A -based space weather conditions machine learning classification model for GNSS PNT performance analysis

# Introduction

The Global Navigation Satellite System (GNSS) and its Positioning, Navigation, and Timing (PNT) service have matured to become an essential part of national infrastructure, public goods, and enablers of a vast number of emerging technology and socio-economic applications . Maintaining the GNSS PNT quality has appeared fundamental for the sustainable development of modern economy and society . Overcoming the shortcomings and vulnerabilities of GNSS PNT is a scientific challenge, and the need of a wide variety of scientists, developers, operators, regulators, and users of GNSS-based systems and services . The Earth’s ionosphere, a part of the Earth’s atmosphere stretching from 50 km to 2000 km above the Earth’s surface and comprised of rare but mostly charged atoms and molecules, is the major natural cause of GNSS PNT degradation . This phenomenon especially affects most currently used GNSS receivers, which work as single-frequency receivers exposed to GNSS ionospheric effects . Driven by powerful and random flows of energy and particles from the Sun (space weather conditions), the ionospheric conditions define the properties of GNSS signal propagation through the Earth’s atmosphere and the resulting ionospheric delay . The GNSS ionospheric delay causes errors in GNSS PNT algorithm deployment, designed to produce position, velocity, and time estimates for a GNSS receiver . The ionosphere affects GNSS satellite signals for position estimation by introducing signal propagation delay . The GNSS ionospheric delay is a stochastic variable, whose value is determined by complex physical processes of space weather . How space weather affects GNSS PNT performance was explained using the Space weather – GNSS PNT performance coupling model , as depicted in Table [1](#Xcd98f75463a4f9443cea1890b285c3eb94af900)

The Space weather – GNSS PNT performance coupling model, after .

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| SPACE | GEOMAGNETIC | IONOSPHERE | GNSS | GNSS |
| WEATHER | FIELD |  | PSEUDO- | POSITIONING |
|  |  |  | RANGES | PERFORMANCE |

The GNSS signal encounters a certain number of charged particles from the satellite aerial to a mobile unit’s (GNSS receiver’s) aerial . Such an encounter is quantified using the Total Electron Content () defined by Equation [[eqn:1]](#eqn:1) in , where denotes the height above the Earth’s mean sea level in , represents the vertical ionospheric profile, the volume density of charged particles at height , in .

It should be noted that in the sense of Equation [[eqn:1]](#eqn:1), is defined as a result, a consequence, of the ionospheric conditions, and not their descriptor . The GNSS ionospheric delay may be determined by derivation from the Appleton-Hartree equation, as given in Equation [[eqn:2]](#eqn:2), where denotes the GNSS ionospheric delay in , denotes the velocity of an electromagnetic wave in vacuum in , and denotes the carrier wave frequency of the satellite signal in .

Combining Equation [[eqn:1]](#eqn:1) and Equation [[eqn:2]](#eqn:2), one can conclude the linear relation between and , as given in Equation [[eqn:3]](#eqn:3), a sub-model incorporated in the Space weather – GNSS PNT performance coupling model .

The GNSS ionospheric delay has been identified as a source of GNSS PNT degradation since the dawn of GNSS. GNSS systems offer various standard GNSS ionospheric delay estimation (correction) models to mitigate the deteriorating effects on GNSS PNT, such as the Klobuchar model . The standard ionospheric correction models are global, and insufficiently flexible to update to mitigate GNSS ionospheric delay to satisfy rising demands on GNSS PNT performance . The development of regional and local models attempts to solve the problem of GNSS PNT sustainable performance in various ionospheric conditions . In recent developments, our team has proposed the Ambient-Aware Application-Aligned (AA)2 PNT to take into account the actual ionospheric and geomagnetic conditions near a mobile unit (a GNSS receiver) . Direct measurements of the immediate geomagnetic and ionospheric condition variables may be supplied to a machine learning-based adapted GNSS ionospheric correction model, thus solving the single-frequency GNSS problem . Previous research has identified predictors and target variables (descriptors of geomagnetic, ionospheric, and GNSS PNT conditions) . The classification of different scenarios of GNSS ionospheric conditions with adverse effects on the GNSS PNT remained an unsolved precondition needed for the development of a machine learning-based GNSS ionospheric delay correction model to render the GNSS PNT algorithm ionospheric conditions-agnostic .

Here we propose a methodology for a machine learning-based classification of the ionospheric conditions based entirely on observations of geomagnetic indices. The proposed method is inspired by previous lightweight machine-learning techniques for classifying sensor readings. The model is sufficiently simple to be applied on computationally capable platforms with suitable geomagnetic field sensors, such as smartphones and connected/autonomous vehicles. The research presented acquires the ambient data and analyses its statistical properties. The data set is split into training and test sets. Several candidates for the GNSS ionospheric delay model are developed in this study. A tailored set of validation methods is used to assess their performance. The optimal GNSS ionospheric delay correction model is identified based on GNSS PNT-related objective criteria, and its performance is demonstrated in an independent case study.

The main contributions of this paper are:

* Defining multiple -based classes using theoretical knowledge and statistics
* Classifying values of and other geomagnetic field variables based on
* Proposing a machine learning-based classification model of space weather events
* Achieving an accuracy over for the Naive Bayes model and any input variables

The findings indicate the classification generated by the Naive Bayes model could be used in real-life systems and support our hypothesis.

The rest of the paper is structured as follows. Section [2](#sec:Dataset) describes this work’s methodology, dataset, and evaluation metrics. Section [3](#sec:Results) presents the results obtained in this study. Section [4](#sec:Discussion) discusses the implications of the results. The main points are summarised and the conclusion is given in Section [5](#sec:Conclusion).

# Method and Data

## Method

A Support Vector Machine (SVM) with a Polynomial Kernel, C5.0 Decision Tree (DT), Naive Bayes (NB), shallow Neural Network (NN), Partial Least Squares (PLS), Flexible Discriminant Analysis (FDA) and shallow Neural Network using Principal Component Analysis (PCA) of the input data were tested based on their ability to classify a set of observations of the geomagnetic field in , and other predictors, into one of the scenarios of space weather conditions based on . Multiple -dependent classes were predefined using theoretical knowledge. Statistical analysis of the data confirmed that distributions of other variables change for different ranges, and not for different ranges, supporting the validity of the classification. The study assumes that the dependent output variable, the class, can be predicted based on the independent variables used as input.

