

CSC 560 Data Science Project -1

Project Report submission

**Team Members:**

**Kiran Mai Bandaru 94966**

**Kavya Janga 94814**

**Pavani Eupuri 94850 Prof: Alex Wu**

**Dataset Name:** HOME MORTGAGE DISCLOSURE ACT data set

**Model chosen:** Decision Tree Classifier

**Objective:** To classify the loan action taken depending on applicant-income, loan-amount, loan Type, loan purpose and agency

**Importing Required Libraries:**

﻿# Importing the libraries

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn import tree

from sklearn import metrics

import pydotplus

from IPython.display import Image

**Command to Install pydotplus through anaconda prompt:**

conda install -c conda-forge pydotplus

**Loading Data:** after importing required libraries load the required dataset using pandas’ read CSV function.

# Importing the dataset

dataset = pd.read\_csv('hmda\_lar.csv')

**Feature Selection:** Here we have select columns and identify the columns with null values and fill the null values.

print(dataset.isnull().sum()) # outputs the columns with null values

# column "applicant\_income" has null values. Replace missing values with the mean

dataset['applicant\_income\_000s'].fillna(dataset['applicant\_income\_000s'].mean(), inplace=True)

Excluded the attributes which have null values for more than 300 out of 500 observations. From the remaining observations we have identified the following features to analyse our objective.

These are the independent variables selected from the dataset

﻿features = ['applicant\_income\_000s','loan\_amount\_000s','loan\_type','loan\_purpose','agency\_code']

and below is the dependant variable

target = ['action\_taken']

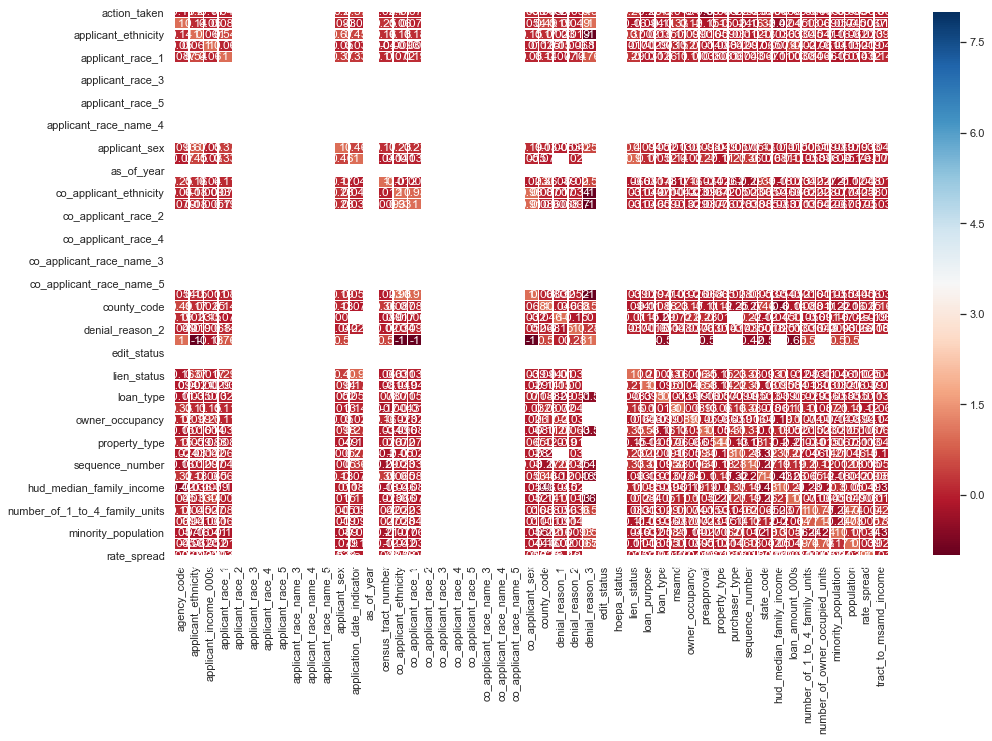
Heatmap to find correlation between attributes:

import seaborn as sns

corr = dataset.corr()

