# Part 2 Classification

* Programmatically downloads Q12005 and Q22005 origination data and pre-processes it.

import requests

import urllib

import os

from bs4 import BeautifulSoup

import http.cookiejar

import shutil

import pandas as pd

url = 'https://freddiemac.embs.com/FLoan/Data/download.php'

cookie={'PHPSESSID':'11vr6kgl2a9lt7b8if0mv08vq0'}

r = requests.post(url,cookies=cookie)

content=r.content

soup = BeautifulSoup(content,'lxml')

all\_href = soup.find\_all('a')

i=0;

dww=['Q12005','Q22005']

lod=len(dww)

url\_list=['https://freddiemac.embs.com/FLoan/Data/']\*lod

for href in all\_href:

for s in dww:

if s in href['href']:

url\_list[i]=url\_list[i]+href['href']

i+=1

if (not os.path.exists('datapart2')):

os.mkdir('datapart2')

os.chdir('datapart2')

for i in range(len(url\_list)):

r = requests.get(url\_list[i],cookies=cookie)

with open(url\_list[i][71:77]+'.zip','wb') as code:

code.write(r.content)

files= os.listdir()

for file in files:

for docu in dww:

if ('.zip' in file) and (docu in file):

shutil.unpack\_archive(file)

os.chdir('..')

Use cookies to get access to the download page, use BeautifulSoup to get content of the webpage, find the link we need, download the zip file, then unzip it by shutil package.

def preprocessing(data):

data['Delinquent']=[0 if x == '0' else 1 for x in data['CURRENT LOAN DELINQUENCY STATUS']]

data.drop(['LOAN SEQUENCE NUMBER','REPURCHASE FLAG','CURRENT LOAN DELINQUENCY STATUS','MODIFICATION FLAG','ZERO BALANCE CODE'

,'ZERO BALANCE EFFECTIVE DATE','DUE DATE OF LAST PAID INSTALLMENT (DDLPI)','MI RECOVERIES'

,'NET SALES PROCEEDS','NON MI RECOVERIES','EXPENSES','LEGAL COSTS','MAINTENANCE AND PRESERVATION COSTS'

,'TAXES AND INSURANCE','MISCELLANEOUS EXPENSES','ACTUAL LOSS CALCULATION','MODIFICATION COST'

,'STEP MODIFICATION FLAG','ESTIMATED LOAN TO VALUE (ELTV)'],axis=1,inplace=True)

dictionary\_25 = {'Y':1,'N':0,' ':-1}

data['DEFERRED PAYMENT MODIFICATION'] = [dictionary\_25[x] for x in data['DEFERRED PAYMENT MODIFICATION']]

return data

Do necessary preprocessing steps.

* Builds a Logistic regression model for the CURRENT LOAN DELINQUENCY STATUS, using Q12005 data as training data (col 4). Note anytime col 4 is > 0, add a new variable as Delinquent.

data['Delinquent']=[0 if x == '0' else 1 for x in data['CURRENT LOAN DELINQUENCY STATUS']]

Generate ‘Delinquent’ column by the CURRENT LOAN DELINQUENCY STATUS, ‘0’ present not delinquent, ‘1’ present delinquent.

logi = LogisticRegression()

logi.fit(train\_x,train\_y)

pred\_y = logi.predict(test\_x)

cm=confusion\_matrix(test\_y,pred\_y)

Build logistic model and train it with Q12005 data. Validate against Q22005, use the confusion matrix to show the performance.

* Validates against Q22005 data and selects the best Classification model. Computes ROC curve and Confusion matrices for training and testing datasets. Repeat this using Random Forest & Neural Network algorithms.Try TPOT, H20.Ai and AutoSKLearn Automl algorithms. Choose the best model amongst the different types of algorithms.

We tried Logistic Regression, SVM, Decision Tree, Naïve Bayes, KNN, Random Forest, Extra Tree, MLP, H2o.ai, TPOT. The best model we got is Random Forest.

rf = RandomForestClassifier(max\_depth=7,n\_estimators=100,verbose=2)

rf.fit(train\_x,train\_y)

pred\_y = rf.predict(test\_x)

cm=confusion\_matrix(test\_y,pred\_y)

tn, fp, fn, tp = cm.ravel()

print(cm)

print(tn, fp, fn, tp)

[[285663 463]

[ 13326 548]]

285663 463 13326 548

prob\_y=rf.predict\_proba(test\_x)

fpr, tpr, thresholds = metrics.roc\_curve(test\_y, prob\_y[:,1],pos\_label=1)

plt.plot(fpr, tpr, 'b')A screenshot of a cell phone

Description generated with very high confidence

* Parameterize the input (example it should take Q12005) and modify the code so that it outputs the 5 parameters listed in the matrix below.

from configparser import ConfigParser

cfg = ConfigParser()

cfg.read('classification.config')

value = cfg.get('cookie','value')

docu = cfg.get('file','name')

Use configparser to parameterize the input.

print('Number of Actual Delinquents:', tp+fn)

print('Number of Predicted Delinquents:', tp+fp)

print('Number of records in the dataset:', tn+tp+fn+fp)

print('Number of Delinquents properly classified:', tp)

print('Number of nondelinquents improperly classified as delinquents:', fp)

Compute the five columns by confusion matrix.

* Write another script that calls the above classification script from Q11999-Q42016 and computes the following matrix.

# -\*- coding: utf-8 -\*-

"""

Created on Thu Nov 29 01:26:05 2018

@author: wenqi

"""

import pandas as pd

import gc

from cla\_main import main\_function

filelist=[

'Q11999',

'Q21999',

'Q31999',

'Q41999',

'Q12000',

'Q22000',

'Q32000',

'Q42000',

]

value='11vr6kgl2a9lt7b8if0mv08vq0'

df = pd.DataFrame(columns = ["Actual Delinquents", "Predicted Delinquents", "Records in the dataset"

, "Delinquents properly classified","nondelinquents improperly classified as delinquents"])

for file in filelist:

tn, fp, fn, tp=main\_function(file,value)

df.loc[df.shape[0]+1]=[tp+fn,tp+fp,tn+tp+fn+fp,tp,fp]

gc.collect()

filelist2=[

'Q21999',

'Q31999',

'Q41999',

'Q12000',

'Q22000',

'Q32000',

'Q42000',

'Q12001',

]

df["Quarter"]=filelist2

print(df)

df.to\_csv('matrix.csv')

* Document your design and results in the report. Comment on the quality of the model and it’s outputs. What can you do to do better? Would you recommend to use to evaluate the performance of your algorithm? Compute those metrics for Q12016.