Plotting the 10 most important variables

Parker Christenson

MSAAI 510

These variables are just what I think are the most important variables to plot...

`Link to the dataset: https://www.kaggle.com/c/home-credit-default-risk/data`

In [ ]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import plotly.graph_objects as go
import umap
```

```
WARNING:tensorflow:From c:\Users\tehwh\anaconda3\Lib\site-packages\keras\src\losses.py:2976: The name tf.losses.sparse_softmax_cross_entropy is deprecated. Please use tf.compat.v1.losses.sparse_softmax_cross_entropy instead.
```

In [ ]:
```
df = pd.read_csv('train_data.csv')

# also getting the cols of the df

#df.columns

'''
So I was not able to get the cols using the df.columns method, so I used the extent
think of it like a power query editor, but you get to write code in python.
'''
```

Out[ ]:
```
'\nSo I was not able to get the cols using the df.columns method, so I used the ex
tention sanddance to get the cols and view the data\nthink of it like a power quer
y editor, but you get to write code in python. \n'
```

In [ ]:
```
# getting the unique vals of the 'TARGET' col
df['TARGET'].unique()

# assuming that the 0 is "DENIED" and 1 is "APPROVED" changing the values into stri
df['TARGET'] = df['TARGET'].map({0:'DENIED', 1:'APPROVED'})
```

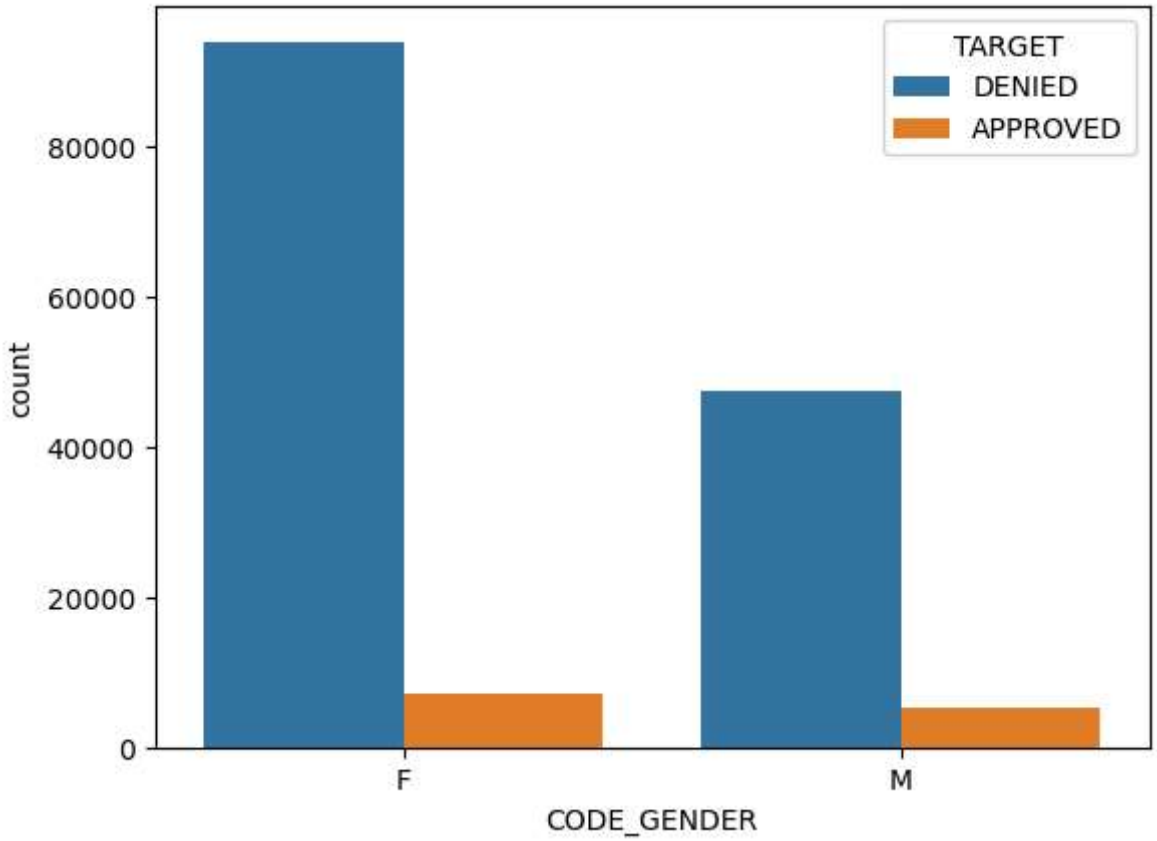
