Table of Content

CHAPTER 1.0

INTRODUCTION

1.1 Project Overview

The proposed project titled “Loan Prediction” is aimed to provide Banking people to know weather their customer is eligible for loan or not. The technology for betterment is the key concept behind selecting the title as it transforms the hand work into the fast computing computer and making the efficient use of time.

1.2 Scope

This project is very useful for the banking sector and people who want to get the loan from the bank either it is home or personal or business. They just need to provide the information about their total income, no. of dependent, Education level and so on. And system will calculate the rest and will tell them weather loan is granted or not.

1.3 Objective

The purpose of this project is to help the people who spend lot of time in bank for getting approval bank and time is wasted. So, this system helps them to know before they go to bank and waste their precious time.

Another purpose of this project can done in respect of bank. Person working in bank can know their customers details and they can approach directly to them regarding the loan and new offers related to loan and in interest of bank.

1.4 Purpose

The purpose of this document is to help to have a clear understanding about project. The main purpose of the project is to provide help in terms of know weather they can get the loan or not. For banks purpose they can get their customers to know about the loan.

1.5 Technology review

Loan Prediction is used for Knowing Weather the loan will be granted to customer or not. We are using programming language as Python and is very easy to understand and is one of the language emerging for Data Science. We have used the machine learning which also one the field for Analysis. Also, we got a much better performance out of project a large amount of time can be saved. So, in that sense, Loan prediction is good but it also have some draw backs.

CHAPTER 2.0

System Requirement Study

2.1 User Characteristics

In Loan Prediction, mainly there are 3 technologies behind it namely,

- Anaconda package
- Jupyter Notebook
- Python 2.7

This 3 technologies are powerful to create any type of Data science project. Anaconda package contains useful libs. Jupter notebook is one platform or IDE for python, which is included in Anaconda Package. Python 2.7 is programming language.

2.2 Tools & Technology

- Jupyter Notebook
- Python 2.7
- Anaconda Package
- Scipy
- Numpy
- Pandas

Scipy: - it python file that helps us to calculate Scientific functions.

Numpy: - it the python file that helps to calculate the N-dimensional array and numerical computation.

Pandas: - It is powerful python Data Structure and analysis toolkit.

Anaconda: - It is an open source, easy-to-install high performance Python and R distribution, with the conda package and environment manager and collection of 1,000+ open source packages with free community support.

Jupyter Notebook: - Web app that allows you to create and share documents that contain live code, equations, visualizations and explanatory text.

