DSC-530: Predictive Modeling Week 3 Assignment: Predictive Modeling Using Decision Trees Author: Jonathan Ibifubara Pollyn College of Science, Engineering and Technology, Grand Canyon University In [43]: #loading the required packages import pandas as pd import numpy as np import statsmodels.tools.tools as stattools from sklearn.tree import DecisionTreeClassifier, export graphviz from sklearn.ensemble import RandomForestClassifier from sklearn.metrics import accuracy score 1.Create a CART model using the training data set that predicts Approval using Debt to Income Ratio, FICO Score, and Request Amount. Visualize the decision tree. Describe the first few splits in the decision tree. #Loading training Data into a variable loan training = pd.read csv('C:/DataSets/Loans Training') In [4]: loan training Out[4]: Approval Debt-to-Income Ratio FICO Score Request Amount Interest 0 F 0.00 397 1000 450.0 0.00 225.0 1 403 500 2 F 0.00 408 1000 450.0 408 2000 900.0 3 0.00 4 F 0.00 2250.0 411 5000 150297 Т 0.38 709 19000 8550.0 150298 Τ 0.38 722 17000 7650.0 4950.0 150299 Т 0.38 747 11000 4500.0 150300 0.39 679 10000 Т 150301 0.39 769 7000 3150.0 150302 rows × 5 columns #Get the predictor variable predictor x = loan training[['Debt-to-Income Ratio','FICO Score','Request Amount','Interest']] In [29]: #Getting the target variable target y = loan training['Approval'] #Running the CART Algorithm cart training = DecisionTreeClassifier(random state=0, max depth=2) In [64]: cart training.fit(predictor x, target y) Out[64]: DecisionTreeClassifier(max\_depth=2, random state=0) loan\_training\_data = export\_graphviz(cart\_training, out\_file=None, feature names=loan training.columns[1:], class names=loan training["Approval"].unique(), filled=True, rounded=True, special characters=True) #Installing the graphviz and pydot conda install python-graphviz conda install pydot In [49]: import graphviz In [104.. graph = graphviz.Source(loan training data) graph FICO Score ≤ 659.5 gini = 0.5samples = 150302value = [75066, 75236] class = T True False FICO Score ≤ 650.5 Debt-to-Income Ratio ≤ 0.305 gini = 0.076gini = 0.417samples = 46002 samples = 104300value = [44194, 1808] value = [30872, 73428] class = F class = T gini = 0.015gini = 0.338gini = 0.364gini = 0.227samples = 39733samples = 6269samples = 91427samples = 12873value = [39423, 310] value = [4771, 1498] value = [19678, 71749] value = [11194, 1679] class = F class = F class = T class = F 2.Develop a CART model using the test data set that uses the same target and predictor variables. Visualize the decision tree. Investigate the splits in the decision tree. Does the tree built using the test data match the tree built using the training data? #Loading training Data into a variable loan test = pd.read csv('C:/DataSets/Loans Test') loan\_test Approval Debt-to-Income Ratio FICO Score Request Amount Interest F 0.00 2000 900.0 413 0.00 449 1000 450.0 2 F 0.00 454 6000 2700.0 0.00 1000 450.0 456 4 F 0.00 457 1000 450.0 49693 Τ 0.38 662 14000 6300.0 49694 0.38 664 16000 7200.0 49695 Τ 0.38 676 4000 1800.0 49696 0.38 680 6000 2700.0 49697 0.39 500 662 225.0 49698 rows × 5 columns #Get the predictor variable for the test data predictor\_test\_x = loan\_test[['Debt-to-Income Ratio','FICO Score','Request Amount','Interest']] #Getting the target variable for the test data target test y = loan test['Approval'] #Running the CART Algorithm cart test = DecisionTreeClassifier(random\_state=0, max\_depth=2) #Running the CART prediction cart test.fit(predictor test x, target test y) Out[77]: DecisionTreeClassifier(max depth=2, random state=0) #Obtaining the graph loan test data = export graphviz(cart test, out file=None, feature names=loan training.columns[1:], class names=loan training["Approval"].unique(), filled=True, rounded=True, special characters=True) graph test = graphviz.Source(loan test data) In [84]: graph test Out[84]: FICO Score  $\leq$  660.5 gini = 0.5samples = 49698value = [24934, 24764] class = F True False