The models were selected because they represent larger families of classification methods. SVMs are supervised maximum margin models. Decision Trees also apply supervised learning. Naive Bayes classifiers are probabilistic classifiers that can be parametric or non-parametric, but this study uses a non-parametric approach. PLS is a non-parametric linear regression model. FDA uses multiple non-parametric linear regression models to create a non-linear classification. PCA is a linear dimensionality reduction technique that extracts a predefined number of components for training a Neural Network. Neural Networks imitate the brain using artificial neurons to produce outputs based on the input and the activation function. Neural Networks require that the model structure be predefined, and hyperparameters are usually tuned or predefined based on assumptions. An automated tuning procedure was used in this study.

All Neural Network models were applied based on research by Kuhn for the *R* *caret* package .

### Support Vector Machine

In machine learning, Support Vector Machines (SVMs) or Support Vector Networks (SVNs) are supervised maximum margin models with associated learning algorithms that analyze data for classification and regression. In addition to linear classification, SVMs are also effective for non-linear classification using the hyperplane kernel trick , implicitly mapping their inputs to high-dimensional feature spaces.

SVMs can also be used for regression tasks where the objective becomes -sensitive. A "soft margin" to separate data that is not linearly separable was proposed by Corinna Cortes and Vapnik in 1993 and published in 1995 . Support Vector Clustering (SVC) applies support vector statistics to unlabeled data in unsupervised learning to find new natural groupings. Meyer, Leisch, and Hornik compared SVM with other classifiers , but it is unclear whether SVM predictions perform better than other linear models, such as logistic, and linear regression.

Support Vector Machine constructs a set of hyperplanes in a high-dimensional space, used for classification, regression, or outlier detection . Intuitively, a good separation is achieved by the hyperplane with the greatest distance to the nearest point in the training data belonging to any class .

The original maximum margin hyperplane algorithm proposed by Vapnik in 1963 constructed a linear classifier. In 1992, Bernhard Boser, Isabelle Guyon, and Vladimir Vapnik proposed creating nonlinear classifiers using the kernel trick . A multidimensional feature space increases the generalization error of Support Vector Machines, so additional samples are needed to enhance performance . To keep the computational burden reasonable, a kernel probability density function is chosen to fit the problem .

SVMs have been used to solve a variety of real-world problems. SVMs are useful in text and hypertext categorization, as their application can significantly reduce the need for labeled training instances in standard inductive and transductive settings . Some methods for shallow semantic parsing are based on SVMs . Image classification and segmentation can also be performed using SVM models, including a modified version of SVM that uses privileged access as proposed by Vapnik .

Classification of satellite data such as Synthetic Aperture Radar (SAR) data is possible using a supervised SVM . Handwritten characters can be recognized using SVM . The SVM algorithm is widely used in biological and other sciences. They were used to classify proteins with up to compounds correctly classified. Permutation tests based on SVM weights have been proposed as a mechanism for interpreting SVM models . SVM weights were also used to interpret SVM models .

The performance of SVM depends on the kernel probability density function, the parameters of the specified kernel probability density function, and the "soft" margin parameter . Each combination of parameter choices is usually cross-validated, and the parameters with the best accuracy are selected. The final model is used to test and classify new data and is trained using the best parameters .

### Decision Tree

Decision Trees are an approach to supervised learning in statistics, data mining, and machine learning. They represent predictive or decision-making model used to make inferences about a series of observations. Tree models in which the target variable can take on a discrete set of values are called classification trees. Decision Trees where the target variable can take on continuous values (usually real numbers) are called regression trees. A regression tree can be extended to any object with pairwise differences such as categorical arrays . Decision Trees are among the most popular machine learning algorithms due to their comprehensibility and simplicity . A Decision Tree describes the input data and the resulting classification tree can be the input for decision-making in data mining .

A tree is built by dividing the original set, which forms the root node, into subsets that form descendants, or successors. The segmentation is made using a set of rules based on classification features . This process is repeated on each derived subset in a recursive manner called recursive partitioning. The recursion is complete when the subset at a node all have the same target variable values or the division no longer adds value to the predictions. This process of Top-Down Induction of Decision Trees (TDIDT) is an example of a greedy algorithm and is by far the most common strategy for learning Decision Trees from data . C5.0, used in the *caret* package in R, has a similar approach and improves the ID3 and C4.5 algorithms.

Decision Trees can also be described as a combination of mathematical and computational techniques that help describe, categorize, and generalize a given data set in data mining. Data comes in records of the form . The dependent variable is the target variable we need to understand, classify, or generalize. Features form the vector used for that task.

### Naive Bayes

In statistics, Naive Bayes (NB) classifiers are a family of linear "probabilistic classifiers" that assume that, given a target class, the features are conditionally independent. The classifier got its name from this strong and possibly naive assumption. These classifiers are among the simplest Bayesian network models . Naive Bayes classifiers are highly scalable and require several parameters. The number of parameters depends linearly on the number of variables. Variables represent features or predictors in the learning problem. Maximum likelihood training can be performed by evaluating a closed-form expression , which requires linear time, instead of the expensive iterative approximation used for many other classifiers.

In the statistical literature, Naive Bayes models are known by various names, including simple Bayes and independent Bayes . These names refer to Bayes’ theorem in the classifier’s decision rule, but a Naive Bayesian classifier is not necessarily a Bayesian method . In other words, one can work with a Naive Bayesian model without accepting Bayesian probability or methods. In many practical applications, parameter estimation for Naive Bayes models uses the maximum likelihood method.