CHAPTER 3.0

SYSTEM DESIGN

3.1 Software Model Used

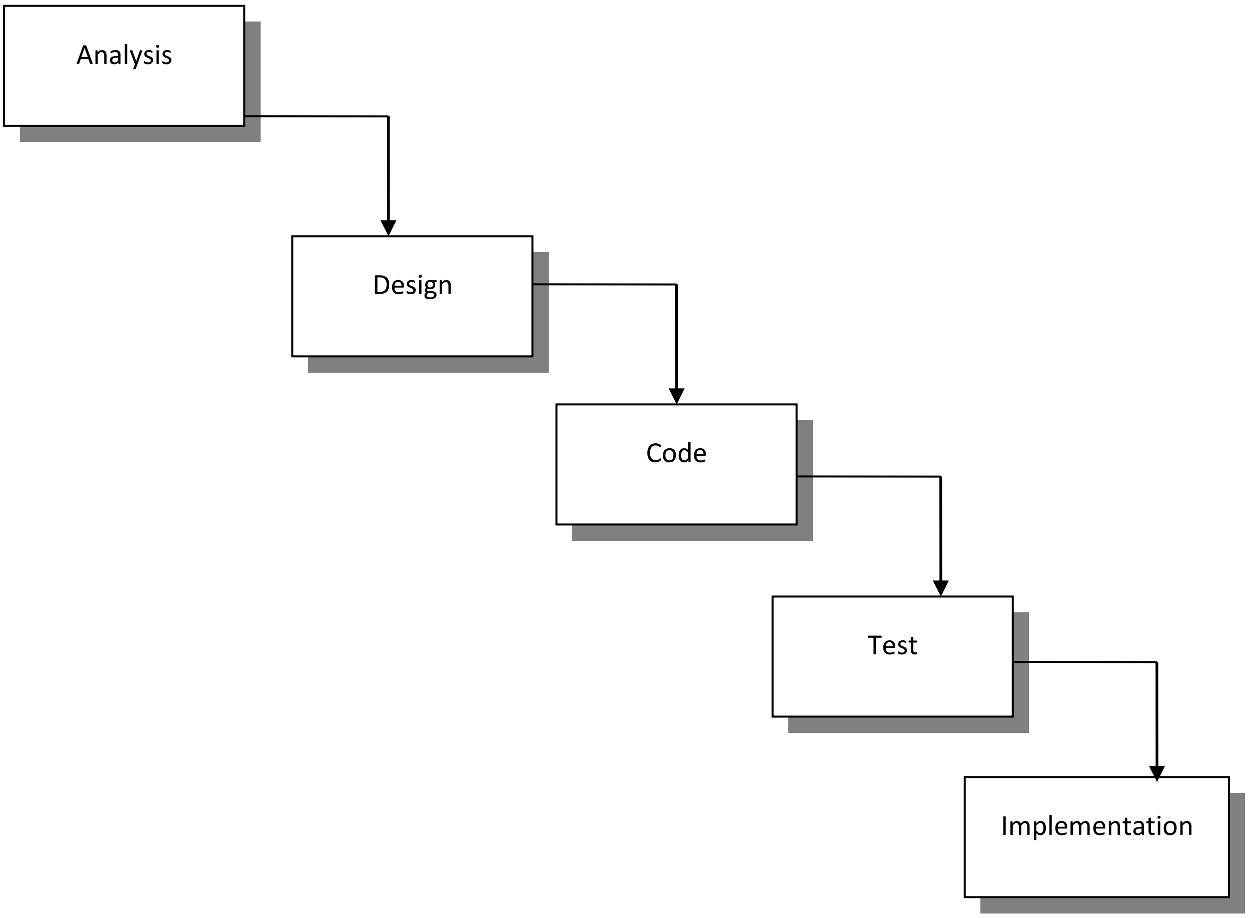

Figure 3.1

3.2 Data Description

Variable	Description
Loan_ID	Unique Loan ID
Gender	Male/ Female
Married	Applicant married (Y/N)
Dependents	Number of dependents
Education	Applicant Education (Graduate/ Under Graduate)
Self_Employed	Self employed (Y/N)
ApplicantIncome	Applicant income
CoapplicantIncome	Coapplicant income
LoanAmount	Loan amount in thousands
Loan_Amount_Term	Term of loan in months
Credit_History	credit history meets guidelines
Property_Area	Urban/ Semi Urban/ Rural
Loan_Status	Loan approved (Y/N)

3.3 Data Dictionary

Train.csv

Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
LP001002	Male	No	0	Graduate	No	5849	0		360	1	Urban	Y
LP001003	Male	Yes	1	Graduate	No	4583	1508	128	360	1	Rural	N
LP001005	Male	Yes	0	Graduate	Yes	3000	0	66	360	1	Urban	Y
LP001006	Male	Yes	0	Not Graduate	No	2583	2358	120	360	1	Urban	Y
LP001008	Male	No	0	Graduate	No	6000	0	141	360	1	Urban	Y
LP001011	Male	Yes	2	Graduate	Yes	5417	4196	267	360	1	Urban	Y
LP001013	Male	Yes	0	Not Graduate	No	2333	1516	95	360	1	Urban	Y
LP001014	Male	Yes	3+	Graduate	No	3036	2504	158	360	0	Semiurban	N
LP001018	Male	Yes	2	Graduate	No	4006	1526	168	360	1	Urban	Y
LP001020	Male	Yes	1	Graduate	No	12841	10968	349	360	1	Semiurban	N
LP001024	Male	Yes	2	Graduate	No	3200	700	70	360	1	Urban	Y
LP001027	Male	Yes	2	Graduate		2500	1840	109	360	1	Urban	Y
LP001028	Male	Yes	2	Graduate	No	3073	8106	200	360	1	Urban	Y
LP001029	Male	No	0	Graduate	No	1853	2840	114	360	1	Rural	N
LP001030	Male	Yes	2	Graduate	No	1299	1086	17	120	1	Urban	Y

Test.csv

Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area
LP001015	Male	Yes	0	Graduate	No	5720	0	110	360	1	Urban
LP001022	Male	Yes	1	Graduate	No	3076	1500	126	360	1	Urban
LP001031	Male	Yes	2	Graduate	No	5000	1800	208	360	1	Urban
LP001035	Male	Yes	2	Graduate	No	2340	2546	100	360		Urban
LP001051	Male	No	0	Not Graduate	No	3276	0	78	360	1	Urban
LP001054	Male	Yes	0	Not Graduate	Yes	2165	3422	152	360	1	Urban
LP001055	Female	No	1	Not Graduate	No	2226	0	59	360	1	Semiurban
LP001056	Male	Yes	2	Not Graduate	No	3881	0	147	360	0	Rural
LP001059	Male	Yes	2	Graduate		13633	0	280	240	1	Urban
LP001067	Male	No	0	Not Graduate	No	2400	2400	123	360	1	Semiurban
LP001078	Male	No	0	Not Graduate	No	3091	0	90	360	1	Urban
LP001082	Male	Yes	1	Graduate		2185	1516	162	360	1	Semiurban
LP001083	Male	No	3+	Graduate	No	4166	0	40	180		Urban
LP001094	Male	Yes	2	Graduate		12173	0	166	360	0	Semiurban
LP001096	Female	No	0	Graduate	No	4666	0	124	360	1	Semiurban

