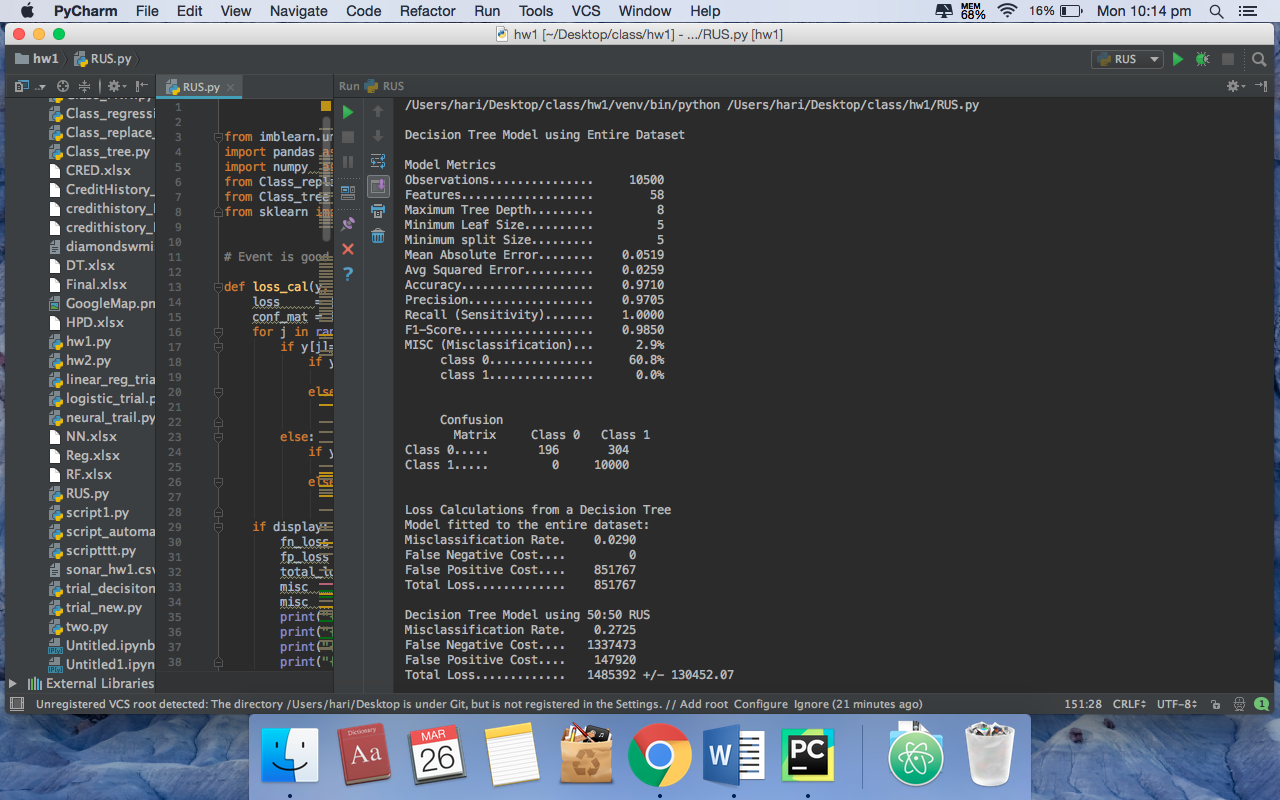
WEEK 8 – RARE EVENT MODELLING

PYTHON SCRIPT

from imblearn.under\_sampling import RandomUnderSampler  
import pandas as pd  
import numpy as np  
from Class\_replace\_impute\_encode import ReplaceImputeEncode  
from Class\_tree import DecisionTree  
from sklearn import tree  
  
  
# Event is good credit = 1 o.e false positive will have greatest loss.  
  
def loss\_cal(y, y\_predict, fp\_cost, fn\_cost, display=True):  
 loss = [0, 0] #False Neg Cost, False Pos Cost  
 conf\_mat = [0, 0, 0, 0] #tn, fp, fn, tp  
 for j in range(len(y)):  
 if y[j]==0:  
 if y\_predict[j]==0:  
 conf\_mat[0] += 1 #True Negative  
 else:  
 conf\_mat[1] += 1 #False Positive  
 loss[1] += fp\_cost[j]  
 else:  
 if y\_predict[j]==1:  
 conf\_mat[3] += 1 #True Positive  
 else:  
 conf\_mat[2] += 1 #False Negative  
 loss[0] += fn\_cost[j]  
 if display:  
 fn\_loss = loss[0]  
 fp\_loss = loss[1]  
 total\_loss = fn\_loss + fp\_loss  
 misc = conf\_mat[1] + conf\_mat[2]  
 misc = misc/len(y)  
 print("{:.<23s}{:10.4f}".format("Misclassification Rate", misc))  
 print("{:.<23s}{:10.0f}".format("False Negative Cost", fn\_loss))  
 print("{:.<23s}{:10.0f}".format("False Positive Cost", fp\_loss))  
 print("{:.<23s}{:10.0f}".format("Total Loss", total\_loss))  
 return loss, conf\_mat  
  
