Credit Approval Dataset:
Predicting Approval Using Neural Networks

Filip Reierson

16/10/2021

Contents

Introduction

The credit approval dataset was obtained from the UC Irvine Machine Learning Repository (Dua and Graff 2017). The attributes in the dataset are modified in such a way that the data is de-identified, while still allowing us to use the data for education and research. In this report we explore the data and then assess neural networks trained using various approaches.

Data Processing

The dataset contains both continuous and nominal attributes which are summarised in Table 1. To prepare the data for analysis nominal data were coded as positive integers according to alphabetical order of factor levels. Logical data was changed to numeric with true values being 1 and false values being 0. The target classes were binary encoded with "+" as 1 and "-" as 0. All columns were normalised to the 0 to 1 range. Finally, we replaced missing values by -1 to distinguish these values from the non-missing data.

Table 1: Attributes in the credit approval dataset.

Attribute	Data Type	Possible Values
A1	nominal	b, a
A2	continuous	$\mathbb{R}$
A3	continuous	$\mathbb{R}$
A4	nominal	u, y, l, t
A5	nominal	g, p, gg
A6	nominal	c, d, cc, i, j, k, m, r, q, w, x, e, aa, ff
A7	nominal	v, h, bb, j, n, z, dd, ff, o
A8	continuous	$\mathbb{R}$
A9	logical	t, f
A10	logical	t, f
A11	continuous	$\mathbb{R}$

Attribute	Data Type	Possible Values
A12	logical	t, f
A13	nominal	g, p, s
A14	continuous	$\mathbb{R}$
A15	continuous	$\mathbb{R}$
Approved	logical	t, f

Data Exploration

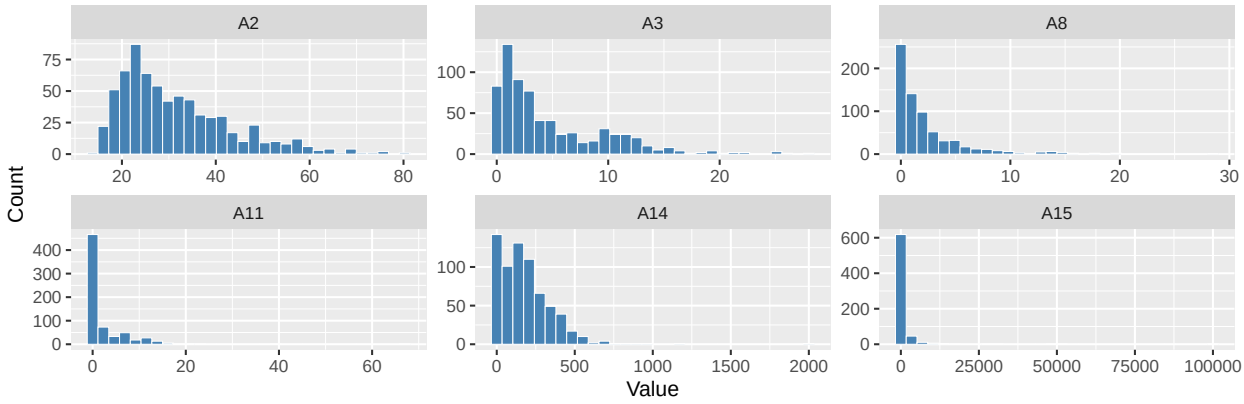

Figure 1: Histograms of continuous features.

