Machine Learning for Binary Classification of Wildfire Size

Dimitar Milenov Angelov
STUDENT NUMBER: 2023197

THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
BACHELOR OF SCIENCE IN COGNITIVE SCIENCE & ARTIFICIAL INTELLIGENCE
DEPARTMENT OF COGNITIVE SCIENCE & ARTIFICIAL INTELLIGENCE
SCHOOL OF HUMANITIES AND DIGITAL SCIENCES
TILBURG UNIVERSITY

Thesis committee:

Dr. Sharon Ong
Dr. Dimitar Shterionov

Tilburg University
School of Humanities and Digital Sciences
Department of Cognitive Science & Artificial Intelligence
Tilburg, The Netherlands
June 2021

Preface

Completing this research and writing the thesis to accompany it has been quite the process for me. I certainly faced challenges, most of them self-imposed in nature, but that is what has allowed me to grow. I would like to take a moment to extend my thanks to those who have helped me through this process. Not only in my thesis, but my time so far at Tilburg University as a whole.

First and foremost, I would like to thank my thesis advisor, Dr. Sharon Ong, for their invaluable guidance at every step of the way. They've been nothing but an excellent mentor and I was incredibly fortunate to have been taken under their wing during for the duration of this thesis.

I would also like to extend my humble thanks to the academic staff of the TSHD department at TiU as a whole. Everyone I have had the pleasure to interact with has been helpful, enthusiastic, and always willing to take a moment to humor my silly questions.

Finally, I would like to thank those closest to me in my personal life. My parents, for never doubting me, and always supporting me in my endeavours. My friends, for always being by my side, and giving me a hand every time I stumble. Last, but not least, my S.O., for all of the emotional support she has given me.

Machine Learning for Binary Classification of Wildfire Size

Dimitar Milenov Angelov

The use of machine learning (ML) classifiers in making predictions about a wildfire's size was investigated. A novel public dataset was added onto, preprocessed, and used to train and evaluate 7 ML architectures. The ML techniques were chosen based on the prior research in the area. A LightGBM model with 1000 Estimators and Maximum Depth of 16 showed an accuracy of 69.5% and F1-score of 70.5%, a 20% increase over the baseline method. Random Forest models were used to explore feature importance. Spatial features were found to be the most important, followed by features describing the meteorological conditions prior to the start of the fire. Vegetation information was found to be comparatively unimportant to making accurate predictions.

1. Introduction

1.1 Project Definition

Wildfires are a major and widespread environmental, wildlife, and economic issue. They are a major driver of greenhouse gas emissions and are responsible for 5-8% of the 3.3 million annual premature deaths from poor air quality (Lelieveld et al. 2015). As the global temperature continues to rise, we are set to see wildfires of larger size and greater intensity. A review done by Jones et al. (2020) found that "climate change increases the frequency and/or severity of fire weather – periods with a high fire risk due to a combination of high temperatures, low humidity, low rainfall and often high winds".

The overarching goal of this research was to explore the extent that machine learning can predict the category of wildfires from remote sensing data. Wildfires are classified into one of 7 categories (classes A through G) relating to the area burned from the fire. See section 1.2 of the appendix for more information on the categories.

The dataset used had samples of over 50,000 fires from classes B through G. However, due to the heavily-skewed data (as described in detail in the Methods section), the wildfire classes were combined into the 'B' class and all larger classes. This changed the classification problem from one with six categories into a binary one with only two categories.

1.2 Motivation

Due to the reasons stated above, fire-fighting resource management and allocation is becoming increasingly important to containing and fighting wildfires. In cases where several wildfires have been identified in a similar region, fire inspectors and land managers must decide how much of their limited resources they should send to each location.

There have been studies which attempted to quantify the effects and interactions of different variables on wildfire size, which found there are complex non-linear interactions between the features and the size of the fire (Cary et al. 2006; Slocum et al. 2010; Parisien and Moritz 2009). Slocum et al. (2010) performed analysis on wildfire size and such variables using quantile regression and found that "different climate conditions served as critical thresholds, influencing wildfire size at different spatial scales".

There have been attempts to solve this problem with mathematical models (Rothermel 1970), probabilistic models such as the ones in the FSPro system, and regression models (Slocum et al. 2010). However, a good predictive model would have the potential of enabling fire inspectors to make more informed decisions in a timelier manner. Due to the large number of variables and the complex non-linear interaction effects between them and the size of a fire, machine learning classifiers are a tool to be considered in this space. Machine learning has already been used in wildfire management research, as well as the specific area of predicting the size of a wildfire from remote-sensed data. Cortez and Morais (2007) explored the performance of several ML models on a small dataset containing spatiotemporal and meteorological features. Calp and Kose (2020) evaluated an ANN on the same dataset, and Sayad, Mousannif, and Moatassime (2019) added more features to it and evaluated SVM and MLP classifiers.

1.3 Research Question

The following overarching research question stood at the core of this research:

Research Question:
"To what extent can machine learning methods classify a wildfire's size?"

There are several features in the dataset such as the place and time of the fire, meteorological factors such as temperature and humidity, as well as topographic data describing the elevation, vegetation, and remoteness of the place of the wildfire. The features are described in detail in section 1.5 of the appendix.

Due to the diverse nature of these features, the only way they could be gathered is from a diverse set of sources. Adding multiple data sources increases the complexity and work required to set up the pipelines to gather data reliably and in a timely manner. Therefore, it will be valuable to explore the importance of the features in predicting a wildfire's size category. This is a factor which has also been prominently explored in prior research (Fang et al. 2015; Wang and Wang 2020).

A limitation of this prior research was the limited variety of ML models they used – perhaps different ML models will find more benefit from certain features. Therefore, a sub-question regarding the feature importance was formed.

Sub-question:
"What impact do different data features have on the results of the models?"

In order to do this, three subsets of features were selected and the models were trained on them, as well as on the full set of features. The resulting datasets and the features of each of them is described in section 1.5 of the appendix. A Feature correlation analysis of each data subset was performed (11b , 11c, 11d) and a correlation matrix of all features was plotted 2.

1.4 Summary of Contributions

This research attempted to answer the questions of "What is the best machine learning algorithm for fire size prediction?" and "What are the most informative features when making such a prediction?". Due to the disbalance of the dataset, the six-class task was transformed into a two-class, binary, task.

8 classification algorithms were explored, including most of the ones found in previous research, as well as the Light Gradient Boosting (LightGBM) algorithm, the performance of which has not been evaluated for this task before. It also includes a large range of features, which have not been evaluated together in prior research, as well as one which has not been used before - namely the remoteness of the location of the fire.

It was found that the LightGBM and Random Forest algorithms have the highest performance, however the LightGBM algorithm an order of magnitude quicker to train and requires less computation. The fact it had an overall better performance, and was quicker to train, makes it the algorithm to use for this task.

The final best performing algorithm found was LightGBM on the full dataset, with 20% improvement over the baseline. It had an accuracy of 0.69, with a precision and recall of 0.71 (f-score 0.71), and a training time of 0.941 seconds on an Intel i7-4790 processor.

The most important features, in descending order, were found to be the meteorological information (combined), latitude/longitude (combined), remoteness & elevation.

2. Related Work

Fire-fighting resource management and allocation is becoming increasingly important to containing and fighting wildfires. In cases where several wildfires have been identified in a similar region, fire inspectors and land managers may have to make decisions regarding how much of their limited resources they should send to each location.

Prior work has been done which attempted to quantify the effects and interactions of different variables on wildfire size, which found there are complex non-linear interactions between the features and the size of the fire (Cary et al. 2006; Slocum et al. 2010; Parisien and Moritz 2009). Slocum et al. (2010) performed analysis on wildfire size and these variables using quantile regression and found that "different climate conditions served as critical thresholds, influencing wildfire size at different spatial scales".

There have been attempts to tackle the task of wildfire size prediction with mathematical models (Rothermel 1970), probabilistic models (such as FSPro), and regression models (Slocum et al. 2010). However, a good predictive model would have the potential of enabling fire inspectors to make more informed decisions in a timelier manner. Due to the large number of variables and the complex non-linear interaction effects between them and the size of a fire, machine learning classifiers are a tool which has been, and continues to be, considered in this space.

