Universiteit Leiden

Master Computer Science

Wildfire Anomaly Detection in Time Series Data Using an Unsupervised Learning Approach

Name:	Athul Sreejith
Student ID:	s3906868
Date:	March 2025
Specialisation:	Data Science: Computer Science
1st supervisor:	Prof. Hao Wang
2nd supervisor:	Prof. Zhaochun Ren

Master's Thesis in Computer Science

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Abstract

Wildfires have become one of most frequent and destructive environmental problem caused by abnormal weather conditions and predicting such events remains a major challenge, particularly in regions with very limited labeled wildfire data. This study explores the use of unsupervised learning techniques mainly LSTM autoencoders, Fourier regression model, and ARIMA models to detect anomalous patterns in weather data that may lead to wildfire occurrences. Instead of relying on past fire labels, these models are trained fully on normal weather behavior by training non wildfire data, allowing the models to flag unusual deviations across multiple climate features. The performance of each approach is evaluated using historical wildfire dates, revealing the importance of seasonal information and strengths and weakness of each models. The findings in the study highlights the potential of unsupervised methods in finding the anomalies in weather data that can lead to wildfire

Keywords: Wildfire Detection, Unsupervised Learning, Anomaly Detection, LSTM Autoencoder, Fourier Regression, ARIMA, Weather Forecast Data, Time Series Analysis, Climate Anomalies, Early Warning System, Latent Space Representation, Reconstruction Error, Seasonal Decomposition, Residual Modeling, Transfer learning

1.2 Introduction

In the recent years, wildfire events have become one of the most destructive and rapidly growing natural disaster in many countries with hot and dry climates. The recent wildfires in Northern California in 2024 caused over 50,000 acres of fire within a matter of days which had led to mass evacuations and confused local firefighting situations [3]. The extreme heatwaves and dry winds made the situation worser, this incident shows the strength in the need for an early wildfire detection and response because of the rapidly changing environmental conditions now a days. This increase in intensity and frequency can be attributed to a variety of causes, including climate change, drought problems and other environmental causes. Most of the time the exact triggers for these wildfires are not identified properly. These wildfires not only harm the human lives and human made infrastructures but also have a very bad effect on biodiversity, habitats and life in general. This has become a big issue, highlighting the need to address these climate related challenges. Since these environmental issues are growing, the need for smarter and more data oriented methods to support early detection also increases. As we look traditional wildfire detection systems mostly rely on heavy satellite imagery and field reports. While these techniques have been valuable but they are often reactive and mostly reporting fire events after they have already begun spreading [10]. This delay can cause evacuation time or prevention measures and by that time we have lost lot of life and habitat. Moreover, traditional systems struggle to offer early warnings especially in fast changing weather conditions. The weather forecast data is rich and continuous set of features like temperature, humidity, wind, solar radiation, and soil moisture that are often very correlated with fire behavior[1]. So these time series signals can be used to detect unusual environmental patterns before a fire breaks out and can stop from a big destruction to life and habitat. Many recent studies have explored the use of machine learning in wildfire detection. However, most of these approaches are supervised learning, which requires labeled historical wildfire data showing either the wildfire happened or not. While these supervised methods such as random forests, support vector machines, and other deep learning classifiers have been shown to be effective but they come with certain limitations like inability in transfer learning from one region to another, incomplete wildfire labels, delayed data labeling which takes lot of time for new wildfire labeled data to publish. This often makes the model struggle while training because we can't add labels when its missing [8]. For example, A model trained on labeled wildfire data from New South Wales in Australia may perform poorly when applied to data from Western Australia, due to environmental and behavioral differences in the regions, here we can aim to train a model to check whether the unsupervised model can learn a general threshold or general residual for transfer learning at least in that same country. Moreover, many wildfires occur in unpredictable ways, caused by abnormal environmental conditions that the traditional classification models may not have learned while training which makes it harder to classify wildfires. These shows need for more flexible approaches ones that can detect abnormal weather behavior without a need to know in advance[7]. Some research has begun to investigate unsupervised learning methods for wildfire detection. One such Unsupervised models developed to dynamically assess wildfire risk by learning contextual weather patterns over time was the (Context-Based Fire Risk) CBFR model to detect abnormal environmental conditions without requiring labeled fire data [9] Still, there is a lack of comparative studies that evaluate multiple unsupervised approaches with different strength and different behaviors and test their practical applicability in transfer learning without the help of labeled wildfire data and finding the limitations of univariate and multivariate models to see how well they can be used for wildfire detections or as an early detection system.

This thesis proposes an unsupervised approach where we explore the capability of unsupervised learning [13] methods to detect anomalous weather conditions that can lead to wildfires without relying on labeled wildfire data during training. In supervised models, both input data and labels are required during training. As a result a model trained in one region may not perform well elsewhere unless it is again trained with the labels of that specific region. So here we use unsupervised models which learn from normal weather data. If a model is trained to understand that normal weather patterns, it can potentially identify anomalies when weather variables with larger reconstruction errors shows and it can apply to any region with similar weather inputs [11]. This design choice is intentional because our goal is to test how far one can go using unsupervised detection methods alone, especially in situations where labels are unavailable or inconsistent. By learning the structure of normal weather data, we aim to detect deviations from normal weather that might correspond to dangerous or fire risk conditions without explicitly showing the model.

1.3 Research Question

The central research question guiding this thesis is:

> *Can unsupervised learning methods applied to weather related time-series data effectively detect anomalous weather conditions like wildfires ?*

To explore this, the following sub-questions are considered:

- Can autoencoders trained on multivariate time series learn normal weather patterns and identify sequences that deviate from them?
- How effective are uni-variate models in capturing anomalies across individual weather features?
- How do these unsupervised methods compare in their ability to flag wildfire-relevant anomalies across time and geographic regions?

1.4 Method Overview

To address these questions, this study explores and compares three unsupervised anomaly detection methods:

- **LSTM Autoencoder**: A neural network designed to reconstruct sequences of weather forecast features. High reconstruction errors are treated as anomalies since the model can't reconstruct which may indicate wildfire risk conditions[4].
- **ARIMA-based Anomaly Detection**: Each weather feature is modeled using classical time-series analysis. If forecasted values deviate beyond a pre-defined threshold, these are flagged as anomalies.
- **Fourier Series Regression**: A seasonal regression framework using Fourier series terms to model expected seasonal variation in features. High residuals between predicted and actual values indicate unusual weather behavior.

All three models are tested using real-world Australian weather datasets. Known wildfire dates are used for evaluation only not during model training to determine how well each method aligns with real fire events. This allows us to assess the performance of unsupervised anomaly detectors in capturing wildfire related weather deviations.

Chapter 2

Data

2.1 Data Description

For this research we are using using three datasets: a historical record of wildfire activities happend across seven regions of Australia namely New South Wales (NSW), Northern, Territory (NT), Queensland (QL), South Australia (SA), Tasmania (TA), Victoria (VI) and Western Australia (WA) and their corresponding weather information and Vegetation information. All these dataset have information about all the seven regions of Australia. These dataset can help in our research to explore the unusual behavior of weather pattern causing these destructive wildfires since it has wildfire information and its corresponding weather information. This approach allows us to assess how well unsupervised models are since the models are trained entirely without labeled wildfire data

2.1.1 Historical Weather Dataset

TABLE 2.1: Details of Weather Features

Feature Category	**Feature Description**	**Range and Unit**
Date	Time span of daily weather records	2005-01-01 to 2021-01-23
Region	Australian states covered in the dataset	NSW, NT, QL, SA, VI, WA, TA
Precipitation	Min, max, mean, and variance of daily precipitation	0 – 509.83 mm/day
Relative Humidity	Min, max, mean, and variance of daily relative humidity	0 – 509.83 mm/day
Soil Water Content	Min, max, mean, and variance of soil moisture content	0 – 0.52 m^3/m^3
Solar Radiation	Min, max, mean, and variance of solar radiation	0.41 – 35.69 $MJ/m^2/day$
Temperature	Min, max, mean, and variance of daily air temperature	-5.05 – 41.73 °C
Wind Speed	Min, max, mean, and variance of daily wind speed	0.25 – 24.27 m/s

The Weather Dataset used in this study is a daily weather data which is derived from the ERA5 reanalysis product. ERA5 is developed by the European Center for Medium-Range Weather Forecasts (ECMWF) [5]. It is widely used in climate research and weather analysis due to its consistency and completeness, particularly in regions where direct measurements are often sparse. The weather information is aggregated to a daily frequency where each data point or row in the dataset represents the average weather conditions for one region in Australia among the seven regions over a single day, starting from 01:00 UTC of that given day to 00:00 UTC of the next day. The weather variables in this data include lot of key factors known to cause fire, such as temperature, precipitation, humidity, wind speed, solar radiation, and soil moisture content. Temperature values in the dataset have provided with minimum, mean, and maximum for each day. Precipitation is calculated from the total rainfall and then converted from meters per hour to millimeters per day. Relative humidity is calculated from temperature and dew point readings. Wind speed is calculated from easterly and northerly wind components at a height of 10 meters. Solar radiation is the total energy from the sun and the soil moisture is estimated for the top 0 to 7 centimeters of soil which is an important layer of the soil which can show the speed of fire ignition and its spread, all values of all features in weather data that we used in the study are mentioned in table2.1. All these variables in the dataset are continuous and are present throughout the dataset which makes the dataset ideal for our wildfire anomaly detection[17].

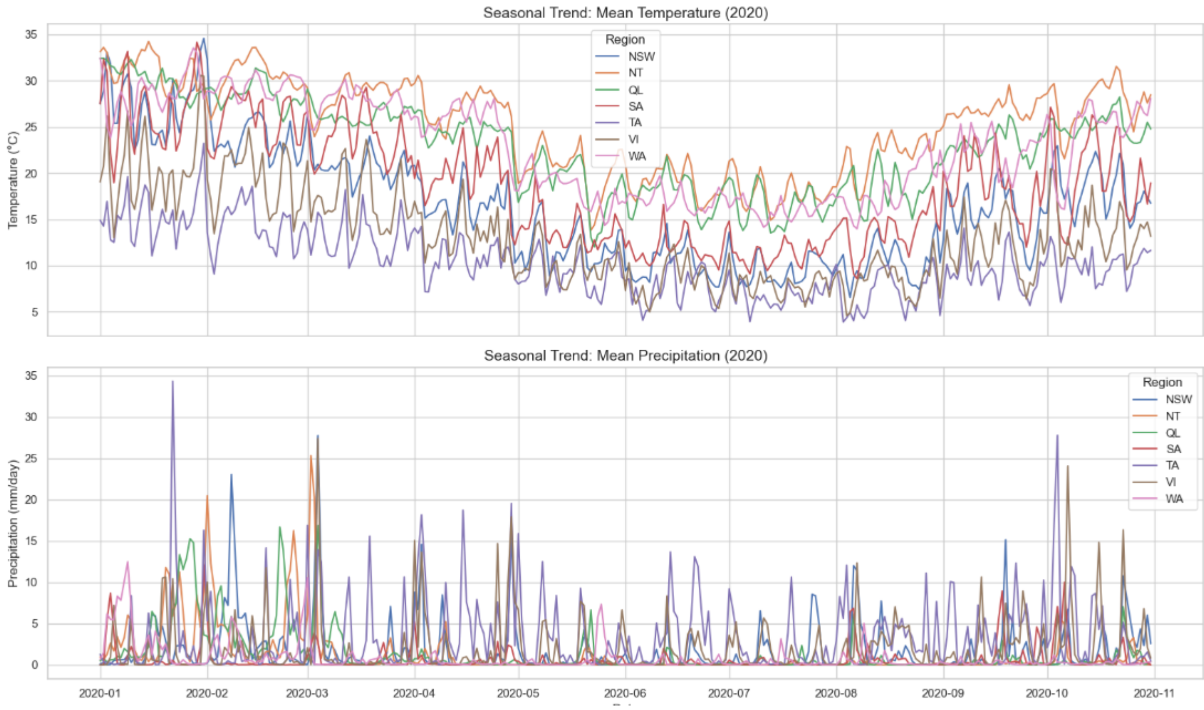

FIGURE 2.1: Seasonal trends in raw temperature and precipitation values for selected regions

2.1.2 Historical Wildfire Dataset

The wildfire data used in this study originates from IBM's PAIRS Geo scope platform [6]. This dataset covers wildfire activity from 2005 to 2021. While the dataset offers detailed information such as fire area, brightness, radiative power, and other variables [18] but we are not directly using any of these variables in our modeling process. We do not use any fire labels or any of the features of this dataset to train or tune our models. Instead, we only extract the dates and regions where wildfires were recorded, and then we use those dates and regions to check the corresponding dates and regions in the weather dataset to separate wildfire and non-wildfire weather data.

