Ada Ates & Sagana Ondande

CSCI 374 Homework 5: Research Report

ABSTRACT

This project focuses on the problem of comparing and therefore picking the optimal algorithm for various datasets which represent real-world problems that are wished to be automated for people. The solution tested throughout is to evaluate the supervised machine learning models of decision trees, random forests, neural networks, and naive bayes. The results demonstrate that random forests and neural networks are the best options among the models chosen for the given real-world problems whereas decision trees are consistently not a good choice for automating such tasks.

INTRODUCTION

Machine learning has become more prominent over the years as a hope to overcome obstacles in certain tasks that humans hope to solve, such as diagnostics in healthcare, automation, and security. In science, improvement and development are only achieved through repetition, evaluation, and comparison of various models, equations, and theories. As humanity aims to evolve into the next phase in the Information Age, it is important for the same standard to be applied to machine learning. Therefore, the purpose of this project is to experimentally determine the ability of supervised machine learning algorithms in automating tasks for people and compare them to one another.

Aside from evaluating the methods of machine learning and their abilities to automate tasks for people, it is also important to analyze which tasks a specific model can solve. This is a vital step in the development process as it helps to determine which models to focus on improving when aiming for accomplishing a specific task. For example, neural networks might work better for facial recognition whereas naive bayes could work better for medical diagnosis. Such distinctions could be made by running these programs on different datasets and comparing their performance. Here, we are evaluating various models on four different datasets to gain an understanding of when one solution approach might be better than another.

To offer a solution to address the problem described above, we evaluated supervised machine learning approaches of decision trees, random forests, neural networks, and naive bayes on four different datasets in order to see how their accuracies, recalls, and confidence intervals change.

The experiments were focused on accuracy values of predictions of all models across each dataset. The recall values of each label in the dataset were also recorded to check whether a model is producing consistent predictions on an individual level. It was seen that random forests and neural networks provided the best, decision trees the worst, and naive bayes the mid-range of values.

PROBLEM

In this project, the problem we are trying to investigate is how naive bayes, decision trees, random forests, and neural networks learn to predict when given instances from real-world data. Our real-world data consists of four different datasets: monks1.csv, mnist_1000.csv, hypothyroid, and votes.csv. Each dataset represents a different real-world problem.

To begin with, monks1.csv is a dataset with nominal attributes and a binary label, which describes two classes of robots. To determine the label, the rule of this data set is to check if head shape equals body shape or if jacket color is red, which should produce yes as the prediction whereas if these conditions are not satisfied, it should predict no. This dataset is an important one to use in this project as the data comes from one of the first challenges addressed in machine learning. It is a good measure to see how well the new models perform on it.

Second dataset is called MNIST_1000.csv, which consists of 1000 randomly selected optical character recognition of numeric digits from images from the overall MNIST dataset. The instances in this dataset represent different grayscale 28x28 pixel images of handwritten numeric digits. In addition, the labels are the attributes in the dataset are ordinal but they were treated as continuous and represent the intensity values of the 784 pixels. This dataset is crucial to include as it was a benchmark in machine learning because it was the first dataset created for ML algorithms to work on understanding and generating from handwriting, and therefore, it provides a good basis for experiments.

The third dataset we used is called votes.csv, which consists of the voting histories of members of the U.S. Congress. The usage of this dataset is to predict a member's political party based on their previous votes. The attributes in this dataset are nominal; however, some values are missing. This dataset was chosen to include because in real life, there has been more and more research going into political parties and voting patterns to understand the general stance of a given country.

The fourth and final dataset used is called hypothyroid.csv, which consists of patient health data with a combination of nominal and continuous attributes that are used to diagnose the health of a patient's thyroid into four possible labels.

These four datasets were fed into all of our models: decision trees, random forests, neural networks, and naive bayes. Here is a table below that summarizes the number of attributes, the number of possible labels and the set proportion of each label in each dataset.

