STAT 451 Project Proposal

Project Title:

Comparison of Logistic Regression vs Random Forests for Predicting Bank Marketing Success

Introduction and Objective:

The goal of this project is to explore whether logistic regression provides a better predictive classification model compared to tree-based models, such as Random Forests, for the Bank Marketing dataset. We will assess the predictive performance of both models in terms of accuracy, interpretability, and computational efficiency.

Dataset Description:

The Bank Marketing dataset, obtained from the UCI Machine Learning Repository, contains data related to direct marketing campaigns (phone calls) of a Portuguese banking institution. The dataset consists of various features, including socio-economic attributes and contact information, with the target variable y indicating whether a client subscribed to a term deposit.

Research Questions:

1. To what extent is logistic regression a better predictor of classification compared to a Random Forest model?
2. How do preprocessing steps such as standardization, one-hot encoding, and regularization affect the predictive performance of these models?

Variables and Preprocessing Steps:

- **Target Variable:** ***y*** (binary classification: "yes" or "no")
- **Independent Variables:** Multiple socio-economic and campaign-related attributes (age, job, marital status, etc.)
- **Preprocessing:**
    - Standardization vs. Normalization of numerical variables.
    - One-hot encoding of categorical variables.
    - Feature selection using Lasso and Ridge regression.
    - Application of L2 and L1 regularization to control overfitting in logistic regression.

Methods and Approach:

1. **Data Preprocessing and Exploration:**
    - Visual exploration of numerical and categorical columns (e.g., distribution plots, histograms).
    - Preprocessing steps such as missing value imputation, encoding categorical variables, and scaling numerical features.

2. **Model Development:**
    - Implementation of a logistic regression model with L1 and L2 regularization.
    - Implementation of a Random Forest model with hyperparameter tuning.
3. **Evaluation Metrics:**
    - Confusion Matrix.
    - F1 Score, Accuracy, AUC-ROC plots.
    - Feature importance visualization for Random Forest.
4. **Visualization:**
    - Visual comparison of model performance metrics.

Code:

In [1]:
```
# Importing necessary libraries
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, roc_auc_score, confusion_
from ucimlrepo import fetch_ucirepo



# Load data

bank_marketing = fetch_ucirepo(id=222)
X = bank_marketing.data.features
y = bank_marketing.data.targets
df = pd.concat([X,y], axis = 1)

# Initial data overview
print(df.head())
print(df.info())

# Data preprocessing (example snippet)
X = df.drop(columns=['y'])
y = df['y'].apply(lambda x: 1 if x == 'yes' else 0)  # Binary encoding for t
X_encoded = pd.get_dummies(X, drop_first=True)  # One-hot encoding for categ

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X_encoded, y, test_size=

# Model training (example snippet)
log_reg = LogisticRegression(max_iter=3000)
log_reg.fit(X_train, y_train)

rf_model = RandomForestClassifier(n_estimators=300, random_state=42)
rf_model.fit(X_train, y_train)

# Model evaluation
y_pred_logreg = log_reg.predict(X_test)
y_pred_rf = rf_model.predict(X_test)

print("Logistic Regression Classification Report:\n", classification_report(
print("Random Forest Classification Report:\n", classification_report(y_test
```

```
   age           job  marital  education default  balance housing loan  \
0   58    management  married   tertiary      no     2143     yes   no   
1   44    technician   single  secondary      no       29     yes   no   
2   33  entrepreneur  married  secondary      no        2     yes  yes   
3   47   blue-collar  married        NaN      no     1506     yes   no   
4   33           NaN   single        NaN      no        1      no   no   

  contact  day_of_week month  duration  campaign  pdays  previous poutcome  
y  
0     NaN            5   may       261         1     -1         0      NaN  
no  
1     NaN            5   may       151         1     -1         0      NaN  
no  
2     NaN            5   may        76         1     -1         0      NaN  
no  
3     NaN            5   may        92         1     -1         0      NaN  
no  
4     NaN            5   may       198         1     -1         0      NaN  
no  
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 45211 entries, 0 to 45210
Data columns (total 17 columns):
 #   Column       Non-Null Count  Dtype 
---  ------       --------------  ----- 
 0   age          45211 non-null  int64 
 1   job          44923 non-null  object
 2   marital      45211 non-null  object
 3   education    43354 non-null  object
 4   default      45211 non-null  object
 5   balance      45211 non-null  int64 
 6   housing      45211 non-null  object
 7   loan         45211 non-null  object
 8   contact      32191 non-null  object
 9   day_of_week  45211 non-null  int64 
 10  month        45211 non-null  object
 11  duration     45211 non-null  int64 
 12  campaign     45211 non-null  int64 
 13  pdays        45211 non-null  int64 
 14  previous     45211 non-null  int64 
 15  poutcome     8252 non-null   object
 16  y            45211 non-null  object
dtypes: int64(7), object(10)
memory usage: 5.9+ MB
None
```

```
/opt/anaconda3/lib/python3.12/site-packages/sklearn/linear_model/_logistic.p
y:469: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regre
ssion
  n_iter_i = _check_optimize_result(
```

Logistic Regression Classification Report:

	precision	recall	f1-score	support
0	0.92	0.97	0.94	11966
1	0.64	0.34	0.44	1598
accuracy			0.90	13564
macro avg	0.78	0.66	0.69	13564
weighted avg	0.88	0.90	0.89	13564

Random Forest Classification Report:

	precision	recall	f1-score	support
0	0.92	0.97	0.95	11966
1	0.67	0.40	0.50	1598
accuracy			0.91	13564
macro avg	0.79	0.69	0.72	13564
weighted avg	0.89	0.91	0.90	13564