EECS349 Machine Learning

Homework 4

Student name: Haikun Liu
NetID: hlg483

Question 1

Consider a data set with 30 attributes, where x_1(randomly pick between 1 and 10) ,x_2(randomly pick between 0 and 1), and x_3(randomly pick between 10 and 12) are numeric, while the rest attributes, x_4~x_{30}, are nominal with value taken 0 or 1. The target function returns value according to following rules:

1). Return 1 if $x_1 < 5$ and $x_7 == 0$
2). Return 1 if $x_2 == x_9 == 0$ and $x_{10} == 1$
3). Return 1 if $x_6 == 1$ and $x_8 == 1$
4). Return 1 if $x_{12} == x_{21} == 0$ and $x_{17} == 1$
5). Otherwise, return 0

	Nearest Neighbor	Decision Trees
10-fold Cross-validation	Correctly Classified Instances 650 65% Incorrectly Classified Instances 350 35% Kappa statistic 0.2985 Mean absolute error 0.3503 Root mean squared error 0.591 Relative absolute error 70.2016 % Root relative squared error 118.3042 % Total Number of Instances 1000	Correctly Classified Instances 993 99.3% Incorrectly Classified Instances 7 0.7% Kappa statistic 0.986 Mean absolute error 0.0084 Root mean squared error 0.0819 Relative absolute error 1.6854 % Root relative squared error 16.3928 % Total Number of Instances 1000
Absolute difference in accuracy = 34.3%		

Question 2:

Consider a data set with two nominal attributes x_1 and x_2 which is 0 or 1, where the target function is "return 1 if $x_1 == x_2$, and 0 other wise" and one nominal output y. Naïve Bayes must attempt to approximate conditional independence between x_1 x_2 and y, which will be erroneous, because probabilities that $x_1 == 1$ or $x_2 == 1$ are the same for $y = 1$ class or $y = 0$ class. On the other hand, Multilayer Perceptron can perfectly represent this $y = x_1 \, XOR \, x_2$ relation.

	Bayes Net	Multilayer Perceptron
10-fold Cross-validation	Correctly Classified Instances 529 52.9% Incorrectly Classified Instances 471 47.1% Kappa statistic 0 Mean absolute error 0.4986 Root mean squared error 0.4996 Relative absolute error 100.0603 % Root relative squared error 100.0979 % Total Number of Instances 1000	Correctly Classified Instances 1000 100% Incorrectly Classified Instances 0 0% Kappa statistic 1 Mean absolute error 0.0053 Root mean squared error 0.0053 Relative absolute error 1.0573 % Root relative squared error 1.0608 % Total Number of Instances 1000
Absolute difference in accuracy = 47.1%		

Question 3

a. Genetic algorithm

Example: Solving N-queen problem

Reason: Genetic algorithm is typically used to provide good global maximum approximation to problems that cannot be solved easily using other techniques. In N-queen problem, gradient descent and hill climbing are easily stuck in local maximum, so that the solution is not completed. Although, genetic algorithm does not guarantee the completed result, with proper genetic representation and fitness function, genetic algorithm can reach a satisfied solution to N-queen problem, which may not be achievable using gradient descent and hill climbing.

b. Gradient descent

Example: Find the value for w and b to minimize $f(x)$ where $f(x) = \sum_j (y_j - wx_j + b)^2$ and $x_j, y_j \in R$

Reason: Gradient descent is a first-order optimization, which takes steps proportional to the negative gradient of $f(x)$ at the current point. In this minimization problem, it is unable to find a suitable genetic representation of w and b in order to use genetic algorithm. While, hill climbing is easily stuck into local maximum in this case. Here, $f(x)$ is differentiable with regard to w and b, and $f(x)$ is a convex-shape function. So it is ideally to used gradient descent to solve this optimization problem.

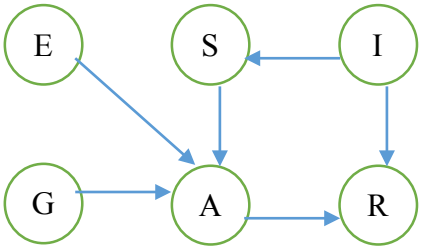
c. Hill climbing

Example: Find a maximum point of $f(x)$ with space complicity equal to $O(1)$

Reason: Hill climbing is a memory-less iterative algorithm that starts with an arbitrary solution to a problem, then continually move in the direction of increasing value of all successor states until a maximum is reached. In this maximization problem, it is unable to store previous states in order to use genetic algorithm or gradient descent. It does not require that the result need to be completed and optimal. So hill climbing can finish the task with minimal computational complexity.

Question 4

a.



b.

Minimum number of probabilities $= (\mathrm{A}: 8) + (G: 1) + (E: 1) + (S: 2) + (R: 4) + (I: 1) = 8 + 1 + 1 + 2 + 4 + 1 = 17$

c.

Number of independent parameters $= 2^6 - 1 = 63$

d.

E and G are converging connected, and A renders them dependent. When there is no value for any other variables, E and G are independent

e.

E and G are converging connected through A. If A has a value as evidence, E conditionally dependent of G given A

Question 5

$P(C) = \sum_a \sum_b P(a, b, C) = \sum_a \sum_b P(C|a, b)P(a)P(b) = P(C|A, B)P(A)P(B) + P(C|A, \sim B)P(A)P(\sim B) + P(C|\sim A, B)P(\sim A)P(B) + P(C|\sim A, \sim B)P(\sim A)P(\sim B) = 1\times0.5\times0.7 + 1\times0.7\times0.5 + 0.7\times0.3\times0.5 + 0.3\times0.3\times0.5 = 0.85$

Appendix:

Python code for question 1:

```
import csv, random
c = csv.writer(open('prob4.csv', 'wb'))
header = []
for i in range(30):
    header.append('X' + str(i+1))
header.append('Y')
c.writerow(header)
for i in range(1000):
    row = []
    row.append(random.uniform(1,10))
    row.append(random.uniform(0,1))
    row.append(random.uniform(10,12))
    for j in range(27):
```

```
            pred_val = random.randint(0,1)
            row.append(pred_val)
      if (row[0] < 5 and row[6] == 0) or (row[9] == 1 and row[8] == 0 and row[1]==0) or (row[7]==1 and row[5]==1) or (row [11]==0 and row[20]==0 and row[16]==1):
            row.append(1)
      else:
            row.append(0)
      c.writerow(row)
```

Python code for question 2:

```
import csv, random
c = csv.writer(open('prob_ec.csv', 'wb'))
header = []
for i in range(2):
      header.append('X' + str(i+1))
header.append('Y')
c.writerow(header)
for i in range(1000):
      row = []
      row.append(random.randint(0,1))
      row.append(random.randint(0,1))
      if row[0] == row[1]:
            row.append(0)
      else:
            row.append(1)
      c.writerow(row)
```