6/21/2021

Raghav Saxena A00279923

Athlone institute of technology

A comparative study of state-of-the art ensembling and general-purpose deep learning algorithms for sun spot detection.

Thesis Draft



Table of Contents

[Acknowledgement: 3](#_Toc80043848)

[Declaration: 3](#_Toc80043849)

[Abstract: 4](#_Toc80043850)

[Chapter 1 - Introduction: 5](#_Toc80043851)

[1.2 Research Problem 6](#_Toc80043852)

[1.3 Scope 6](#_Toc80043853)

[1.4 Limitations 6](#_Toc80043854)

[1.5 Dissertation Roadmap: 7](#_Toc80043855)

[Chapter 2 - Literature Review: 8](#_Toc80043856)

[Chapter 3 – Tools and Data Methodology: 13](#_Toc80043857)

[Chapter 4 – Data Analysis and Model Results: 29](#_Toc80043858)

[4.1 Exploratory Data Analysis (EDA): 29](#_Toc80043859)

[4.2 Model Results: 33](#_Toc80043860)

[4.3 Feature Importance: 36](#_Toc80043861)

[Chapter 5 – Discussions: 41](#_Toc80043862)

[5.1 Data Overview: 41](#_Toc80043863)

[5.2 Model Results: 41](#_Toc80043864)

[Chapter 6 – Conclusion: 43](#_Toc80043865)

[References 44](#_Toc80043866)

**List of Tables:**

[Table 3. 1: Column descriptions for the solar flare dataset from the NGDC catalogue. 15](#_Toc80043708)

[Table 3. 2: Column descriptions for the sunspot dataset from the NGDC catalogue. 16](#_Toc80043709)

[Table 3. 3: Description of columns of the final dataset achieved after processing. 19](#_Toc80043710)

[Table 4.2 1: Hyperparameters used for the random forest model. 34](#_Toc80043785)

[Table 4.2 2: Hyperparameters used for the gradient boosted trees model. 34](#_Toc80043786)

[Table 4.2 3: Hyperparameters used for the deep learning model. 34](#_Toc80043787)

[Table 4.2 4: Performance metrics of the different machine learning algorithms in RapidMiner. 35](#_Toc80043788)

[Table 4.2 5: Performance metrics of the different machine learning algorithms in Python. 35](#_Toc80043789)

[Table 4.2 6: Performance metrics of different machine learning algorithms in h2o.ai. 36](#_Toc80043790)

## Acknowledgement:

First and foremast I would like to thank my dissertation supervisor Prof. Terri Hoare of MSc in Data Analytics course at the Dublin Business School. It was through the constant support and guidance of Prof. Terri Hoare that I was able to complete this dissertation. It was through her suggestions to incorporate new technologies, which I was previously not aware about, that improved the results that were expected from this research. She made sure that the entire project was my own work and gave insights on how it could be improved.

Second, I would like to thank Dr. Mark Daly lecturer at Athlone Institute of Technology (AIT), who helped me find a suitable project for my dissertation when I reached out to him with my area of interest. He was kind enough to provide me with the topics and resources to start with my research and was available to address any queries while I tried to gain an understanding of what the particular research demanded.

Finally, I would like to thank Jonny O'Dwyer of Athlone Institute of Technology for his well taught lectures that helped to understand how to carry out the research in a proper and planned way.

Thank You,

Raghav.

# Declaration:

I have read the Institute’s code of practice on plagiarism. I hereby certify this material, which I now submit for assessment on the programme of study leading to the award of (Master of Science in Data Analytics ) is entirely my own work and has not been taken from the work of others, only to the extent that such work has been cited within the text of my work.

Student ID Number: A00279923

Name of Candidate: Raghav Saxena

Signature of Candidate:

Date: 25th August 2021

# Abstract:

In this study we try to compare the performance of various ensemble based and deep learning models on their ability to classify sunspots as flaring or non-flaring sunspots with those of traditional machine learning methods. A sunspot is dark region on the surface of the sun which appears dark because it is cooler than the rest of the surface due to the presence of high magnetic field within that region. A solar flare is a sudden and short-lived burst of energy from these spots which can have major effects on the space weather and can also cause major effects on earth. The study explores two datasets, sunspots and solar flares which were gathered from the publicly available catalogue from the National Geophysical Data Center (NGDC). The data corresponding to C, M and X class flares is considered in the study. Morphological and magnetic properties of the sunspots are used as input. The output is a binary yes no classification on whether or not a flare will be produced or not. The research is done across three different platforms, each having its own advantages over the others. The performance of these models is compared on a number of machine learning based evaluation metrices. A basic understanding of how much each of the predictor in the dataset contributes towards the final prediction is also calculates using two metrices. These metrices helps to get a understanding of the global and local contribution of the predictors for a particular model. The use of hyper parameter optimization was done to further improve the performance of the models under study. With this research it was proven that the ensemble machine learning methods did a better job at the classification problem compared to the deep-learning and traditional machine learning methods. Even though the margin by which the ensemble methods performed better is small it is still significant when it comes to predicting solar flares. The two ensemble models, gradient boosted trees and random forests, used in this study were able to create a degree of separation between the flaring and non-flaring classes with a AUC score of 0.88 and 0.84 each.

# Chapter 1 - Introduction:

Space weather is the term given to the changing weather conditions in the space as the result of the activity on the surface of the sun. The space weather affects the entire solar system and can have adverse effects on earth – damaging electrical grids and satellites. Earth is surrounded by strong magnetic field that’s protects us from the harmful radiation that comes from the changing conditions from the surface of the sun. However, even with the protective environment around us the radiations can still have adverse effects to instruments and personnel in space and larger phenomenon can even affect the ground-based equipment's. With the advancement in science and technology it is becoming more and more important to understand space weather and its effects on earth. The disruptions that emerge from the surface of the sun have the potential to have adverse effects on Earth. It can disrupt electrical grids, railway signals, wireless communications, space-based systems and human life in space. Some of the conditions even have the potential to cause devastating and can cost countries billions of dollars.

The sun is the centre of our solar system and it bounds all the objects and planets around it by its gravitational force. It comprises almost 99% of mass of the entire solar system and is the largest object in it. Th sun is a huge ball of electrically-charged hot gas that capable of generating powerful magnetic fields. Space weather is a consequence of the changing magnetic field on the sun. This change in magnetic field goes through a cycle of low activity to high activity every 11 years which is called the solar cycle. At the end of every solar cycle the magnetic poles of the sun completely flip, that is the south pole becomes the north pole and the north pole becomes the south pole. During a solar cycle the sun goes through a period of low activity called the solar minimum to a period of high activity called the solar maximum and as this magnetic fields in the sun change so does the activity on the sun.

During this solar cycle, the sun develops regions of high magnetic field called the sunspots or the active regions (AR). These spots generally appear dark on the surface of the sun because they are cooler than the rest of the sun. They may vary in shape, size and their magnetic polarity. The magnetic fields inside a sunspot can tangle, reorganize or cross over each other resulting in violent explosions called the solar flares. A solar flare is a sudden release of energy, light and radiation from the surface of sun that could last up to a few minutes. On the basis of their X-ray wavelength the solar flares are classified into 5 categories: A, B, C, M and X, in increasing order of their intensity. Out if these A and B class flares are negligible and have are of no consequence when studying the effects of flares on Earth, the X-class flares are big flares and the potential to cause brief radio blackouts on earth and long-lasting ration storms, M-class flares are medium class flares and cause minor radiation storms along with brief radio blackouts, C-class flares in are the smallest types when compared against M- and X- class they have few noticeable consequences on earth.

The need to predict solar flares arises from the fact that these can directly affect the communication systems, ionosphere and the personnel and technology in space. A lot of work has been done to understand the origin of solar flares. The works of (MclNTOSH, 1990), which define the morphological properties of the sunspots, and the Mt. Wilson magnetic classification, which define the magnetic properties of a sunspot group are one of the early works in understanding what factors affect the eruptions of solar flares from a active region (AR). While a lot of work has been done on predicting the solar flares from a AR's magnetic or morphological properties only a few have used both the parameters. This study focuses on the use of both the properties, magnetic and morphological, of a sunspot to train a number of models that can create a degree of separation between a flaring and a non-flaring sunspot. These models are compared on a number of parameters and a analysis is done to evaluate how much each of the predictor contributes towards the final prediction.

## 1.2 Research Problem

A large solar flare can be very disastrous and lead to damages worth billions of dollars. Although solar flare has been studied a lot and different parameters and methods have been used to forecast them, none of them can be considered as a perfect model for the flares because it is not clearly know what conditions combined together lead to the eruption of a solar flare from a sunspot.

**Research Question:** Comparison of deep-learning and ensembling techniques to classify which sunspot can produce a solar flare.

**Aim:** The aim of this research is to compare the performance of different ensemble and deep learning-based algorithms with traditional machine learning algorithms at classifying the research data into flaring and non-flaring classes and checking which model does the best job using all the predictors in that are there in the research data.

**Objective:** The objective of the research is to check the performance of a number of models like random forests, gradient boosted trees, neural nets by comparing their accuracy at segregating the two classes in the research data and also drawing a performance comparison of these models with the traditional machine learning models like logistic regression, svm, etc.

## 1.3 Scope

The scope of the research is to do a comparative study of ensemble and deep learning machine learning models in order to find out a model that will do the best job at classifying sunspots into flaring and non-flaring groups. It also focuses on finding factors that contribute towards the final predictions.

## 1.4 Limitations

This research is restricted by a number of limitations, some of them are data related while the others are mainly due the hardware:

* Solar flares happen in different classes and while some classes of flares happen quite often their data is available abundantly and is of little of help towards the research and the flares that are big are often rare and the data on these is very less. Therefore, the research is limited by the data it has on very large flares.
* Data Processing: The data used is downloaded from two different catalogues and they are compared and combined into one. Since the data is very vast comparing and processing them is limited by the kind of hardware that is available.
* The hardware available to conduct is research is very much limited by the speed at which it can carry out model building and evaluation tasks.

## 1.5 Dissertation Roadmap:

This dissertation is organized as follows, Chapter 1 – introduction introduces the research problem and puts forwards the research question, aim and the objective of the research.

Chapter 2 is the literature review which focuses on summarising the works that have been done with respect to this field and draws a conclusion on all the research discussed.

Chapter 3 discusses upon the tools and methodology that has been used to complete this research. The data, algorithms and evaluation metrices that are used in this research are also discussed in this section.

Chapter 4 is the data analysis and model results section. In this section data visualisations are used to discuss upon the data, the second part of this section focuses on discussing upon the results of the various algorithms implemented on different platforms. The last part of this section gives and understanding to how much weight the different predictors have towards the final prediction.

Chapter 5 is the discussion section. It discusses upon the results of the various algorithms and draws upon a comparison between them.

Chapter 6 which is the conclusion chapter. This chapter draws discusses the entire research and draws upon a final conclusion for the dissertation.

# Chapter 2 - Literature Review:

Many researchers have used different approaches to figure out an effective solar flare prediction model. The following research journals were referred in order to understand the concept behind developing a model.

In his work (Colak and Qahwaji, 2007) tries to "investigate the degree of correlation between sunspots and the occurrence of solar flares" using the data from the National Geophysical Data Center (NGDC). The NGDC dataset classifies sunspots using two different systems: McIntosh and Mt. Wilson. The McIntosh classifies based on the shape, density, size (MclNTOSH, 1990), Mt. Wilson classifies based on magnetic polarities. Colak used this data to train a feed-forward backpropagation neural net. The model takes in 3 values of McIntosh classification along with an average sunspot group number and predicts binary classification of whether or not a particular sunspot generates a solar flare and a second target value to determine the class of flare (M or X). The study was able to demonstrate that there is a clear association between flares and specific McIntosh groups of sunspots and that an NN-based system is efficient at predicting solar flares with an accuracy of 91.7%

While most of the work up until now has been based on the morphological properties of ARs or the current characteristics of the magnetic field, none of them focus on the properties leading to the eruption of a solar flare. (Yu et al., 2009) build a short-term solar forecast model that focuses on the evolutionary properties of the magnetic field inside an AR. In their research, they focused on three main predictors – maximum horizontal gradient, length of neutral line, number of singular points – from the SOHO/MDI magnetograms. The influence of the evolutionary information on flare level is analysed using autocorrelation functions, then the sequential machine learning problem is converted to a standard supervised machine learning problem, and the AR's are classified into flaring Class 1 and non-flaring Class2 regions using decision tree classifiers and learning vector quantization (LVQ) machine learning algorithms. The results showed an evolutionary influence on the flare by the 3-day magnetic field information before an eruption. The accuracy of the model was calculates using the True Positive (TP) rate (74.88%).

