Owen Hope

UCID: Oeh3

[Oeh3@njit.edu](mailto:Oeh3@njit.edu)

**CS634 Data Mining**

**Midterm Project**

**Apriori Algorithm**

**Database 1:**

|  |
| --- |
| **pots,trashbins,cookies,soda,beans,steak,sheets,blanket,underwear,chips,cookies**  **shampoo,keyboard,toiletpaper,soap,steak,socks,beans,pans,blanket,trashbins**  **salsa,pants,socks,bananas,underwear,pizza,cookies,pillow**  **pillow,toiletpaper,underwear**  **pans,trashbins,blanket,apples,salsa,steak**  **keyboard**  **steak,beans,bananas,mouse,blanket**  **shirt,tv,keyboard,chips,salsa,pans,shampoo,knives,pants,oj**  **chips**  **apples,rice,shampoo**  **cookies,pizza,rice**  **trashbins,knives,oj,pants,pots,shirt,sheets,computer,steak,blanket**  **pots,trashbins,shampoo,sheets,underwear,steak,blanket,papertowels,rice**  **pants,pans,tv,underwear,steak,chips,keyboard,shirt,apples,soap,rice**  **soap,trashbins**  **soda,toiletpaper,blanket,steak,mouse,blanket**  **pillow,mouse,oj,keyboard**  **shampoo,rice,cookies,chips,underwear,papertowels,bananas,salsa,oj,apples,soap,pans**  **underwear,blanket,soap**  **computer,beans,blanket,soap,underwear,chips,pants** |

**Database 2:**

|  |
| --- |
| **steak,pants,sheets,computer,pizza,salsa,cookies,pans,mouse**  **soda,oj,mouse,sheets,cookies**  **trashbins,shirt,apples,oj,pots,chips,sheets,salsa,knives,rice,soap,mouse,steak,keyboard,tv,underwear,shampoo**  **oj,soda,keyboard,apples,salsa,rice,papertowels,underwear,pizza,computer,tv,pants,cookies,beans,socks,chips**  **soap,pots,cookies,knives,salsa,apples,trashbins,chips,oj,steak,tv,papertowels,rice,computer,blanket,underwear,socks,beans,mouse,pizza,pants**  **soda,shirt,rice,tv,pizza,pans,steak,chips,bananas,pots,beans,blanket,papertowels,soap,cookies,toiletpaper,socks**  **pants,beans,pizza,chips,oj,steak**  **bananas,shirt,sheets,pants,cookies,pillow,socks,rice,apples,pots,pans,computer,trashbins,shampoo,pizza,salsa,soap**  **bananas,blanket,trashbins,apples,toiletpaper,rice**  **pots,knives,trashbins,soda,toiletpaper,cookies,keyboard,pans,tv,sheets,oj,socks,shirt,papertowels,blanket,steak,apples**  **blanket,bananas,pots,shirt**  **beans,socks,sheets,keyboard,rice,steak,soda,mouse,tv,papertowels,chips,cookies,pots,apples,oj,bananas,pants,shirt,soap,underwear**  **knives,soap,bananas,apples,pillow,keyboard,socks**  **soda,oj,knives,pants,chips**  **pizza**  **soda,oj,tv,mouse,apples,cookies,underwear,papertowels,pillow,sheets,beans,shirt,toiletpaper,blanket,socks,trashbins,shampoo,salsa**  **soap,salsa,pants,pillow,shirt,toiletpaper,tv,socks,underwear,blanket,beans,steak,oj,chips,trashbins,papertowels,rice,keyboard,knives**  **computer,rice,beans,pillow,underwear,soda,pants,apples,sheets,cookies**  **pizza,apples,soda,shampoo,steak,mouse,salsa,socks,pants,keyboard,blanket,beans,knives,sheets,toiletpaper,oj**  **soda,steak,pots,pants,pizza,socks,trashbins,beans,pans,toiletpaper,blanket,bananas,knives,soap,underwear,rice** |

