

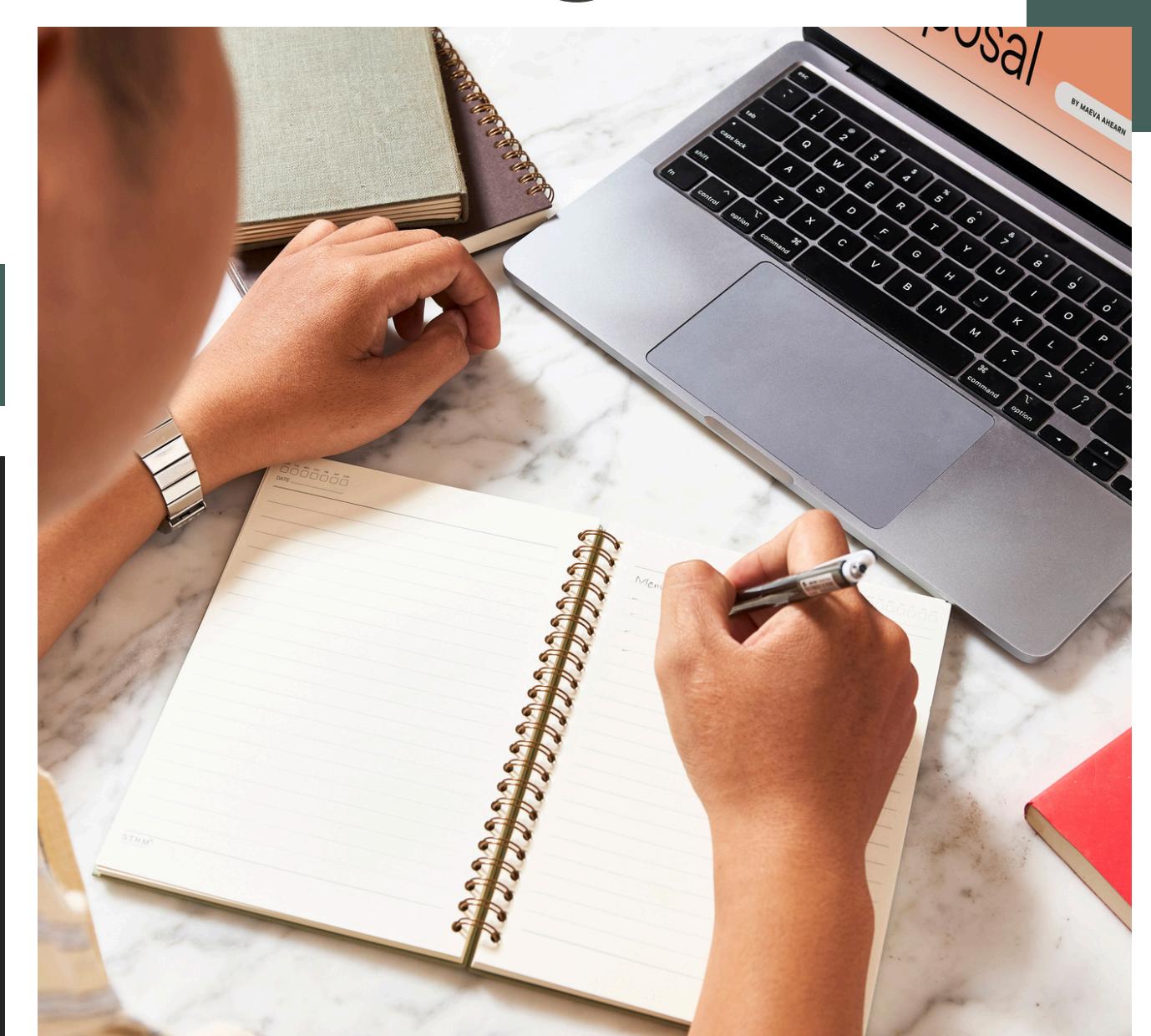
# Predicting the Success of Bank Telemarketing for Selling Long-term Deposits

**Presented By:**

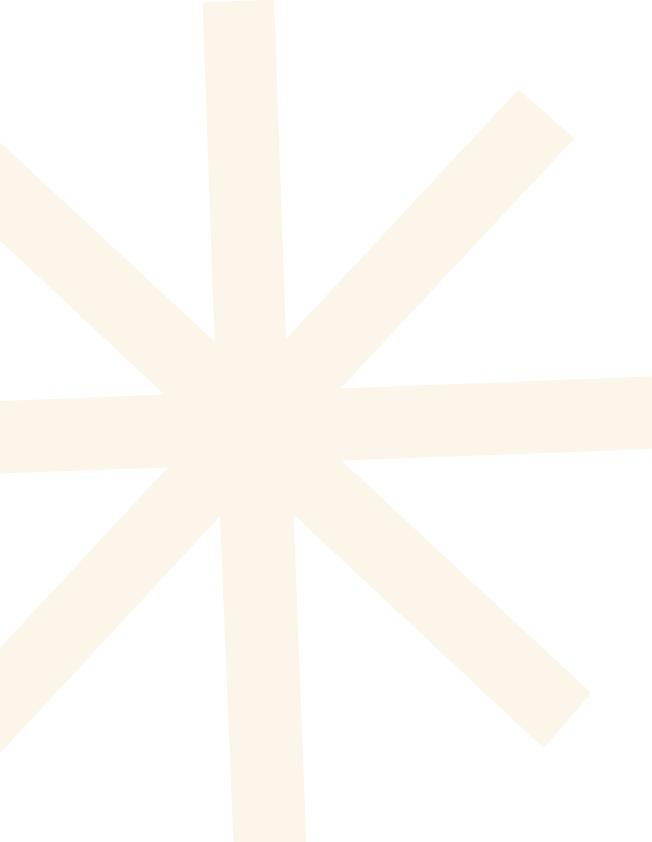
Group2  
Alfonso Garcia  
Anna Ikeda  
Erick Espinoza

