The Astrophysical Journal, 964:163 (17pp), 2024 April 1 https://doi.org/10.3847/1538-4357/ad276c

Investigating Performance Trends of Simulated Real-time Solar Flare Predictions: The Impacts of Training Windows, Data Volumes, and the Solar Cycle

Griffin T. Goodwin, Viacheslav M. Sadykov, and Petrus C. Martens
Physics & Astronomy Department, Georgia State University, Atlanta, GA 30303, USA

Received 2023 December 4; revised 2024 January 26; accepted 2024 February 6; published 2024 March 28

Abstract

This study explores the behavior of machine-learning-based flare forecasting models deployed in a simulated operational environment. Using Georgia State University's Space Weather Analytics for Solar Flares benchmark data set, we examine the impacts of training methodology and the solar cycle on decision tree, support vector machine, and multilayer perceptron performance. We implement our classifiers using three temporal training windows: stationary, rolling, and expanding. The stationary window trains models using a single set of data available before the first forecasting instance, which remains constant throughout the solar cycle. The rolling window trains models using data from a constant time interval before the forecasting instance, which moves with the solar cycle. Finally, the expanding window trains models using all available data before the forecasting instance. For each window, a number of input features (1, 5, 10, 25, 50, and 120) and temporal sizes (5, 8, 11, 14, 17, and 20 months) were tested. To our surprise, we found that, for a window of 20 months, skill scores were comparable regardless of the window type, feature count, and classifier selected. Furthermore, reducing the size of this window only marginally decreased stationary and rolling window performance. This implies that, given enough data, a stationary window can be chosen over other window types, eliminating the need for model retraining. Finally, a moderately strong positive correlation was found to exist between a model's false-positive rate and the solar X-ray background flux. This suggests that the solar cycle phase has a considerable influence on forecasting.

Unified Astronomy Thesaurus concepts: Space weather (2037); Solar flares (1496); Support vector machine (1936); Solar cycle (1487)

1. Introduction

Due to humanity's growing technological advancements over the past century, solar eruptive events have emerged as a significant threat to society and its infrastructure. Electromagnetic radiation, solar energetic particles, and coronal mass ejections produced during solar flares have the potential to interfere with radio communications, GPS, and power grids (Natras et al. 2019; Hudson 2021), which are crucial components to our everyday lives. Furthermore, these events pose considerable health risks to humans, in particular astronauts who are not shielded by Earth's magnetosphere and may receive increased doses of radiation. Considering these effects, the need for robust forecasting models that provide accurate and timely predictions of solar flares has become increasingly important. Traditional forecasting methods, such as those used by the National Oceanic and Atmospheric Administration (NOAA), have long relied on a blend of statistical analyses and human intuition (Crown 2012). However, given the advancements of artificial intelligence in recent years, there has been a gradual shift toward utilizing machine learning (ML) to automate and improve current forecasting capabilities. ML centers on training computers to make predictions on unseen data, given their previously acquired knowledge of some available data set (Florios et al. 2018). For flares, this can be physics-based parameters of active region (AR) vector magnetograms, extreme ultraviolet images of ARs, or even sunspot properties and McIntosh classifications (Li et al. 2007; Bobra & Couvidat 2015; Nishizuka et al. 2018). Since its initial application to space weather in the early 1990s (Camporeale 2019), ML has grown significantly, showing great promise within the community. However, despite this success, several notable issues continue to limit its implementation in operational forecasting:

1. ML models are commonly trained and tested using a random set of flaring and non-flaring data, which is not necessarily consistent with real-time forecasting. In an operational setting, predictions must be based solely on data available prior to the forecasted event. This raises the question: How do ML classifiers perform when utilizing chronological training and testing partitions? Sadykov & Kosovichev (2017), Nishizuka et al. (2018), and Leka et al. (2019a) have considered this idea through static training and testing windows; however, to the best of our knowledge, no studies have attempted to implement a dynamic temporal training strategy to improve operational forecasts.
2. Complicated ML algorithms are often considered black boxes, providing little insight into their predictive reasoning (Camporeale 2019). This makes it challenging for forecasters to rely on them confidently. Thankfully, relatively basic models exist that provide easily interpretable predictions. However, there is no guarantee that these models perform as well as their more complex counterparts. A previous study from Deshmukh et al. (2023) found that, for flare forecasting, ML models of different complexities were quite comparable, but a

similar investigation has yet to be done for a real-time forecasting environment.

3. ML models are frequently trained on all available data to maximize performance. However, this can result in a time-consuming training and hyperparameter optimization phase, which is not ideal for real-time forecasting. A middle ground between performance and run time likely exists, but the amount of data necessary to generate effective flare forecasts is currently poorly understood.
4. The performance of ML-based flare-forecasting models is heavily influenced by the selection of training data. Previous studies have shown that skill scores may vary significantly when training on different parts of the solar cycle (Wang et al. 2020). It is unclear whether these impacts can be mitigated through dynamic training windows.

The goal of this work is to thoroughly examine each of these concerns. To address **Problem** (**1**), we deploy a training and testing methodology that simulates a real-time predictive environment. We accomplish this through three training windows we label as stationary, rolling, and expanding. For **Problem** (**2**), we apply our training methodology to three different ML models of increasing complexity: decision tree, support vector machine, and multilayer perceptron. We then explore how performance scales with the number of magnetogram features used in a prediction. To tackle **Problem** (**3**), we investigate the impact of data volume on performance by implementing different stationary and rolling window sizes. Finally, to handle **Problem** (**4**), we explore the relationship between classifier performance and the solar background soft X-ray (SXR) flux. We use this as a probe to investigate if the solar cycle has an effect on real-time forecasts, as well as how this potential dependency interacts with the dynamic training windows. We would like to emphasize that the main goal of this work is not necessarily to produce the best-performing classifier, but rather to exhaustively examine the problems we have mentioned above.

The remaining sections of this work are organized as follows: Section 2 details the data we use to construct our forecasts. Section 3 describes the methodology used to tune, train, test, and analyze our ML models. Finally, Sections 4 and 5 highlight the results and conclusions of our study.

2. Data

In this section, we provide a description of the three key data sets employed in this work. Section 2.1 provides an overview of the Space Weather Analytics for Solar Flares database, while Section 2.2 briefly describes the data used to analyze the performance dependency on the solar cycle.

2.1. Space Weather Analytics For Solar Flares (SWAN-SF)

Georgia State University's Space Weather Analytics for Solar Flares (SWAN-SF) benchmark data set (Angryk et al. 2020a, 2020b) is a comprehensive, ML-ready collection of multivariate time-series samples extracted from ARs present during Solar Cycle 24 (2010 May–2018 August). For each AR, 24 physics-based features (see Angryk et al. 2020b, Table 1) are derived from photospheric vector magnetograms taken by the Solar Dynamics Observatory Helioseismic and Magnetic Imager (Scherrer et al. 2012). Throughout an AR's lifetime, time-series data are sliced into temporally successive overlapping files (offset by 1 hr), each containing 12 hours' worth of data, at a 12 minute cadence. Files are then labeled based on the strongest flaring event that occurs in the following 24 hr. We categorize flare strength using NOAA's logarithmic classification scale: A (weakest), B, C, M, and X (strongest). For this study, M- and X-class flares are labeled as flaring events, considering they have the greatest potential for societal impacts. Weaker flares, in addition to flare-quiet time series, are labeled as non-flaring events.

In total, there are 331,185 AR multivariate time-series files in SWAN-SF. Given this, along with the high dimensionality of each file, we decided to eliminate the contiguous temporal dimension of our data. This process not only allowed for easier integration with our proposed ML models, discussed in Section 3.1, but significantly reduced training and testing times, which is an important aspect to consider when deploying models operationally. By extracting the summary statistics (mean, median, standard deviation, maximum, and minimum) of each magnetic field parameter, within the 12 hr window, all files were reduced to a single point-in-time datum, with a dimension of 1 by 120. Any files containing columns with missing data were linearly interpolated before calculating their summary statistics, while files with empty columns were dropped altogether. To ensure that the data reflected a real-time forecasting scenario as much as possible, all ARs in the original SWAN-SF data set were retained. Ultimately, we were left with 330,169 data points: 6234 flaring and 323,935 non-flaring.

Interested readers who would like to learn more about the original data processing techniques for SWAN-SF may refer to Angryk et al. (2020b).

2.2. Geostationary Operational Environmental Satellite SXR Flux and Hale Classifications

To investigate potential correlations between a classifier's performance and the phase of the solar cycle, we utilized daily SXR flux data from Geostationary Operational Environmental Satellite (GOES) (1–8 Å channel) provided by Ali et al. (2024). A proxy for the solar SXR background flux was then determined by selecting the minimum X-ray flux measurement for each day during Solar Cycle 24. Additionally, we made use of AR Hale classifications provided by Marroquin et al. (2023), to study the frequency of complex ARs throughout the solar cycle. We then used these data in conjunction with each other to synthesize our results discussed in Section 4.3.

3. Methodology

The structure of SWAN-SF frames this forecasting problem as a binary classification task, with the ultimate goal of determining whether an AR will produce a $\geqslant$M-class solar flare within the next 24 hr. Previous work has shown that ML-based classifiers such as decision tree, logistic regression, random forest, support vector machine, and multilayer perceptron provide relatively reasonable forecasting performance when utilizing magnetogram feature sets (Yu et al. 2009; Yuan et al. 2010; Bobra & Couvidat 2015; Florios et al. 2018). In this particular study, we focus on the simulated real-time performance of three models: decision tree, support vector machine, and multilayer perception. This subset covers a wide gambit of complexities, ensuring we obtain robust results.

The following sections provide a basic overview of each model (Section 3.1), the methodology for data preprocessing,

Table 1
A List of Hyperparameters Used in the DT, SVM, and MLP Grid Search

ML Classifier	`scikit-learn` Grid Search Hyperparameters

Decision Tree	`criterion: [``gini'', ``entropy'']`
	`class_weight: [``balanced'']`
	`max_depth: [2, 3, 4, 5, 10, 20, 30, 40, 50, 100]`
	`min_samples_leaf: [1, 10, 20, 30, 40, 50, 100]`
	`min_samples_split: [2, 10, 20, 30, 40, 50, 100]`

Support Vector Machine	`kernel: [``rbf'']` `class_weight: [``balanced'']` `C: [0.0001, 0.001, 0.01, 0.1, 1, 10, 100, 1000]` `gamma: [scale, 0.0001, 0.001, 0.01, 0.1, 1, 10, 100, 1000]`
Multilayer Perceptron	`hidden_layer_sizes: [(50, 25, 12)]` `solver: [``adam'']` `activation: [``relu'']` `learning_rate: [``adaptive'']` `alpha: [0.0001, 0.001, 0.01, 0.1, 1, 10, 100]` `max_iter: [5, 10, 20, 30, 40, 50, 100, 200]`

Notes. The SVM `scale` hyperparameter for `gamma` uses the inverse of the number of features times the variance of the feature vector. See the `scikit-learn` API (https://scikit-learn.org/stable/modules/classes.html) for additional details on each parameter.

feature selection, and hyperparameter tuning (Section 3.2), the design of each training window (Section 3.3), the performance metrics used to analyze our results (Section 3.4), and our approach for studying the solar cycle dependence (Section 3.5).

