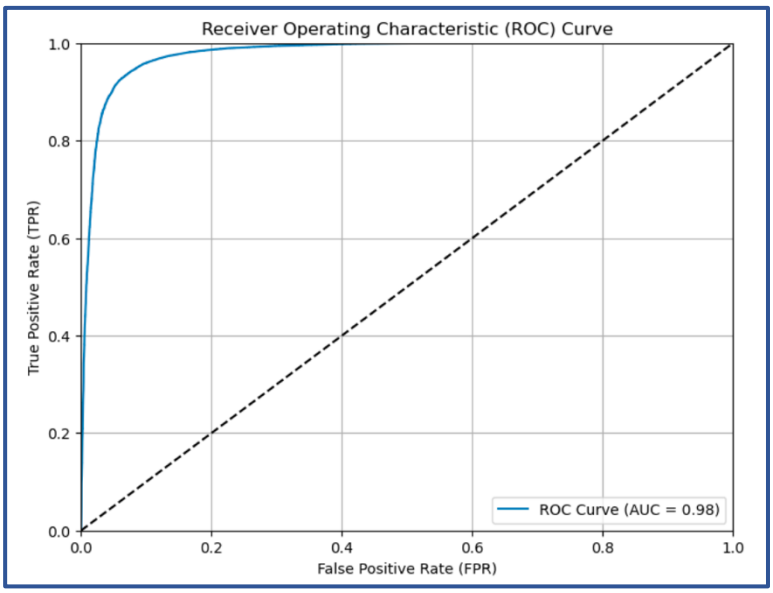




The Neuronal Network Model

We trained a neuronal network using a sequential model (TensorFlow's Keras API). **Compiling the Model,** the optimizer is set to 'nadam' (a variant of Adam optimization), the loss function is set to binary cross-entropy (suitable for binary classification), and 'accuracy' is chosen as the metric to monitor during training.
tf.keras.callbacks.ModelCheckpoint is used to set up a callback that will save the model's weights during training, monitoring validation accuracy and saving the weights only when the validation accuracy is maximized. This is a way to keep track of the best model during training.
The results from the test set predictions (**y_test_pred**) have been visualized using a confusion matrix.
The model achieves 94% accuracy, correctly identifying 97% of positive instances with a recall of 0.97 while maintaining 95% precision. The F1-score stands at 0.96, showcasing a well-rounded performance. Visualizations, including the Precision-Recall and ROC curves, provide further insights into the model's behavior at different thresholds. While the Precision-Recall curve is not as optimal as its KNN counterpart, it still represents a well-balanced tradeoff and with its high AUC (0.98), the ROC curve indicates the model distinguishes well between the positive and negative classes across varying thresholds.





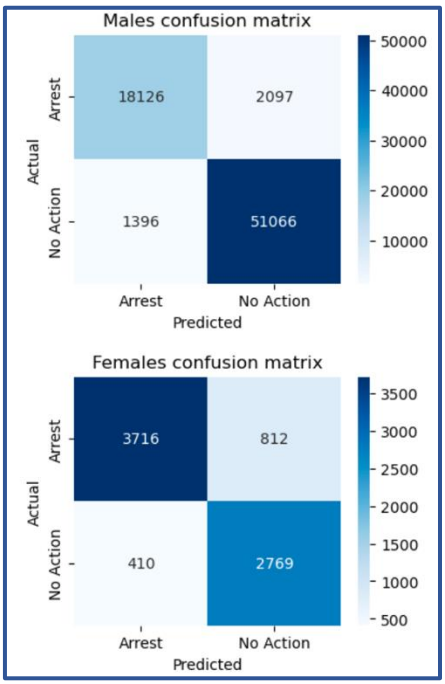
Comparison with dummy classifier

While our dummy classifier achieves an accuracy of 94%, accuracy alone may not always provide a complete picture of a model's performance, especially in imbalanced datasets where one class dominates the other. Additional metrics like recall, F1-score, precision, ROC curve, and precision-recall curve take into account both true positives and various types of errors (false positives and false negatives), providing a more comprehensive assessment of a model's ability to correctly classify instances, especially in scenarios where class distribution is imbalanced. Therefore, if the KNN and Neural Network models demonstrate superior performance in these metrics compared to the Dummy Classifier, it signifies their effectiveness in making meaningful predictions beyond random guessing and confirms that they outperform the baseline model.