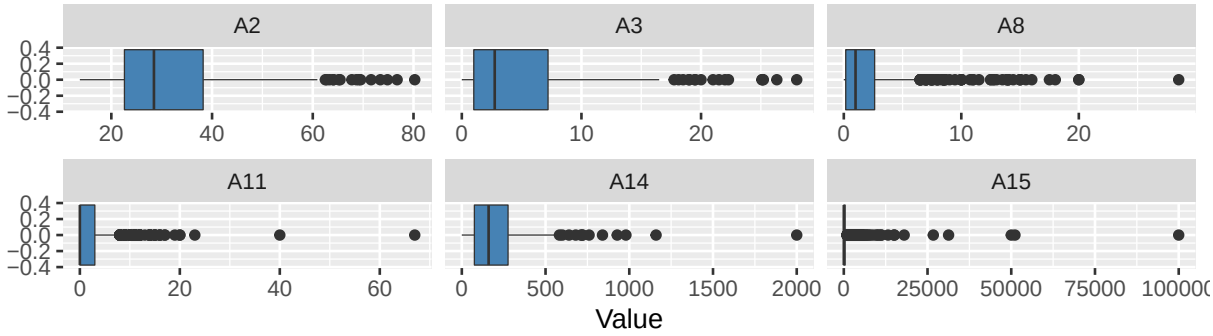

Figure 2: Boxplots of continuous features

Histograms of continuous features are shown in Figure 1. Based on the histograms we see that all distributions are right skewed, but to varying degrees. We also observe that A3 appears to be bimodal, with the second peak appearing around 10. The boxplots in Figure 2 gives us an idea of central tendency and technical outliers. We see that A15 appears to be zero inflated which is why many values are displayed as technical outliers by the boxplot. We could imagine A15 may be something like existing credit card debt, many will have close to 0 while a few outliers will have large compounding debts. Barplots are used to visualise nominal and logical features in Figure 4. We see that A6 and A7 have many categories, so each category doesn't have many observations to learn from. We also see that A13 has a very imbalanced distribution between categories. However, the target "Approved" is reasonably balanced so simple randomisation can be used for train test split.

