Counterfeit Currency Detection

Thomas Powell

Tagliatela College of Engineering, University of New Haven

DSCI 6003: Introduction to Machine Learning

Dr. Md Moinuddin Bhuiyan

June 25, 2024

Abstract

Counterfeit currencies pose significant challenges to global economies, necessitating robust detection and classification systems. This project aims to develop a comprehensive approach for classifying counterfeit currencies using machine learning techniques. By leveraging a dataset containing various attributes of non-counterfeit and counterfeit banknotes, we implement and compare the efficacy of two algorithms: logistic regression and SGD classification. The project involves extensive pre-processing of data, exploratory data analysis, and the deployment of learning models to achieve it's desired goals. Initial results demonstrate that both models achieved an accuracy rate of ~50.0%, attributed to the uniform nature of the data utilized. This study underscores the potential of machine learning to enhance the reliability and efficiency of counterfeit detection systems, providing a scalable solution for financial institutions worldwide.

Introduction

The circulation of counterfeit currencies remains a persistent and evolving threat to global financial stability, undermining the integrity of monetary systems and eroding public trust in economic institutions. The advent of sophisticated printing technologies and the proliferation of high-quality counterfeits have made traditional detection methods increasingly inadequate. Consequently, there is a critical need for advanced, automated solutions capable of reliably distinguishing between genuine and counterfeit banknotes. Recent advancements in machine learning offer promising avenues for enhancing counterfeit detection systems.

This project focuses on developing a machine-learning-based classification system for counterfeit currencies. We aim to implement and compare the effectiveness of two machine learning models, logistic regression and SGD classification, in accurately identifying counterfeit banknotes. The study leverages a comprehensive dataset that captures various attributes of both genuine and counterfeit currencies, enabling a detailed analysis and comparison of different models.

Through this research, we seek to address the limitations of existing counterfeit detection methods and contribute to the development of more reliable and efficient systems. By enhancing the ability to detect counterfeit currencies, this project aims to support the efforts of financial institutions in safeguarding economic transactions and maintaining public confidence in monetary systems.

Methods

Data Collection

I obtained the data for this project from the "Fake Currency Data" dataset on Kaggle. This dataset contains 1,000,000 examples of currency, with three currencies representing Great British Pounds and European Euros, each with ~250,000 examples, and United States Dollars with ~500,000 examples. The currency examples contain data for size, value, serial number, security features, and country of origin. Overall, the collection is comprehensive enough to enable an exploration of counterfeit classification methods.

	Country	Denomination	Counterfeit	SerialNumber	SecurityFeatures	Weight	Length	Width	Thickness
0	USA	$100	1	25973198	Hologram	1.731759	130.243185	66.537999	0.098488
1	USA	$20	1	95903230	Security Thread	1.002179	152.596364	76.135834	0.094119
2	EU	€10	0	82937914	Hologram	2.306713	152.857126	66.772442	0.061393
3	USA	€20	1	23612989	Microprint	1.366965	143.133672	78.377052	0.053114
4	EU	€20	1	56025342	Watermark	1.796075	129.664777	75.916093	0.051438
...	...	...	...	...	...	...	...	...	...
999995	EU	$100	1	24436622	Watermark	1.472511	134.888731	75.425943	0.093939
999996	EU	£20	1	82654212	Hologram	2.355633	147.830149	65.232274	0.097358
999997	USA	$5	0	59174754	Microprint	1.393764	150.050308	69.273269	0.068363
999998	EU	£10	0	55268089	Watermark	2.026417	142.852137	77.878841	0.081160
999999	EU	£10	0	59464296	Watermark	0.867139	127.645125	72.608513	0.083379

1000000 rows × 9 columns

Data Preprocessing

Prior to model implementation, I used several preprocessing techniques to prepare the data for classification. First, I renamed the Denomination column to Value, created a new Denomination column based on the monetary symbols in the Value column, and removed all the monetary symbols from the Value column.

