

# Wildfire Challenge – Team PSR

## Table of Contents

Data Pre-Processing .....	2
Acquiring Weather Data .....	2
Weather Data Scraping .....	3
Dataset balancing .....	3
Correlation with Fire_Size .....	4
Dropped Parameters Explanation .....	5
Weather Parameters used to train the model .....	5
Weather Forecasting .....	7
Weather forecasting model .....	7
Model Results & Evaluation .....	9
Final Thoughts on Weather Forecasting Model .....	11
Wildfire Model Building & Training .....	12
Pipeline .....	13
Model Results .....	14
Performance of Final Model .....	15
Prediction Application .....	16
Inputs to the model .....	16
Features of the Application .....	18
Zip/ Postal Code as the Geocode .....	18
Prediction Location History .....	19
Exception Handling .....	20
Demo Video for New Users .....	22
Prediction from the Application .....	23
References .....	24

## Data Pre-Processing

The dataset for the h2o.ai wildfire challenge was taken from Rachael Tatman's 1.88 Million US wildfires dataset. It's available as a public dataset on Kaggle containing wildfires occurred in United States from 1992 to 2016.[1]

Dataset in the Kaggle is in the form of SQL lite database file. Database file is then converted into panda dataframe.

```
conn = sql.connect('Data.sqlite')
fire = pd.read_sql('SELECT * FROM Fires', conn)
fire.head(10)
```

After the dataframe is created the following data columns were dropped from the dataframe.

- Discovery Date (discovery\_date)
- Discovery Time (discovery\_time)
- Continuous Date (cont\_date)
- Continuous Time (cont\_time)

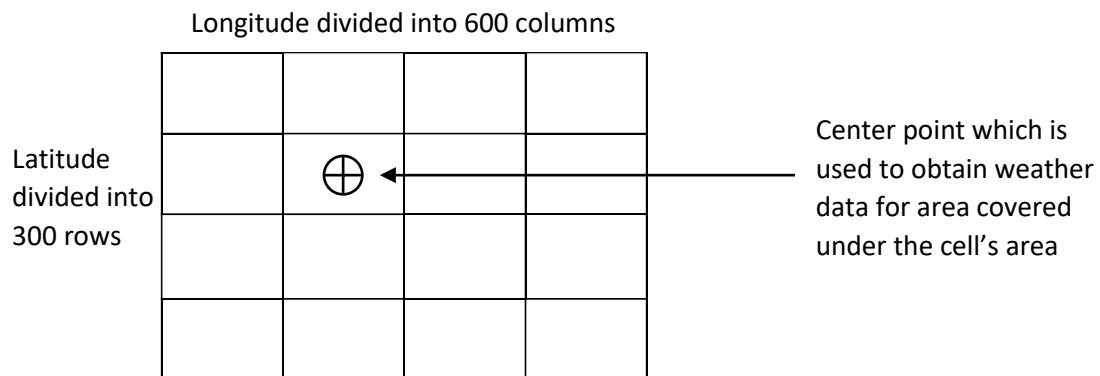
### Acquiring Weather Data

Above dataset comprises with 1.88 million wildfire occurrences throughout the United states. The application is focused on making the predictions for a wildfire using the weather data of the incident is recorded. In order to minimize the weather data that needs to be taken from NASA Langley Research Centre [2] webpage, the geo-locations available in the dataset have been divided into a grid of 600 columns and 300 rows. The grid's corners are determined using the dataset's minimum and maximum values for latitude and longitude.

Corner of the Grid	Value
Bottom Left	17.9397, -178.8086
Bottom Right	17.9397, -65.2569
Top Left	70.3306, -178.8086
Top Right	70.3306, -65.2569

## Weather Data Scraping

After the grid is created, each record has been placed into the relevant cell according to the geo-coordinate. Then weather data for the cell's centre point (latitude & longitude) is used to obtain data from NASA weather dataset mentioned above.



Since the grid has created 300x600 data points which need to obtain weather data from, a web-scraping script was created to automate the task. Selenium web-driver has been used to automate the data scraping tasks and it was able to successfully scrape weather data from 1992 to 2016 related to the incident recording date from the wildfire dataset. Scraped data has been used to train the machine learning model and another set of data from 2020 to 2021 have been used to get predictions with user input.

Weather data for each wildfire occurrence has been taken for **7 days prior from the incident record** date. Those weather data is then placed in each row with the wildfire dataframe.

## Dataset balancing

The wildfire dataset was only containing incidents of wildfire occurring under severity of the fire. Allowing the model to learn about occurrence where wildfires haven't recorded, it has been **assumed that 3 months after an incident has occurred, there hasn't another wildfire incident happened**. Under the above assumption weather data has been scraped and inserted into the model training dataset with zero severity (no wildfire risk) value set. This has been done in order to have a balanced dataset before the training of model.

## Correlation with Fire Size

Correlation matrix with fire size column has been then plotted to identify the most important features from the dataset.

