BANK MARKETING ANALYTICS

Bank Marketing Classification

Moataz Samara, DS 301

Research Paper & Problem

Statement on Predictions

This project references the **Moro et al. (2014)** paper, focusing on predicting term deposit subscriptions through bank telemarketing, aiming to classify clients' likelihood of subscription effectively.

Importance of This Issue

Enhancing Telemarketing Efficiency

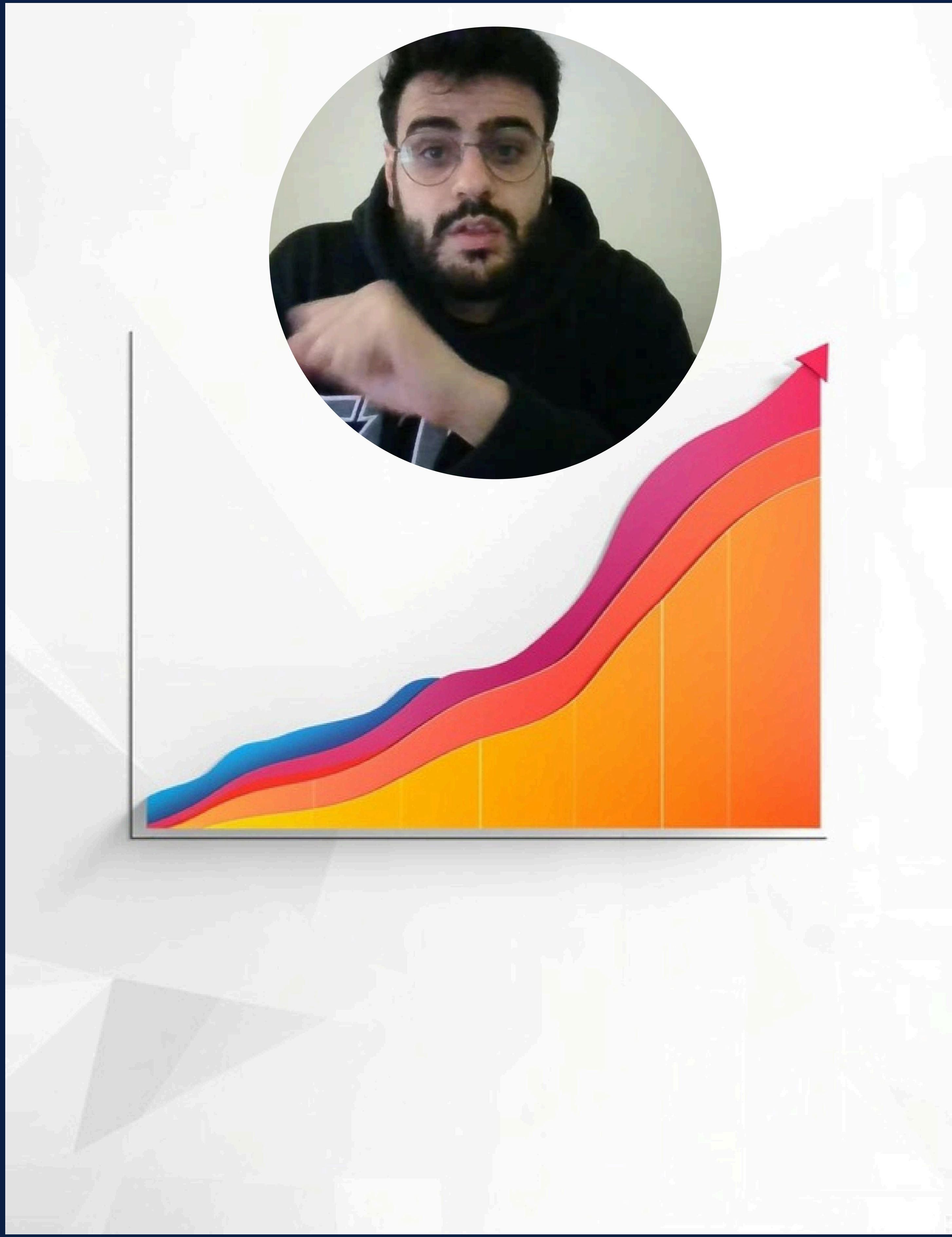
The challenges of high telemarketing costs and low subscription rates necessitate effective models. By prioritizing calls, banks can reduce waste and improve ROI significantly, optimizing their marketing strategies.

