Utilization of Pyphtopatholigcal data for Pathogen Forecasting

Bernstein, Joshua[a]

[a]*Stevens Institute of Technology, Hoboken, NJ, USA*
[b]*I pledge my honor that I have abided by the Stevens Honor System.*

Abstract

Plant pathogens are an occurrence that many may never think of, however, they have a tremendous impact on the production of food worldwide. Hence, systems that are able to accurately predict the spread of these pathogens are extremely valuable and important. This paper will go over the history of the study of plant pathogens, called plant pathology or phytopathology, as well as the forecasting systems that would be as a result of these innovations and discoveries. Moreover, these forecasting systems are primarily derived from very few variables using linear regression, those mostly relating to the ideal conditions of a specific pathogen. Due to the fact that it relies on such few variables, there is the potential for incorrect predictions if there is an issue with external equipment or calculation. On the other hand, the fact that it does only need few variables, and that they are typically readily available, such as temperature or humidity, allows these systems to be made just about anywhere. Reliable forecasting systems are able to predict and hopefully prevent the spread of plant pathogens, mitigating a common problem with food production that causes hundreds of billions of dollars each year. This paper will seek to create such a forecasting system, based on Moonachie, NJ, that will model the spread of disease pathogen. This will be modeled using the matplot library in Python.

Keywords: Phtyopathology, Forecasting Systems, Matplot

Highlights

- Summarized the history of plant pathology(phytopatholgy) and disease forecasting
- The derivation of disease forecasting models
- Applications of plant disease forecasting models
- Disease Forecasting system of Common corn rust
- Python graphing of forecasting system
- Climate variables over designated time interval

1. Introduction

1.1. History

The early beginnings of plant pathology is traced back to the times of ancient Greece with Theophrastus. Theophrastus lived from approximately 371 to 287 BC and wrote 10 treatises on plants and trees entitled "Enquiry into Plants"[9]. Of the nine that survive today, there are the first known mentions of diseases affecting plants, as well as speculations to their causes [9]. After Theophrastus, there is a great gap in time between significant discoveries, the most impactful being the discovery of bacteria in 1683 by Antonie Van Leeuwenhoek, using the compound microscope developed a few years earlier. Half a century later, in 1729, fungi spores would be observed by Pier Antonio Micheli, as said by Ashrafuzzaman's 2013 leccture "History of Plant Pathology"[3]. Thus, by 1729, two of the three main plant pathogens have been discovered (bacteria, viruses, fungi). The next major innovation would come in the early to mid 1800s, with the creation of lime sulfur, the first known synthetic pesticide, that was used as plant pathogen panacea until the 1940s. This is covered in Ainsworth's "Introduction to the History of Plant Pathology"[2].

After that, the next major innovation is the publishing of Germ Theory by Louis Pasteur in 1867, which helped to entrench the idea that bacteria and other microorganisms are responsible for many diseases humans, animals, and plants face. Very soon after that, during the 1880s and 1890s, viral plant diseases were being discovered and catalogued. This was covered in Agrios's "Plant Pathology"[1]. It was at this time that a large amount of the knowledge we utilize today regarding pathogen cause and effects was documented. Moreover, another significant innovation was the discovery that mycoplasma bacteria was pathogenic to plants and trees in 1967[2]. During the mid 1900s, international efforts were made to combat plant pathogens in order to protect food production worldwide. In 1951, the European and Mediterranean Plant Protection Organization (EPPO) was established to predict and hopefully prevent plant pathogens. The organization consisted of 32 member countries from all over Europe, Africa, and Asia. Today, the data gathered via plant pathology is used around the globe to create forecasting systems to mitigate the spread and damage done by plant pathogens.

In terms of plant disease forecasting, this did not occur until 1926, and it was in the Netherlands to combat a potato pathogen sweeping Europe, as said by Charaya et al. in "Plant Disease Forecasting: Past Practices to Emerging Technologies"[5]. Traditionally before this, plant diseases were controlled through the scheduled application of fungal and pesticides, en lieu of a dynamic forecasting system. This traditional system, while effective, did not take into account the effect of the pesticides/fungicides on humans and their environment [4]. This is opposed to a forecasting model, which implements pathogen halting measures sparingly, and only when an outbreak is noticed or predicted. Within the last decade, specifically, many forecasting models have become web-based. Using information freely available on the worldwide web such as weather, humidity, and temperature to create accurate forecasting systems, and hosting them online so anyone may see them. Such an idea was postured by Fernandes et al. in 2011[6], and was quickly adopted by organizations such as USPEST[10]. These web-based models are now an invaluable tool to fighting the spread of plant pathogens.

