ML LAB-1

EXPT #1: Creating attributes and test and train sets

1. 1000 random examples were chosen from TRAINlabeledBow.feat and 1000 from TESTlabeledBow.feat such that half of them are positive and half are negative.

2. For attributes, I chose 5000 most frequently used words with 2500 having polarity>1 and 2500 with polarity<-0.5.

3. For different runs, the examples in train and test sets change as they are chosen randomly and so, the accuracy also changes by a slight amount

EXPT #2: ID3 with and without early stopping

Without early stopping:

Accuracy(avg) : 69.5%

Num of leaves: 189

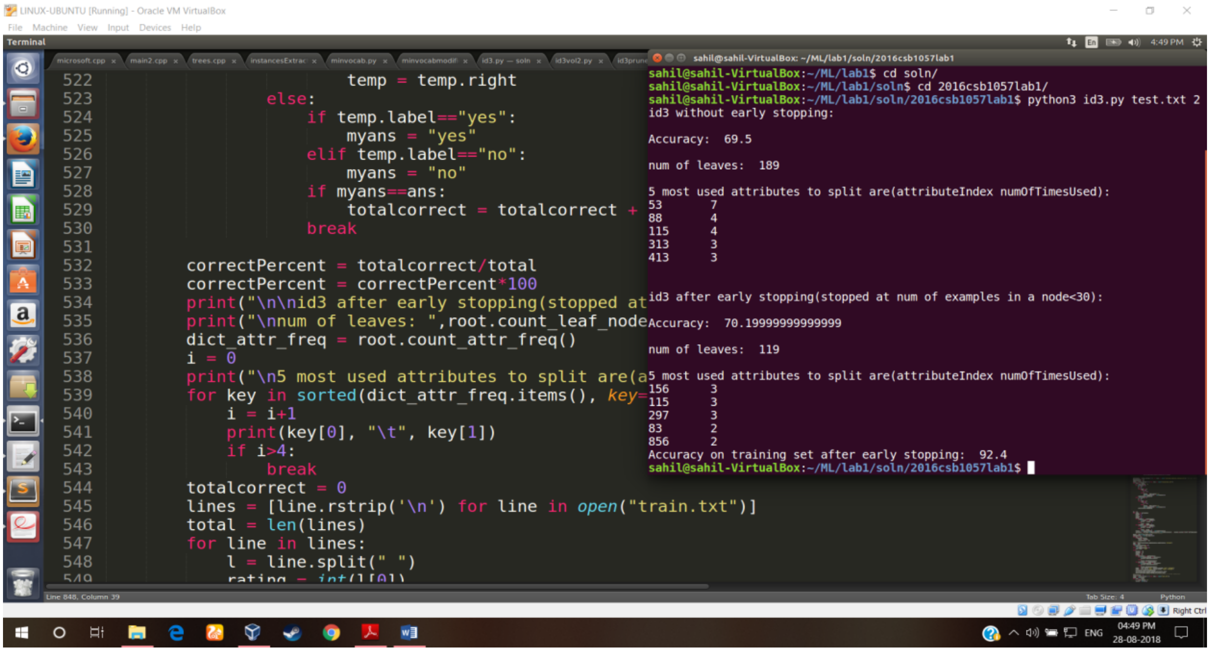
5 most used attributes(for splitting)	Frequency
53	7
88	4
115	4
313	3
413	3

After early stopping:

Accuracy: 70.19%

Num of leaves: 119

5 most used attributes(for splitting)	Frequency
156	3
115	3
297	3
83	2
856	2

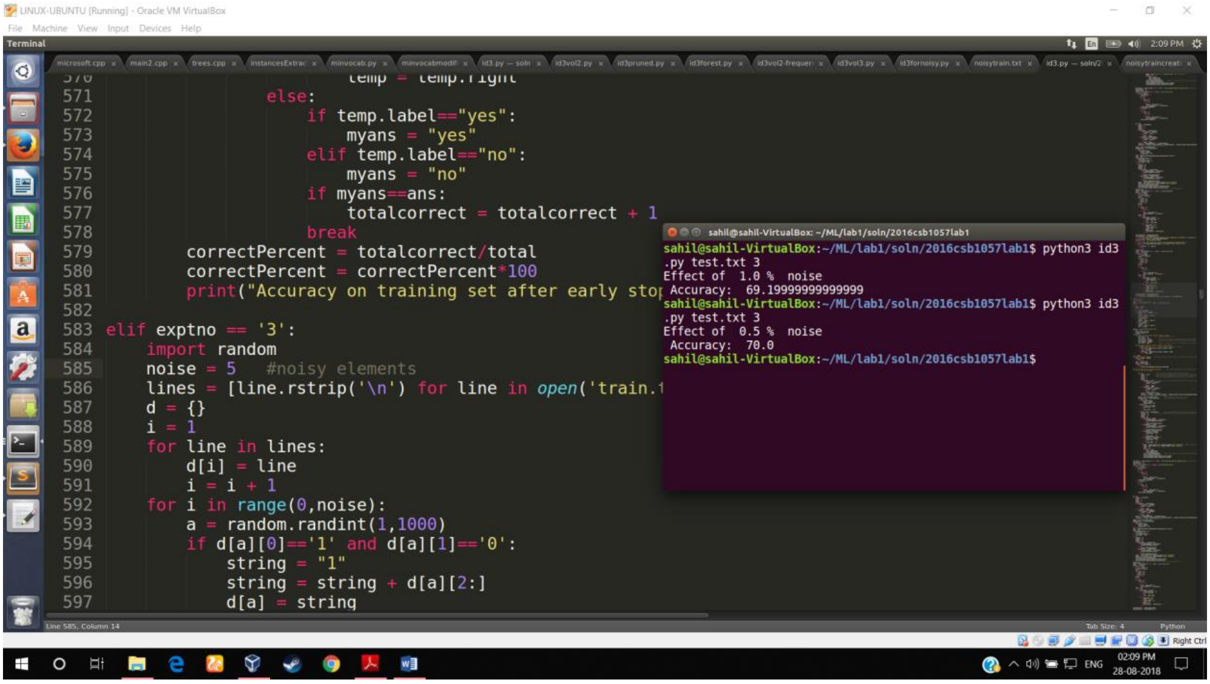


So, as we can see early stopping increases the accuracy by a percent as the size of tree decreases and thus the overfitting also decreases. But the stopping criteria must not be too high, in my case it was 30 examples in a node.

