POST 29
DATA SCIENCE FOR ALL

Predicting Loan Default Risk Using Machine Learning

- Predicting loan default risk is a critical task for banks and financial institutions.
- Accurate predictions enable these entities to identify high-risk loan applicants, mitigate financial losses, and enhance decision-making processes.
- This article provides a comprehensive guide to building a classification model using Python and machine learning techniques to predict loan default risk.

DineshLal

Dataset and Prerequisites

Before we dive into the model, ensure you have the following:

- **Dataset:** A dataset containing historical loan data with features like:
  - Applicant's financial details (income, debt, assets).
  - Credit history (credit score, payment history).
  - Loan details (amount, duration, interest rate).
  - Target variable: Default status (0 for no default, 1 for default).
- **Libraries:** Python packages required:


```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import roc_auc_score, classification_report
```

Step 1: Data Preprocessing

1.1 Load and Inspect the Data


```
# Load the dataset
data = pd.read_csv('loan_data.csv')

# Inspect the dataset
print(data.head())
print(data.info())
```

1.2 Handle Missing Values


```
# Fill missing values or drop rows/columns with excessive missingness
data.fillna(data.median(), inplace=True)
```

Step 1: Data Preprocessing

1.3 Encode Categorical Variables


```
# Convert categorical features to numerical using one-hot encoding
data = pd.get_dummies(data, drop_first=True)
```

1.4 Feature Scaling


```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
data_scaled = scaler.fit_transform(data.drop('default_status', axis=1))
```

Step 1: Data Preprocessing

1.5 Train-Test Split


```
X = data_scaled
y = data['default_status']

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.3, random_state=42)
```

Step 2: Model Building

2.1 Logistic Regression


```
# Train the Logistic Regression model
logreg = LogisticRegression()
logreg.fit(X_train, y_train)
```

```
# Predictions and evaluation
y_pred_lr = logreg.predict(X_test)
roc_auc_lr = roc_auc_score(y_test, logreg.predict_proba(X_test)
[:, 1])
print(f"Logistic Regression ROC-AUC: {roc_auc_lr}")
```

Step 2: Model Building

2.2 Random Forest Classifier


```
# Train the Random Forest model
rf = RandomForestClassifier(random_state=42)
rf.fit(X_train, y_train)
```

```
# Predictions and evaluation
y_pred_rf = rf.predict(X_test)
roc_auc_rf = roc_auc_score(y_test, rf.predict_proba(X_test)[:, 1])
print(f"Random Forest ROC-AUC: {roc_auc_rf}")
```

Step 3: Model Evaluation

3.1 Evaluation Metrics

- ROC-AUC Score: Measures the model's ability to distinguish between classes.
- Classification Report: Provides precision, recall, and F1-score.


```
from sklearn.metrics import classification_report

# Logistic Regression report
print("Logistic Regression Report:\n", classification_report(y_test, y_pred_lr))

# Random Forest report
print("Random Forest Report:\n", classification_report(y_test, y_pred_rf)
```

Step 3: Model Evaluation

3.2 Comparing Models

If the Random Forest model has a higher ROC-AUC score and better classification metrics, it may be a better choice for deployment.

Why Random Forest May Be Preferred:

1. **Better Generalization:** Random Forest models can handle complex relationships and are less prone to overfitting due to their ensemble nature.
2. **Feature Importance:** They provide insights into which features are most predictive, helping in refining credit risk assessments.
3. **Robust Performance:** Random Forests tend to perform well even with unbalanced datasets or when there are noisy features.
4. **Scalability:** The model can be easily scaled to handle larger datasets, making it suitable for real-world applications.

.

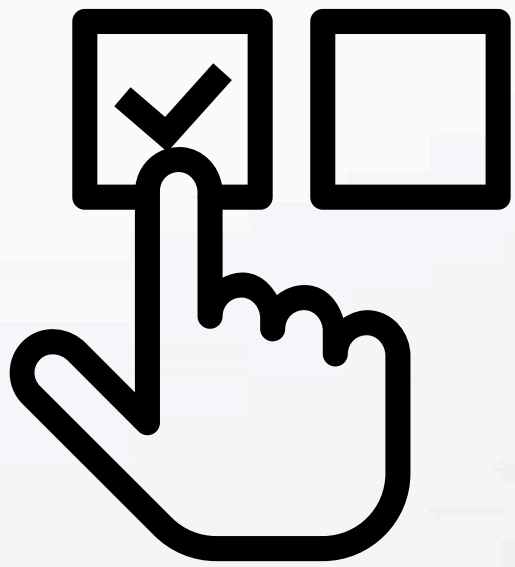
Step 3: Model Evaluation

3.2 Comparing Models

Steps to Confirm Model Selection:

- **Evaluate Metrics:** Compare precision, recall, F1-score, and accuracy alongside ROC-AUC.
- **Conduct Cross-Validation:** Ensure consistent performance across different data splits.
- **Test Real-World Scenarios:** Simulate predictions with recent applicant data to check practical reliability.
- **Consider Interpretability:** Logistic Regression may be preferred if model transparency is crucial for stakeholders or regulatory compliance.

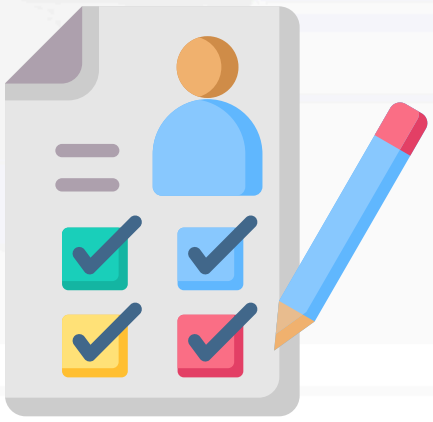
By thoroughly comparing models, you ensure the selected model aligns with the business goals and operational needs



Real-World Implications

Benefits

- **Improved Risk Assessment:** Enhanced ability to identify high-risk applicants.
- **Reduced Financial Losses:** Avoid granting loans to applicants likely to default.
- **Data-Driven Decision-Making:** Empowers credit managers with actionable insights.
    -



Real-World Implications

- **Challenges**
- **Bias in Data:** Historical biases in loan data can affect predictions.
- **Dynamic Financial Environments:** Models may need regular updates to stay relevant.

Summary

- Predicting loan default risk using machine learning provides banks with a powerful tool to mitigate risks and enhance financial stability.
- By leveraging advanced classification models like Logistic Regression and Random Forests, along with robust evaluation metrics like ROC-AUC, banks can make informed decisions about loan approvals.

Special Thanks to ChatGPT
and Gemini for Content support