1. STATISTICAL DESCRIPTIONS

`german_cat.shape` => ***There are 1000 records with 20 variables for them***

`german_cat.describe()` => ***High Standard Deviation can be seen in Credit Amount, Duration and Ages***

	DURATION	CREDIT_AMT	INSTALLMENT_RATE	PRESENT_RESIDENCE_SINCE	AGE	NO._EXISTING_CREDITS	NO_LIABLE	RESPONSE
count	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000
mean	20.903000	3271.258000	2.973000	2.845000	35.546000	1.407000	1.155000	1.300000
std	12.058814	2822.736876	1.118715	1.103718	11.375469	0.577654	0.362086	0.458487
min	4.000000	250.000000	1.000000	1.000000	19.000000	1.000000	1.000000	1.000000
25%	12.000000	1365.500000	2.000000	2.000000	27.000000	1.000000	1.000000	1.000000
50%	18.000000	2319.500000	3.000000	3.000000	33.000000	1.000000	1.000000	1.000000
75%	24.000000	3972.250000	4.000000	4.000000	42.000000	2.000000	1.000000	2.000000
max	72.000000	18424.000000	4.000000	4.000000	75.000000	4.000000	2.000000	2.000000

`german_cat.nunique()=>`

```
ACCT_STATUS                      4
DURATION                        33
CREDIT_HISTORY                   5
PURPOSE                         10
CREDIT_AMT                     921
SAVE_ACCTS/BONDS                 5
EMPLOYMENT_SINCE                 5
INSTALLMENT_RATE                 4
GENDER AND PERSONAL STATE        4
GURANTORS                        3
PRESENT_RESIDENCE_SINCE          4
PROPERTY                         4
AGE                             53
INSTALLMENT PLANS                3
HOUSING                          3
NO._EXISTING_CREDITS             4
JOB                              4
NO_LIABLE                        2
TELEPHONE                        2
FOREIGN_WORKER                   2
RESPONSE                         2
dtype: int64
```

`german_cat.isnull().sum()=>` ***No Missing Values in the Dataset***

```
ACCT_STATUS                          0
DURATION                             0
CREDIT_HISTORY                       0
PURPOSE                              0
CREDIT_AMT                           0
SAVE_ACCTS/BONDS                     0
EMPLOYMENT_SINCE                     0
INSTALLMENT_RATE                     0
GENDER AND PERSONAL STATE            0
GURANTORS                            0
PRESENT_RESIDENCE_SINCE              0
PROPERTY                             0
AGE                                  0
INSTALLMENT PLANS                    0
HOUSING                              0
NO._EXISTING_CREDITS                 0
JOB                                  0
NO_LIABLE                            0
TELEPHONE                            0
FOREIGN_WORKER                       0
RESPONSE                             0
dtype: int64
```

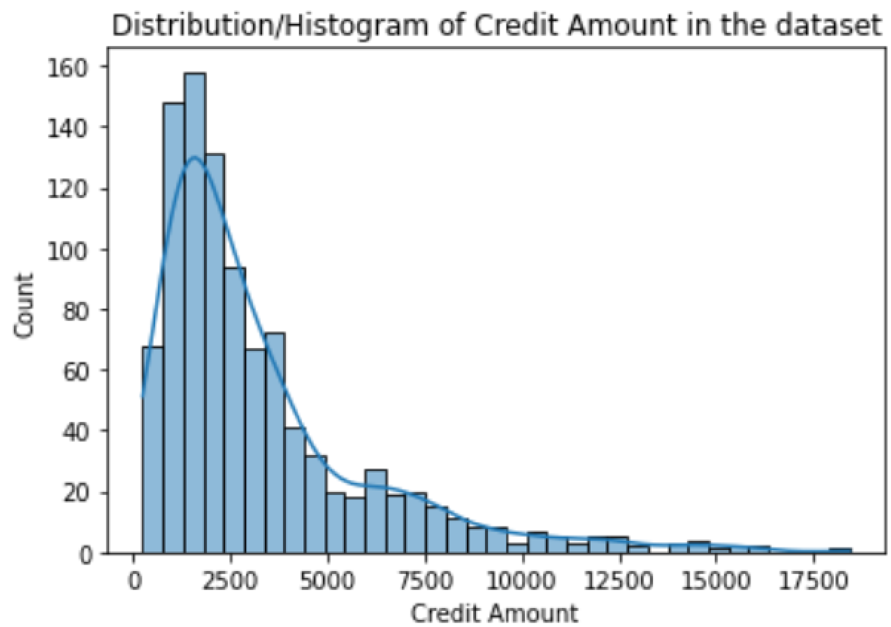
1.1 Correcting formats

```
cat_columns=[columns[0],columns[2],columns[5],columns[6],columns[8],
columns[9],columns[11],columns[13],columns[14],columns[16],columns[18],col
umns[19]]
for column in cat_columns:
  german_cat[column+'0']=german_cat[column].str[-1].astype(int)
PURPOSE0_COL=german_cat['PURPOSE0'].apply(lambda x: x%40 if x!=10
else x)
```

