Stein Dijkstra
Jelle Willekes
Liza Mints

Group 77

Report

In this report we train and select and algorithm for the preprocessed "Porto Seguro's Safe Driver Prediction" dataset [15]. The datasets consists of a binary target variable and 91 predictive variables. The goal of the algorithm is to predict the binary target variable, whether an individual claims insurance, with the highest possible accuracy. The hyper parameter optimization and model selection is based on the procedure described in section 3.7 of [13]. Starting with separating the data in a test (20%) and training (80%) set, we select the best hyperparameters using gridsearch along with 3-fold cross validation (CV) on the training set. Then the model selection and expected out of sample accuracy is computed using the independent test set. The result of the test set such as accuracy and optimal set of hyper parameters are presented Table 1.

The best model used to predict the target value is a feedforward Neural Network [3]. Due to computational constraints we perform a nested gridsearch. The specific details of the gridsearches can be found in the appendix. In the first gridsearch we optimise the batch size and the number of neurons in the layer. Our found optimal number of neurons is in line with prior research which show a model with $n = 91$ input variables perform best with $\frac{4n^2+3}{n^2-8} \approx 4$ neurons in the hidden layer [7]. Additionally, we chose the number of neurons to be powers of 2, due to computational efficiency. Moreover, we found that networks with a higher batch size and lower number of neurons in the layer achieve higher accuracy.

Next, using the result of the first gridsearch, we further optimize dimension of our Neural Network. In general, 2 hidden layers are sufficient to enable a feed forward network to model any arbitrary function between the input variables [5]. Hence, we analyzed any combination between a few hidden layers (1, 2, 3) and a low number of neurons (4, 8, 16) in the hidden layers. We find that networks with decreasing number of neurons in a multiple layer networks outperform those with increasing number of neurons in accordance with [4].

In the third gridsearch, the activation functions (ReLU, Tanh, Sigmoid and Softplus), initialization functions (uniform, normal) and optimizers (SGD, RMSprop, Adam) that achieve the highest test score were analyzed. We found that a ReLU activation function with a uniform initialization function and a RMSprop optimizer achieve the highest accuracy.

Besides other models we optimized a random forest (RF) using the scikit-learn package [12]. The found optimal hyperparameter are: number of estimators 200, max features log2, criterion entropy, max depth 10, and minimum sample split 40. The initial chosen values for the grid search are based on [8]. Additionally, we estimated an AdaBoost classifier (AB tree), with as the base classifier a decision tree. Similarly to the random forest we use the scikit-learn package [12]. Since the Adaboost classifier uses a base classifier we have two types of the hyper parameters, those corresponding to the boosting and to the decision tree. The optimal values for the boosting are: number of estimators 10, and learning rate 0.5. The optimal hyper parameters of the decision tree are: max features auto, criterion entropy, max depth 2, and minimum sample split of 40. These chosen values are based on [2].

Similar to the winner of the original competition [10], we have also tried to combine multiple models using a simple linear averaging. However we have found no combination of models for which we obtain a better result than the best neural network. Since the tasks we were provided was to estimate an algorithm with the highest accuracy, we selected the neural network as our model. The accuracy of this neural network on our independent test set was 58.74% and thus we expect our out of sample accuracy to also be 58.44%.

Table 1: Test set accuracy and set of best hyper parameters for the estimated models

Model	Accuracy	Hyperparameters
NN	58.74	Layers: 16, 8, 8 Activation: ReLU Epochs: 300 Dropout: 0.5 Optimizer: RMSprop Batch size: 256
RF	57.87	Estimators: 200 Criterion: gini Max depth: 10 Max features: log2 Min sample split: 40
AB Tree	57.23	Estimators: 50 Learning rate: 0.5 Criterion: entropy Max depth: 2 Max features: auto Min sample split: 40

Questions

1. The task of this assignment is to train a machine learning algorithm that predicts a binary variable, insurance claim, as accurate as possible. Consequently, selecting an algorithm based on maximum accuracy is reasonable as it suits our goal of specifying correctly a highest achievable number of labels. Moreover, literature supports using the accuracy measure when applying classification to (non-trivial) predictive algorithm such as predicting insurance cases [14][16][1].

2. When comparing our final algorithm Feedforward Neural Networks (NN) to a random coin flip benchmark, we find that our model significantly outperforms the benchmark. Not only does our method produce a higher accuracy of 58.44% compared to approximately 50%, if we apply the McNemar test [9], we find a test statistic of 50.3 with a p-value 0.00 which rejects the null hypothesis of equal performance.

3. If a dataset satisfies Manifold Hypothesis, the instances lie near a manifold of lower dimensionality relative to the original dataset. Subsequently, one could apply a neural network to such dataset but it would require careful selection of the weights for the layers. If the selection is done correctly, it would be possible to mimic the low dimension of the hypothesis to find the observations instances and produce accurate predictions. This is supported by the universal approximation theorem which states that NN can approximate any Borel measurable function [6]. Thus, NN should be capable approximate the lower space dimension to achieve good predictions. Nevertheless, [11] suggest that finding the proper weight can be challenging and even when selected NN may not work suitably due to problems complexity.

4. With previous knowledge on NN and Manifold Hypothesis, we examine whether our dataset satisfies the Manifold hypothesis based on the algorithm and performance. The found accuracy, of 58.44%, for NN is relatively low which implies that the algorithm does not find the weight for the layers which would mimic the expected lower dimensionality for predictions in the dataset. This suggest that there is no Manifold Hypothesis in this dataset. In order to reach the highest level of certainty on whether the hypothesis holds, one requires a relatively higher accuracy.

