Vehicle Fraud Prediction Project Report

Dan Bilsker

November 2022

1 Data Cleaning

I began by first scanning the data. I found in the age column that there were many zeroes. Obviously, a zero year old cannot purchase an insurance policy. So I deleted these rows. To make things go smoother, I changed the data from categorical to continuous numerical data(I saved my own data dictionary in the Excel comments and later learned that these comments were deleted, which I believe was due to saving the workbook in a csv and not file format;although it is easy to reconstruct the meaning of the variables by working backwards from my original Excel sheet, which I saved).I noticed later on the many entries had empty cells, which I deleted as well.

2 Data Tree Regressor

The first model I tried was the decision tree regressor. I initially input all data from the worksheet, including seemingly irrelevant features like marital status, day of the week, policy number etc. I split the test-training set into a ratio of 1:2. The training tree had fault as the root node, and Vehicle type and deductible as split nodes.The tree therefore believed that the type of car an insurance policy holder owned was a significant variable to use in the data to predict insurance fraud.I'm not sure if this makes sense on the human level. Maybe a car is indicative of a certain personality, but I doubt it. More concerning was the testing tree, which differed significantly from the training tree. The training tree sort of believed(it was inconsistent on this as well) that low deductibles best predicted fraud. Makes sense. If a person wants to defraud an insurance company, they need money, so they don't want to pay a lot in the form of a deductible. But the testing tree had a low deductible as having a zero likelihood of fraud.I tried the decision tree regressor again after deleting vehicle category from the data(since this is really redundant because policy type includes vehicle), but the training and testing tree were still inconsistent and not in alignment with any common sense. For example, there's no reason policy number, which is arbitrary, should predict fraud.

Thus I learned about some of the disadvantages of a decision tree regressor:it doesn't do a great job on continuous numerical data, and slight changes in data can result in a significant changes between training and testing tree.

3 Random Forest

The next logical step would be to try a random forest, which uses the results of many decision trees together. While the random forest did have a 94 percent accuracy rate. Insurance companies would mostly be concerned about catching potential fraud, so our key statistic was the number of frauds our model correctly predicted. Of the $2.8*10^2$ cases of actual fraud, our model only correctly selected three, a paltry sum.Eliminating all but the most relevant features from the data set did result in better performance but only by an insignificant amount.

4 Logistic Regression

Logistic regression seemed the next best guess. I standardized the data and again the accuracy was very high but now the model wasn't predicting any fraud at all. I had success after refining the data set to include only fault, deductible, and policy type and adjusting the threshold with the ROC curve.With the optimal threshold I was predicting 162 out of 278 of fraud in the test data. The model also wrongly predicted 2682 cases of fraud where there was none but in real life erroneously flagging a case as potentially fraudulent is still cost-efficient so long as the model is catching a sufficient number of real fraud cases. I used the lbfgs solver which is a tad bit less accurate than the newton-cg solver but computes the same algorithm faster by approximating the Hessian matrix. If I reduce the data set further to only two features like fault and deductibles, the model does not perform as well. The cost function of the training set is 0.20756309073344958, and the cost function of the test set is 0.19820254290404488.

5 Math for Logistic Regression

For logistic regression, we have the sigmoid function: $\hat{y} = 1/(1 + e^{-z})$

We see to minimize the following cost function:
Cost($\hat{y}$,y) $= -log(\hat{y})$ if $y = 1$
Cost($\hat{y}$,y) $= -log(1 - \hat{y})$ if $y = 0$

