DecisionTree

2016059198 진승현

1. 프로그램 개요

Training Samples를 입력받아 Decision Tree를 생성하고 이를 기반으로 Test Samples의 Class를 Prediction하여 그 결과를 출력하는 프로그램이다.

2. Decision Tree

Decision Tree는 top-down recursive devide-and-conquer manner를 통하여 tree를 구성한 것으로 적절한 candidate를 정하고 이를 기반으로 한 partition으로 나누어 평가하는 방식을 사용하고 있다.

여기서 중요한 것은 어떻게 적절한 candidate를 구하는가 이며 Information Gain, Gain Ratio, Gini index 등의 방법이 알려져 있다.

본 프로그램에는 Information Gain 방식과 Gain Ratio를 구현하였으며 Gain Ratio 방식을 적용하였다.

3. 코드 설명(함수 기준)

1) Find Gain Ratio & Select Candidate

```
def getInfo(D): # Info(D) = -sum(1->m) (p(i)*log2(p(i))
  m, counts = np.unique(D.iloc[:, -1], return_counts = True)
  return -np.sum( np.fromiter( (counts[i]/len(D) * np.log2(counts[i] / len(D)) for i in range(len(m))), float ) )

def getGain(D, attr): #InfoA(D) = sum(1->v) |D(j)|/|D| * Info(D(j))
  v, counts = np.unique(D[attr], return_counts = True) # v = attribute의 class#
  return getInfo(D) - np.sum( np.fromiter( (counts[j] / len(D) * getInfo(D.query(f"{attr} == '{v[j]}'")) for j in range(len(v))), float ) )

def getGainRatio(D, attr): #GainRatio = Gain(A) / SplitInfo(A)
                           #SpitInfoA(D) = -sum(1->v) |D(j)|/|D| * log2(|D(j)|/|D|)
  v, counts = np.unique(D[attr], return_counts = True)
  splitInfo = - np.sum( np.fromiter( (counts[j] / len(D) * np.log2(counts[j] / len(D)) for j in range(len(v)) ), float  ) )
  return getGain(D, attr) / splitInfo

def getCandidate(D): #get max-gain attribute name
  gains = np.array([getGainRatio(D, attr) for attr in D.columns[:-1]])
  return D.columns[gains.argmax()]
```

2) get Decision Tree

```
Tree_Node = {
    'Candidate' : '',
    'Attribute' : '',
    'Childs' : {},
}

def getDecisionTree(D, Node): #D는 samples / attr은 나눌 attribute

  New_Node = {
    'Candidate' : '',
    'Attribute' : '',
    'Childs' : {},
  }
  if D.size == 0: #더이상 남은 샘플이 없을 때
    return None

  elif D.keys().size == 1: #더 이상 나눌 수 있는 attribute가 없을 때
    classes, counts = np.unique(D, return_counts=True)
    return classes[counts.argmax()]

  elif len(np.unique(D.values[:,-1:])) == 1: #남은 샘플의 클래스가 동일할 때
    return np.unique(D.values[:,-1:])[0]

  else:

    Node['Candidate'] = getCandidate(D)
    New_Node['Attribute'] = Node['Candidate']
    classes = np.unique(D[Node['Candidate']]) #attribute의 classes
    for cls in classes:
      Child_Node = New_Node.copy() #child노드 생성
      if len(Child_Node['Childs']) != 0:
        Child_Node['Childs'] = {}
      New_D = D.query(f"{Node['Candidate']} == '{cls}'").drop(Node['Candidate'], axis=1) #data split
      (Node['Childs'])[cls] = (getDecisionTree(New_D, Child_Node),len(New_D)) #node와 node의 길이 set

    #Node['Childs'] = childs #부모노드에 자식노드들 추가

  return Node
```

3) prediction

```
def getPredict(DTree, sample):
  if type(DTree) is tuple and type(DTree[0]) is not dict: #leaf node
    return DTree[0]
  else:
    if type(DTree) is tuple:
      DTree = DTree[0]
    if sample[DTree['Candidate']] in DTree['Childs'].keys(): #해당 attribute 값이 tree에 있다면
      return getPredict(DTree['Childs'][sample[DTree['Candidate']]], sample)
    else: #없다면
      max_key = max(DTree['Childs'], key = (lambda k: DTree['Childs'][k][1]))
      return getPredict(DTree['Childs'][max_key], sample)
```

4. 실행 예제

1)Argument Format

Python dt.py [traning data src] [test data src] [result file src]

```
2021_ite4005_2016059198₩DecisionTree>python dt.py ./data/dt_train1.txt ./data/dt_test1.txt ./test/dt_result1.txt
```

2) test 결과

```
st>dt_test.exe dt_answer1.txt dt_result1.txt
313 / 346
```

5. Specification of Testing

OS : Windows 10 Home (x64)

Language version : Python 3.7.0