**GitHub link:**

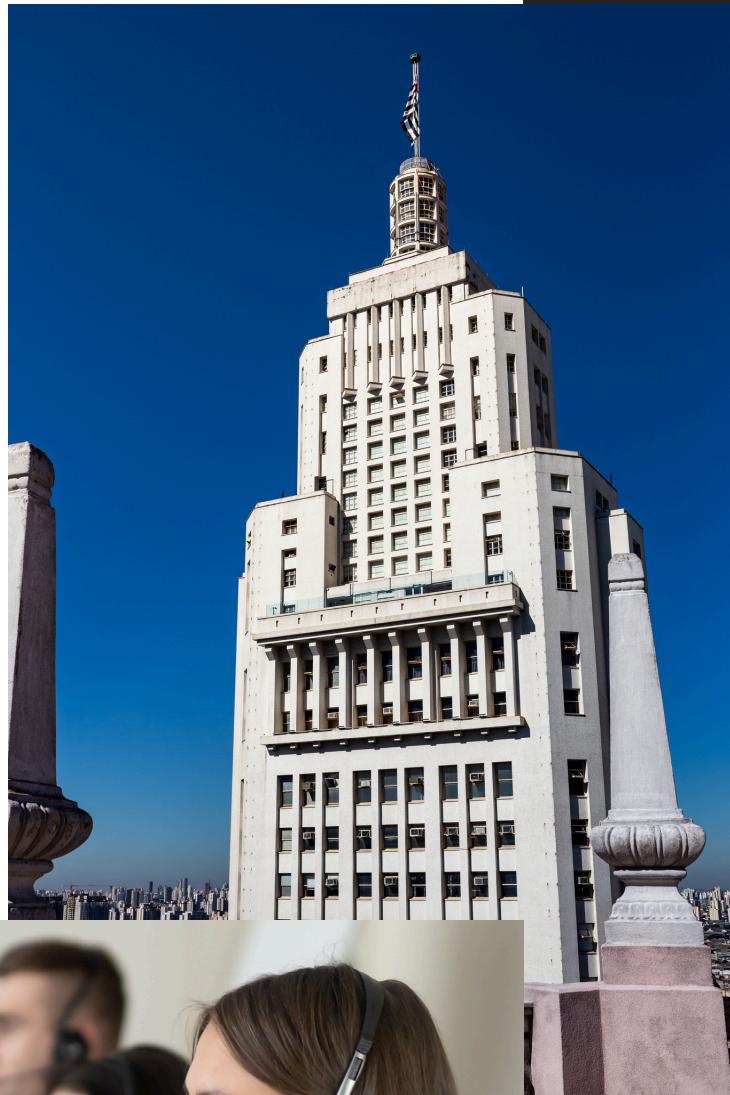
<https://github.com/Anna-Ikeda/Predicting-the-Success-of-Bank-Telemarketing-for-Selling-Long-term-Deposits.git>



# Agenda Overview

- 
- 01** Introduction of our research paper
  - 02** Introduction of the problem and the dataset
  - 03** The methodology
  - 04** Reproduced code
  - 05** Suggest improvements to the model

# Introduction of our research paper



**Predicting the Success of Bank Telemarketing  
for Selling Long-term Deposits:  
An Application of Machine Learning  
Algorithms(2021)**

Written by  
**Premkumar Borugadda**  
**Dr. Prabhakar Nandru**  
**Dr. Chendragiri Madhavaiah**

(Pondicherry University, Karaikal Campus, Karaikal - 609605, India )

[https://www.researchgate.net/publication/352755139 Predicting the Success of Bank Telem\[...\]](https://www.researchgate.net/publication/352755139)  
[erm Deposits An Application of Machine Learning Algorithms](https://www.researchgate.net/publication/352755139)

# Background of our research paper

## Globalization

- Expanded connections with customers across borders

!

## Technological development

- Improved customer reach and service quality

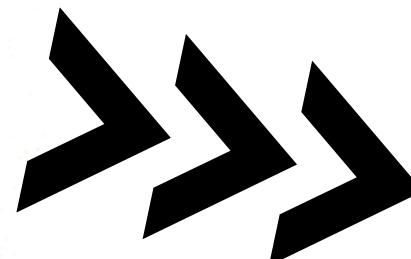
!