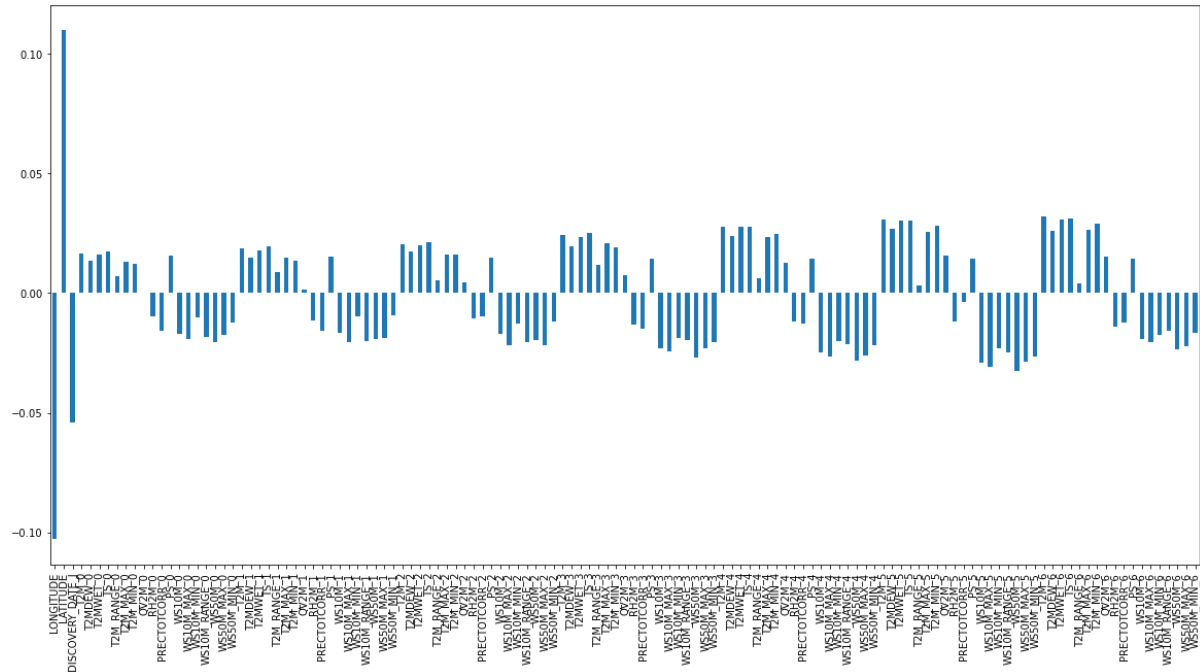


Figure 1 - Correlation Matrix

Considering the correlation values with the dependent variable (fire\_size) following columns were further dropped from the dataframe.

### Determining correlation values less than 0.008

```
Col_to_drop = []
for index, val in Plot_data.items():
    if(abs(val) < 0.008):
        Col_to_drop.append(index)
print(Col_to_drop)
```

### OUTPUT

```
['T2M_RANGE_0', 'QV2M_0', 'QV2M_1', 'T2M_RANGE_2', 'QV2M_2', 'QV2M_3', 'T2M_RANGE_4', 'T2M_RANGE_5', 'PRECTOTCORR_5', 'T2M_RANGE_6']
```

### Dropped Parameters Explanation

Name	Unit	Description
<b>T2M_RANGE_0</b>	C - Celsius	Temperature at 2-meter range (0 = present day)
<b>QV2M_0</b>	g/kg	Specific Humidity at 2 meters (0 = Present day)
<b>QV2M_1</b>	g/kg	Specific Humidity at 2 meters (1 = One day before)
<b>T2M_RANGE_1</b>	C - Celsius	Temperature at 2-meter range (1 = One day before)
<b>QV2M_2</b>	g/kg	Specific Humidity at 2 meters (2 = Two days before)
<b>QV2M_3</b>	g/kg	Specific Humidity at 2 meters (3 = Three days before)
<b>T2M_RANGE_4</b>	C - Celsius	Temperature at 2-meter range (4 = Four days before)
<b>T2M_RANGE_5</b>	C - Celsius	Temperature at 2-meter range (5 = Five days before)
<b>PRECTOTCORR_5</b>	mm/day	Precipitation Corrected (Rainfall) (5 = Five days before)
<b>T2M_RANGE_6</b>	C - Celsius	Temperature at 2-meter range (6 = Six days before)

### Weather Parameters used to train the model

Please see the table below.



## Weather Forecasting

The Wildfire prevention application is currently designed to predict wildfire based upon selected day's weather dataset. The weather dataset is obtained from separate website in the application. Currently the application is only limited to selected time range which the predictions can be obtained on. The reason behind limiting the time range is because currently the application is not fitted with the function to fetch weather data for relevant day or date that are in future.

In order to address the limitation of having to fetch weather data every time and to predict future wildfire risks, possibility to design **accurate weather prediction model** has been tested as following.

### Weather forecasting model

Free to use dataset was available on 'MeteoBlue' web page [3]. The dataset was based in Basel a city in Switzerland. The data which were acquired were having 6 weather parameters in daily basis and hourly basis. The daily basis weather data were taken for a period of 9 years between 2010 January to 2018 December. Then the hourly basis weather taken only for 4-year period between 2015 January to 2018 December. Weather data which came as hourly basis contained 24 values per day sub-dataset for representing each hour in a day, but the daily weather data set only had the mean values regarding the parameters which were requested when downloading from web page

The RNN model has been fed only with one input at a time, but it has modelled to predict several type of forecasts such as precipitation, temperature and wind speed. The model has been fed with two set of datasets, where one set was daily record based and other with hourly record based. The model, which is built on daily data, is capable of predicting 1 day ahead of weather forecast considering past data which belongs to 3 days. The hourly based model evaluates 24 hours of past data to predict 1 hour ahead weather forecast.

The model has been built with Long Short-Term Memory (LSTM) cells included in every dense layer. Input layer consists of 120 neurons and the output layer only with 1 unit. The dense layer set with 3 layers, and 4 drop out layers were used to model both RNN models for daily and hourly dataset. Dropout layer in a RNN structure means that, if the threshold value of dropout is passed, the relevant neuron will be deactivated. Combined with this model it allows neurons, not to depend on other neurons out of order weights. Following table represent the parameters which have been selected to build the model along with training and testing dataset selection criteria.

Table 1 - RNN Model Specification

Layer Rank	Layer Name	Layer Size (Neurons)	Dropout Rate
1	Input Layer	120	-
2	Dropout Layer 1	-	0.2
3	Dense Layer 1	120	-
4	Dropout Layer 2	-	0.2
5	Dense Layer 2	120	-
6	Dropout Layer 3	-	0.2
7	Dense Layer 3	120	-
8	Dropout Layer 4	-	0.2
9	Output Layer	1	-
Optimizer = 'Adam'; loss = 'Mean Squared Error'			
Epochs = 25; Batch size = 500 (hourly set), 100 (daily set)			

Following table represents the total number of samples used to train both hourly and daily weather forecasting models along with the time-period it was taken.