In prior research, ML classification techniques have been used in wildfire management research, as well as specifically for predicting the size of a wildfire from remote-sensed data. A review of the use of ML in wildfire management as a whole was done by Jain et al. (2020). Cortez and Morais (2007) explored the performance of a number of ML models on a small dataset containing spatiotemporal and meteorological features. Calp and Kose (2020) evaluated another technique - an Artificial Neural Network (ANN), on the same dataset. In addition, (Sayad, Mousannif, and Moatassime 2019) added more features to the dataset initially made by (Cortez and Morais 2007) and evaluated a Support Vector Machine (SVM) and Multi-Layer Perceptron (MLP). Feature importances when making predictions were explored for two other datasets by Tonini et al. (2020) and Fang et al. (2015).

One of the main challenges in creating these classification models has been the unevenly distributed and highly correlated data. ML models which perform well when modelling highly complex data have been found to perform best for this task. (Cortez and Morais 2007; Sayad, Mousannif, and Moatassime 2019)

3. Methods

3.1 Dataset

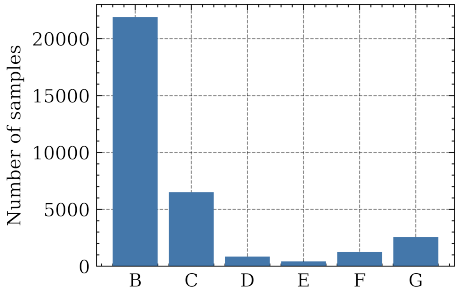
The dataset used was a combination of a publicly available dataset of 50,000 U.S. wildfires and elevation information taken from the Open-Elevation project.[1]

The fire dataset was published on Kaggle (Ramesh 2020). It consists of a 50,000 fire random subset of the 1.88 million U.S. wildfires dataset, also published on Kaggle (Short 2017). Although it has a smaller number of samples, it is still more than sufficient for training and testing the machine learning algorithms described in the Classification Algorithms subsection. The dataset by Ramesh (2020) has the advantage of having a large number of features which are not present in the original dataset by Short (2017).

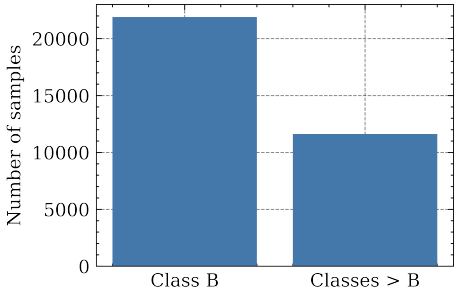
1 Open-Elevation Website

Figure 1: Plot of label distributions of the full dataset.

(a) All classes.



(b) Label distribution with all fire sizes greater than B combined.



The features include historical weather information (humidity, precipitation, temperature, and wind speed), historical vegetation data (28 types), and the spatiotemporal information regarding the fire (latitude/longitude, date, remoteness). Information for the vegetation features was interpolated from the research work done by (Meiyappan and Jain 2012) using the latitude and longitude of the location of the fire. The remoteness feature represents a non-dimensional distance to the nearest populated place. Information about the towns and cities was taken from Simplemaps' World Cities Database. [2] The rest of the methods by which the feature information was gathered and the specifics of what each feature represents are described in more detail by Ramesh (2020).

The elevation information was found to be an important feature by Tonini et al. (2020) in their findings on feature importance regarding wildfire classification. Therefore, efforts were taken to gather information about the elevation at the latitude/longitude location of each fire and include it as one of the topological features. In order to do this a Python program was written which would take each latitude/longitude and pull the information from the public Open-Elevation API.

In addition, the date information was used to create another feature which points to whether the fire started during the weekend or not - a feature which was also explored in prior work (Cortez and Morais 2007).

A problem which has been noted and explored by prior research was also present in this data. The data is severely skewed toward the smaller fires, with 65% of the samples belonging to the B class - the smallest wildfire size featured in the dataset. This is a positive skew in the same nature of what has been observed in Canada (Malarz, Kaczanowska, and Kułakowski 2002), China (Fang et al. 2015), Portugal (Cortez and Morais 2007), and Italy (Tonini et al. 2020). You can see the label distribution for this dataset in figures 1a and 1b.

2 Simplemaps World Cities Database

3.2 Feature Selection

One of the techniques used to evaluate the importance of features was to split the features into three feature subsets based on their source and the type of information they provide. The three datasets and the features they contained are briefly described here. For more detailed information you can look at table **??** in the appendix.

- Full dataset. (**F**)
  - Contains all of the features.
- Meteorological dataset. (**M**)
  - Only contains meteorological features - temperature, humidity, etc.
- Spatiotemporal dataset. (**SPT**)
  - Only contains features related to the place and time of the fire - latitude/longitude, season, etc.
- Topological dataset. (**T**)
  - Only contains features related to the topology at the location of the fire - the vegetation & elevation.

This was done with practical applications in mind. One can assume that the information regarding a fire can, in certain situations, be limited in scope. This can happen due to multiple reasons, but one of the most likely ones is that one of the sources of information for an area is not operational at the time of a fire.

For example, a full range of meteorological data will usually come from one source - information regarding humidity, precipitation, temperature, etc. Therefore, it would not be useful to evaluate these features separately (i.e., how a model will perform if given the temperature, but not the wind speed).

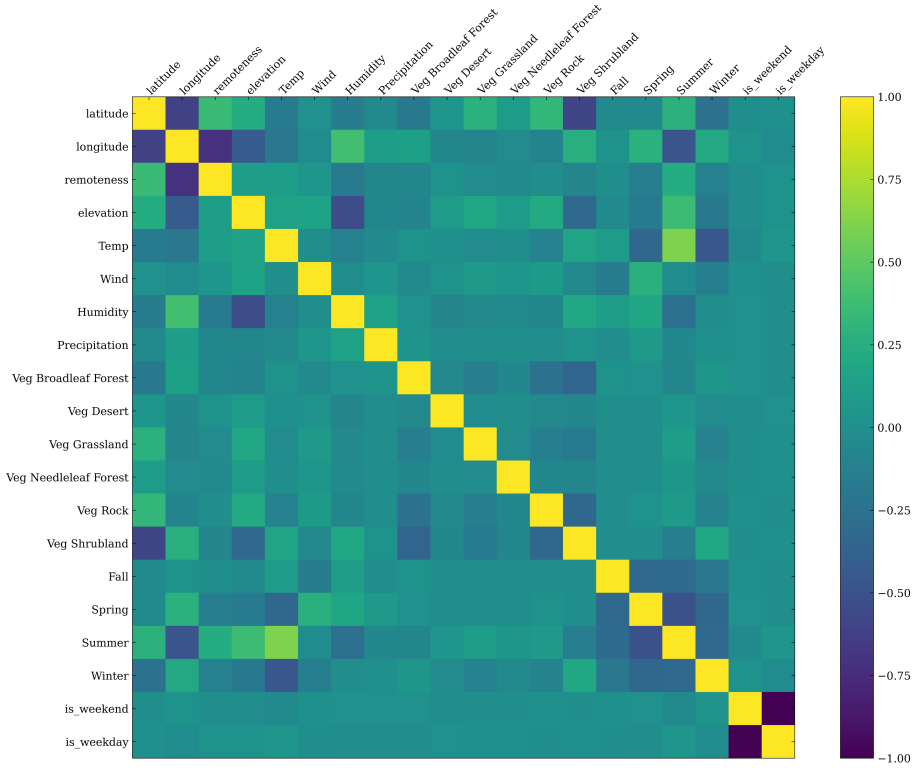
In order to evaluate the *relative* importance of features, a feature importance plot was created for each of the datasets. In order to do this, a Random Forest model trained on each dataset was used in combination with the feature_importance_ attribute provided by scikit-learn.[3] The importances were evaluated based on the mean decrease in impurity, as described in Breiman et al. (1983). The final feature importances were plotted, and are shown in figure 11, in the results section.

3.2.1 Feature Correlation. A Pearson's pairwise correlation matrix was created for the features of the **F** dataset. The implementation provided by the corr attribute of DataFrames within Pandas was used.[4] You can see the plot in figure 2.

It was used during the feature selection, as a check for highly correlated features. Feature correlation matrices can also be used to show a causal relationship. For example, we see a correlation between the Summer variable and the temperature.

3 Scikit-learn feature_importance_ documentation.
4 Pandas corr documentation.

Figure 2: Pairwise Correlation Matrix of all Features



1

3.3 Data Preparation

A number of steps were taken to prepare the raw data. The steps taken are described in the following subsections, but their general purpose was to transform the original dataset in such a way that it would allow for the best performance of the algorithms.

