BENCHMARKING MACHINE LEARNING MODELS FOR COLLISION RISK PREDICTION IN LOW EARTH ORBIT

Rasit Abay[(1)], Francisco Caldas[(1)], Mariana Filipe[(1)], Marta Guimarães[(1)]

[(1)]Neuraspace, R.Pedro Nunes, Coimbra, Portugal, {rasit.abay, francisco.caldas, mariana.filipe, marta.guimaraes}@neuraspace.com

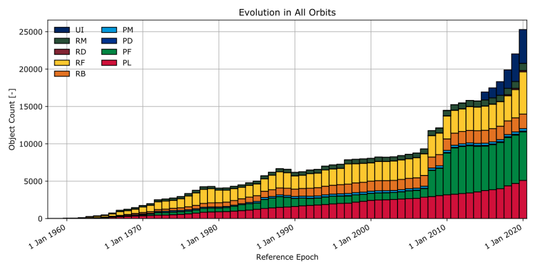
ABSTRACT

The number of trackable resident space objects (RSOs) is increasing due to new launches and novel space surveillance sensors. Therefore, it is desired to automate the collision risk prediction and mitigation (CREAM) to keep pace with the number of conjunction data messages (CDMs). Recently, the feasibility of leveraging machine learning models has been investigated to reduce the false detections in the literature, and ESA collision risk prediction challenge hosted on Kelvins platform provided the publicly available dataset. The proposed work benchmarks the machine learning models and frameworks that have been studied for collision risk prediction to evaluate their suitability for real world deployment. Therefore, machine learning models are benchmarked against the naive solution, which considers the risk value at cut-off day as risk at close approach time, using JSpOC CDMs issued for satellites that ESA maintains. This work shows that Bayesian neural networks, Siamese-embedding, Boosting decision trees models, and Linear Discriminant Analysis (LDA) are promising machine learning approaches for collision risk prediction.

1 INTRODUCTION

Since 1957, thousands of satellites have been launched. Currently, there are 2600 active satellites in low earth orbit (LEO) [1,2]. The number of resident space objects (RSOs) has been increasing in the last 10 years, as shown in Figure 1. In addition, with the launch of mega-constellations and advanced space surveillance systems, the number of trackable RSOs will also increase more drastically when compared to the current pace and proliferation. Adequate space debris mitigation strategies are the key to ensure that spaceflight does not become impossible to manage. In-orbit collisions easily can reach the collisional cascading point known as "Kessler Syndrome" [3] unless effective space traffic management practices are introduced.

Collision risk estimation and mitigation (CREAM) is an essential routine task to keep satellites safe from hazards due to the other orbiting RSOs [5]. Every time a close encounter occurs, several conjunction messages are generated in order to monitor the criticality of the event. Current space surveillance capabilities can not provide persistent observations for all RSOs. Not only the physical properties of space debris are unknown and modeled with simplistic approximations (cannonball approximation), but also the space environment state and the associated interactions with RSOs can not be modeled with desired precision. These uncertainties in states of RSOs and in the space environment lead to hundreds of conjunction data messages generated per week that need to be analysed by human experts, and only very few end up as high-risk close encounters. Therefore, the automation of CREAM is desired to scale up to the number of CDMs generated every day to



reduce the workload on human experts and lead to better decisions by allowing late manoeuvre decisions [6].

Figure 1. Evolution of RSOs in space from 1957 to 2020 [4].

In 2019, ESA organized the Collision Avoidance Challenge [7] that invited multidisciplinary teams around the World to develop machine learning (ML) models that can predict the criticality of close encounters. During the competition, it has been shown that a naive approach (using the latest available risk value at cut-off time as the target risk value) is a strong predictor, as it yields good results. Therefore, the proposed work benchmarks the performances of ML models against the naive solution. During the challenge, the Siamese neural networks and gradient boosting trees have performed marginally better than the naive approach, and the highest ranking solution at the challenge has leveraged a step-by-step approach to optimise the private dataset [6].