2.1.3 Normalized Difference Vegetation Index

TABLE 2.2: Details of Normalized Difference Vegetation Index Features

Feature Category	**Feature Description**	**Range and Unit**
Date	Time span of daily weather records	2005-01-01 to 2021-01-23
Region	Australian states covered in the dataset	NSW, NT, QL, SA, VI, WA, TA
Vegetation Index	Min, max, mean, and variance of NDVI	$0 \leq NDVI \leq 1$

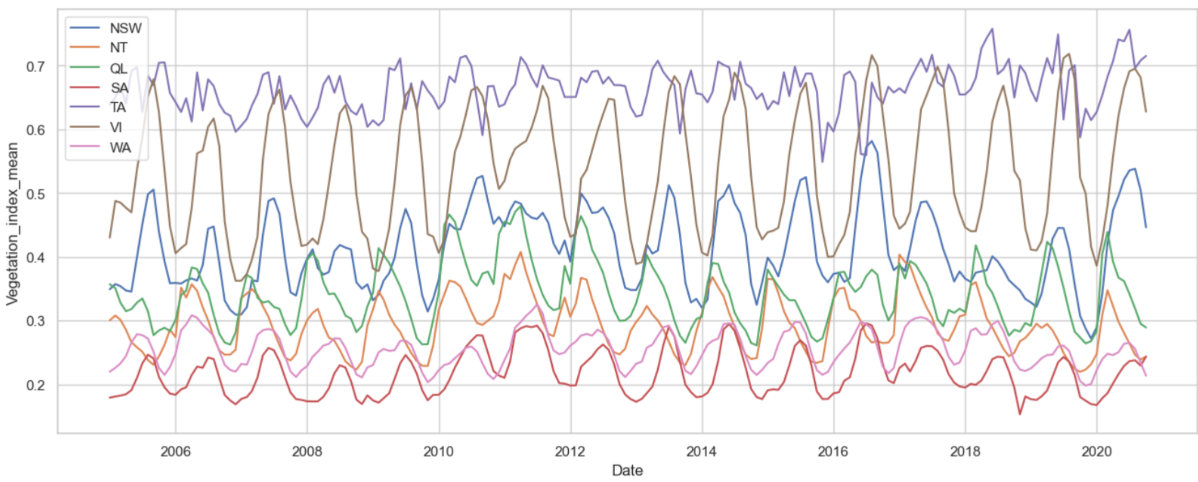

FIGURE 2.2: Distribution of NDVI values by all seven regions in Australia from 2005 to 2021

The Normalized Difference Vegetation Index (NDVI) is a satellite measure that indicates the greenness and the density of the overall vegetation of a specific area. The image below shows the NDVI average of Australia from 1 Dec 2012 to 31 May 2013[12].

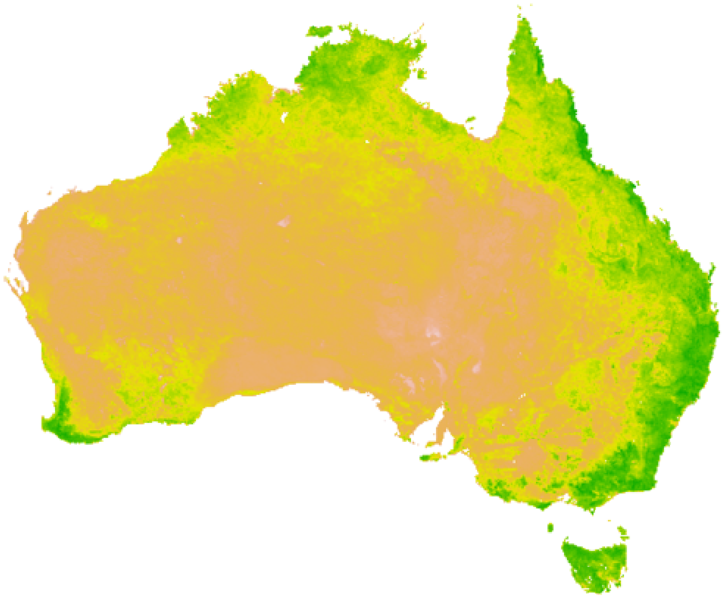

FIGURE 2.3: NDVI Average of all seven regions in Australia

It is calculated based on the reflection of near infrared light where healthy vegetation means it reflects more strongly, The values of this NDVI ranges from 0 to 1 with higher values showing dense vegetation 2.2. Vegetation can act as a strong variable for wildfires like areas with low NDVI are more probable to catch fire under high temperatures and high NDVI suggest vegetation is still moist which is less probable to catch or spread fire.

The NDVI data originally were monthly values in the dataset [16] but for this research we aggregated those values into daily values to merge with the corresponding weather data. This allows our training data more diverse where we are not just training with atmospheric and soil conditions of each day but also including the landscape details like vegetation. By Integrating this we aim to provide a more complete environment behavior to our unsupervised models. This additional information can enhance the ability for the model to find anomalies better. Larger the information means better the model can learn.

2.2 Data Preprocessing

For our research we want all the dataset to be aligned for merging and for separating wildfire and non wildfire weather data for training the model with proper normal weather data. The preprocessing for weather data included handling missing values and converting the data into proper clean structure for better usage because the original format contains multiple nested measurements across several weather parameters. These includes statistical aggregations such as minimum, mean, maximum and variance but it was in a long format structure that was not ideal for our data modeling. So the first objective of preprocessing was to restructure and standardize the dataset.

2.2.1 Wildfire and Weather dataset

The first step in this included renaming and simplifying column headers to proper labeling like mean() and variance() which were replaced with Mean and Variance, and the area-related column `count()[unit:km^2]` to Area. This renaming of dataset made the work more reliable. Then we restructured the weather data using a pivot operation. A pivot operation is a way of rearranging the dataset so that in our case, instead of having many row values for one measurement which was the original format, pivoting will make all the relevant values for a given day and region into a single row, making the dataset more structured and easier to analyze and work with. The pivoted structure is known to be more compatible with time-series modeling, which allows the models to learn across all the weather variables at the same time. Once the structuring was done, the next step was handling missing values, which are often common in weather datasets due to the satellite data gaps in transmission and sensor issues. Each features were treated individually since these features have different characteristics, so we can't use one method for all the features. This was a challenging part because we can't simply add values for weather data, which will then affect the pattern of the weather forecast data.

For precipitation variables, the missing values were filled using forward fill which means it carries the last known value from the previous row forward. This was done based on the assumption that the rainfall patterns will sustain over short periods and these short gaps can then be filled from the previous day without a big change in the feature pattern. For relative humidity and temperature variables, we have used the linear interpolation method to fill missing values. Since these are atmospheric conditions that generally change gradually from day to day, applying linear interpolation might be a good approach. Linear interpolation is a method that works by filling in missing value by considering a straight line between the nearest known values before and after the long gap, which means it connects these two points to form a smooth, proper transition. For example, if the temperature value on Monday was 28°C and on Wednesday it was 30°C, but the value of Tuesday was missing, then the linear interpolation method will fill Tuesday's temperature as 29°C showing a steady increase between the two points. For solar radiation and wind speed, we used a 3-day rolling average to smooth the data. This method works by replacing each value with the average of its value and the two days around it. For example, if the wind speed on Monday was 12 km/hr and Tuesday was 18 km/hr and on Wednesday 15 km/hr then it is calculated the average of these three days which is 15 km/hr. This will create more stable trend without removing the original time-based pattern and if any values were still missing, we used linear interpolation like we used before for Humidity and temperature. However, some missing values may appear at the front or end where interpolation is not possible. In such cases, we used median values of that region to fill the gaps. All these handling of missing values are done for individual regions. For example, missing values in the Temperature variable of Western Australia (WA) was interpolated only using values from Western Australia, avoiding cross regional inputs. After all the missing variables were properly handled, the next step was to align with the wildfire dataset. Since our approach was based on unsupervised modeling, we do not use wildfire data for training the model, but we do need wildfire occurrence dates for evaluating the model to see how extent unsupervised models can go. For this we took the date and region combination in the wildfire dataset with those in the weather dataset and then separated into yeswildfire_df, which contains all the wildfire records and nowildfire_df with all the non-wildfire records as you see in the figure 2.4. This separation ensures that our model is only learning from normal weather data without being exposed to wildfire weather data.

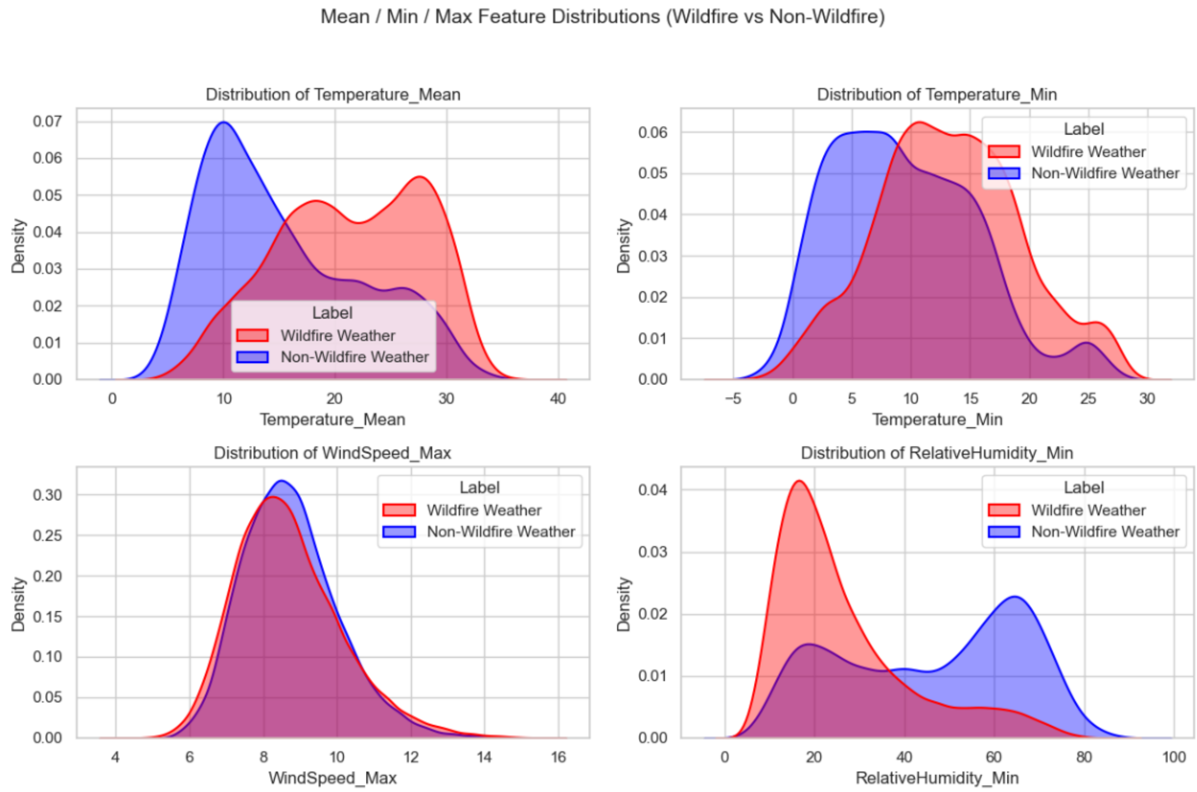

FIGURE 2.4: Density distribution of main features comparing wildfire and non-wildfire days

2.2.2 Vegetation dataset

For better weather information, we also wanted to include the vegetation data for each region. Since NDVI data is a satellite-derived data it can have data gaps due to cloud cover or image loss. The NDVI data was originally available as monthly data for each region, which means only one NDVI value per month for one region but for our modeling, we need daily values to align with the weather dataset for merging. To achieve this, we first created a complete daily date set which covers the entire time of the NDVI dataset and then paired each day with every region using a MultiIndex. This allowed us to re-index the NDVI dataset to daily data so that each Date-Region combination is there even if the original monthly data had no value for that day. From this process most of the newly created rows were initially filled with missing values (NaN) and then we addressed this through linear interpolation to fill the missing values by joining the first two know values which is the value from first month and the value from second month per region with a gradual increase or decrease in the new values based on the two points. Thus, we are able to generate continuous daily values for NDVI, which then will help with the merging process. After preprocessing NDVI data our next step was to merge the NDVI data with the weather data, For this we have to ensure that all the Date-Region combinations were perfectly aligned. To do this we checked the common date range shared by both NDVI and weather data using the earliest start date and latest end dates from each. Then we created a complete Multiindex which consists of all date region combinations available from both of the datasets. To generate a full index, we used

```
pd.MultiIndex.from_product([date_range, regions], names=["Date", "Region"])
```

and then applied reindex() to both of our datasets. Then we checked again for missing values, this step was crucial because for our approach as a time-series model, the models expect complete data across all the features so making sure everything is cleaned and structured was very important. After checking this we applied a left merge using Date-Region index as the key. The result of this was a single data frame that contains both the weather and vegetation daily data for all the region. This is the final input data we are using for our modeling pipeline which provides a rich combination of both atmospheric and vegetation data for the detection of anomalies in weather data that leads to wildfire.

