Student: Ellen McGrory

Logistic regression in healthcare applications

1.0) Introduction

In statistics, datasets can be leveraged to predict outcomes based on a series of input variables which can be done using linear regression type models or decision tree methods (and more). For this project, the machine learning technique of logistic regression will be discussed. While this machine learning technique was not taught as part of this module, I wanted to learn how to implement and evaluate it, in addition to understanding how logistic regression works.

2.0) Literature review

Machine learning techniques can be generally divided into either supervised or unsupervised methods. Unsupervised learning techniques mainly involve clustering and regression such as principal component analysis (PCA) for data dimensionality reduction (Géron, 2017). These unsupervised techniques do not require labeled data so that the model output is predicted or classified based on patterns on the input data.

In comparison, supervised techniques require required the data to be labelled in addition to being split into training and testing datasets. Supervised learning techniques mainly deal with classification (training a model to predict what animal is in a photo) or regression (predicting the price of a house based on number of rooms, neighborhood, area, etc.) (Field et al., 2012). However, in the case of supervised classification algorithms, they cannot predict an animal if it was not present during model training. Data is split so that the model can be evaluated on data it has not seen before (i.e., testing data). After a model has been fitted a confusion matrix can be generated to evaluate the model in addition to other metrics (Géron, 2017).

A commonly supervised regression technique is linear regression where an independent variable (or explanatory) is used to estimate dependent variable (i.e., hours of study to estimate grades). This analogy can be extended to multiple independent variables are used to estimate a dependent variable in multiple linear regression (MLS) (i.e., hours of study, extracurricular activity, number of days sick used to predict grades). The output of MLS is a continuous value (i.e., price of house or grades), in comparison, logistic regression returns the probability of a binary outcome (for example is an email spam or not) (Field et al., 2012).

Logistic regression (or logit regression) is a supervised classification algorithm (despite regression in the name) that is used to estimate the probability that an occurrence belongs to a classification (e.g. spam folder of email) where a threshold probability of >50% the model will predict that the occurrence belongs to the positive class (denoted as 0, i.e. normal email) or else predicts that the classification belongs to the negative class (denoted as 1, i.e. spam) (Géron, 2017). The logistic function (denoted as σ) is a sigmoid function with the output bound between 0 and 1 and defined by Equation 1 which is subsequently illustrated in Fig. 1 (Géron, 2017).

$$\sigma(X) = \frac{1}{1 + \ exp^{-x}} \quad Equation\ 1$$

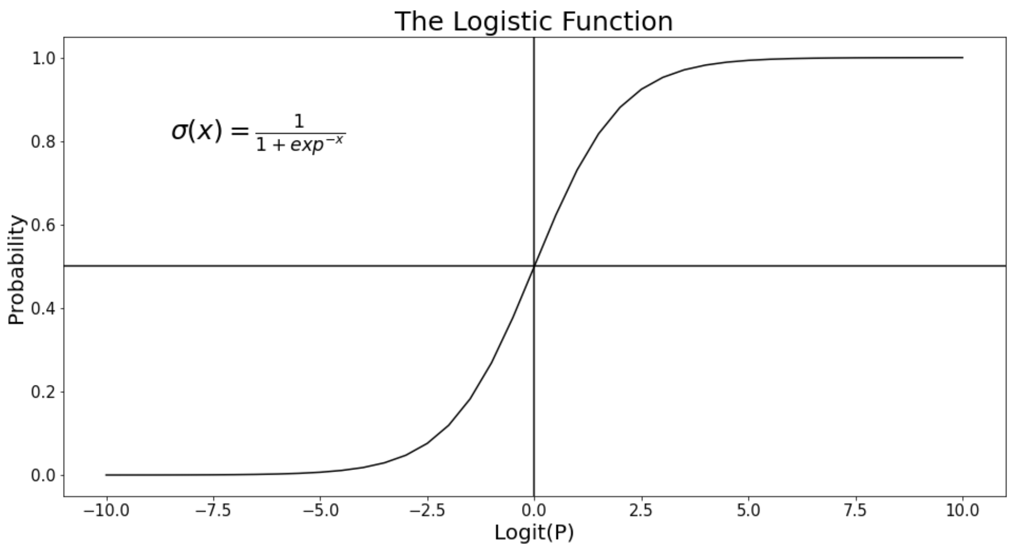


Fig. 1. The logistic function (modelled in Python)

