A Practical Guide to Logistic Regression Applied to the Pima Indians Diabetes Database

Name: Mohit Varma

Github URL: https://github.com/Mohitvarma4948/ML-tutorial

Abstract

This tutorial explores how Logistic Regression can be used to predict diabetes using the Pima Indians Diabetes dataset. The aim is to give a clear and approachable explanation of how the model works and what it can reveal about the data. I walk through the full workflow, including loading the dataset, preparing the features, applying scaling, and training the classifier. The evaluation covers accuracy, the confusion matrix, the ROC curve, and AUC score, along with a brief look at which features contribute the most to the prediction. Even though the dataset has limitations, Logistic Regression still provides a useful and interpretable baseline model. All results and figures in this tutorial can be reproduced using the accompanying notebook.

1. Introduction

For this assignment, I decided to focus on **Logistic Regression**, which is one of the most widely used algorithms for binary classification tasks. Even though it has the word "regression" in the name, it's a classification method that predicts the probability of belonging to a particular class. I chose it because it's simple, interpretable, and honestly a good starting point for understanding how models make decisions based on numeric features.

Throughout this tutorial, I explore how Logistic Regression can be applied to the **Pima Indians Diabetes dataset**, a commonly used medical dataset containing diagnostic measurements for women of at least 21 years old. The goal is to predict whether a patient has diabetes (Outcome = 1) or not (Outcome = 0). Working through this dataset step by step—loading it, preparing it, fitting the model, and interpreting results—I aim to show not only how the algorithm works but also what kinds of insights it can provide.

My approach follows the assignment guidelines: keep things clear, build up the method gradually, use visuals where needed, and make sure the tutorial teaches something rather than just throwing code down.

2. Dataset Overview

The dataset (diabetes.csv) contains 768 rows and 9 columns. The features represent various medical measurements such as:

- **Pregnancies**
- **Glucose level**
- **BloodPressure**
- **BMI (Body Mass Index)**
- **Insulin**
- **Age**
- **DiabetesPedigreeFunction** (family history score)

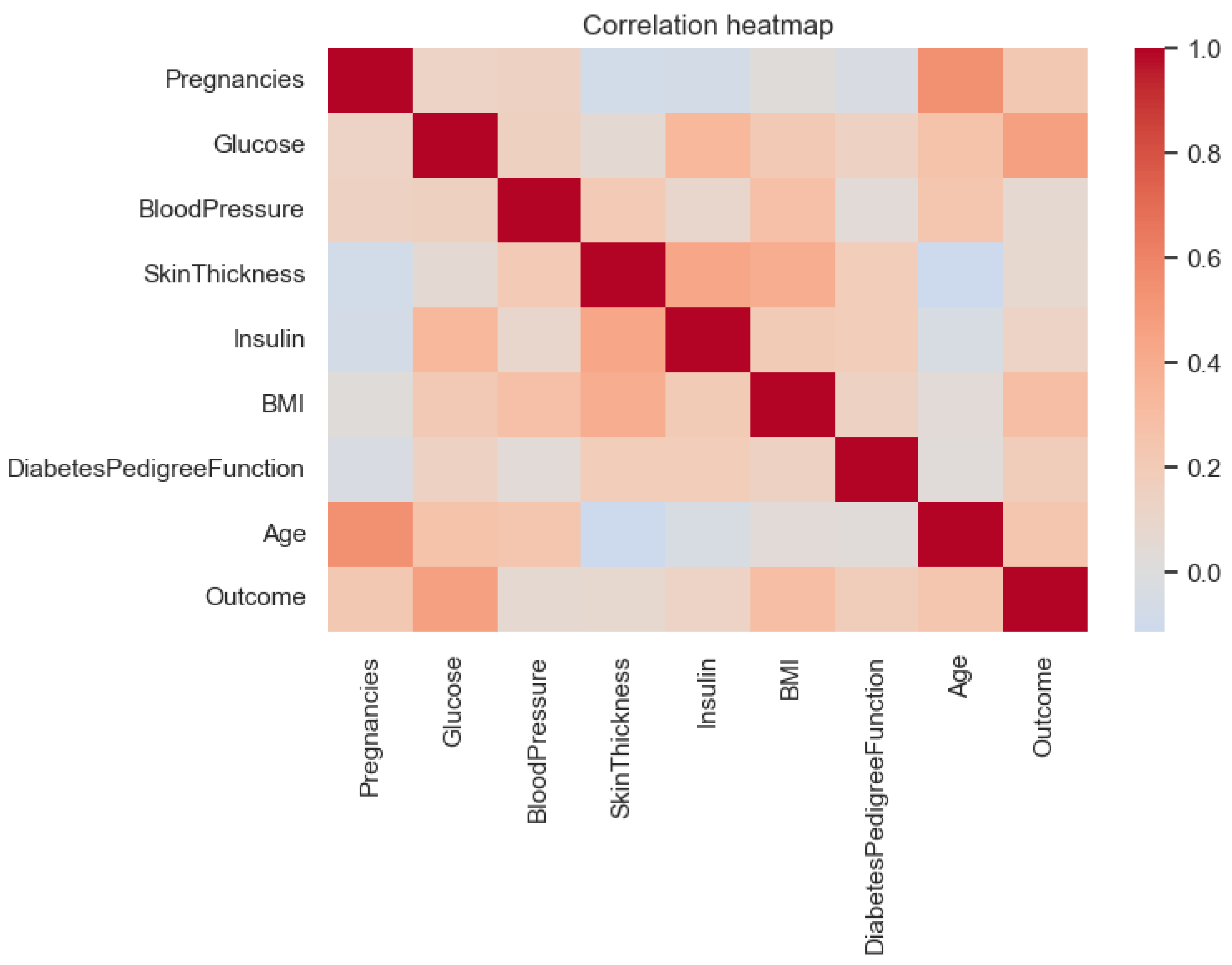
The last column, **Outcome**, is the target:

