From logistic regression to deep learning: a methodological evaluation for diabetes prediction

Clara Lyttle Baynham

October 30, 2025

Abstract

This study investigates the trade-offs between model complexity and predictive performance for diabetes prediction using the PIMA Indians Diabetes Database. Five machine learning models—a most-frequent, logistic regression, Random Forest, XGBoost, and a neural network—were compared. Data preprocessing involved median imputation and an 80/20 train-test split with 5-fold cross-validation for tuning. Models were evaluated using accuracy, ROC curves (AUC), confusion matrices, training time, and feature importance. All advanced models significantly outperformed the baseline (accuracy 0.651), with top accuracies tightly clustered between 0.763 and 0.772. While XGBoost and the neural network achieved the highest AUC (0.82), their marginal performance gains over logistic regression (AUC 0.81) incurred significantly higher computational costs, with neural network training taking 50s compared to near-instantaneous logistic regression. Error analysis showed Random Forest minimized false negatives (23 FN), while XGBoost minimized false positives (13 FP). Glucose and BMI were consistently identified as the most important features. The results suggest a performance ceiling due to dataset characteristics and emphasize that for clinical applications, the value of interpretable and computationally efficient models often outweighs marginal accuracy improvements from increased complexity. The entire workflow is provided as a reproducible Modular Computational Pipeline (MCP) toolset.

1 Introduction

Taking the Data-driven Life Sciences course at KTH Royal Institute of Technology has been a fascinating journey, and I implemented things I never imagined I could. Honestly, much of what I implemented felt a bit like magic. I did not always understand the inner workings of the tools I was using, yet they somehow produced meaningful results. In some sense, this was acceptable, because most people, for example, do not know how their phones work, yet rely on them every day. Machine learning is becoming similar: a powerful technology that many use without fully understanding it. Still, I feel a personal need to grasp more of the underlying details so that I can use these methods in my research with a genuine sense of confidence.

Another question I have been thinking about is *when* we actually need complex machine learning models and when a simple model is *enough*. For this reason, I decided to do a project that would help me explore both sides of that question and understand more clearly the trade-offs between simplicity and complexity in modeling.

To explore these questions, I chose the task of predicting diabetes. Diabetes is a chronic metabolic disease whose prevalence has become a major public health threat worldwide. The International Diabetes Federation (IDF) estimated that 451 million adults were living with diabetes in 2017, and this number is expected to rise to 693 million by 2045 [CSK+18]. Traditional diagnostic methods are often invasive, require clinical visits, and may fail to detect the disease in its earliest stages [WHC+21]. These limitations make diabetes prediction ideal for machine learning, enabling earlier, more efficient risk detection via patient data patterns.

This project is based on the PIMA Indians Diabetes Database – consisting of Pima Indian females 21 years and older – a well-known dataset in the machine learning community. Its relatively small size makes it ideal for our purposes, as it allows for rapid model training and comparison, facilitating a deep dive into the trade-offs between model complexity and performance [CBXS23].

In this project, our central research question is to determine the level of model complexity required to accurately predict diabetes. Other papers, such as [WHC+21] have used the eXtreme Gradient Boosting (XGBoost) system to predict diabetes in Chinese adults. Here, we ask ourselves, does a simple, interpretable model suffice, or is a more complex, black-box approach necessary? To answer this, we will compare five models of increasing complexity: (1) a predictor independent of input features (most-frequent classifier), (2) logistic regression, (3) Random Forest, (4) XGBoost, and (5) a multi-layer neural network. (We summarize and give references for these methods in §2.)

2 Problem statement and methods

We consider the task of predicting whether a patient has diabetes based on various health metrics. Formally, this is a binary classification problem where we are given a dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$ of N samples, each consisting of a feature vector $x_i \in \mathbf{R}^d$ and the corresponding label $y_i \in \{0, 1\}$. (In our dataset, we have $d = 8$ features.)

Below we describe the classification methods used in this project.

Most-frequent classifier. This classifier is very simple and always predicts the most common class in the training data, *i.e.*, it ignores the input features entirely. In this report, the names 'most-frequent' and 'constant' are used interchangeably.