The final, prepared, datasets contained 23236 samples with 2 labels and 20 features.

3.3.1 Data Cleaning. The data cleaning involved identifying and removing any samples which contain no values. The main source of these were the meteorological features, which were missing in about 32% of the samples. Samples which contained NA values, values of '-1', or values in which all the meteorological data values were equal to '0' were removed. This initial step reduced the total dataset size from 55367 samples to 37247 samples.

3.3.2 Dimensionality Reduction. In general, one may assume that the higher the number of features of a dataset, the greater the information available to the algorithms, and so the greater the accuracy of the resulting model. However, it has been found that machine learning models have a threshold of dimensionality, which is dependent on

the model and dataset, above which their performance (in terms of accuracy) will begin to deteriorate (Trunk 1979; Hughes 1968).

This is especially important for categorical features, as it is necessary to one-hot encode them. The result of this process is that each label becomes a separate feature, which can cause the dimensionality to increase significantly.

Therefore, intuitive techniques to reduce the number of features were employed. The vegetation categories, of which there was 28 in total, as described in Ramesh (2020), were consolidated into 12 groups to reduce the number of categorical features of the dataset. For the same reason, the months were grouped into seasons. You can see detailed information on how the categories were consolidated in sections 1.3 and 1.4 of the appendix.

3.3.3 Data encoding and Scaling. The categorical features were one-hot encoded using the Pandas implementation for dummy encoding.[5]

All continuous features were scaled into the $0-1$ range using the scikit-learn MinMaxScaler implementation.[6]

3.3.4 Data Balancing. In figure 3, it was shown that the labels have a very positively skewed distribution. This was mitigated by reducing the number of labels from six to two, however, as shown in figure 1b, the B class still has a significantly larger number of samples than the combined class. To address this further, a random undersampling technique was used for the training data, in which a selection of random samples taken from the larger class was created until it reached the size of the smaller class. As a result, both classes have an equal number of samples.

The implementation provided in the RandomUnderSampler class of the imbalanced-learn package was used.[7]

3.4 Classification Algorithms

The algorithms described in the following subsections were evaluated for this research. The algorithms which have been used in prior research most commonly were chosen. The implementations of the algorithms provided in the scikit-learn package, version 0.24.2, was used (Pedregosa et al. 2011). A large range of hyper-parameters were evaluated. These are described in table 1.

The values for the hidden layers of the MLP were the most difficult to determine, since there is such a large possible range. At the start small single-layer networks were evaluated (with up to 50 perceptrons), however it quickly became apparent that the network did not achieve a sufficient accuracy on the training set, thus larger and larger networks were tested. It has been shown that a single-layer MLP can approximate any arbitrary function (Hornik 1991), the process of determining the ideal number of

5 Pandas one-hot encoding documentation.
6 Scikit-learn MinMaxScaler documentation.
7 Imbalanced-learn RandomUnderSampler documentation.

neurons is difficult and very time consuming in practice. More layers were added to allow for an easier representation of the interactions within the features. It also allows for more abstract features to be learned and subsequently used as input for the next hidden layer.

Each of the final, preprocessed, datasets got split into a training and testing set in a 9-1 ratio. All of the models were trained on the training set and their performance was evaluated on the testing set. The labels in the training set were balanced using a random undersample, but the testing set was left as is, so as to be representative of the distribution of fires in the real world.

Grid Search with 10-fold cross-validation was used to train the algorithms and to determine the best hyper-parameters, shown in table 1.[8] Using 10-fold cross-validation ensures that the results are representative of the overall dataset.

The models were trained on the CPU of a desktop computer equipped with an Intel i7-4790 (4-core, 8-thread, 4.0GHz) CPU.

Table 1: All hyper-parameters which were tested for each model.

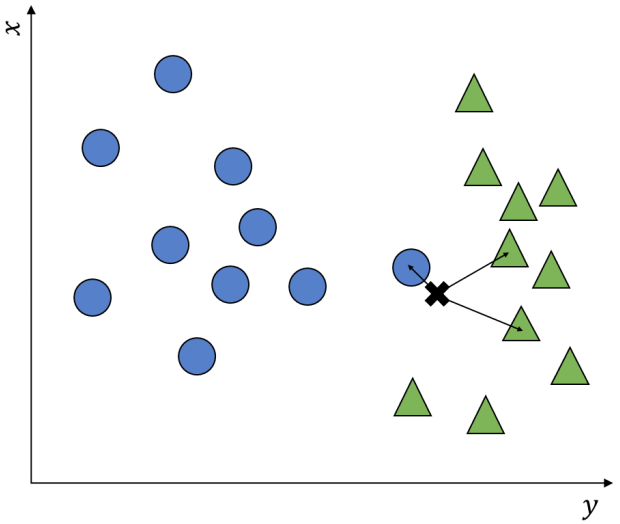
Algorithm	Hyper-parameters tested
Dummy Classifier	-
K-Nearest Neighbours	k: Odd numbers between 1 and 100 Weights: Uniform & Distance
Gaussian Naïve Bayes	-
Support Vector Machine	C: 0.01, 0.1, 0.25, 0.5, 1, 2 Kernel: poly, rbf, sigmoid
Decision Tree	Max Depth: 2, 4, 16, 64, 128, 256, None Criterion: Giny, Entropy
Random Forest	Max Depth: 2, 4, 16, 64, 128, 256, None Criterion: Giny, Entropy Estimators: 100, 500, 1000
LightGBM	Max Depth: 2, 4, 16, 32, 64, None Estimators: 10, 50, 100, 250, 500, 1000, 2500
MLPClassifier	Activation: Tanh, ReLU, Logistic Alpha: 0.0001, 0.05 Hidden Layer Sizes: (5), (50), (100), (250), (1000), (10, 10), (25, 25), (50, 50), (100, 100), (250, 250)

3.4.1 Dummy Classifier. In supervised learning, it is common to have a baseline classifier against which one compares the results of more complex estimators. Usually, they will be simple rules, such as always picking the most common class or randomly choosing from some prior distribution set by the researchers. The DummyClassifier of Sci-Kit Learn has implementations of several such simple classification strategies and serves as the baseline against which we compare the rest of the classification algorithms.

The strategy chosen was “stratified”, which will generate random predictions with a distribution respecting the label distribution of the training set. Since we under-sample to balance the dataset, this effectively means that one of the two classes will be randomly chosen as the predicted label.

8 Scikit-learn Grid Search Documentation

Figure 3: Visualization of the K-Nearest Neighbours Classifier
Note that for an unweighted KNN classifier, the X point would be labelled as belonging to the triangle class. However, in a weighted KNN classifier, the circle-labelled data point would be weighted more and X would be classified as a circle.



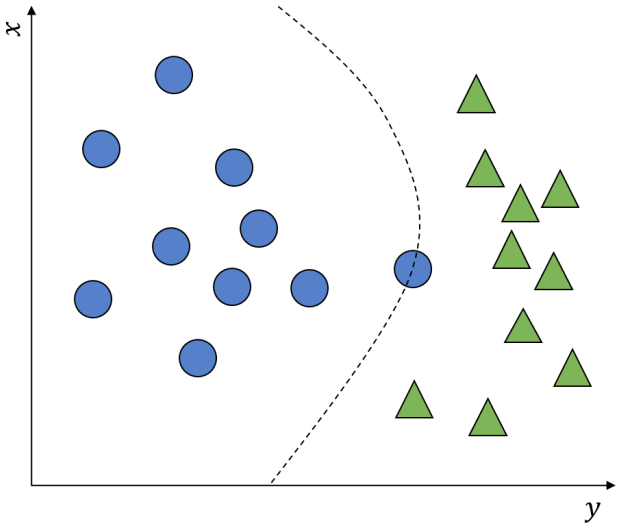
3.4.2 K-Nearest Neighbours. The K-Nearest Neighbours (KNN) is a conceptually simple, yet powerful, supervised machine learning algorithm (Cover and Hart 1967). First, a value k is specified, which selects the number of 'neighbours' a data point should take into account when performing classification (Goldberger et al. 2005). For example, if we were to set $k = 3$, the algorithm would take the three data points closest to the one it is trying to label, and looks at their classes. If there is a majority in the data points, it classifies the unassigned data point as the majority class.

