Aatmay S. Talati

Machine Learning (CS 4641)

Project 1: Supervised Learning

Date: Jan 28, 2018 (Spring 2018)

Analysis of Supervised Learning

Machine Learning: A Gentle Introduction

Machine learning is a field of computer science that gives computers the ability to learn without being explicitly programmed [3]. There are three sub-types of machine learning: Supervised Machine Learning, Unsupervised Machine Learning and Reinforcement Learning. In terms of this assignment, we will focus on Supervised Learning only. Engineers, Computer Scientists and Researchers increasingly rely on machine learning algorithms now-a-days, and it has been of significant help in terms of analyzing data and looking for patterns that might lead to novel conclusions.

Choosing the Datasets:-

Choosing the correct, and interesting datasets for the project was a bit challenging and time-consuming task for me. Via using the links provided on Canvas, and conducting a little research in terms of the finding the datasets on the internet, I found two interesting datasets EEG Eye State and HR (Human Resources) Analytics from UCI (University of California, Irvine) Machine Learning Repository and Kaggle, respectively.

About the Datasets:

EEG Eye State:

All data is from one continuous EEG measurement with the Emotiv EEG Neuroheadset. The duration of the measurement was 117 seconds. The eye state was detected via a camera during the EEG measurement and added later manually to the file after analyzing the video frames. '1' indicates the

eye-closed and '0' the eye-open state. All values are in chronological order with the first measured value at the top of the data [1].

HR Analytics:

The main goal of this dataset is to know knowing why the best and most experienced employees of the company leave the company prematurely, and also, we can predict which employee will leave the company – depending upon the inputs we have so far. This dataset includes the reason of leaving the company/firm due to many reasons which are included in the attributes: satisfaction level, last evaluation, number of projects, average monthly salaries, time spent in company, work accidents, people left, promotion from last 5 years, salary, and department.

Why the Datasets are Interesting?

EEG Eye State:

As a researcher working on a ground-breaking research project at Georgia Tech in the field of the Brain Computer Interface, we primarily reply on brain signals. In order to obtain brain signals we primarily use few techniques like fMRI, EEG, SSEVEP and P300.

Having this dataset has its practical use along with its real-world applicability, was the primary reason behind choosing this dataset for my machine learning project dataset. By using this dataset, we can predict what exact combination of different attributes of one EEG reading determine if the eye of the subject is open or closed and thus helping us go deeper into understanding our brain and its functioning.

HR Analytics:

In terms of the growth of the company or firm, it is very important to make sure that employees do not leave the company/firm prematurely. There are many affecting factors behind the reasons of leaving the company other than explained scenarios in the datasets, such as employer-employee relations,

colleague relations, etc. This dataset has approximately 15,000 instances.

This dataset also gives some insights and makes me think that this scenario may take place one day in future, and I will know how to drill-down the main cause, like is the salary main concerned? Are promotions required for an employee to grow in his career? Is the working environment important in a company? Does the relationship between coworkers needs to be good to communicate well and stay comfortable while working? And many more.

Decision Trees:

Decision tree learning uses a decision tree to go from observation about an item to the conclusion about the item's targeted value. Tree models where the target variable can take a discrete set of values are called classification trees; in these tree structures, leaves represent class labels and branches represent conjunctions of features that lead to those class labels. [4]

Decision Tree: EEG Eye State:

UnPruned	Confidence Interval	minNumberObject	Tree Size	Numbr of Leaves	Time Taken to test model on training data	Mean Abs. Error Training	Training %	Time Taken to build Model	Mean Abs.Error Testing	Cross Validation %
F	0.2	1	1825	913	0.03	0.1338	97.93%	0.84	0.3838	84.53%
F	0.2	2	1503	752	0.03	0.1631	96.92%	0.82	0.3748	84.61%.
F	0.2	4	1215	608	0.02	0.1991	95.33%	0.8	0.3697	84.27%
F	0.4	1	1949	975	0.03	0.119	98.33%	0.88	0.3864	84.43%
F	0.4	2	1651	826	0.04	0.1488	97.40%	0.81	0.3797	84.41%
F	0.4	4	1339	670	0.02	0.189	95.74%	0.81	0.3703	84.31%
F	0.5	1	1961	981	0.02	0.118	98.35%	0.85	0.3866	84.44%
F	0.5	2	1657	829	0.02	0.1483	97.41%	0.88	0.3796	84.43%
F	0.5	4	1363	682	0.02	0.1874	95.80%	0.79	0.3702	84.29%
T	N/A	1	2097	1049	0.02	0.1075	98.59%	0.65	0.3878	84.39%
T	N/A	2	1727	864	0.02	0.1442	97.53%	0.7	0.38	84.38%
T	N/A	4	1397	699	0.03	0.1858	95.82%	0.58	0.3696	84.26%