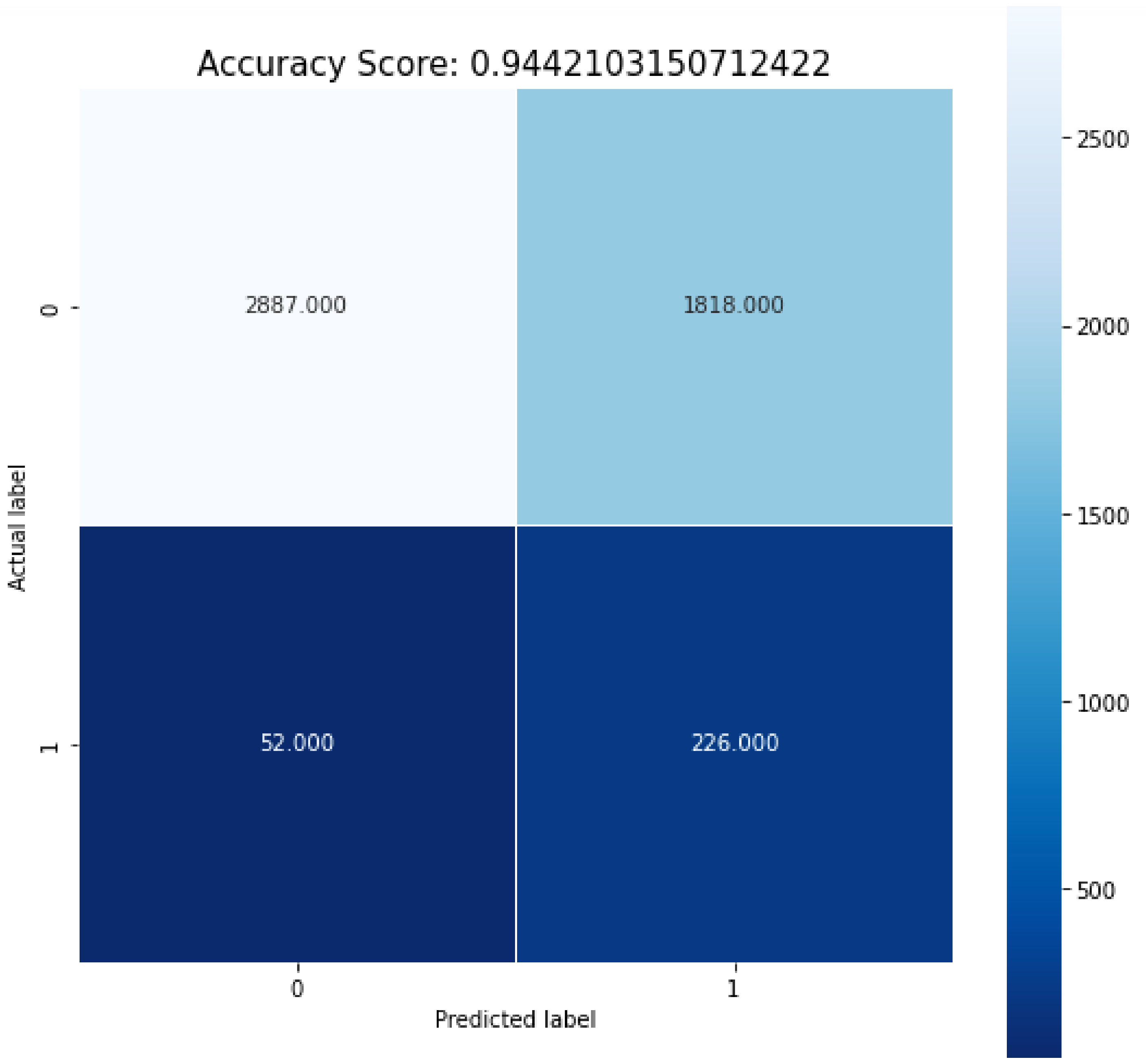
$\hat{y}$ is bounded between 0 and 1.The first equation captures the intuition that the cost to $\hat{y}$ should be high if $y = 1$ and $\hat{y}$ is zero, and the cost should be zero if $\hat{y}$ is one and $y = 1$.And the same logic applies for why this makes sense in the case of $y = 0$

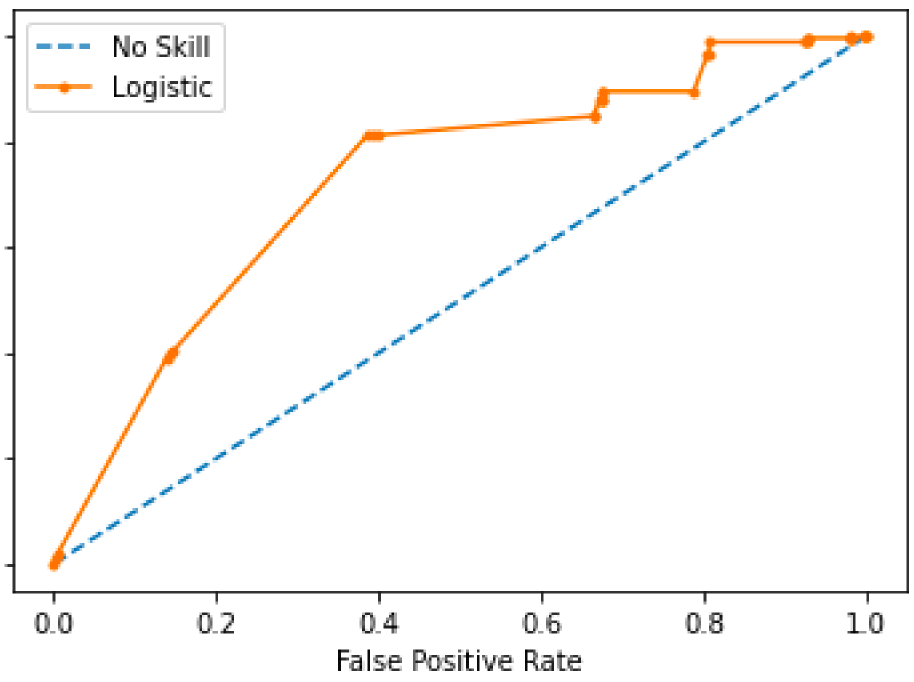
We can put this into one equation.

$Cost(\hat{y}, y) = -y * log(\hat{y}) - (1 - y) * log(1 - \hat{y})$ and thus
$Cost(\hat{y}, y) = -(1/n)\Sigma_{i=0}^{n}(y_i * log(\hat{y_i})) + (1 - y_i) * log(1 - \hat{y_i})$

6 Conclusion

The below is the confusion matrix and ROC curve for the best result in logistic regression. It might be worth trying lasso regression as well.


Accuracy Score: 0.9442103150712422
2887.000
1818.000
52.000
226.000
Actual label
Predicted label
0
1
0
1
2500
2000
1500
1000
500


No Skill
Logistic
0.0
0.2
0.4
0.6
0.8
1.0
False Positive Rate