Write a Report on the Neural Network Model

The report should contain the following:

1. **Overview** of the analysis: Explain the purpose of this analysis.

The nonprofit foundation Alphabet Soup wants a tool that can help it select the applicants for funding with the best chance of success in their ventures. The purpose of analysis was to predict whether applicants will be successful if funded by Alphabet Soup.

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?

Target variable refers to the 'IS_SUCCESSFUL' column in the application_df

    - What variable(s) should be removed from the input data because they are neither targets nor features?

Both 'EIN' and 'NAME' columns were dropped because they were neither targets nor features for the dataset.

```
# Drop the non-beneficial ID columns, 'EIN' and 'NAME'.
application_df = application_df.drop(columns = ['EIN', 'NAME'])
application_df
```

- Compiling, Training, and Evaluating the Model
    - How many neurons, layers, and activation functions did you select for your neural network model, and why?

In the first attempt, I used 80 hidden_nodes_layer1 & 30 hidden_nodes_layer2, these were chosen randomly.

Compile, Train and Evaluate the Model

```
# (First Attempt)
# Define the model - deep neural net, i.e., the number of input features and hidden nodes for each layer.
input_features_total = len(X_train[0])
hidden_nodes_layer1 = 80
hidden_nodes_layer2 = 30
```

- Were you able to achieve the target model performance?

Reaching the goal of 75% model accuracy was not attainable and the first attempt was 72.7%.

```
# Evaluate the model using the test data (First Attempt)
model_loss, model_accuracy = nn.evaluate(X_test_scaled,y_test,verbose=2)
print(f"Loss: {model_loss}, Accuracy: {model_accuracy}")
```

```
268/268 - 0s - loss: 0.5762 - accuracy: 0.7276 - 341ms/epoch - 1ms/step
Loss: 0.5761507153511047, Accuracy: 0.727580189704895
```

- What steps did you take in your attempts to increase model performance?

I increased and decreased additional hidden nodes from the first attempt and switched up the activation functions in the third attempt in order to achieve higher model accuracy to no avail. Third attempt resulted in 72.8% accuracy which was still short of the desired 75%.

```
# (Second Attempt)
# Define the model - deep neural net, i.e., the number of input features and hidden nodes for each layer.
input_features_total = len(X_train[0])
hidden_nodes_layer1 = 100
hidden_nodes_layer2 = 50

nn = tf.keras.models.Sequential()

# First hidden layer
nn.add(tf.keras.layers.Dense(units=hidden_nodes_layer1, input_dim = input_features_total, activation = "relu"))

# Second hidden layer
nn.add(tf.keras.layers.Dense(units=hidden_nodes_layer2, activation = "relu"))

# Output layer
nn.add(tf.keras.layers.Dense(units=1, activation="sigmoid"))

# Check the structure of the model
nn.summary()
```

```
# (Third Attempt)
# Define the model - deep neural net, i.e., the number of input features and hidden nodes for each layer.
input_features_total = len(X_train[0])
hidden_nodes_layer1 = 20
hidden_nodes_layer2 = 20

nn = tf.keras.models.Sequential()

# First hidden layer
nn.add(tf.keras.layers.Dense(units=hidden_nodes_layer1, input_dim = input_features_total, activation = "tanh"))

# Second hidden layer
nn.add(tf.keras.layers.Dense(units=hidden_nodes_layer2, activation = "sigmoid"))

# Output layer
nn.add(tf.keras.layers.Dense(units=1, activation="relu"))

# Check the structure of the model
nn.summary()
```

3. **Summary**: Summarise the overall results of the deep learning model. Include a recommendation for how a different model could solve this classification problem, and then explain your recommendation.

The deep learning model achieved an overall accuracy of approximately 73% in addressing the classification problem. Improving the correlation between input and output variables is crucial for enhancing prediction accuracy. This improvement may involve thorough upfront data cleanup and the exploration of alternative activation functions in the model. Clearly a different approach needs to be adopted to increase the overall accuracy of more than 75% possibly by doing additional data cleanup up, as well as by using a model with different activation functions and iterating until higher accuracy is reached.