After the challenge, research has been conducted to investigate the feasibility of leveraging machine learning for CREAM. Metz has investigated various machine learning models to predict collision risk by regressing the final chaser position uncertainties, and using those predictions to compute the probability of collision [8], Acciarini et. al. have incorporated simulated data that are conditioned on the real data into probabilistic programming framework [9], Pinto et. al. have studied the feasibility of generating CDMs using Bayesian deep learning with recurrent neural networks and Monte Carlo dropout [10].

This work aims at benchmarking state-of-the-art ML models in the literature that have been investigated for automating and improving the collision risk prediction at real deployment environment. Therefore, the proposed work conducts benchmarking for ML models in the literature against the naive solution, which is known to be a strong predictor [7] for the problem, and this is the significance of the work.

The outline of the rest of the proposed work is :Section 2 introduces the data and the main characteristics of the features considered for modeling. Section 3 gives some background regarding the ML models used in this work, as well as the architectures and parameters chosen. Section 4 presents the obtained results and lastly, Section 5 discusses the main conclusions of this work.

2 DATA

The data used in this work is provided by ESA and consists of CDMs from 2015 to 2019. CDMs corresponding to the same conjunction event are grouped under the same event ID. Therefore, each event represents a time series of CDMs. All variables and parameters contained in each CDM consist of propagated values at the closest approach time (TCA) (as opposed to values corresponding to the creation date of the CDM). As the operator needs time to make the decision whether to maneuver or not, the events used during this work satisfied the following constraints [6]:

i) The event should contain at least 2 CDMs, one to learn from and the other to be used as the target.
ii) The last CDM released for the event should be within 1 day to the TCA.
iii) The first CDM released for the event should be at least 2 days before TCA and all the CDMs that are within two days from TCA are removed.

By performing an exploratory data analysis (EDA) it is possible to understand the main characteristics of the dataset, detect noise, and discover patterns and trends in the data.

This dataset consists of 2202566 samples/CDMs, grouped into 119852 events, and 230 features. Some of these features can not be utilised, as they are completely missing or the amount of missing data is beyond 50%. Others, specifically the ones that reference the chaser and target id, are repeated for the different databases, such as DISCOS and NORAD.

The main goal of this work is benchmarking machine learning models in the literature investigated for predicting the collision risk at close approach time. The risk value is defined to be the logarithm to the base 10 of the probability of collision, which is a variable present in the given set of features. The distribution of this feature shows that the dataset is imbalanced. In other words, the minority class (high-risk) is relatively rare when compared to the majority class (low-risk). The proportion of these classes is 0.84% of high-risk CDMs and 99.2% of low-risk. Dealing with an imbalanced dataset is the major challenge for the proposed work.

The first step in the data preparation process is the removal of events with only one CDM as well as events where it is not confirmed that no manoeuvre is performed. Some parameters have significant outliers. For example, the covariance matrix of the target and chaser has a maximum value for its entries of $4.068062x10^{15}\,m^2$. That value repeats itself in more than 6000 CDMs, and besides being unrealistic from a physical perspective, it also affects others features, most importantly, the probability of collision. When this outliers occurs, due to how the probability of collision is computed, the probability is concentrated around $\approx 1x10^{-14}\,m^2$.

During the data cleaning phase, the dataset is kept as original as possible, in order to benchmark machine learning models by their performances in real World, and nothing is removed unless errors are detected.

To increase the performance of the models, new features are derived from the dataset, namely the mahalanobis distance between chaser and target and the determinant of the covariance matrix, while other features are engineered to acquire some of the time-series characteristics of the sequence of CDMs that belong to an event.

3 MODELS

The dataset is divided in two sets of data: training set and test set for this work. The training set is used to fit the ML models, while the test set is not used until the very end to evaluate the final model. The split ratio used is 80:20. After this split, the training set is randomly split to perform cross-validation.

3.1 Naive Solution

It is important to establish the baseline performance in order to provide a point of comparison for each model.

