SRM
INSTITUTE OF SCIENCE & TECHNOLOGY
(Deemed to be University u/s 3 of UGC Act, 1956)

LOAN DEFAULT PREDICTION

TEAM MEMBERS

SONIA PAUL[RA2011003010451]
ADARSH KUMAR[RA2011003010459]

ABSTRACT

Loan default prediction refers to the process of using data analysis and statistical models to predict the likelihood of a borrower failing to repay a loan.

To make these predictions, data scientists analyze various factors that could impact a borrower's ability to repay a loan, such as their credit score, income, debt-to-income ratio, employment status, and payment history. They then use machine learning algorithms to build predictive models that can estimate the likelihood of loan default based on these factors.

Once a model is trained, it can be used to predict whether a new loan applicant is likely to default on their loan. The lender can then use this information to make a more informed decision about whether or not to approve the loan, and if so, what interest rate to offer.

Overall, loan default prediction is a valuable tool for lenders because it helps them manage their risk and avoid lending money to borrowers who are unlikely to be able to repay their loans.

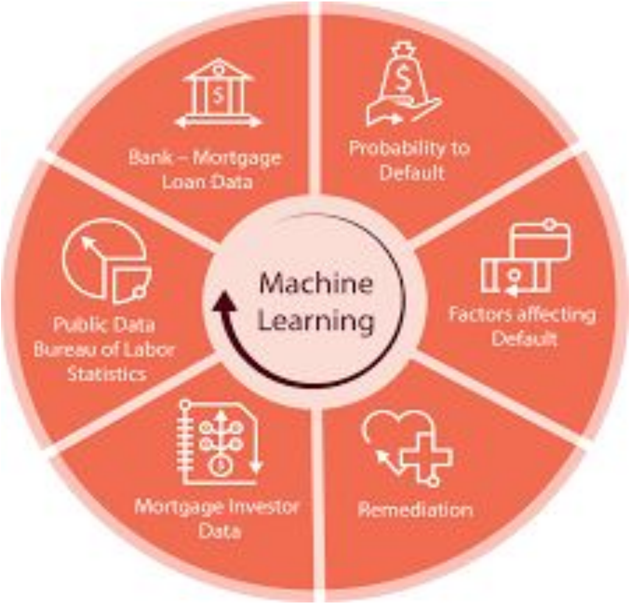
WHAT IS THE NEED ?

Loan default prediction is important for both lenders and borrowers. Lenders need to predict the likelihood of a borrower defaulting on a loan in order to manage their credit risk and protect their financial interests. Borrowers, on the other hand, need to be aware of the factors that could lead to a default so that they can take appropriate measures to avoid defaulting and potentially damaging their credit score.

HOW IT WORKS.....

Loan default prediction models typically work by analyzing historical loan data to identify patterns and trends that are associated with loan defaults. These models use statistical algorithms and machine learning techniques to identify the most important factors that influence loan repayment, such as credit score, debt-to-income ratio, loan amount, and employment status.

The loan default prediction model will typically be trained on a dataset of past loans, where each loan record includes information on the borrower's characteristics and whether or not they defaulted on the loan. The model will then use this data to identify patterns and relationships that can help it predict the likelihood of loan default for new loan applicants.


Machine Learning
Bank – Mortgage Loan Data
Probability to Default
Factors affecting Default
Remediation
Mortgage Investor Data
Public Data Bureau of Labor Statistics

LITERATURE SURVEY

S. N	Title	Author Name	Methodology	Inference	Drawbacks
1.	Loan default prediction using a credit rating-specific and multi-objective ensemble learning scheme	Yu Song [a b 1], Yuyan Wang [a c 1], Xin Ye [a], Russell Zaretzki [b], Chuanren Liu [b]	OC-SVM ensemble models are adaptively constructed for data with distinct imbalance ratios.	An adaptive classification boundary adjustment approach is further proposed to identify a better label assignment threshold for each individual in the credit risk assessment task.	Loan default prediction models rely heavily on historical data to make accurate predictions. However, if the historical data is incomplete or inaccurate, the model's predictions may not be reliable.
2.	Credit Risk Analysis Using Machine and Deep Learning Models	Peter Martey Addo, Dominique Guegan, Bertrand Hassani	the tree-based models are more stable than the models based on multilayer artificial neural networks.	we have shown that it is important to consider a pool of models that match the data and business problem. This is clearly deduced by looking at the difference in performance metrics.	that we did not consider a combination of the different models.

EXISTING SYSTEMS	MERITS	DEMERITS
FICO Score System	The merits of the FICO score system include its simplicity and widespread use, as well as its ability to provide a standardized measure of creditworthiness.	its demerits include its focus on past credit history rather than other relevant factors, as well as its potential to perpetuate discrimination and biases.
Credit Risk Assessment Models	The merits of credit risk assessment models include their ability to handle large and complex datasets and their high accuracy.	their lack of interpretability and the potential for overfitting if not properly tuned.

CHALLENGES

Limited Data Availability: One of the major challenges in loan default prediction is the limited availability of data, particularly for new borrowers or those with limited credit histories. This can make it difficult to build accurate predictive models.

Imbalanced Data:Loan default data is often imbalanced, with a small number of default cases compared to non-default cases. This can make it difficult to build accurate predictive models, as traditional machine learning algorithms tend to be biased towards the majority class.