Plot 1

In [ ]:
```
# using plotly to graph the amount of approved and denied applications by the loan
fig = px.histogram(df, x='NAME_CONTRACT_TYPE', color='TARGET', title='Loan Type vs
fig.show()
```

Plot 2

In [ ]:
```
# using sns to plot the amount of approved loans by the CODE_Gender col
sns.countplot(x='CODE_GENDER', data=df, hue='TARGET')
```

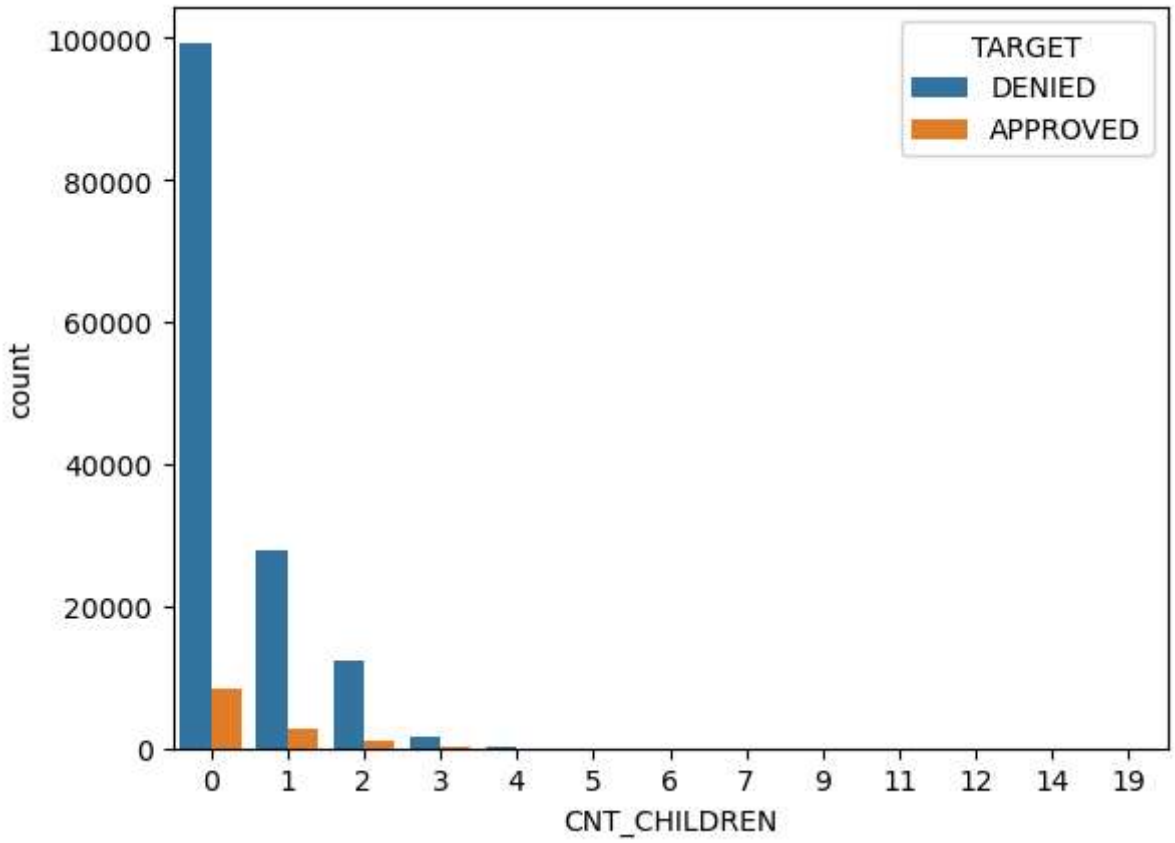
Out[ ]:
```
<Axes: xlabel='CODE_GENDER', ylabel='count'>
```



Plot 3

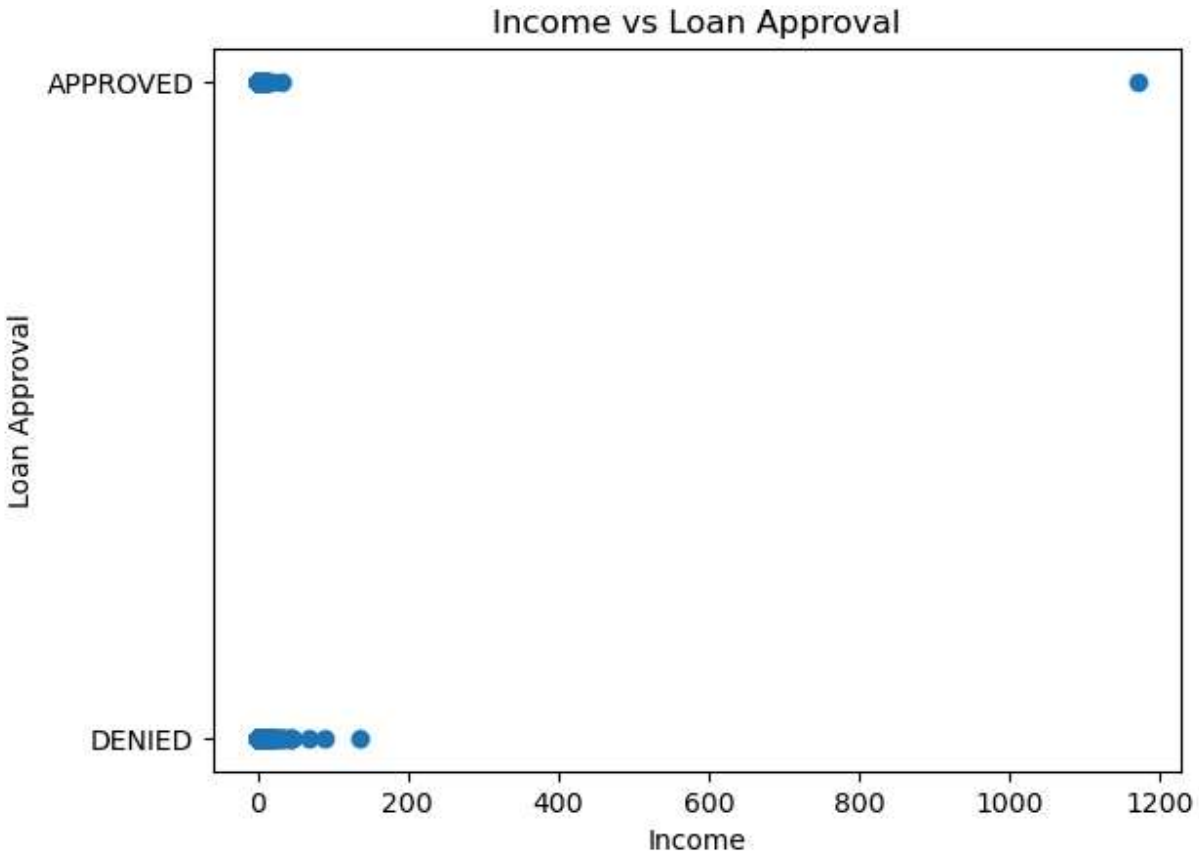
In [ ]:
```
# using the 'CNT_CHILDREN' col to plot the amount of approved loans by the amount o
sns.countplot(x='CNT_CHILDREN', data=df, hue='TARGET')