One of the most common ways to choose which k points are closest is via their Euclidean distance ($dist\,(x,\ y) = \sqrt{\sum_{i=1}^{n}\left(x_i - y_i\right)^2}$). KNN networks can also leverage this distance and improve their performance by giving greater importance to closer points' labels. This is called a weighting scheme, and was one of the features, along with k, evaluated when training this classifier. You can see figure 3 for a visualization of the decision plane of a KNN classifier with $k = 3$.

3.4.3 Naïve Bayes. There are several supervised learning algorithms which are based on the application of Bayes' theorem $P\left(A|B\right) = \frac{P(B|A)P(A)}{P(B)}$ with a "naive" assumption of conditional independence between features. This set of methods is referred to as the Naïve Bayes classifiers ().

Naïve Bayes (NB) classifiers are some of the most computationally simple classifiers commonly used for Machine Learning. Despite their relative simplicity and low data requirements, naïve bayes classifiers have been demonstrated to work well in many real-world applications, such as document classification and spam filtering (Mccallum

Figure 4: Visualization of the Naïve Bayes Classifier
Notice the non-linearity exhibited.



and Nigam 1998; Metsis and Paliouras 2006). You can see a visualization of the decision boundary a Naïve Bayes classifier in figure 4. The formula in (1) is a mathematical representation of a Bayes classifier.

$$\hat{y} = \underset{y}{\operatorname{argmax}} \; P(y) \prod_{i=1}^{n} P(x_i \mid y) \tag{1}$$

The differences between the assumptions various naïve Bayes classifiers make about the distribution of $P(x_i \mid y)$ are what distinguishes them.

In figure 4 there is a visualization of the approximate decision boundaries a naïve classifier makes when performing binary classification. Notice the non-linearity exhibited.

3.4.4 Support Vector Machine. Support vector machines (SVM) are supervised learning algorithms which can be used for classification, regression, or other tasks. They create a set of decision boundaries in order to find the one which maximizes the distance to the nearest data points of any class (called support vectors), intuitively achieving a good separation of the classes. Generally, the larger the distance to the support vectors, the lower the error of the model. (Cortes and Vapnik 1995)

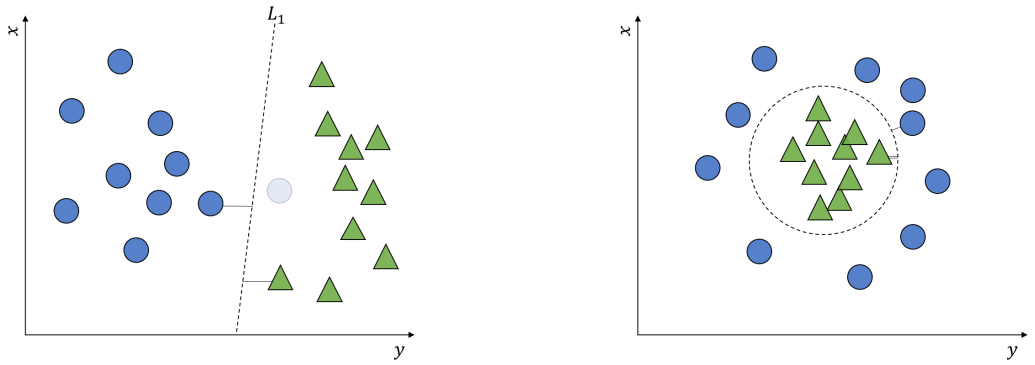
In cases where the data is not linearly separable, one of several types of kernels can be applied. Kernels will transform the classification space by adding a dimension, which can effectively allow non-linear separations to be made.

Support vector machines are effective in high dimensional spaces, such as classification tasks in which there is a large number of features. Thanks to the possibility of

Figure 5: Visualizations of Support Vector Machine Classifiers.

(a) Linear SVM.
Notice the outlier circle has which is not being considered due to regularization.

(b) Kernel SVM.



specifying a kernel, they are very versatile. However, they can suffer from a lack of tolerance to outliers when no regularization (C) is applied.

You can see figures 5a and 5b for visualizations of Linear and Kernel SVMs respectively.

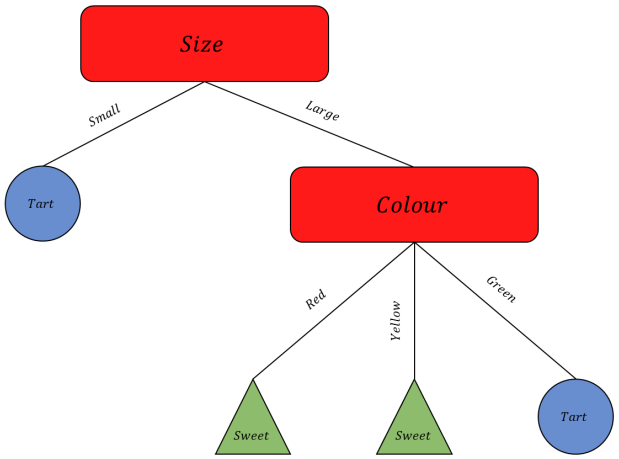
3.4.5 Decision Tree. The decision tree (DT) classifier (or classification tree) continuously splits the data according to a set of features. It begins with a 'root' node, and has a number of branches (features) and leaves (labels). Data is split along branches with the goal of maximising information gained per split (different measures of information gained can be used). The data continues to be split until the first of several scenarios occurs: a pre-specified maximum depth is reached; a branch has all the same values of leaves; or when splitting no longer adds value to the predictions (Morgan and Sonquist 1963).

Decision trees are relatively simple to understand and interpret due to the boolean logic used to make decisions. They are computationally simple and quick to both train and infer. One of the main hurdles for decision trees is the fact they are prone to overfit to the training set. There are several ways to minimize this, the most common and straightforward of which is setting a maximum number of splits (depth).

You can see figure 6 for a simple decision tree for a binary classification task – whether an apple is sweet or tart. There are two features (size, colour).

3.4.6 Random Forest. The random forest (RF) algorithm is an ensemble learning model – it uses the output of a number of other models in order to make predictions (Ho 1995). RFs use a large number of shallow decision tree models, each of which is trained on a bootstrap-sampled subset of the training set. Furthermore, when deciding the split for each branch during the training of the trees, the best split is found from a random subset of size $max\_features$. Generally, on inference, each decision tree will produce a

Figure 6: Visualization of the Decision Tree Classifier.
A binary classification task – whether an apple is sweet or tart.
There are two features (size, colour).



prediction about the input, and the class which was picked the highest number of times will be chosen as the output. You can see a visualization of a random forest model in figure 7.

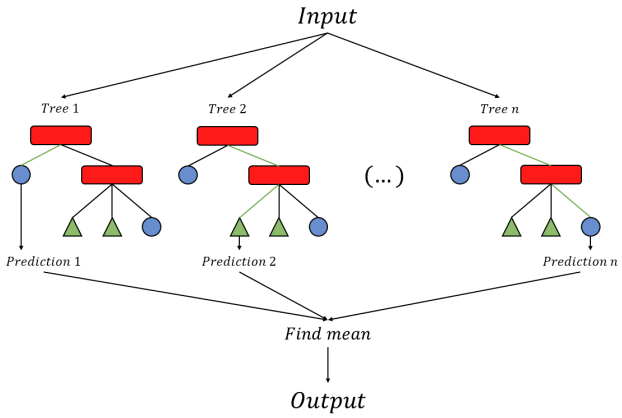
Note that the sci-kit learn implementation differs from this and the output will be taken by "averaging their probabilistic prediction, instead of letting each classifier vote for a single class".[9] This was initially proposed in Breiman (1996).

There are several advantages to random forests. They correct for decision trees' tendencies to overfitting to their training set and will generally outperform large individual decision trees. They are, however, generally considered to be black-box models, as there is no clear path taken to produce an output. As such, they can be more difficult to fine tune accurately and understand in general.

3.4.7 Light Gradient Boosting Decision Trees. Gradient Boosting Decision Trees (GBDT) are a set of tree-based classifiers which, similarly to Random Forests, use the output of large number of Decision Trees to make their prediction (Ke et al. 2017). However, unlike Random Forests, in which the trees are trained independently and their bagged output is taken as the output of the network, the GBDT algorithm trains trees in sequence, with each subsequent tree minimizing the residual error of the tree before it. This makes GBDT models perform very well, but the training time and computation required is greater than the other algorithms evaluated, especially for large datasets. As is pointed out by Ke et al. (2017), "The main cost in GBDT lies in learning the decision trees, and the most time-consuming part in learning a decision tree is to find the best split points.". You can see a visualization of a GBDT in figure 8.