	Country	Value	Counterfeit	SerialNumber	SecurityFeatures	Weight	Length	Width	Thickness	Denomination
0	USA	100	1	25973198	Hologram	1.731759	130.243185	66.537999	0.098488	USD
1	USA	20	1	95903230	Security Thread	1.002179	152.596364	76.135834	0.094119	USD
2	EU	10	0	82937914	Hologram	2.306713	152.857126	66.772442	0.061393	EUR
3	USA	20	1	23612989	Microprint	1.366965	143.133672	78.377052	0.053114	EUR
4	EU	20	1	56025342	Watermark	1.796075	129.664777	75.916093	0.051438	EUR
...	...	...	...	...	...	...	...	...	...	...
999995	EU	100	1	24436622	Watermark	1.472511	134.888731	75.425943	0.093939	USD
999996	EU	20	1	82654212	Hologram	2.355633	147.830149	65.232274	0.097358	GBP
999997	USA	5	0	59174754	Microprint	1.393764	150.050308	69.273269	0.068363	USD
999998	EU	10	0	55268089	Watermark	2.026417	142.852137	77.878841	0.081160	GBP
999999	EU	10	0	59464296	Watermark	0.867139	127.645125	72.608513	0.083379	GBP

1000000 rows × 10 columns

Next, I used Pandas' getdummies function to one-hot encode the categorical data in the Country, Security Features, and Denomination columns. Lastly, I moved the counterfeit column to the last position for cleanliness.

SecurityFeatures_Hologram	SecurityFeatures_Microprint	SecurityFeatures_Security Thread	SecurityFeatures_Watermark	Country_EU	Country_UK	Country_USA
1	0	0	0	0	0	1
0	0	1	0	0	0	1
1	0	0	0	1	0	0
0	1	0	0	0	0	1
0	0	0	1	1	0	0
...	...	...	...	...	...	...
0	0	0	1	1	0	0
1	0	0	0	1	0	0
0	1	0	0	0	0	1
0	0	0	1	1	0	0
0	0	0	1	1	0	0

Exploratory Data Analysis

For the first portion of my EDA, I performed univariate analysis to examine the characteristics of the variables I would be using in my model. Using Pandas' describe function, I was able to determine that the data did not contain any outliers, missing values, errors, or inconsistencies. The histograms I produced using Seaborn also supported this. (See Appendix)

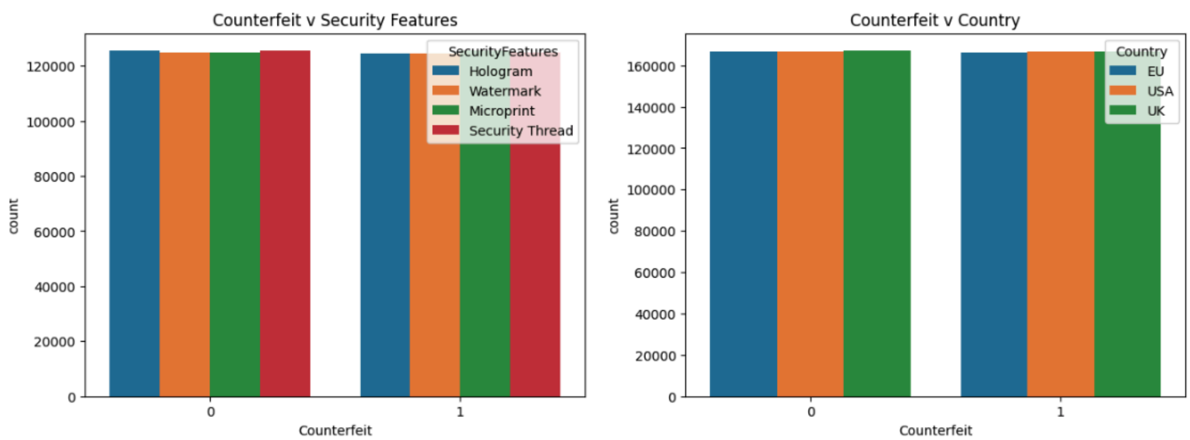
	Value	SerialNumber	Weight	Length	Width	Thickness
count	1000000.000000	1.000000e+06	1000000.000000	1000000.000000	1000000.000000	1000000.000000
mean	21.367511	5.502259e+07	1.649766	140.020542	70.003944	0.074995
std	26.828397	2.598490e+07	0.490712	11.544293	5.772709	0.014442
min	1.000000	1.000015e+07	0.800003	120.000073	60.000005	0.050000
25%	5.000000	3.249784e+07	1.224855	130.034878	64.999762	0.062487
50%	10.000000	5.506594e+07	1.649137	140.032496	70.008440	0.074992
75%	20.000000	7.751115e+07	2.074540	150.022309	75.006372	0.087499
max	100.000000	9.999994e+07	2.499999	159.999961	79.999983	0.100000

In the second portion of my EDA, I performed bivariate analysis to examine the relationship between the variables I would be using in my model. Using Pandas' corr function, I was able to determine there was no significant linear dependency between the continuous variables.