# also printing out the unique count of the 'CNT_CHILDREN' col
df['CNT_CHILDREN'].value_counts()
```

Out[ ]:
```
CNT_CHILDREN
0     107584
1      30670
2      13415
3       1811
4        210
5         44
6         10
7          4
12         2
19         2
14         1
11         1
9          1
Name: count, dtype: int64
```



Plot 4

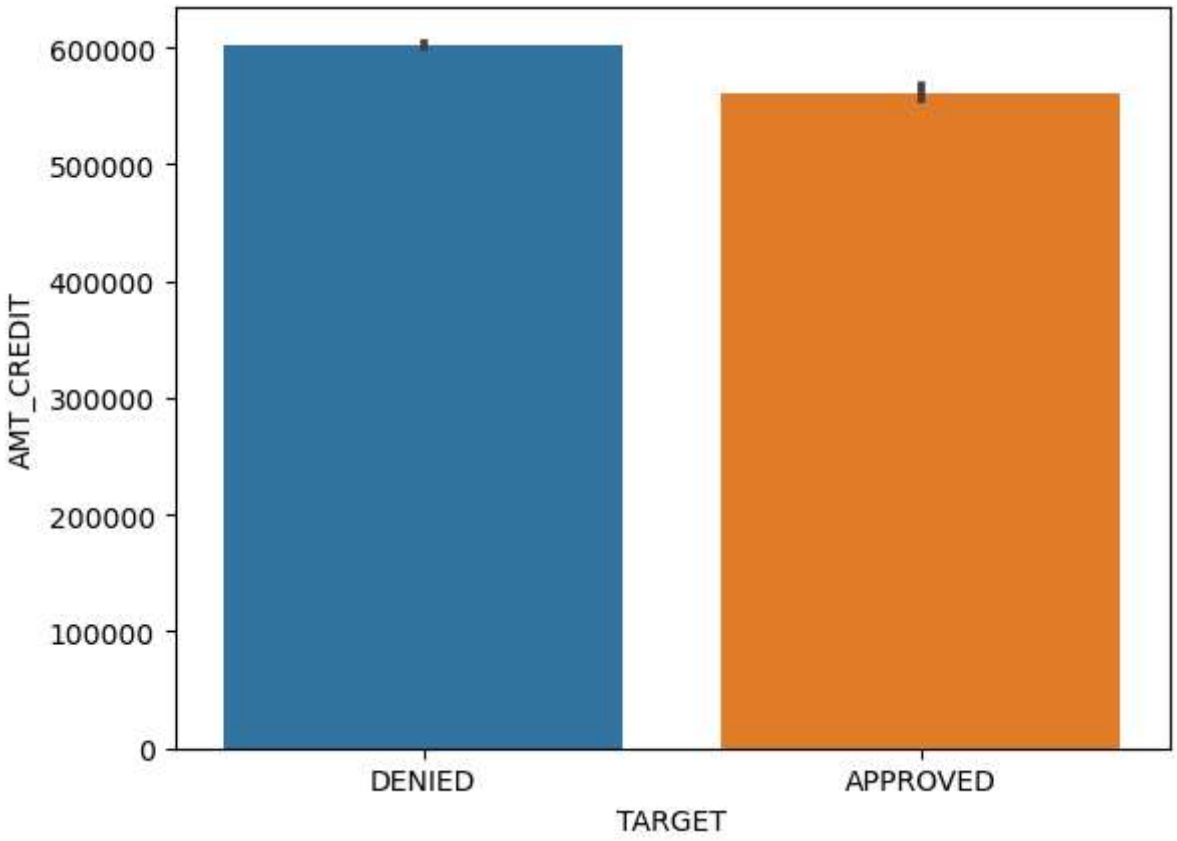
In [ ]:
```
# creating a new col to group the 'AMT_INCOME_GROUP' by 100K increments
df['AMT_INCOME_GROUP'] = df['AMT_INCOME_TOTAL'] // 100000