1.2. Model Derivation

Models for plant disease forecasting systems are primarily derived from the specific conditions of the pathogen. Due to the variability of a pathogen's preferred temperature, humidity, or climate from pathogen to pathogen, most forecasting systems are used to predict the spread of a particular pathogen. Therefore, the model typically used is one of linear regression (y = mx + b), where only a few variables, in this case that of the surrounding environment, is used to predict the spread of a particular pathogen. For example, a forecasting system may be based solely on the the closeness in temperature of the surrounding area to that of a pathogen's preferred temperature. Meaning that the closer the area's temperature is to the ideal, the higher risk the model predicts of a pathogen's potential spread[5]. More advanced models will use multiple variables to increase the overall accuracy, such as humidity or weather.

1.3. Model Properties

Due to the simplicity of the model, it has very few properties. It can essentially only weight the percentage risk a pathogen may appear in a given location. It uses very few variables, primarily just temperature and humidity (although depending on the pathogen, others may be used) to weight the risk percentage in the model. A very good source for these risk models is USPEST.org, a website maintained by the University of Oregon State, that monitors the infection risk of various plant pathogens across the country[10]. Thus, an unfortunate downside to this model is that it is very sensitive to inaccurate data, as an incorrect reading for temperature or humidity could cause for a radically incorrect risk percentage. On the other hand, because it relies on such few variables, it is incredibly easy to apply a model essentially anywhere with very little external equipment needed. Since most models utilize data that is freely available in most places, it is very easy to adapt a model to anywhere the pathogen might spread to. Furthermore, since these models are not dependent on size, they are useful to both farmers with acres upon acres of farmland and farmers with only an acre or two.

1.4. Model Applications

Plant disease forecasting systems have tremendously important applications to the production of food. If forecasting systems are able to reliably predict where and when a pathogen will begin to spread, it can save an industry that is vital to society and our species. Plant pathogens worldwide are responsible for about 30 percent of all lost food production, amounting to over 200 billion dollars each year [5]. Therefore, being able to create a reliable and accurate disease forecasting system has amazing applications for the agriculture industry.

1.5. Experiment Objectives

This paper will seek to create a disease forecasting system. In particular, this paper will create a forecasting system for Common Rust in and around Moonachie, New Jersey, USA in the month of June. Common rust is a fungal disease primarily affecting corn species, and with several farms nearby, being able to predict the spread of common rust could be incredibly useful. Going into this experiment, it is predicted that the model should find a moderate risk of rust infection, as the late spring weather is within the approximate ideal conditions of this fungus. This pathogen flourishes typically with temperatures between 60

and 76 degrees Fahrenheit, with a a relative humidity of 95 percent or higher, and a dew point of about 60 degrees Fahrenheit or higher[7]. Moreover, this fungus typically takes about 6 hours to germinate properly. This model will forecast common rust spread of June 1st through June 4th 2022, and will be created using the Python programming language.

2. Materials and Method

2.1. Materials

2.1.1. Pathogen Data

The most important material required for this experiment is the essential data about the pathogen. This includes data such as its ideal temperature, humidity, or dew point. For this experiment, this information was obtained via the University of Minnesota [7]. There is not a centralized source for this information, however, so it is recommended for other pathogens one searches for this information at institutions nearby the pathogen naturally spreads.

2.1.2. Weather Data

After that, another material required for this experiment is meteorological information. This includes the information necessary to predict the spread of the pathogen. In this case, temperature, humidity, and dew point. There are several free, online resources that can provide this information for most areas of the United States. For the purposes of this experiment, all weather-related information was supplied by WeatherUnderground[12]. Once you obtain this data, it is important to transfer it into either a csv or excel file, so it can be more easily read by the program.

2.1.3. Python Setup

Next, one needs a software to program and run python code. Like weather data, there are numerous free alternatives. For python specifically, one may download python's IDE IDLE[8], or Microsoft's Visual Studio Code (VS Code)[11]. This experiment will be done using VS Code, but the results would remain the same regardless of the coding environment. Once you have your coding environment running, you will need to install a python library that handles creating graphs. For this experiment, we are using the MatPlot library. This library can be downloaded onto your python install by navigating to your operating system's command line, and typing "pip install matplotlib". This command will automatically install the library into your python install.

2.2. Method

The first step is to isolate the ideal conditions for the pathogen you are forecasting. For common rust, these conditions include a temperature of 60-76 degrees Fahrenheit, Humidity of 95 percent or greater, and a dew point of 60 degrees or higher. Once you have these conditions, you can begin creating your program.