Appendix

Table 2: Hyper parameters of the Random Forest used in gridsearch with corresponding options (optimal parameters in bold)

Parameter	Options
Number of estimators	100, **200**, 400, 800
Max features	auto, sqrt, **log2**
criterion	gini, **entropy**
max depth	1, **10**, 20, 40
minimum sample split	1, 5, 20, **40**

Table 3: Hyper parameters of the AdaBoosted tree used in gridsearch with corresponding options (optimal parameters in bold)

Parameter	Options
Number of estimators	10, **50**, 100, 200
Learning rate	**0.5**, 1
Max features	**auto**, sqrt, log2
criterion	gini, **entropy**
max depth	**2**, 5, 10, 40
minimum sample split	2, 10, **40**

Table 4: Hyperparameter settings for Gridsearch 1

Gridsearch 1	Hyperparameters							
	batch size	epochs	layers	neurons	initializer	activation	optimizer	dropout
Options	32	100	1	4	uniform	relu	adam	0.5
	64			8				
	128			16				
	256			32				
				64				
				128				
				256				

Table 5: Hyperparameter settings for Gridsearch 2

Gridsearch 2	Hyperparameters							
	batch size	epochs	layers	neurons	initializer	activation	optimizer	dropout
Options	256	100	1	4	uniform	relu	adam	0.5
			2	8				
			3	16				

Table 6: Hyperparameter settings for Gridsearch 3

Gridsearch 3	Hyperparameters							
	batch size	epochs	layers	neurons	initializer	activation	optimizer	dropout
Options	256	Early Stop	3	(16, 8, 8)	Uniform	ReLU	Adam	0.2
					Normal	Sigmoid	RMSprop	0.5
					Zero	Softplus	SGD	0.8
						Tanh		

Table 7: Top performing networks per Gridsearch

Models	Hyperparameters								Accuracy
	batch size	epochs	layers	neurons	initializer	activation	optimizer	dropout	Test set
Grid1-1	256	Early Stop	1	(4)	Uniform	ReLU	Adam	0.5	0.5864
Grid1-2	64	Early Stop	1	(16)	Uniform	ReLU	Adam	0.5	0.5852
Grid1-3	256	Early Stop	1	(8)	Uniform	ReLU	Adam	0.5	0.5850
Grid1-5	256	Early Stop	1	(16)	Uniform	ReLU	Adam	0.5	0.5849
Grid1-5	256	Early Stop	1	(32)	Uniform	ReLU	Adam	0.5	0.5846
Grid2-1	256	Early Stop	3	(16, 8, 8)	Uniform	ReLU	Adam	0.5	0.5894
Grid2-2	256	Early Stop	3	(16, 16, 16)	Uniform	ReLU	Adam	0.5	0.5867
Grid2-3	256	Early Stop	2	(16, 8)	Uniform	ReLU	Adam	0.5	0.5867
Grid2-4	256	Early Stop	2	(16, 4)	Uniform	ReLU	Adam	0.5	0.5857
Grid2-5	256	Early Stop	2	(16, 16)	Uniform	ReLU	Adam	0.5	0.5853
Grid3-1	256	Early Stop	3	(16, 8, 8)	Normal	ReLU	RMSprop	0.5	0.5874
Grid3-2	256	Early Stop	3	(16, 8, 8)	Normal	ReLU	Adam	0.5	0.5868
Grid3-3	256	Early Stop	3	(16, 8, 8)	Uniform	Tanh	Adam	0.5	0.5855
Grid3-4	256	Early Stop	3	(16, 8, 8)	Normal	Tanh	Adam	0.5	0.5854
Grid3-5	256	Early Stop	3	(16, 8, 8)	Uniform	ReLU	RMSprop	0.5	0.5853

References

[1] P. L. Brockett, L. L. Golden, J. Jang, and C. Yang. A comparison of neural network, statistical methods, and variable choice for life insurers' financial distress prediction. *Journal of Risk and Insurance*, 73(3):397–419, 2006.

[2] S. Chaudhury. Tuning of adaboost with computational complexity, Aug 2020.

[3] M. K. G. Zaccone. Deep learning with tensorflow : explore neural networks and build intelligent systems with python, 2018.

[4] J. Heaton. Artificial intelligence for humans : Deep learning and neural networks, 2017.

[5] K. Hornik. Approximation capabilities of multilayer feedforward networks, 1991.

[6] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989.

[7] S. D. K.G. Sheela. Review on methods to fix number of hidden neurons in neural networks, May 2013.

[8] W. Koehrsen. Hyperparameter tuning the random forest in python, Jan 2018.

[9] P. A. Lachenbruch. *McNemar Test*. American Cancer Society, 2014.

[10] j. michael. 1st place with representation learning, 2017.

[11] C. Olah. Neural networks, manifolds, and topology.

[12] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[13] S. Raschka. Model evaluation, model selection, and algorithm selection in machine learning, 2020.

[14] G. S., P. S., S. P., and B. A. Health insurance claim prediction using artificial neural networks. *International Journal of System Dynamics Applications*, 9:40–57, 2020.

[15] P. Seguro. Porto seguro's safe driver prediction, Nov 2017.

[16] C. Yan, M. Li, W. Liu, and M. Qi. Improved adaptive genetic algorithm for the vehicle insurance fraud identification model based on a bp neural network. *Theoretical Computer Science*, 817:12–23, 2020.