Loan Prediction Using Decision Tree

January 22, 2020

[1]:
```
# importing basic libraries
import numpy as np
import pandas as pd
from pandas import Series, DataFrame
from sklearn.model_selection import train_test_split
from sklearn import tree
from sklearn import metrics
from sklearn import preprocessing
#creating labelEncoder
le = preprocessing.LabelEncoder()

# load datasets
train = pd.read_csv("train.csv")
test = pd.read_csv("test.csv")
train
```

[1]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	\
0	LP001002	Male	No	0	Graduate	No	
1	LP001003	Male	Yes	1	Graduate	No	
2	LP001005	Male	Yes	0	Graduate	Yes	
3	LP001006	Male	Yes	0	Not Graduate	No	
4	LP001008	Male	No	0	Graduate	No	
5	LP001011	Male	Yes	2	Graduate	Yes	
6	LP001013	Male	Yes	0	Not Graduate	No	
7	LP001014	Male	Yes	3+	Graduate	No	
8	LP001018	Male	Yes	2	Graduate	No	
9	LP001020	Male	Yes	1	Graduate	No	
10	LP001024	Male	Yes	2	Graduate	No	
11	LP001027	Male	Yes	2	Graduate	NaN	
12	LP001028	Male	Yes	2	Graduate	No	
13	LP001029	Male	No	0	Graduate	No	
14	LP001030	Male	Yes	2	Graduate	No	
15	LP001032	Male	No	0	Graduate	No	
16	LP001034	Male	No	1	Not Graduate	No	
17	LP001036	Female	No	0	Graduate	No	
18	LP001038	Male	Yes	0	Not Graduate	No	
19	LP001041	Male	Yes	0	Graduate	NaN	

```
20   LP001043    Male     Yes          0  Not Graduate            No
21   LP001046    Male     Yes          1      Graduate            No
22   LP001047    Male     Yes          0  Not Graduate            No
23   LP001050     NaN     Yes          2  Not Graduate            No
24   LP001052    Male     Yes          1      Graduate           NaN
25   LP001066    Male     Yes          0      Graduate           Yes
26   LP001068    Male     Yes          0      Graduate            No
27   LP001073    Male     Yes          2  Not Graduate            No
28   LP001086    Male      No          0  Not Graduate            No
29   LP001087  Female      No          2      Graduate           NaN
..        ...     ...     ...        ...           ...           ...
584  LP002911    Male     Yes          1      Graduate            No
585  LP002912    Male     Yes          1      Graduate            No
586  LP002916    Male     Yes          0      Graduate            No
587  LP002917  Female      No          0  Not Graduate            No
588  LP002925     NaN      No          0      Graduate            No
589  LP002926    Male     Yes          2      Graduate           Yes
590  LP002928    Male     Yes          0      Graduate            No
591  LP002931    Male     Yes          2      Graduate           Yes
592  LP002933     NaN      No         3+      Graduate           Yes
593  LP002936    Male     Yes          0      Graduate            No
594  LP002938    Male     Yes          0      Graduate           Yes
595  LP002940    Male      No          0  Not Graduate            No
596  LP002941    Male     Yes          2  Not Graduate           Yes
597  LP002943    Male      No        NaN      Graduate            No
598  LP002945    Male     Yes          0      Graduate           Yes
599  LP002948    Male     Yes          2      Graduate            No
600  LP002949  Female      No         3+      Graduate           NaN
601  LP002950    Male     Yes          0  Not Graduate           NaN
602  LP002953    Male     Yes         3+      Graduate            No
603  LP002958    Male      No          0      Graduate            No
604  LP002959  Female     Yes          1      Graduate            No
605  LP002960    Male     Yes          0  Not Graduate            No
606  LP002961    Male     Yes          1      Graduate            No
607  LP002964    Male     Yes          2  Not Graduate            No
608  LP002974    Male     Yes          0      Graduate            No
609  LP002978  Female      No          0      Graduate            No
610  LP002979    Male     Yes         3+      Graduate            No
611  LP002983    Male     Yes          1      Graduate            No
612  LP002984    Male     Yes          2      Graduate            No
613  LP002990  Female      No          0      Graduate           Yes

     ApplicantIncome  CoapplicantIncome  LoanAmount  Loan_Amount_Term  \
0               5849                0.0         NaN             360.0
1               4583             1508.0       128.0             360.0
2               3000                0.0        66.0             360.0
3               2583             2358.0       120.0             360.0
```

