PlantDoc.

Group 1:

Inês Rodrigues| MBq

Joana Inverno | MEEC

Maria Alvarez| MIEMN

Professors/ Mentors:

Aneesh Zutshi (Coordinator)

Tahereh Nodehi

Sina Fazlollah Zadeh

António Coelho

David Belo

Simão Gonçalves

Index

Abstract

The United States (US) is considered one of the leading agriculture producers and exporters worldwide, with its agricultural productivity playing a crucial role in the overall country's economic sector. Additionally, the recent increase in food consumption has led to a requirement for a higher supply and, thus, a demand for healthier crops. Considering that, nowadays, the agricultural industry is still technologically underdeveloped and heavily labour-intensive, automatic disease detection in plants plays a notable role in early diagnosis and correct control decision-making. In turn, this will result in a negligible yield loss, lower production costs, decreased pesticide exposure for agricultural workers, and the promotion of healthy and sustainable agricultural development.

Convolutional Neural Networks (CNN) is an algorithm that allows for the detection of features automatically. This research project presents a deep learning architecture, which allow the detection and classification of specific infections in a panoply of plant leaves.

First, the data (New Plant Diseases Dataset) was collected from Kaggle, and the image files were reorganized, cleaned and re-labelled. Exploratory Data Analysis (EDA) was made before and after the data transformation to explore and analyze the properties of the images, labels and the distribution of each category. After EDA, the dataset was divided into batches, then a second EDA process was done and lastly the implementation of a baseline.

Different techniques were applied in order to increase the accuracy of the models after which, the best final model reached an accuracy of 91.7%, correctly predicting the health state of the random images fed to the program. These were obtained after slight changes in the number of batches, image sizes, structure of the model, number of epochs, the proportion of healthy images versus the others (diseased), and the number of times the whole train batch was executed through the model.

Applying this algorithm in an app, that is versatile both for B2B and B2C, it is possible to create a full integrated solution, where a client can diagnose their plants and create and implement a treatment plan with other features built into the application.

1. Introduction

1.1. Convolutional Neural Network (CNN)

Convolutional Neural Networks (CNNs) are efficient at modelling spatial data. They can extract features and patterns, enabling tasks such as image classification or object detection.

The convolutional layers are the backbone of CNN, as they are applied to extract the prominent features within the images. Once progressing into the network, the system begins to recognize increasingly complicated characteristics, such as shapes, digits, and various regions of the face.

The 2D convolution starts with a kernel that strides over the input data, and the resulting convolution is a 2-dimensional array containing the image's correlation with the applied kernel. The input array is transformed into a smaller array during the convolution process while maintaining the spatial correlation between the pixels by applying these filters.

To further reduce the size of the feature map and the amount of computing power needed to process the data generated from convolution, pooling is applied before further processing. This helps to compress the dimensions of the feature map further. For this reason, pooling is also referred to as subsampling. Pooling summarizes the features within a group of cells in the feature map and helps to prevent overfitting.

Convolutions and pooling can be applied to CNN many times. The metrics of the model generated depend on finding the correct number of times these steps should be repeated.

After this, the output is flattened and converted to a single-dimensional vector to make it compatible with an Artificial Neural Network (ANN). This flattened output is passed through one or more fully connected neural networks.

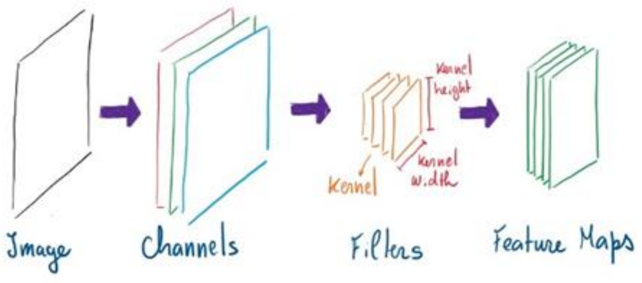


Figure 1 -Visual representation of the CNN inner workings.

2. Business Model

2.1. Problem

The United States (US) is one of the largest countries in the agricultural sector (Boss, 2022). However, traditional disease detection methods employed identify differences in an infected leaf sometimes late, resulting in financial losses leading to a reduction in quality, which could, in turn, lead to unnecessary economic loss. On the other hand, early diagnosis and treatment of such illnesses are advantageous to assure high quantity and optimal quality, according to prior studies using automated disease recognition on rice crops (Sarkar, 2020). Thus, feature selection techniques and image preprocessing are significant steps in crop disease identification and diagnosis.

2.2. Solution

The project's primary goal is to create "PlantDoc.", an app that uses deep learning for the automatic detection and recognition of diseases in the leaves of plants. For that, a program must be trained to differentiate between healthy and diseased leaves and distinguish between the different diseases.

2.3. Product

The first step in creating a business model was to identify the problem, which, as previously mentioned, lies in how underdeveloped the current agricultural sector is in identifying crop diseases. Additionally, the larger the area to analyze, the less efficient becomes the manual method currently employed. Thus, PlantDoc is an app created as a new opportunity to automatically identify crop diseases with AI support. Additionally, this software recommends products and treatment tips for plant recovery or prevention of spreading the infection to surrounding healthy plants.

Finally, PlantDoc. 's unique value is having an integrated CNN algorithm, which the client can quickly use to diagnose the health of their plants through a photo. Additionally, after identifying the plant's disease, this app can give out a selection of products and tips catered to the client's needs, as shown in Figure 2. In addition to these options, the client can purchase the suggested products through the app, making this application an Integrated Solution. To conclude, at a business level, and together with drone technology, this app allows for more efficient monitoring of larger areas.

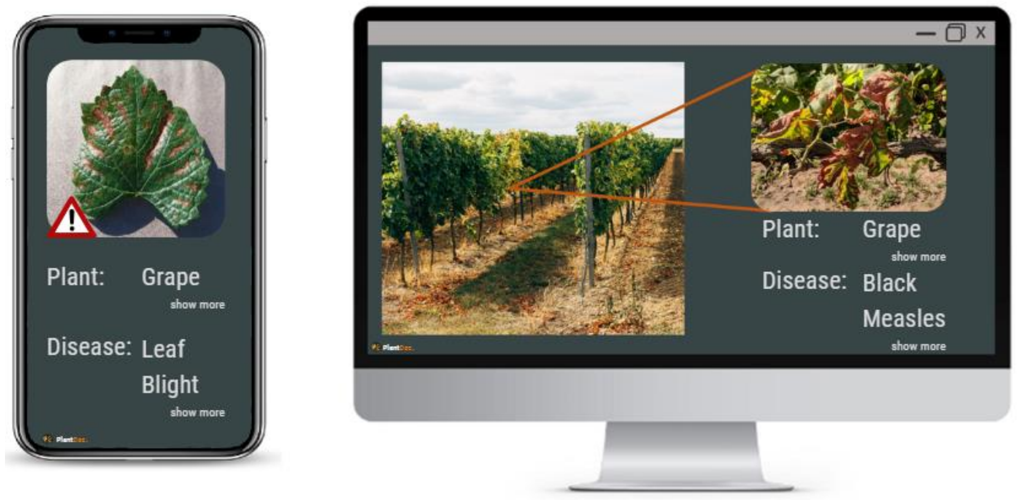


Figure 2 - Visualization of the PantDoc.'s app general layout

2.4. Market Strategy

PlantDoc. 's market is directed both to B2B and B2C markets. For Business to Consumer, which consists of gardeners and plant enthusiasts, the app is used for smaller-scale operations and more sporadic use. In Business to Business, the sectors involved are agricultural entities such as farmers and agricultural producers alike, in which the app is used for a larger scale operation, potentially employing drones or cameras to allow for surveillance of wider areas. Furthermore, these clients may use these services through an Android/IOS app or a web-based application.

The US was chosen to initially launch the product since this is one of the largest countries in the agricultural sector. Therefore, our marketing strategies would be mostly targeted to the top three largest Agricultural Companies, Cargill, the Archer-Daniels-Midland Company (ADM) and John Deere company, as represented in **Table 2**.

Table 2 | Main contributors to the agricultural sector in the US (Saud, 2022; Agriculture, 2022).

Name of Company	**Revenue (USD billions)**	**Headquarters**
Cargill	114.69	Minnesota, US
ADM	64.34	Illinois, US
John Deere	37.35	Illinois, US

2.5. Costs and Revenue

The PlantDoc.'s app would be divided into a free version, where the customer could access some basic features, including disease classification. However, it only allows up to 10 images to be analyzed monthly. On the other hand, premium customers, which correspond to the customers with a paid version of the app, could have access to unlimited image recognition and extra tools and functionalities (*e.g.*, product and treatment suggestions, where these products can be found, the application process, and purchases through the app).

