Classifying Lender Loan Payback Using Linear Support Vector Machine - CS37300
Hyeong Kyun Park

Task: Predictive modeling of whether or not the lender will or will not pay off their loan.

Dataset:

1. lending_topredict.csv
   345k observations, 5 columns
2. lending_train.csv
   1M observations, 5 columns

All datasets have: (Notice some has null values. We filled them with 0's;)

```
1 pd.read_csv("lending_train.csv").info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000000 entries, 0 to 999999
Data columns (total 25 columns):
 #   Column                    Non-Null Count    Dtype
---  ------                    --------------    -----
 0   ID                        1000000 non-null  int64
 1   requested_amnt            1000000 non-null  float64
 2   loan_duration             1000000 non-null  object
 3   employment                936438 non-null   object
 4   employment_length         941848 non-null   object
 5   race                      1000000 non-null  object
 6   reason_for_loan           1000000 non-null  object
 7   extended_reason           987575 non-null   object
 8   annual_income             1000000 non-null  float64
 9   debt_to_income_ratio      999722 non-null   float64
 10  employment_verified       1000000 non-null  object
 11  public_bankruptcies       999484 non-null   float64
 12  zipcode                   999999 non-null   object
 13  state                     1000000 non-null  object
 14  home_ownership_status     1000000 non-null  object
 15  delinquency_last_2yrs     1000000 non-null  float64
 16  fico_score_range_low      1000000 non-null  float64
 17  fico_score_range_high     1000000 non-null  float64
 18  fico_inquired_last_6mths  999999 non-null   float64
 19  months_since_last_delinq  495691 non-null   float64
 20  revolving_balance         1000000 non-null  float64
 21  total_revolving_limit     949769 non-null   float64
 22  type_of_application       1000000 non-null  object
 23  any_tax_liens             999972 non-null   float64
 24  loan_paid                 1000000 non-null  int64
dtypes: float64(12), int64(2), object(11)
memory usage: 190.7+ MB
```

```
1 pd.read_csv("lending_train.csv").iloc[0,:]
```

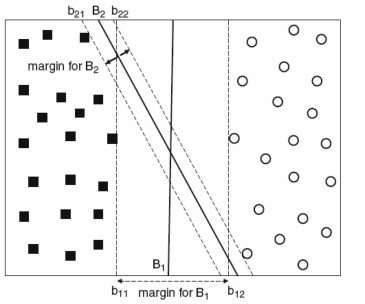
```
ID                                      0
requested_amnt                    32000.0
loan_duration                   60 months
employment                            SVP
employment_length                4 years
race                                    W
reason_for_loan           debt_consolidation
extended_reason           Debt consolidation
annual_income                   250000.0
debt_to_income_ratio               16.35
employment_verified             Verified
public_bankruptcies                  0.0
zipcode                            333xx
state                                 FL
home_ownership_status               RENT
delinquency_last_2yrs                0.0
fico_score_range_low               775.0
fico_score_range_high              779.0
fico_inquired_last_6mths             0.0
months_since_last_delinq             NaN
revolving_balance                22480.0
total_revolving_limit           105700.0
type_of_application           Individual
any_tax_liens                        0.0
loan_paid                              1
Name: 0, dtype: object
```

Model & Model Space (Knowledge Representation

Linear Support Vector Machine

$$y = sign\left[\sum_{i=1}^{m} w_i x_i + b\right]$$

- Model space: set of weights w and b
- Input: Categorical variables encoding using CatBoostEncoder. Continuous variables passed just the way they are. All features are scaled with StandardScaler.
- Output: Classification, so categorical variable. In our situation, it would be whether the lender pays back his or her loan or not (1 or 0).



Essentially we want to choose, among many equivalent hyperplanes, the one that maximizes the margin, described in the figure above.

The score function formula (must contain the parameters)

This maximizes the margin subject to constraint that all training data is correctly classified.

- w: weight
- b: y-intercept
- x(i), y(i) = i'th training sample
- ai = coefficient associated with the i'th training sample

$$L_P = \frac{1}{2}||w||^2 - \sum_{i=1}^{I} \alpha_i y(i)[x(i) \cdot w + b] + \sum_{i=1}^{I} \alpha_i$$

The search function (how are you finding good models?)

I chose SVM for classification to increase the accuracy over logistic regression. The model was able to achieve 0.1254 training accuracy (Matthew's Correlation Coefficient).

I did not perform hyperparameter tuning for the RFC, I just ran it with the default parameters. If I were to tune my hyperparameters, my search function would have been GridSearchCV.

For feature selection, I dropped ID, which does not contribute to the prediction and only is used for identifying the observations. I also dropped the state and zipcode, since the location itself does not say much about an individual and their likelihood of paying back his or her loan.