Final ISE 364

Winton Wong

# Introduction

The objective is to create a model with data from data.csv and be used to predict target value with new.csv.

# Summary

The data from data.csv was first explored and then used to create a neural network, which was then used to predict the target (column 11) with data from new.csv. Several neural network design, with different activation functions, and other techniques were explored. The final model is a 3-layer model, 1 hidden layer of 25 nodes, using sigmoid as activation functions.

# Experimental Details

* Load the data.csv as df
* Conduct Exploratory Data Analysis for project\_data.csv
* Use OneHotEncoder to transform categorical columns into multiple numeric columns
* Apply Train Test Split to df
* Train Test sets are scaled
* Construct the neural network as the base model
* Train the neural network, and test it with test data
* Evaluate the model with accuracy score
* Retrain and valuate the model with different variations
  + Variations includes, using sgd as the optimizer instead of adam, reduce data dimensions from 2-50 using Principal Component Analysis (PCA), use standard scaler instead of MinMax scaler, use different activation functions like tanh, sigmoid, softmax, in addition to relu.
* Load the new.csv as df\_new
* Use OneHotEncoder to transform categorical columns into multiple numeric columns
* Scale and transform the data df\_new for prediction
* Predict and save the target column (11) to predictions.csv

# Base Model

model = Sequential()

model.add(Dense(*units*=50, *activation*='relu'))

model.add(Dropout(0.5))

model.add(Dense(*units*=25, *activation*='relu'))

model.add(Dropout(0.5))

model.add(Dense(*units*=1, *activation*='sigmoid'))

# loss for binary

model.compile(*loss*='binary\_crossentropy',

*optimizer*='adam', *metrics*=['accuracy'],)

# fit

model.fit(*x*=X\_train, *y*=y\_train, *epochs*=1000, *validation\_data*=(

X\_test, y\_test), *verbose*=0, *callbacks*=early\_stop)

Base model consists of 1 input layer with nodes equal to the total feature, 50, one hidden layer with nodes half of the input layer, and one node for the out put layer. The first and second layer has a Dropout of 0.5 and uses relu activation function, and the output layer uses sigmoid as the activation function. The model is also being validated with the test data. Callbacks are set with a patience of 50 base on loss, epochs at 1000, and accuracy are displayed.

# Results and discussions

The following chart is the accuracy of the base model and model with different variations.