Regressor

Defining $\hat{y}_i$ as the predicted risk value for the i^{th} event, and y_{-2i} as the latest known risk value of the CDM released at least two days before the TCA, then the baseline's prediction can be defined as:

$$\hat{y}_i = y_{-2i}. \quad (1)$$

Classifier

Besides predicting the exact risk value at TCA, another major goal of this problem is to predict whether that risk value is high or low, that is, the classification of events into high-risk or low-risk. The threshold used to binarize the data is -6, such that: 1 stands for high-risk ($y_i \geq -6$) and 0 represents low-risk events ($y_i < -6$).

3.2 Bayesian Neural Network (BNN)

A BNN can be defined as a stochastic neural network, (a network whose weights and biases are expressed as a distribution rather than a deterministic value) trained using Bayesian inference [11]. This type of network is an interesting tool for active learning, as it provides an approach to quantify uncertainty in deep learning and, consequently, allows easier interpretability of the predictions when compared to non-BNNs. This uncertainty can be related to the estimation of the model's parameters (epistemic uncertainty), or related to the aleatoric nature of the data used as input (aleatoric uncertainty) [12].

The model outputs a distribution estimated for the target. This is accomplished by placing a prior distribution $p(\theta)$ over the model parametrization θ and finding the posterior distribution, $p(\theta|D)$ (where D represents the training set) given by the Bayes' theorem:

$$p(\theta|D) = p(D|\theta)p(\theta)/p(D). \quad (2)$$

The posterior distribution can be used to compute the output y' given a certain input x' :

$$p(y'|x', D) = \int_{\theta} p(y'|x', \theta)p(\theta|D)d\theta. \quad (3)$$

Computing the distribution $p(\theta|D)$ analytically usually becomes an intractable problem. To address this major issue, one possible solution is to use a variational inference approach. With this method, a variational distribution q is defined to approximate the posterior distribution. In other words, the objective is to find the values of β, such that the difference between the variational distribution $q(\theta|\beta)$ and the true posterior distribution $p(\theta|D)$is minimum. This difference is measured by the Kullback-Leibler (KL) divergence, therefore the problem consists in minimizing the KL-divergence, $KL(q(\theta|\beta)||p(\theta|D))$, given by:

$$KL(q(\theta|\beta)||p(\theta|D))= \\ = KL(q(\theta|\beta)||p(\theta)) - \mathbb{E}_{q(\theta|\beta)}[log\, p(D|\theta)] + \\ log\, p(D). \quad (4)$$

Since the model evidence ($log\, p(D)$) is a constant, minimizing $KL(q(\theta|\beta)||p(\theta|D))$ is equivalent to maximizing the evidence lower bound (ELBO) function:

$$ELBO = log(p(D)) - KL(q(\theta|\beta)||p(\theta|D)), \quad (5)$$

where $ELBO = \mathbb{E}_{q(\theta|\beta)}[log\, p(D|\theta)] - KL(q(\theta|\beta)||p(\theta))$. The second term of the right-hand side of the equation can be analytically computed and the first term can be estimated by sampling parameter values from the variational distribution, that is, via Monte Carlo.

The model's target is defined to be the change in risk value between the CDM at least 2 days away from TCA and the event's final CDM. This approach has the following advantages: i) reduces the bias towards the most represented target risk value (-30) therefore facilitating the learning task and ii) leverages the baseline solution [7].

3.3 Siamese-Embedding Model

Siamese models have been developed for learning similarities between input pairs. The first implementation of Siamese neural networks dates back to 1994, and it has been proposed for distinguishing forged signatures from the original ones [13]. Compared to the classical approaches in computer science and statistics for determining similarity, namely Euclidean distance, Pearson correlation coefficient, and others, Siamese neural networks provide a framework that learns the similarities between input pairs by sharing the same neural network architecture and outputting the similarity measures [14]. Siamese models map input pairs in a lower dimensional space that allows to evaluate the distance between their latent representations.

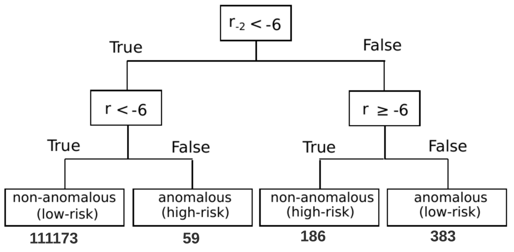


Figure 2. Number of anomalous and non-anomalous events from the training set.