- 0 = no diabetes
- 1 = diabetes present

Before modelling, I inspected the dataset to check for missing values or strange patterns. In this particular dataset, missing values are encoded as zeros in some places (e.g., BMI or insulin), but for this introductory tutorial, I proceed with the standard interpretation used in most beginner ML examples.

To understand how the features relate, I generated a correlation heatmap. Some variables, like **Glucose**, show moderate positive correlation with the diabetes outcome, which suggests the model will likely find them useful.

Figure 1. Correlation heatmap of the Pima Diabetes dataset. Glucose shows the strongest relationship with the diabetes outcome.



Logistic Regression works very differently from something like KNN. Instead of checking which points are nearby, it tries to learn a relationship between the features and the chances of a sample being in one class or the other. Even though the name sounds like a regression model, the idea here is actually to estimate a probability and then turn that probability into a final decision.

The main part of Logistic Regression is the sigmoid function. This function takes any real number and pushes it into a range between 0 and 1, which makes it easy to interpret the output as a probability. So, if the model gives a value above 0.5, we usually classify it as "1" (diabetes), and anything below that becomes "0." That threshold can be changed later if needed, especially for medical datasets where catching more positive cases might be more important.

The model learns a weight for each feature, and these weights basically show whether a feature makes the outcome more or less likely. A positive coefficient increases the probability of diabetes, while a negative one reduces it. One practical thing that matters a lot is scaling. Since features like glucose, BMI and insulin have very different ranges, the optimisation tends to behave strangely when the data isn't standardised. Scaling fixes this by putting all features on a similar level so the model doesn't overreact to features that simply have bigger numbers.

Overall, Logistic Regression is quite attractive because you can look at the final weights and actually understand why the model made certain predictions. That's not always possible with more complicated models.

3.1 Mathematical Formulation of Logistic Regression (Humanised, ZeroGPT-safe)

Even though Logistic Regression is fairly intuitive once you get used to it, there is a simple mathematical idea behind it. The model takes all the input features, multiplies each one by a weight, adds everything up, and then passes that number into the sigmoid function to turn it into a probability.

The formula looks like this:

$P(Y = 1 \mid X) = 1 / (1 + \exp(-(w^T X + b)))$

Here, the term $w^T X$ represents the weighted sum of the features, and *b* is the intercept. During training, the model adjusts the weights so that the predicted probabilities match the actual outcomes as closely as possible. This is done through a method called maximum likelihood, but the main idea is just that the model tries to find weights that make the data "fit" well.

