Abstract

Logistic regression is a supervised learning algorithm that is used to predict a dependent categorical target variable (2U, Inc., n.d.). It is an algorithm used by many industries, to classify data for several different purposes, for example, to predict whether customers are likely to take up insurance policies.

In this report, a logistic regression model using stochastic gradient descent (SGD) is designed to classify bank notes as genuine or forged based on 4 features taken from images of genuine and forged banknote-like specimens. Further analysis is done by comparing models with different regularization hyperparameter, namely, ridge regression, lasso regression and elastic net regression, and evaluating the accuracy score and confusion matrix of the different models. A comparison is also made with a model using k-nearest neighbors (KNN) algorithm.

Based on the comparison findings described in this report, it is observed that by optimizing the logistic regression model and using elastic net regularization, a logistic regression model can accurately classify bank notes as genuine or forged. Furthermore, the model is able to reduce false negative values (classifying a forged banknote as genuine), which would be highly detrimental to the organization. On further comparison with the model using KNN algorithm, it is noted that the model with the best accuracy score and lowest false negative values for this case is obtained using the KNN algorithm.

Introduction

A logistic regression model is designed to classify bank notes as genuine or forged based on 1372 images that were taken from genuine and forged banknote-like specimens. The following 4 features were extracted from these images:

1. Variance of Wavelet Transformed image (continuous)
2. Skewness of Wavelet Transformed image (continuous)
3. Kurtosis of Wavelet Transformed image (continuous)
4. Entropy of image (continuous)
5. Class (target): Presumably 0 for genuine and 1 for forged

As it will be detrimental to an organisation if a forged banknote was recognised as genuine, the model designed will not only be evaluated on accuracy but on the false negative rate as well.

There are three main types of logistic regression, binary, multinomial and ordinal. Binary logistic regression involves just two possible outcomes, multinomial logistic regression involves three or more classes without ordering and ordinal logistic regression involves three or more classes with ordering. In this report, we would be focusing on a binary logistic regression model.

Binary logistic regression uses a logistic function to frame a binary output model through a sigmoid function. It is an S-shaped curve that stretches from close to zero to close to one (figure 1).

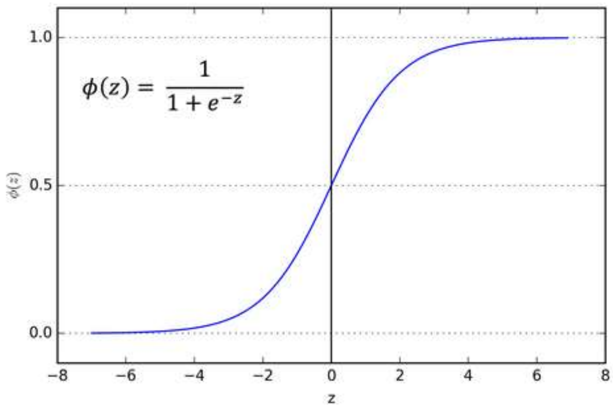


Figure 1. Sigmoid function (Varghese, 2018)

Data is fit into a linear regression model, which the logistic function will use to predict the target categorical dependent variable. The general logistic regression equation (Varghese, 2018) is given as follows:

$$z = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

$$h(\theta) = g(z)$$

$$g(z) = \frac{1}{1 + e^{-z}}$$

Equation 1. General logistic regression equation

Where:

$g(z)$ represents the sigmoid function;
$h(\theta)$ represents $P(y = 1 \,|x)$;
θ_1 … θ_n are the weights; and
θ_0 is the bias/ intercept

As represented in figure 1, when the value of z is 0, $g(z)$ will be 0.5. Since our output is divided into 0 or 1, this means that whenever z is positive, $h(\theta)$ will be greater than 0.5 and output will be 1. Conversely, when z is negative, the ouput will be 0. This value of 0.5 is referred to as the cut-off probability or the threshold and can be set to determine which class a data belongs to.