2.2.3 Deseasonalization

The weather dataset with features like temperature, precipitation and soil moisture all follow strong and predictable seasonal patterns and these natural cycles can be a problem in our analysis where the model can see this as anomalies. For example, sudden spikes in temperature in summer is normal and a part of the cycle but having an unusual temperature spike in spring has to be considered. So, for addressing these issues, we applied a decomposition process called Seasonal-Trend decomposition using Loess (STL). STL is a widely used decomposition method to break down time-series data. This method will decompose into three components as seasonal, trend, and residual. The seasonal component will have the regular repeating patterns like annual cycle and the trend component will have longer patterns over time such as gradual warming toward summer and finally, the residual component will have the short-term fluctuations which does not have the trend or seasonal information. In our case, this residual was the exact signal we needed. It was very valuable because unusual spikes or anomalies in this component can train the model to learn properly without any noise from the seasonal up spikes and down spikes.

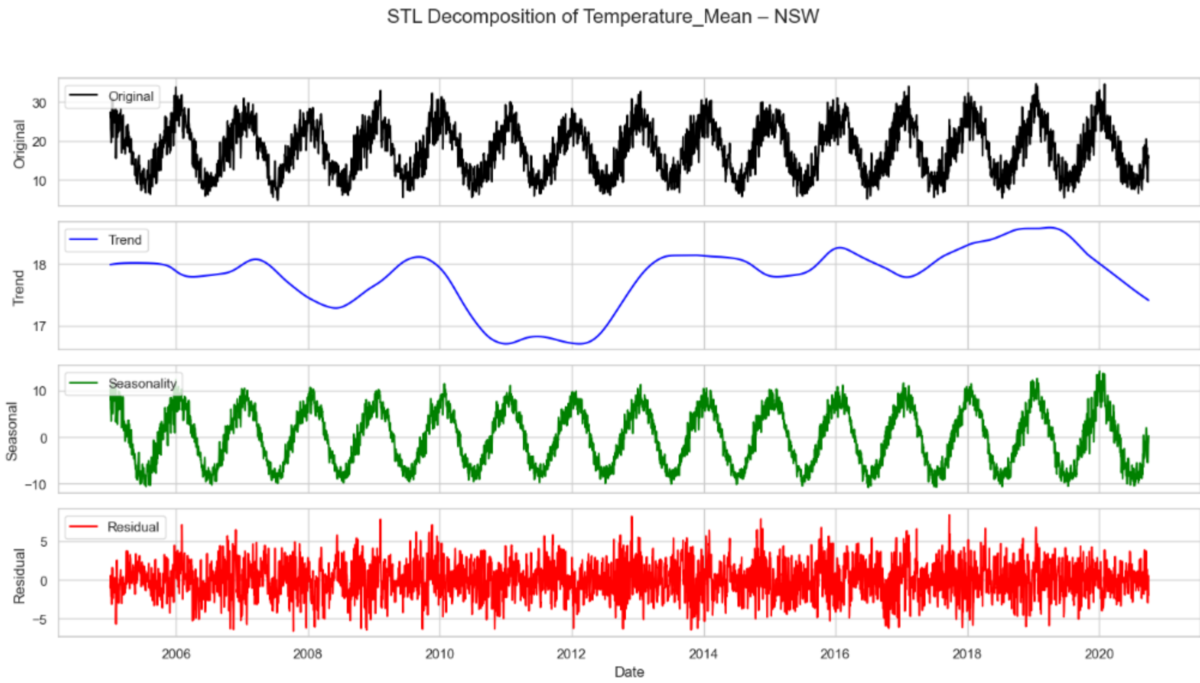

FIGURE 2.5: Decomposition of Temperature Mean

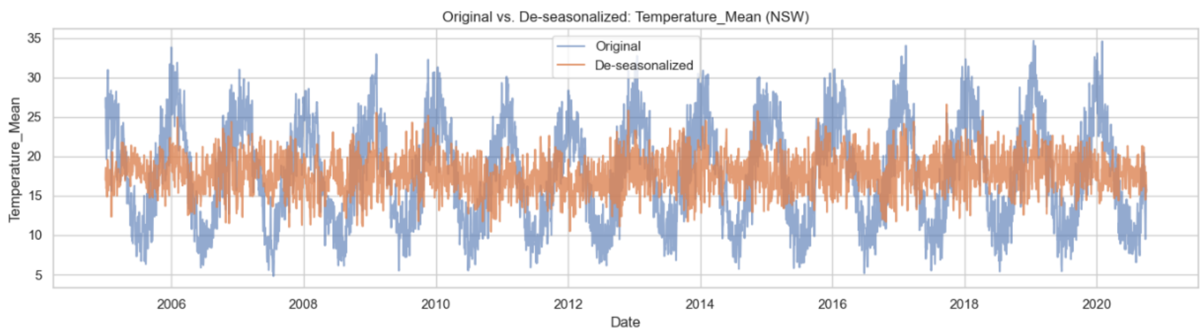

FIGURE 2.6: Decomposed Temperature Mean vs original Temperature Mean of the region NSW

We applied the STL decomposition separately for each region as each region has different climate conditions and therefore it can have different seasonal patterns. So for each region we extracted each feature of those respective region and then applied STL decomposition with a period of 365 for each features separately and stored their residual results in a new data frame as new input for our models, the figure 2.5 and figure 2.6 shows the decomposition of feature temperature mean. By removing seasonality, we enabled the model to focus more on true unusual weather and vegetation patterns, which we thought would improve the reliability and precision of our unsupervised approach.

Chapter 3

Methodology

Since labeled data are sparse, delayed, and sometimes geographically limited for supervised learning, exploring unsupervised learning can be another solution which do not need any label for training and can learn to find anomalies by training the models with normal weather data. To explore the effectiveness of our unsupervised learning for identifying weather anomalies that cause wildfires, we are here using three different models: a deep learning based LSTM Autoencoder, an ARIMA time series model, and a seasonal Fourier regression model, which is inspired by the Prophet model by Facebook. All models are only trained on non-wildfire weather data and then evaluated on both wildfire and unseen non-wildfire data.

3.0.1 Input data

All the models in this study are trained with daily, region-specific data that has been carefully preprocessed, which includes weather information and vegetation information. To prepare for our models, we used two datasets, non-wildfire data and wildfire data, which we had separated earlier. Since our focus was on unsupervised learning, we only trained our models with non-wildfire data, which allows our models to learn normal weather. The wildfire information has been placed out and only used during the evaluation process to check how well the models detect deviations related to wildfire events which will provide what all models performed well. We have also applied seasonal decomposition to analyze whether the model can learn better without the seasonal cycles or not.
Since we are using different models with different abilities, we used different representations of data: For LSTM Autoencoder, we used data into a fixed-length sequence like a 10-day window, where 10 rows at a time will be used for the model. In this way the model can learn temporal patterns across all days. The ARIMA and Fourier-based regression models were applied with each features individually. In Univariate models like this, Each feature has to be trained for each models thus the model can learn properly for that specific feature. This input foundation will allow us to explore how different modeling strategies responds to the same data, and then we can evaluate which method is more effective for finding anomalies in which all conditions.

3.0.2 LSTM Autoencoder

The first model we explored for this study is a Long Short-Term Memory (LSTM) Autoencoder, which is a type of recurrent neural network architecture model which is designed for sequential data reconstruction. An autoencoder is simply a neural network that learns a compressed representation of input data called an encoding and then reconstructs the original input from this compressed form. It mainly has two parts: an encoder which compresses the input data into lower lower-dimensional latent space, and a decoder which tries to reconstruct this lower-dimensional compressed data back to normal representation. We chose this model in our approach because it can learn the temporal patterns in the historical time-series data by training the model with normal weather which is the weather data that did not lead to wildfire in our case and we will find if the model can reconstruct this data or not by checking the reconstruction error[15].
Since our input data is time series which had ordered daily measurements, we thought of using LSTM layers instead of standard dense layers because LSTM recurrent neural network (RNN) are good at capturing temporal dependencies because they can retain information over time using a memory cell and its gates like input, forget, and output and this allows the model to focus on what to remember and what to forget across time steps.

Input data preparation
Our input data for the LSTM model consists of daily weather and vegetation information for seven regions in Australia. In order to capture short-term information efficiently, the model has a 10-day sequence window, which means that at a time 10 days will be given as input to the model. That is each training sample is a matrix of shape (10, n), where 10 is the number of days and n is the number of input features. Here we have 29 features including both weather and vegetation information. Before giving these features to the model, we have also applied MinMax scaling. This scaling will convert all the feature values into the range [0, 1]. This is done by subtracting the minimum value from the feature and dividing by the feature range. This way we will get a normalized input data, which is essential in neural networks as it ensures that the model can learn patterns more effectively without being biased toward features with larger numerical values.

Model Architecture
The goal of our LSTM Autoencoder is to learn what normal weather looks like over short time windows as of 10-day period, and then detect when future data significantly deviates from that pattern it learned to be normal.

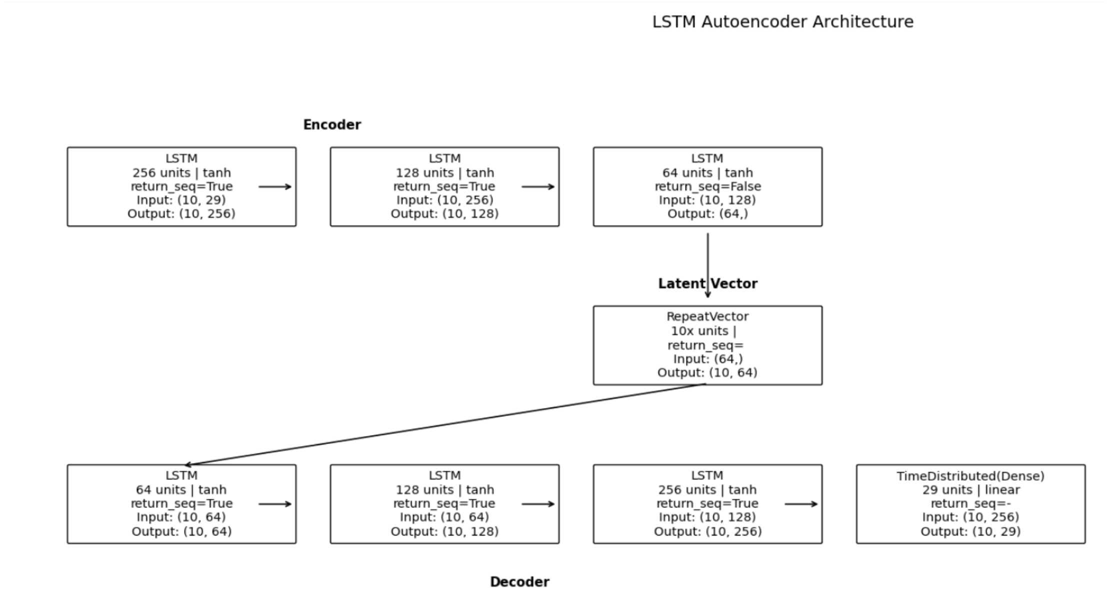

FIGURE 3.1: Long Short-Term Memory Autoencoder Architecture

`Encoder layer LSTM (256 units, return sequences = True):` This layer has 256 units means it has 256 memory cells, also known as neurons. Each unit can learn patterns over time, like what to remember through input gates, what to forget through forget gates, and what to output through the output gates, and this information is stored. The first layer processes the input 10-day sequence and then learns the short-term patterns across all time steps, and then it returns the entire sequence so that the next layer can continue learning from it.
LSTM (128 units, return sequences = True): This layer is built by the patterns learned from the previous layer, but compresses them by reducing the number of units from 256 to 128. LSTM (64 units, return sequences = False): This is the final layer of encoder. This layer only outputs the last hidden state, which is the information of that entire 10-day sequence from all the above layers.`Decoder layer` RepeatVector (length = 10): This layer represents the 64-dimensional latent vector 10 times, this way it can match the original sequence length which is 10 in our case. LSTM (64 units, return sequences = True): This layer starts the process of decoding. Each of the 10 time steps is processed here to learn how to reconstruct the original feature values from the compressed vector. LSTM (128 units, return sequences = True): This layer expands the learned reconstruction into 128 units. LSTM (256 units, return sequences = True): The final decoder LSTM layer is the same as the encoder's starting size. It then tries to restore the details of the original sequence. Time Distributed (Dense layer): This layer applies a fully connected network to each of the 10 time steps separately, which then produces a vector of predicted features for each day. This outputs the reconstructed version of the original 10-day input. We chose this architecture after a lot of experiments with many simpler versions and adjusting layer sizes to find a good balance between model complexity, reconstruction accuracy, and its anomaly detection sensitivity.
In this model, we have compiled using the Mean Squared Logarithmic Error (MSLE) as the loss

function. We used MSLE because it won't underestimates more than overestimates, means it focuses more on relative errors, making it better at detecting small yet meaningful deviations in weather patterns especially when values are small and vary across different scales, as it often happens before a wildfire, this is effective that it won't miss many potential dangers when we are working with normalized time-series data.