**Database 3:**

|  |
| --- |
| **bananas,shirt,pillow,toiletpaper**  **mouse,shampoo,rice,blanket,trashbins**  **chips,shirt,rice,beans,blanket,trashbins,papertowels,pants,keyboard,salsa,toiletpaper,knives,sheets,computer,shampoo,tv**  **soda,sheets,pants,trashbins,computer,blanket,pans,soap**  **papertowels,apples,soda,rice,beans,socks,shampoo,soap,pizza,sheets,pillow,chips,knives,oj,blanket,mouse,computer,pots,shirt,salsa,underwear,trashbins,keyboard,tv**  **sheets,socks,soda,oj,salsa**  **soda,salsa,pizza,rice,blanket,shampoo,sheets,socks,pans,beans,keyboard,apples,toiletpaper,pots,mouse,pants,computer,soap,trashbins,oj,chips,shirt,underwear**  **soda,underwear,beans,toiletpaper,salsa,pants**  **toiletpaper,keyboard**  **soda,oj,cookies,salsa,socks,bananas,pots,beans,computer,shampoo,mouse,soap,trashbins,toiletpaper,chips,papertowels,pizza,pillow,blanket,sheets,pants,underwear,knives**  **mouse,soap,pizza,knives,computer,pots,beans,tv,papertowels,socks,bananas,salsa,blanket,cookies,pillow,shirt,chips**  **chips,papertowels,soda,trashbins**  **pillow,pants,trashbins,beans,shirt,keyboard,apples,shampoo,bananas,pans,papertowels**  **pizza,cookies,shirt,soap,salsa,socks,sheets,trashbins,blanket,beans,oj,mouse,soda,papertowels,computer,apples,chips,pans,rice,shampoo**  **socks,toiletpaper,tv,shirt,mouse,pillow,rice,knives,pizza,papertowels,trashbins,keyboard,beans,blanket,sheets,oj,computer,pans,pots,bananas,cookies,apples,chips**  **toiletpaper,bananas,underwear,trashbins,rice,socks,sheets,pizza,shampoo,cookies**  **trashbins,oj,soap,underwear,chips**  **oj,pillow,computer,rice,underwear,chips,soda,pots,blanket,shampoo,papertowels,pants,pans,knives,keyboard,toiletpaper,shirt,socks,pizza,cookies**  **mouse,pizza,toiletpaper,knives,soap,computer,underwear,socks,sheets,tv,shirt,pillow,keyboard,papertowels,pants,apples,oj**  **soap,pizza,trashbins,chips,bananas,papertowels,soda,blanket,keyboard** |

**Database 4:**

|  |
| --- |
| **apples,pizza,underwear,sheets,keyboard,knives,pans,mouse,toiletpaper,beans,blanket,shirt,pants,cookies,oj,soap,papertowels**  **cookies,soda,beans,pillow,computer,chips**  **bananas,oj,salsa,pillow,keyboard,sheets,rice,underwear,chips,papertowels,shampoo,pizza,socks**  **pizza,toiletpaper,salsa,apples**  **pizza,knives,shirt,computer,pillow,tv,soap,salsa,pants,socks,papertowels,chips,pans,soda,trashbins,beans,bananas**  **computer,pants,pans,socks,knives,pizza,underwear,shampoo,tv,keyboard,oj,shirt,chips,rice,bananas,trashbins,salsa,pillow**  **computer,oj,bananas,socks,mouse,sheets,pans,tv,chips,pillow,pizza,cookies,pants,beans,toiletpaper,papertowels,salsa,knives,underwear,shirt,blanket,soda,pots,trashbins,keyboard,rice**  **computer,knives,pizza,rice,soap,shirt,cookies,apples,oj,pots,papertowels,underwear,shampoo,chips,soda,trashbins,bananas,keyboard,socks,toiletpaper**  **cookies,pans,socks,sheets,pants,trashbins,blanket**  **computer,pans,underwear,papertowels,mouse,salsa,trashbins,oj,sheets,socks,pizza,shampoo,toiletpaper,bananas,pots,keyboard,apples,chips,soda,shirt,beans,rice,tv,pillow,pants**  **pants,toiletpaper,blanket,bananas,sheets,trashbins,papertowels,soap,chips,mouse,underwear,apples,computer,tv,pizza,cookies,shampoo,socks,beans,knives,keyboard,pillow,pans,oj,salsa,shirt,pots**  **pizza,rice,socks,computer,oj,tv,blanket,apples,papertowels,mouse,soda,pans,soap**  **oj,trashbins,shampoo,knives,blanket,pants,underwear,tv,cookies,socks,mouse,papertowels,beans,toiletpaper,sheets,salsa,soap,pots,keyboard,bananas,computer**  **underwear,rice**  **soda,shirt**  **pots,socks,sheets,beans,underwear,pans**  **socks,sheets,soda,mouse,apples,shirt,papertowels,pillow,toiletpaper,keyboard,shampoo,trashbins,pans,salsa,pizza,computer,bananas,cookies,soap,blanket,chips,oj,rice,pots**  **pots,salsa,pans**  **soap,pillow,tv,chips,shirt,apples,pots,salsa,trashbins,shampoo,blanket,oj,pans,socks,knives,rice,papertowels,pizza,bananas,cookies,keyboard**  **trashbins,salsa,soap,pots,sheets,bananas,beans,soda,chips,pizza,toiletpaper,rice,knives,mouse,apples,cookies,pants,socks,blanket,keyboard,pillow,pans,oj,papertowels,tv,shampoo,underwear,computer** |