4	6000	0.0	141.0	360.0
5	5417	4196.0	267.0	360.0
6	2333	1516.0	95.0	360.0
7	3036	2504.0	158.0	360.0
8	4006	1526.0	168.0	360.0
9	12841	10968.0	349.0	360.0
10	3200	700.0	70.0	360.0
11	2500	1840.0	109.0	360.0
12	3073	8106.0	200.0	360.0
13	1853	2840.0	114.0	360.0
14	1299	1086.0	17.0	120.0
15	4950	0.0	125.0	360.0
16	3596	0.0	100.0	240.0
17	3510	0.0	76.0	360.0
18	4887	0.0	133.0	360.0
19	2600	3500.0	115.0	NaN
20	7660	0.0	104.0	360.0
21	5955	5625.0	315.0	360.0
22	2600	1911.0	116.0	360.0
23	3365	1917.0	112.0	360.0
24	3717	2925.0	151.0	360.0
25	9560	0.0	191.0	360.0
26	2799	2253.0	122.0	360.0
27	4226	1040.0	110.0	360.0
28	1442	0.0	35.0	360.0
29	3750	2083.0	120.0	360.0
..	...	...	...	...
584	2787	1917.0	146.0	360.0
585	4283	3000.0	172.0	84.0
586	2297	1522.0	104.0	360.0
587	2165	0.0	70.0	360.0
588	4750	0.0	94.0	360.0
589	2726	0.0	106.0	360.0
590	3000	3416.0	56.0	180.0
591	6000	0.0	205.0	240.0
592	9357	0.0	292.0	360.0
593	3859	3300.0	142.0	180.0
594	16120	0.0	260.0	360.0
595	3833	0.0	110.0	360.0
596	6383	1000.0	187.0	360.0
597	2987	0.0	88.0	360.0
598	9963	0.0	180.0	360.0
599	5780	0.0	192.0	360.0
600	416	41667.0	350.0	180.0
601	2894	2792.0	155.0	360.0
602	5703	0.0	128.0	360.0
603	3676	4301.0	172.0	360.0

604	12000	0.0	496.0	360.0
605	2400	3800.0	NaN	180.0
606	3400	2500.0	173.0	360.0
607	3987	1411.0	157.0	360.0
608	3232	1950.0	108.0	360.0
609	2900	0.0	71.0	360.0
610	4106	0.0	40.0	180.0
611	8072	240.0	253.0	360.0
612	7583	0.0	187.0	360.0
613	4583	0.0	133.0	360.0

	Credit_History	Property_Area	Loan_Status
0	1.0	Urban	Y
1	1.0	Rural	N
2	1.0	Urban	Y
3	1.0	Urban	Y
4	1.0	Urban	Y
5	1.0	Urban	Y
6	1.0	Urban	Y
7	0.0	Semiurban	N
8	1.0	Urban	Y
9	1.0	Semiurban	N
10	1.0	Urban	Y
11	1.0	Urban	Y
12	1.0	Urban	Y
13	1.0	Rural	N
14	1.0	Urban	Y
15	1.0	Urban	Y
16	NaN	Urban	Y
17	0.0	Urban	N
18	1.0	Rural	N
19	1.0	Urban	Y
20	0.0	Urban	N
21	1.0	Urban	Y
22	0.0	Semiurban	N
23	0.0	Rural	N
24	NaN	Semiurban	N
25	1.0	Semiurban	Y
26	1.0	Semiurban	Y
27	1.0	Urban	Y
28	1.0	Urban	N
29	1.0	Semiurban	Y
..	...	...	...
584	0.0	Rural	N
585	1.0	Rural	N
586	1.0	Urban	Y
587	1.0	Semiurban	Y

```
588             1.0     Semiurban           Y
589             0.0     Semiurban           N
590             1.0     Semiurban           Y
591             1.0     Semiurban           N
592             1.0     Semiurban           Y
593             1.0         Rural           Y
594             1.0         Urban           Y
595             1.0         Rural           Y
596             1.0         Rural           N
597             0.0     Semiurban           N
598             1.0         Rural           Y
599             1.0         Urban           Y
600             NaN         Urban           N
601             1.0         Rural           Y
602             1.0         Urban           Y
603             1.0         Rural           Y
604             1.0     Semiurban           Y
605             1.0         Urban           N
606             1.0     Semiurban           Y
607             1.0         Rural           Y
608             1.0         Rural           Y
609             1.0         Rural           Y
610             1.0         Rural           Y
611             1.0         Urban           Y
612             1.0         Urban           Y
613             0.0     Semiurban           N

[614 rows x 13 columns]
```

