Wildfire Analysis Through Lightning Data

Project Report

CMPE 195B/F

Group 29

by

Sowmya Bijjala

Mathew Kaneda

Danielle Shen

Ryota Suzuki

_____ _____

Dr. Jerry Gao, Project Advisor Date

May 2021

APPROVED FOR THE COLLEGE OF ENGINEERING

DocuSigned by:
Jerry Gao
A92F1FED2B04411...

Dr. Jerry Gao, Project Advisor

Professor Wencen Wu, Instructor

Dr. Xiao Su, Computer Engineering Department Chair

ABSTRACT

[Wildfire Analysis Through Lightning Data]

By: Sowmya Bijjala
Mathew Kaneda
Danielle Shen
Ryota Suzuki

Modern Wildfire analysis is crucial for the state of California and the rising risk of fires. Fires in California have been a common occurrence for years however the fires have become increasingly more complex. There are many causes for fires but the most common cause of fires is natural causes. According to Calfire, there have been 5,356 wildfires this year alone in California burning almost a hundred thousand acres of land. Wildfires not only affect the land and air quality, but they also cause property damage and economically impact the state's resources.

Today's wildfire risk analysis does not include lightning data in its factors. With the recent rise in thunderstorms leading to the cause and spread of wildfires, using data about lightning and factoring it would be more crucial. In addition to the thunderstorm data, we would look at the weather conditions, and fire history to determine the risk of a fire.

With this project, we aim to assess the risk of wildfire in California by analyzing real-time lightning data, weather history, and wildfire history. We will use these datas and apply machine learning algorithms such as logistic regression and random forest models to predict and prevent the damage caused by these wildfires. This timely analysis of real-time data and reliable information about the potential for the rate of fire spread and extreme fire behaviors would be essential to saving lives and property.

Acknowledgments

Our advisor Jerry Gao for leading us through this process.

Table of Contents

References

Appendix

Appendix 1

List of Figures

Appendix 2

List of Tables

Chapter 1. Project Overview

(Sowmya Bijjala, Mathew Kaneda, Danielle Shen, Ryota Suzuki)

1.1 Project Goals and Objectives

Our objective for our project is to create a system that can predict a wildfire that is caused by lightning in California. Within our report, we will cover the various data that is fed into our machine learning, the different models that are available for our machine learning, and the API that were used to gather live data. Through our research and experiment, we tested which method is best suited for us in terms of cost, performance, complexity and we made a user interface that presents our data and prediction.

1.2 Problem and Motivation

California's budget for wildfire prediction and prevention is $209 million, which the state has already run out of with its wildfire season in the year 2020. In the growing concern of climate change, more unusual weather patterns have been occurring including dry lighting storms. These dry lightning storms have caused many wildfires to break out across the state of California resulting in loss of lives and structures. With the wildfire analysis system the group created, there is further notice for firefighters to prepare and anticipate for a lightning-caused fire.
According to Cal Fire's website, in 2020 alone there have been 7982 incidents of fire, with more than 3.5 million acres of land burned. That is around 3.7% of California's land burned, and marked a record for largest wildfire season in California history. The top two are the SCU lightning complex and LNU lightning complex, both caused by lightning.

1.3 Project Application and Impact

Our project helps the prediction of wildfires caused by lightning. By being able to predict these wildfires, we hope that we can stop it while it is small and containable. If we are able to successfully predict and stop the wildfire before it gets too big and out of control, then we are able to prevent property and environment damage. In addition, this would help the state spend their wildfire budget more efficiently if we can help prevent wildfire from spreading. Overall,

our goal is to help predict wildfire risks caused by lightning instead of reacting to a wildfire caused by lightning.

1.4 Project Results and Deliverables

Out of the three models discussed earlier, we built a system that can predict a wildfire based on lightning. This prediction along with the environmental data is shown on a webpage for users to view.

Chapter 2. Background and Related Work

(Sowmya Bijjala, Mathew Kaneda, Danielle Shen, Ryota Suzuki)

2.1 Background and Technologies

Previously fire spreading and other fire assessments could only be completed through fire history, now fire assessments can be completed with new technologies. In recent years, there have been more papers published on using Artificial Intelligence and Machine Learning to predict wildfires. Each paper uses different methods of Machine Learning to predict and assess wildfires.

Back-Propagation Neural Network (BPNN): This is a popular classification algorithm used in conjunction with a multitude of data sources to give predictions. This algorithm was sometimes used with other clustering models to make a hybrid model.

Random Forest (RF) Model: This algorithm builds but different decision trees through the use of the bagging method. Random Forest searches for the best features in the subset of random features. This model has been used to model human-caused wildfires.

Logistic Regression Model: Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable. It is especially suited when the outcome of a model is a binary output, in this case ‘1’ if there a wildfire and ‘0’ if there isn’t

SQL/SQL Alchemy: This database technology will allow this system to store large amounts of data that can be accessed at any time through the backend API. This is where a bulk of the data will be stored. One limitation could be that as we acquire more data, each query will become slower. This can be improved through database indexing in how we create the database models as well as caching which can be done through services such as Redis cache servers.

Redis Cache: This technology allows us to cache data for a duration of time that we specify in our project. Redis can store millions of data-entries while allowing us to quickly find specific data entries due to its time complexity of O(1) within Redis Sets. This can improve large database queries and computation time.

Javascript: This technology will allow us to create an interactive user interface that can communicate with our backend API. This is what will drive the functionality of the app/webapp. Javascript will also allow us to implement lazy-loading which will allow us to load bits of necessary data rather than loading all the data. This will improve performance as the user will not have to wait for every piece of data to be processed.

2.2 State-of-the-art

This project deals with a very serious problem that has been increasing in intensity over the past few years. In order to reduce the damages caused by the wildfires, this project system will predict wildfires, specifically those caused by dry lightning.