Despite their simple design and seemingly oversimplified assumptions, Naive Bayes classifiers have performed quite well in many complex real-world situations. In 2004, an analysis of the Bayesian classification problem showed that there are reasonable theoretical reasons for the seemingly incredible performance of Naive Bayesian classifiers . However, a comprehensive comparison with other classification algorithms in 2006 showed that Bayesian classification performed worse than other approaches, such as boosted trees or random forests . The advantage of Naive Bayes is that it requires only a small amount of training data to estimate the parameters needed for classification .

In the abstract, Naive Bayes is a conditional probability model. The model assigns probabilities to each of the possible outcomes or classes given the problem instance to be classified, represented by the vector which encodes features representing mutually independent variables . The problem with the above formulation is that if the number of features is large or the feature has a large value range, it is infeasible to base a model on probability tables. The model must therefore be reformulated using conditional probability represented by Bayes’ theorem in Equation [[eqn:4]](#eqn:4).

To estimate feature distribution parameters, we must assume a parametric distribution, or generate non-parametric models for the features from the training data set . Assumptions about the distribution of features are called an event model or Bayesian classifier, which can introduce confusion by creating both a continuous and discrete model . Multinomial and Bernoulli distributions have often been used for discrete features in document classification, or spam filtering. The Gaussian Naive Bayesian classifier was used since the features used in this study assumed continuous values.

Clustering is often used to discretize continuous values. The new set of features obtained by discretization follows the Bernoulli distribution. Some literature claims that discretization is necessary to use a Naive Bayesian classifier, but discretization may discard the information needed to distinguish between classes .

Sometimes the distribution of class-based marginal densities is far from normal. The marginal probability density of each class can be used to estimate the kernel probability density function in such cases. This method, introduced by John and Langley , can significantly increase the accuracy of the classifier .

### Neural Networks

The neurons of human or animal brains provide the basis for a Neural Network (NN) or Artificial Neural Network (ANN) with connected units or nodes called artificial neurons in machine learning .

Shallow Neural Networks are simple and easy to train quickly with fewer parameters, and computational resources. They typically contain only a few hidden layers for processing between the input layer that receives the data, and the final layer that produces the output . A network with at least two hidden layers is considered a deep Neural Network.

Neural Networks are typically trained through empirical risk minimization. This method optimizes the network’s parameters to minimize the difference or empirical risk, between the predicted output and the actual target values in a given dataset . Gradient-based methods such as backpropagation are usually used to estimate network parameters . During the training phase, ANNs learn from labeled training data by iteratively updating their parameters to minimize a defined loss function .

Using Artificial Neural Networks requires an understanding of their characteristics. The model suitable for a specific task varies depending on how the data is stored and other practical considerations. Many model parameters may need to be adjusted and differ for each model. The hyperparameters for the learning algorithm may also be modified to suit the problem during an extensive tuning process experimenting on unseen training data, and various learning algorithms are available.

The network design process can be simplified using previously defined machine learning techniques like neural architecture search (NAS). These networks have performed well even when pitted against architectures selected by humans. The proposed model is tested and the responses are used to train the NAS network . AutoML and AutoKeras provide extensive frameworks for this process. The *scikit-learn* library features various classification, regression, and clustering algorithms including support-vector machines, random forests, gradient boosting, k-means, and Density-Based Spatial Clustering (DBSCAN). The hyperparameters must be predefined, are not trained, and include how many neurons are in each layer, learning rate, step, stride, depth, receptive field, and padding for CNNs . TensorFlow and Keras are used to build a custom network in Python, and the user can define layers, models, or metrics. The data, number of hidden layer units, learning rate, and number of iterations are required parameters for model training based on a book by Liu .

### Partial Least Squares

Partial Least Squares (PLS) regression, or projection to latent structures, , is a linear regression statistical model that transforms the predicted and the observable variables to a new space instead of identifying maximum variance hyperplanes used in a linear regression model. Svante and Herman O. A. Wold, a Swedish father and son, founded this approach. PLS methods are bilinear factor models because the and are projected to new spaces. In Partial Least Squares Discriminant Analysis (PLS-DA), is categorical .

Using paired observations . PLS finds the normalized direction that maximizes the covariance in the first step , shown in Equation [[eqn:5]](#eqn:5). Matrix notation is used in further discussion of the model.

The scores form an orthogonal basis in PLS regression, and the loadings are chosen to achieve this. In Principal Component Analysis (PCA) orthogonality is not imposed on scores but onto loadings instead . Many versions of PLS exist for estimating the factor and loading matrices, such as the PLS1 algorithm .

### Flexible Discriminant Analysis

Flexible Discriminant Analysis (FDA) is a general methodology that creates the discriminant surface for a multigroup non-linear classification model based on a mixture of non-parametric linear regression models , such as Multivariate Adaptive Regression Splines (MARS) and linear discriminant analysis (LDA).

Many predictors can be used at once in FDA , variable interactions are automatically noted , it is complex but execution time and computational load are adequate . The algorithm is not significantly affected by outliers .

Modifying different settings has a high impact on FDA , and the estimation might be unsuccessful if the predictor distributions are highly correlated. FDA is prone to overfitting , and challenging to comprehend or explain . The assumption of normality for the continuous independent variables is necessary to determine the response or grouping categorical variable.

Linear discriminant analysis (LDA), normal discriminant analysis (NDA), or discriminant function analysis is a generalization of Fisher’s linear discriminant defined in 1936 . The results of LDA may be utilized directly or to reduce dimensionality before classification, which is more often used . LDA finds a linear combination of features or measurements differentiating two or more sample classes . Discriminant analysis has continuous independent variables and a categorical dependent variable representing the groups .

LDA is closely related to PCA and factor analysis since they identify linear functions of multiple variables that fit the input with the smallest error . Discriminant correspondence analysis is equivalent to LDA for categorical independent variables. Points in discriminant function analysis are rated based on quantitative predictors and a group measure to represent classification .

The assumptions of MANOVA are also true for discriminant analysis. The analysis is highly affected by outliers so there must be fewer predictor variables than samples in the smallest class . In every level of the grouping variable, independent variables must be normal to achieve multivariate normality . Box’s M statistic test whether homogeneity of variance/covariance is established and whether variances among group variables change for predictor levels.

