Assignment #3

Jonah Muniz

Abstract

The purpose of this research is to determine if unsupervised learning methods can be leveraged to improve the performance of supervised learning techniques. This research will look to see if running the German credit card dataset through an autoencoder prior to training and validating a logistic regression model predicting customer status will led to better performance. A baseline logistic regression model will be trained and validated to act as a baseline. Overall model performance will be evaluated on a precision-recall-based method. Conclusion of this research will show whether leveraging autoencoders prior to building a logistic regression model on the German credit card dataset will improve model performance.

Keywords: Unsupervised Learning, Supervised Learning, Logistic Regression, Autoencoders, Credit Worthiness Prediction

Introduction

When a credit card company is deciding on extending credit ensuring the customer will be a good customer is very important. Extending credit to bad customers exposing the company to a lot of risk. Previous studies at the company have shown that extending credit to bad customers is five times the cost of granting credit to good customers. Due to this the company wants to develop an accurate model that can predict whether a customer is a good or bad customer. The company is also curious if leveraging both unsupervised and supervised learning methods will lead to a better performing model then if only supervised learning methods were used. Improving model overall performance even by the most minimal percentages will lead to a large decrease in cost and risk for the credit card company. Autoencoders will be the primary unsupervised learning method used in this research. Logistic

regression will be the supervise learning method used in both the baseline model as well as the unsupervised + supervised learning model.

Literature Review

Utilizing autoencoders to try to improve supervised learning model performance is not a novel concept. Similar analysis has been performed even in the credit card business. The first research that will be covered was analyzing three different supervised learning methods and wanted to measure their performance at detecting fraudulent activity. The research also wanted to see how incorporating autoencoders prior to the training process would impact model performance. The research analyzed and compared the performance of six different model configurations: logistic regression, autoencoder plus logistic regression, gradient boosted trees, autoencoder plus gradient boosted trees, deep learning, and autoencoders plus deep learning. The research concluded that adding autoencoders to the models slightly boosted predictive power (Rushin and Stancil 2017). The next piece of research wanted to see if adding a stacked autoencoder prior to performing logistic regression would lead to better classification performance than the popular support vector machine classifier. The dataset used in this research contained hyperspectral data. Due to this incorporating a stacked autoencoder with rectified linear units (ReLU) as the activation function in conjunction with a logistic regression model led to a better performing classification model then just a support vector machine classifier (Xing, Ma and Yang 2015).

Methods

In order to understand the value of the autoencoder, the performance of the logistic regression classification without the autoencoder must be assessed. A baseline regression model must be trained, validated, and evaluated. Prior to any models being trained first the data must be ingested into R. Once the data has been ingested into R exploratory data analysis and data cleansing will be performed. Through exploratory data analysis it can be seen that personal_status female_single has zero records. Due to this the female_single personal_status can be removed. The purpose field can also be

consolidated to reduce the amount of purpose fields. Analyzing the summary statistics for each feature in the dataset it can be seen that the credit amount has a large range. A new feature, log_credit_amount, can be created and inserted into the dataset to reduce the range of values. A cost matrix can now be created to capture the cost effect of giving credit to a bad customer.

Now that all data preparations have been completed k-fold validation can be conducted to train and validate the baseline logistic regression model. The model will be trained and validated on 5 folds. The measure of performance will be precision, recall, F1 score, and an overall cost metric due to bad customers. The model with the highest precision, recall, F1 score and lowest cost will be considered the better performing model. Now that the baseline model and performance has been obtained the autoencoder will need to be created and fit to the data.

The autoencoder will consist of an encoder and decoder. The encoder portion of the autoencoder will reduce the dataset from its original 44 features to 15 features. The decoder will then decode the 15 features and will reconstruct the data to 44 features. The encoder will consist of four layers. The activation function used will be the tanh activation function. The decoder will consist of four layers and will also use the tanh activation function. Once the autoencoder is compiled and fit to the dataset the encoder is saved.

The encoder is then used to reduce the train and test dataset to 15 features. These new values are then used as the base training and test dataset for the logistic regression model. The customer status is then combined with the new training and test datasets to enable training and validation. The logistic regression is trained and validated in the same manner that the base model was. The same metrics were generated to evaluate the new encoder plus logistic regression model. F1 score and cost are the primary metrics the two models will be evaluated and compared on.

Results

The German credit card dataset was ingested into R using the readARFF function. This dataset was then converted to a data frame and called credit. A cost matrix and cutoff point value was also created. The cutoff point for labeling whether a customer will be good or bad is 0.086. The cutoff point is 0.086 because a bad customer costs the bank 5 times more, (1/5 = 0.086). Exploratory analysis was then performed on the credit data frame and some feature engineering was performed. The final dataset prior to the training and validation of the base model consisted of 1000 observations and 22 features.