3.1. Machine-learning Classifiers

Decision trees (DT) are a simple, yet effective, ML algorithm for classification. Their foundation stems from a series of feature-based inequalities, which guide an input to a prediction. The overall structure of this model is hierarchical in nature, consisting of interconnected nodes, children, and leaves. Each node contains a test that compares a particular feature to some threshold value. These nodes are then split into child nodes, each with their own thresholds, further subdividing the tree. Eventually, enough splits are made to obtain a leaf, which determines the prediction for a given input. Mathematically, DTs are constructed by minimizing the impurity of successive splits of the training data. This can be considered analogous to determining the feature and decision boundary that best separates two labeled distributions (Kingsford & Salzberg 2008; Kotsiantis 2013; Deshmukh et al. 2023). In this work, we focus on optimizing two key hyperparameters of our DT model: the tree depth and the number of training samples needed for a split/leaf node to occur. We also explore a variety of impurities (Gini and entropy), when splitting nodes (for more details, see Kingsford & Salzberg 2008; Kotsiantis 2013).

In its simplest form, the support vector machine (SVM) algorithm identifies a multidimensional hyperplane in feature space that maximizes the separation between labels. This hyperplane is then used to make predictions on input data. Typically, more complex, nonlinear structures are needed to separate labels adequately. Thus, kernels may be applied to map the feature space into a higher dimension. In this paper, we employ a radial basis function (RBF) kernel (Bobra & Couvidat 2015), which is influenced by two fundamental hyperparameters: C and γ. C is a penalty parameter for the misclassification of training data, where large values of C result in overfitting and low values lead to underfitting. On the other hand, γ is the "width" of the kernel. High values of γ increase the complexity of the decision boundary, while smaller values generate a smoother division, resulting in performance similar to that of a linear boundary.

Multilayer perceptrons (MLP) are a subclass of feed-forward neural networks containing a set of fully connected nodes (or neurons) across consecutive neuron layers. These nodes, which constitute the building blocks of the algorithm, possess a series of inputs, outputs, and weights that facilitate the creation of a nonlinear decision boundary. All data coming into a node are multiplied by its corresponding weight and summed together. The result is then transformed using an activation function and passed to the output, which connects to each node within the subsequent layer (Gardner & Dorling 1998; Deshmukh et al. 2023). Typically, MLPs have three stages: a single input layer, a single output layer, and some arbitrarily large hidden layer sandwiched in between. For binary classification tasks, the input layer contains the same number of nodes as the size of the feature space, the output layer contains two nodes, and the hidden layers may contain any number of nodes. In this study, after some trial and error, we settled on a three-stage hidden layer, with 50, 25, and 12 nodes. To optimize the node weights, we utilize Adam, a stochastic gradient descent algorithm (Kingma & Ba 2014). For our nonlinear activation function, we chose the rectified linear unit (ReLU), a reliable transformation used in most modern networks. Two key hyperparameters we consider in this work, which can be tweaked to improve performance, are α and the number of training iterations. α serves as the strength of the L2 regularization. If α is large, there is a high penalty for misclassification, leading to a greater likelihood of overfitting the training data. The number of training iterations is how often the weights of the MLP are updated. The larger this value, the more vulnerable the MLP is to overfitting.

To implement these models, we use Python's `scikit-learn` library (Pedregosa et al. 2011). This package provides excellent support for data preprocessing, feature selection, and hyperparameter tuning.

3.2. Data Preprocessing, Feature Selection, and Hyperparameter Tuning

Data preprocessing is a key aspect to consider when training ML models, as incorrectly formatted or inconsistent data can lead to significantly worse predictions. In particular, disagreement in feature scales (due to differences in units) can pose problems, because features with larger scales tend to be given additional weight. We address this problem by rescaling each training data set feature distribution to a mean of 0 and a variance of 1, using `scikit-learn`’s `Standard Scaler` module. This transformation is calculated using the following formula: $z = \frac{x-u}{s}$, where z is the transformed value, x is the input value, u is the mean of the training samples, and s is the standard deviation of the training samples. The transformation is then applied to the testing data set, using the same training values for u and s to ensure that no testing data bias is introduced into our predictions.

Feature selection is another crucial facet to include, as utilizing features with little predictive capacity will result in poor performance. To determine the optimal features to select, we employ `scikit-learns SelectKBest` module, which calculates the analysis of variance (ANOVA) F-value for each feature in the training data set. This univariate statistic provides an estimate for the separation of variances between two distributions (in this case flaring/non-flaring events). Narrow and widely spaced distributions will produce large F-values, while significantly overlapping distributions with large standard deviations will result in small F-values. The metric is mathematically defined in the following way (Bobra & Couvidat 2015):

$$F(i) = \frac{\left(\bar{x}_i^{+} - \bar{x}_i\right)^2 + (\bar{x}_i^{-} - \bar{x}_i)^2}{\frac{1}{n^{+}-1}\sum_{c=1}^{n^{+}}\left(\bar{x}_{c,i}^{+} - \bar{x}_i\right)^2 + \frac{1}{n^{-}-1}\sum_{c=1}^{n^{-}}\left(\bar{x}_{c,i}^{-} - \bar{x}_i\right)^2}, \quad (1)$$

where, for a given feature i, $\bar{x}_i^{+}$ is the average value across flaring events, $\bar{x}_i^{-}$ is the average value across non-flaring events, $\bar{x}_i$ is the average value across all events, n^{+} is the number of flaring events, and n^{-} is the number of non-flaring events. The k features with the highest F-values are then kept, as they have the best-separated distributions and thus are beneficial to use when training our models. Determining the appropriate value for k can be challenging, so several were tested: 1, 5, 10, 25, 50, and 120 (see Section 4.1). We apply this selection methodology to every new training data set, prior to undersampling (see Section 3.3). Ultimately, the main reason we settled on this feature selection approach over others is its proven success within SWAN-SF. Generally, F-values have been shown to provide reasonable insight into a feature’s importance (Yeolekar et al. 2021).

Last, to address hyperparameter tuning, a necessary step for maximizing predictive performance, we implement a grid search using `scikit-learn's GridSearchCV` module. This enables us to exhaustively test combinations of hyperparameters and select those that result in the best-performing model. For each training data set, a stratified group fivefold cross-validation was applied. This ensures that, within the training and testing folds, no data overlap between ARs, and a similar number of flaring and non-flaring events are present. Using the generated folds, a model for every possible combination of hyperparameters shown in Table 1 was tested. The model that produced the highest true skill statistic score (see Section 3.4) was then selected for application to the full training data set. Once again, we apply this process for all new data fed to the model.

3.3. Simulated Real-time Training Windows

To explore the performance of a classifier in an operational setting, we designed a simulated real-time environment centered on training and testing ML models chronologically throughout Solar Cycle 24. Training data were produced using three different dynamic temporal windows: stationary, rolling, and expanding (see Figure 1). The stationary window paradigm generates forecasts using a single set of data that is available before the *first* forecasting instance. This training data is always selected from the beginning of the solar cycle (2010 May onward). The rolling window paradigm generates forecasts based on data from a constant time interval before the currently observed forecasting instance. This is similar to the stationary window; however, the window now moves with the testing data. Finally, models trained using the expanding window paradigm utilize all available data before the currently observed forecasting instance.

The boundary conditions for a given window were defined to best emulate data acquired in real time. For a given window lower boundary date, denoted as X, all data with time-series start dates $\geqslant X$ were retained. For a given window upper boundary date, denoted as Y, non-flaring data whose 24 hr forecasting window ends $\leqslant Y$ were kept. Data instances with forecasting windows extending beyond this date were excluded, as operators would need to wait the full 24 hr to confirm an AR as non-flaring. In the case of flaring data, time series can instantly be labeled as a flaring event, once an M- or X-class flare occurs. Thus, data corresponding to flares that took place at times $\leqslant Y$ were kept, even if their 24 hr forecasting window had extended beyond the boundary.

For the testing data, windows were generated in sequential 3 month blocks starting 2012 January 1. The boundaries for the blocks were set to encompass all flaring data and any non-flaring time series whose 24 hr forecasting window end date fell within the 3 month block. Any testing blocks that did not contain flaring data were not considered in our analysis of true skill statistic and Heidke skill score in Sections 4.1 and 4.2. This includes the period between 2016 April and 2017 March, as well as any time after 2017 September.

For each training data set, an undersampling approach was applied to mitigate the effects of class imbalance. Within a given window, all flaring data were retained. However, non-flaring data were randomly sampled to match the number of flaring events, while preserving the original ratio of C-class to B-class to flare-quiet events. For example, let us consider a training window consisting of 119 X-class, 974 M-class, 5481 C-class, 5184 B-class, and 51,160 flare-quiet data. There are a total of 1093 flaring events, which we want to retain, and 61,825 non-flaring events, which we want to trim down. By calculating the ratio between the number of flaring and non-flaring events (1093/61,825) and multiplying it by the number of C-class, B-class, and flare-quiet events, we can determine the sample size required for each class to preserve its original ratio while adding up to the desired 1093 non-flaring events. Consequently, when applying this technique to the previous example, we get 97 C-class, 92 B-class, and 904 flare-quiet events. Generally, this approach has proven to be successful when training and testing with SWAN-SF (Ahmadzadeh et al. 2021).

Finally, to investigate how performance scales with data volume, a series of stationary and rolling window sizes (5, 8, 11, 14, 17, and 20 months) were tested. For the stationary windows, data were selected starting from 2010 May up until the added