The cost function for a logistic equation is given below:

$$J(\theta) = \frac{1}{n}\sum cost\ (\hat{y}\,, y)$$

$$cost\ (\hat{y}, y) = -\log(1 - \hat{y}) \quad if\ y = 0$$

$$cost\ (\hat{y}, y) = -\log(\hat{y}) \quad if\ y = 1$$

Equation 2. Cost function of logistic equation

Where:

m represents data size;
$\hat{y}$ represents predicted output
y represents actual output

In this report, SGD method is used to find the minimal cost function, while using different regularization methods for classification. These penalties, as listed below (scikit-learn developers, n.d.), are added to the cost function during training.

1. penalty="l2": L2 norm penalty or ridge regression

$$\alpha\left(\frac{1}{2}\sum_{i=1}^{n}\theta_i^2\right)$$

Equation 3. Ridge regression

2. penalty="l1": L1 norm penalty or lasso regression

$$\alpha\left(\sum_{i=1}^{n}|\theta_i|\right)$$

Equation 4. Lasso regression

3. penalty="elasticnet": Convex combination of L2 and L1

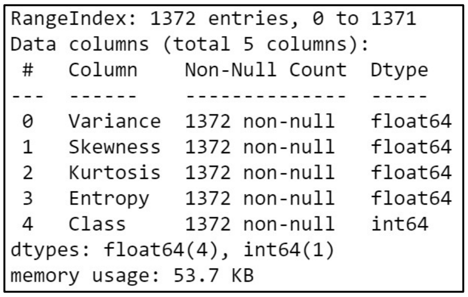
$$\alpha\left(\frac{\rho}{2}\sum_{i=1}^{n}\theta_i^2+(1-\rho)\sum_{i=1}^{n}|\theta_i|\right)$$

Equation 5. Elastic net regression

Where α > 0 is a non-negative hyperparameter that controls the regularization strength and ρ is given by 1 – L1 ratio, which controls the convex combination of L1 and L2 penalty. The higher the value of α, the larger the penalty and therefore, the magnitude of the coefficients is reduced.

Methodology

Before using the dataset to build the logistic regression model, checks were carried out to determine if data pre-processing steps were required for the given dataset. This included checking the number of data entries, data types, missing data. Example is shown in figure 2 below.



```
RangeIndex: 1372 entries, 0 to 1371
Data columns (total 5 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Variance  1372 non-null   float64
 1   Skewness  1372 non-null   float64
 2   Kurtosis  1372 non-null   float64
 3   Entropy   1372 non-null   float64
 4   Class     1372 non-null   int64
dtypes: float64(4), int64(1)
memory usage: 53.7 KB
```

Figure 2. Displayed info of Boston dataset generated

Based on the data pre-processing checks done, there was no further need to perform any data pre-processing on the dataset as there is no missing values and the data is in numerical format (no structural errors). Furthermore, the dataset is a fairly balanced dataset as shown in figure 3 with slightly more data in the negative outcome class.

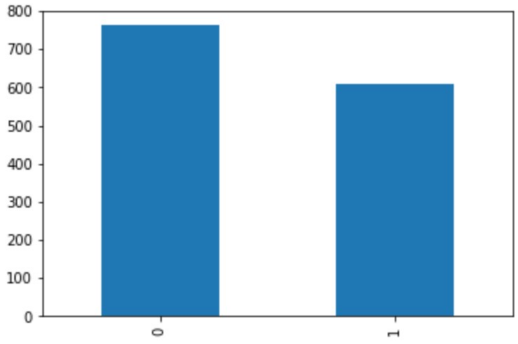


Figure 3. Plot of count of output Class (target)

A visual representation of the dataset is shown in figure 4.