3. Features and Fairness

Drivers' race, age, district, and gender are all features suitable to form groups and be checked for fairness. We decided to focus on gender as our initial step towards fairness assessment, primarily for the sake of clarity and to maintain a manageable scope for our analysis. By concentrating on gender, we aim to establish a clear and interpretable baseline for evaluating fairness metrics. This approach enables us to thoroughly investigate and understand any potential gender-based disparities in policing outcomes before expanding our analysis to other features. Gender is a salient and socially significant characteristic that often draws attention in discussions of fairness and bias. Therefore, beginning with gender allows us to address a topic of substantial public interest and establish a foundation for more comprehensive fairness evaluations across various demographic factors.

We grouped the dataset by gender, calculated confusion matrices, and various fairness metrics for each group. The lower false positive rate and higher recall, precision, and F1-score for males compared to females suggest that the model may perform differently between the two gender groups. Fairness metrics, as portrayed by the table on the right, support this finding.



Metric	male	female
False positive rate:	0.1	0.18
Recall:	0.97	0.87
Precision:	0.96	0.77
F1:	0.97	0.82
Accuracy:	0.95	0.84

1. **Dependence Metric:** The independence metric, which should ideally be equal for both genders, indicates a significant difference (0.27) between males and females. A higher value for males suggests potential bias against females in the model's predictions.
2. **Accuracy (True Positives):** The accuracy for true positives is higher for males (0.7) compared to females (0.36). This further supports the observation that the model's performance is biased against females.

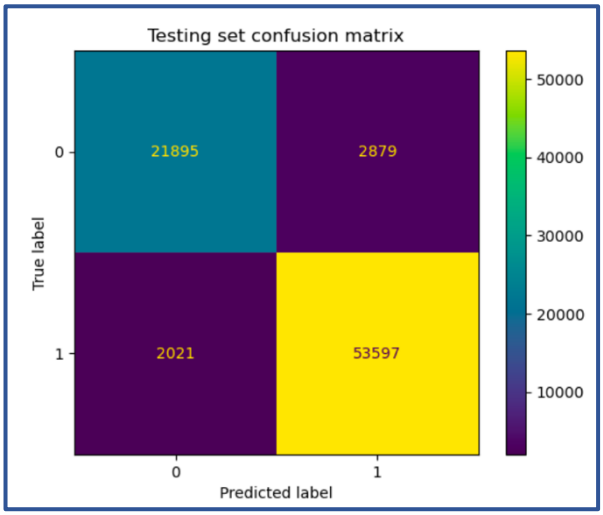
Independence males:	0.73
Independence females:	0.46
Difference:	0.27
tp Separation males:	0.96
tp Separation females:	0.77
Difference:	0.19
tn Separation males:	0.93
tn Separation females:	0.9
Difference:	0.03
tp Sufficiency males:	0.97
tp Sufficiency females:	0.87
Difference:	0.1
tn Sufficiency males:	0.1
tn Sufficiency females:	0.18
Difference:	-0.08
Accuracy(tp) males:	0.7
Accuracy(tp) females:	0.36

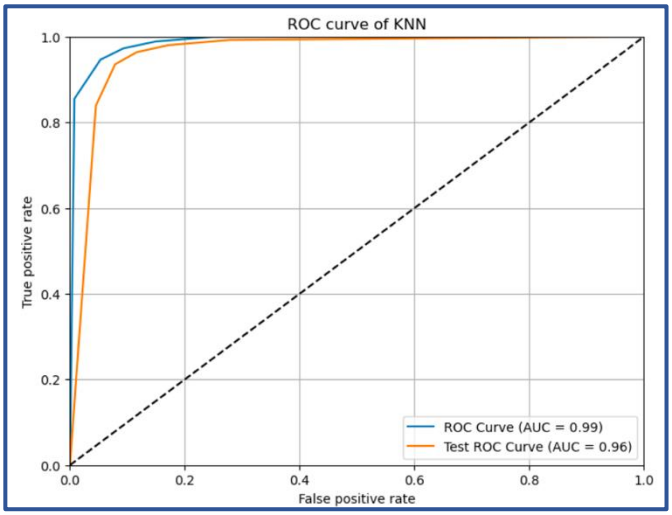
In summary, these metrics suggest that the classifier might be biased against women, as it appears to perform less accurately and consistently for female individuals compared to male individuals.

4. Excluding Sensitive Features

In this section of the analysis, a kNN classifier is trained to predict outcomes while excluding sensitive characteristics.

a. We begin by excluding “gender.” A confusion matrix was generated to visualize the results.