All the preceding A's and the number are removed for the categorical columns so that their format can be easily understood by the models

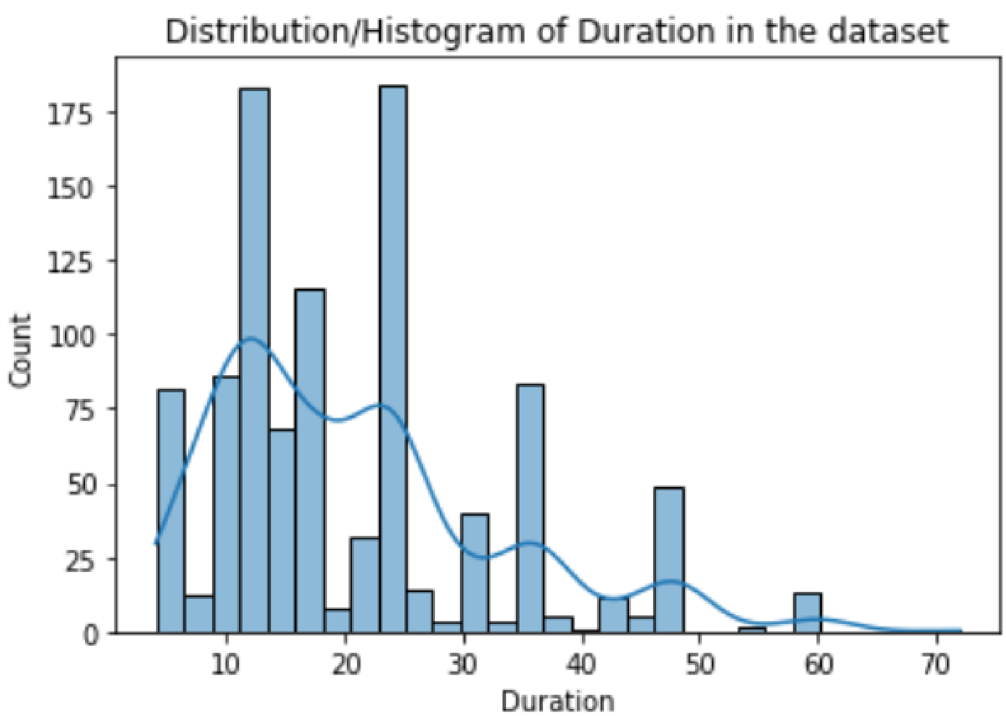
1.2 Distributions of Important Variables

```
sns.histplot(german_cat['CREDIT_AMT'],kde=True)
plot_labels("Credit Amount","Count","Distribution/Histogram of
Credit Amount in the dataset")
```