Dataset Overview

UCI Bank Marketing Dataset

The UCI Bank Marketing dataset contains **45,211 client records** used for predicting term deposit subscriptions. It includes various feature categories such as demographics, contact information, and campaign history.

Column 1 (Feature group)	Column 2 (Example columns)
Client / Demographic	age, job, marital, education
Financial status	default, housing, loan, balance
Contact details	contact, month, day_of_week
Campaign history	campaign, pdays, previous, poutcome
Economic indicators	emp.var.rate, cons.price.idx, euribor3m, nr.employed



Data Preprocessing Steps

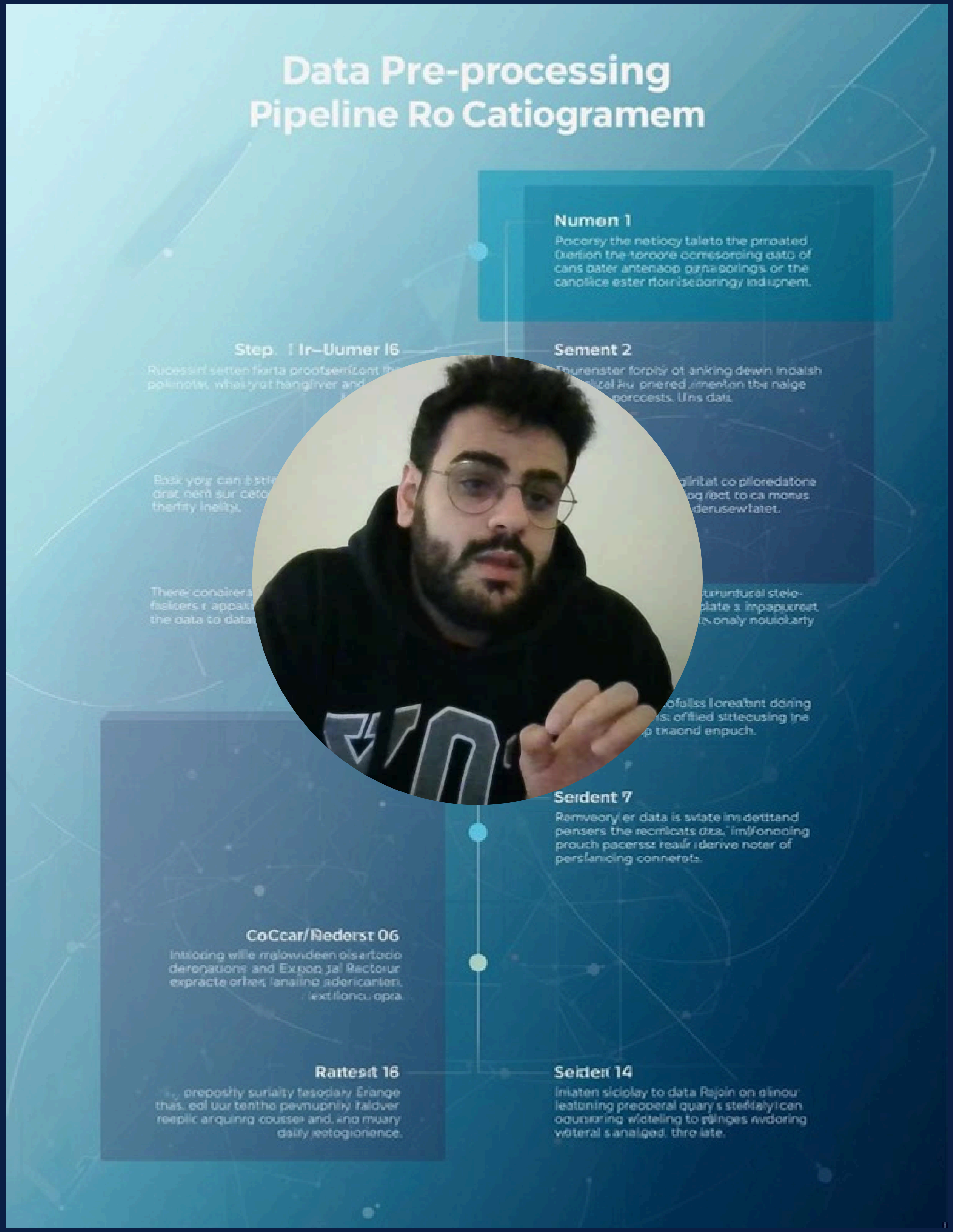
Essential Preparation Workflow

In this phase, we make a copy of the dataset, eliminate the **duration** column to mitigate data leakage risks, and perform a **stratified train/test split** of 70/30.