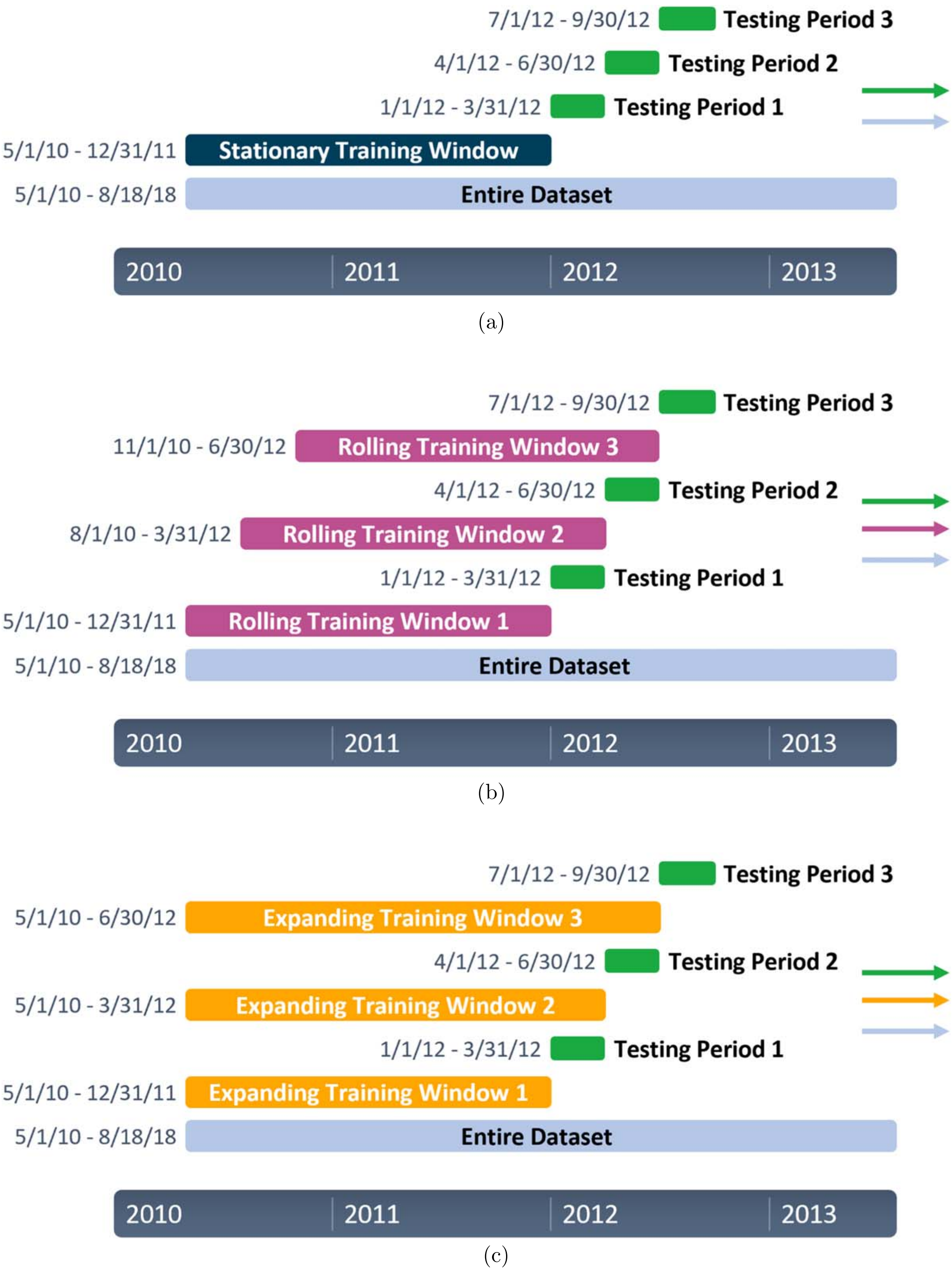


Figure 1. (a) An example of the stationary training window methodology. Each model is trained on a set portion of the data at the beginning of the solar cycle (in this example, it is the first 20 months). Model performance is then analyzed in consecutive 3 month blocks after training. (b) An example of the rolling training window methodology. Each model is trained similarly to the stationary window; however, the window now moves with the testing blocks. (c) A depiction of the expanding window methodology. Here, the models are trained using the entire available data set, prior to the forecasting instance. The arrows in each figure emphasize the temporal continuation of the training windows, testing windows, and the data set itself.

window size. For the rolling windows, data were selected between the testing window start date and extended into the past by the rolling window size. For each model, a total of three trials were run, each with different randomly undersampled non-flaring data to ensure robust results. Models with training data lacking flaring events were disregarded, and instead the previously available trained model would be used. This was only pertinent to the rolling window.

3.4. Performance Metrics

To evaluate the performance of each classifier, the true skill statistic (TSS) and Heidke skill score (HSS_2) were calculated for every 3 month testing block. These metrics are defined in the following way (Bobra & Couvidat 2015):

$$\mathrm{TSS} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} - \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}} \tag{2}$$

$$\mathrm{HSS}_2 = \frac{2 \times [(\mathrm{TP} \times \mathrm{TN}) - (\mathrm{FN} \times \mathrm{FP})]}{(\mathrm{TP} + \mathrm{FN}) \times (\mathrm{FN} + \mathrm{TN}) + (\mathrm{TP} + \mathrm{FP}) \times (\mathrm{FP} + \mathrm{TN})}, \tag{3}$$

where TP = true positives (the number of correctly predicted flaring events), TN = true negatives (the number of correctly predicted non-flaring events), FP = false positives (the number of non-flaring events predicted as flaring), and FN = false negatives (the number of flaring events predicted as non-flaring). TSS ranges from −1 to +1, with a score of 0 reflecting a classifier that makes random or purely positive/negative forecasts, a score of −1 reflecting a classifier that is always wrong, and a score of +1 reflecting a perfect classifier (Ahmadzadeh et al. 2021). This metric is particularly advantageous, as it is unbiased toward the class-imbalance problem prevalent within flare forecasting (Bobra & Couvidat 2015). When using TSS, one must keep in mind that two models with the same score do not necessarily produce an identical number of true positives and true negatives. This is because the metric is dependent on the balance of the true-positive rate ($\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}$) and the false-positive rate ($\frac{\mathrm{FP}}{\mathrm{FP}+\mathrm{TN}}$), which can be individually tweaked to achieve the same score (Ahmadzadeh et al. 2021).

Like TSS, HSS_2 ranges from −1 to +1 and provides an insight into a model's improvement over a random forecast. However, the minimum score is now dependent on the class-imbalance ratio. As it reaches 1:1 (an equal number of flaring and non-flaring events), the lower boundary approaches −1 (Ahmadzadeh et al. 2021). A score of 0 is equivalent to a random classifier, a negative score is representative of a classifier that performs worse than random, and a score of +1 reflects a perfect classifier. We have selected this definition of the Heidke skill score over the original (HSS_1), highlighted in Bobra & Couvidat (2015), as it tends to be less sensitive to the effects of class imbalance. Nevertheless, compared to TSS, both definitions are significantly more susceptible, with scores decreasing as the class-imbalance ratio increases (see Figures 2 and 4 in Bobra & Couvidat 2015 and Ahmadzadeh et al. 2021 for an illustrative example). Overall, both metrics are widely used in the community, enabling others to make comparisons to this work, provided that they apply methodology similar to that shown here.

3.5. Dependency on the Solar Cycle

Finally, to highlight performance dependencies on the solar cycle, we investigate the interplay between a model's false-positive rate ($\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP}+\mathrm{TN}}$) and the solar SXR background flux. We approach this by calculating the Spearman correlation between these parameters for each model utilizing 25 features. All window types, window sizes, classifiers, and trials were included. Following this, we back up our results by performing a study on how FPR is influenced by the largest flare class generated by an AR, as well as how the frequency of complex ARs changes throughout the solar cycle.

4. Results and Discussion

In the following sections, we discuss the results of our work in detail. In Section 4.1, we explore the effects of feature selection on model performance and highlight the magnetogram features most frequently chosen in our forecasts. In Section 4.2, we compare model performance between the different window types and investigate how the temporal sizes of the stationary and rolling training windows affect our predictions. Finally, in Section 4.3, we determine a correlation between the solar SXR background flux and the FPR.

4.1. Impacts of Feature Selection

The dimensionality of the feature space significantly influences the training time and complexity of a model. In an operational environment, unnecessary delays and complications must be avoided. Thus, we explore how the number of features selected for a given model affects performance, with the hope of establishing a baseline feature requirement for forecasts utilizing point-in-time magnetogram data. Figure 2 summarizes our results. The columns highlight a particular skill score: TSS (left) and HSS_2 (right), while the rows correspond to our three tested classifiers: DT (top), SVM (middle), and MLP (bottom). The radar plots are divided into six sections, one for every feature count tested (1, 5, 10, 25, 50, and 120 features). Within each wedge, the radial extent of the bars denotes the skill score for a given feature set and window type (color), averaged across all available 3 month testing blocks (2012 January 1–2016 March 31, 2017 April 1–September 30) associated with the 20 month stationary and rolling windows. This 20 month window was selected to remove any dependencies on data volume, which will be explored in Section 4.2.

At first glance, we find that TSS and HSS_2 scores tend to increase as more features are included in a forecast. This is expected, given that a higher-dimensionality feature space offers additional means to distinguish between the flaring and non-flaring distributions. However, it is rather surprising that, for a given classifier and window type, skill scores improve on average by only 0.035 when jumping from 1 to 120 features. To check whether these improvements are statistically significant, we can compare the absolute difference between the two feature scores ($|\bar{X}_{120} - \bar{X}_1|$) and their combined standard errors ($\sqrt{\sigma^2_{\bar{X},120} + \sigma^2_{\bar{X},1}}$). When we do this, we find that 88.8% of the improvements have a larger absolute difference than their combined errors, implying that they are statistically significant measurements at a 1σ or 68% confidence interval. If we extend this to 2σ, we discover that over 61% of the improvements are statistically significant at a 95% confidence interval. This suggests that, in general, our observations are meaningful and not simply due to the uncertainties in our data. However, the general similarity between scores still warrants further investigation.

To explore this topic in more detail, we examine the features that are typically chosen for a forecast, the flaring and non-flaring populations of those features, and the correlations between them. Figure 3 depicts the average normalized F-value for the 25 highest-scoring features. This metric can be considered a proxy for the selection frequency of a particular parameter. To calculate this measure, we first determined the