	Value	SerialNumber	Weight	Length	Width	Thickness
Value	1.000000	0.001795	-0.001224	-0.000296	0.000373	-0.000434
SerialNumber	0.001795	1.000000	0.000688	-0.000604	-0.000613	0.000410
Weight	-0.001224	0.000688	1.000000	-0.000037	0.000589	0.000269
Length	-0.000296	-0.000604	-0.000037	1.000000	-0.000152	-0.001139
Width	0.000373	-0.000613	0.000589	-0.000152	1.000000	0.000497
Thickness	-0.000434	0.000410	0.000269	-0.001139	0.000497	1.000000

I was also able to check the dependency of the continuous and categorical variables on the target variable, Counterfeit. I found that the target had no significant dependency on any of the features in the dataset.

	Value	SerialNumber	Weight	Length	Width	Thickness
Counterfeit						
0	21.398738	5.503707e+07	1.650006	140.037239	70.000000	0.074988
1	21.336208	5.500808e+07	1.649525	140.003805	70.007899	0.075002



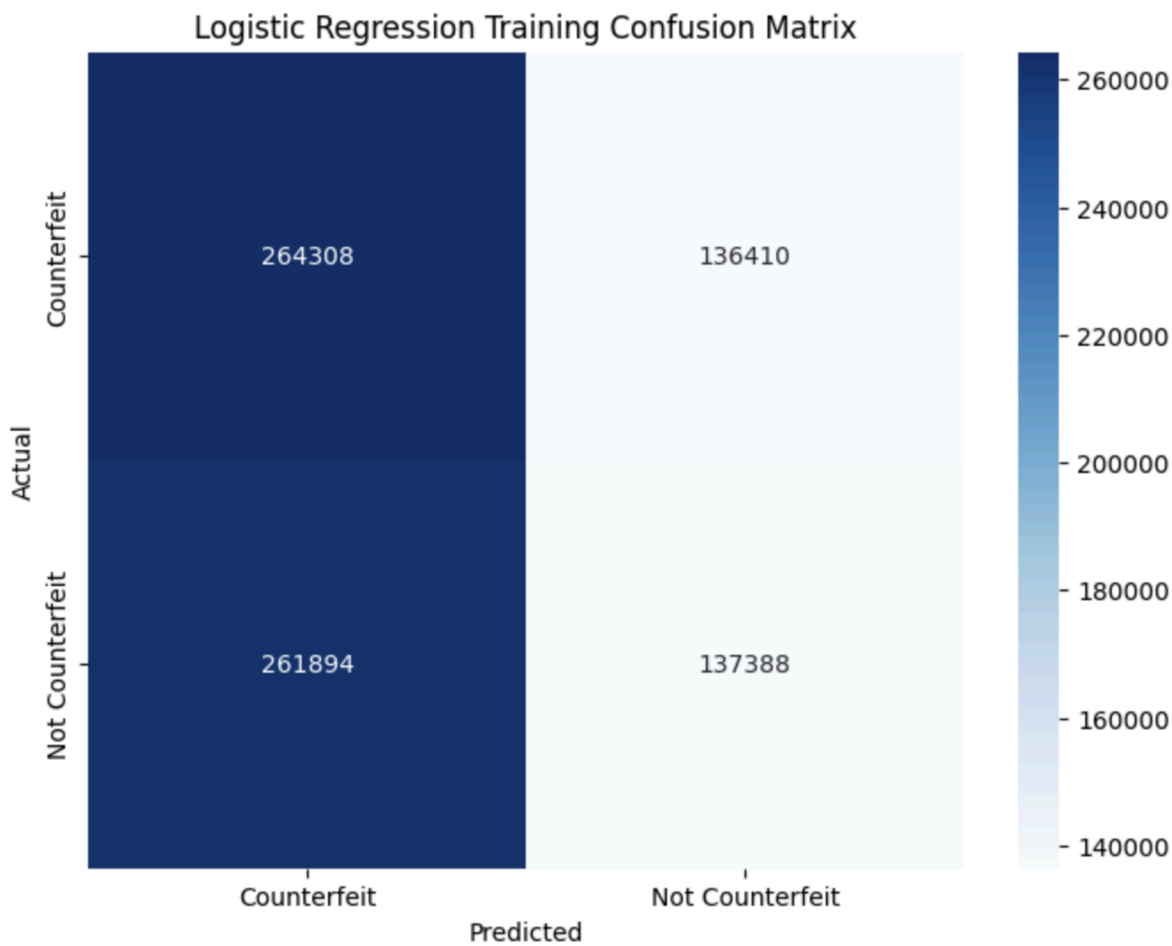
Model Development

Before fitting and training the models, I split my data into an 80% training set and a 20% testing set using Sklearn's train_test_split function. This division allowed me to train my model on the majority of the data while leaving a smaller portion for evaluation of model