Machine Learning Algorithms from Scratch Using Python

Topgyal Gurung

December 16, 2019

1 Abstract

Nowadays, with the help of libraries like Scikit-learn and Weka, machine learning predictions like regression and classification can be easily performed on a given problem and data set. But as a computer science student and a machine learning enthusiast, understanding how machine learning algorithms are built from scratch is important. There are many machine learning algorithms developed to solve real life problems. K Nearest Neighbor, Decision Tree and K Means Clustering algorithms are basic fundamental machine learning algorithms. In this paper, I present my research and work on those algorithms from scratch to understand their inner workings and mathematics. The data sets used to implement the algorithms are Breast Cancer Wisconsin and Monk's Problem Data sets from UCI Machine Learning Repository.

2 Introduction

Machine learning is a sub field of computer science and subset of Artificial Intelligence. Using machine learning algorithms we can built statistical model based on data sets and use it to solve practical problems. Samuel Arthur coined the term Machine Learning in 1969 while at IBM. It is conglomeration of data modeling, probability and statistics. Machine learning algorithms acquire knowledge by extracting patterns from raw data. It achieves such by using and processing data through an error function, a loss function and an objective function(Burkov, 2019)

In this project, the main goal is to understand details on some supervised and unsupervised machine learning algorithms and build it from scratch using python with minimum use of libraries and trace their operation on actual data. The initial phase of the project was devoted on learning concepts about those algorithms. There is a use of Pandas and Numpy python libraries for reading data sets and data manipulation. The process of building those algorithms involve using Jupyter notebook to be able to see step by step of data preparation process and feature engineering. Model performance has been examined and also compared with the results from scikit-learn and Weka.

3 Background

Machine learning has been one of the most attractive fields in computer science. For the past few decades, the amount of data collected by computers has been enormous due to which we are able to use it to find useful insights by applying statistical and machine learning methods. Big companies along with small companies have adopted this new technology for the efficiency and return on investment.

It has been used in voice assistants, search engine, recommendation, facial recognition, spam filtering and many more. As a machine learning enthusiast, by building these classic algorithms from scratch will help me understand how the functions and algorithms are being able to learn from data and adapt with change.

4 Algorithms

4.1 K Nearest Neighbor

K Nearest Neighbor algorithm is a non-parametric method and a supervised machine learning algorithm for classification and regression. It works by finding optimal k closest to query from all the training data, then votes for most frequent label of the unknown sample for classification. KNN keeps all training examples in memory. We can use any distance metric measure that satisfies distance function properties. There are Euclidean distance, Manhattan distance, cosine similarity, Minkowski distance and many more.

There is a weighted KNN algorithm where nearest k points are given a weight using function called kernel function. Observation within learning set close to new observation get higher weight. One such example is inverse distance function where votes for test data point will be weighted by inverse distance of k points closest to it.

4.2 Decision Tree

Decision Tree is a supervised learning algorithms that can be used for both classification and prediction. Decision tree is a tree where each node represents feature or attribute, each link or branch represents decision or rule and each leaf represents an outcome whether a categorical or continuous value. It is an acyclic graph that can be used to make decisions.

Tree Construction and Tree pruning are two phases in decision tree algorithm. In Tree Construction, at first all instances of the training set are at the root of the tree. And then they are splitted into subsets based on attribute value test recursively. There are many ways for attribute selection to divide the examples. There is CART (Classification and Regression Tree) which uses Gini Index as metrics and there is ID3 (Iterative Dichotomiser 3) which uses entropy function and information gain as metric. Information Gain is based on Shannon Entropy which is a measure of uncertainty. High entropy corresponds to little information while low entropy substantial information. If entropy is zero, it means that it contains instance of only one class and if entropy is one, it contains equal number in binary class. The entropy is defined using following Equation:

$$E(S) = -^{n}\Sigma_{i=1} \times p_i \times log_2 \times p_i \tag{1}$$

n = number of class labels and p= proportion of instances in the data set

Information gain is difference in entropy before and after splitting. It is given by following equation:

$$InformationGain = Entropy(S) - [(WeightedAverage \times Entropy(eachfeature)] \tag{2}$$

Higher entropy represents more information gain and vice versa. Tree Pruning is removing of branches which do not contribute to a decision process and to avoid over-fitting. Two ways of pruning trees are pre-pruning and post-pruning. Pre-pruning is stop growing tree when data split is insignificant and post-pruning is to grow full tree then backtrack to identify and remove branches that reflect noise.

