The goal of this project is to enable the capture, manipulation and grading of decision trees in support of classroom and course needs.

General Guidelines from Herrmann

* Preferred implementation
  + Website that has compatibility with phones
* Standards
  + Boxes for decisions, circles for uncertain events
  + Both edges and nodes should have text
  + Triangles are results
* Interface
  + Quizlet
  + Instructor has an account where they can upload problems and solutions
  + Students have accounts where they can see the problem, submit, get instant feedback
  + Trees and results are saved so that the instructor can review them and see how students are doing
* Weblinks provided
  + <https://github.com/SilverDecisions/SilverDecisions/wiki/Gallery>
    - Software for creating and analyzing decision trees
  + <http://www.public.asu.edu/~kirkwood/DAStuff/decisiontrees/index.html>
    - Presents methods for analyzing decision trees
  + <https://www.palisade.com/precisiontree/>
    - Quantitative decision analysis in Microsoft Excel

**Feasibility Report - Decision Trees**

**Team:** Ameen Ateeqi, Dustin Brashear, Lucas Cheng, David Kaplan, Guanhua Liu, Yoni Margulies, Greg Segala, Hugo Wang

**Client:** Dr. Jeffery Herrmann

**Table of Contents:**

**Clients Needs**

Our task is to create an application for Dr. Herrmann to ease creation, manipulation and grading of decision trees - as these activities typically require communication through hand drawn graphic interface. Current hand writing methods can be confusing if the student makes an unclear drawing, and grading must be done manually by hand.

**Solution Proposal**

For the decision trees project, we are building a platform for the professor Jeffrey Herrmann so that he can streamline the process of grading decision trees. To get a better idea of what Dr. Herrmann wants us to create, we contacted him and asked him his ideas about the project. Based on the input from Dr. Herrmann and discussing what is feasible to create in 8 weeks, our group is going to build a website where instructors can submit decision tree assignments and then students can submit their answers and receive grading on their work.

There are three distinct sections of the project: The user interface, the logic of grading the decision trees, and the website itself (database, pages to create and view assignments, accounts, etc.).

In order to ensure ease of use, we acknowledge the need for a streamlined front end, and while we still are planning what this will exactly entail, we imagine a click and drag diagram for nodes, which the user can add as many edges as the user wants from that node. This allows both the teacher to build the correct answer template, and provides the student with easy access to create their own answer as well. After storing both the teacher’s template and recording the student’s answers we’ll analyze each student’s reply with the correct answer and judge accordingly through an comparison algorithm. This algorithm will need to take into account synonyms, misspellings, and incorrect relationships - and will output comments on any errors that it judges to exist upon a “layer” above the tree that the student submitted (after submission the tree effectively becomes inactive) - which also can allow the teacher to submit any additional comments beyond the program checking for incorrect answers

**The user interface**

Our goal for this project is to provide an easier way to capture decision trees, and enable an instructor to characterize rules for recognizing valid trees. With the help of our program, users can visually map out, organize, and analyze decisions using decision trees. Decision trees are quantitative diagrams with nodes and branches representing different possible decision paths and chance events. This helps users identify and calculate the value of all possible alternatives, so users can choose the best option with confidence. Furthermore, we also introduce a way to compare and analyze two different trees by splitting structures and presenting a clear way of visualizing similarity and difference. For this reason, when we design our interface we should follow these three rules: simplicity, security, and intelligence. We can approach this question by splitting our system into four different stages.

**1. Login interface**

Upon following a link to our website, users will be prompted to login with via a CAS authentication process. After the user verifies his or her identity, this user will be able to use their according student or instructor account.

**2. Submission interface**

Following login authentication, students will be able to view a list of assignments posted by the instructor. By clicking on one of the assignments, the student will be able to view any saved work, edit their saved work, submit their saved work to be graded, or delete the work that have already been submitted. If a student wishes to create or work on a new decision tree for a given assignment, they will be directed to an interactive click and drag tree construction webpage. Meanwhile, the instructor will be able to create new assignments. He or she will be able to submit a rubric document and create a tree (using the same interactive decision tree constructor) that will be used to grade the students submissions.