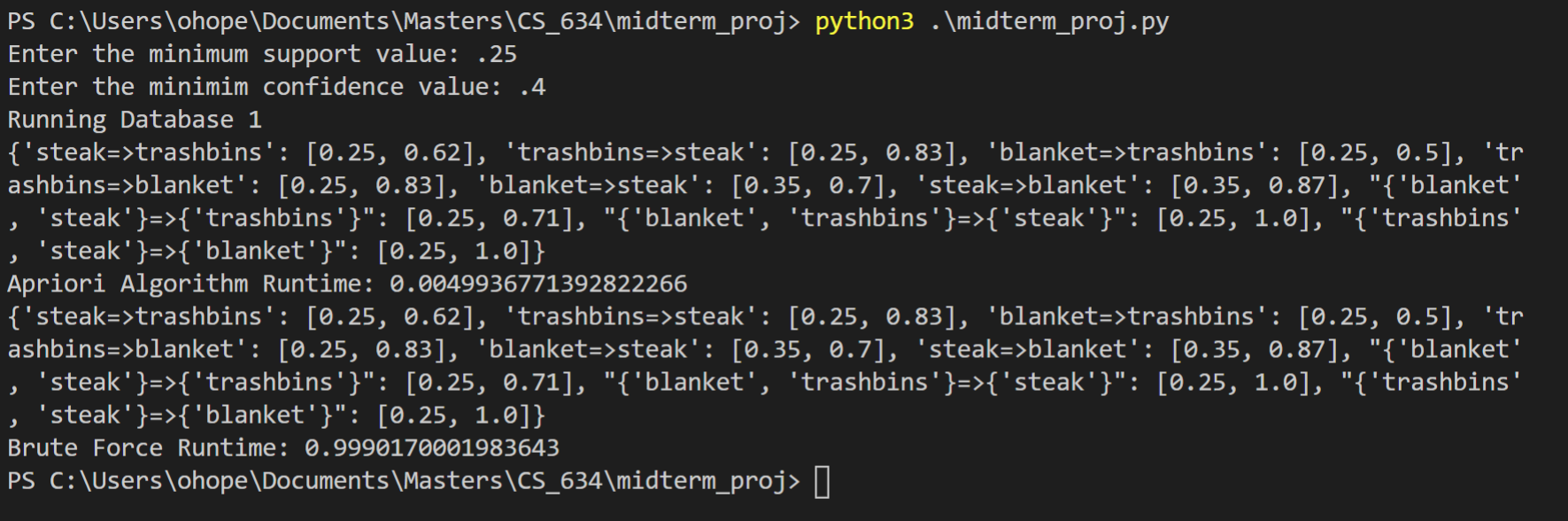
**Database 5:**

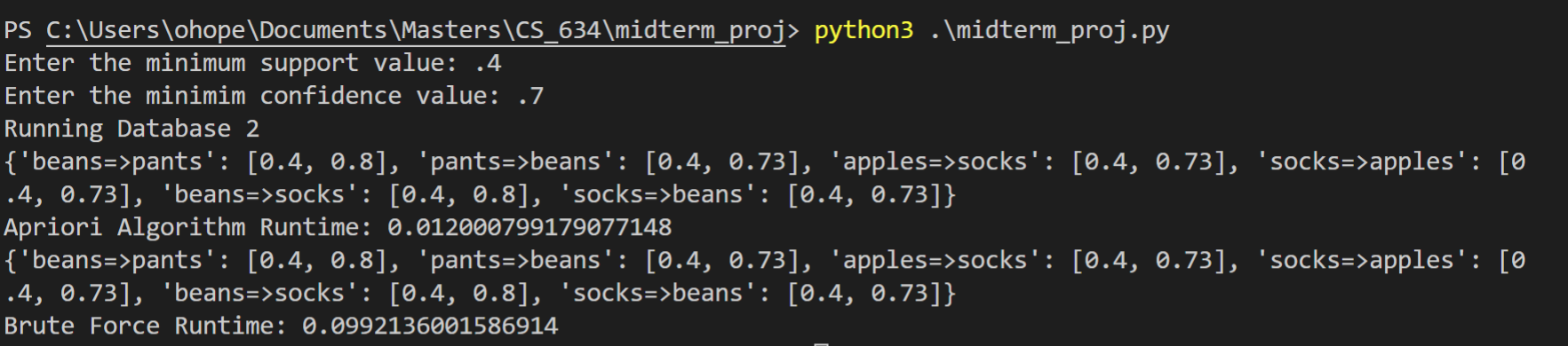
|  |
| --- |
| **soap,sheets,pots,keyboard,trashbins,pants,socks,computer,pizza,tv,shampoo,papertowels**  **shampoo,toiletpaper,computer,soda,pizza,pants,beans,mouse,pans,soap,trashbins,shirt,oj,pots**  **pillow,blanket,underwear,salsa,shampoo,oj,pizza,toiletpaper,bananas,cookies,computer,socks,beans,soda,shirt,pants,pans,pots,papertowels**  **pants,blanket,trashbins,underwear,mouse,apples,shirt,soda,rice,tv,computer,pans**  **salsa,toiletpaper,soda,knives,shirt,bananas,cookies,trashbins,pans,blanket,socks,soap,apples,underwear,mouse,pots,sheets,rice,beans**  **pillow,papertowels,chips,salsa,apples,oj,socks**  **tv,soda,chips,oj,pants,pots,shirt,bananas,salsa,trashbins,rice,computer,pizza,papertowels,underwear,mouse,knives,socks**  **papertowels,bananas,trashbins,chips,pants,rice,soap,apples,cookies,knives,tv,beans,salsa**  **pants,socks**  **salsa,shampoo,pants,shirt,sheets,bananas,cookies,soda,pillow,underwear,toiletpaper,keyboard,beans,apples**  **rice,tv,soap,computer,soda,pants**  **soda,computer,beans,socks,tv,apples,underwear**  **shampoo,papertowels,socks,pants**  **blanket,shirt,toiletpaper,trashbins,apples,pizza,oj,pots,knives,rice,chips,cookies,computer,sheets,pants,tv,soda,underwear,soap,bananas,papertowels,pans,socks,mouse**  **tv,socks,pants**  **soap,pizza,shirt,blanket,pots,papertowels,apples,knives,mouse,bananas,socks,tv,keyboard,pillow,salsa,underwear,computer,sheets,toiletpaper,rice,trashbins,beans,shampoo,chips,cookies,oj**  **cookies,apples,shampoo,socks,pants**  **keyboard,pants,shampoo,socks,pants**  **sheets,tv,trashbins,papertowels,soap,pots,chips,mouse,shirt,knives,soda,bananas**  **pillow,tv,oj,trashbins,pants,keyboard,pots,pans,beans,soap,computer,apples,mouse,knives,socks,soda,shirt,chips,sheets,blanket,salsa,papertowels,underwear,bananas,pizza,toiletpaper,cookies,shampoo** |