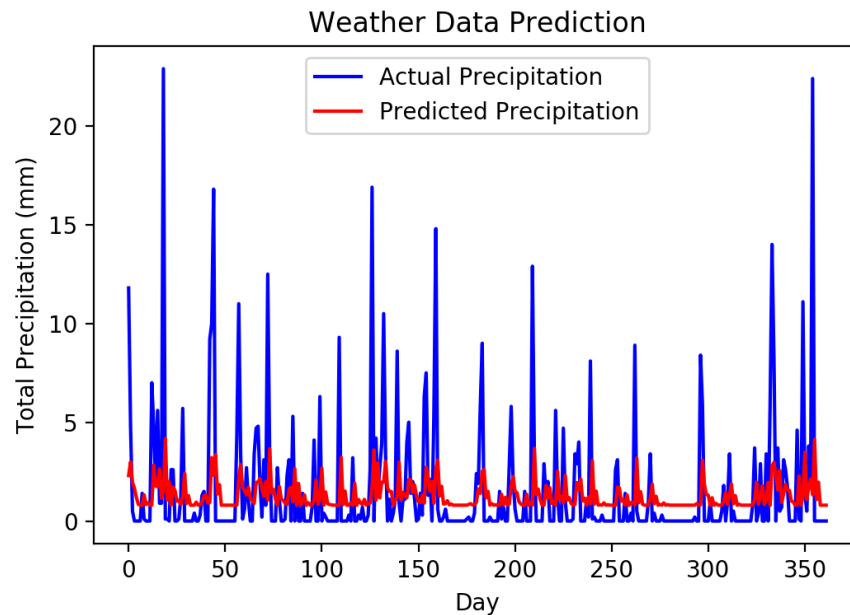
Table 2 - Dataset sizes used on RNN Model

RNN Model	Data Type	Time Period of Dataset	Total Samples
<b>Hourly based model</b>	Training Dataset	2017 January to December	8736
	Testing Dataset	2018 January to December	8736
<b>Daily based model</b>	Training Dataset	2010 January to 2017 December	2920
	Testing Dataset	2018 January to December	365

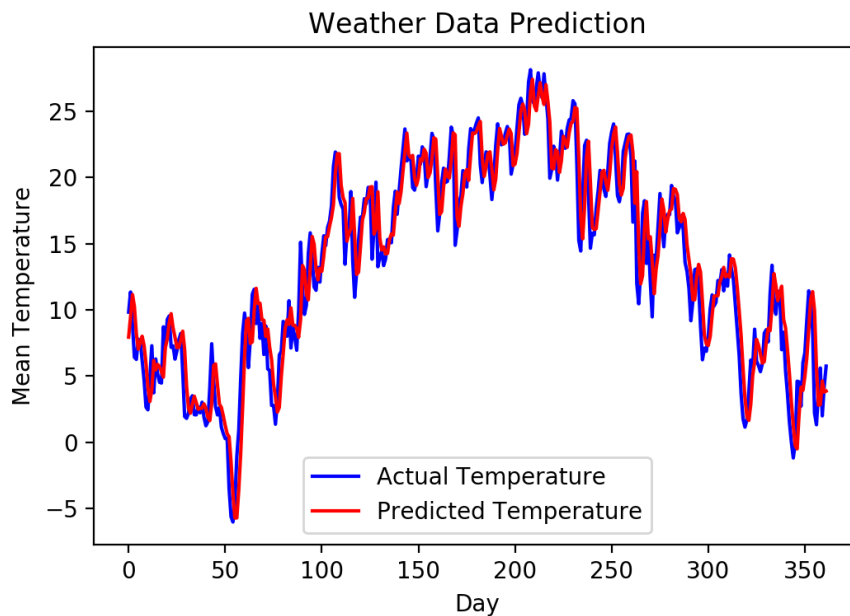


## Model Results & Evaluation

Following figures represents the prediction results and actual data for model which predicts weather one day ahead. Parameters which can be organized in a time series array has only been trained with the model.



*Figure 2 - Daily Precipitation Forecast using RNN*



*Figure 3 - Daily Temperature Forecast using RNN*

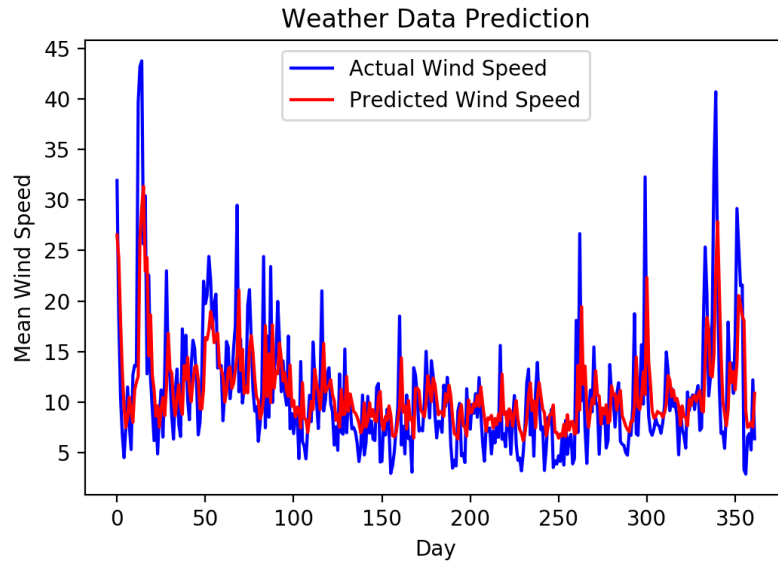


Figure 4 - Daily Wind Speed Forecast using RNN

Following figures represents the prediction results and actual data for model which predicts relevant weather parameter one hour ahead.