2.2.1. Importing Libraries

For this experiment, you will need to import two python libraries: matplot, where the tutorial download can be seen above, and csv, which comes natively with the python installation. This can be imported as seen below (note that colors do not matter and is a stylistic preference).

```
15  import matplotlib.pyplot as plt #Used to create the graph
16  import csv #Used to read data
```

2.2.2. Day Class

From there, in order to better organize the data, I created a class "day" to hold all the information from each day. This class is initalized using the data of the day, and later a time block can be added using the addTime() function. Other functionality includes the getLst() method, which returns a list of every time block with every variable entered up to that point, as well as the print() method, which is primarily used for debugging and can be safely excluded.

```
25  class day:
26      '''Used to store the information for one day'''
27      def __init__(self,name):
28          self.label = name
29          self.times = []
30      def addTime(self,tLst):
31          '''Adds a time block to the list'''
32          self.times.append(tLst)
33      def getLst(self):
34          '''Returns a list of all time blocks and their corresponding variables'''
35          return self.times
36      def print(self):
37          '''Prints out full class, used solely for debugging'''
38          print(f'Date: {self.label}')
39          for i in range(len(self.times)):
40              print(f'Time Reading #{i+1}:\t{self.times[i]}')
```

2.2.3. Formatting to CSV

To start, open the file in python, then open it using the csv library. This will automatically create a list containing the contents of each line in the file. From there, you organize the data so that you add the correct hourly information to each data. Due to how the source data of this experiment was formatted, we had to perform some unusual maneuvers to ensure our data was being stored properly in python. The crucial step is that at the end, each day object must contain a correct list of each hour block's variables.

```
55  days = []
56  with open("Days.csv",'r') as csv_file:
57      file_content = csv.reader(csv_file) #Converts the file into a list of lines
58      index = 0
59      for row in file_content:#Loops through each line
60          if index == 0:
61              inner = 0
62              for col in row:#Creates a list of days
63                  if col != "" and inner == 0:
64                      days.append(day(col[3:]))
65                  elif col != "":
66                      days.append(day(col))
67                  inner +=1
68          elif index > 1:#Adds each time block to the appropriate day
69              innerIndex = 0
70              days[0].addTime(row[0:4])
71              days[1].addTime(row[5:9])
72              days[2].addTime(row[10:14])
73              days[3].addTime(row[15:19])
74          index += 1
```

2.2.4. Creating the table of risk values

The last step is to create the table that will be modeled by the graphing software. This table is made by looping over every hour block in every day, and checking to see if the weather conditions if that hour block satisfy the ideal conditions, if they do, then we add 1 to the variable accumTime. As this variable increases, the risk percentage increases as well. The higher the accumTime, the larger the risk percentage. Conversely, the longer time the data has been outside of the ideal conditions lowers the risk percentage. After a chosen time of 20 1-hour blocks, the risk percentage is reset to 10 percent, as we have been outside of the ideal conditions for a long time at that point. At the end of each iteration of the loop, the hour count, and its associate risk percentage, are each added to a pair of parallel lists that will become the axes of the model.

```
hourCount = 0
timeLst = []
riskLst = []
for d in days:
    temporary = d.getLst()
    for col in temporary:
        temp = int(col[1])
        humidity = int(col[3])/100
        dewPoint = int(col[2])

        if isIdeal(temp,humidity,dewPoint):#If the climate is ideal
            accumTime += 1
            timeOutsideIdeal = 0
        else:# If the weather is not ideal
            timeOutsideIdeal += 1
            if timeOutsideIdeal >= 10:
                accumTime = 0
        risk = 0#Intializes the risk to 0
        if timeOutsideIdeal < 20:#If the weather has not been ideal for less than 20 hours
            if 0 <= accumTime <= 2:
                risk = 0.25
            elif 2 <= accumTime <= 6:
                risk = 0.5
            elif 6 <= accumTime <= 10:
                risk = 0.75
            else:
                risk = 0.95
        else:#If the weather has not been ideal for 20 hours or more
            risk = 0.1


        timeLst.append(hourCount)
        riskLst.append(risk)
        hourCount +=1
```

3. Calculation

As previously stated, the model for a plant disease forecasting system is incredibly simple, with simple linear regression being the only real driver behind the model. To gain the risk percentage, we simply determine if the current conditions at some time block match the ideal conditions of the pathogen, and if they do, then we compare with previous time intervals. If earlier time intervals also housed these ideal conditions, then the risk percentage may rise accordingly.