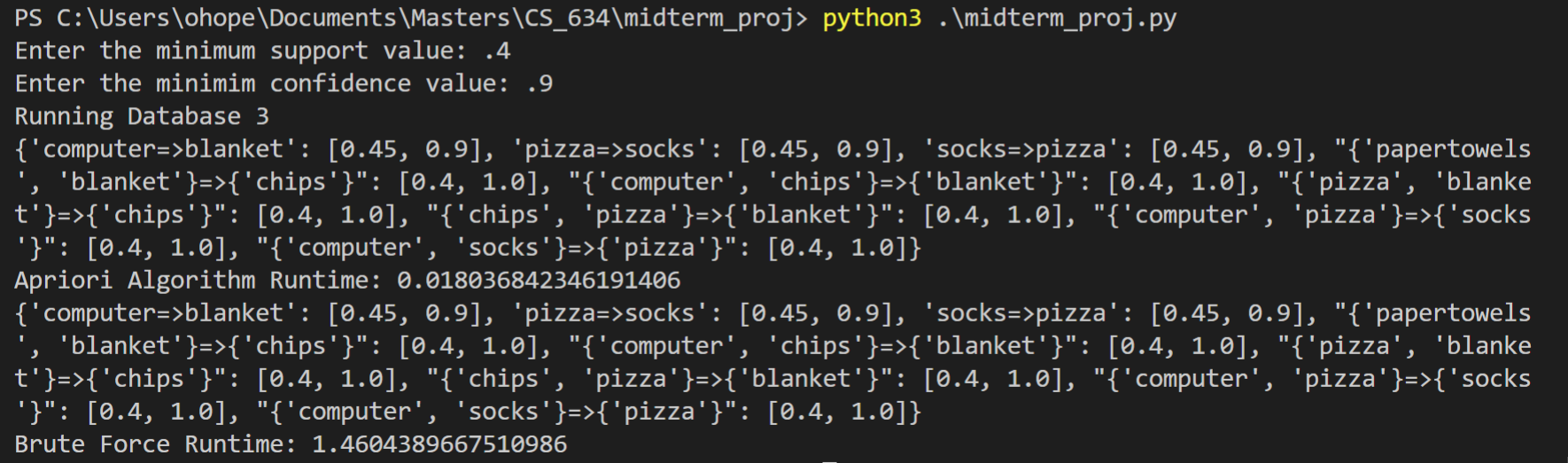
**Source Code:**

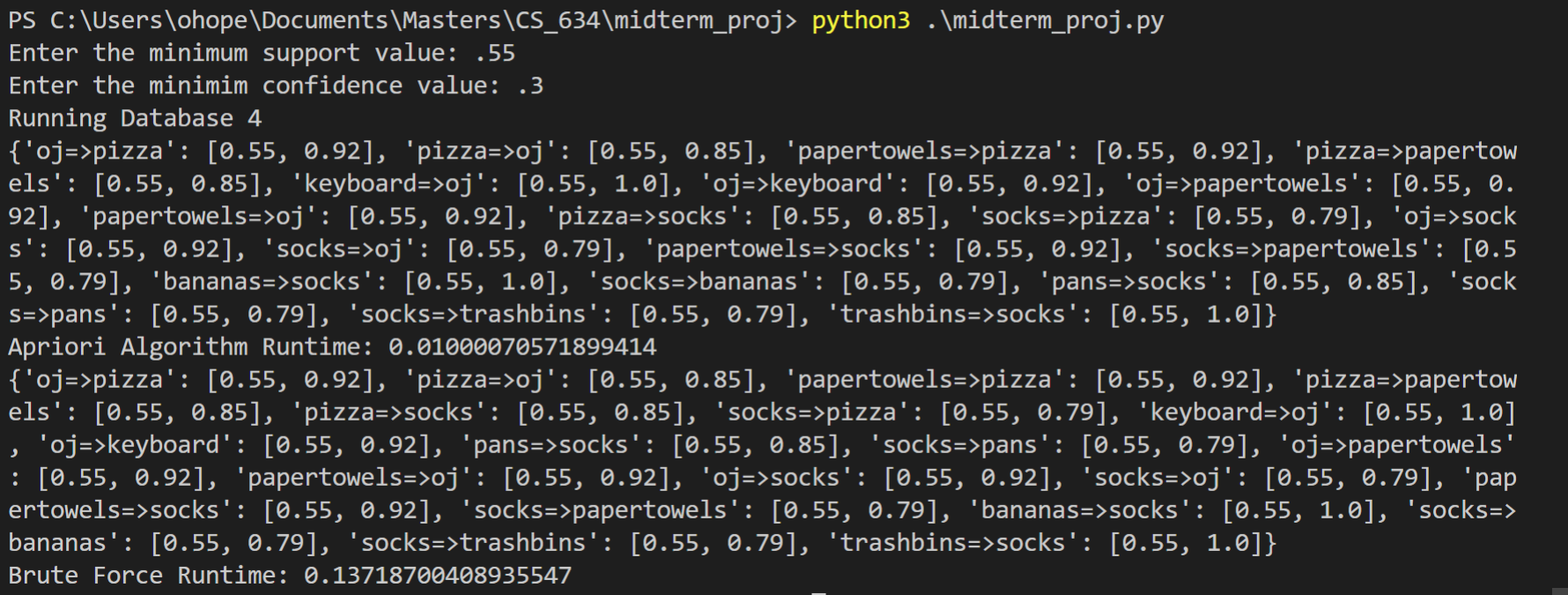
|  |
| --- |
| import csv  import time  min\_support = float(input("Enter the minimum support value: "))  min\_confidence = float(input("Enter the minimim confidence value: "))  # Replacement for itertools combinations  # Use of recursion  def join\_step(keys\_list, k):      # Used as recursion check      if k == 0:          return [[]]        return\_list = []      for key\_index in range(0, len(keys\_list)):          n = keys\_list[key\_index]          remList = keys\_list[key\_index + 1:]          for m in join\_step(remList, k - 1):              return\_list.append([n] + m)      return return\_list  def read\_file(file):      file\_list = []      with open(file, mode='r') as csv\_file:          csv\_reader = csv.reader(csv\_file, delimiter=',')          for row in csv\_reader:              file\_list.append(row)      return file\_list  def support\_calc(min\_support, freq\_dict, transacton\_num):      support\_dict = {}      for item, freq\_num in freq\_dict.items():          support\_val = freq\_num / transacton\_num          if support\_val >= min\_support:              support\_dict[item] = support\_val      return support\_dict  def confidence\_calc(min\_confidence, combination\_support\_dict):      association\_dict = {}      for item\_set, support\_val in combination\_support\_dict.items():          if isinstance(item\_set,str):              continue          list\_item = list(item\_set)            #print("Association rules {}".format(association\_rule))          item\_set\_size = len(item\_set)          if len(item\_set) == 2:              support\_left\_val = combination\_support\_dict[item\_set[0]]              conf = round(support\_val / support\_left\_val,2)              association = str(item\_set[0]) + "=>" + str(item\_set[1])              if conf >= min\_confidence:                  association\_dict[association] = [support\_val, conf]              support\_left\_val = combination\_support\_dict[item\_set[1]]              flipped\_rule\_conf = round(support\_val/support\_left\_val,2)              flipped\_association = str(item\_set[1]) + "=>" + str(item\_set[0])              if flipped\_rule\_conf >= min\_confidence:                  association\_dict[flipped\_association] = [support\_val, flipped\_rule\_conf]          else:              while item\_set\_size-1 > 1:                  association\_rule = join\_step(list\_item, item\_set\_size-1)                  for x in association\_rule:                      left\_side = set(x)                      right\_side = set(item\_set) - set(x)                        conf = round(support\_val / combination\_support\_dict[tuple(sorted(x))],2)                      if conf >= min\_confidence:                          association = str(left\_side) + "=>" + str(right\_side)                          association\_dict[association] = [support\_val, conf]                  item\_set\_size -= 1      return association\_dict    def apriori\_alg(min\_support, min\_confidence, file):      freq\_dict = {}      transacton\_num = 0      # Get data set initially      file\_values = read\_file(file)      for row in file\_values:          transacton\_num += 1          for item in row:              if item