plt.scatter(df['AMT_INCOME_GROUP'], df['TARGET'])
plt.xlabel('Income')
plt.ylabel('Loan Approval')
plt.title('Income vs Loan Approval')
plt.show()
```



Plot 5

In [ ]:
```
# bar plot based off what I am assuming is the credit limit which is the 'AMT_CREDI
sns.barplot(x='TARGET', y='AMT_CREDIT', data=df)
```

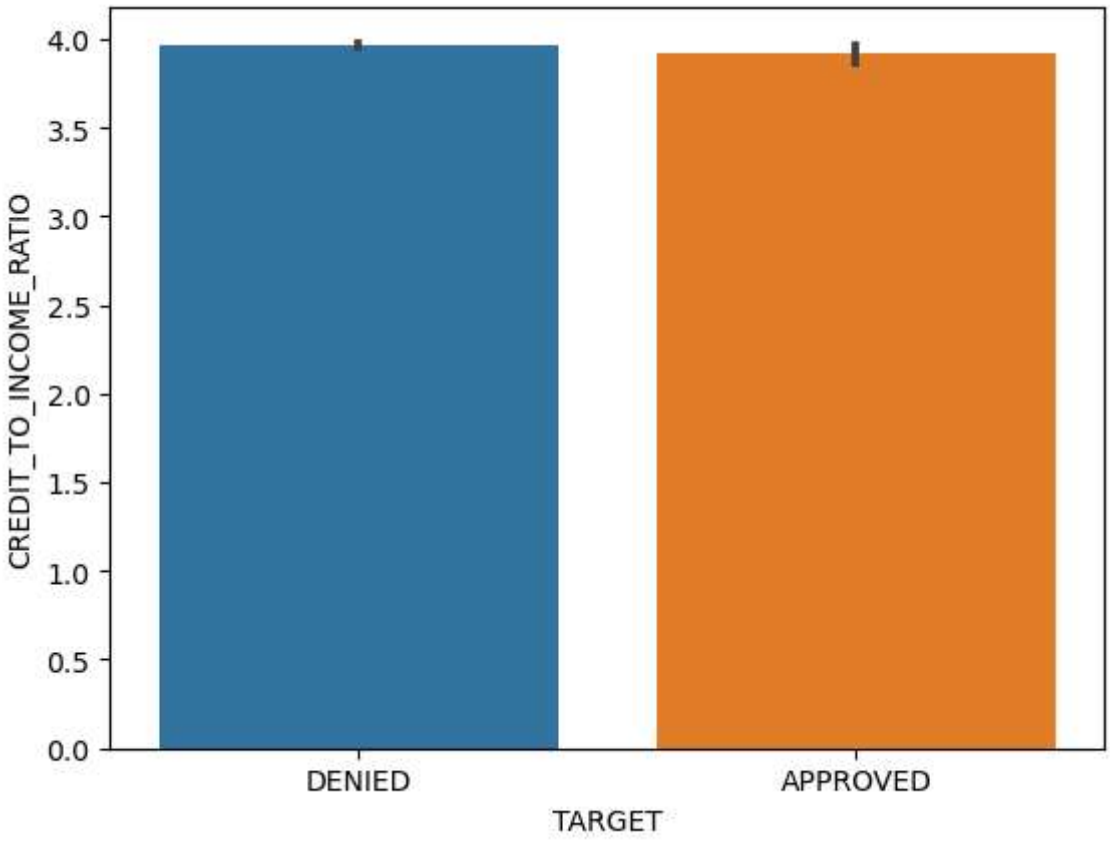
Out[ ]:
```
<Axes: xlabel='TARGET', ylabel='AMT_CREDIT'>
```



Plot 6

In [ ]:
```
# Doing some feature extraction here from the data set to get ratios of the 'AMT_CR
df['CREDIT_TO_INCOME_RATIO'] = df['AMT_CREDIT'] / df['AMT_INCOME_TOTAL']

