

Brandon Chavez
Build-It Bear Inc.
123 Quite Finite Loop
Codeville, WA 98225

Senior Leadership Personnel @ Fresh Enough Grocery Stores
2520 Vegetate Lane
Schenectady, NY 12345

Dear Senior Leadership Personnel,

It has come to my attention that our company's resolution to exclusively hire employee candidates with severe, uncorrectable near vision impairment has made sorting some of our best-selling products a daunting proposition – in particular, those fruits which have similar weights, textures, and shapes. Oranges and grapefruits are a key example of this.

I recommend that we use a publicly available dataset on key citrus fruit characteristics to create a Machine Learning (ML) algorithm which can easily and quickly sort oranges and grapefruits from each other with at least a 90% success rate. Such an algorithm can then be used in conjunction with simple scales and camera systems to allow employees to rapidly sort incoming produce. This will dramatically reduce customer complaints, with improvement in sales and our overall reputation being sure to follow! I am highly qualified for this undertaking, as an upcoming front end web development intern, and WGU Computer Science graduate. With a budget of \$10,000, I can deliver such a ML algorithm and a UI which can be used to test its classification abilities. Please review the details below.

Kind Regards,

Brandon Chavez

Part A: Project Recommendation

Problem Summary: This project, referred to hereafter as the Citrificate Project (or simply Citrificate for short) will aim to produce a Machine Learning (ML) model capable of reliably and inexpensively discerning between two very similar citrus fruits which together constitute a large portion of Fresh Enough Grocery Stores' sales: oranges and grapefruits. Citrificate will include a simple web application UI which can be used to identify whether or not a fruit is an orange or a grapefruit, and present visualized data which testers can refer to. The Citrificate Project's deliverables will be intended to form the core of future efforts to create a computer system, including color scanning cameras and scale hardware, which employees can then use to rapidly sort oranges and grapefruits in real-time. Such a system will be necessary because Fresh Enough Grocery Stores (alternatively referred to as simply, "The Company" from here on out) previously made a decision to exclusively hire employees with critically impaired near vision, whose vision cannot be improved satisfactorily with corrective ophthalmic lenses. The company has subsequently found itself in the midst of crisis concerning its employees' ability to sort fruits and vegetables with similar qualities, and the ML model delivered by Citrificate will allow us to restore this situation before customers lose confidence in our ability to sell them the correct merchandise altogether.

Application Benefits: As this company's senior leadership is already well aware, the trouble the company has been having with sorting grapefruits and oranges is only one instance of a widespread inability to sort similar products in general. Once the Citrificate project delivers a ML model capable of distinguishing grapefruits and oranges, it will prove that a similar approach can be taken to help employees identify other similar items in store inventory. With sufficient data on other products, further ML models can be trained for the purpose of differentiating these different items. In this way, the company can continue its mission of employing people with visual impairment and

still maintain a reputation for competence and efficiency with its customer base. Furthermore, employee morale will be dramatically improved thanks to a newfound lack of complaints from customers on account of purchasing wrongly categorized produce and other items from our store. Having successfully worked around our employee base's abilities, the company will be lauded for its altruistic employment practices and benefit and enhanced reputation in the public eye.

Application Description: The Citrificate project will feature, at its core, a ML learning model trained to classify reliably between grapefruits and oranges. It will accomplish this by studying a large dataset featuring 10,000 samples of known citrus fruits' color, weight, and diameter. By doing this, the model will identify patterns in these attributes which will allow it to predict the identity of yet unidentified grapefruit and oranges in the future. Although the ML model produced by this project will not immediately be integrated with camera and scale hardware for obtaining fruit data in real-time, which would be ideal for in-store use and should be the subject of future development, testers will be able to interact with the model through a web application UI to further verify the accuracy of the model's predictions before integrating the aforementioned hardware.