Reconstruction

Once the model is properly trained, the model receives new 10-day sequences and attempts to reconstruct them with the learning it had from the previous training. For each sequence X, the model produces a predicted sequence $\hat{X}$. The reconstruction error is then calculated as the mean absolute difference between the original and reconstructed sequence over all time steps and features:

$$\text{Reconstruction Error} = \frac{1}{T \times N} \sum_{t=1}^{T} \sum_{i=1}^{N} \left| X_{t,i} - \hat{X}_{t,i} \right| \tag{3.1}$$

Where:

- T is the sequence length (10 days),
- N is the number of features,
- $X_{t,i}$ is the value of feature i on day t,
- $\hat{X}_{t,i}$ is the reconstructed value for the same feature and day.

So when the model sees patterns similar to what it has seen during training, that is meant to be normal weather, Now the model tries to reconstructs it with error percentages and when the model sees unfamiliar pattern or inputs such as those related with wildfire conditions or unusual spike then the reconstruction becomes poor which then leads to higher reconstruction error.

Finding Anomalies

To find what is normal and anomalous from the data, we compute reconstruction errors on the training set as discussed above and now we select a threshold at the 95th percentile of the reconstruction error from training which means only the top 5% of highest reconstruction errors among the normal (non-wildfire) training sequences are considered abnormal and all the remaining are considered normal.

$$\text{Threshold} = \text{Percentile}_{95}(\text{Reconstruction Errors}_{\text{Training Set}}) \tag{3.2}$$

Any new sequence with a reconstruction error above this threshold is flagged as an anomaly, which potentially indicates a high-risk wildfire day or unusual weather pattern. This threshold is calculated without using any wildfire labels, maintaining the unsupervised nature of our study approach.

Performance Evaluation

Once anomaly scores are calculated for the test set, which also includes known wildfire days, we then compare the model's predictions with the ground truth labels, keeping in mind that the model never saw these labels during training. This comparison helps us evaluate how well the LSTM Autoencoder can actually find the real wildfire events.
For this analysis, we used the following metrics:

- **Mean Absolute Error (MAE)** and **R^2 Score** between reconstructed and true sequences to measure internal reconstruction quality in the training.
- **Symmetric Mean Absolute Percentage Error (SMAPE)** to see the magnitude of deviation,
- Standard classification metrics including **Precision**, **Recall**, **F1-score**, and a Confusion matrix to assess the anomaly detection performance of the model.

3.0.3 ARIMA

The second method we used for our unsupervised wildfire detection approach was to build upon the classical statistical model known as ARIMA, which stands for Auto Regressive Integrated Moving Average. This is very different from deep learning models. Unlike LSTM, ARIMA gives a fully interpretable and computationally efficient approach that is learning individual feature patterns over time instead of a multivariate learning. We choose ARIMA because it explicitly models the dependencies between previous observations which is called the autoregression and also it adjusts for trends and seasonality through differencing, which is integration, and it can smooth short-term fluctuations based on the past forecast errors called moving average[14]. This method is highly effective when the time-series data with mean and variance remain stable over time, thus making it a good approach for forecasting normal patterns in our data and then for flagging if it deviates.

Model Setup and Feature Selection

We focused the ARIMA analysis on the New South Wales (NSW) region for further transfer learning exploration and applied this to all the features, and each feature was treated as a separate univariate time series. This approach assumes that even an unusual pattern in a single feature may lead to wildfire conditions.

ARIMA Model Specification

We used a fixed model configuration with the order (p, d, q), which means:

- $p = 2$: the model looks at the previous two values (autoregression),

- $d = 0$: no differencing is applied, assuming the data is stationary after deseasonalization,
- $q = 2$: it considers the last two error terms (moving average component).

Although the model order was not tuned separately for each feature, this configuration can still provide a reasonable balance between learning temporal structure and avoiding overfitting. Once the model was fit on the historical data for a feature like `Temperature_Mean`, it was then used to generate forecasts for the full duration of the evaluation period, including wildfire and non-wildfire days.

Detecting Anomalies from Residuals

For each feature, the residuals were calculated as:

$$\text{Residual}_t = |\text{Actual}_t - \text{Forecast}_t| \tag{3.3}$$

We then took the absolute value of residuals and computed the 95th percentile of residuals from the training data. The value we get from this is used as our anomaly threshold, which is a boundary beyond which the model is considered to have seen something that is unexpected. If the residual for a day exceeded this threshold, that data point is then flagged as an anomaly for that feature. This process was repeated for all features, resulting in a binary anomaly flag for each feature per day, and then with this we can compare with the actual wildfire occurrences and can find how well the model performs.

Aggregating Risk from Multiple Features

For a given day, we aggregated the anomaly flags from all features by summing them into a Risk Score as given below :

$$\text{RiskScore}_t = \sum_{i=1}^{29} \text{Anomaly}_{t,i} \tag{3.4}$$

If the Risk Score was greater than or equal to 1, the day was flagged as a potential wildfire day (`FireRiskFlag = 1`). This way it assumes that even one highly abnormal feature could signal the risk of wildfire conditions.

Evaluation and Results

To assess the model's effectiveness, we compared the predicted `FireRiskFlag` values against the actual wildfire labels (known values but not used while training). Using these predictions, we computed several evaluation metrics as give below:

- **Precision:** how many of the days flagged by the model actually had wildfires.
- **Recall:** how many of the actual wildfire days were successfully identified.

- **F1-score:** a balanced measure of precision and recall.
- **Confusion Matrix:** to show the count of true positives, false positives, etc.

This evaluation allowed us to assess how well the ARIMA model could detect wildfire related deviations in weather patterns by only modeling each feature independently.

3.0.4 Fourier Series-Based Regression Model

The third method we used for our study was inspired by Facebook's Prophet model, which uses a Fourier series for modeling seasonality. We built a linear regression framework that has Fourier terms in it to capture the seasonality in the data [2]. So by using the Fourier series to capture the repeating patterns in individual features, we can find any deviations from these expected patterns, which is considered as seasonal behavior. This represents a periodic function as a sum of sine and cosine waves. These waves are used as seasonal functions in our regression model.

Model working

The core idea is to train a separate univariate linear regression model for each weather feature using only non-wildfire data. Each regression learns how that feature changes over time using a combination of:

- A time variable t (representing days since the start),
- Multiple Fourier terms: sine and cosine functions that represent different seasonal frequencies.

For each feature, we created a dataset with:

- **Target** y: the value of the weather feature (e.g., `Temperature_Mean`),
- **Inputs** X:
  - A time index t,
  - Fourier terms of the form $\sin\left(\frac{2\pi kt}{365}\right)$ and $\cos\left(\frac{2\pi kt}{365}\right)$ for orders $k = 1$ to 3.

The regression equation for a feature like Temperature_Mean is expressed as:

$$\text{Temperature}_t = \beta_0 + \beta_1 t + \sum_{k=1}^{3} \left[\alpha_k \cdot \sin\left(\frac{2\pi kt}{365}\right) + \gamma_k \cdot \cos\left(\frac{2\pi kt}{365}\right)\right] + \epsilon_t \tag{3.5}$$

Where:

- β_0 is the intercept,
- β_1 models any slow linear drift over time,
- α_k and γ_k are the learned weights for the sinusoidal seasonal components,
- ϵ_t is the residual error — which we later use to flag anomalies.

Training and Prediction

The model was trained using ordinary least squares linear regression, with the training data restricted to non-wildfire days. Once trained, the same Fourier-based model was used to generate predictions for unseen data, including wildfire days.

Anomaly Detection Using Residuals

After generating predictions, we computed the residuals as the difference between the actual observed value and the model's predicted value for each day:

$$\text{Residual}_t = |\text{Observed}_t - \text{Predicted}_t| \tag{3.6}$$

Anomalies are identified when the magnitude of residuals exceeds the 95th percentile threshold computed from the training residuals as in the equation 3.6. This threshold helps distinguish typical seasonal variability from genuinely unusual deviations. If on a particular day, the temperature was much higher or lower than what the seasonal model expects, that day would be flagged as anomalous for that feature.

Aggregating Anomalies into Risk Scores

Since we repeat this process independently for each of the 29 weather features, a single day might show anomalies in multiple variables (e.g., high radiation and low humidity). To combine this information, we assigned each day a Risk Score, defined as the sum of anomaly flags across all features. A higher risk score indicates that more features exhibited unusual behavior on that day. We then applied a final rule: if the Risk Score was greater than or equal to 2, the day was classified as a potential wildfire risk day.

This aggregation logic assumes that real wildfire conditions typically do not stem from one isolated weather anomaly, but from a convergence of multiple abnormal indicators (such as hot, dry, windy, and low vegetation moisture).

Model Evaluation

To evaluate how well this method could detect wildfire-prone conditions, we compared the model's predictions with actual wildfire event dates (known but not used during training). We computed standard classification metrics give below:

- **Precision:** how many of the flagged days were actual fire days,
- **Recall:** how many of the actual fire days were correctly flagged,
- **F1-score:** the harmonic mean of precision and recall,
- **Confusion Matrix:** to visualize true/false positives and negatives.

Chapter 4

Results and Discussion

In this section, we are going to discuss the results and performance of all Unsupervised models that we used for this study. We start with LSTM, which is trained with seasonal and non-seasonal data. Then Fourier series model, which is trained with seasonal data and finally ARIMA, which is trained with non-seasonal data and we also performed how well this model works with transfer learning, whether the model is able to detect anomalies of other regions even though its only trained for one specific region.

4.1 LSTM

Here we are checking the performance of the LSTM model with both seasonal and non-seasonal data, considering its reconstruction errors, ROC curve, training and validation loss, Performance matrices and Latent space visualization.

4.1.1 LSTM model with seasonal data

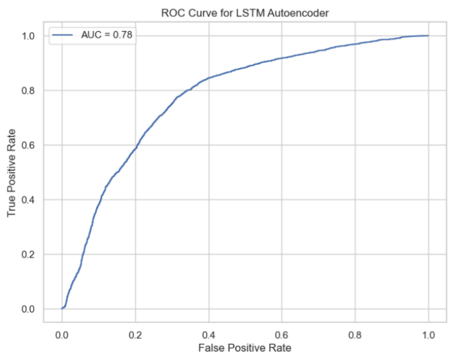

FIGURE 4.1: ROC curve for LSTM autoencoder (seasonal) showing the model's capability to distinguish wildfire and non-wildfire days using reconstruction error

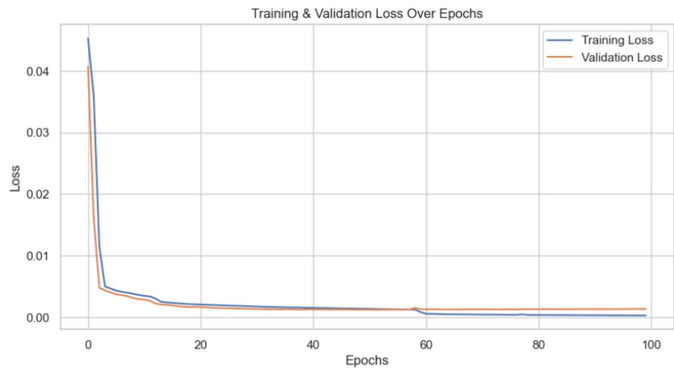

FIGURE 4.2: Training and validation loss curve for the LSTM autoencoder trained on seasonal weather data

The loss curve of the LSTM model shows how the model's reconstruction loss evolved during training as in the figure 4.2. This is used to check whether the model is able to learn anything, or is it overfitting, or is it generalizing well to unseen data. Training loss in the graph shows how well the model reconstructed the data it was trained on, and the Validation loss shows how well the model have reconstructed new data it had never seen. The x-axis of the graph is the number of training epochs or iterations it had, and the y-axis is the reconstruction error. As we look at the graph, we can see both the training and validation loss decrease sharply in the beginning of the iterations and then finally converge and flatten, which means it reached a point where there is not much left to learn.
The ROC curve in the figure 4.1 is a graph that shows the trade-offs between true positive rates and false positive rates. In our case we want to separate wildfire and non-wildfire days based on its reconstruction errors. The x-axis of the ROC curve is the False Positive Rate which means it shows how often the model falsely thinks there is a wildfire when there is not any and the y-axis is the True Positive Rate means it shows how often the model correctly identifies the actual wildfires. So the curve we see in the graph is drawn by sliding the threshold across the reconstruction errors and recalculating its True Positive Rate(TPR) and False Positive Rate (FPR) at every point. We got an AUC curve of 0.78,indicating that the model is generally able to assign higher anomaly scores to wildfire days compared to non-wildfire days across various thresholds. This suggests that the model is not simply making random predictions, but is also learning meaningful temporal patterns from the seasonal weather data through its unsupervised training.