Other sources of revenue can be through collaborations with product-selling companies or advertising.

The cost sources of the of the application consist on the costs of setting up a website with its content, all the marketing strategies and legal fees, mainly concerning the collaborations with other companies and the advertisement in the app. Furthermore, to ensure that the data in the app is always up to date and both the algorithm and other functionalities of the app work correctly, there is a need for constant research and development in this sector and maintenance of the app after its development and implementation.

2.6. Competitors

There are similar applications, that are this product's main competitors, which are divided into three main types. Some can only identify the type of plant (e.g., *PictureThis*). Others (e.g., *Plant Disease Detector*) can identify and determine the type of disease. However, in both these cases, a lack of product and treatment suggestions makes our product stand out by comparison. Lastly, the applications that give product suggestions are mainly directed to pest control (e.g., *Agrobase*) and not as much disease (bacterial and viral).

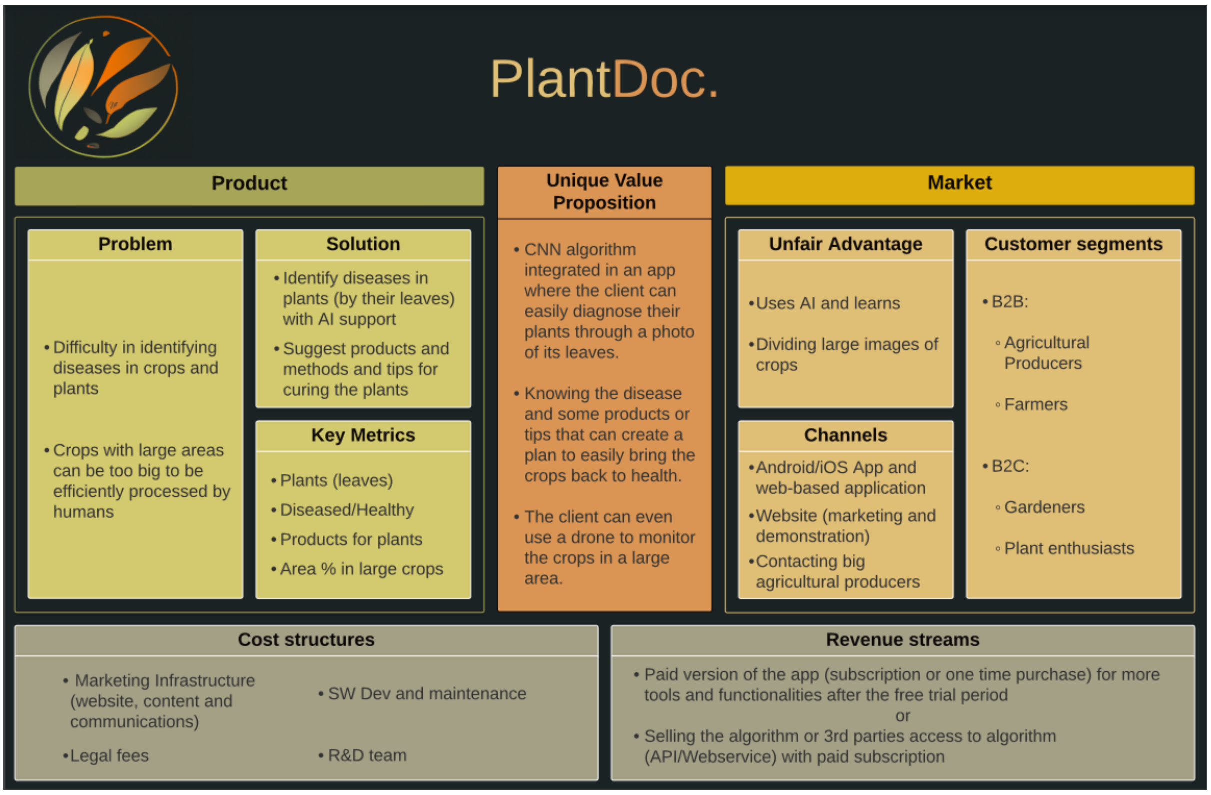
2.7. Evaluation and Performance Metrics

Regarding performance prospects, for similar applications the lowest number of transfers is around 10 thousand. In the begging, PlantDoc. will be more directed to the individual user, with higher percentages of the premium subscribers being the more enthusiastic gardeners. Afterwards, PlantDoc. is expected to expand to more prominent agricultural industries, from where a higher revenue is also expected.

In order to evaluate the performance and progress of the app, different parameters can be analyzed over time. These can be the number of times the app is downloaded, the number of pictures the app analyzed in a certain period, the geographical spread of the clients, the number of paid subscribers, the client reviews and complaints and the number of purchases made through the app.

2.8. Business Model Canvas and SWOT Analysis

To summarize the previously mentioned topics, the graphic representation of the business model canvas and the SWOT analysis is presented in Figure 2.



PlantDoc.

Product

Problem

- Difficulty in identifying diseases in crops and plants
- Crops with large areas can be too big to be efficiently processed by humans

Solution

- Identify diseases in plants (by their leaves) with AI support
- Suggest products and methods and tips for curing the plants

Key Metrics

- Plants (leaves)
- Diseased/Healthy
- Products for plants
- Area % in large crops

Unique Value Proposition

- CNN algorithm integrated in an app where the client can easily diagnose their plants through a photo of its leaves.
- Knowing the disease and some products or tips that can create a plan to easily bring the crops back to health.
- The client can even use a drone to monitor the crops in a large area.

Market

Unfair Advantage

- Uses AI and learns
- Dividing large images of crops

Channels

- Android/iOS App and web-based application
- Website (marketing and demonstration)
- Contacting big agricultural producers

Customer segments

- B2B:
  - Agricultural Producers
  - Farmers
- B2C:
  - Gardeners
  - Plant enthusiasts

Cost structures

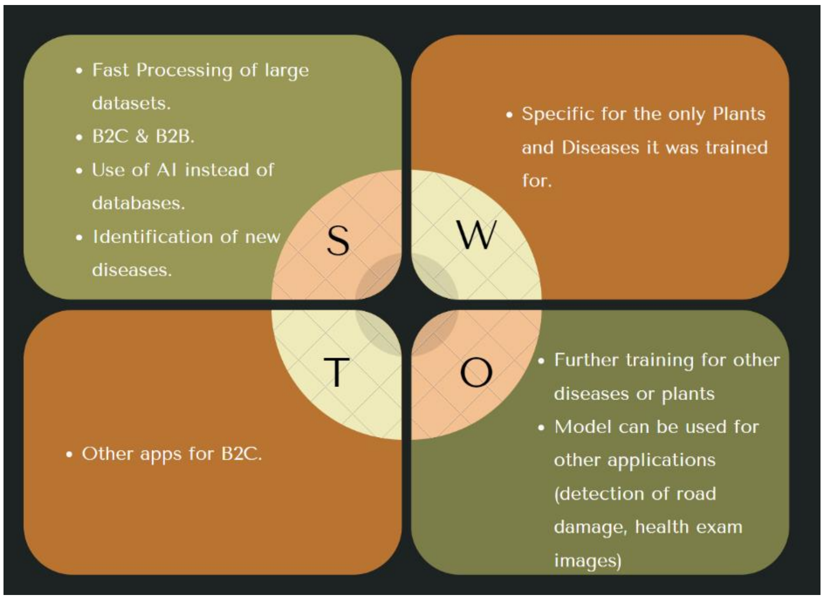
- Marketing Infrastructure (website, content and communications)
- SW Dev and maintenance
- Legal fees
- R&D team

Revenue streams

- Paid version of the app (subscription or one time purchase) for more tools and functionalities after the free trial period

or

- Selling the algorithm or 3rd parties access to algorithm (API/Webservice) with paid subscription



S

- Fast Processing of large datasets.
- B2C & B2B.
- Use of AI instead of databases.
- Identification of new diseases.

W

- Specific for the only Plants and Diseases it was trained for.

T

- Other apps for B2C.

O

- Further training for other diseases or plants
- Model can be used for other applications (detection of road damage, health exam images)

Figure 2 - Representation of the PlantDoc's business model canvas.

3. Methodology

3.1. Image Acquisition

The data used in this project is a dataset from Kaggle called “New Plant Diseases Dataset”, which consists of 87K RGB images of healthy and diseased crops categorized into 38 different classes.