Dataset	# of Attributes	# of Possible Labels	Proportion of Labels
monks1.csv	432	2	'yes' = 0.5 'no' = 0.5
MNIST_100.csv	10000	10	'zero' = 0.1 'one' = 0.1 'two' = 0.1 'three' = 0.1 'four' = 0.1 'five' = 0.1 'six' = 0.1 'seven' = 0.1 'eight' = 0.1 'nine' = 0.1
votes.csv	435	2	'democrat' = 0.62 'republican' = 0.38
hypothyroid.csv	3772	3	'negative' = 0.92 'compensated_hypothyroid' = 0.05 'primary_hypothyroid' = 0.03

Table 1. *A summary of the statistics for each data set, the number of attributes, the number of possible labels, and the proportion of each label in the data set.*

SOLUTION

Our defined problem in this project is to experiment with various machine learning algorithms training on different real-world problem data and to compare their performance to one another. To solve this problem, we decided to use four supervised machine learning approaches, which are decision trees, random forests, neural networks, and naive bayes. Decision trees were chosen as a model because in theory, it is expected to work very well, especially with binary classification. However, real world data tends to be more complicated than binary classification or might be missing some data; therefore, we thought decision trees are a perfect choice to illustrate the difference between theoretical and practical reality of machine learning. Secondly, we chose random forests as it was created as an improved version of the idea behind decision trees. Thirdly, neural networks were chosen as they are quite prominent in the ML world as they stem from a theory of how the human brain works. They were proven to be a quite accurate depiction of human learning that other models such as recurrent neural networks and convolutional neural networks were built upon. If the goal is for machines to try to replicate human learning and ideally be better at solving real world problems, then this model is the on-point approach to test these expectations. The final approach chosen was naive bayes, it was chosen mainly to illustrate the power of decisions based on probability.

To implement our solution and each approach, we first preprocessed our data by creating a dataframe using pandas, then determining if the values are nominal, and applying hot one-coding. Then, we used scikit-learn python library, which has each of the models pre-implemented. Each implementation was then run on each of the datasets, their accuracy and recall values of each label in respective datasets were recorded. Further statistical analysis was done on accuracies to answer our problem.

EXPERIMENTAL SETUP

The performance measures that were used for these experiments were to calculate the accuracy of the model through calculating the sum of the correct predictions divided by the total number of predictions made. This can bring some insight into the general accuracy of the model as how it does with multiple labels and how well they predict the values across the input dataset. Alongside this, we calculate the recall of each label in the input dataset as we check how well for each individual label is predicted properly. This is calculated by taking the number of correct predictions divided by the sum of all predictions that included that label. This can tell us how well we predict a given label in a dataset correctly.

For our experiments, we utilized the sci-kit learn python library, pandas dataframes, and numpy to help us with the data preprocessing, model creation, training, testing, and calculation of metrics. The training set percentage was 80% and testing was 20% of the dataset.

RESULTS

The results of our experiments aligned with our expectations. Decision trees performed worse than other machine learning approaches when the values were numerical. Their accuracy was 18.5% for the mnist_1000.csv dataset where all the values were numerical whereas the rest of the approaches were at least 82% accurate. These values show a big gap between the decision trees and the other approaches, which make sense as decision trees perform better with nominal values. It also was the worst performer with the dataset monks1.csv with an accuracy of 72%. As seen in tables 2 and 3, the recall values of labels in the monks1.csv and mnist_1000.csv had very low or inconsistent values when they were used to train decision trees. The decision tree produced a recall value of 0.48 for label ‘yes’ in monks1.csv whereas it consistently produced a recall value of 0.0 for several labels in mnist_1000.csv. These results suggest that decision trees are not a good choice for datasets with numerical values or classification with more than 2 labels for each attribute. This should be taken into account when one chooses a model for a real-world problem.

Dataset: monks1.csv	**Decision Trees**	**Random Forests**	**Neural Networks**	**Naive Bayes**
Accuracy	0.723076923076923	0.89615384615384	1.0	0.74230769230769
Confidence Intervals	[0.66091375077292, 0.785240095380922]	[0.8537751257871, 0.93853256652]	[1.0, 1.0]	[0.6815496107898, 0.8030657738255]
Labels	**Recall Values**			
'no'	1.0	1.0	1.0	1.0
'yes'	0.478260869565217	0.80434782608695	1.0	0.51449275362318

Table 2. *This table illustrates the accuracy and confidence intervals of monks1.csv dataset across all models implemented, as well as the recall values for each label in the dataset.*

Secondly, naive bayes does not seem to be the best choice for these data sets either. After decision trees, naive bayes was the worst performer in monks1.csv and mnist_1000.csv whereas it was the primary worst performer in hypothyroid.csv and votes.csv. Although this is the case, its accuracy was pretty consistent across all datasets with a range of 72-90% as observed across tables 2-5, and produced overall consistent recall values for all labels in the datasets. This observation suggests that although naive bayes might not produce the best result, it is a good bet as a choice as it will be consistent with its prediction accuracy. For instance, if there is not much information on a given dataset, naive bayes might be a good option as it produces good results across varying datasets regardless of nominal or numerical values, or abundance missing information.