Quadratic instead of linear discriminant analysis should be utilized if covariances are different . The scores of one predictor variable should not depend on other predictor variables for any sample, and the samples should be chosen at random . Discriminant analysis can handle small deviations from these requirements and may be used even if multivariate normality is not achieved, such as for dichotomous variables .

The discriminant rules suggest that if , then for a group , and sets of sample space. The discriminant analysis method identifies regions of that decrease errors of classification . A discriminant score describes how well a discriminant differentiates classes. .

The discriminant analysis eigenvalue represents the characteristic root of each function, indicating the quality of the group separation produced by the function. The function differentiates better for a larger eigenvalue . Eigenvalues do not have an upper limit , so the results should not be taken at face value.

Even though it is not universally accepted, some suggest eigenvalues be utilized as effect size measures . The percent of correctly classified data, such as the kappa value, can also be used as the effect size while accounting for random matches in classification . Kappa is not biased for any particular class, whether performance is high or low, and instead normalizes across all classes .

Fisher’s linear discriminant and LDA are sometimes considered synonyms. However, Fisher describes a slightly different discriminant, which does not make some of the assumptions of LDA such as a normal distribution or the equality of covariances for groups. The application of the Fisher discriminant is expanded to find a subspace capturing all class variability for multiple groups, as suggested by C. R. Rao .

### Principal Component Analysis

Principal component analysis (PCA) is a linear dimensionality reduction technique with applications in exploratory data analysis, visualization, and data preprocessing. It is used in preprocessing before training a Neural Network in the method in the *caret* package in *R* . The data is linearly transformed into a new coordinate system such that the directions (principal components) capturing the largest variation in the data can be easily identified. Many studies use the first two principal components to plot the data and visually identify clusters of closely related data points .

PCA is closely related to factor and canonical correlation analysis (CCA). PCA is the simplest of the true eigenvector-based multivariate analyses. PCA defines a new orthogonal coordinate system that optimally describes the variance in a single dataset . CCA does the same for the cross-covariance between two datasets. Robust and L1-norm-based variants of standard PCA have also been proposed .

Karl Pearson conceived of PCA in 1901 as a counterpart of the principal axis theorem in mechanics . Harold Hotelling discovered the Hotelling transform in multivariate quality control in the 1930s without knowledge of the previous work . Synonyms for PCA include the discrete proper orthogonal decomposition (POD) in mechanical engineering and the Kosambi–Karhunen–Loève or Karhunen–Loève theorem (KLT) in signal processing. Singular value decomposition (SVD) of X, established in the final quarter of the 19th century , is also similar to PCA. Eigenvalue decomposition (EVD) of a matrix in linear algebra shares the mathematical foundations of PCA . Factor analysis differs in several key features , but has many aspects in common with PCA. Empirical orthogonal functions (EOF) in meteorological science were developed by Lorenz in 1956 . The Eckart–Young theorem and quasiharmonic modes are both connected to PCA. Empirical modal analysis in structural dynamics and spectral decomposition in noise and vibration also take a similar approach to PCA.

The data in PCA is converted to a new coordinate system using a scalar projection, an orthogonal linear transformation on a real inner product space. The first coordinate or principal component covers the largest portion of the variance, decreasing with each following component . The significance of the principal components can be checked with parametric bootstrap for a smaller dataset to help decide how many principal components to keep .

## Data Description and Analysis

Many parameters can describe the Earth’s geomagnetic field, but this study focuses on those describing disturbances of the geomagnetic field, most importantly , , , and . The index quantifies global geomagnetic activity by measuring disturbances in the horizontal component of Earth’s magnetic field. The index is based on 3-hour-based indices from 12 geomagnetic locations.

The index is defined by Table[2](#tab:K).

The index.

|  |  |
| --- | --- |
| Geomagnetic | The intensity of a |
| observatory index | geomagnetic storm |
| - | Quiet geomagnetic conditions |
| - | Unsettled |
|  | Minor storm |
|  | Large storm |
|  | Severe geomagnetic storm |
| - | Very intense (extreme) geomagnetic storm |

The index is an integer in the range to ( lower scare is , lower scale is ), the largest range of geomagnetic disturbances and in the two horizontal components and during a -hour UT interval.

The index is a single value of magnetic activity for the entire day, the average of indexes.

Elevated levels can increase signal delay which can cause positioning errors, especially critical for high-precision applications. can be calculated as indicated in Equation [[eqn:1]](#eqn:1), Equation [[eqn:2]](#eqn:2), and Equation [[eqn:3]](#eqn:3).

The GPS Ionospheric delay on the radio ray path between satellite and receiver is defined in Equation [[eqn:6]](#eqn:6) where represents the GPS ionospheric delay, is the velocity of light, and is the refraction index on the path .

The GPS ionospheric delay related to distance measurements can be defined using Equation [[eqn:7]](#eqn:7) .

Appleton’s formula in Equation [[eqn:8]](#eqn:8) reasonably approximates the relationship between the refraction index and the free electron density distribution . is the vertical distribution of the free electron density over a height above the surface of the Earth, and is the frequency of the radio signal.

The combination of Equation [[eqn:6]](#eqn:6) and Equation [[eqn:8]](#eqn:8) yields Equation [[eqn:9]](#eqn:9) that directly describes the relationship between the free electron density profile and the GPS ionospheric delay .

The in Equation [[eqn:10]](#eqn:10) is calculated using the integral on the right side of Equation [[eqn:9]](#eqn:9).

Using equation Equation [[eqn:10]](#eqn:10), Equation [[eqn:9]](#eqn:9) can be written as shown in Equation [[eqn:11]](#eqn:11).

Following the same analogy, the equivalent GPS ionospheric delay is calculated in Equation [[eqn:12]](#eqn:12).

Considering the pseudo-range measurements taken on both GPS transmitting frequencies ( and ) and using Equation [[eqn:12]](#eqn:12), the relation between the and GPS pseudo-range measurements can be established in Equation [[eqn:13]](#eqn:13) :

Equation [[eqn:13]](#eqn:13) can be rewritten as in Equation [[eqn:14]](#eqn:14), defining . The actual value of the can be computed utilizing a dual-frequency GPS receiver and Equation [[eqn:14]](#eqn:14).