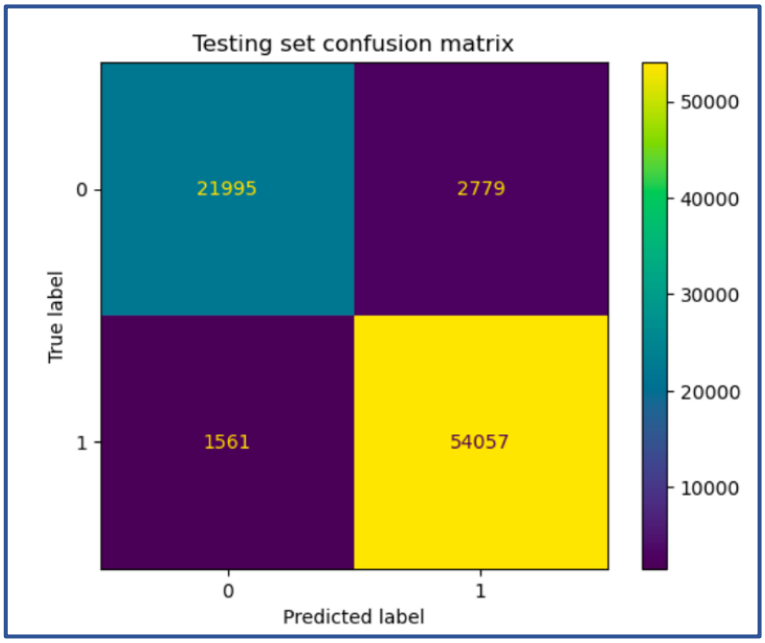


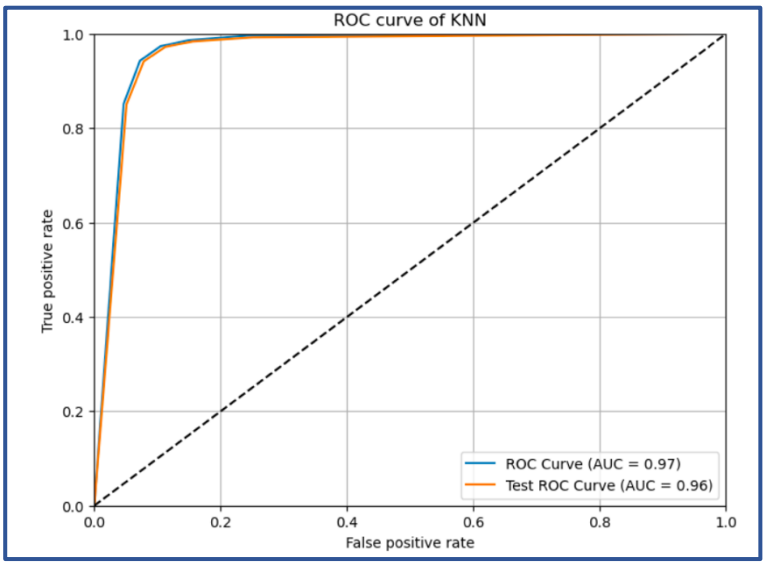


The model performed very well on the dataset, with high accuracy (0.94), recall (0.96), precision (0.95) and F1-Score (0.96). The ROC Curve with an AUC of 0.96 demonstrated its strong discrimination capability between positive and negative cases in test sets.

This suggests that removing the "Gender" feature did not significantly impact the model's predictive performance.

b. In this section, the kNN classifier is trained without using any features that are considered sensitive, including "Gender," "Age," and "Race."





The model performed very well on the dataset, with a high accuracy (0.95), recall (0.97), precision (0.95) and F1-Score (0.96). The ROC Curve with an AUC of 0.96 demonstrated its strong discrimination capability between positive and negative cases in test sets.
This suggests that further removing sensible features as "Race" and "Age", additionally to "Gender", did not impact the model's predictive performance, with accuracy and recall even slightly higher than in the previous model.

The fairness analysis of the model reveals interesting insights. When we exclude sensitive features, both males and females have nearly identical proportions of positive predictions, as shown by the low independence values of around 0.05. In terms of separation, both groups exhibit similar true positive and true negative rates, with values around 0.04 and 0.28, respectively. The model's sufficiency for correctly identifying positives is rather low for both groups, with values close to zero.
Comparing these results to when sensitive features were included, there are notable differences. Without sensitive features, the model demonstrates a more balanced outcome in terms of independence, separation, and sufficiency between gender groups.

Differences	With Sensitive	Without Sensitive
Independence	0.27	-0.0
Separation 1	0.19	0.0
Separation 2	0.03	0.01
Sufficiency 1	0.1	0.0
Sufficiency 2	-0.08	-0.01

5. Performance and Fairness Metrics at Different Thresholds