3.2. Data Treatment

In order to ease data extraction and further analysis, all the folders and their respective images underwent a process of name changing according to a rule, where the name consisted of two parts divided by a space. The first part corresponded to the name of the species, and the second part was the name of the label (healthy or name of the disease). If any of these parts had more than one word, these were separated by an underscore.

3.3. Image Augmentation

Even though the images used for the model building had already gone through an augmentation process, a small group of the initial untreated data was selected for an augmentation process. First, the images from the original dataset were resized to (128,128) pixels. Afterwards, a random selection of 4 images was selected to exemplify the type of transformations employed during the augmentation process.

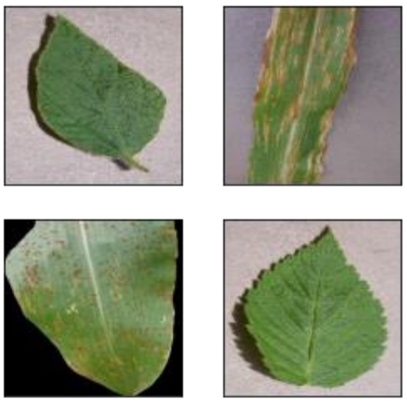


Figure 3 - Plot of random leave images from the original dataset.

The difference between an image's maximum and minimum pixel intensities determines the contrast (**Figure 4**). Here, an enhancement factor of 0.0 returns a solid grey image, while a value of 1.0 gives the original image. Images with an enhancement factor (f < 1) look greyer, while those with a higher factor (f > 1) are more saturated than the original counterpart.

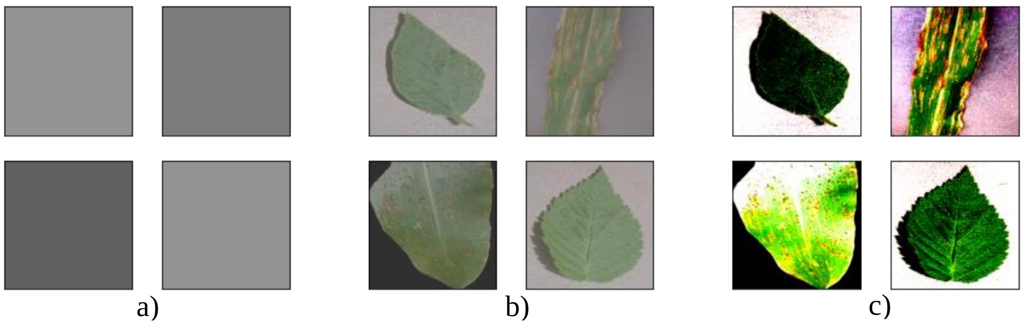


Figure 4 -Plot of random leave images with different levels of contrast. a) Images with a minimum brightness level (f = 0); b) Images with grey undertones (f < 1); c) Images saturated (f > 1).

An image's brightness or luminous brightness (**Figure 5**) is a relative that can be defined as the amount of light relative to an established source. Here, an enhancement factor of 0.0 leads to an output of entirely black images. On the contrary, an enhancement factor of 1.0 gives back the original image.

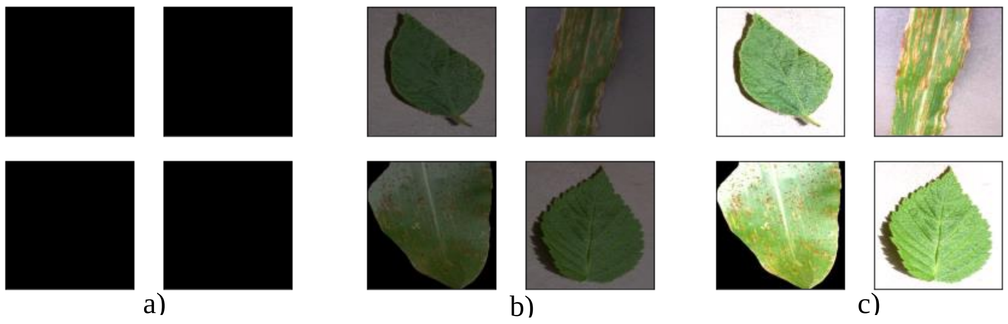


Figure 5 - Plot of random leave images with different levels of brightness. a) Images with no brightness (f = 0); b) Images with lower brightness (comparing to the original) (f = 0.5); b) Images with higher brightness compared to the original (1.5).

Sharpness (**Figure 6**) is a class that describes the clarity of detail found in an image. Here, an enhancement factor of 0.0 gives blurred images back, a factor of 1.0 returns the original image, and a value of 2.0 sharpens the image. However, factors with superior values (f > 2) return images with an emphasized texture (*i.e.*, the image returns with an increased amount of noise).

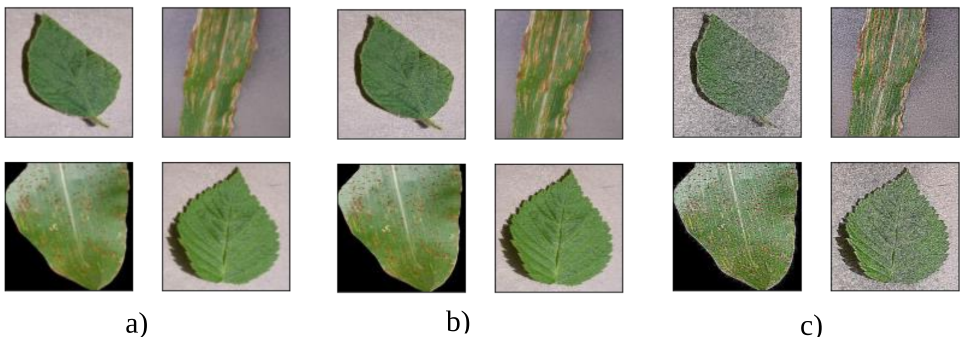


Figure 6 - Plot of random leave images with different levels of sharpness. a) Images with low sharpness (f = 0); b) Images with enhanced sharpness (f = 2); c) Images with excessive sharpness (f > 2).

In color (**Figure 7**), an enhancement factor of 0.0 gives a black-and-white image, while a value of 1.0 gives back the original image. Additionally, negative values return images with complementary colours to those from the original image. Positive values lead to images with colours tending to green.

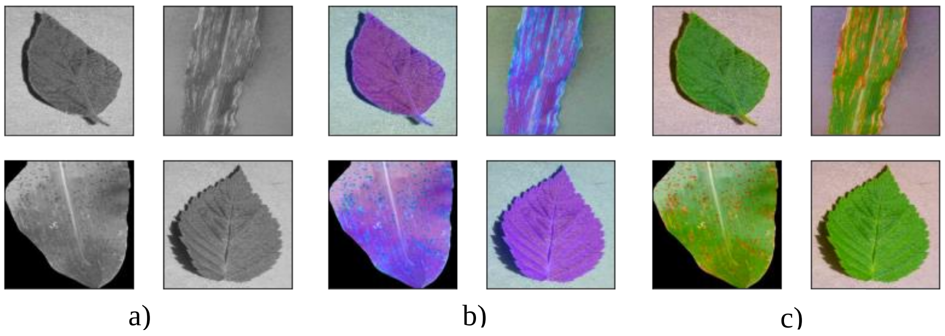


Figure 7 - Plot of random leave images with different levels of color. a) Images with no color (f = 0); b) Images with a negative color factor (f < 0); c) Images with a positive color factor (f > 1).

3.4. Data Extraction

The first step in data extraction was to place the dataset in a zip file in Google Drive, followed by extracting this into a *tmp* file in Colab. Verifying the number of files uploaded into Colab was done to ensure that previous steps were correctly done and check the number of files being used.

3.5. Exploratory Data Analysis

The Exploratory Data Analysis (EDA) consisted in seeing how the augmented data was presented. Thus, the folder names were grouped in different ways (*i.e.*, by species, category (label) or both simultaneously) to analyze the amount of data in each and verify the distribution of images. It is noteworthy that the files and, thus, the label distribution were the only things being analyzed, rather than the images themselves.

3.6. Data Treatment

After EDA, two main lists were created to treat the data better. Firstly, *all_paths* is a list containing all of the paths that led to the images, which will later be used to access each image and load it into the model. Secondly, *all_labels* is a list with all the labels of the images. These lists have the same order, which means that an image in position *x* in *all_paths* has its corresponding label in the list *all_labels* in the position *x*. An additional list containing the species was created for carrying n further analysis of EDA.