|  |  |  |
| --- | --- | --- |
| Model | Accuracy Score | Model compared to base model |
| Base | 0.8267 |  |
| Base SGD | 0.8247 | Use sgd optimizer instead of adam |
| PCA 50 SGD | 0.8260 | Applied PCA with n=50 to data, trained with sgd optimizer instead of adam |
| Base Underbalance | 0.7427 | Resample data for underbalanced case with imblearn.under\_sampling |
| Base Overbalance | 0.7890 | Resample data for overbalanced case with from imblearn.over\_sampling |
| Standard Scaler | 0.8263 | Used standard scaler instead of MinMax scaler |
| Complex 1 | 0.8253 | Introduce a hidden layer of 11 nodes and 0.5 Dropout after 2nd layer |
| Complex 2 | 0.8293 | 2nd layer is 11 nodes instead of 25 |
| Tanh | 0.8267 | Used tanh as activation function for the first 2 layer |
| sigmoid | 0.8297 | Used sigmoid as activation function for the first 2 layer |
| complex 1 sigmoid | 0.8263 | Introduce a hidden layer of 11 nodes and 0.5 Dropout after 2nd layer, and used sigmoid as activation function for the first 2 layer |
| softmax | 0.8190 | Used softmax as activation function for the first 2 layer |
| PCA n=1 | 0.7560 | Applied PCA with n=1 to data |
| PCA n=2 | 0.7630 | Applied PCA with n=2 to data |
| PCA n=3 | 0.7793 | Applied PCA with n=3 to data |
| PCA n=4 | 0.7560 | Applied PCA with n=4 to data |
| PCA n=5 | 0.7720 | Applied PCA with n=5 to data |
| PCA n=6 | 0.7750 | Applied PCA with n=6 to data |
| PCA n=7 | 0.7860 | Applied PCA with n=7 to data |
| PCA n=8 | 0.7957 | Applied PCA with n=8 to data |
| PCA n=9 | 0.8020 | Applied PCA with n=9 to data |
| PCA n=10 | 0.8037 | Applied PCA with n=10 to data |
| PCA n=11 | 0.8057 | Applied PCA with n=11 to data |
| PCA n=12 | 0.8023 | Applied PCA with n=12 to data |
| PCA n=13 | 0.7910 | Applied PCA with n=13 to data |
| PCA n=14 | 0.8067 | Applied PCA with n=14 to data |
| PCA n=15 | 0.8137 | Applied PCA with n=15 to data |
| PCA n=16 | 0.8177 | Applied PCA with n=16 to data |
| PCA n=17 | 0.8100 | Applied PCA with n=17 to data |
| PCA n=18 | 0.8113 | Applied PCA with n=18 to data |
| PCA n=19 | 0.8073 | Applied PCA with n=19 to data |
| PCA n=20 | 0.8167 | Applied PCA with n=20 to data |
| PCA n=21 | 0.8120 | Applied PCA with n=21 to data |
| PCA n=22 | 0.8123 | Applied PCA with n=22 to data |
| PCA n=23 | 0.8213 | Applied PCA with n=23 to data |
| PCA n=24 | 0.8103 | Applied PCA with n=24 to data |
| PCA n=25 | 0.8123 | Applied PCA with n=25 to data |
| PCA n=26 | 0.8157 | Applied PCA with n=26 to data |
| PCA n=27 | 0.8240 | Applied PCA with n=27 to data |
| PCA n=28 | 0.8177 | Applied PCA with n=28 to data |
| PCA n=29 | 0.8253 | Applied PCA with n=29 to data |
| PCA n=30 | 0.8270 | Applied PCA with n=30 to data |
| PCA n=31 | 0.8263 | Applied PCA with n=31 to data |
| PCA n=32 | 0.8187 | Applied PCA with n=32 to data |
| PCA n=33 | 0.8213 | Applied PCA with n=33 to data |
| PCA n=34 | 0.8260 | Applied PCA with n=34 to data |
| PCA n=35 | 0.8240 | Applied PCA with n=35 to data |
| PCA n=36 | 0.8177 | Applied PCA with n=36 to data |
| PCA n=37 | 0.8200 | Applied PCA with n=37 to data |
| PCA n=38 | 0.8197 | Applied PCA with n=38 to data |
| PCA n=39 | 0.8187 | Applied PCA with n=39 to data |
| PCA n=40 | 0.8230 | Applied PCA with n=40 to data |
| PCA n=41 | 0.8253 | Applied PCA with n=41 to data |
| PCA n=42 | 0.8270 | Applied PCA with n=42 to data |
| PCA n=43 | 0.8230 | Applied PCA with n=43 to data |
| PCA n=44 | 0.8263 | Applied PCA with n=44 to data |
| PCA n=45 | 0.8263 | Applied PCA with n=45 to data |
| PCA n=46 | 0.8193 | Applied PCA with n=46 to data |
| PCA n=47 | 0.8213 | Applied PCA with n=47 to data |
| PCA n=48 | 0.8210 | Applied PCA with n=48 to data |
| PCA n=49 | 0.8280 | Applied PCA with n=49 to data |
| PCA n=50 | 0.8279 | Applied PCA with n=50 to data | |

Changing the dimensions of the input data down did not improve the accuracy, and the graphs (can be found inside the folder PCA\_1000) for each of the PCA showed that as n increases, the validation losses also increase. Each of the PCA n from 1 to 50 are ran to 1000 epoch with no call backs.

Using sigmoid as activation function for the first 2 layers provided the greatest accuracy with consistency. Although it is only a 0.07 increase in accuracy compared to the base model, unlike the more complex model, it consistently results in a 0.07 increase. Therefore, a model using sigmoid as activation function for the first 2 layers was used to train the data.