performance. Since, what I am trying to predict has a binary output, Counterfeit or Not Counterfeit. I decided the best models to use would be basic logistic regression and a SGD classifier, since they both work well with large datasets and binary classification.

Logistic Regression

Using Sklearn's make_pipeline function, I constructed a pipeline using the Standard Scaler to scale to unit variance and subtract the mean from the characteristics to standardize them, and the

Logistic Regression function using the saga (Stochastic Average Gradient) solver, with 1000 being the maximum number of iterations. After fitting the model to my training set, I predicted upon it, using Sklearn's score function to determine the training accuracy, which was 50.21%.

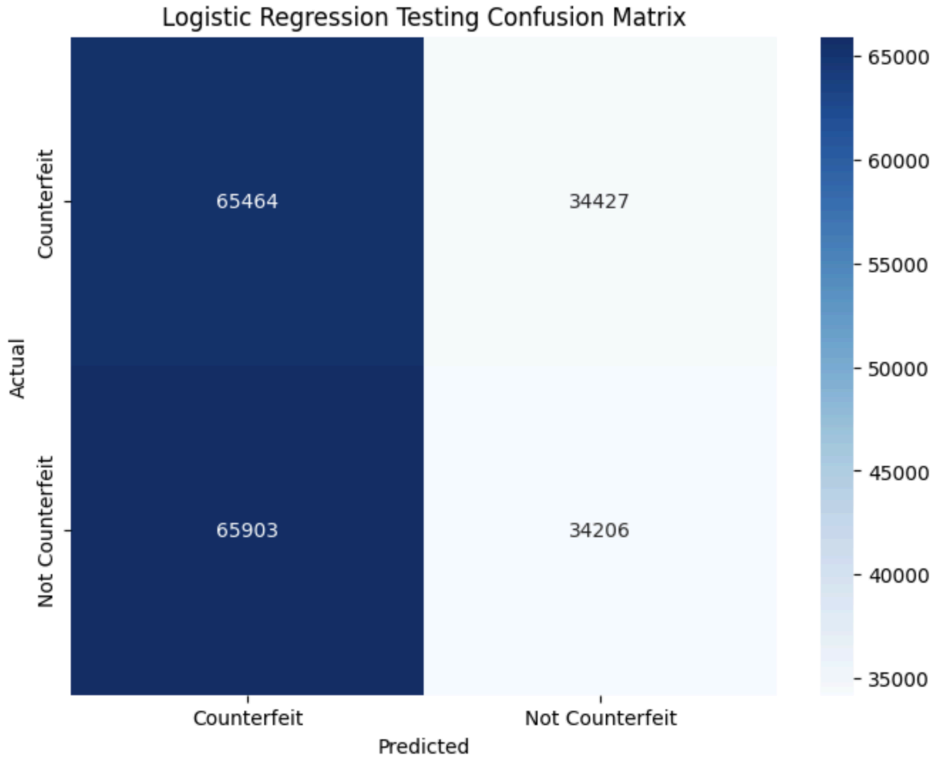


SGD Classifier

Similarly to the Logistic Regression model, I made a pipeline with the Standard Scaler function and the SGD Classifier function, which is a linear SVM classifier with stochastic gradient descent enabled and early stopping enabled. After fitting this new model to my training set, I predicted upon it, using Sklearn's score function to determine the training accuracy, which was 50.07%.