4.3 K-Means Clustering

K Means Clustering is an unsupervised learning that uses unlabeled data to extract information from them. It is used in graph clustering to classify similar data in groups. One of the main reasons behind coming up with unsupervised learning algorithms is because collecting and labeling a large set of sample patterns can be costly. In clustering, model returns the id of cluster for each feature vector in the data set.

In K- Means clustering algorithms, we choose pre-determined number of clusters k which is a hyperparameter. We use distance metric like Euclidean distance to calculate the similarity or categorize those data sets into groups. K-means problem is to find cluster centers that minimize the intra-class variance. To avoid exhaustive search, better approach involves using Expectation-Maximization iterative process. Expectation step is to assign points to nearest cluster center and Maximization step is to set the cluster centers to the mean.

5 Experiments and Results

5.1 Data set Details

5.1.1 UCI Breast Cancer Wisconsin

The data set has been already labeled so there is no prior labeling needed in order to use it for supervised learning algorithm. If we were to use the original data set, we needed to split into a training, validation and test set and also keeping in mind to maintain same class distribution of Benign and Malignant which is about 65:35. But we used the already provided train, valid and test set from professor. The total number of instances in the data set is 699 and 10 plus the class attribute for number of attributes. The provided data set contains 477 instances of train set, 67 test set and 135 validation set which is around 70 percent, 20 percent and 10 percent for train, valid and test set respectively.

5.1.2 MONK'S Problem

Monk's Problem data set is the comparison of three learning algorithms. The comparisons were performed by researchers and they wanted to answer which three problems is most difficult for a decision tree algorithm to learn. The three Monk's problems contain same set of six attributes, id column and binary class label 0 or 1. Monk 3 problem has 5 percent noise added to the training set. There are total 432 number of instances. The training and test sets have been already separated. The target concepts associated to the problems are:

MONK-1: (a1 = a2) or (a5 = 1)

MONK-2: EXACTLY TWO of a1 = 1, a2 = 1, a3 = 1, a4 = 1, a5 = 1, a6 = 1

MONK-3: (a5 = 3 and a4 = 1) or (a5 /= 4 and a2 /= 3) (5 percent class noise added to the training set)

The idea of implementing built decision tree model on these data sets is to see if the model can learn the correct rule from the training data provided.

5.2 Experimental Results

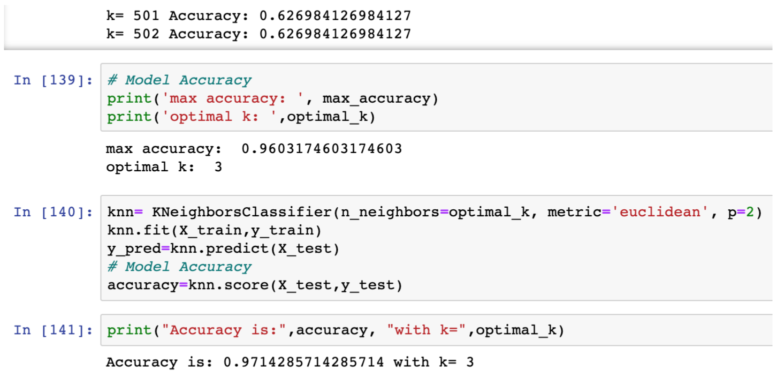
5.2.1 K Nearest Neighbor

I implemented the K Nearest Neighbor algorithm to UCI Breast Cancer Wisconsin Dataset. The first thing needed to do with the dataset before training the model was data preparation. The data file has been read using pandas library. The 'id' column was removed since it serves no purpose for training the model. The distance metric I used to find the nearest neighbor is Euclidean distance which calculates distance between two points in the coordinate plane.

$$\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \tag{3}$$

To find the optimal hyperparameter k, accuracies for k=1 to train size has been calculated after testing on validation set. One of the issue it could arise is that there could be some k with same accuracies. In that case, we could pick the greater k. Other important concept I learnt is if there is some point with zero distance while taking k points, inverse distance could backfire. To solve it, we need to do smoothing which is to use 1/(d+1) to ensure that distance(d) never gets to 0.