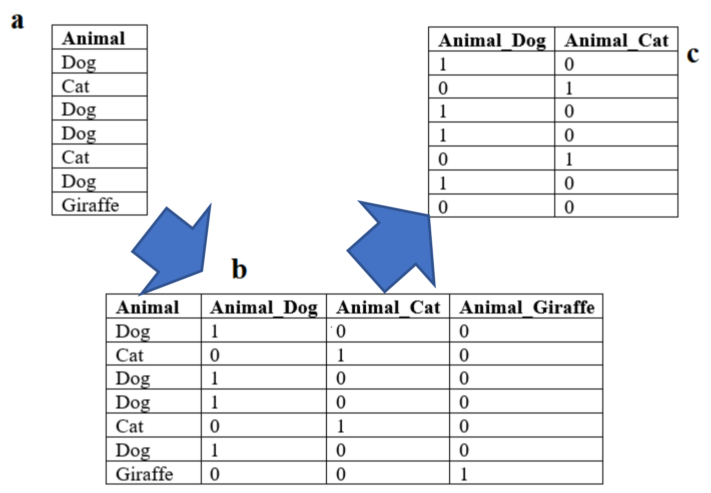
Logistic regression finds many applications, such as in healthcare settings to predict the likelihood that a cancer is benign or cancerous from nth variables (Field et al., 2012). A database of patients with several variables is split into two with the training dataset being used to train the model while the test dataset is used for model evaluation. These variables can be measured for a new patient to predict if cancer is benign or cancerous. While this type of logistic regression is known as binary logistic regression, it can also be used to predict membership to more than two categories (i.e., is a person married, single or divorced) then multinomial logistic regression can be used (Field et al., 2012).

In order to use logistic regression, there are several general assumptions of the model (Field et al. 2012; Midi et al., 2010);

- The true conditional probabilities are a logistic function of the independent variables.
- Independent variables are measured without error.
- Observations are independent.

- Errors are binomially distributed.
- Independent variables are not linear combinations of each other.
- No extraneous variables are included, and no important variables are omitted.

While a lot of data include nominal type data, logistic regression cannot use these variables for modelling (including linear regression too). To overcome this, dummy variable can be computed. For example, a dataset has a nominal data column (Fig. 2.a) of different animals. Dummy variables are computed where three additional columns are created with the different categorical data types where presence (1) or absence (0) is denoted. If animal was Dog then Animal_Dog would be demarcated as 1 (Fig. 2b). One of the dummy variables of Animal can be removed (Fig. 2c) so that perfect multicollinearity can be avoided. Since Giraffe is removed, this is the base category which other dummy variables are compared to. Multicollinearity can also affect the estimate of the dependent variables if the correlation coefficient of several independent variables is high (>0.8) and effects the interpretation of independent coefficients of the logistic regression model (Midi et al., 2010).



a

Animal
Dog
Cat
Dog
Dog
Cat
Dog
Giraffe

b

Animal	Animal_Dog	Animal_Cat	Animal_Giraffe
Dog	1	0	0
Cat	0	1	0
Dog	1	0	0
Dog	1	0	0
Cat	0	1	0
Dog	1	0	0
Giraffe	0	0	1