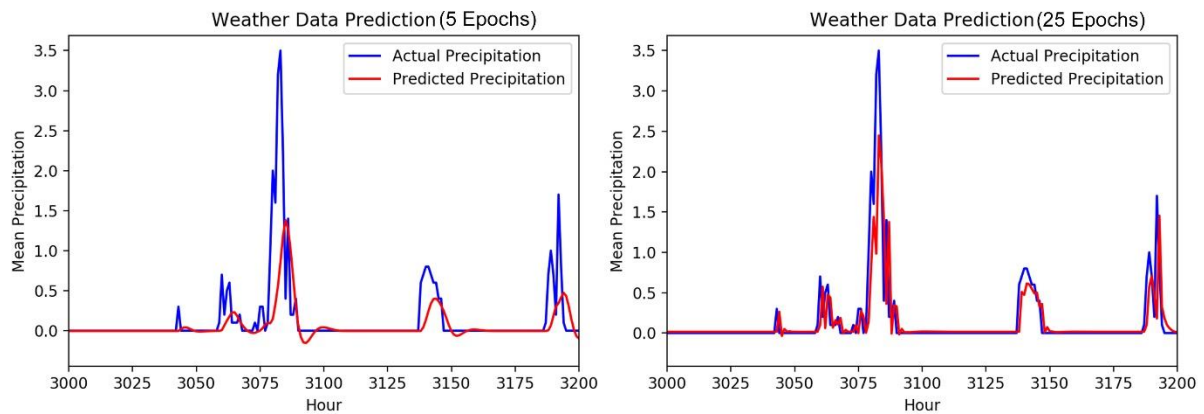


Figure 5 - Hourly Precipitation Forecasting RNN Model

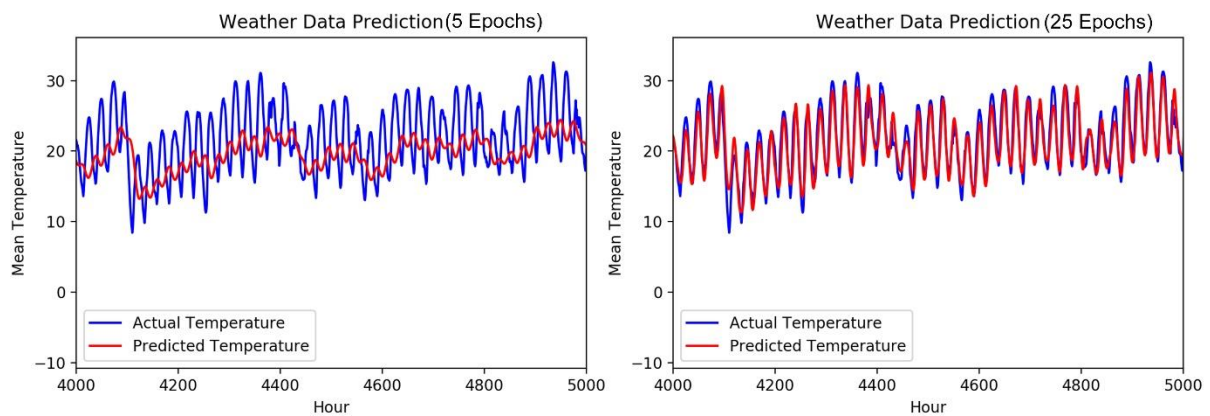
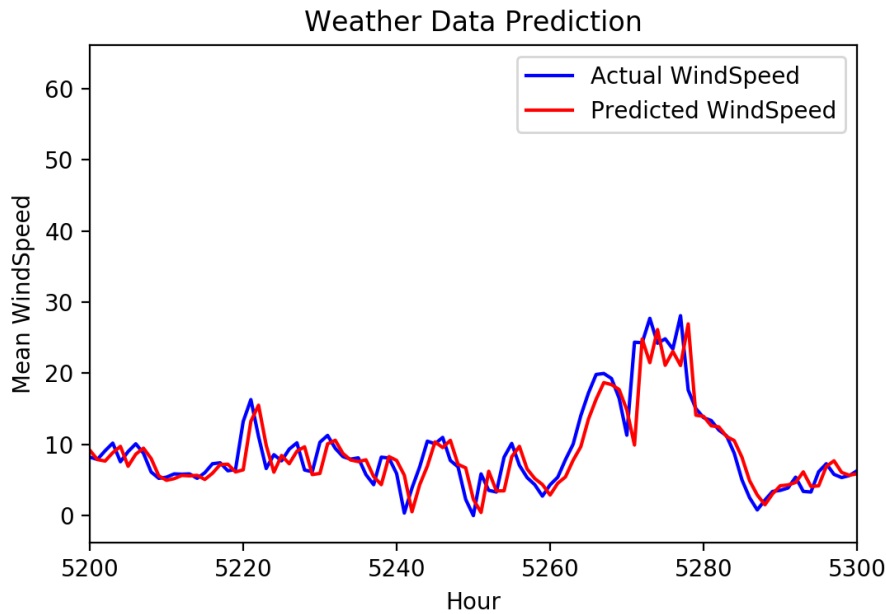


Figure 6 - Hourly Temperature Forecasting RNN Model Comparison



*Figure 7 - Hourly Wind Speed Forecasting using RNN [Scaled]*

The Recurrent Neural Network model with LSTM cells has performed well throughout the testing phase. It has been shown that a well-trained RNN model can provide accurate results on weather forecasting. The RNN model does only consider the present and past data set to predict the behavior of the weather in future time. When comparing both daily based and hourly based models, it could be seen that the hourly predicting model, in other words short-term predicting model perform well.

### Final Thoughts on Weather Forecasting Model

However above results clearly shows that precipitation, temperature & windspeed which are in a time series form can be predicted using above model with a quite good accuracy. The issue arises when the Wildfire application requires more weather parameters which cannot be predicted with RNN model as above. **Specific humidity (g/kg)** and **precipitation (mm/day)** parameters are holding dependency with more advance metrological parameters such as cloud coverage, air pressure and require more advanced neural networks for prediction. Thus, designing a separate machine learning model dedicated to predicting weather data instead of properly utilizing a third-party service for fetching relevant weather data, could lead to less accurate result in wildfire prediction application.

## Wildfire Model Building & Training

After pre-processing the raw data, dataset was saved as a CSV file format. Then it was added to the H2O Driverless AI platform in order to determine the best suitable model and parameters.

Prior to the dataset being processed, dataset was split into train, test & validation. The percentage for each set is respectively 70%, 15% & 15%.



The screenshot shows the H2O.ai Datasets interface. At the top, there's a navigation bar with links like PROJECTS, DATASETS, AUTOVIZ, EXPERIMENTS, etc. Below the header, there's a search bar and a table of datasets. The table has columns: Name, Path, Size, Rows, Columns, Status, and Created. There are four datasets listed: Test, Validation, Train, and Data\_final.csv.

Name	Path	Size	Rows	Columns	Status	Created
Test	...f7/Test.163911864.0255008...	2MiB	7K	35	[Click for Actions]	12/10/2021, 10:21:03 AM
Validation	...lidation.163911863.958916...	2MiB	7K	35	[Click for Actions]	12/10/2021, 10:21:03 AM
Train	...7/Train.163911828.5664454...	9MiB	33K	35	[Click for Actions]	12/10/2021, 10:20:28 AM
Data_final.csv	...nal.csv.163911631.5117447.bin	13MiB	48K	35	[Click for Actions]	12/10/2021, 10:17:11 AM

Figure 8 - Dataset Split

After the dataset is prepared by splitting as above, an Experiment Setup was created inside the platform. Experiment setup was targeting “fire\_size” column as the dependent variable and experiment type is set to “supervised”. Afterwards the experiment setup is started and following results has been obtained by the platform.