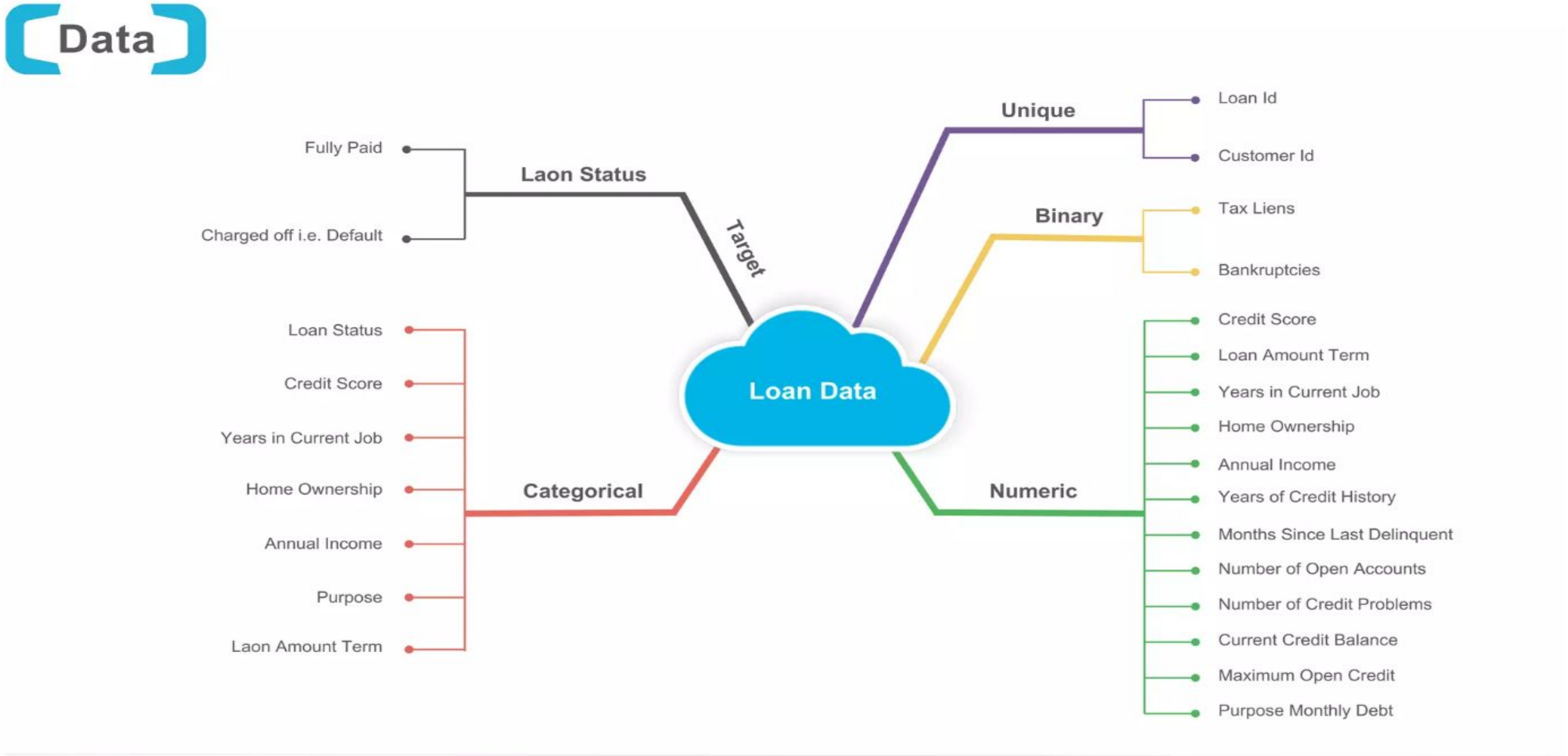
Changing Economic Conditions: Loan default prediction is also affected by changing economic conditions, such as recessions or periods of economic growth. These changes can affect borrower behavior and make it difficult to accurately predict loan default rates.

Lack of Transparency:The lack of transparency in some lending practices and the complexity of loan products can make it difficult to accurately assess the risk of loan default. For example, borrowers may not fully understand the terms of their loans, or lenders may not disclose all relevant information.

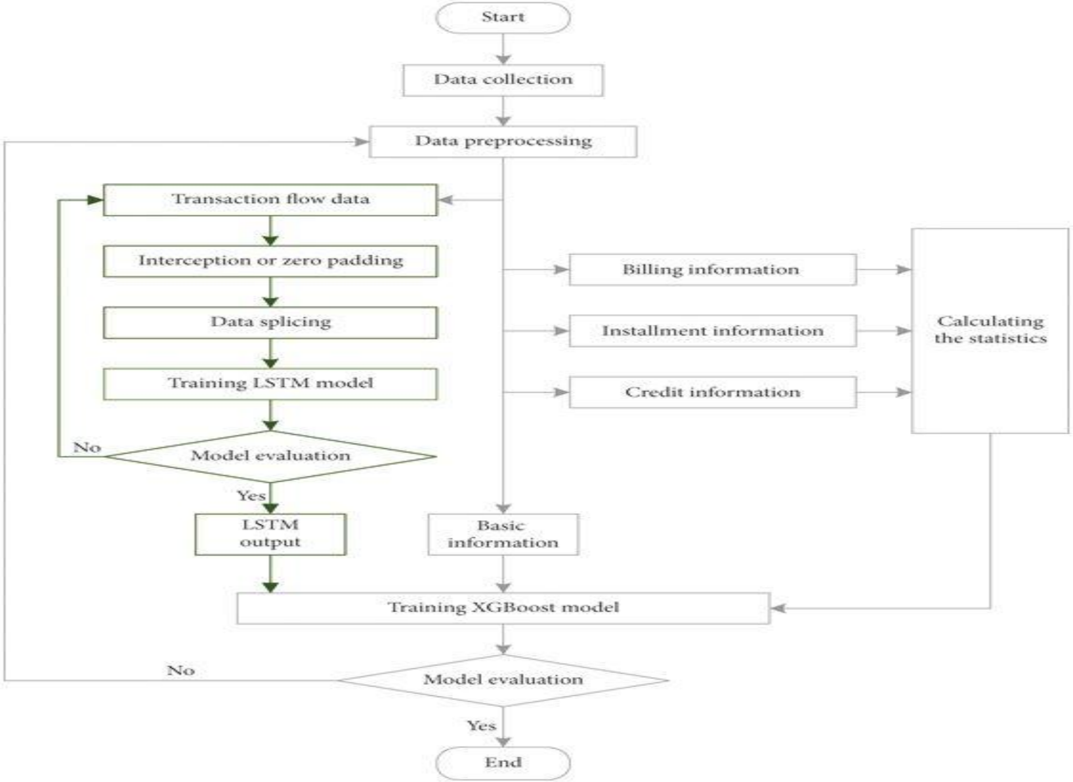
Problem Statement and Objectives

The problem statement of loan default prediction is to identify borrowers who are at high risk of defaulting on their loans, and the objective is to develop a predictive model that can accurately and reliably identify these high-risk borrowers to enable lenders to make informed lending decisions and minimize the risk of loan default.