## Telemarketing:

A salesperson checks customers' willingness to buy their products by phone

More and more banks are using **telemarketing** as a new marketing method to compete with other banks



For prospective customers

Telemarketing helps acquire new customers

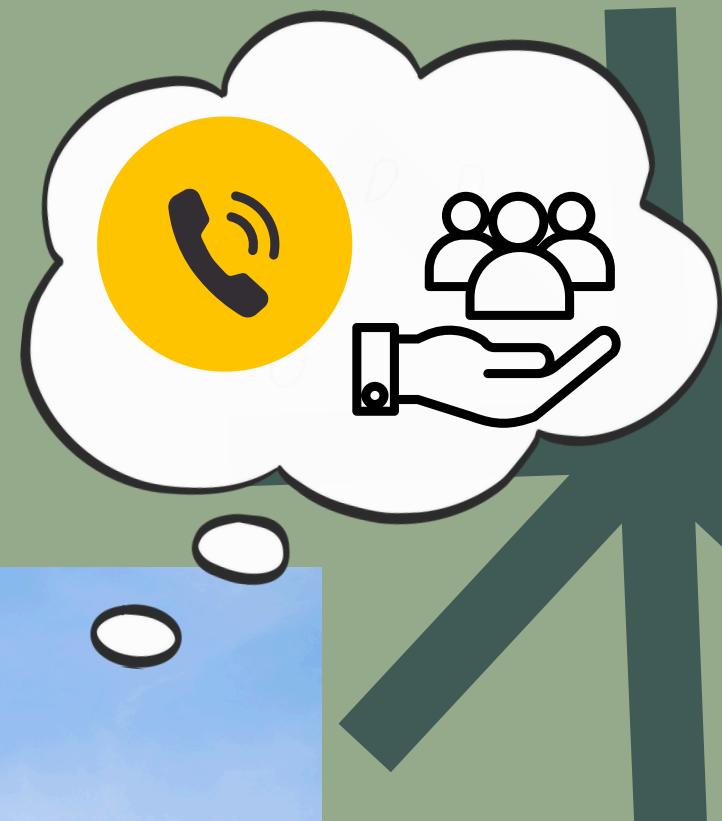
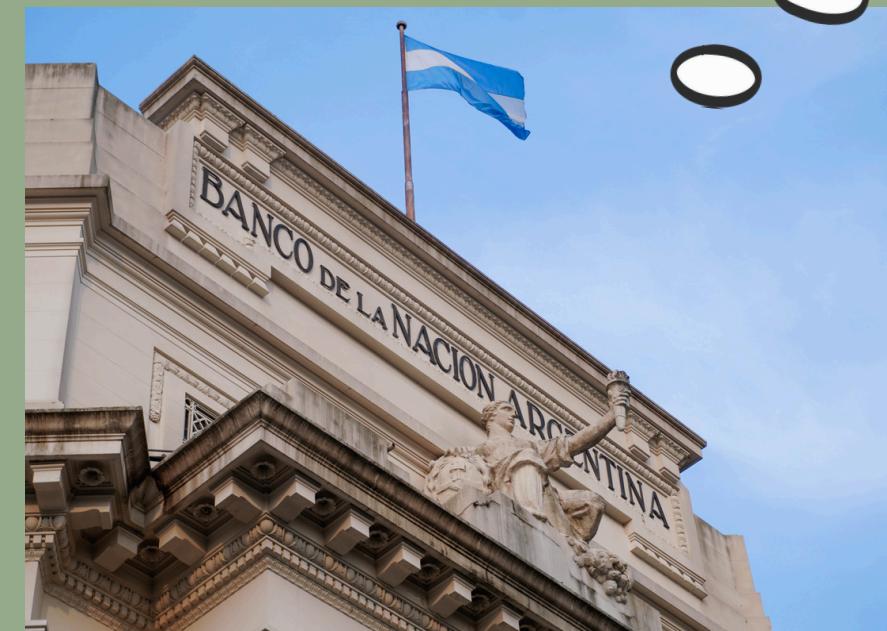
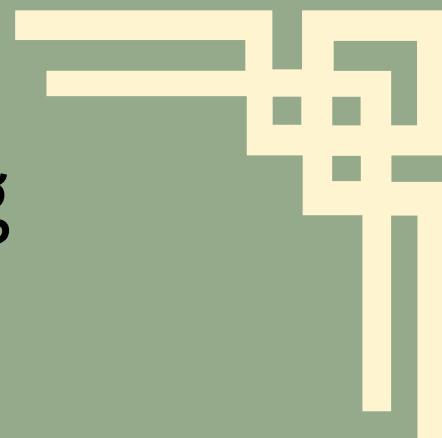
For existing customers

Telemarketing enables better, more personalized service

# Introduction of the problem being solved

**Whether telemarketing is effective  
for encouraging customers  
to subscribe to long-term bank deposits**

- Few studies have predicted telemarketing success for long-term deposits using ML
- This research provided **insightful information to banks** for improving their telemarketing strategies for long-term bank deposits



# Introduction of the dataset

## Overview

- Our dataset is related with **direct marketing campaigns (phone calls) of a Portuguese banking institution**
- It collected from the UC Irvine Machine Learning Repository

<https://archive.ics.uci.edu/dataset/222/bank+marketing>

## Number of Variables

- 21 features

## Number of Rows

- 4,119 instances

## Target and descriptive features

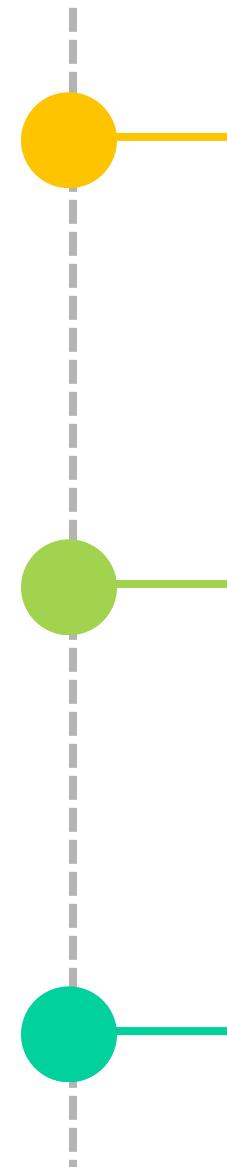
- Target feature: "y (Label)"
- Descriptive features: The rest of the features