Preprocessing Pipeline

Data Transformation Steps

The preprocessing pipeline involves transforming numeric columns using **StandardScaler** and categorical columns through **OneHotEncoder**, ensuring data is ready for modeling and analysis.



Algorithms Overview

Logistic Regression vs Decision Trees

This study employs **Logistic Regression** for linear classification and **Decision Trees** for non-linear predictions, highlighting their unique strengths in handling binary outcomes and interpretability in model outputs.

Code Approach

train_and_evaluate() Workflow

The **train_and_evaluate() function** streamlines the model training process, integrating preprocessing with model fitting, and evaluating accuracy using essential metrics to inform performance.

```python
def train_and_evaluate(model, model_name):
    """
    Fit a pipeline (preprocessing + model),
    print metrics, and return the fitted pipeline, probabilities and ROC-AU
    """
    print(f"\n{'='*70}")
    print(f"Model: {model_name}")
    print(f"{'='*70}\n")

    # Full pipeline: preprocessing + model
    pipe = Pipeline(
        steps=[
            ("preprocess", preprocessor),
            ("model", model)
        ]
    )

    # Fit the model
    pipe.fit(X_train, y_train)

    # Predictions
    y_pred = pipe.predict(X_test)
    y_prob = pipe.predict_proba(X_test)[:, 1]  # prob. for class 1 (y = 1)

    # Metrics
    acc = accuracy_score(y_test, y_pred)
    prec = precision_score(y_test, y_pred)
    rec = recall_score(y_test, y_pred)
    f1 = f1_score(y_test, y_pred)
    roc_auc = roc_auc_score(y_test, y_prob)

    print(f"Accuracy     : {acc:.4f}")
    print(f"Precision    : {prec:.4f}")
    print(f"Recall       : {rec:.4f}")
    print(f"F1-score     : {f1:.4f}")
    print(f"ROC-AUC      : {roc_auc:.4f}")

    print("\nClassification report:\n")
    print(classification_report(y_test, y_pred, digits=4))
```

```python
    # Confusion Matrix
    cm = confusion_matrix(y_test, y_pred)
    disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=["no", "yes"])
    disp.plot(values_format="d")
    plt.title(f"Confusion Matrix - {model_name}")
    plt.show()

    # ROC Curve
    fpr, tpr, thresholds = roc_curve(y_test, y_prob)
    plt.plot(fpr, tpr, label=f"{model_name} (AUC = {roc_auc:.3f})")
    plt.plot([0, 1], [0, 1], linestyle="--", label="Random")
    plt.xlabel("False Positive Rate")
    plt.ylabel("True Positive Rate")
    plt.title(f"ROC Curve - {model_name}")
    plt.legend()
    plt.grid(True)
    plt.show()

    return pipe, y_prob, roc_auc
```

Test Metrics Comparison

Evaluating Model Performance

Logistic Regression

Logistic Regression achieved an **accuracy of 75.6%**, demonstrating reliable identification of potential subscribers while maintaining a strong balance between precision and recall metrics.

Decision Tree

The Decision Tree model recorded an **accuracy of 84.2%**, but its precision and recall were lower, indicating possible class imbalance may be affecting its overall usefulness.

Model Insights

While the Decision Tree showed higher accuracy, **Logistic Regression is preferred** for this task due to its better recall and ROC-AUC, making it more effective for prioritizing calls.