3.1. Risk Percentage

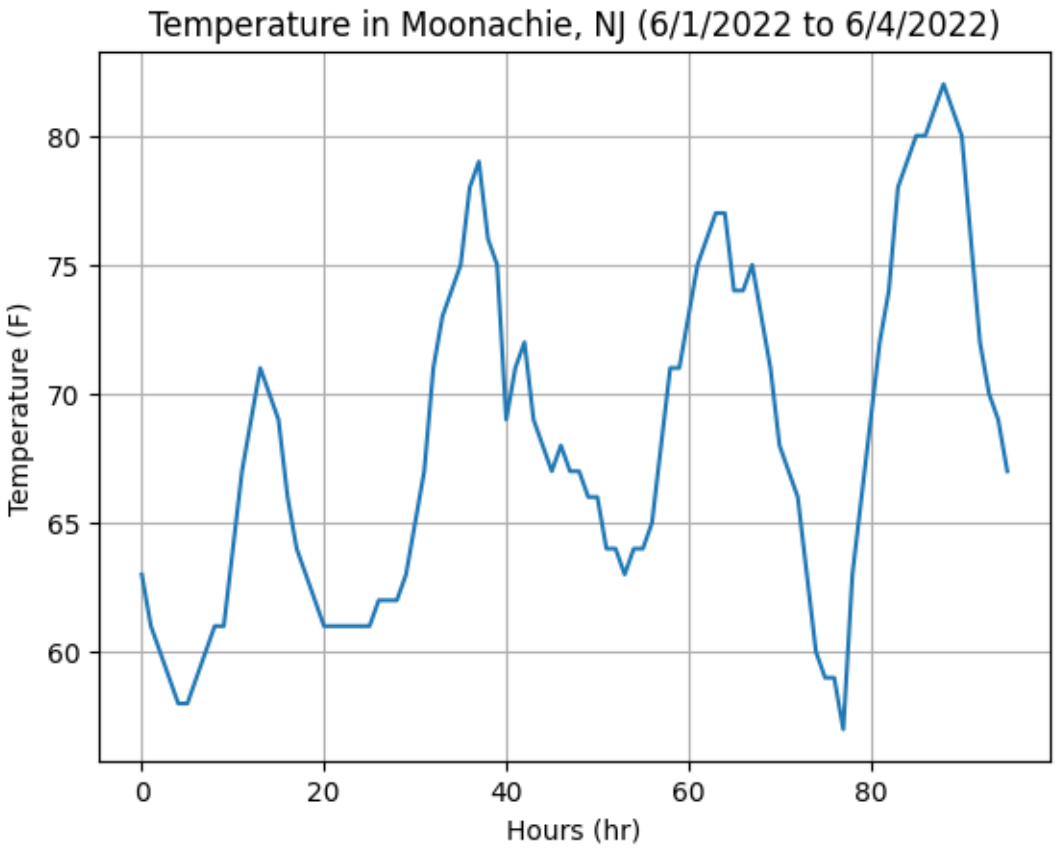
To calculate risk percentage, we first look at the accumulated time of ideal conditions. This variable, as can be seen in the code above, denotes how long there have been ideal conditions in our experimental area. For this experiment, we have created a table that denotes the approximate risk percentage for each range of accumulated time. To start, we assume the risk percentage is at 25 percent, as we are not charting the risk percentage before this date. If the accumulated time is between 0 and 2 hours, then the risk percentage is at 25 percent. If it is between 2 and 6 hours, then the risk percentage is at about 50 percent, meaning there is now a decent likelihood of disease spread. Next, between 6 and 10 hours, there is a significant likelihood of pathogen infection and spread, and for over 10 hours straight of ideal conditions, there is a very significant chance of pathogen infection.

4. Results

For this section, all graphs and figures are shown over a time period of 4 full days (96 hours).