Thus far, wildfire prediction has been based on the factors of fire history, weather conditions, land coverage and vegetation growth. The predictions have been based on either statistical approaches and simulation based approaches. We are planning on using a data-driven machine learning approach for this project, in which machine learning models are trained using large datasets to predict wildfires.

While there have been papers that researched the impact of lightning and other factors on causing wildfires, there have been few, if any, papers published on Machine Learning models including lightning in their predictions.

Chapter 3. Project Requirements

(Sowmya Bijjala, Mathew Kaneda, Danielle Shen, Ryota Suzuki)

3.1 Domain and Business Requirements

Process summary diagram

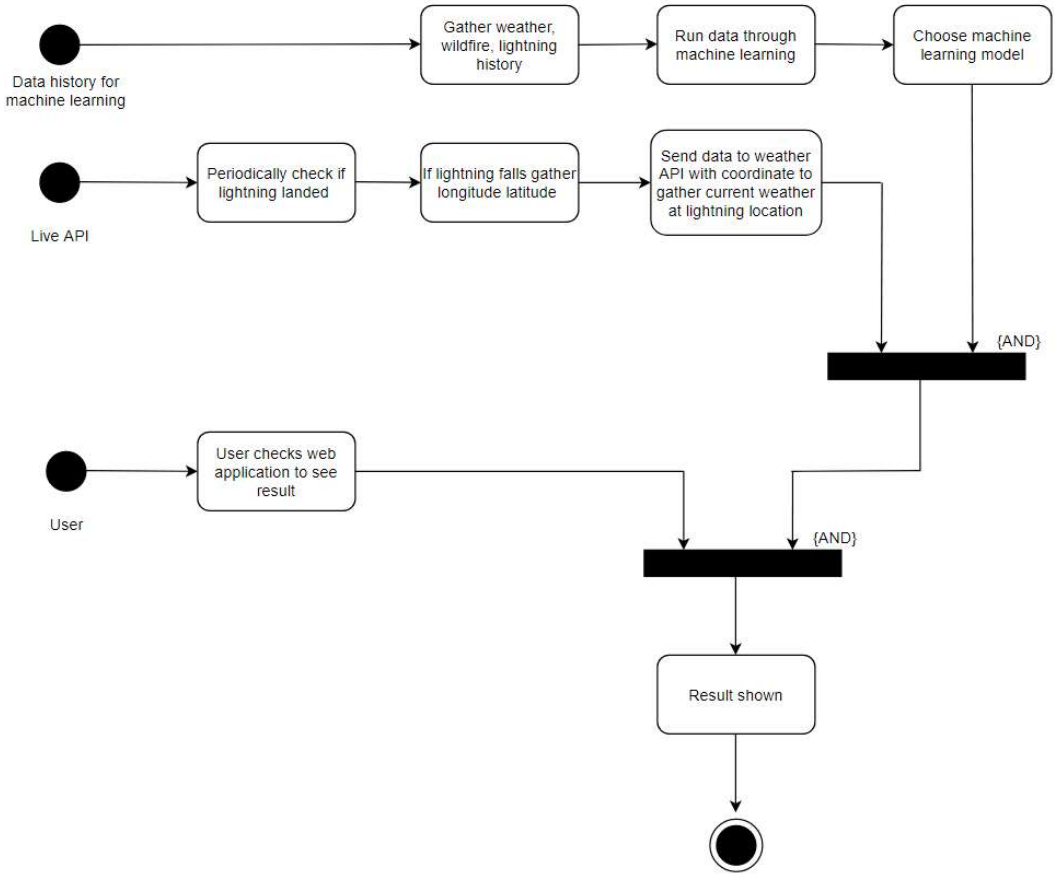

Figure 1: Process summary diagram

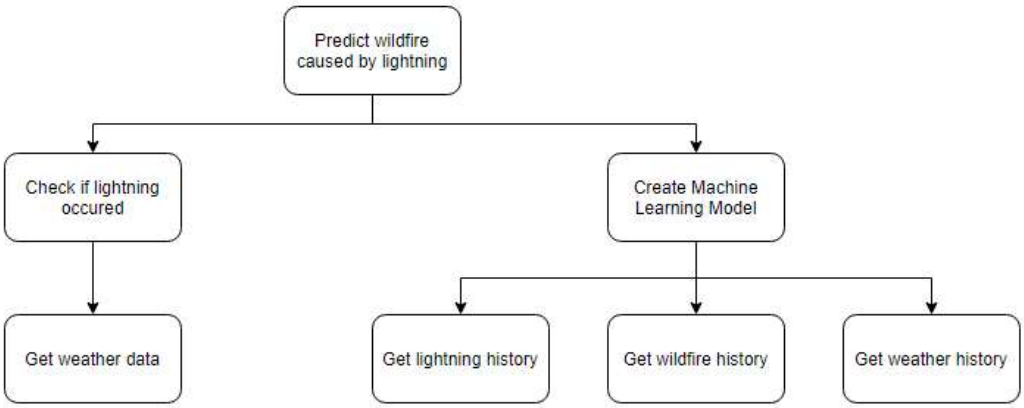

Figure 2: Process decomposition diagram

3.2 System (or Component) Functional Requirements

Our Wildfire Prediction System shall have a visible and interactive map showing lightning strikes in the supported counties. This system shall show wildfires that were directly caused by lightning and should have previous fires for a given county shown on the map. The system shall also have a page or component to tell if a lightning strike can cause a fire based on the weather conditions. The system should also have a navigation bar which will allow future features and modules to be added on with ease.

Table 1: Functional Requirements

Shall have	Visible and interactive map showing lightning strikes and wildfires caused by lightning
	Component/page to show if a given lightning strike can cause a fire based off current weather conditions
Should have	Previous wildfires for given county shown on the map

	Navigation bar for easy navigation and expandability for future addons

3.3 Non-functional Requirements

Our Wildfire Prediction Application will follow file, variable, and function naming conventions such as snake-case for python and camel-case for JavaScript. Our application will also consist of a filesystem that separates each group of endpoints or each component to its own folder to allow for easy navigation through code as well as easy scalability in the future. Our application will handle each edge case so as to never result in a server error and each error will be handled with a verbose error message given to the user. Our application will also be easily run through a webpage so that there is no environmental setup required for users.

