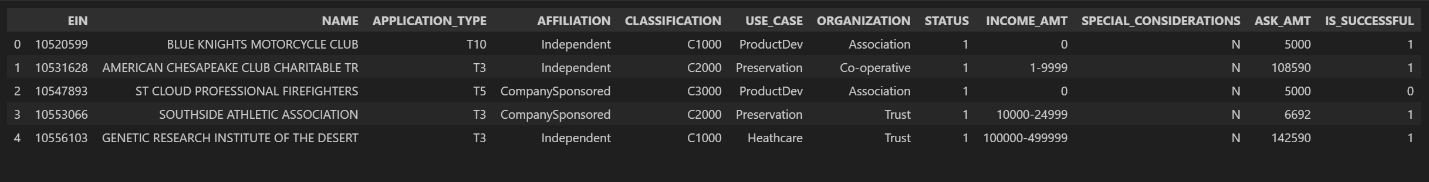
**Report on the Neural Network Model**

**Overview** **of the analysis**:

To develop and implement a neural network model utilizing advanced machine learning techniques with scikit-learn and TensorFlow. The model aims to serve as a binary classifier, predicting the likelihood of the applicants' success if funded by Alphabet Soup.

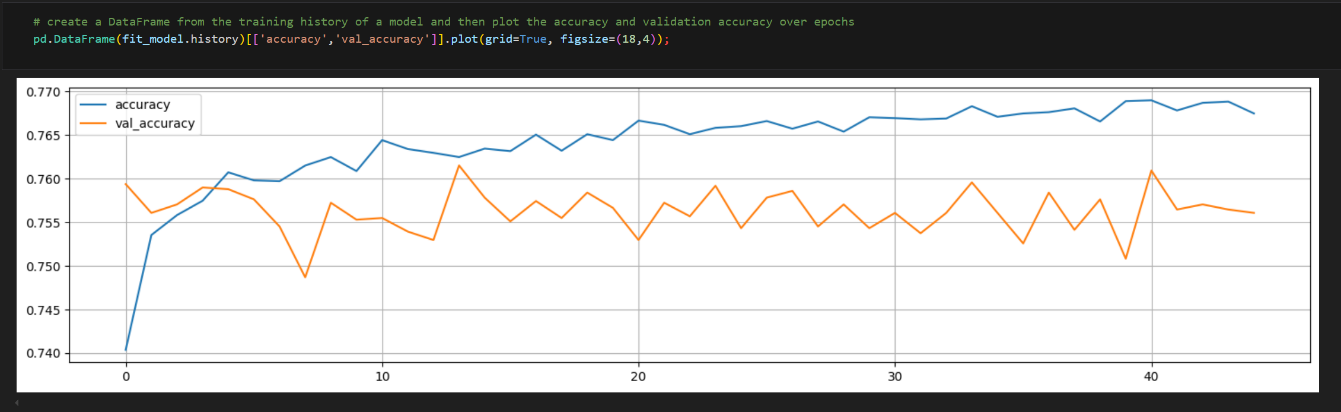
**Snapshot of the original DataFrame**:



**Results**:

* Data Preprocessing
  + What variable(s) are the target(s) for your model? The 'IS\_SUCCESSFUL' numerical column in the dataset was used as the target variable for the model to predict whether a charity campaign was successful or profitable.
  + What variable(s) are the features for your model? The rest of the columns (‘NAME’, ‘APPLICATION\_TYPE’, ‘AFFILIATION’, ‘CLASSIFICATION’, ‘USE\_CASE, ORGANIZATON’, ‘STATUS’, ‘INCOME\_AMT’, ‘SPECIAL\_CONSIDERATIONS’, and ‘ASK\_AMT’) were used as features of the model. The ‘STATUS’ and ‘ASK\_AMT’ were the only numerical columns, and the rest were categorical.



* + What variable(s) should be removed from the input data because they are neither targets nor features? The 'EIN' column was dropped because it contained unique values, making it non-beneficial to our model.
* Compiling, Training, and Evaluating the Model
  + How many neurons, layers, and activation functions did you select for your neural network model, and why? The neural network model was optimized with a total of 275 neurons: 140 in the first hidden layer and 135 in the second. The model consisted of three layers in total—two hidden layers and an output layer. ReLU and sigmoid activation functions were selected for their superior performance with this dataset. The choice of neurons and layers was based on the observed improvement in prediction accuracy, enabling the model to capture more complex patterns effectively.
  + Were you able to achieve the target model performance? Yes, the model’s accuracy was 75.46%
  + What steps did you take in your attempts to increase model performance? In the original starter code (AlphabetSoupCharity.ipynb), I dropped the 'EIN' and 'NAME' columns as suggested and followed most of the recommendations. For binning, I grouped values below 600 in the 'APPLICATION\_TYPE' column and those below 780 in the 'CLASSIFICATION' column. After splitting the preprocessed data, I performed fitting and scaling. For model compilation and training, I used three dense layers with 170 neurons in the first hidden layer and 140 neurons in the second. ReLU and sigmoid activation functions were selected, and the model was trained over 45 epochs. However, this configuration resulted in an accuracy of only 73.09%. In the optimization phase (AlphabetSoupCharity\_Optimization notebook), I reintroduced the 'NAME' column, which surprisingly improved the model's performance. Since the 'NAME' column contained 19,568 unique values, I grouped values with fewer than 100 occurrences. Additionally, I adjusted the binning thresholds for 'APPLICATION\_TYPE' to 500 and for 'CLASSIFICATION' to 200. I also applied binning to the 'AFFILIATION' column (grouping values below 100), but binning the 'USE\_CASE' and 'ORGANIZATION' columns did not improve performance, so they were removed. The number of neurons was revised to 140 in the first hidden layer and 135 in the second, while keeping the number of epochs the same. These optimizations increased the model's accuracy to 75.46%. The graph below illustrates the model's training history by plotting accuracy and validation accuracy across the number of epochs. Beyond 45 epochs, the gap between accuracy and validation accuracy widens, indicating diminishing returns from additional epochs and suggesting potential overfitting.

**Summary**:

While the deep learning model achieved an acceptable accuracy, it faced challenges along the way. Removing certain columns caused the model to exhibit signs of overfitting. To improve its performance and achieve higher accuracy, further adjustments may be necessary to enhance the model's reliability. Exploring a different type of deep learning model could also be more effective for this classification problem. For instance, the Random Forest classifier could be a strong alternative, as it calculates feature importance, helping to identify the most relevant features for better predictions. Additionally, Random Forest is well-suited for handling non-linear relationships and is less prone to overfitting.