LOAN PREDICTION

project

It is a Machine learning project given by skillsanta

dhruv singhal
Dhruv.singhal2612@gmail.com

Contents

Section 1: preparing of data

1.1 Finding missing data and converting categorical variable(code section 1.1)

Columns	missing values
Gender	24
Married	3
Dependents	25
Education	0
Self_Employed	55
ApplicantIncome	0
CoapplicantIncome	0
LoanAmount	27
Loan_Amount_Term	20
Credit_History	79
Property_Area	0
Loan_Status	367

- **Gender :** it contains 24 missing values and they are replaced by male as it is majority, it is a categorical variable so converted into int values(categorical)
- **Married:** 3 missing, majority are married(categorical)
- **Dependents:** 25 missing, majority are '0'(categorical)
- **Education:** 0 missing, majority are 'graduate'(categorical)
- **Self_employed:** 55 missing, majority are not self employed(categorical variable)
- **Applicantincome:** 0 missing and it is a float variable
- **Coapplicantincome:** 0 missing and it is a float variable
- **loanAmount:** 27 missing, missing values replaced by median value of columns(float variable)
- **Loan_amount_term:** 20 missing, missing values replaced by median of column(float variable)
- **Property_area:** 0 missing and categorical variable
- **Credit_history:** 79 missing and categorical variable

All categorical variables are replaced by integer values using **.cat.codes**

1.2 removal of outliers(code section 1.2)

As we can see from boxplot diagrams ,there are many outliers in continuous variables and outliers in **applicant income , coapplicantincome and loanAmount .** outliers from **loan_amount_term** are not removed as most of the values are 360 as boxplot is a line parallel to x-axis , so all other points are

considered as outliers and if we try to remove those outliers, it will destroy a feature for us

Section 2: EDA(code section 2)

2.1 **correlation**(code section 2.1)

As it is evident from heatmap **loanAmount and applicantincome** are strongly correlated , also **married** has noticeable correlation with **gender, dependents** so we can consider removing them in future.

2.2 data visualisation(code section 2.2)

- loan_status approval density as well as rejection density decreases with increase in dependents
- married has more both loan approval and rejection
- male has more both loan approval and rejection
- on average if applicant income increases ,coapplicant income decreases
- as applicant income amount applied for loan also increases

2.3 checking skewness in continuous variables(code section 2.3)

LoanAmount and Applicantincome are pretty normally distributed but that's not the case with coapplicantincome which has a positive skewness in its curve dur to presence of outliers but if we try to remove it using np.log1p() function it gives two equal global maxima which is poorer for modelling so we will keep the shape of features same.

Section 3 modelling(code section 3.1 onwards)

3.1 choice of algorithm

1. **random forest:** random forest algorithm is a great algorithm for classifying high dimentional data , it is also very helpful in calculating the feature importances and its decision boundary is more stable than single decision tree due to presence of many decision trees
2. **logistic regression:** it is a great algorithm for binary classification purposes. Moreover , it does not easily overfit as it allows bias error in data while training as it is based on linear regression and thus, model is simple than it can get and therefore generalises well on unseen data

3.2 hyperparameter tuning(code section 3.3 and 4.2)

1. **random forest**
   - n_estimators: it is the number of trees to be used while modeling and its range should be not too small(underfit) and not too large (overfit)
   - max_depth: it is the maximum height of tree, if given none height will increases until we achieve pure leaves,
   - min_samples_split: it is the minimum number of samples required to split an internal node
   - max_features: it is the number that are kept in mind while splitting a node in best possible way.
   - Min_samples_leaf: it is the minimum number of samples required to be at the lea node, it should be as high that the processor can handle but very high value might not generalize new data well

2.logistic regression

- **C:** it is inverse of regularisation, very high or low value might overfit and underfit the data respectively
- **Penalty:** it is the type of regularisation applied.
- **Class_weight:** it is weigths associated with the classes, if none then weigths of all classed be same i.e. 1

3.3 Feature engineering(code section 3.2):

1.we can see the feature importances using random forest feature_importances attribute.
2.we have created two new features-
Total_income = applicant_income + coapplicant_income
Which is the total income of a person before applying for loan

Unit_loan_amount = loanAmount/loan_amount_term
Which is the loan taken in unit (whatever be the unit of loan_amount_term)
And we will consider these features while creating our model and see if they improves our model or not.

3.4 Best model(code section 5.2)

Our best two models are rf8_std_grid and rf9 as evident from our classification report and confusion matrix

Models	BAC	REC	AUC
Rf8_std_grid	0.70	0.95	0.77
Rf9	0.81	0.86	0.90

Section 4: innovation

- ***Resampling :*** it is a technique to deal datasets with imbalanced classes , we have applied over sampling on our data as our target variable is imbalanced and it may be giving us biased results which have improved our overall model as evident from metric scores.
- ***Feature_enginerring:*** I have plotted different plots to undertand the data and engineered two new features out of 4 which decreases the complexity of our model .
- ***Use of polynomial features:*** I have tried to create new polynomial features using inbuilt class and fit them in logistic regression as our model becomes more complex , our model may overfit but logistic regression allow bias in training data which prevents it from being overfitted.