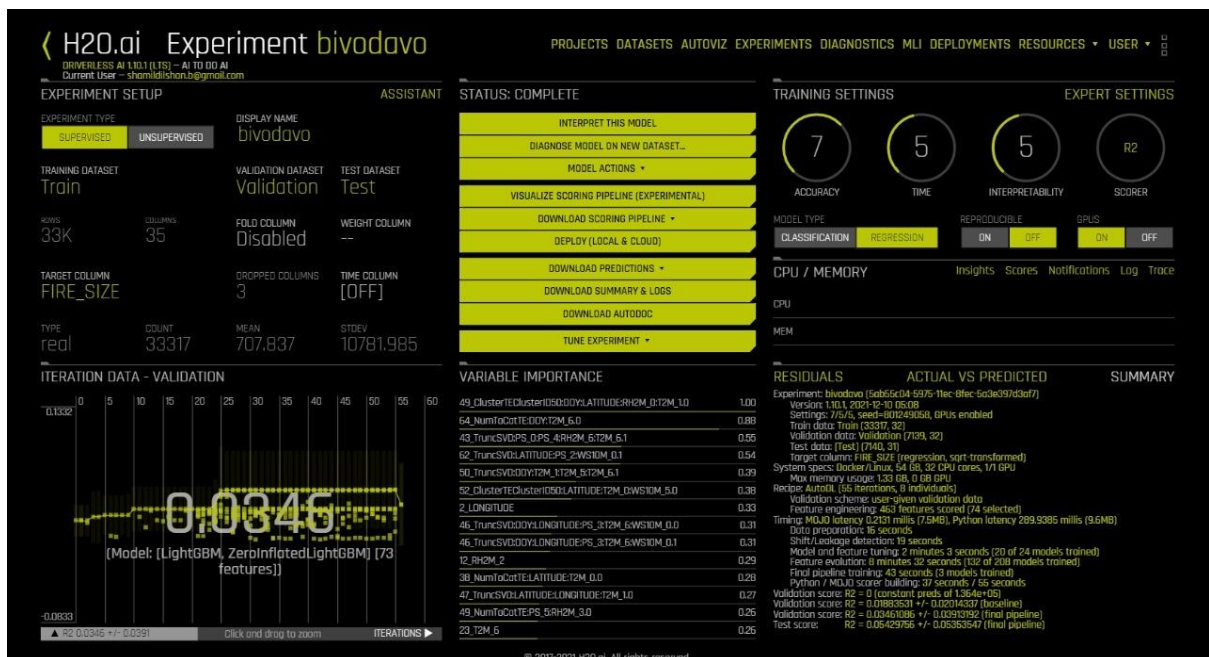


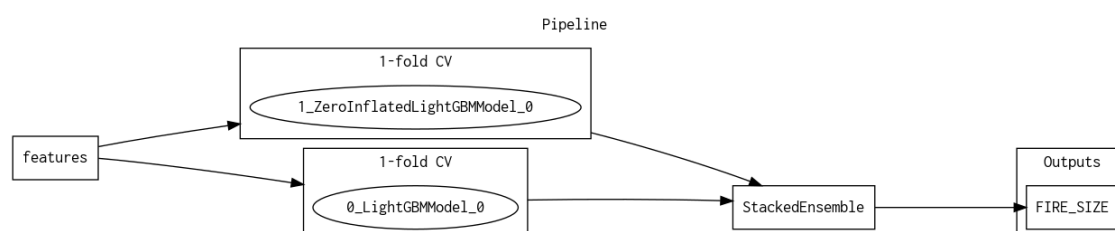
Figure 9 - Experiment Setup Results

“Autodoc” was generated and downloaded at the end of the experiment in order to identify the performance results for each model. The document results were used to determine the best model and its hyper-parameters. The best model that was considered for model building was “LightGBM”.

The model used in the application is LightGBM which is a fast-processing algorithm. The selected model is a gradient boosting framework that makes use of tree-based learning algorithms. This algorithm grows vertically which means leaf-wise. It chooses the leaf with large loss or grows, to lower down more in the next step. One of the perspectives of choosing LightGBM as our model is its’ lightness. It takes less memory to run, can deal with a large amount of data, and give good accuracy of results.

## Pipeline

Final StackedEnsemble pipeline with ensemble\_level=2 transforming 31 original features -> 74 features in each of 3 models each fit on external validation set then linearly blended:



- Model Index: 0 has a weight of 0.5 in the final ensemble

Type	grow policy	index	learning rate	max depth	Split Type	max leaves	colsample bytree	subsample	model class name	tree method
LightGBM Model	lossguide	0	0.03	5	External	32	0.45	0.7	LightGBM Model	

Figure 10 - Hyper Parameter for LightGBM model

Above hyper-parameters were then utilized to build the model LGB regression model. Following code snippet represents the relevant section of the notebook which was used to create the model.

```

import lightgbm as lgb
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn import metrics

data = pd.read_csv('Data_final_Dropped.csv')
data.head()

# To define the input and output feature
x = data.drop(['FIRE_SIZE', 'DISCOVERY_DATE'], axis=1)
y = data.FIRE_SIZE
# train and test split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_state=42)

# Model parameters got based on Driverless AI output
model = lgb.LGBMRegressor(learning_rate=0.03, max_depth=5, random_state=42, num_leaves=32, subsample=0.7, colsample_bytree=0.45)
model.fit(x_train, y_train, eval_set=[(x_test, y_test)], verbose=20, eval_metric='l2')

pred = model.predict(x_test)
print("Testing Accuracy (R Square Value),")
print(metrics.r2_score(y_test, pred))

lgb.plot_importance(model)
lgb.plot_metric(model)

model.booster_.save_model("wildfire_detector.model")