c

Animal_Dog	Animal_Cat
1	0
0	1
1	0
1	0
0	1
1	0
0	0

Fig. 2. Dummy variables of nominal data

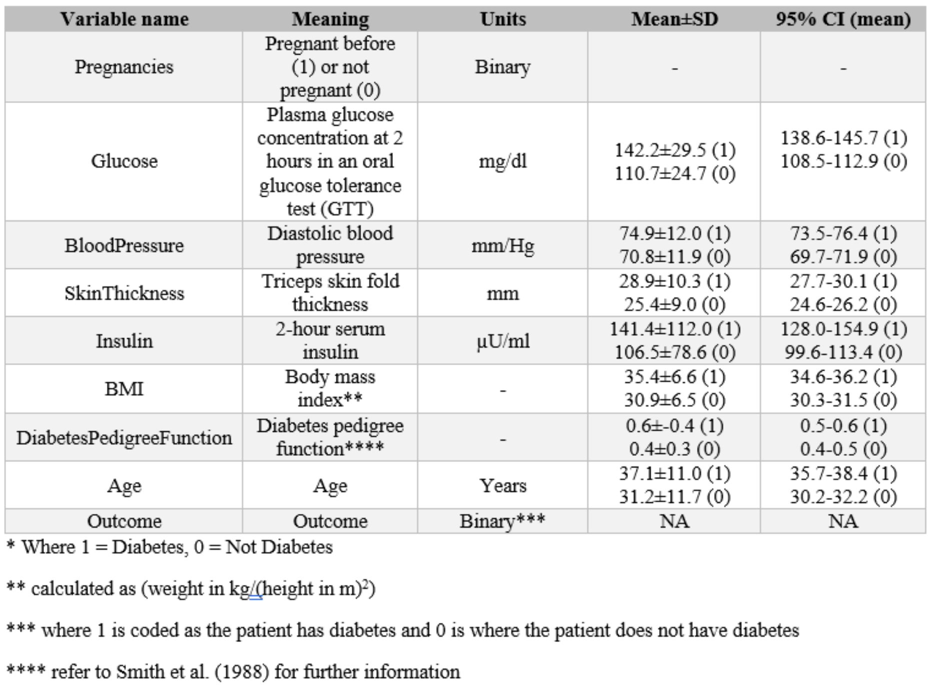
Each algorithm has default parameters when used as part of SkiKit-Learn python package, but these parameters or hyper parameters of logistic regression can be fine-tuned to improve model metrics and prediction (Géron, 2017).

3.0) Logistic regression example – the dataset and preprocessing

For this section, a binary logistic regression model will be implemented using Python in Jupyter Notebooks with an example of predicting if a patient has diabetes mellitus using binary logistic regression. Diabetes mellitus is a series of metabolic disorders where there is hyperglycemia (i.e., high blood glucose levels) which can be broadly divided into type 1 diabetes (patients cannot produce insulin, mainly affecting juveniles), type 2 diabetes (patients have insulin resistance) and gestational diabetes (female patients who while pregnant can be less susceptive to insulin) (Tigga and Garg, 2021). Type 2 diabetes is the most prevalent form of diabetes.

The dataset used was the Pima Indian Diabetes dataset from Machine Learning Repository (Dua and Graff, 2019) (originally from National Institute of Diabetes and Digestive and Kidney Disease) (Smith et al., 1988) which contains 8 medical diagnostic attributes and one target variable (i.e, Outcome) of 768 female patients with 34.9% having diabetes (268 patients) (Table 1). The variance for insulin for both categories was quite high. An independent t-test was used to compare the values between diabetic and non-diabetic patients showing that differences occurred for all 8 independent variables. For example, there is evidence to suggest that the average blood glucose concentration for diabetics mean=142.2 mg/dl (95% CI: 138.6, 145.7) was higher than non-diabetics, mean=110.7 mg/dl (95% CI: 108.5, 112.9); t(766)= 15.67, $p < 0.001$ at the 95% confidence level. Before an algorithm like logistic regression can be implemented data quality must be checked.

Table 1. Diabetes dataset features



Variable name	Meaning	Units	Mean±SD	95% CI (mean)
Pregnancies	Pregnant before (1) or not pregnant (0)	Binary	-	-
Glucose	Plasma glucose concentration at 2 hours in an oral glucose tolerance test (GTT)	mg/dl	142.2±29.5 (1) 110.7±24.7 (0)	138.6-145.7 (1) 108.5-112.9 (0)
BloodPressure	Diastolic blood pressure	mm/Hg	74.9±12.0 (1) 70.8±11.9 (0)	73.5-76.4 (1) 69.7-71.9 (0)
SkinThickness	Triceps skin fold thickness	mm	28.9±10.3 (1) 25.4±9.0 (0)	27.7-30.1 (1) 24.6-26.2 (0)
Insulin	2-hour serum insulin	µU/ml	141.4±112.0 (1) 106.5±78.6 (0)	128.0-154.9 (1) 99.6-113.4 (0)
BMI	Body mass index**	-	35.4±6.6 (1) 30.9±6.5 (0)	34.6-36.2 (1) 30.3-31.5 (0)
DiabetesPedigreeFunction	Diabetes pedigree function****	-	0.6±-0.4 (1) 0.4±0.3 (0)	0.5-0.6 (1) 0.4-0.5 (0)
Age	Age	Years	37.1±11.0 (1) 31.2±11.7 (0)	35.7-38.4 (1) 30.2-32.2 (0)
Outcome	Outcome	Binary***	NA	NA