CHAPTER 4.0

Implementation Environment

4.1 Implementation Environment

The purpose of this project is to develop a platform were the bank's people /customer can check weather the can get the loan or not. The basic idea is to change the hand work to computerized work keep it record for both customer and bank's people.

Sometimes you come across small problems where you to need to go home and fetch the documents and then back to bank again. Or we can take it as the Bank's people to find the customer for the appropriate loan and the can guide them well to approach bank and get loan for customer's benefits.

4.2 Coding Standards

```
import pandas as pd
import numpy as np

from sklearn.metrics import roc_auc_score,accuracy_score
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier


trainData = pd.read_csv('H:/train.csv')

trainData.head(2)
trainData.shape
trainData.isnull().sum()
trainData.columns

#Handling with missing data

trainData.Gender.fillna(trainData.Gender.max(),inplace =True)
trainData.Married.fillna(trainData.Married.max(),inplace=True)
trainData.Credit_History.fillna(trainData.Credit_History.max(),inplace=True)
trainData.LoanAmount.fillna(trainData.LoanAmount.mean(),inplace=True)
trainData.Loan_Amount_Term.fillna(trainData.Loan_Amount_Term.mean(),inpl
ace=True)
trainData.Self_Employed.fillna(trainData.Self_Employed.max(),inplace=True)
trainData.Dependents.fillna(0,inplace=True)

#Convert string values to numerical values because to algorithm can understand
only numerical value not string values

trainData.Gender.value_counts()
```

```
gender_cat = pd.get_dummies(trainData.Gender,prefix='gender').gender_Female
trainData.Married.value_counts()
married_category =
pd.get_dummies(trainData.Married,prefix='marriage').marriage_Yes
trainData.Education.value_counts()
graduate_category =
pd.get_dummies(trainData.Education,prefix='education').education_Graduate
trainData.Self_Employed.value_counts()
self_emp_category =
pd.get_dummies(trainData.Self_Employed,prefix='employed').employed_Yes
loan_status = pd.get_dummies(trainData.Loan_Status,prefix='status').status_Y
property_category = pd.get_dummies(trainData.Property_Area,prefix='property')
trainData.shape

trainNew =
pd.concat([trainData,gender_cat,married_category,graduate_category,self_emp_
category,loan_status,property_category],axis=1)
trainNew.head()
trainNew.columns
feature_columns =
['ApplicantIncome','CoapplicantIncome','LoanAmount','Loan_Amount_Term','Cr
edit_History','gender_Female','marriage_Yes','education_Graduate','employed_Y
es','property_Rural','property_Semiurban','property_Urban']

X = trainNew[feature_columns]
y =  trainNew['status_Y']
y

from sklearn.cross_validation import train_test_split

X_train,X_test,y_train,y_test =
train_test_split(X,y,test_size=0.01,random_state=42)
X_train.shape
X_test.shape

randForest = RandomForestClassifier(n_estimators=25, min_samples_split=25,
max_depth=7, max_features=1)
randForest.fit(X_train,y_train)
y_pred_class  = randForest.predict(X_test)
randForestScore = accuracy_score(y_test,y_pred_class)
get_ipython().magic(u'time print "Random forest accuraccy
score",randForestScore')

#Import test data and do real test of our model
randForestNew = RandomForestClassifier(n_estimators=25,
min_samples_split=25, max_depth=7, max_features=1)
randForestNew.fit(X,y)
testData = pd.read_csv('H:/test.csv')
testData.shape
testData.head()
```

```
testData.isnull().sum()
testData.Gender.fillna(testData.Gender.max(),inplace =True)
testData.Married.fillna(testData.Married.max(),inplace=True)
testData.Credit_History.fillna(testData.Credit_History.max(),inplace=True)
testData.LoanAmount.fillna(testData.LoanAmount.mean(),inplace=True)
testData.Loan_Amount_Term.fillna(testData.Loan_Amount_Term.mean(),inplace=True)
testData.Self_Employed.fillna(testData.Self_Employed.max(),inplace=True)
testData.Dependents.fillna(0,inplace=True)

gender_cat = pd.get_dummies(testData.Gender,prefix='gender').gender_Female
married_category = pd.get_dummies(testData.Married,prefix='marriage').marriage_Yes
graduate_category = pd.get_dummies(testData.Education,prefix='education').education_Graduate
self_emp_category = pd.get_dummies(testData.Self_Employed,prefix='employed').employed_Yes
property_category = pd.get_dummies(testData.Property_Area,prefix='property')

testDataNew = pd.concat([testData,gender_cat,married_category,graduate_category,self_emp_category,property_category],axis=1)

X_testData = testDataNew[feature_columns]

X_testData.head()

y_test_pread_class = randForestNew.predict(X_testData)

randForestFormat = ["Y" if i == 1 else "N" for i in y_test_pread_class ]

pd.DataFrame({'Loan_ID':testData.Loan_ID,'Loan_Status':randForestFormat}).to_csv('radom_forest_submission.csv',index=False)

#Solve using logistic regression

from sklearn.linear_model import LogisticRegression

logReg = LogisticRegression()
logReg.fit(X_train,y_train)
logREg_predict =logReg.predict(X_test)
accuracy_score(y_test,logREg_predict)

logReg_y_prediction_class = logReg.predict(X_testData)

logRegPredictionFormat = ["Y" if i == 1 else "N" for i in logReg_y_prediction_class ]

#zip(logRegPredictionFormat,logReg_y_prediction_class)
```

```
pd.DataFrame({'Loan_ID':testData.Loan_ID,'Loan_Status':logRegPredictionFormat}).to_csv('logReg_submission.csv',index=False)
```