**Table 1. Features of the dataset**

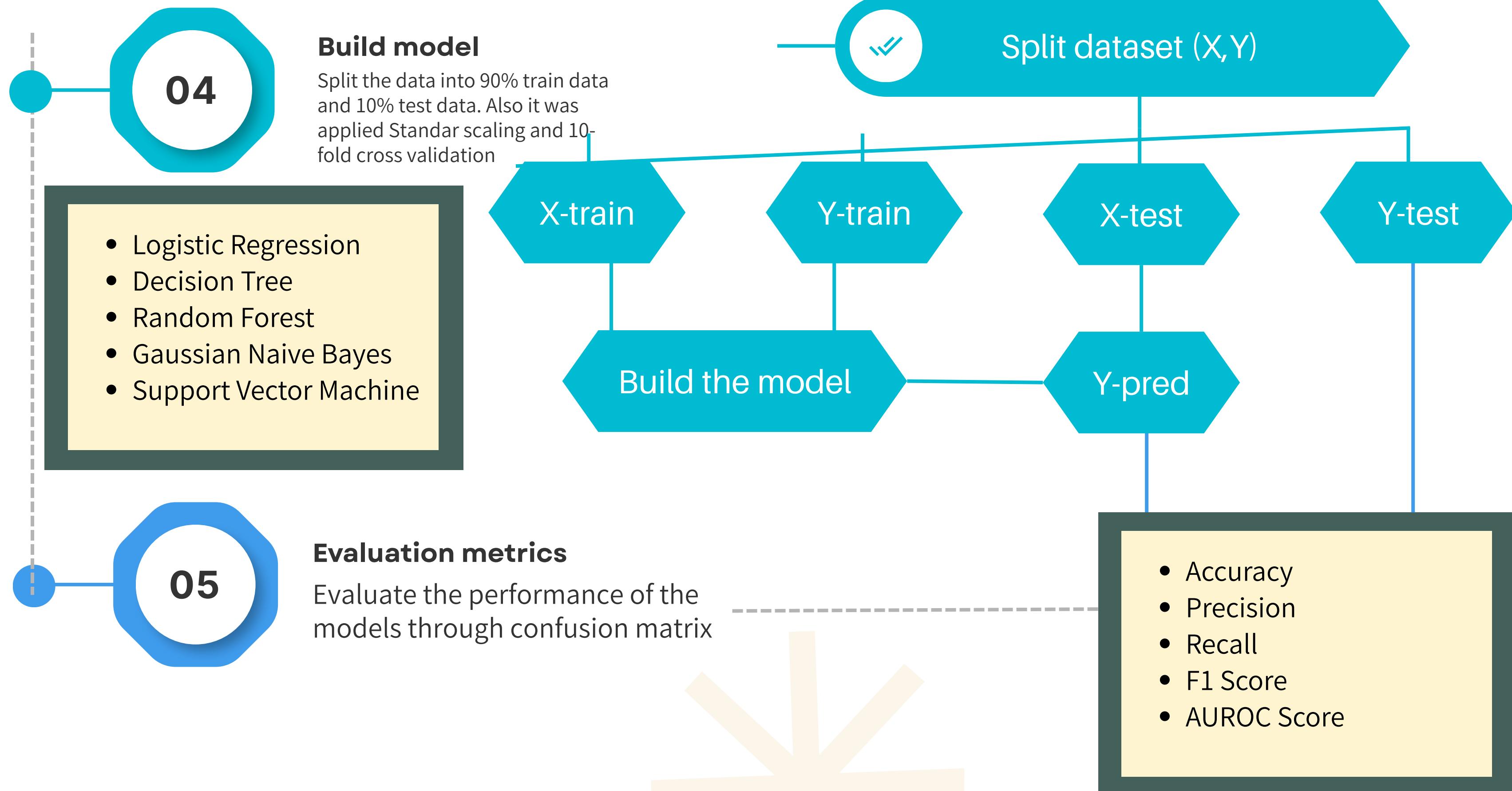
S. No	Features	Description of Features
1	Age	Age of the client
2	Job	Type of client's job
3	Marital	Client's marital status
4	Education	Highest education of the client
5	Default	Client credit
6	Housing	Housing loan
7	Loan	Personal loan
8	Contact	Type of contact communication with the client
9	Month	Last month of the year contracting to the client
10	Day of week	Last day of the week contracting to the client
11	Duration	Duration of client contact
12	Campaign	Number of contacts performed during the campaign and for this client
13	Plays	Number of days elapsed after the client's last visit
	Previous	Number of contacts performed before this campaign and for this client
14	Poutcome	Outcome of the previous marketing campaign
15	Emp.var.rate	Employment variation rate
16	Cos.price.IDX	Consumer price index
17	Cons.conf.IDX	Consumer confidence index
18	Euribor3m	Euribor 3-month rate
19	Nr. employed	Number of employees
20	Label	Client subscription

*Source:* <https://archive.ics.uci.edu/ml/datasets/Bank+Marketing>

# Methodology

- 
- 01**  
**Data Acquisition**  
UCI Machine Learning Repository.  
21 attributes  
4119 records
  - 02**  
**Exploratory data analysis**  
Check data's shape  
Unique labels of the target variable.  
Features data types  
Data's descriptive statistics
  - 03**  
**Feature engineering**  
Convert categorical variable into numeric  
Apply the standard scaler technique