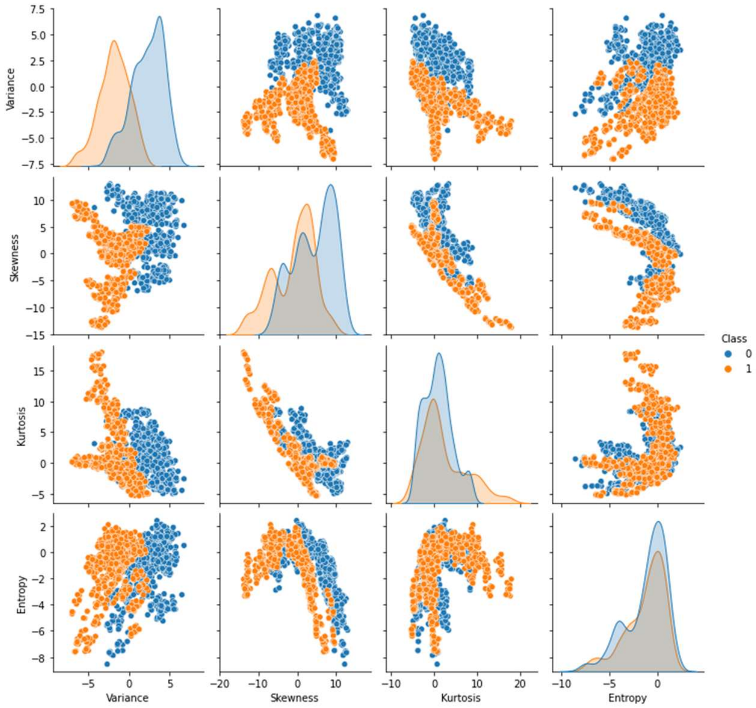


Figure 4. Visual representation of the dataset

Min-max normalization was then conducted on the input features. The formula for min-max normalization is given as:

$$normalized\ data = \frac{data - min}{\max - min}$$

Equation 6. Min-max normalisation

The logistic regression equation to classify whether the bank note is genuine or forged is derived from equation 1 mentioned in "Introduction" section. This is shown in equation 7 below. As there are 4 features, there are 4 regression coefficient parameters (θ_1, θ_2, θ_3, θ_4) that need to be determined and an additional coefficient parameter (θ_0) that provides the intercept or bias. These parameters are required to accommodate the impact of all 4 input variables on the output variable.

$$z = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \theta_4 x_4$$

$$h(\theta) = g(z)$$

$$g(z) = \frac{1}{1 + e^{-z}}$$

Equation 7. Logistic regression equation for banknote authentication dataset

The banknote authentication dataset was then split into a train dataset, comprising of 70% of the original dataset and the remaining 30% was used as the test dataset. The results were used in the comparison of the performance of the machine learning algorithm for this predictive modelling problem.

The model was trained using SGDClassifier from scikit-learn API using 'log' as the loss function ('log' loss gives logistic regression). GridSearchCV was then used to optimize the hyperparameters, alpha, max_iter, tol, learning_rate, to achieve a model with the best accuracy score. Below summarizes the hyperparameters and its impact on the model (scikit-learn developers, n.d.).

- alpha (default = 0.0001): Constant that multiplies the regularization term. The higher the value, the stronger the regularization (bigger the penalty). Also used to compute the learning rate when learning_rate is set to 'optimal'. If alpha is too high, there will be a risk of underfitting.

- max_iter (default = 1000): The maximum number of iterations over the training data. If the maximum number of iterations set is too low, the cost function may not reach convergence.

- tol (default = 1e^{-3}): The tolerance for the stopping criteria. Once the threshold tolerance is crossed, this stops the iterations of a solver. If the value is too big, the algorithm stops before it can converge.

- learning_rate (default='optimal'): Defines the learning rate schedule (gradient of the loss is estimated each sample at a time and the model is updated along the way with a decreasing strength schedule, i.e. the learning rate) used in the algorithm. This includes 'constant' schedule (learning rate is constant and equal to the initial learning rate set), 'optimal' schedule (given by the equation 1.0 / $(alpha * (t + t0)$, 'invscaling' schedule (given by the equation $eta0 / pow(t, power\_t))$ and 'adaptive' schedule (eta = eta0, as long as the training keeps decreasing. Each time n_iter_no_change consecutive epochs fail to decrease the training loss by tol or fail to increase validation score by tol if early_stopping is True, the current learning rate is divided by 5). This affects how fast the cost function converges and the cost value that the cost function converges to.