The next step was dealing with the labels in string form (*healthy*, *scab*, *black rot*, and others). Here, the labels had to be converted from strings to integers using a dictionary (each key corresponding to a label in a string and the value to a number) because the models work best with data presented as numbers rather than text. During this process, it was essential to verify that the labels were unique and that there were no empty keys.

Afterwards, the data were divided into batches not to exceed the amount of RAM freely available by GoogleColab. Furthermore, these batches can easily be changed to ensure that the Colab limitations can be overcome without compromising the model's results.

Afterwards, the data were divided into easily mutable batches to avoid exceeding the amount of RAM freely available by GoogleColab. This was all assured without compromising the model's results. Additionally, the number of batches was optimized towards better results, and, in this stage, it was made sure that the data was distributed equally throughout the batches, ensuring that each label had around the same amount of data in each batch and that the total quantity of data across all batches was roughly equal.

3.7. Second Exploratory Data Analysis

The second phase of EDA was executed after dividing the data into batches to guarantee safe access.

Firstly, a sample of images was analyzed and applied a random index, which entailed that every time the code was executed, a new group of images was presented. The random plot of images in **Figure 8** made it possible to assess the image quality in terms of brightness, close-ups, rotations, and flips (already present in the images due to augmentation done in the data before loading it).

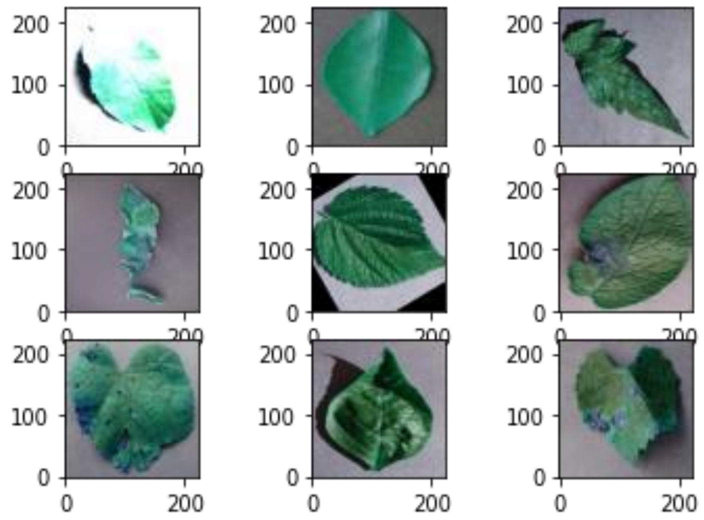
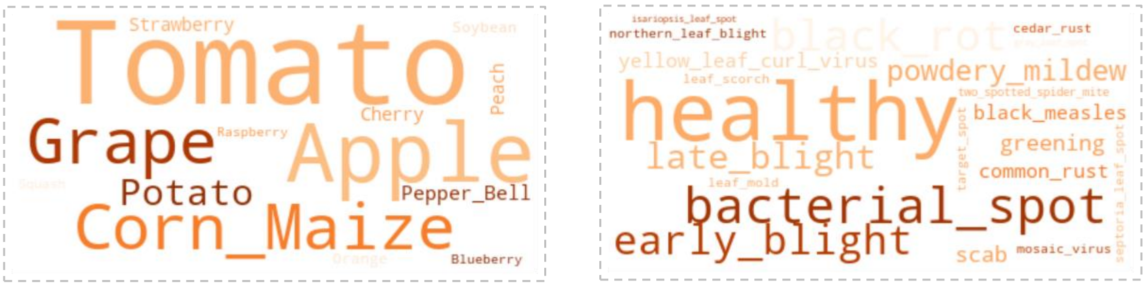


Figure 8 – Plot of randomly assimilated images

To better see the data distribution, a word cloud containing the different types of plants (**Figure 9. a**) and another representing the diseases (**Figure 9. b**) were created.



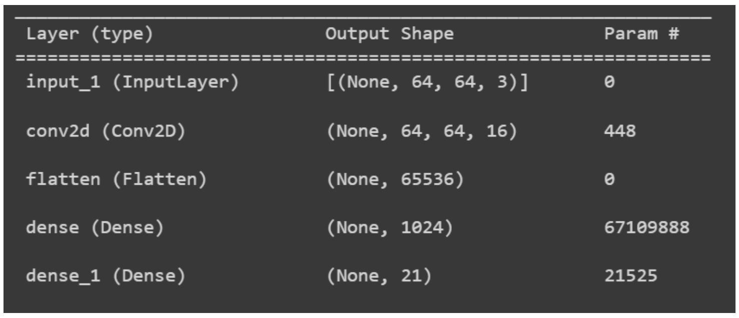
a) b)

Figure 9 – Wordclouds of the species and the labels.

3.8. Baseline

After data extraction, EDA process and data treatment, the baseline was created to check if the data had been correctly organized during the previous steps (so it could be used as input in a model) and to check where optimization steps could be later on implemented.

This first model (**Figure 10**) consisted of a single convolutional layer, a flattened layer and a hidden layer, and its fitting was done with 5 epochs.

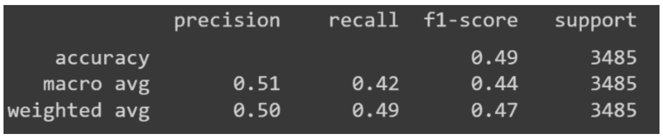


```
Layer (type)                Output Shape              Param #
=================================================================
input_1 (InputLayer)        [(None, 64, 64, 3)]       0

conv2d (Conv2D)             (None, 64, 64, 16)        448

flatten (Flatten)           (None, 65536)             0

dense (Dense)               (None, 1024)              67109888

dense_1 (Dense)             (None, 21)                21525
```

Figure 10 - Summary of the baseline model.

For this model, 10 batches were created in the previous steps (8 used as train data and two as test data). As a first step in creating a model, the number of layers and epochs was initially small, focused on creating results fast without worrying about accuracy and loss. All images were also resized from their original size to (64,64) to guarantee a faster process. Later, these parameters were optimized based on the results obtained from this and the following models.

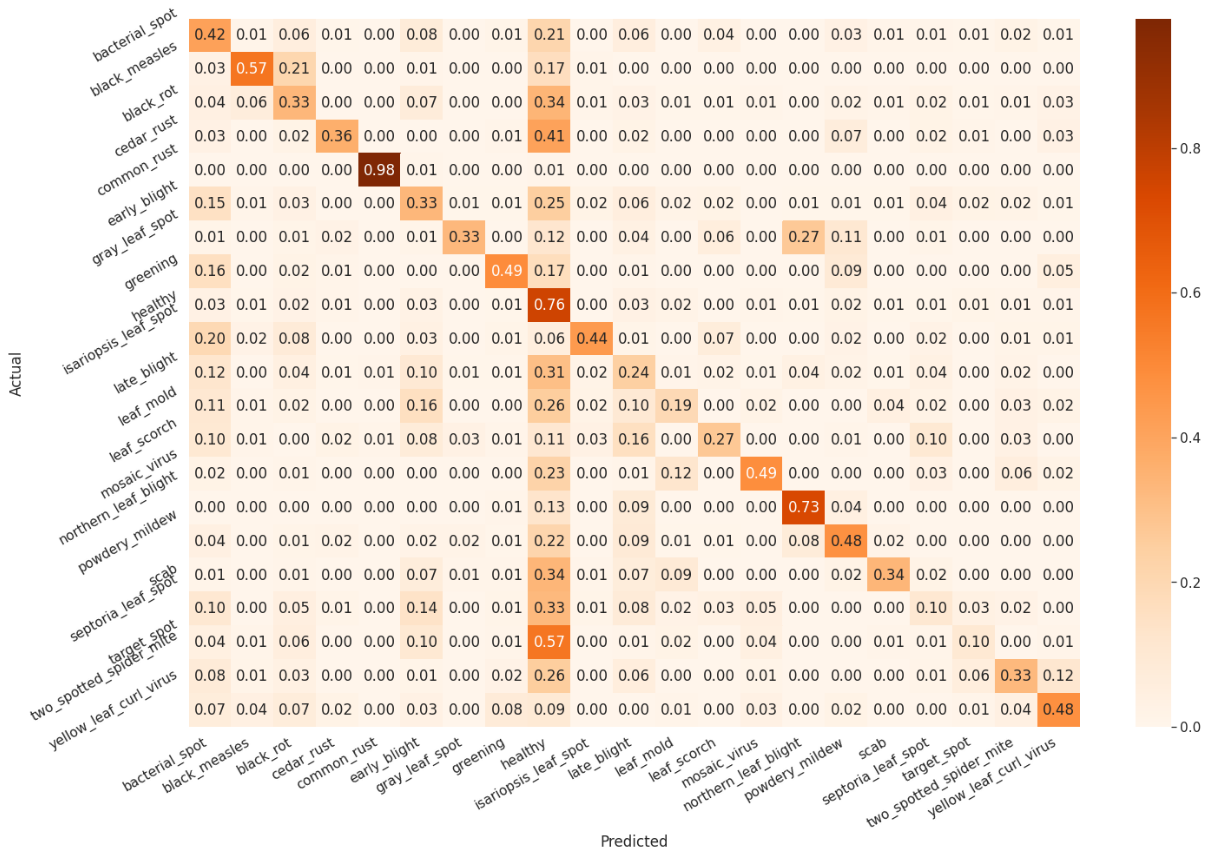
Essential parameters calculated after the model execution, such as precision, recall and f1-score, are represented in **Figure 11**.



```
              precision    recall  f1-score   support

    accuracy                           0.49      3485
   macro avg       0.51      0.42      0.44      3485
weighted avg       0.50      0.49      0.47      3485
```

Figure 11 - Summary of the baseline model performance metrics.