Converting train data string labels into numbers and filling Na values of Item_Weight By Mean Values According to Fat_Content.

[2]:
```
train['Education'] = le.fit_transform(train['Education'])
train['Education'].value_counts()
train['LoanAmount'] = train.groupby('Education')['LoanAmount'].transform(lambda␣
 ↪x: x.fillna(x.mean()))
```

[3]:
```
train['Property_Area'] = le.fit_transform(train['Property_Area'])
train['Self_Employed'] = train['Self_Employed'].fillna(train['Self_Employed'].
 ↪mode()[0],inplace=True)
train['Credit_History'].fillna(1, inplace=True)
train['Self_Employed'] = le.fit_transform(train['Self_Employed'])
train['Loan_Status'] = le.fit_transform(train['Loan_Status'])
train['Loan_Amount_Term'] = train['Loan_Amount_Term'].fillna(360,inplace=True)
train['Loan_Amount_Term'] = le.fit_transform(train['Loan_Amount_Term'])
```

[4]:
```
#Specifyin input features and output features
feat_colums = ['Education', 'Self_Employed',␣
 ↪'ApplicantIncome','CoapplicantIncome','Credit_History','Loan_Amount_Term','Property_Area']
```

```
X = train[feat_colums]
y = train.Loan_Status
#Splitting train and test in train dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=1)
```

[5]:
```
#Creating Tree Object With Entropy Criterias
model = tree.DecisionTreeClassifier(criterion='entropy',max_depth = 5)
#Decision Tree Classifier
model = model.fit(X_train,y_train)
```

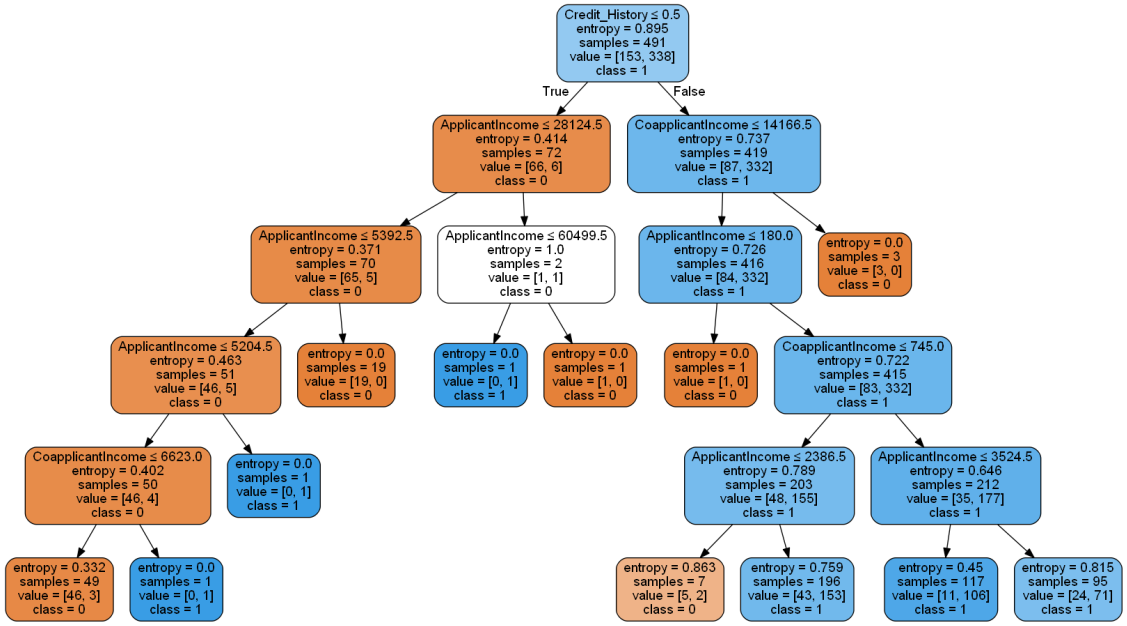
[6]:
```
#Predict the response for test dataset
y_pred = model.predict(X_test)
# Model Accuracy, how often is the classifier correct?
print("Accuracy Using Entropy Criterion:",metrics.accuracy_score(y_test,␣
 ↪y_pred))
```

```
Accuracy Using Entropy Criterion: 0.7967479674796748
```