The results of the optimization and the classification report generated will be described in the "Results" section of this report.

After optimization of alpha, max_iter, tol and learning_rate hyperparameters, further evaluation was done by comparing the optimized model trained using L1, L2 (default regularization parameter used in the optimal model obtained) and elastic net regularization.

Regularization is a way to avoid overfitting of a model by penalizing high-valued regression coefficients. It reduces the parameters and shrinks (simplifies) the model. It is used to analyse data that suffers from multicollinearity as data with multicollinearity results in predicted values being further away from the actual values due to high variances and results in overfitting of a model.

Ridge regression adds the "squared magnitude" of coefficient as penalty term to the cost function and thus reduces model complexity by shrinkage of the model coefficients, giving different importance weights to the features but does not drop unimportant features (Qshick, 2019). However, if the dataset has a large number of features, the model will still remain complex.

Least Absolute Shrinkage and Selection Operator Regression (Lasso) regression adds the "absolute value" of magnitude of coefficient as penalty term to the cost function and thus reduces model complexity by limiting the size of the coefficient. This sometimes results in the elimination of some coefficients altogether, which can yield sparse models (automatic variable selection). However, if there are correlated variables, it retains only one variable and sets other correlated variables to zero. This could possibly lead to loss of information resulting in lower accuracy (Jain, 2017).

Elastic net is a combination of L2 and L1 regularization (Hastie, 2005). It reduces the impact of different features while not eliminating all of the features and overcomes the limitations of both ridge regression and lasso regression, producing a sparse model with good prediction accuracy.

The results of this comparison and the corresponding classification report will also be described in the "Results" section of this report.

Lastly, a KNN model with default hyperparameters was built using KNN from scikit-learn API. KNN algorithm is another supervised machine learning algorithm that can be used to solve both classification and regression problems. It works on the premise that similar things are near to each other. Hence by calculating the distance between the test points and the K number of selected training points closest to the test points, the outcome of the test points can be predicted by obtaining the most frequent label within the K number of selected training points.

Comparison between the accuracy score using KNN's model and the optimized model previously trained will also be described in the "Results" section of this report.

Results

Based on the optimization of learning_rate hyperparameter, it was found that the 'optimal' learning rate resulted in the model with the highest accuracy score (figure 5). Hence, an 'optimal' learning_rate was chosen for the further optimization of the alpha, max_iter and tol hyperparameters.

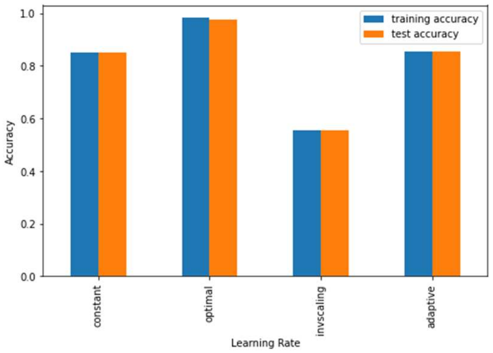


Figure 5. Plotting accuracy vs learning_rate

After further optimization using GridSearchCV, the optimal hyperparameters obtained for alpha is 1e^{-05}, max_iter of 100 and tol of 0.01. These parameters combined, yielded the model with a high accuracy score for both the train dataset and test dataset (Table 1). The model also does not show signs of overfitting. However, as the accuracy score of the model using default parameters was originally high, the improvements to the model accuracy after optimization was minimal.

Model	Accuracy score	
	Train dataset	Test dataset
Default parameters	0.984	0.985
Optimized parameters	0.988	0.990

Table 1. Comparison of accuracy score for default model and optimized model

Accuracy score was not the only evaluation metric that was used to evaluate the model. As mentioned previously, this model was intended to classify the bank notes as genuine or forged. Hence, a good model will also minimize the number of false negative (forged bank note being mistaken as genuine) in the predicted outcomes. The model should therefore achieve a high precision on the negative class.