4.1. Temperature Graph

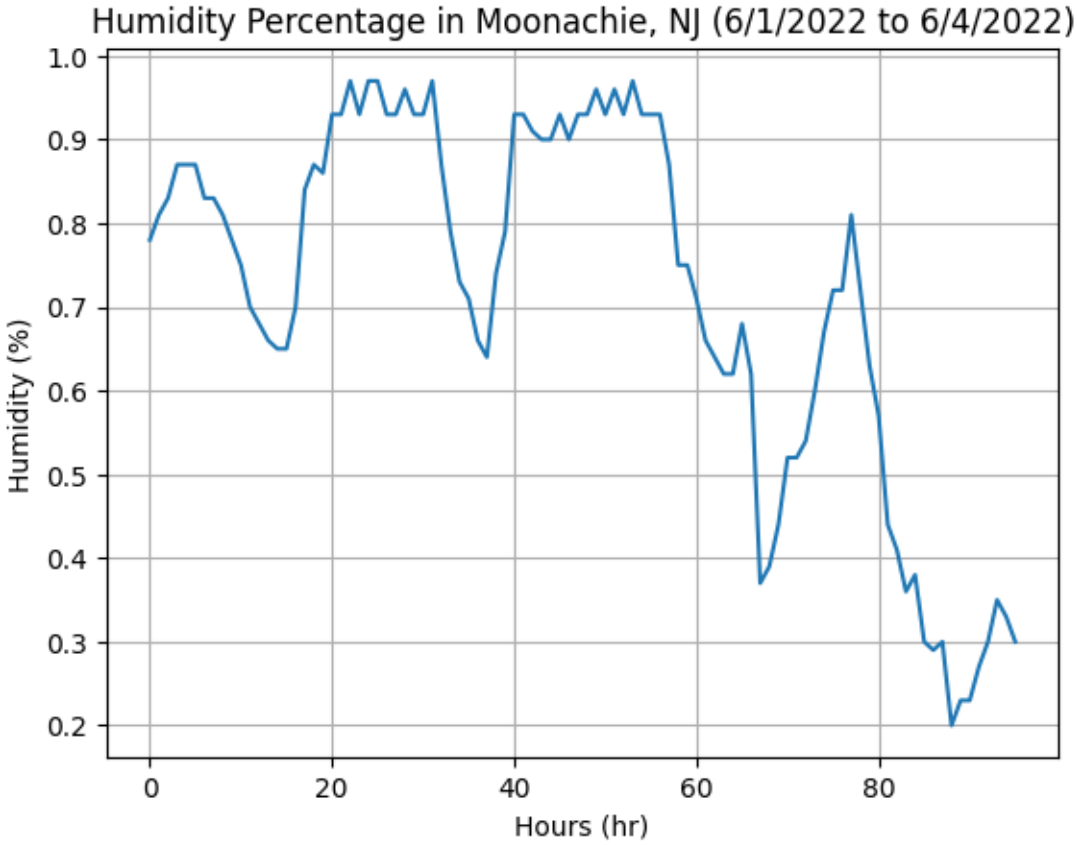
For the first figure, it is a graph of the temperature over the time interval.



As can be seen, the temperature over the time interval was almost wholly within the ideal conditions of the pathogen. With notable increases at around the 40 and 90 mark, with significant dips a the approximate 5 and 75 hour mark going below the ideal temperature range. Overall, the lowest temperature recorded was about 57 degrees, with a high point of about 88 degrees.

4.2. Humidity Graph

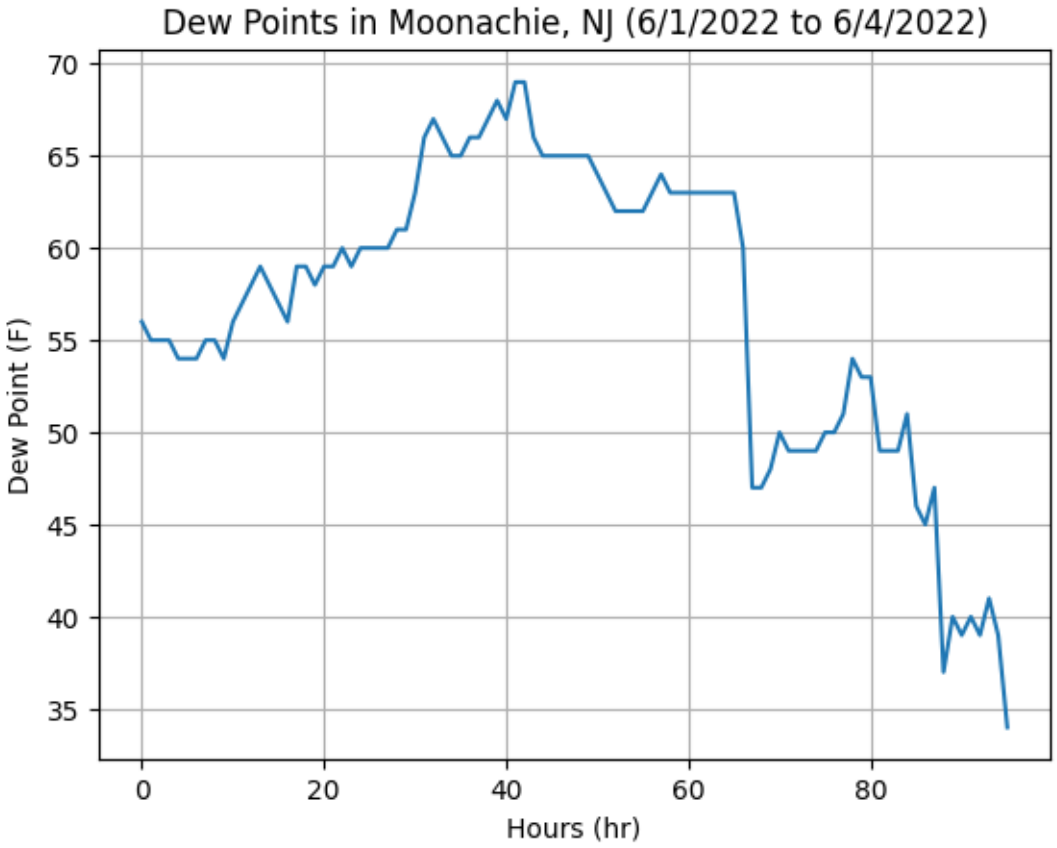
For the next figure, it is a graph of the humidity percentage over the time interval.