**3. Analysis interface**

The main part for this interface is to and compare and contrast the outcomes between the teacher’s decision tree and the student’s answer. In this stage, the instructor will be able to see a side by side display of his own decision tree and each student’s answer. Our program ideally would be able to highlight differences, and allow any commentary on the student’s decision graph as a separate “layer” above the student’s decision tree.

**4.Grading interface**

After the instructor grades student work we immediately apply changes to student side view and update the grade result in our database. We allow the instructor to write down comments and will send this feedback to students in their webpage view. The instructor also has the option to regrade the changes he makes, and can receive messages from the students about the assignment. Specifics on grading are still being resolved, and we anticipate this as a potential risk below.

**How it Solves the Problem**

This software will solve the problem by providing a quick and easy way for the professor to create, assign, grade, and manage his assignments. Our program also provides a simple and efficient way for students to see, execute, and submit their assignments, while providing quick and accurate grading. This eliminates the use of paper assignments, and having the professor grade each assignment by hand, ultimately improving the time spent on this problem.

Specifically, our processing tool is one that we will use to provide a visual frontend aspect to the project, which is something that would otherwise be done by hand. Even drawing decision trees by hand can be time consuming and unclear depending on how complex the tree is. Using this tool will give the user a standard way to draw decision trees to simplify both the process in creating a decision tree as well as eliminating any confusion within the decision tree itself - something that can very easily happen if the decision tree was drawn by hand.

In addition to improving grading time, simplifying the creation of a decision tree, and minimizing confusion - our application has the added bonus of being both fully accessible at all times to both students and teachers while being stored in a convenient location. Any correction made to a problem after being graded will always be there, so the student can always access any decision tree to study it further in preparation for future exams.

**Possible timeline**

The first part of the project that we would need to complete would be establishing the XML tree definitions for representing a decision tree, which should take no more than a day. After that, we would set up a database to store user submissions and professor provided trees to compare them to, which should be about a week to setup and configure. We would also need a front end web page where students and professors can authenticate using CAS, which would take a week or two to design and code.

We would need to determine how to grade trees and display said grades to the students. Ideally, this would involve communication with Dr. Herrmann as to how exact he wants the grade to be (either a pass/fail or a letter or points system). This step of the timeline will take another week to complete, but can run concurrently as to while we are still coding the initial versions of our website. We would also need to do research on how to acknowledge misspellings, synonyms and incorporate how to account for alternative structure of trees if a prior assumption was wrong. This we anticipate will be the most time consuming on the research side, and allot about two to three weeks to ensure that we are able to satisfy all grading criteria and are able to capture a majority of student’s answers correctly.

Finally, we believe the majority of the work will be required to actually record and interpret a decision tree and translate it into XML. For creating trees on the website, it will take two to three weeks to code up and thoroughly test a solution involving the Processing language. Afterwards, based on our research in how to grade decision trees we will need to implement our research into code, which will have to do only after we have finished that research. We anticipate, due to the difficulty of this specific topic that this could take an additional one to two weeks as well (although we intend to start the research of this immediately).

Overall, this timeline starts with research and coding a skeleton of our website. We then work on the frontend graphics to make user interaction as simple as possible while we continue talking with Herrmann on making all grading criteria specified (along with research on recognizing similar or synonymous trees). Afterwards, this leaves only the coding of grading and comparing decision trees. Ultimately, this all leaves one to two weeks as a buffer we can use to refine our project, conduct additional tests, and address unanticipated problems and changes to the requirements.

**Possible Issues and Resolutions**

Currently, one of the major risks that we face is our research and ability to quickly decide on how to integrate what we know into coding. As such we plan to (and have already) started to do research into how to solve our various problems of storing decision trees, what symbols to include for what element within the decision tree, and particularly how to grade a student’s decision tree. As stated, to counteract the time spent on this, we both have gotten in contact with Herrmann and are active both on Slack and have met in order to quickly decide at least where to take the research we have done.