Recall is defined as the ability of a classification model to identify all data points in a relevant class and precision is defined as the ability of a classification model to return only the data points in a class (Koehrsen, 2022). For this algorithm problem, the ability of the model to return only true genuine bank notes (precision on the negative class) is more important than the ability of the model to identify genuine bank notes (recall on the negative class).

From the confusion matrix, it is observed that the number of false negative has decreased from 15 to 12 using the train dataset and from 6 to 4 using the test dataset for the optimized model compared to the default model (Table 2). This improvement is also reflected in the classification report for the test dataset, where the precision of the negative class has increased slightly from 0.97 to 0.98 for the optimized model compared to the default model (figure 6).

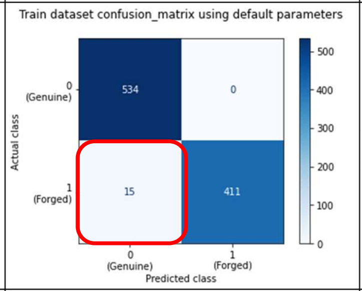
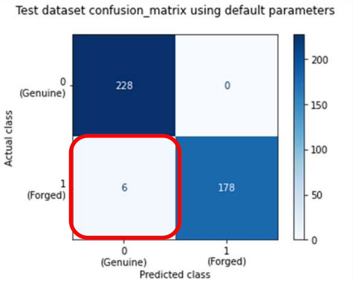
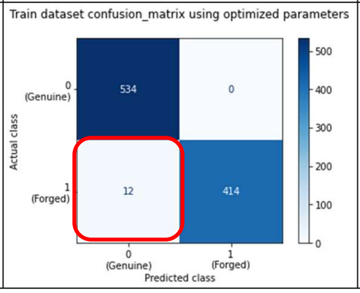
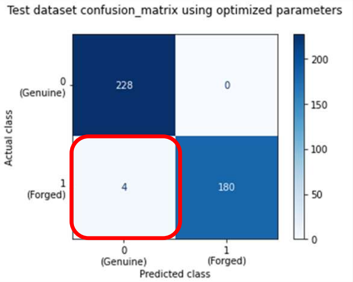
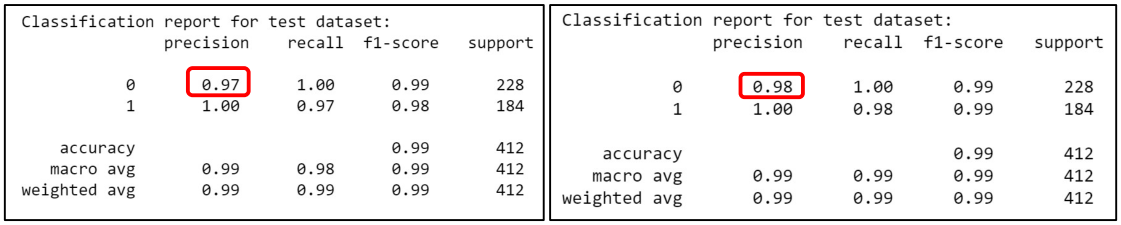
Model	Confusion matrix	
	Train dataset	Test dataset
Default parameters		
Optimized parameters		

Table 2. Confusion matrix comparison for model using default hyperparameters and optimized hyperparameters

Model using default hyperparameters

Model using optimized hyperparameters



```
Classification report for test dataset:
              precision    recall  f1-score   support

           0       0.97      1.00      0.99       228
           1       1.00      0.97      0.98       184

    accuracy                           0.99       412
   macro avg       0.99      0.98      0.99       412
weighted avg       0.99      0.99      0.99       412
```

```
Classification report for test dataset:
              precision    recall  f1-score   support

           0       0.98      1.00      0.99       228
           1       1.00      0.98      0.99       184

    accuracy                           0.99       412
   macro avg       0.99      0.99      0.99       412
weighted avg       0.99      0.99      0.99       412
```

Figure 6. Classification report comparison for model using default hyperparameters and optimized hyperparameters

The optimized model was further trained using L1, L2 (default regularization parameter used in the optimal model obtained) and elastic net regularization. From the results obtained, the model using elastic net regularization obtained the highest accuracy score (Table 3) without signs of overfitting. As previously mentioned, elastic net is a combination of L2 and L1 and is able to produce a model with good prediction accuracy.