For Siamese-embedded models, the collision risk prediction problem is casted as an anomaly detection problem for a total of 4 classes using the naive solution (Eq. 1). The naive solution assumes the final PoC available at the cut-off time, which is 2 days before the close approach, as the target risk value. The proposed model regards the false positives of the naive approach as anomalies, and it learns the dissimilarity between false positives and true positives. The fact that the physical models, i.e. naive solution, have an inferior performance for some samples is mostly due to the errors in propagating states and uncertainties to the close approach time (dynamical errors), lower-fidelity in modeling space environment (non-gravitational forces), simplistic physical modeling for space objects (non-gravitational forces), and errors in the observation data. The proposed model learns from the patterns of input features over the dataset, and detects false positives of the naive solution. Since Siamese models can be used for one-shot learning, a smaller number of samples for each class can be utilised for building generalisable machine learning models. Therefore, the proposed Siamese-embedded model can be trained by leveraging the naive solution, which has been shown to be a strong predictor [15].

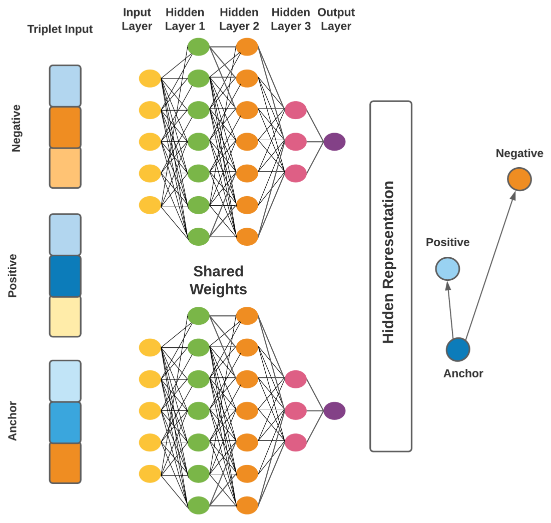


Figure 3. Siamese neural networks with triplet loss.

The total number of features used for training the proposed model are 16 which have been selected based on how discriminative they are for the proposed model experimentally. There are 186 samples for non-anomalous (high-risk) and 383 samples for anomalous (low-risk) classes. Since similarity-based learning using triplet loss is leveraged for the proposed model, 71052 input triplets are generated. Each triplet has two samples from the similar class and one from the dissimilar class (Figure 3). To avoid the drastic impact of outliers in the data, the triplet input features are standardised using only the training data statistics, i.e. mean and standard deviation of each feature. Five-fold cross validation is used for validation of the results from the training data.

Three different neural network architectures, namely fully-connected, convolutional, and recurrent (LSTM) neural networks, are investigated with the proposed Siamese-embedded model. Since the number of features are small, simple neural network architectures are preferred over complex ones Figure 4. The triplet loss (Eq. 6) enforces anchor representations to positive representations and further away from the negative ones during the training phase.

$$L = max(d(a,p) - d(a,n) + margin, 0) \quad (6)$$

where d is the Euclidean distance and margin is selected as 0.2 for the proposed model.