  
attribute\_map = {  
 'age':[0,(1, 120),[0,0]],  
 'amount':[0,(0, 20000),[0,0]],  
 'duration':[0,(1,100),[0,0]],  
 'checking':[2,(1, 2, 3, 4),[0,0]],  
 'coapp':[2,(1,2,3),[0,0]],  
 'depends':[1,(1,2),[0,0]],  
 'employed':[2,(1,2,3,4,5),[0,0]],  
 'existcr':[2,(1,2,3,4),[0,0]],  
 'foreign':[1,(1,2),[0,0]],  
 'good\_bad':[1,('bad', 'good'),[0,0]],  
 'history':[2,(0,1,2,3,4),[0,0]],  
 'housing':[2,(1, 2, 3), [0,0]],  
 'installp':[2,(1,2,3,4),[0,0]],  
 'job':[2,(1,2,3,4),[0,0]],  
 'marital':[2,(1,2,3,4),[0,0]],  
 'other':[2,(1,2,3),[0,0]],  
 'property':[2,(1,2,3,4),[0,0]],  
 'resident':[2,(1,2,3,4),[0,0]],  
 'savings':[2,(1,2,3,4,5),[0,0]],  
 'telephon':[1,(1,2),[0,0]] }  
  
df = pd.read\_excel("CRED.xlsx")  
# Encode for Logistic Regression, drop last one-hot column  
rie = ReplaceImputeEncode(data\_map=attribute\_map, nominal\_encoding='one-hot', \  
 interval\_scale = 'std', drop=False, display=False)  
encoded\_df = rie.fit\_transform(df)  
# Create X and y, numpy arrays  
# bad=0 and good=1  
y = np.asarray(encoded\_df['good\_bad']) # The target is not scaled or imputed  
X = np.asarray(encoded\_df.drop('good\_bad',axis=1))  
  
# Setup false positive and false negative costs for each transaction  
fp\_cost = np.array(df['amount'])  
fn\_cost = np.array(0.15\*df['amount'])  
  
treeclassifier = tree.DecisionTreeClassifier(criterion='gini', max\_depth=8, min\_samples\_split=5, min\_samples\_leaf=5)  
treeclassifier = treeclassifier.fit(X, y)  
  
print("\nDecision Tree Model using Entire Dataset")  
col = rie.col  
col.remove('good\_bad')  
DecisionTree.display\_binary\_metrics(treeclassifier, X, y)  
print("\nLoss Calculations from a Decision Tree")  
print("Model fitted to the entire dataset:")  
loss, conf\_mat = loss\_cal(y, treeclassifier.predict(X), fp\_cost, fn\_cost)  
  
# Setup random number seeds  
rand\_val = np.array([1, 12, 123, 1234, 12345, 654321, 54321, 4321, 321, 21])  
# Ratios of Majority:Minority Events  
ratio = [ '50:50', '60:40', '70:30', '75:25', '80:20', '85:15' ]  
# Dictionaries contains number of minority and majority events in each ratio sample  
# n\_majority = ratio x n\_minority  
rus\_ratio = ({0:500, 1:500}, {0:500, 1:750}, {0:500, 1:1166}, {0:500, 1:1500}, {0:500, 1:2000}, {0:500, 1:2833})  
  
# Best model is one that minimizes the loss  
min\_loss = 9e+15  
best\_ratio = 0  
for k in range(len(rus\_ratio)):  
 rand\_vals = (k+1)\*rand\_val  
 print("\nDecision Tree Model using " + ratio[k] + " RUS")  
 fn\_loss = np.zeros(len(rand\_vals))  
 fp\_loss = np.zeros(len(rand\_vals))  
 misc = np.zeros(len(rand\_vals))  
 for i in range(len(rand\_vals)):  
 rus = RandomUnderSampler(ratio=rus\_ratio[k], \  
 random\_state=rand\_vals[i], return\_indices=False, \  
 replacement=False)  
 X\_rus, y\_rus = rus.fit\_sample(X, y)  
 dtree = tree.DecisionTreeClassifier(criterion='gini', max\_depth=8, \  
 min\_samples\_split=5, min\_samples\_leaf=5)   
 dtree.fit(X\_rus, y\_rus)  
 loss, conf\_mat = loss\_cal(y, dtree.predict(X), fp\_cost, fn\_cost,\  
 display=False)  
 fn\_loss[i] = loss[0]  
 fp\_loss[i] = loss[1]  
 misc[i] = conf\_mat[1] + conf\_mat[2]  
 misc = np.sum(misc)/(10500 \* len(rand\_vals))  
 fn\_avg\_loss = np.average(fn\_loss)  
 fp\_avg\_loss = np.average(fp\_loss)  
 total\_loss = fn\_loss + fp\_loss  
 avg\_loss = np.average(total\_loss)  
 std\_loss = np.std(total\_loss)  
 print("{:.<23s}{:10.4f}".format("Misclassification Rate", misc))  
 print("{:.<23s}{:10.0f}".format("False Negative Cost", fn\_avg\_loss))  
 print("{:.<23s}{:10.0f}".format("False Positive Cost", fp\_avg\_loss))  
 print("{:.<23s}{:10.0f}{:5s}{:<10.2f}".format("Total Loss", avg\_loss, \  
 " +/- ", std\_loss))  
 if avg\_loss < min\_loss:  
 min\_loss = avg\_loss  
 best\_ratio = k  
  