Considering the imposed regulations for common GPS dual-frequency receiver use, a different method for estimating the and GPS ionospheric delay is needed. A standard GPS ionospheric delay model, such as the Klobuchar model in Equation [[eqn:15]](#eqn:15) . A cosine-like daily dynamic of GPS ionospheric delay values is assumed in the Klobuchar model, with the daily maximum appearing around 14:00 in local time .

The next relevant parameter is , also known as the index, a measure of geomagnetic storm intensity. describes ring currents forming above the sub-equatorial region and affecting the ionospheric regions in mid-latitudes.

The index is derived from measurements taken by a network of ground-based magnetometer stations located near the magnetic equator, which continuously monitor the horizontal component of Earth’s magnetic field. To calculate the index, variations in the horizontal magnetic field from these stations are averaged, and this average is then subtracted from a baseline value representing the quiet-time magnetic field. The resulting value, expressed in nanoteslas (), measures the intensity of geomagnetic disturbances, with more negative values indicating stronger geomagnetic storms. The performance of GNSS is linked to many dynamic conditions of space weather. Solar activity can induce various effects that degrade the accuracy, availability, and reliability of GNSS PNT. Incorporating parameters such as the and indexes, which provide global measures of geomagnetic activity, alongside and the Disturbance Storm Time () index, allows for a more detailed assessment of the space environment and its potential effects on GNSS signals. In this paper, , , and were used with ,, , and to train machine-learning models.

To generate classes, samples were split into ranges, P (positive, from to ), N (normal, from to ), R (recovery, from to ), T (through, from to ), and E (extreme, from to ), based on values derived from theoretical knowledge of different storm phases. The ranges are listed in Table [3](#tab:Dstranges).

-based classification rules.

|  | Storm phase classification |
| --- | --- |
|  | positive phase (P) |
|  | normal (N) |
|  | recovery phase (R) |
|  | through (T) |
|  | extreme (E) |

Figure [1](#fig:correlation) contains a heat map of the correlation between all variables used in this study. Blue represents a low correlation, while red represents a higher correlation. Swapping the row and column leads to the same combination of variables, so the matrix is symmetrical concerning the secondary diagonal. The values on the secondary diagonal equal and can be disregarded because all variables are fully correlated with themselves. Besides these values on the secondary diagonal, the highest correlation of is between and , as expected based on previously outlined theoretical aspects of these variables. and have a correlation coefficient of , the second highest value not on the secondary diagonal. This is the only other value larger than not on the secondary diagonal. A high correlation of and supports the hypothesis that we can predict values, ranges, or classes using . The third highest correlation coefficient not on the secondary diagonal is between and and equals . The properties and definitions of these variables explain this correlation.



A heat map of the correlation between all variables used in this study, where red represents a higher correlation, while blue represents a low correlation. All variables are fully correlated with themselves, so all values on the secondary diagonal equal . The matrix is symmetrical concerning the secondary diagonal because the same combination of correlated variables is achieved when swapping the row and column.

The box plots of all variables for different ranges of values in Figure [2](#fig:dataset2boxplot), without restriction on the range of , and in Figure [3](#fig:iono3boxplot), where the is less than , demonstrate that the minimum, maximum, and arithmetic mean of decrease for larger values regardless of ranges. The opposite is true for , as indicated by a high correlation of and in Figure [1](#fig:correlation). exhibits the same trend as , but it is less prominent. is the most stable variable with the smallest changes related to when observing , , and , but an increase for larger values can also be observed. Similar conclusions can be reached from the scatter plots of all variables for different ranges of in Figure [4](#fig:iono3scatterplot).

The scatter plots of all variables for different ranges of values in Figure [5](#fig:dataset2scatterplot) demonstrate that their distribution is not significantly dependent on . This is why the box plots in Figure [2](#fig:dataset2boxplot) and Figure [3](#fig:iono3boxplot) that only differ in ranges look very similar. A minimal difference is also evident between the histograms of the relative frequency for all variables in Figure [6](#fig:allTEC), without restriction on the range of , and in Figure [7](#fig:300TEC), where the is less than . Table [3](#fig:iono3boxplot) provides the minimum, quartile, median, arithmetic mean, quartile, and maximum values for all variables, supporting the conclusions made based on Figure [2](#fig:dataset2boxplot) and Figure [6](#fig:allTEC).



Box plots of all variables without restriction on the range of for different ranges of values defining the class label used in this study.



Box plots of all variables when the is less than for different ranges of values defining the class label used in this study.



Scatter plots of all variables when the is less than for different ranges of values defining the class label used in this study.



Scatter plots of all variables for different ranges of values without restriction.



Histograms of the relative frequency for all variables without restriction on the range of .



Histograms of the relative frequency for all variables when the is less than .

The minimum, quartile, median, arithmetic mean, quartile, and maximum values for all variables.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Min. |  |  |  |  |  |
| Qu. |  |  |  |  |  |
| Median |  |  |  |  |  |
| Mean |  |  |  |  |  |
| Qu. |  |  |  |  |  |
| Max. |  |  |  |  |  |

### Data Pre-processing

Data pre-processing is recommended to increase classification accuracy , so this step was also applied to the input data before training the models in this study. There are many ways to standardize data, such as minimum-maximum, normalization by decimal scaling, and Z-score . Subtracting the mean and dividing by the variance for each feature are commonly used for Support Vector Machines (SVMs) and other models tested in this study, so this approach was chosen.

The values *scale* and *center* were used in the code for this study in the *preProcess* parameter for the *train* function from the *caret* package developed for *R*. The option *center* subtracts the mean of each feature while *center* divides by the standard deviation.

### Training and Testing Dataset

The samples are divided into training and testing datasets as close as possible to a ratio of for training and for testing. During the division, the share of classes in the original data was taken into account, that is, the division was stratified by class so that an approximately equal ratio of classes was present in both the training data and the testing data, which is a feature of the *createDataPartition* function from the *caret* R library that was used .