9 Sci-Kit Learn Random Forest Implementation

Figure 7: Visualization of the Random Forest Classifier.

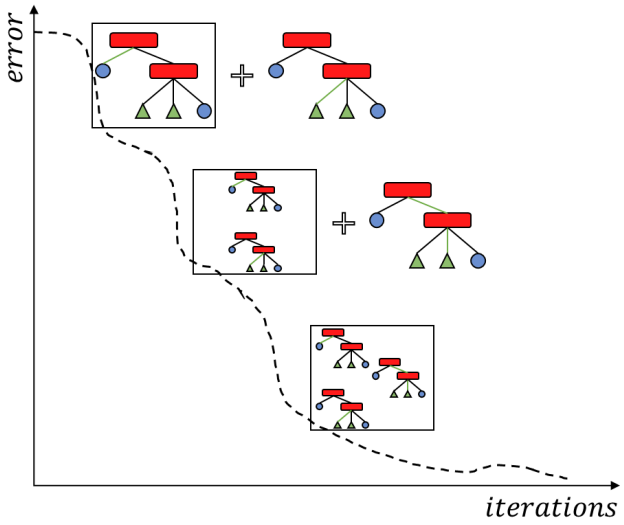


The Light Gradient Boosting Decision Trees (LightGBM) algorithm has been found to perform as well or better than other GBDT variants (such as the one initially proposed by Friedman (2001)), while having a significantly lower training time. This is achieved thanks to two techniques:

- Gradient-based One-Side Sampling (GOSS)
  - GOSS reduces the amount of data the trees use for learning.
  - Each data instance in GBDT provides a gradient associated with the error for that instance. If the gradient is small, the error is small, and thus the model is well-trained for that instance.
  - GOSS selects the instances of data to be trained on based on their gradient. All instances with a large gradient are kept and a random sampling is performed on the instances with a small gradient.
  - When calculating the information gain, the data with small gradients is amplified. This allows GOSS to place a higher importance on under-trained data while also not affecting the data distribution.
- Exclusive Feature Bundling (EFB)
  - A method to effectively reduce the number of features by bundling exclusive categorical features.
  - Very often, features are sparse and mutually exclusive. This is especially true for one-hot encoded categorical data.
  - EFB takes these *exclusive features* and *bundles* them, reducing the total number of features while not affecting the informativeness of the data.

GBDT-based models have found extensive use in many different areas of both research and industry. They generally have comparative or higher performance than

Figure 8: Visualization of the Gradient Boosting Decision Trees Classifier



Random Forests, at the cost of greater computation and memory requirements (Bentéjac, Csörgő, and Martínez-Muñoz 2021). LightGBM models specifically have been successfully used in market forecasting (Sun, Liu, and Sima 2020), phishing webpage detection (Li et al. 2019), EEG-based driver state classification (Zeng et al. 2019), and many more.

3.4.8 Multi-Layered Perceptron. The Multilayer Perceptron (MLP) algorithm is a subclass of the artificial neural network (ANN) family of machine learning models. There are, at a minimum, 3 layers of neurons in an MLP – an input layer, a hidden layer, and an output layer. All neurons are interconnected between layers (Rosenblatt 1958; Hastie, Tibshirani, and Friedman 2009). You can see this visualized in figure 9.

Each neuron in the hidden layer will perform a weighted linear summation ($w_1\ x_1 + w_2\ x_2 + ... + w_m\ x_m$) to the values from the previous layer, followed by a non-linear activation function. There is a number of activation functions which can be used, but the logistic (2a), hyperbolic tangent (2b), and rectified linear unit (2c) functions were evaluated for this research.

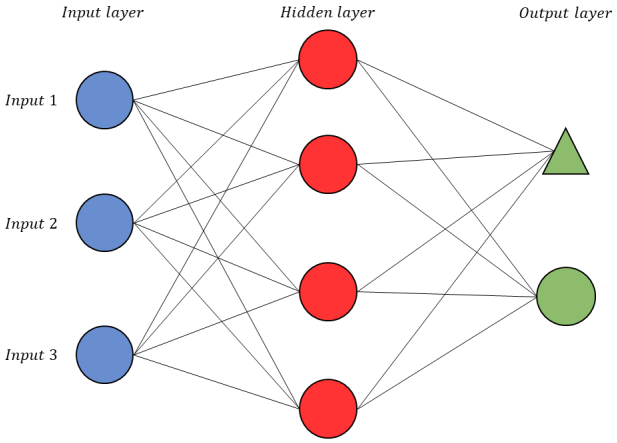
$$logistic\,(x) = \frac{1}{1 + e^{-x}} \tag{2a}$$

$$tahn\,(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{2b}$$

$$ReLU\,(x) = \max(0, x) \tag{2c}$$

During training, backpropagation is used. The goal of backpropagation is straightforward - change the weights of the parameters of the network such that the overall

Figure 9: Visualization of the Multi-Layered Perceptron Classifier
A simple 1-hidden-layer network with three inputs and 2 outputs.



error is reduced. The error is usually measured by the Mean Squared Error (3). Backpropagation is performed recursively for each sample the network is trained on.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \tag{3}$$

MLP have been used in many areas of both research and in industry. They have found success in a great number of tasks in a broad variety of scientific and industrial fields - from text classification (Lee and Choeh 2014), to forest biometrics and modelling (Chiarello et al. 2019), and many more. They can handle non-linearly separable data, which makes them ideal for use in tasks with high dimensionality and a large number of features. A disadvantage compared to the other models tested is that it requires a greater amount of compute and takes significantly longer to train, as described in the results section.

4. Results

This section will provide a detailed overview of the classification performance for the models described in Methods section, the best hyperparameters which were found for the models, as well as go over the findings regarding the feature importance and feature correlation metrics. All models were trained on the preprocessed training set and the results reported are from the test set, unless otherwise specified.

4.1 Models

4.1.1 Hyperparameters. The hyperparameters tested for all models are shown in table 1. Generally, a range large enough to allow for the ideal value to be within it, while also be-

ing viable from a practical point of view, was chosen for all parameters. Via grid search, all combinations were trained and evaluated. You can see the best hyperparameters for each of the models, across datasets, in section 1.1 of the appendix.

One thing to take note of is the fact that the k value of the KNN classifier reached its high-limit for the **M** dataset, with the value stopping one step before, at 97, when trained on the **SPT** and **T** datasets. This is in contrast to the value of k for the **F** dataset, which was only 37. It seems that the KNN classifier's performance in terms of accuracy stagnated, and it was not able to fit a better line, regardless of the value of k. This, in combination with the comparatively low performance of the model, is why it was not deemed necessary to increase the range of k.

4.1.2 Performance. All models were trained using Grid Search with F-scoring and 10-fold cross-validation on each of the datasets. The hyper-parameters, shown in table 1, were tuned for each dataset. You can see the hyper-parameters, F1-Score, and training time of all of the models on each of the datasets in section 1.1 of the appendix. The training time reported is the time it took to train the model with those specific hyper-parameters and does not include the time taken to perform Grid Search over the total number of hyperparameters. As mentioned, all training and evaluation was done on an Intel i7-4790 CPU.

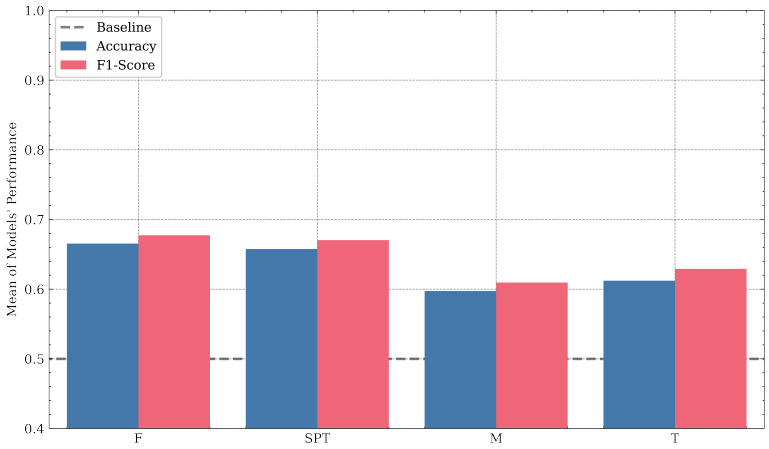
The Dummy Classifier model will be used as a baseline, against which all other models will be compared. On each of the datasets, the baseline result was an accuracy of approximately 50%, a precision and recall of approximately 50%, and an F-1 score of approximately 48%. These results make sense in a binary classification task, since the Dummy Classifier simply creates a random choice based on the distribution of the input data.

