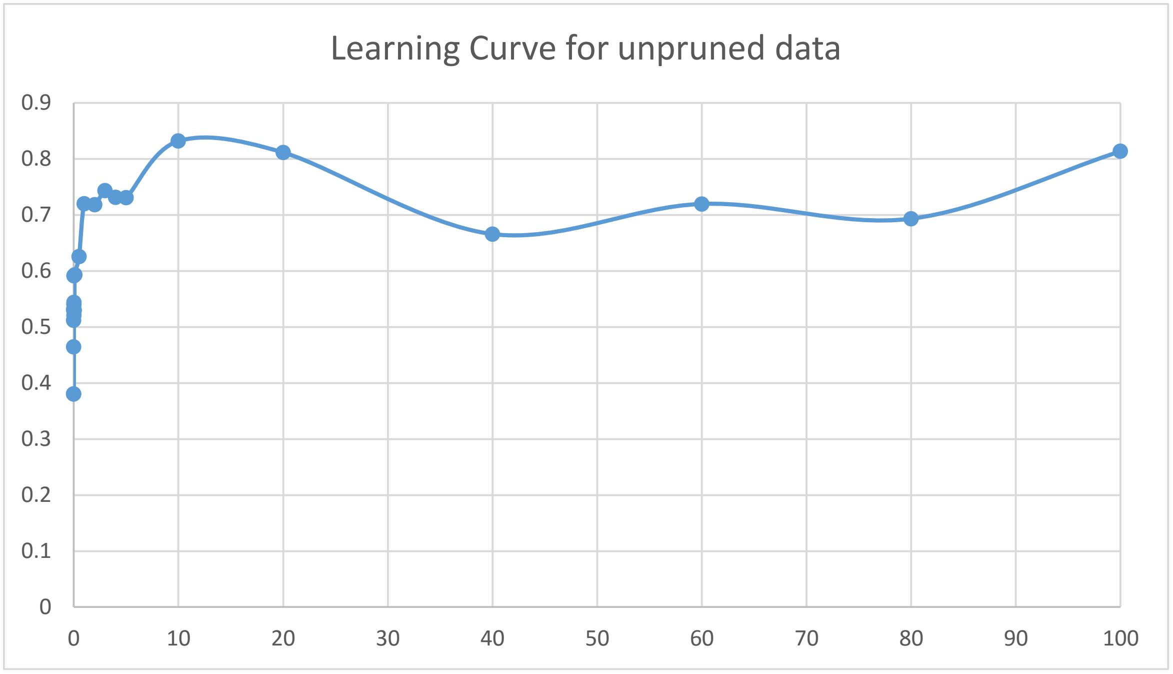
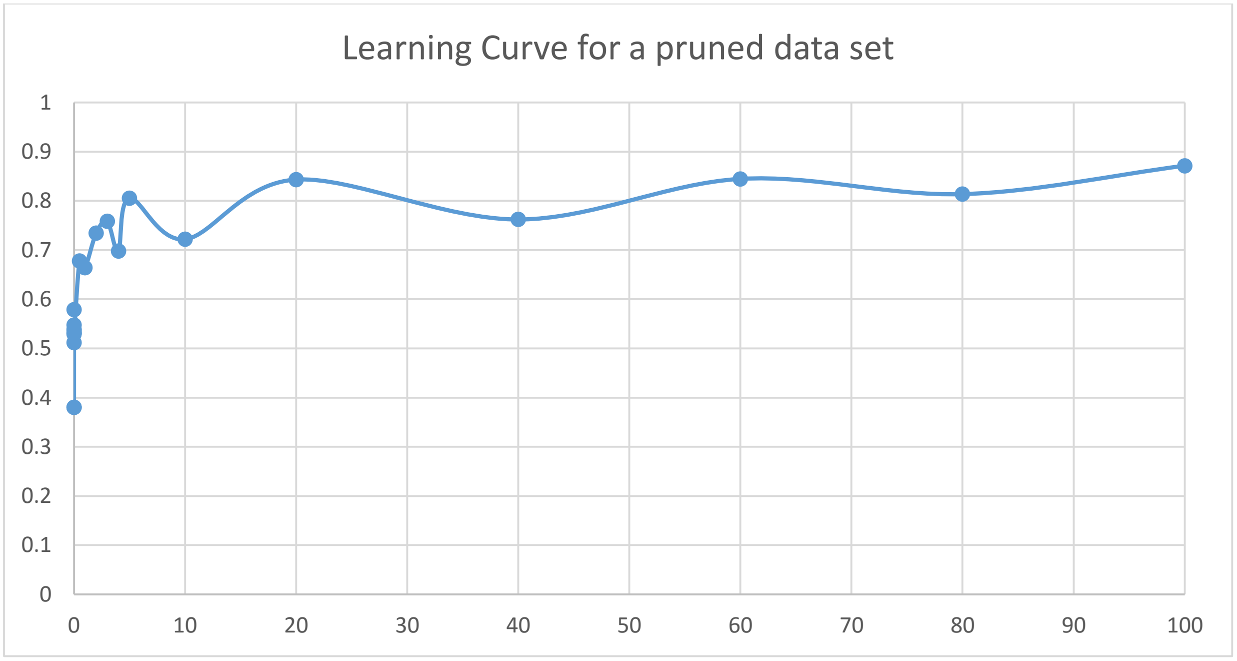
Roman Grigorii