Data Description: The raw data utilized in this project will be sourced from a publicly available dataset hosted on Kaggle bearing the CC0 Public Domain license, which can be found here: <https://www.kaggle.com/datasets/joshmcadams/oranges-vs-grapefruit>. The dependent variable here will be the class of the citrus fruit in question, and the independent variables will be the fruit's diameter in centimeters, its weight in grams, and its color, represented as the composite of its red, green, and blue values (given on a scale of 0-255, as this is the standard way in which computerized colors are typically represented, for technical reasons). The dataset being used, although lacking in missing data or outliers, is limited in that it has been artificially inflated to make the dataset a size suitable for creating a ML model. As such, although this "dummy" data has been generated based on data

measured from real instances of grapefruits and oranges, the ML model may likely perform unusually well during initial testing performed with this dataset. As such, extensive testing with real data collected from instances of our produce will have to be conducted as soon as possible following the project's completion to ensure the model's efficacy.

Objectives & Hypothesis: The ideal outcome of the Citrificate project would be producing a ML model and intuitive UI which can successfully differentiate between grapefruits and oranges with at least a 90% success rate. My present hypothesis is that it will be possible to derive such a model from the initial dataset, given that the average person (with typical vision) is typically able to make such a distinction with little difficulty and a high degree of success.

Methodology: The Waterfall Methodology will be utilized for this project's development and implementation. Given that I shall be the project team's sole member, this methodology will ensure that careful attention to detail is given to each sequential step in the process and that any backtracking is kept to the minimum. Furthermore, being a project with fairly small scope, and due to the nature of the product being delivered, it would be difficult and unintuitive to attempt to develop different components of the project simultaneously: Each step in the project's completion is a prerequisite to the next step, so the Waterfall Methodology appears to be the most logical choice.

Funding Requirements: The Citrificate project will require funding for a single software engineer acting as both a web developer and a machine learning/data engineer, available at a flat rate of \$10,000, all paid upfront. No additional costs for equipment or licensing will be necessary as the engineer in question will already have a suitable workstation by virtue of his profession and will exclusively use free or open-source tools to create the project's deliverable.

Data Precautions: None of the data which will be used in this project is sensitive or protected in

any way – the data is publicly available at Kaggle under the CC0 Public Domain license.

Developer's Expertise: As the project's lead (and sole) developer, I am qualified through experience writing simple web applications similar to the one being proposed here, and within a few days time, I will also be an alumni of Western Governors University with a Bachelors Degree in Computer Science. Finally, but not of the least importance, I have recent experience with creating simple ML models as well as the libraries and tools instrumental in creating such models.

Part B: Project Proposal

Project Statement: Fresh Enough Grocery Stores is in the process of implementing a company-wide policy to exclusively hire employees who suffer from irreparable and uncorrectable near vision loss, causing a crisis in, among other things, our operational ability to properly sort and distribute produce which have similar shape and texture. Nowhere is this situation better exemplified than in the case of sorting grapefruits and oranges. The negative effects of this recent development on the company's reputation and the number of customer complaints received daily are already readily observable, and will surely cause catastrophic damage to the company's ability to retain its customers on the long term if left unabated.

Client (or Customer) Summary: The client for this project shall be the company itself – Fresh Enough Grocery Stores Inc. In particular, a small group of internal ML engineers and full stack web developers, including myself, and key members of the company's executive leadership. The data product procured by this product, in this case a cloud based web application, will allow the clients to further test the ML model encapsulated by the web application and, in the case of executive leadership, serve as a proof of concept demo which will convince them to grant further resources for the project's true endgame: A hardware system capable of supplying visual and weight data to

the ML model in real-time. Such a system could then be deployed to stores and offer employees a rapid, reliable, and cost-effective means of sorting grapefruits and oranges, helping restore customer confidence in our ability to place the correct items in the correct lots.

Existing System Analysis: The company's existing workaround to the difficulties introduced by the shortcomings of its new staff is to use handheld or stationary barcode scanners wherever and whenever possible. If an item is, for some reason, not labelled, the visually impaired employee may be able to identify the item simply by shape, weight, texture, etc. While this works somewhat well for items which are consistently labelled, as with most processed, packaged, or canned foods, it is highly ineffective when an employee must differentiate produce of similar size, weight, and texture. Especially since many produce items, especially oranges and grapefruits, arrive unlabelled and often in mixed lots. The proposed solution would resolve this, and could be extended to differentiating different items in stores in the future if successful.