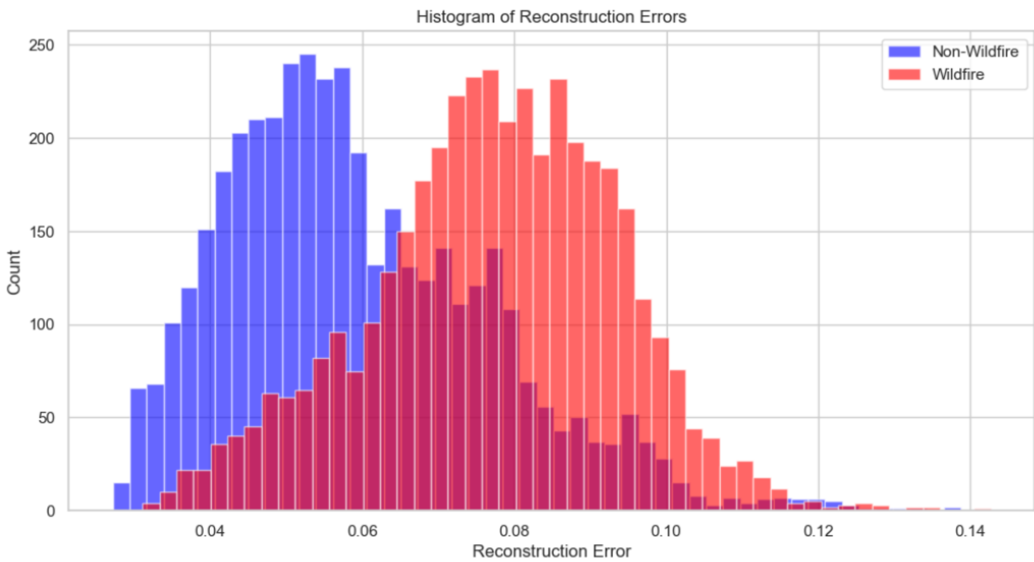

FIGURE 4.3: Histogram of reconstruction errors from LSTM (seasonal), comparing distribution for wildfire (red) and non-wildfire (blue) sequences

This is the histogram of reconstruction error, it shows how often reconstruction error occurred. Since the LSTM is trained only on non-wildfire data, the expectation is the normal data will be reconstructed well and the wildfire which are considered anomalies will confuse the model which causes high errors. The x-axis of this graph shows the reconstruction error, showing how far the model's predictions were from the true values and the y-axis shows the number of counts that had that much amount of error. There are two color groups blue bars for non-wildfire data and red bars for wildfire data. As we look close in the figure 4.3 we can see the blue bars are concentrated on left side which has lower errors, there are more counts of non wildfire data with error 0.04-0.06 and the red bard are shifted more towards right side which have higher error and most of its count has an error of 0.07-0.09 as per our expectation but there are also lot of counts with some overlap in the middle but the two groups are clearly not identical. This is a promising result that the model is doing what it was supposed to do, but still it does not have a complete separated reconstruction error for wildfire and non wildfire data.

Classification Report:

	precision	recall	f1-score	support
Non-Wildfire	0.80	0.55	0.65	4129
Wildfire	0.66	0.86	0.75	4129
accuracy			0.71	8258
macro avg	0.73	0.71	0.70	8258
weighted avg	0.73	0.71	0.70	8258

FIGURE 4.4: Classification report for LSTM model trained on seasonal data, indicating precision, recall, and F1-score for both classes

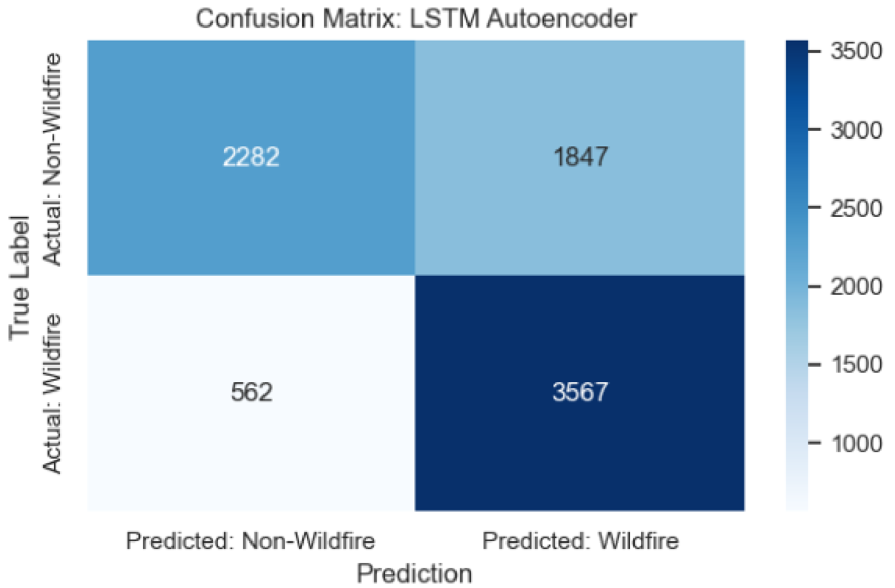

FIGURE 4.5: Confusion matrix for LSTM (seasonal), showing true/false positive and negative predictions of wildfire vs non-wildfire

The classification report mainly shows four main key matrices: Precision, Recall, F1-score and

Support for both Wildfire and Non-Wildfire classes. Precision means how many wildfires the model said were actual wildfires. Recall means from all the wildfire cases, how many we were able to find. F1-score is a balance between precision and recall. Support is the count of how many actual data points were in each class. As we see the results figure 4.4 and figure4.5 , For Non-wildfire, the model was very precise 80% of the time for predicting normal weather with no wildfire. For Wildfire, the precision was lower which is 66% meaning it sometimes flags a normal day as a wildfire. For Non-wildfire, recall was only 55% meaning some days are being flagged as wildfire and for Wildfire, recall was high with 86%, meaning the model was able to detect the majority of wildfire cases, which is what we needed for a strong wildfire detection system. Wildfire F1-score was 0.75 means the model have a quite good balance with precision and recall. The overall accuracy for the LSTM model with seasonal data was 71% which is impressive with an unsupervised approach.

We analyzed the performance of the model which has the anomaly threshold from its reconstruction error. We used a Confusion matrix which is a 2x2 table that shows how many time our model was correct with true positives(TP) and true negatives (TN) and was wrong with false positives (FP) and false negatives (FN). In our case TP=2282, FP=1847, FN=562 and TN=3567. The model was able to correctly detect 3567 wildfires showing it's not missing too many critical events but also it falsely labels 1847 normal days as wildfires and it actually missed 562 real wildfire cases which are more serious in safety applications.

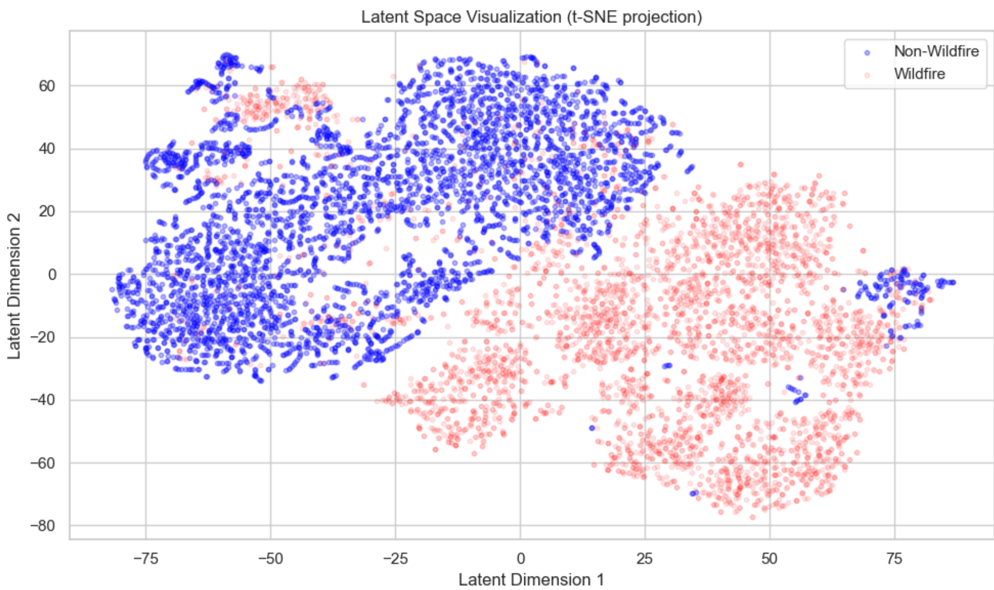

Figure 4.6: t-SNE latent space visualization of LSTM autoencoder trained on seasonal data. Blue points represent non-wildfire sequences, red points indicate wildfire sequences

The figure 4.6 above is the representation of a high-dimensional latent space using t-SNE(t-distributed stochastic neighbor embedding), It is a powerful technique that is used for visualizing

complex patterns and relationships between data points. This graph will show how the data points are structured in the model's latent space. So normal sequences, which are the non-wildfire data points, should get mapped into consistent, repeatable patterns by considering that it follow a regular seasonal trend and the anomalous sequences which are wildfire points should get mapped differently because they should deviate from the learned pattern from normal weather data.
So the LSTM autoencoder learns a compressed representation of the input data called a latent vector which captures the main information of the input data for reconstructing it, and for each input sequence both wildfire and non-wildfire, we will extract this latent vector from the encoder part of our autoencoder. As you can see in the image red points are wildfire sequences and blue are non wildfire and if we closely look we can see the seperation of datat points in the latent space some blue clusters are packed tightly in few areas towards upper left and the formed cluster will have similar patterns showing the learning from normal weather conditions and the red points are also separated from the blue points not much overlapping but still some data points are plotted in similar spaces. These partial groupings are similar in normal weather patterns, indicating their seasonal behaviour and most of the wildfire sequences do not belong to these clusters, showing separations between wildfire and non-wildfire data points.