Decision Tree: HR Analytics

UnPruned	Confidence Interval	minNumberObject	Tree Size	Number of Leaves	Time Taken to test model on training data	Mean Abs.Error Training	Training %	Time Taken to build Model	Mean Abs.Error Testing	Cross Validation %
F	0.2	1	12977	6491	0.63	0.1079	92.04%	2.06	0.3303	40.70%
F	0.2	2	7867	3936	0.1	0.1799	75.58%	1.62	0.4485	54.19
F	0.2	4	1503	768	0.14	0.3736	67.00%	0.56	0.4442	53.09%
F	0.4	1	3683	1862	0.04	0.3164	75.21%	0.67	0.4513	56.52%
F	0.4	2	2875	1458	0.03	0.3347	72.92%	0.57	0.4503	55.20%
F	0.4	1	1972	1002	0.02	0.189	95.74%	0.81	0.3703	84.31%
F	0.5	1	3819	1930	0.03	0.3138	75.51%	0.57	0.4527	56.61%
F	0.5	2	2967	1504	0.03	0.3329	73.14%	0.56	0.4513	55.23%
F	0.5	4	2017	1025	0.03	0.3618	68.82%	0.47	0.447	53.37%
T	N/A	1	4589	2319	0.03	0.3035	75.58%	0.47	0.4531	57.23%
T	N/A	2	3291	1670	0.03	0.3286	73.48%	0.48	0.4518	55.47%
T	N/A	4	2213	1127	0.03	0.3588	68.99%	0.34	0.4472	53.37%

During running the decision trees, I set pruning on (Unprune = False), three Confidence Intervals and three minNumbObjects. Decision tree with pruning have worked completely differently on my both datasets. Confidence Interval and minNumbObject has significant impact on tree size, number of leaves, Mean Abs. Error and Training%. It was interesting to notice that as the confidence factor increases the size and the number of leaves decreases.

$$Confidence\ Factor\ \&\ minNumbObj \uparrow => Tree\ Size \downarrow \&\ Number\ of\ Leaves \downarrow$$

Without having any doubts, I can clearly say that decision tree algorithm worked extremely well on EEG dataset. It predicted the outcome pretty accurately compared to HR dataset. I think due to excessive number of attributes in second dataset, it didn't perform as well as first dataset. Also, we can clearly notice that when we turn off the pruning (Unpruned = True) the dataset did a bad job in terms of the cross validation on 10 folds.

Neural Nets:

Neural Networks are one of the robust learners, but only when given the

right set of parameters they can model any function. Personally, I feel that neural nets are really complex to construct, and it requires to have a lot of patience, time and energy. Finding the correct parameter to perform neural net is really challenging. Running Neural Nets on weka was interesting and extremely time consuming as often I was running out of the memory power. Then I decided to switch to Linux Machine, and that helped a bit. Eventually I ended up trimming the data by 1/3rd, and still those processes took well over couple hours on an 8 GB RAM Linux Machines.

Dataset: EEG Eye State

M	L	H	Mean Abs Error - Training	Epochs (N)	% Correct	Time taken for Training dataset	Mean Abs Error- Cross Validation	%Correct	Time taken for Cross Validation
0.4	0.5	a	0.3629	500	63.93%	0.01	0.2692	80.26%	6.12
		a	0.2757	1000	79.70%	0.01	0.236	82.12%	14.53
		a	0.2295	1500	83.12%	0.02	0.3362	85.34%	17.05
0.2	0.1	a	0.2976	500	77.49%	0.01	0.2473	82.99%	5.88
		a	0.2215	1000	84.43%	0.01	0.194	86.58%	11.44
		a	0.2923	1500	88.67%	0.01	0.295	88.25%	16.74
0.2	0.3	a	0.2605	500	55.74%	0.04	0.247	82.56%	5.42
		a	0.2235	1000	8459%	0.01	0.1979	87.07%	10.79
		a	0.1742	1500	89.82%	0.02	0.1748	88.86%	16.23
0.2	0.5	a	0.3265	500	77.00%	0.01	0.2697	80.74%	5.41
		a	0.2411	1000	82.77%	0.01	0.234	83.47%	10.77
		a	0.2328	1500	82.32%	0.01	0.207	85.74%	16.16