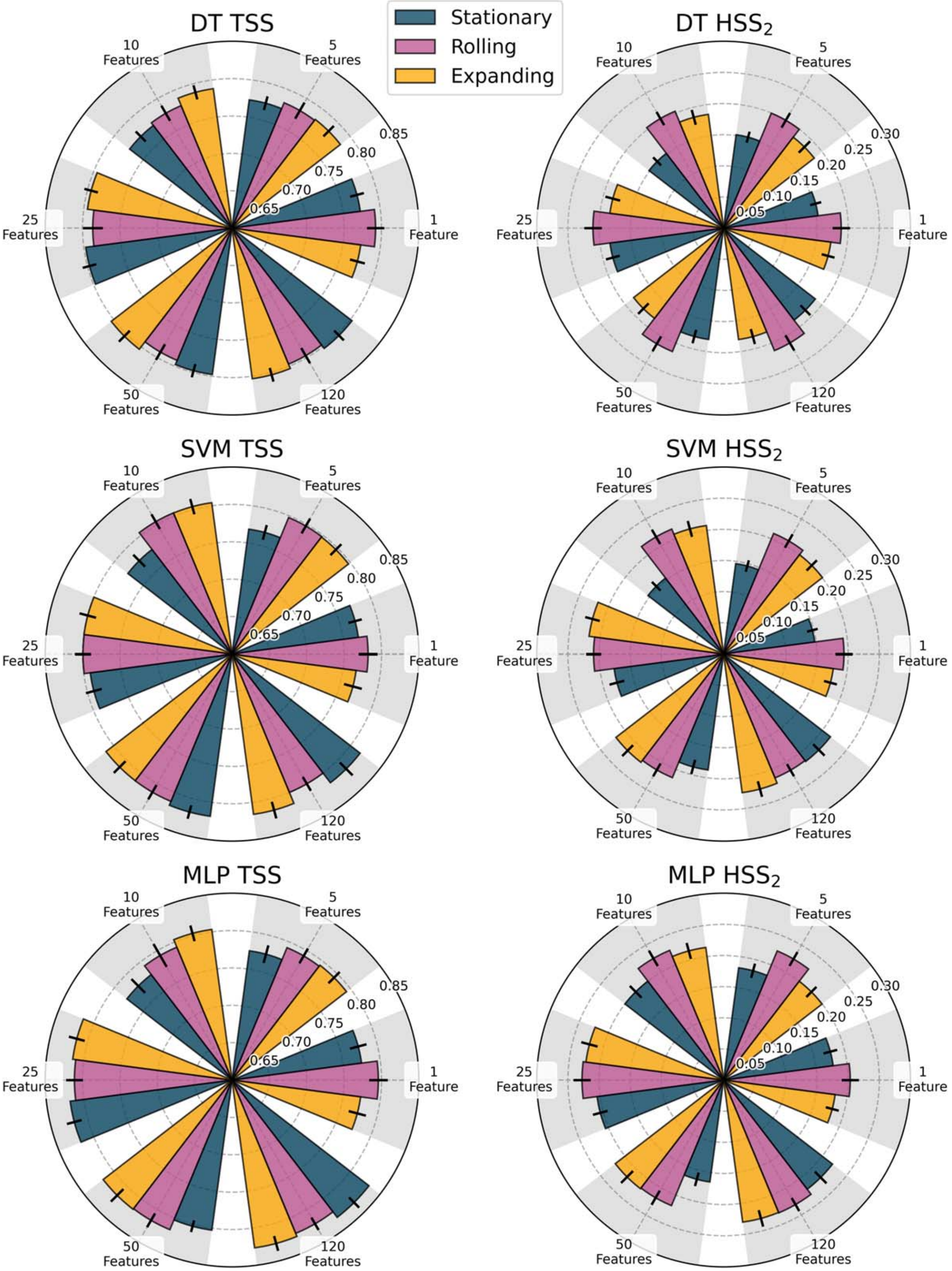


Figure 2. Average TSS and HSS_2 scores for the DT, SVM, and MLP classifiers with varying feature counts (1, 5, 10, 25, 50, and 120) and window types. The error bars illustrate the standard error on the mean. Note: These results were obtained using stationary and rolling windows of 20 months.

F-value of each feature across all stationary, rolling, and expanding training data sets. For a particular data set, all *F*-values were normalized with respect to the highest achieved *F*-value. The scores for each feature were then averaged over all data sets and finally organized in decreasing order. It should be noted that, even though we display the averages for individual window types, the order shown is solely based on the average across all training data sets. Because there is only one instance of the stationary window, but 19 instances (one for each new testing data set) for both the rolling and expanding windows, the stationary window only provides a small contribution to this order. From the figure, it is evident that

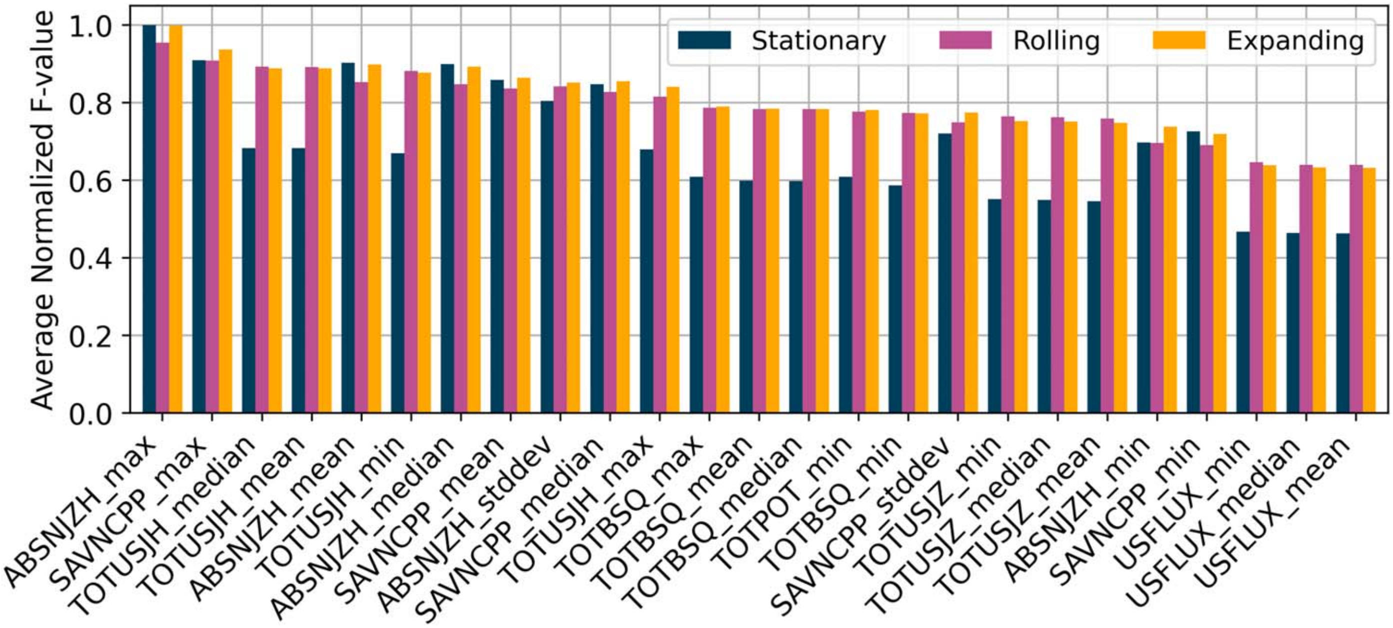


Figure 3. The average normalized F-value for the 25 highest-scoring features. This metric can be thought of as a proxy for selection frequency. Features with values close to 1 tend to be those with the highest-scoring F-value (and thus more likely to be chosen) in a given training window.

the summary statistics from only a few magnetogram parameters tend to be chosen for a given forecast: ABSNJZH (absolute value of the net current helicity), SAVNCPP (sum of the absolute value of net current polarity), TOTUSJH (total unsigned current helicity), TOTBSQ (total magnitude of the Lorentz force), TOTPOT (total photospheric magnetic free energy density), TOTUSJZ (total unsigned vertical current), and USFLUX (total unsigned flux). These features align well with those highlighted in other work within the field (Bobra & Couvidat 2015; Yeolekar et al. 2021; Zhang et al. 2022).

For a more comprehensive look, we have plotted the flaring and non-flaring distributions of the highest-ranking statistics for these features (see Figure 4). These figures reveal an overarching similarity between the distributions, with each of them having a right-skewed non-flaring and left-skewed flaring population. While significant overlaps exist, demonstrating the difficulty of flare forecasting, a separation between the medians of these populations can still be resolved. This provides enough distinction to make reasonable forecasts utilizing even a single feature (most frequently ABSNJZH_max), which explains the relatively high TSS scores we have obtained.

Calculating the Spearman correlation between the seven features in Figure 4, we find that a strong positive correlation exists between all of them (see Figure 5). Extending this analysis to all 25 features in Figure 3, it comes as no surprise that a similar trend is found, with all unique correlations being $\geqslant$0.72 and 80% of them being $\geqslant$ 0.90. Though correlation does not necessarily mean that two features are not complementary (Guyon & Elisseeff 2003), we believe that this could still be a plausible explanation for our results. Highly correlated features often provide similar information about the target class. Therefore, combining them will result in only minor improvements in the separability of the population. In our case, the 25 most important features are highly correlated, which may explain why the performance from 1 to 25 features is fairly comparable. Beyond 25, the additionally included features have diminishing F-values, making it significantly more difficult to separate between flaring and non-flaring events (at least in a one-dimensional sense). This may explain the only marginal performance improvements found in this range.

Shifting our focus to individual window types, we find that the rolling window consistently matches or outperforms the stationary and expanding windows, particularly when utilizing only 1 or 5 features. We speculate that this may be a consequence of the window's ability to capture the current flare occurrence rate. On a large scale, performance differences between the three window types are relatively minimal. This is unexpected given that, during the latter half of the solar cycle, the expanding window has access to significantly more flaring data than the other windows (see Figure 6). This suggests that, with a sufficient amount of data, a stationary classifier may be chosen over other window types. This not only saves time but dramatically reduces the difficulty of implementing an operational model. Of course, these results utilize a relatively large training window. Utilizing a smaller stationary or rolling window may not produce the same results. We explore this further in Section 4.2.

Finally, comparing results across the three tested classifiers, it becomes evident that MLPs yield the best TSS and HSS_2 scores, regardless of the number of features or window type selected (see Figure 7). Given the algorithm's complexity, this is expected. However, the narrow difference between the skill scores of all three classifiers is rather surprising. This clearly suggests that easily interpretable models, such as DTs, may be a viable alternative to more complicated models when provided with a 20 month training window. We investigate this trend for different window sizes in Section 4.2.

4.2. Impacts of Training Window Size

In addition to feature selection, data volumes are critical to producing effective flare-forecasting models. Without the proper amount of training data, ML algorithms fail to capture an adequate decision boundary, which can significantly degrade performance. Because data volumes can vary during operational deployment, it is essential to explore how skill scores are affected by this aspect