Table 2: Non-Functional Requirements

Non-Functional Requirement	**Description**
Scalability and maintainability	Following file, variable, and function naming convention
	File system separating each group of like endpoints to their own file
Reliability/Usability	Handling edge cases with data validation to ensure there are no server (500) errors.
Environmental	Application will be hosted so the user is not required to go through any environment setup to get the application working

3.4 Context and Interface Requirements

1. The machine learning model has to have greater than 60% accuracy.
2. Lightning strikes and the risk of fire shall be displayed on a website that can be accessed by the user.
3. Analyze real-time lightning to determine risk of wildfire, and update the website if there is a possibility of wildfire.

3.5 Technology and Resource Requirements

Table 3: Technology and Resource Requirements

Resource	Resource Details
Python	Python 3 or higher for the ML libraries and website to work
Flask	Web Framework for backend API
Lightning API	Real time data about lightning strikes
Weather API	To get real time information about the weather at a particular location
Lightning, wildfire, weather history (csv)	Data used to create machine learning model
Skitlearn, Google Colab	Machine learning
Visual Code Studio	Used VS code as the workspace to test and develop the front end
Create React App	Used to bootstrap the project and create the frontend build
Laptops	To code and run our program

Chapter 4. System Design

(Sowmya Bijjala, Mathew Kaneda, Danielle Shen, Ryota Suzuki)

4.1 Architecture Design

Below is a diagram detailing how the different models come together to create our system. Once the different data collection is completed, then the data will enter preprocessing in order to prepare for the machine learning model. Once the pre-processing is completed, then the different machine models can be applied to the processed data. Once the model finishes running, then the outputs are compared within the ensemble model.

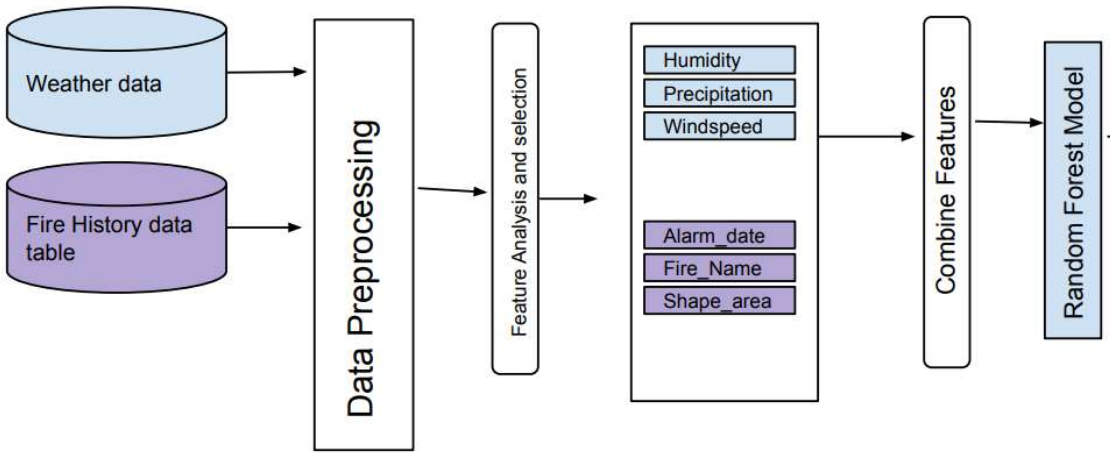

Figure 3: Architecture design

4.2 Interface and Component Design

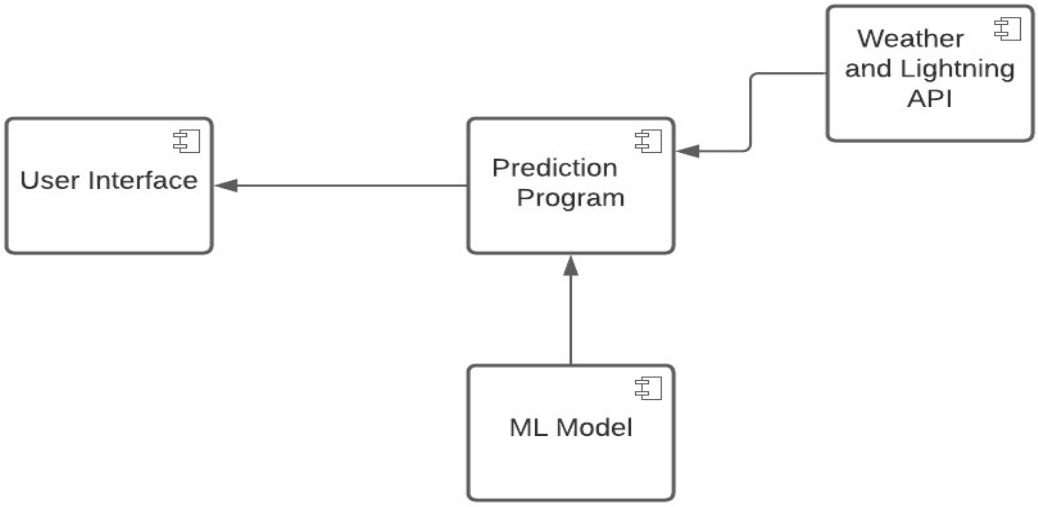

Figure 4: Component diagram

The program will also get real time lightning information from the Lightning API and predict if there is a risk of wildfire based on the output from the machine learning model. The data extracted from the API and the prediction results will be updated on the user interface to alert users when there is a possibility of wildfire.