## Performance Metrics

### Confusion Matrix

The metrics and terminology used to evaluate the classifier performance were taken from the confusion matrix defined in the R function *confusionMatrix* in the *caret* library . The confusion matrix is suitable for use in multiclass classification, which is the goal of this project. A confusion matrix is a table where the number of rows and columns corresponds to the number of classes. The cells report the number of samples in each class classified into any class. This enables a more detailed analysis than by observing the share of correct classifications, i.e. accuracy. Accuracy will give results that can lead us to a wrong conclusion if the data set is unbalanced, that is, if the numbers of observations in different classes vary significantly, which was the case in this paper, because out of a total of samples, of them belonged to the largest N class, and to the smallest E class.

If there are only two classes (Yes and No), the metrics are calculated using an example of a confusion matrix in Table [5](#tab:cm).

A true positive (TP) classification result correctly indicates that the sample belongs to the positive class.

A true negative (TN) classification result correctly indicates that the sample belongs to the negative class.

A false positive (FP) classification result or type I error, incorrectly indicates that the sample belongs to the positive class because it truly belongs to the negative class.

A false negative (FN) classification result or type II error, incorrectly indicates that the sample belongs to the negative class because it truly belongs to the positive class.

The confusion matrix for a two-class problem.

|  | Reference | |
| --- | --- | --- |
| Prediction | Yes | No |
| Yes | TP | FP |
| No | FN | TN |

Sensitivity, recall, hit rate, or true positive rate (TPR), is calculated as , that is, the proportion of samples that are correctly classified as positive among all samples that truly belong to the positive class. Higher TPR values indicate better results for the positive class.

Specificity, selectivity, or true negative rate (TNR), is calculated using the expression , that is, the proportion of samples that are correctly classified as negative among all samples that truly belong to the negative class. Higher TNR values indicate better results for the negative class.

Prevalence is calculated as for the positive class, that is, the proportion of samples that truly belong to the positive class among all samples. Prevalence values other than an even split by the number of classes indicate a class imbalance.

Precision, or positive predictive value (PPV), is calculated as for a two-class problem, i.e. the proportion of truly positive results among all samples classified as positive. Higher PPV values indicate better results for the positive class.

The negative predictive value (NPV) is calculated as for a two-class problem, i.e. the share of truly negative results among all samples that are classified as negative. Higher NPV values indicate better results for the negative class.

The detection rate (DR) is calculated as for the positive class, that is, that is, the proportion of samples that are correctly classified as positive among all samples.

The detection prevalence (DP) is calculated as for the positive class, that is, that is, the proportion of samples that are correctly or incorrectly classified as positive among all samples.

The accuracy (Acc) is calculated as , that is, the proportion of samples that are correctly classified among all samples.

Since accuracy is not usable in the case of unbalanced classes, balanced accuracy (BA) is introduced, which is calculated as and is arithmetic mean TPR and TNR, which are respectively centered on the positive and on the negative class separately.

Another metric that can be used instead of accuracy to consider each class separately is the score, i.e. the harmonic mean of PPV and TPR, equal to .

If there are more than two classes, results are calculated for each class separately by viewing that class as the positive class and samples of all other classes as negative, a "one versus all" approach. The unweighted Kappa statistic and a *p*-value from McNemar’s test are also computed. McNemar’s test produces NA values with sparse tables, such as the one in this experiment, so it was not included in the results.

A confidence interval for the accuracy rate is computed using *binom.test* and a one-sided test to see if the accuracy is significantly larger than the "no information rate," which is the percentage of the largest class.

### The Binomial Test

The binomial test for two classes is an exact test of the statistical significance of deviations from a theoretically expected distribution.

The binomial test is often utilized to validate a hypothesis about the probability () of success , where users define between and . The expected number of successes would be if the null hypothesis were correct. If out of attempts are successful, the binomial distribution formula defines the probability in Equation [[eqn:16]](#eqn:16).

The probability of an outcome equally or more extreme than actual occurrences is used to calculate the -value. This is a simple operation for a one-tailed test.

Equation [[eqn:17]](#eqn:17) contains the expression for the -value when testing the hypothesis that . We can use a range from to to test .

A binomial distribution isn’t symmetric if , so calculating a -value for a two-tailed test is not as simple. The -value from the one-tailed test cannot be doubled to obtain the result since all events must be considered, including ones more extreme than previously seen. These extreme events are equally or less probable than . Equation [[eqn:18]](#eqn:18) denotes all such events. Equation [[eqn:19]](#eqn:19) is used to compute the two-tailed -value.

### Cohen’s Kappa Coefficient

Cohen’s kappa coefficient (, lowercase Greek kappa) indicates inter- and intra-rater reliability for classification . As it accounts for random matches in ratings, it is a better metric than simply computing the percentage of agreement. Interpreting indices of agreement is not straightforward, and is still a point of contention among experts. Using disagreement instead of agreement in ratings might be simpler . Galton was the first to use a statistic similar to Cohen’s kappa in 1892 . Jacob Cohen formally described the kappa coefficient in 1960 in the journal "Educational and Psychological Measurement" .

Two raters group items into mutually exclusive classes, and Cohen’s kappa measures their agreement. Equation [[eqn:20]](#eqn:20) defines , where is the relative observed agreement among raters, and is the hypothetical probability of chance agreement. The observed data influences the computed probabilities of each observer randomly selecting each category.

If the raters completely align then . When there is no agreement other than what would be expected by chance and is given by , . If statistic is negative there is no match between the raters, which can reflect a real tendency of the raters to give differing ratings.

Equation [[eqn:21]](#eqn:21) is derived from the construction in Equation [[eqn:22]](#eqn:22) and holds for categories with observations to categorize if is the number of times rater predicted category . In these equations, is the estimated probability that both rater and rater will classify the same item as , and is the estimated probability that rater will classify an item as . A similar claim can be made for rater .

If ratings are independent Equation [[eqn:23]](#eqn:23), can be used.

Equation [[eqn:24]](#eqn:24) is used to estimate the term by using the number of items classified as by rater () divided by the total items to classify ().