```

## Model Results

Following model results were taken after the model is saved. Training accuracy and testing accuracy in the following figure represents the **R2 Score**.

```

C:\Users\Shamil\anaconda3\envs\ML_Project\lib\site-packages\lightgbm\
a future release of LightGBM. Pass 'log_evaluation()' callback via 'ca
_log_warning("'verbose' argument is deprecated and will be removed i

[20]  training's l2: 1.12892e+08      valid_0's l2: 6.31603e+07
[40]  training's l2: 1.05989e+08      valid_0's l2: 6.28162e+07
[60]  training's l2: 1.00678e+08      valid_0's l2: 6.29359e+07
[80]  training's l2: 9.64645e+07      valid_0's l2: 6.33668e+07
[100] training's l2: 9.27741e+07      valid_0's l2: 6.39151e+07
Training accuracy 0.2442
Testing accuracy 0.0314

```

Figure 11 - Model Results

### Performance of Final Model

<b>Scorer</b>	<b>Better score is</b>	<b>Final ensemble scores on validation (internal or external holdout(s)) data</b>	<b>Final ensemble standard deviation on validation (internal or external holdout(s)) data</b>	<b>Final test scores</b>	<b>Final test standard deviation</b>
R2	higher	0.03461086	0.03913192	0.05429756	0.05353547
GINI	higher	0.8891321	0.01445112	0.8950952	0.0148617
MAE	lower	602.3961	101.0958	581.8617	111.8432
MAPE	lower	11580.52	1411.909	12526.36	1660.284
MER	lower	0	0.2302509	12.11784	7.080475
MSE	lower	8.128958e+07	5.133636e+07	7.805699e+07	5.133636e+07
R2COD	higher	0.01607631	0.01671084	0.02129041	0.01671084
RMSE	lower	9016.073	2001.861	8834.986	2335.224
RMSLE	lower	2.161031	0.01938655	2.174275	0.02108419
RMSPE	lower	174614.9	40862.74	218430.3	70200
SMAPE	lower	176.9048	0.5324957	176.7379	0.5324957



## Prediction Application

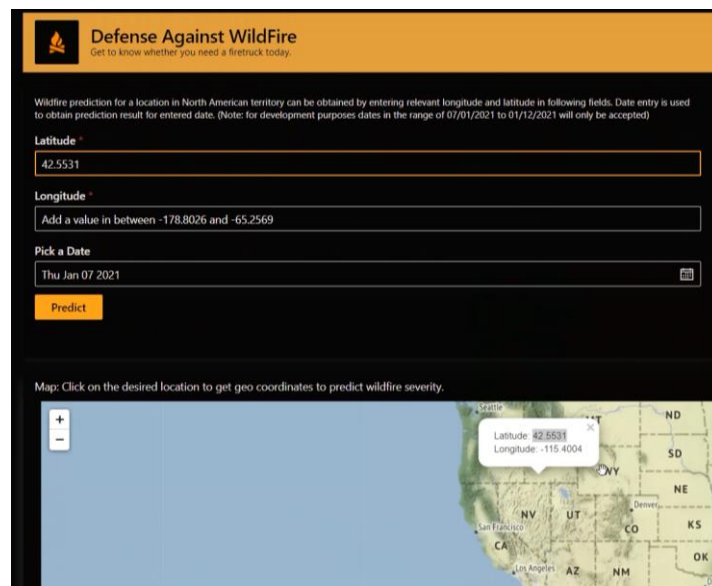
Application instance is deployed in H2O AI cloud and it can be accessed using the following URL.

- **Instance ID:** 001f7563-5768-437e-8553-b6c00f296ed2
- **URL:** <https://001f7563-5768-437e-8553-b6c00f296ed2.challenge.h2o.ai/>

### Inputs to the model

The prediction system accepts several inputs if the user needs to get predictions for a wildfire. Following image shows the user interface for inputting data which is required for the system (Model) to predict the severity parameter for the given geo-location.

Geo-Location can be viewed using the **“Mark on the Map”** or **“Converting Zip Code”** button and selecting the desired point using the graphical map of the region. After the point is selected, coordination data can be input to the system manually using Latitude & Longitude.



*Figure 12 - Selecting geo location using the Map*

The system needs user to pick a date which the user wants the system to predict the wildfire on the desired geo-location. Ideally the above application can be modified to fetch the current date and time from the Internet therefore the user doesn't get to input the date. Main reason the application has designed in this way manner is due to complexity in updating weather data automatically. After the user interface provide necessary values to fetch the weather data from a pre-saved dataset which is not being automatically updated.



Table 3 - Input Parameters for the Application by User

Parameter	Description	Input Range
Latitude	Latitude of the geo-location	17.9397 to 70.3306
Longitude	Longitude of the geo-location	-178.8026 to -65.2569
Zip code (Optional)	Text box which accepts zip codes in the United States – Value will be converted to geo-coordinates and used as the input parameters	Valid US zip codes only
Date	Date to fetch the weather data from pre-saved dataset	07/01/2021 to 01/12/2021

Figure 13 - User Interface of the Application

## Features of the Application

Application user experience has been improved after the first submission was submitted on the H2O wildfire challenge. Following features are implemented and deployed in the H2O cloud instance.

### **Zip/ Postal Code as the Geocode**

App was initially developed where user must enter valid longitude and latitude to get predictions. A map which was shown in the bottom portion of the screen is used to select and pin the point where geo-coordinates were shown. Since this task is time consuming, a function to translate valid United States zip codes to relevant address and geo location was added.

Once the user input the zip code, app automatically inputs the longitude and latitude for the relevant text boxes and marks the geo-location on the map in the bottom half of the screen. Following figures represent the app functionality.

---

*Note: Zip code translation was developed using Geo-Py service in the first place, but when application is deployed to H2O cloud it was having **connection timeout error**. As a workaround CSV file containing most of the US zip codes has been used to translate zip codes to geo-coordinates. Recently published zip codes won't translate into coordinates with this offline method.*

---

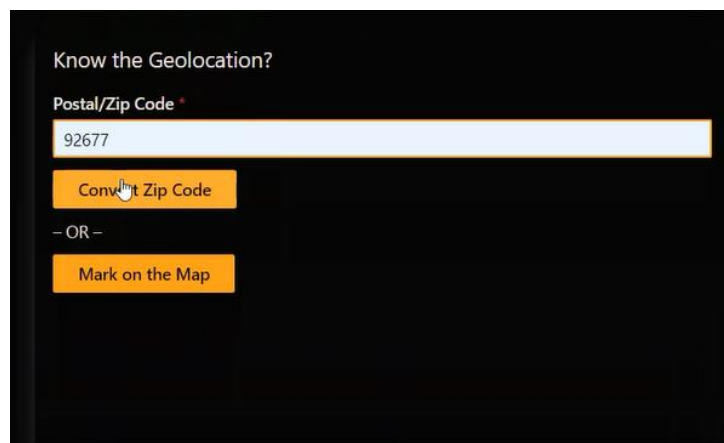
A screenshot of a web application interface with a dark background. At the top, the text "Know the Geolocation?" is displayed. Below it, the label "Postal/Zip Code \*" is followed by a text input field containing the value "92677". Underneath the input field is a yellow button labeled "Convert Zip Code". Below this button is the text "- OR -", followed by another yellow button labeled "Mark on the Map".