4.4 Snapshots of project

```
In [3]: import pandas as pd
        import numpy as np

        from sklearn.metrics import roc_auc_score,accuracy_score

        from sklearn.linear_model import LogisticRegression
        from sklearn.ensemble import RandomForestClassifier

In [4]: trainData = pd.read_csv('H:/train.csv')

In [5]: trainData.head(2)

Out[5]:
```

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History
0	LP001002	Male	No	0	Graduate	No	5849	0.0	NaN	360.0	1.0
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.0	1.0

```
In [6]: trainData.shape

Out[6]: (614, 13)

In [7]: trainData.isnull().sum()

Out[7]: Loan_ID              0
        Gender              13
        Married              3
        Dependents          15
        Education            0
        Self_Employed       32
        ApplicantIncome      0
        CoapplicantIncome    0
        LoanAmount          22
        Loan_Amount_Term    14
        Credit_History      50
        Property_Area        0
        Loan_Status          0
        dtype: int64

In [8]: trainData.columns

Out[8]: Index([u'Loan_ID', u'Gender', u'Married', u'Dependents', u'Education',
               u'Self_Employed', u'ApplicantIncome', u'CoapplicantIncome',
               u'LoanAmount', u'Loan_Amount_Term', u'Credit_History', u'Property_Area',
               u'Loan_Status'],
              dtype='object')

In [9]: #Handling with missing data

In [10]: trainData.Gender.fillna(trainData.Gender.max(),inplace =True)

In [11]: trainData.Married.fillna(trainData.Married.max(),inplace=True)

In [12]: trainData.Credit_History.fillna(trainData.Credit_History.max(),inplace=True)

In [13]: trainData.LoanAmount.fillna(trainData.LoanAmount.mean(),inplace=True)

In [14]: trainData.Loan_Amount_Term.fillna(trainData.Loan_Amount_Term.mean(),inplace=True)

In [15]: trainData.Self_Employed.fillna(trainData.Self_Employed.max(),inplace=True)

In [16]: trainData.Dependents.fillna(0,inplace=True)

In [17]: #Convert string values to numerical values because to algorithm can understand only numerical value not string values

In [18]: trainData.Gender.value_counts()
         gender_cat = pd.get_dummies(trainData.Gender,prefix='gender').gender_Female

In [19]: trainData.Married.value_counts()
         married_category = pd.get_dummies(trainData.Married,prefix='marriage').marriage_Yes
```

In [17]:
```
#Convert string values to numerical values because to algorithm can understand only numerical value not string values
```

In [18]:
```
trainData.Gender.value_counts()
gender_cat = pd.get_dummies(trainData.Gender,prefix='gender').gender_Female
```

In [19]:
```
trainData.Married.value_counts()
married_category = pd.get_dummies(trainData.Married,prefix='marriage').marriage_Yes
```

In [20]:
```
trainData.Education.value_counts()
graduate_category = pd.get_dummies(trainData.Education,prefix='education').education_Graduate
```