Data: The raw data set features 10,000 rows of data, devoid of any null entries (no doubt owing to the partially fabricated nature of the dataset), with each row's classification denoted by the "name" column. The value of the "name" column in the raw dataset is a string value, either "orange" or "grapefruit", although the rest of the data is numeric. In particular, the "diameter" and "weight" columns contain float data values, and the values of the columns describing color ("red", "green", and "blue"), are all integers from 0 to 255, corresponding to a standard RGB hex value. For the initial training of the LinearSVC ML algorithm used in this project, no further data collection will be necessary, as the dataset procured from Kaggle already contains more than enough samples for such a model. However, the data will have to be transformed in order to properly classify data in predictions, and likely for the ML algorithm to converge within a reasonable amount of iterations over the data. Specifically, the "name" column will have to be encoded into binary format, using the

One Hot Encoding method or similar, as all of the data in a dataset must be numeric to be understood by the ML model during training. Also, because the independent variables' values have very different ranges, it may be necessary to scale the data down to a normal distribution such that the ML model can more easily recognize patterns in data during training on the dataset. In addition to being used to train the initial ML model, the dataset will be stored in a comma separated value file residing in the web application's root directory where it will be used to generate static graphs comparing key values which appear in the web application UI, as a means for the user/tester to better visualize the dataset. The complete dataset itself will also appear as a user interactive data frame in the web application interface. Finally, the dataset will be maintained and expanded upon as more fruit data is aggregated over time to help improve the ML model iteratively and assess it's accuracy. The initial dataset, being partially fictitious at this time, does not feature any known data anomalies, so no action will be required to prune the dataset of such things during this project.

Project Methodology: The Waterfall Method will be utilized for the development and eventual deployment of the Citrificate project. Given the small size of the team working on this project initially (only myself), there will be limited opportunities to work accurately on multiple tasks in parallel. It is also extremely unlikely that the requirements for this project will change, a situation where the Agile approach might have been more suitable, and the quality of the ML model produced will benefit greatly from advance time and consideration given to each step in the process – making sure the dataset is properly formatted and that there is enough difference between orange and grapefruit that a ML model can be trained to distinguish between the two, for example. Finally, the problem being encountered by the company here is well understood, and this is a scenario where the completion of each phase in the project is a logical prerequisite for the next phase in the project. For example, testing the ML model cannot be performed prior to training it, which in turn cannot be performed before the dataset is transformed into a trainable format, etc. Planned development by

phase is as follows:

1. **Requirements:** This proposal will be written, identifying the business problem to be solved by the proposed solution, setting a cost estimate for the completion of the project, and conditions for the projects success will be defined.
2. **Design:** A mockup of the final web application to be delivered will be created in Figma, to guide the development of the UI. The desired inputs and outputs of the ML model to be produced will be defined, as a guide to shaping the dataset and the ML model training process.
3. **Implementation:** The ML model will be produced by training, tweaking hyper-parameters, and then testing model prediction performance on validation and testing datasets. Once this is implemented, the model will be exported for use in the UI. Finally, the web application front end will be coded using the Streamlit framework and the ML model will be integrated into it, allowing predictions to be made through the UI and the dataset to be visualized
4. **Verification or Testing:** Distinct from the verification and testing of the ML model itself, herein the web application UI itself will be tested prior to deployment on a local machine for browser compatibility, unexpected behavior, and other bugs which might hinder the user experience or yield incorrect data.
5. **Deployment and Maintenance:** The web application shall be deployed for use by the client into the cloud via Streamlit's free hosting service. Seeing as the application's source code when deployed in this manner originates from the project's GitHub page, any updates from future work which are merged into the main branch will automatically be deployed as soon as available, ensuring that future improvements are delivered automatically and in a timely

fashion.

Project Outcomes:

- An initial version of a ML model capable of making predictions about fruit classification between oranges and grapefruits with the desired accuracy of 90% or better on the test dataset.
- An integrated web application user interface for making predictions based on additional data recorded following project completion.
- A README.md serving as a user guide on the web application's predictive features and data visualization metrics.

Implementation Plan: The overall strategy of this project will be to find a ML model with which the 10,000 fruit samples we have will be sufficient to distinguish between grapefruits and oranges with the desired degree of accuracy. Preliminary visualizations of the traits contained in the chosen dataset suggest that while there is considerable overlap in the dimensions and color of the two fruits (as was expected), it may still be possible for a Linear SVC model to correctly classify fruits when considering all factors described in the dataset at once. As such, the dataset will first be transformed into entirely numeric format and scaled for training using the Python language within the Jupyter Notebook environment. The untrained ML model itself will be imported from the well-reputed Scikit-learn library, with it's API being used to train the model and adjust hyper-parameters proper. The APIs necessary for data manipulation and visualization will depend upon the Numpy, Pandas, and Matplotlib libraries.