* Where 1 = Diabetes, 0 = Not Diabetes

** calculated as (weight in kg/(height in m)^2)

*** where 1 is coded as the patient has diabetes and 0 is where the patient does not have diabetes

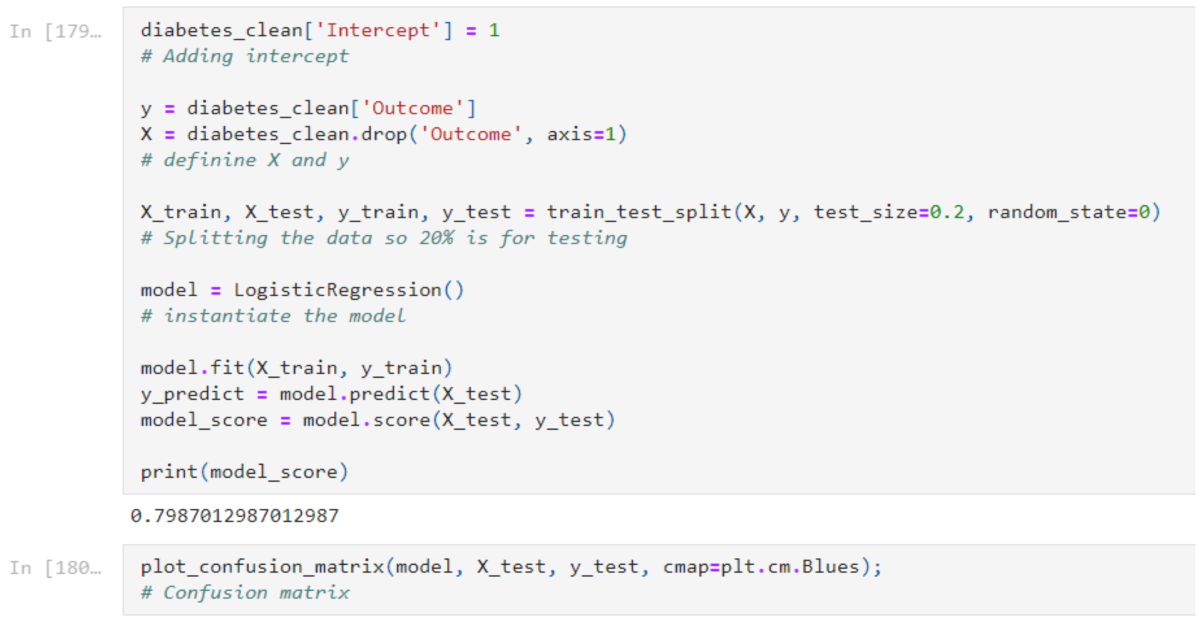
**** refer to Smith et al. (1988) for further information

While no values were missing on initial inspection, it was discovered that several variables had values of 0, which are not relevant clinical values. For example, a BMI of 0 would not be a valid number. Five attributes had these types of missing data which ranged from 0.65% (Glucose) to 48.7% (Insulin). Whilst this is a violation of an assumption of logistic regression, this can be overcome here. While there are methods to deal with missing data, imputation using the mean value was implemented here. To reduce the complexity of the analysis, as noted by other studies (Wu et al., 2018), the Pregnancies column was transformed into either never pregnant (as 0) or has been pregnant before (1).

As all the independent values are continuous or discrete data and no character data is present, dummy variables do not need to be compued. A heatmap illustrating Pearson correlation coefficients of the independent variables showed that the highest correlations were between BMI and SkinThickness (0.536), and Glucose and Insulin (0.397) indicating whilst correlation exists between these variables, multicollinearity was not an issue.