[7]:
```
#Plotting The Decision Tree
from sklearn.externals.six import StringIO
from IPython.display import Image
from sklearn.tree import export_graphviz
import pydotplus
import os
os.environ['PATH'] = os.environ['PATH']+';'+os.
 ↪environ['CONDA_PREFIX']+r"\Library\bin\graphviz"
dot_data = StringIO()
export_graphviz(model, out_file=dot_data,
                filled=True, rounded=True,
                special_characters=True, feature_names =␣
 ↪feat_colums,class_names=['0','1'])
graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
graph.write_png('prediction.png')
Image(graph.create_png())
```

```
C:\Users\Trilo\Anaconda3\lib\site-packages\sklearn\externals\six.py:31:
DeprecationWarning: The module is deprecated in version 0.21 and will be removed
in version 0.23 since we've dropped support for Python 2.7. Please rely on the
official version of six (https://pypi.org/project/six/).
  "(https://pypi.org/project/six/).", DeprecationWarning)
```

[7]:



[8]:
```
test['Education'] = le.fit_transform(test['Education'])
test['Education'].value_counts()
test['LoanAmount'] = test.groupby('Education')['LoanAmount'].transform(lambda x:
 ↪ x.fillna(x.mean()))
```

[9]:
```
test['Property_Area'] = le.fit_transform(test['Property_Area'])
test['Self_Employed'] = train['Self_Employed'].fillna(test['Self_Employed'].
 ↪mode()[0],inplace=True)
test['Credit_History'].fillna(1, inplace=True)
test['Self_Employed'] = le.fit_transform(test['Self_Employed'])
feat_colums = ['Education', 'Self_Employed',␣
 ↪'ApplicantIncome','CoapplicantIncome','Credit_History','Property_Area']
X = test[feat_colums]
test['Loan_Status'] = model.predict(X)
test['Loan_Status'].replace([0,1],['No','Yes'],inplace=True)
test[['Loan_ID','Loan_Status']].to_csv("loan_prediction.csv")
```

```
      ␣
↪---------------------------------------------------------------------------

      ValueError                                Traceback (most recent call␣
↪last)

      <ipython-input-9-7ab68d4f110b> in <module>
        5 feat_colums = ['Education', 'Self_Employed',␣
↪'ApplicantIncome','CoapplicantIncome','Credit_History','Property_Area']
```

```
      6 X = test[feat_colums]
----> 7 test['Loan_Status'] = model.predict(X)
      8 test['Loan_Status'].replace([0,1],['No','Yes'],inplace=True)
      9 test[['Loan_ID','Loan_Status']].to_csv("loan_prediction.csv")


      ~\Anaconda3\lib\site-packages\sklearn\tree\tree.py in predict(self, X,␣
↪check_input)
      428         """
      429         check_is_fitted(self, 'tree_')
--> 430         X = self._validate_X_predict(X, check_input)
      431         proba = self.tree_.predict(X)
      432         n_samples = X.shape[0]


      ~\Anaconda3\lib\site-packages\sklearn\tree\tree.py in␣
↪_validate_X_predict(self, X, check_input)
      400                                 "match the input. Model n_features is␣
↪%s and "
      401                                 "input n_features is %s "
--> 402                                 % (self.n_features_, n_features))
      403
      404         return X


      ValueError: Number of features of the model must match the input. Model␣
↪n_features is 7 and input n_features is 6
```