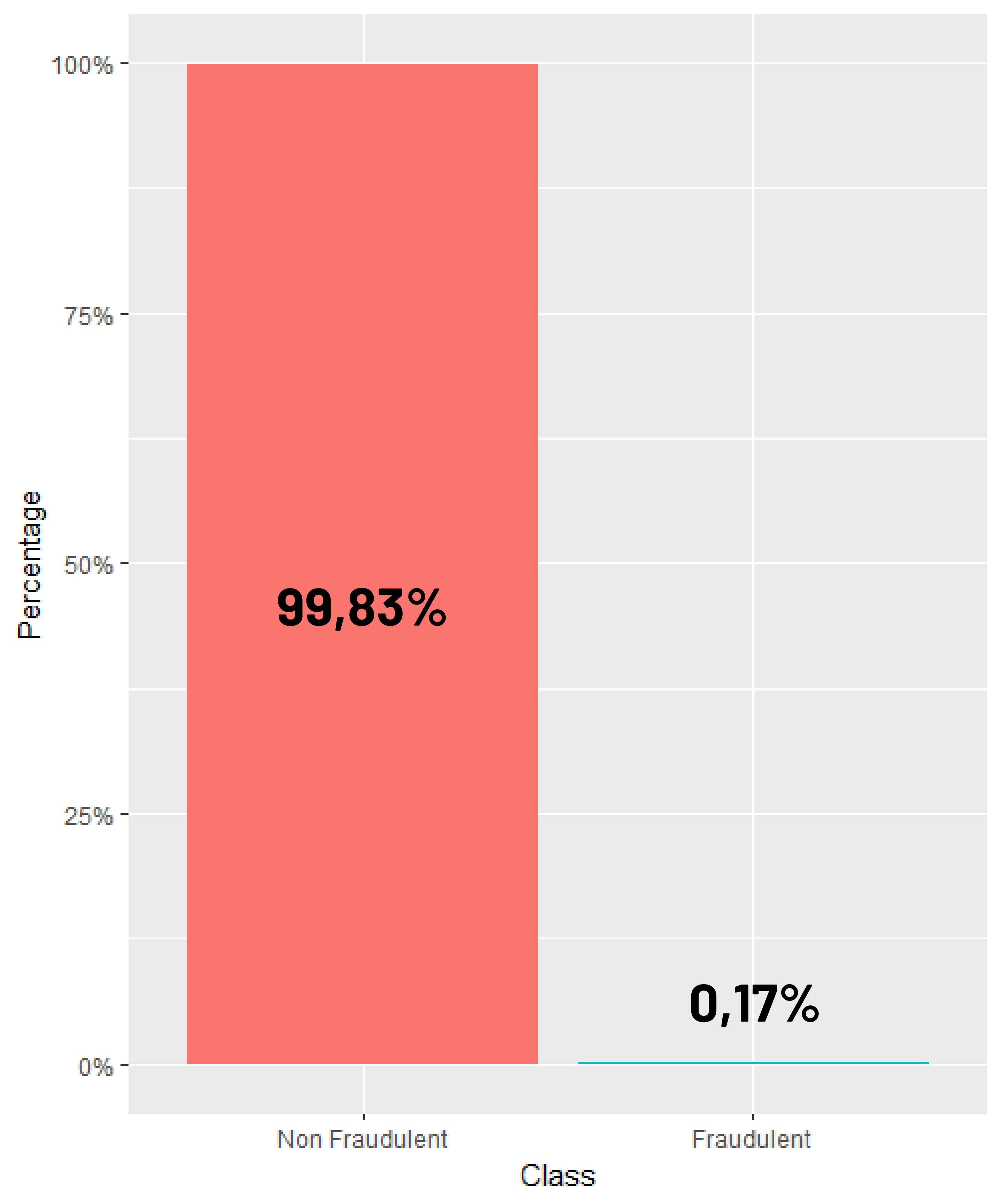
Fraud Predictive Modelling

A presentation made by:

Aldo Caumo, 866626
Eleonora Fiorentino, 899000
Meron Kedir Hussen, 898703
Rasmita Shrestha, 898598
Dipta Roy, 898395



Dataset description

The dataset contains credit card transactions in September 2013 by European cardholders over two days, consisting of **492** frauds out of 284,807 transactions.

The dataset contains only numerical input variables which are the result of a **PCA transformation**, to preserve customer privacy.

The only features which have not been transformed with PCA are **Time** and **Amount**.

Feature **Class** is the response variable and it takes value 1 in case of fraud and 0 otherwise.

Splitting the Data into Train and Test Set

Dataset Division for Credit Card Fraud Detection

- Total Transactions: 284,807
- Fraudulent Transactions (Class 1): 492
- Non-Fraudulent Transactions (Class 0): 284,315

Training Set:

- Size: 80% of the data
- Contains: 227,845 transactions
- Composition: 80% Non-Fraudulent, 20% Fraudulent
- Use: Model training

Test Set:

- Size: 20% of the data
- Contains: 56,962 transactions
- Composition: 80% Non-Fraudulent, 20% Fraudulent
- Use: Model evaluation

Now we will compare 3 different Class forecast models with the curved ROCs.