(Qahwaji and Colak, 2007) presented a short-term machine learning based flare prediction model. They used the sunspot and solar flare data from the NGDC data centre. They used the data from the two datasets to train three different machine learning models – Cascade Correlation Neural Networks (CCNs), SVMs and Radial Basis Function Networks (RBFN). The initial dataset was created by associating flares with their corresponding sunspots by checking if their NOAA number matches and weather the time, they were recorded was within 6 hours of each other. From this data only the McIntosh Classification and the NOAA number attributes were selected. The McIntosh groups were further converted into their numerical values as machine learning models work better with numerical data, further a mathematical model was used to simulate the behaviour of solar cycle. The CCNN, RBFN and SVM models are compared on a number of parameters like training time, topology, accuracy etc, in this research they found SVM was doing a better job at weather a sunspot group will flare based on its McIntosh Classification.

(Yi et al., 2020) used to two deep learning models to predict the solar falres using the X-ray flux profiles of the flares. For this research he used the data form the NOAA GOES x-ray catalogue from the years August 1998 to May 2006, which contained data in 845 major solar flares events. This data set was split into training and testing sets where training set contained 90% of the events whereas the testing set contained 10% of the events. He used this data to train two different types Long Short Term Memory Loss (LSTM) models: namely the seq2seq and seq2seq with attenuation. The model gave a 30-minute forecast of solar flares during their rise phase with a 1 minute time cadence. The models were evaluated using the 10-fold cross validation and rms error (RMSE), for this three different RMSE values were used, one for all forecasting results, one for peak results, and one for forecast time. The performance of these models was further compared with the performance of traditional deep learning and regression models. The models used for comparing results are MLP, LSTM model, Auto-Regressive Integrated Moving Average (ARIMA), KNN and SVM. The results of the research proved that the seq2seq and seq2seq with attenuation do a better job compared to rest of the models to predict the X-ray flux. The models proved to be efficient at predicting flux for weaker flares compared to stronger flares.

(Li and Zhu, 2013) while most of the works till now were focused on sunspot characterization and flare occurrence, none of them have taken into the account the evolutionary process of an active region. Li built a learning network quantization network and multi-layered perceptron using the sequential data of a sunspot. This data included the evolutionary features of an active region over a three-day cycle. Other attributes like the: magnetic classification, radio flux, Mcintosh classification were taken and converted into numerical forms. This model was built to forecast flares within 48h window. Using a sliding window technique time-series data was added to the parameters of the sunspot. For this research he used the flare catalogue and the sunspot data of the solar cycle 23 from the NGDC's GOES satellites. The data used contained 21655 samples, to these samples two pre-processing steps were done, one: some variables were computed using the flare productivity equation; two: the data was normalized to add the sequential data to it; the final data that was achieved was used to build a classification model that tells the importance of each solar flare class within the next 48h frame. To train this model the dataset was divided into ten folds were 9 were used for training and one was used for testing. True-positive rate, True-negative rate and accuracy were taken in as a parameter to check the performance of the model. The results proved that the evolutionary properties of the sunspots do provide better forecasting results compared to the current properties of the sunspots. The models that were trained on evolutionary data were able to give a higher accuracy of 82.3% for MLP and 84.25% for LVQ compared to the ones that were trained on the current information that gave am accuracy of 77.77% for MLP and 82.80% for LVQ.

(Song et al., 2009) used the data from SOHO MDI magnetograms to develop a logistic regression model that estimates the probability of active regions to produce C, M or X class flare with a 24hr cadence. He used three parameters – unsigned magnetic flux, length of gradient neutral line and magnetic dissipation. He used the data from the years 1995 to 2005 to gather a total of 230 solar events the data gather was turned into nominal forms. They were able to prove that all the three predictive parameters showed positive correlation and length of the gradient line was the most significant at predicting solar flares.

(Wang et al., 2008) quantitively studied the evolutionary magnetic field inside a sunspot. Although a number of parameters like: length of neutral line, shear angle, horizontal gradient, etc. have been previously used to map the magnetic field, Wang in his research used only three parameters - number of singular points, maximum horizontal gradient and length of neutral line. The magnetic field data for this research was extracted from SOHO/MDI longitudinal magnetograms and X-Ray flares data from the GOES data repository for the years April 1996 to December 2001 was taken. A threshold value of 10, that corresponds to a solar flare of M1.0 equivalent was selected to classify these flares with a looking period of 48h. A backpropagation artificial neural net was trained on this data and the data from Jan 2002 to Dec 2002 was used as a test set for this research. The results showed a ratio of 69% - 31% for correct to incorrect forecasts for the next 48hrs.

(Falco et al., 2019) developed a solar flare forecasting model based on properties of sunspot groups like number of sunspots and pores, area, importance between density of sunspot and leading spot, Zurich class, type of penumbra and a statistical technique. For this research the data from the Equatorial Spar of INAF-Cantania Astrophysical Observatory (INAF-OACt) was used, this observatory stored data about the morphological properties of sunspots on a 24h cadence. In his research he assumed that the flaring capability of sunspot groups follows a poison distribution. Based on this they computed a flare rate for each of the above-mentioned parameters. To evaluate the performance of the model the predictions were compared with the records obtained from the GOES satellites, probability distributions for the observations and forecasts were calculated and compared against a skill score. Using the characterization of sunspots groups and statistical measures, Falco was able to get the more accurate forecasts for some the stronger flares.

A method to automatically forecast solar flares was presented by (Yuan et al., 2010) in this research paper. This method is trained on the magnetograms from the MDI instrument on-board the SOHO satellite, which sends real-time data for space weather predictions. Three variables are extracted from this data: magnetic polarity inversion line length, total magnetic energy, unsigned magnetic flux. The model gives an a yes or no occurrence for a certain type of flare within the next 24hour from the moment the data is recorded by the SOHO observatory. Yuan's works builds up on the work of (Song et al., 2009), hence the statistical variables that were used by Song were also used on this study. The model builds up into two steps: firstly, using a logistic regression model the three magnetic predictors are used to map the active region data into four probabilities, namely: level 0 for regions that produce only A or B class flares, level 1 for C-class, level 2 for M-class, and level 3 for X class flares; second: a SVM model is used to map the probabilities into binary labels. 230 samples were extracted from the solar cycle 23, corresponding to the years 1996 to 2005 and leave-one-out cross-validation was used to access the performance of the model. For each test 229 were used for training and 1 was used for testing the model, the same procedure was repeated for 230 times each time using a different subset for training and testing. The experimental results shows that the method introduced in this paper outperforms the SVM-based method on level one and level three flares but does marginally less well for level two flares.

(Qu et al., 2003) did a comparative study of three different models, namely: support vector machines (SVM), multi layered perceptron's (MLP), radial basis functions (RBF), to build an automatic flare detection model using the hydrogen-alpha images data from the Big Bear Solar Observatory (BBSO). The BBSO captures images in 2032 X 2032-pixel format, these images were then labelled as flare and non-flare and later nine features were extracted from these images which were separately put into training and testing sets. In this study Qu tried to build a model that could catch the scalability and complexity of solar flares. The nine features that were extracted from the images are: frame mean brightness, brightness standard deviation, the difference in the mean brightness between consecutive images, key pixel's absolute mean brightness, key pixel radial position, in a 7X7 the contrast between key pixel and its minimum neighbour, in a 50x50 window: mean brightness, standard deviation, difference of brightness between previous and next image. Out of the three models that were trained SVM performed the best with a accuracy rate of 96.7% and a very low testing and training speed, followed by a RBF with a accuracy of 95% and finally the MLP with a accuracy of 94.2%.

(Li et al., n.d.) used a method that combined the predictive powers of both the SVM and K-nearest neighbour (KNN) called the SVM-KNN model to build a solar flare forecasting model. SVM model give a low classification accuracy when the model is built on complex data, in order to improve this accuracy, the SVM model was combined with a KNN model. For this research the sunspot region data from NOAA was taken from the years 1996 to 2004. Magnetic classification, sunspot area, McIntosh classification and radio flux data was used as input to train the SVM-KNN model. The output of the algorithm gives us a importance of solar flare class for the next 48 hours, there are two outcomes: larger than or equal to M, or lower than M. The performance of the SVM-KNN model was compared to the performance of two other models namely the SVM and the LVQ model. While the SVM model did a very good job at predicting M class flares its performance was increased when it was combined with KNN model. The SVM-KNN model performed the best out of the tree models at predicting the M class flares over the next 2 days period.

(Benvenuto et al., 2018) The aim of this paper was to provide a novel flare forecasting method that can combine the prediction accuracy while being able to identify relevant variables. In this method the outputs of a supervised learning model, LASSO, are applied to a unsupervised fuzzy learning technique called the Fuzzy C-means. In this hybrid model, firstly the regularization constructs weights on how much each feature will contribute to the prediction, while the second part of the model, classification, an automatic thresholding of the regression outcomes is done. The method described has several advantages over other one-step approaches, the regularization helps select the significant features while the classification produces a flexible and data-adaptive partition between the variables. For this research the data from the NOAA Space Weather Prediction Center (SWPC) from the dates August 1996 to December 2010 is taken for test set which contains 22222 events and the data between December 1998 to June 1966 is taken as train data which contains 17600 events, all these events correspond to C and M class flares. The performance of the hybrid model is compared against standard machine learning models like the SVM, MLP, K-Means on various accuracy metrices like sensitivity, accuracy, false positive ratios. The regularization – classification model performs better than all others models in predicting the flares that are greater than C1 class flares. It also gives better sensitivity and FAR and its accuracy is comparable to all the other models. For flares that are greater than M1 class, the proposed model proves to be the most effective one.

(MclNTOSH, 1990) The sunspots over the surface of the sun can be observed easily even by a small telescope. Most of the data that is observed is reduced to a classification system that is correlated with solar flares. The Zurich Classification is one such classification system that correlates solar flares to a classification scheme for the data that is observed and recorded, but in this system of classification the probability of a flare happening is too low even in its most active class. Therefore, this system of classification was modified to improve the definitions and to add descriptors for stability, complexity and size. The general form for a McIntosh class is Zpc, where Z refers to the Zurich class which are defined on the basis weather penumbra is present, its distribution and the the length of sunspot group, p refers to the type of principal spot – it defines the size, stability, maturity and complexity and lastly c which refers to the relative spottedness inside a sunspot group and gives additional information about the area, polarity in a group. These three classes together combine to define 60 different types of sunspot groups and helps provide better correlation with solar flares.

Knowing how machine learning arrived at a certain prediction can help empower better models in a number of ways. Because of this interpretability of models has become of paramount interest. (Ribeiro et al., 2016) in his paper discussed the limitations of model-agnostic approaches and then introduces how Local Interpretable Model-agnostic Explanations (LIME) handle some of the issues of model agnostics. Model agnostics is the process of treating the model as a black box and extracting post-hoc explanations form it. Restricting the models to be interpretable results in them losing accuracy, usability and flexibility and therefore the model agnostics approach was introduced. This approach solves a number of problems of interpretable models like: explanation, representation, switch cost, comparison by treating the models as black box thus allowing them to have a trade-off between interpretability and different constraints. However, there are certain drawbacks to the model-agnostics approach like getting global understanding of models, their exact explanations and actionability of models. These problems are tackled in lime, which is an approach that explains the predictions of a model by learning the model locally around a certain prediction. Even though interpretable model are able to provide insights about how a certain model arrived at a prediction they impose restrictions on models thus the model agnostics approach is used to overcome these limitations in the models.

Most of the researches summarised in the literature review are either based on the McIntosh classification or Mt. Wilson Magnetic classification of sunspots. Previous researches have used the data from different repositories, like NOAA, SOHO/MDI, SDO to build and compare the performances of various machine learning algorithms. Most of the researches showed that models that are able to handle complex data very well, like: neural nets, random forests, SVM, etc., are the ones that gave a higher accuracy score compared to other machine learning algorithms. While the work of most of the researchers is focused on using either the Morphological or the magnetic data, none of them has used both the data to build classification models, also most of the research's models like neural nets, SVM, etc. only a few have used random forests to classify the problem. Based on the understanding of the previous works, ensemble methods like gradient boosted trees, random forests and deep-learning models like neural nets will be used to classify the problem using both the morphological and magnetic data.

# Chapter 3 – Tools and Data Methodology:

Machine learning tools are software's and packages that are built to make machine learning easy for all. These are powerful tools that help to do complex machine learning tasks like visualisation, modelling, evaluation, etc. in a simple and straightforward way. Due to their ability to perform complex tasks easily, three machine learning tools and platforms will be used in this research:

1. **RapidMiner:**

RapidMiner Studio is an easy-to-use data science software platform. It provides a environment for predictive modelling, data analytics, machine learning, deep learning etc. RapidMiner has the tools and user interface to provide a comprehensive data science experience from data prep to model deployment. With the RapidMiner Studio's interactive and easy to use user interface it becomes very easy to follow the life-cycle phases of a data science project that are defined by the CRISP-DM methodology.