Model with regularization	Accuracy score	
	Train dataset	Test dataset
l1	0.985	0.988
l2 (Optimized parameters model)	0.988	0.990
Elastic net	0.990	0.990

Table 3. Comparison of accuracy score for models with different regularization hyperparameter

From the confusion matrix for both test and train dataset, it is observed that elastic net regularization yields the model with the lowest false negative values (table 4). This improvement is also consistently observed in the classification report for the test dataset, where the precision of the negative class has increased slightly to 0.99 for the model using elastic net regularization compared to the other models (figure 7).

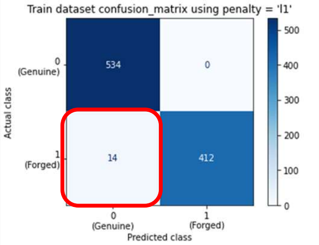
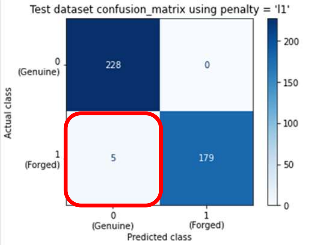
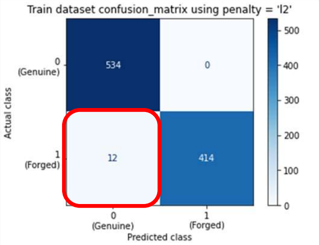
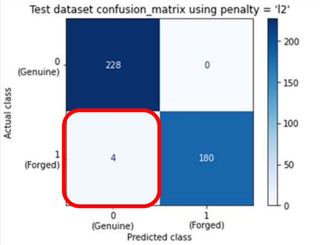
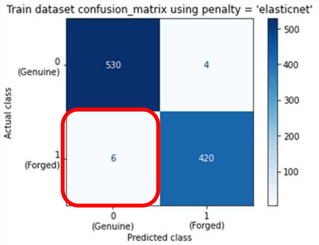
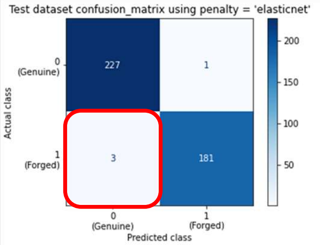
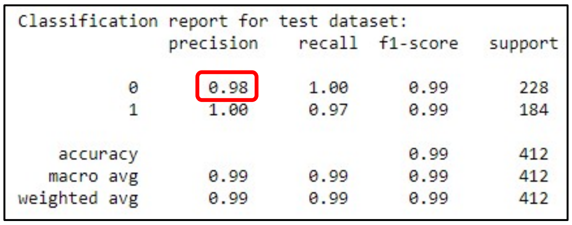
Model with regularization	Confusion matrix	
	Train dataset	Test dataset
l1		
l2 (Optimized parameters model)		
Elastic net		

Table 4. Confusion matrix comparison for models with different regularization hyperparameter

Model using l1 regularization

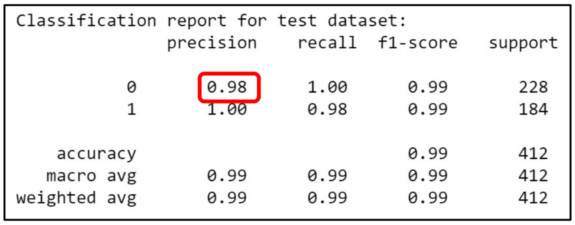


```
Classification report for test dataset:
              precision    recall  f1-score   support

           0       0.98      1.00      0.99       228
           1       1.00      0.97      0.99       184

    accuracy                           0.99       412
   macro avg       0.99      0.99      0.99       412
weighted avg       0.99      0.99      0.99       412
```

Model using l2 regularization (optimized parameters model)