  
# Ensemble Modeling - Averaging Classification Probabilities  
avg\_prob = np.zeros((len(y),2))  
# Setup 100 random number seeds for use in creating random samples  
np.random.seed(12345)  
max\_seed = 150000  
rand\_value = np.random.randint(1, high=max\_seed, size=10)  
# Model 100 random samples, each with a 70:30 ratio  
for i in range(len(rand\_value)):  
 rus = RandomUnderSampler(ratio=rus\_ratio[best\_ratio], \  
 random\_state=rand\_value[i], return\_indices=False, \  
 replacement=False)  
 X\_rus, y\_rus = rus.fit\_sample(X, y)  
 dtree = tree.DecisionTreeClassifier(criterion='gini', max\_depth=8, \  
 min\_samples\_split=5, min\_samples\_leaf=5)  
 dtree.fit(X\_rus, y\_rus)  
 avg\_prob += dtree.predict\_proba(X)  
avg\_prob = avg\_prob/len(rand\_value)  
# Set y\_pred equal to the predicted classification  
y\_pred = avg\_prob[0:,0] < 0.5  
y\_pred.astype(np.int)  
# Calculate loss from using the ensemble predictions  
print("\nEnsemble Estimates based on averaging",len(rand\_value), "Models")  
loss, conf\_mat = loss\_cal(y, y\_pred,fp\_cost,fn\_cost)

SCREENSHOT

OUTPUT:

Decision Tree Model using Entire Dataset

Model Metrics

Observations............... 10500

Features................... 58

Maximum Tree Depth......... 8

Minimum Leaf Size.......... 5

Minimum split Size......... 5

Mean Absolute Error........ 0.0519

Avg Squared Error.......... 0.0259

Accuracy................... 0.9710

Precision.................. 0.9705

Recall (Sensitivity)....... 1.0000

F1-Score................... 0.9850

MISC (Misclassification)... 2.9%

class 0............... 60.8%

class 1............... 0.0%

Confusion

Matrix Class 0 Class 1

Class 0..... 196 304

Class 1..... 0 10000

Loss Calculations from a Decision Tree

Model fitted to the entire dataset:

Misclassification Rate. 0.0290

False Negative Cost.... 0

False Positive Cost.... 851767

Total Loss............. 851767

Decision Tree Model using 50:50 RUS

Misclassification Rate. 0.2725

False Negative Cost.... 1337473

False Positive Cost.... 147920

Total Loss............. 1485392 +/- 130452.07

Decision Tree Model using 60:40 RUS

Misclassification Rate. 0.1962

False Negative Cost.... 949554

False Positive Cost.... 258551

Total Loss............. 1208105 +/- 132413.99

Decision Tree Model using 70:30 RUS

Misclassification Rate. 0.1021

False Negative Cost.... 468444

False Positive Cost.... 423333

Total Loss............. 891777 +/- 69286.81

Decision Tree Model using 75:25 RUS

Misclassification Rate. 0.0742

False Negative Cost.... 333280

False Positive Cost.... 515937

Total Loss............. 849217 +/- 75304.62

Decision Tree Model using 80:20 RUS

Misclassification Rate. 0.0566

False Negative Cost.... 227334

False Positive Cost.... 583129

Total Loss............. 810463 +/- 90470.82

Decision Tree Model using 85:15 RUS

Misclassification Rate. 0.0396

False Negative Cost.... 120918

False Positive Cost.... 650800

Total Loss............. 771718 +/- 75140.48

Ensemble Estimates based on averaging 10 Models

Misclassification Rate. 0.0246

False Negative Cost.... 0

False Positive Cost.... 595382

Total Loss............. 595382

Process finished with exit code 0