Dataset: HR Analytics

M	L	H	Mean Abs Error - Training	Epochs (N)	% Correct	Time taken for Training dataset	Mean Abs Error- Cross Validation	%Correct	Time taken for Cross Validation
0.4	0.5	a	0.3364	500	54.10%	0.02	0.3469	51.33%	8.29
		a	0.3334	1000	54.53%	0.02	0.3468	50.74%	17.3
		a	0.3326	1500	54.88%	0.02	0.3468	51.33%	23.83
0.2	0.1	a	0.3305	500	60.16%	0.07	0.3466	53.76%	8.82
		a	0.327	1000	60.98%	0.02	0.3465	53.89%	17.33
		a	0.3255	1500	61.28%	0.03	0.3466	54.05%	26.62
0.2	0.3	a	0.3353	500	59.86%	0.02	0.3456	53.52%	8.33
		a	0.3322	1000	60.64%	0.02	0.3461	53.46%	16.19
		a	0.3388	1500	60.74%	0.02	0.3463	53.33%	24.24
0.2	0.5	a	0.3352	500	53.97%	0.02	0.3461	52.05%	8.65
		a	0.3323	1000	54.66%	0.02	0.3456	52.21%	16.1
		a	0.331	1500	54.93%	0.02	0.3458	52.16%	27.52

In terms of the Neural Nets, I've changed the values of M, L and N. Recorded Cross Validations% are over the 10 folds only. My first reflection by looking at the data is, as we increase the number of epochs, the accuracy of the neural nets also increases and mean abs. error decreases simultaneously.

$$Number\ of\ Epochs \uparrow => Accuracy\ of\ Neural\ Nets \uparrow \& Mean\ Abs.Error \downarrow$$

Second reflection is, as learning rate increases, the accuracy decreases.

$$Learning\ Rate \uparrow => Accuracy\ of\ Neural\ Nets \downarrow$$

Third reflection is, momentum increases, accuracy Decreases.

$$Momentum \uparrow => Accuracy\ of\ Neural\ Nets \downarrow$$

Reason behind our reflections are as following:

- Epochs are linearly proportional to training time. Thus, longer training time

leads to higher accuracy as algorithm fits the data more accurately.

- Learning rate determines the magnitude of weights and a higher learning rate results in more less accurate output but takes relatively shorter to get it.
- Momentum controls the convergence at a local minimum.

As a summary of my both dataset, I can say that in my case, EEG dataset did much better than HR dataset.

KNN

For the K-Nearest Neighbor algorithm, I've used number of neighbors as 1, 5, 10, 15, 20, 25 for two types of distance weightings: Unweights and 1/Distance. After conducting a research, I found out that I should use 1/distance, because 1/distance is more appropriate for normalized datasets. The results are clear below. KNN didn't work well on EEG dataset as its not as normalized as HR dataset, and in order to demonstrate the same I didn't change 1/distance to 1-distance for EEG Dataset.

In the data tables below, F represents "Unweighted" and T represents "1/distance" distaceWeighting. Just like every other algorithm I did for this project I kept folds to "10" for cross-validations.

Dataset: HR Analytics

Weighted	Nearest Neighbors	Time to build on Training	Training %	Mean Abs. Error Training	Testing %	Mean Abs Error Testing	Time taken to Build Model
F	1	41.01	100%	0.0007	62.21%	0.2517	0
F	5	55.66	67.45%	0.2675	52.55%	0.3387	0
F	10	61.01	61.98%	0.3161	51.71%	0.354	0
F	15	60.16	59.23%	0.3334	51.35%	0.3581	0
F	20	79.94	57.95%	0.3417	51.31%	0.3604	0
F	25	77.52	57.02%	0.3472	51.08%	0.3686	0.1
T	1	43.5	99.92%	0.0006	62%	0.2517	0.01
T	5	56.42	99.88%	0.032	62.88%	0.266	0.01
T	10	55.97	99.76%	0.0529	63.24%	0.2742	0.01
T	15	65.49	99.69%	0.0681	63.82%	0.279	0.01