Dataset: mnist_1000.csv	**Decision Trees**	**Random Forests**	**Neural Networks**	**Naive Bayes**
Accuracy	0.1845	0.934	0.87083333333333	0.824833333333333
Confidence Intervals	[0.173282849131, 0.195717150868]	[0.926820109861, 0.9411798901384]	[0.8611345972568, 0.88053206940982]	[0.81384121923613, 0.83582544743053]
Labels	**Recall Values**			
'eight'	0.0	0.9038142620232	0.79601990049751	0.784411276948590
'five'	0.0	0.9378238341968	0.85837651122625	0.675302245250431
'four'	0.0	0.9147540983606	0.85245901639344	0.734426229508196
'nine'	0.9719471947194	0.9026402640264	0.79537953795379	0.834983498349835
'one'	0.0	0.9729729729729	0.95777027027027	0.935810810810810
'seven'	0.0	0.9471074380165	0.89256198347107	0.824793388429752
'six'	0.0	0.9542483660130	0.90522875816993	0.918300653594771
‘three’	0.0	0.9126853377265	0.83196046128500	0.830313014827018
‘two’	0.0	0.9277108433734	0.88640275387263	0.836488812392426
‘zero’	0.8561983471074	0.9669421487603	0.93388429752066	0.869421487603305

Table 3. *This table illustrates the accuracy and confidence intervals of mnist_1000.csv dataset across all models implemented, as well as the recall values for each label in the dataset.*

On the other hand, random forests and neural networks produced the overall best results across all datasets, which suggests that these approaches also perform well regardless of missing information and nominal vs. numerical values. Random forests were the best performer, as seen from tables 3-5, it had the best accuracy and confidence intervals as its accuracy range was 93-99%. Yet, neural networks performed better on monks1.csv with an accuracy of 100% as seen in table 1.

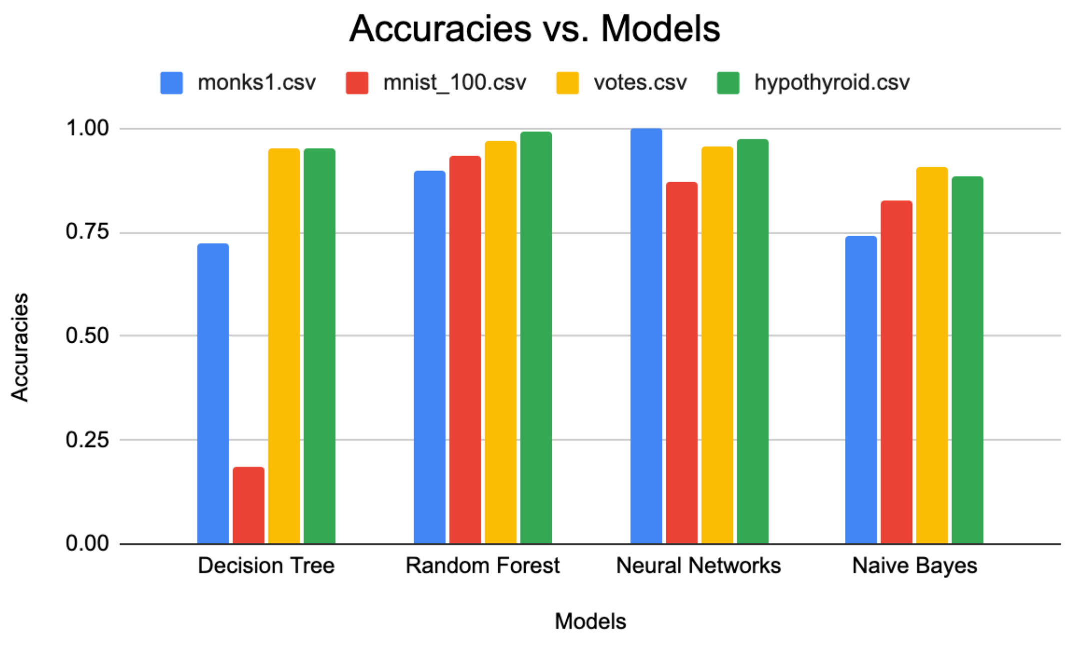
Neural networks also provided the second best accuracies and confidence intervals on the rest of the datasets which can be observed in tables 3-5, its accuracy range was 87-100%. Neural networks gave an accuracy of 100% with monks1.csv, which suggests that this approach performs the best when there are multiple nominal labels for an attribute. On the other hand, although its recall value for the label 'compensated_hypothyroid' was 0.66, overall recall values for labels were very consistent and in the 0.80-1.00 range. Recall values of labels produced by random forests were more consistent and precise than those of neural networks, meaning that although neural networks sometimes gave the top recall value for a certain label of a dataset, it also had big gaps between recall values of the rest of the labels of the same dataset. However, this was not observed for random forests. Overall, random forests performed the most consistent and the best by values for accuracy and recall values.