First Model

Logistic Regression

Second Model

LDA / QDA

Third Model

XGBoost

Logistic Regression

Probabilistic Model

Suitable for classification tasks where you want to estimate class probabilities.

Linear Relationship

It assumes a linear relationship between the independent variables and the log-odds of the dependent variable.

Interpretability

Provides interpretable coefficients

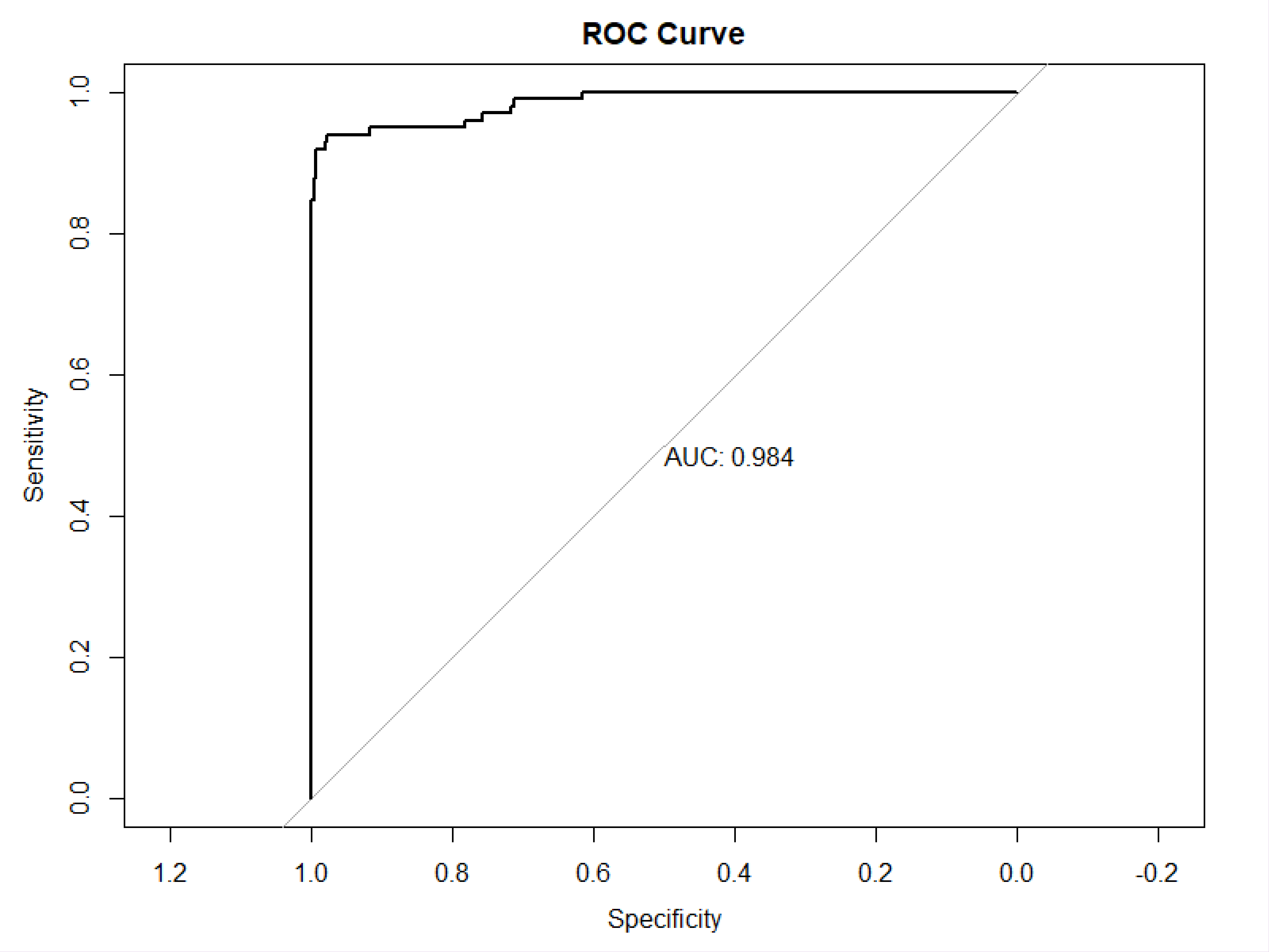
Sigmoid Function

The logistic or sigmoid function used in logistic regression ensures that the predicted probabilities are within the range [0, 1].

Maximum Likelihood Estimation

The coefficients are estimated using maximum likelihood estimation, which finds the values that maximize the likelihood of the observed data given the model.