Debt-to-Income Ratio ≤ 0.305 FICO Score ≤ 651.5 gini = 0.09gini = 0.418samples = 15481samples = 34217value = [14754, 727] value = [10180, 24037] class = F class = T gini = 0.395gini = 0.339gini = 0.225gini = 0.021samples = 13309 samples = 2172samples = 29978samples = 4239value = [13170, 139] value = [1584, 588] value = [6488, 23490] value = [3692, 547] class = F class = T class = Fclass = F 3.Build a C5.0 model using the training data set that predicts Approval using Debt to Income Ratio, FICO Score, and Request Amount. Specify a minimum of 1,000 cases per terminal node. Visualize the decision tree. Describe the first few splits in the decision tree. #Obtain decision tree using entropy c50\_training = DecisionTreeClassifier(criterion='entropy', max\_leaf\_nodes=5).fit(predictor\_x, target\_y) In [74]: loan\_training\_c50 = export\_graphviz(c50\_training, out\_file=None, feature\_names=loan\_training.columns[1:], class\_names=loan\_training["Approval"].unique(), filled=True, rounded=True, special\_characters=True) c50 graph = graphviz.Source(loan training c50) c50\_graph FICO Score ≤ 656.5 entropy = 1.0samples = 150302value = [75066, 75236] class = T **False** True FICO Score ≤ 644.5 Debt-to-Income Ratio ≤ 0.315 entropy = 0.168entropy = 0.886samples = 43749samples = 106553value = [42659, 1090] value = [32407, 74146] class = F class = T FICO Score ≤ 669.5 entropy = 0.453entropy = 0.025entropy = 0.578entropy = 0.773samples = 36483samples = 7266 samples = 12077 samples = 94476value = [6265, 1001] value = [36394, 89] value = [10930, 1147] value = [21477, 72999] class = Fclass = F class = F class = T entropy = 0.991entropy = 0.725samples = 84387 samples = 10089value = [4489, 5600] value = [16988, 67399] class = T class = T 5.Create a C5.0 model using the test data set that utilizes the same target variable, predictor variables, and minimum cases criterion. Visualize the decision tree. Does the tree built using the test data match the tree built using the training data? #Obtain decision tree using entropy c50\_test = DecisionTreeClassifier(criterion='entropy', max\_leaf\_nodes=5).fit(predictor\_test\_x, target\_test\_y) loan\_test\_c50 = export\_graphviz(c50\_test, out\_file=None, feature names=loan training.columns[1:], class\_names=loan\_training["Approval"].unique(), filled=True, rounded=True, special\_characters=True) c50\_graph\_test = graphviz.Source(loan\_test\_c50) c50\_graph\_test FICO Score ≤ 655.5 entropy = 1.0samples = 49698value = [24934, 24764] class = F False True Debt-to-Income Ratio ≤ 0.335 entropy = 0.152entropy = 0.895samples = 14182 samples = 35516value = [13871, 311] value = [11063, 24453] class = F class = T Debt-to-Income Ratio ≤ 0.265 entropy = 0.243entropy = 0.801samples = 3379samples = 32137value = [3243, 136] value = [7820, 24317] class = F class = T FICO Score ≤ 666.5 entropy = 0.998entropy = 0.744samples = 3297samples = 28840value = [1732, 1565] value = [6088, 22752] class = F class = T entropy = 1.0entropy = 0.693samples = 2320samples = 26520value = [1150, 1170] value = [4938, 21582] class = T class = T 1. Use random forests on the training data set to obtain the predicted value of Approval using the same predictor variables as in the CART and C5.0 models. random\_for\_y = np.ravel(target\_y) #Create the Random Forest random\_for\_train\_y = RandomForestClassifier(n\_estimators=100, criterion='gini').fit(predictor\_x,random\_for\_y) random for train y.predict(predictor x) Out[95]: array(['F', 'F', 'F', ..., 'T', 'T'], dtype=object) 1. Use random forests on the test data set to obtain the predicted value of Approval in the test data set. Build a table comparing the predictions from the training and test data sets. How do they compare? random for test y = np.ravel(target test y) #Create the Random Forest random for test y = RandomForestClassifier(n estimators=100, criterion='gini').fit(predictor test x,random for random\_for\_test\_y.predict(predictor\_test\_x) Out[100... array(['F', 'F', 'F', ..., 'T', 'T'], dtype=object)