Results and Discussion

After completing training with both the Logistic Regression model and the SGD classifier, I applied the predictor on the test set. With the logistic regression, I achieved a test accuracy of 49.84%, and with the SGD classifier, I achieved a test accuracy of 50.00%.


Logistic Regression Testing Confusion Matrix
Actual
Counterfeit
Not Counterfeit
65464
34427
65903
34206
Counterfeit
Not Counterfeit
Predicted
65000
60000
55000
50000
45000
40000
35000


SGD Classifier Testing Confusion Matrix
Actual
Counterfeit
Not Counterfeit
50672
49219
50777
49332
Counterfeit
Not Counterfeit
Predicted
50600
50400
50200
50000
49800
49600
49400

Around 50.00% accuracy is expected in the case of this data. During the bivariate analysis, I observed that there was no significant correlation between the features and the target variable. This dataset is most likely evenly and randomly generated, with each value generated from an acceptable range and a counterfeit value being randomly assigned. The slight lean toward incorrect counterfeit predictions is probably due to the training set having more non-counterfeit examples, causing it to lean toward its predictions more that way.

Conclusion

In this study, I explored the effectiveness of machine learning models in detecting counterfeit currencies, utilizing logistic regression and SGD classifiers. Despite the thorough data preprocessing and rigorous model training, both models achieved an accuracy rate of approximately 50%, which is indicative of the dataset's uniform and randomized nature. Our analysis revealed no significant correlation between the features and the target variable, highlighting the challenge of distinguishing genuine from counterfeit banknotes using the given dataset.

The random assignment of counterfeit values likely contributed to the models' inability to learn distinctive patterns, resulting in accuracy levels akin to random guessing. Future research should consider leveraging more nuanced datasets with inherent patterns and dependencies that better mimic real-world scenarios. Additionally, exploring more advanced machine learning techniques, such as ensemble methods or neural networks, could potentially enhance detection accuracy.

Ultimately, while the current models did not achieve high accuracy, this study highlights the potential of machine learning in counterfeit detection. Continued advancements and refinements in data collection and model development will be crucial for creating robust, reliable systems that can effectively combat the evolving threat of counterfeit currencies.