# Methodology



# Reproduced code

## Decision Tree - (Model evaluation and confusion matrix)

The Result from our reproduced code

```
evaluate_results("Decision Tree", y_test, y_pred_dt, y_prob_dt)
✓ [33] 112ms

Accuracy      : 90.29%
Precision     : 58.62%
Recall        : 37.78%
F1 Score      : 45.95%
AUC Score     : 87.90%
Confusion Matrix:
[[355  12]
 [ 28 17]]
```

The Result from the research paper

**Table 2. The performance comparison of five classification machine learning models**

S. No.	Classification Models	Evolution Metrics of Classification Models				
		Accuracy	Precision	Recall	F1-Score	AUROC -score
1.	LR	92.72	70.58	54.54	61.53	93.62
2.	GNB	85.0	38.55	72.72	50.39	84.76
3.	RF	91	59.37	43.18	50.0	93.53
4.	SVM	90.53	58.62	38.63	46.57	91.4
5.	DT	91.0	56.75	47.72	51.85	71.69

*Source:* Jupyter notebook output

The Decision Tree model achieved a solid accuracy of 90.29%, showing it predicts the overall outcome well. However, its precision (58.62%) and recall (37.78%) suggest it struggles to correctly identify actual positive cases, often missing them. The F1 score (45.95%) confirms this trade-off between precision and recall. Still, the AUC score of 87.90% shows it does a good job separating the two classes. The confusion matrix highlights that while the model correctly predicted 355 "no" cases, it only identified 17 true "yes" cases and missed 28, indicating a need for improvement if we want better detection of subscribers.

# Reproduced code

## Decision Tree - Classification report

The Result from our reproduced code

Classification Report:				
	precision	recall	f1-score	support
No	0.93	0.97	0.95	367
Yes	0.59	0.38	0.46	45
accuracy			0.90	412
macro avg	0.76	0.67	0.70	412
weighted avg	0.89	0.90	0.89	412

The Result from the research paper  
**Classification Report of DT:**

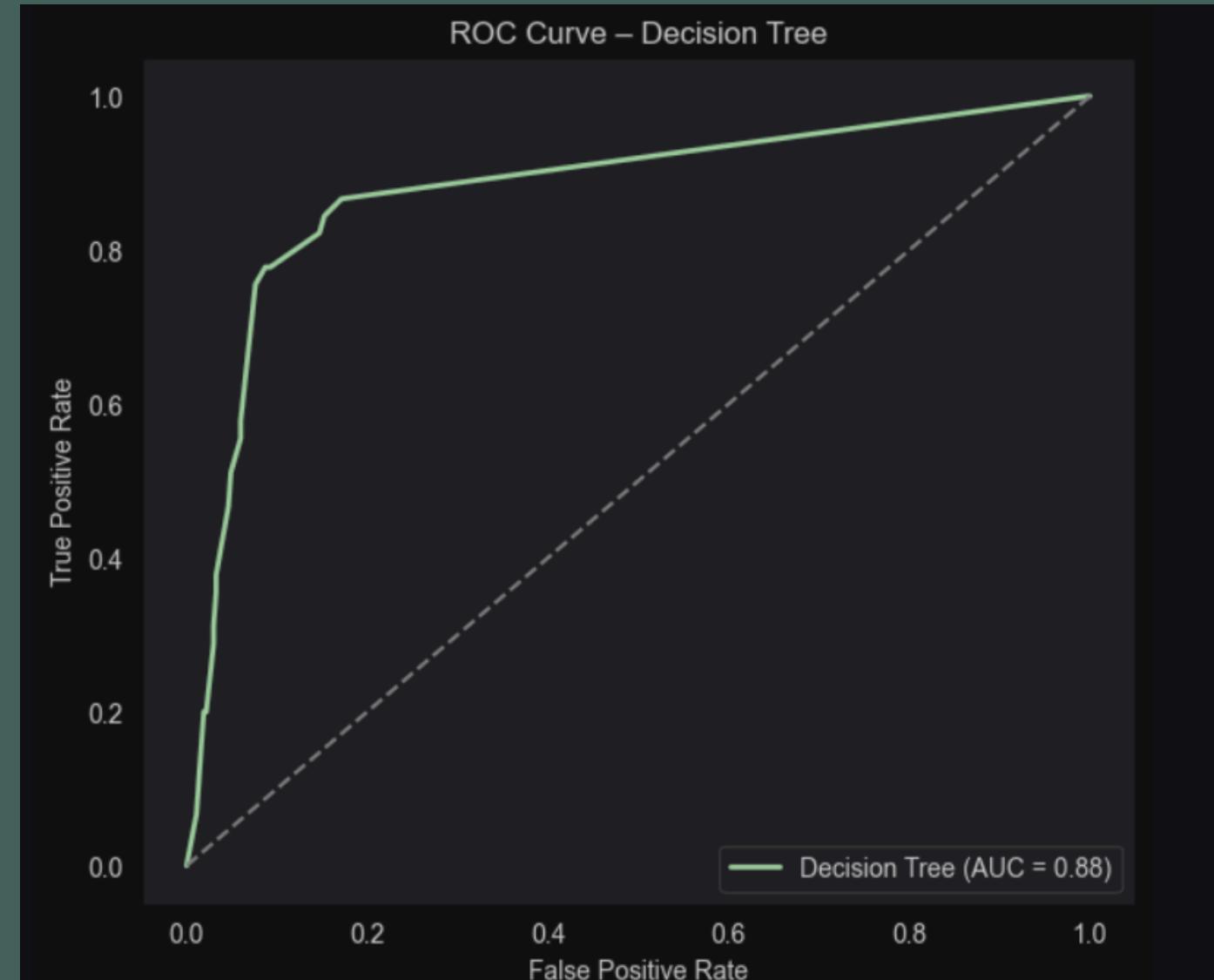
	precision	recall	f1-score	support
0	0.9387	0.9565	0.9475	368
1	0.5676	0.4773	0.5185	44
accuracy			0.9053	412
macro avg	0.7531	0.7169	0.7330	412
weighted avg	0.8990	0.9053	0.9017	412