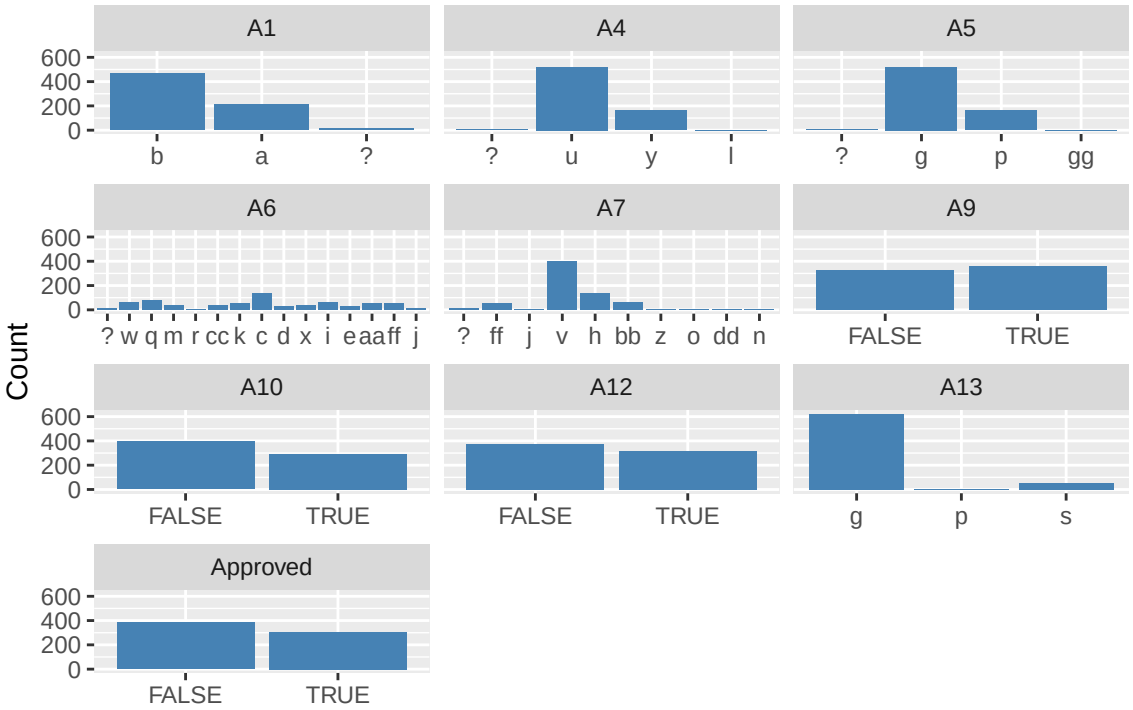

Figure 3: Barplots of nominal and logical features

A heatmap of all features after making nominal variables numeric and normalising all features is shown in in Figure 4. We can see that A9 and to a lesser extent A8, A10, and A11 are positively correlated with the target. On the other hand, A4, A5, and A6 are negatively correlated with the target. The other features don't have very strong correlations. However, it is worth noting that we shouldn't expect the relationship between the target and features to be linear, especially nominal values. Finally, the heatmap also shows that A4 and A5 are highly collinear. In fact A4 and A5 are perfectly collinear, i.e., it adds no information, therefore we dropped A5 from the data.

Now that we have some understanding of the data and have processed it, we can move on to modelling.

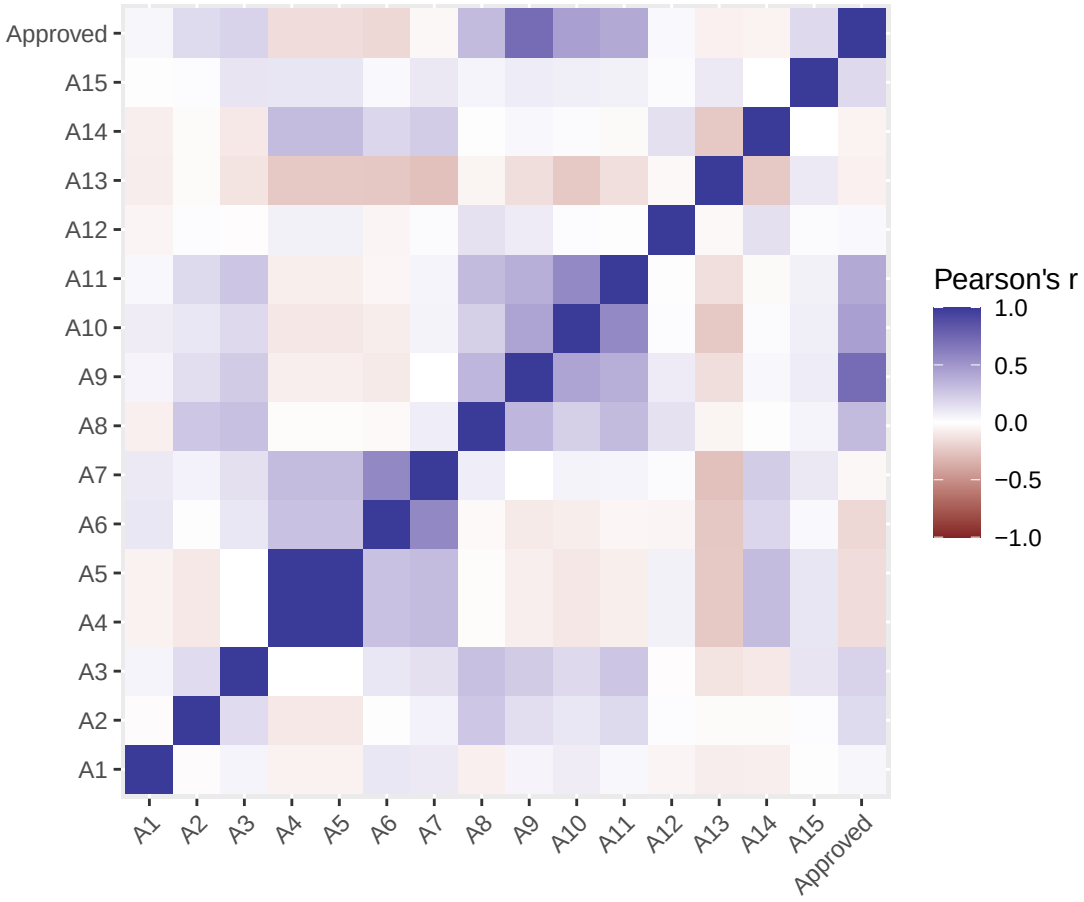

Figure 4: Heatmap of feature correlation matrix.