The dataset will be split into training and test datasets using 80% and 20% of the original

dataset respectively, with the test dataset being further split in a similar fashion into a validation and testing dataset. Once the ML model has been trained on the training dataset to recognize patterns and make predictions accordingly, it's efficacy will be assessed by comparing classification predictions on the validation dataset compared against the set's true labels. Metrics to be assessed include accuracy, and a confusion matrix cross-comparing the model's predictions v. the actual labels. Once this analysis has been complete, the ML model will be retrained on the training dataset with altered hyper-parameters if necessary, in a bid to improve prediction accuracy. Finally, the ML model will best tested on the test dataset, and a scoring process similar to validation will take place. If the model's results are deemed satisfactory, meeting a 90% minimum accuracy score, it will be integrated into the web application. The web application front end itself will be written in Python and rely heavily on a web application framework called Streamlit, which will also double as a hosting service for distributing the app to the client for use and further testing with data accumulated in the future.

Evaluation Plan: Evaluation and verification during key stages in development will take place as follows:

- **Design:** The mockup of the eventual web application will be assessed as meeting UI criteria. Namely, it will be guaranteed to include a means of inputting fruit parameters and using the ML model to make a prediction based on that data, and visualizations of the dataset via graphs and data frames.
- **Implementation:** The trained ML model will be deemed satisfactory if it can achieve a 90% accuracy on the test dataset or better, and the front end will be deemed successful if it can reliably convert input data into predictions which continue to meet 90% accuracy or better, with reasonable prediction confidence. For example, 100% confidence in predictions for

every input would be unreasonable, as no ML model should be 100% confident for every prediction. Especially if such predictions are routinely false. This will help rule out an overestimation of performance on the test dataset due to data leakage between the training and test sets.

- **Deployment (Project Completion Checkpoint):** The model's functionality will be tested upon deployment to the cloud to ensure that the prediction model and other visual elements work similarly compared to their functionality on a local machine. Any package dependency errors will be addressed by adjusting the package dependency file for compatibility with the Streamlit cloud environment.

Resources and Costs: The project will incur no hardware or software costs, as the development team's sole member will operate entirely on their own, pre-existing computer hardware and use entirely free, mostly open-source software, libraries, and frameworks to accomplish the project's goals. Deploying, hosting, and maintaining the app will also be free via the Streamlit cloud hosting service. The app's development, however, is estimated to take 10 working hours and cost a flat rate labor fee of \$10,000 paid to the ML/Front End engineer upfront. Additional work hours will be provided at no additional cost to the company, by said engineer as necessary to complete the deliverables promised herein.

Timeline and Milestones:

Project Start Date (05/01/22):

- Completion of project requirements review, mockup UI evaluation, and initial ML model (4 hours).

Project End Date (05/02/22):

- Completion of ML model tuning and test performance evaluation, web application UI, ML model integration, UI testing, and deployment (6 hours).

Part D: Post-implementation Report

A Business (or Organization) Vision: Fresh Enough Grocery Stores was facing a crisis concerning its employees' ability to accurately sort items with similar weights and textures, grapefruits and oranges being a salient and impactful example of this dilemma. With the successful development of a ML model capable of differentiating grapefruit and oranges with an accuracy exceeding 90%, the company was primed to being work on a hardware system to provide the required color, size, and weight data for the ML model in real-time, which would ultimately provided employees with a reasonably accurate, inexpensive, and fast solution for sorting citrus fruits. This project thus helped make the company's initiative to employ people with uncorrectable near vision impairment possible while maintaining its reputation with customers and keeping sorting-related customer complaints to a minimum.