Here the accuracy was only 48.95%, and the loss was 2.53. The confusion matrix can also be seen in **Figure 12**.



	bacterial_spot	black_measles	black_rot	cedar_rust	common_rust	early_blight	gray_leaf_spot	greening	healthy	isariopsis_leaf_spot	late_blight	leaf_mold	leaf_scorch	mosaic_virus	northern_leaf_blight	powdery_mildew	scab	septoria_leaf_spot	target_spot	two_spotted_spider_mite	yellow_leaf_curl_virus
bacterial_spot	0.42	0.01	0.06	0.01	0.00	0.08	0.00	0.01	0.21	0.00	0.06	0.00	0.04	0.00	0.00	0.03	0.01	0.01	0.01	0.02	0.01
black_measles	0.03	0.57	0.21	0.00	0.00	0.01	0.00	0.00	0.17	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
black_rot	0.04	0.06	0.33	0.00	0.00	0.07	0.00	0.00	0.34	0.01	0.03	0.01	0.01	0.01	0.00	0.02	0.01	0.02	0.01	0.01	0.03
cedar_rust	0.03	0.00	0.02	0.36	0.00	0.00	0.00	0.01	0.41	0.00	0.02	0.00	0.00	0.00	0.00	0.07	0.00	0.02	0.01	0.00	0.03
common_rust	0.00	0.00	0.00	0.00	0.98	0.01	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
early_blight	0.15	0.01	0.03	0.00	0.00	0.33	0.01	0.01	0.25	0.02	0.06	0.02	0.02	0.00	0.01	0.01	0.01	0.04	0.02	0.02	0.01
gray_leaf_spot	0.01	0.00	0.01	0.02	0.00	0.01	0.33	0.00	0.12	0.00	0.04	0.00	0.06	0.00	0.27	0.11	0.00	0.01	0.00	0.00	0.00
greening	0.16	0.00	0.02	0.01	0.00	0.00	0.00	0.49	0.17	0.00	0.01	0.00	0.00	0.00	0.00	0.09	0.00	0.00	0.00	0.00	0.05
healthy	0.03	0.01	0.02	0.01	0.00	0.03	0.00	0.01	0.76	0.00	0.03	0.02	0.00	0.01	0.01	0.02	0.01	0.01	0.01	0.01	0.01
isariopsis_leaf_spot	0.20	0.02	0.08	0.00	0.00	0.03	0.00	0.01	0.06	0.44	0.01	0.00	0.07	0.00	0.00	0.02	0.00	0.02	0.00	0.01	0.01
late_blight	0.12	0.00	0.04	0.01	0.01	0.10	0.01	0.01	0.31	0.02	0.24	0.01	0.02	0.01	0.04	0.02	0.01	0.04	0.00	0.02	0.00
leaf_mold	0.11	0.01	0.02	0.00	0.00	0.16	0.00	0.00	0.26	0.02	0.10	0.19	0.00	0.02	0.00	0.00	0.04	0.02	0.00	0.03	0.02
leaf_scorch	0.10	0.01	0.00	0.02	0.01	0.08	0.03	0.01	0.11	0.03	0.16	0.00	0.27	0.00	0.00	0.01	0.00	0.10	0.00	0.03	0.00
mosaic_virus	0.02	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.23	0.00	0.01	0.12	0.00	0.49	0.00	0.00	0.00	0.03	0.00	0.06	0.02
northern_leaf_blight	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.13	0.00	0.09	0.00	0.00	0.00	0.73	0.04	0.00	0.00	0.00	0.00	0.00
powdery_mildew	0.04	0.00	0.01	0.02	0.00	0.02	0.02	0.01	0.22	0.00	0.09	0.01	0.01	0.00	0.08	0.48	0.02	0.00	0.00	0.00	0.00
scab	0.01	0.00	0.01	0.00	0.00	0.07	0.01	0.01	0.34	0.01	0.07	0.09	0.00	0.00	0.00	0.02	0.34	0.02	0.00	0.00	0.00
septoria_leaf_spot	0.10	0.00	0.05	0.01	0.00	0.14	0.00	0.01	0.33	0.01	0.08	0.02	0.03	0.05	0.00	0.00	0.00	0.10	0.03	0.02	0.00
target_spot	0.04	0.01	0.06	0.00	0.00	0.10	0.00	0.01	0.57	0.00	0.01	0.02	0.00	0.04	0.00	0.00	0.01	0.01	0.10	0.00	0.01
two_spotted_spider_mite	0.08	0.01	0.03	0.00	0.00	0.01	0.00	0.02	0.26	0.00	0.06	0.00	0.00	0.01	0.00	0.00	0.00	0.01	0.06	0.33	0.12
yellow_leaf_curl_virus	0.07	0.04	0.07	0.02	0.00	0.03	0.00	0.08	0.09	0.00	0.00	0.01	0.00	0.03	0.00	0.02	0.00	0.00	0.01	0.04	0.48

Figure 12 - Confusion matrix of baseline.

From this confusion matrix, it is possible to observe that the diagonal, the part of the matrix that should have bigger numbers and, therefore, darker colours, does not match the expectation. Besides that, the column corresponding to the healthy category has higher values and, consequently, darker colours than the other columns. This means that the model thinks that the images being fed to it are mostly healthy, even if they are not. Considering this and the distribution of data from EDA, it is possible to see that the label corresponding to the healthy plants has many more images than every other label. Therefore, the model is biased towards this label, which was fixed in the following steps (normalization).

On the other hand, it is also possible to observe that some labels in the confusion matrix lack predictions. In the distribution of the images through the labels, these categories have fewer images than more common labels. This also means the model was not properly trained for that label due to insufficient images.

Both these issues were uncovered because, for the model under test, only the diseases—not the plant species—were considered when dividing the labels.

The last step in testing the model was to check if it could correctly predict the disease of an image it had never seen before. For this, an image of a tomato leaf with early blight was employed. As seen in **Figure 13**, the model predicted incorrectly, which was expected from the previous level of accuracy.

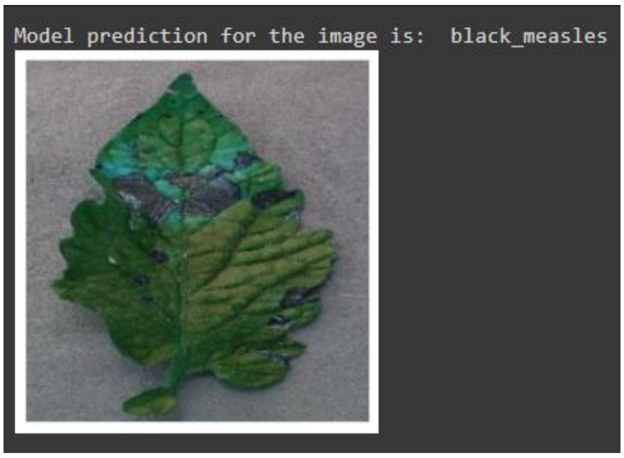

```
Model prediction for the image is:  black_measles
```

Figure 13 - Prediction of the baseline model using a plant diseased with early blight.