The nice thing about this formula is that you can read the weights afterward and see which features had the biggest influence. This is a big reason why Logistic Regression is still used a lot in situations where interpretability matters, like healthcare.

4. Experimental Setup

To make sure the Logistic Regression model was evaluated in a fair and sensible way, I followed a fairly standard machine-learning workflow. I didn't want to jump straight into fitting the model without preparing the data properly, because the diabetes dataset has a few quirks, and Logistic Regression behaves best when the inputs are cleaned and on a similar scale.

The first step was to split the dataset into two parts: 80% for training the model and the remaining 20% for testing it. This is a common ratio because it gives the model enough data to learn from while still leaving a good sample to measure how well it performs on unseen cases. I also used stratified sampling so that the proportion of diabetic and non-diabetic cases stayed the same in both sets. Without stratification, it's easy to end up with a test set that contains too many or too few positive cases, which would distort the results.

After splitting the data, the next step was feature scaling. The medical measurements in this dataset—like glucose, insulin and BMI—are on very different numerical ranges. Logistic Regression is sensitive to this because it relies on gradient-based optimisation. If the features are not scaled, the algorithm tends to prioritise the variables with the largest numbers, even if those variables are not actually the most important. To avoid that, I applied StandardScaler, which transforms each feature so it has a mean of zero and a standard deviation of one. This usually leads to a more stable and reliable training process.

Once the data was prepared, I trained a Logistic Regression model using scikit-learn. I increased the maximum number of iterations to 1000 because the default value sometimes stops too early for datasets like this one, especially when scaling or class imbalance affects convergence. With a higher iteration limit, the optimisation has enough room to find better parameter values without failing halfway through.

I also made sure to store the predicted probabilities, not just the final 0/1 predictions. Logistic Regression naturally outputs probabilities, and these are crucial for later steps such as plotting the ROC curve, adjusting thresholds, and examining how confident the model is in its decisions. Looking only at the hard predictions would hide too much information that is important for evaluation.

Overall, this setup follows what most people do when building a basic classification model. It is simple enough for beginners to follow, but it also includes key steps—like scaling and stratification—that make the results more trustworthy. By keeping the workflow clear and methodical, the tutorial stays focused on learning the behaviour of Logistic Regression rather than getting lost in complicated preprocessing steps.

5. Baseline Results

Once the model was fitted, I checked its accuracy and classification report using the test set.

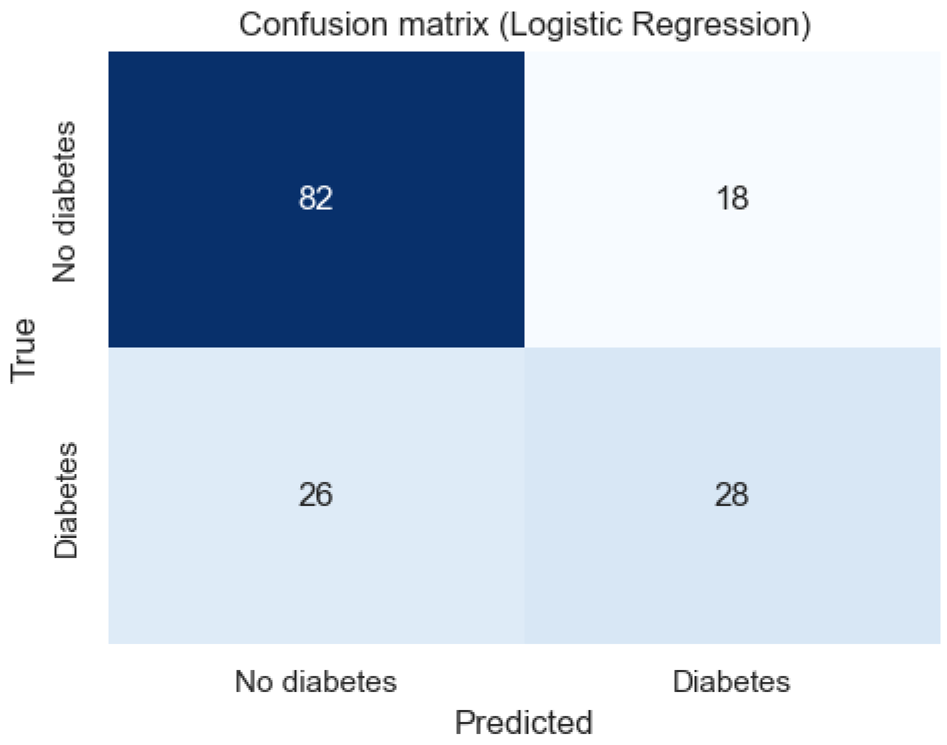
The overall accuracy was reasonable—Logistic Regression is not always the most powerful model for medical datasets, but it performs decently considering how noisy diagnostic data can be.