4.3 Structure and Logic Design

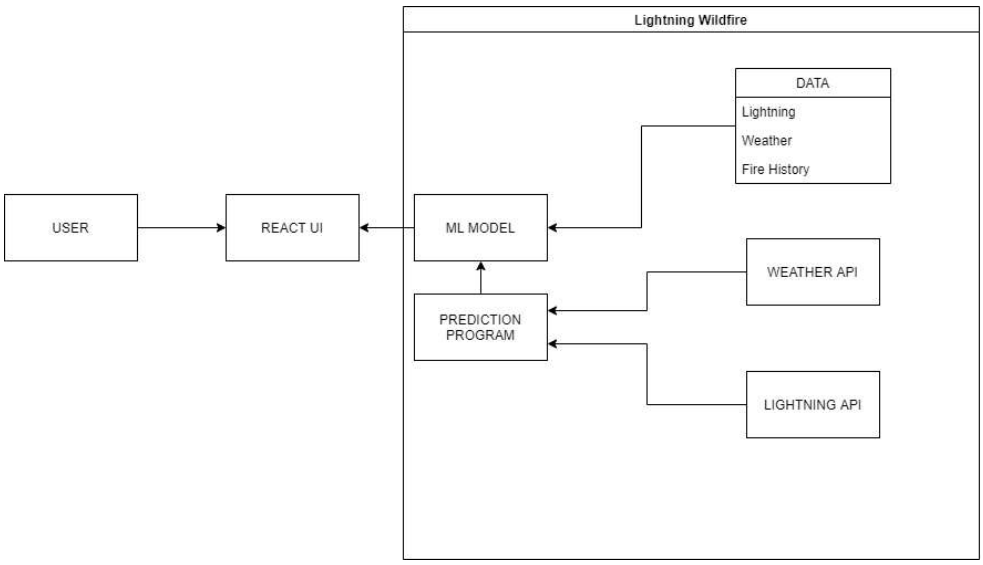

Figure 5: Structure and logic design

The structure design above vially describes the structure and logic of the system. The prediction program will call and retrieve data from the APIs. This data will then be used in the ML model along with the previous data it already has. Once a prediction is made then the react user interface will display that prediction.The user will then be able to interact with the UI and see the result of the model on a map.

4.4 Design Constraints, Problems, Trade-offs, and Solutions

Some design constraints problems we have are that the data provided are not accurate, or have data up to date. These datas were hard to find since many of them were unusable since they were missing a lot of critical information. Our problem was that we couldn't find any usable data until recently, and we were not able to start our machine learning training. Therefore, we had to take the best data we were given, and use it. The trade off was that our accuracy may fall due to the quality of data given to us. However, accuracy may go up once better data is given to us.

4.4.1 Design Constraints and Challenges

Some design constraints and challenges that the team has faced include adequate quality of collected data, real time data and limited knowledge on machine learning topics.

Due to the new nature of this project, there has not been nearly as much accessible resources to find apis for real time data. Quite a bit of time is spent collecting data then having to sort through the data collected. In addition to that, the data that is needed has to fit certain criteria. This is all the data that is used to both train the machine learning model and data to improve the accuracy of the predictions.

The group is also limited by time constraints. Some optional deliverables are optimizations that will be addressed if time allows. Because of the team's lack of previous experience, trying to determine what is necessary for different ML models have been a struggle.

4.4.2 Design Solutions and Trade-offs

Since we are using machine learning, we require massive amounts of data that are accurate and reliable. To get quality data, our group is currently researching many sources and contacting sources for access. However, many of the datas we are looking at are in a format that we can not view. To read these datas we are required to extract it, which we are currently working on. Once we gather the data, we can go into using them for our machine learning.

Due to the group's limited knowledge of machine learning, it's been decided to utilize a few different machine learning models instead of the original number. We are currently taking an online machine learning course to better understand the concept and use case so we can apply it to our project. In addition, our group is working closely with Dr. Gao to determine and run the different machine learning models.

Chapter 5 System Implementation

(Sowmya Bijjala, Mathew Kaneda, Danielle Shen, Ryota Suzuki)

5.1 Implementation Overview

The three main components of the implementation are the data sources, machine learning model and the user interface. The data sources are several APIs to retrieve the real-time data for the machine learning model to use. The machine learning model also needs data for training and testing. This data is from various CSV files, satellite imaging data and other sources. The machine learning model chosen to process our data is the random forest tree model. The user interface is made using react to present the outcome of the machine learning model.

5.2 Implementation of Developed Solutions

The system's core is the Machine Learning model. The team used the logistic regression model to predict the wildfire output. The model's decision tree is based on a number of data points including humidity, precipitation and weather events. Weather events are reported weather instances to NOAA such as TS for thunder and RA for rain. The ML model needs training, test and real data. The real-time data will be from a couple of APIs. Lighting strike and weather APIs are implemented to provide the ML the real-time data.

When there is a lightning strike, the lightning api will provide the longitude and latitude of the strike. With this longitude and latitude, the weather of the strike location will be checked by calling the NOAA weather API. The NOAA weather API will receive the longitude and latitude of the lightning strike and return the current weather conditions including the specified factors used in the decision tree. Once all the data is fed into the Linear Regression Model, the model will output whether there is a wildfire caused by lightning.

The user interface was made with React. The user interface will display the locations of lighting strikes and whether or not they caused a fire. When a lightning strike is detected, a pin will

appear on the map. The color of the pin will determine whether or not that specific lightning strike caused a fire.