As stated, a main risk involves accurate grading of the student submitted decision tree when compared to a canonical solution. There will have to be clearly established criteria so that there is clarity when a student receives a grade so that they understand where points were lost, and areas that contributed to loss of points should be highlighted. This can be refined by discussion with the client on how their grading currently judges differences between trees and adapting our grading schema accordingly. In addition, grading can be influenced by incorrect spelling, synonymous terms, and decision trees influenced by incorrect assumptions. In order to make sure no unnecessary points are lost, when we compare we also have to define what constitutes as being within the correct solution.

Another risk we may face is the possible miscommunication about intended functionality of the end product with the client. If the team and the client are not on the same page, then we may end up developing software that does not align with the client’s vision, ending up in a failed delivery. Clear communication about the development process and product features with the client is important to prevent this - meaning we must regularly keep in contact with Herrmann at every step of the project (yet also keep in mind to not be too overbearing).

The last risk lies in how our logic reasons in specifically giving points off. It could be easy if the program just said 25/30 based on the percent difference between the two trees, but Dr. Herrmann might actually want a program that takes into consideration points for each concept instead of each element. Figuring out how to incorporate that into the product will likely be an issue.

**Required Resources**

The main resource that will be required for this project is time. Besides the obvious time required for coding, we will also need time for group meetings. This will require about a few hours per week during early evening hours, in the first half of the project and doubled in the second half of the projects. This is because of the increase from weekly to bi-weekly meetings. Another resource that we will need for these meetings is a location. Having our meetings during school hours will allow us to use the engineering library to meet, eliminating the need for reserving rooms.

Baring time resources aside, we still are on the process of deciding on the framework that we’re deciding to use. For backend, we anticipate using an SQL database where we store the tree in XML form, meaning we prefer to use any full stack framework as it already integrates front-end and back-end capabilities, allowing us as a group to easily connect separate parts (the UI and actual storage of the tree). It also allows us to build custom classes/data-structures to aid in the storage of our decision tree with whatever full stack framework language we eventually decide to use (eg. Ruby, python, javascript).

In terms of research resources, we still are trying to find any existing projects or similar products that we can use as guidelines in our own development of this decision tree grading application. While this is ongoing, we already have quite some valuable resources (which are attached in our group SVN as .pdf files) sent by Dr. Herrmann. We anticipate using his c05.pdf file heavily as these 41 pages are the basis of the material Herrmann covers and thus will be incorporating our modelling based on the paper. In addition, Herrmann gave us another paper (Smith IEEE 2013.pdf) which while shows grading schemes that go beyond decision trees gives good concepts - such as the idea of MMU (Minimum meaningful units) where we can break the decision tree down into their individual smallest components and start from there. It also delves into assigning synonyms to words, and talks about comparison algorithms in general. While we anticipate incorporating more research documents beyond these two, we feel that these are a great resource to start in helping our problem in how to grade decision trees.

**Workflow Proposal**

Our process to produce this product will involve clear lines of communication, and assistance by all group members. This will involve a Slack channel and weekly meetings. The Slack channel will be our primary research sharing channel as well - as we have dedicated groups to look into solutions for the various problems within our product (eg. grading), and will cover briefly as a part of our weekly meeting. We will break down all tasks we need to do and spread it out over the eight week timeline we have. We can achieve this by working in an agile format, using eight one week sprints. In addition to a weekly meeting, we plan on getting in touch with our client Dr. Herrmann at least once a week - if not in person at least through email.

Towards the last weeks of the project, we also plan on increasing the frequency of our weekly meetings to bi-weekly meetings. This will allow for better communication and ensure that everyone is on schedule for the application’s deployment.