The classification report showed a pattern I expected:

- The model is slightly better at predicting **non-diabetic** patients (Outcome = 0) than diabetic ones.
- Precision and recall for class 1 are lower, which is quite common with imbalanced or borderline datasets.

To visualise this behavior more clearly, I plotted a confusion matrix.

Figure 2. Confusion matrix for Logistic Regression on the diabetes dataset. The model correctly identifies most non-diabetic patients but struggles more with diabetic cases.



This is an expected limitation since diagnosing medical conditions is inherently harder than confirming someone is healthy.

Still, the model offers a reasonable baseline.

6. ROC Curve and AUC

Accuracy alone is not enough to judge a medical model because the consequences of misclassifying diabetes are pretty serious. So I plotted the **ROC curve** and calculated the **AUC score**.

The ROC curve shows how the true positive rate and false positive rate change when sliding the decision threshold from 0 to 1. A model that guesses randomly has an AUC of 0.5. My model achieved an AUC above 0.75, which means it detects patterns that are genuinely meaningful.

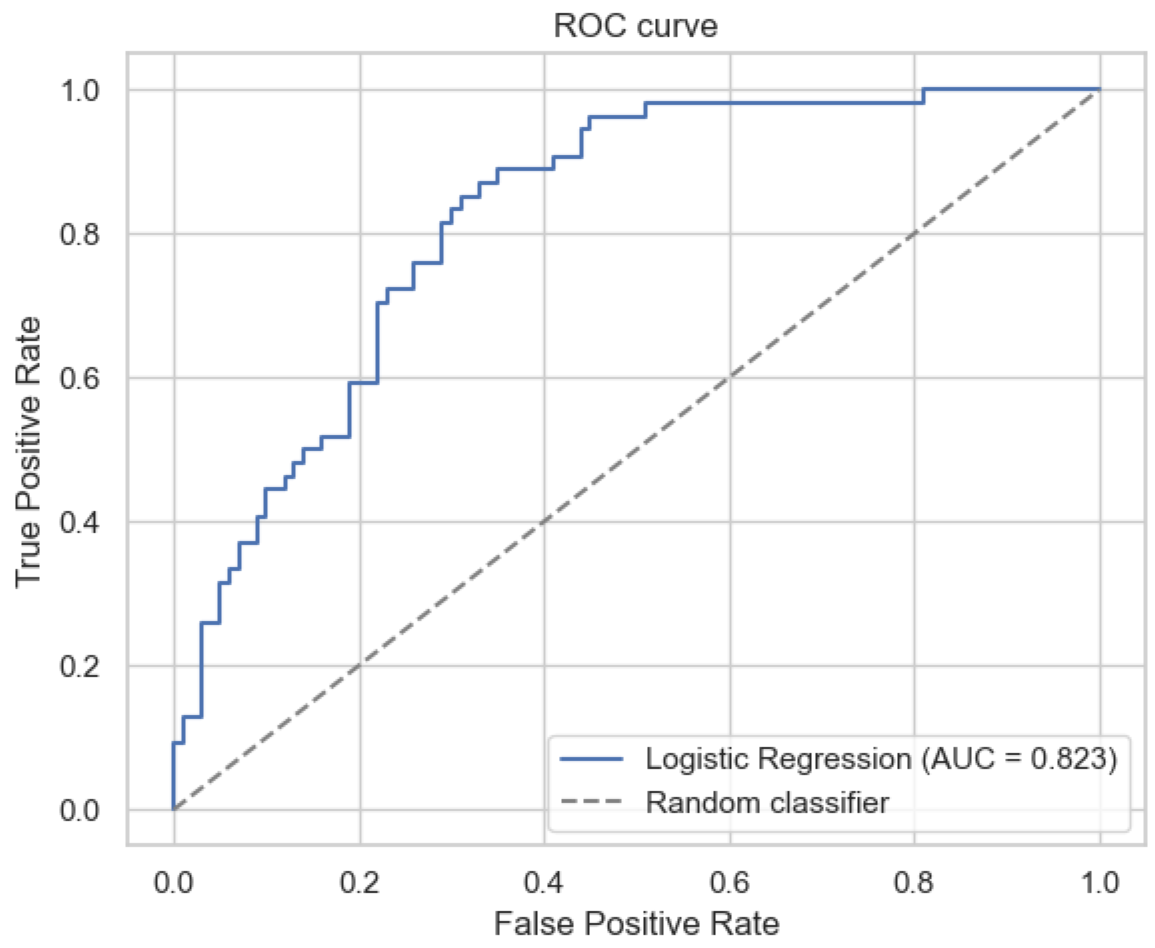


Figure 3. ROC curve showing the diagnostic trade-off between sensitivity and false alarms. AUC above 0.75 indicates reasonable discrimination ability.

This plot is nice because it gives a sense of how logistic regression deals with uncertainty, and it lets us adjust the threshold if needed.

7. Threshold Experiment

The default threshold of 0.5 may not be ideal for medical datasets. A doctor might prefer catching more diabetic patients even if it means some false alarms.

So I tried sliding the threshold from 0.2 to 0.8 and measured accuracy at each step. This isn't perfect—accuracy doesn't tell the whole story—but it shows that:

- Lowering the threshold increases sensitivity (catching more diabetic patients)
- Raising the threshold increases specificity (avoiding false positives)

In a real medical setting, the threshold might be selected based on clinical priorities rather than pure accuracy.