Datasets: The raw data used to train the ML model in this project was sourced from a partially fictitious dataset available on Kaggle.com under the CC0 Public Domain license, containing 10,000 samples of grapefruits and oranges described in terms of their diameters in centimeters, weight in grams, and colors as a combination of red, green, and blue values corresponding to a RGB hex value. An example of a data sample from the raw dataset can be seen below:

	name	diameter	weight	red	green	blue
0	orange	2.96	86.76	172	85	2

Due to the ML model requiring entirely numeric data for training, the “name” column’s values were binary encoded using the One Hot Encoder method, implemented in the Scikit-learn library, before being separated from the rest of the table as the target values the ML model learned to predict. The remaining columns were scaled to a normal distribution using the StandardScaler object also implemented in the Scikit-learn library, thus making it easier for the LinearSVC ML model to identify patterns in the dataset, converging on such patterns in a reasonable amount of iterations (set to 10,000 in the model eventually trained). A sample of the training data eventually passed to the ML model to fit it to our classification problem is shown below, where columns 0 – 4 represent the diameter, weight, red, green, and blue columns in that order:

	0	1	2	3	4
0	1.964091	2.020444	0.101717	-2.747355	0.069225

The original dataset can be found at <https://www.kaggle.com/datasets/joshmcadams/oranges-vs-grapefruit>.

Data Product Code: The ML model’s data analysis started with formatting the data samples into an entirely numeric format suitable for fitting the model, a process which is described in greater detail above. The project’s dataset was descriptively visualized by way of two plots comparing key data sample characteristics and a data frame implemented by the Pandas library, which interactively presents the original, unaltered dataset to the user. The plots were created through the Matplotlib library’s 2D and 3D subplot implementation, depicting the weight and diameter distributions of grapefruit and orange samples, and the color distribution of those same samples respectively.

For the predictive (that is, non-descriptive) portion of the project, the input parameters

entered by the user in the UI are simply passed to a copy of the trained ML model, which then produces prediction probabilities. These probabilities are then interpreted and displayed to the user, along with the ML model's "guess" – simply the greater of the two probabilities returned. The ML model's performance was considered satisfactory if it's accuracy was at least 90% on all samples in the test dataset, which was deemed an appropriate method for ensuring sorting errors in actual usage would be kept to a tolerable minimum. The ML model was trained by fitting it to a randomized subset consisting of 80% of all samples in the original dataset, and then assessing it's performance on a validation subset consisting of 16% of all remaining samples. The model's performance on this validation set exceed the required accuracy, so it was tested on the final test dataset comprising the remaining 6% of original data, where it continued to meet the desired accuracy benchmark and was deemed a success. Following completion of training, it was exported via the Python Pickle library for use in the final web application. The analysis of validation testing data helped ensure the reliability of this project's predictive method by offering evidence that the ML model could, in fact, properly classify the grapefruit and orange data samples. Furthermore, that same analysis inspired my choices for the project's descriptive methods: I chose to show the correlation and overlap in key traits between the two fruits, and hoped to show a visual explanation as to why grapefruits and oranges could be so reliably differentiated from each other. The charts chosen for the descriptive method ultimately did a good job of showing the marked difference in proportion and color ranges for each fruit.

The ML model resides in the same directory as the web application itself in the Streamlit cloud. Since the exported Scikit-learn ML model produced in Jupyter Notebook was less than 5 MB, no stand alone database solution was necessary. It is worth noting that a copy of the original dataset in csv format also exists in the same directory as the web application, as all descriptive methods shown in the UI are based on that data. All source code and software dependencies for this

project, as well as the dataset and other resources used for the development of this project shall be submitted with this proposal, along with the project's Github link as an alternative.

