Extended Essay

Title: The use of predictive algorithms in predicting recidivism in defendants

Research Question: How far will a neural network model be more accurate in predicting recidivism than the logistic regression model used by COMPAS?

Computer Science Extended Essay

May 2024

Word Count: 3983

Table of Contents

1. Introduction 1
2.Background Information 3
  2.1 Machine Learning 3
  2.2 Logistic Regression 3
  2.3 Deep Learning 5
    2.3.1 Neural Networks (NNs) 5
    2.3.2 Feed-forward Networks 5
  2.4 Feature Selection 7
  2.5 Performance Metrics 8
3. Methodology 9
  3.1 Hypothesis 9
  3.2 Variables 10
    3.2.1 Dependent Variables: 10
    3.2.2 Independent Variables: 10
  3.3 Data Set 12
  3.4 Procedure 13
  3.5 Structure of the original Neural Network Model 13
4. Results 15
  4.1 Results for initial variables 15
    4.1.1 Logistic regression model 15
    4.1.2 Neural Network 17
  4.3 Results for reduced variables: 19
  4.4 Results with increased variables: 21
  4.5 Graphical representation of results 23
5. Data Analysis 25
6. Implications and Limitations 29
7. Conclusion 30
  References 31

1. Introduction

Recidivism can be defined as the future possibility that a criminal will re-offend after being released from prison. The United States has the highest rates of recidivism in the world, with almost 76.6% of all its criminals re-offending within 5 years of release.[1] Predicting recidivism before it occurs will allow more resources to be devoted to rehabilitating individuals that are at a high risk of recidivating, preventing further crime. Additionally, lower-risk defendants will be released, saving resources for defendants that require them, while helping one-time offenders re-integrate into society much faster. Hence, accurate predictions of recidivism are crucial to the justice system. To reduce any human error in such crucial matters, courts in the U.S. have begun using Machine Learning algorithm-provided scores to aid in their predictions. However, despite their best efforts, algorithms used in most courts do not provide sufficient accuracy nor rid the system of bias. I discovered that this was the case with a very popular recidivism prediction algorithm, COMPAS, which has been used to make risk assessments in U.S. courts since 1998[2], during personal investigations of its algorithm through an open-sourced Python notebook. The notebook modelled a study conducted by ProPublica in 2016. ProPublica analyzed the accuracy of 'COMPAS', which used a Logistic Regression model (LR), in predicting recidivism. Through the notebook that modelled ProPublica's study, I was able to see that COMPAS gave inaccurate assessments.[3]

[1] Liz Benecchi, "Recidivism Imprisons American Progress," Harvard Political Review, August 8, 2021, accessed January 4, 2024, https://harvardpolitics.com/recidivism-american-progress/.
[2] Jeff Larson et al., "How We Analyzed the COMPAS Recidivism Algorithm," ProPublica (ProPublica, May 23, 2016), accessed October 23, 2023, https://www.propublica.org/article/how-we-analyzed-the-compas-recidivism-algorithm.
[3]ibid.

LRs have been used to predict recidivism for decades due to their simplicity and low computational load. However, when dealing with predicting human behavior, which is unpredictable, its simplistic basis for functioning might be its downfall. During my modelling of the ProPublica study, I pondered whether this simplicity in an LR was the reason for the model's low accuracy. Many studies have found that the use of Neural Networks (NNs), models that can account for complex and non-linear patterns, boosted the accuracy of recidivism predictions such that they outperformed LR models. I hypothesized that they may also fare better than LR models if used by COMPAS, and thus formulated my research question: **"How far will a neural network model be more accurate in predicting recidivism than the logistic regression model used by COMPAS?"**

To comprehend exactly 'how far' the NN would fare better than the LR model, the data constraints had to be considered. LR models are known to function better on smaller sets of data. Hence, a feature selection algorithm will be used on both models to see if an NN is still the better alternative to an LR that is placed in optimum conditions. Additionally, NNs are known to perform better with larger amounts of data, hence, additional input data that was present in the data set but not used by ProPublica will be incorporated into both models to determine whether the extra computational load of an NN can be justified by its performance on larger amounts of data, which is typically available, in its prime conditions, over LR models.

In this study I will provide the necessary background information, a justification behind and set up of the methodology and experiment, and analysis on data collected from the experiment. The data will be evaluated based on specific accuracy metrics. Implications of these findings for the judicial system and their limitations will also be discussed.

2.Background Information

2.1 Machine Learning

Machine learning is a branch of Artificial Intelligence that employs statistical analysis on a given set of data to either classify data or make predictions based on it. The statistical analysis is executed via algorithms.[4]

2.2 Logistic Regression

Logistic regression is a machine learning model that is used to model the probability that a dependent variable (for instance *Y*) with a set of characteristics/features ($X_i = X_{i1}, X_{i2}, X_{i3} \dots. X_{iu}$) will succeed or fail. [5]

An LR model uses a logit function to create a relationship between the independent variable (X) and the dependent variable (Y). The function it uses is the natural log of odds that Y equals either 1 or 0. This function can be represented as:

$$Ln\left(\frac{P}{1-P}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_k X_k \tag{1}$$ [6]

Here, $Y = ln\frac{P}{1-P}$ represents that log odds ratio between (P) the probability of success and probability of failure (1- P). ($X_1, X_2, X_3 \dots. X_k$) represents the set of independent variables that

[4] IBM, "What Is Machine Learning?," IBM, 2023, accessed October 23,2023, https://www.ibm.com/topics/machine-learning.

[5]Rolando et al., "Modeling Recidivism through Bayesian Regression Models and Deep Neural Networks," *Mathematics* 9, no. 6 (March 17, 2021): 639–39, n.d., accessed on November 25, 2023, https://doi.org/10.3390/math9060639.

[6] Karen Martin, "What Is a Logit Function and Why Use Logistic Regression? - the Analysis Factor," The Analysis Factor, May 11, 2015, October 23, 2023, https://www.theanalysisfactor.com/what-is-logit-function/.

will determine the value of the dependent variable. B_1, B_2, B_3 …. B_k are the coefficients that define how Y will be impacted by a change in X.

The logit function maps Y as a sigmoid function of X:

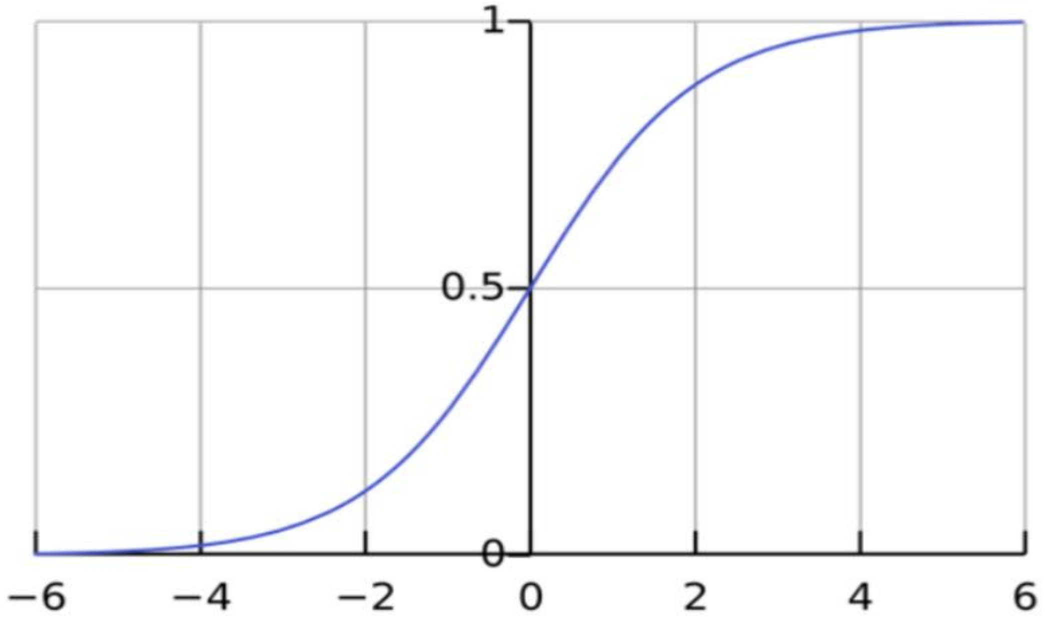


Image 1- Graph of Logistic Regression[7]

As displayed in the graph, the logit function only returns values between 0 and 1 for the dependent variable, regardless of the independent variables.[8] These results are used to categorize the outputs.

[7] "What Is Logistic Regression? - Logistic Regression - AWS," Amazon Web Services, Inc., n.d., accessed October 22, 2023, https://aws.amazon.com/what-is/logistic-regression/#:~:text=Logistic%20regression%20is%20a%20data.
[8] Ibid.

2.3 Deep Learning

2.3.1 Neural Networks (NNs)

Neural networks are classification and regression models that are designed to mimic the functioning of a human brain, specifically neurons in the brain. They are, thus, able to recognize patterns simply from the data provided, as a way of learning from the past like humans do. Hence, they can prove to be useful when a functional relationship between the inputs and the outputs is not known.[9]

2.3.2 Feed-forward Networks

In many studies that use machine learning models to predict recidivism, models that had the most accuracy were shown to be neural network models. Although their advantages in individual metrics like precision or recall were varied, they ultimately provided an increase in accuracy over LR models.[10]

A feed-forward neural network has an input layer with several nodes, which feed-data into hidden-layers which in turn feed-data into an output layer, producing a desired output. Each layer consists of several nodes which receive a summed weighted input (z_i) from the previous layer:

$$z_i = \sum(W_i X_i + b_i) \qquad (2)$$[11]

[9] Turgut Ozkan, "PREDICTING RECIDIVISM through MACHINE LEARNING," The University of Texas at Dallas, May 2017, accessed November 15, 2023, https://utd-ir.tdl.org/bitstream/handle/10735.1/5405/ETD-5608-7434.54.pdf.
[10] Ibid.
[11] Sakshi Tiwari, "Activation Functions in Neural Networks - GeeksforGeeks," GeeksforGeeks, February 6, 2018, accessed November 15, 2023, https://www.geeksforgeeks.org/activation-functions-neural-networks/.

W_i represents the weight of the neuron, i.e. how much the node will influence the input of the next node. X_i represents the inputs from the previous node while b_i represents the biases of the node. During training, the nodes learn the patterns between the previous layers and the next by altering the weights with each iteration of training data. These weights, when finalized by the training data, are then multiplied with the inputs from the testing data through a vector dot product. A bias that determines whether the activation function to connect this node to the next node should be "activated" is added to each sum. These transformations on the input, however, would only create linear relationships between layers.[12] To account for non-linear relationships, NNs also use activation functions in between hidden layers.

The ReLU function is an activation function that allows the model to account for non-linear relationships. If the function receives a negative weighted input, it returns 0 and if it receives a positive weighted input, it returns that value, as shown in the graph.

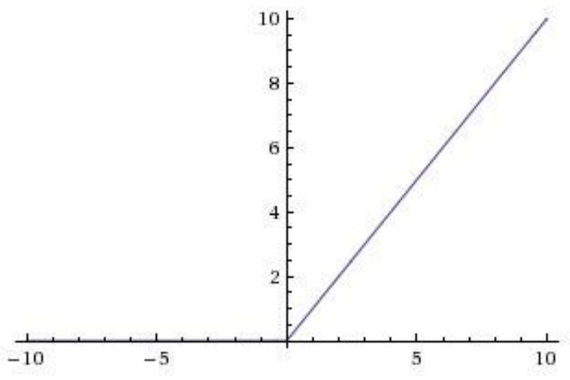


Image 2 – Graph of the ReLU function[13]

[12] Sakshi Tiwari, "Activation Functions in Neural Networks - GeeksforGeeks," GeeksforGeeks, February 6, 2018, accessed November 15, 2023, https://www.geeksforgeeks.org/activation-functions-neural-networks/.
[13] "Rectified Linear Units (ReLU) in Deep Learning," kaggle.com, 2018, accessed November 15, 2023, https://www.kaggle.com/code/dansbecker/rectified-linear-units-relu-in-deep-learning.