The main function of the RapidMiner studio that will be used in this research is its auto model feature. The auto model features automatically trains a number of algorithms on the data it is fed; it also calculates the performance of these algorithms using a number of performances metrices. This feature will be used to get an initial intuition at how the various machine learning and deep-learning algorithms will perform at finding a relationship between the dependent and independent variables. These results will be then used as a benchmark to build upon. Other models that will be built using python and H2O will try to improve the scores that will be given by the RapidMiner studio.

1. **H20.ai:**

H2O is an in-memory, fast, distributed, scalable and open-source predictive modelling and machine learning platform that allows to scalable and distributed machine learning models (H2O.ai, 2016). It helps to automate some of the most difficult machine learning and data science processes like visualizations, validation, model tuning, selection and deployment. The aim of the H2O model is to attain the highest accuracy for complex and simple machine learning algorithms in the minimum amount of time. It also offers automatic visualisation tools and machine learning interpretability (MLI). H2O uses in-memory compression techniques to effectively handle big data-sets. H2O supports the seamless implementation of all the major deep- learning and machine learning algorithms like linear regression, logistic regression, principal component analysis (PCA), gradient boosting, random forests etc. It also implements some of the best-in-class deep learning algorithms. H2O's auto machine learning feature (Auto ML) allows non-engineers to implement scalable and effective machine learning algorithms without having much knowledge of the processes and procedures of data science, it makes it easier for them to implement a whole data science project easily. With H2O it becomes easier to handle complex model building processes like hyperparameter tuning, feature generation etc.

This project makes use of python to connect to the H2O environment. H20 uses REST API to connect with python. Since the size of the dataset being used for this research is not very large, the local system will be used to run an instance of H2O. This instance of the H2O will be used to run the common machine learning algorithms along with the state-of-the-art ensembling and some general purpose deep learning algorithm like neural nets. The result generated from H2O will be compared to the one's that will be generated using the scikit-learn package of python.

A methodological approach may refer to way in which a researcher intends to carry out his/her research from the specified or known methods in a discipline. This gives a framework of how the research should be carried out gives a detailed summary of each of the phases of the research. It gives the readers of the research on how the research was carried and what was done in the research. It may include a number phases of like how the data was collected, how it was analysed, what was the research problem, etc. This study makes use of the Cross Industry Standard Process for Data-Mining methodology also known as CIRSP-DM methodology. It is a standard process model that aims at describing the various approaches taken by data mining experts. It gives out guidelines on how to organize, plan and execute data science projects.

1. **Python:**

Python is a general purpose high level programming language. Python is an open source programming language that offers numerous libraries and inbuilt functions to create and implement data visualisations and machine learning models.

For this research, pythons scikit learn library will be used to implement machine learning models. The greater

**CRISP-DM:**

Cross Industry Standard Process for Data-Mining is a methodological approach that defined the life-cycle of a Data Science project from its initialisation till its end. This process includes 6 phases that layout the flow of a data science project. It also includes a description of each phase of a data mining project and the tasks related to each phase. CRISP-DM provides a flexible and customizable methodology involving 6 phases, many of the activities can be completed in any sequence, and it will frequently be necessary to go back and repeat some procedures. The 6 phases of CRISP-DM methodology are :

* Business Understanding
* Data Understanding
* Data Preparation
* Modelling
* Evaluation
* Deployment



Figure 3.1: CRISP - DM Methodology

1. **Business Understanding:**

Business understanding is the very first phase in the CRISP-DM methodology which aims at understanding what needs to accomplished from the research. In this stage the factors that can affect the outcome of a project are discussed upon and the requirements and objectives are defined in a business perspective. By the end of this stage the goals are understood and what will be required of the project is also stated.

A solar flare is a huge burst of light, energy and radiation which is caused by the entangling of magnetic field lines in the regions of high magnetic field concentration called the sunspots. These flares after eruption effect the space weather in the solar system and when they reach earth they can interfere with the earth's magnetic fields, electric guilds, satellites and personnel in space and can also cause radio blackouts, which can cost countries billions of dollars to fix. Therefore, a model can accurately predict the emergence of solar flares before they happen is essential to minimize such costs. All the models that have been developed so far are not very effective as there is not much information on what are all the factors that affect the eruptions of solar flares from a sunspot. This research tries to map the various morphological and magnetic properties of sunspots in order to develop a model that can accurately separate a flaring sunspot from a non-flaring sunspot. This research also aims to find a best model for this research by testing training the data on a number of different models on three different platforms. Two different data catalogues are used for this research - the sunspots and the solar flares dataset from the NGDC open source dataset.

1. **Data Understanding:**

In this phase the data that is collected for the project is explored. This data is explored using visualisations and tables to gain more insights and to identify the quality of the data. This is a very important phase in the methodology as it helps to get the feel of the data and also gives an early opportunity to check if the data collected is indeed the one that will help with the research.

For this research the data has been collected from the NOAA's GOES data centre about two separate events that happen on the surface if the sun: solar flares catalogue, which contains information about the various solar flares that have happened, and sunspots catalogue, which catalogues information on sunspots groups that have appeared on the surface of the sun, from the period ranging from the beginning of 1996 to the end of 2016. The main columns in each of the dataset are described below:

**Solar Flare dataset:**

|  |  |  |
| --- | --- | --- |
| Column | Description | Data Type |
| Date | The date the solar flare was recorded. | Date Time |
| Start Time | Start time of the flare. | Date Time |
| End Time | The time at which the flare reaches its peak intensity. | Date Time |
| Max Time | The duration till which the flare lasted. | Date Time |
| Location | Longitude and Latitude of the sunspot from which flare originated. | String |
| X-Ray Class | The class of X-Ray (A, B, M, X). | String |
| X-Ray Intensity | The total intensity. | Number |
| NOAA Number | NOAA Number of the sunspot from which the flare originated. | Number |
| Integrated Flux | The total integrated flux of the flare that was observed. | Number |
| Total Region Area |  |  |
| Total Intensity |  |  |

Table 3. : Column descriptions for the solar flare dataset from the NGDC catalogue.

Since the research only focuses on a binary classification of the sunspots into flaring and non-flaring groups a lot of information on time and date becomes irrelevant towards the final result. But this data will still be kept in order to develop upon it in the future works.

**Sunspots Dataset:**

|  |  |  |
| --- | --- | --- |
| Column | Description | Data Type |
| Date | The date the spot was observed. | Date Time |
| Time of Observation | Time of observation | Time |
| Location | Location in Latitude and Longitude. | String |
| Mt. Wilsom Magnetic Classification | Magnetic polarity classification of sunspot. | Object |
| NOAA Number | NOAA Number of the class. | Number |
| Zurich Class | Describes the evolutionary sequence of spots. | Object |
| Penumbra Class | Describes the size, maturity, stability of the spot. | Object |
| Compactness Class | Describes the information about area and spottedness of the spots. | Object |
| Number of Spots | Total number of spots in a sunspot. | Number |
| Longitudinal Extent |  |  |

Table 3. : Column descriptions for the sunspot dataset from the NGDC catalogue.

The categorical columns (Mt. Wilson Magnetic Classification, Zurich class, Compactness Class, Penumbra Class) have different classes wherein each class defines a certain characteristic of the sunspot groups. To be able to understand the project and know the behaviour of these models with respect to correlation the final target will have with these columns, it is essential to first gain a understanding of what each class will signify.

1. Mt. Wilson Magnetic Classification:

The magnetic classification contains classes that describe how the magnetic field is distributed within a sunspot. There are a total of eight classes (alpha, beta, gamma, beta-gamma, delta, beta-delta, beta-gamma-delta, gamma-delta) that define the magnetic characteristic of a sunspot group but the data that have been gathered for the research contains only four of the classes, these four classes are:

**Alpha (A):** unipolar sunspot group.

**Beta (B):** A bipolar group with a simple separation between the polarities.

**Gamma (G):** Complex group with positive and negative polarities irregularly distributed making it difficult to classify it as a bipolar group.

**Beta-Gamma (B-G):** Bipolar sunspot group with complexity enough to make it impossible to distinguish between positive and negative polarities.

1. Zurich class:

The Zurich classification makes one part of the McIntosh classification of sunspots groups. The Zurich classes are defined on the basis of weather a penumbra is present or not. Zurich classification can take up to seven (A, B, C, D, E, F, H) different classes depending on the size and complexity of the sunspot group.

**A:** It is the simplest class and refers to a unipolar sunspot with no penumbra.

**B:** It is a bipolar group without any penumbra.

**C:** A bipolar group with penumbra surrounding one end of the groups.

**D:** Bipolar with both ends surrounded by penumbra.

**E:** Similar to D class but with a length that is greater than the D class (10° < length < 15°).

**F:** Bipolar with penumbra on each end and length > 15°.

**H:** A unipolar group with no penumbra.

1. Penumbra Class:

This class defines the maturity, stability, size and complexity in terms of penumbra of the sunspot groups. This contains six classes that define the varying complexity of the groups.

**X:** no penumbra

**R:** An incomplete penumbra that partially surrounds the largest spot.

**S:** A small and symmetric penumbra.

**A:** small and asymmetric penumbra.

**H:** large and symmetric penumbra.

**K:** large and asymmetric penumbra.

1. Compactness Class:

This class gives the information about the relative spottedness of the sunspot's groups. Gives additional information about the area of the group, more importantly gives information on whether or not there are strong spots near the line of polarity. This class contains four categories:

**X:** for unipolar groups it is undefined.

**O:** Open. Few spots, if any on the line of polarity. Spots are very small in size.

**I:** Intermediate. Numerous spots lie on the line of polarity, without the presence of penumbra in any spot.

**C:** compact. Strong spots exist between the leading and the trailing spots with some of them possessing mature penumbra.

1. **Data Preparation**

Data preparation is third phase of the CRISP-DM methodology. It is the process of converting the data that was gathered in a from that is required for the research. This step involves five steps:

* **Select Data:** This step involves deciding upon the data sets that will be required for the analysis. It also involves selecting the columns and rows that will be used.
* **Clean Data:** This step involves the process of raising the quality of the data to level where it is satisfactory for the research. This task may require to remove, impute or correct the data.
* **Construct Data:** The process of deriving new attributes from the original data.
* **Integrate Data:** This step involves combining data from different datasets to create new data.
* **Format Data:** This is a crucial phase is the data cleaning process. Some attributes may require re-formatting such as changing the data type of the column.

(Put chart)

The datasets for from the NGDC open catalogue were taken and a number of steps were taken to process and get them in a format that is required for the research. The process follows the steps of CRISP-DM data preparation phase and are explained below:

**Selecting Data:**

The research explored two data catalogues namely the solar flares dataset and the sunspots dataset for the years 1996 to 2016. This data was available in space separated texts files on the where each file contained data of a particular year.

**Clean, Construct and Integrate Data:**

Since the data was stored in a space separated text file a number of steps were taken to clean the data and bring it in a format suitable for the research.

* Firstly, a function was made each for sunspots and solar flares data which would take in yearly text files as input and convert them into a csv file with each column of data properly labelled.
* In the second step all of the csv files that are made, the year, month and day column are combined together to form a continuous date.
* Finally, two csv files are made. The first csv files are termed flares and it contains the merged datasets for all the yearly flare data and the second csv file contains the merged datasets for all the yearly sunspots data.

**Formatting Data:**

To get the final data for the research the solar flare data and the sunspots data have to be combined on two columns: NOAA Number and Date. The two datasets are combined using the algorithm defined in (Qahwaji and Colak, 2007). The process was done in three steps:

First, the data two datasets (solar flares and sunspots data) were processed together row by row to find the rows that matched in both the tables based on the fact that they have the same NOAA number and if their NOAA number match then the time difference between the two recorded rows should be lesser than eight hours. If the above two criteria matched then the two rows are combined and stored in a different dataset and a new column 'Status' is labelled as Associated. This dataset contains information on all the sunspots that flared and what type of flare they produced.

Second, after getting the data on the spots that were associated with flares a different dataset was build which contained all the spots that did not produce a flare. For this dataset any spot whose NOAA number did not match with the ones in the flare dataset was taken out and then that record was stored in a separate dataset with its label column 'Status' changed to unassociated.

Third, the two new datasets are combined into one. Then any NA or blank value that may exist in the dataset is dropped. Final the columns that are not considered valuable for the final research data are identified and dropped from the dataset.

