The report should contain the following:

1. **Overview** – The purpose of this analysis is to create an accurate model from a binary classifier which will predict whether or not a recipient of an Alphabet Soup donation will be successful or not and explain the factors leading to the models accuracy across three attempts .
2. **Results**: Using bulleted lists and images to support your answers, address the following questions:

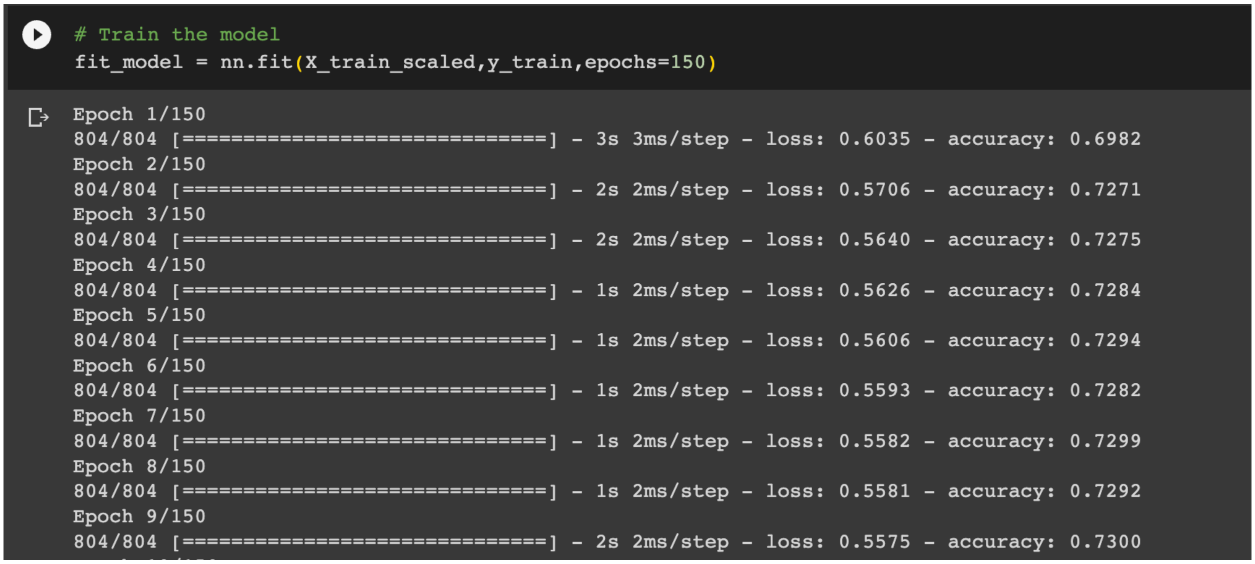
- Data Preprocessing
    - What variable(s) are the target(s) for your model?
        - The 'Is_Successful' column is the target variable
    - What variable(s) are the features for your model?
        - The features are the remaining columns: Name, application_type, affiliation, classification, use_case, organization, status, income_amt, special_consideration, and ask_amt
    - What variable(s) should be removed from the input data because they are neither targets nor features?
        - I achieved peak performance by excluding only the EIN column
- Compiling, Training, and Evaluating the Model
    - How many neurons, layers, and activation functions did you select for your neural network model, and why?
        - For my working model, I used 3 hidden layers with 15, 30, and 70 neurons, respectively. After trying tanh and sigmoid, I used the relu activation method for the hidden layers and sigmoid for the output layer
    - Were you able to achieve the target model performance?
        - No
    - What steps did you take in your attempts to increase model performance?

- Attempt 1: For attempt 1, I used 8 nodes in hidden layer 1 and 5 in hidden layer 2 with 100 epochs of iteration. Both hidden laters were using the relu activation method while the output layer used the sigmoid method. From that arrangement, I only achieved a 72.58% accuracy with 55.3% loss. With only 72.58% of the models predictions matching the correct data, I will need to modify other parameters for more optimal performance.

- Attempt 2: For attempt 2, I increased the amount of neurons with 15 in later 1 and 20 in layer 2. I also changed the activation method to sigmoid for all three layers. Sigmoid is beneficial when trying to predict the probability as an output. This attempt yielded worse performance at only 72.57% accuracy and 55.2% model loss.

- · 268/268 - 0s - loss: 0.5547 - accuracy: 0.7247 - 430ms/epoch - 2ms/step
- · Loss: 0.5546526312828064, Accuracy: 0.7246647477149963

- Attempt 3: As I notice the first three epochs present the biggest increase in accuracy, attempt 3 starts 3 hidden layers using the relu activation method ended with a sigmoid activation layer for output. Adding back the 'name' feature produced too many parameters which would clog up my run time, so I removed the 'status' feature and increased to 150 epochs instead. I also increased the cutoff for the binning of the classification and application features and increased the neurons to 10, 20 and 40, respectively for the three hidden layers. See below how the first few iterations started higher than 70% but then quickly plateaued.



```
# Train the model
fit_model = nn.fit(X_train_scaled,y_train,epochs=150)

Epoch 1/150
804/804 [==============================] - 3s 3ms/step - loss: 0.6035 - accuracy: 0.6982
Epoch 2/150
804/804 [==============================] - 2s 2ms/step - loss: 0.5706 - accuracy: 0.7271
Epoch 3/150
804/804 [==============================] - 2s 2ms/step - loss: 0.5640 - accuracy: 0.7275
Epoch 4/150
804/804 [==============================] - 1s 2ms/step - loss: 0.5626 - accuracy: 0.7284
Epoch 5/150
804/804 [==============================] - 1s 2ms/step - loss: 0.5606 - accuracy: 0.7294
Epoch 6/150
804/804 [==============================] - 1s 2ms/step - loss: 0.5593 - accuracy: 0.7282
Epoch 7/150
804/804 [==============================] - 1s 2ms/step - loss: 0.5582 - accuracy: 0.7299
Epoch 8/150
804/804 [==============================] - 1s 2ms/step - loss: 0.5581 - accuracy: 0.7292
Epoch 9/150
804/804 [==============================] - 2s 2ms/step - loss: 0.5575 - accuracy: 0.7300
```

-
- ·

· 268/268 - 0s - loss: 0.5624 - accuracy: 0.7250 - 473ms/epoch - 2ms/step
Loss: 0.5624184608459473, Accuracy: 0.7250145673751831

Summary: Overall the model could not exceed 72% accuracy. Another way to increase meaningful parameters might be to actually add back the ‘Name’ feature and bin it, so that there are not too many parameters. Too many parameters could lead to overfitting of the model.