# using the 'CREDIT_TO_INCOME_RATIO' col to plot the amount of approved loans by th
sns.barplot(x='TARGET', y='CREDIT_TO_INCOME_RATIO', data=df)
```

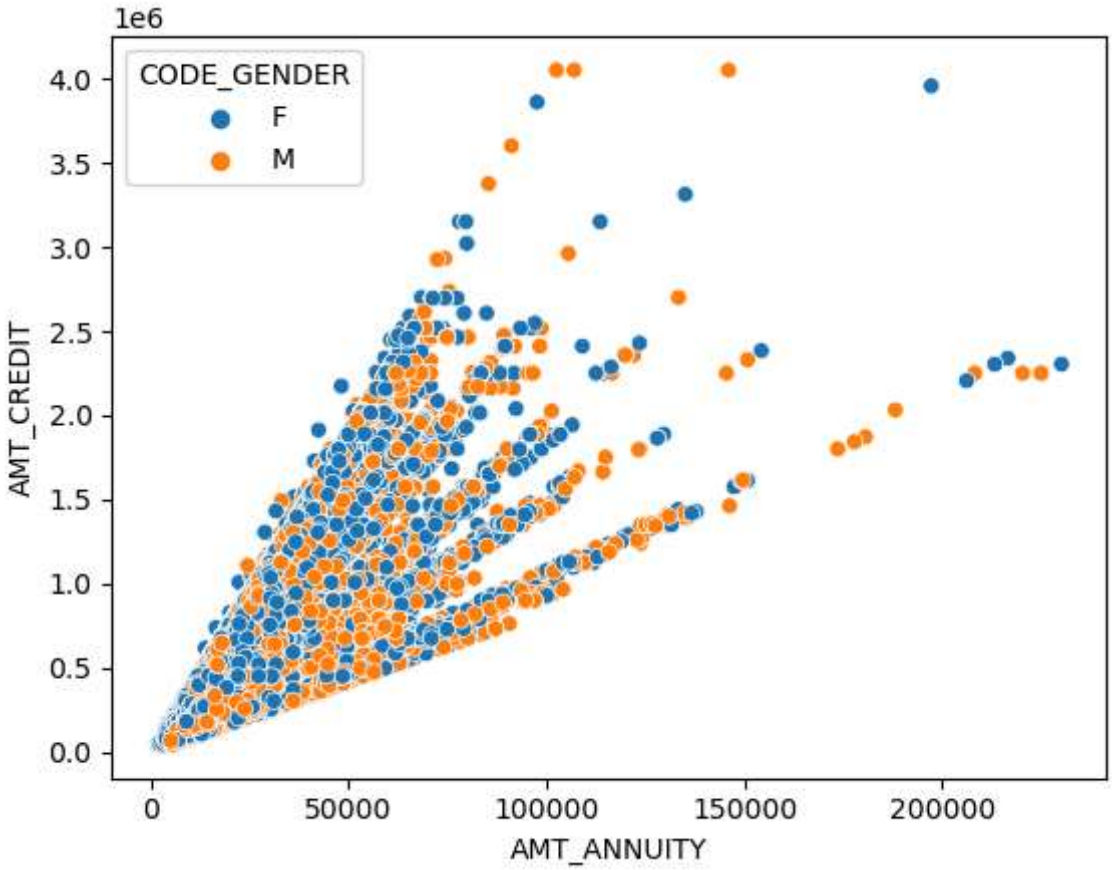
Out[ ]:
```
<Axes: xlabel='TARGET', ylabel='CREDIT_TO_INCOME_RATIO'>
```



Plot 7

In [ ]:
```
# doing two scatter plots a scatter plot based off the 'AMT_ANNUITY' and 'AMT_CREDI
sns.scatterplot(x='AMT_ANNUITY', y='AMT_CREDIT', data=df, hue='CODE_GENDER')
```

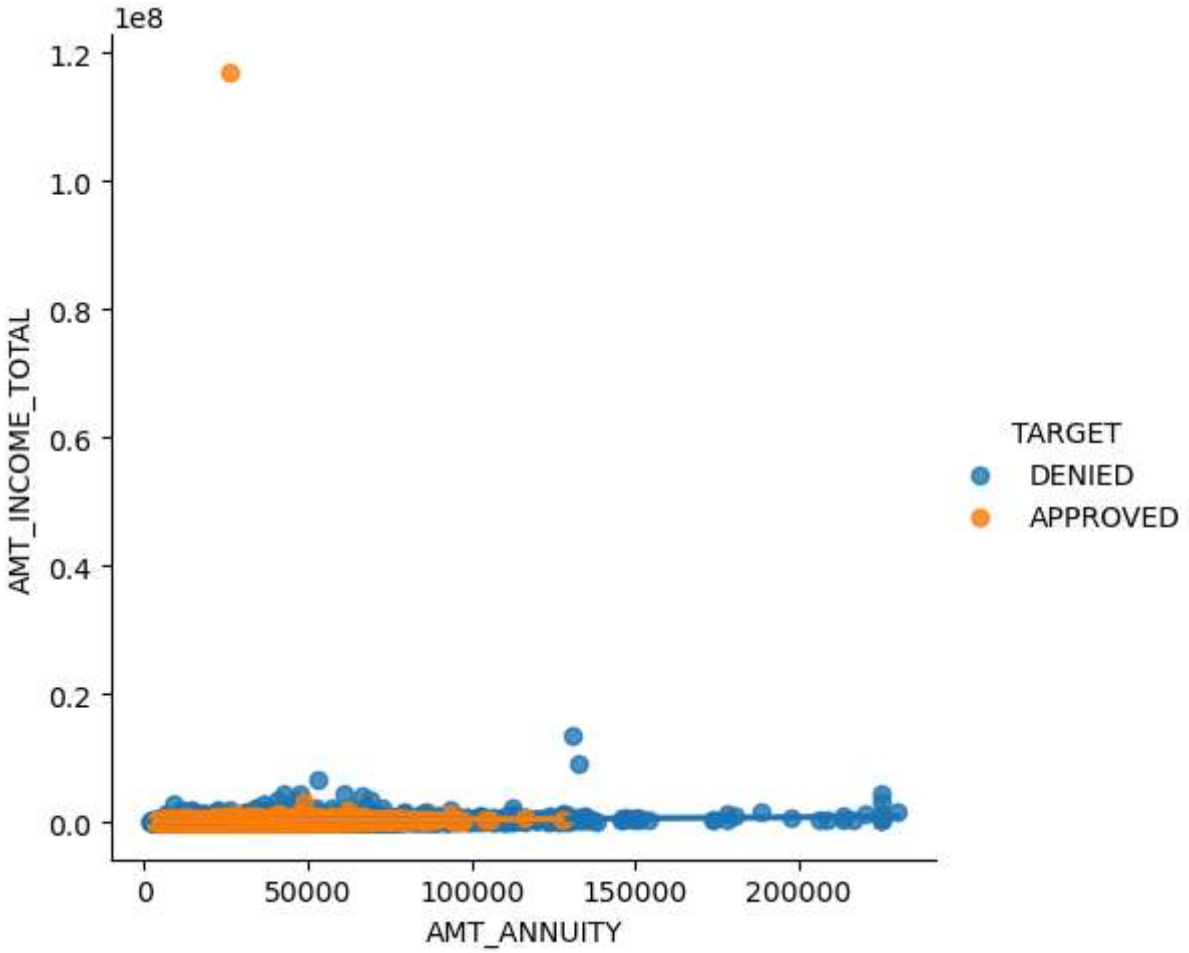
Out[ ]:
```
<Axes: xlabel='AMT_ANNUITY', ylabel='AMT_CREDIT'>
```



Plot 8

In [ ]:
```
# Now doing it based off the 'AMT_ANNUITY' and 'AMT_INCOME_TOTAL' columns, but with
sns.lmplot(x='AMT_ANNUITY', y='AMT_INCOME_TOTAL', data=df, hue='TARGET')
```

Out[ ]:
```
<seaborn.axisgrid.FacetGrid at 0x17f22884210>
```



In [ ]:
```
# now getting the description of the data set
df.describe()