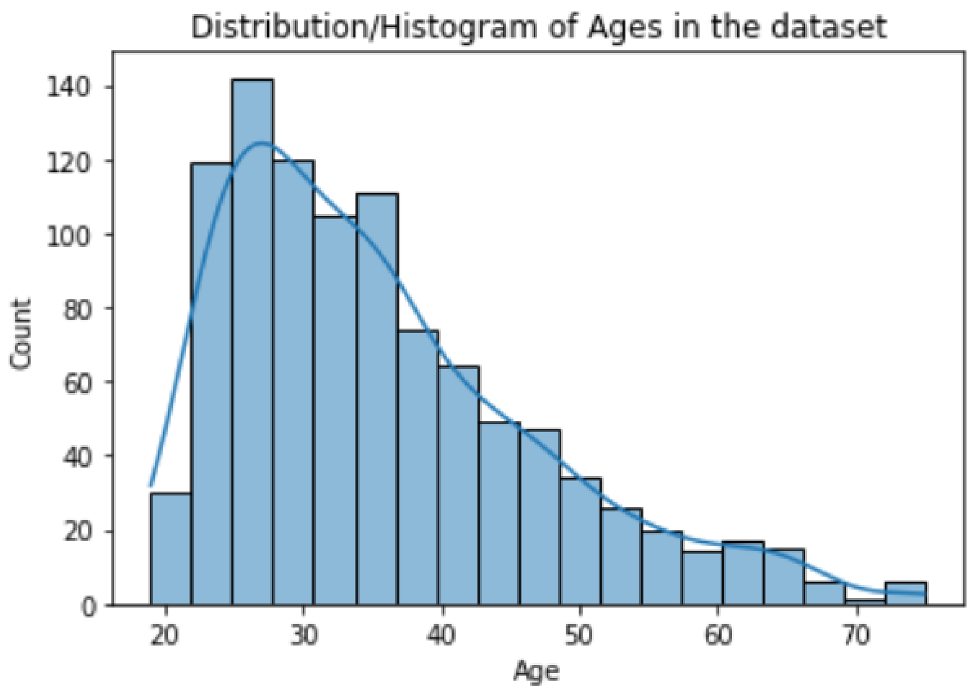
Highly skewed distribution and right-tailed distribution

```
sns.histplot(german_cat['DURATION'],kde=True)
plot_labels("Duration","Count","Distribution/Histogram of Duration
in the dataset")
```



Moderately skewed distribution

```
sns.histplot(german_cat['AGE'],kde=True)
plot_labels("Age","Count","Distribution/Histogram of Ages in the dataset")
```



Highly skewed distribution and right-tailed distribution

1.3 Feature Engineering

```
age_german_cat['AGE0']=pd.cut(age_german_cat['AGE'],bins=age_bins,labels=age_labels)
age_german_cat['AGE1']=pd.cut(age_german_cat['AGE'],bins=age_bins,labels=age_labels2)
```

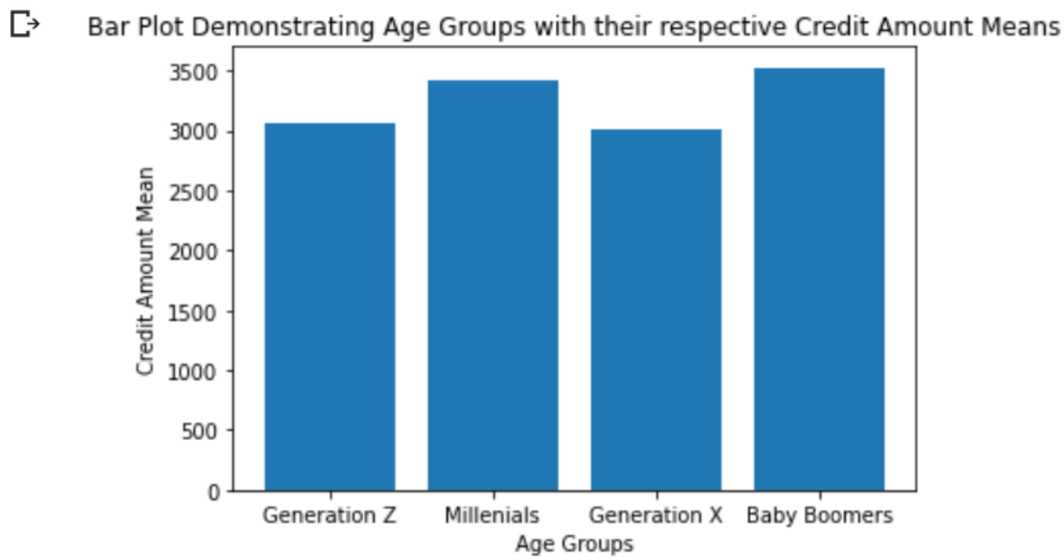
Ages have been grouped on Baby Boomers, Generation X, Millennials and Generation Z

```
scaler=preprocessing.MinMaxScaler()
cols=normal_german_cat.columns
normal_german_cat = scaler.fit_transform(normal_german_cat)
```