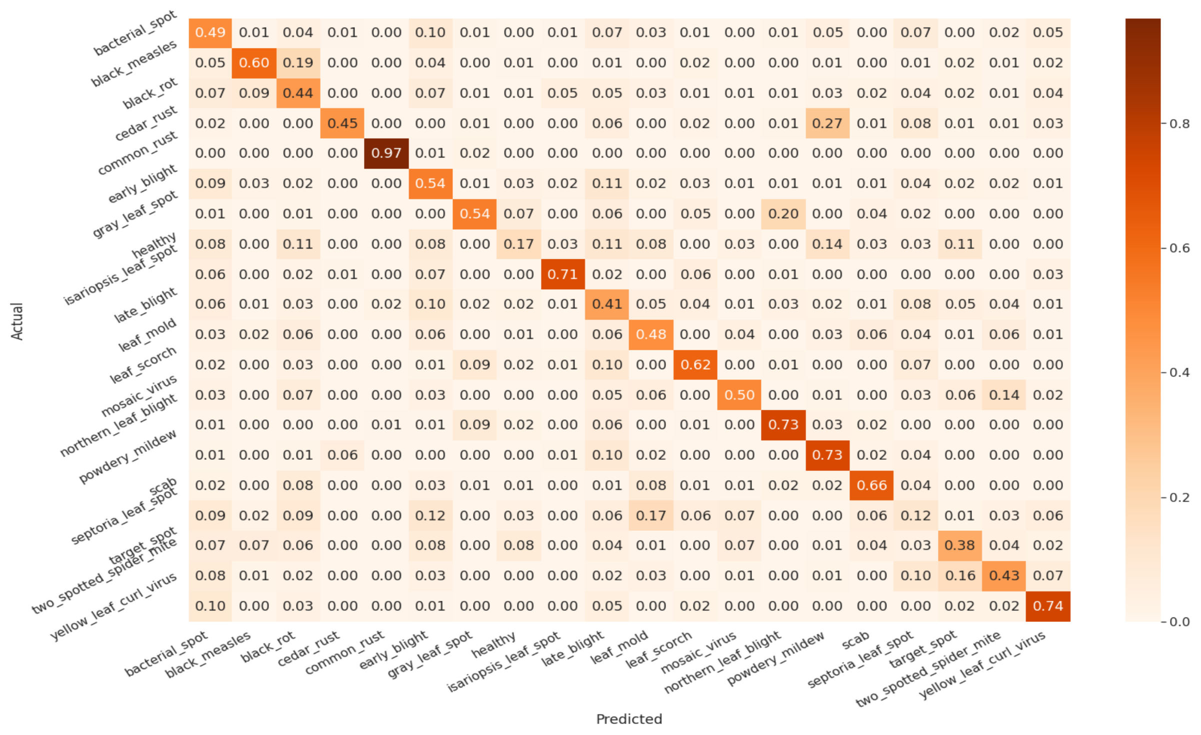
3.9. Normalization

The normalization's main objective was to decrease discrepancies between the amounts of data throughout the labels to achieve better accuracy (seen in the confusion matrices).

During the EDA, some plants appear only to have one correspondent folder, either healthy or just one disease. Thus, since the model needs a large amount of data on each label to identify it correctly, these folders were eliminated during the data loading process. These included blueberry, orange, raspberry, soybean and squash. Afterwards, it was possible to increase accuracy slightly to 51% by executing the same model from the baseline. However, the model continued to be biased towards the healthy category due to the high number of images associated with this label. Thus, a solution to increase accuracy was to decrease the number of healthy images without compromising the variety of species in the healthy category.

After utilizing these normalization strategies on the previously created baseline, a 53% of accuracy was reached.

After analyzing the matrix (**Figure 14**), it is then possible to conclude that, without improving the layers of the model or any other parameter, these two steps of normalization have helped improve the accuracy, and the model has become less biased towards the healthy label. Additionally, it was gathered that in the main diagonal (overall much darker), the healthy label has a slightly lower frequency (0.17) than the other labels (0.6). This can be explained by the number of images of healthy leaves used being lower than what was supposed to be to have an optimized model. After the bias was countered, the following models were expected to increase in accuracy.



	bacterial_spot	black_measles	black_rot	cedar_rust	common_rust	early_blight	gray_leaf_spot	healthy	isariopsis_leaf_spot	late_blight	leaf_mold	leaf_scorch	mosaic_virus	northern_leaf_blight	powdery_mildew	scab	septoria_leaf_spot	target_spot	two_spotted_spider_mite	yellow_leaf_curl_virus
bacterial_spot	0.49	0.01	0.04	0.01	0.00	0.10	0.01	0.00	0.01	0.07	0.03	0.01	0.00	0.01	0.05	0.00	0.07	0.00	0.02	0.05
black_measles	0.05	0.60	0.19	0.00	0.00	0.04	0.00	0.01	0.00	0.01	0.00	0.02	0.00	0.00	0.01	0.00	0.01	0.02	0.01	0.02
black_rot	0.07	0.09	0.44	0.00	0.00	0.07	0.01	0.01	0.05	0.05	0.03	0.01	0.01	0.01	0.03	0.02	0.04	0.02	0.01	0.04
cedar_rust	0.02	0.00	0.00	0.45	0.00	0.00	0.01	0.00	0.00	0.06	0.00	0.02	0.00	0.01	0.27	0.01	0.08	0.01	0.01	0.03
common_rust	0.00	0.00	0.00	0.00	0.97	0.01	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
early_blight	0.09	0.03	0.02	0.00	0.00	0.54	0.01	0.03	0.02	0.11	0.02	0.03	0.01	0.01	0.01	0.01	0.04	0.02	0.02	0.01
gray_leaf_spot	0.01	0.00	0.01	0.00	0.00	0.00	0.54	0.07	0.00	0.06	0.00	0.05	0.00	0.20	0.00	0.04	0.02	0.00	0.00	0.00
healthy	0.08	0.00	0.11	0.00	0.00	0.08	0.00	0.17	0.03	0.11	0.08	0.00	0.03	0.00	0.14	0.03	0.03	0.11	0.00	0.00
isariopsis_leaf_spot	0.06	0.00	0.02	0.01	0.00	0.07	0.00	0.00	0.71	0.02	0.00	0.06	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.03
late_blight	0.06	0.01	0.03	0.00	0.02	0.10	0.02	0.02	0.01	0.41	0.05	0.04	0.01	0.03	0.02	0.01	0.08	0.05	0.04	0.01
leaf_mold	0.03	0.02	0.06	0.00	0.00	0.06	0.00	0.01	0.00	0.06	0.48	0.00	0.04	0.00	0.03	0.06	0.04	0.01	0.06	0.01
leaf_scorch	0.02	0.00	0.03	0.00	0.00	0.01	0.09	0.02	0.01	0.10	0.00	0.62	0.00	0.01	0.00	0.00	0.07	0.00	0.00	0.00
mosaic_virus	0.03	0.00	0.07	0.00	0.00	0.03	0.00	0.00	0.00	0.05	0.06	0.00	0.50	0.00	0.01	0.00	0.03	0.06	0.14	0.02
northern_leaf_blight	0.01	0.00	0.00	0.00	0.01	0.01	0.09	0.02	0.00	0.06	0.00	0.01	0.00	0.73	0.03	0.02	0.00	0.00	0.00	0.00
powdery_mildew	0.01	0.00	0.01	0.06	0.00	0.00	0.00	0.00	0.01	0.10	0.02	0.00	0.00	0.00	0.73	0.02	0.04	0.00	0.00	0.00
scab	0.02	0.00	0.08	0.00	0.00	0.03	0.01	0.01	0.00	0.01	0.08	0.01	0.01	0.02	0.02	0.66	0.04	0.00	0.00	0.00
septoria_leaf_spot	0.09	0.02	0.09	0.00	0.00	0.12	0.00	0.03	0.00	0.06	0.17	0.06	0.07	0.00	0.00	0.06	0.12	0.01	0.03	0.06
target_spot	0.07	0.07	0.06	0.00	0.00	0.08	0.00	0.08	0.00	0.04	0.01	0.00	0.07	0.00	0.01	0.04	0.03	0.38	0.04	0.02
two_spotted_spider_mite	0.08	0.01	0.02	0.00	0.00	0.03	0.00	0.00	0.00	0.02	0.03	0.00	0.01	0.00	0.01	0.00	0.10	0.16	0.43	0.07
yellow_leaf_curl_virus	0.10	0.00	0.03	0.00	0.00	0.01	0.00	0.00	0.00	0.05	0.00	0.02	0.00	0.00	0.00	0.00	0.00	0.02	0.02	0.74

Figure 14 - Confusion matrix after normalization.