The model shows strong performance for predicting the "No" class (not subscribing), with high precision (93%) and recall (97%). However, for the "Yes" class (subscribing), both precision (59%) and recall (38%) are significantly lower, meaning many actual positive cases are missed. Overall accuracy is solid at 90%, but the model is biased toward the majority class and struggles to detect the minority class effectively. This suggests a class imbalance issue that could be addressed in future model improvements.

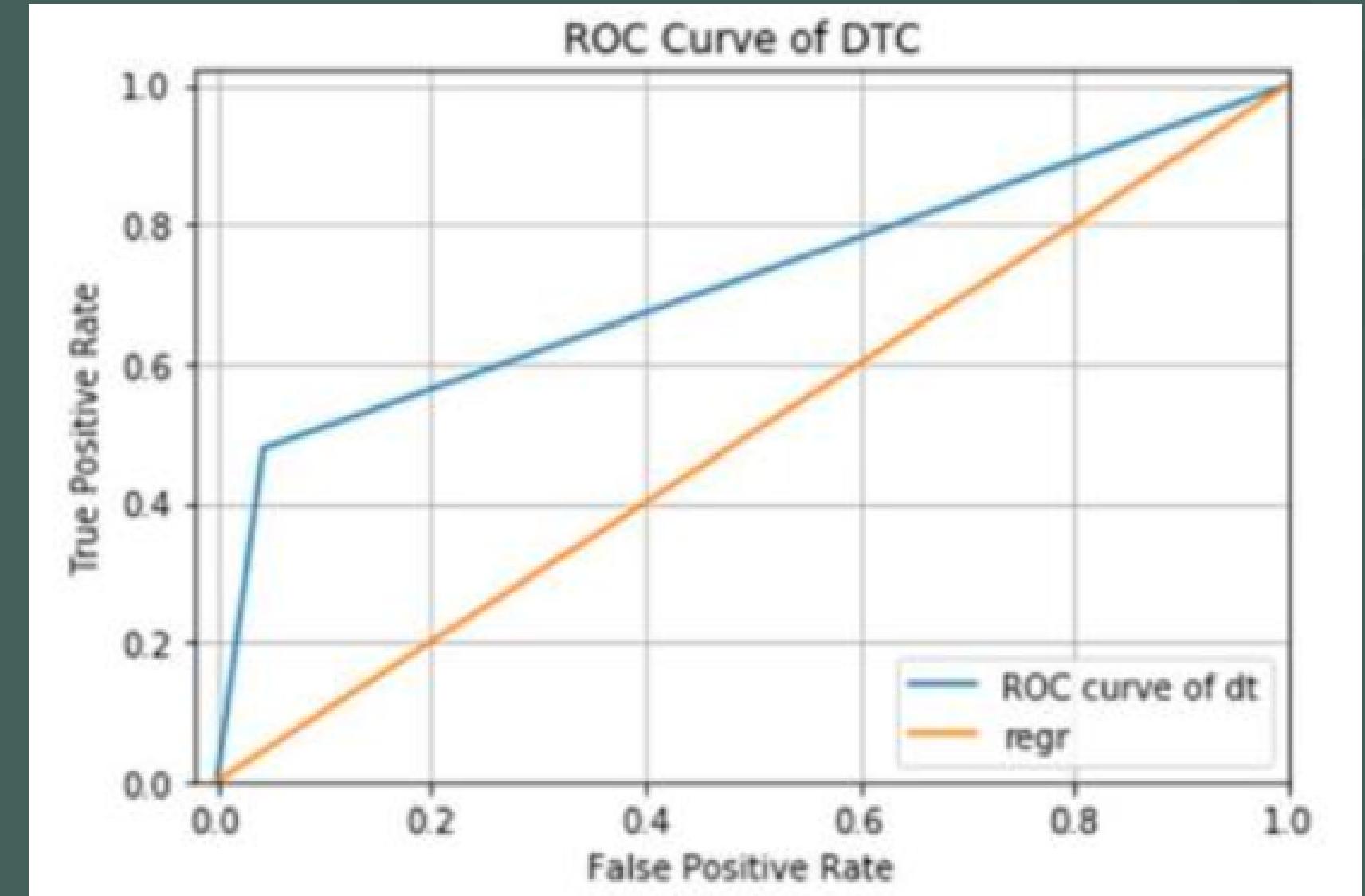
# Reproduced code

## Decision Tree ROC Curve

The Result from our reproduced code



The Result from the research paper



The ROC curve for the Decision Tree model shows a strong ability to distinguish between the two classes, with an AUC score of 0.88. This indicates high overall performance, as the curve stays well above the diagonal baseline, meaning the model correctly separates positive and negative classes much better than random guessing. However, while AUC is high, other metrics like recall for the "Yes" class suggest the model may still miss some true positives.