So, k is tested from 1 to train size of 476. I got the optimal k =102 with my algorithm. But, comparing it with the Scikit-learn libraries, it gives optimal k=3 with accuracy of around 96 percent. So, my model still needs an improvement to get the right solution. Using my optimal k to test the model with test set, it gives accuracy of around 87 percent. While the accuracy with scikit learn gives 97 percent.



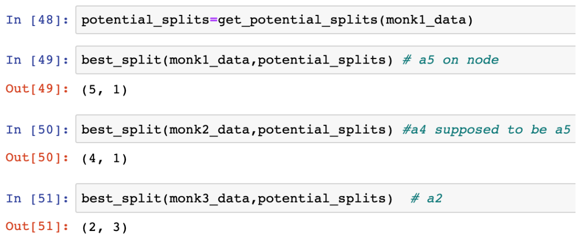
```
k= 501 Accuracy: 0.626984126984127
k= 502 Accuracy: 0.626984126984127

In [139]: # Model Accuracy
          print('max accuracy: ', max_accuracy)
          print('optimal k: ',optimal_k)

          max accuracy:  0.9603174603174603
          optimal k:  3

In [140]: knn= KNeighborsClassifier(n_neighbors=optimal_k, metric='euclidean', p=2)
          knn.fit(X_train,y_train)
          y_pred=knn.predict(X_test)
          # Model Accuracy
          accuracy=knn.score(X_test,y_test)

In [141]: print("Accuracy is:",accuracy, "with k=",optimal_k)

          Accuracy is: 0.9714285714285714 with k= 3
```

Figure 1: Optimal K and Accuracy from Scikit-learn KNN

5.2.2 Decision Tree

Decision Tree model is implemented on UCI Monk's Problem dataset. Monk's Problem also contains Id column which serve no purpose for training the model. So, id column has been dropped.Then the training set has been converted into numpy 2D arrray. Entropy for all parent node in the tree is calculated. I got the right entropy for all parent of three monk's problem. Monk-1 entropy is 1.0 since it contains equal number of class 0 and 1. Monk-2 problem's entropy is 0.95711748264771 as the number of classes is not homogenous. And Monk-3 problem's entropy is 0.999806132804711. The sum of entropy of children in the trees is also calculated to find information gain. This helps to see if entropy has decreased or not.

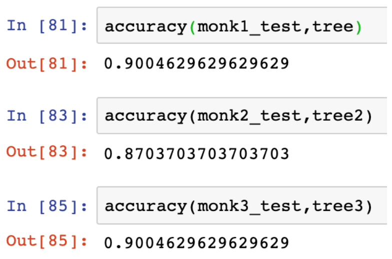
The result for the best split node for monk-1 is a5=1, monk-2 is a4=1 and monk-2 is a2=3. For monk-2, the best split node is supposed to be a5. Monk-2 is hardest to learn because every variable



```
In [48]: potential_splits=get_potential_splits(monk1_data)

In [49]: best_split(monk1_data,potential_splits) # a5 on node
Out[49]: (5, 1)

In [50]: best_split(monk2_data,potential_splits) #a4 supposed to be a5
Out[50]: (4, 1)

In [51]: best_split(monk3_data,potential_splits)  # a2
Out[51]: (2, 3)
```

Figure 2: Best split for each monk's problem

is independent from each other. Monk-1 is also hard as it takes time to realize a1 and a2 to be equal. The accuracies I got testing on test-set for monk-1 is 90 percent, monk-2 is 87 percent and monk-2 is 90 percent.



```
In [81]: accuracy(monk1_test,tree)
Out[81]: 0.9004629629629629

In [83]: accuracy(monk2_test,tree2)
Out[83]: 0.8703703703703703

In [85]: accuracy(monk3_test,tree3)
Out[85]: 0.9004629629629629
```

Figure 3: Accuracy for each monk's problem on test set

5.2.3 K Means Clustering

K Means is implemented on Breast Cancer Wisconsin dataset provided with separate training, validation and test set. The datasets contain no header since we are using unsupervised learning algorithm which do not need label. The last column diagnosis or class is removed. So, the problem is how we choose the number of clusters. Centroid of a data is the average or mean of the data. For the distance metric, we can use Euclidean distance same as K Nearest Neighbor.