All models, across all datasets, exhibited a score higher than the baseline, with the highest increase being found in the LightGBM model trained on the dataset with all of the features (**F**). The optimal hyperparameters for this model were found to be a 'Max Depth' of 16 and 1000 Estimators. It's results showed a 21.5% F1-score increase over the baseline. The model has an accuracy of 69.50%, a precision and recall of 71.38% and 71.41% respectively, and an F1-score of 71.39%.

For all datasets, the ensemble tree models had the highest accuracy. LightGBM models had a higher accuracy than Random Forests and a training time at least an order of magnitude lower. The naïve bayes and DT classifiers were generally trailing behind. This can be explained by their lower complexity and inability to fit to the highly dimensional problem presented by fire size classification.

Overall, all of the models performed best on the **F** dataset, however, the models trained on the **SPT** dataset were close in accuracy and F1-score. The LightGBM model trained on the **SPT** dataset had an F1-Score only 1% lower than that of the **F**-trained LightGBM model, while being trained on a dataset with less than half the features. It had an accuracy of 69.18%, a precision of 70.54%, recall of 70.86%, and an F1-score of 70.69%. Thanks to the lower number of features, the training time was lowered by 19%,

Figure 10: Mean Accuracy and F1-score of all Models.
The baseline Dummy Classifier model was not included. Average of all models performs better than baseline, with **F** and **SPT** performing similarly, ahead of the **M** and **T** datasets.



although it is not a large difference in real-time - training was reduced from 0.941s to 0.754s.

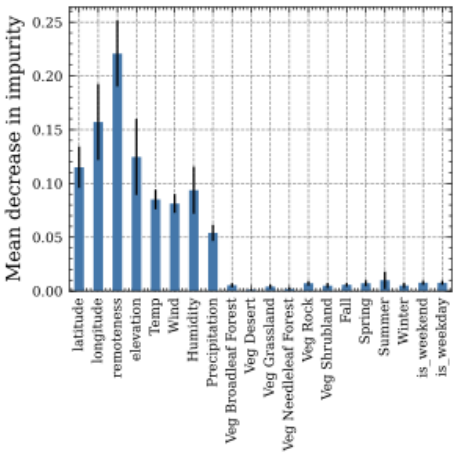
4.2 Feature Importance

In order to further explore the overall performance of the datasets, a mean of the models' accuracy and F1-score was taken (without the baseline). This allows for a comparison to be made between the datasets independently overall, rather than focusing on singular classifiers and their hyperparameters. You can see a plot of this in figure 10. The **F** dataset has the best overall performance with an average accuracy of 66.54% and average F1-score of 67.73%. The **SPT** dataset has an accuracy and F1-score within a percent of the **F** dataset. This is in contrast to the **M** and **T** datasets which are trailing behind in accuracy by 6.81% and 5.33% respectively. While the differences are small, one should consider the fact that these figures are the mean across all classifiers and are used to show a trend in the classification performance. There are larger differences between datasets when looking at individual models. For more information, please refer to section 1.1 of the appendix.

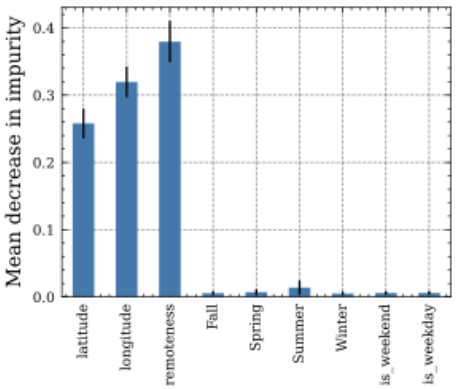
Another way used to evaluate the feature importance is through the feature_importance_ attribute, part of the scikit-learn Random Forest implementation, as described in the Methods section. This method allows for a more detailed look, highlighting the features that have the highest decrease in impurity, which can be interpreted as importance. The feature importances were computed for the RF models trained on each of the datasets and were plotted in figure 11. The plots for all of the features relative to each other can be seen in figure 11a, with more detailed information on the individual sets of features better visible in figures 11b, 11c, and 11d.

Figure 11: Feature importance plots.

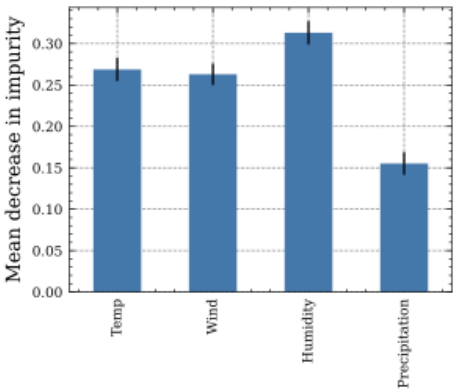
(a) Full Dataset (**F**)



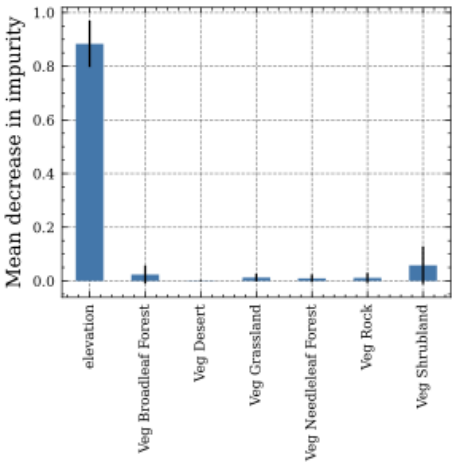
(b) Spatiotemporal Dataset (**SPT**)



(c) Meteorological Dataset (**M**)



(d) Topological Dataset (**T**)



Overall, the most important feature was the remoteness of the fire, however the location of the fire - the latitude and longitude, follows closely. In fact, if taken latitude and longitude are to be considered together, the location of the fire is the most important feature. The elevation and meteorological features are also informative, but less that the ones previously mentioned. The rest of the features - the seasons, whether it was the weekend, and the vegetation information, are essentially irrelevant in how little information is gained from them.

This large difference in information gain is highlighted when looking at the plots for the **SPT** and **T** datasets - figures 11b and 11d. It should be kept in mind that the values with lower importance are categorical, and as a result their informativeness is

spread out over each label, however even when taken together, the temporal features of the spatio*temporal* dataset (seasons and whether it was the weekend at the time of the fire) account for <10% of the total informativeness. The same is repeated with the vegetation data of the topological dataset. When combined, the total mean decrease in impurity of all vegetation classes is still only around 15%.

Within the meteorological features, shown in figure 11c, humidity was found to have the highest informativeness, and thus be the most important. Surprisingly, the precipitation was found to have the lowest mean decrease in impurity at 15%.

5. Discussion

The overarching goal of this research was to implement and evaluate the performance of a variety of machine learning algorithms at the task of wildfire size classification. The main research question was: "To what extent can machine learning methods classify a wildfire's size?". A subquestion relevant to the impact of the features the models use was formed: "What impact do different data features have on the results of the models?".

The primary research question was evaluated by training 7 ML models and one baseline model, and evaluating their performance in terms of accuracy, precision, recall, F1-score, and training time. The findings, shared in the results section, show that all models exhibit a consistent increase in performance above the baseline. The highest increases were exhibited by the ensemble tree models (RF and LightGBM).

The results were significant, but far from perfect. It is difficult to determine the reason for the disparity in performance between this research and prior findings (). While the research questions do not overlap directly, they are similar in nature and one could have expected better overall accuracy and F1-score from the research shown here. A point to investigate for future work is the creation of a dataset comprised only of the n most important features. A different direction will be to look at larger and more complex models and deep-learning approaches. Deep learning has consistently been shown to deal well with highly-dimensional non-trivial tasks. Deep learning has been explored in use for wildfire detection (Zhao et al. 2018) and wildfire spread prediction (Radke, Hessler, and Ellsworth 2019).