ARCHITECTURE DIAGRAM



BLOCK DIAGRAM



Modules Description

There are several modules involved in the implementation of loan default prediction. These modules can be broadly classified into three categories:

1. Data Preprocessing: This module involves preparing the data for analysis. It includes tasks such as data cleaning, data transformation, and data normalization. Data cleaning involves removing irrelevant and duplicate data, filling in missing values, and correcting inconsistencies in the data. Data transformation involves converting data into a suitable format for analysis, such as converting categorical variables into numerical variables. Data normalization involves scaling the data to a common range to avoid bias.
2. Feature Engineering: This module involves selecting the most relevant features for the analysis. Features are characteristics of the loan or borrower that can influence the likelihood of default. Feature selection involves choosing the most relevant features based on their predictive power and eliminating features that are irrelevant or redundant. Feature engineering also involves creating new features based on domain knowledge or statistical analysis.

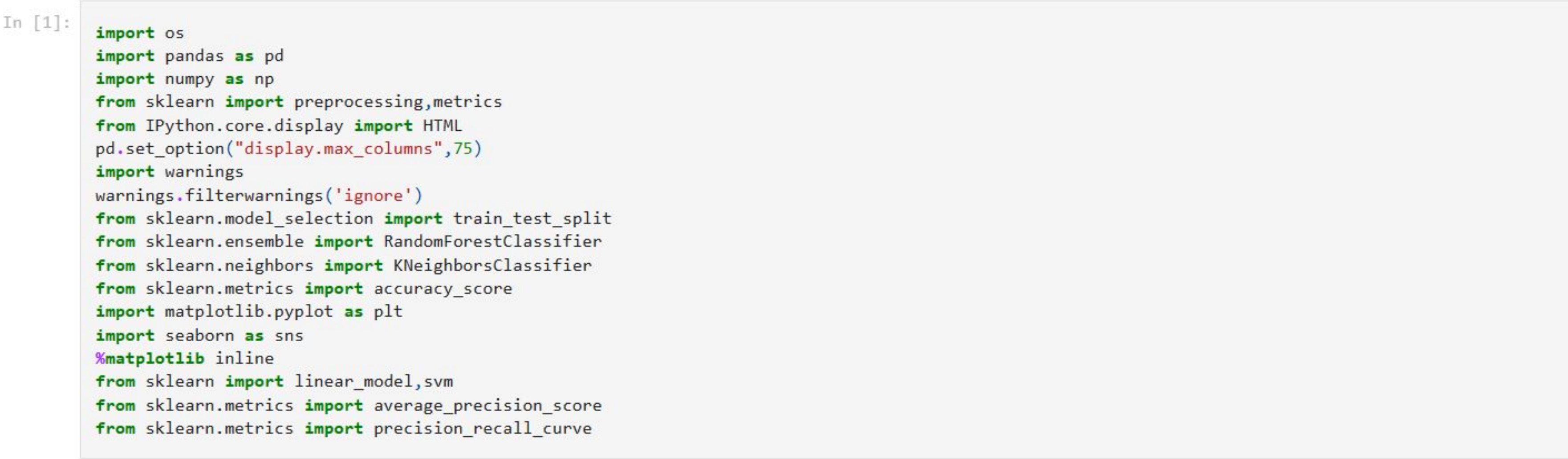
3.

Machine Learning: This module involves training and evaluating machine learning models to predict loan default. Machine learning models can be divided into two categories: supervised and unsupervised. Supervised learning involves training models on labeled data, where the outcome variable (default or non-default) is known. Common supervised learning algorithms used for loan default prediction include logistic regression, decision trees, random forests, and neural networks. Unsupervised learning involves training models on unlabeled data to identify patterns and group similar borrowers together.

Implementation

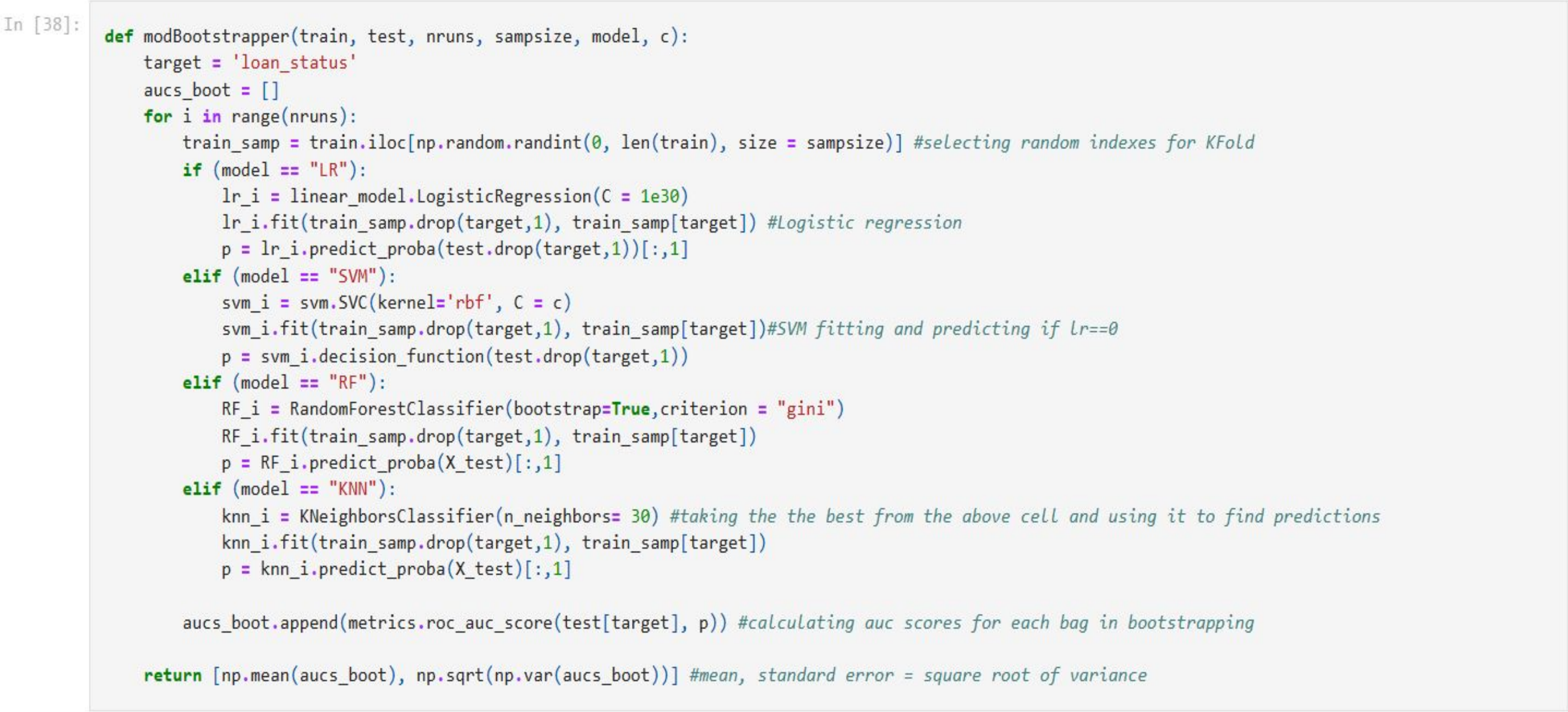
1. Data Collection: Collecting data from various sources, such as credit bureaus, financial institutions, and government agencies.
2. Data Cleaning: Cleaning and pre-processing the data to remove irrelevant or duplicated information.
3. Model Selection: Selecting appropriate machine learning algorithms for the task and evaluating their performance using cross-validation.
4. Model Training: Training the selected machine learning algorithm on the pre-processed data.
5. Model Evaluation: Evaluating the performance of the trained model using evaluation metrics such as accuracy, precision, recall, and F1-score.
6. Model Deployment: Deploying the trained model to predict the likelihood of loan default for new borrowers.
7. Model Monitoring: Monitoring the performance of the deployed model and updating it periodically to improve its accuracy and predictive power.