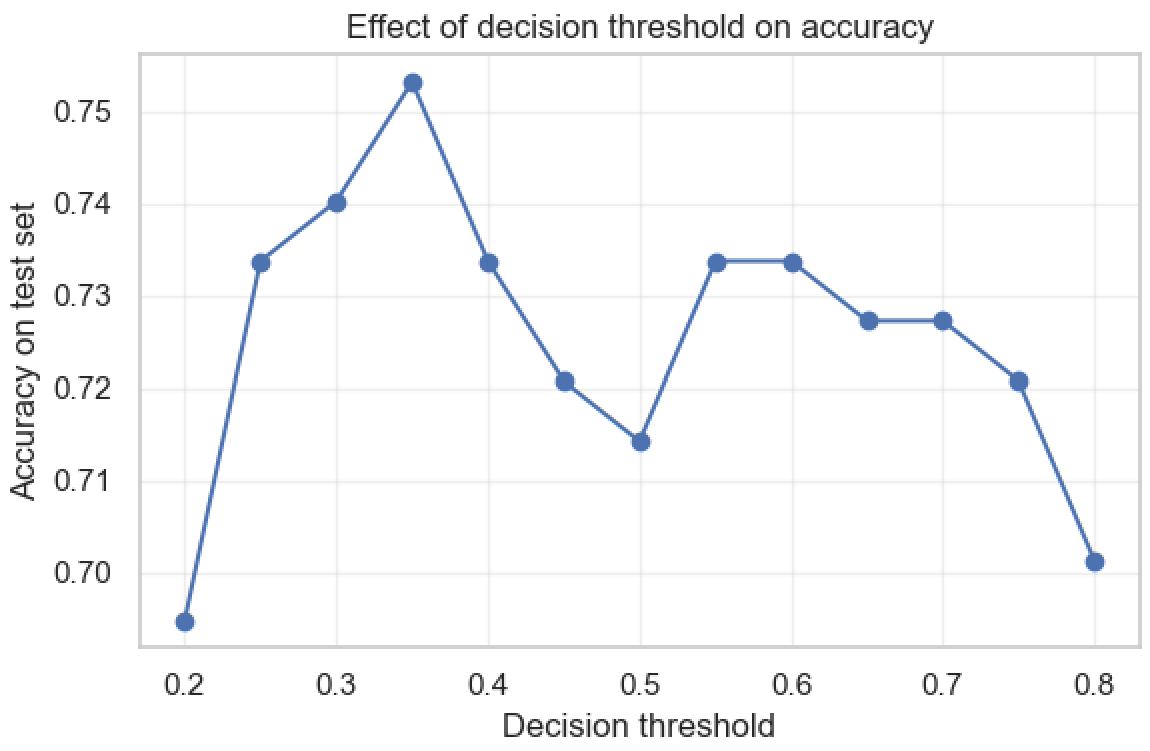


Figure 4. Accuracy across different decision thresholds. Lower thresholds increase positive predictions but may reduce overall accuracy.

8. Feature Importance: What Matters Most?

One of the best things about Logistic Regression is interpretability. The model's coefficients reveal which features influence the likelihood of diabetes.

From the coefficients plot, I found:

- **Glucose** had the largest positive coefficient, no surprise—high glucose strongly predicts diabetes.