1) In the training set, I handled the missing cases by ignoring these examples during the stage of calculating the information gain upon splitting on a particular attribute. However, for nominal splits I would place the examples with missing values into the largest set within the given dictionary. In other words, I would put the missing attributes with the splitting attribute which was mode of the known attribute values. For the numerical splits, I would place the missing attributes into the child with the maximum number of entries. This is equivalent to finding the average of the attribute values, assigning this average to the examples missing this attribute, and then lacing them into the tree according this is average value.
   In the validation set, I handled the missing attribute values by filling them with either the mode (for nominal) or the average (the numerical) of the entire data set.
2)
```
(numinjured<2.0^(homeaway=0^((oppnuminjured<2.0^(numinjured<1.0^(oppnuminjured<1.0^(numinjured<0.0
^(numinjured=>-
1.0^(temperature=>59.90583102^v(dayssincegame=>1.0))))v(numinjured=>0.0^(oppnuminjured=>-
1.0^(oppdayssincegame<3.0^(oppnuminjured<0.0^(oppdayssincegame<2.0^(startingpitcher=1^((oppstartingpit
cher=1)v(oppstartingpitcher=2)))v(startingpitcher=2^((weather=0)v(weather=-
1)))v(startingpitcher=3^((weather=0)vv(weather=-
1)))v(startingpitcher=4^((oppstartingpitcher=2)v(oppstartingpitcher=4)))v(startingpitcher=5^((oppwinningperce
nt=>0.614672704))))(oppdayssincegame=>2.0^(winpercent=>0.060863252^(oppstartingpitcher=1^((dayssincega
me=>2.0^(opprundifferential<76.0^(oppwinningpercent=>0.405493803^(temperature=>52.93667041^v(startin
gpitcher=5^((winpercent<0.588968144(winpercent=>0.588968144))))))v(opprundifferential=>76.0^(startingpitc
her=1)))))v(oppstartingpitcher=3^((oppwinningpercent<0.903209529(oppwinningpercent=>0.903209529)))
```

3) I used a small portion of the DNF output in problem 2. The interpretation is as following: If the number of injured players is below 2 and the number of home-away games is 0 and the number of injured players on opposing team is less than 2 or the number of injured players on own team is equal to -1 and the temperature is higher than 60 degrees with days since last game being greater than 1, the game will be won. The game can also be won with a number of other conditions. For example, if the number of injured players is greater or equal to 0 and number of injured players on opposing team is greater than -1 and there have been less than 3 days since last game for the opponent, and the number of injured opponent teammates is less than 0 with a starting pitcher for our team being #1 and the opposing pitcher being 1 or 2, we will win the game is long as the weather is 1 or 0. However, if the starting pitcher is 4, we need the weather to be either -1 or 0 in order to win.
4) To prune the tree I used several functions. The first (backProb) given an array of attribute values (referred to as MAP from here on out) , goes down the tree based on these values, and when it reaches the end of the array it replaces the node it arrives at with a labeled node, with a value a user specifies. The second function (check_error) takes MAP and a tree, and checks if replacing a split node with the given MAP, would increase the validation accuracy. If it does, the function returns the tree with the replaced node, and if it doesn't it returns the original tree. The last function, reduced_error_pruning, goes through all the possible MAP values, and returns a tree with maximized validation accuracy, after the splitting nodes that reduced the validation accuracy have been replaced with labeled nodes (leafs).
5)
```
(numinjured<2.0^(homeaway=0^((oppnuminjured<2.0^(numinjured<1.0^(oppnuminjured<1.0^(numinjured<0.0
^(numinjured=>-
1.0^(temperature=>59.90583102^v(dayssincegame=>1.0)))))(numinjured=>0.0^(oppnuminjured=>-
1.0^(oppdayssincegame<3.0^(oppnuminjured<0.0^(oppdayssincegame<2.0^(startingpitcher=1^((oppstartingpit
cher=1)v(oppstartingpitcher=2)v))v(startingpitcher=2^((weather=0)v(weather=-
1)))v(startingpitcher=3^((weather=0)vv(weather=-
1)))v(startingpitcher=4^((oppstartingpitcher=2)v(oppstartingpitcher=4)))v(startingpitcher=5^((oppwinningperce
nt=>0.614672704))))
```

6) The pruned set is approximately 4/5 of the original tree. Since for every split there are a little bit above two Nodes (it would have been 2 if we only had numerical data, but the nominal data increases the number of splits). So if the number of splits is approximately 2/5 of the number of splits in the original set.
7) The validation accuracy for unpruned tree is 86.4% and for pruned it is 89.1% (done for 10 trials each). The higher validation accuracy for the pruned tree is due to the fact that we are no longer overfit our data. Overfitting reduces the accuracy, because overfit models are extremely sensitive to noise, or outliers, in the data and thus result in worse predictions for validation data.
8)





The learning curve for pruned data is slightly offset upwards than the curve for unpruned data. Also, the margin of accuracy between the two curves increases with increased percentage of data that was used to train our model. This make sense, because when we rely on false leads that come up in data, they propagate the error to their children and so forth, and result in more error.

9) I think the pruned model will perform better because I handle the missing data points by finding the average for numerical and mode for nominal cases, and since the pruned model generally predicts better, I think it will perform better here as well. However, one must remember that pruning a tree means that some of it will be cut away. So, if the prediction set shares some of the entries from the training set, then we may see a decrease in accuracy, just because the overfitting that was a problem before is beneficial because it will fit to many entries of the prediction set.

10) I worked on the entire homework on my own because I did not receive any contributions from my groupmates. The exception is predictions code that was shared between me and Gale Curry.

11) Limited step size is not a bad option. However, I noticed that very small step sizes provide several percentages higher in validation accuracy. I ran ID3 on the first 20,000 of the data set with step size = 1. It took several hours, but my accuracy, after the tree has been pruned, was a little over 91%. So, to learn something from the data, a large step size is not bad, but making really great prediction will need a smaller step size.
Using plain IG for finding the best splitting attribute, I found the validation accuracy to decrease by less than 1%. I didn’t see that using IG versus IG/IV has improved the model significantly.