4.0) Logistic regression example – model implementation

In Python, logistic regression was coded using SkiKit-Learn package (Fig. 3). The independent variables (X) are used to predict outcome (y) with the dataset being split into testing and training datasets using the *train_test_split()* function with an 80:20 ratio for training and testing respectively (Géron, 2017). The model was fitted using the *.fit()* function. A confusion matrix of the logistic regression model was returned. The confusion matrix is a N*N matrix used for evaluating the performance of a classification algorithm. The matrix contains the actual values (0 and 1) with the predicted values (0 and 1) using the test dataset, in this case a 2*2 matrix. For example, this model predicted 27 patients had diabetes who had diabetes (i.e., true positives TP) while the model successfully predicted 96 patients who did not have diabetes (i.e., true negatives TN). In comparison, this model predicted 11 patients who have diabetes when they do not have diabetes (i.e., false positive FP) while the model predicted 20 patients do not have diabetes, but they actually do (i.e., false negative FN). A perfect classification algorithm would only have TP and TN values (Géron, 2017). Using the confusion matrix (Fig. 3) several of the metrics of the model can be calculated, such as accuracy, precision, recall and F_1 score (Equation 2-5).



```
In [179…  diabetes_clean['Intercept'] = 1
          # Adding intercept

          y = diabetes_clean['Outcome']
          X = diabetes_clean.drop('Outcome', axis=1)
          # definine X and y

          X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
          # Splitting the data so 20% is for testing

          model = LogisticRegression()
          # instantiate the model

          model.fit(X_train, y_train)
          y_predict = model.predict(X_test)
          model_score = model.score(X_test, y_test)

          print(model_score)

          0.7987012987012987

In [180…  plot_confusion_matrix(model, X_test, y_test, cmap=plt.cm.Blues);
          # Confusion matrix
```

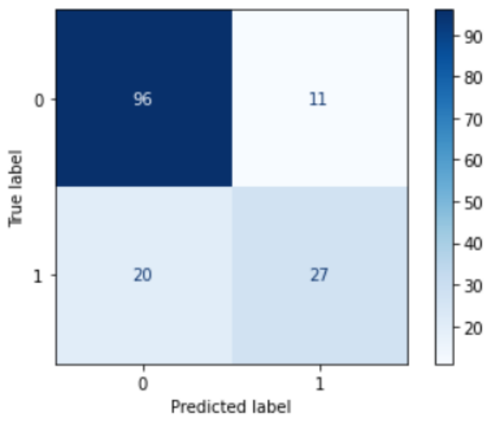


Fig. 3. Python code for logistic regression model (note: the intercept is the expected mean of y when X=0).

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} = \frac{27 + 96}{27 + 11 + 96 + 20} = 79.9\% \quad Equation\ 2$$

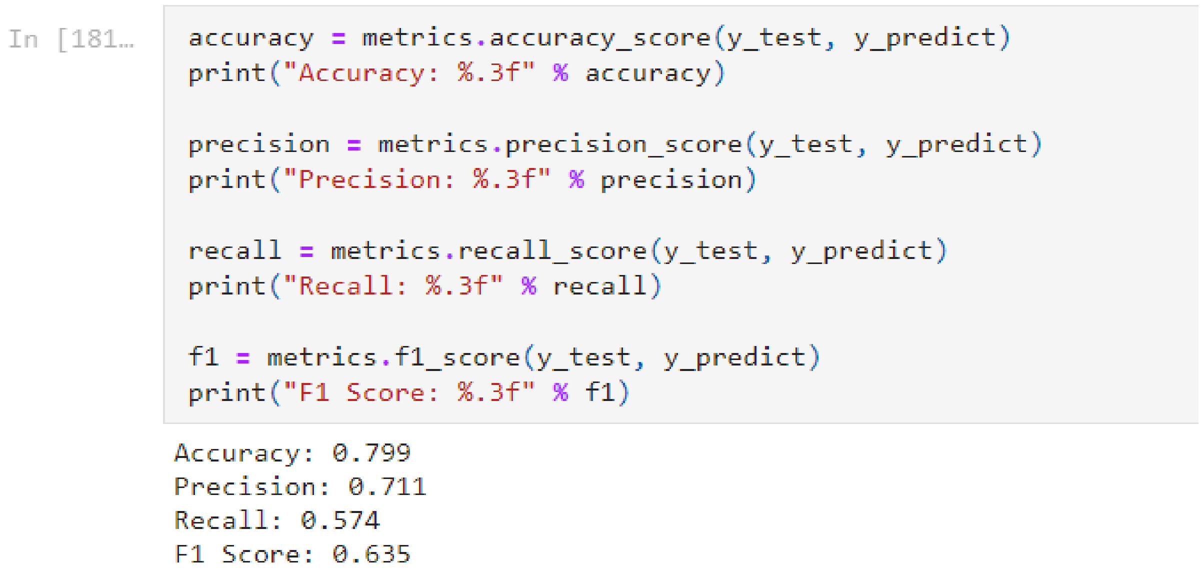
$$Precision = \frac{TP}{TP + FP} = \frac{27}{27 + 11} = 71.1\% \quad Equation\ 3$$