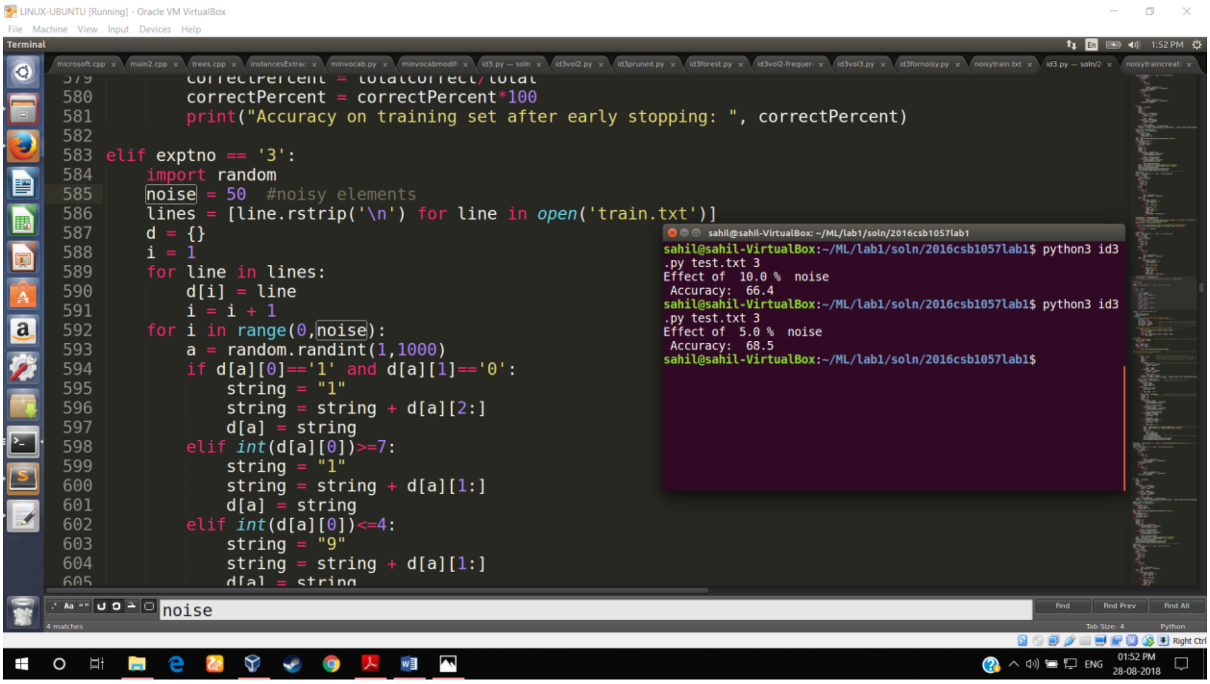
Too high or too low stopping criteria decreased the accuracy.

EXPT #3: ID3 with noisy examples:

Noise(in percent)	Accuracy
0.5%	70%
1%	69.19%
5%	68.5%
10%	65%



LINUX-UBUNTU [Running] - Oracle VM VirtualBox
else:
if temp.label=="yes":
myans = "yes"
elif temp.label=="no":
myans = "no"
if myans==ans:
totalcorrect = totalcorrect + 1
break
correctPercent = totalcorrect/total
correctPercent = correctPercent*100
print("Accuracy on training set after early stop
elif exptno == '3':
import random
noise = 5 #noisy elements
lines = [line.rstrip('\n') for line in open('train.t
d = {}
i = 1
for line in lines:
d[i] = line
i = i + 1
for i in range(0,noise):
a = random.randint(1,1000)
if d[a][0]=='1' and d[a][1]=='0':
string = "1"
string = string + d[a][2:]
d[a] = string
sahil@sahil-VirtualBox: ~/ML/lab1/soln/2016csb1057lab1
sahil@sahil-VirtualBox:~/ML/lab1/soln/2016csb1057lab1$ python3 id3
.py test.txt 3
Effect of 1.0 % noise
Accuracy: 69.19999999999999
sahil@sahil-VirtualBox:~/ML/lab1/soln/2016csb1057lab1$ python3 id3
.py test.txt 3
Effect of 0.5 % noise
Accuracy: 70.0
sahil@sahil-VirtualBox:~/ML/lab1/soln/2016csb1057lab1$
02:09 PM
28-08-2018