The baseline logistic regression model was trained and validated using k-fold validation. 5 folds were used to train and validate the dataset. The precision, recall, f1 score and cost was determined for each fold. Two different cutoff points were used to evaluate the model's performance, 0.5 and the cutoff value 0.086. A table of the baseline logistic regression's performance across both cutoff points can be seen in the appendix as Table 1. The baseline model was trained and validated on 19 features. The list of features used can be found in the appendix as Figure 1. The average F1 score for the baseline model using the cutoff value of 0.086 was 0.532. The average cost for the baseline model using the cutoff value of 0.086 was 116. Now that the baseline model has been created and evaluated the encoder and decoder can be compiled and fitted.

The encoder's structure consists of four layers. The first layer is the input layer with the shape of the original dataset. The next layer has a shape of 30, reducing the dataset from 44 to 30 features. The third layer in the encoder has a shape of 20, reducing the dataset from 30 to 20 features. The last layer has a shape of 15, reducing the dataset from 20 to 15 features. The decoder's structure is a mirror image of the encoders. The first layer of the decoder has a shape of 15. The next layer has a shape of 20. The third has a shape of 30. The last layer of the decoder is the original datasets shape of 44. Both encoder and decoder where then combined to create the autoencoder. The tanh function was used as the activation function for each layer of the encoder and decoder. The autoencoder is then compiled and fit to the dataset. The optimizer for the autoencoder is adam, the loss function is the mean squared error

and accuracy was calculated for each epoch in addition to loss. The autoencoder was fit on 100 epochs. The lowest loss value for the autoencoder was 6.7 with a val_accuracy of 0.8253. A plot of the loss and accuracy of each epoch can be found in the appendix as Figure 2. A summary of both the encoder and decoder structure can be found in the appendix as Figure 3. Now that the autoencoder has been compiled and fitted the encoder can be used to transform the training and testing dataset prior to logistic regression model training and validation.

The encoder was used to create a new dataset transforming the original dataset to 15 features. The customer status for each observation was then combined with the new dataset to create the final dataset that will be used to train the autoencoder plus logistic regression model. The new model was trained and validated using k-fold validation and 5 folds. The same metrics and cutoff values were used to evaluate the model. A table of the autoencoder plus logistic regression model's performance across both cutoff points can be seen in the appendix as Table 2. The average F1 score for the model using the cutoff value of 0.086 was 0.46. The average cost for the baseline model using the cutoff value of 0.086 was 142.2.

Conclusions

Comparing the performance of the two models it can be seen that the baseline model actually outperformed the autoencoder plus logistic regression model for predicting a customer's status. The baseline model outperformed the autoencoder model in F1 score and cost. The baseline model has an F1 score that was higher then the autoencoder model by 0.07 and a cost that was lower by 36. The baseline model outperforming the autoencoder model is shocking at first but when looking at external research it is not uncommon. Autoencoders have been found to marginally improve logistic regression model performance. Autoencoder's main benefit is that it reduces the amount of noise in a dataset. If the dataset has a lot of noise that will reduce the performance of the logistic regression model. If the dataset does not have a lot of noise the autoencoder will not provide much improvement in

performance. Another reason why the baseline outperformed the autoencoder model is due to the number of features that were used to train and validate. The baseline model was trained and validated using 19 features while the autoencoder reduced the training and validation dataset to 15 features. These two factors can be responsible for why the baseline model outperformed the autoencoder model. For the German credit card dataset using autoencoders prior to performing logistic regression did not improve the overall model performance.

Appendix

baseprecision	baserecall	basef1Score	basecost	ruleprecision	rulerecall	rulef1Score	rulecost
0.538	0.475	0.505	179	0.362	0.864	0.510	130
0.607	0.557	0.581	157	0.418	0.967	0.584	92
0.592	0.509	0.547	160	0.338	0.930	0.495	124
0.609	0.475	0.533	173	0.372	0.932	0.531	113
0.652	0.469	0.545	186	0.381	0.922	0.539	121

Table 1: *Table showing the performance of the baseline logistic regression model.*

```
credit_model = "class ~ checking_status + duration +
    credit_history + purpose + log_credit_amount + savings_status +
    employment + installment_commitment + personal_status +
    other_parties + residence_since + property_magnitude +
    age + other_payment_plans + housing + existing_credits +
    job + num_dependents + own_telephone"
```

Figure 1: *List of features used to train and validate the baseline logistic regression model.*