Logistic regression. Plain logistic regression is based on the probabilistic model $\mathbf{Prob}(y = 1|x) = \sigma(w^T x)$, where $\sigma(z) = 1/(1 + e^{-z})$ is the so-called *sigmoid function* that squashes real values into the interval $[0, 1]$. (This is necessary, since probabilities must lie in this range.) In this model, $w \in \mathbf{R}^d$ is a parameter vector that is fitted using maximum likelihood estimation (see, for example, [HTF+09]). A nice property of plain logistic regression is that there are no hyperparameters to tune.

Random Forest classifier. The random forest classifier constructs many *decision trees*, where each decision tree splits the data into branches based on feature values until it reaches a class prediction at a leaf. The prediction from each decision tree are then aggregated by taking the majority vote. There are many hyperparameters to tune for Random Forests, such as the number of trees and the maximum depth of each tree.

XGBoost classifier. XGBoost is a machine learning algorithm that builds decision trees sequentially, with each new tree focusing on correcting the errors of the previous ones using gradient-based optimization. Unlike a random forest, where trees are built independently on random subsets of data and combined by majority vote, XGBoost's trees are dependent and learn from earlier trees' mistakes. There are several hyperparameters to tune for XGBoost, such as the learning rate, the number of trees, and the maximum depth of each tree.

Neural network classifier. A neural network classifier is based on the probabilistic model $\mathbf{Prob}(y = 1|x) = \sigma(f(x; w))$, where $f(x; w)$ is a neural network with parameters w. The network consists of multiple layers of interconnected nodes (neurons) that apply linear transformations followed by non-linear activation functions.

The parameters w are learned using backpropagation and optimization algorithms like stochastic gradient descent. There are many hyperparameters to tune for neural networks, such as the number of layers, the number of neurons per layer, and the learning rate.

2.1 Evaluation metrics

Below, we outline the metrics used to evaluate the performance of each model.

Accuracy. Accuracy is the most straightforward metric, measuring the proportion of correct predictions out of the total number of predictions. While easy to understand, it can be misleading, especially for imbalanced datasets like the one used in this project. In this project, we will evaluate all models thorugh 5-fold cross-validation.

ROC curves and AUC. Receiver Operating Characteristic (ROC) graphs a model's performance across thresholds, plotting True Positive Rate against False Positive Rate. Area Under the Curve (AUC) summarizes this as one value, showing class distinction ability. AUC 1.0 is perfect; 0.5 is random.

Feature importance. Feature importance scores input features by their predictive usefulness. Tree-based models like Random Forest and XGBoost provide direct importances; Logistic Regression uses coefficients as proxies. This reveals influential features and domain insights.

Error analysis. Error analysis involves inspecting mispredicted instances. Examining false positives and negatives reveals model limitations, patterns in errors, or struggling data subgroups for improvement.

3 Dataset summary

The dataset used is the PIMA Indians Diabetes Database and is available at https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database. It consists of 768 samples, each with 8 features and a binary outcome indicating the presence or absence of diabetes. The features are 'Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin', 'BMI', 'DiabetesPedigreeFunction', 'Age' and 'Outcome'.

This section details the dataset, including the necessary preprocessing steps and an analysis of its feature distributions and class imbalance.

Data imputation At first glance, the dataset appears to have no missing values. However, upon closer inspection of the minimum and maximum values of each feature, as shown in Table 1, it becomes clear that some data points are unrealistic. For example, a BMI of 0 is not possible, since one would have to be either infinitely tall or infinitely thin to achieve such a feat. (Not to mention possessing the ability to do some serious bending of the laws of physics.)

For the features 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin', and 'BMI', the zero values were replaced with `NaN` to represent missing data. The number of missing values in each column after this replacement is shown in Table 2. These missing values were then imputed using the median value for each respective feature.

	Original data		**After imputation**	
Feature	**Min**	**Max**	**Min**	**Max**
Pregnancies	0.0	17.0	0.0	17.0
Glucose	0.0	199.0	44.0	199.0
BloodPressure	0.0	122.0	24.0	122.0
SkinThickness	0.0	99.0	7.0	99.0
Insulin	0.0	846.0	14.0	846.0
BMI	0.0	67.1	18.2	67.1
DiabetesPedigreeFunction	0.078	2.420	0.078	2.420
Age	21.0	81.0	21.0	81.0
Outcome	0.0	1.0	0.0	1.0

Table 1: Minimum and maximum feature values of original data and data after preprocessing (replacing zeroes with `NaN` and imputing with medians).