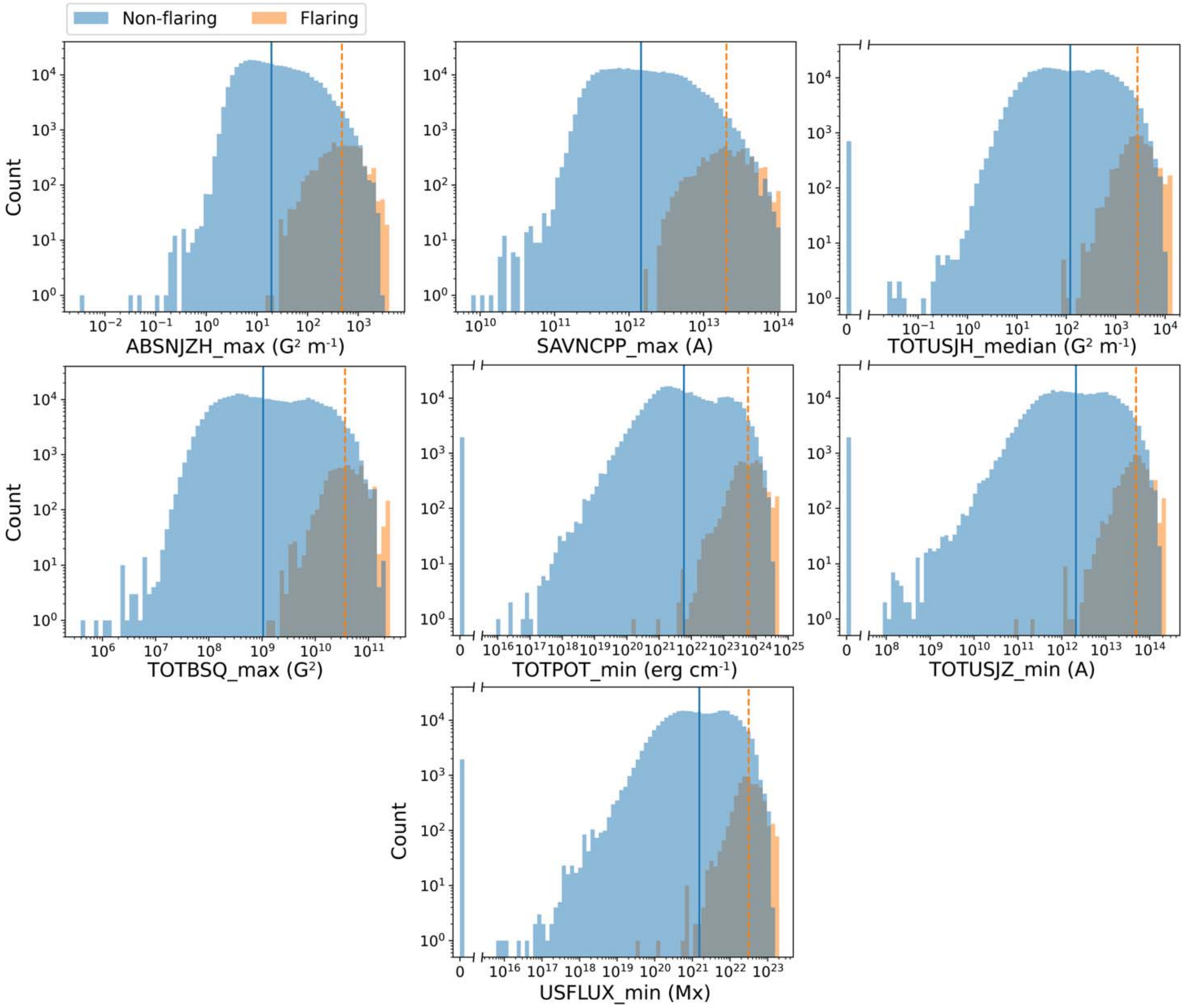


Figure 4. The flaring and non-flaring distributions for ABSNJZH_max (the maximum absolute value of the net current helicity), SAVNCPP_max (the maximum sum of the absolute value of net current polarity), TOTUSJH_median (the median total unsigned current helicity), TOTBSQ_max (the maximum total magnitude of the Lorentz force), TOTPOT_min (the minimum total photospheric magnetic free energy density), TOTUSJZ_min (the minimum total unsigned vertical current), and USFLUX_min (the minimum total unsigned flux) over the entire SWAN-SF data set. Distributions are plotted on a log–log scale. Bins containing zeros are plotted before the break in the *x*-axis. The solid blue line indicates the median of the non-flaring distribution. The dotted orange line indicates the median of the flaring distribution.

and how these restrictions interact with our custom training windows. Figure 8 summarizes our results. Once again, each column highlights a specific skill score, and each row a particular model. The radar plots are divided into six sections, one for every stationary and rolling window size tested (5, 8, 11, 14, 17, and 20 months). Within each wedge, the radial extent of the bars denotes the skill score for a given window size and type, averaged across all testing data (2012 January 1–2016 March 31, 2017 April 1–September 30) associated with the 20 month stationary and rolling training windows. This ensures that our findings are comparable across window sizes. Because the expanding window utilizes all data prior to the forecasting instance, its scores are the same across all wedges.

First, examining the general skill score trends, we find that the effects of window size on TSS and HSS_2 are dependent on the classifier and window type selected. For certain combinations, such as the DT with rolling window, removing training data results in a steady decline in performance, as one might expect. However, we find that this is not universal, with a majority of scores being completely uncorrelated with one another. For example, the stationary SVM TSS is larger for the 5 month window than the 20 month window, even though it has significantly fewer flares. This hints that there may be some underlying limitations to our data set, which we suspect are imposed by our methodology. When calculating the summary statistics of each time series, we remove potentially significant knowledge related to the dynamics of an event. This gives us a lighter data set that is easier to work with but may not be as informative. It is apparent that our models are able to capture some important aspects needed to predict flares, as they achieve

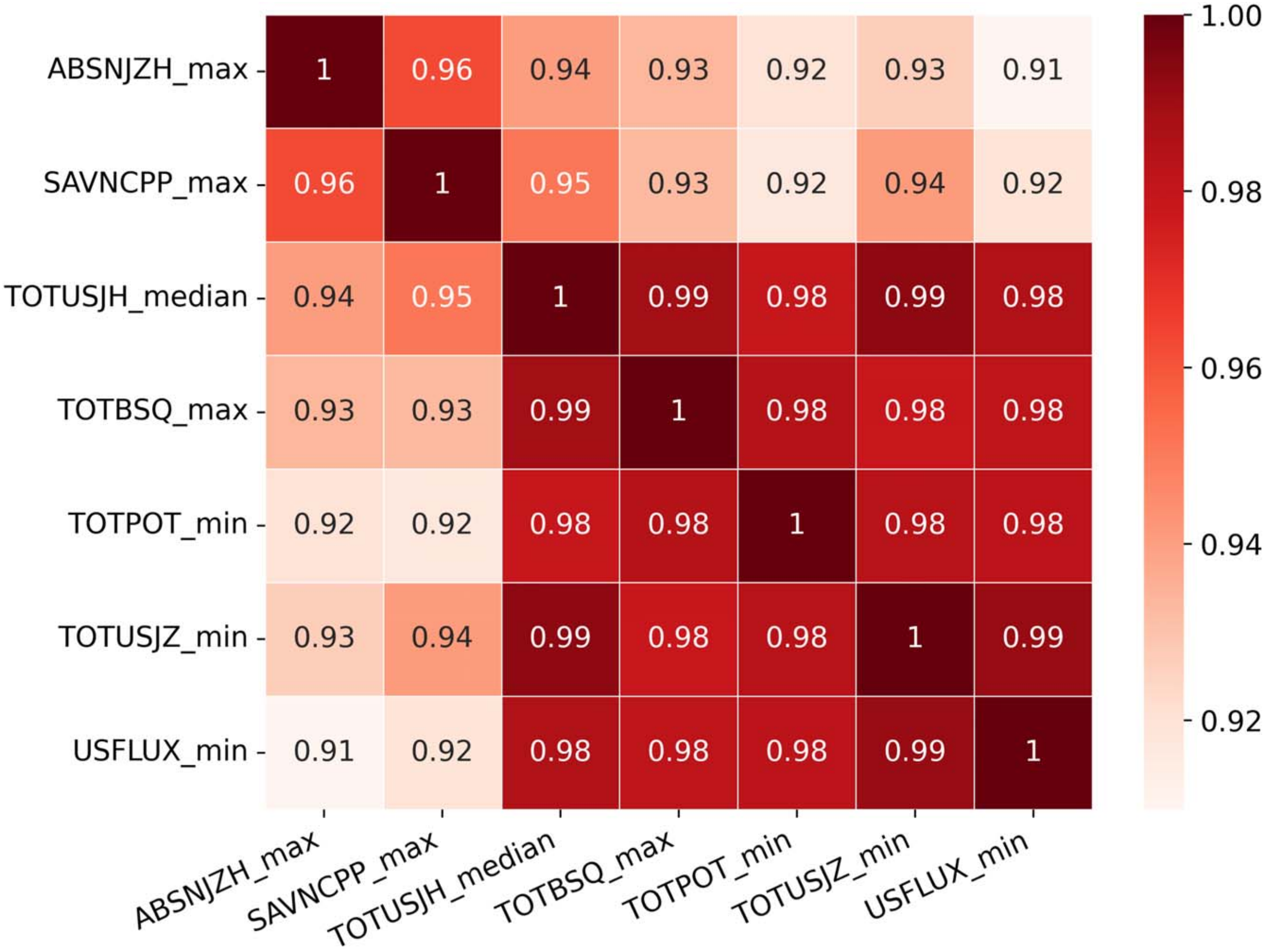


Figure 5. A Spearman correlation heatmap for the features selected in Figure 4.

fairly high skill scores, but with additional data, this can only improve so much. Without more exhaustive features, which can be taken advantage of by our ML models, increasing data volumes will not provide enough new information about the flaring population to significantly affect performance. A potential solution to this problem is to train models utilizing the entire time series, which has been shown to improve skill scores in SWAN-SF (Ji et al. 2020). However, it remains to be seen how, or even if, this would affect our data volume results. Finally, a recent study has shown that magnetogram data alone does not provide significant improvements over human-based forecasting (Leka et al. 2019b). This hints that there may be some inherent simplicity to magnetogram data itself, which limits its predictive capability and contributes to the findings shown here.

Taking a deeper dive into our results, we find that the stationary window almost always produces better TSS but noticeably worse HSS_2 scores than the rolling window, when the window size is less than 20 months. Because the stationary window covers the beginning of the solar cycle, where the number of flaring events is low, models will be biased toward capturing each flaring event in the training data. This is because missing a single flare has a large impact on the true-positive rate ($\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}$) and in turn the TSS score, which we are attempting to maximize when training the model. This leads to the stationary window producing fewer false negatives and more false positives while testing, which has less of an effect on TSS than HSS_2.

Finally, when comparing performance across classifiers (see Figure 9), we again find that MLPs yield the best TSS and HSS_2 scores by only a small margin. SVMs and DTs follow closely behind, even occasionally outperforming MLPs in select window types, sizes, and skill scores. This extends our conclusion made in 4.1, that less complex models can be reliably used in place of more sophisticated algorithms, to varying data volumes. However, we find that, when window sizes get too small, one must be cautious. The 8 month stationary DT window produces dreadful TSS and HSS_2 scores, which we suspect may be a consequence of the algorithm itself. It is well known that DTs tend to struggle with instabilities and under/overfitting, with small changes in their training data producing vastly different trees (Li & Belford 2002). These effects, compounded by the fact there is only a small amount of training data, lead to a higher chance of producing a decision boundary that does not accurately separate the entire flaring population in the testing data set.