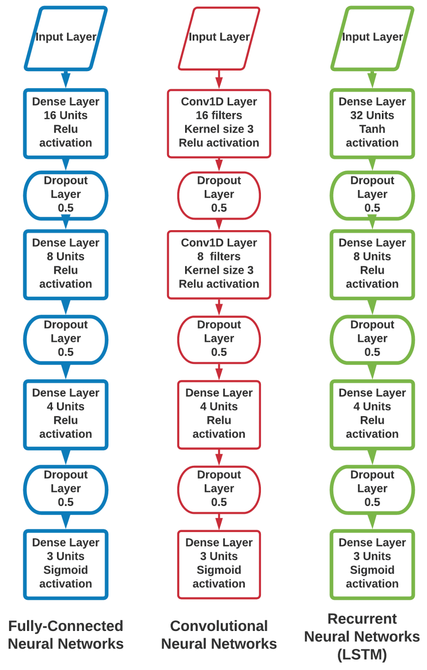

Input Layer
Dense Layer 16 Units Relu activation
Dropout Layer 0.5
Dense Layer 8 Units Relu activation
Dropout Layer 0.5
Dense Layer 4 Units Relu activation
Dropout Layer 0.5
Dense Layer 3 Units Sigmoid activation
Fully-Connected Neural Networks
Input Layer
Conv1D Layer 16 filters Kernel size 3 Relu activation
Dropout Layer 0.5
Conv1D Layer 8 filters Kernel size 3 Relu activation
Dropout Layer 0.5
Dense Layer 4 Units Relu activation
Dropout Layer 0.5
Dense Layer 3 Units Sigmoid activation
Convolutional Neural Networks
Input Layer
Dense Layer 32 Units Tanh activation
Dropout Layer 0.5
Dense Layer 8 Units Relu activation
Dropout Layer 0.5
Dense Layer 4 Units Relu activation
Dropout Layer 0.5
Dense Layer 3 Units Sigmoid activation
Recurrent Neural Networks (LSTM)

Figure 4. Neural network architectures used in the proposed framework.

For validation and testing, anchors come from validation and test dataset, and positive and negative samples are sampled from the training dataset. The distances between anchor-positive and anchor-negative are computed and stored for determining the class that the sample belongs to by leveraging the resultant distribution of differences (Figure 5).

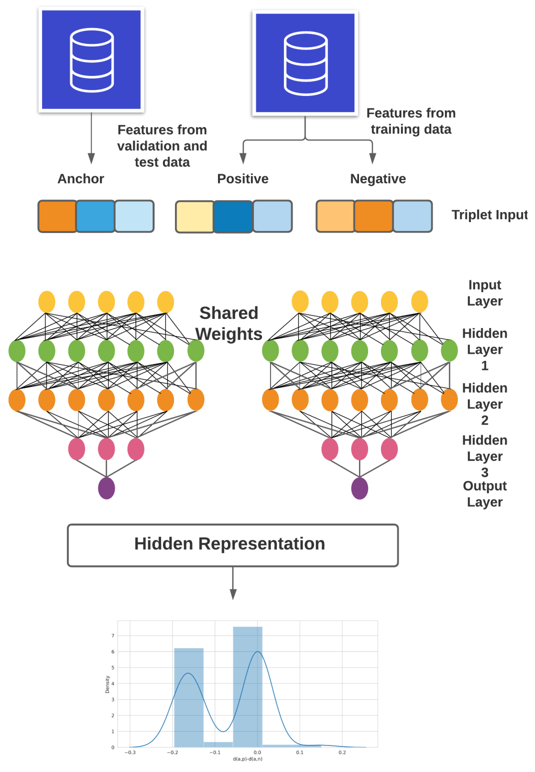


Figure 5. The visual representation of validation and testing within the proposed framework.

3.4 Categorical Boosting (CatBoost)

A decision tree algorithm is a supervised learning method that can be used for classification. These methods are popular due to their interpretability and capability to capture non-linear patterns with good performance [16].

Gradient Boosting is a learning algorithm that leverages weak learners, such as decision trees, to produce a predictive model. These weak learners are sequentially fitted using the information of the loss function of the previous one. This iterative process intends to minimize the error using the gradient descent technique.

This powerful model framework, GBM, can be implemented in the form of the well-known XGBoost (which stands for extreme gradient boosting) [17], LightGBM [18], or CatBoost [19]. All algorithms are investigated, however, the performance of CatBoost is relatively more promising.

The main differences in the implementation of these techniques consists of feature-splitting, leaf growth, missing values handling, feature importance methods, categorical feature handling [20].

CatBoost's main advantages over the standard gradient boosting implementations are in:

- Handling categorical features in a more efficient way;
- Using a new schema to select the next tree structure. This helps to reduce the overfitting;
- Its GPU implementation enables faster training.

This model's implementation is very straight forward. The *catboost* package is already available as a python library and it does not require any preprocessing.