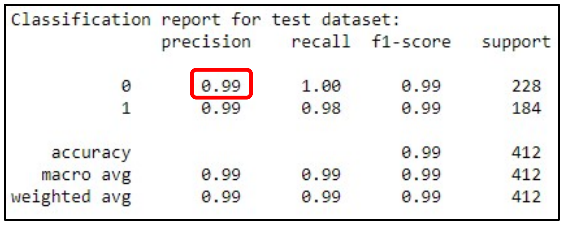


```
Classification report for test dataset:
              precision    recall  f1-score   support

           0       0.98      1.00      0.99       228
           1       1.00      0.98      0.99       184

    accuracy                           0.99       412
   macro avg       0.99      0.99      0.99       412
weighted avg       0.99      0.99      0.99       412
```

Model using elastic net regularization



```
Classification report for test dataset:
              precision    recall  f1-score   support

           0       0.99      1.00      0.99       228
           1       0.99      0.98      0.99       184

    accuracy                           0.99       412
   macro avg       0.99      0.99      0.99       412
weighted avg       0.99      0.99      0.99       412
```

Figure 7. Classification report comparison for models with different regularization hyperparameter

Lastly, a KNN model with default hyperparameters was designed using the same dataset as the logistic regression model. As seen by the results, the KNN model performs better than the logistic regression model with optimized hyperparameters in terms of accuracy score (Table 5).

Model	Accuracy score	
	Train dataset	Test dataset
Optimized parameters	0.988	0.990
KNN	0.999	0.998

Table 5. Comparison of accuracy score for optimized model and KNN model

The KNN model, however, greatly reduces the number of false negatives (Table 6) of both the train and test dataset (to zero false negatives) and improves the precision of the negative class (figure 8) from 0.98 to 1.00, without signs of overfitting. This could be specific to the current dataset as seen from figure 4 above, most of the data points with the same class are grouped near to each other. However, the KNN algorithm may not be suitable for noisy or large datasets.

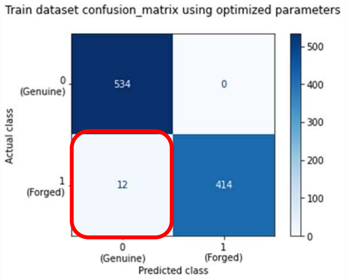
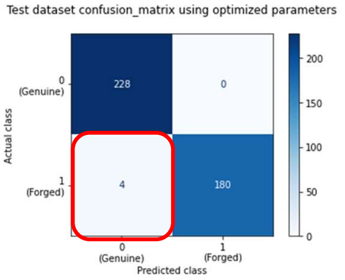
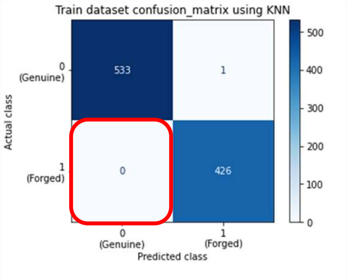
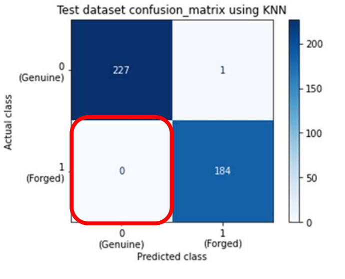
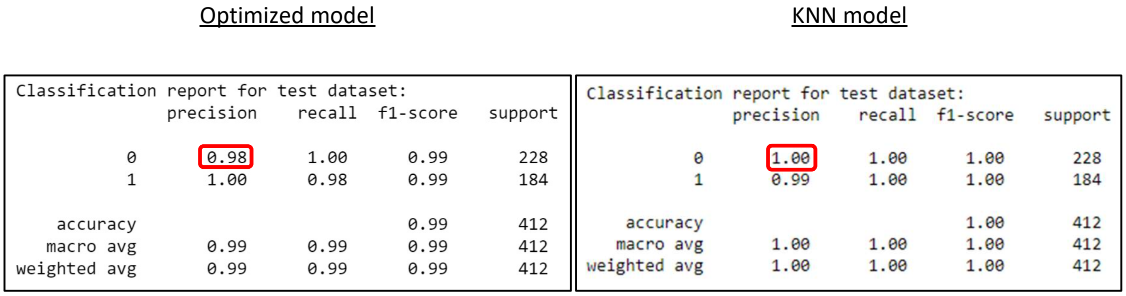
Model	Confusion matrix	
	Train dataset	Test dataset
Optimized parameters		
KNN		