4.3. Dependency on the Solar Cycle

Finally, to investigate the impact of the solar cycle on model performance, we explore the relationship between FPR and a proxy for the SXR background flux (the minimum daily GOES flux value). Figure 10 summarizes our results. Here, we find

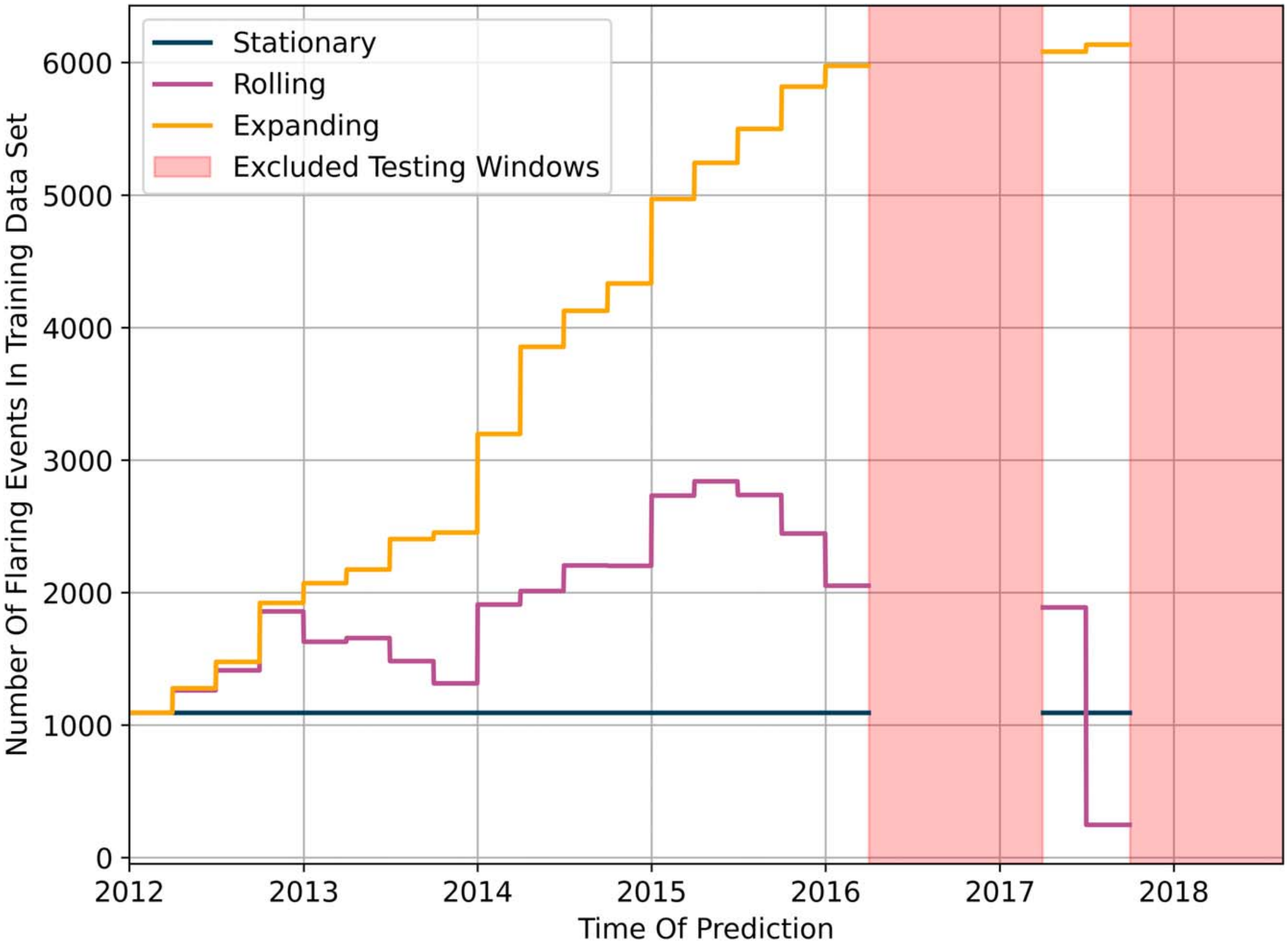


Figure 6. The number of flaring events in the 20 month stationary, rolling, and expanding windows as time progresses. The red regions illustrate periods where no testing windows exist, so no model was trained.

that a moderately strong positive correlation (mean of $\rho = 0.570$) exists for the conglomeration of our 25 feature trials (see the *All Data* label in Figure 10). This indicates that, as we reach solar maximum (when background levels are high), the FPR also increases. We find that models utilizing the rolling window or the MLP classifier tend to be more susceptible to this trend. Interestingly, no window type or classifier is impervious to this correlation.

To explore this further, we then examine how the largest flare class generated by an AR influences its FPR. To accomplish this, we first divide our ARs into flaring groups dependent on the strongest event they produce within their lifetime (X, M, C, B, or A/flare-quiet). Any point-in-time data associated with a particular AR are placed within the same group. We then recalculate the FPR for the data in each AR category and average our findings across all models utilizing 25 features. Table 2 presents our results. It is evident that ARs producing M- or X-class flares generally have elevated FPRs in comparison to ARs generating weaker flares. This is somewhat expected, given that B- and C-class flares, as well as flare-quiet periods, occur intermittently throughout flaring episodes, which can be challenging to detect. ARs may appear to have high magnetic activity over the previous 12 hr (in comparison to a typical non-flaring event) but do not end up flaring. This, of course, leads to significantly more false-positive predictions for these ARs.

Building on this, we then consider the frequency of complex ARs (those more likely to produce M- and X-class flares) throughout the solar cycle. In Figure 11, we plot the ratio of ARs with a Hale classification $>\beta$ (this includes γ, $\beta-\gamma$, δ, $\beta-\delta$, $\beta-\gamma-\delta$, and $\gamma-\delta$) to the total number of ARs, binned monthly for Solar Cycle 24. Here, we find that more complex ARs have a higher likelihood of existing during the peak of the solar cycle (near 2014) compared to the beginning or end. Tying this back to our findings from Table 2: if the probability of having a more complex AR is higher during the peak of the cycle, and ARs producing stronger flares tend to have larger FPRs, then there is reason to believe that the FPR will increase with background SXR flux. Of course, it is important to note that our forecasts are based solely on magnetic field parameters, with no direct relationship to the background SXR flux. Thus, we would like to emphasize that this result is merely a statistical observation rather than a causal relationship.

5. Summary and Conclusions

In this study, we focused on producing a simulated real-time prediction environment, which can be used as a test bed to analyze how a variety of classifiers, features, data volumes, and the solar cycle impact operational performance. From this work, we have identified the following key results:

1. Across all window types, the most frequently chosen magnetogram features are ABSNJZH (absolute value of the

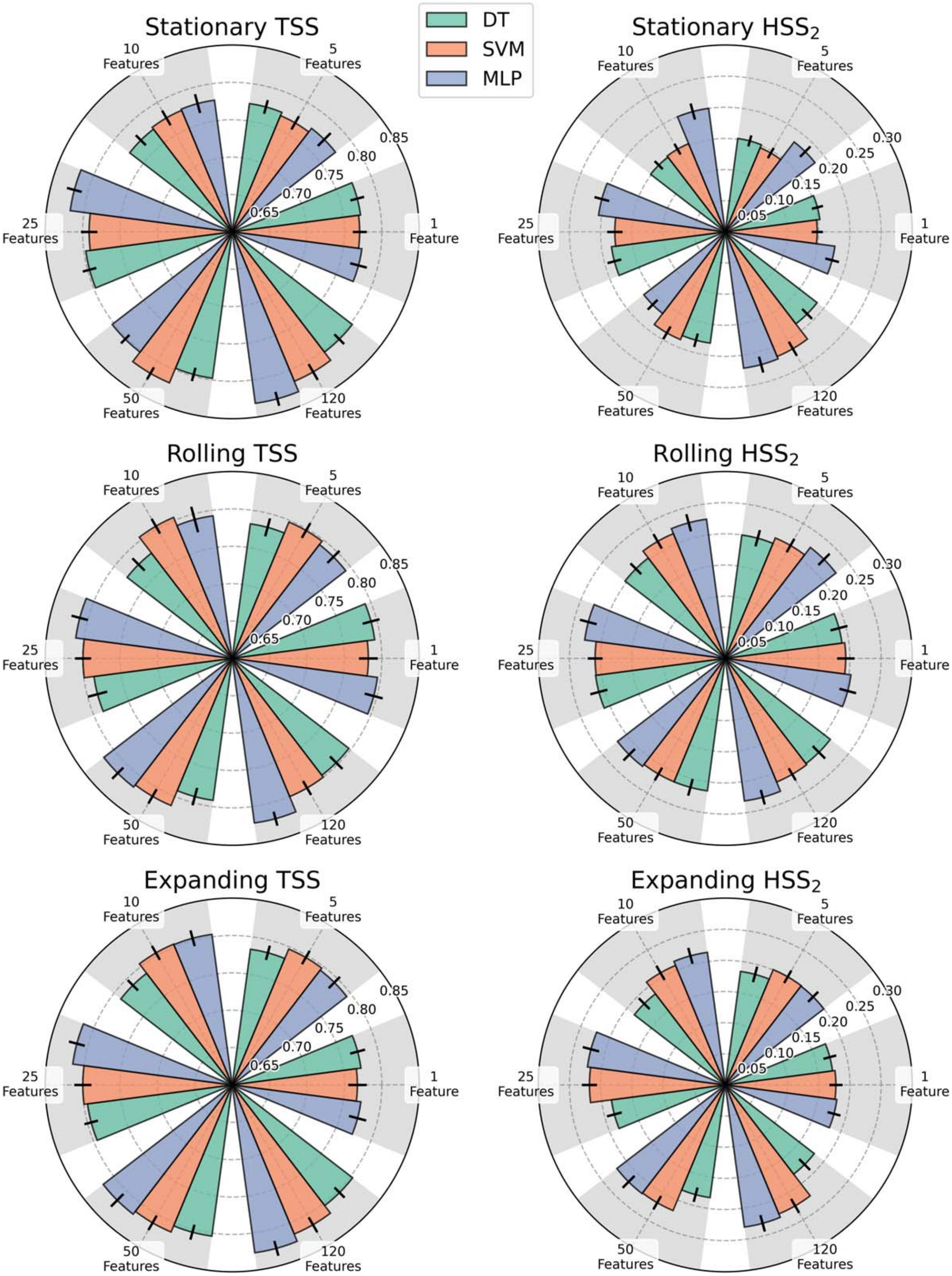


Figure 7. Average TSS and HSS$_2$ scores for the DT, SVM, and MLP classifiers with varying feature counts (1, 5, 10, 25, 50, and 120) and window types. The error bars illustrate the standard error on the mean. This plot is similar to Figure 2, except that a comparison is now being made between classifiers instead of window type. Note: These results were obtained using stationary and rolling windows of 20 months.

net current helicity), SAVNCPP (sum of the absolute value of net current polarity), TOTUSJH (total unsigned current helicity), TOTBSQ (total magnitude of the Lorentz force), TOTPOT (total photospheric magnetic free energy density), TOTUSJZ (total unsigned vertical current), and USFLUX (total unsigned flux). This corresponds well with results from other papers within the field (Bobra & Couvidat 2015; Yeolekar et al. 2021; Zhang et al. 2022).