T	20	59.42	99.60%	0.0803	63.73%	0.284	0.01
T	25	80.72	99.50%	0.0906	81.16%	0.2913	0.01

Dataset: EEG Eye State

Weighted	Nearest Neighbors	Time to build on Training	Training %	Mean Abs. Error Training	Testing %	Mean Abs Error Testing	Time taken to Build Model
F	1	12.12	100%	0.0001	83.65%	0.1635	0
F	5	25.34	90.06%	0.1683	83.79%	0.2184	0
F	10	27.14	86.37%	0.22	82.49%	0.2489	0
F	15	28.26	85.02%	0.2475	81.96%	0.2686	0
F	20	29.37	83.59%	0.2663	81.19%	0.284	0
F	25	30.38	82.98%	0.2811	80.93%	0.2958	0.1
T	1	47.19	100	0	100%	0	0.01
T	5	64.86	90.34%	0.1501	83.79%	0.216	0.01
T	10	68.14	90.07%	0.2043	83.81%	0.2457	0.01
T	15	69.42	86.36%	0.2334	82.05%	0.2649	0.01
T	20	74.84	86.03%	0.2532	82.18%	0.2798	0.01
T	25	77.54	84.53%	0.2686	81.16%	0.2913	0.01

We can clearly see that when it comes to 1/distance in terms of the distanceWeighting, the testing data is always approximately 100%, and that's the evidence that KNN overfitted for weighted distance. In the continuation of the same, we can observe that KNN overfitted to a much larger extent for HR as compared to EEG. However, in terms of the EEG dataset, I can make a statement by looking at the table that, it has training precision and testing precision are much closer to each other, especially for the unweighted distanceWeighting. That means that KNN is executing very fitting this dataset almost perfectly. For EEG

$$Number\ of\ Nearest\ Neighbors \uparrow => Mean\ Abs. Error \uparrow\ \&\ Accuracy \downarrow$$

Although increasing the number of neighbors tend to lead towards appearing increment in mean abs error and decrement in tendency to overfit. The reason behind that statement could be that, EEG is relatively precise and adding more neighbors could lead to adding more noise and that's the reason behind

dropped accuracy. As EEG has more numerical values than HR dataset, it could have led to higher accuracy as explained above.

As a conclusion I can say that

$$Weighted\ Runs\ \Rightarrow\ Overfits\ the\ data$$

Whereas,

$$Unweighted\ \Rightarrow\ Peforms\ Better$$

Boosting

In terms of the testing accuracy, EEG dataset and HR dataset – both of them have significantly higher training accuracy during boosting.

Dataset: EEG Eye State

weightThreshold	Seed	Iteration	Confidence Factor	minNumbOj	Tree Size	Leaves	Root Mean squared error	Relative Abs Error	Cross Validation	Time Taken
100	1	10	0.05	2	1275	638	0.2622	16.19 %	92.04%	24.8
100	1	10	0.25	2	1371	686	0.2683	16.83 %	91.69%	22.98
100	1	10	0.5	2	1371	686	0.2661	16.60 %	91.74%	14.11
100	1	20	0.05	2	1219	610	0.2409	12.75 %	93.71%	43.4
100	1	20	0.25	2	1085	543	0.2381	12.41 %	93.84%	59.66
100	1	20	0.5	2	1035	518	0.2359	12.31 %	94%	35.28
100	1	30	0.05	2	1667	634	0.2309	11.40 %	94.27%	61.95
100	1	30	0.25	2	1081	541	0.2239	10.76 %	94.65%	83.67
100	1	30	0.5	2	1189	595	0.2319	11.49 %	94.37%	73.09
100	1	10	-	2	1535	768	0.2726	17.34 %	91.40%	8.64

Dataset: HR Analytics

weightThreshold	Seed	Iteration	Confidence Factor	minNumbOj	Tree Size	Leaves	Root Mean squared error	Relative Abs Error	Cross Validation	Time Taken
100	1	10	0.05	2	2143	1084	0.4299	79.40 %	59.99%	7.72
100	1	10	0.25	2	3127	1580	0.451	67.40 %	63.05%	7.21
100	1	10	0.5	2	3627	1830	0.4561	66.55 %	62.89%	7.08
100	1	20	0.05	2	1799	908	0.4344	72.12 %	62.70%	13.33
100	1	20	0.25	2	3122	1591	0.4649	64.48 %	63.29%	14.78
100	1	20	0.5	2	3673	1889	0.4699	64.55 %	63.30%	14.31
100	1	30	0.05	2	1925	971	0.437	70.22 %	63.01%	20.27
100	1	30	0.25	2	2495	1276	0.4714	64.32 %	63.33%	20.45
100	1	30	0.5	2	2001	1057	0.4723	64.15 %%	63.43%	20.59
100	1	10	-	2	4423	2224	0.4582	65.98 %	63.07%	6.47