(and similarly for rater ).

Equation [[eqn:25]](#eqn:25) contains the Cohen’s Kappa formula for the traditional confusion matrix for binary classifications in Table [5](#tab:cm).

In this scenario, Cohen’s Kappa equals the Heidke skill score introduced by Myrick Haskell Doolittle in 1888 .

# Research Results

The accuracy, CI, NIR, -value, and Kappa statistic for each model when using all variables (all), all variables except (no ), all variables except , , and d (no ), only , , and (coord), only , , and ( ), only , , and ( ), or only , , and ( ) is displayed in Table [6](#tab:stats:reverse:all), Table [7](#tab:stats:reverse:noDst), Table [8](#tab:stats:reverse:no$TEC$), Table [9](#tab:stats:reverse:coord), Table [10](#tab:stats:reverse:xyap), Table [11](#tab:stats:reverse:xzap), and Table [12](#tab:stats:reverse:yzap).

The accuracy, CI, NIR, -value, and Kappa statistic for each model when using all variables as input.

| Model | Acc | CI | NIR | -value | Kappa |
| --- | --- | --- | --- | --- | --- |
| svmPoly |  |  |  |  |  |
| C5.0 |  |  |  |  |  |
| nb |  |  |  |  |  |
| nnet |  |  |  |  |  |
| pls |  |  |  |  |  |
| fda |  |  |  |  |  |
| pcaNNet |  |  |  |  |  |

The accuracy, CI, NIR, -value, and Kappa statistic for each model when using all variables except as input.

| Model | Acc | CI | NIR | -value | Kappa |
| --- | --- | --- | --- | --- | --- |
| svmPoly |  |  |  |  |  |
| C5.0 |  |  |  |  |  |
| nb |  |  |  |  |  |
| nnet |  |  |  |  |  |
| pls |  |  |  |  |  |
| fda |  |  |  |  |  |
| pcaNNet |  |  |  |  |  |

The accuracy, CI, NIR, -value, and Kappa statistic for each model when using all variables except , , and d as input.

| Model | Acc | CI | NIR | -value | Kappa |
| --- | --- | --- | --- | --- | --- |
| svmPoly |  |  |  |  |  |
| C5.0 |  |  |  |  |  |
| nb |  |  |  |  |  |
| nnet |  |  |  |  |  |
| pls |  |  |  |  |  |
| fda |  |  |  |  |  |
| pcaNNet |  |  |  |  |  |

The accuracy, CI, NIR, -value, and Kappa statistic for each model when using only , , and as input.

| Model | Acc | CI | NIR | -value | Kappa |
| --- | --- | --- | --- | --- | --- |
| svmPoly |  |  |  |  |  |
| C5.0 |  |  |  |  |  |
| nb |  |  |  |  |  |
| nnet |  |  |  |  |  |
| pls |  |  |  |  |  |
| fda |  |  |  |  |  |
| pcaNNet |  |  |  |  |  |

The accuracy, CI, NIR, -value, and Kappa statistic for each model when using only , , and as input.

| Model | Acc | CI | NIR | -value | Kappa |
| --- | --- | --- | --- | --- | --- |
| svmPoly |  |  |  |  |  |
| C5.0 |  |  |  |  |  |
| nb |  |  |  |  |  |
| nnet |  |  |  |  |  |
| pls |  |  |  |  |  |
| fda |  |  |  |  |  |
| pcaNNet |  |  |  |  |  |

The accuracy, CI, NIR, -value, and Kappa statistic for each model when using only , , and as input.

| Model | Acc | CI | NIR | -value | Kappa |
| --- | --- | --- | --- | --- | --- |
| svmPoly |  |  |  |  |  |
| C5.0 |  |  |  |  |  |
| nb |  |  |  |  |  |
| nnet |  |  |  |  |  |
| pls |  |  |  |  |  |
| fda |  |  |  |  |  |
| pcaNNet |  |  |  |  |  |

The accuracy, CI, NIR, -value, and Kappa statistic for each model when using only , , and as input.

| Model | Acc | CI | NIR | -value | Kappa |
| --- | --- | --- | --- | --- | --- |
| svmPoly |  |  |  |  |  |
| C5.0 |  |  |  |  |  |
| nb |  |  |  |  |  |
| nnet |  |  |  |  |  |
| pls |  |  |  |  |  |
| fda |  |  |  |  |  |
| pcaNNet |  |  |  |  |  |

The execution time in seconds calculated using the R *system.time* function for each model when using different combinations of variables as input is displayed in Table [13](#tab:time:total). The experiment was run on Windows 11 using R Studio version 2024.04.2+764 and R version 4.4.1, the AMD Radeon RX 6600 GPU, 16GB of RAM, and the AMD Ryzen 5 PRO 4650G CPU with 6 cores.

The execution time in seconds for each model when using different combinations of variables as input.

| Model | svmPoly | C5.0 | nb | nnet | pls | fda | pcaNNet |
| --- | --- | --- | --- | --- | --- | --- | --- |
| all |  |  |  |  |  |  |  |
| no |  |  |  |  |  |  |  |
| no |  |  |  |  |  |  |  |
| coord |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

## Comprehensive Results for the Naive Bayes Model

The confusion matrix and the performance indicators derived from it when using the Naive Bayes model and all input variables are depicted in Table [14](#tab:cm:all:nb) and Table [15](#tab:cs:reverse:all:nb). The same values when using all input variables except are marked in Table [16](#tab:cm:noDst:nb) and Table [17](#tab:cs:reverse:noDst:nb), and Table [18](#tab:cm:yzap:nb) and Table [19](#tab:cs:reverse:yzap:nb) when using only , , and variables as input. These results are displayed since the Naive Bayes model has low computational load and high accuracy. The combinations of input variables that produced a accuracy when used for the Naive Bayes model were not included since there are no errors in classification.