**Addendum: Alternative Computer Vision Approach**

Another additional approach that can potentially be pursued in parallel to the other implementation is to use computer vision to interpret pictures of decision trees. This implementation of the website would be a tool that Dr. Herrmann mainly uses to convert physically drawn decision trees into a digital copy. Eventually, this tool would serve to compare and grade the correctness of even physically drawn decision trees between the students and Dr. Herrmann’s answer.

This path involves scanning an existing student’s answer on paper and then converting this image into the backend storage. The software would then interpret, compare, and then grade the student’s answer against Dr. Herrmann’s answer, which can be created digitally or scanned as well. There are a couple of parts to this computer vision:

* Identifying the shapes, words, and relational structure of the decision tree
* Creating relationships between the different node objects by just looking at directions
* If there are words inside the decision tree then reading the words

Once the logical structure of the decision tree is created, comparing and grading the decision tree is the same, and we anticipate integration with our main program to be relatively easy.

The biggest risk to this approach is that the translation using machine learning and computer vision from image to XML file has to be very consistent and very accurate. We must make absolutely sure that any training we give this part of the program take into account for all situation and make it explicit when the program cannot run properly (eg. We only train this decision tree recognition software on white computer paper, but it ends up trying to read from college-ruled lined paper). If the interpretation of the tree is not accurate then the grading will not be accurate. A way to mitigate this risk would be to provide a gauge that tells Dr. Herrmann how confident the software is in the conversion and then the interpretation and the grade. If the threshold confidence is not met, then Dr. Herrmann would have to grade the decision tree by hand. If this product could drastically but not absolutely eliminate grading by hand, the project would be a success.

Since this implementation has a lot of technical difficulties, it is going to be pursued as a side, parallel project to the other implementation. If this computer vision implementation works, then we anticipate shifting more resources to this approach. However, a lot of research first has to be done in order to figure out how this will work. This computer imaging approach is preferable over our main approach because it dramatically reduces cost to students for tree capture than with the other approach - and allows students to continue with their existing workflow with only the added step of scanning and submitting it online. The only change that might occur is that they might have to do their trees on white printer paper.

1. what you would potentially build as a solution;
   1. We would build a website where an instructor can submit assignments with correct solutions, students would be able to submit their answers. The software would then compare the answer and the
   2. Have the ability to highlight divergent areas or where the solutions begin to diverge
   3. Pages
      1. Students and teacher to build trees
      2. Students to see grades?
      3. Teacher to see students grades?
      4. Login page
      5. Teacher pages
         1. Creating assignments
         2. Creating answers (part i)
         3. Ability to add comments/highlights
2. how an objective observer would recognize it solves the problem;
   1. Provides a grade that corresponds to how correct a decision tree is
   2. Easy to build visual representation of a decision tree
      1. Both for students and teachers
3. a likely timeline for the major mileposts along the way (and what they are);
   1. Translating tree to XML and back to tree
   2. User building the tree
      1. The way that we accept input
      2. Teacher needs to start from scratch, where does the student start from
   3. Website authentication
      1. Make users and stuff
   4. Database to store the trees/how to store the trees
   5. Comparing two trees
      1. How wrong is something
   6. Other aspects about the website
      1. How students see their grades
      2. How teachers see their students grades
      3. How teachers create
4. The chief risks facing you at this point, and what you will do to reduce them;
   1. How would you judge how wrong something is
   2. Miscommunication about product details
5. The resources you anticipate you'll need (and thus put in your proposal);
   1. Web services
      1. “Tree building API” -> so far Processing
   2. Time -> weekly, bi-weekly meetings, meeting rooms
   3. Input from the professor (good line of communication for what he really wants)
   4. Authentication API?
6. The process you will likely follow, and how you will ensure you are following it.
   1. How we check how we are doing in terms of progress
      1. Make timelines
         1. 4 2 week sprints
         2. 8 1 weeks sprints
   2. Agile format
   3. Stand-up meeting weekly
      1. Increased meetings towards end of meeting.