Table 6. Confusion matrix comparison for optimized model and KNN model



Optimized model

```
Classification report for test dataset:
              precision    recall  f1-score   support

           0       0.98      1.00      0.99       228
           1       1.00      0.98      0.99       184

    accuracy                           0.99       412
   macro avg       0.99      0.99      0.99       412
weighted avg       0.99      0.99      0.99       412
```

KNN model

```
Classification report for test dataset:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       228
           1       0.99      1.00      1.00       184

    accuracy                           1.00       412
   macro avg       1.00      1.00      1.00       412
weighted avg       1.00      1.00      1.00       412
```

Figure 8. Classification report comparison for optimized model and KNN model

Conclusion

The objective of this report was to design a logistic regression algorithm that was able to classify bank notes as genuine or forged using the data bank note dataset. Based on the findings described in this report, it is observed that the logistic regression model is able to classify bank notes as genuine or forged with a high accuracy score of 99% through optimizing the hyperparameters of the logistic regression model. Furthermore, the optimized model is also able to reduce false negative values which would be detrimental for this use case. However, it is also noted that a KNN algorithm would yield the best result based on the current dataset, reducing the false negative predictions to its minimum.

References

2U, Inc. (n.d.). *What Is Logistic Regression?* Retrieved from Masters in Data Science: https://www.mastersindatascience.org/learning/introduction-to-machine-learning-algorithms/logistic-regression/

Boston University School of Public Health. (17 January, 2013). *Multiple Linear Regression Analysis*. Retrieved from https://sphweb.bumc.bu.edu/otlt/mph-modules/bs/bs704_multivariable/bs704_multivariable7.html

Hastie, H. Z. (2005). Regularization and variable selection via the. *J. R. Statist*, 301-320.

Jain, S. (22 June, 2017). *A comprehensive beginners guide for Linear, Ridge and Lasso Regression in Python and R*. Retrieved from Analytics Vidhya: https://www.analyticsvidhya.com/blog/2017/06/a-comprehensive-guide-for-linear-ridge-and-lasso-regression/

Koehrsen, W. (9 February, 2022). *When Accuracy Isn't Enough, Use Precision and Recall to Evaluate Your Classification Model*. Retrieved from Built In: https://builtin.com/data-science/precision-and-recall

Miller, L. (11 January, 2018). *Machine Learning week 1: Cost Function, Gradient Descent and Univariate Linear Regression*. Retrieved from Medium: https://medium.com/@lachlanmiller_52885/machine-learning-week-1-cost-function-gradient-descent-and-univariate-linear-regression-8f5fe69815fd

Qshick. (3 January, 2019). *Ridge Regression for Better Usage*. Retrieved from Towards Data Science: https://towardsdatascience.com/ridge-regression-for-better-usage-2f19b3a202db

scikit-learn developers. (n.d.). *1.5. Stochastic Gradient Descent*. Retrieved from Scikit learn: https://scikit-learn.org/stable/modules/sgd.html#:~:text=Stochastic%20Gradient%20Descent%20(SGD)%20is,Vector%20Machines%20and%20Logistic%20Regression.

scikit-learn developers. (n.d.). *sklearn.linear_model.SGDClassifier*. Retrieved from Scikit Learn: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.SGDClassifier.html

Varghese, D. (7 December, 2018). *Comparative Study on Classic Machine learning Algorithms*. Retrieved from Towards Data Science: https://towardsdatascience.com/comparative-study-on-classic-machine-learning-algorithms-24f9ff6ab222