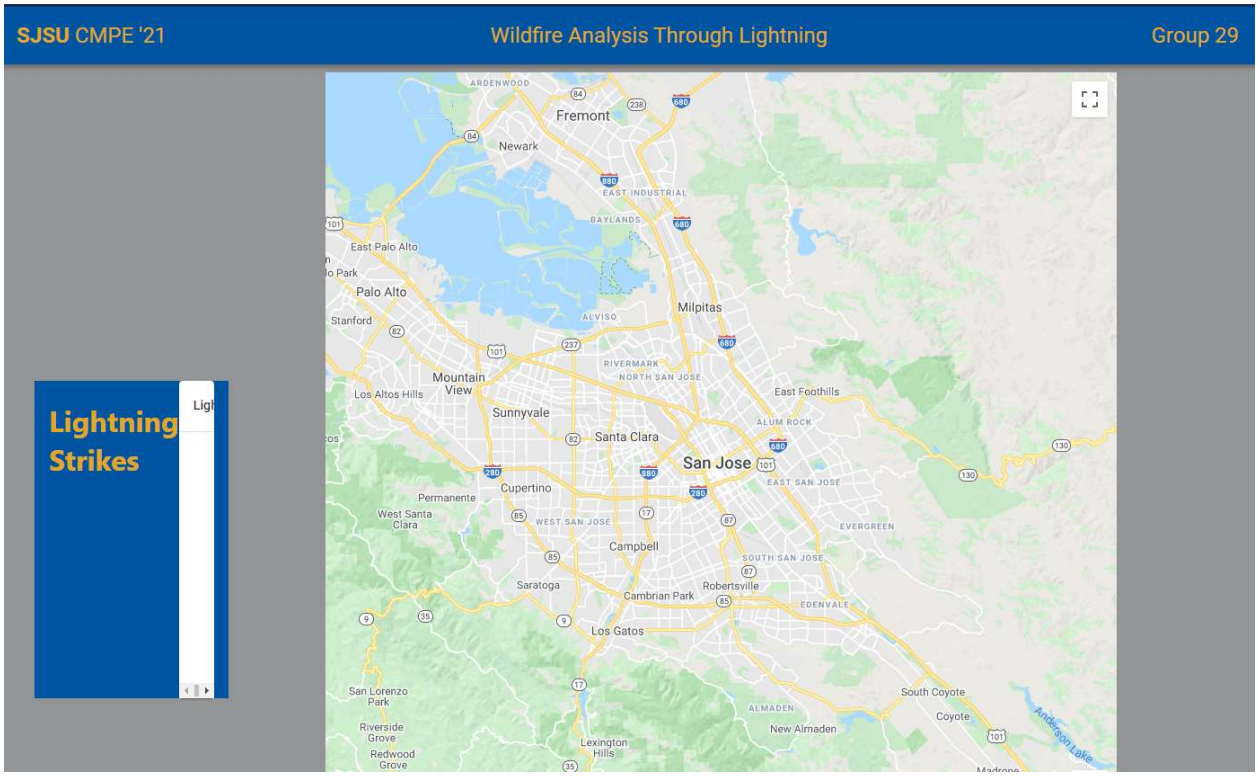

Figure 6: UI of Wildfires results

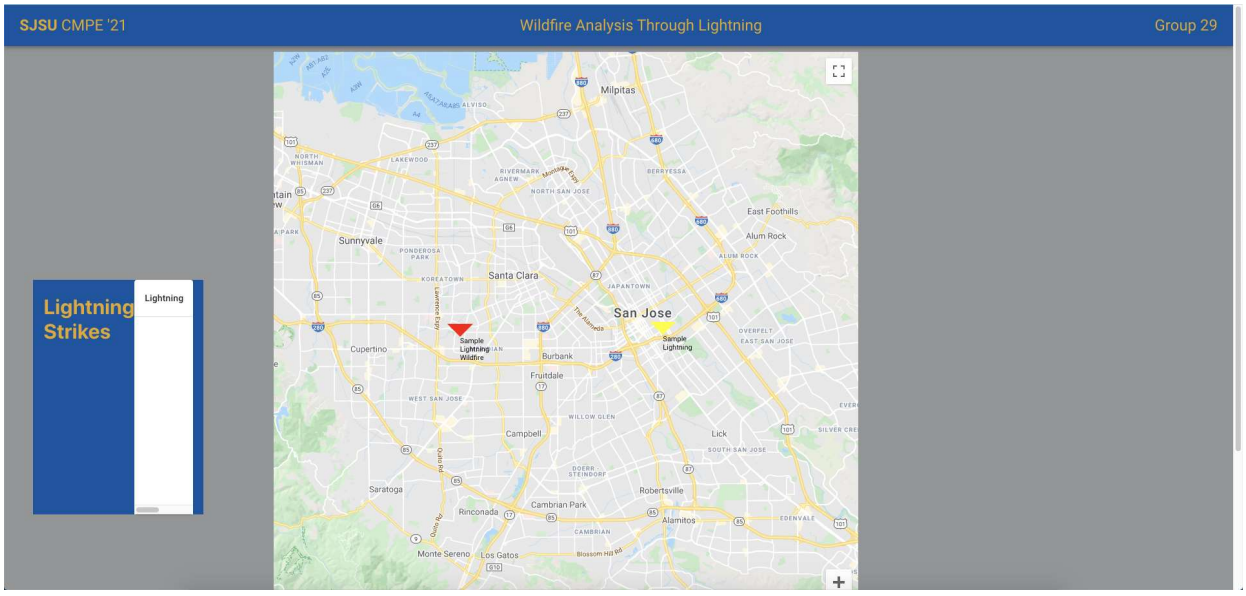

Figure 7: UI Sample Lightning Strike Markers on Map Component

5.3. Implementation Problems, Challenges, and Lessons Learned

Implementation challenges include the struggle with data collection, data formatting for the ML model.A constant challenge has been finding accurate and useful data for training the machine learning model and testing said model. Due to the team's unfamiliarity with machine learning, the model adjustment and data cleaning were challenging tasks to perform. While APIs have been used by the team members before, it is new to use APIs in conjunction with the machine learning model. The team has learned how to integrate the APIs with the user interface as well as the ML model.

Another challenge we faced was the limited data that were accessible. For meteomatics, since we had free trial accounts, we only got access to lightning within 24 hours. In the best case scenario, we would have wanted to get access to all lightning so that we can list them in another page. We could have also tested using these data values. We also had free trial accounts for our openweather API. If we had full access, we would have been able to access more specific data and made our prediction better.