ROC curve Logistic Regression



Confusion matrix

		Actual	
		0	1
Predicted	0	56853	34
	1	10	64

$$Sensitivity = \frac{TP}{(TP + FN)}$$

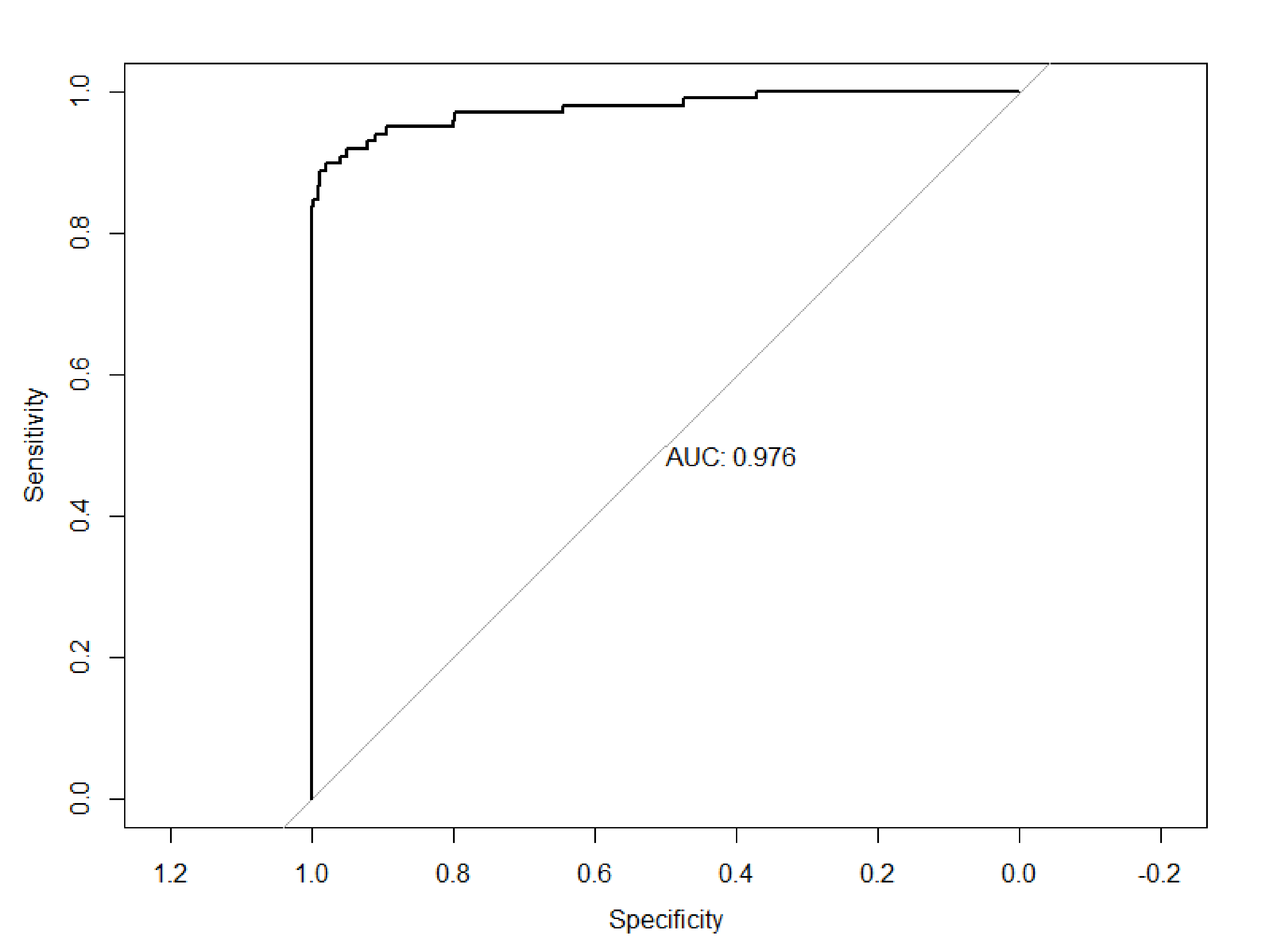
LDA and QDA

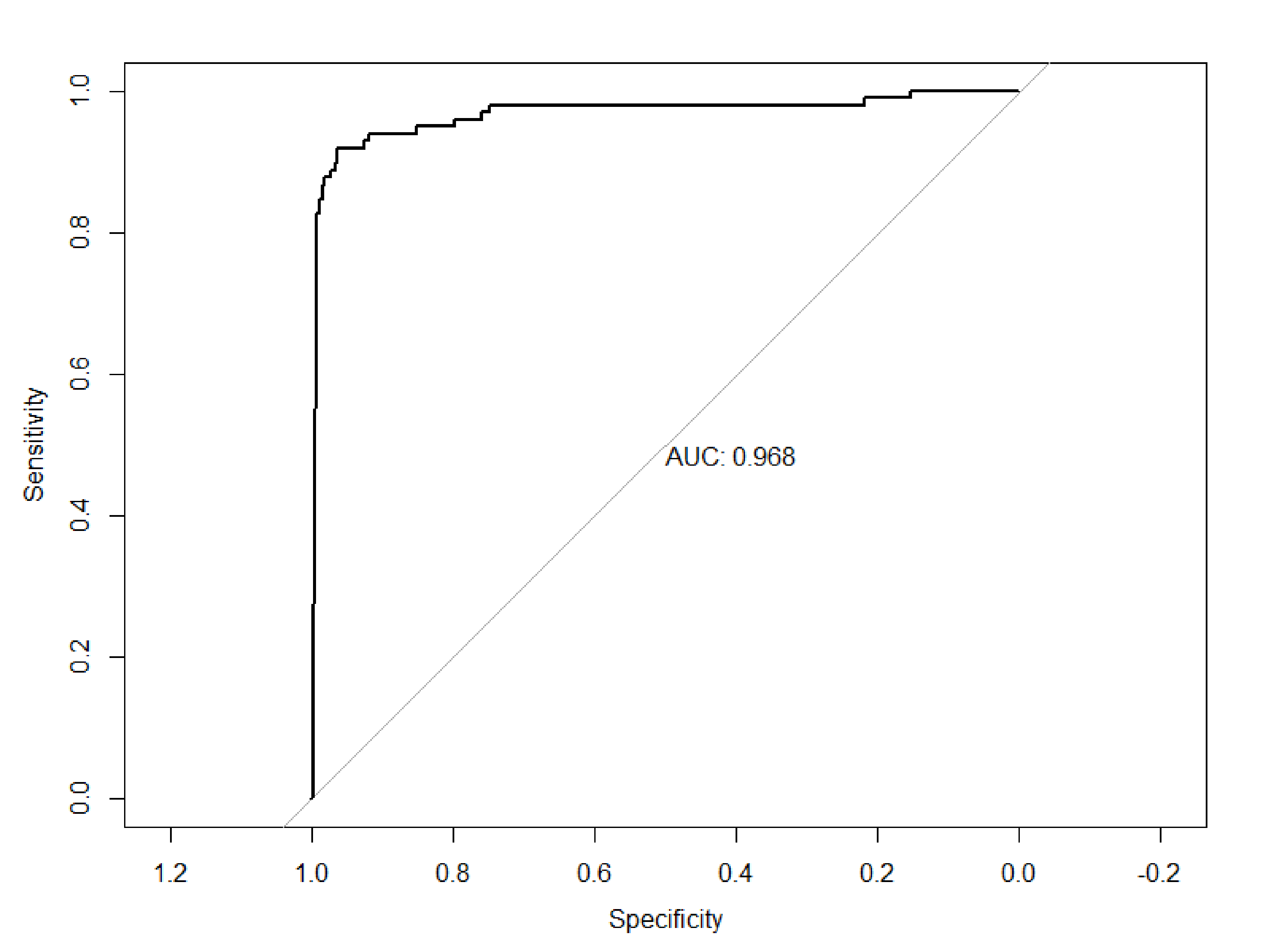
Linear Discriminant Analysis (LDA) and Quadratic Discriminant Analysis (QDA) are both statistical methods used for classification and dimensionality reduction in machine learning and statistics. They are primarily employed in supervised learning tasks where you want to assign observations to predefined classes or categories based on a set of features.

Simplified comparison between LDA and QDA:

- **Assumption about Covariance:**
  - LDA assumes equal covariance matrices for all classes.
  - QDA allows different covariance matrices for each class.
- **Computational Complexity:**
  - LDA is computationally less demanding compared to QDA.
  - QDA can be computationally expensive, especially with high-dimensional data or many classes.
- **Flexibility:**
  - LDA is less flexible than QDA due to its assumption of equal covariances.
  - QDA is more flexible and can model more complex relationships between features and classes.

ROC curve LDA / QDA





		Actual	
		0	1
Predicted	0	56853	25
	1	10	73

		Actual	
		0	1
Predicted	0	55490	12
	1	1373	86

Extreme Gradient Boosting

Ensemble Method

XGBoost is an ensemble learning method based on the gradient boosting framework. It builds a strong predictive model by combining the predictions of multiple weak models (usually shallow decision trees).