For implementation, we can choose number of iterations to find convergence and choose number of k clusters. the centroids is also randomly chosen from the data. Randomly choosing centroids is not efficient and there are techniques to solve this problem such as K-means++. K-means++ initializaition is guaranteed to find a solution that is O(log k) competitive to the optimal k-means solution. But I am using random initialization to start with. Euclidean distance is used to calculate distance from each point to centroids. Variance is used as a measure of cluster scatter. We store the regrouped data points based on cluster index and compute mean or average of clusters and assign a new centroids. So, this continues until we achieve convergence at some point and also depends on iteration limits. The other hyperparameter to be tuned is k. For every data point, its cluster center should be nearest . One common way to find appropriate k is ELBOW method. With scikit-learn, the elbow is around 2 or 3 which can be used as appropriate k.

Finding optimal clustering with given k is NP-hard and k-means is limited to linear cluster boundaries because each iteration of k-means must access every point in the dataset, the algorithm can be relatively slow as the number of sample grows O(n^2).

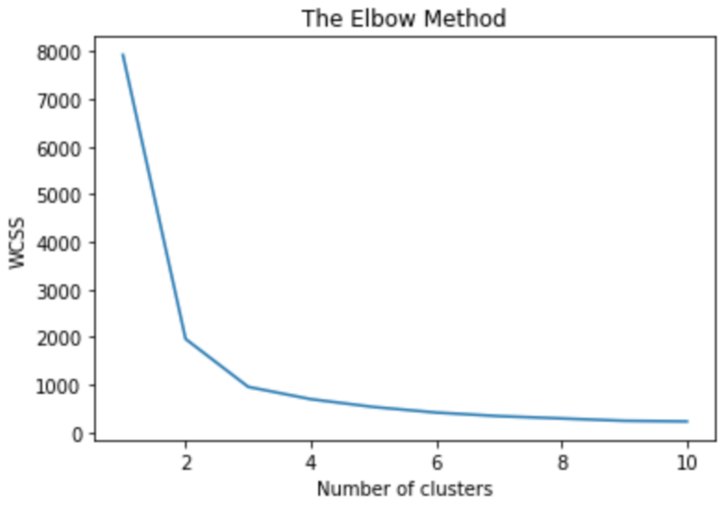


Figure 4: Elbow method

6 Conclusion

The three machine learning algorithms I used for the project served as useful topics to help understand important concepts in supervised and unsupervised learning algorithms. UCI Monk's Problem dataset was used for decision tree algoirthms and UCI Breast Cancer Wisconsin dataset was used for K Nearest Neighbor and K Means algorithms. I managed to research a lot on algorithms and what it serves as a foundational purpose for other high level machine learning algorithms although I did not get the best results for the models and accuracies. Since, it was my first hand work on machine learning algorithms and learning python to building it from scratch, I did struggled but I gained valuable understanding of important concepts which will further help me to my journey to explore greater depth of machine learning field.

References

1. Burkov, Andriy. The Hundred -Page Machine Learning Book. 2019

2. "Ch-3: Decision Tree Learning" Machine Learning, Tom Mitchell, McGraw-Hill. Slides for Instructors. http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/mlbook/ch3.pdf

3. Dua, D. and Graff, C. (2019). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml].Irvine, CA: University of California, School of Information and Computer Science.

4. Duda, Richard et al.Pattern classification,2nd Edition. 2006

5. Hechenbichler, Schliep et al. Weighted K-Nearest-Neighbor Techniques and Ordinal Classification. 13th October 2004.

6. "How to implement the Decision Tree Algorithm from Scratch in Python". November 9, 2016. Jason Brownlee. "Machine Learning Mastery. " https://machinelearningmastery.com/blog/

7. "Implementation of K Nearest Neighbors." MrDupin. https://www.geeksforgeeks.org/implementation-k-nearest-neighbors/

8. "Unsupervised Learning- Clustering" CIS520 Machine Learning — lectures/ Clustering https://alliance.se

9. Witten et al. Data Mining. Practical Machine Learning Tools and Techniques. Fourth Edition. 2017