# Reproduced code

## Logistic Regression - (Model evaluation and confusion matrix)

The Result from our reproduced code

```
evaluate_results("Logistic Regression", y_test, y_pred_lr, y_pr
✓ [32] 88ms

Accuracy      : 91.99%
Precision     : 75.00%
Recall        : 40.00%
F1 Score      : 52.17%
AUC Score     : 93.06%
Confusion Matrix:
[[361  6]
 [ 27 18]]
```

The Result from the research paper

**Table 2. The performance comparison of five classification machine learning models**

S. No.	Classification Models	Evolution Metrics of Classification Models				
		Accuracy	Precision	Recall	F1-Score	AUROC -score
1.	LR	92.72	70.58	54.54	61.53	93.62
2.	GNB	85.0	38.55	72.72	50.39	84.76
3.	RF	91	59.37	43.18	50.0	93.53
4.	SVM	90.53	58.62	38.63	46.57	91.4
5.	DT	91.0	56.75	47.72	51.85	71.69

*Source:* Jupyter notebook output

The Logistic Regression model achieved a high accuracy of 91.99% and an impressive AUC score of 93.06%, showing strong overall classification performance. It correctly identified the "Yes" class with 75% precision, though the recall is only 40%, meaning it missed some actual positives. The F1 score of 52.17% reflects this trade-off. The confusion matrix shows 18 true positives, 27 false negatives, and only 6 false positives, indicating that while the model is precise, it could improve in capturing more of the positive cases.

# Reproduced code

## Classification Report of Logistic Regression

The Result from our reproduced code

Classification Report:				
	precision	recall	f1-score	support
No	0.93	0.98	0.96	367
Yes	0.75	0.40	0.52	45
accuracy			0.92	412
macro avg	0.84	0.69	0.74	412
weighted avg	0.91	0.92	0.91	412

The Result from the research paper

## Classification Report of Logistic Regression:

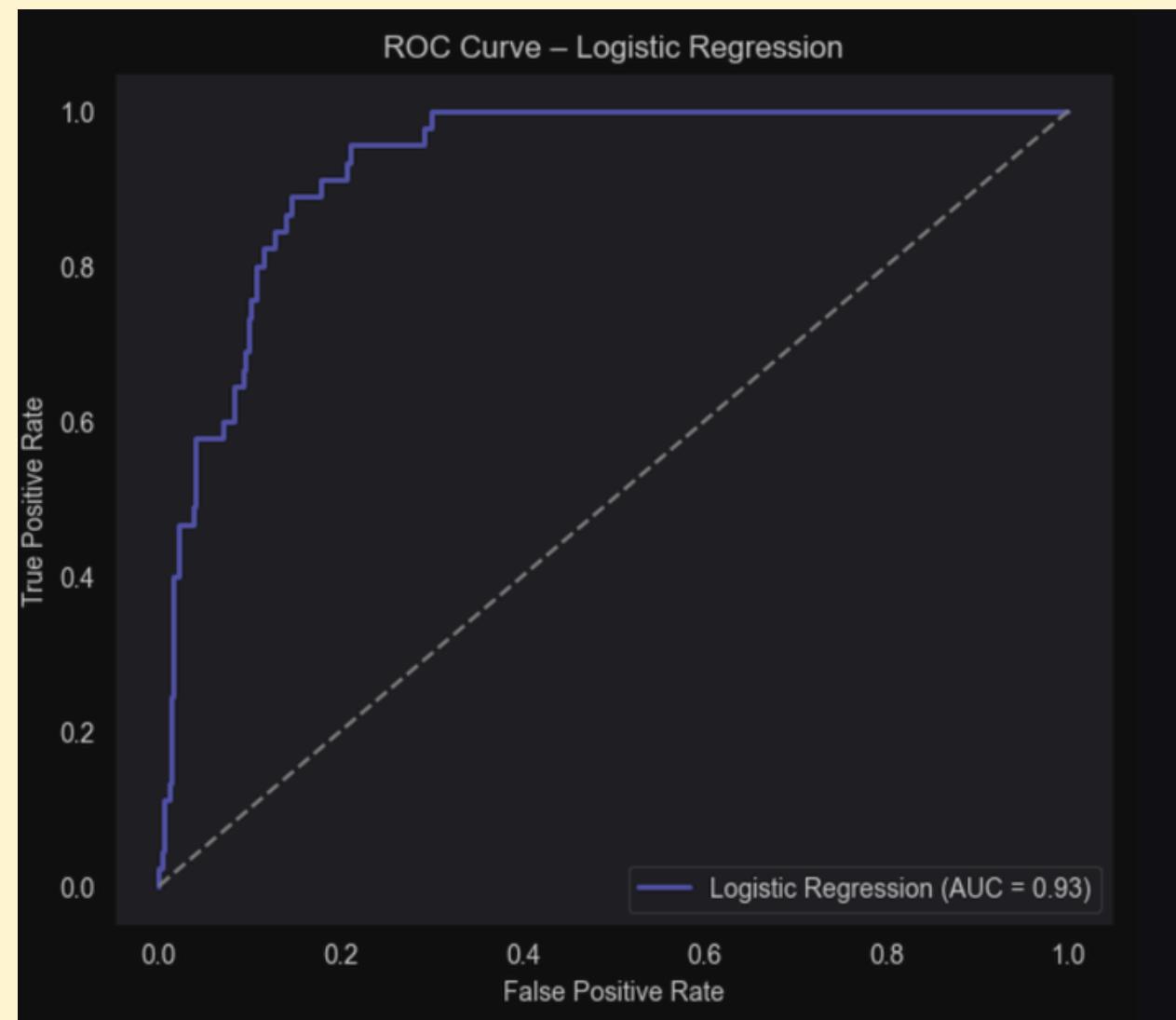
	precision	recall	f1-score	support
0	0.9471	0.9728	0.9598	368
1	0.7059	0.5455	0.6154	44
accuracy			0.9272	412
macro avg	0.8265	0.7591	0.7876	412
weighted avg	0.9213	0.9272	0.9230	412