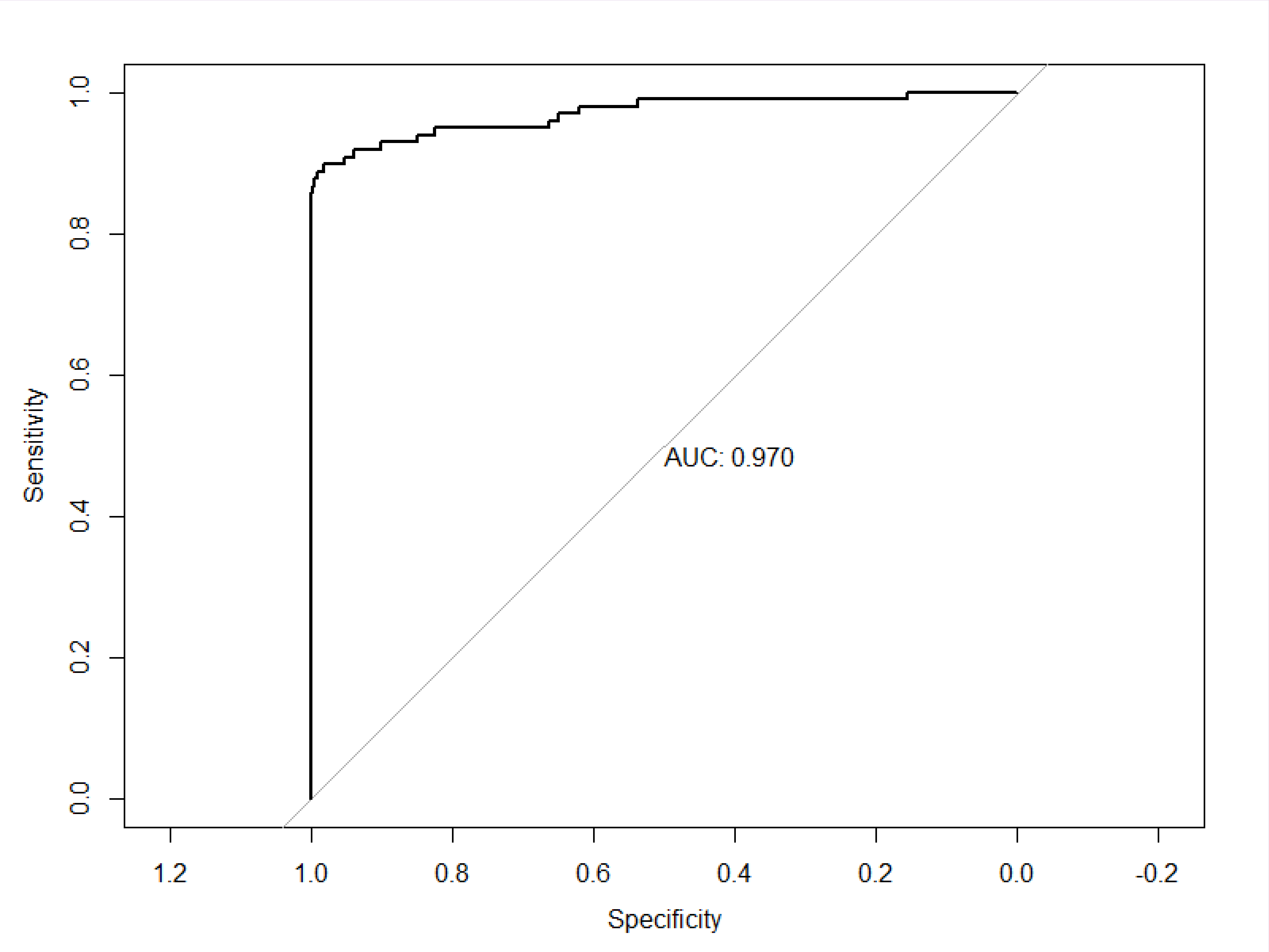
Sequential Training

Unlike Random Forest, which trains decision trees independently, XGBoost trains trees sequentially. Each new tree is trained to correct the errors made by the previous ones.

Regularization

XGBoost includes regularization terms in the objective function, which helps prevent overfitting and improves model performance.

ROC curve XGBoost



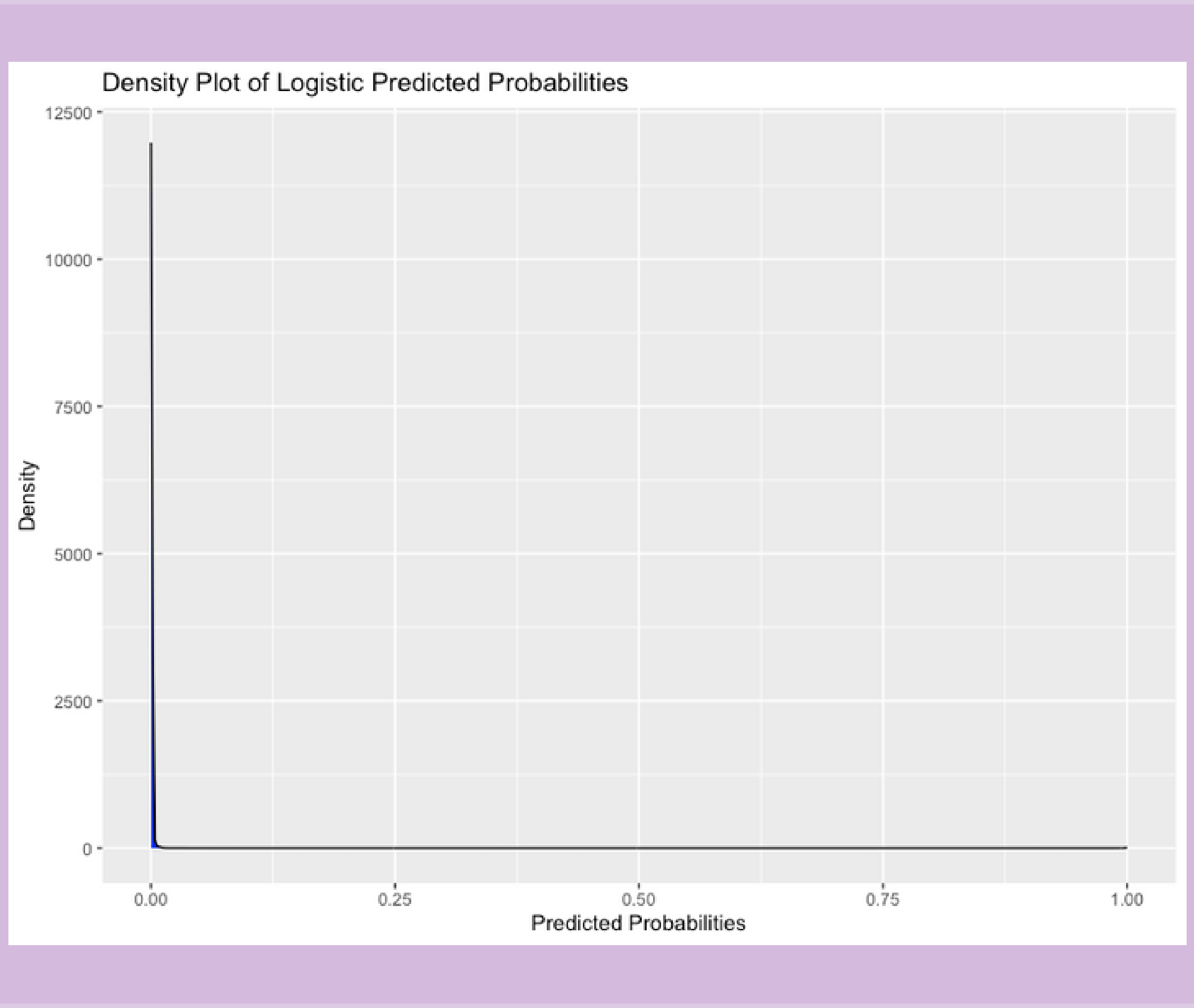
		Actual	
		0	1
Predicted	0	56859	22
	1	4	76