After performing the pre-processing steps, the dataset that will be generated will have the following columns:

|  |  |  |
| --- | --- | --- |
| Column | Description | Data Type |
| X-Ray Class | Class of X-Ray (X or M) | String |
| X-Ray Intensity |  | Number |
| Date and Time | Date on which the flare/spot was recorded. | Date Time |
| Magnetic Classification | Contains the Mt. Wilson Magnetic Classification. | String |
| Zurich Class | Describes the evolutionary sequence of spots. | String |
| Penumbra Class | Describes the size, maturity, stability of the spot. | String |
| Compactness Class | Describes the information about area and spottedness of the spots. | String |
| Number of Spots | Total number of spots in a sunspot | Number |
| Longitudinal Extent |  | Number |
| Area | The area of sunspot | Number |
| Status | Weather the sunspot have a associated flare or not (Associated or Unassociated) | String |

Table 3. : Description of columns of the final dataset achieved after processing.

1. **Modelling:**

After preparing the data the next step is to create a model. The tools for modelling were already decided in the business understanding phase. This phase focuses on the what sorts of models will be built and how they will be assessed. This phase includes four steps:

* **Selecting modelling technique:** In this step the models on which the data will be trained is decided upon.
* **Generate test design:** After the model is decided, a mechanism like splitting is needed to test the validity and quality of the model.
* **Build Model:** After deciding upon the model and validating technique the model is trained and the parameter values for the models are also specified.
* **Assess model:** The models are interpreted for their performance based on industry knowledge. The models are assessed and ranked on the basis of the performance criteria that is decided.

The aim of this model is to do a comparative study of state-of-the-art ensemble and deep learning models at their ability to effectively classify flaring and non- flaring sunspots. For conducting this research, a number of models are selected like: random forests, gradient boosted trees, neural nets and some general purpose machine learning models likes logistic regression and svm to check how much better the methods in study are better than the general purpose methods. The data is split into a 80-20 ratio for 80% of the data is used for training and the remaining 20% is used for testing. The models are trained on a combination of number of different parameters so that the best possible values for the parameters can be found. These models are later validated using there area under the curve (AUC), which tells about the degree of separation that the model was able to create, closer the value to 1 the better the model is able to separate the classes, and are ranked accordingly. The methods that are used in this research are described below:

* Ensembling
  + - Random Forests
    - Gradient Boosted Trees
* Neural Nets
* Logistic Regression
* SVM

1. Enembling:

Ensemble techniques are based on the idea of 'wisdom of crowds' (“Hands On Machine Learning with Scikit Learn Keras and Tensorflow.pdf,” n.d.) which means that a decision made by a larger group of people will be better than the decision made by an individual person. Ensembles are a type of machine learning approach that integrates many base models to create a single best-fit prediction model. Ensemble methods rather than relying on just one decision tree makes a number of decision trees by calculating which feature to use at each split and then make a final prediction based on the aggregate result of all the decision trees, i.e. they combine a number of models together to improve the predictive power and stability of a model. Ensembling techniques are further classified into Bagging and Boosting.

Bagging:

Bagging or bootstrap aggregating is the process of generating random subset of data from the training set with replacement. A individual data point in bagging can be chosen multiple times. The purpose of bagging is to reduce the variance produced in individual models. In bagging the weak learners are trained in parallel i.e., that each model is trained individually without the influence of any other model and then the average of the models is taken to make predictions. Bagging happens in three phases:

* Create random datasets by taking in randomly selected data from the original dataset.
* Fit a number of models each using different subset of the bootstrap data and using different parameters at each split to produce a better model.
* Taking an average of all the predictions to make a final overall prediction.

Boosting:

Unlike bagging where the models are trained in parallel, the models in boosting are trained in a sequential manner, that is the next model learns upon the mistakes of the previous model and tries to improve upon the predictions. In boosting the values that are predicted incorrectly by a previous classifier are chosen more often than the ones predicted correctly in the next classifier (Opitz and Maclin, 1999).

This research focuses on the use of two main ensemble based machine learning models: random forest which is based on the concept of bagging and gradient boosted trees which is based on the concept of boosting.

1. Random Forests:

A decision tree is limited to how accurately it can perform classification, to overcome this problem a number of decision trees are used together to arrive at the result and this is known as Random Forest. It works on the concept that a large number of uncorrelated trees working together will outperform a single individual tree. A random forest, as the name suggests is a forest of Decision Trees wherein each decision tree is trained using a random subset of data, this random subset of data is generated using bagging. In a Random Forest N number of decision trees are built using the bootstrapped dataset. Each tree in this forest has different set of parameters and subsets of data. Instead of searching for the best features at every split random forest works by looking for the best feature from a random subset of features, this results in a higher bias and a lower bias. After these trees are built, the out-of-bag data is used to test the accuracy of the model. An instance of data is run down all the trees and the vote is recorded for each of the tree. The class that was predicted by the majority of the trees is taken to be the prediction.

Random forests are effective because all the trees in the forests are unrelated to each other and since the final prediction of the forest is a averaged prediction of all the trees it helps in achieving higher accuracy.

In the figure below, an instance is given to the random forest. The algorithm then runs that instance through every tree in the forest and keeps the vote on the classification given by each tree. In the end the class that was voted by majority of the trees is given to be the prediction of the forest.



Figure 3.2: Steps that a random forest follows to make a prediction

1. Gradient Boosting:

Gradient-boosted trees is a machine learning algorithm that is used to optimize the predictive power through successive steps in learning process. Every iteration of decision tree in a gradient boosted model is involved in adjusting the weights, biases and coefficients in order to reduce the loss function. Gradient Boosting is a ensemble technique that relies on the concept of boosting in order to improve the predictive power of the model. It is based on the assumption that when the best feasible future model is merged with prior models, the overall prediction error is minimized. Gradient Boosting operates by adding predictors to an ensemble in a systematic manner, each one correcting the previous one's errors. Instead of adjusting instance weights with each iteration, this technique seeks to fit a new predictor to prior predictors' residual errors. The output of the new tree is then added to the output of the previous trees in the model.

This method is continued until a predetermined number of trees have been achieved or the loss has been minimized to below a specific level. Gradient boosting involves three main elements:

* A loss function:

Loss function is a way to see how effectively an algorithm models a dataset. It will provide a larger value if the forecasts are completely wrong. If they're decent, it'll give a lower number. It depends in the type of problem being solved. Since the problem at hand is a classification problem a logarithmic loss function can fit our model perfectly.

* A weak learner:

In gradient boosting, decision trees are generally used as weak learners.

* Additive model:

The additive component of a gradient boosting model derives from the fact that trees are added to the model over time, and when this happens, the values of existing trees aren't changed.

GBT are one of the most popular machine learning algorithms, what makes them so popular is their ability to provide the best possible accuracy, they provide a lots of flexibility with hyperparameter tuning, and can handle missing data easily. However, they run the problem of overfitting the model, since the number of trees are high they can take a lot of time and are computationally heavy.

Python's scikit-learn library is used to build a Gradient Boosted Classifier. The algorithm will be paired with the grid search cv to test out different hyperparameters for the tress in order to find the best fir parameters for the algorithm. The final model will then be trained using the parameters that were calculated in the previous step.

1. Multi Layered Perceptron:

An artificial neural network is machine learning algorithm that is based on the network of biological neurons that are present in out brain. A multi layered perceptron is a back propagation neural network made up of many perceptron's stacked on top of each other. One input layer, one or more hidden layers, and an output layer make up each MLP. The lower layers are those closest to the input, while the output layers are those closest to the output. A bias is present in every layer except the output layer.

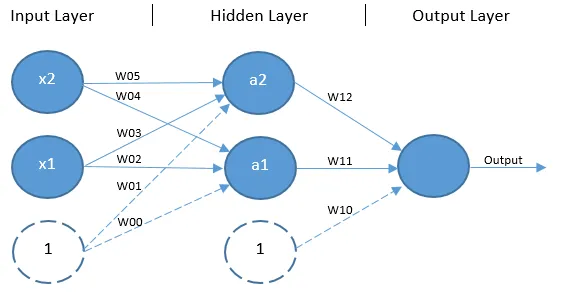


Figure 3.3: Architecture of MLP with two input, one hidden layer and an output layer.

In the above figure x1, x2 are out input data sets, w00 … w12 are weights associated with each neuron and '1' are bias terms.

A perceptron is the most basic unit of a MLP. The job of a perceptron in a MLP is to calculate the weighted sum of all the inputs.

It applies a activation function to the value. This activation function is used to convert the values between a range of 0 and 1. There are multiple activation function that can be used but for this study a sigmoid function will be used. Then these values are sent to the next layer and it performs a similar calculation.

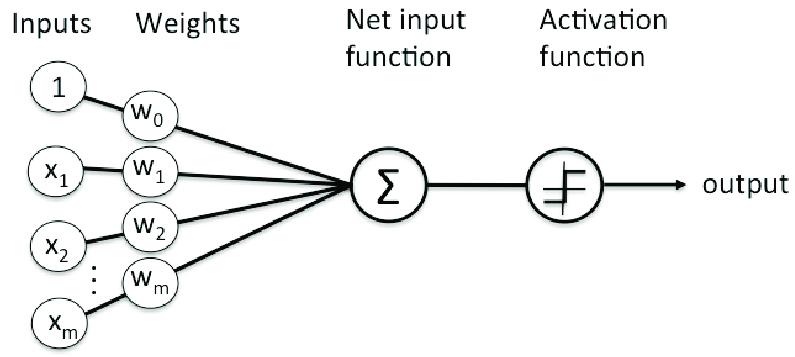


Figure 3.4: An artificial neuron that computes weights and applies activation function. (put ref.)

A backpropagation neural network uses the following steps:

* It works with one tiny batch at a time and runs through the entire training set several times. Each pass is referred to as an era.
* Each batch is sent to the input layer of the network, which then transmits it to the first hidden layer. The method then computes all neurons output and passes the result to the next hidden layer, and so on, until the model reaches the output layer. This is known as the forward pass.
* The method then calculates the output error of the network (loss function)
* It then calculates the contribution of each output link to the error.
* Working backwards until it reaches the input layer, it then calculates how much of these error contributions occurred from each connection layer. This pass in known as the backward pass.
* At last a Gradient decent step is used to tweak all the connection weights in the network.

Equation: Logistic sigmoid step function.

1. Support Vector Machines (SVM) :

SVM is a versatile and powerful supervised machine learning model that can do classification, regression, and outlier identification. SVM's are best for classification of complex datasets. They work by plotting each data item as a point in n-dimensional space (where n is the number of features you have), with the value of each feature being the value of a certain coordinate in the SVM algorithm. Then classification is done by plotting the hyperplane that best separates the different classes of the classification problem.

It works by fitting a line to the data that clearly segregates the points on two sides. As shown in figure 3.2, a best-fit line for a SVM model is a line that does a better job at separating two different classes, here the line L2 does the best job at separating the blue and the red dots and hence is the best fit line for the model. The underlying assumption is that the farther SV points are from the hyperplane, the more likely they are to be accurately classified in their corresponding area or classes. The location of the vectors affects the position of the hyperplane, therefore SV points are highly important in calculating the hyperplane.

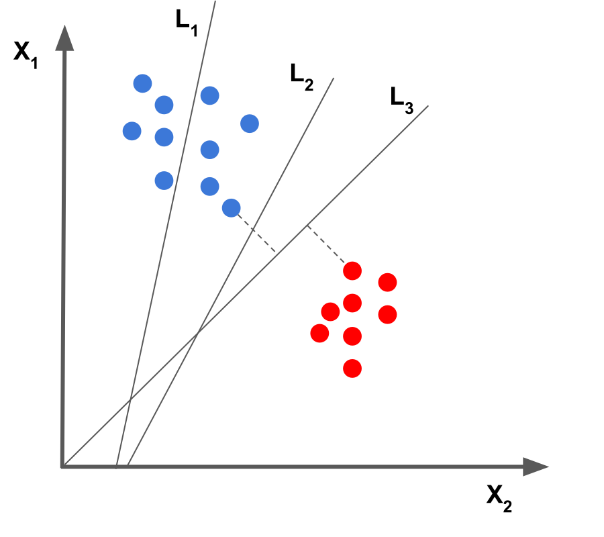


Figure 3.2: SVM Model with 3 different separation lines

The above example is based on the assumption that the SVM line is able to perfectly separate the two classes and that the data is linearly separable, this is an example of a hard margin classification. Hard margin classifier is very sensitive to outliers and may result in a bad model if outliers are present. To solve this model a soft margin classification is made, this is based on the idea to allow SVM to make a few mistakes and keep the margins as wide as possible.

The hyperplane is a function that may be used to distinguish between distinct characteristics. The function used to classify between features in 2-D is a line, whereas the function used to classify features in 3-D is a plane, and the function used to classify points in higher dimensions is a hyperplane. If there are 'm' dimensions then the formula for hyperplane can be given as:

Where w1,w2,…wn = vectors; w0 = bias term; x1, x2, … xn = variables.

Advantages of SVM:

* This algorithm is really effective in higher dimensions.
* When classes are separable, this is the best algorithm.
* Only the support vectors affect the hyperplane, therefore outliers have a less impact.