A likely reason ensemble tree models generally performed best is the feature correlation exhibited in the features. This is visible in the correlation matrix plotted in figure 2. Specifically, the pairs of latitude/longitude, remoteness/longitude, elevation/humidity, and latitude/shrubland show a (negative) correlation. Ensemble tree models have been shown to deal well with highly-dimensional data with some correlation between the features (Dormann et al. 2013). The same reasoning may also apply to the MLP models, which consistently performed best with multiple hidden layers of neurons across all datasets (see section1.1 of the appendix).

Although it should generally be reduced as much as possible, in this case the correlation is difficult to tackle. Since the features related to the location of the fire are some of the most informative (see figure 11), but also most highly correlated, one would

assume that they cannot simply be removed without a due drop in the classification accuracy.

This brings me onto the subquestion regarding the effects of the different features on the models' performance. The feature importance plots, shown in figure 11, give a great overview of the informativeness (and lack-there-of) of the features. We can tell that there is very little mean decrease in impurity given by the features related to vegetation. This is useful information, as specific information about vegetation for an area can be difficult to find, especially in rural and/or unpopulated areas. Thus, it may be practical for future academic research and real-world end-to-end systems to consider using models which do not require, or do not consider, vegetation information - specifically in cases when such information is not readily available.

6. Conclusion

This research project investigated the use of machine learning classifiers in making predictions about a wildfire's size, as well as look into which features are most important in order to make a good prediction. Seven model architectures were trained, tested, and compared against a 50% baseline. These architectures were chosen based on the investigations and results of prior research (Calp and Kose 2020; Cortez and Morais 2007; Sayad, Mousannif, and Moatassime 2019).

The most effective model was found to be the LightGBM classifier, closely followed by the Random Forest classifier - the other ensemble tree-based architecture. A maximum increase to accuracy and F1-score of 19.5% and 20.5% respectively was observed over baseline with the LightGBM model trained on the **F** dataset.

The most important features were related to the location of the fire, followed by the meteorological information. Vegetation and temporal features did not have a significant decrease to the impurity and thus can be considered inconsequential for the purposes of this research.

7. Self Reflection

This project was one of the largest undertakings I have taken so far in my life - certainly academically, most likely as a whole. If I could do it again I would do many things differently.

First and foremost, I would have managed my time and tasks better. I would have liked to work on the project consistently, over time, rather than in bursts of energy followed by lulls of little-to-no work being done. I would have taken greater care to see exactly what is wanted from me and created a more detailed plan of the steps required to get there. This would have allowed me to better know what is in front of me rather than being surprised by the magnitude of work required to do something.

Secondly, I would have been much more communicative and collaborative with my thesis advisor. They have been incredibly kind, helpful, and patient with me at every step of the way, but I was often not ready to ask them for guidance, but rather chose to try and solve the problems on my own, wasting invaluable time and effort in the

process. This was no doubt exacerbated, if not enabled entirely, by the pandemic and the fact I was making this thesis while 3000km away from my advisor, but that is certainly no excuse for what I allowed to happen.

References

Bentéjac, Candice, Anna Csörgő, and Gonzalo Martínez-Muñoz. 2021. A comparative analysis of gradient boosting algorithms. *Artificial Intelligence Review*, 54(3):1937–1967.

Breiman, By Leo and Leo B eiman. 1994. Bagging predictors.

Breiman, L., J. Friedman, R. Olshen, and C. J. Stone. 1983. Classification and regression trees.

Breiman, Leo. 1996. Bagging predictors. *Machine Learning*, 24.

Calp, M. Hanefi and Utku Kose. 2020. Estimation of burned areas in forest fires using artificial neural networks. *Ingeniería Solidaria*, 16:1–22.

Cary, Geoffrey J., Robert E. Keane, Robert H. Gardner, Sandra Lavorel, Mike D. Flannigan, Ian D. Davies, Chao Li, James M. Lenihan, T. Scott Rupp, and Florent Mouillot. 2006. Comparison of the sensitivity of landscape-fire-succession models to variation in terrain, fuel pattern, climate and weather. *Landscape Ecology*, 21:121–137.

Chiarello, Flávio, Maria Teresinha Arns Steiner, Edilson Batista de Oliveira, Júlio Eduardo Arce, and Júlio César Ferreira. 2019. Artificial neural networks applied in forest biometrics and modeling: State of the art (january/2007 to july/2018). *Cerne*, 25.

Cortes, Corinna and Vladimir Vapnik. 1995. Support-vector networks. *Machine learning*, 20(3):273–297.

Cortez, Paulo and Aníbal Morais. 2007. A data mining approach to predict forest fires using meteorological data.

Cover, Thomas and Peter Hart. 1967. Nearest neighbor pattern classification. *IEEE transactions on information theory*, 13(1):21–27.

Dormann, Carsten F, Jane Elith, Sven Bacher, Carsten Buchmann, Gudrun Carl, Gabriel Carré, Jaime R García Marquéz, Bernd Gruber, Bruno Lafourcade, Pedro J Leitao, et al. 2013. Collinearity: a review of methods to deal with it and a simulation study evaluating their performance. *Ecography*, 36(1):27–46.

Fang, Lei, Jian Yang, Jiaxing Zu, Guicai Li, and Jiashen Zhang. 2015. Quantifying influences and relative importance of fire weather, topography, and vegetation on fire size and fire severity in a chinese boreal forest landscape. *Forest Ecology and Management*, 356:2–12.

Friedman, Jerome H. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232.

Goldberger, Jacob, Sam Roweis, Geoff Hinton, and Ruslan Salakhutdinov. 2005. Neighbourhood components analysis.

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman. 2009. *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media.

Ho, Tin Kam. 1995. Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*, volume 1, pages 278–282, IEEE.

Hornik, Kurt. 1991. Approximation capabilities of multilayer feedforward networks. *Neural networks*, 4(2):251–257.

Hughes, G. 1968. On the mean accuracy of statistical pattern recognizers. *IEEE Transactions on Information Theory*, 14(1):55–63.

Jain, Piyush, Sean CP Coogan, Sriram Ganapathi Subramanian, Mark Crowley, Steve Taylor, and Mike D Flannigan. 2020. A review of machine learning applications in wildfire science and management. *Environmental Reviews*, 28(4):478–505.

Jones, Matthew W, Adam Smith, Richard Betts, Josep G Canadell, I Colin Prentice, and Corinne Le Quéré. 2020. Climate change increases the risk of wildfires rapid. *ScienceBrief Review*, 116.

Ke, Guolin, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie Yan Liu. 2017. Lightgbm: A highly efficient gradient boosting decision tree. volume 2017-December.

Lee, Sangjae and Joon Yeon Choeh. 2014. Predicting the helpfulness of online reviews using multilayer perceptron neural networks. *Expert Systems with Applications*, 41.

Lelieveld, J., J. S. Evans, M. Fnais, D. Giannadaki, and A. Pozzer. 2015. The contribution of outdoor air pollution sources to premature mortality on a global scale. *Nature*, 525:367–371.

Li, Yukun, Zhenguo Yang, Xu Chen, Huaping Yuan, and Wenyin Liu. 2019. A stacking model using url and html features for phishing webpage detection. *Future Generation Computer Systems*, 94.

Malarz, K, S Kaczanowska, and K Kułakowski. 2002. Are forest fires predictable? *International Journal of Modern Physics C*, 13(08):1017–1031.

Mccallum, Andrew and Kamal Nigam. 1998. A comparison of event models for naive bayes text classification.

Meiyappan, Prasanth and Atul K. Jain. 2012. Three distinct global estimates of historical land-cover change and land-use conversions for over 200 years. *Frontiers of Earth Science*, 6:122–139.

Metsis, Vangelis and Georgios Paliouras. 2006. Spam filtering with naive bayes-which naive bayes? *.

Morgan, James N and John A Sonquist. 1963. Problems in the analysis of survey data, and a proposal. *Journal of the American statistical association*, 58(302):415–434.

Parisien, Marc-André and Max A Moritz. 2009. Environmental controls on the distribution of wildfire at multiple spatial scales.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Radke, David, Anna Hessler, and Dan Ellsworth. 2019. Firecast: Leveraging deep learning to predict wildfire spread. In *IJCAI*, pages 4575–4581.

Ramesh, Varun. 2020. U.s. wildfire data (plus other attributes) https://www.kaggle.com/capcloudcoder/us-wildfire-data-plus-other-attributes.

Rosenblatt, Frank. 1958. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386.

Rothermel, R C. 1970. A mathematical model for fire spread predictions in wildland fuels.