It is usually denoted by the function f(x) = max(0, x). In this manner, the activation function non-linearly selects which nodes need to be activated in the next layer, based on its own outputs.

Sigmoid functions are activation functions generally used for the output node and are best suited for binary classification. The functioning of which has been detailed in the LR section.

NN models also tend to have loss functions. In this essay, a binary cross-entropy loss function was used to measure the difference between the predicted and actual values (error) once output values are calculated. This loss function is specifically used for the categorization of output into two outcomes, as it is best suited to alter weights of a neural network to minimize the error.[14] An Adam optimizer is also used to adjust the weights and biases of the NN once this difference between predicted and actual values.

2.4 Feature Selection

Feature selection has been known to increase accuracy of performance in several Machine learning models. It is the process of eliminating features that impact the output the least, to prevent overfitting[15]. To understand how far and under what conditions an NN model can perform better than a LR model, Recursive Feature Elimination (RFE) was used on both models. In this process, features are recursively removed based on their importance to the model's performance. This is done by fitting the given machine learning algorithm used by each model to the input features, ranking the features by importance, and re-fitting the model. This process is

[14]Daniel Godoy, "Understanding Binary Cross-Entropy / Log Loss: A Visual Explanation," *Towards Data Science* (Towards Data Science, November 21, 2018), last modified November 21, 2018, https://towardsdatascience.com/understanding-binary-cross-entropy-log-loss-a-visual-explanation-a3ac6025181a.

[15] Rung-Ching Chen et al., "Selecting Critical Features for Data Classification Based on Machine Learning Methods," *Journal of Big Data* 7, no. 1 (July 23, 2020), accessed January 1, 2024, https://doi.org/10.1186/s40537-020-00327-4.

repeated till a desired number of features remains.[16] As, RPE has found to boost accuracy well in both LR and NN models, it was chosen for this investigation.[17]

2.5 Performance Metrics

Any prediction falls in one of 4 categories. The predicted case can be true positive, false negative, true negative, and false positive. False Positive (FP) and False Negative (FN) rates are the least desired outcomes. False positive indicates that the model predicts a non-recidivist as a recidivist impacting the decision to let them out of the judicial system. A false negative indicates that the model predicts a recidivist as a non-recidivist, which means dangerous people are set free.

The impact of these values can be derived from the below formulas:

$$False\ Positve\ Rates\ (FPR) = \frac{False\ Positives\ (FP)}{False\ Positives\ (FP) + True\ Negatives (TN)} \qquad (3)^{18}$$

$$False\ Negative\ Rates\ (FNR) = \frac{False\ Negatives\ (FN)}{False\ Negatives (FN) + True\ Negatives\ (TN)} \qquad (4)^{19}$$

[16] Jason Brownlee, "Recursive Feature Elimination (RFE) for Feature Selection in Python," Machine Learning Mastery, May 24, 2020, accessed January 1, 2024, https://machinelearningmastery.com/rfe-feature-selection-in-python/.

[17] RITHP, "Logistic Regression for Feature Selection: Selecting the Right Features for Your Model," Medium, January 1, 2023, accessed January 4, 2024, https://medium.com/@rithpansanga/logistic-regression-for-feature-selection-selecting-the-right-features-for-your-model-410ca093c5e0#:~:text=Logistic%20regression%20is%20a%20popular.

[18] Turgut Ozkan, "PREDICTING RECIDIVISM through MACHINE LEARNING," The University of Texas at Dallas, May 2017, accessed on 23 October, 2023, https://utd-ir.tdl.org/bitstream/handle/10735.1/5405/ETD-5608-7434.54.pdf.

[19] Ibid.

Measures that account for more nuances are recall and precision. High recall indicates that the model detects recidivism well, at the cost of labelling non-recidivists as recidivists.

$$Recall = \frac{True\ Positives\ (TP)}{True\ Positives\ (TP) + False\ Negatives\ (FN)} \qquad (5)$$[20]

On the other hand, precision illustrates the relevancy of the recidivism predictions.

$$Precision = \frac{True\ Positives\ (TP)}{True\ Positives\ (TP) + False\ Positives\ (FP)} \qquad (6)$$[21]

F1 score combines precision and recall providing a middle ground between them. It is the most significant accuracy metric for this reason. It can be calculated using:

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \qquad (7)$$[22]

3. Methodology

This section will further explain the research question and the methodology used for the investigation.

3.1 Hypothesis

The neural network model will predict recidivism with higher accuracy than the logistic regression model used by COMPAS, as it can account for nuances in the data that a logistic regression model cannot.

[20]Ibid.

[21] Turgut Ozkan, "PREDICTING RECIDIVISM through MACHINE LEARNING," The University of Texas at Dallas, May 2017, accessed on October 23, 2023, https://utd-ir.tdl.org/bitstream/handle/10735.1/5405/ETD-5608-7434.54.pdf.

[22] Ibid.

3.2 Variables

3.2.1 Dependent Variables:

Accuracy: Metrics will be calculated by comparing decile scores of the model to actual recidivism rates, data that the models have not been trained on. The aim is to optimize the accuracy of the model.

Recidivism:

The inputs will be used to produce a binary output (0, 1) deciding whether a specific defendant will recidivate in the next two years, and these values will be compared with the existing ones in the data set.

3.2.2 Independent Variables:

The independent variables have been chosen and formatted to mimic the ProPublica Study.

Priors Count: This variable represents information on the number of prior offenses that a defendant has. This variable will allow models to use past experiences and patterns of the defendant to predict whether they will offend in the future.

Sex: This variable indicates the sex of the defendant. It has been made to be a binary variable where 0 represents male and 1 represents female.

Age: The ages of the defendants have widely been divided into three categories. Greater than 45, 45-25, and less than 25. Using one-hot encoding this categorical data has been converted into 1s and 0s.[23] This is further illustrated below:

[23] "Using Categorical Data with One Hot Encoding," kaggle.com, accessed August 22, 2023, https://www.kaggle.com/code/dansbecker/using-categorical-data-with-one-hot-encoding.

Table 1- Table depicting one-hot encoding for age category.

Defendant Number	**Greater than 45**	**45 - 25**	**Less than 25**
1	1	0	0
3	0	1	0
4	0	0	1

As the table depicts, when a defendant falls into one category, that category is marked as one while the others are marked as 0. This simplifies the large amount of data into three categories, simplifying the process of establishing a relationship between age and recidivism.

Race: One input that is also taken into consideration is the race of defendants. Certain communities might not have access to the proper resources or support systems to assist released offenders which leads to their imprisonment. Using one-shot encoding once more, races are categorized into Caucasian, Hispanic, Native American, African American, Asian, and other.

Charge: It is categorized into M for misdemeanor which is not as severe of a crime as F for felony. 1 is assigned to M while 0 is assigned to F once again by one-shot encoding.

Additional variable:

Juvenile counts:

Three additional variables were added. Juvenile other count, juvenile felony count, and juvenile misdemeanor count. These variables store information on the type of defendant's past crimes, if any, and the frequency at which each type of crime occurred. These values were taken as three separate variables, instead of categorical variables to represent the separate columns for these values in the dataset. These values were not considered in the Python notebook that modeled the

ProbPublica study, which influenced my investigation, however, they are part of the existing dataset.

3.3 Data Set

The Data Set chosen is a set of 11,001 Broward County, Florida defendants from the ProPublica 2016 study. In the study, these 11,001 defendants were narrowed down to 6172 defendants, after cases that were not within 30 days of the score were removed. The data was collected by surveying defendants in the county jail about information through a questionnaire.[24]

It consisted of the following demographics:

Table 2- Table depicting number of defendants across different demographics.

Demographic	**Number of defendants**
Asians	31
Caucasians	2103
Hispanic People	509
African Americans	3175
Native Americans	11
Others	11

[24] ds-modules, "HCE-Materials/COMPAS/COMPAS Project.ipynb at A3bc3b02f2c1a6a0af6646b021f2e605d074e4a2 · Ds-Modules/HCE-Materials," GitHub, 2020, accessed August 22, 2023. https://github.com/ds-modules/HCE-Materials/blob/a3bc3b02f2c1a6a0af6646b021f2e605d074e4a2/COMPAS/COMPAS%20Project.ipynb

3.4 Procedure

1) The data was cleaned to rid it of any defendants that have incomplete information (see Appendix).
2) The data was split data into training and testing data (an 80-20 split) for the LR model.
3) The independent and dependent variables were defined and the LR model was fit with the training data.
4) Once the data was fit, performance metrics for the model on training data were found.
5) Steps 2-4 were repeated to account for random errors.
6) Steps 2-5 were repeated for the NN model.
7) RFE was employed on the LR model.
8) Split the data 5 times.
9) The accuracy metrics during each trial were notes.
10) Steps 7-9 were repeated with the NN model.
11) The additional juvenile count variables were added to the inputs of both models, they were fitted and their performance metrics on testing data were recorded.

3.5 Structure of the original Neural Network Model:

Studies found that a structure with more than 4 layers did not perform optimally. Additionally, adding non-linearity to these structures best aided in accuracy. [25] Hence, after multiple trials, the final model had 4 layers with the ReLU activation function except for the output layer which had a sigmoid function. The number of nodes in each layer was modeled after a study with 80 input

[25] Rolando et al., "Modeling Recidivism through Bayesian Regression Models and Deep Neural Networks," *Mathematics* 9, no. 6 (March 17, 2021): 639–39, https://doi.org/10.3390/math9060639.

nodes.[26] For this investigation, I scaled down the architecture to fit my 11 input nodes, for each of the 11 inputs, as seen in the diagram below:

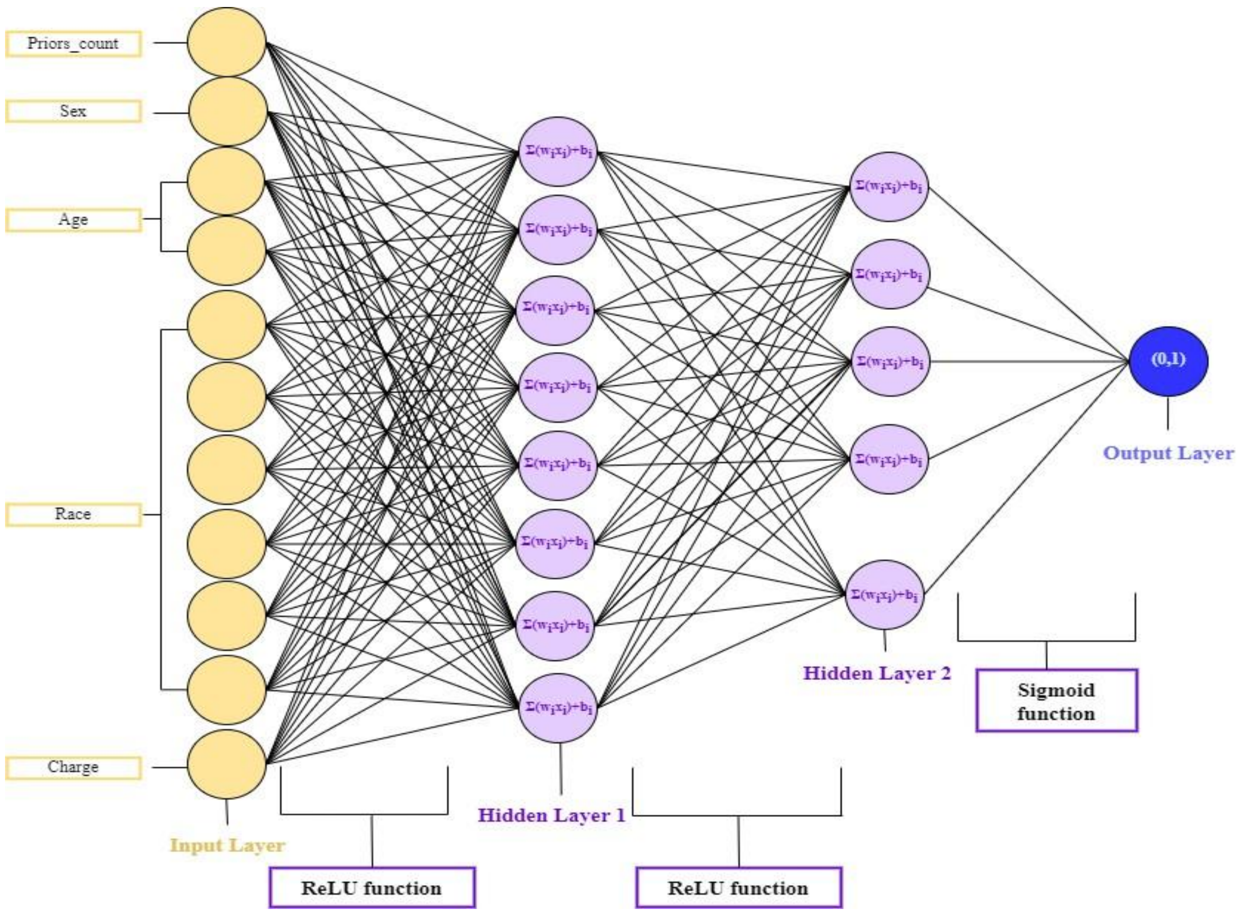


Image 3 – Structure of proposed neural network model

As depicted above, in image 3, the hidden layers made use of ReLU functions to introduce non-linearity into the model and differentiate it from an LR model. This activation function was then switched to a sigmoid function for the final layer, with one output node to allow the model to make binary classifications – recidivist (1) or non-recidivist (0).