Variables are scaled to minimize the effect of high standard deviations in credit_amount and duration

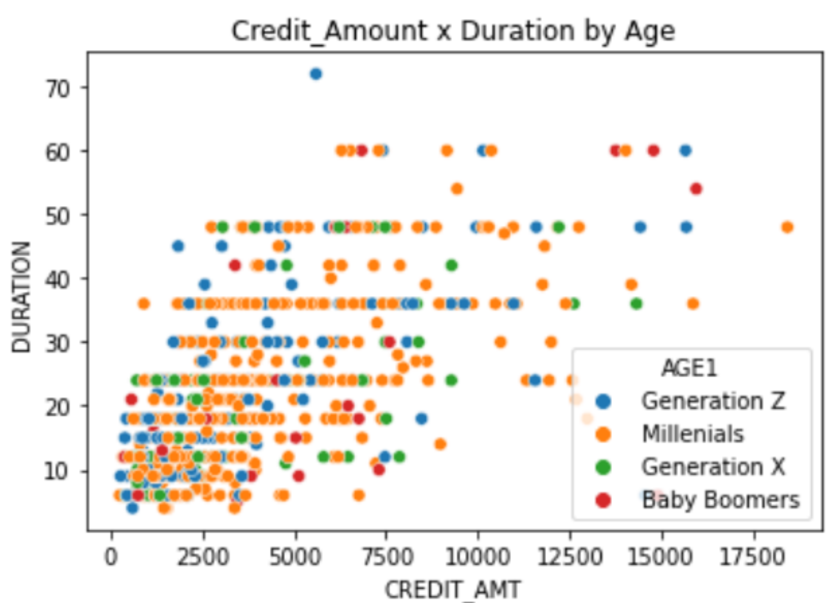
1.4 Plots

```
plt.bar(age_labels2,means)
plot_labels("Age Groups","Credit Amount Mean","Bar Plot Demonstrating Age Groups with their respective Credit Amount Means")
```



Age Groups don't have much difference between their credit amounts. The amount credited doesn't depend on the age of the person

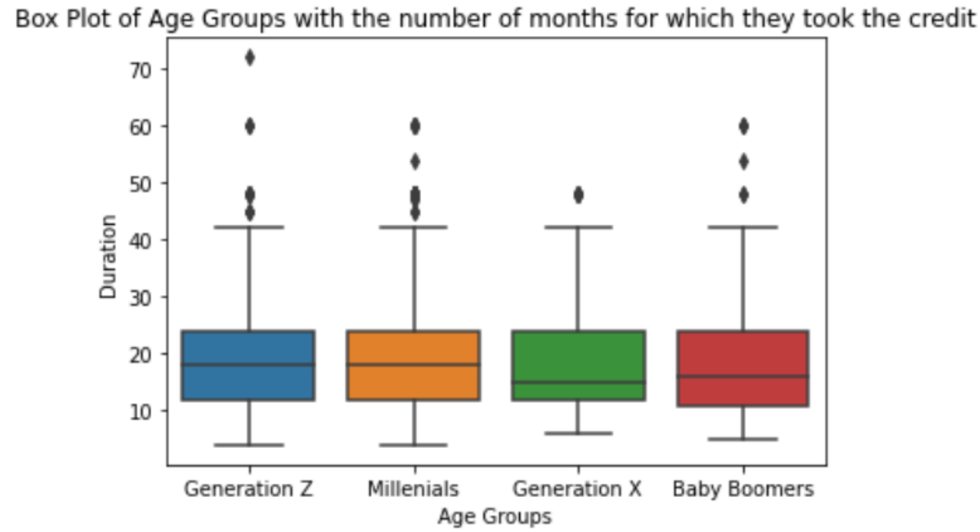
```
sns.scatterplot(x='CREDIT_AMT',y='DURATION',hue='AGE1',data=age_german_cat)
plt.title("Credit_Amount x Duration by Age")
plt.show()
```



No specific relationship between Credit Amount and duration when grouped by Age. However, generally there is a linear relationship between credit amount and duration as the duration tends to be more when a high amount is credited.