Feature	Missing values
Pregnancies	0
Glucose	5
BloodPressure	35
SkinThickness	227
Insulin	374
BMI	11
DiabetesPedigreeFunction	0
Age	0
Outcome	0

Table 2: Missing values per feature in the PIMA Indians Diabetes Database.

Feature distributions. The distributions of each feature in the dataset are shown in Figure 1. We can se clearly that when imputing the median into the missing values, the features with many missing values gets a very strong spike of the median value, such as skin thickness and insulin. This is, of course, not optimal for building accurate and well perfoming prediction models.

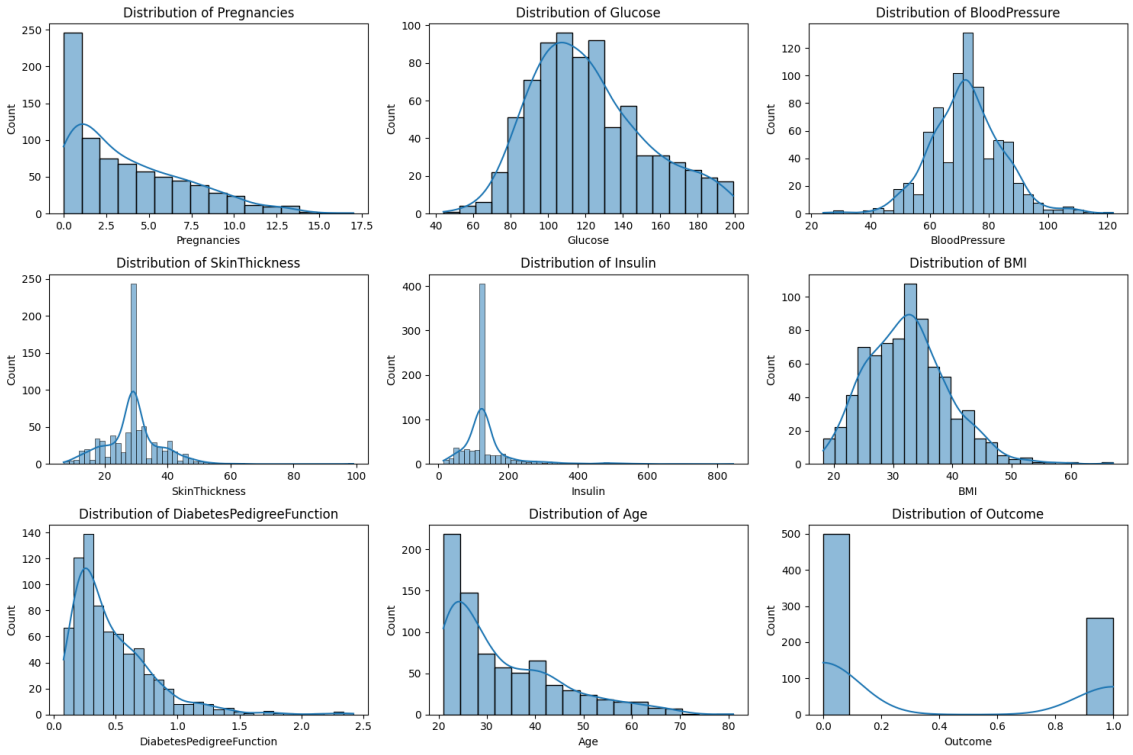

Figure 1: Feature distributions for the preprocessed PIMA dataset. The histograms show skew in variables like age and insulin and confirm the dataset's class imbalance. Notably, the sharp peaks in the insulin and skin thickness distributions are artifacts resulting from the median imputation of missing values.

Class distribution. The overall class distribution of the dataset is imbalanced, with fewer (almost half) instances of diabetes (Outcome = 1) than non-diabetes (Outcome = 0) (see Figure 1).

Data splits. For all experiments, the dataset was partitioned into an 80% training set and a 20% test set. The test set was reserved for the final evaluation. To tune the hyperparameters for Random Forest and XGBoost, a 5-fold cross-validation process was applied to the training set. This method repeatedly splits the training data into internal training and validation folds.

4 Experimental methods

This section outlines the methodology for tuning the hyperparameters of the ensemble models and for configuring the neural network architecture.