Screenshots And Result

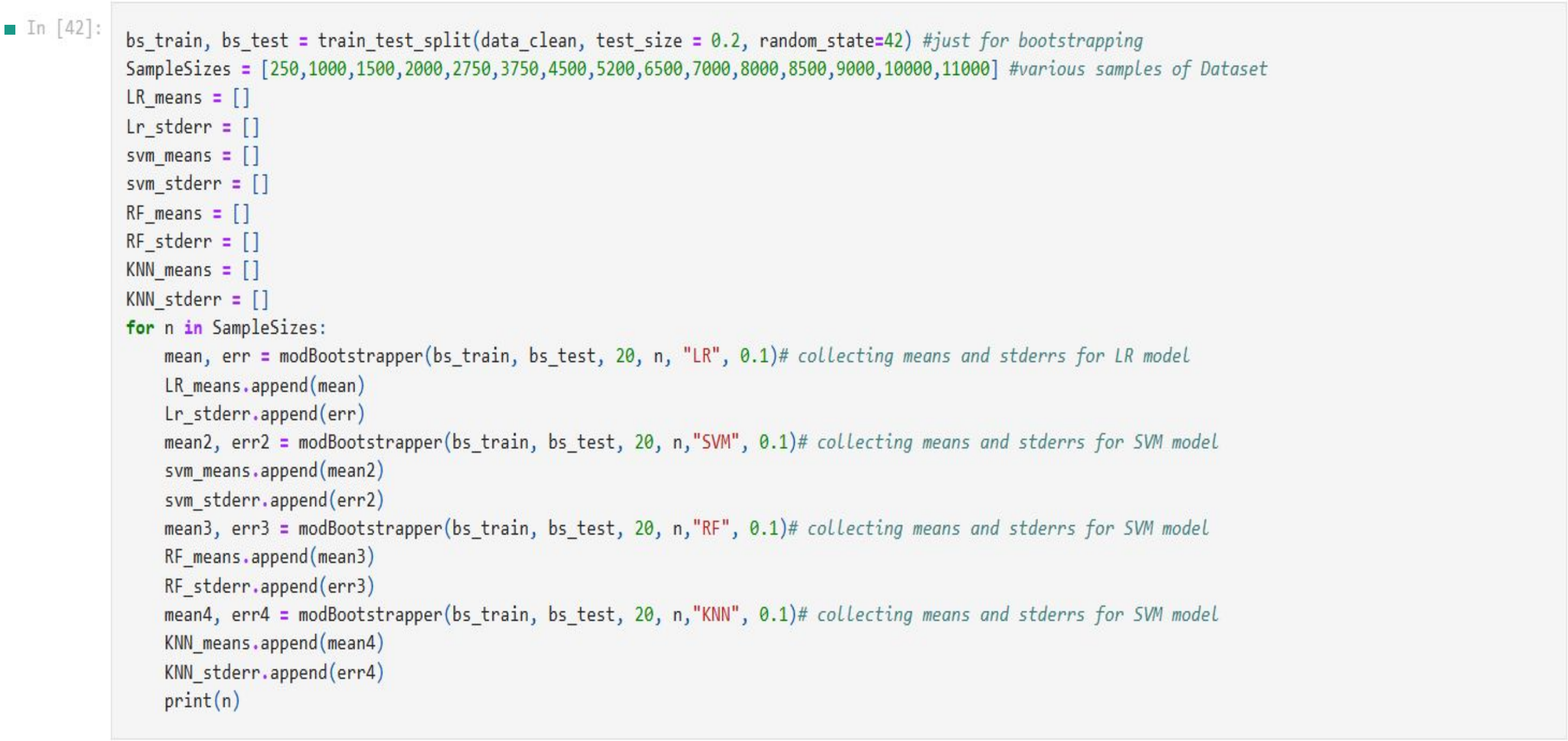


In [1]:

```
import os
import pandas as pd
import numpy as np
from sklearn import preprocessing,metrics
from IPython.core.display import HTML
pd.set_option("display.max_columns",75)
import warnings
warnings.filterwarnings('ignore')
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn import linear_model,svm
from sklearn.metrics import average_precision_score
from sklearn.metrics import precision_recall_curve
```