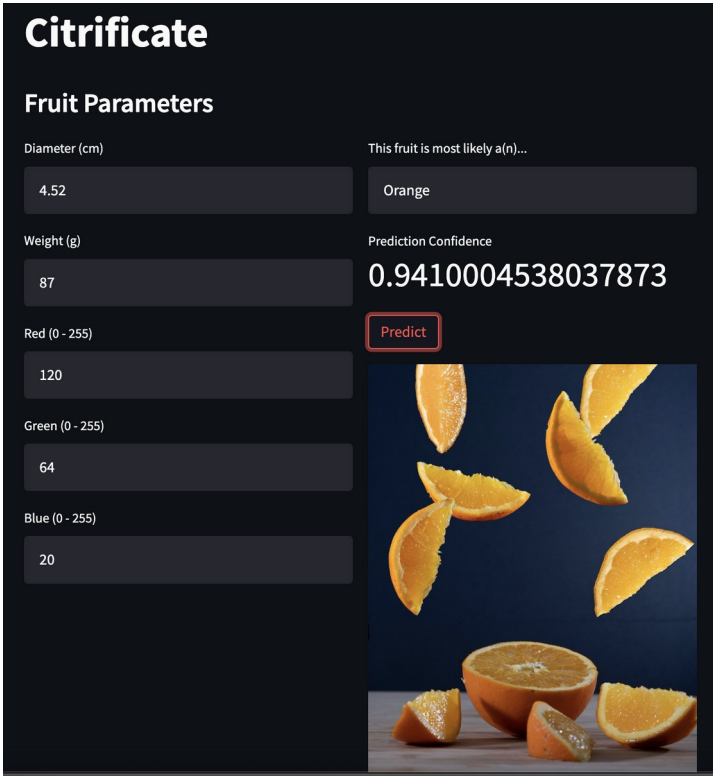
Objective (or Hypothesis) Verification: The ideal outcome of the Citrificate project was to produce a ML model and intuitive UI which could successfully differentiate between grapefruits and oranges with at least a 90% success rate. The operating hypothesis was that it would be possible to derive such a model from the initial dataset, given that the average person (with typical vision) is typically able to make such a distinction with little difficulty and a high degree of success. Both the apparent accuracy of the final product, and the data visualizations highlighting the different (though overlapping) ranges that grapefruits and oranges inhabit in terms of proportion and color support this hypothesis, though evidence to the contrary may yet be discovered if the ML model fails to perform accurately on additional data samples collected in the future.

Effective Visualization and Reporting: The visualizations produced during the Citrificate project aided during testing of the final product in one sense by suggesting input values to test for edge cases in classification: The two plots depicting color and proportion distribution at a glance give an impression of which values should be predicted as being either a grapefruit or orange with a very high degree of confidence, and even more importantly, values which lie in an area of significant color and proportion overlap – predictions for inputs in these areas would be expected to have a lesser degree of confidence. The latter is especially important because testing values in this hard to differentiate zone would give us some of the lowest possible confidence values for any prediction made by the model. In other words, if the ML model made 90% accurate predictions or better on fairly ambiguous data values, it was reasonable to think that it would perform at least as well or better on any other set of values.

Another descriptive method of considerable importance was the confusion matrix comparing

the ML model's false positive and false negative rate, offering a visualization of the model's performance on the test dataset in a more absolute manner – instead of showing how accurate the model was in terms of a percentage of fruits which were correctly classified, it simply shows how many fruits were classified as grapefruits, when they were, in fact, grapefruits, and vice versa. This helped establish even grater confidence in the trained model's efficacy.

Accuracy Analysis: The target minimum accuracy for this product's deliverable to be considered successful was 90% on the final test dataset, consisting of 1,600 randomized samples from the original dataset of 10,000 samples. The model exceeded this requirement with a score of 96.05% on the test dataset. This was consistent with results yielded by the ML model when entering inputs on the same scale as the original dataset, although a better test for future developments of this project would be to test the model's accuracy on a much larger dataset of actual data samples aggregated from real in-store produce (at least 1,000 samples would be ideal). Example of prediction method outputs can be seen below:



The screenshot displays the 'Citrifcate' web application interface. On the left, under 'Fruit Parameters', there are five input fields: 'Diameter (cm)' with value 4.52, 'Weight (g)' with value 87, 'Red (0 - 255)' with value 120, 'Green (0 - 255)' with value 64, and 'Blue (0 - 255)' with value 20. On the right, the prediction results are shown: 'This fruit is most likely a(n)...' with the answer 'Orange', and 'Prediction Confidence' with the value '0.9410004538037873'. A red 'Predict' button is located between the input fields and the prediction results. Below the text, there is a photograph of several orange slices and a whole orange on a wooden surface.

Citrificate

Fruit Parameters

Diameter (cm)

7.5

Weight (g)

130

Red (0 - 255)

114

Green (0 - 255)

81

Blue (0 - 255)

15

This fruit is most likely a(n)...