In stark contrast to the previous graph, only two portions were inside of the ideal conditions of the pathogen. Only from time intervals of 20-30 hours and 40 to 55 hours is the humidity within the ideal range for common rust to flourish. This graph's highest point is at a humidity percentage of about 97, and a low point of about 20 percent.

4.3. Dew Point Graph

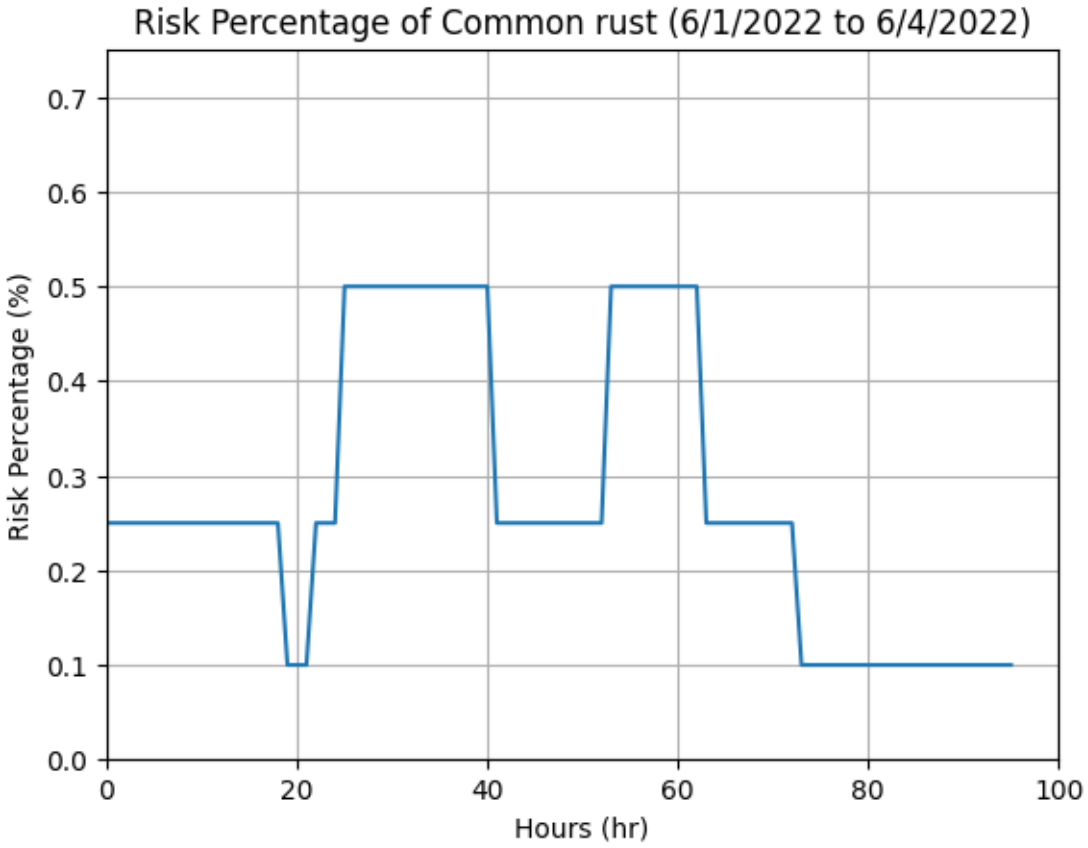
After that, this figure corresponds to the dew point (in Fahrenheit) over the recorded time interval.



Much like the humidity graph, only a small subset of this figure is within the ideal conditions of dew point for this pathogen (approximately greater than or equal to 60 degrees). This range is from approximately the 25 to 65 hour mark, with all other ranges being outside of the ideal conditions of this pathogen. This graph has a high point of about 68 degrees, and a low point of about 32 degrees.

4.4. Overall Risk Percentage Graph

Finally, the last graph showcases the result of the forecasting model, with the risk percentage charted over the entire time interval.



Throughout this graph, the risk percentage is very low, at about 25 percent, but jumps at about the 25 and 55 hour mark. Each of these jumps eventually falls back down within about 15 hours. Moreover, the risk percentage is never charted as being above 50 percent, but reaches a low of 10 percent.