Dataset: votes.csv	**Decision Trees**	**Random Forests**	**Neural Networks**	**Naive Bayes**
Accuracy	0.9501915708812	0.9693486590038	0.95785440613026	0.90804597701149
Confidence Intervals	[0.920027859689, 0.9803552820725]	[0.945448955597, 0.9932483624098]	[0.9299961707032, 0.9857126415572]	[0.8679808013510, 0.94811115267192]
Labels	**Recall Values**			
'democrat'	0.9177215189873	0.9746835443037	0.96835443037974	0.89240506329113
'republican'	1.0	0.96116504854368	0.94174757281553	0.93203883495145

Table 4. *This table illustrates the accuracy and confidence intervals of votes.csv dataset across all models implemented, as well as the recall values for each label in the dataset.*

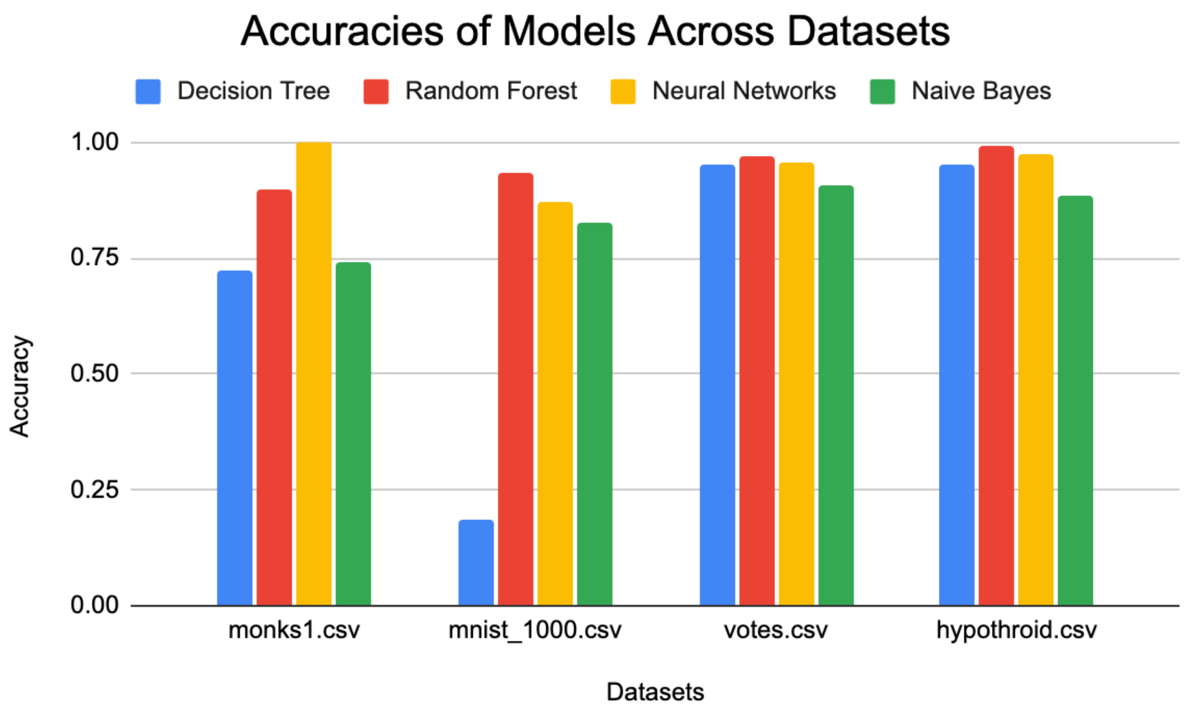
Dataset: hypothyroid.csv	**Decision Trees**	**Random Forests**	**Neural Networks**	**Naive Bayes**
Accuracy	0.953180212014	0.99072438162	0.972614840989	0.8851590106007
Confidence Intervals	[0.94323504599, 0.963125378032]	[0.9862114612, 0.99523730197]	[0.96493171620, 0.98029796577]	[0.870149417563, 0.900168603637]
Labels		**Recall**	**Values**	
‘compensated_hypothyroid’	0.965217391304	0.93913043478	0.660869565217	0.7739130434782
‘negative’	0.976622137404	0.99666030534	0.992366412213	0.8959923664122
‘primary _hypothyroid’	0.0	0.86792452830	0.867924528301	0.6981132075471