Interpretation of Results

Insights on Model Performance

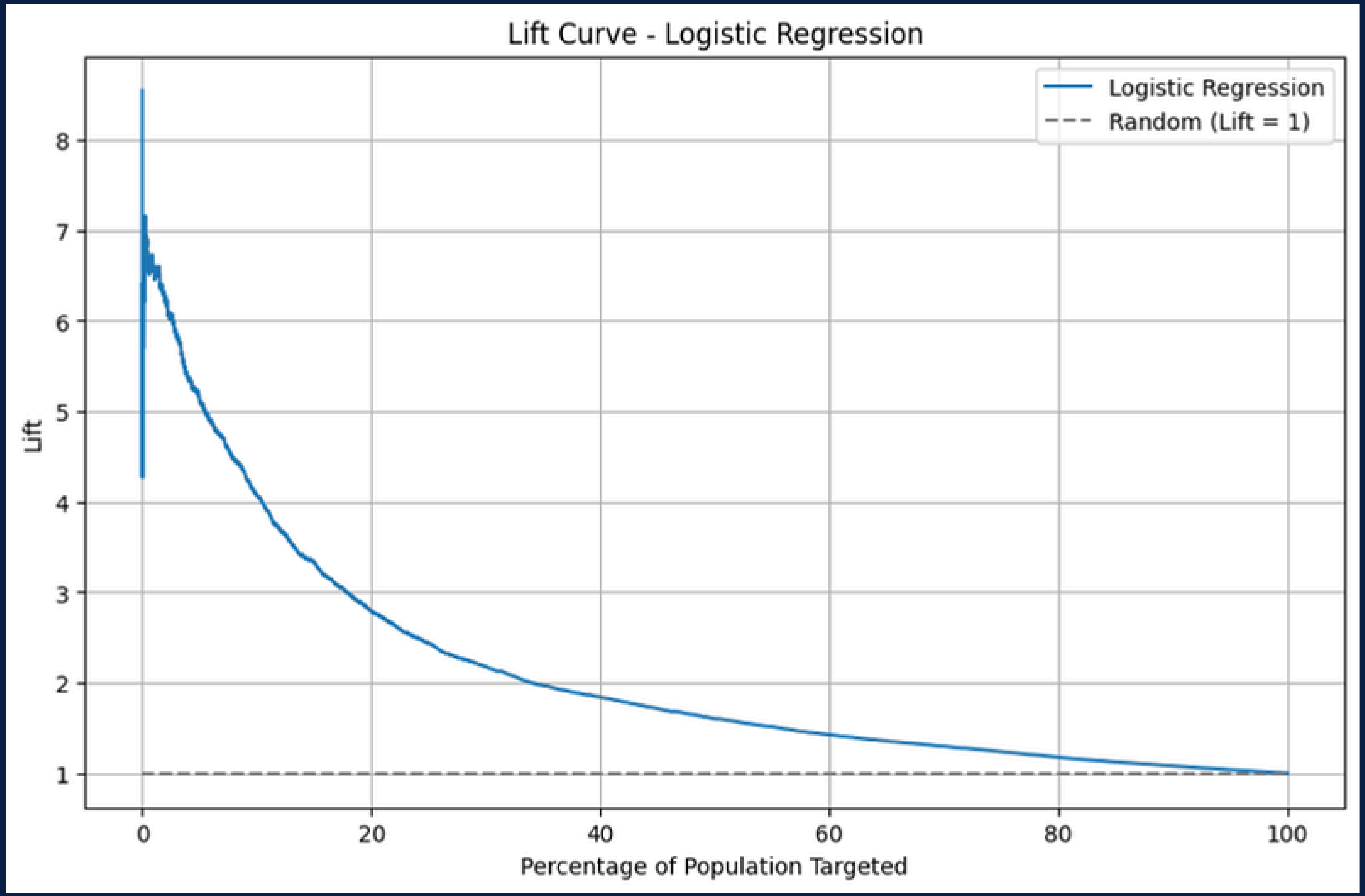
Logistic Regression proves more effective in identifying potential subscribers due to its higher recall and ROC-AUC metrics, while the Decision Tree's accuracy improvement stems from class imbalance bias.

Model	Accuracy	Precision	Recall	F1-score	ROC-AUC
Logistic Regression	75.60%	26.80%	62.60%	37.60%	0.77
Decision Tree	84.20%	32.10%	31.70%	31.90%	0.61

Lift Curve Explained

Improving Client Contact Efficiency

By sorting clients based on predicted subscription probabilities, contacting the top X% first enhances efficiency and maximizes marketing campaign effectiveness.



Suggested Improvements

Enhancing Model Performance

To optimize results, we recommend **hyperparameter tuning** for models, exploring additional algorithms such as Neural Networks and Random Forest, and implementing cross-validation for more robust evaluations.

Project Recap and Insights

Key Findings and Achievements

This project successfully implemented a comprehensive machine learning pipeline, demonstrating that **Logistic Regression** is the most effective model for predicting term deposit subscription based on recall and ROC-AUC metrics.

BANK MARKETING ANALYTICS

THANK YOU!