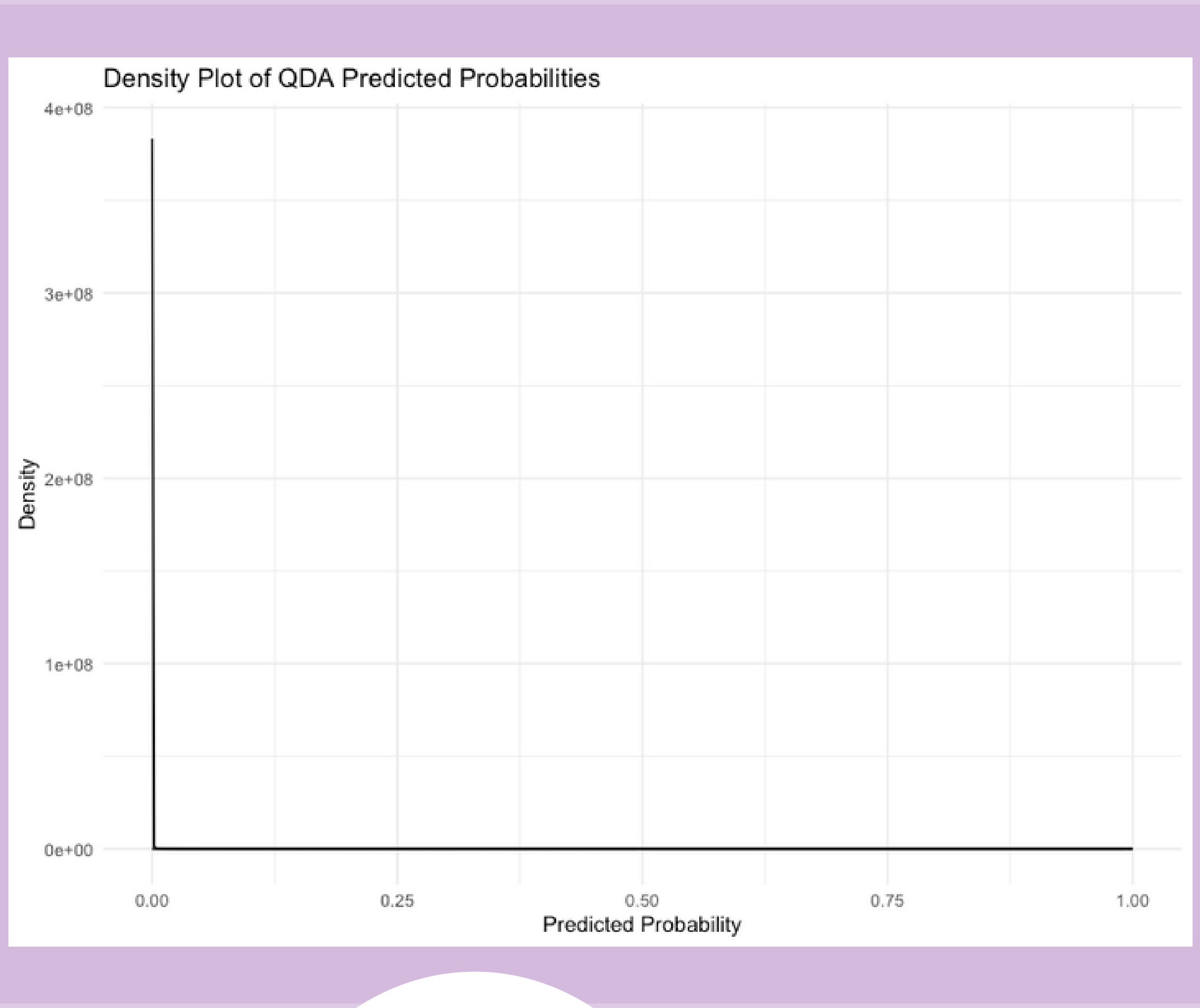
Results

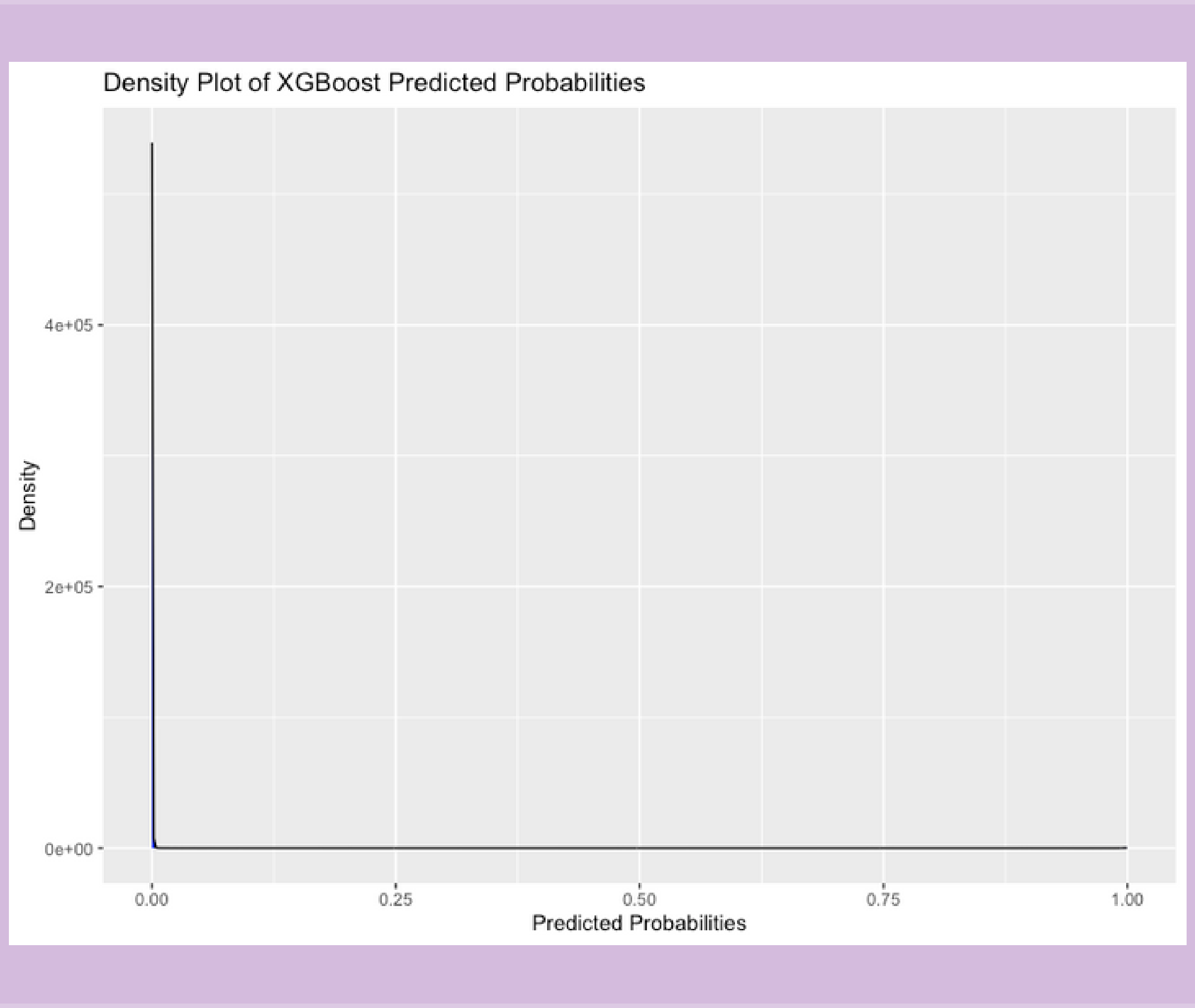
Confronting the three models

Logistic	QDA	XGBoost
AUC: 0.984	**AUC**: 0.968	**AUC**: 0.970
SENSITIVITY: 0.653	**SENSITIVITY**: **0.877**	**SENSITIVITY**: 0.775
SPECIFICITY: 0.999	**SPECIFICITY**: 0.976	**SPECIFICITY**: 0.999

Type of error and probability distributions


Density Plot of Logistic Predicted Probabilities


Density Plot of QDA Predicted Probabilities


Density Plot of XGBoost Predicted Probabilities

There’s clearly an L shape

Unbalanced dataset leads to **misclassification** for frauds

Possible solutions

Up-sampling

Synthetically generated data that corresponds with the fraudulent class are injected into the data set.