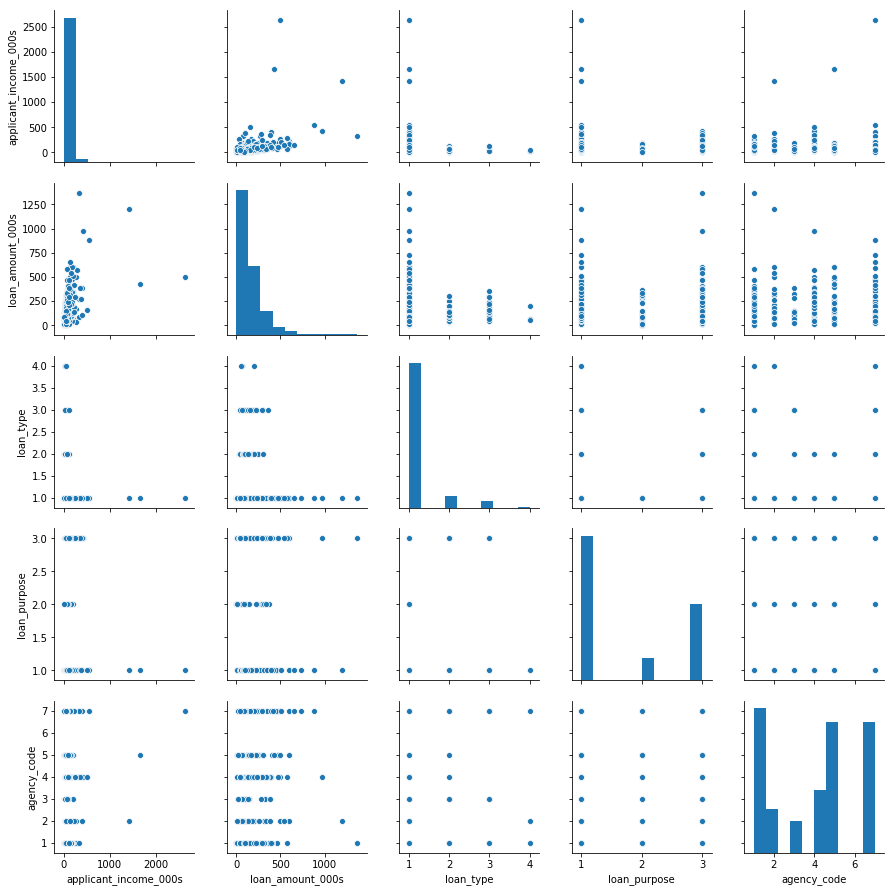
fig, ax = plt.subplots(figsize=(16,10))

sns.heatmap(corr.iloc[:, 1:78:], annot = True, linewidth = 0.5, vmax= 8, cmap ='RdBu')



**Feature analysis:**

correlation output on the independant features.



**Splitting Data:**

To understand model performance, divide the dataset into training set(75%) and test set(25%). Training set to fit the model and test set to verify the performance.

We used sklearn.model\_selection.train\_test\_split to split the dataset.

#Split test and train data

X = dataset[features]

y = dataset[target]

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.25, random\_state=0)

**Fitting The Model:**

clf = tree.DecisionTreeClassifier(criterion="entropy",max\_depth=5)

clf.fit(X\_train, Y\_train)

y\_pred=clf.predict(X\_test)

**Evaluating Model:** Accuracy can be computed by comparing actual test set values and predicted values.

Confusion Matrix

from sklearn.metrics import confusion\_matrix

cm = confusion\_matrix(y\_test, y\_pred)

print(cm)

A screenshot of a cell phone

Description automatically generated

ACCURACY = sum of correct classifications/ total no of classifications

﻿Decision Tree Accuracy: 0.512

A screenshot of a social media post

Description automatically generated

**Visualizing Decision trees:** we used scikit-learn's export\_graphviz function to visualize the tree.

# View the decision tree

import pydotplus

from IPython.display import Image

data\_feature\_names = ['Applicant Income', 'Loan Amount','Loan Type','Loan Purpose','Agency']

plotData=tree.export\_graphviz(clf, out\_file=None, filled=True, rounded=True, special\_characters=True,

               feature\_names= data\_feature\_names)

graph = pydotplus.graph\_from\_dot\_data(plotData)

graph.write\_png('DecisionTree.png')

Image(graph.create\_png())

**Decision Tree Visualization:** with criterion – entropy and max\_depth = 5

A picture containing text

Description automatically generated

**Decision Tree Visualization:** with criterion – entropy and no depth parameter

A picture containing screenshot

Description automatically generated

When depth parameter is not mentioned the entropy of all the terminal nodes are “0” because the split recursively takes place till the terminal node has the entropy of “0”.

The more the depth, more the no.of splits and the possibility of over-fitting.

**Conclusion:**

The model resulted in only 50% accuracy. In this case random forest classifier would be a better classifier instead of a single decision tree.

A large number of relatively uncorrelated models together can produce predictions that are more accurate than any individual model.

Reference:

<https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html>

Dataset Link:

[**https://docs.google.com/spreadsheets/d/16zviDzVYK6H-aUP-Z6qonYPt7Ui62o8DwfjzbNfn17k/edit#gid=453885575**](https://docs.google.com/spreadsheets/d/16zviDzVYK6H-aUP-Z6qonYPt7Ui62o8DwfjzbNfn17k/edit#gid=453885575)