References

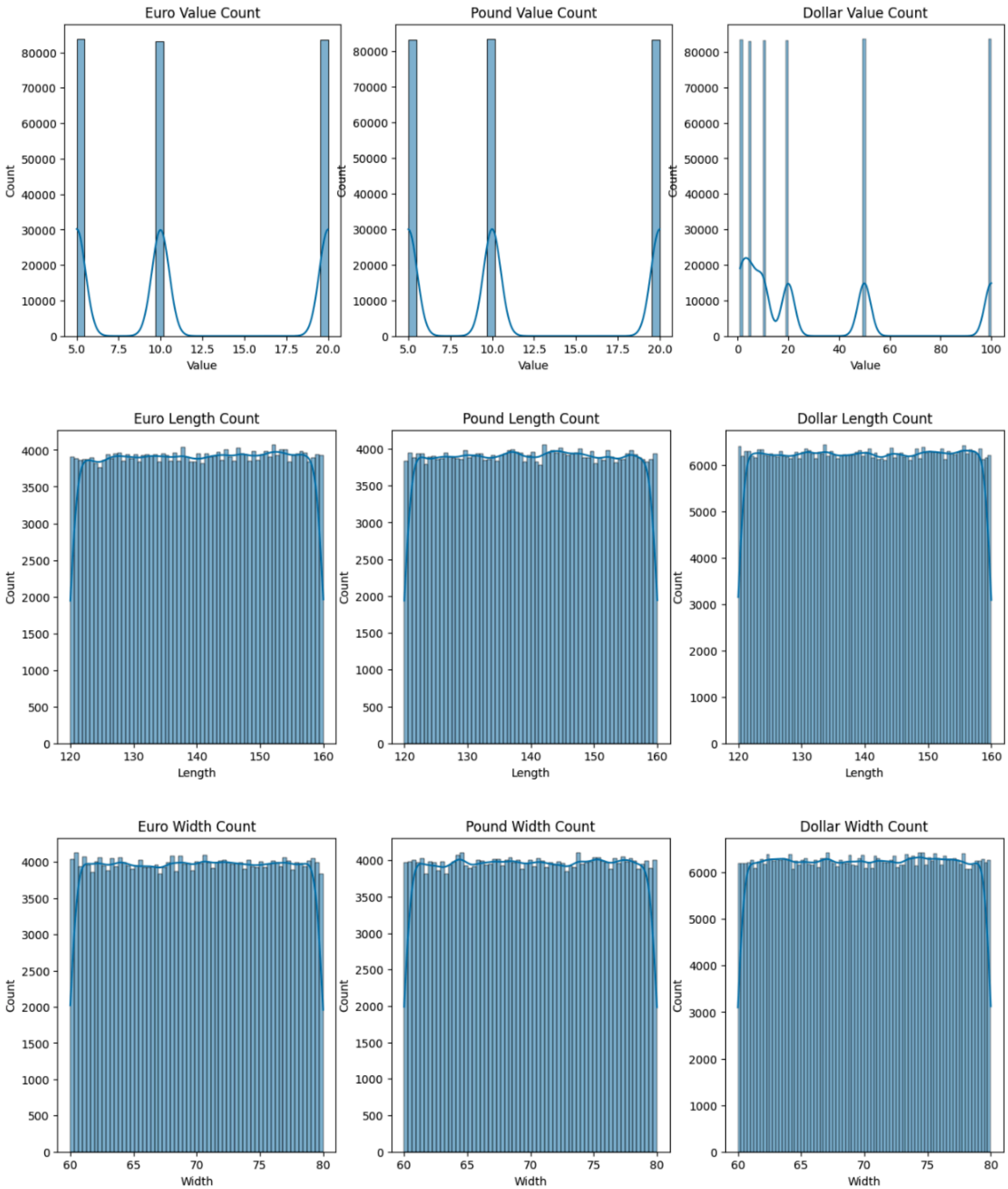
Mdladla. (n.d.). Fake currency data. Kaggle. Retrieved June 25, 2024, from https://www.kaggle.com/datasets/mdladla/fake-currency-data

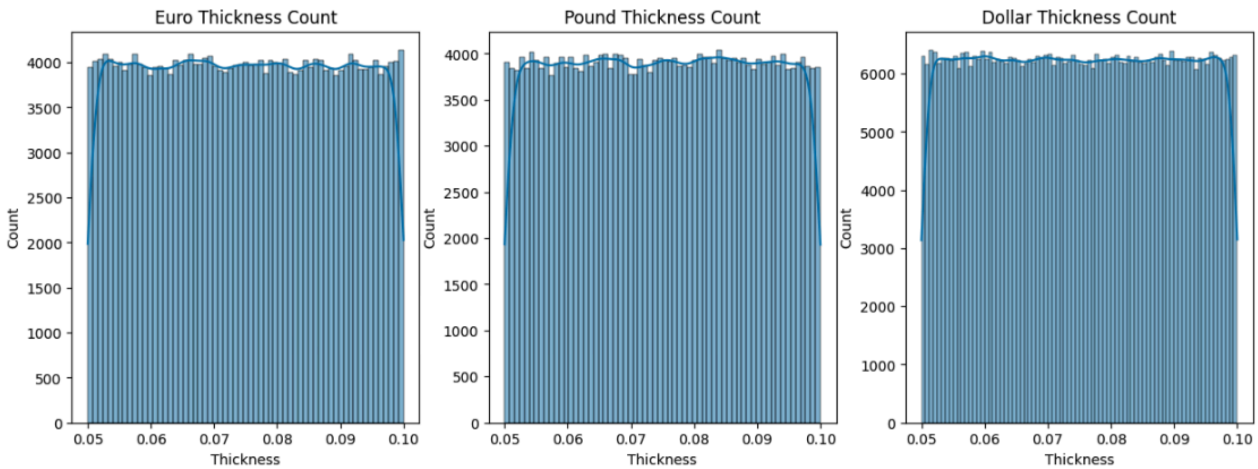
Scikit-learn. (n.d.). sklearn.linear_model.LogisticRegression. Retrieved June 25, 2024, from https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