```

Out[ ]:

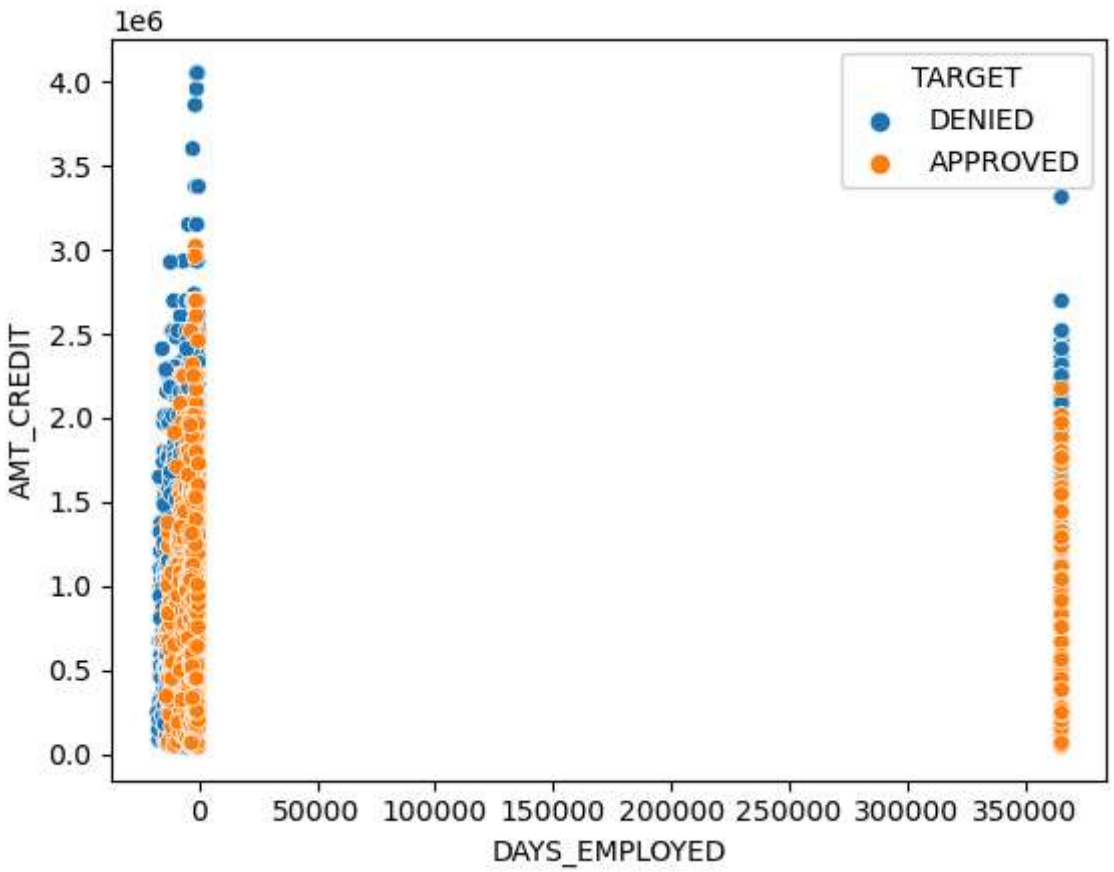
	SK_ID_CURR	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREDIT	AMT_ANNUITY
count	153755.000000	153755.000000	1.537550e+05	1.537550e+05	153750.000000
mean	277867.616930	0.417398	1.692611e+05	5.988824e+05	27083.127015
std	102831.742645	0.722523	3.180805e+05	4.023748e+05	14468.883776
min	100004.000000	0.000000	2.565000e+04	4.500000e+04	1615.500000
25%	188542.000000	0.000000	1.125000e+05	2.700000e+05	16506.000000
50%	277749.000000	0.000000	1.462500e+05	5.135310e+05	24903.000000
75%	366718.000000	1.000000	2.025000e+05	8.086500e+05	34587.000000
max	456255.000000	19.000000	1.170000e+08	4.050000e+06	230161.500000

8 rows × 107 columns

Plot 9

In [ ]:
```
# plotting the days employed vs the amount of credit together with the target
sns.scatterplot(x='DAYS_EMPLOYED', y='AMT_CREDIT', data=df, hue='TARGET')
```

Out[ ]: <Axes: xlabel='DAYS_EMPLOYED', ylabel='AMT_CREDIT'>



Plot 10

In [ ]:
```
# making a correlation matrix
data = df[['CODE_GENDER', 'NAME_CONTRACT_TYPE', 'AMT_CREDIT', 'AMT_ANNUITY', 'AMT_G
print(data['TARGET'].unique())
target_mapping = {'DENIED': 0, 'APPROVED': 1}
data['TARGET'] = data['TARGET'].map(target_mapping)
data_encoded = pd.get_dummies(data, columns=['CODE_GENDER', 'NAME_CONTRACT_TYPE', '
corr = data_encoded.corr()

# plotting
plt.figure(figsize=(12, 10))
cmap = sns.diverging_palette(230, 20, as_cmap=True)
sns.heatmap(corr, annot=True, cmap=cmap, vmax=.3, center=0,
            square=True, linewidths=.5, cbar_kws={"shrink": .5})
plt.show()
```

```
['DENIED' 'APPROVED']
```

```
C:\Users\tehwh\AppData\Local\Temp\ipykernel_16988\1390636616.py:13: SettingWithCopyW
arning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
```