The confusion matrix for the Naive Bayes model when using all variables as input.

|  | Reference | | | | |
| --- | --- | --- | --- | --- | --- |
| Prediction | E | N | P | R | T |
| E |  |  |  |  |  |
| N |  |  |  |  |  |
| P |  |  |  |  |  |
| R |  |  |  |  |  |
| T |  |  |  |  |  |

The performance indicators derived from the confusion matrix for the Naive Bayes model when using all variables as input.

|  | Class | | | | |
| --- | --- | --- | --- | --- | --- |
| Statistics | E | N | P | R | T |
| Sensitivity |  |  |  |  |  |
| Specificity |  |  |  |  |  |
| PPV |  |  |  |  |  |
| NPV |  |  |  |  |  |
| DR |  |  |  |  |  |
| DP |  |  |  |  |  |
| BA |  |  |  |  |  |

The confusion matrix for the Naive Bayes model when using all variables except as input.

|  | Reference | | | | |
| --- | --- | --- | --- | --- | --- |
| Prediction | E | N | P | R | T |
| E |  |  |  |  |  |
| N |  |  |  |  |  |
| P |  |  |  |  |  |
| R |  |  |  |  |  |
| T |  |  |  |  |  |

The performance indicators derived from the confusion matrix for the Naive Bayes model when using all variables except as input.

|  | Class | | | | |
| --- | --- | --- | --- | --- | --- |
| Statistics | E | N | P | R | T |
| Sensitivity |  |  |  |  |  |
| Specificity |  |  |  |  |  |
| PPV |  |  |  |  |  |
| NPV |  |  |  |  |  |
| DR |  |  |  |  |  |
| DP |  |  |  |  |  |
| BA |  |  |  |  |  |

The confusion matrix for the Naive Bayes model when using only , , and as input.

|  | Reference | | | | |
| --- | --- | --- | --- | --- | --- |
| Prediction | E | N | P | R | T |
| E |  |  |  |  |  |
| N |  |  |  |  |  |
| P |  |  |  |  |  |
| R |  |  |  |  |  |
| T |  |  |  |  |  |

The performance indicators derived from the confusion matrix for the Naive Bayes model when using only , , and as input.

|  | Class | | | | |
| --- | --- | --- | --- | --- | --- |
| Statistics | E | N | P | R | T |
| Sensitivity |  |  |  |  |  |
| Specificity |  |  |  |  |  |
| PPV |  |  |  |  |  |
| NPV |  |  |  |  |  |
| DR |  |  |  |  |  |
| DP |  |  |  |  |  |
| BA |  |  |  |  |  |

# Discussion

The Neural Network model has the highest execution time for any subset of the input variables due to its complexity and extensive training, evident from the data in Table [13](#tab:time:total). However, in each case, it achieved a accuracy, which is shown in Table [6](#tab:stats:reverse:all), Table [7](#tab:stats:reverse:noDst), Table [8](#tab:stats:reverse:no$TEC$), Table [9](#tab:stats:reverse:coord), Table [10](#tab:stats:reverse:xyap), Table [11](#tab:stats:reverse:xzap), and Table [12](#tab:stats:reverse:yzap). The Naive Bayes model has the second highest accuracy, over . The Naive Bayes model achieved a accuracy, except when using all input variables, all input variables except , or only , , and . The difference in accuracy between the Neural Network and the Naive Bayes model is negligible, and the training time for the Neural Network model is more than twice as long. Despite their simple design and seemingly oversimplified assumptions, Naive Bayes classifiers have performed quite well in many complex real-world situations. In 2004., an analysis of the Bayesian classification problem showed that there are reasonable theoretical reasons for the seemingly incredible performance of Naive Bayesian classifiers .

All other models fail to achieve an accuracy over , except when using all input variables, including . The class was derived by thresholding the continuous value and discretizing it by converting it to a single character, so the original value should not be used as model input. This labeling method explains the increase in accuracy when adding the input variable in all except the two best-performing models. For example, the C5.0 model using Decision Trees did not consider any variable except when included in the input, indicated by a variable importance. The svmPoly model is consistently the worst-performing for any subset of input variables, never achieving an accuracy over . None of the models achieved an accuracy under , so they were all moderately successful.

When studying the performance of the Naive Bayes model by individual class in Table [14](#tab:cm:all:nb), Table [15](#tab:cs:reverse:all:nb), Table [16](#tab:cm:noDst:nb), Table [17](#tab:cs:reverse:noDst:nb), Table [18](#tab:cm:yzap:nb) and Table [19](#tab:cs:reverse:yzap:nb), samples of the T class are all correctly classified, but the testing is less extensive since it is the second smallest class. Samples of the R class are sometimes erroneously assigned to the N, P, or less commonly the T class. Samples of the P class are rarely mistaken for the N class. Two of the most common misclassifications were assigning samples of the N class with the highest detection rate and prevalence to the P or R classes with the third and second highest detection rate and prevalence. This is due to the largest number of samples in the N class between the P and R classes with the narrowest range. Samples of the N class are least often included in the E class, whose range is the furthest apart from the N class. A sample of the E class was attributed to the N class only once. The E class is the smallest with only three samples, which must be accounted for when interpreting these results, such as the lowest balanced accuracy of and sensitivity of when using only , , and as input, or the lowest positive predictive value of when using all variables as input.

# Conclusion

The presented study aims to classify ambient conditions of space weather events for sub-equatorial regions. GNSS PNT performance is significantly affected by such events. it would be beneficial to indicate to users that a geomagnetic/ionospheric storm is in progress.

Machine learning classification models were applied to descriptions of the geomagnetic field expressed in and other input data. It was assumed observations contained independent variables to generate the dependent variable representing the class.

Statistical analysis confirmed that other variables change distribution based on , not . Continuous values in different ranges were converted into discrete classes based on statistics, previous theories, and research.

An SVM with a Polynomial Kernel, C5.0 Decision Tree, Naive Bayes, Neural Network, PLS, FDA, and PCA Neural Network model created a -based classification from multiple combinations of input variables.

The Naive Bayes model achieved perfect accuracy for most tested combinations of input variables, and over in all cases. It is slightly less successful than a Neural Network, and the total execution time is at least two times shorter, making it more suitable for use in compact, low-performance, and low-cost portable devices such as smartphones.