Modelling

We start by selecting the number of neurons in the hidden layer using the formula as a guide (hobs 2015):

$$N_h = \frac{N_s}{\alpha \times (N_i + N_o)}$$

After dropping collinear features there are $N_i = 14$ input neurons. The output is binary encoded, so there is only one output neuron, so $N_o = 1$. There are $N_s = 345$ samples in the training data set. We may arbitrarily select scaling factor $\alpha = 4$. As a result of these numbers we may choose to use $N_h = 5.75 \approx 6$ as a starting point. Experimentation indicated that this configuration works fine. Based on experimentation I found that 100 epochs was sufficient with margin to let both Adam and SGD converge, so I will use 100 epochs. Also to make SGD converge a bit quicker I used a learning rate of 0.1.

We split the data into train and test with 50% in each. We trained the neural networks with Adam and SGD, using 10 different initial positions in weight space, and a single hidden layer with six neurons. The performance of both optimisers are shown in Table 2. The table shows that SGD actually outperformed Adam in terms of average train and test performance over the 10 runs.

The performance using L2 compared to dropout regularisation running an equivalent model to above is shown in Table 3. Interestingly the difference isn't very large between various types of regularisation, possibly due to the low complexity of the neural network.

For the first run of the experiment we get the confusion matrix in Table 4. The confusion matrix shows a fairly high false positive rate. Neither L2 nor dropout method appear to suffer from over fitting based on the accuracy and confusion matrix.

For the first run of the experiment the AUC-ROC for each regularisation method is shown in Figure 5. The AUC metric is shown in Table 5.

4. The confusion matrix shows a fairly high false positive rate. Neither L2 nor dropout method appear to suffer from over fitting based on the accuracy and confusion matrix.

Table 2: Performance of Adam and SGD for training a 6 neuron single hidden layer neural network.

optimiser	train_accuracy	test_accuracy
Adam	0.8713	0.8510
SGD	0.8730	0.8614

Table 3: Performance of regularisation strategies for training a 6 neuron single hidden layer neural network using Adam optimiser.

regularisation	train_accuracy	test_accuracy
L2 (0.01)	0.8649	0.8446
Dropout (0.1)	0.8780	0.8542
None	0.8713	0.8510

Limitations and further research

A limitation of this investigation is that we don't know what the business impact of a false positive and a false negative would be, which would be an important consideration if a model like this were to be used in practice. For example, approving credit which has a high risk of not being paid back is likely to be much

Table 4: Confusion matrix for L2 and dropout-based regularisation.

	Reference			
	Train		Test	
	Approved	Rejected	Approved	Rejected
L2				
Predicted Approved	121	20	131	29
Predicted Rejected	28	176	27	158
Dropout				
Predicted Approved	119	11	129	23
Predicted Rejected	30	185	29	164

