Lending Club Loan Default Prediction Project

Complete Data Science Project Plan

🎯 PROJECT GOAL

Primary Objective: Build a machine learning model to predict which loans are likely to default (not be fully paid back)

Why This Matters: Help lenders make better decisions about who to give loans to and at what interest rate, reducing financial losses from bad loans.

💼 BUSINESS PROBLEMS YOU'RE SOLVING

1. Credit Risk Assessment

- **Problem:** Lending money is risky - some borrowers won't pay back
- **Your Solution:** Predict default probability before approving loans
- **Business Impact:** Reduce loan losses by 20-30%

2. Interest Rate Optimization

- **Problem:** How much interest should we charge different borrowers?
- **Your Solution:** Higher risk borrowers get higher rates to compensate
- **Business Impact:** Balance risk and profit

3. Loan Approval Automation

- **Problem:** Manual loan review is slow and inconsistent
- **Your Solution:** Automated scoring system for faster decisions
- **Business Impact:** Process 10x more applications

4. Portfolio Management

- **Problem:** Which existing loans need monitoring?
- **Your Solution:** Flag high-risk loans for early intervention
- **Business Impact:** Reduce defaults through proactive management

---

🔍 WHAT TO LOOK FOR IN THIS DATA

Key Questions to Answer:

1. **What makes a loan go bad?**
   - High debt-to-income ratio (DTI > 20%)?
   - Low credit scores (FICO < 650)?
   - Recent delinquencies?

- Short employment history?

2. **What patterns exist in good loans?**
   - Stable employment (10+ years)?
   - Home ownership (MORTGAGE)?
   - Low credit utilization (<30%)?
   - Higher income levels?
3. **Do certain loan characteristics predict defaults?**
   - Loan amount relative to income?
   - Loan term (36 vs 60 months)?
   - Loan purpose (debt consolidation vs home improvement)?
   - Grade assigned (A through G)?
4. **Are there geographic or temporal patterns?**
   - Which states have higher default rates?
   - Do loans from certain months/years perform worse?

📊 PROJECT PHASES

PHASE 1: DATA EXPLORATION & UNDERSTANDING (Week 1)

Goals:

- Understand your dataset structure
- Identify data quality issues
- Discover initial patterns

Tasks:

1. **Load and inspect data**

python

```
import pandas as pd
df = pd.read_csv('lending_club_data.csv')
print(df.shape)
print(df.info())
print(df['loan_status'].value_counts())
```

2. **Check target variable distribution**
   - How many Fully Paid vs Charged Off?
   - Is there class imbalance? (Usually yes - most loans are good)
3. **Analyze missing values**
   - Which columns have >50% nulls?
   - Can you impute or must you drop them?
4. **Explore key features**
   - Distribution of loan_amnt, int_rate, annual_inc
   - Correlation with loan_status
   - Visualize: histograms, box plots, scatter plots

Deliverables:

- ✅ Data quality report
- ✅ Initial visualizations (5-10 key charts)
- ✅ List of columns to drop/keep

PHASE 2: DATA CLEANING & PREPROCESSING (Week 2)

Goals:

- Clean and prepare data for modeling
- Handle missing values and outliers
- Create analysis-ready dataset

Tasks:

1. **Filter to completed loans only**

python

```
# Keep only loans with known outcomes
df = df[df['loan_status'].isin(['Fully Paid', 'Charged Off'])]
```

2. **Handle missing values**
    - Drop columns with >70% nulls (joint app, hardship fields)
    - Impute numerical: median or mean
    - Impute categorical: mode or 'Unknown'
3. **Create binary target variable**

python

```
# 1 = Default/Bad Loan, 0 = Fully Paid/Good Loan
df['default'] = (df['loan_status'] == 'Charged Off').astype(int)
```

4. **Handle outliers**
    - Cap extreme values in annual_inc, loan_amnt
    - Use IQR method or percentile capping
5. **Parse date columns**

python

```
df['issue_d'] = pd.to_datetime(df['issue_d'])
df['issue_year'] = df['issue_d'].dt.year
df['issue_month'] = df['issue_d'].dt.month
```

6. **Drop irrelevant columns**
    - IDs (id, member_id, url)
    - Post-loan data (total_pymnt, recoveries - these leak future info!)
    - Text descriptions (desc, title, emp_title)

Deliverables:

- ✅ Clean dataset saved as CSV
- ✅ Documentation of cleaning decisions
- ✅ Before/after statistics

PHASE 3: FEATURE ENGINEERING (Week 3)

Goals:

- Create new predictive features
- Transform and encode existing features
- Select most important features

Tasks:

1. **Create ratio features**

python

```
df['loan_to_income'] = df['loan_amnt'] / df['annual_inc']
df['installment_to_income'] = df['installment'] / (df['annual_inc'] / 12)
df['revol_bal_to_income'] = df['revol_bal'] / df['annual_inc']
```

2. **Create time-based features**

python

```
df['credit_history_years'] = (df['issue_d'] - pd.to_datetime(df['earliest_cr_line'])).dt.days / 365
```

