3.5 Learning Response: Comparing Algorithms

Jason “Scott” Person

20250422

1. ***From Jedamski, Defining model vs. Algorithm:*** Explain and clarify the definitions of and differences between “algorithm” and “model.”

An algorithm is a technique or equation outline. A model is an actual implementation of an algorithm with specific parameter values that is created by learning from data.

Some examples of algorithms (and models):

* Linear Regression (a line that defines the heigh and weight of students: Weight (lbs) = -220.5 + (6.16 \* Height (in) )
* Decision Tree (sequence of binary questions that determine likelihood of loan prepayment)
* Logistic Regression (The equationA mathematical equation with numbers and symbols

  Description automatically generatedthat provides a probability of passing an exam based on number of hours studied)

In these examples the algorithm defines the technique or type of process/equation that will be used. The model is the outcome of the machine learning and is an actual implementation.

1. ***From Doug Rose’s assigned video on Linear Regression:*** Define and describe **linear regression**.
   1. Summarize it with an easy-to-understand sentence or two, and illustrate how it works.
   2. What is the “line of best fit”? And what are two common synonyms?
   3. What problem do outliers create for linear regression prediction?
   4. What kind of prediction is linear regression used for? (Hint: It’s not binary classification.)

Linear regression creates a best-fit line function that models the data as closely as possible. It works best when there is a continuous relationship between the predictor and the outcome.

The line of best fit is the line through the data that minimizes the aggregate error calculated as the sum of the distances between each point and the line. Two common synonyms for this line are the hyperplane and trendline. In Figure 1 below you can see the trendline representing a model predicted ice-cream sales based on temperature. This figure also shows the model being used to predict sales on a 95-degree day.

A graph of sales and sales

Description automatically generated

Figure 1: Linear Regression (Rose)

Figure 1 shows a couple of outliers in the data. One with low sales at a temperature of around 77 degrees and another day with high sales at 95 degrees. While these don’t impact the models significantly in this case, outliers can potentially distort the model leading to inaccurate results – this is one reason we validate and test the models.

Linear regression is best used to predict continuous numeric values where there is generally a linear relationship between the predictor and the outcome.

1. ***From Jedamski:*** Define and describe **logistic regression**.
   1. Summarize it with an easy-to-understand sentence or two, and illustrate how it works.
   2. Include an explanation of the difference between *linear regression* and logistic regression — the way they work, and the kind of problem for which each is best designed. (Hint: One of these is designed for binary classification; the other is not.)
   3. What are its strengths?
   4. When should it be used?
   5. What shape of data works best with logistic regression?
   6. What are its weaknesses, and when should it *not* be used?

Logistic regression models binary data with a function called an S-curve otherwise known as sigmoid. This is shown below in Figure 2. The significance of the S-curve is that for most values of X it returns a value of 0 or value 1. This makes it appropriate for modeling data that is of similar shape. Binary data is not continuous – there is a big gap between 0 and 1 but due to the shape of the logistic curve we can model this discontinuous data using a continuous function. Linear regression (also shown in Figure 2) is better for modeling continuous data where the values for the dependent variable are distributed across the Y axis.

Logistic regression works well for binary target values, when a clear model is important, for well-behaved data without a lot of complex relationships or outliers, and when a basic model is needed to set a benchmark for baseline performance.

Logistic regression should not be used when the data is short and fat or tall and skinny. Avoid the extremes.

A two graphs of equations

Description automatically generated with medium confidence

Figure 2: Linear vs. Logistic Regression (Rose)

1. ***From Doug Rose’s assigned video on Decision Trees:*** Define and describe Decision Trees.
   1. Summarize it with an easy-to-understand sentence or two, and illustrate how it works.
   2. Briefly explain and illustrate these key terms:
      1. Split
      2. Root node
      3. Decision node
      4. Leaf nodes

Decision trees set up a sequence of binary decisions to model binary data. The values of one or more predictors define the path through a tree of decisions the result of which is a percentage that the outcome is true. Figure 3 below illustrates an outline of a decision tree.

Split: This is where the dataset is divided into two or more groups based on the input features. A split occurs at a node.

Root node: the “trunk” of the tree and represents the entire dataset before any splits.

Decision node: this is the point where the values of the predictor determine the path through the tree.

Leaf node: the final node for any path through the decision tree. This node represents the prediction for the given predictors.

A diagram of a tree

Description automatically generated

Figure 3: Decision Tree

1. ***From Jedamski:*** Define and describe **Random Forest**.
   1. Summarize it with an easy-to-understand sentence or two, and illustrate how it works.
   2. Briefly explain ensemble methods — and by inference, explain why a random forest model frequently outperforms a decision tree model.
   3. What is the purpose and role of multiple decision trees in a random forest model?
   4. By what means does the Random Forest model combine the predictions of (or “learn from”) the multiple decision trees?
   5. What are its strengths?
   6. When should it be used?
   7. What are its weaknesses, and when should it *not* be used?

