Report on performance of the deep learning models

Analysis Overview:

First Model

In this analysis, the primary objective is to construct a robust deep learning model for predicting the success of funding applications within Alphabet Soup Charity. The dataset, extracted from "charity_data.csv" revolves around the "IS_SUCCESSFUL" variable, serving as the target to delineate funding outcomes.

Flow Steps

Data Preprocessing

Targets: "IS_SUCCESSFUL"
Features: Inclusion of all columns.
Removed : "EIN" and "NAME.
APPLICATION_TYPE :
Aggregated applications with counts < 500 into the "Other" category.
CLASSIFICATION :
Collated classifications with counts < 100 into the "Other" category.
Categorical Encoding:
Employed one-hot encoding through pd.get_dummies to transform categorical columns into numerical equivalents.

Compiling, Training, and Evaluating the Model

num_input_features = len(X_train_scaled[0])
hid_nodes_layer1 = 80 activation="relu"
hid_nodes_layer2 = 50 activation="relu"
hid_nodes_layer3 = 20 activation="relu"
Output layer = 1 activation="sigmoid"
epoch = 100

Results

268/268 - 0s - loss: 0.5678 - accuracy: 0.7270 - 469ms/epoch - 2ms/step
Loss: 0.5678246021270752, Accuracy: 0.7269970774650574

Summary

we have a good performance of the model with as 0.73 accuracy and Loss: 0.567

Optimizations Models

1-Optimisation parameters

Trial 1

Binning and encoding for app type < 100 and for classification < 100 ,
number of Layers 128-64-32 ', epochs=100 ,
fct activation relu and segment for output

Results

268/268 - 1s - loss: 0.5669 - accuracy: 0.7268 - 582ms/epoch - 2ms/step
Loss: 0.5668720602989197,
Accuracy: 0.7267638444900513

Summary

we still have the same accuracy as the First model

Trial 2

Binning and encoding for app type < 100 and for classification < 100

modify number of Layers 80-30-20 ', epochs=150 ,
fct activation relu and segment for output

Results
268/268 - 0s - loss: 0.5610 - accuracy: 0.7248 - 497ms/epoch - 2ms/step
Loss: 0.5609777569770813,
Accuracy: 0.724781334400177

Trial 3
'Binning and encoding for app type < 100 and for classification < 100
modify number of Layers 80-30-20 ,
modify epochs=100 ,fct activation tanh and segment for output

Results
268/268 - 1s - loss: 0.5545 - accuracy: 0.7263 - 773ms/epoch - 3ms/step
Loss: 0.554489254951477,
Accuracy: 0.7262973785400391

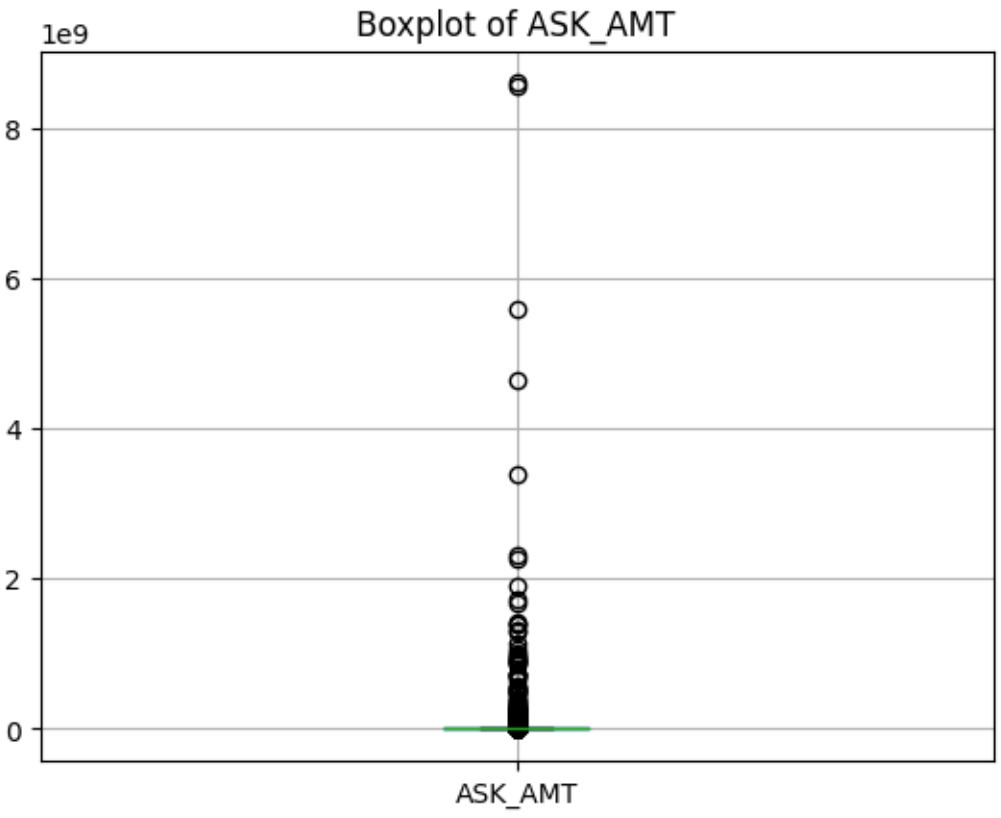
Summary of those 3 trials
when we use the same data processing algorithm and the same target and features ,
we can easily conclude that the modification for differents parametrs as number of layer ,number of epoch ,functions used ,
have a very small impact in the accuracy arround 0.73 .
we can check to modify our data processing algorithm ,the target and features and check the performance of our model .

2-Optimisation data processing

***Process Flow**
we use the same data ,we will explore numerical features for potential outliers
EIN,IS_SUCCESSFUL,STATUS, (ASK_AMT they are some outlier)

AlphabetSoupCharity_Optimization\AlphabetSoupCharity_Optimization_data\img\
boxplot3.png

*** Data



processing

```
#Drop outliers in 'ASK_AMT' column

#keep  just 'APPLICATION_TYPE', 'CLASSIFICATION', and 'USE_CASE'
columns_to_keep = ['APPLICATION_TYPE', 'CLASSIFICATION', 'USE_CASE',
'IS_SUCCESSFUL']
application_df = application_df[columns_to_keep]

#Convert categorical data to numeric with pd.get_dummies
#Split the data
X = application_df_encoded.drop('IS_SUCCESSFUL', axis=1)
y = application_df_encoded['IS_SUCCESSFUL']

#Split,create,fit ,scale the data

#define the model
3 layers (100,50,20),activation fct Relu and sigmoid for Output

# Compile the model
nn.compile(loss="binary_crossentropy", optimizer="adam", metrics=["accuracy"])

# Train the model with 100  epochs
```

Results

```
204/204 - 0s - loss: 0.6399 - accuracy: 0.6452 - 477ms/epoch - 2ms/step
Loss: 0.6399290561676025,
Accuracy: 0.6451563239097595
```

Summary global

after this trail we can conclude that the data processing can impact the performance of the model ,
in our case we have decrease in the performance of the model we increase number of loss 0.639 and we decrease the Accuracy: 0.6451 instead of 0.726