[26] Turgut Ozkan, "PREDICTING RECIDIVISM through MACHINE LEARNING," The University of Texas at Dallas, May 2017, accessed on October 23, 2023, https://utd-ir.tdl.org/bitstream/handle/10735.1/5405/ETD-5608-7434.54.pdf.

For the sub-experiments using RFE new NN model followed a similar decrease in nodes in each layer by about 1/4th in each iteration (See Appendix). For the NN with increased inputs, the structure was slightly changed to account for the higher complexity demand caused by increased inputs (See Appendix).

4. Results

4.1 Results for initial variables

4.1.1 Logistic regression model

Table 3 – Raw data table for Logistic Regression Model

Metrics	**Values**				
	V_1	**V_2**	**V_3**	**V_4**	**V_5**
True Positive (TP)	324	324	321	323	303
False Positive (FP)	197	191	188	192	207
False Negative (FN)	220	219	244	241	215
True Negative (TN)	494	501	482	479	510
False Negative Rate (FNR)	0.404	0.403	0.432	0.427	0.415
False Positive Rate (FPR)	0.285	0.276	0.281	0.286	0.289
Recall	0.596	0.597	0.568	0.573	0.585
Precision	0.622	0.629	0.631	0.627	0.594
F1 Score	0.608	0.612	0.598	0.599	0.589

Table 4 – Processed data table for Logistic Regression

Metrics	**Average Values**
True Positive (TP)	319
False Positive (FP)	195
False Negative (FN)	228
True Negative (TN)	493
False Negative Rate (FNR)	0.416
False Positive Rate (FPR)	0.283
Recall	0.584
Precision	0.621
F1 Score	0.601

The final F1 score of the model was discovered to be 0.601 (60.1%). This value mirrors the 61% accuracy of the decile score discovered by the ProPublica analysis. [27]

[27] Jeff Larson et al., "How We Analyzed the COMPAS Recidivism Algorithm," ProPublica (ProPublica, May 23, 2016), accessed October 23 ,2023, https://www.propublica.org/article/how-we-analyzed-the-compas-recidivism-algorithm.

4.1.2 Neural Network

Table 5 – Raw data table for Neural Network Model

Metrics	**Values**				
	V_1	**V_2**	**V_3**	**V_4**	**V_5**
True Positive (TP)	347	336	332	360	339
False Positive (FP)	188	181	191	189	201
False Negative (FN)	199	219	210	224	214
True Negative (TN)	501	499	502	462	481
False Negative Rate (FNR)	0.364	0.395	0.387	0.384	0.387
False Positive Rate (FPR)	0.273	0.266	0.276	0.29	0.295
Recall	0.636	0.605	0.613	0.616	0.613
Precision	0.649	0.65	0.635	0.656	0.628
F1 Score	0.642	0.627	0.623	0.635	0.62

Table 6 – Processed data table for Neural Network model

Metrics	**Average Values**
True Positive (TP)	343
False Positive (FP)	190
False Negative (FN)	213
True Negative (TN)	489
False Negative Rate (FNR)	0.383
False Positive Rate (FPR)	0.28
Recall	0.617
Precision	0.644
F1 Score	0.629

4.3 Results for reduced variables:

4.3.1 New LR model:

Table 7 – Table with five iterations of random splitting of training and testing data, the variables eliminated by RFE, and the resultant performance metrics of the LR model.

	Values				
	V_1	**V_2**	**V_3**	**V_4**	**V_5**
Variables eliminated	Caucasian Native American	African-American Asian	African-American Other	Caucasian Hispanic	Caucasian African-American
True Positive (TP)	309	306	281	325	263
False Positive (FP)	145	132	160	122	104
False Negative (FN)	256	269	282	262	295
True Negative (TN)	525	528	512	526	573
False Negative Rate (FNR)	0.453	0.468	0.501	0.446	0.529
False Positive Rate (FPR)	0.216	0.200	0.238	0.188	0.154
Recall	0.547	0.532	0.500	0.554	0.471
Precision	0.681	0.697	0.637	0.727	0.717
F1 Score	0.606	0.604	0.560	0.628	0.569

4.3.2 Neural Network model:

Table 8 – Table with five iterations of random splitting of training and testing data, the variables eliminated by RFE, and the resultant performance metrics of the NN model with a modified structure.

	Values				
	V_1	**V_2**	**V_3**	**V_4**	**V_5**
Variables eliminated	Asian Native American	Sex Greater than 45	Caucasian Native American	Greater than 45 Less than 45	African-American Hispanic
True Positive (TP)	309	312	335	321	293
False Positive (FP)	141	158	171	183	153
False Negative (FN)	231	272	231	248	276
True Negative (TN)	554	493	498	483	513
False Negative Rate (FNR) 35	0.428	0.466	0.408	0.436	0.485
False Positive Rate (FPR)	0.203	0.243	0.256	0.275	0.235
Recall	0.572	0.534	0.592	0.564	0.515
Precision	0.687	0.664	0.662	0.637	0.657
F1 Score	0.624	0.592	0.625	0.598	0.577

The highlighted column shows the feature elimination combination that creates the best F1 score.

4.4 Results with increased variables:

4.4.1 LR model:

Table 9 – Logistic regression model trained and tested on data with additional juvenile record features.

Metrics	**Values**					**Average values**
	V_1	**V_2**	**V_3**	**V_4**	**V_5**	
True Positives (TP)	349	345	328	335	347	341
False Positives (FP)	181	174	168	180	209	182
False Negatives (FN)	223	220	217	222	213	219
True Negatives (TN)	482	496	522	498	466	493
False Negative Rate (FNR)	0.39	0.389	0.398	0.399	0.38	0.391
False Positive Rate (FPR)	0.26	0.26	0.243	0.265	0.31	0.268
Recall	0.611	0.611	0.602	0.601	0.62	0.609
Precision	0.658	0.665	0.661	0.65	0.624	0.652
F1 Score	0.633	0.637	0.63	0.625	0.622	0.629

4.4.2 Neural Network model:

Table 10 – Neural network model trained and tested on increased number of juvenile record variables

Metrics	**Values**					**Average values**
	V_1	**V_2**	**V_3**	**V_4**	**V_5**	
True Positive (TP)	355	339	328	345	320	337
False Positive (FP)	176	182	170	194	165	177
False Negative (FN)	229	213	218	201	226	217
True Negative (TN)	475	501	519	495	524	503
False Negative Rate (FNR)	0.392	0.386	0.399	0.368	0.414	0.392
False Positive Rate (FPR)	0.27	0.266	0.246	0.282	0.239	0.261
Recall	0.608	0.614	0.601	0.632	0.586	0.608
Precision	0.669	0.651	0.659	0.64	0.66	0.656
F1 Score	0.637	0.632	0.628	0.636	0.621	0.631

4.5 Graphical representation of results

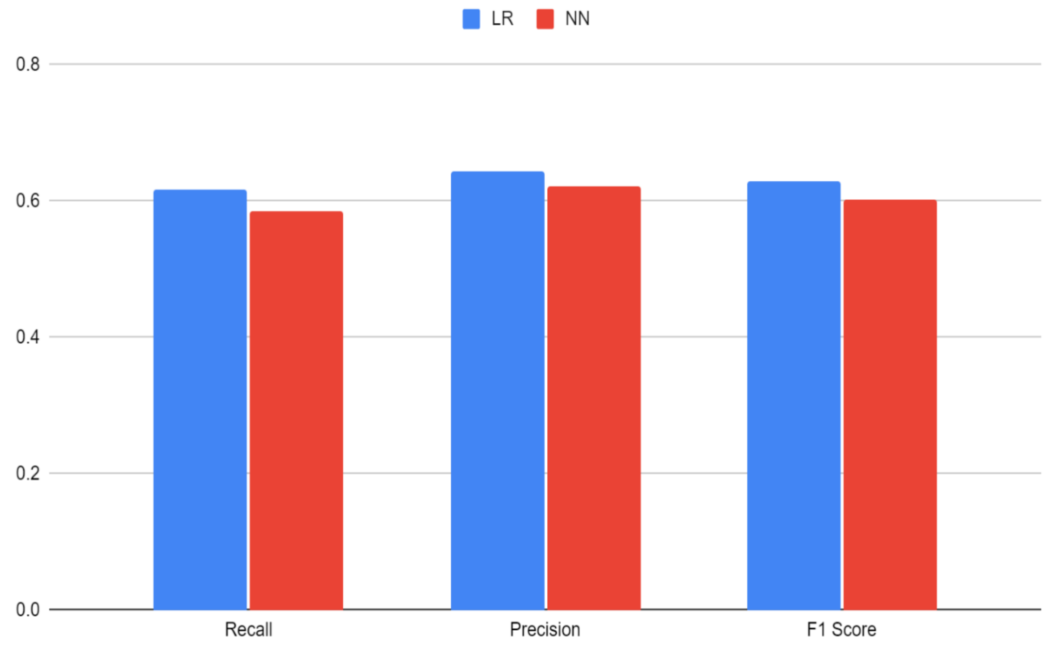
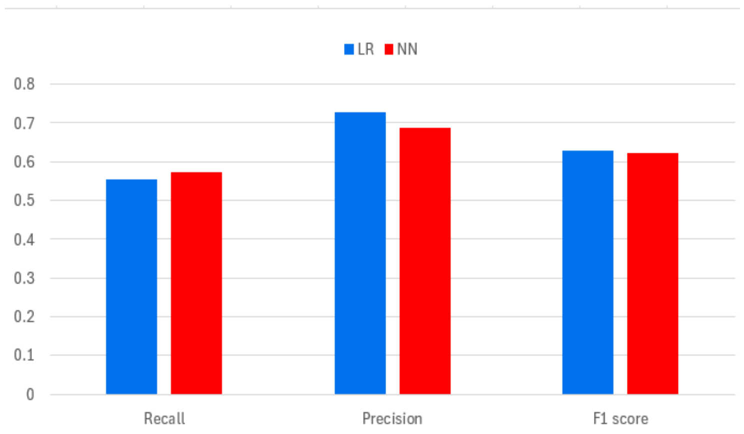


Image 5 – Graph comparing the recall, precision, and F1 scores of both LR and NN models when trained and tested against initial data.

a)



b)

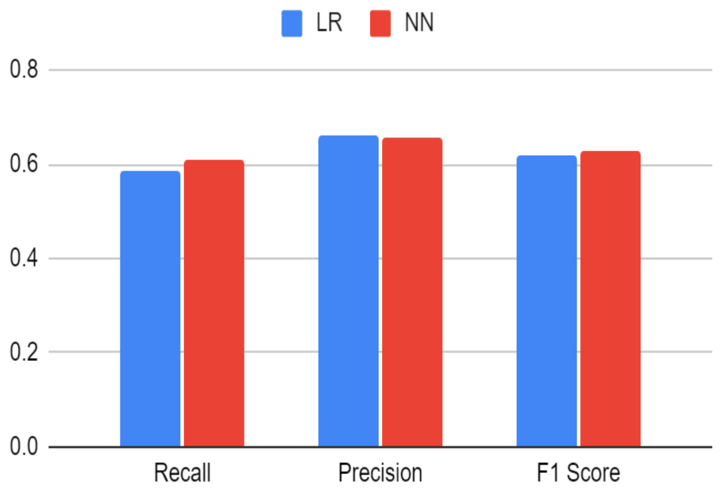


Image 6 – a) Graph comparing the recall, precision, and F1 scores of both LR and NN models when trained and tested on reduced variables, b) Graph comparing metrics of both models when trained and tested on increased variables.