The classification report shows that the model performs very well in predicting the "No" class (customers who won't subscribe), with high precision (93%) and recall (98%). However, for the "Yes" class (customers who will subscribe), precision drops to 75% and recall to 40%, meaning it misses many true positives. The overall accuracy is strong at 92%, but the imbalance in class performance indicates the model is better at detecting non-subscribers than actual subscribers.

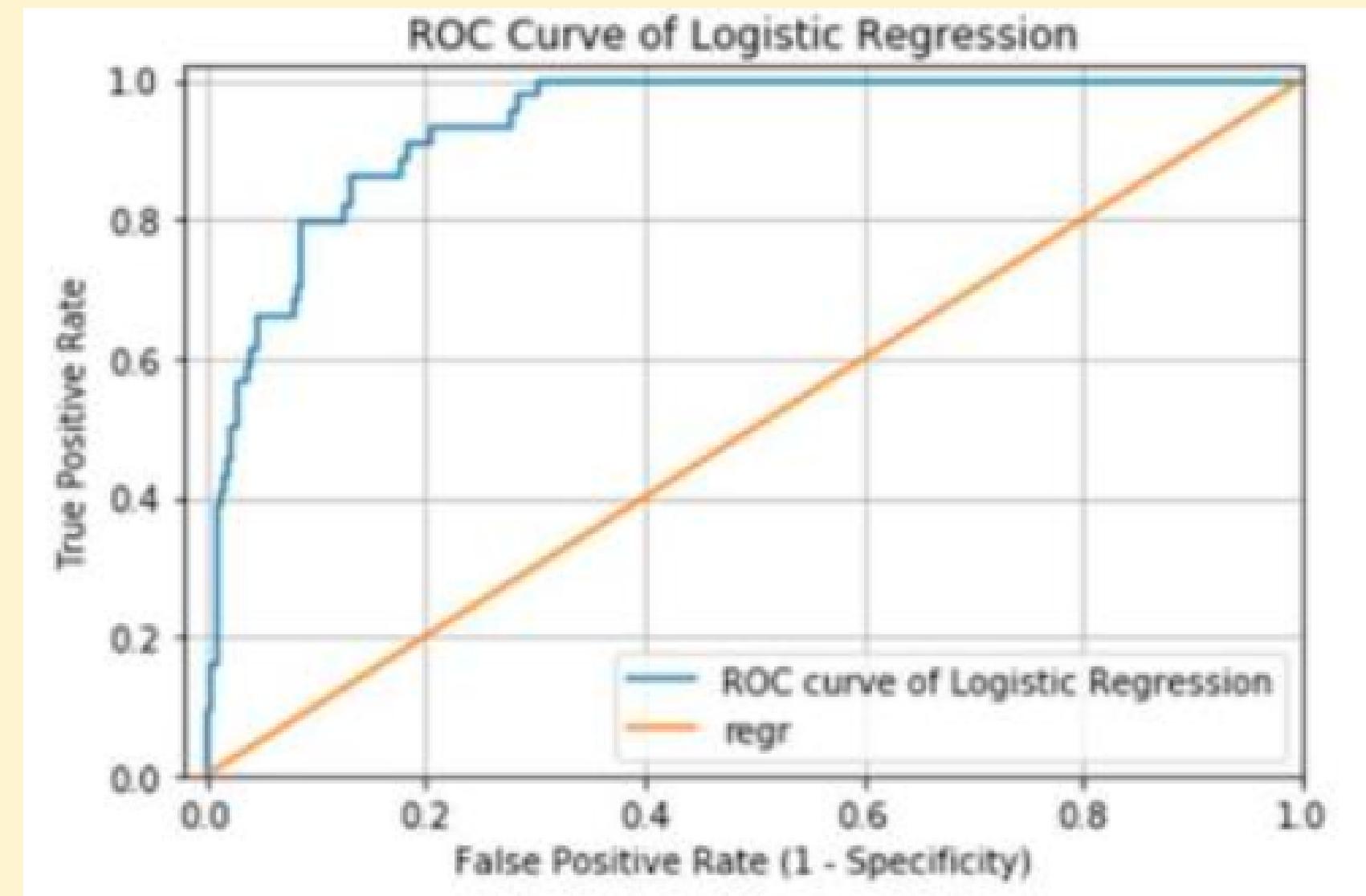
# Reproduced code

## ROC Curve of Logistic Regression

The Result from our reproduced code



The Result from the research paper



The ROC Curve for the Logistic Regression model shows strong performance, with an AUC score of 0.93, meaning the model has a high ability to distinguish between the two classes. The curve hugs the top-left corner, which indicates a good balance of high true positive rate and low false positive rate—ideal for a binary classifier.

# Suggest improvements to the model

**01**

## Preprocessing

- Use One Hot Encoder instead of get\_dummies
- Scale only numeric features, and don't scale before tree training.

**02**

## Models

- Use pipelines to avoid leakage
- Add class\_weight = 'balanced'



# Thank You