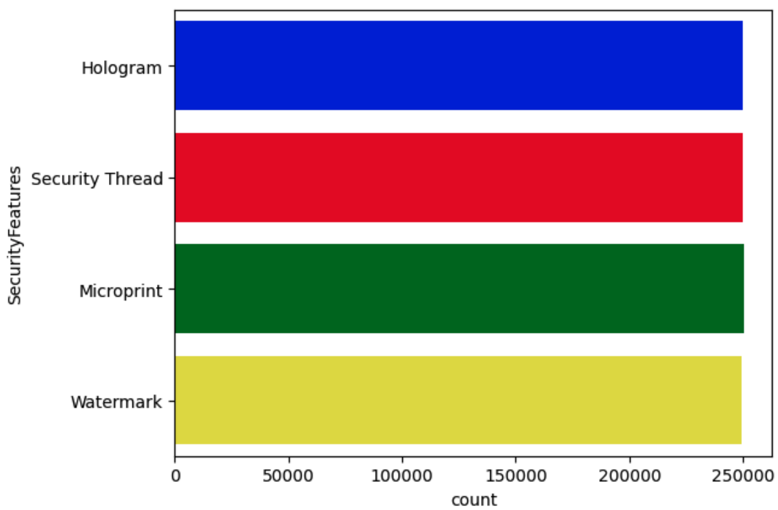
Scikit-learn. (n.d.). sklearn.linear_model.SGDClassifier. Retrieved June 25, 2024, from https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.SGDClassifier.html

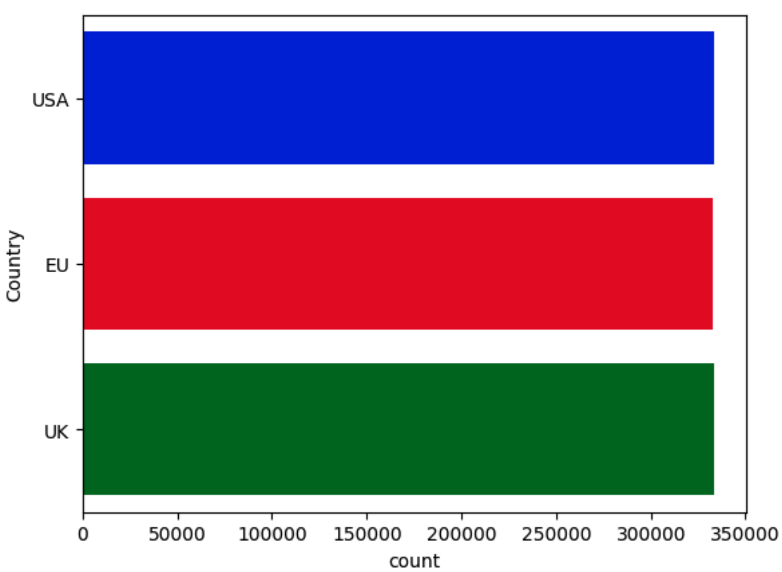
Tanwar, G. (2021, April 19). Univariate, bivariate, and multivariate data analysis in Python. Medium. Retrieved June 25, 2024, from https://gauravtanwar1.medium.com/univariate-bivariate-and-multivariate-data-analysis-in-python-341493c3d173

Appendix


Euro Value Count
Pound Value Count
Dollar Value Count
Count
Value
Euro Length Count
Pound Length Count
Dollar Length Count
Length
Euro Width Count
Pound Width Count
Dollar Width Count
Width


Euro Thickness Count
Pound Thickness Count
Dollar Thickness Count
Count
Thickness
0.05
0.06
0.07
0.08
0.09
0.10


Hologram
Security Thread
Microprint
Watermark
SecurityFeatures
count
0
50000
100000
150000
200000
250000


USA
EU
UK
Country
count
0
50000
100000
150000
200000
250000
300000
350000