SVM algorithm can be implemented using the widely used Python's scikit-learn package. The scikit-learn package also provides an way to tune hyperparameters of the algorithm.

1. Logistic Regression:

It is a machine learning algorithm that is based on the concept of probability and is used for classification problems. A logistic regression model can be binary or multinomial based on the number of output variables that are there in the problem. For the sunspots association problem there are only have two output variables , associated and unassociated, the logistic regression model that will be used here is a binary model.

In a binary problem the probability if the outcome is given by the equation:

Where Where X(i) are independent variables. The logistic regression model fits a 'S' shaped curve to the data, this curve is the sigmoid function which maps the values between 0 and 1.

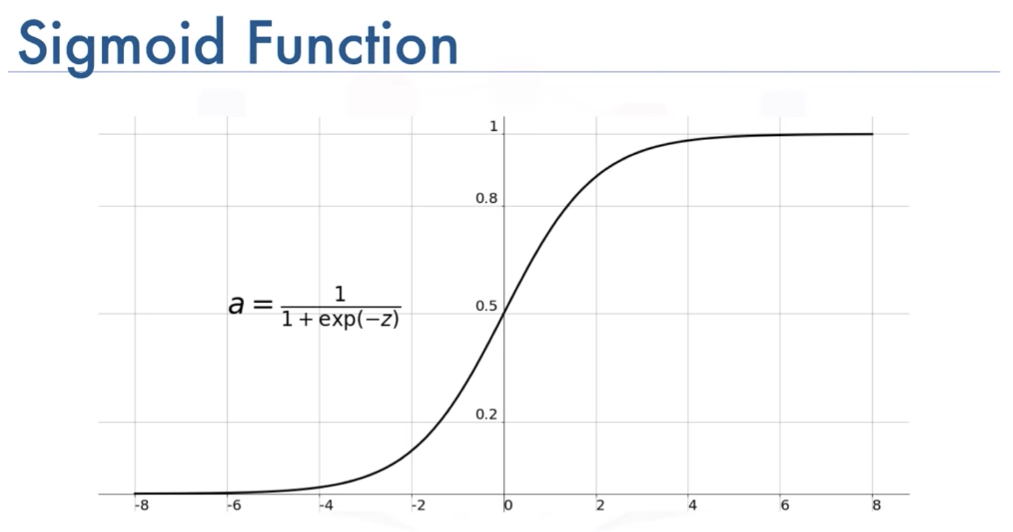


Figure 3.3:

The model gives us a set of output probabilities based on the input predictors that are fed into the model. A output is given out to be either 0 or 1 based on the fact weather the probabilistic value of the class lies above or below a certain threshold value.

1. **Evaluation:**

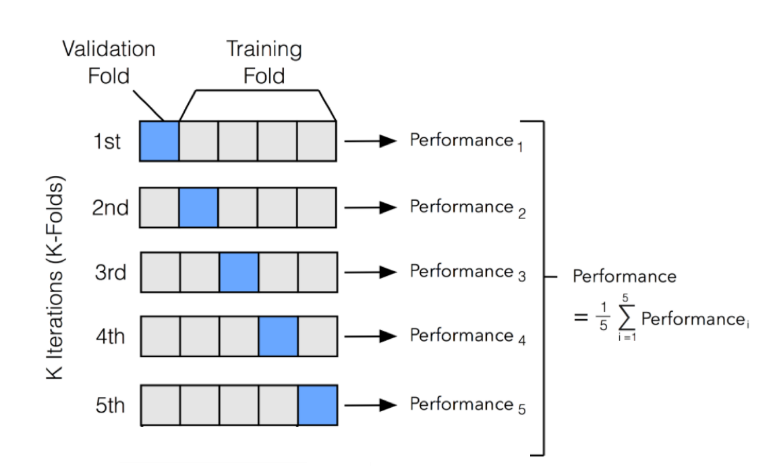
The next phase in the CRISP-DM methodology is Evaluation. It is the process by which it is checked that weather or not the model built using the training will perform well on the data it has not seen before. In this phase the results are evaluated on the basis that defines our criteria for success. This phase consists of three different steps:

* Evaluating results: Check if the results match with the success criteria.
* Review the process: In this step a thorough review of the results is done to check if some important criteria have been missed or not. This step also covers quality issues i.e., that the model was built accurately and whether or not it violates any business criteria.
* Determining the next step: In this step an analysis is made on whether to proceed with the results that have been generated or to go back and reiterate over the entire model to improve performance.

In this research a number of different performance criteria are considered in order to get a thorough evaluation of how well the various models perform on the data and to draw a comparison between different models.

1. Cross Validation:

Cross validation is a machine learning evaluation approach that involves training the model on a part of the available data and then assessing it on the remaining input data. This study makes use of the K-fold cross validation process to perform random splits on the training data. In the K-fold cross validation the entire data is broken into equal folds depending on the value of K that is defined. While training the model is trained on K-1 parts of the data and the remaining 1 part is used for testing. This process continues until each part has been used for testing, giving each point equal opportunity to be used as testing set. The model returns K accuracy values, which are then averaged to get the final value.



1. Confusion Matrix:

It is used to assess a classification model's performance using a set of test data for which the real values are known. It provides a summary of the classification problem results. Confusion matrix are also used for calculating other performance measures such as accuracy, precision, recall, F1-score.



Where TP stands for True Positive, FP stands for False Positives, FN stands for False Negatives and TN stands for True Negatives.

* 1. Accuracy: It is the measure of determination which helps to decide which models does a better job at identifying the relationships between the dependent and independent variables. It is calculated using the formula:

where TP stands for the True Positive (the values that are positive and have been correctly predicted by the model to be positive) and TN stands for True Negative (the values that are negative and have been correctly identified by the model to be negative).

* 1. Precision: It tells us about how many were correctly identified to be positive out of all the positive cases that were predicted. This is given by the formula:
  2. Recall: It tells us that out of all the positive cases how many were correctly identified to be positive. This is given by the formula
  3. F1 Score: F1 score is the average between precision and recall. This score is used when a balance between precision and recall is required. It is a better measure to use for a unbalanced data. It is given by the formula:

1. Area Under the Curve (AUC):

The AUC curve tells us how much better the model is at distinguishing between classes. The higher the value of AUC the better the model is at drawing a line between different classes. The AUC value is between 0 and 1 where the value closer to 1 is considered good compared to a value closer to 0.

The main goal in a data-science project is to focus on optimizing the performance of a model but sometimes it is also very important to understand the way it is behaving and how the various predictors are contributing towards the final prediction. Most of the models that are used in this research are black box models i.e., how the model arrived to a particular prediction is not known and since the events that lead to the eruption of solar flares from the surface of the sun are not fully understood, it is very much important to understand how the different features contributed towards the final prediction. Understanding how a particular model came to a particular prediction can also help us develop better warning systems with lesser features hence making the model less complex. There are a number of ways in which the importance of the features can be checked, in this research only two of them will be used to understand and evaluate the model:

1. Feature Importance:

This is a model specific evaluation matrix which tells us about how each of the feature contributed towards the final prediction. This will be used to check the importance of each of the feature in the model.

1. LIME:

Local Interpretable Model Agnostics Explanations (LIME) is a model agnostics approach that aims at recognizing an interpretable model over a representation that is locally faithful to the model. This approach is used to locally check some predictions of the model in order to understand how the model reached at the prediction. This evaluation metric will help to provide a better understanding of how each of the variable is contributing towards the final predictions.

1. **Deployment:**

The final stage in the CRISP – DM methodology is the deployment phase. In this phase a strategy to deploy evaluation results is though of. This phase consists of four steps:

* Deployment Plan: This step is required to develop a comprehensive plan for deploying a model.
* Maintenance and monitoring plan: A well documented maintenance strategy is a good practice in avoiding excessively lengthy periods of inaccurate data results. In this step a maintenance and monitoring strategy is developed.
* Final report: a summary document of the entire project that includes data mining results.
* Project review: A review report of the entire project that includes what went right or what went wrong or things that can be improved about the project.

The research problem is a binary classification problem that classifies data into flaring and non-flaring classes. To do this a number of ensemble and deep learning based models are developed and are compared on the basis of evaluation criteria's discussed earlier.. After doing a thorough evaluation of the models and finalizing any improvements that could have been needed in the project, a model that does the best job at creating the highest degree of separation between the two target classes will be selected as the final model and may be used develop a system that can predict flaring sunspots and give an early warning prediction of solar flares.

# Chapter 4 – Data Analysis and Model Results:

This section is divided into three parts. The first part focuses on getting a understanding of the various features of the dataset by discussing a number of visualisations, in part two the performance of each of the model is presented for each of the platforms they have been trained on based on the evaluation metrices that have were mentioned in chapter 3, in part 3 focuses on finding out how much each of the feature has contributed towards the score for the final prediction using the feature importance matrix and LIME analysis method in both python and h20.ai

**4.1 Exploratory Data Analysis:**

Exploratory Data Analysis (EDA) refers to the process of performing a initial understanding on the data in order to find new patters, spot out anomalies, and to get a feel of the data with the help of summary statistics and data visualisations.

**4.2 Model Results:**

Since a number of models were built using a number of different platforms, the performance of each of the model for each of the performance metrics is depicted with the help of tables. The three platforms that were used to train these models are:

* RapidMiner
* Python
* H2O.ai

**4.3 Feature Importance:**

The feature importance analysis will help us to understand how each of the variable in our research dataset contributes in predicting the outcome of the model. For this purpose the use of two different feature evaluation is made:

* Feature Importance:

This is a build in function in both python and h20.ai that gives a general idea of how much each of the factor in the dataset contributed towards the model.

* LIME:

LIME is a model agnostics approach that explains how the model came a particular prediction locally. Three different cases will be used to explain the results generated through the implementation of LIME. The three different predictions are: a positive case, a negative case, and a inaccurate prediction.

## 4.1 Exploratory Data Analysis (EDA):

Data visualisation is one of the best ways to get an understanding of the data that is being used to carry out some research. The data for this contains a number of categorical columns that describe the complexity of the sunspots as discussed in the data understanding section of chapter 3. A number of bar plots will be drawn out to compare how the different complexities of sunspots affect the number of spots within a sunspot group and the area associated with them while at the same time looking at how each of these variables are corelated with flaring and non-flaring sunspots. Each of the categorical predictor will be analysed one by one and a conclusion will be drawn at the very end of this section.

1. Magnetic Classification:

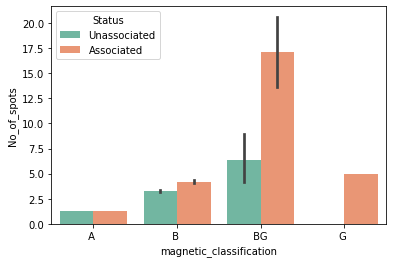


Figure 4.1 : A comparison of magnetic field vs the number of spots for associated and unassociated flares.

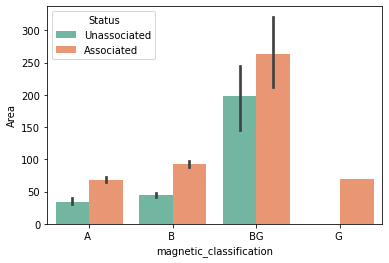


Figure 4.1 A comparison of magnetic field vs the number of spots for associated and unassociated flares.

As The Mt. Wilson magnetic classification classifies magnetic field within a sunspot group into a total of eight classes that define the magnetic field complexity and the polarity of the spots. The more complex the magnetic field gets within a sunspot group the more are it's chance of tangling with each other and resulting in an eruption of solar flare. The Gamma (G) and Beta – Gamma ( BG ) classes signify higher complexity of magnetic fields inside a sunspot and from the graph it can also be inferred that a higher complexity is associated with more number of and a bigger area of sunspot and hence more number of flares.

1. Zurich class in comparison with number of spots and area.

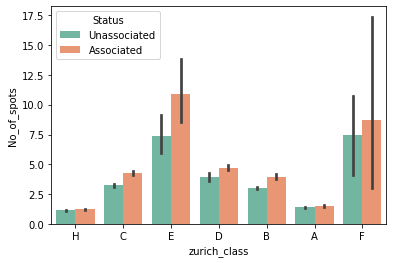


Figure 4.1 : A comparison of zurich class vs the number of spots for associated and unassociated flares.

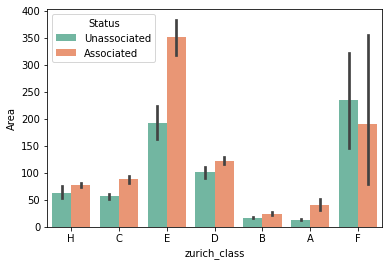


Figure 4.1 : A comparison of zurich class vs area for associated and unassociated flares.