2. The number of magnetogram features used to make a prediction does not have a significant effect on TSS or

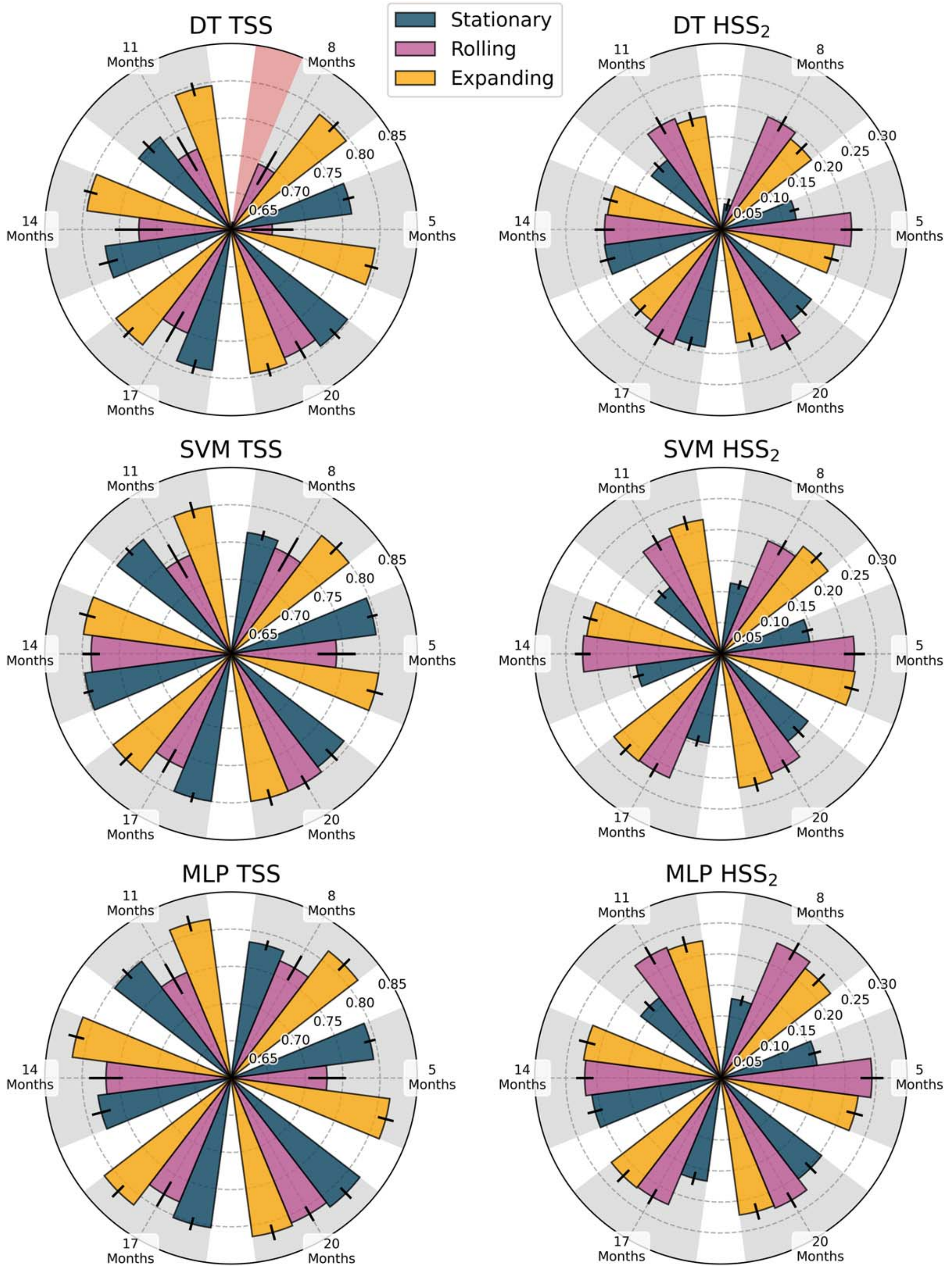


Figure 8. Average TSS and HSS_2 scores for the 25 feature DT, SVM, and MLP with varying stationary and rolling window sizes (5, 8, 11, 14, 17, and 20 months). Naturally, the skill scores for the expanding window are the same across different window sizes. They are included for reference. The error bars illustrate the standard error on the mean. Note: the 8 month stationary DT (the red wedge) has an average TSS score of 0.26 ± 0.05.

HSS_2 scores. Only a marginal increase in performance is observed as additional features are included in a forecast. We believe this may be an outcome of the highly correlated nature of our features.

3. When utilizing a 20 month stationary or rolling window, performance is generally comparable to the expanding window. Only a minor decrease in performance is observed for the stationary and rolling windows when

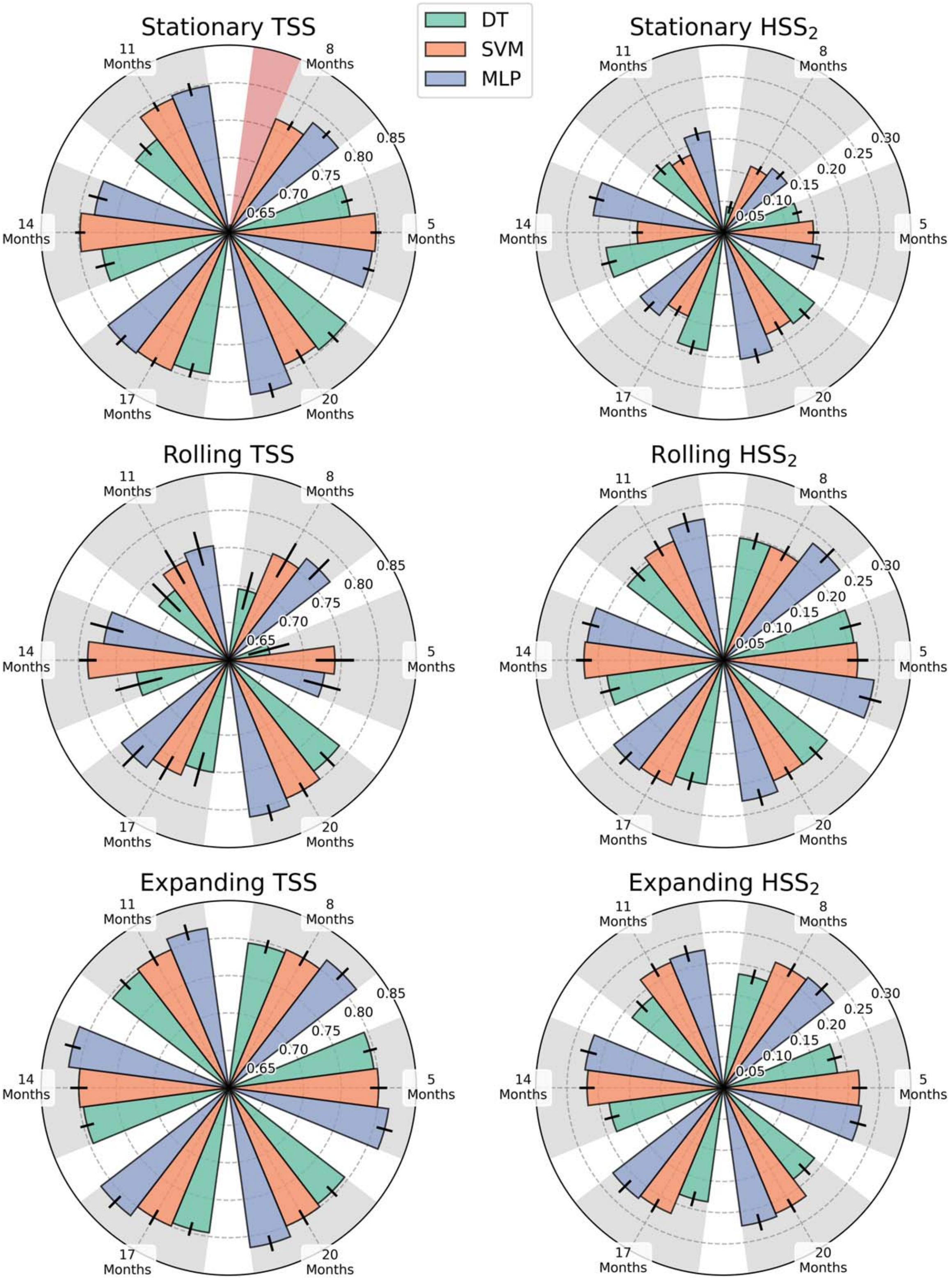
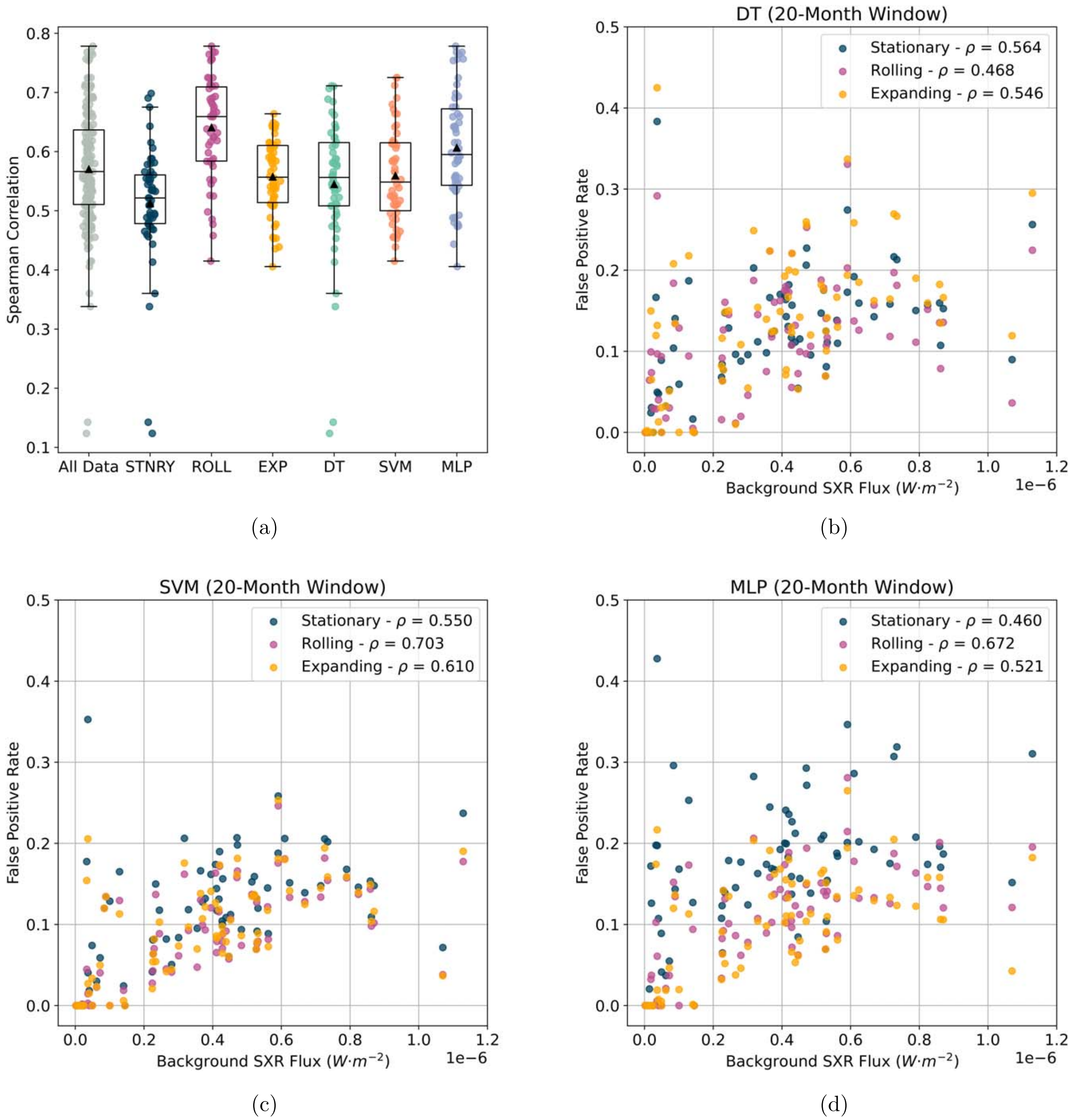


Figure 9. Average TSS and HSS_2 scores for the 25 feature DT, SVM, and MLP with varying stationary and rolling window sizes (5, 8, 11, 14, 17, 20 months). The error bars illustrate the standard error on the mean. This plot is similar to Figure 8, except a comparison is now being made between classifiers instead of window type. Note: the 8 month stationary DT (the red wedge) has an average TSS score of 0.26 ± 0.05.

their size is reduced. This suggests that, provided with a sufficient amount of data, a stationary classifier can be chosen over other window types, removing the need for retraining. We believe this to be a consequence of our methodology or potentially an inherent simplicity of the magnetogram data itself.