5. Discussion

From these results, it is clear that over the time interval, the chance of common corn rust spreading was not very high. With a maximum recorded percentage of about 50 percent, there was no significant risk of common rust during the recorded time interval. This low average percentage indicates that corn was relatively safe during this time interval from this particular pathogen in this particular region of the country. Moreover, it should be noted that a large shortcoming of this experiment is the unknown conditions before this experiment began. It is possible that before the designated time interval, the risk percentage could have been much higher than what this experiment used as its starting risk percentage.

An interesting observation is the effect of humidity and dew point on the final graph. The corresponding rises in the humidity and dew point graph can be approximately seen represented in the final graph of risk percentage. Humidity, in particular, spends very little

of the time interval within the ideal conditions of the pathogen. This indicates that it was humidity and dew point, not temperature, that was keeping the risk percentage relatively low. This notion is supported when looking at the temperature graph, which is almost always within the ideal conditions of the fungus.

Furthermore, another shortcoming of this model is that it does not take into account the host population of the region. If there is a very low population of corn, then the risk percentage would in reality, be made even lower. However, since this experiment does not take into account host population, the risk percentage is slightly skewed. Also, the reason for choosing June dates en lieu of December ones is to showcase actual fluctuations in risk percentage, as in winter, the risk percentage is always at incredibly low.

Overall, this experiment demonstrates that disease forecasting models can be made incredibly easily, and with entirely-publicly available information. Thus, this enables them to be made portable, with simply inputting the weather data from many different regions into the same model and you will accurate results for all of them. This portability allows them to readily applied to any area where a particular pathogen may reside, including low-income areas, which can lead to amazing contributions to the protection of plant health and suppression of pathogens. This of course will yield to better food yields, helping both the farmers growing them and worldwide food economy.

6. Conclusion

In short, this paper explains that plant disease forecasting models can be created and applied relatively easily, and that these models can be used to positively benefit the world as a whole. In particular, through creating our own forecasting model of Moonochie, NJ, we charted the risk percentage of the common corn rust fungus over a period of 4 days, we demonstrated exactly how one would go about creating such a forecasting system, and at the end, we found that the risk percentage over time was very low, with a height of only 50 percent, meaning that there was a very low chance of the pathogen cultivating and spreading.

References

[1] Agrios, G. N. (2005). Plant Pathology.

[2] Ainsworth, G. C. (1981). Introduction to the History of Plant Pathology. *Cambridge University Press*.

[3] Ashrafuzzaman, M. (2013). History of Plant Pathology.

[4] Caffi, T. and Rossi, V. (2018). Fungicide models are key components of multiple modelling approaches for decision-making in crop protection. *Phytopathologia Mediterranea*, 57(1):153–169.

[5] Charaya, M. U., Upadhyay, A., Bhati, H. P., and Kumar, A. (2021). Plant Disease Forecasting: Past Practices to Emerging Technologies.

[6] Fernandes, J. M. C., Pavan, W., and Sanhueza, R. M. (2011). Sisalert-a generic web-based plant disease forecasting system. pages 225–233.

[7] Malvick, D. (2018). Common rust on corn. *University of Minnesota*, (1).

[8] Python (2022). https://www.python.org/downloads/release/python-3110/.

[9] Theophrastus and Hort, A. (1916). Enquiry Into Plants and Minor Works on Odours and Weather Signs (Vol. 1-5).

[10] USPEST.org (2022). https://uspest.org/risk/models.

[11] VSCode (2022). https://www.vscode.dev/.

[12] WeatherUnderground (2022). https://www.wunderground.com/.

7. Appendix

7.1. Weather Data Used

Listed below are the tables used for all risk percentage calculation, as obtained from weatherUnderground[12].

7.1.1. Table 1: Weather Data for 6/1/2022

June 1st, 2022

Time	Temp	Dew Point	Humidity
11:51 PM	63 °F	56 °F	78 %
12:51 AM	61 °F	55 °F	81 %
1:51 AM	60 °F	55 °F	83 %
2:51 AM	59 °F	55 °F	87 %
3:51 AM	58 °F	54 °F	87 %
4:51 AM	58 °F	54 °F	87 %
5:51 AM	59 °F	54 °F	83 %
6:51 AM	60 °F	55 °F	83 %
7:51 AM	61 °F	55 °F	81 %
8:51 AM	61 °F	54 °F	78 %
9:51 AM	64 °F	56 °F	75 %
10:51 AM	67 °F	57 °F	70 %
11:51 AM	69 °F	58 °F	68 %
12:51 PM	71 °F	59 °F	66 %
1:51 PM	70 °F	58 °F	65 %
2:51 PM	69 °F	57 °F	65 %
3:51 PM	66 °F	56 °F	70 %
4:51 PM	64 °F	59 °F	84 %
5:51 PM	63 °F	59 °F	87 %
6:51 PM	62 °F	58 °F	86 %
7:51 PM	61 °F	59 °F	93 %
8:51 PM	61 °F	59 °F	93 %
9:51 PM	61 °F	60 °F	97 %
10:51 PM	61 °F	59 °F	93 %