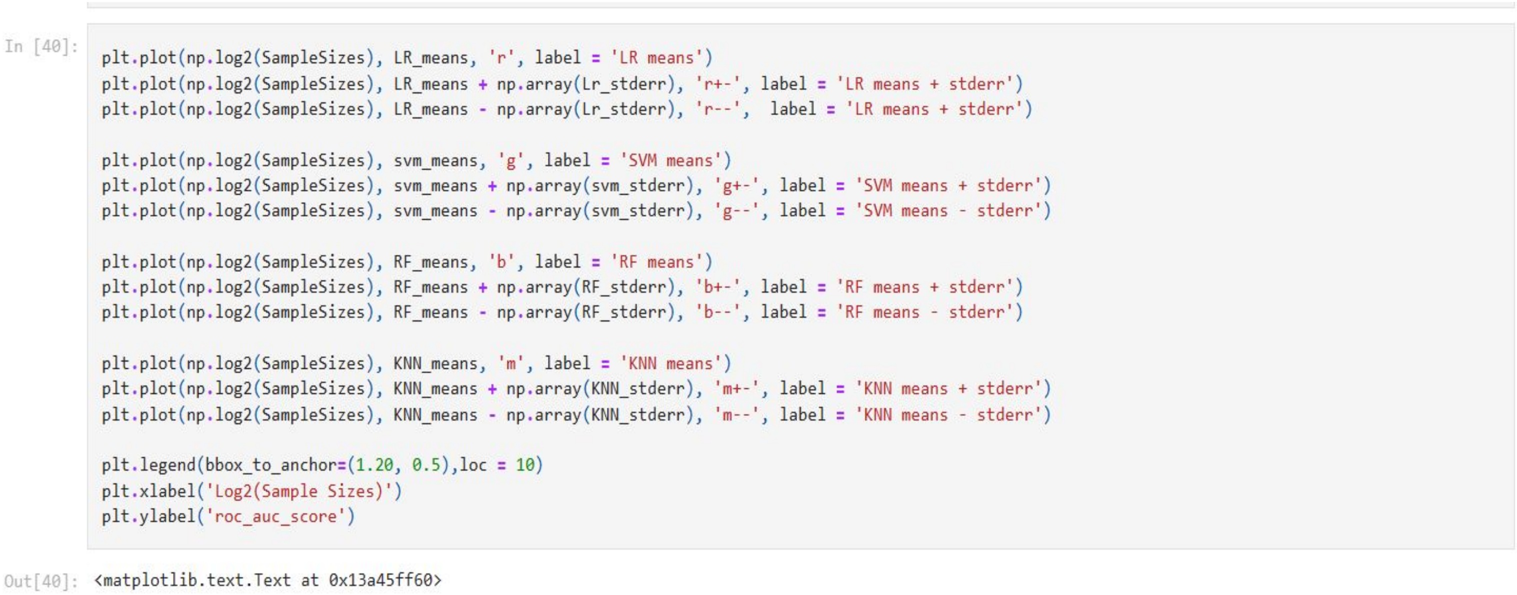


```
In [38]:
def modBootstrapper(train, test, nruns, sampsize, model, c):
    target = 'loan_status'
    aucs_boot = []
    for i in range(nruns):
        train_samp = train.iloc[np.random.randint(0, len(train), size = sampsize)] #selecting random indexes for KFold
        if (model == "LR"):
            lr_i = linear_model.LogisticRegression(C = 1e30)
            lr_i.fit(train_samp.drop(target,1), train_samp[target]) #Logistic regression
            p = lr_i.predict_proba(test.drop(target,1))[:,1]
        elif (model == "SVM"):
            svm_i = svm.SVC(kernel='rbf', C = c)
            svm_i.fit(train_samp.drop(target,1), train_samp[target])#SVM fitting and predicting if lr==0
            p = svm_i.decision_function(test.drop(target,1))
        elif (model == "RF"):
            RF_i = RandomForestClassifier(bootstrap=True,criterion = "gini")
            RF_i.fit(train_samp.drop(target,1), train_samp[target])
            p = RF_i.predict_proba(X_test)[:,1]
        elif (model == "KNN"):
            knn_i = KNeighborsClassifier(n_neighbors= 30) #taking the the best from the above cell and using it to find predictions
            knn_i.fit(train_samp.drop(target,1), train_samp[target])
            p = knn_i.predict_proba(X_test)[:,1]

        aucs_boot.append(metrics.roc_auc_score(test[target], p)) #calculating auc scores for each bag in bootstrapping

    return [np.mean(aucs_boot), np.sqrt(np.var(aucs_boot))] #mean, standard error = square root of variance
```