Here is a loss, accuracy, validation loss, validation accuracy graph of the base model.

A picture containing graphical user interface

Description automatically generated

Here is a loss, accuracy, validation loss, validation accuracy graph of the model using sigmoid as activation function for the first 2 layers:

Graphical user interface

Description automatically generated

While both models are set to stop with call backs patients of 50, the sigmoid model provided better overall consistency and a narrower gap between loss and validation loss.

# Body

After the data was loaded into data frames, Exploratory Data Analysis was conducted to attempt to discover trends, to check for nulls, data types, any special relationship between characteristics and targets, etc. The data has 11 different type of characteristics 0-10, and the target column 11 was binary of 1s and 0s. Some characteristics of the data are categorical data, OneHotEncoder was used to transform them, and each unique value in the categorical data got transformed into their own column. As the result, the input data can be categorized to have 50 characteristics. Although it can be reduced by removing 5 columns that from each of the 4 categorical columns, it was left alone.

The data set was then split with 30% for testing after training the model, and then scaled using the MinMax scaler.

Following the data clean up and setting up for training. A base model was set and trained and evaluated for accuracy. A PCA model was constructed and trained after reducing data to 2 dimensions and evaluated for accuracy. Another base model was constructed and trained using sgd as the optimizer. The accuracies for the 3 models that can be found inside the main\_eda.ipynb are 0.825 , 0.8, and 0.827.

Next, the goal was to find a model that would be more accurate. The data was then reduced dimensions using PCA to n=2 to n50, total of 49 models. However, as the data for the accuracies for the 49 models did not produce a significantly better results than the base model.

During the exploratory data analysis stage, it was discovered that the ratio of 0s and 1s in the target columns was about 7:0. The set of data was then resampled using oversampling and under sampling techniques, but none of them produced better results.

Two models with additional layers were then tried, they would sometimes produce a better result, but they lack the consistency due to the more complex network.

Lastly, the base model was then reconfigured using different activation functions in the first 2 layers, it includes tanh, sigmoid, and softmax. The results of those were 0.8268, 0.8927, and 0.8190. And the sigmoid has the highest accuracy so far and it was very reproducible due to its simple network. Then predictions were made using the model that uses sigmoid as activation functions.

# Conclusions

After comparing the base model and a few variations models, a 3 layers model (50-25-1) using sigmoid as activation functions was then used to predict data from new.csv, and the accuracy is about 0.83.

# Recommendations

It would be nice if there is a description of the different features, then each feature can be deeply analyzed and to be used to the advantage. Such as the categorical data, if the relationship between the kinds within a characteristic can be known, different data transformation techniques can the be apply in attempt to create a more accurate model.

# Files

main\_eda.ipynb

* This files contains the exploratory analysis and explorations of training a few models

predict.ipynb

* This file trains the base model with sigmoid as activation functions and predict and save to the prediction.csv

**PCA\_1000**/

* This is a folder that contains the script that runs PCA from n=2 to n=50, and the results of each run in a consolidated csv files, and each run’s graph

base\_model.py

* This is a script to run the base model

base\_sgd.py

* This is a script to run the base model with sgd optimizer

pca\_n50\_sdg.py

* This is a script to run the base model with PCA set n=50

base\_sigmoid.py

* This is a script to run the base model with sigmoid as activation functions

base\_tanh.py

* This is a script to run the base model with tanh as activation functions

base\_softmax.py

* This is a script to run the base model with softmax as activation functions

base\_under.py

* This is a script to run the base model with under-balancing

base\_over.py

* This is a script to run the base model with overbalancing

standard\_scaler.py

* This is a script to run the base model with the standard scaler

complex\_1\_model.py

* This is a script to run the complex 1 model

complex\_1\_sigmoid.py

* This is a script to run the complex 1 model with sigmoid as activation functions

complex\_2.py

* This is a script to run the complex 2 model

**result\_graph**/

* This folder contains the loss, accuracy, validation loss, validation accuracy graphs for each result