3. **Encode categorical variables**
    - Ordinal encoding for grade (A=1, B=2, ... G=7)

- Ordinal encoding for emp_length (0-1 years=1, ... 10+ years=11)
- One-hot encoding for purpose, home_ownership, addr_state
- Target encoding for high-cardinality features

4. **Bin numerical features**

python

```
df['fico_bin'] = pd.cut(df['fico_range_low'],
                        bins=[0, 650, 700, 750, 850],
                        labels=['Poor', 'Fair', 'Good', 'Excellent'])
```

5. **Feature selection**
   - Calculate correlations with target
   - Use Random Forest feature importance
   - Keep top 30-50 features

Deliverables:

- ✅ Engineered features dataset
- ✅ Feature importance ranking
- ✅ Feature correlation heatmap

PHASE 4: EXPLORATORY DATA ANALYSIS (EDA) (Week 3-4)

Goals:

- Visualize relationships between features and defaults
- Identify strong predictors
- Generate insights for business

Key Analyses:

1. **Univariate Analysis**
   - Distribution of each feature
   - Identify skewness and outliers
2. **Bivariate Analysis**
   - Default rate by grade
   - Default rate by DTI ranges
   - Default rate by FICO score ranges
   - Default rate by loan purpose
   - Default rate by state
3. **Multivariate Analysis**
   - Correlation matrix
   - Pair plots for top features
   - Group comparisons (high vs low income, homeowners vs renters)
4. **Time Series Analysis**
   - Default rate trends over time
   - Seasonal patterns

Deliverables:

- ✅ 10-15 key visualizations
- ✅ Insights document (What did you learn?)
- ✅ Presentation-ready charts

PHASE 5: MODEL BUILDING (Week 4-5)

Goals:

- Build and compare multiple models
- Handle class imbalance
- Tune hyperparameters

Tasks:

1. **Split data**

python

```python
from sklearn.model_selection import train_test_split

X = df.drop(['default', 'loan_status'], axis=1)
y = df['default']

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42, stratify=y
)
```

2. **Handle class imbalance**
   - Option A: Use SMOTE (synthetic oversampling)
   - Option B: Use class_weight='balanced' in models
   - Option C: Undersample majority class
3. **Build baseline models**

python

```python
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
from xgboost import XGBClassifier

models = {
    'Logistic Regression': LogisticRegression(),
    'Decision Tree': DecisionTreeClassifier(),
    'Random Forest': RandomForestClassifier(),
    'XGBoost': XGBClassifier()
}
```

4. **Evaluate models**
    - Don't just use accuracy! (misleading with imbalanced data)
    - Use: AUC-ROC, Precision, Recall, F1-Score
    - Confusion matrix analysis
    - Precision-Recall curve
5. **Hyperparameter tuning**

python

```python
from sklearn.model_selection import GridSearchCV

param_grid = {
    'n_estimators': [100, 200, 300],
    'max_depth': [10, 20, 30],
    'min_samples_split': [2, 5, 10]
}

grid_search = GridSearchCV(RandomForestClassifier(),
                           param_grid,
                           cv=5,
                           scoring='roc_auc')
```

Deliverables:

- ✅ 4-5 trained models
- ✅ Model comparison table
- ✅ Best model saved (pickle or joblib)

PHASE 6: MODEL EVALUATION & INTERPRETATION (Week 5-6)

Goals:

- Validate model performance
- Understand what drives predictions
- Ensure model is fair and reliable

Tasks:

1. **Evaluate on test set**

python

```
from sklearn.metrics import classification_report, roc_auc_score, roc_curve

y_pred = model.predict(X_test)
y_proba = model.predict_proba(X_test)[:, 1]

print(classification_report(y_test, y_pred))
print(f'AUC-ROC: {roc_auc_score(y_test, y_proba):.3f}')
```

2. **Business metric calculation**
    - Cost of false negative (missing a default): $10,000 loss
    - Cost of false positive (rejecting good loan): $1,000 opportunity cost
    - Calculate expected profit
3. **Model interpretation**

python

```
import shap

explainer = shap.TreeExplainer(model)
shap_values = explainer.shap_values(X_test)
shap.summary_plot(shap_values, X_test)
```

4. **Feature importance analysis**
    - Which features matter most?
    - Are they intuitive?
    - Any surprising findings?
5. **Error analysis**
    - What loans are we getting wrong?
    - False positives vs false negatives patterns

- Can we improve specific segments?

