Funding Success Analysis

1. **Overview**: Purpose of this analysis.

The purpose of this analysis was to determine whether funding provided by Alphabet Soup to different applicants will result in the execution of a successful venture or not. What success means is not clearly defined, but it is indicated as a boolean variable. However, defining success differently (such as achieving certain valuation for their venture) might result in a higher quality model.

Furthermore, the income is provided as a range, while the asked amount is provided as a number. My opinion is that utilizing a number for income instead of a range would result in a higher quality model.

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:
            - IS_SUCCESSFUL
    - What variable(s) are the features for your model?
        - The features are:
            - APPLICATION_TYPE
            - AFFILIATION
            - CLASSIFICATION
            - USE_CASE
            - ORGANIZATION
            - STATUS
            - INCOME_AMT
            - SPECIAL_CONSIDERATIONS
            - ASK_AMT
    - What variable(s) should be removed from the input data because they are neither targets nor features?
        - Neither targets nor features:
            - EIN
            - NAME
- Compiling, Training, and Evaluating the Model
    - How many neurons, layers, and activation functions did you select for your neural network model, and why?
        - Neurons: 129; it’s the recommended amount of 2-3 times the amount of features (43).
        - Layers: 2 hidden layers + 1 output layer; started with 2, then did 3 without seeing significant improvement.
        - Activation functions:

        - Hidden layer 1: relu; it’s better to use more complex functions for hidden layers.
        - Hidden layer 2: relu
        - Output layer: tanh; tanh was slightly better than sigmoid.
    - Were you able to achieve the target model performance?
        - No, the accuracy stayed at around 74%. My guess is that the data isn’t of enough quality.
    - What steps did you take in your attempts to increase model performance?
        - I added 2 more layers
        - I added more neurons
        - I removed 2 columns (Classification and Application Type): This negatively affected accuracy but not for a lot, which means these features don’t explain a big part of the outcome.

Old data:

	EIN	NAME	APPLICATION_TYPE	AFFILIATION	CLASSIFICATION	USE_CASE	ORGANIZATION	STATUS	INCOME_AMT	SPECIAL_CONSIDERATIONS	ASK_AMT	IS_SUCCESSFUL
0	10520599	BLUE KNIGHTS MOTORCYCLE CLUB	T10	Independent	C1000	ProductDev	Association	1	0	N	5000	1
1	10531628	AMERICAN CHESAPEAKE CLUB CHARITABLE TR	T3	Independent	C2000	Preservation	Co-operative	1	1-9999	N	108590	1
2	10547893	ST CLOUD PROFESSIONAL FIREFIGHTERS	T5	CompanySponsored	C3000	ProductDev	Association	1	0	N	5000	0
3	10553066	SOUTHSIDE ATHLETIC ASSOCIATION	T3	CompanySponsored	C2000	Preservation	Trust	1	10000-24999	N	6692	1
4	10556103	GENETIC RESEARCH INSTITUTE OF THE DESERT	T3	Independent	C1000	Heathcare	Trust	1	100000-499999	N	142590	1

Old model:

```
# Define the model - deep neural net, i.e., the number of input features and hidden nodes for each layer.
 # Define the deep learning model
nn_model = tf.keras.models.Sequential()

# First hidden layer
nn_model.add(tf.keras.layers.Dense(units=129, activation="relu", input_dim=43))

# Second hidden layer
nn_model.add(tf.keras.layers.Dense(units=129, activation="relu"))

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

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

New data:

	AFFILIATION	USE_CASE	ORGANIZATION	STATUS	INCOME_AMT	SPECIAL_CONSIDERATIONS	ASK_AMT	IS_SUCCESSFUL
0	Independent	ProductDev	Association	1	0	N	5000	1
1	Independent	Preservation	Co-operative	1	1-9999	N	108590	1
2	CompanySponsored	ProductDev	Association	1	0	N	5000	0
3	CompanySponsored	Preservation	Trust	1	10000-24999	N	6692	1
4	Independent	Heathcare	Trust	1	100000-499999	N	142590	1
...	...	...	...	...	...	...	...	...
34294	Independent	ProductDev	Association	1	0	N	5000	0
34295	CompanySponsored	ProductDev	Association	1	0	N	5000	0
34296	CompanySponsored	Preservation	Association	1	0	N	5000	0
34297	Independent	ProductDev	Association	1	0	N	5000	1
34298	Independent	Preservation	Co-operative	1	1M-5M	N	36500179	0

34299 rows × 8 columns

New model:

```
# Define the model - deep neural net, i.e., the number of input features and hidden nodes for each layer.
 # Define the deep learning model
nn_model_new = tf.keras.models.Sequential()

# First hidden layer
nn_model_new.add(tf.keras.layers.Dense(units=300, activation="relu", input_dim=28))

# Second hidden layer
nn_model_new.add(tf.keras.layers.Dense(units=100, activation="relu"))

# Third hidden layer
nn_model_new.add(tf.keras.layers.Dense(units=100, activation="relu"))

# Fourth hidden layer
nn_model_new.add(tf.keras.layers.Dense(units=100, activation="relu"))

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

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

3. **Summary**: Summarize 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 performed better with the Classification and Application Type features, although just slightly (73.71% vs 70.04% accuracy). However, these features are categories, and when encoded they result in a lot of added new features.

I believe this is a poor model because we have poor data with a lot of categorical features with a lot of possible values. My guess is that if we had numerical data instead, and had a more clear definition for success, our model would perform a lot better.

My recommendation would be to gather data in a different manner so that we can have numerical values instead of categorical values, which would lead to a model that is far more accurate.