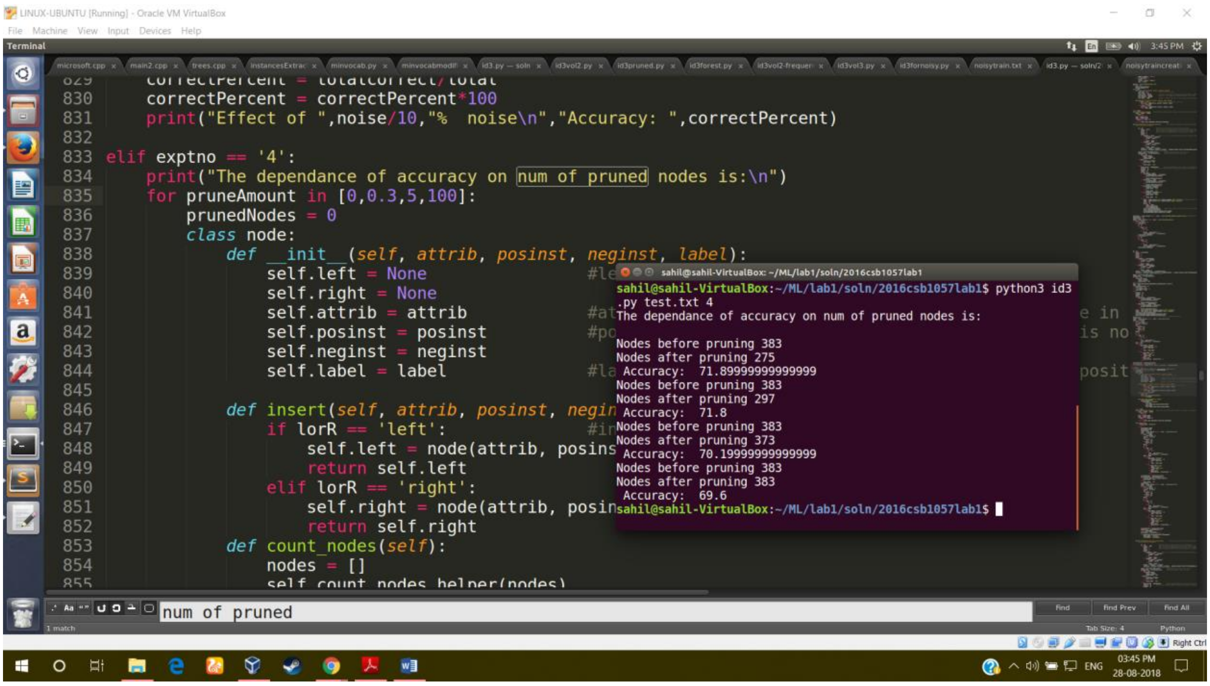


LINUX-UBUNTU [Running] - Oracle VM VirtualBox
correctPercent = correctPercent*100
print("Accuracy on training set after early stopping: ", correctPercent)
elif exptno == '3':
import random
noise = 50 #noisy elements
lines = [line.rstrip('\n') for line in open('train.txt')]
d = {}
i = 1
for line in lines:
d[i] = line
i = i + 1
for i in range(0,noise):
a = random.randint(1,1000)
if d[a][0]=='1' and d[a][1]=='0':
string = "1"
string = string + d[a][2:]
d[a] = string
elif int(d[a][0])>=7:
string = "1"
string = string + d[a][1:]
d[a] = string
elif int(d[a][0])<=4:
string = "9"
string = string + d[a][1:]
sahil@sahil-VirtualBox: ~/ML/lab1/soln/2016csb1057lab1
sahil@sahil-VirtualBox:~/ML/lab1/soln/2016csb1057lab1$ python3 id3
.py test.txt 3
Effect of 10.0 % noise
Accuracy: 66.4
sahil@sahil-VirtualBox:~/ML/lab1/soln/2016csb1057lab1$ python3 id3
.py test.txt 3
Effect of 5.0 % noise
Accuracy: 68.5
sahil@sahil-VirtualBox:~/ML/lab1/soln/2016csb1057lab1$
noise
01:52 PM
28-08-2018

EXPT #4: ID3 with post-pruning(reduced error pruning)

Nodes before pruning	Nodes after pruning	Accuracy
383	275	71.9
383	297	71.8%
383	383(no pruning)	69.6%

So, the number of nodes pruned directly affects the accuracy, if a lot of nodes are pruned then the accuracy decreases but up until a level the accuracy increases by upto 2% on pruning



EXPT #5: Random forests using feature bagging

Randomly 2000 attributes were chosen from the list of 5000 attributes and then a number of trees were created and the maximum output of those trees was chosen as my output.

Number of trees	Accuracy
5	72.1%
10	72.8%
15	73%
25	77.3%