Chapter 6 Tools and Standards

(Sowmya Bijjala, Mathew Kaneda, Danielle Shen, Ryota Suzuki)

6.1. Tools Used

Our project consists of multiple parts and each part is created using different tools. For our back-end, we saved our training and testing data for our machine learning in a CSV file. Then, we fed that data into our tensorflow machine learning to train and test. We also have multiple API calls that help us get the necessary data for our machine learning to predict if lightning will cause a wildfire or not. First, we periodically check if there has been any lightning strike in california. This first API, called meteomatics, will give us the location of a lightning strike in longitude and latitude within California. Then, if there is any lightning, we relay the location to our weather data API. The weather API called openweather will then check the weather data at that location, looking for temperature, dew point temperature, humidity, windspeed, weather and precipitation value. We will then feed this information to our tensorflow machine learning to get a prediction. Our front-end portion of our project was bootstrapped with Create React App. The UI will use Google Maps API in order to display the gathered data. The code will consist of JavaScript, React, and CSS to build the fronted. We will also be using multiple UI toolkits such as material-UI and UI components from react-bootstrap.

6.2. Standards

For our project, we had to maintain standards when we were finding training and testing data for our machine learning model. Since these data will be responsible for precision and accuracy, we had to choose data that best fit our use. It took a long time to find a good dataset that fulfilled our needs. First, we had to find wildfire data that had an accurate start date of fire. Next, we had to find a dataset that was not missing any fire. Finally, we had to choose a dataset that had a detailed cause of the fire since we were only interested in fires that were caused by lightning. Throughout our research we found multiple wildfire history data but many of them were missing one of these requirements. We finally came across calfire's database and used it as one of our

training and testing models of our machine learning. To ensure that each fire was really caused by lightning, we checked each fire entry to see if it was actually caused by lightning, as well as checking the weather on that day to make sure that there was lightning on that day. Another dataset we needed was weather data. We also struggled to find data that fit our needs, as we needed data that had as many details as possible. For this dataset, we used NOAA's weather database as it provided enough information for our machine learning to train and test. Finally, we combined these two files into one so that our machine learning could see the correlation between weather and wildfire.

Chapter 7 Testing and Experiment

(Sowmya Bijjala, Mathew Kaneda, Danielle Shen, Ryota Suzuki)

7.1 Testing and Experiment Scope

Our project consists of 3 units to test: the ML model, the api to get real time weather and lightning data and the user interface. Our focus for testing the ML model is to gain a model accuracy of above 85% with our current data. We also want the precision of the model to be above 90% so that we can ensure that our results are accurate and usable. For testing our api calls, we must ensure that the api returns values that can be used by our ML model. Another requirement of the api calls is that it should output results immediately without any lag since we are looking at real-time hourly weather and lighting data. The requirements for our UI component are that it has to be user friendly and pleasing to the eye. The UI should also be able to display the data received from the api and ML model and display it accurately on a map on the website homepage. In addition to this, the UI should also display a table to lightning and wildfire history. These three components will be first tested to meet requirements individually, and then will be combined to perform system testing.

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.1,random_state=0)
```

Figure 8: Splitting data into training and testing data

```
[ ]  from sklearn import metrics
     print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
     print("Precision:",metrics.precision_score(y_test, y_pred, zero_division=1))
     print("Recall:",metrics.recall_score(y_test, y_pred))

     #Accuracy: 0.9072916666666666
     #Precision: 0.7659574468085106
     #Recall: 0.5806451612903226
```

Figure 9: Accuracy precision, recall rate

7.2 Testing and Experiment Approach

To train and test our machine learning model, our dataset was split into 90% training data and the remaining 10% to test our data. This data split was made randomly using skitlearn's library functions to perform the splitting of data. The accuracy and precision of our model was calculated using the results from testing our model on the test data and ensuring that the output that we got from our model is the same as our expected output. To get the best model to use, we tried testing the accuracy of multiple classification models such as Logistic regression and Random Forest models. To test our api component, we validated the JSON response by sending multiple types of queries and comparing the responses received. We also measured the response time and made sure that it took less than a millisecond to receive response from the api. To make sure that our UI was user friendly, we made sure that our website scales to any screen-size and ensured that the content was easily visible. We tested the UI by having users interact with it.