Random Forest is a collection or “ensemble” of n independent decision trees created by sampling the data (rows and features) n times and training each model on a sample (see Figure 4). Predictions from all these models are aggregated into a single prediction. It is more resistant to outliers and overfitting than any single model. Figure 5 below shows how multiple decision trees are tested and can have different predictions, which are combined by voting. This is shown with test data from the Titanic dataset; prediction is similar (Figure 5) except that each model gets the same input. Each decision tree uses its own subset of features to arrive at the prediction.

A diagram of a training data

Description automatically generated

Figure 4: Random Forest Training (Jedamski)

A diagram of a test data

Description automatically generated

Figure 5: Random Forest Testing (Jedamski)

Random Forests work well for categorical or continuous target variables with messy data (bad/missing values). As with logistic regression, random forests are great for arriving at a quick, benchmark model.

Avoid random forests when you need the absolute best possible model or need a model that is transparent. The random nature of the training set selection both with respect to rows and features makes the model less intuitive. Also, while the model is relatively quick and easy to train, its prediction time is high because of the normally large n value.

1. ***From Jedamski:*** Define and describe **Boosting**.
   1. Summarize it with an easy-to-understand sentence or two, and illustrate how it works.
   2. Why does boosting build on *weak* models?
   3. Explain how boosting uses weak models differently than random forest uses its individual decision trees.
   4. How does “voting” work differently in boosting as opposed to random forest?
   5. What are its strengths?
   6. When should it be used?
   7. What are its weaknesses, and when should it *not* be used?

Boosted trees is like random forest in that it is an ensemble model. With boosting many weak models are trained in sequence based on the outcome of the previous model.

It is surprising that boosting places emphasis on using weak models. Weak models are cheaper to train, allow for the process of boosting (learning from the misclassified examples) to work without model redundancy, and help to minimize overfitting (OpenAI, 2025).

Each model that makes up a random forest is trained separately against its own randomly selected (with replacement) training data. Random forest models do not impact each other. With boosting we take the output of one model, overweight the misclassified data, and use that as the input for training the next model (See Figure 6 below). This allows each model to learn from the mistakes of the previous models.

A diagram of a training data

Description automatically generated

Figure 6: Boosting Training (Jedamski)

Random forest uses a straight vote with each model being equally weighted to determine the outcome. Boosting uses weighted voting (Figure 7) where the models that performed better in training have a higher weight (more influence) on the voting outcome than poorer performing models.

A diagram of a test data

Description automatically generated

Figure 7: Boosting Testing and Implementation (Jedamski)

Boosted trees is particularly time efficient when predicting because all the models can be executed in parallel – and they are simple models. Boosting is also quite flexible. It works for both categorical and continuous predictions.

Avoid boosting when an in-depth understanding of the specific model is important. Also, boosting can be compute and time intensive during training as the component models are trained sequentially. Finally, boosting tends to be sensitive to overfitting so it is not a particularly good model for using with noisy data.

1. ***From Jedamski, Summary:*** Answer the question: ***Why do you need to consider so many different models?***
   1. Jedamski references Khabaza’s “No Free Lunch Theorem.” What is the relevance of that theorem to this question?
   2. What are “the questions that we need to be asking”?
   3. Even after narrowing down our algorithm options based on strengths and weaknesses, why do we often still wind up with 3-4 algorithms to test?

The free lunch theorem: “No algorithm works best for every problem”. Each algorithm has its strengths – different capabilities that make it appropriate to solve specific types of problems given the associated constraints. There are compromises associated with each algorithm. After asking the questions below and narrowing down the options, we are often still left with 3-4 options. At this point we need to test the applicability of the model to our specific requirements and data. The tradeoffs can be situational where there is no perfect answer.

Questions to ask:

* How efficient is the algorithm during training?
* How efficient is the algorithm when making predictions?
* What is the shape and makeup of the data?
* What are the relationships within the data?
* How complex is the data?

(Jedamski)

1. ***From Jedamski, Summary, Conceptual Comparison:*** Screenshot and paste Jedamski’s table summarizing the strengths and weaknesses of the algorithms he considers.  
   Then, below the table, answer:
   1. What is Jedamski’s overall favorite algorithm in terms of performance? Why is this so?
   2. What are the next favorites in terms of performance?
   3. Which algorithm excels with “limited” or “small” data — and data with few examples and many features?

A table of speed and speed

Description automatically generated

Figure 8: Machine Learning Algorithm Comparison

Jedamski’s favorite model overall is boosted trees because of its accuracy performance and flexibility to solve many different types of problems.

Multilayer perception and random forest techniques are also noted as being favorites by Jedamski.

For limited data with lots of features, support vector machines is a strong algorithm choice.

References:

OpenAI. (2025, April 23). *Why does boosting use weak models?* [Large language model response]. ChatGPT. https://chat.openai.com/