Grapefruit

Prediction Confidence

0.9894891451140285

Predict

Application Testing: Per usage of the Waterfall Method, the web application delivered in this product was tested incrementally and strictly sequentially as each feature was written. First, the app's boilerplate code was written which allowed the empty app to run an instance on the local workstation being used, in turn allowing it to be opened in a browser page. From that point onward, the app and source code were kept open side by side. As new elements were added to the page, the app was configured to automatically reload the source code. In this way, changes to the page could be seen in real-time, the page interacted with freely, and adjustments made to the layout according to this feedback. This also made it possible to identify visual bugs as soon as they manifested, helping ensure that further features to the app were built on a functional foundation. Once the project met functional completion, different values were plugged in (including blatantly incorrect or missing values) to the input parameters to ensure that the model and UI behaved as expected. This notably helped debug an issue where the login screen UI elements would persist even after the user

had logged in with correct credentials and loaded the app's main interface.

Application Files: No installation is necessary to use Citrificate on a Windows 10 machine, as the web app is accessible online as described in the user guide below. The files included in the submission folder are as follows, and can also be found on Github at

<https://github.com/builditbear/wgu-capstone>:

1. capstone-proposal.odt: This paper.
2. capstone-workbook.ipynb: Jupyter Notebook containing all original data exploration, analysis, and ML model training as well as validation and test analysis.
3. citrificate.py: source code for the web application.
4. citrus.csv: The original dataset from <https://www.kaggle.com/datasets/joshmcadams/oranges-vs-grapefruit>.
5. fruit-classifier.pkl: The final, trained ML model utilized in the web application in serialized form.
6. grapefruit.jpg: Picture sourced from <https://unsplash.com/photos/GLFt9RL9kDY> and used under the [Unsplash License](#).
7. input-scaler.pkl: Serialized, fitted scaler for scaling input data to a normal distribution prior to insertion into ML model.
8. orange.jpg: Picture sourced from <https://unsplash.com/photos/jBHv766AKrE> and used under the [Unsplash License](#).

9. README.md

10. requirements.txt: List of project dependencies.

User Guide:

1. Navigate to the Citrificate web app at

<https://share.streamlit.io/builditbear/wgu-capstone/main/citrificate.py>

2. Enter the following credentials to gain access to the user interface.

- Username: “test-user”

- Password: “test-password”

3. Enter all requested input parameters as listed on the left side of the screen. Click the “Predict” button on the right hand side of the screen, and a prediction will be presented after a few moments, along with a visualization of that prediction, including the ML model’s self-assessed confidence in that prediction. Please note that if *not all parameters are entered*, an error message will be thrown. This message will vanish once you attempt to re-interact with any of the fields.

4. To make another prediction, simply repeat step 3 with different input parameters.

Summation of Learning Experience: In retrospect, my experiences in Intro to Machine Learning here at WGU helped prime me for success in this project by instilling a basic sense of what Machine Learning is about and how it functions. Similarly, one of the tasks in that course was

writing a project proposal very similar to this one, although actual implementation of said proposal was out of scope for that particular project. My own personal projects working on small web apps and building something considerably larger in scale in Software 2 was also instrumental in helping me rapidly prototype and construct the app implemented in this project. Despite all of that, I had never worked with Machine Learning libraries or tools prior to this undertaking, so I did take the time to work through key sections in a Udemy ML/Data Science course that a lot of my peers recommended to get up to speed on the essentials in a timely fashion. I also took the time to get acquainted with using Streamlit – a front end framework that dramatically reduced the time I would have otherwise spent on cobbling together a decent looking UI through more traditional tools like HTML/CSS and JavaScript. Even if I never use it again, it got the job done and it was enjoyable getting exposure to another framework!

I feel the greatest contribution this project has made to my concept of lifelong learning is reinforcement of something I already knew: It will often be the case in my career as a software engineer that I will simply not know what I need to get a job done – I'll be out of my depth on a library in use, encounter a design pattern or work methodology I'm only vaguely acquainted with, or I'll have to use a programming language out of my comfort zone because the tools for a task happen to be built around that language. With that in mind, perhaps the most important skill I've developed, both in my formal education and in my personal time, is the ability to rapidly accrue these skills and locate the knowledge I need to keep growing and meet expectations. Lifelong learning isn't just a virtue in this field – it's the key to surviving in a rapidly evolving industry!

Part E:

No in-text citations or references were necessary for this paper as no content herein was quoted, paraphrased, or summarized.