4.1.2 LSTM model With no seasonal data

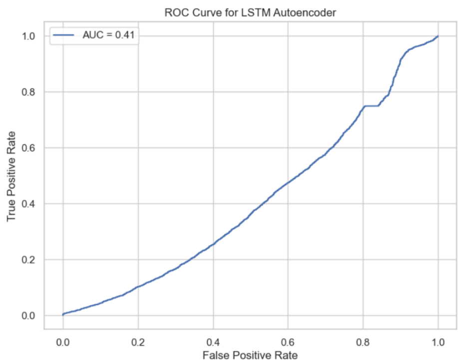

FIGURE 4.7: ROC curve for LSTM autoencoder trained on deseasonalized weather data, indicating worse performance even while random guessing

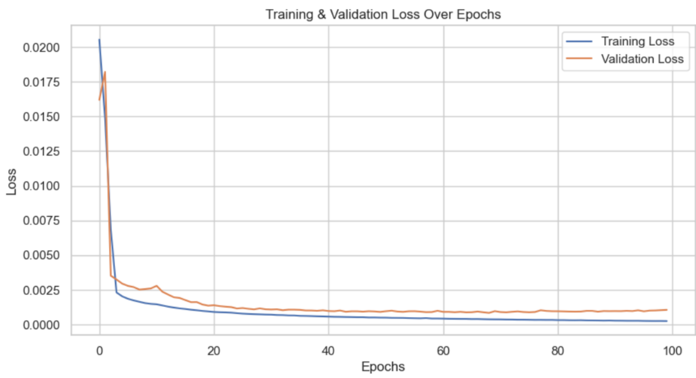

FIGURE 4.8: Training and validation loss for non-seasonal LSTM, showing convergence but lack of meaningful anomaly detection capability

The ROC curve for the LSTM model trained on non-seasonal data shows a significant decline in performance with an AUC of 0.41 compared to the seasonal model with 0.78. This value is below the 0.5 threshold of random classification, which indicates that the model performs worse than guessing the chances of wildfire events based on reconstruction error. This result suggests that the model has not learned that much meaningful features that correlate with the occurrence of wildfires and it is in fact, not classifying the majority of sequences properly. This poor separation is likely due to

the absence of seasonality in the input data, which removes much of the natural structure and other repeating weather patterns that would otherwise help define what normal weather looked like.
After the low predictive performance, when we look at the loss curve for both training and validation sets, it shows a smooth and consistent decrease across epochs, eventually converging without signs of overfitting. This indicates that the model has successfully minimized reconstruction error in a general sense but these learning patterns do not contribute to anomaly detection, which means the model is stable and capable of reproducing input sequences but the information it learns is not enough to separate wildfire-related behavior from regular weather patterns. This shows the importance of seasonal structure in the data when using sequence-based unsupervised models for anomaly detection with LSTM based autoencoders.

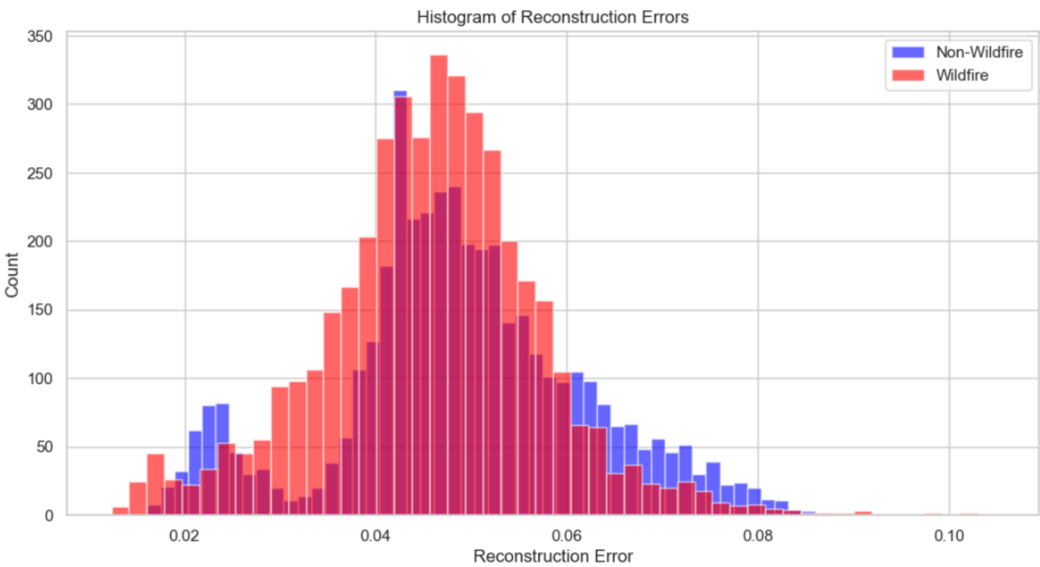

FIGURE 4.9: Histogram of reconstruction errors for the LSTM model trained on non-seasonal data. Overlap between wildfire and non-wildfire points shows poor separation

From the reconstruction error distribution for the non-seasonal model, it shows a substantial overlap between wildfire and non-wildfire sequences with both classes concentrated around similar error ranges from 0.04-0.06. The separation here is very minimal or absent compared to the reconstruction error from the seasonal LSTM. This indicates that the model struggles to differentiate between normal and anomalous weather patterns when seasonality is removed from the input data. This clear lack of divergence in this error distributions suggests that the model perceives both wildfire and non-wildfire sequences as equally reconstructable showing that the input features lack the necessary information for distinguishing anomalous behavior. From this result we can say without periodicity or consistent climatic trends, the model is struggling to find what a normal weather looks like.

```
Classification Report:
              precision    recall  f1-score   support

 Non-Wildfire       0.61      0.06      0.11      4169
     Wildfire       0.51      0.96      0.66      4169

     accuracy                           0.51      8338
    macro avg       0.56      0.51      0.39      8338
 weighted avg       0.56      0.51      0.39      8338
```

FIGURE 4.10: Classification report for LSTM trained on non-seasonal data, showing lower precision and overprediction of wildfire cases

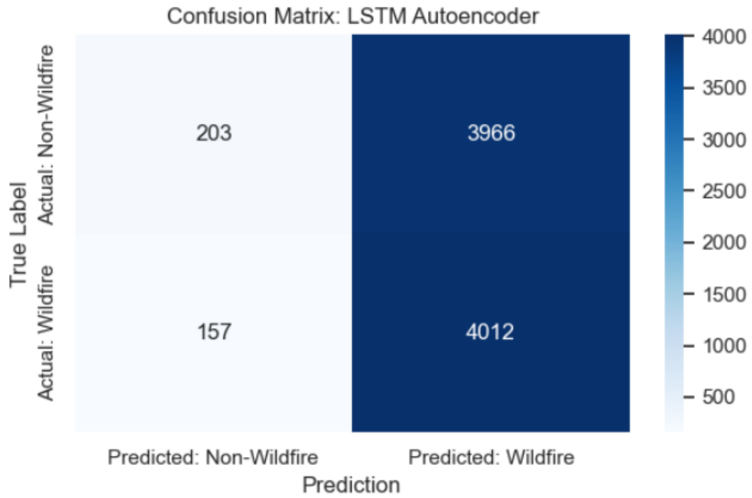

FIGURE 4.11: Confusion matrix for non-seasonal LSTM indicating high false positive rate and poor non-wildfire recognition

As you look at the classification report and confusion matrix for the non-seasonal LSTM model, it indicates a strong imbalance in the model's predictions, with more tendency to classify the majority of sequences as wildfires regardless of their true label. The confusion matrix also shows a high number of false positives, where non-wildfire sequences are incorrectly flagged as wildfire and a significantly low true negative count. We can see this reflected in the performance metrics: the recall for the wildfire class is very high (0.96), but its precision drops to 0.51, indicating that a large proportion of the predicted wildfire cases are incorrect. Conversely, the non-wildfire class exhibits extremely low recall (0.06) and an F1-score of only 0.11, showing that the model fails to recognize normal sequences accurately.

From these results it shows the model fails to recognize normal weather. The overall accuracy and F1-scores are very low than in the seasonal model, suggesting that the absence of seasonal information has decreased the model's ability to form a reliable baseline for anomaly detection, the model appears to treat nearly all input sequences as anomalous, resulting in widespread misclassification as you see in the report.

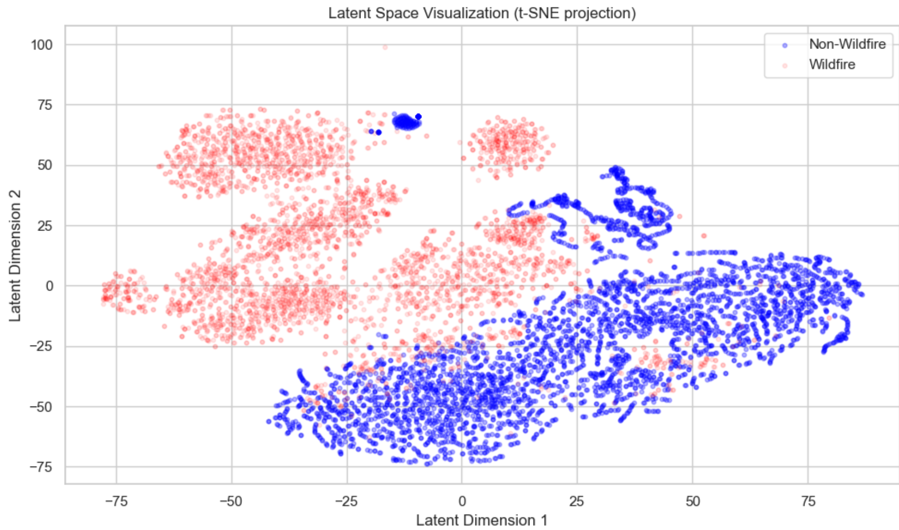

FIGURE 4.12: t-SNE visualization of latent space from non-seasonal LSTM model, showing quite good seperation with strange dense points

The latent space visualization of the LSTM autoencoder trained without seasonal data shows separation between wildfire and non-wildfire points but as you see in the figure 4.12 there are strange dense point of non-wildfire data near the wildfire regions and also we can see some wildfire points overlapped in the non-wildfire region which can confuse the model to predict which is non-wildfire and which is wildfire. Even though we can see visible separation, the model still produces poor classification performance 4.11. This can be related to the latent space which doesn't directly convert into clean reconstruction behavior, and since the model was trained on deseasonalized data, it likely struggled to learn the periodic structure that helps distinguish small weather signals preceding wildfires. As a result, the reconstruction errors for both wildfire and non-wildfire inputs begin to overlap, making it difficult for the thresholding to confidently identify anomalies even if the latent space representations appear separated visually. This result indicates that the autoencoder's decoder network failed to reconstruct non-seasonal data which resulted in very poor classification performance and overlapping reconstruction errors observed earlier 4.9.

As we look at both the LSTM autoencoders trained on seasonal and non-seasonal weather data, it clearly highlights the importance of temporal seasonal structure in anomaly detection. With the seasonal data, the LSTM model learns effectively, showing converging loss curves and an AUC of 0.78 and a meaningful separation in reconstruction errors even though there were slight overlapping. The model was able to achieve high recall of 86% for wildfire events and its structured separate latent representations forming clusters and learning similar patterns for both wildfire and non-wildfire data points. On the other hand, the LSTM with a non-seasonal data showed proper learning graphs but its AUC drops below 0.41 and the reconstruction errors had lot of overlapping 4.9 and the latent representations had strange dense points near wildfire data point regions even though it looks like separable data point the model struggled to generate causing high reconstruction errors and

overlapping and thus poor classification results.
These results show that seasonality is very important for LSTM autoencoders to build a reliable baseline of normal weather. Without it, the model fails to recognize anomalies, thus making seasonal structure a very important component for unsupervised wildfire detection for LSTM models.

4.2 Fourier Series-Based Regression Model

In this section, we evaluate Fourier series-based regression model to detect anomaly with seasonal weather features. Each weather variable are treated individually which means each features have its own models. In this section we discuss the results and performance of the model using a detailed view of model fitting for one feature Temperature Mean for NSW, a combined multi-feature view across all 29 weather variables used in the model and the classification report.