Table 5. *This table illustrates the accuracy and confidence intervals of mnist_1000.csv dataset across all models implemented, as well as the recall values for each label in the dataset.*



Graph 1. *An illustration of accuracy values of implemented machine learning approaches across various datasets that represent real-world problems.*

As explained in the paragraphs above, random forests overall presented more consistent accuracy and recall values, which can also be observed in graph 1 above. Although neural networks provided the best accuracy and recall values for monks1.csv, its values for other datasets varied more often than random forests. On the other hand, naive bayes couldn't reach the level of neural networks and random forests; yet, its results were pretty precise across all datasets even though not as accurate as wished. Finally, decision trees were the most variant approach of all, it performed significantly worse for mnist_100.csv where the values were numerical. It also performed below the usual 75% range for monks1.csv, which we interpreted as not working well with classification problems with several labels for attributes. In sum, random forests seem to compare the best to other approaches but neural networks also generally perform well.



Graph 2. *An illustration of accuracy values of implemented machine learning approaches according to every dataset that represents real-world problems, and which model is the best for a certain type of dataset.*

As seen in graph 2 below and based on the results presented above, certain models performed best for specific datasets. For monks1.csv, the neural network model was the best as it produced an accuracy of 100%, which to reiterate, shows us that it works especially well with datasets that have multiple classification. However, random forests performed best for the rest of the datasets. Decision trees performed overall the worst but it took a dip especially with the mnist_100.csv dataset, which made us interpret that it doesn't perform well on datasets with numerical values. Finally, naive bayes performed very consistently across all datasets even though it failed to produce the best results.

CONCLUSION & FUTURE WORK

The aim of this project was to find a solution to the problem of making a wise choice for a machine learning algorithm for different real-world problems and their respective data for automation. We implemented four machine learning approaches as a solution to this problem as well as to observe how they compare and contrast in varying scenarios. These approaches were decision trees, random forests, neural networks, and naive bayes. Our findings shows that while decision trees produced the worst results generally, random forests produced the best and most consistent results across chosen datasets. Neural networks competed well with random forests whereas naive bayes were in the mid-range of results but always consistent. Considering that the datasets had missing information and consisted of various types of data inside, these results are valuable as they come from a good range of data indicating distinct real-world problems. Overall, it can be concluded that decision trees are not a good option for automation of such tasks whereas random forests and neural networks are well suited for automation of these tasks.

For future work, datasets that involve more complex imagery, have varying types of data within, or lack information in general would be a good starting point to explore machine learning algorithms further. Another option would be to try other algorithms such as k-nearest neighbors or logistic regression. Also, hyperparameters could be played around with to see if each model has an optimal point in each type of dataset. These would provide more information on our given problem, which is essentially how to choose the best machine learning algorithm.