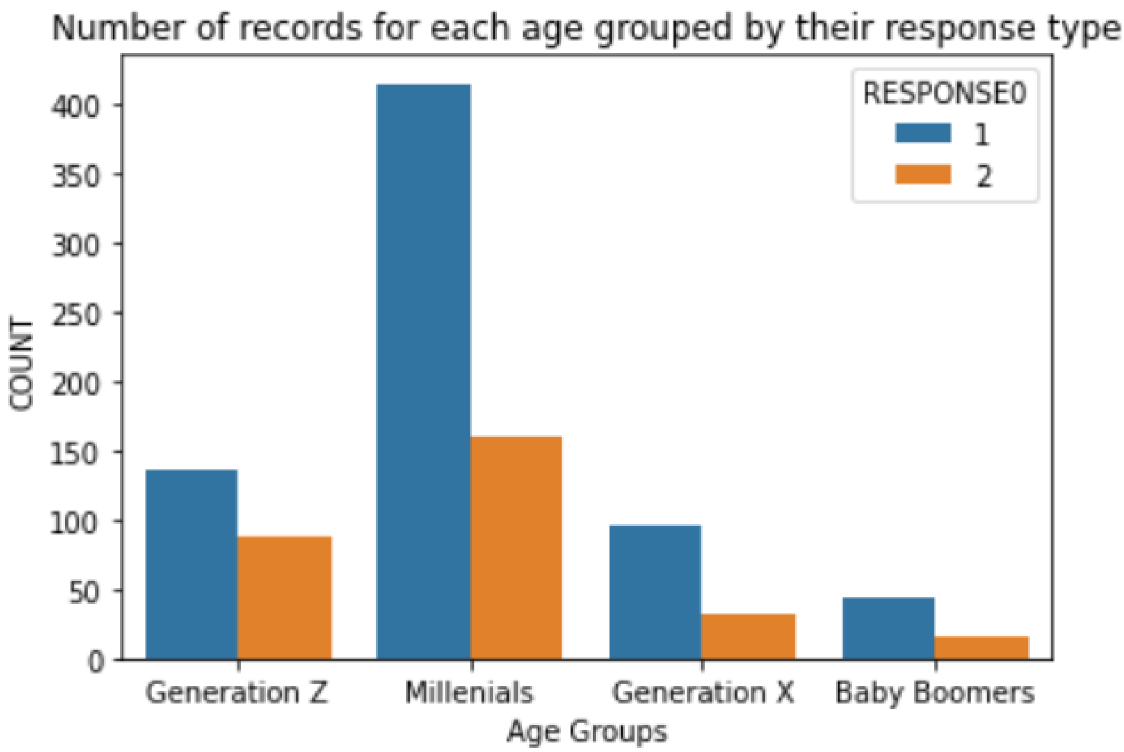
```
sns.boxplot(x='AGE1',y='DURATION',data=age_german_cat)
```

```
plot_labels("Age Groups","Duration","Box Plot of Age Groups with the number of months for which they took the credit")
```



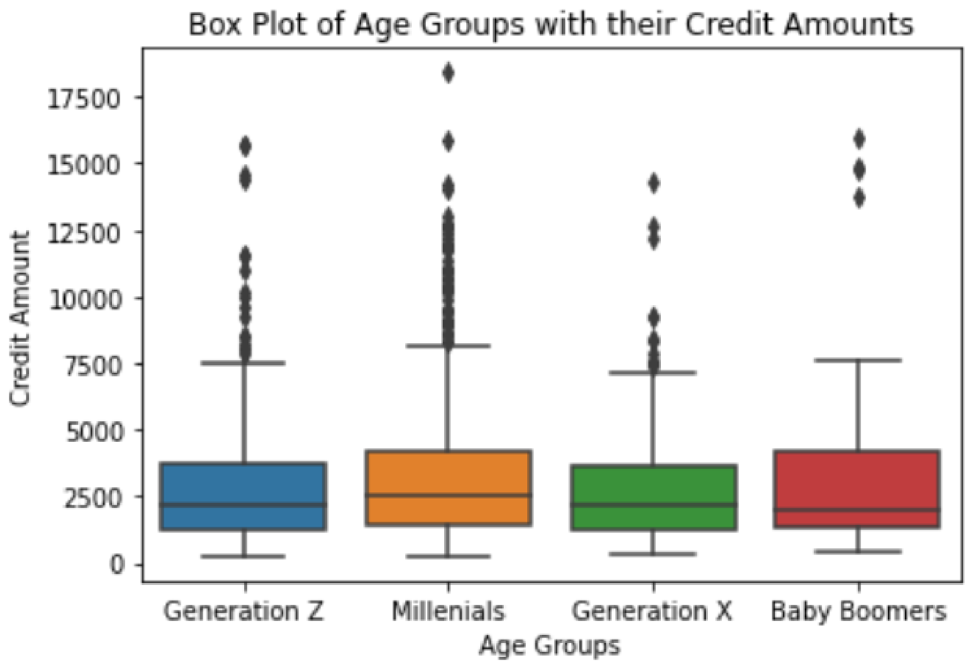
Some millennials have higher durations than usual but generally there are not that many outliers that can be seen

```
sns.countplot(x='AGE1',hue='RESPONSE0',data=age_german_cat)
plot_labels("Age Groups","COUNT","Number of records for each age grouped by their response type")
```



