```
# NAME: Mihail Chitorog
# ASGT: Activity 5
# ORGN: CSUB - CMPS 3500
# FILE: Neural_Network.pdf
# DATE: 11/16/2024
```

`Cleaning the Credit Score Data Set`

- What is the size of the input dataset (`credit_score_data.csv`)?

The size of the input dataset is 80,000 records of customers with 28 columns.

- What is the size of the clean data set (`Credit_score_cleaned_data.csv`)?

The size of the clean data set is 80,000 records of customers with 32 columns.

- What are the 3 most common data issues in the columns of the input dataset?

1. **Many columns have missing values shown as null.**
2. **Some columns have data types that don't match the kind of data they hold. For example, a column that should have numeric values might contain textual characters, most likely because of placeholders or incorrect entries.**
3. **Some missing data isn't marked as null but is instead stored as strings.**

- In your own words describe the purpose of function `summarize_numerical_column_with_deviation`.

The purpose of this function is to summarize and analyze data in a specific column of a dataset in two steps. The first step gives a summary of the column, like the smallest and largest values, average and how the data is spread out. It also shows charts like a boxplot and histogram to visualize the data. And the second step, it can compare the data for each customer and find out how much their numbers are different from the usual value for that customer, called median standardization. This tells us which customers are far from the average behavior.

- In your own words describe the purpose of function `return_max_MAD`.

The purpose of this function is to calculate the maximum "Median Absolute Deviation" (MAD) for a column of data. It looks at the data for each customer, calculates how much it deviates from the usual, and then finds the largest MAD among all customers.

- Describe the steps taken to clean columns `Annual Income` and `monthly inhand salary`.

The steps taken to clean Annual Income are:

1. **The data was inspected for any outliers or extreme values. A boxplot and histogram showed some high values that didn't seem realistic.**

2. **Extreme values were probably entered incorrectly during data entry, as they are much higher than the usual income levels.**
3. **The income was grouped by Customer ID and missing or incorrect values were replaced using statistical methods like the mode (most common value) or neighboring values within the same group.**
4. **Remaining extreme outliers were removed or replaced based on their deviation from the median value of the customer's records.**
5. **After these corrections the distribution of Annual Income looked more reasonable and clean.**

The steps taken to clean Monthly Inhand Salary are:

1. **The data was also checked for missing values and unusual patterns. This column had missing values and inconsistent records compared to Annual Income.**
2. **Since Monthly Inhand Salary is related to Annual Income (Annual Income = Monthly Salary × 12), missing values in Monthly Inhand Salary were filled in.**
3. **Missing values were replaced by looking at nearby or same-group records with valid Monthly Salary for the same customer.**
4. **Any remaining unrealistic values were adjusted using methods like deviation from the median value within the customer group.**
5. **A clean and realistic distribution was achieved for Monthly Inhand Salary.**

Running a Simple Neural Network

- RELU (or rectified linear unit) is one type of activation function that is used above. What is the point of having an activation function?

The point of having an activation function like RELU in a neural network is to let non-linearity into the model. Real-world data has complex, non-linear relationships and an activation function allows the neural network to learn and approximate these non-linear patterns.

- How many neurons are in the input layer and what do these neurons correspond to? What about the output layer?

The input layer has 24 neurons and they correspond to the preprocessed features selected for modeling: Age, Annual_Income, Monthly_Inhand_Salary, Num_Credit_Card, Interest_Rate, Credit_Utilization_Ratio, One-hot encoded categorical features like Occupation and Credit_Mix.

The output layer has 3 neurons, corresponding to the three target classes: Good, Poor, and Standard credit scores.

- Explain in your own words the process of back-propagation and how a neural net “learns".

Back-propagation is the process of updating the weights in a neural network based on the error in predictions. First the data flows through the network from the input layer to the output layer, generating predictions. Then the difference between the predicted values and the actual values is calculated using a loss function. After that the gradient of the loss with respect to each weight is computed using the chain rule. These gradients are propagated backward from the output layer to the input layer and using an optimizer, weights are updated in the direction that minimizes the loss. The learning rate controls the step size for updates. Eventually this process is repeated for multiple epochs until the network converges to a solution, which is usually until the sum of squared errors is less than 0.001.

- Is this model a good fit for the data? Why or why not?

I think the model is not a perfect fit as it may require better preprocessing, feature engineering or other improvements. It is because it achieves a test accuracy of 62.3%, which is quite low for a classification task. It also struggles with certain misclassifications, like Poor is often misclassified as Standard or Good, Standard is sometimes misclassified as Good or Poor. Other indicators like Precision, Recall, F1-Score are around 60%, which means more improvement is needed. The training and test loss of around 0.81 shows that the model has not fully learned to classify the data accurately yet.

- What changes would you make to the `Basic_Neural_Networks Notebook` to ensure that your group will build a better Neural Network.

I would normalize numerical features for better performance, then check for unnecessary features or high connection for example between Annual_Income and Monthly_Inhand_Salary. I would also try to experiment with different numbers of hidden layers and neurons, as well as use random search to optimize learning rate, batch size, and number of epochs. It would be wise to set up a way to stop training when the model stops getting better. We could test it multiple times with different data to make sure our model isn't just getting lucky.