$$Recall = \frac{TP}{TP + FN} = \frac{27}{27 + 20} = 57.4\% \quad Equation\ 4$$

$$F_1 Score = \frac{2}{\frac{1}{Recall} + \frac{1}{Precision}} = \frac{2}{\frac{1}{57.4} + \frac{1}{71.1}} = 63.5\% \quad Equation\ 5$$

The accuracy of the model shows the successfully predicted true positives and negatives as a function of total predictions at 79.9% (the corresponding error rate is 21.1%) (Fig. 4). While this is useful, it does not give good information in relation to the ability of the model to successfully predict values. For this, precision and recall are used. Precision shows the accuracy of the positive predictions while recall is the ratio of positive instances that are correctly identified (Géron, 2017). The F_1 score is a combination of precision and recall with a classifier having similar precision and recall values having a high F_1 score. In some circumstances precision or recall would be more valued in an application. However, an increase in recall reduces precision (and vice versa) known as the precision/recall tradeoff (Géron, 2017).

A final metric used for binary logistic regression such as this example is the receiver operating characteristic (ROC) curve which plots the true positive rate (TPR or recall) as a function of the false positive rate (FPR) (Géron, 2017). Using this graph, the area under the curve (AUC) value can be calculated as a tool to compare different classification algorithms with a random classifier having a ROC AUC value of 0.5. Better performing algorithms will have a higher ROC AUC value > 0.5. This model had an ROC AUC value of 0.834 (Fig. 5).


```
In [181…

accuracy = metrics.accuracy_score(y_test, y_predict)
print("Accuracy: %.3f" % accuracy)

precision = metrics.precision_score(y_test, y_predict)
print("Precision: %.3f" % precision)

recall = metrics.recall_score(y_test, y_predict)
print("Recall: %.3f" % recall)

f1 = metrics.f1_score(y_test, y_predict)
print("F1 Score: %.3f" % f1)
```

```
Accuracy: 0.799
Precision: 0.711
Recall: 0.574
F1 Score: 0.635
```