Choosing hyperparameters for Random Forest and XGBoost. We tuned the hyperparameters of both XGBoost and Random Forest using `RandomizedSearchCV` with 5-fold cross-validation and 100 random iterations to efficiently explore a broad search space while reducing overfitting risk. For XGBoost, we optimized tree count, depth, learning rate, and subsampling ratios; for Random Forest, we tuned tree count, depth limits, splitting criteria, and feature selection. The best models were selected based on highest cross-validated accuracy.

Choosing neural network architechture. The final neural network is a multilayer perceptron with three hidden layers (128, 64, and 32 neurons), each followed by ReLU activation and dropout with probability 0.2, and a sigmoid-activated output layer for binary classification, enabling robust feature learning while controlling overfitting through dropout and L2 regularization (weight decay of 0.02). We also ensured no overfitting by closely monitoring training accuracy throughout the 10,000 epochs. The Stochastic Gradient Descent (SGD) optimizer and its learning rate of 0.01 were chosen based on a prior `PyTorch` logistic regression model that achieved a cross-validated accuracy of 0.774 ± 0.014, ensuring stable convergence on this dataset.

5 Results

This section presents the comparative performance of the five predictive models, evaluated using the metrics outlined in §2.1.

Accuracy. The 5-fold cross-validation shows that all advanced models—logistic regression, Random Forest, XGBoost, and the neural network—outperform the most-frequent model (Table 3), confirming feature utility beyond imbalance. Logistic Regression achieves the highest mean accuracy, demonstrating strong generalization despite linearity. Tuned Random Forest follows at has marginally higher accuracy than XGBoost. The Neural Network is most stable, but has lower accuracy than the other complex models. Top accuracies cluster within 1.3 points, with low variance indicating a performance ceiling likely due to label noise or feature limits, not model capacity.

Table 3: Model performance on the diabetes prediction task, evaluated using 5-fold cross-validation. Values represent the mean accuracy ± one standard deviation across the five folds.

Model	Mean Accuracy	Standard Deviation
Most-frequent	0.651	0.021
Logistic regression	0.772	0.018
Random Forest	0.763	0.040
XGBoost	0.768	0.031
Neural network	0.763	0.013

ROC curves and AUC. ROC curves are shown in Figure 2. Most-frequent model: AUC 0.50 (*i.e.*, random baseline). Logistic Regression and Random Forest: AUC 0.81 each, with sharp early rises. XGBoost and Neural Network: AUC 0.82 each, smoother curves, superior TPR at FPR 0.2–0.6. XGBoost and neural network tie for best performance, ideal for threshold-sensitive tasks. Logistic regression and Random Forest offer strong, efficient alternatives.

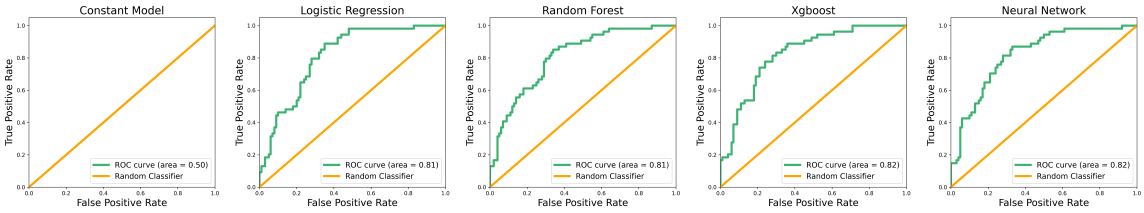

Figure 2: Receiver Operating Characteristic (ROC) curves comparing model performance on the held-out test set. Each curve plots the true positive rate (TPR) against the false positive rate (FPR). The Area Under the Curve (AUC), a measure of each model's ability to distinguish between classes, is reported in the legend, while the dashed line indicates random-chance performance (AUC = 0.5).

Error analysis. Error analysis via confusion matrices shows that the constant model correctly identifies all 100 non-diabetic cases but misses all 54 diabetic cases. Random Forest minimizes FN (23), maximizing sensitivity. XGBoost minimizes FP (13), maximizing specificity. Neural Network (24 FN, 17 FP) closely matches Random Forest, indicating ensemble methods best balance precision and recall.