In both datasets we can see that weighThresholds seeds are same over 10, 20 and 30 iterations. Interestingly I observed that

$$Number\ Of\ Iterations \uparrow => Testing\ Accuacy \uparrow Relative\ Abs\ Error \downarrow$$

The reason behind this is, in every iteration the biased distribution of the dataset changes more and that leads to

$$Weak\ Learners\ => Stronger\ Learners$$

$$Stronger\ learners\ => Testing\ Accuacy \uparrow \&\ Errors\ \downarrow$$

Also, I've marked that pruning while boosting significantly helps in terms of the accuracy.

$$Pruning \uparrow => Accuracy \uparrow$$

And, in the same way I can say that lower pruning leads to underfitting. I think as HR data has more noise, pruning significantly helps in terms of getting better accuracy. EEG data didn't work well when I lowered the confidence factor, because EEG dataset have lower amount of noise data compared to EEG dataset. So, in that case after a certain extent of confidence factor, it just reduces the tree size.

SVMs

In terms of the SVMs, I've primarily performed on 3 different types of Kernels: Radial, Polynomial and Linear. I've conducted experimented each kernel with degree or gamma of 1 or 2.

Dataset: EEG Eye State

Kernal	Degree / Gamma	Time to build on Training	Training %	Mean Abs. Error Training	Testing %	Mean Abs Error Testing	Time taken to Build Model
Linear	1	0.43	74.17%	0.2582	74.04%	0.2595	5.59
Linear	2	0.24	74.17%	0.2582	74.04%	0.2595	5.57
Poly	1	0.3	74.41%	0.2558	73.93%	0.2606	2.88
Poly	2	0.33	77.91%	0.2208	74.14%	0.2558	2.29
Radial	1	2.66	100%	0	57.08%	0.4291	3.81
Radial	2	2.69	100%	0	57.08%	0.4291	3.74

Dataset: HR Analytics

Kernal	Degree / Gamma	Time to build on Training	Training %	Mean Abs. Error Training	Testing %	Mean Abs Error Testing	Time taken to Build Model
Linear	1	0.47	55.25%	0.2983	55.25%	0.2983	56.56
Linear	2	0.49	55.25%	0.2983	55.25%	0.2983	56.59
Poly	1	0.53	55.25%	0.2983	55.25%	0.2983	80.95
Poly	2	0.66	54.74%	0.3017	51.78%	0.3214	70.23
Radial	1	1.97	58.61%	0.2759	52.98%	0.5598	2.45
Radial	2	2.92	58.61%	0.2759	52.98%	0.5598	3.6

Looking at the data tables primarily I can say that

$$Degree/Gamma \downarrow => Testing\ Accuacy \uparrow Relative\ Abs\ Error \downarrow$$

In terms of the polynomial kernels

$$Degree\ or\ Gamma \uparrow => Complicated\ Kernals \uparrow => Fit\ Data\ More\ Correctly => Accracy \uparrow \& Errors \downarrow$$

In terms of the EEG data, we can see that kernel is not corresponding with the function, which leads to approx. 100% accuracy in training but comparatively much lower accuracy for testing. That tells us that RBF erroneously didn't perform well on that dataset.

Conclusion:

After conducting every algorithm on both datasets, I can tell with an absolute certainty that EEG dataset stood out to have an outstanding performance for mostly all aforementioned algorithms. We can also see that KNN worked the best on EEG dataset, and it gave a highest accuracy among all others. It splits all the instances consistently based on class values.

HR dataset does include a lot of noise in it, and that's the reason behind failing into multiple scenarios. Plus compared to EEG datasets, it has more attributes – which significantly leads to higher processing time when it comes to cross validation on 10 folds.

References:

1) https://archive.ics.uci.edu/ml/datasets/EEG+Eye+State
2) https://www.kaggle.com/ludobenistant/hr-analytics-1
3) https://en.wikipedia.org/wiki/Machine_learning
4) https://en.wikipedia.org/wiki/Decision_tree_learning