The zurich class is one part of the Mc Intosh classification of sunspot groups. This class is defined on the basis of penumbra present around a sunspot and the magnetic polarity nature of the group. The same trend as was observed earlier can be noticed here too, a more complex class is associated with more number of spots and a greater area. In the graph depicted above classes D, E, F have more number of spots for associated flares as compared to unassociated ones and the trend is more prominent for classes with higher complexity.

1. Compactness in comparison with number of spots and area.

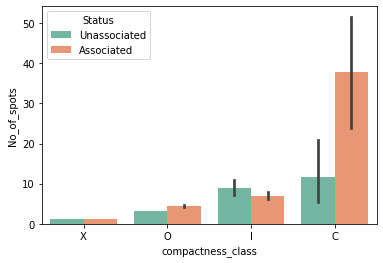


Figure 4.1 : A comparison of compactness class vs number of spots for associated and unassociated flares.

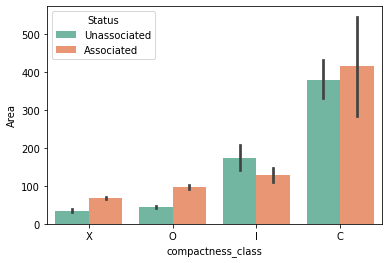


Figure 4.1 : A comparison of compactness class vs number of spots for associated and unassociated flares.

The compactness shows the relative distance between the spots in a sunspot group i.e., how close the spots are, therefore a higher compactness class represents a more compact sunspot group and hence more number of spots within it. Looking at the graph it can be seen that the flares associated with the groups that has a greater number of spots are also the ones that more compact. A similar observation can be made about the area when compared with the compactness of the sunspot groups, the more complex and compact groups have a bigger area.

1. Penumbra vs number of spots:

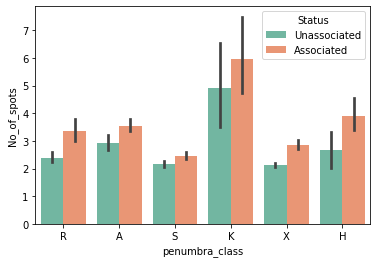


Figure 4.1 : A comparison of penumbra class vs number of spots for associated and unassociated flares.

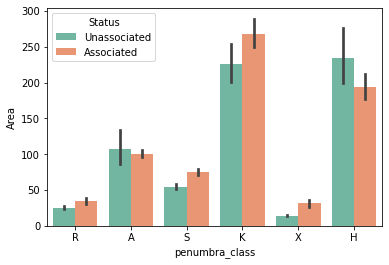


Figure 4.1 : A comparison of penumbra class vs area for associated and unassociated flares.

The presence of penumbra around a sunspot group is very important in order to figure out flaring capabilities of a sunspot group. The penumbra class is used to define the size, maturity and complexity in terms of the kind of penumbra present. The chances of flares increase with the increase in size and complexity of the penumbra. As can be seen from the graph the spots that are associated with flares generally have more number of spots as compared to the spots that are not associated with flares. The relation between area and the penumbra class is same as what have been seen for other classes too. The area of the spots increases as the complexity of the class increases and the same is evident from the plots.

There is a general trend that can be seen with the plots that were made between the categorical predictors against the number of spots and area is that as complexity of the classes increased so does the area and the number of spots within the sunspot groups. Also, for the flaring sunspots the number of spots with the group and the area of the group is generally higher than those of the non-flaring once. Hence, it can be expected to see more complex classes to be more associated with flares than those with lesser complexity.

## 4.2 Model Results:

This section gives the results of the various performance metrices for each of the models that were trained in RapidMiner, Python and H2O. In RapidMiner the default model parameters were used to train the algorithms without any hyper-parameter tuning and these results were used as an initial benchmark to improve upon in other platforms. The models that were trained using python and H2O try to improve upon the results of RapidMiner by the use hyper-parameter tuning and k-fold cross validation. For each ensemble and deep learning model a number parameter was passed in using GridSearchCV. The combination of parameters that were returned after running GridSearchCV, which gave the highest score on the training data, were then used to train the final model on which the prediction will be made. Both python and h20.ai support different kind of hyperparameter tuning. Even though the parameters are mostly kept same for most of the model, python offers a wider range of tuning options and hence some of the hyper-parameters that are available in python are not in h2o and hence were left untouched.

1. Random Forest:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Parameters > Method ˅ | n\_estimators | max\_features | criterion | max\_depth |
| Python | 50, 100, 200, 500 | auto, sqrt, log2 | gini, entropy | 4, 6, 7, 8, 10 |
| H2O | 50, 100, 200, 500 |  |  | 4, 6, 7, 8, 10 |

Table 4.2 : Hyperparameters used for the random forest model.

The n\_estimmators refers to the number of trees that will be trained in the random forest, the max\_depth refers to the maximum number of nodes that a tree can have, criterion measures the quality of split and two different criterion were used in python and max\_features is used to calculate the number of features that will be used in an node.

1. Gradient Boosted Trees:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Parameters >  Method ˅ | n\_estimators | max\_features | max\_depth | learning\_rate |
| Python | 50, 100, 200, 500 | auto, sqrt, log2 | 4, 6, 7, 8, 10 | 0.01, 0.05, 0.06 |
| H2O | 50, 100, 200, 500 |  | 4, 6, 7, 8, 10 | 0.01, 0.05, 0.06 |

Table 4.2 : Hyperparameters used for the gradient boosted trees model.

The hyper parameters discussed in random forest are exactly the same for the gradient boosted trees as well. The only new one is learning rate which defines the rate at which at the model adjusts the weights of the tree to arrive at an ideal weights for the model.

1. Neural Nets:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Parameters >  Method ˅ | Neurons | Activation | Epochs | Batch Size |
| Python | [20, 50, 100] | ['softmax', 'relu', 'sigmoid'] | [10, 50 100, 200] | [5, 10, 15] |
| H2O | [[20, 20], [50, 50], [100, 100]] | ['Tanh', 'Rectifier'] | [10, 50, 100, 200] |  |

Table 4.2 : Hyperparameters used for the deep learning model.

The above parameters are defined for a neural network model that has two hidden layers and an output layer. The neurons parameters define the number of neurons in the hidden layers, activation defines how a neuron in the model will arrive at an output given a set of inputs. Epochs refers to the number of times the algorithm will go through the entire dataset, batch size is defines the number of samples of the data the algorithm will go through before updating the model parameters.

* 1. **RapidMiner:**

Rapidminers Auto model tool is and extension of Rapidminer studio that makes the process of building and validating models easier. The auto model extension is used here to train a number of machine learning models that will give an initial understanding of how the various models will perform on creating a degree of separation between the flaring and the non-flaring sunspot groups. The results that were generated by running the data through a number of algorithms will be used as a benchmark and then these results will be improved upon in the two other platforms: python and h2o.ai.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Performance Metric | Naïve Bayes | Generalized Linear Model | Logistic Regression | Deep Learning | Decision Trees | Random Forest | Gradient Boosted Trees | SVM |
| Accuracy | 63.9% ± 0.7% | 63.5% ± 1.4% | 63.7% ± 1.4% | 58.8% ± 0.6% | 61.0% ± 2.7% | 58.5% ± 1.7% | 58.7% ± 1.7% | 51.4% ± 1.4% |
| F1 Score | 67.6% ± 1.2% | 54.2% ± 2.2% | 54.0% ± 2.2% | 34.9% ± 1.8% | 43.9% ± 5.2% | 34.2% ± 4.6% | 33.7% ± 3.3% | 42.4% ± 2.5% |
| AUC | 0.713 ± 0.014 | 0.719 ± 0.016 | 0.720 ± 0.017 | 0.734 ± 0.015 | 0.643 ± 0.038 | 0.713 ± 0.033 | 0.715 ± 0.021 | 0.548 ± 0.027 |
| Precision | 61.3% ± 0.8% | 72.7% ± 2.7% | 73.7% ± 2.9% | 82.9% ± 3.7% | 77.9% ± 5.1% | 81.9% ± 3.1% | 84.8% ± 2.6% | 52.0% ± 2.0% |
| Recall | 75.5% ± 3.5% | 43.2% ± 2.6% | 42.6% ± 2.5% | 22.1% ± 1.6% | 30.6% ± 4.3% | 21.7% ± 3.7% | 21.1% ± 2.6% | 35.9% ± 3.0% |
| Classification Error | 36.1% ± 0.7% | 36.5% ± 1.4% | 36.3% ± 1.4% | 41.2% ± 0.6% | 39.0% ± 2.7% | 41.5% ± 1.7% | 41.3% ± 1.2% | 48.6% ± 1.4% |

Table 4.2 : Performance metrics of the different machine learning algorithms in RapidMiner.

Most of the models give an accuracy score of over 60% with a standard deviation greater than 2.0%. The accuracy seems to be pretty good given the fact that these models haven’t been tuned at all. Out of all the models SVM shows to be performing very bad with only a accuracy of 50% while the ensembling techniques are performing pretty good with accuracy of 66% and a AUC curve above 0.7. These results will be used as a starting point and will be improved upon using python and H2O.ai.

* + 1. **Python:**

Python is a coding language that provides an easy implementation of various machine learning algorithms through the use of it's scikit-learn library. The scikit-learn library presents easy to implement functions of different ensemble, deep-learning and common machine learning algorithms. It also gives an easy way to do hyperparameter tuning using the GridSearchCV function. The table below represents the results that were calculated on the predictions made on the model that was trained on the best fit parameters found using the GridSearchCV method.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Performance Metric | Logistic Regression | Deep Learning | Random Forest | Gradient Boosted Trees | SVM |
| Accuracy | 0.66 | 0.714 | 0.75 | 0.80 | 0.65 |
| F1 Score | 0.65 | 0.715 | 0.76 | 0.81 | 0.65 |
| AUC | 0.72 | 0.715 | 0.84 | 0.88 | 0.72 |
| Precision | 0.69 | 0.73 | 0.74 | 0.79 | 0.67 |
| Recall | 0.61 | 0.78 | 0.78 | 0.84 | 0.64 |
| Mean Squared error | 0.33 | 0.28 | 0.24 | 0.19 | 0.34 |

Table 4.2 : Performance metrics of the different machine learning algorithms in Python.

The results shown above are an improvement on what was achieved in RapidMiner. These results are a result of hyper-tuning a number of model parameters using different values and calculating the accuracy metrices after training the models on the best fit parameters. Ensemble based machine learning models achieved the highest values of the performance metrices.

* + 1. **H2O.ai:**

H2O's API makes It possible for a system to run and train the models faster than the normal python environment, it can handle huge amount of data effectively and efficiently. Since the parameter tuning options are lesser in h20,ai compared to that of python, the models are trained on comparatively lesser combinations of parameter values.

Table 4.2 : Performance metrics of different machine learning algorithms in h2o.ai.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Performance Metric | Logistic Regression | Deep Learning | Random Forest | Gradient Boosted Trees | SVM |
| MSE | 0.21 | 0.41 | 58.5% ± 1.7% | 58.7% ± 1.7% | 51.4% ± 1.4% |
| RMSE | 0.45 | 34.9% ± 1.8% | 34.2% ± 4.6% | 33.7% ± 3.3% | 42.4% ± 2.5% |
| LogLoss | 0.61 | 0.734 ± 0.015 | 0.78 | 0.715 ± 0.021 | 0.548 ± 0.027 |
| AUC | 0.73 | 82.9% ± 3.7% | 81.9% ± 3.1% | 84.8% ± 2.6% | 52.0% ± 2.0% |
| Gini | 0.46 | 22.1% ± 1.6% | 21.7% ± 3.7% | 21.1% ± 2.6% | 35.9% ± 3.0% |
| Classification Error | 36.3% ± 1.4% | 41.2% ± 0.6% | 41.5% ± 1.7% | 41.3% ± 1.2% | 48.6% ± 1.4% |

## 4.3 Feature Importance:

This section discusses upon the results of two evaluation metrices. The first evaluation metric is the feature importance metric. This metric scores the different features of the predictors on the bases of their contribution towards predicting the final class, this gives an global idea of how important the various features are. The second evaluation metric is the Local Interpretable Model-agnostic Explanations also known as LIME. This evaluation metric gives an local interpretability of a prediction i.e., how the model arrived at a particular prediction and how much the various factors contributed towards the final prediction can be found by the use of this metric. This section will discuss both the feature importance and the LIME metric separately for python and h20.ai for each of the machine learning model.

**Python:**

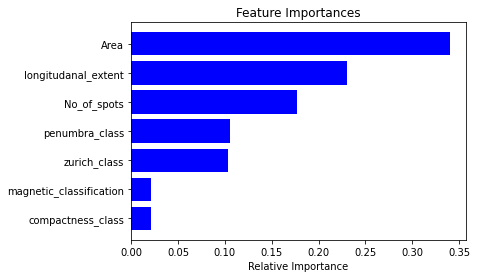
****

Figure 4.3 : Feature importance graph for Gradient Boosted Trees.