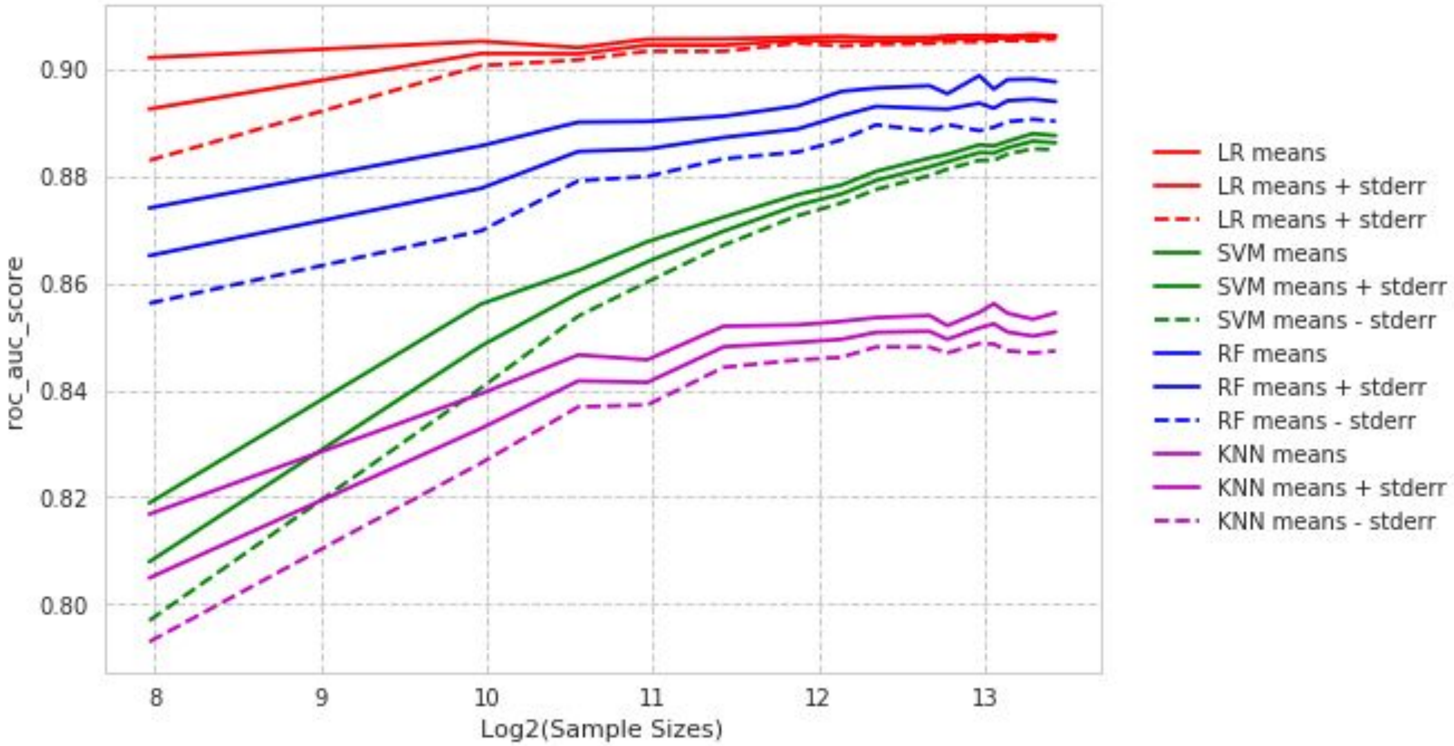


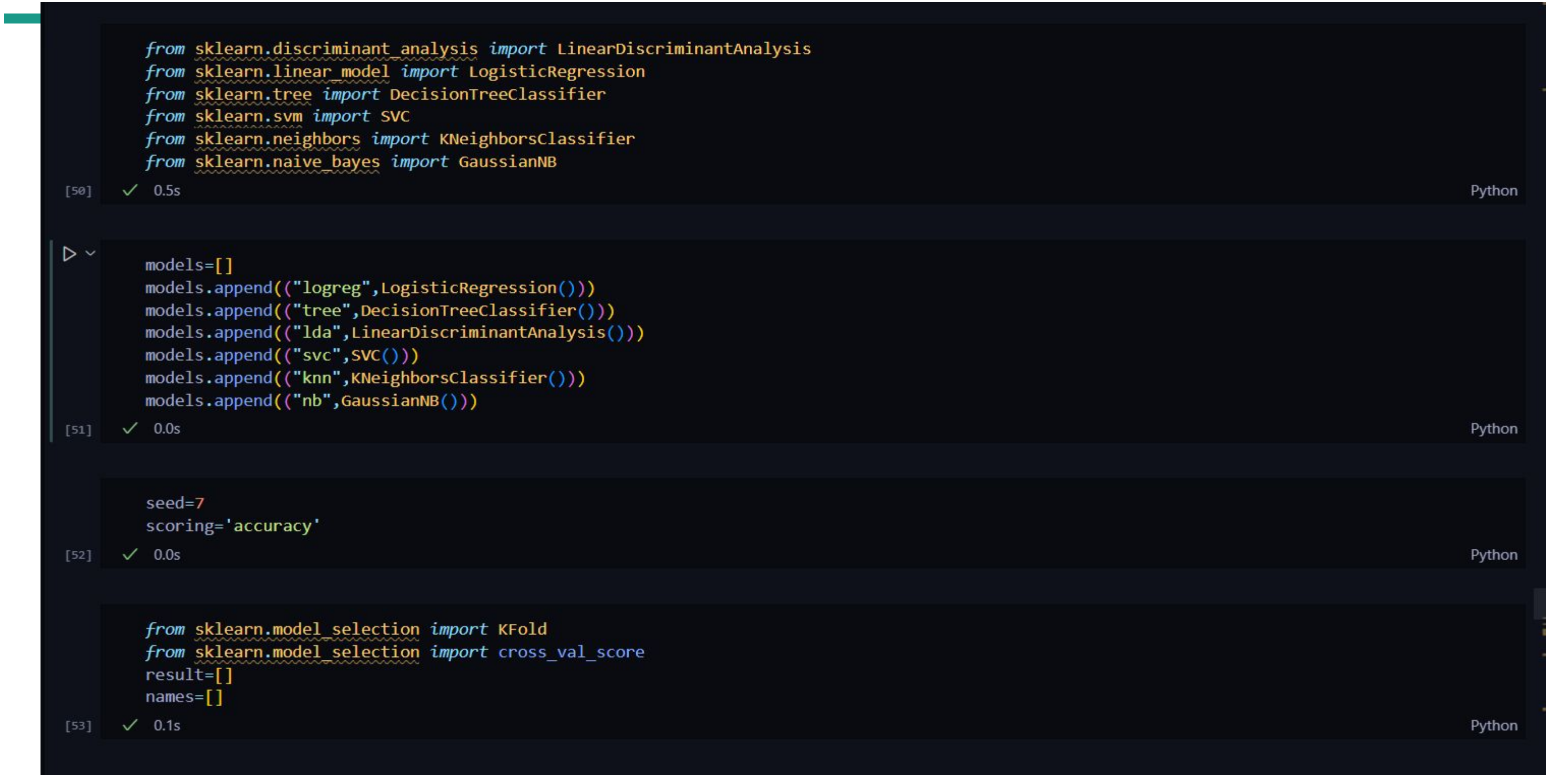
In [42]:

```
bs_train, bs_test = train_test_split(data_clean, test_size = 0.2, random_state=42) #just for bootstrapping
SampleSizes = [250,1000,1500,2000,2750,3750,4500,5200,6500,7000,8000,8500,9000,10000,11000] #various samples of Dataset
LR_means = []
Lr_stderr = []
svm_means = []
svm_stderr = []
RF_means = []
RF_stderr = []
KNN_means = []
KNN_stderr = []
for n in SampleSizes:
    mean, err = modBootstrapper(bs_train, bs_test, 20, n, "LR", 0.1)# collecting means and stderrs for LR model
    LR_means.append(mean)
    Lr_stderr.append(err)
    mean2, err2 = modBootstrapper(bs_train, bs_test, 20, n,"SVM", 0.1)# collecting means and stderrs for SVM model
    svm_means.append(mean2)
    svm_stderr.append(err2)
    mean3, err3 = modBootstrapper(bs_train, bs_test, 20, n,"RF", 0.1)# collecting means and stderrs for SVM model
    RF_means.append(mean3)
    RF_stderr.append(err3)
    mean4, err4 = modBootstrapper(bs_train, bs_test, 20, n,"KNN", 0.1)# collecting means and stderrs for SVM model
    KNN_means.append(mean4)
    KNN_stderr.append(err4)
    print(n)
```



```
In [40]:
plt.plot(np.log2(SampleSizes), LR_means, 'r', label = 'LR means')
plt.plot(np.log2(SampleSizes), LR_means + np.array(Lr_stderr), 'r+-', label = 'LR means + stderr')
plt.plot(np.log2(SampleSizes), LR_means - np.array(Lr_stderr), 'r--',  label = 'LR means + stderr')

plt.plot(np.log2(SampleSizes), svm_means, 'g', label = 'SVM means')
plt.plot(np.log2(SampleSizes), svm_means + np.array(svm_stderr), 'g+-', label = 'SVM means + stderr')
plt.plot(np.log2(SampleSizes), svm_means - np.array(svm_stderr), 'g--', label = 'SVM means - stderr')

plt.plot(np.log2(SampleSizes), RF_means, 'b', label = 'RF means')
plt.plot(np.log2(SampleSizes), RF_means + np.array(RF_stderr), 'b+-', label = 'RF means + stderr')
plt.plot(np.log2(SampleSizes), RF_means - np.array(RF_stderr), 'b--', label = 'RF means - stderr')

plt.plot(np.log2(SampleSizes), KNN_means, 'm', label = 'KNN means')
plt.plot(np.log2(SampleSizes), KNN_means + np.array(KNN_stderr), 'm+-', label = 'KNN means + stderr')
plt.plot(np.log2(SampleSizes), KNN_means - np.array(KNN_stderr), 'm--', label = 'KNN means - stderr')

plt.legend(bbox_to_anchor=(1.20, 0.5),loc = 10)
plt.xlabel('Log2(Sample Sizes)')
plt.ylabel('roc_auc_score')

Out[40]: <matplotlib.text.Text at 0x13a45ff60>
```