On closer inspection, it's seen that Generation Z has more than 50% of bad credit which reveals that they should be addressed with care. As they are young, they are less likely to pay back due to limited assets with them.

```
sns.boxplot(x='AGE1',y='CREDIT_AMT',data=age_german_cat)
plot_labels("Age Groups","Credit Amount","Box Plot of Age Groups
with their Credit Amounts")
```



Millennials and Baby Boomers have lots of outliers in Credit_Amount. Due to old age, they have more money. Normalization is necessary for this variable.

2. ML MODELS

```
def loss_calc(report_mat):
  loss=0
  loss+=report_mat[0][1]*100
  loss+=report_mat[1][0]*500
  return loss
x_train, x_test, y_train, y_test =
train_test_split(scaled_normal_german_cat[red_cols],
normal_german_cat_target['RESPONSE0'],
test_size=0.4,random_state=42)
```

2.1 Logistic Regression

```
logRegressor = LogisticRegression()
metric_show(y_test,predictions,"Logistic Regression")
```

```
Accuracy on Logistic Regression: 0.765
Precision on Logistic Regression: 0.7781065088757396
Recall on Logistic Regression: 0.9326241134751773
[[ 43  75]
 [ 19 263]]
Loss on Regression:  17000
```

2.2 Decision Tree

```
classifier=DecisionTreeClassifier(criterion='entropy',max_depth=6)
metric_show(y_test,predictions,"Decision Tree")
```

```
Accuracy on Decision Tree: 0.7475
Precision on Decision Tree: 0.7767584097859327
Recall on Decision Tree: 0.900709219858156
[[ 45  73]
 [ 28 254]]
Loss on Regression:  21300
```

2.3 Neural Network

```
mlp = MLPClassifier(hidden_layer_sizes=(3,3,3))
metric_show(y_test,predictions,"Neural Network")
```

```
Accuracy on Neural Network: 0.7175
Precision on Neural Network: 0.7253333333333334
Recall on Neural Network: 0.9645390070921985
[[ 15 103]
 [ 10 272]]
Loss on Regression:  15300
```

2.4 Discriminant Analysis

```
lda=LinearDiscriminantAnalysis()
metric_show(y_test,predictions,"Discriminant Analysis")
```

```
Accuracy on Discriminant Analysis: 0.765
Precision on Discriminant Analysis: 0.7831325301204819
Recall on Discriminant Analysis: 0.9219858156028369
[[ 46  72]
 [ 22 260]]
Loss on Regression:  18200
```

3. CONCLUSION:

Young people should be given credit carefully due to a high risk of bad credit.

Highest Accuracy is provided by Logistic Regression and Discriminant Analysis at 76.5%.
Neural Network provides us with the highest at 96.45% recall when the hidden layer has 3x3x3 neurons. This does result in a steep decline in its accuracy which becomes 71.75%

In this scenario, recall has the highest priority as a bad credit being classified as good is much more harmful than a good credit being classified as bad. The best two models thus are Logistic Regression and Neural Network with losses of 17000 and 15300 respectively. With a higher volume of data, Logistic Regression may perform better than Neural Network but with the current data, Neural Network will be the preferred model.