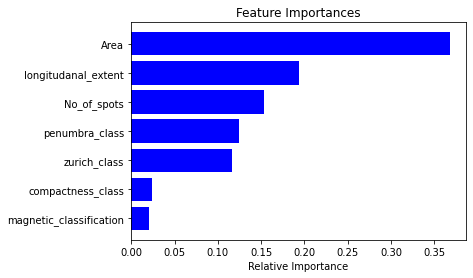
****

Figure 4.3 : Feature importance graph for Random Forests.

The above bar graphs shows the feature importance plots for two methods: gradient boosted tress and random forests. The graph depicts how useful the various input features are at predicting the target column, in this case weather or not a spot will flare or not. Looking at graph it becomes clear that the morphological properties: area, number of spots, longitudinal extent are the most useful variables at predicting the outcome variable. Amongst the McIntosh classes the penumbra class has the highest score and the magnetic classification contributes the least towards predicting the final class.

**H2O:**

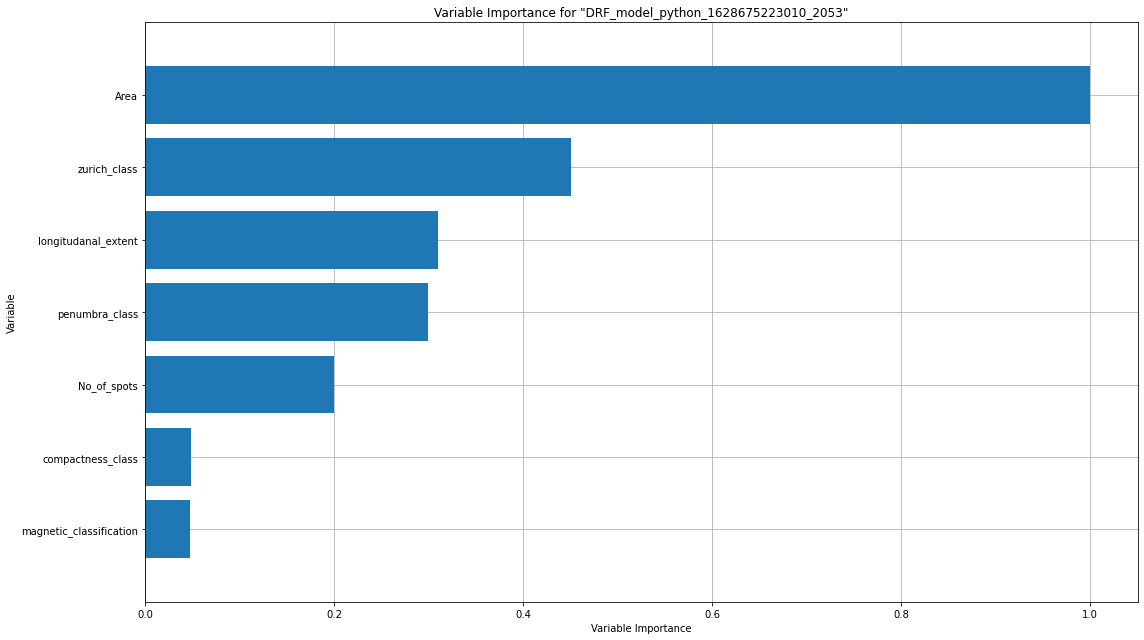
****

Figure 4.3 : Feature importance graph for Random Forest in H2O.ai.

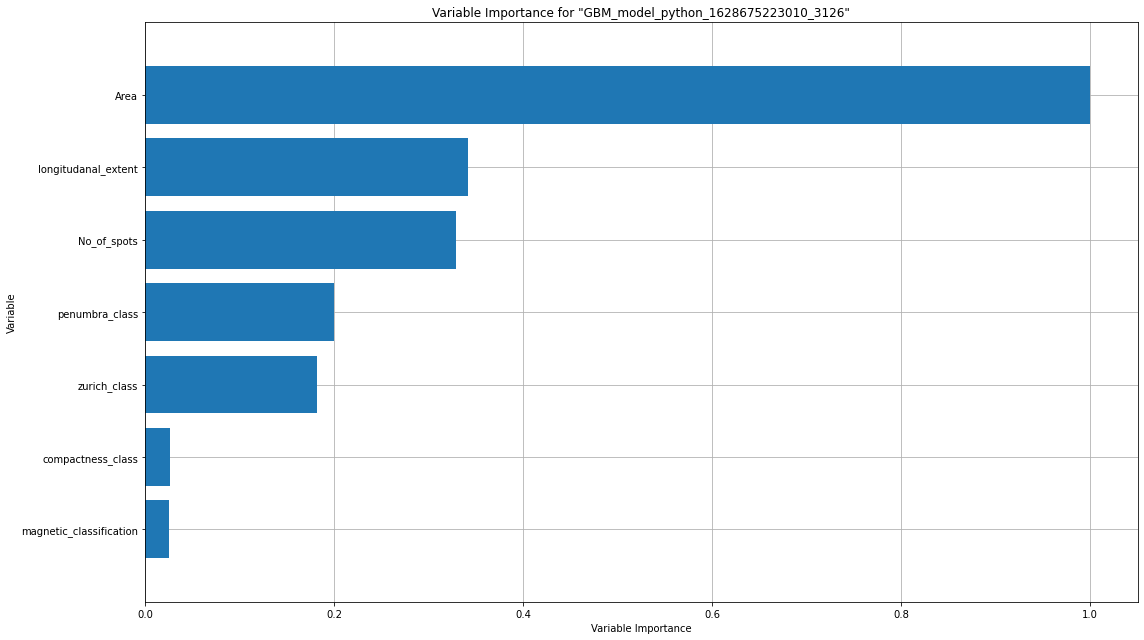
****

Figure 4.3 : Feature importance graph for Gradient Boosted Trees in H2O.ai.

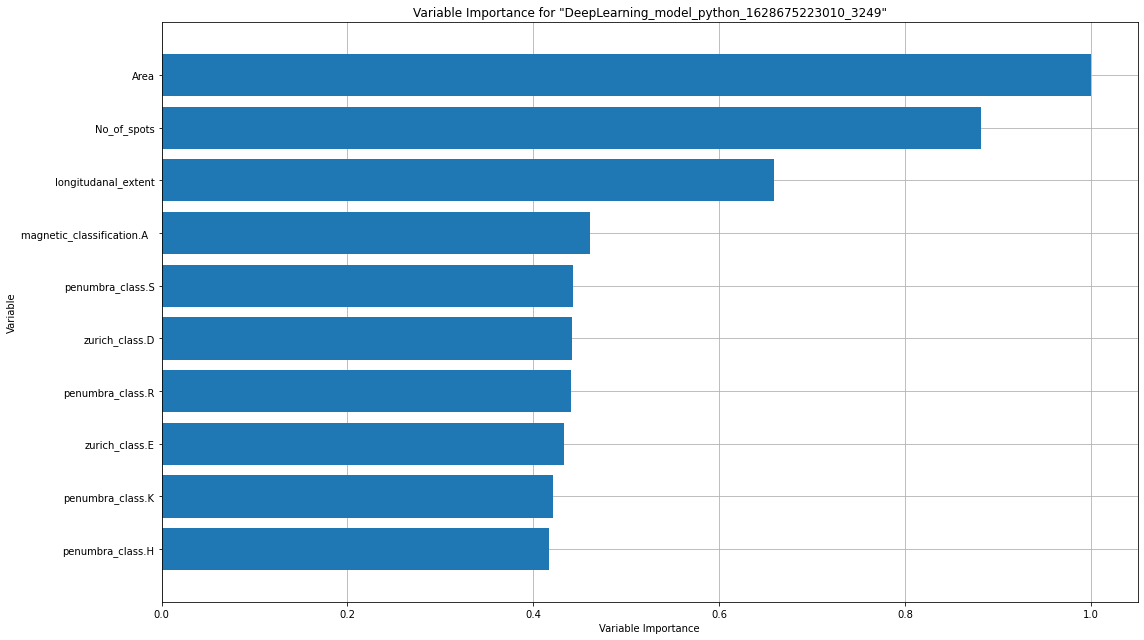
****

Figure 4.3 : Feature importance graph for Neural Nets in H2O.ai.

Looking at feature dependencies graphs for the random forest, gradient boosted trees and neural network models trained in H2O.ai, a number of interesting insights can be drawn from them. The plots of each of the model will be analysed one by one and a final comparison will be made in the end. For the random forests the area and the zurich class have the highest score and are the main input variables when predicting the target class followed by longitudinal extent and number oof spots with magnetic classification contributing the least towards the final prediction. For gradient boosted trees the area, longitudinal extent and the number of spots played a vital role towards the final prediction followed by the McIntosh classification and here too it can be seen that the magnetic classification plays the least role in predicting the outcomes. For neural network the feature importance graphs become a little more detailed, as seen in rest of the models area, number of spots and longitudinal extent are the best predictors a odd feature is the magnetic classification class A variable that high score compared to the rest of the remaining variables. The rest of the classes of penumbra and zurich shown in the Figure 4.3 5 are complex classes and hence their importance score is expected.

**LIME:**

Using LIME can help get an local interpretability of how the models arrive at a particular prediction. Downside of lime is that it is not supported in h2o.ai as of yet. So to understand the local interpretability of, the best performing model, gradient boosted tree's outcome will be used. In this three examples will be taken: one that shows the correct prediction of a flaring class, one that shows the correct prediction of a non flaring class and the last that gives a wrong prediction.

**Case 1: Class that was positive and was predicted positive:**

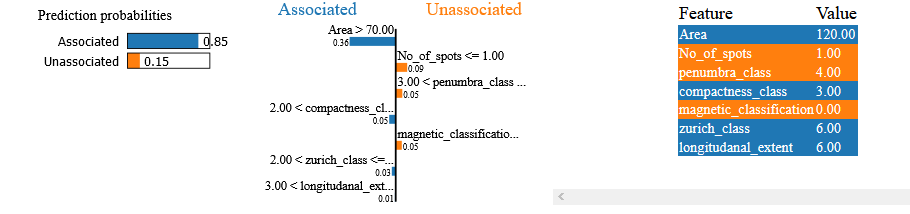


Figure 4.3 : Local interpretation of a record that flared and the model predicted it as a flaring spot.

The above image shows the model came to the prediction that the record is a associated one i.e., it is a flaring sunspot. How the model came to the prediction and how much the different variables contributed towards the prediction and the values that each of the predictors held for the record can be all seen in the image. The ones marked red showed that the prediction may be the positive class while those in orange pointed out otherwise. Looking at the tree plot it can be inferred that the area, zurich class, compactness class, longitudinal extent all hold high values and more complex classes and hence their score was more towards a positive class while the remaining held lower values and hence their score pointed towards the negative class.

**Case 2: Class that was negative and was predicted negative:**

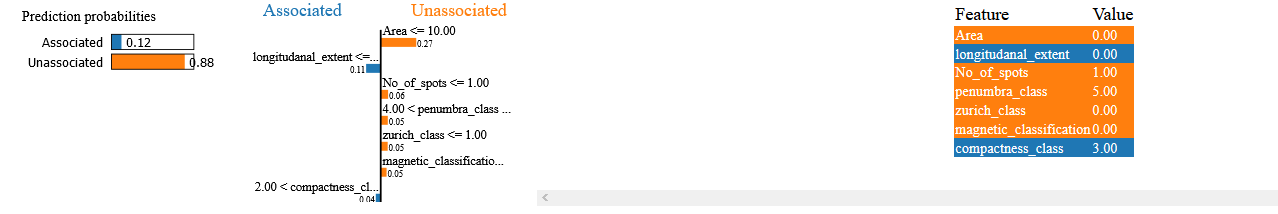


Figure 4.3 : Local interpretation of a record that did not flare and the model predicted it as a non-flaring spot.

The above image of a record that does not have a flare associated with it and the model predicted it correctly to not have any flare associated with it, i.e, it is an unassociated spot. The model had an 88% accuracy at predicting it as an non-flaring spot and only 12% pointed it towards being an associated spot. The global dependencies that were discussed in the above section hold true here, area contributed the most towards the prediction with a value of 0.27 followed by number of spots with value of 0.06 and then the rest of the McIntosh classes each contributing 0.05 towards the final prediction. The two predictors that pointed out otherwise in this model are the longitudinal extent and the compactness class. The higher complexity of the compactness class is understandable as to why it contributed towards a flaring spot prediction.