```
[ ]  feature_cols = ['HourlyDewPointTemperature', 'HourlyDryBulbTemperature', 'HourlyPrecipitation','RA', 'HourlyRelativeHumidity','HourlyWetBulbTemperature','HourlyWindSpeed']
     X = data[feature_cols] # Features
     y = data.Wildfire # Target variable
```

Figure 10: Variables used for training and testing

DATE	HourlyDewPointTemperature	HourlyDryBulbTemperature	HourlyPrecipitation	HourlyPresentWeatherType	RA	HourlyRelativeHum	HourlySkyConditions	HourlyWetBulbTemperature	HourlyWindSpeed	Wildfire
2014-02-28T08:33:00	48	55	0.05	VCTS:7 RA:02 |RA |	1	77	FEW:02 20 BKN:07 50 OVC:08 65	51	15	0
2014-02-28T08:38:00	48	55	0.07	VCTS:7 +RA:02 |RA |	1	77	FEW:02 24 BKN:07 50 OVC:08 70	51	15	0
2014-04-01T02:01:00	39	48	0	VCTS:7 ||	0	71	CLR:00	44	9	0
2014-12-15T13:54:00	51	54	0.35	VCTS:7 RA:02 FG:2 |FG RA |	1	90	FEW:02 3 BKN:07 20 OVC:08 50	52	6	0
2015-05-14T10:52:00	45	61	0	VCTS:7 ||	0	55	OVC:08 50	53	9	0
2015-05-14T10:54:00	45	60	0	VCTS:7 ||	0	58	BKN:07 50 OVC:08 70	52	10	0
2015-07-19T03:43:00	55	66	0	TS:7 |TS TS |	0	68	BKN:07 110	60	3	0
2015-07-19T03:54:00	55	66	0	TS:7 |TS TS |	0	68	BKN:07 100	60	3	0
2015-07-19T04:39:00	54	64	0	VCTS:7 ||	0	68	FEW:02 100 SCT:04 120	58	0	0
2015-07-19T04:54:00	55	65	0	VCTS:7 ||	0	70	FEW:02 55 SCT:04 110	59	5	0
2015-07-19T05:54:00	58	64	0.01	TS:7 -RA:02 |RA TS TS |	1	81	BKN:07 60 OVC:08 75	60	6	0
2015-07-19T06:54:00	60	64	0.03	VCTS:7 ||	0	87	FEW:02 43 SCT:04 70 BKN:07 90	62	3	0
2015-07-19T07:32:00	61	64	0.02	VCTS:7 ||	0	88	FEW:02 55 BKN:07 100 BKN:07 120	62	0	0
2015-08-06T19:10:00	57	63	0	VCTS:7 ||	0	83	OVC:08 10	60	8	0
2015-08-06T20:12:00	57	63	0	VCTS:7 ||	0	83	OVC:08 10	60	10	0
2015-08-06T20:54:00	58	62	0.02	TS:7 |TS TS |	0	86	OVC:08 10	60	5	0
2017-01-08T00:03:00	55	56	0.05	VCTS:7 RA:02 BR:1 |RA |	1	97	FEW:02 3 BKN:07 7	55	10	0
2017-01-08T00:08:00	55	56	0.2	VCTS:7 +RA:02 BR:1 |RA |	1	97	FEW:02 3 BKN:07 7 OVC:08 16	55	14	0
2017-01-08T00:13:00	55	55	0.6	|FG TS |	0	100		55	7	0
2017-01-08T00:23:00	55	56	0.79	|TS |	0	97		55	8	0
2017-01-08T00:27:00	55	56	0.79	TS:7 RA:02 |RA TS TS |	1	97	SCT:04 8 OVC:08 17	53	10	0
2017-01-19T03:18:00	47	51	0.01	VCTS:7 -RA:02 |RA |	1	86	FEW:02 45 BKN:07 55 OVC:08 80	49	6	0
2017-01-20T17:21:00	43	50	0	VCTS:7 ||	0	77	FEW:02 28 SCT:04 37 BKN:07 55	47	17	0
2017-09-11T04:52:00	59	66	0	VCTS:7 ||	0	78	FEW:02 120	62	11	1
2017-09-11T04:54:00	59	68	0	VCTS:7 ||	0	73	FEW:02 120	63	10	1
2017-09-11T07:14:00	62	73	0	VCTS:7 ||	0	69	FEW:02 48 SCT:04 100 BKN:07 120	66	9	1
2017-09-11T12:44:00	59	87	0	VCTS:7 ||	0	39	FEW:02 110	69	9	1
2017-09-11T14:07:00	57	86	0	VCTS:7 ||	0	37	FEW:02 49 FEW:02 75 FEW:02 110	68	10	1
2017-09-13T02:31:00	60	64	0	VCTS:7 ||	0	87	OVC:08 11	62	6	0
2017-09-13T04:52:00	61	63	0	TS:7 |TS |	0	94	OVC:08 11	62	3	0
2017-09-13T04:54:00	60	64	0	TS:7 |TS TS |	0	87	OVC:08 11	62	5	0
2018-01-03T17:23:00	53	58	0.07	VCTS:7 -RA:02 |RA |	1	84	BKN:07 42 BKN:07 48 BKN:07 60	55	0	0
2018-04-10T15:41:00	56	59	0	VCTS:7 ||	0	90	OVC:08 5	57	8	0
2018-11-29T02:32:00	53	57	0.02	VCTS:7 ||	0	87	FEW:02 20 SCT:04 33 BKN:07 50	55	20	0
2018-11-29T07:27:00	48	57	0	VCTS:7 -RA:02 |RA |RA	1	72	FEW:02 31 BKN:07 45 OVC:08 85	52	20	0
2019-02-08T21:09:00	40	45	0	VCTS:7 ||	0	83	SCT:04 15 BKN:07 32 OVC:08 45	43	7	0
2019-05-19T06:19:00	49	55	0.02	VCTS:7 RA:02 |RA |RA	1	80	FEW:02 39 OVC:08 55	52	8	0
2019-05-19T06:26:00	49	55	0.07	VCTS:7 +RA:02 |RA |RA	1	80	FEW:02 30 BKN:07 48 OVC:08 55	52	17	0
2019-05-19T06:32:00	45	51	0.24	|TS |	0	80		46	22	0
2019-05-19T06:39:00	45	49	0.37	|TS |	0	86		66	8	0
2020-08-16T01:28:00	57	77	0	TS:7 |TS TS |	0	50	CLR:00	65	8	1
2020-08-16T01:54:00	59	74	0	VCTS:7 -RA:02 |RA |RA	1	60	FEW:02 50 FEW:02 70 SCT:04 110	65	3	1
2020-08-16T03:08:00	59	74	0	TS:7 |TS TS |	0	60	FEW:02 37	65	6	1
2020-08-16T03:23:00	59	74	0	VCTS:7 ||	0	60	CLR:00	65	13	1
2020-08-16T05:14:00	58	79	0	TS:7 |TS TS |	0	49	CLR:00	66	5	1
2020-08-16T07:19:00	59	84	0	VCTS:7 ||	0	43	SCT:04 90	68	0	1
2020-08-16T07:54:00	59	78	0	VCTS:7 ||	0	52	SCT:04 90	66	10	1

Figure 11: Sample training and testing data

7.3 Testing and Experiment Results and Analysis

Our Machine learning model had an accuracy of 90% and Precision value or 99% when tested on our current dataset. Our API has a fast response rate, and outputs results without lagging. In our UI, we were able to eliminate many bugs such as field width errors and usability errors. We were also able to incorporate scrolling features, which made our website more user friendly.

Table 4: Functional Testing

Case	Result
Visible and interactive map showing lightning strikes and wildfires caused by lightning	Google Maps API used to show map and appropriately colored markers to indicate lightning strikes and the result of analysis
Component/page to show if a given lightning strike can cause a fire based off current weather conditions	Webpage displaying the current lightning strikes, machine learning model used and analysis result.
Previous wildfires for given county shown on the map	Not implemented as this would clutter the map interface. History data is displayed on the same page as the ML algorithms.
Navigation bar for easy navigation and expandability for future addons	Navigation bar at the side implemented for users to switch between the different pages

Figure 12: UI Testing

Table 5: Context Testing

Context Requirements	Result
The machine learning model has to have greater than 60% accuracy.	The model has a 90% accuracy
Lightning strikes and the risk of fire shall be displayed on a website that can be accessed by the user.	The above figure shows the UI displaying the lightning and risk of fire in a table