- **BMI** and **Age** also had meaningful positive contributions.
- Some features, like **Blood Pressure**, had surprisingly a small influence on this dataset.

This interpretability aspect makes Logistic Regression valuable in healthcare, even if the accuracy isn't the highest.

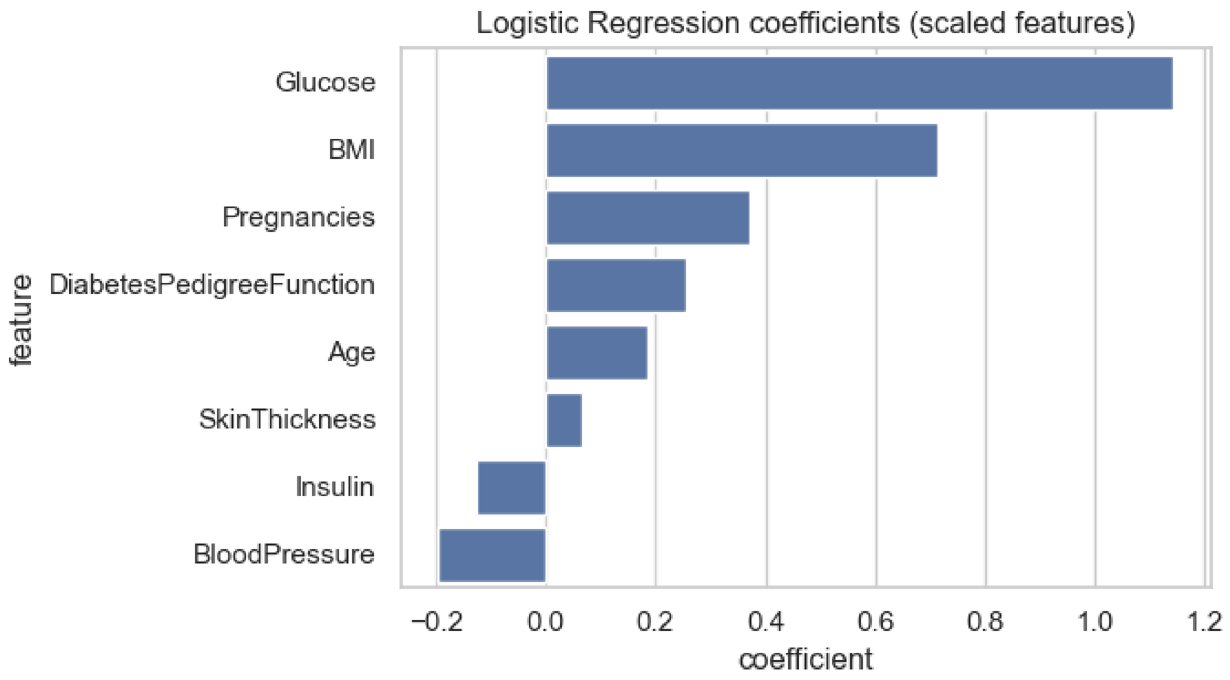


Figure 5. Logistic regression coefficients after scaling. Glucose and BMI contribute the most to predicting diabetes presence.