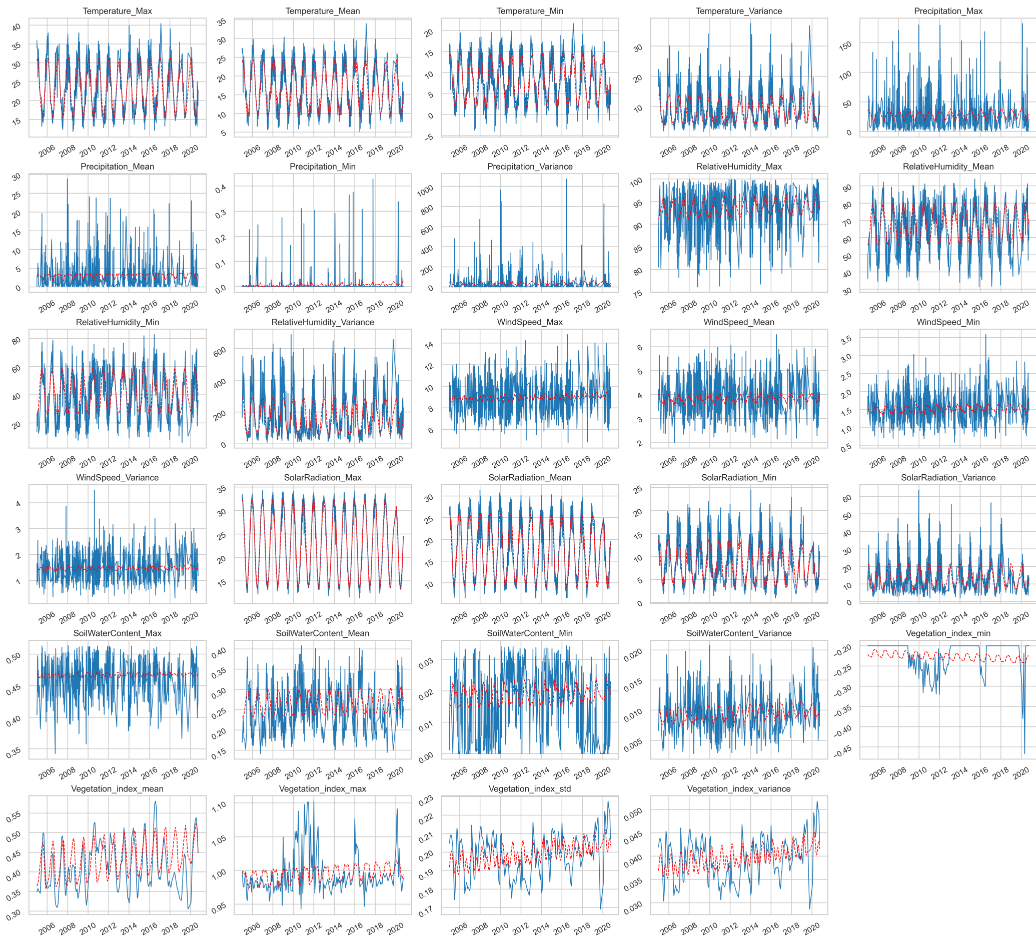

FIGURE 4.13: Fourier series model forecast on all features for Region NSW (with season) showing strong alignment with actual values for features exhibiting smooth seasonal patterns, but performs less accurately on more irregular variables, highlighting its strength in modeling periodicity and limitations with non-cyclic trends

The above large figure shows Fourier regression fits for all 29 different weather variables, X-axis in all figures shows the date and Y-axis show the value ranges of that respective feature. Blue line in the figure is the actual value and the red dashed line is the fourier models prediction. When you closely look 4.13 you can see that feature like Temperature, Solar Radiation, and Humidity shows strong and regular seasonal patterns, and thus the model fits them quite well but other features like percipitation or windspeed, the actual values shows lot of irregular spikes which makes the model harder to capture using periodic functions but still the model was able to capture the broad seasonal structure across all features.

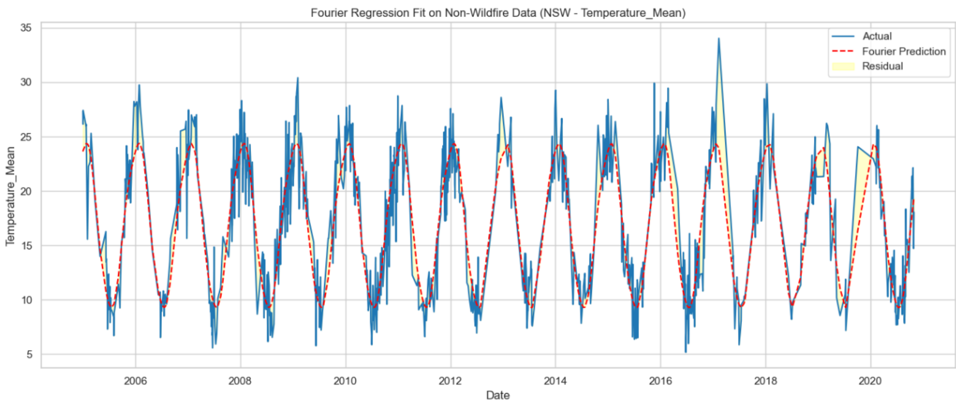

FIGURE 4.14: Fourier series model forecast on temperature mean for Region NSW (with season) showing how well the model predicts the smooth feature

In this figure for temperature mean you can see that the model was able to predict it quite well because it follows strong seasonal pattern.

```
Classification Report:
              precision    recall  f1-score   support

           0       0.38      0.51      0.43     13994
           1       0.68      0.55      0.61     26277

    accuracy                           0.54     40271
   macro avg       0.53      0.53      0.52     40271
weighted avg       0.57      0.54      0.55     40271

Confusion Matrix:
[[ 7203  6791]
 [11935 14342]]
```

FIGURE 4.15: Classification report and confusion matrix of the Fourier model which performs moderately better in detecting wildfire days with a precision of 0.68, compared to non-wildfire days with a low precision of 0.38, indicating a tendency to over-predict wildfire events. The overall accuracy is 54%, and the imbalance in false positives suggests that while the model captures periodic wildfire-related anomalies but it struggles to generalize over irregular patterns, particularly for non-wildfire sequences.

The Fourier model achieves a moderate F1-score of 0.61 for wildfire detection. The recall for wildfire is 55%, indicating that slightly more than half of actual wildfire cases are detected. However, the false positive rate remains high as shown by 6,791 normal days were incorrectly flagged as wildfire risk. Also the non-wildfire class suffers from both low precision and F1-score with many of its data points misclassified as wildfires. Overall accuracy is 54%, which is not that great.
In conclusion, the Fourier model successfully captured some seasonal trends in weather data, allowing it to flag deviations as potential wildfire risks but even though the model detected many wildfire days through these anomalies, it also misclassified many normal days as wildfires. Overall, it can be seen as a simple but effective baseline for identifying weather irregularities but not a great way to approach features with very irregular spikes without strong seasonal information or patterns.

4.3 ARIMA

In this section, we evaluate the ARIMA (AutoRegressive Integrated Moving Average) model's ability to capture anomalies that may indicate wildfire risk. The data we used for this model is without any seasonal information and its a univariate model, each features are treated individually. For performance analysis we provided one large figure showing the model's prediction across all weather features as we did for the Fourier model and another figure zoomed into a single variable Temperature Mean of NSW. We also review its classification performance using anomaly-based fire risk labeling.

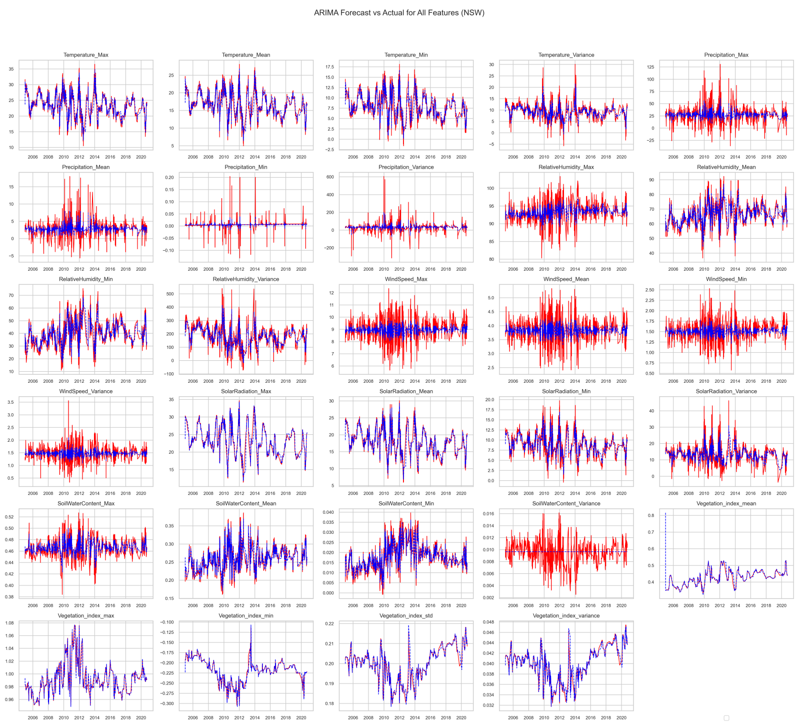

FIGURE 4.16: Residual-based anomaly detection across all features using ARIMA model for deseasonalized data in NSW showing how well the model predict features even without any smooth seasonal features but struggles with other features like percipitation and windspeed

In this figure the X-axis shows date and y-axis are the value ranges of the variables. Here red line shows the actual values and the blue dashed line is the forecasted values by ARIMA. As you can see the model closely follows the actual values even without the seasonal trends. But in certain variables like windspeed due to its sudden drops and high, the model struggles to predict the values because the residuals here still behaves more like a noise or random ups and downs.

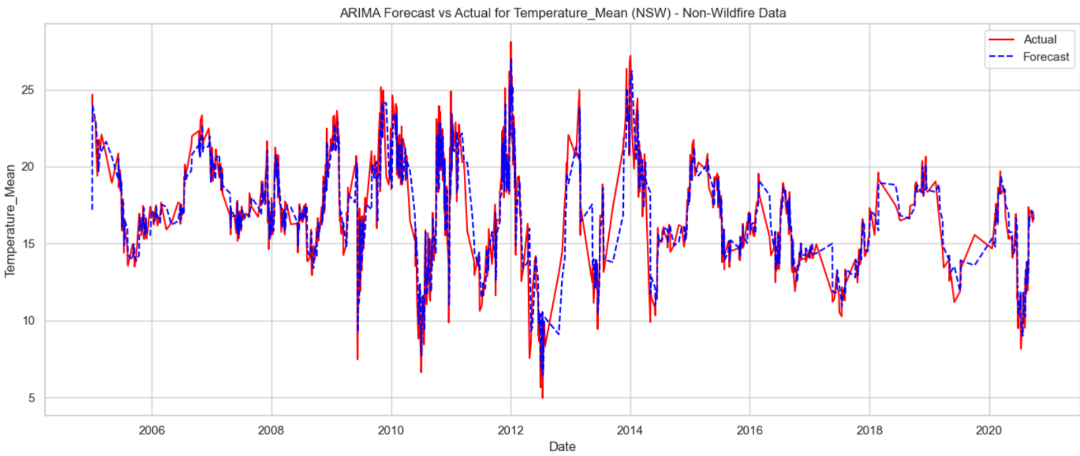

FIGURE 4.17: ARIMA forecast on temperature mean for Region NSW (without season) shows the ability of the model to predict the irregular spikes of the feature

If you look at the temperature mean, without any seasonal trend the model really worked well with its prediction even though the signal was irregular, this shows the models ability to learn non periodic structures properly.

Classification Report:

	precision	recall	f1-score	support
0	0.285	0.639	0.395	1155
1	0.868	0.598	0.708	4598
accuracy			0.606	5753
macro avg	0.577	0.619	0.551	5753
weighted avg	0.751	0.606	0.645	5753

FIGURE 4.18: Classification report for ARIMA model. Model trained on deseasonalized data, tested against wildfire labels with high precision for wildfire and descent recall for both classes

When you look at the classification report, For wildfire the precision is 0.86 which is high that means the model can predict the actual wildfire correctly. Recall is 0.59 which is moderate, this indicate that even though the model is predicting a good amount of wildfire still it misses a significant portion of it. For Non-wildfire precision is quite where many of the cases flagged as non-wildfire which was not true, this suggest that the model tends to over predict wildfires and the recall is 0.63 which is good that it captures a good amount of actual non-wildfire cases even with limited precision.