Fig. 4. Python code for logistic regression model metrics

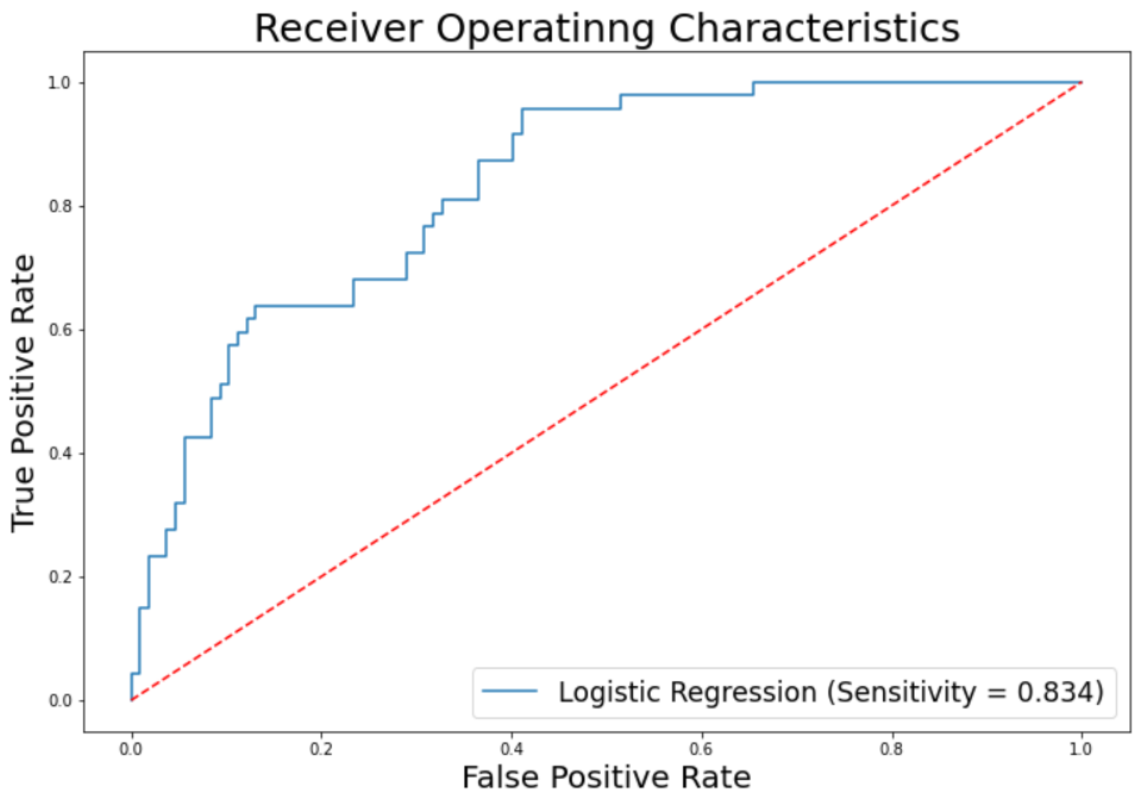


Fig. 5. Python code for logistic regression ROC curve

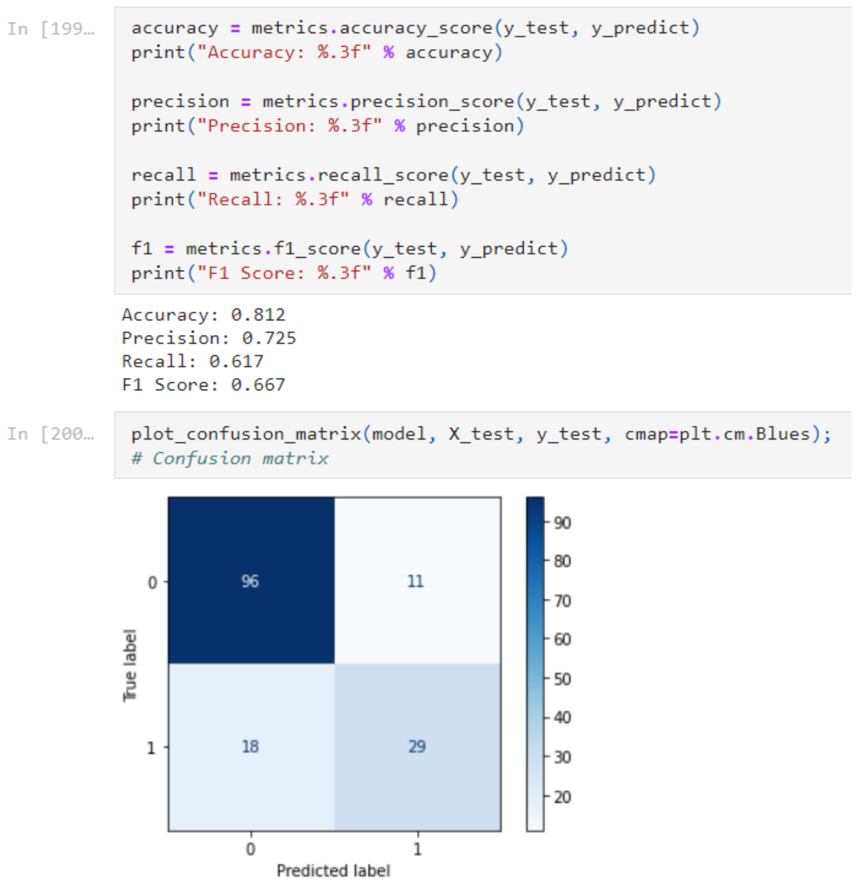
5.0) Hyperparameter tuning

Based on the previous model it was showed based on coefficient values that the independent variables of DiabetesPedigreeFunction, BMI, Glucose and Age contributed significantly to the model. As a result, these independent variables were used for this example of hyperparameter tuning.

Since this is a medical based application, the focus will be on improving the recall of the model using hyperparameter optimization. Recall is important in this application as patients are predicted as not having diabetes when they have diabetes has important clinical implications.

