Machine Learning

Important points and calculations

- Kathirmani Sukumar

Disclaimer

- Please use LMS notes for introduction and theory about each alogorithm
- This notes is mainly to cover important points discussed in class

Naïve Bayes

Outlook	Temperature	Humidity	Wind	Play Tennis
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Overcast	Hot	High	Weak	Yes
Rain	Mild	High	Weak	Yes
Rain	Cool	Normal	Weak	Yes
Rain	Cool	Normal	Strong	No
Overcast	Cool	Normal	Strong	Yes
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Rain	Mild	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes
Rain	Mild	High	Strong	No

Frequency Tables

Outlook	Play=*Yes*	Play=*No*
Sunny	2/9	3/5
Overcast	4/9	0/5
Rain	3/9	2/5

Temperature	Play=*Yes*	Play=*No*
Hot	2/9	2/5
Mild	4/9	2/5
Cool	3/9	1/5

Humidity	Play=*Yes*	Play=*No*
High	3/9	4/5
Normal	6/9	1/5

Wind	Play=*Yes*	Play=*No*
Strong	3/9	3/5
Weak	6/9	2/5

P(Play=Yes) = 9/14

P(Play=No) = 5/14

Predictions

Given a new instance, predict its label

X = (Outlook=Sunny, Temperature=Cool, Humidity=High, Wind=Strong)

P(Outlook=Sunny | Play=Yes) = (2 + 1) / (9 + 3) = 0.25

P(Temperature=Cool | Play=Yes) = (3 + 1) / (9 + 3) = 0.33

P(Humidity=High | Play=Yes) = (3 + 1) / (9 + 2) = 0.36

P(Wind=Strong | Play=Yes) = (3 + 1) / (9 + 2) = 0.36

P(Play=Yes) = 9 / 14 = 0.64

P(Outlook=Sunny | Play=No) = (3 + 1) / (5 + 3) = 0.5

P(Temperature=Cool | Play=No) = (1 + 1) / (5 + 3) = 0.25

P(Humidity=High | Play=No) = (4 + 1) / (5 + 2) = 0.71

P(Wind=Strong | Play=No) = (3 + 1) / (5 + 2) = 0.57

P(Play=No) = 5 / 14 = 0.35

P(Yes | X) = [P(Sunny|Yes) P(Cool|Yes) P(High|Yes) P(Strong|Yes)] P(Play=Yes) = 0.0068

P(Yes | X) = [P(Sunny|No) P(Cool|No) P(High|No) P(Strong|No)] P(Play=No) = 0.0177

Given the fact P(Yes | X) < P(No | X). We label X to No

K Nearest Neighbor

X1	X2	Y
7	7	Bad
7	4	Bad
3	4	Good
1	4	Good

Predict Y if X1=3, X2=7; K= 3

K Nearest Neighbor

X1	X2	Euclidian Distance	Neighbor	Y
7	7	Sqrt((7-3)^2 + (7-7)^2) = 4	3	Bad
7	4	Sqrt((7-3)^2 + (4-7)^2) = 5	4	Bad
3	4	Sqrt((3-3)^2 + (4-7)^2) = 3	1	Good
1	4	Sqrt((1-3)^2 + (4-7)^2) = 3.6	2	Good

Among the first 3 Neighbors, 2 samples are Good and 1 sample is Bad; Prediction is Good

X1	X2	Euclidian Distance	Neighbor	Y
7	7	Sqrt((7-3)^2 + (7-7)^2) = 4	3	Bad
7	4	Sqrt((7-3)^2 + (4-7)^2) = 5	4	Bad
3	4	Sqrt((3-3)^2 + (4-7)^2) = 3	1	Good
1	4	Sqrt((1-3)^2 + (4-7)^2) = 3.6	2	Good

Random Forest

- Each tree is grown fully to reduce bias; Due to this variance will be high
- To reduce variance, multiple such trees are grown
- For each tree, 60% of samples are randomly selected
- While randomly selecting samples, each sample is given equal weight
- For each tree, Sqrt(No. of cols) are randomly selected
- Prediction is based on polling of predictions from all trees

Adaboost

- In Random Forest, each sample is given equal weight while sampling
- In Adaboost, while building first decision tree, each sample is given equal weight
- While building second decision tree, for each sample new weights are calculated
- Samples which are misclassified in first decision tree are given higher weights in second decision tree

Sample Data

Outlook	Temperature	Humidity	Wind	Target Original	Target New
Sunny	Hot	High	Weak	No	-1
Sunny	Hot	High	Strong	No	-1
Overcast	Hot	High	Weak	Yes	1
Rain	Mild	High	Weak	Yes	1
Rain	Cool	Normal	Weak	Yes	1
Rain	Cool	Normal	Strong	No	-1
Overcast	Cool	Normal	Strong	Yes	1
Sunny	Mild	High	Weak	No	-1
Sunny	Cool	Normal	Weak	Yes	1

N = 9

First Tree Prediction on Training Data

Sample Weight	Actual	Predicted	Correct Prediction?
1/N = 1 / 9	-1	1	Wrong
1 / 9	-1	-1	Correct
1 / 9	1	-1	Wrong
1 / 9	1	1	Correct
1 / 9	1	1	Correct
1 / 9	-1	1	Wrong
1 / 9	1	1	Correct
1 / 9	-1	-1	Correct
1 / 9	1	1	Correct

Error Rate

Error Rate = Wrong / Total Predictions

Error Rate = 3 / 9 = 0.33

Classifier's weight

$$\alpha_t = \frac{1}{2} ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$$

```
(1 / 2) * np.log((1-0.33) / 0.33)
```

```
0.3540925289622428
```

Samples new weight

$$D_{t+1}(i) = \frac{D_t(i)\exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

- Y_i = Actual value of i^{th} sample
- $H_t(x_i)$ = Prediction value of i^{th} sample
- $Alpha_t$ = Classifiers weight

Sample Weight (D_1)	Actual	Predicted	Correct Prediction?	Sample Weight (D_2) (numerator)
1/N = 1 / 9	-1	1	Wrong	(1/9) exp(-0.35*(-1)*(1)) = 0.15
1 / 9	-1	-1	Correct	(1/9) exp(-0.35*(-1)*(-1)) = 0.07
1 / 9	1	-1	Wrong	(1/9) exp(-0.35*(1)*(-1)) = 0.15
1 / 9	1	1	Correct	(1/9) exp(-0.35*(1)*(1)) = 0.07
1 / 9	1	1	Correct	…
1 / 9	-1	1	Wrong	…
1 / 9	1	1	Correct	…
1 / 9	-1	-1	Correct	…
1 / 9	1	1	Correct	…

- Samples which are misclassified given weight as 0.15
- Samples which are correctly classified given weight as 0.07
- Samples which are misclassified are given more weight
- Zt is nothing but summation of all new weights
- Mainly used to make values add up to 1
- Second decision tree gives more weights to misclassified samples

Weighted Polling

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

- While polling prediction from each tree is added together along with classifier weight
- It is called as weighted polling
- If sum comes out to be positive, final prediction is +1
- If sum comes out to be negative, final prediction is -1