Bank Marketing Analysis: Term Deposit Predictions

Summary

Analyzed 41,188 records from a Portuguese bank's marketing campaign. Goal: predict who subscribes to term deposits. Found severe class imbalance (7.88:1), identified key predictive patterns, and recommend XGBoost for implementation.

EDA Findings

The dataset has 20 features plus the target variable. Only 11.27% of customers subscribed - that's our main challenge. No actual missing values, but 20.87% of credit default statuses are marked 'unknown'. This isn't random - it's customers choosing not to share financial info.

Outliers exist in campaign contacts (up to 56 attempts) and call duration (up to 4,918 seconds). These aren't errors - they're real edge cases of persistent sales efforts. Keep them for model robustness.

Economic indicators are highly correlated (emp.var.rate and euribor3m at 0.972). Makes sense - they all measure the same economic climate. Subscribers experienced vastly different economic conditions: employment variation of -1.23 vs 0.25 for non-subscribers. That's a 600% difference.

The killer insight: customers with previous campaign success convert at 65.11% vs 8.83% baseline. Previous success is our strongest predictor. This changes everything - focus on warm leads, not cold calling.

Duration correlates strongly with outcome (0.405) but it's useless - you don't know call length until after the call. Classic data leakage. Drop it for modeling, keep it for post-campaign analysis.

Algorithm Selection

Tested four algorithms against our constraints:

XGBoost (9.5/10) - Winner - Handles imbalance natively with scale_pos_weight=7.88 - Captures non-linear patterns between age, contacts, and economics - Feature importance for business interpretability - Robust to outliers without preprocessing - Proven in banking applications

Random Forest (8.5/10) - Runner-up
- Good with non-linear relationships - Parallel training for speed - But struggles with severe imbalance - Memory intensive with large forests

Logistic Regression (7/10) - Baseline - Fast and interpretable - Good for stakeholder communication - But assumes linearity (wrong for this data) - Needs extensive preprocessing

SVM (6.5/10) - Skip it - Too slow for 41k records - Kernel computations kill iteration speed

XGBoost wins. It handles our imbalance, works with mixed data types, and gives us feature importance for the business team.

If we had <1,000 records instead of 41,188? Different story. I'd use Logistic Regression with heavy regularization. Complex models overfit small datasets. Focus would shift to manual feature engineering and domain expertise.

Preprocessing Strategy

Data Cleaning: - Remove 12 duplicates - Drop duration (leakage) - Keep 'unknown' as a category - it's a signal, not noise

Feature Engineering: - Convert pdays=999 to 'never_contacted' flag - Create total_contacts = campaign + previous
- Add has_previous_success binary - Build economic_pressure index from correlated indicators - Extract quarter from month

Handle Imbalance: - XGBoost: set scale_pos_weight=7.88 - Alternative: SMOTE for synthetic samples - Metrics: F1-score and PR-AUC, not accuracy

Outlier Strategy: - Cap at 1.5 IQR boundaries - Don't remove - they're real customer behaviors

Multicollinearity: - PCA on economic indicators (keep 2-3 components) - They're measuring the same underlying economic stress

Implementation Pipeline

```
# Quick implementation
preprocessor = BankPreprocessor()
df_processed = preprocessor.transform(df)

X = df_processed.drop('y', axis=1)
y = df_processed['y'].map({'no': 0, 'yes': 1})

pipeline = ImbPipeline([
    ('preprocessor', preprocessor.create_preprocessing_pipeline()),
    ('smote', SMOTE(random_state=42)),
    ('classifier', XGBClassifier(
        scale_pos_weight=7.88,
        n_estimators=100,
        max_depth=5
    ))
])
```

Business Recommendations

Immediate Actions: 1. Target customers with previous success first - 7x better conversion 2. Time campaigns during economic uncertainty (track employment rate) 3. Quality over quantity - relationship beats volume

System Changes: - Build economic monitoring dashboard - Track customer interactions across campaigns - Implement A/B testing framework - Monitor model drift monthly

Success Metrics: - F1-score for model performance - Cost per acquisition for business impact - Campaign ROI for profitability

Conclusion

This isn't just a classification problem - it's about understanding when customers are receptive to fixed-return investments. Previous relationship success dominates all other factors. Economic timing matters more than demographics.

XGBoost with proper preprocessing handles the technical challenges. The real win comes from focusing on warm leads during economic uncertainty. That's how you turn a 11% baseline into profitable campaigns.

Next steps: implement the pipeline, validate with 2024 data, deploy with monitoring. Focus on the 65% conversion segment first for quick wins, then expand to colder segments with refined messaging.

Word Count: 792 words