3.10. Optimization Strategies

The next step was to improve the model by adjusting various parameters. Therefore, multiple strategies were applied and analyzed for the best outcome. The following paragraphs present a summary of the applied strategies, and the conclusion for each.

The number of batches that the data was divided into could be altered. This had to be balanced because the smaller the number, the better the results would be. However, with fewer batches, the bigger the risk was of reaching the maximum of the allowed RAM in Colab. With this in mind, the number of batches was settled at 10.

Then the structure of the model could be improved. For this, different layers were created, different structures were experimented with, and the results were compared. The structure that was found with the best results consisted of 6 convolutional layers. Another parameter continuously adjusted was the number of filters on the convolutional layer, which settled at 128.

The size of the images could also be adjusted, and the sizes tried were (224,224), (128,128), (64, 64). Here it was seen that the bigger the image, the longer it took to train the model, which could, at times, prevent the model's training altogether because of limitations imposed by Colab in terms of allowed execution time. Besides, the bigger the images, the higher the risk of reaching the maximum allowed RAM. Additionally, the results would not get significantly better for bigger images, so the size of the images was settled at (64,64).

Another parameter was the number of epochs. Here the same problem of the time of execution of Colab was reencountered, so there needed to be a balance between the minimum number of epochs needed to improve the results and the maximum number of epochs allowed without Colab shutting down. The best number of epochs was found at 5 epochs.

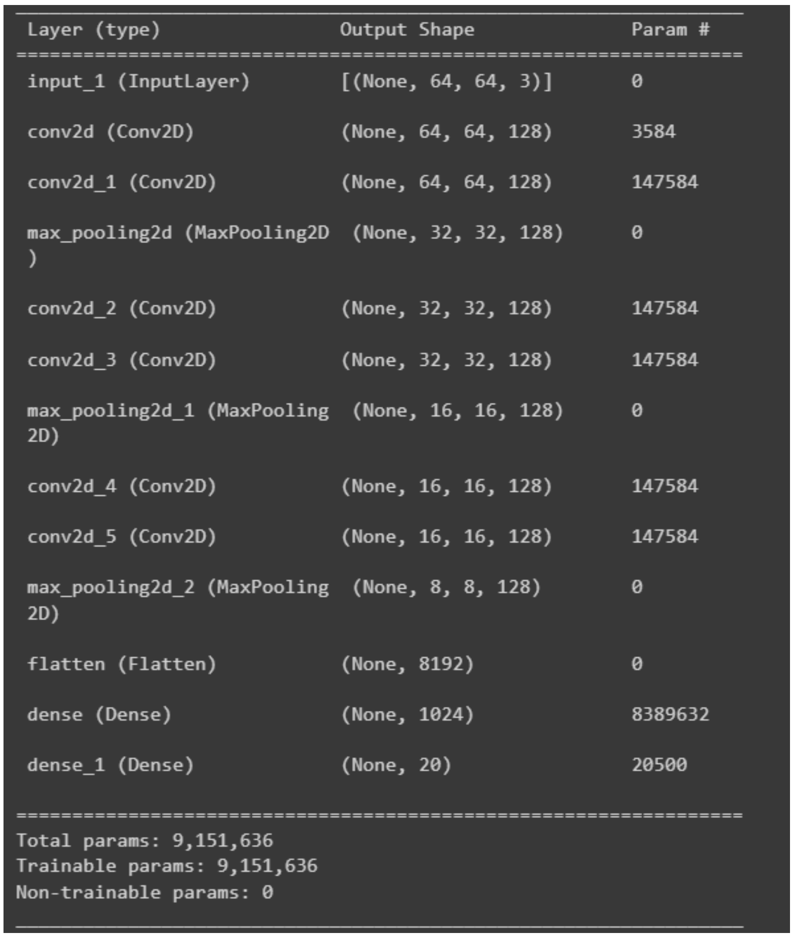
Another parameter that had to be optimized was the proportion of healthy images in each used folder. The bigger the number of images, the more the model was trained towards that label, so it was essential to ensure that this label had approximately the same amount of data as the other labels. If too many images of healthy plants existed, then the model would become biased towards this label. If too few images existed, the model would not be adequately trained for this label. This parameter depends heavily on the number of labels in the data and the amount of data in each of these labels. For the data that was here being used, only de valid folder, due to the limitations imposed by Colab, both in RAM and execution time, only 10% of each folder of healthy was used.

Lastly, the number of times the whole train batch was executed through the model is another parameter that could be adjusted. This means the model is trained for all the train batches once and then a specific number of times more, even if these are always the same images. This was done with a while loop. The same problem was encountered here, as the number of repetitions could take too much time, and the Colab would shut down. Considering this, the ideal number of repetitions in the while loop was 6.

One last parameter that needs to be considered is the data being used. The whole data in the dataset is comprised of approximately 87K images. However, these are too many images for the Colab to handle. Ideally, all images should have been used, but only a portion was used, corresponding to 20% of the data. If it were possible to use the whole dataset, better results would have been obtained.

3.11. Best Model

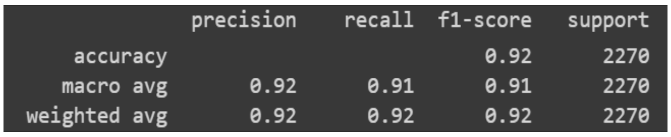
The best model was constructed, considering the optimization parameters previously presented. The data (only the one contained in the valid folder) was firstly divided into 10 batches (8 used for training and 2 for testing), and the images were resized to (64,64). Then the model was constructed following the structure shown in **Figure 15**.



```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 64, 64, 3)]       0

 conv2d (Conv2D)             (None, 64, 64, 128)       3584

 conv2d_1 (Conv2D)           (None, 64, 64, 128)       147584

 max_pooling2d (MaxPooling2D  (None, 32, 32, 128)      0
 )

 conv2d_2 (Conv2D)           (None, 32, 32, 128)       147584

 conv2d_3 (Conv2D)           (None, 32, 32, 128)       147584

 max_pooling2d_1 (MaxPooling  (None, 16, 16, 128)      0
 2D)

 conv2d_4 (Conv2D)           (None, 16, 16, 128)       147584

 conv2d_5 (Conv2D)           (None, 16, 16, 128)       147584

 max_pooling2d_2 (MaxPooling  (None, 8, 8, 128)        0
 2D)

 flatten (Flatten)           (None, 8192)              0

 dense (Dense)               (None, 1024)              8389632

 dense_1 (Dense)             (None, 20)                20500

=================================================================
Total params: 9,151,636
Trainable params: 9,151,636
Non-trainable params: 0
_________________________________________________________________
```

Figure 15 - Summary of the best model.

For the model fitting, a while loop was used to run all the training batches 6 times, where each batch was randomly chosen for training. Here, the achieved accuracy was 91.7%, with a loss of 0.47. Other performance parameters can be seen in **Figure 16**.



```
              precision    recall  f1-score   support
    accuracy                           0.92      2270
   macro avg       0.92      0.91      0.91      2270
weighted avg       0.92      0.92      0.92      2270
```

Figure 16 - Summary of the best model performance metrics.

A confusion matrix was also created to analyze the results better, as shown in **Figure 17**.