7.1.2. Table 2: Weather Data for 6/2/2022

June 2nd, 2022

Time	Temp	Dew Point	Humidity
11:51 PM	61 °F	60 °F	97 %
12:51 AM	61 °F	60 °F	97 %
1:51 AM	62 °F	60 °F	93 %
2:51 AM	62 °F	60 °F	93 %
3:51 AM	62 °F	61 °F	96 %
4:51 AM	63 °F	61 °F	93 %
5:51 AM	65 °F	63 °F	93 %
6:51 AM	67 °F	66 °F	97 %
7:51 AM	71 °F	67 °F	87 %
8:51 AM	73 °F	66 °F	79 %
9:51 AM	74 °F	65 °F	73 %
10:51 AM	75 °F	65 °F	71 %
11:51 AM	78 °F	66 °F	66 %
12:51 PM	79 °F	66 °F	64 %
1:51 PM	76 °F	67 °F	74 %
2:51 PM	75 °F	68 °F	79 %
3:51 PM	69 °F	67 °F	93 %
4:51 PM	71 °F	69 °F	93 %
5:51 PM	72 °F	69 °F	91 %
6:51 PM	69 °F	66 °F	90 %
7:51 PM	68 °F	65 °F	90 %
8:51 PM	67 °F	65 °F	93 %
9:51 PM	68 °F	65 °F	90 %
10:51 PM	67 °F	65 °F	93 %

7.1.3. Table 3: Weather Data for 6/3/2022

June 3rd, 2022			
Time	Temp	Dew Point	Humidity
11:51 PM	67 °F	65 °F	93 %
12:51 AM	66 °F	65 °F	96 %
1:51 AM	66 °F	64 °F	93 %
2:51 AM	64 °F	63 °F	96 %
3:51 AM	64 °F	62 °F	93 %
4:51 AM	63 °F	62 °F	97 %
5:51 AM	64 °F	62 °F	93 %
6:51 AM	64 °F	62 °F	93 %
7:51 AM	65 °F	63 °F	93 %
8:51 AM	68 °F	64 °F	87 %
9:51 AM	71 °F	63 °F	75 %
10:51 AM	71 °F	63 °F	75 %
11:51 AM	73 °F	63 °F	71 %
12:51 PM	75 °F	63 °F	66 %
1:51 PM	76 °F	63 °F	64 %
2:51 PM	77 °F	63 °F	62 %
3:51 PM	77 °F	63 °F	62 %
4:51 PM	74 °F	63 °F	68 %
5:51 PM	74 °F	60 °F	62 %
6:51 PM	75 °F	47 °F	37 %
7:51 PM	73 °F	47 °F	39 %
8:51 PM	71 °F	48 °F	44 %
9:51 PM	68 °F	50 °F	52 %
10:51 PM	67 °F	49 °F	52 %

7.1.4. Table 4: Weather Data for 6/4/2022

June 4th, 2022			
Time	Temp	Dew Poin	Humidity
11:51 PM	66 °F	49 °F	54 %
12:51 AM	63 °F	49 °F	60 %
1:51 AM	60 °F	49 °F	67 %
2:51 AM	59 °F	50 °F	72 %
4:51 AM	57 °F	51 °F	81 %
5:51 AM	63 °F	54 °F	72 %
6:51 AM	66 °F	53 °F	63 %
7:51 AM	69 °F	53 °F	57 %
8:51 AM	72 °F	49 °F	44 %
9:51 AM	74 °F	49 °F	41 %
10:51 AM	78 °F	49 °F	36 %
11:51 AM	79 °F	51 °F	38 %
12:51 PM	80 °F	46 °F	30 %
1:51 PM	80 °F	45 °F	29 %
2:51 PM	81 °F	47 °F	30 %
3:51 PM	82 °F	37 °F	20 %
4:51 PM	81 °F	40 °F	23 %
5:51 PM	80 °F	39 °F	23 %
6:51 PM	76 °F	40 °F	27 %
7:51 PM	72 °F	39 °F	30 %
8:51 PM	70 °F	41 °F	35 %
9:51 PM	69 °F	39 °F	33 %
10:51 PM	67 °F	34 °F	30 %