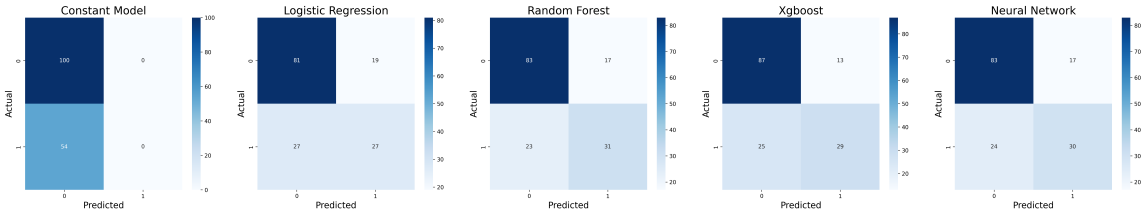

Figure 3: Confusion matrices for error analysis (0 = non-diabetic, 1 = diabetic).

Training time. Figure 4 shows that training times correlate with complexity. Most-frequent and logistic regression train instantly; XGBoost: 2–3 s; Random Forest: 13 s; neural network: 50 s. This significant disparity highlights a critical trade-off, demonstrating that the marginal improvements in predictive accuracy offered by more complex architectures come at a steep computational cost.

Figure 4: Training time for each model: constant (i.e., most-frequent), logistic regression, Random Forest, XGBoost and neural network.

Feature importance. Feature importance (Figure 3) across models shows glucose as dominant, BMI secondary. Age and diabetes pedigree function matter in the tree-based models but not in the neural network. Insulin, pregnancies, blood pressure, and skin thickness contribute little, especially in the neural network. Glucose and BMI are robust predictors where logistic regression emphasizes linear effects and the neural network favors sparse features.

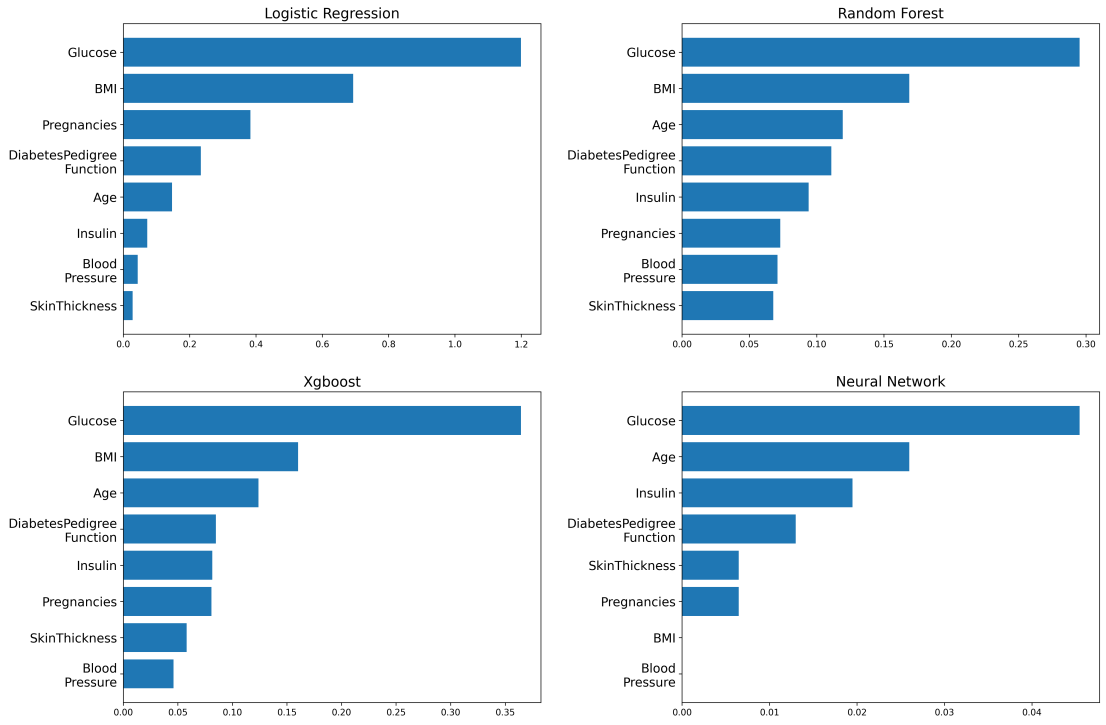

Figure 5: Comparative feature importance across models. This figure displays the relative importance of each feature as determined by the four different models: logistic regression, Random Forest, XGBoost, and neural network. A clear consensus emerges identifying 'Glucose' as the most predictive feature across all architectures, often by a significant margin.