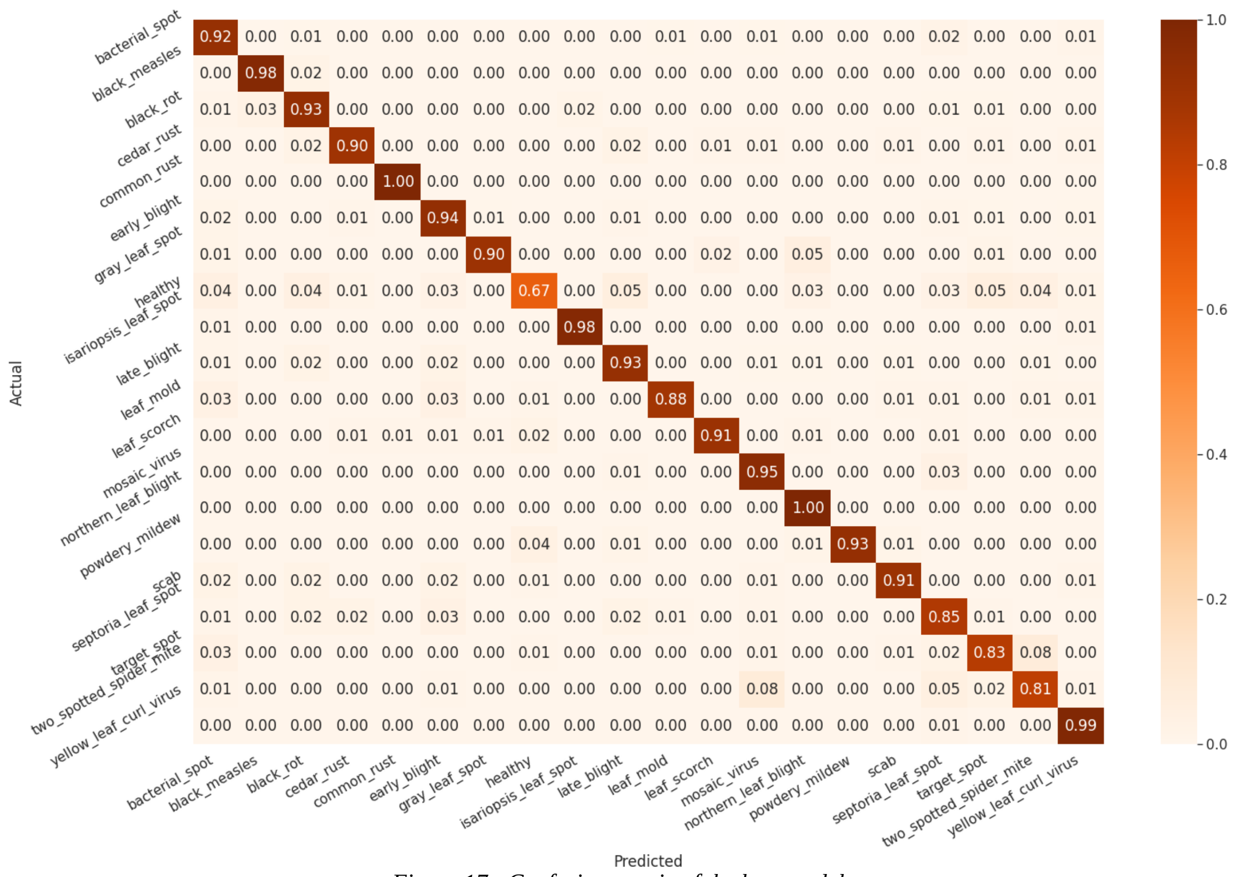
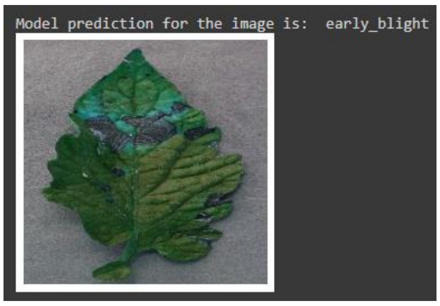


Figure 17 - Confusion matrix of the best model.

From the performance parameters and the confusion matrix, the main conclusion was that this is an excellent model that can predict labels with very high accuracy. From the matrix, it is possible to observe a strong diagonal, which means that the model is predicting most of the labels correctly. However, some faults in the model are found, such as in the healthy leaf mold, Septoria leaf spot, target spot and two-spotted spider mite, where the percentage is below 90%. To improve the performance of this model, more images corresponding to those specific labels should be introduced in the model's training. Specifically, the problem could easily be solved for the healthy label by increasing the proportion of images included when loading the data. Lastly, the model was tested to see if it could predict the disease of an example of an image it had never seen before. For this, the image of a tomato leaf with early blight was used, and, as can be seen in **Figure 18**, the model predicted the label correctly.



```
Model prediction for the image is:  early_blight
```

Figure 18 – Correct prediction of the best model using a plant diseased with early blight.

4. Final Thoughts

With this project, it was possible to present a potential implementation of a business model and its prototype through an app called PlantDoc. This app aimed to classify diseases in plants through a picture of leaves. In the Business Model, the Unique Value Proposition was presented, as well as the cost structures, revenue streams, prospective customers, and market and product detailed information.

The algorithm for the prototype was implemented using deep learning, and the steps involved in developing this algorithm are presented throughout the report. This consisted of an Exploratory Data Analysis, data transformation and normalization, followed by the creation of a Baseline. This first model was then optimized to achieve the final model, which presented fascinating results with an accuracy of 91%.

Throughout the project, the main obstacles observed were Google Colab not collaborating with the utilized data due to the limited amount of RAM available. This limitation restricted the number of images that could train the model and increased the complexity of the model's structure. Furthermore, the limited execution time also prevented further training of the models since these would take too much time to process through the necessary amount of time. In sum, these restrictions prevented the use of the whole data and the creation of an even better model.

This way, to improve the algorithm and achieve better results, the structure of the model would not need to be changed, but the used amount of data should be even larger. Better results are essential for applying the algorithm since incorrectly categorizing a plant could lead to killing the plant of a plant enthusiast or even taking a hit on the production of a big agricultural company since a whole crop could be ruined.

Overall, to finalize the prototype and create a fully integrated solution, the following steps would be to develop the app that implemented this algorithm, combined with a structure to generate tips and suggestions of products that could be purchased through the app.

APPENDIX

Table 1 | Examples of plant disease commonly found among commonly planted crops.

Disease denomination	**Characterization**	
Black Measles	Fungi infection recognized by the round brown-coloured spots and/ or lesions.	**Fungi**
Black Rot	Fungi disease recognized by the black spots in plants' leaves.	
Cedar Rust	Fungi infection, also known as *Gymnosporangium juniperi-virginianae*, can be easily identified by the appearance of pale-yellow pinhead-sized spots on the upper surface of the leaves shortly after bloom.	
Early Blight	Fungi infection characterized by small brown spots with concentric rings that form a "bull's eye" pattern.	
Gray Leaf Spot	Fungi infection that originates spots on leaves with round or oval shapes, tan colour and dark brown border.	
Isariopsis Leaf Spot	Also known as Leaf Blight is a fungi infection detected by the appearance of scattered, somewhat angular, purple-brown spots on the upper leaf surface.	
Leaf Mold	Fungal infection that consists off circular, powdery white spots.	
Late Blight	*Phytophthora infestans* is an oomycete or fungus-like infection that leads to the appearing of large brown blotches with a green-gray edge.	
Northern Leaf Blight	Fungal infection that leads to the appearance of canoe-shaped lesions that are initially gray-green, but can get darker.	
Powdery Mildew	Fungal disease that is recognized by circular, powdery white spots that cover the upper part of the leaves.	
Scab	Fungal infection that causes spotting and drop.	
Septoria Leaf Spot	One of two common fungal diseases consisting of spots that first appear at the base of affected plants.	
Target Spot	Disease is caused by the fungus *Alternaria solani*. Its main symptoms start with the appearing of small circular to oval dark brown to black spots on leaves that latter enlarge, becoming oval to angular, and are normally confined within the main veins of the leaflets.	
Greening	Bacterial (*Candidatus Liberibacter*) infection that leads to yellow spotting and veins.	**Bacterial**
Mosaic Virus	Viral infection that can easily be spread to other nearby plants. It is identified by yellow, white or green stripes/ streaks/ spots on foliage, dark green blisters…	**Viral**
Yellow Leaf Curl Virus	Viral infection transmitted from diseased to healthy plants by silverleaf whitefly. Diseased plants have mall leaves that become yellow between the veins or curl upwards and towards the middle of the leaf.	
Common Rust	Originated from rust spores and is mainly identified by the appearance of brown pustules.	**Other**
Leaf Scorch	Not an infection and occurs by unfavorable environmental conditions. It is recognized for the appearance of a brown coloration blotches.	
Two-Spotted Spider Mite	Occurs due to the presence of two-spotted spider mite, *Tetranychus urticae*, is a type of arachnid, related to insects. It leads mostly to tiny white or yellow spots.	

References

Agriculture, B. (2022, Oct. 14). *Top 10 Largest Agricultural Companies in the World 2020, Top Agriculture Companies*. Retrieved from BizVibe: https://blog.bizvibe.com/blog/largest-agricultural-companies

Boss, S. (2022, August 02). *4 Countries That Produce the Most Food*. Retrieved from Investopedia: https://www.investopedia.com/articles/investing/100615/4-countries-produce-most-food.asp

Sarkar, S. G. (2020). Rice Leaf Diseases Classification Using CNN With Transfer Learning. *IEEE Calcutta Conference (CALCON)*, pp. 230-236, doi: 10.1109/CALCON49167.2020.9106423.

Saud, D. (2022, October 11). *Top 5 Biggest Agriculture Companies in The World 2022!* Retrieved from foloi3: https://dynamics.folio3.com/blog/top-agriculture-companies/