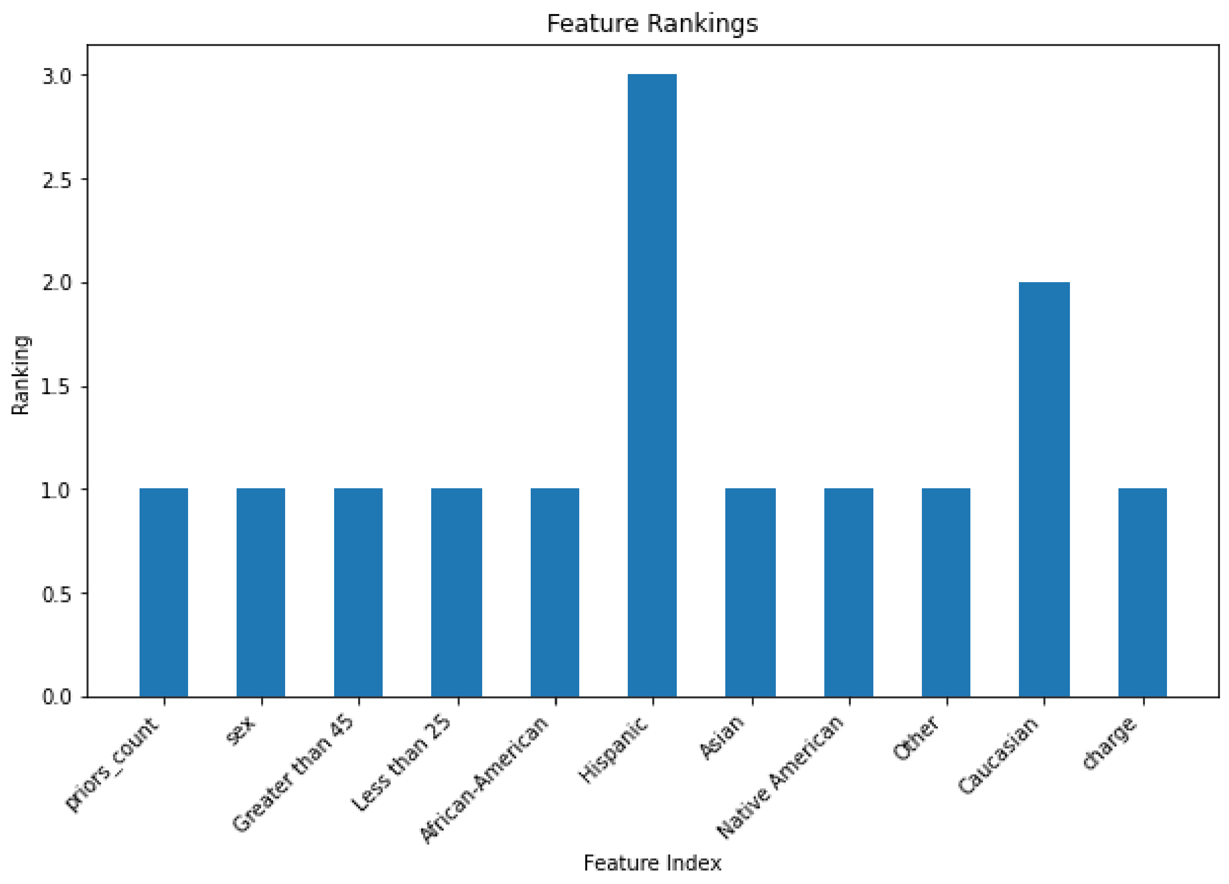


Image 7 – Graph depicting ranks of variables allotted, which gave the best results, by RFE in the NN model.

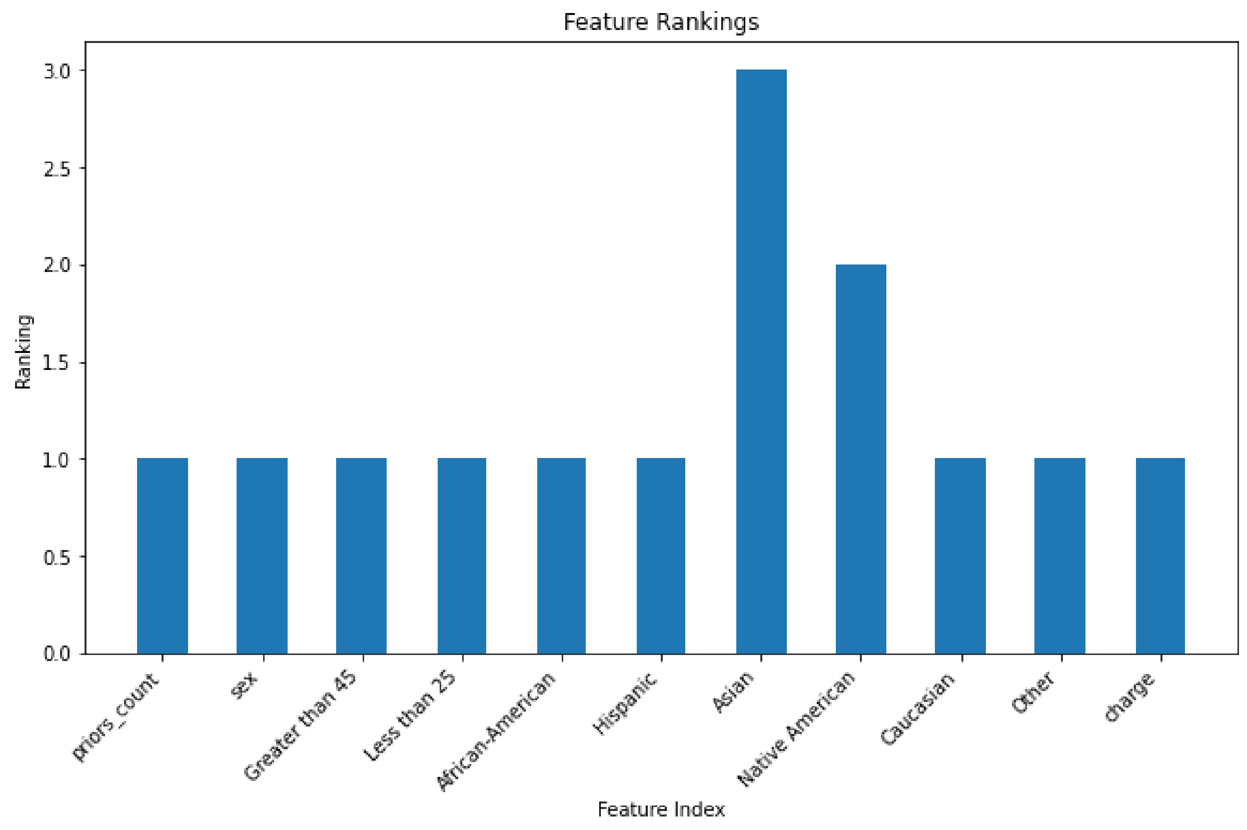


Image 8 – Graphs depicting ranks allotted by RPE, which gave the best results, in the LR model.

5. Data Analysis

Overall, the fairness metrics for the NN model were comparatively better than those of the LR model.

In Table 2, in each trial, the training and testing data were changed. The accuracy metrics for the model were calculated using their averages, as seen in Table 3. This was done to allow the accuracy metrics to contain as little error as possible. In this process, the difference in training and testing data seemed to impact the 5th trial in the LR model the most. The values for all the other trials had less fluctuation with the others. Its TPs dropped to 303 from the previous trial's 323. Additionally, it can be observed that for this trial the number of False Positives also increased. This can be explained by the moderately high recall, but low precision. This indicates that with the wrong data available for the LR to train on, it is capable of over-predicting defendants as recidivists. This problem, despite being partially corrected by the higher TNs and lower FNs, negatively impacted the overall F1 score of the trial, making it crucial to be included in the average metric values.

From Table 4, it is observed that the NN model can correct any over-predictions that it makes. In V_4, the TNs are the lowest of all the trials, this is because the FNs were the most out of all the trials. This would typically mean that the model is overpredicting non-recidivism because of improper training data. However, the model makes up for this problem as it can predict the highest TPs from all the trials, and maintain a relatively low number of FPs. Hence, any over or underprediction by the NN model didn't largely affect its accuracy, as seen by V_4's 63.5% accuracy. This difference in the impact of errors on the F1 score could be a potential reason for the NN model performing better than the LR model when the metrics were averaged.

By comparing Tables 3 and 5, it can be seen that the NN model was able to predict 24 more TPs, indicating a substantial increase in performance. Additionally, the model predicted 5 fewer FPs and 15 fewer FNs. However, the number of true negatives decreased by 4 defendants. It is evident from the data that the precision of the model has increased by a smaller amount than the recall has. Thus, the NN model was more likely to predict a defendant as recidivating in the future. However, the relevance of these predictions would be lesser due to the lower precision. This would reduce the chances of getting a negative prediction from the model, explaining the drop in TNs. As the other metrics, however, are much better than that of the LR model, this comparison indicates that the NN model can correct overprediction of recidivism as well.

Overall, the NN's F1 score was 0.028 (2.8%) greater than that of the LR model, which is explained by the advantages discussed earlier. This increase in accuracy is in line with the empirical consensus of many studies about the prediction of recidivism through machine learning. Many of them have found a similar increase in accuracy of about 1-3%.[28]

When RFE was used to determine how far this observation was true, different results were uncovered. As seen in Tables 7 and 8, the best-case feature selection in the LR model provided a 0.03% increase in accuracy compared to that of the NN model. These results indicate that an LR model can narrowly outperform an NN model if RFE is used to reduce the number of inputs. This was because LR models perform well on limited amounts of data and variables. Thus, reducing the number of variables available to the model prevented the dilution of true associations between inputs and output, which prevented large errors.[29] These findings, however, do not outperform the controlled NN model, which performs well without the extra computational step of RFE. The reason for the continued limit to the accuracy of the model despite RFE could be because the limited number of inputs when further cut down, increases overfitting, hindering the model's ability to recognize the right patterns in novel testing data.

Another observation that can be made from Tables 6 and 8 is that the NN model's F1 score falls with the use of RFE. In this instance, the NNs may have been at a disadvantage due to their structure. Studies have shown that the non-linear nature of the relationships such models fit to

[28] Turgut Ozkan, "PREDICTING RECIDIVISM through MACHINE LEARNING," The University of Texas at Dallas, May 2017, accessed November 15, 2023, https://utd-ir.tdl.org/bitstream/handle/10735.1/5405/ETD-5608-7434.54.pdf.
[29] "What Is Logistic Regression?," www.ibm.com (IBM, 2022), accessed January 5, 2023. https://www.ibm.com/topics/logistic-regression.

the data is unable to accurately function without a significant number of inputs.[30] Considering that the current model only has 11 variables, it is plausible that this advantage the LR model has over the NN model might not apply in the real world, when much more data is available. Additionally, despite extra precautions taken during the experiment to ensure comparability by using the same algorithm for feature selection to remove the same number of variables from each model, the reduction in the F1 score for the NN could also be due to a difference in how NN models and LR models react to this algorithm. Both models assign weights and coefficients to the same variables differently. This is seen in images 7 and 8 where the importance of different features is ranked differently by RFE in both models. From the images, the NN model is depicted to attribute less significance to Asian and Native American races – which constitute 31 and 11 defendants in the dataset. This contrasts with the LR model's removal of "Caucasian" and "Hispanic" as a race affecting 2103 and 509 defendants as seen in table 2. The NN model removed far fewer race inputs for defendants than the LR model did. This limitation caused by a skew in race demographics would limit the advantage of RFE for the NN. Thus, even though the LR shows an advantage with RFE, the data limitations might negate this advantage's application in the real world.