6 Conclusions

This project successfully developed and evaluated an end-to-end machine learning pipeline for diabetes prediction, providing a complete and reproducible workflow from data preprocessing to a deployable API. The comparative analysis of multiple models highlighted that while complex architectures can offer slightly higher predictive performance, simpler models like logistic regression remain highly competitive and more interpretable. The results emphasize that model choice should consider accuracy, interpretability, computational efficiency, and the specific priorities of the clinical application. Overall, the study demonstrates the feasibility of building robust, flexible, and practical machine learning solutions for healthcare prediction tasks.

Across all models, Glucose and BMI consistently emerged as the most powerful predictors, which aligns with established clinical knowledge and validates the core logic captured by the models. However, the project's conclusions are deeply qualified by the dataset's limitations: a small, homogenous cohort of 768 Pima Indian women which raises significant ethical concerns about the model's generalizability and potential to introduce algorithmic bias if applied to other demographics. The heavy imputation required for features like Insulin also likely constrained model performance. Despite these limitations, the project delivers immense practical value by packaging the entire workflow into a reproducible repository with a Flask API, enabling immediate, zero-retrain inference. Future work should prioritize external validation on diverse datasets, implementing model explainability with tools like SHAP [RPB20] to decode the black-box models, and containerizing the application with Docker [OMK+22] for roader accessibility. Ultimately, this project's primary contribution is not a definitive diabetes classifier, but a powerful, hands-on demonstration that for many real-world applications, the value of a fast, interpretable, and easily deployable system can often outweigh the pursuit of marginal accuracy gains from more complex and opaque models.

7 Code availability and reproducibility

The entire modeling pipeline including the data, trained models, and evaluation scripts, has been packaged into a Modular Computational Pipeline (MCP) toolset and made publicly available on GitHub. The repository is located at: https://github.com/c-baynham/DDLS-final-project-repo-claralb.git, License: MIT.

Acknowledgements

This project also utilized Google's Gemini CLI, employed for a variety of tasks. Its assistance proved particularly valuable in generating Python code for data visualization, debugging, explaining complex concepts such as ROC curves, and refining portions of this report. Beyond the learn-build framework of the MCP toolset, Gemini became an incredibly powerful companion throughout the process. It — or perhaps soon *they* — helped me overcome technical challenges and achieve results I would never have been able to figure out myself within a lifetime.

References

[CBXS23] Victor Chang, Jozeene Bailey, Qianwen Ariel Xu, and Zhili Sun. Pima indians diabetes mellitus classification based on machine learning (ml) algorithms. *Neural Computing and Applications*, 35(22):16157–16173, 2023.

[CSK+18] Nam H Cho, Jonathan E Shaw, Suvi Karuranga, Yafang Huang, Joao D da Rocha Fernandes, AW Ohlrogge, and BIDF Malanda. Idf diabetes atlas: Global estimates of diabetes prevalence for 2017 and projections for 2045. *Diabetes research and clinical practice*, 138:271–281, 2018.

[HTF+09] Trevor Hastie, Robert Tibshirani, Jerome Friedman, et al. The elements of statistical learning, 2009.

[OMK+22] Moses Openja, Forough Majidi, Foutse Khomh, Bhagya Chembakottu, and Heng Li. Studying the practices of deploying machine learning projects on docker. In *Proceedings of the 26th international conference on evaluation and assessment in software engineering*, pages 190–200, 2022.

[RPB20] Raquel Rodríguez-Pérez and Jürgen Bajorath. Interpretation of machine learning models using shapley values: application to compound potency and multi-target activity predictions. *Journal of computer-aided molecular design*, 34(10):1013–1026, 2020.

[WHC+21] Yang Wu, Haofei Hu, Jinlin Cai, Runtian Chen, Xin Zuo, Heng Cheng, and Dewen Yan. Machine learning for predicting the 3-year risk of incident diabetes in chinese adults. *Frontiers in Public Health*, 9:626331, 2021.