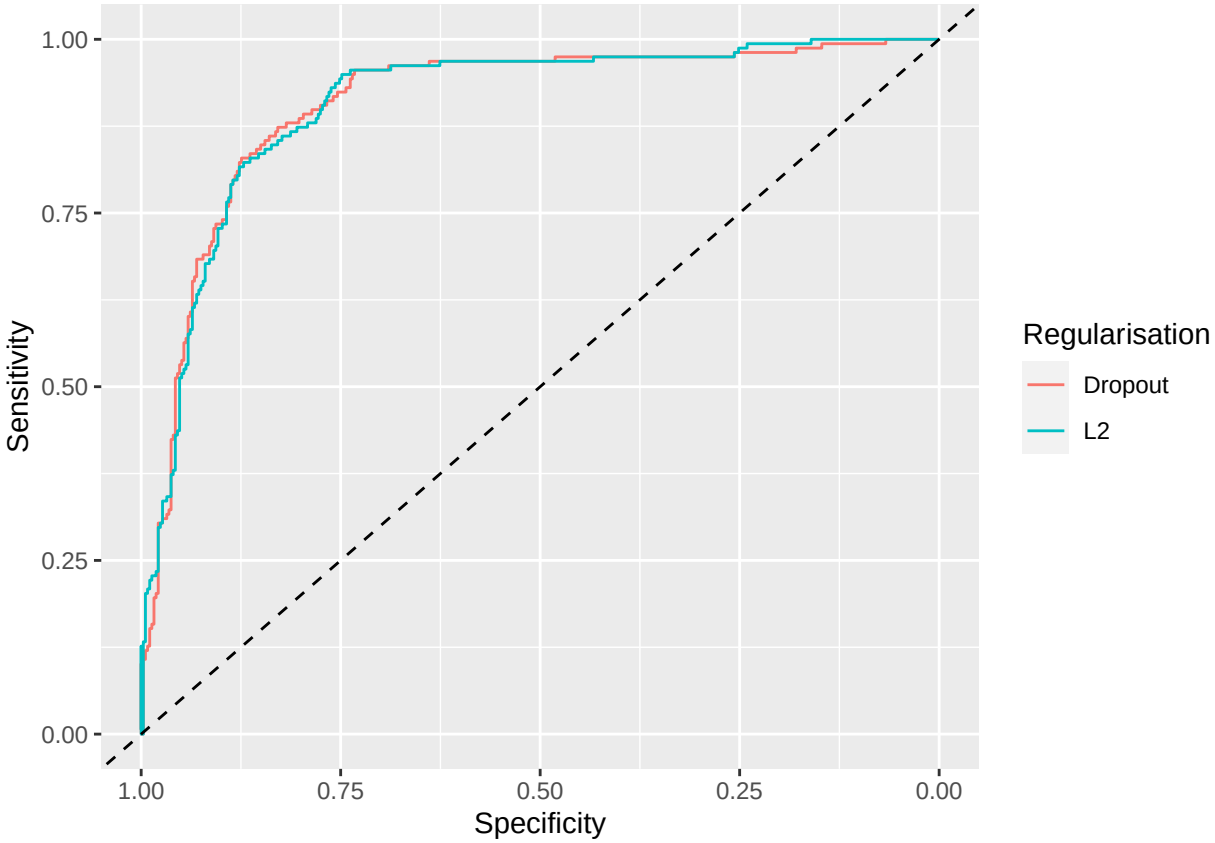

Figure 5: AUC-ROC curves for L2 and dropout-based regularisation.

Table 5: AUC metric for L2 and dropout-based regularisation.

Regularisation	AUC
L2	0.9048
Dropout	0.9050

more problematic than missing out on a potential borrower. If we had this information we could define a new loss function which better reflect the needs of the use case.

Additionally since we don't know what the nominal and continuous variables represent we can't use domain knowledge to engineer features which could improve the usefulness of the model.

Conclusion

We can see that regularisation has a fairly negligible impact on how well the model performs due to its simplicity. Both Adam and SGD perform similarly with a slight edge to SGD in terms of accuracy. If the model were to be used we would be able to predict whether credit would be approved with an ~85% accuracy if the new data is similar to the existing data. However, the false positive rate is too high for full automation, but the neural network could be useful for screening given the fairly low false negative. Other cut-offs than 0.5 could be chosen to achieve the desired balance between sensitivity and specificity.

References

Dua, Dheeru, and Casey Graff. 2017. "UCI Machine Learning Repository." University of California, Irvine, School of Information; Computer Sciences. http://archive.ics.uci.edu/ml.

hobs. 2015. "How to Choose the Number of Hidden Layers and Nodes in a Feedforward Neural Network?" *Cross Validated*. https://stats.stackexchange.com/questions/181/how-to-choose-the-number-of-hidden-layers-and-nodes-in-a-feedforward-neural-netw.