Hence, a final sub-experiment was conducted to see if an increase in variables would vary the benefit that NN models have over LR models. Especially, considering that the actual COMPAS model used 137 features to make its predictions. As seen in Tables 9 and 10, the increased number of features increased the F1 score of the LR model significantly by 2.1%, matching the

[30] "Neural Networks | a Beginners Guide," GeeksforGeeks, January 17, 2019, accessed January 4, 2024, https://www.geeksforgeeks.org/neural-networks-a-beginners-guide/.

NN's original F1 score. This could occur because the model can account for more nuances due to the increased amount of variables in the input data. However, the NN's accuracy also increased by 0.1%, once again outperforming the LR. The reason for the NN's lower increase in F1 score is that it not only requires more data, but data that has significance in terms of predicting recidivism which juvenile records might not have[31]. This could have reduced its significance to the NN[32] when this trend was reflected in the training data as well.

This further proves the idea that if both the models had access to larger data and better data, the NN model would perform better than the LR model as an NN model thrives on increased amounts of data.

6. Implications and Limitations

The experiment proved that the use of NNs is mostly guaranteed to increase model performance. However, as even an NN's best F1 score was 63.1%, its independent use in courts must be approached with caution. Especially, since ML models are black-box models that make decisions with rationales that are unclear to programmers, it is hard to ensure predictions are made under the right assumptions. Additionally, even though there is scope for the accuracy of an NN to improve, with an increased number of features, whether or not 63% accuracy increase is enough to justify the extra computational load that an NN requires must also be pondered. It may not be enough to justify the extra run time and memory required to run such a complex algorithm.

[31] Kelly Richards, "What Makes Juvenile Offenders Different from Adult Offenders? | Office of Justice Programs," *Www.ojp.gov*, February 2011, accessed January 4, 2024. https://www.ojp.gov/ncjrs/virtual-library/abstracts/what-makes-juvenile-offenders-different-adult-offenders.
[32] Ibid.

One limitation of the investigation was the use of random splitting of testing and training data instead of the use of folds, that ensure the model is trained and tested on all the data in the dataset. Folds would have made use of RFE less time-consuming and more accurate, and hence should be pursued during further experiments on this topic.

Another limitation was that only a few measures of accuracy were tested. However, tracking measures such as loss or error would have provided results that we able to account for the different aspects of successful performance.

7. Conclusion

The experiment proved that a sophisticated algorithm would predict recidivism more accurately, resulting in a 2.8% increase. Hence, a simple NN model seems to beat the accuracy of the COMPAS model used in actual courts. It can thus reasonably be argued that with greater complexity a NN model could produce far more accurate results. Hence, the hypothesis is proven.

Even with the possibility of RPE increasing the accuracy of an LR model, this increase in accuracy is not likely to beat an NN model that is equipped with sufficient data. Thus, the NN model is the better alternative, regardless of an increase in or decrease in data, under the right conditions.

References

Benecchi, Liz. “Recidivism Imprisons American Progress.” Harvard Political Review, August 8, 2021. https://harvardpolitics.com/recidivism-american-progress/.

Brownlee, Jason. “Keras Tutorial: Develop Your First Neural Network in Python Step-By-Step.” Machine Learning Mastery, July 4, 2019. Accessed January 5, 2024. https://machinelearningmastery.com/tutorial-first-neural-network-python-keras/

ds-modules. “HCE-Materials/COMPAS/COMPAS Project.ipynb at A3bc3b02f2c1a6a0af6646b021f2e605d074e4a2 · Ds-Modules/HCE-Materials.” GitHub, 2020. Accessed August 22, 2023. https://github.com/ds-modules/HCE-Materials/blob/a3bc3b02f2c1a6a0af6646b021f2e605d074e4a2/COMPAS/COMPAS%20 Project.ipynb.

Godoy, Daniel. “Understanding Binary Cross-Entropy / Log Loss: A Visual Explanation.” *Towards Data Science*. Towards Data Science, November 21, 2018. Last modified November 21, 2018. https://towardsdatascience.com/understanding-binary-cross-entropy-log-loss-a-visual-explanation-a3ac6025181a.

GeeksforGeeks. “Neural Networks | a Beginners Guide,” January 17, 2019. https://www.geeksforgeeks.org/neural-networks-a-beginners-guide/.

IBM. “What Is Machine Learning?” IBM, 2023. Accessed October 23,2023. https://www.ibm.com/topics/machine-learning.

Larson, Jeff, Surya Mattu, Lauren Kirchner, and Julia Angwin. “How We Analyzed the COMPAS Recidivism Algorithm.” ProPublica. ProPublica, May 23, 2016. Accessed October 23, 2023. https://www.propublica.org/article/how-we-analyzed-the-compas-recidivism-algorithm.

Martin, Karen. “What Is a Logit Function and Why Use Logistic Regression? - the Analysis Factor.” The Analysis Factor, May 11, 2015. Accessed October 23, 2023. https://www.theanalysisfactor.com/what-is-logit-function/.

Ozkan, Turgut. “PREDICTING RECIDIVISM through MACHINE LEARNING by Turgut Ozkan APPROVED by SUPERVISORY COMMITTEE,” 2017. Accessed November 15, 2023. https://utd-ir.tdl.org/server/api/core/bitstreams/d3f2b73e-e86f-4406-b701-fa110160d761/content.

“Rectified Linear Units (ReLU) in Deep Learning.” kaggle.com, 2018. Accessed November 15, 2023. https://www.kaggle.com/code/dansbecker/rectified-linear-units-relu-in-deep-learning.

RITHP. “Logistic Regression for Feature Selection: Selecting the Right Features for Your Model.” Medium, January 1, 2023 Accessed January 4, 2024. https://medium.com/@rithpansanga/logistic-regression-for-feature-selection-selecting-the-right-features-for-your-model-410ca093c5e0#:~:text=Logistic%20regression%20is%20a%20popular.

Richards, Kelly. “What Makes Juvenile Offenders Different from Adult Offenders? | Office of Justice Programs.” *Www.ojp.gov*, February 2011. Accessed January 4, 2024. https://www.ojp.gov/ncjrs/virtual-library/abstracts/what-makes-juvenile-offenders-different-adult-offenders.

Rolando, Oslando Padilla, Mauricio A Valle, and Gonzalo A Ruz. “Modeling Recidivism through Bayesian Regression Models and Deep Neural Networks.” *Mathematics* 9, no. 6 (March 17, 2021): 639–39. Accessed on November 25, 2023. https://doi.org/10.3390/math9060639.

Rung-Ching Chen et al., “Selecting Critical Features for Data Classification Based on Machine Learning Methods,” *Journal of Big Data* 7, no. 1 (July 23, 2020). Accessed January 1, 2024. https://doi.org/10.1186/s40537-020-00327-4.

Tiwari, Sakshi. “Activation Functions in Neural Networks - GeeksforGeeks.” GeeksforGeeks, February 6, 2018. Accessed January 4, 2024. https://www.geeksforgeeks.org/activation-functions-neural-networks/.

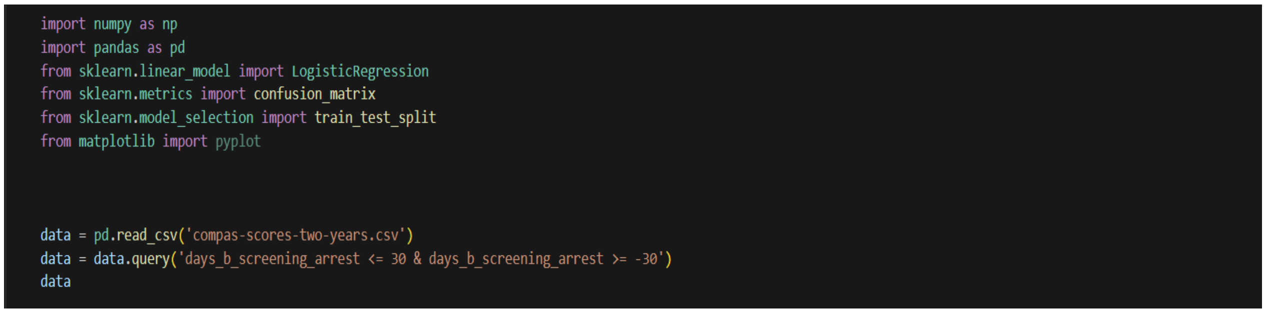
“Using Categorical Data with One Hot Encoding.” kaggle.com. Accessed August 22, 2023. https://www.kaggle.com/code/dansbecker/using-categorical-data-with-one-hot-encoding.

“What Is Logistic Regression? - Logistic Regression - AWS.” Amazon Web Services, Inc. Accessed August 22, 2023. https://aws.amazon.com/what-is/logistic-regression/#:~:text=Logistic%20regression%20is%20a%20data.

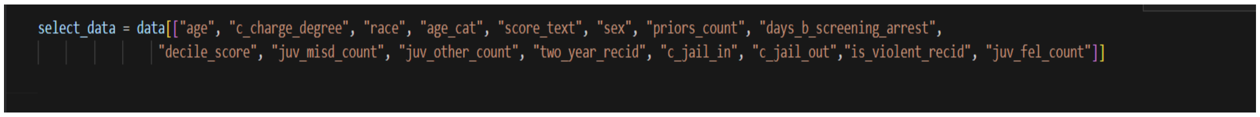
“What Is Logistic Regression?” www.ibm.com. IBM, 2022. Accessed January 5, 2024. https://www.ibm.com/topics/logistic-regression.

Appendix:

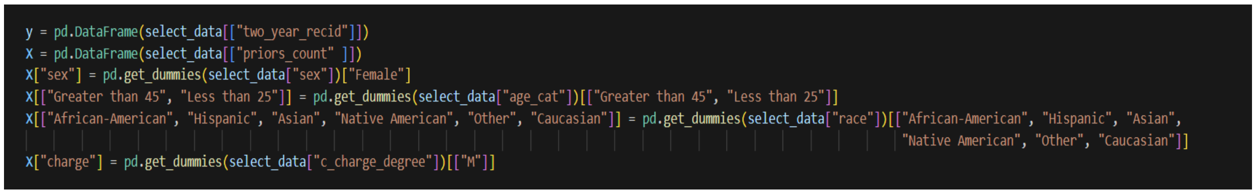
Python code to load the data for the initial Logistic Regression model:



```
import numpy as np
import pandas as pd
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import train_test_split
from matplotlib import pyplot



data = pd.read_csv('compas-scores-two-years.csv')
data = data.query('days_b_screening_arrest <= 30 & days_b_screening_arrest >= -30')
data
```

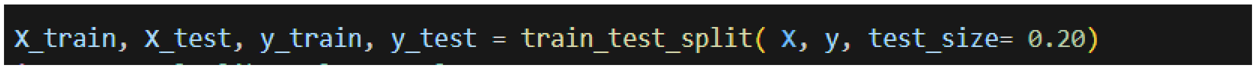


```
select_data = data[["age", "c_charge_degree", "race", "age_cat", "score_text", "sex", "priors_count", "days_b_screening_arrest",
                "decile_score", "juv_misd_count", "juv_other_count", "two_year_recid", "c_jail_in", "c_jail_out","is_violent_recid", "juv_fel_count"]]
```