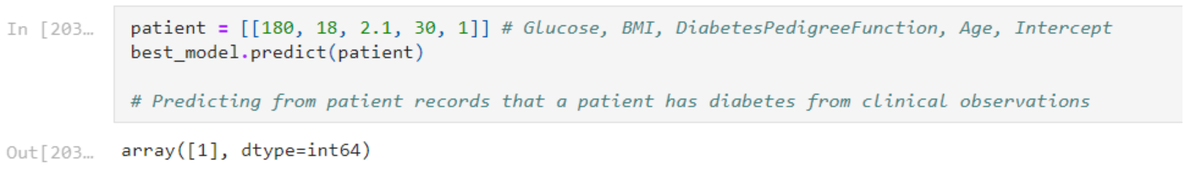
A common way to fine tune a model is to adjust the hyperparameters using the *GridSearchCV()* function of Scikit-Learn to find a good combination of parameters for the dataset under investigation.

As can be seen (Fig. 6) by optimizing for hyperparameters (in this instances 'penalty', 'C', and 'max_iter' of the *LogisticRegression()* function) the recall score has improved. One less patient is now in the FN segment of the confusion matrix. In addition, the ROC AUC value has now increased to 0.861. Finally, this model can be used to predict if a new patient has diabetes (Fig. 7).



```
In [199…  accuracy = metrics.accuracy_score(y_test, y_predict)
          print("Accuracy: %.3f" % accuracy)

          precision = metrics.precision_score(y_test, y_predict)
          print("Precision: %.3f" % precision)

          recall = metrics.recall_score(y_test, y_predict)
          print("Recall: %.3f" % recall)

          f1 = metrics.f1_score(y_test, y_predict)
          print("F1 Score: %.3f" % f1)

          Accuracy: 0.812
          Precision: 0.725
          Recall: 0.617
          F1 Score: 0.667

In [200…  plot_confusion_matrix(model, X_test, y_test, cmap=plt.cm.Blues);
          # Confusion matrix
```

Fig. 6. Logisitic regression model metrics using hyperparameer optizmising



```
In [203…  patient = [[180, 18, 2.1, 30, 1]] # Glucose, BMI, DiabetesPedigreeFunction, Age, Intercept
          best_model.predict(patient)

          # Predicting from patient records that a patient has diabetes from clinical observations

Out[203…  array([1], dtype=int64)
```

Fig. 7. Predicting that a new patient has diabetes using the trained model

6.0) Conclusions

Binary logistic regression can be used to predict a medical outcome. By optimizing the hyperparameters, the model could be improved by increasing the recall (i.e., decrasing FNs). While this model can be used to predict if a patient has diabetes, this model is not appropriate to predict diabetes in patients with different demographics to the demographics used in the original dataset.

7.0) References

Dua, D. and Graff, C., 2019. UCI Machine Learning Repository Irvine, CA: University of California, School of Information and Computer Science. http://archive.ics.uci.edu/ml [Date Assessed: 29/03/2021].

Field, A., Mills, J., Field, Z., 2012. Discovering Statistics using R. Sage, Los Angeles, pp. 958.

Géron, A., 2017. Hands-on Machine Learning with Scikit-Learn & TensorFlow. O'Reilly, Bejing, pp. 549.

Midi, H., Sarkar, S.K., Rana, S., 2010. Collinearity diagnostics of binary logistic regression model. *Journal of Interdisciplinary Mathematics* **13**, 253-267.

Smith, J.W., Everhart, J.E., Dickson, W.C., Knowler, W.C., Johannes, R.S., 1988. Using the ADAP learning algorithm to forecast the onset of diabetes mellitus. In: Proceedings of the Symposium on Computer Applications and Medical Care. IEEE Computer Society Press, pp. 261-265.

Tigga, N.P., Garg, S., 2021. Predicting type 2 diabetes using logistic regression. In: Proceedings of the Fourth International Conference on Microelectronics, Computing and Communication Systems (Nath V., Mandal J.K., Eds). Springer, Singapore, pp. 491-500.

Wu, H., Yang, S., Huang, Z., He, J., Wang, X., 2018. Type 2 diabetes mellitus prediction model based on data mining. *Informatics in Medicine Unlocked* **10**, 100-107.