Figure 14 - Zip code input

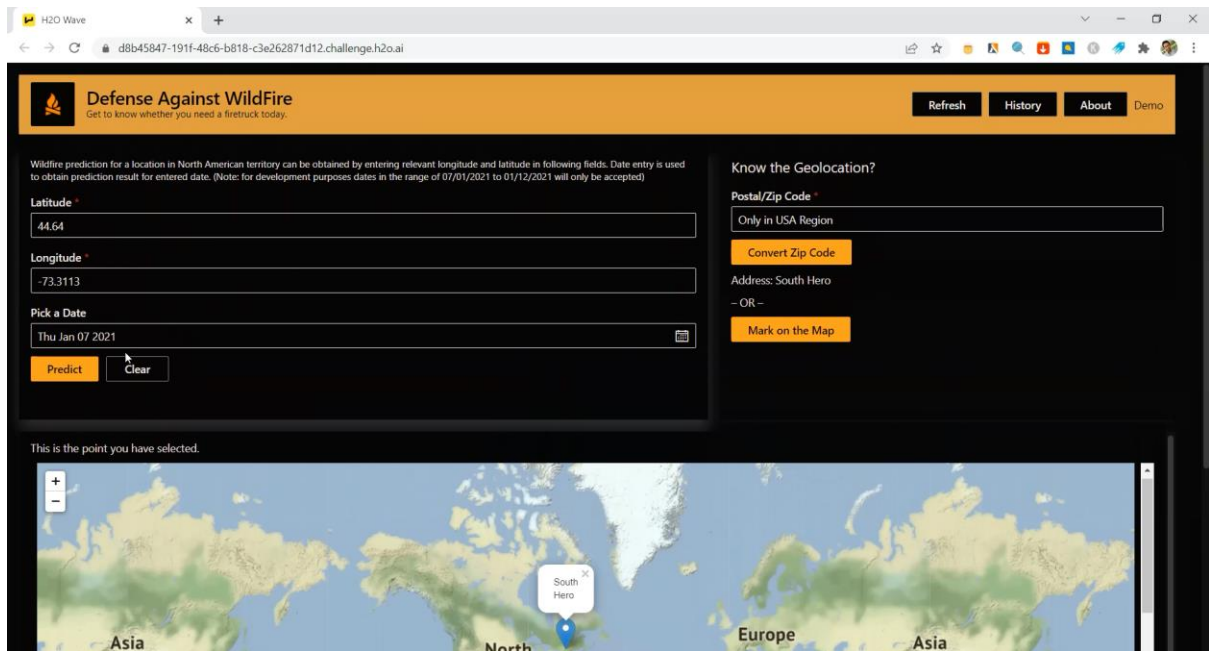


Figure 15 - Zip Code translated to geo-location

## Prediction Location History

Once the users get the predictions from the application for a location, it is stored in the application and visualized on the map. This is allowing users to keep track of previous locations which they have looked on wildfire predictions.

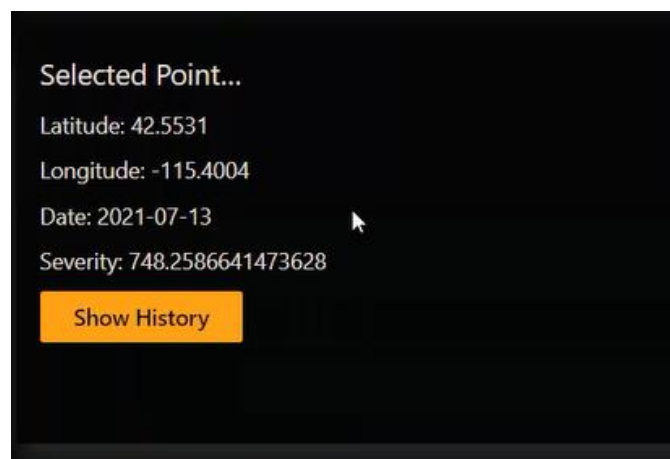


Figure 16 - Accessing history of predictions

Each location user enters is marked on the map with a **coloured marker** demarcating the wildfire severity with colours mentioned in the following table.

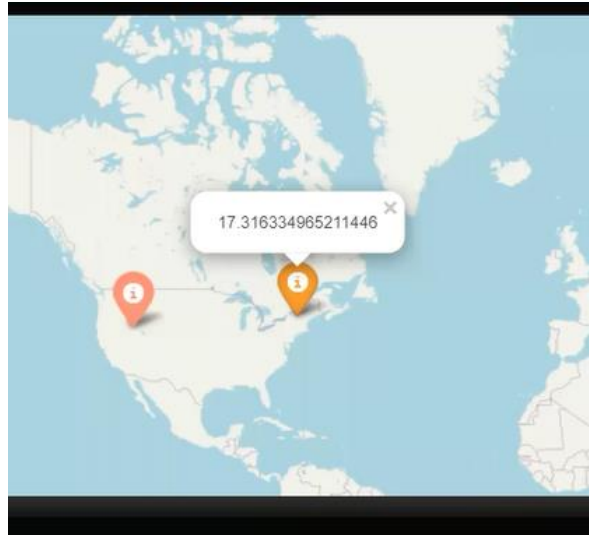


Figure 17 - Prediction history shown on the map

Table 4 - Fire Class and Demarcation Colour

Severity Value	Class	Class Colour
<b>0</b>	None	Dark Green
<b>0.1 – 0.2499</b>	Class A	Green
<b>0.25 – 9.99</b>	Class B	Light Green
<b>10 – 99.99</b>	Class C	Orange
<b>100 – 299.99</b>	Class D	Pink
<b>300 – 999.00</b>	Class E	Light Red
<b>1000 - 4999</b>	Class F	Red
<b>5000 and above</b>	Class G	Dark Red

## Exception Handling

The application has following set of limitations and those limitations has been checked prior the prediction result is being fetched.