roc_auc_score
0.90
0.88
0.86
0.84
0.82
0.80
8
9
10
11
12
13
Log2(Sample Sizes)
LR means
LR means + stderr
LR means + stderr
SVM means
SVM means + stderr
SVM means - stderr
RF means
RF means + stderr
RF means - stderr
KNN means
KNN means + stderr
KNN means - stderr

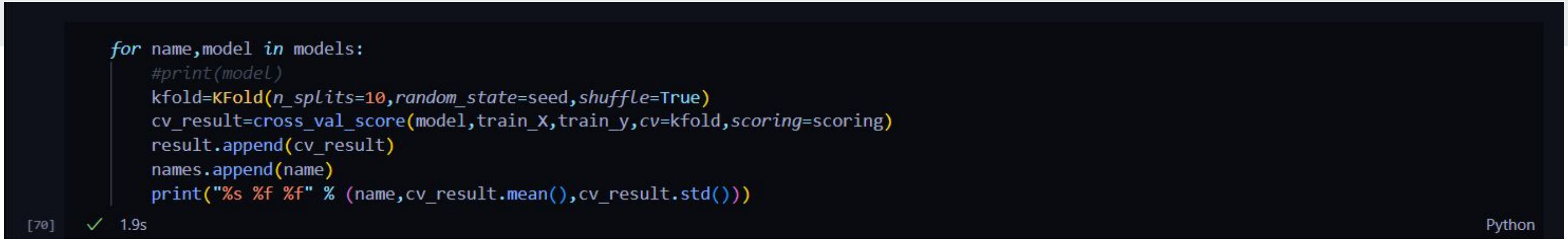


```python
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
```

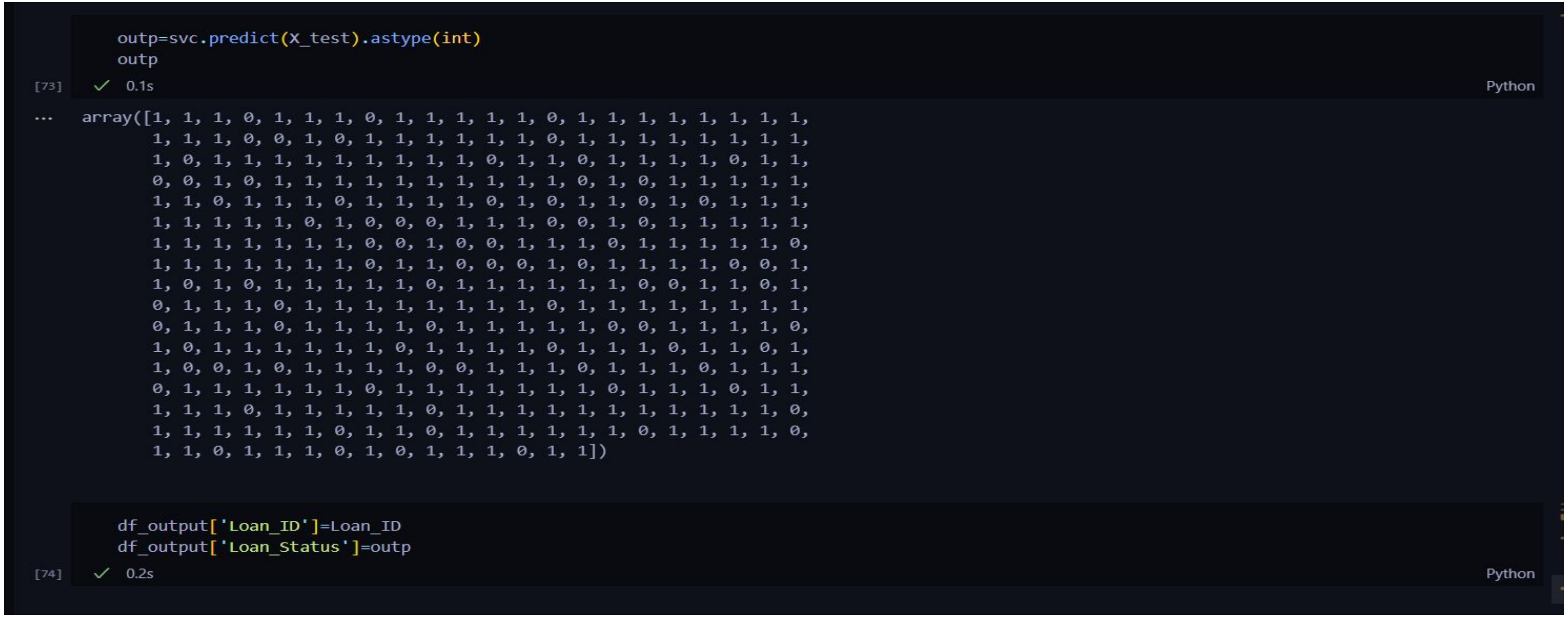
```python
models=[]
models.append(("logreg",LogisticRegression()))
models.append(("tree",DecisionTreeClassifier()))
models.append(("lda",LinearDiscriminantAnalysis()))
models.append(("svc",SVC()))
models.append(("knn",KNeighborsClassifier()))
models.append(("nb",GaussianNB()))
```

```python
seed=7
scoring='accuracy'
```

```python
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
result=[]
names=[]
```