4.3.1 Transfer learning

Here we have checked the transfer learning abilities of ARIMA to assess whether a model trained on a specific region like NSW could effectively generalise to other regions. We used deseasonalised data here as well.

```
Starting evaluation...
Evaluating region: NT

 TRANSFER EVALUATION: Trained on NSW, Tested on NT
              precision    recall  f1-score   support

 Non-Wildfire       0.00      0.00      0.00       730
     Wildfire       0.84      0.77      0.80      5023

     accuracy                           0.67      5753
    macro avg       0.42      0.38      0.40      5753
 weighted avg       0.73      0.67      0.70      5753

Confusion Matrix:
 [[   0  730]
 [1178 3845]]
Evaluating region: SA

 TRANSFER EVALUATION: Trained on NSW, Tested on SA
              precision    recall  f1-score   support

 Non-Wildfire       0.69      0.44      0.54      3767
     Wildfire       0.37      0.62      0.46      1986

     accuracy                           0.50      5753
    macro avg       0.53      0.53      0.50      5753
 weighted avg       0.58      0.50      0.51      5753

Confusion Matrix:
 [[1671 2096]
 [ 764 1222]]
Evaluating region: QL

 TRANSFER EVALUATION: Trained on NSW, Tested on QL
              precision    recall  f1-score   support

 Non-Wildfire       0.06      1.00      0.12       250
     Wildfire       1.00      0.33      0.49      5503

     accuracy                           0.35      5753
    macro avg       0.53      0.66      0.30      5753
 weighted avg       0.96      0.35      0.47      5753

Confusion Matrix:
 [[ 250    0]
 [3712 1791]]
```

FIGURE 4.19: Transfer evaluation performance of ARIMA model trained on NSW and applied to NT, SA and QL

```
Evaluating region: TA

 TRANSFER EVALUATION: Trained on NSW, Tested on TA
              precision    recall  f1-score   support

 Non-Wildfire       0.58      0.04      0.08      4353
     Wildfire       0.23      0.90      0.37      1400

     accuracy                           0.25      5753
    macro avg       0.41      0.47      0.23      5753
 weighted avg       0.50      0.25      0.15      5753

Confusion Matrix:
 [[ 195 4158]
 [ 141 1259]]
Evaluating region: VI

 TRANSFER EVALUATION: Trained on NSW, Tested on VI
              precision    recall  f1-score   support

 Non-Wildfire       0.52      0.39      0.45      3583
     Wildfire       0.29      0.41      0.34      2170

     accuracy                           0.40      5753
    macro avg       0.41      0.40      0.39      5753
 weighted avg       0.43      0.40      0.41      5753

Confusion Matrix:
 [[1391 2192]
 [1275  895]]
Evaluating region: WA

 TRANSFER EVALUATION: Trained on NSW, Tested on WA
              precision    recall  f1-score   support

 Non-Wildfire       0.13      1.00      0.23       156
     Wildfire       1.00      0.81      0.90      5597

     accuracy                           0.82      5753
    macro avg       0.56      0.91      0.56      5753
 weighted avg       0.98      0.82      0.88      5753

Confusion Matrix:
 [[ 156    0]
 [1059 4538]]
```

FIGURE 4.20: Transfer evaluation performance of ARIMA model trained on NSW and applied to TA, VI and WA

NT (Northern Territory): The model completely failed to recognise non-wildfire days by classifying all days as wildfire. This shows the residual pattern in NT, probably due to its high monsoonal weather, it differed a lot from NSW. The ARIMA model trained on NSW interprets that even the normal NT

variations as anomalies.
SA (South Australia): The model shows moderate balance between both classes with precision 0.67 for non-wildfire, 0.36 for wildfire, but still lacks strong performance with overall accuracy 52%. SA shares more weather structure to NSW than NT but many false positives and false negatives were there.
QL (Queensland): We can see extremely biased predictions here where the model thinks everything is non-wildfire recall is 1.00 for non-wildfire. This reversal in behavior compared to NT suggests the NSW-trained model underestimates anomalies in QL. Probably the noise in QL might be lower or smoother, so the model sees deviations as normal instead of anomalies.
TA (Tasmania): Opposite trend from QL, Here most wildfire days are correctly flagged, but non-wildfires are not, it had a lot of false positives with a recall of 0.11 for non-wildfires. In this region, even slight irregularities are flagged which caused the model to overreact.
VI (Victoria): Here we have balanced but underperforming results. Both classes have a poor predicted accuracy with 41% with lot of confusions between wildfire and non-wildfire days.
WA (Western Australia): High performance on wildfire detection with recall is 0.66, but it completely ignores non-wildfire days, recall is 1.00 for non-wildfire, but sample size is very small. The small number of non-wildfire sample with only 156. The model performs relatively well on wildfires but the confidence is very low, probably due to sample size imbalance. Here matrix looks good due to the data imbalance but for proper results we need more balanced support.

Overall, the residual-based threshold from NSW was not generalizable across other regions with very different weather profiles, especially when seasonal patterns were removed. Climatic diversity is important, as even though we removed the seasonal trend still the noise and residuals change with locations. From this, it's clear that ARIMA residuals are highly region-sensitive, and a single region-based threshold is not enough to generalize to other regions.

Chapter 5

Conclusion

In this thesis, we have explored whether unsupervised learning techniques can be effectively used for wildfire risk detection by using weather forecast time-series data without relying on labeled wildfire events. This study needs to be explored because of many real-world problems, especially across diverse geographic regions where reliable labeled wildfire data may be sparse, delayed, or even unavailable, and since this natural disaster is very dangerous to our environment and habitat finding a solution was necessary. To address this challenge we explored and compared three distinct unsupervised anomaly detection strategies: LSTM Autoencoders, Fourier Series Regression and ARIMA-based model each offering a different way of modeling normal environmental behavior and identifying these models deviations as anomalies which can be used for wildfire risk.

From all our evaluations of all models, we end up with some meaningful conclusions. First, the LSTM Autoencoder showed clear evidence that deep sequence-based models trained fully on seasonal non-wildfire data was able to learn meaningful representations of normal weather and also the model demonstrated a strong ability to detect wildfire-related anomalies when seasonality was retained by achieving an AUC of 0.78 and an 86% recall on wildfire days. Also the latent space visualization and reconstruction error distributions confirmed the model's ability to separate anomalous sequences from regular ones. However, when we removed seasonal structure,we saw that the model's performance dropped very bad, with poor reconstruction error showing the importance of temporal seasonal patterns. Without seasonality, the LSTM essentially lost its ability to define what "normal" weather looks like which then resulted in high false positives. This confirms that the autoencoders can identify anomalies but only when the data preserves meaningful seasonal patterns, also showing we can approach with unsupervised learning to find wildfire events quite well with descent overall performance.

The second approach was the Fourier Series Regression, which was focused on modeling individual weather features using periodic functions by retaining the seasonality of weather variables. The model produced smooth and cyclic predictions that aligned with features that have similar strong annual trends like temperature and solar radiation. However, it struggled to read sharp ups and downs, particularly in irregular features like wind speed and precipitation. These limitations are not bad but are limited to its design where the Fourier model assumes that any sharp deviation from the expected seasonal cycle is anomalous which makes it function well as a baseline seasonal anomaly detector but lacks the sensitivity to pick up small but meaningful nonlinear interactions across features. But other than that the model helped validate our hypothesis that univariate models can still flag useful deviations, although there are limitations in a multi-dimensional situation. These specific univariate approaches are functional but less robust than multivariate alternatives for real-world anomaly detection when we focus on applying smooth and periodic approximations.

Finally, the ARIMA-based anomaly detection model which was fully trained on deseasonalized data which revealed strong insights when we tested its transferability across regions and its ability to find deviations as a univariate model with no seasonal information. We can see that while using this univariate model we got an overall accuracy of 60% compared to overall accuracy of 54% from Fourier model and also pretty good f1-score of 70% for wildfire. ALso from the figure it was clear that ARIMA model forecasting was way better than Fourier regressive model which was able to forecast even quick ups and downs but there were still features which ARIMA was also struggling. And when we attempted to try the transfer learning with its residual-based anomaly, it was clear that the model trained on NSW which applied to other regions like NT, WA, and SA showed poor performance, often overpredicting or underpredicting wildfires depending on local weather information. For instance, in NT, it predicted nearly all days as wildfire due to unfamiliar residual distributions, while in QL, it failed to flag enough wildfire days. These patterns shows high limitation of residual based anomaly detection, meaning that the models are highly region sensitive and thresholds learned from one area may not generalize across other diverse weather conditions of other regions. Therefore, this method is computationally light and easy to interpret but it lacks robustness for broad deployment unless localized changes is done. This shows transferability of unsupervised models is possible in certain conditions like the regions with similar residuals like NSW, but can't be considered as a general residual threshold.

Bringing these findings together, We can conclude that Unsupervised learning methods can detect wildfire-relevant anomalies using weather forecast data even without access to labeled fire events but their effectiveness is highly dependent on data characteristics. Multivariate models that capture sequential dependencies like LSTM autoencoders require seasonality to learn effective baselines and Univariate methods like Fourier and ARIMA offer simplicity and interpretability but struggle in local variance or structural shifts across regions in general.

While this study has shown the potential of unsupervised learning methods to detect weather anomalies that can cause wildfires, there are several meaningful directions in which this research can be extended for more effective results. One of the most important observations from our results is that seasonal patterns were very important in anomaly detection, particularly with deep learning models like LSTM autoencoders. Future work could explore hybrid models that preserve seasonal structure while learning residual patterns more robustly and also combine both seasonal and non-seasonal components explicitly during training and also if we manage to combine ARIMA and Fourier models for selective features where taking features with irregular spikes which lacks smooth trends for ARIMA models and fourier model which is good for smooth features when combined can give better prediction and good results. Wildfire behavior is often influenced by regional climate trends, So future studies could incorporate spatial relationships using graph-based models or spatiotemporal architectures, which may improve the robustness and generalization of anomaly detection across multiple regions. These enhancements altogether can offer a practical path forward for building promising hybrid models into reliable tools for detecting wildfire risks in the future.

References

[1] John T. AbatzoglouA **and** Jennifer K. BalchB Bethany A. BradleyC Crystal A. KoldenD. *Human related ignitions concurrent with high winds promote large wildfires across the USA*. 2018. URL: http://www.pyrogeographer.com/uploads/1/6/4/8/16481944/abatzoglou_etal_2018_ijwf.pdf.

[2] Fourier analysis. *Fourier analyis and trignometric functions*. 2024. URL: https://en.wikipedia.org/wiki/Fourier_analysis.

[3] California Fire report by ca.gov. *California Wildfire Incidents*. **june** 2024. URL: https://www.fire.ca.gov/incidents/.

[4] Miguel Arana Catania1, Alexander Farr2 **and** Ivan Petrunin. *Unsupervised Anomaly Detection in Wildfire Prediction*. 2024. URL: https://arxiv.org/pdf/2411.09844.

[5] *ERA5 the fifth generation ECMWF reanalysis for the global climate and weather for the past 8 decades*. URL: https://cds.climate.copernicus.eu/datasets/reanalysis-era5-single-levels?tab=overview.

[6] *Geospatial Analytics IBM PAIRS Geoscope IBM PAIRS*. URL: https://www.ibm.com/docs/en/environmental-intel-suite?topic=components-geospatial-analytics.

[7] Üstek İrem **andothers**. *Deep Autoencoders for Anomaly Detection in Wildfire Prediction*. 2024. DOI: 10.1029/2024EA003997.

[8] Sanaz Moghim & Majid Mehrabi. *Wildfire assessment using machine learning algorithms in different regions*. 2024. URL: http://www.pyrogeographer.com/uploads/1/6/4/8/16481944/abatzoglou_etal_2018_ijwf.pdf.

[9] Mahsa Salehi **andothers**. *Dynamic and robust wildfire risk prediction system: An unsupervised approach*. 2015. URL: https://research.ibm.com/publications/dynamic-and-robust-wildfire-risk-prediction-system-an-unsupervised-approach.

[10] Piyush Jain piyush Sean C.P. Coogan Sriram Ganapathi Subramanian Mark Crowley Steve Taylor **and** Mike D. Flannigan. *A review of machine learning applications in wildfire science and management*. 2020. URL: https://cdnsciencepub.com/doi/10.1139/er-2020-0019.

[11] Irem Ustek1 **andothers**. *Deep Autoencoders for Unsupervised Anomaly Detection in Wildfire Prediction*. 2024. URL: https://arxiv.org/pdf/2411.09844.

[12] *vegetation index image*. URL: https://en.wikipedia.org/wiki/Vegetation_index.

[13] Lawrence Wong **andothers**. *Auto-Encoder with Regression for Time Series Anomaly Detection*. 2022. URL: https://arxiv.org/pdf/2212.13558.

[14] ARIMA working. *Autoregressive integrated moving average*. 2024. URL: https://en.wikipedia.org/wiki/Autoregressive_integrated_moving_average.

[15] LSTM working. *Long short-term memory*. 2024. URL: https://en.wikipedia.org/wiki/Long_short-term_memory.

[16] *zvegetation dataset Australia*. URL: https://github.com/Call-for-Code/Spot-Challenge-Wildfires/tree/main/data.

[17] *zweather dataset Australia*. URL: https://github.com/Call-for-Code/Spot-Challenge-Wildfires/tree/main/data.

[18] *zwildfire dataset Australia*. URL: https://github.com/Call-for-Code/Spot-Challenge-Wildfires/tree/main/data.

Acknowledgements

I would like to express my deepest gratitude to my supervisor, Prof. Hao Wang, for his invaluable guidance, patience, and continuous support throughout this research. His insightful feedback and encouragement have played a crucial role in shaping both the technical and academic quality of this thesis. I am equally thankful to Prof. Zhaochun Ren for his thoughtful suggestions and kind support. I also wish to extend my sincere thanks to my family and friends, whose unwavering belief in me gave me the strength to persevere. This work would not have been possible without their collective support and kindness.