Sayad, Younes Oulad, Hajar Mousannif, and Hassan Al Moatassime. 2019. Predictive modeling of wildfires: A new dataset and machine learning approach. *Fire Safety Journal*, 104:130–146.

Short, Karen C. 2017. Spatial wildfire occurrence data for the united states, 1992-2015 [fpa_fod_20170508].

Slocum, Matthew G., Brian Beckage, William J. Platt, Steve L. Orzell, and Wayne Taylor. 2010. Effect of climate on wildfire size: A cross-scale analysis. *Ecosystems*, 13:828–840.

Sun, Xiaolei, Mingxi Liu, and Zeqian Sima. 2020. A novel cryptocurrency price trend forecasting model based on lightgbm. *Finance Research Letters*, 32:101084.

Tonini, Marj, Mirko D'Andrea, Guido Biondi, Silvia Degli Esposti, Andrea Trucchia, and Paolo Fiorucci. 2020. A machine learning-based approach for wildfire susceptibility mapping. the case study of the liguria region in italy. *Geosciences*, 10(3):105.

Trunk, G. V. 1979. A problem of dimensionality: A simple example. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-1(3):306–307.

Wang, Sally S.C. and Yuxuan Wang. 2020. Quantifying the effects of environmental factors on wildfire burned area in the south central us using integrated machine learning techniques. *Atmospheric Chemistry and Physics*, 20:11065–11087.

Zeng, Hong, Chen Yang, Hua Zhang, Zhenhua Wu, Jiaming Zhang, Guojun Dai, Fabio Babiloni, and Wanzeng Kong. 2019. A lightgbm-based eeg analysis method for driver mental states classification. *Computational intelligence and neuroscience*, 2019.

Zhao, Yi, Jiale Ma, Xiaohui Li, and Jie Zhang. 2018. Saliency detection and deep learning-based wildfire identification in uav imagery. *Sensors*, 18(3):712.

1. Appendix

1.1 All Models' Performance on all Datasets

Dataset	Classifier	Best Parameters	Accuracy	F1 Score	Training (s)
Full	Dummy Classifier	-	0.50357	0.48942	0.001
	K-Nearest Neighbours	k: 37 Weights: Distance	0.66952	0.67534	0.007
	Gaussian Naïve Bayes	-	0.60075	0.62085	0.011
	Support Vector Machine	C: 2 Kernel: Poly	0.65825	0.67506	13.286
	Decision Tree	Criterion: Entropy Max Depth: 16	0.66053	0.66400	0.285
	Random Forest	Criterion: Entropy Max Depth: 16 Estimators: 1000	*0.69496*	0.71058	41.976
	LightGBM	Max Depth: 16 Estimators: 1000	**0.69840**	**0.71392**	0.941
	MLP	Activation: ReLU Alpha: 0.05 Hidden Layer Sizes: (250, 250)	0.67533	0.68448	25.893
Spatiotemporal	Dummy Classifier	-	0.50108	0.48314	0.001
	K-Nearest Neighbours	k: 97 Weights: Distance	0.67542	0.68756	0.230
	Gaussian Naïve Bayes	-	0.58530	0.59715	0.007
	Support Vector Machine	C: 2 Kernel: Poly	0.62653	0.63549	13.122
	Decision Tree	Criterion: Entropy Max Depth: 16	0.67111	0.68365	0.132
	Random Forest	Criterion: Entropy Max Depth: 16 Estimators: 500	0.68200	0.70207	15.813
	LightGBM	Max Depth: 16 Estimators: 500	**0.69182**	0.70688	0.754
	MLP	Activation: ReLU Alpha: 0.0001 Hidden Layer Sizes: (250, 250)	0.67116	0.67659	37.572
Meteorological	Dummy Classifier	-	0.49372	0.47757	0.001
	K-Nearest Neighbours	k: 99 Weights: Distance	0.59825	0.61715	0.029
	Gaussian Naïve Bayes	-	0.57553	0.58878	0.005
	Support Vector Machine	C: 2 Kernel: RBF	0.59606	0.60387	16.485
	Decision Tree	Criterion: Entropy Max Depth: 4	0.59675	0.61008	0.029
	Random Forest	Criterion: Entropy Max Depth: 16 Estimators: 500	0.60574	0.61680	24.799
	LightGBM	Max Depth: None Estimators: 2500	0.60665	0.62637	0.255
	MLP	Activation: ReLU Alpha: 0.0001 Hidden Layer Sizes: (50, 50)	0.60200	0.62019	14.945
Topological	Dummy Classifier	-	0.49923	0.48660	0.001
	K-Nearest Neighbours	k: 97 Weights: Uniform	0.60221	0.62257	0.047
	Gaussian Naïve Bayes	-	0.57295	0.59238	0.005
	Support Vector Machine	C: 0.25 Kernel: Poly	0.60320	0.62395	17.767
	Decision Tree	Criterion: Entropy Max Depth: 4	0.61857	0.64562	0.016
	Random Forest	Criterion: Giny Max Depth: 4 Estimators: 2500	0.62365	0.65094	9.838
	LightGBM	Max Depth: None Estimators: 1000	0.63273	0.65013	0.305
	MLP	Activation: Logistic Alpha: 0.0001 Hidden Layer Sizes: (25, 25)	0.63135	0.64802	4.056

1.2 Fire Size Classes and Sizes

Table 2: US wildfire size classification

Class	Min	Max
Class A	-	¼ acre
Class B	¼ acre	10 acres
Class C	10 acres	100 acres
Class D	100 acres	300 acres
Class E	300 acres	1 000 acres
Class F	1 000 acres	5 000 acres
Class G	5 000 acres	-

1.3 Vegetation Groups

Table 3: The original labels and how the new classes into which they were grouped

# label	Original Class Label	New Class
1	Tropical Evergreen Broadleaf Forest	Broadleaf Forest
2	Tropical Deciduous Broadleaf Forest	Broadleaf Forest
3	Temperate Evergreen Broadleaf Forest	Broadleaf Forest
4	Temperate Evergreen Needleleaf Forest	Needleleaf Forest
5	Temperate Deciduous Broadleaf Forest	Broadleaf Forest
6	Boreal Evergreen Needleleaf Forest	Needleleaf Forest
7	Boreal Deciduous Needleleaf Forest	Needleleaf Forest
8	Savanna	Savanna
9	C3 Grassland/Steppe	Grassland
10	C4 Grassland/Steppe	Grassland
11	Dense Shrubland	Shrubland
12	Open Shrubland	Shrubland
13	Tundra Tundra	Tundra
14	Desert	Desert
15	Polar Desert/Rock/Ice	Rock
16	Secondary Tropical Evergreen Broadleaf Forest	Broadleaf Forest
17	Secondary Tropical Deciduous Broadleaf Forest	Broadleaf Forest
18	Secondary Temperate Evergreen Broadleaf Forest	Broadleaf Forest
19	Secondary Temperate Evergreen Needleleaf Forest	Needleleaf Forest
20	Secondary Temperate Deciduous Broadleaf Forest	Broadleaf Forest
21	Secondary Boreal Evergreen Needleleaf Forest	Needleleaf Forest
22	Secondary Boreal Deciduous Needleleaf Forest	Needleleaf Forest
23	Water/Rivers Water	Water
24	C3 Cropland	Cropland
25	C4 Cropland	Cropland
26	C3 Pastureland	Pastureland
27	C4 Pastureland	Pastureland
28	Urban land	Urban

1.4 Season Categorizations

Table 4: Month-to-season transformations

Month	Season
March	Spring
April	Spring
May	Spring
June	Summer
July	Summer
August	Summer
September	Autumn
October	Autumn
November	Autumn
December	Winter
January	Winter
February	Winter

1.5 Features and their Descriptions

Table 5: Features and their Descriptions

Features	Description (units)
Latitude	Latitude of origin point of the fire.
Longitude	Longitude of origin point of the fire.
Season	Season when the fire was discovered.
Weekend	Whether it was the weekend when the fire started.
Temperature	Average temperature at the location of the fire over last 30 days. (c)
Wind speed	Average wind speed at the location of the fire over last 30 days. (m/s)
Humidity	Average humidity at the location of the fire over last 30 days. (%)
Precipitation	Average precipitation at the location of fire over last 30 days. (mm)
Vegetation type	Dominant vegetation at the origin point of the fire.
Remoteness	Calculated by evaluating the distance to the closest city from the site of the fire.
Elevation	Elevation above sea level at the place the origin of the fire (meters)