In [21]:
```
trainData.Self_Employed.value_counts()
self_emp_category = pd.get_dummies(trainData.Self_Employed,prefix='employed').employed_Yes
```

In [22]:
```
loan_status = pd.get_dummies(trainData.Loan_Status,prefix='status').status_Y
```

In [23]:
```
property_category = pd.get_dummies(trainData.Property_Area,prefix='property')
```

In [24]:
```
trainData.shape
```

Out[24]:
```
(614, 13)
```

In [26]:
```
trainNew.head()
```

Out[26]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	...	Property_Ar
0	LP001002	Male	No	0	Graduate	No	5849	0.0	146.412162	360.0	...	Urt
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.000000	360.0	...	Ru
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.000000	360.0	...	Urt
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.000000	360.0	...	Urt
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.000000	360.0	...	Urt

5 rows × 21 columns

In [32]:
```
X_train.shape
```

Out[32]:
```
(607, 12)
```

In [33]:
```
X_test.shape
```

Out[33]:
```
(7, 12)
```

In [34]:
```
randForest = RandomForestClassifier(n_estimators=25, min_samples_split=25, max_depth=7, max_features=1)
randForest.fit(X_train,y_train)
y_pred_class  = randForest.predict(X_test)
randForestScore = accuracy_score(y_test,y_pred_class)
%time print "Random forest accuraccy score",randForestScore
```

```
Random forest accuraccy score 1.0
Wall time: 17 ms
```

In [49]:
```
#Solve using logistic regression
```

In [50]:
```
from sklearn.linear_model import LogisticRegression
logReg = LogisticRegression()
logReg.fit(X_train,y_train)
logREg_predict =logReg.predict(X_test)
accuracy_score(y_test,logREg_predict)
```

Out[50]:
```
1.0
```

In [51]:
```
logReg_y_prediction_class = logReg.predict(X_testData)
```

In [52]:
```
logRegPredictionFormat = ["Y" if i == 1 else "N" for i in logReg_y_prediction_class ]
```

In [53]:
```
#zip(logRegPredictionFormat,logReg_y_prediction_class)
```

In [54]:
```
pd.DataFrame({'Loan_ID':testData.Loan_ID,'Loan_Status':logRegPredictionFormat}).to_csv('logReg_submission.csv',index=False)
```

CHAPTER 5

Limitation and Future Enhancement

5.1 Limitation:

So many tools and techniques have been used to develop the software according to their requirements. It is not a complete project with its own. But at least there are few limitations of the project which are described below-

From the user's point of view:

- ✓ There is no provision to make an additional information.
- ✓ There is no specific user Interface for them to do.

From the programmers point of view:

- ✓ We have used the Python 2.7 instead we can use python 3.5.
- ✓ We have used the Anaconda 2, which has limited functionality then Anaconda 4.
- ✓ There are less parameters compare to actual parameters.
- ✓ No specific format is followed.

5.2 Future Enhancements:

After the development of any project, there is some specific points gets missed. Because we work at the time of development by considering only required problems. But in real life project working we get lots of requirements in our developed project, and then enhancement comes in near future. Some of the important point which is to be considering in enhancement are as follows:-

- ✓ As per development, the main considering point for enhancement is to enable the developed software to achieve “User Interface”.
- ✓ And few more parameters can be added to make it more Accurate and efficient.

CHAPTER 6

Conclusion

The package “Loan Prediction” being developed to help and assist the people to know that weather they are can get the loan or not. The system saves lot of time and effort of the same. Still it need some upgradation in terms of GUI but we can run that in console based.

CHAPTER 7

References

1. https://www.datasciencecentral.com/m/blogpost?id=6448529:BlogPost:434520
2. https://www.python.org/about/gettingstarted/
3. https://www.analyticsvidhya.com/blog/2016/01/complete-tutorial-learn-data-science-python-scratch-2/
4. http://pandas.pydata.org/pandas-docs/stable/10min.html
5. https://www.analyticsvidhya.com/blog/2016/01/12-pandas-techniques-python-data-manipulation/
6. https://www.analyticsvidhya.com/blog/2015/01/scikit-learn-python-machine-learning-tool/
7. https://www.analyticsvidhya.com/blog/2017/09/common-machine-learning-algorithms/
8. https://www.analyticsvidhya.com/blog/2015/11/improve-model-performance-cross-validation-in-python-r/
9. https://www.analyticsvidhya.com/blog/2015/11/beginners-guide-on-logistic-regression-in-r/
10. https://www.analyticsvidhya.com/blog/2015/09/random-forest-algorithm-multiple-challenges/