4. Simple and interpretable machine-learning classifiers, such as decision trees, provide skill scores similar to those of more complex models.
5. A moderately strong positive Spearman correlation exists between a model's false-positive rate and the background soft X-ray flux. We hypothesize that this is a consequence



(a)

(b)

(c)

(d)

Figure 10. (a) Boxplots of the Spearman correlation between the FPR and the background SXR flux. Plots are made for the entire collection of results, window types, and classifiers. Results are shown only for models that utilize 25 features. A swarm plot is overlaid to emphasize the distribution of data. The triangles indicate the mean of the distributions. ((b), (c), and (d)). Scatter plots of the FPR vs. background SXR flux for single trials of the 20 month DT, SVM, and MLP models. The correlation for each window type is given by ρ. The FPR and background SXR were binned monthly to reduce noise due to daily fluctuations.

> of highly complex active regions (those more likely to produce M- and X-class flares) appearing more frequently during the peak of the solar cycle. From our analysis, we observed that these active regions tend to be accompanied by larger false-positive rates.

Overall, we can conclude that, for operational forecasts utilizing point-in-time magnetogram data, the number of features, window size, window type, and classifier used have a minimal impact on performance, at least for those we tested.

Regarding future studies, there are numerous paths we can explore. First, it may be valuable to investigate whether utilizing temporally dependent features, such as the time-series derivative of a parameter, has any impact on the forecasting results shown here. These descriptive statistics could give a model better insight into how an active region is growing/decaying over time, which may lead to improved performance. However, recent work by Nishizuka et al. (2017) found that these features are ineffective on timescales less than 24 hr. This indicates that we would likely need to extend our 12 hr observation window to benefit from them. A better alternative would be to train models directly on the time-series data, rather than the point-in-time summary statistics. This could be accomplished through more complicated deep-learning algorithms such as long short-term memory (LSTM)

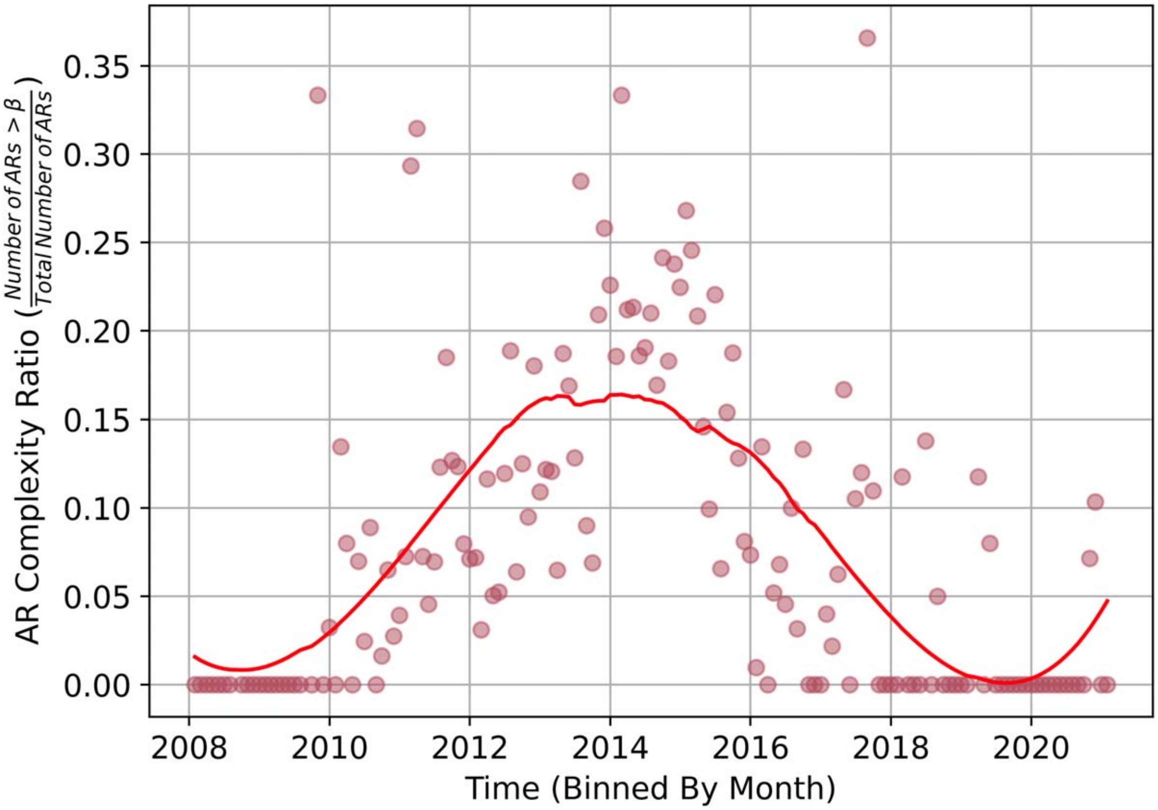


Figure 11. The AR complexity ratio vs. time over Solar Cycle 24. AR complexity ratio is calculated by dividing the number of ARs with a Hale classification more complex than β (this includes γ, $\beta-\gamma$, δ, $\beta-\delta$, $\beta-\gamma-\delta$, and $\gamma-\delta$) by the total number of ARs across each month. A Savitzky–Golay filter (solid red line) has been applied to illustrate the general trend of the data.

Table 2
The FPR for ARs Grouped by the Strongest Flare Produced during Their Lifetime

Largest Flare Produced By AR	False-positive Rate of ARs within Flare Group
Flare Quiet / A	0.010 ± 0.001
B	0.077 ± 0.004
C	0.312 ± 0.007
M	0.699 ± 0.008
X	0.823 ± 0.008

Notes. Results are averaged across data from all models utilizing 25 features. The standard error on the mean is shown as well.

networks, which have been employed in other studies (Liu et al. 2019; Sun et al. 2022). Of course, with these models comes added training time and a need for increasingly powerful computational resources, which is not ideal for operational purposes. Finally, a major drawback of the current SWAN-SF iteration is its focus on 24 hr forecasting. With the rapidly approaching NASA Artemis missions, it will be critical that we have the capability of predicting flaring events even farther (potentially, we hope, up to 72 hr) in advance. While not addressed in this paper, it may be worthwhile in future studies to modify the current data set labels for several extended forecasting windows (36, 48, and 72 hr). This may reveal hidden intricacies between our various training methodologies that were not found in this work.

Acknowledgments

We would like to thank the anonymous reviewer for providing excellent feedback, which greatly improved the clarity of the text. This work was supported by NASA FINESST grant 80NSSC23K1639, NASA LWS grant 80NSSC22K0272, and NSF FDSS grant 1936361.

ORCID iDs

Griffin T. Goodwin https://orcid.org/0000-0003-3493-9174
Viacheslav M. Sadykov https://orcid.org/0000-0002-4001-1295
Petrus C. Martens https://orcid.org/0000-0001-8078-6856

References

Ahmadzadeh, A., Aydin, B., Georgoulis, M. K., et al. 2021, ApJS, 254, 23
Ali, A., Sadykov, V., Kosovichev, A., et al. 2024, ApJS, 270, 15
Angryk, R., Martens, P., Aydin, B., et al. 2020a, SWAN-SF, v1, Harvard Dataverse, doi:10.7910/DVN/EBCFKM
Angryk, R. A., Martens, P. C., Aydin, B., et al. 2020b, Sci. Data, 7, 227
Bobra, M. G., & Couvidat, S. 2015, ApJ, 798, 135
Camporeale, E. 2019, SpWea, 17, 1166
Crown, M. D. 2012, SpWea, 10, S06006
Deshmukh, V., Baskar, S., Berger, T. E., Bradley, E., & Meiss, J. D. 2023, A&A, 674, A159
Florios, K., Kontogiannis, I., Park, S.-H., et al. 2018, SoPh, 293, 28
Gardner, M., & Dorling, S. 1998, AtmEn, 32, 2627
Guyon, I., & Elisseeff, A. 2003, JMLR, 3, 1157
Hudson, H. S. 2021, ARA&A, 59, 445
Ji, A., Aydin, B., Georgoulis, M. K., & Angryk, R. 2020, in IEEE Int. Conf. on Big Data, 4218
Kingma, D. P., & Ba, J. 2014, arXiv:1412.6980
Kingsford, C., & Salzberg, S. L. 2008, NatBi, 26, 1011

Kotsiantis, S. B. 2013, Artificial Intelligence Review, 39, 261
Leka, K. D., Park, S.-H., Kusano, K., et al. 2019a, ApJS, 243, 36
Leka, K. D., Park, S.-H., Kusano, K., et al. 2019b, ApJ, 881, 101
Li, R., Wang, H.-N., He, H., Cui, Y.-M., & Du, Z.-L. 2007, ChJAA, 7, 441
Li, R.-H., & Belford, G. G. 2002, in Proc. of the ACM SIGKDD International Conf. on Knowledge Discovery and Data Mining 8, 570
Liu, H., Liu, C., Wang, J. T., & Wang, H. 2019, ApJ, 877, 121
Marroquin, R. D., Sadykov, V., Kosovichev, A., et al. 2023, ApJ, 952, 97
Natras, R., Horozovic, D., & Mulic, M. 2019, SN Appl. Sci., 1, 49
Nishizuka, N., Sugiura, K., Kubo, Y., Den, M., & Ishii, M. 2018, ApJ, 858, 113
Nishizuka, N., Sugiura, K., Kubo, Y., et al. 2017, ApJ, 835, 156
Pedregosa, F., Varoquaux, G., Gramfort, A., et al. 2011, JMLR, 12, 2825
Sadykov, V. M., & Kosovichev, A. G. 2017, ApJ, 849, 148
Scherrer, P. H., Schou, J., Bush, R. I., et al. 2012, SoPh, 275, 207
Sun, Z., Bobra, M. G., Wang, X., et al. 2022, ApJ, 931, 163
Wang, X., Chen, Y., Toth, G., et al. 2020, ApJ, 895, 3
Yeolekar, A., Patel, S., Talla, S., et al. 2021, in 2021 International Conference on Data Mining Workshops ICDMW, 1067
Yu, D., Huang, X., Wang, H., & Cui, Y. 2009, SoPh, 255, 91
Yuan, Y., Shih, F. Y., Jing, J., & Wang, H.-M. 2010, RAA, 10, 785
Zhang, H., Li, Q., Yang, Y., et al. 2022, ApJS, 263, 28