1. US region dataset has ONLY been used to train the ML model thus geo-points that can be entered are only within a limited area. (Latitude: 17.9397 to 70.3306 | Longitude: -178.8026 to -65.2569)
2. Automatic LIVE weather data acquisition function has not been implemented. Hence, valid dates which can be entered to the app is from 07.01.2021 to 01.12.2021

**Defense Against WildFire**  
Get to know whether you need a firetruck today.

Wildfire prediction for a location in North American territory can be obtained by entering relevant longitude and latitude in following fields. Date entry is used to obtain prediction result for entered date. (Note: for development purposes dates in the range of 07/01/2021 to 01/12/2021 will only be accepted)

**Latitude \***  
Add a value in between 17.9397 and 70.3306

**Longitude \***  
Add a value in between -178.8026 and -65.2569

**Pick a Date**  
Thu Jan 07 2021

**Predict**

**Know the Geolocation?**  
Postal/Zip Code \*  
Only in USA Region  
Convert Zip Code  
- OR -  
Mark on the Map

**ERROR! Fill all blank areas**

Figure 18 - Empty input field error handler

**Defense Against WildFire**  
Get to know whether you need a firetruck today.

Wildfire prediction for a location in North American territory can be obtained by entering relevant longitude and latitude in following fields. Date entry is used to obtain prediction result for entered date. (Note: for development purposes dates in the range of 07/01/2021 to 01/12/2021 will only be accepted)

**Latitude \***  
Add a value in between 17.9397 and 70.3306

**Longitude \***  
Add a value in between -178.8026 and -65.2569

**Pick a Date**  
Thu Jan 07 2021

**Predict**

**Know the Geolocation?**  
Postal/Zip Code \*  
Only in USA Region  
Convert Zip Code  
- OR -  
Mark on the Map

**ERROR! Geolocations are out of range**

Figure 19 - Out of range geo-location error handler

**Defense Against WildFire**  
Get to know whether you need a firetruck today.

Wildfire prediction for a location in North American territory can be obtained by entering relevant longitude and latitude in following fields. Date entry is used to obtain prediction result for entered date. (Note: for development purposes dates in the range of 07/01/2021 to 01/12/2021 will only be accepted)

**Latitude \***  
Add a value in between 17.9397 and 70.3306

**Longitude \***  
Add a value in between -178.8026 and -65.2569

**Pick a Date**  
Thu Jan 07 2021

**Predict**

**Know the Geolocation?**  
Postal/Zip Code \*  
Only in USA Region  
Convert Zip Code  
- OR -  
Mark on the Map

**ERROR! Enter geolocations in valid format**

Figure 20 - Invalid Zip/ Postal Code input error handler

### **Demo Video for New Users**

Application is included with a demonstration video where the new users can view and understand the basic functionality of the application. It can be accessed through the application by clicking the “Demo” hyperlink or visiting following URL.

URL:

<https://drive.google.com/file/d/1ku94g6s0lCYsipLuMcTtIOL06SI435Ek/view?usp=sharing>

## Prediction from the Application

As the user provides necessary values to the application, model which has been trained in the application fetches the relevant weather data. The application requires weather data for past 7 days from the user picked date. Application is defined to collect all the weather data into one row and then provide them into the model.

The model out is produced as the severity value which starting from zero and zero represents that there is no risk of a wildfire event considering the geo-location and weather **data for past 7 days from the user picked date**. If there is a severity value presented it can be further understood by referring following table. (Table 5 - Wildfire Size Categorization)

**Defense Against WildFire**  
Get to know whether you need a firetruck today.

Refresh History About Demo

Wildfire prediction for a location in North American territory can be obtained by entering relevant longitude and latitude in following fields. Date entry is used to obtain prediction result for entered date. (Note: for development purposes dates in the range of 07/01/2021 to 01/12/2021 will only be accepted)

Latitude \*  
Add a value in between 17.9397 and 70.3306

Longitude \*  
Add a value in between -178.8026 and -65.2569

Pick a Date  
Thu Jan 07 2021

Predict

Selected Point...  
Latitude: 42.5531  
Longitude: -115.4004  
Date: 2021-07-13  
Severity: 748.2586641473628  
Show History

Know the Geolocation?  
Postal/Zip Code \*  
Only in USA Region  
Convert Zip Code  
- OR -  
Mark on the Map

Severity Value	Class	Class Description
0	None	No fire hazard
0.1 – 0.2499	Class A	one-fourth acre or less
0.25 – 9.99	Class B	more than one-fourth acre, but less than 10 acres
10 – 99.99	Class C	10 acres or more, but less than 100 acres
100 – 299.99	Class D	100 acres or more, but less than 300 acres
300 – 999.00	Class E	300 acres or more, but less than 1,000 acres
1000 - 4999	Class F	1,000 acres or more, but less than 5,000 acres
5000 and above	Class G	5,000 acres or more

Figure 21 - Output from the Application (Severity)

Table 5 - Wildfire Size Categorization

Severity Value	Class	Class Description
0	None	No fire hazard
0.1 – 0.2499	Class A	one-fourth acre or less
0.25 – 9.99	Class B	more than one-fourth acre, but less than 10 acres
10 – 99.99	Class C	10 acres or more, but less than 100 acres
100 – 299.99	Class D	100 acres or more, but less than 300 acres
300 – 999.00	Class E	300 acres or more, but less than 1,000 acres
1000 - 4999	Class F	1,000 acres or more, but less than 5,000 acres
5000 and above	Class G	5,000 acres or more

## References

- [1] R. Tatman, "1.88 Million US Wildfires," 13 05 2020. [Online]. Available: <https://www.kaggle.com/rtatman/188-million-us-wildfires>.
- [2] NASA Langley Research Center, [Online]. Available: <https://power.larc.nasa.gov/data-access-viewer/>.
- [3] Meteoblue, "MeteoBlue," [Online]. Available: <https://content.meteoblue.com/en/specifications/weather-variables/precipitation>. [Accessed 20 January 2022].
- [4] National Wildfire Coordinating Group, "Size Class of Fire," [Online]. Available: <https://www.nwcg.gov/term/glossary/size-class-of-fire>.