**Case 3: Class that was positive and was predicted negative by the model**:

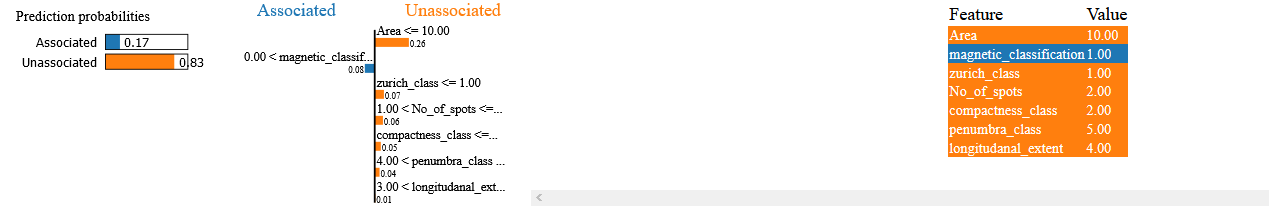
****

Figure 4.3 : Figure 4.3 : Local interpretation of a record that flared and the model predicted it as a non-flaring spot.

The above image represents the scenario of a wrong prediction. In this case the data for a spot that is a flaring spot is fed into the model and the model predicted it as a non-flaring spot i.e., a unassociated spot. Area and number of spots should have higher values in order for them to contribute towards a flaring spot, since both these predictors have lower values it is understood why they score higher towards a negative class. However, the longitudinal extent and penumbra class both have higher values of complexity and should be pointing towards a associated spot instead of an unassociated one. The above case could be because of an extreme event as a flaring sunspot, in general, has a higher area, with more number of spots and has relatively complex classes for some of the predictors.

# Chapter 5 – Discussions:

## 5.1 Data Overview:

After conducting a exploratory data analysis (EDA) it a general pattern was observed in the data used in this research. For each of the predictor that was plotted against number of spots and the area it was observed that the more complex classes and greater number of spots and a larger area compared to the less complex classes. Also, the flaring sunspots groups have higher number of spots and area compared to the ones that did not flare. The feature importance graphs for each of the model showed a global score for each one of the predictors and the local interpretation of a number of predictors proved the global importance's to hold true at an macro level. In general, a sunspot group with more complex characteristics, more number of spots and a greater area will have more chances of flaring compared to the one that is less complex, has lesser number of spots and has a smaller area. Some exception cases were also observed which can be treated as an outlier.

## 5.2 Model Results:

**RapidMiner:**

RapidMiner was used in order get an understanding of weather or not there exists a relationship between the predictor columns and the target columns. A number of different algorithms were trained upon the final data and the performance of each of the model was evaluated on the basis of number performance metrices like AUC, accuracy, f1 score etc. Table 4.2-4 shows the performance results of a number of metrices of all the models that were trained in RapidMiner. From the results it becomes clear that there do exists a certain relationship between our input variables and the target column. The data from the table shows that almost all the models seem to perform equally when creating a degree of separation between the flaring and the non-flaring sunspots groups. Random forest gave an area under the curve score (AUC) of 0.713, gradient boosted trees gave an AUC score of 0.715 and the deep learning model gave an AUC score of 0.734. The other machine learning models also gave similar AUC values when trained on this data, logistic regression – 0.72, svm – 0.54. Even though the scores for each of the model are almost similar, ensemble and deep learning models seem to be doing a good job at classifying the data into flaring and non-flaring classes. Since the ensemble and deep learning methods combine the predictive powers of a number of classifiers they were expected to perform better than the rest of normal machine learning models. The results obtained in RapidMiner helped to confirm that a study can be done to find correlation between the different predictor variables and since these results were used to get an initial benchmark score to improve upon in python and h2o.ai, no hyper parameter tuning were done on them to improve the results.

Both python and H2O.ai are platforms that provides easy implementation and evaluation methods for machine learning models. In both of them, ensemble (random forest and gradient boosted trees), deep learning (neural networks) and general purpose models (logistic regression and svm) are trained and scored on the evaluation metrices that were discussed in section 5 of chapter 3. Even though h2o is quite new and very robust it does not allow as much flexibility in hyper tuning as python does. The parameters used to train models in each of these platforms have been discussed in chapter 4 and the predictions generated from the best fit parameters were used to evaluate scores and compare models.

**H2O.ai:**

After training each of the models on different combination of parameters and finding the best fit parameters, the final models the final models are trained and then compared on the basis of the scores generated from them. For random forests the best fit parameters were 50 number of trees in a model with each tree having the maximum depth of 7, the AUC score and the mean squared error (MSE) that was generated from this model is 0.79 and 0.186 respectively. Gradient boosted trees, in this model the final parameters were 120 number of trees with a maximum depth of trees equal to 7. This model gave an AUC value of 0.84 and a MSE of 0.160, these values are better than what was observed in random forests. Neural nets on the other hand performed slightly better than the random forests but little worse than boosted trees, they gave an AUC score of 0.81 and a MSE of 0.41. Logistic regression model on the other gave an AUC value of 0.72. All three algorithms in study performed better than the traditional machine learning methods, logistic regression and svm. However, comparing the three models it becomes clear that the gradient boosted trees performed the best with a degree of separation score equal to 0.84, followed by deep-learning model with a score of 0.81 and finally random forests with a score of 0.79. Even though the scores are pretty much similar for all three models, gradient boosted trees performs slightly better on the unseen data and hence should be the model of choice for a sunspot classifier.

**Python:**

Since python's scikit-learn library offers more flexibility play around with the hyper parameter values, the model's trained in python were tested on a greater number of criteria's and hence most of them ended up performing better than they did in h2o.ai. The random forests model, the best fit parameters for this is 500 number of trees with a maximum depth of each tree equal to 10 and a gini-index criterion used to measure the quality of split in the model. This model gave an AUC score of 0.84 and MSE of 0.24. Gradient boosted trees gave an AUC of 0.88 and a MSE of 0.19, while the maximum depth of tress remained the same as random forests the number of estimators in the model were only 200 compared to the 500 in random forests, the learning rate used in this model is equal to 0.06 which is pretty high. For the best fit parameters in the neural network, the model was trained for 100 epochs where each epoch contained data in batch size of 10 with a relu activation function and 100 neurons in each of the two hidden layers in the model, an AUC score of 0.71 and a MSE of 0.28 was calculated for this model. Comparing these results to the SVM and logistic regression models which gave a low AUC value 0.72, the ensemble methods and the deep learning models performed better than the traditional machine learning methods. And as was seen in h2o.ai the gradient boosted models performed the best here too with a slightly improved degree of separation of 0.88 compared the 0.84 in h2o.ai

The RapidMiner results gave an understanding that a comparative study was possible and that there exists a relationship between the input predictors and the output. Although the methods under study did not significantly perform better than the traditional machine learning models like logistic regression and svm, they were able to create a greater degree of separation between the flaring and non-flaring classes. The models in h2o.ai were the first step in trying to build better models for the classification task. With the help of parameter optimization, the models were able to significantly improve the performance scores for different models with 0.84 AUC for gradient boosted trees and 0.81 AUC for neural networks, compared to the ones in RapidMiner where gradient boosted trees had an AUC 0.715 and 0.734 AUC for neural networks. The python programming language offers the highest level of flexibility when it comes to optimizing a model, therefore the models in python were tested on a greater number of parameters compared to the ones in h20.ai as a result the python gave better performance of the models on the test data. Out of the three models understudy the gradient boosted trees performed the best in all the three models with AUC score of 0.88 in python, 0.84 in h2o.ai and 0.715 in RapidMiner. The higher degree of separation makes the GBT the best algorithm for the task of classifying sunspots groups into flaring and non-flaring groups.

## Chapter 6 – Conclusion:

This work aims to compare the performance of various machine learning, ensemble and deep learning models at predicting the probability of whether or not a sunspot will produce a flare. For this study the data from two publicly available records, solar flare data corresponding to C, M, X class flares and sunspots data, from the National Geophysical Data Center (NGDC) is used. The data is processed using the algorithm given by (Qahwaji and Colak, 2007) and stored in two different datasets, one containing the data on spots that produced a flared and the other on spots that did not produce a flare. A general idea how on how the machine learning models will perform is gathered from implementing the various models in RapidMiner and checking the accuracy metrices that were produced. This result was then improved upon by implementing the same methods in python and h2o.ai using cross-validation and hyper-parameter tuning.

With this work, the results were improved upon a from what was set as a benchmark and it was proved that the ensemble and deep learning models, with the right parameterization were able to produce a better classification between the two different classes. Gradient boosted trees performed the best amongst all the three models in consideration, 0.88 (python) and 0.84 (h2o.ai), followed by random forests, 0.84 (python) and 0.79 (h2o.ai), and finally the deep learning model.

The practical findings of this research show that the eruptions of flares from a sunspot is dependent upon a number of factors. Area of a sunspot shows the highest correlation towards the eruptions of flares. From the Mcintosh classification, the penumbra class and zurich class contribute more towards the prediction than the compactness class. Finally, the magnetic class seemed to be the poorest predictor amongst all the parameters.

The quality of this work can be further improved by converting the categorical classes into their respective numerical values and make it efficient enough to be able to determine the intensity of a solar flare that will originate from a sunspot. For a future work this model can also be used to build a forecasting model that will forecast the occurrence of a solar flare within a time span of few hours to a day depending on the model and the data. The work can be further extended to add more predictors that can be calculated using statistical methods based on the work of (Song et al., 2009).

In this work a comparative study of various ensemble and deep learning based machine learning methods was performed in order to find the model that does the most effective job of separating the two classes from one another and also to get an understanding of which of the features contribute the most towards the final class prediction. The methods were run in three different environments, each having its own advantages over the other. This research was able to prove that gradient boosted tress were the ones that gave the best accuracy metrices amongst all the other models and that there exists a strong relationship between the morphological properties and the occurrence of solar flares.

# References:

Benvenuto, F., Piana, M., Campi, C., Massone, A.M., 2018. A Hybrid Supervised/Unsupervised Machine Learning Approach to Solar Flare Prediction. ApJ 853, 90. https://doi.org/10.3847/1538-4357/aaa23c

Colak, T., Qahwaji, R., 2007. Automated Prediction of Solar Flares Using Neural Networks and Sunspots Associations, in: Saad, A., Dahal, K., Sarfraz, M., Roy, R. (Eds.), Soft Computing in Industrial Applications, Advances in Soft Computing. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 316–324. https://doi.org/10.1007/978-3-540-70706-6\_29

Falco, M., Costa, P., Romano, P., 2019. Solar flare forecasting using morphological properties of sunspot groups. J. Space Weather Space Clim. 9, A22. https://doi.org/10.1051/swsc/2019019

Hands On Machine Learning with Scikit Learn Keras and Tensorflow.pdf, n.d.

Li, R., Wang, H.-N., He, H., Cui, Y.-M., Du, Z.-L., n.d. Support Vector Machine combined with K-Nearest Neighbors for Solar Flare Forecasting 8.

Li, R., Zhu, J., 2013. Solar ﬂare forecasting based on sequential sunspot data 10.

MclNTOSH, S., 1990. The classification of sunspot groups 17.

Opitz, D., Maclin, R., 1999. Popular Ensemble Methods: An Empirical Study. jair 11, 169–198. https://doi.org/10.1613/jair.614

Qahwaji, R., Colak, T., 2007. Automatic Short-Term Solar Flare Prediction Using Machine Learning and Sunspot Associations. Sol Phys 241, 195–211. https://doi.org/10.1007/s11207-006-0272-5

Qu, M., Shih, F.Y., Jing, J., Wang, H., 2003. [No title found]. Solar Physics 217, 157–172. https://doi.org/10.1023/A:1027388729489

Ribeiro, M.T., Singh, S., Guestrin, C., 2016. Model-Agnostic Interpretability of Machine Learning. arXiv:1606.05386 [cs, stat].

Song, H., Tan, C., Jing, J., Wang, H., Yurchyshyn, V., Abramenko, V., 2009. Statistical Assessment of Photospheric Magnetic Features in Imminent Solar Flare Predictions 25.

Wang, H.N., Cui, Y.M., Li, R., Zhang, L.Y., Han, H., 2008. Solar flare forecasting model supported with artificial neural network techniques. Advances in Space Research 42, 1464–1468. https://doi.org/10.1016/j.asr.2007.06.070

Yi, K., Moon, Y.-J., Shin, G., Lim, D., 2020. Forecast of Major Solar X-Ray Flare Flux Profiles Using Novel Deep Learning Models. ApJ 890, L5. https://doi.org/10.3847/2041-8213/ab701b

Yu, D., Huang, X., Wang, H., Cui, Y., 2009. Short-Term Solar Flare Prediction Using a Sequential Supervised Learning Method. Sol Phys 255, 91–105. https://doi.org/10.1007/s11207-009-9318-9

Yuan, Y., Shih, F.Y., Jing, J., Wang, H.-M., 2010. Automated flare forecasting using a statistical learning technique. Res. Astron. Astrophys. 10, 785–796. https://doi.org/10.1088/1674-4527/10/8/008