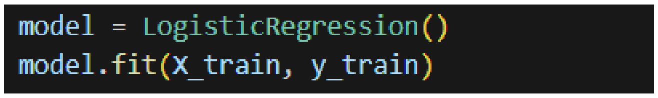
```
y = pd.DataFrame(select_data[["two_year_recid"]])
X = pd.DataFrame(select_data[["priors_count" ]])
X["sex"] = pd.get_dummies(select_data["sex"])["Female"]
X[["Greater than 45", "Less than 25"]] = pd.get_dummies(select_data["age_cat"])[["Greater than 45", "Less than 25"]]
X[["African-American", "Hispanic", "Asian", "Native American", "Other", "Caucasian"]] = pd.get_dummies(select_data["race"])[["African-American", "Hispanic", "Asian",
                                                                                                                               "Native American", "Other", "Caucasian"]]
X["charge"] = pd.get_dummies(select_data["c_charge_degree"])[["M"]]
```

Code for splitting the data in training and testing data:



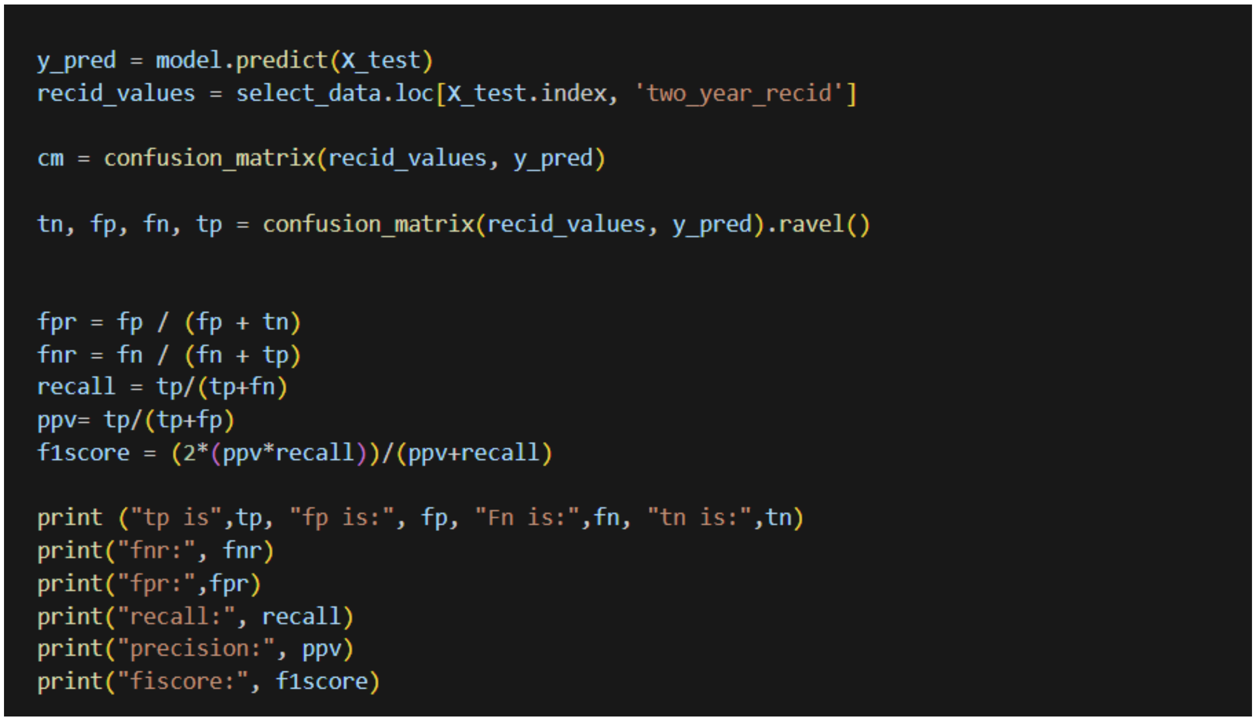
```
X_train, X_test, y_train, y_test = train_test_split( X, y, test_size= 0.20)
```

Python code to train the Logistic Regression Model:



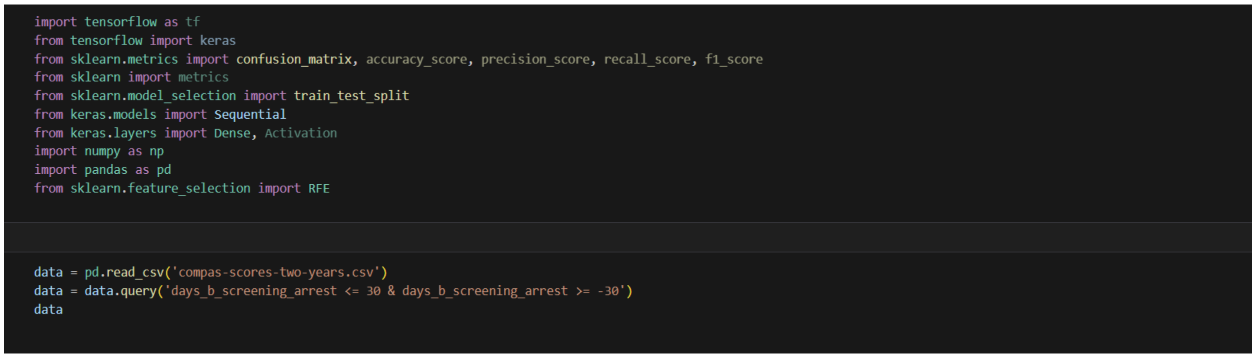
```
model = LogisticRegression()
model.fit(X_train, y_train)
```

Printing and calculation of accuracy Metrics:



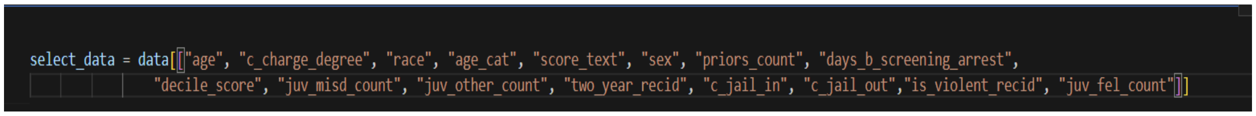
```python
y_pred = model.predict(X_test)
recid_values = select_data.loc[X_test.index, 'two_year_recid']

cm = confusion_matrix(recid_values, y_pred)

tn, fp, fn, tp = confusion_matrix(recid_values, y_pred).ravel()


fpr = fp / (fp + tn)
fnr = fn / (fn + tp)
recall = tp/(tp+fn)
ppv= tp/(tp+fp)
f1score = (2*(ppv*recall))/(ppv+recall)

print ("tp is",tp, "fp is:", fp, "Fn is:",fn, "tn is:",tn)
print("fnr:", fnr)
print("fpr:",fpr)
print("recall:", recall)
print("precision:", ppv)
print("fiscore:", f1score)
```

Python code to load data for a neural network model:

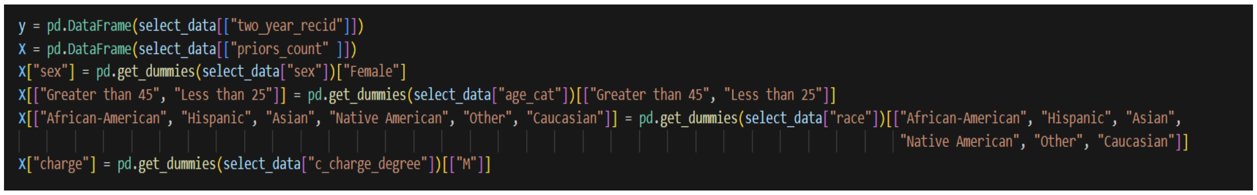


```python
import tensorflow as tf
from tensorflow import keras
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score
from sklearn import metrics
from sklearn.model_selection import train_test_split
from keras.models import Sequential
from keras.layers import Dense, Activation
import numpy as np
import pandas as pd
from sklearn.feature_selection import RFE
```

```python
data = pd.read_csv('compas-scores-two-years.csv')
data = data.query('days_b_screening_arrest <= 30 & days_b_screening_arrest >= -30')
data
```

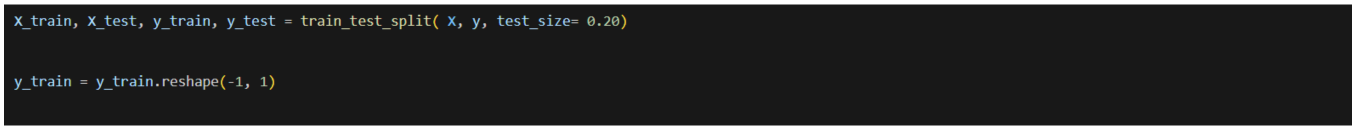


```python
select_data = data[["age", "c_charge_degree", "race", "age_cat", "score_text", "sex", "priors_count", "days_b_screening_arrest",
                    "decile_score", "juv_misd_count", "juv_other_count", "two_year_recid", "c_jail_in", "c_jail_out","is_violent_recid", "juv_fel_count"]]
```



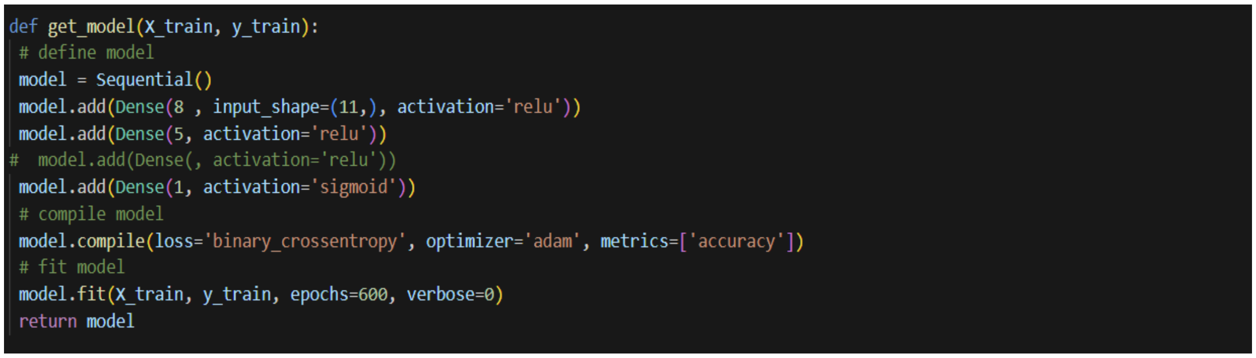
```python
y = pd.DataFrame(select_data[["two_year_recid"]])
X = pd.DataFrame(select_data[["priors_count" ]])
X["sex"] = pd.get_dummies(select_data["sex"])["Female"]
X[["Greater than 45", "Less than 25"]] = pd.get_dummies(select_data["age_cat"])[["Greater than 45", "Less than 25"]]
X[["African-American", "Hispanic", "Asian", "Native American", "Other", "Caucasian"]] = pd.get_dummies(select_data["race"])[["African-American", "Hispanic", "Asian",
                                                                                                                            "Native American", "Other", "Caucasian"]]
X["charge"] = pd.get_dummies(select_data["c_charge_degree"])[["M"]]
```

Splitting and reshaping data:

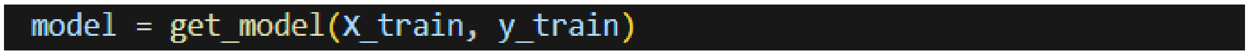


```
X_train, X_test, y_train, y_test = train_test_split( X, y, test_size= 0.20)


y_train = y_train.reshape(-1, 1)
```

Python code to train the final neural network model:

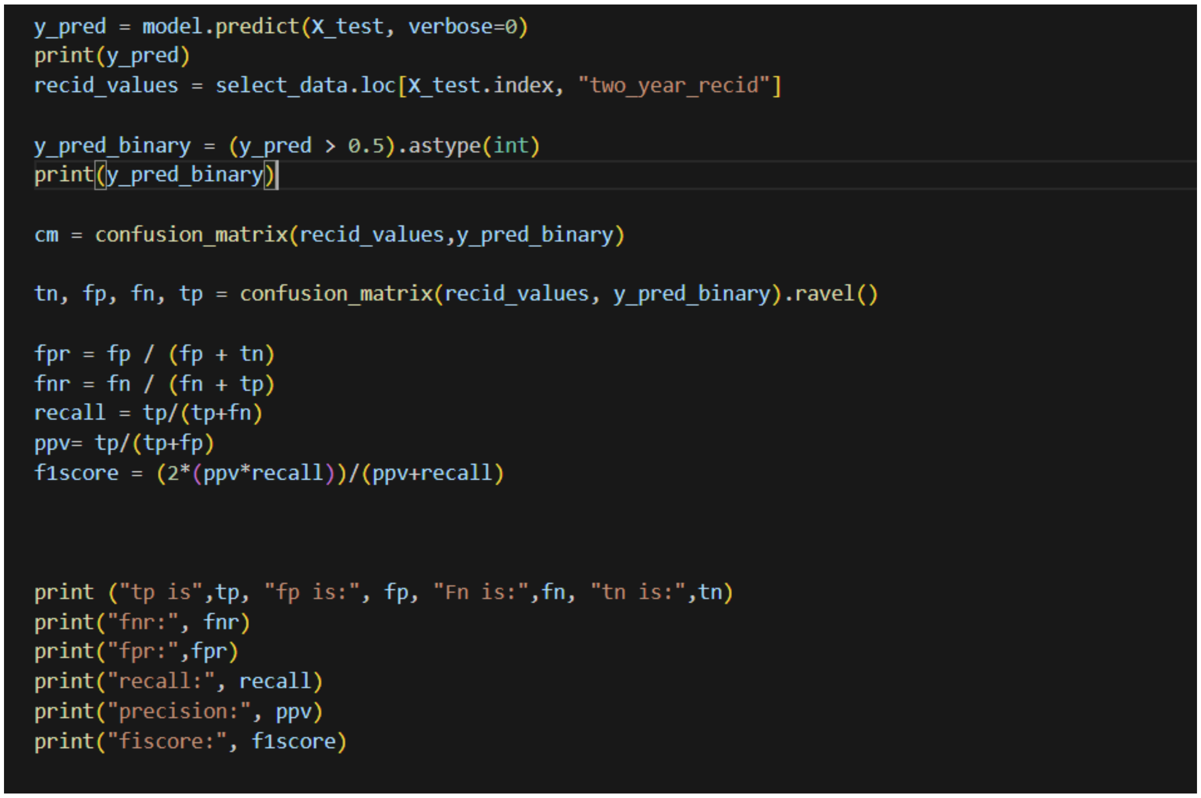


```
def get_model(X_train, y_train):
 # define model
 model = Sequential()
 model.add(Dense(8 , input_shape=(11,), activation='relu'))
 model.add(Dense(5, activation='relu'))
#  model.add(Dense(, activation='relu'))
 model.add(Dense(1, activation='sigmoid'))
 # compile model
 model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
 # fit model
 model.fit(X_train, y_train, epochs=600, verbose=0)
 return model
```



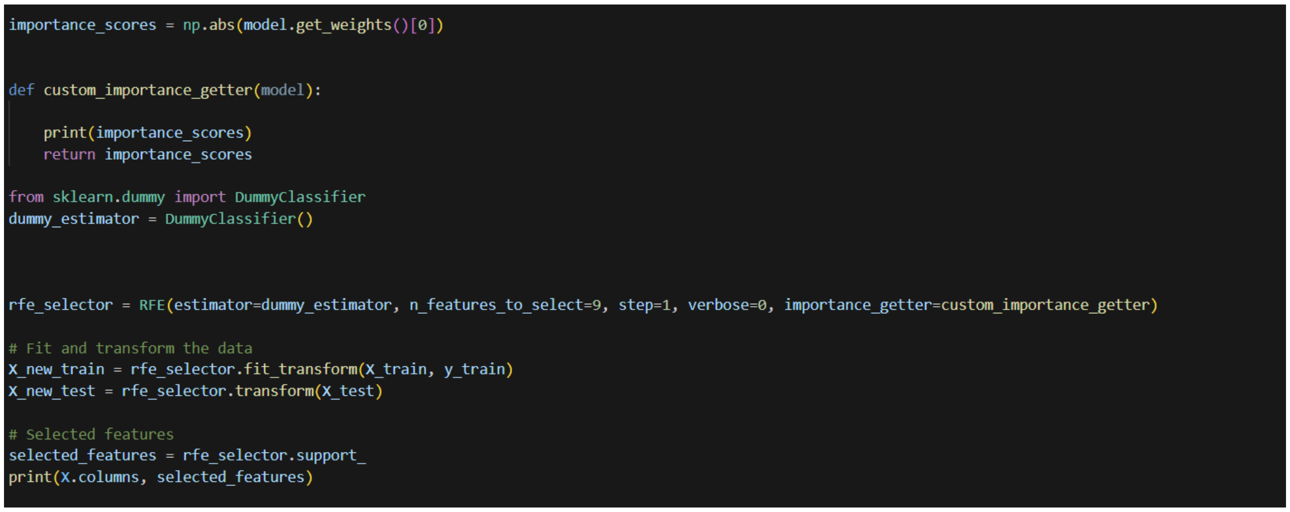
```
model = get_model(X_train, y_train)
```

Printing and calculation of accuracy Metrics:

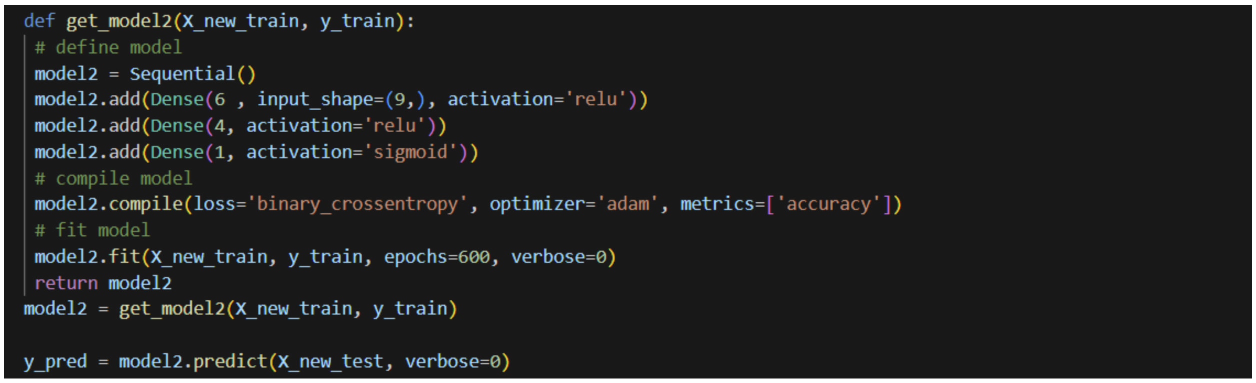


```
y_pred = model.predict(X_test, verbose=0)
print(y_pred)
recid_values = select_data.loc[X_test.index, "two_year_recid"]

y_pred_binary = (y_pred > 0.5).astype(int)
print(y_pred_binary)

cm = confusion_matrix(recid_values,y_pred_binary)

tn, fp, fn, tp = confusion_matrix(recid_values, y_pred_binary).ravel()

fpr = fp / (fp + tn)
fnr = fn / (fn + tp)
recall = tp/(tp+fn)
ppv= tp/(tp+fp)
f1score = (2*(ppv*recall))/(ppv+recall)



print ("tp is",tp, "fp is:", fp, "Fn is:",fn, "tn is:",tn)
print("fnr:", fnr)
print("fpr:",fpr)
print("recall:", recall)
print("precision:", ppv)
print("fiscore:", f1score)
```

Using RFE for NN model:

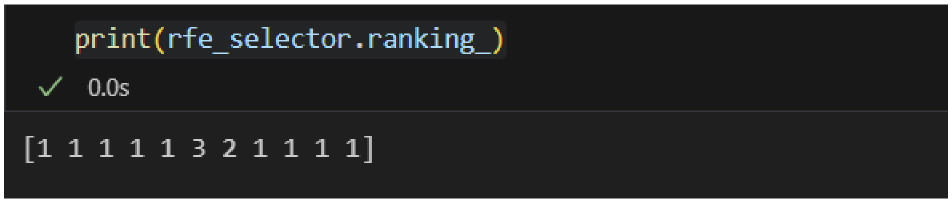


```
importance_scores = np.abs(model.get_weights()[0])


def custom_importance_getter(model):

    print(importance_scores)
    return importance_scores

from sklearn.dummy import DummyClassifier
dummy_estimator = DummyClassifier()



rfe_selector = RFE(estimator=dummy_estimator, n_features_to_select=9, step=1, verbose=0, importance_getter=custom_importance_getter)

# Fit and transform the data
X_new_train = rfe_selector.fit_transform(X_train, y_train)
X_new_test = rfe_selector.transform(X_test)

# Selected features
selected_features = rfe_selector.support_
print(X.columns, selected_features)
```

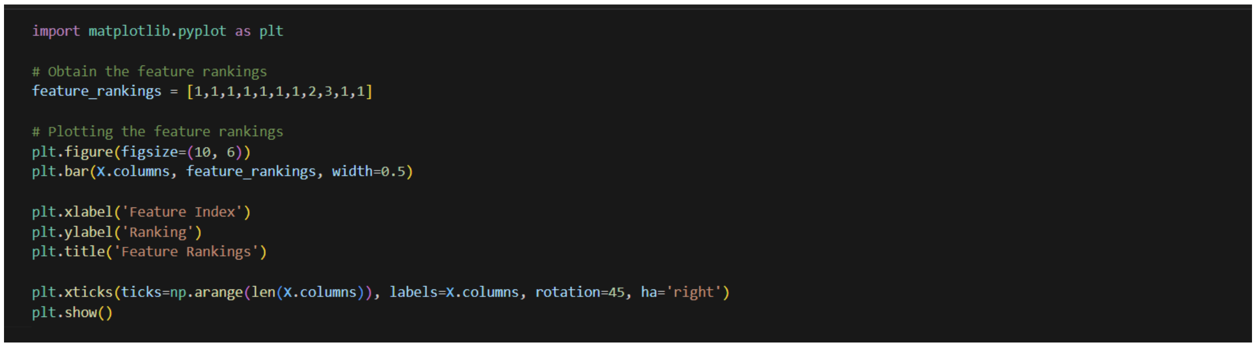


```
def get_model2(X_new_train, y_train):
    # define model
    model2 = Sequential()
    model2.add(Dense(6 , input_shape=(9,), activation='relu'))
    model2.add(Dense(4, activation='relu'))
    model2.add(Dense(1, activation='sigmoid'))
    # compile model
    model2.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
    # fit model
    model2.fit(X_new_train, y_train, epochs=600, verbose=0)
    return model2
model2 = get_model2(X_new_train, y_train)

y_pred = model2.predict(X_new_test, verbose=0)
```

Plotting the RFE rankings:

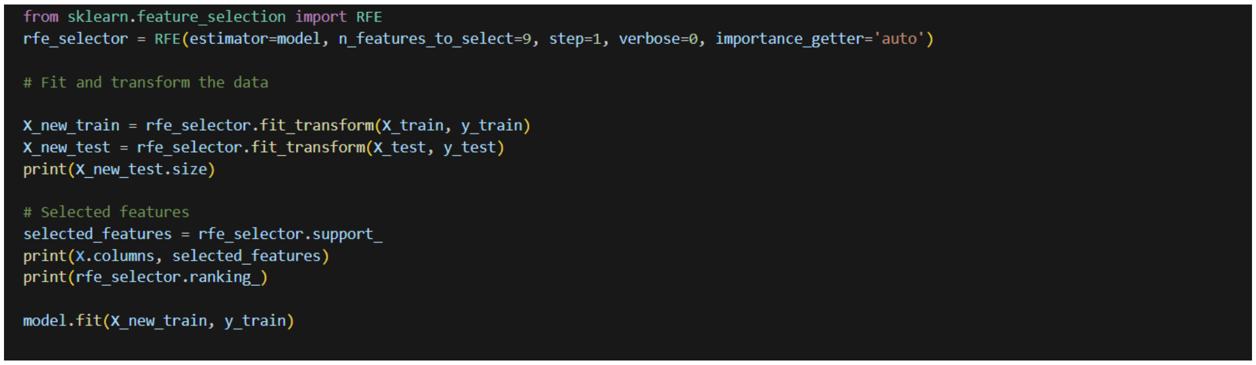


```
print(rfe_selector.ranking_)
✓ 0.0s

[1 1 1 1 1 3 2 1 1 1 1]
```