Deliverables:

- ✅ Test set performance report
- ✅ ROC and PR curves
- ✅ Feature importance charts
- ✅ SHAP analysis
- ✅ Error analysis document

PHASE 7: BUSINESS RECOMMENDATIONS (Week 6)

Goals:

- Translate technical results to business value
- Provide actionable recommendations
- Create deployment plan

Deliverables:

1. **Executive Summary (1-2 pages)**
   - Model accuracy: "Correctly predicts 85% of defaults"
   - Business impact: "Could save $5M annually in bad loans"
   - Implementation requirements
2. **Top Risk Factors Identified**
   - List of 10 strongest default predictors
   - Thresholds for automatic approval/rejection
   - Manual review criteria
3. **Loan Approval Strategy**
   - Risk score ranges (0-1000 scale)
   - Recommended approval thresholds
   - Interest rate pricing by risk tier
4. **Monitoring & Deployment Plan**
   - How often to retrain model
   - Key performance indicators (KPIs)
   - A/B testing approach

📈 SUCCESS METRICS

Technical Metrics:

- **AUC-ROC Score:** Target > 0.70 (good), > 0.80 (excellent)
- **Precision:** > 60% (of predicted defaults are actual defaults)
- **Recall:** > 70% (catch 70% of actual defaults)

Business Metrics:

- **Default Rate Reduction:** 15-25% improvement
- **Profit Increase:** Calculate expected ROI
- **Processing Speed:** 10x faster than manual review

🛠️ TOOLS & LIBRARIES YOU'LL USE

Core Tools:

python

```python
import pandas as pd              # Data manipulation
import numpy as np               # Numerical operations
import matplotlib.pyplot as plt  # Visualization
import seaborn as sns            # Advanced visualization

# Machine Learning
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix, roc_auc_score
from xgboost import XGBClassifier

# Imbalanced data
from imblearn.over_sampling import SMOTE

# Model interpretation
import shap
```

📝 FINAL PROJECT STRUCTURE

```
lending_club_project/
│
├── data/
│   ├── raw/
│   │   └── lending_club_data.csv
│   ├── processed/
│   │   ├── cleaned_data.csv
│   │   └── engineered_features.csv
│   └── final/
│       ├── train.csv
│       └── test.csv
│
├── notebooks/
│   ├── 01_data_exploration.ipynb
│   ├── 02_data_cleaning.ipynb
│   ├── 03_feature_engineering.ipynb
│   ├── 04_eda.ipynb
│   ├── 05_modeling.ipynb
│   └── 06_evaluation.ipynb
│
├── models/
│   ├── random_forest_model.pkl
│   └── xgboost_model.pkl
│
├── reports/
│   ├── data_quality_report.pdf
│   ├── eda_insights.pdf
│   ├── model_evaluation.pdf
│   └── business_recommendations.pdf
│
└── README.md
```

🚀 GETTING STARTED - NEXT STEPS

Week 1 Action Items:

1. ✅ Set up your Python environment
2. ✅ Load the dataset and run basic info commands
3. ✅ Check loan_status distribution
4. ✅ Identify columns with >50% missing values
5. ✅ Create 3-5 initial visualizations

Your First Code Block:

python

```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Load data
df = pd.read_csv('lending_club_data.csv')

# Basic info
print(f"Dataset shape: {df.shape}")
print(f"\nLoan Status Distribution:")
print(df['loan_status'].value_counts())

# Missing values
missing_pct = (df.isnull().sum() / len(df)) * 100
print(f"\nColumns with >50% missing:")
print(missing_pct[missing_pct > 50].sort_values(ascending=False))

# Quick visualization
plt.figure(figsize=(10, 6))
df['loan_status'].value_counts().plot(kind='bar')
plt.title('Loan Status Distribution')
plt.xlabel('Status')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

💡 PRO TIPS FOR SUCCESS

1. **Start Simple:** Don't try to use all 151 columns at once. Start with 20-30 key features.
2. **Document Everything:** Write down WHY you made each decision (why you dropped a column, why you chose a threshold).
3. **Visualize Often:** Charts help you spot issues and patterns faster than numbers.
4. **Think Like a Business Person:** Always ask "What would a loan officer do with this information?"
5. **Handle Imbalanced Data Properly:** Most loans are good - don't let your model just predict "all good" and claim 90% accuracy!

6. **Don't Leak Future Data:** Avoid using columns like `total_pymnt` or `recoveries` that wouldn't be known at loan approval time.
7. **Validate Rigorously:** Use cross-validation and hold out a test set you NEVER touch until final evaluation.
8. **Iterate:** First model won't be perfect. Build → Evaluate → Improve → Repeat.

🎓 LEARNING RESOURCES

- **Lending Club Data Dictionary:** Look up any column you don't understand
- **Scikit-learn Documentation:** Your best friend for ML
- **Kaggle Kernels:** See how others approached similar problems
- **SHAP Documentation:** For model interpretation

✅ PROJECT COMPLETION CHECKLIST

- [ ] Data loaded and explored
- [ ] Missing values handled
- [ ] Target variable created (binary: default yes/no)
- [ ] Features engineered and selected
- [ ] EDA completed with 10+ visualizations
- [ ] Multiple models trained and compared
- [ ] Best model selected and tuned
- [ ] Model evaluated on test set
- [ ] Feature importance analyzed
- [ ] Business recommendations written
- [ ] Code cleaned and documented
- [ ] Final presentation prepared

Ready to start? Begin with Phase 1 - Data Exploration! 🚀

Ask me for help with any specific phase or task!