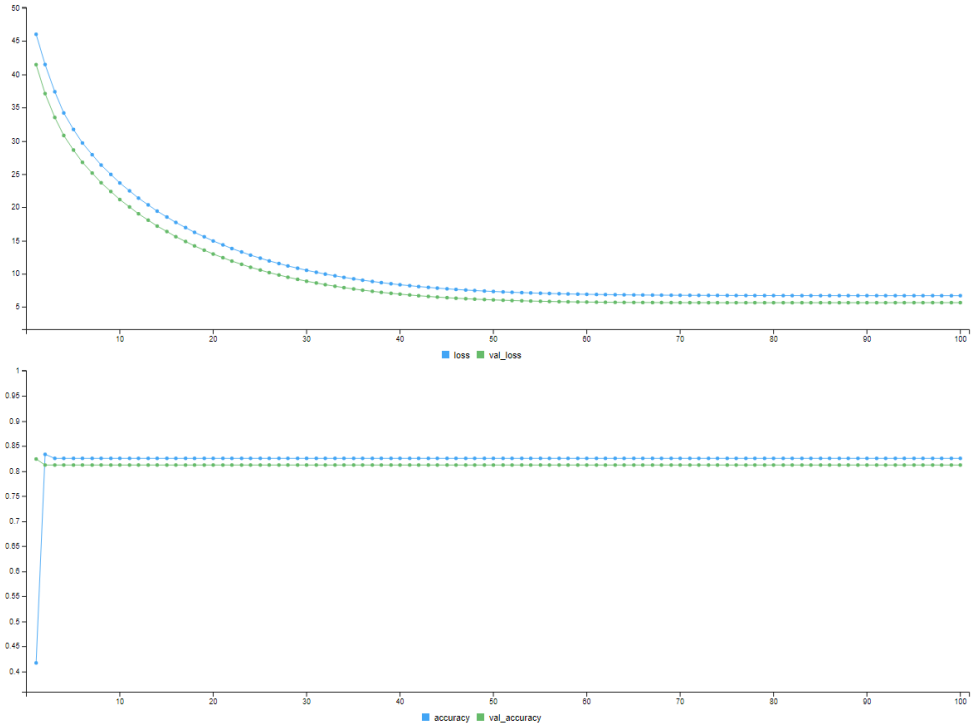

Figure 2: *The Loss and Accuracy of the autoencoder at each epoch*

```
> summary(encoder)
Model: "model_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_3 (InputLayer)         [(None, 44)]              0
_________________________________________________________________
dense_8 (Dense)              (None, 30)                1350
_________________________________________________________________
dense_7 (Dense)              (None, 20)                620
_________________________________________________________________
dense_6 (Dense)              (None, 15)                315
=================================================================
Total params: 2,285
Trainable params: 2,285
Non-trainable params: 0
_________________________________________________________________
> summary(decoder)
Model: "model_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_4 (InputLayer)         [(None, 15)]              0
_________________________________________________________________
dense_11 (Dense)             (None, 20)                320
_________________________________________________________________
dense_10 (Dense)             (None, 30)                630
_________________________________________________________________
dense_9 (Dense)              (None, 44)                1364
=================================================================
Total params: 2,314
Trainable params: 2,314
Non-trainable params: 0
_________________________________________________________________
```

Figure 3: *Summary of the encoder and decoder structure*

	encodebaseprecision	encodebaserecall	encodebasef1Score	encodebasecost	encoderuleprecision	encoderulerecall	encoderulef1Score	encoderulecost
1	0.579	0.183	0.278	253	0.300	1.000	0.462	140
2	0.500	0.082	0.141	285	0.305	0.984	0.465	142
3	0.312	0.082	0.130	291	0.306	0.984	0.467	141
4	0.727	0.121	0.208	293	0.330	0.970	0.492	140
5	0.778	0.135	0.230	227	0.260	1.000	0.413	148

Table 2: *Table showing the performance of the autoencoder plus logistic regression model.*

References

1. Rushin, Gabriel, and Cody Stancil. “Horse Race Analysis in Credit Card Fraud—Deep Learning, Logistic Regression, and Gradient Boosted Tree.” IEEE Xplore temporarily unavailable. Accessed November 2, 2021. https://ieeexplore.ieee.org/abstract/document/7937700.
2. Xing, Chen, Li Ma, and Xiaoquan Yang. “Stacked Denoise Autoencoder Based Feature Extraction and Classification for Hyperspectral Images.” Journal of Sensors. Hindawi, November 30, 2015. https://www.hindawi.com/journals/js/2016/3632943/.