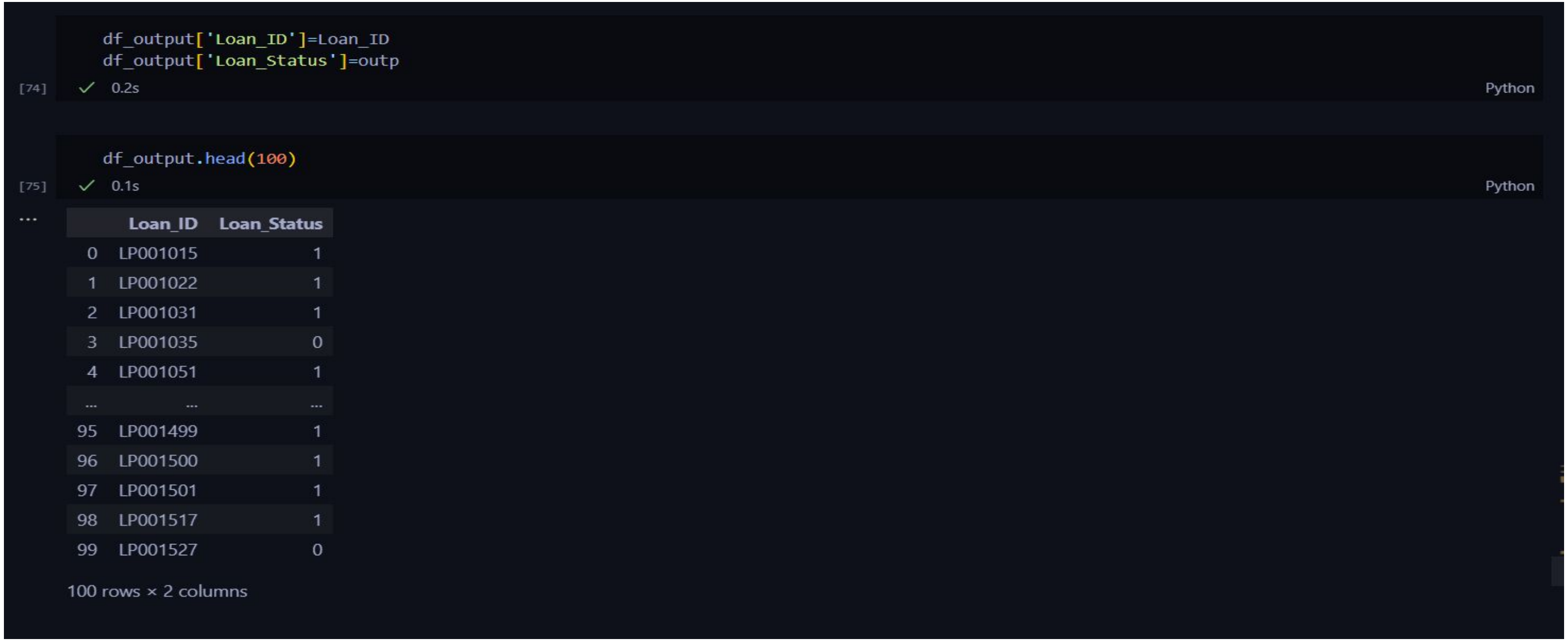
```python
for name,model in models:
    #print(model)
    kfold=KFold(n_splits=10,random_state=seed,shuffle=True)
    cv_result=cross_val_score(model,train_X,train_y,cv=kfold,scoring=scoring)
    result.append(cv_result)
    names.append(name)
    print("%s %f %f" % (name,cv_result.mean(),cv_result.std()))
```



```python
outp=svc.predict(X_test).astype(int)
outp
```

```
array([1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1,
       0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1,
       1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1,
       1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0,
       1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1,
       1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1,
       0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1,
       0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0,
       1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1,
       1, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1,
       0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1,
       1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0,
       1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0,
       1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1])
```

```python
df_output['Loan_ID']=Loan_ID
df_output['Loan_Status']=outp
```



```
df_output['Loan_ID']=Loan_ID
df_output['Loan_Status']=outp
```

[74] ✓ 0.2s Python

```
df_output.head(100)
```

[75] ✓ 0.1s Python

```
      Loan_ID  Loan_Status
 0  LP001015             1
 1  LP001022             1
 2  LP001031             1
 3  LP001035             0
 4  LP001051             1
...       ...           ...
95  LP001499             1
96  LP001500             1
97  LP001501             1
98  LP001517             1
99  LP001527             0

100 rows × 2 columns
```

REFRENCES

"Predicting loan default probability using machine learning techniques" by S. Y. Lim, K. Kim, and S. Lee

"Loan Default Prediction using Machine Learning Techniques" by M. Aboualfa and M. Al-garadi

"Loan default prediction using machine learning techniques: A review" by K. S. K. Prasad and S. G. Umarani

THANK YOU