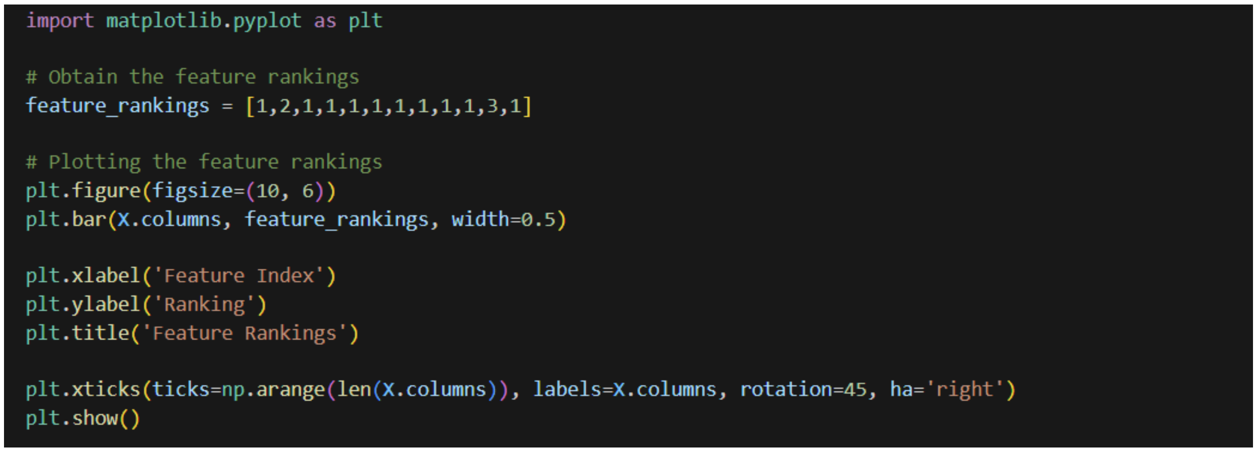
```
import matplotlib.pyplot as plt

# Obtain the feature rankings
feature_rankings = [1,1,1,1,1,1,1,2,3,1,1]

# Plotting the feature rankings
plt.figure(figsize=(10, 6))
plt.bar(X.columns, feature_rankings, width=0.5)

plt.xlabel('Feature Index')
plt.ylabel('Ranking')
plt.title('Feature Rankings')

plt.xticks(ticks=np.arange(len(X.columns)), labels=X.columns, rotation=45, ha='right')
plt.show()
```

RFE for LR model:



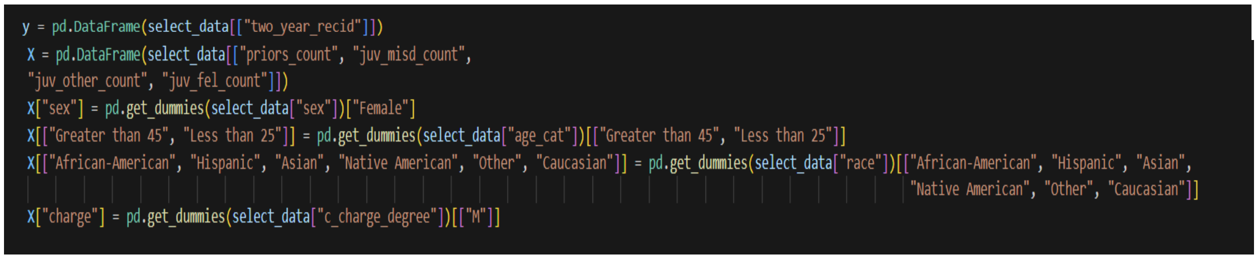
```
from sklearn.feature_selection import RFE
rfe_selector = RFE(estimator=model, n_features_to_select=9, step=1, verbose=0, importance_getter='auto')

# Fit and transform the data

X_new_train = rfe_selector.fit_transform(X_train, y_train)
X_new_test = rfe_selector.fit_transform(X_test, y_test)
print(X_new_test.size)

# Selected features
selected_features = rfe_selector.support_
print(X.columns, selected_features)
print(rfe_selector.ranking_)

model.fit(X_new_train, y_train)
```

Plotting rankings for LR model:

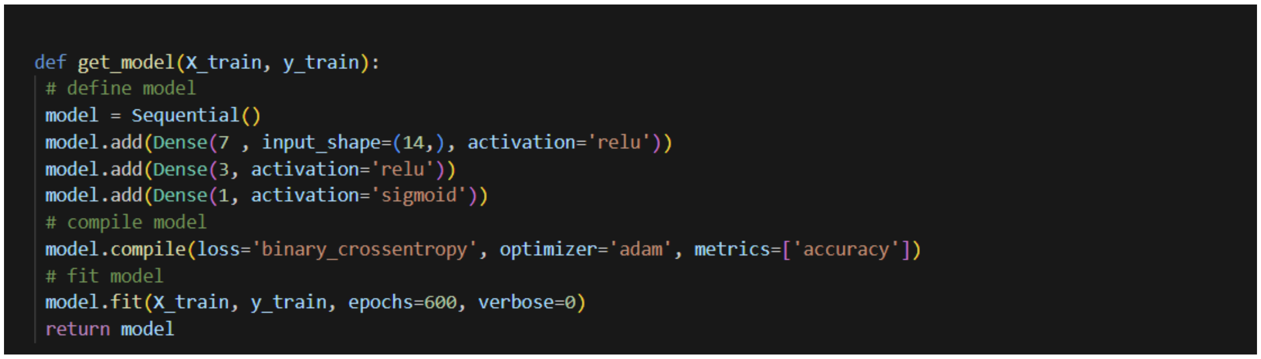


```
import matplotlib.pyplot as plt

# Obtain the feature rankings
feature_rankings = [1,2,1,1,1,1,1,1,1,1,3,1]

# Plotting the feature rankings
plt.figure(figsize=(10, 6))
plt.bar(X.columns, feature_rankings, width=0.5)

plt.xlabel('Feature Index')
plt.ylabel('Ranking')
plt.title('Feature Rankings')

plt.xticks(ticks=np.arange(len(X.columns)), labels=X.columns, rotation=45, ha='right')
plt.show()
```

Model architecture of NN with increased features:

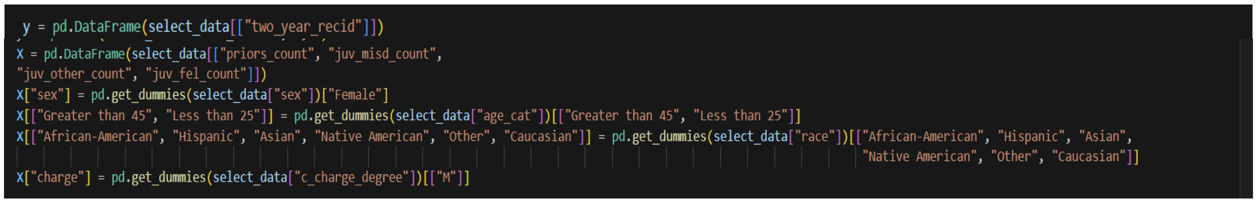


```
y = pd.DataFrame(select_data[["two_year_recid"]])
 X = pd.DataFrame(select_data[["priors_count", "juv_misd_count",
 "juv_other_count", "juv_fel_count"]])
 X["sex"] = pd.get_dummies(select_data["sex"])["Female"]
 X[["Greater than 45", "Less than 25"]] = pd.get_dummies(select_data["age_cat"])[["Greater than 45", "Less than 25"]]
 X[["African-American", "Hispanic", "Asian", "Native American", "Other", "Caucasian"]] = pd.get_dummies(select_data["race"])[["African-American", "Hispanic", "Asian",
                                                                                                              "Native American", "Other", "Caucasian"]]
 X["charge"] = pd.get_dummies(select_data["c_charge_degree"])[["M"]]
```



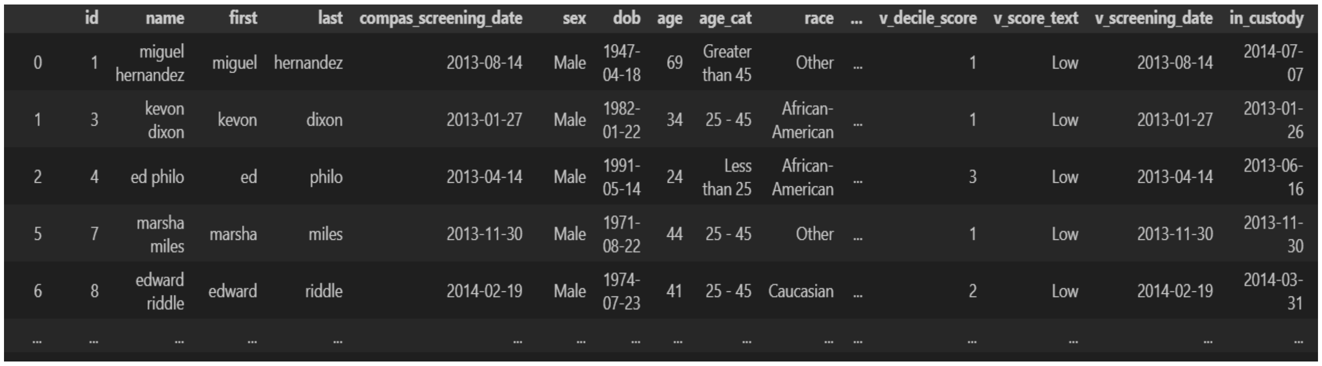
```
def get_model(X_train, y_train):
    # define model
    model = Sequential()
    model.add(Dense(7 , input_shape=(14,), activation='relu'))
    model.add(Dense(3, activation='relu'))
    model.add(Dense(1, activation='sigmoid'))
    # compile model
    model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
    # fit model
    model.fit(X_train, y_train, epochs=600, verbose=0)
    return model
```

Declaration of variables for LR model:



```
 y = pd.DataFrame(select_data[["two_year_recid"]])
X = pd.DataFrame(select_data[["priors_count", "juv_misd_count",
"juv_other_count", "juv_fel_count"]])
X["sex"] = pd.get_dummies(select_data["sex"])["Female"]
X[["Greater than 45", "Less than 25"]] = pd.get_dummies(select_data["age_cat"])[["Greater than 45", "Less than 25"]]
X[["African-American", "Hispanic", "Asian", "Native American", "Other", "Caucasian"]] = pd.get_dummies(select_data["race"])[["African-American", "Hispanic", "Asian",
                                                                                                              "Native American", "Other", "Caucasian"]]
X["charge"] = pd.get_dummies(select_data["c_charge_degree"])[["M"]]
```

Input data used:



	id	name	first	last	compas_screening_date	sex	dob	age	age_cat	race	...	v_decile_score	v_score_text	v_screening_date	in_custody
0	1	miguel hernandez	miguel	hernandez	2013-08-14	Male	1947-04-18	69	Greater than 45	Other	...	1	Low	2013-08-14	2014-07-07
1	3	kevon dixon	kevon	dixon	2013-01-27	Male	1982-01-22	34	25 - 45	African-American	...	1	Low	2013-01-27	2013-01-26
2	4	ed philo	ed	philo	2013-04-14	Male	1991-05-14	24	Less than 25	African-American	...	3	Low	2013-04-14	2013-06-16
5	7	marsha miles	marsha	miles	2013-11-30	Male	1971-08-22	44	25 - 45	Other	...	1	Low	2013-11-30	2013-11-30
6	8	edward riddle	edward	riddle	2014-02-19	Male	1974-07-23	41	25 - 45	Caucasian	...	2	Low	2014-02-19	2014-03-31
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...