Analyze real-time lightning to determine risk of wildfire, and update the website if there is a possibility of wildfire	The API are based on real-time data and the website is updated after analysis of lightning strike

Table 6: Non-Functional Testing

Non-Functional Requirement	**Description**	**Result**
Scalability and maintainability	Following file, variable, and function naming convention	All pages are in separate files. Variable names following ES6 naming convention.
	File system separating each group of like endpoints to their own file	Files are grouped and separated into individual modules and initialized separately
Reliability/Usability	Handling edge cases with data validation to ensure there are no server (500) errors.	Edge cases handled and data validation done to the best of our ability. With 500 errors, both the backendAPI and the webpage will not crash but will handle the error accordingly.
Environmental	Application will be hosted so the user is not required to go through any environment setup to get the application working	The backend API will be hosted on heroku and if possible, the frontend will be as well. These will be pipelined for a 2 stage system of staging and production where most of the testing will be on staging.

Chapter 8 Conclusion and Future Work

(Sowmya Bijjala, Mathew Kaneda, Danielle Shen, Ryota Suzuki)

8.1 Conclusion

After many months of researching, gathering data, and coding, we were able to create a web application that incorporates machine learning to predict if a lightning will cause a wildfire. The prediction occurs when lightning strikes within our California boundary box. When it does, our lightning API will send the location of the lightning using latitude and longitude to our weather API call. Then, the weather API will send the current weather information such as temperature, humidity, precipitation, etc and feed it into our machine learning model we created. The machine learning model we trained uses weather history, lightning history, and fire history throughout various counties within California. After testing multiple machine learning models and comparing accuracy and precision rate, we decided to use logistic regression for our machine learning model. Then, our machine learning will output our prediction if that lightning will cause a wildfire or not. Afterwards, this will show up on our Google Map API of our webpage. This will show the location of the lightning strike with markers to indicate the result of our prediction for users to easily see it.

To train and test our machine learning model, our dataset was split into 90% training data and the remaining 10% to test our data. This data split was made randomly using skitlearn's library functions to perform the splitting of data. The accuracy and precision of our model was calculated using the results from testing our model on the test data and ensuring that the output that we got from our model is the same as our expected output. Our Machine learning model had an accuracy of 90% and Precision value or 99% when tested on our current dataset. Our API has a fast response rate, and outputs results without lagging.

8.2 Future Work

For our future work, we want to make our accuracy higher as our current accuracy of 90% can be improved. This can be done by getting better datasets that have more weather variables as well as

feeding more data samples to our model. However, this was where we struggled the most, as these data were not easily found or accessible. We spent multiple months trying to find data that had the required variables, enough datasets, formatted correctly, and most importantly accessible. Most of the time, these datas were subscription based or only accessible to military or government officials, making it hard for us. If we were able to find a dataset that was better, we believe that our prediction model will have higher accuracy. In addition, we would make the UI better so that it is cleaner and users have easier time to navigate through our web application. We were not able to give our UI enough time because we were busy trying to find ideal data for our machine learning model.

References

Andagostino. (2019, August 16). Bounding boxes for all US States. Retrieved April 30, 2021, from https://anthonylouisdagostino.com/bounding-boxes-for-all-us-states/

California Department of Forestry and Fire Protection (CAL FIRE). (n.d.). Welcome to Cal FIRE. Retrieved April 30, 2021, from https://www.fire.ca.gov/

Chiao, S., Gao, J., Xu, G. (2020) Wildfire Prediction: Big Data and Machine Learning Approaches. 2020.05.06

Lightning forecasts. (n.d.). Retrieved April 30, 2021, from https://www.meteomatics.com/en/api/available-parameters/lightnings/

Lori Jean Mitchener & Albert J. Parker (2005) Climate, Lightning, and Wildfire in the National Forests of the Southeastern United States: 1989-1998, Physical Geography, 26:2, 147-162, DOI: 10.2747/0272-3646.26.2.147

National oceanic and Atmospheric Administration. (n.d.). Retrieved April 30, 2021, from https://www.noaa.gov/

OpenWeatherMap.org. (n.d.). Current weather and forecast. Retrieved April 30, 2021, from https://openweathermap.org/

Rolinski, T., S. B. Capps, R. G. Fovell, Y. Cao, B. J. D'Agostino, and S. Vanderburg, 2016: The Santa Ana Wildfire Threat Index: Methodology and Operational Implementation. Wea. Forecasting, 31, 1881–1897, https://doi.org/10.1175/WAF-D-15-0141.1.

Sayad, Y. O., Mousannif, H., & Moatassime, H. A. (2019). Predictive modeling of wildfires: A new dataset and machine learning approach. *Fire Safety Journal,*

104, 130-146. doi:10.1016/j.firesaf.2019.01.006