in freq\_dict.keys():                  val = freq\_dict[item]                  freq\_dict[item] = val + 1              else:                  freq\_dict[item] = 1      # We need to get support value after we have added up the number of same items      sup\_dic\_l1 = support\_calc(min\_support, freq\_dict, transacton\_num)      item\_sets = []      item\_sets.append(list(sup\_dic\_l1.keys()))      comb\_itemset\_dic = {}      comb\_itemset\_dic.update(sup\_dic\_l1)      # Since range is n-1 we need to do 21 to run for 20 transactions      keys = list(sup\_dic\_l1.keys())        for i in range(2,21):          # Need to do the join step          if keys == []:              break          keys\_item\_set = join\_step(keys, i)          #print("Keys item set {}".format(keys\_item\_set))          item\_sets.append(keys)          if keys\_item\_set == []:              break          freq\_dict\_c2 = {}          # Scan dataset each time when we try to calculate the support.          # This is so we don't keep this information in memory.          file\_values = read\_file(file)          for row in file\_values:              for item\_set in keys\_item\_set:                  item\_set = tuple(sorted(item\_set))                  check = all(item in row for item in item\_set)                  if check is True:                      if tuple(item\_set) in freq\_dict\_c2.keys():                          val = freq\_dict\_c2[tuple(item\_set)]                          freq\_dict\_c2[tuple(item\_set)] = val + 1                      else:                          freq\_dict\_c2[tuple(item\_set)] = 1          sup\_dic\_l = support\_calc(min\_support, freq\_dict\_c2, transacton\_num)          if not sup\_dic\_l:              keys = []              continue          comb\_itemset\_dic.update(sup\_dic\_l)          list\_item\_set = list(sup\_dic\_l.keys())          keys = []          for item\_set in list\_item\_set:              for item in item\_set:                  if item not in keys:                      keys.append(item)        #print(comb\_itemset\_dic)      conf\_dict = confidence\_calc(min\_confidence, comb\_itemset\_dic)      print(conf\_dict)    def brute\_force(min\_support, min\_confidence, file):      k\_items\_frequent = True      k = 2      support\_dict = {}        single\_item\_transactions = {}      transaction\_num = 0      file\_values = read\_file(file)      for row in file\_values:          transaction\_num += 1          for item in row:              if item in single\_item\_transactions.keys():                  val = single\_item\_transactions[item]                  single\_item\_transactions[item] = val + 1              else:                  single\_item\_transactions[item] = 1      single\_item\_support = support\_calc(min\_support, single\_item\_transactions, transaction\_num)      single\_item\_keys = list(single\_item\_transactions.keys())      support\_dict.update(single\_item\_support)      while k\_items\_frequent:          list\_item\_set = join\_step(single\_item\_keys, k)            item\_freq = {}          # Scan dataset each time when we try to calculate the support.          # This is so we don't keep this information in memory.          # As said in the frequently asked questions section          file\_values = read\_file(file)          for item\_set in list\_item\_set:              item\_set = tuple(sorted(item\_set))              for row in file\_values:                  key\_list = item\_freq.keys()                  if set(item\_set).issubset(row):                      if item\_set in key\_list:                          val = item\_freq[item\_set]                          item\_freq[item\_set] = val + 1                      else:                          item\_freq[item\_set] = 1          #print(double\_item\_freq)          item\_support = support\_calc(min\_support, item\_freq, transaction\_num)            support\_dict.update(item\_support)          if not item\_support:              break          k += 1      conf\_dict = confidence\_calc(min\_confidence, support\_dict)      print(conf\_dict)  '''  # Database 1  print("Running Database 1")  db1\_start\_apriori\_time = time.time()  apriori\_alg(min\_support, min\_confidence, 'db1.txt')  db1\_time\_apriori\_alg = time.time() - db1\_start\_apriori\_time  print("Apriori Algorithm Runtime: {}".format(db1\_time\_apriori\_alg))  db1\_start\_brute\_time = time.time()  brute\_force(min\_support, min\_confidence, 'db1.txt')  db1\_time\_brute\_alg = time.time() - db1\_start\_brute\_time  print("Brute Force Runtime: {}".format(db1\_time\_brute\_alg))  # Database 2  print("Running Database 2")  db2\_start\_apriori\_time = time.time()  apriori\_alg(min\_support, min\_confidence, 'db2.txt')  db2\_time\_apriori\_alg = time.time() - db2\_start\_apriori\_time  print("Apriori Algorithm Runtime: {}".format(db2\_time\_apriori\_alg))  db2\_start\_brute\_time = time.time()  brute\_force(min\_support, min\_confidence, 'db2.txt')  db2\_time\_brute\_alg = time.time() - db2\_start\_brute\_time  print("Brute Force Runtime: {}".format(db2\_time\_brute\_alg))  # Database 3  print("Running Database 3")  db3\_start\_apriori\_time = time.time()  apriori\_alg(min\_support, min\_confidence, 'db3.txt')  db3\_time\_apriori\_alg = time.time() - db3\_start\_apriori\_time  print("Apriori Algorithm Runtime: {}".format(db3\_time\_apriori\_alg))  db3\_start\_brute\_time = time.time()  brute\_force(min\_support, min\_confidence, 'db3.txt')  db3\_time\_brute\_alg = time.time() - db3\_start\_brute\_time  print("Brute Force Runtime: {}".format(db3\_time\_brute\_alg))  # Database 4  print("Running Database 4")  db4\_start\_apriori\_time = time.time()  apriori\_alg(min\_support, min\_confidence, 'db4.txt')  db4\_time\_apriori\_alg = time.time() - db4\_start\_apriori\_time  print("Apriori Algorithm Runtime: {}".format(db4\_time\_apriori\_alg))  db4\_start\_brute\_time = time.time()  brute\_force(min\_support, min\_confidence, 'db4.txt')  db4\_time\_brute\_alg = time.time() - db4\_start\_brute\_time  print("Brute Force Runtime: {}".format(db4\_time\_brute\_alg))  '''  # Database 5  print("Running Database 5")  db5\_start\_apriori\_time = time.time()  apriori\_alg(min\_support, min\_confidence, 'db5.txt')  db5\_time\_apriori\_alg = time.time() - db5\_start\_apriori\_time  print("Apriori Algorithm Runtime: {}".format(db5\_time\_apriori\_alg))  db5\_start\_brute\_time = time.time()  brute\_force(min\_support, min\_confidence, 'db5.txt')  db5\_time\_brute\_alg = time.time() - db5\_start\_brute\_time  print("Brute Force Runtime: {}".format(db5\_time\_brute\_alg)) |