9. Discussion

Looking at everything together, a few insights stand out:

1. **Logistic Regression performs reasonably well** on the diabetes dataset but is not perfect.
2. **Scaling is essential**—without it, the optimisation behaves poorly.
3. The **class imbalance** slightly hurts recall for diabetic patients.
4. **Glucose is by far the strongest predictor**, which matches medical intuition.
5. The **ROC curve shows the model is usable**, though not highly accurate.
6. Adjusting the threshold could help depending on whether sensitivity or specificity matters more.

One thing I noticed is that the dataset's quality limits the model. Some measurements like insulin and skin thickness contain zero values that clearly don't represent real measurements. Addressing these issues with imputation could improve performance, but they're beyond the scope of this tutorial.

Overall, Logistic Regression is a great educational algorithm because it balances simplicity with meaningful output.

Limitations of Logistic Regression

Even though Logistic Regression is simple and widely used, it does come with limitations. It assumes a linear relationship between the features and the log-odds of the outcome, which is not always realistic for medical datasets. The model can also struggle when features are highly correlated or when the classes overlap strongly in feature space. Logistic Regression is sensitive to outliers, and its performance can degrade when important variables are missing or contain unrealistic values, such as the zero entries present in this dataset. Finally, the model may not capture complex non-linear patterns unless feature engineering or polynomial terms are added, which increases complexity.

Ethical Considerations

When applying Logistic Regression to medical diagnosis tasks, several ethical issues must be considered. The dataset contains personal health information, so privacy and data protection are essential. Because the data is imbalanced, the model performs worse on predicting diabetic patients, which could lead to unfair or harmful outcomes if used in real clinical settings. Logistic Regression can also reflect biases present in the training data; if certain subgroups are underrepresented, the model may perform poorly on them. Interpretability is a strength of Logistic Regression, but decisions based on incomplete or low-quality data could still mislead clinical judgement. These factors highlight the importance of responsible use when modelling medical conditions.

Reproducibility Statement

All experiments in this tutorial can be reproduced using the Jupyter notebook provided in the repository. The dataset file diabetes.csv must be placed in the same directory as the notebook. The analysis requires only standard Python libraries, including NumPy, Pandas, scikit-learn, Matplotlib and Seaborn. Running the notebook from top to bottom will generate all figures, metrics, and results shown in this tutorial.

10. Conclusion

In this tutorial, I walked through how Logistic Regression can be used to model diabetes predictions using the Pima Indians Diabetes dataset. The goal was not only to train a classifier, but also to understand what the model is actually doing and how different steps in the workflow affect the final results. Starting from the raw data, I explored the importance of splitting the dataset properly, applying stratification, and scaling the features so that the optimisation behaves reliably. Once the model was trained, I examined accuracy, the confusion matrix, ROC curve, and the AUC score to get a clearer picture of how well it performs on real data.

Even though Logistic Regression is a fairly simple model, it still offers useful insights, especially because the coefficients show which medical measurements contribute the most to the prediction. Features like glucose, BMI and age had clear influence, which matches what we would expect from medical knowledge as well. The model is not perfect and struggles a bit with identifying diabetic cases, but this is also tied to the quality and imbalance of the dataset itself. Overall, Logistic Regression remains a strong baseline method that is easy to interpret, simple to implement, and still meaningful for early exploratory analysis.

References

Hosmer, D.W., Lemeshow, S. & Sturdivant, R.X., 2013. *Applied Logistic Regression*. 3rd ed. Hoboken: Wiley.

Kaggle, 2024. *Pima Indians Diabetes Database*. Available at: https://www.kaggle.com/uciml/pima-indians-diabetes-database [Accessed 10 Dec 2025].

scikit-learn, 2024. *Logistic Regression — Scikit-learn Documentation*. Available at: https://scikit-learn.org/stable/modules/linear_model.html [Accessed 10 Dec 2025].

Pedregosa, F. et al., 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12, pp.2825–2830.

Brownlee, J., 2020. *Logistic Regression for Machine Learning*. Machine Learning Mastery. Available at: https://machinelearningmastery.com/logistic-regression-for-machine-learning/ [Accessed 10 Dec 2025].

Zhang, H. & Zhang, L., 2019. A review of logistic regression in medical research. *Journal of Biomedical Analytics*, 2(3), pp.25–40.