Cons: introduces bias since we are presenting additional data.

Down-sampling

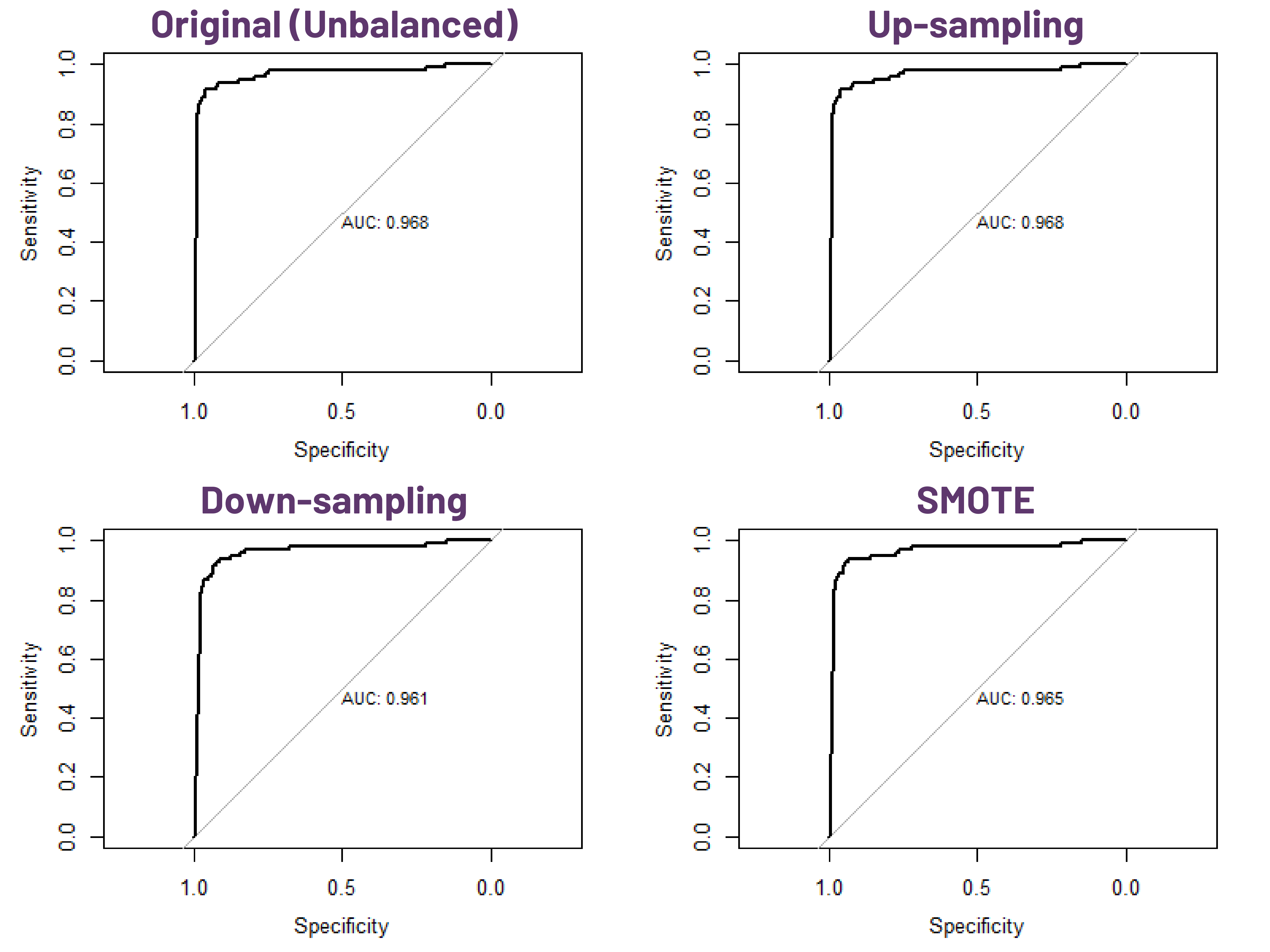
Observations from the non-fraudulent transactions are randomly removed until the counts of both classes are approximately the same.

Cons: may remove observations that might contain useful information when it comes to fitting a good decision boundary.

SMOTE

Non-fraudulent class is down-sampled while the fraudulent minority class is synthesized to create new instances by interpolating between existing ones (via k-nearest neighbor).

Cons: can increase overlapping of classes and introduce additional noise.



Sensitivity	
Original	0,877
Up-sampling	**0,898**
Down-sampling	0,877
SMOTE	0,888

Conclusions

Accuracy is not all

This was an exercise in dealing with highly imbalanced data that required careful analysis. Proceeding without accounting for the imbalance would have given **misinformed** results.

Balancing

Balancing classes can lead to better overall model performance, it helps prevent the model from being biased towards the majority class.

Metrics like **precision**, **recall**, and **F1-score** become more reliable when classes are imbalanced.

Thank you