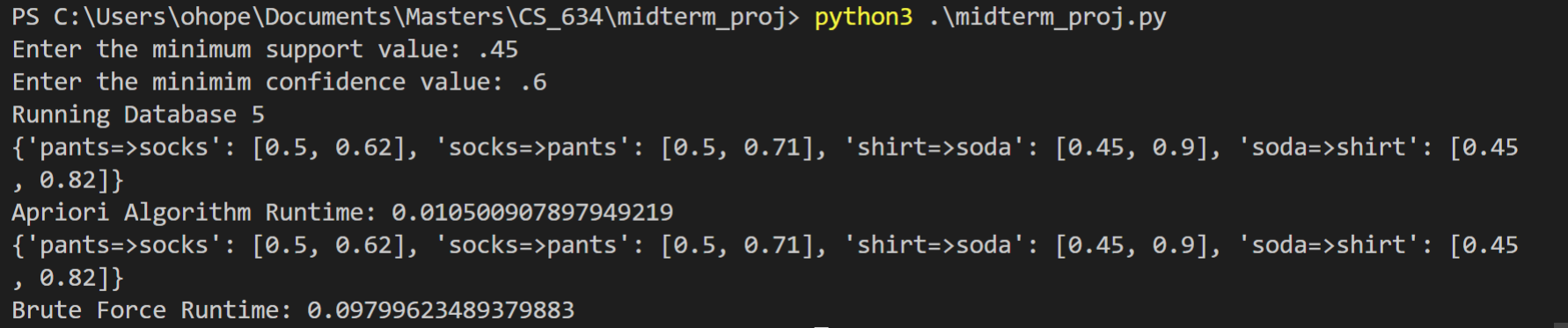
Each database is run by commenting out the specific code pertaining to it then running the midter\_proj.py file.

**Database 1 Execution Screenshot:**

**Database 2 Execution Screenshot:**

**Database 3 Execution Screenshot:**

**Database 4 Execution Screenshot:**

**Database 5 Execution Screenshot:**