This model is used for classification, therefore before training, the risk feature is binarized into two classes: high-risk and low-risk. Using the same threshold as the previous models -6.

The features used in this approach contain approximately 13% categorical features, which aren't preprocessed because they are encoded within the model.

As mentioned before the dataset is considerably imbalanced, therefore the way to overcome this problem is by tuning the parameter that attributed weights to the classes the best way possible.

3.5 Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) is a model that tries to find a linear combination of variables that can separate two or more classes. It is different from Principal Component Analysis (PCA) in the sense that LDA uses the class label and maximizes the differences between each class. Unlike PCA, LDA creates $n - 1$ Linear Discriminant Functions, being n the number of classes, whereas PCA can create up to p Principal Components, being p the number of variables, and it does not take in consideration the classes of the observations. If the number of variables is smaller than the number of classes then LDA creates p LD functions. LDA in particular assumes that for each group, the conditional probability density function

$p(x|y = i)$is a Multivariate normal(μ_i, Σi), and that variance is the same across groups, meaning: $\Sigma i = \Sigma j = \Sigma\, \forall i, j \in C$, being C the set of classes. LDA as a predictive model also suffers from multicollinearity, which is a phenomenon that happens when one predictor variable can be obtained from a linear regression of other variable also used in the model, in that case the LD coefficients of the linear correlated variables may change erratically, without affecting the accuracy or reliability of the model. Our dataset suffers from this, since many variables are highly correlated, so during our application of the LDA we will refrain from trying to interpret the coefficients in depth. While LDA can be used for dimensionality reduction in this case it is used for classification.

For classification, the Bayes Rule is used and the classification is done by choosing the $i \in C$, that maximizes:

$$P(y = i|x) = \frac{P(x|y=i)P(y=i)}{P(x)} \tag{7}$$

by doing the log probability the above can be written as such [21]:

$$\log P(y = i \mid x) = -\frac{1}{2}(x - \mu_i)^T \Sigma^{-1}(x - | \\ + \log P(y = i) + Cst. \tag{8}$$

Meaning that the class probability depends on which class mean is closest by the Mahalanobis distance plus a value that represents the prior probability of belonging to the class [22].

4 RESULTS

A key factor in assessing the performance of the model is the evaluation criteria. A common technique used to determine the performance of a classifier is through the use of the confusion matrix. In a confusion matrix, true positives (TP) is the number of instances correctly classified as positive, false positives (FP) is the number of negative instances incorrectly classified as positive, true negatives (TN) is the number of negative instances incorrectly classified as negative and false negatives (FN) is the number of positive instances correctly classified as negative. In the framework of imbalanced datasets, the evaluation of the classifier's performance must consider the class distribution. To avoid producing a biased illusion on imbalance data, some metrics can be derived from the confusion matrix:

$$precision = \frac{TP}{TP+FP} \tag{9}$$

$$recall = \frac{TP}{TP+FN} \tag{10}$$

In the context of this problem, precision quantifies what proportion of predicted high-risk events is actually correct, and recall measures what proportion of real high-risk events is correctly identified.

Another commonly used metric is the F-score, which is computed from precision and recall

$$F_\beta = (1 + \beta^2)\frac{precision \cdot recall}{\beta^2 precision + precision}, \tag{11}$$

where β is chosen such that recall is considered β times as important as precision. The traditional F-score is the F_1 score, which is the harmonic mean of precision and recall. In this work, F_2 score is also considered as it gives more attention on minimizing FN than minimizing FP.

In regression analysis, all the metrics mentioned above can not be applied. Thus, root mean squared error (RMSE) is chosen. RMSE is a commonly used metric given by:

$$RMSE = \left(\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)\right)^{1/2} \tag{12}$$

	F_1	**F_2**	**Precision**	**Recall**	**RMSE**
Naive	0.497	0.638	0.362	0.791	**29.06**
BNN	-	-	-	-	29.22
Siamese Embedding Conv1D	**0.552**	0.631	**0.458**	0.696	-
Catboost	0.388	0.568	0.264	**0.839**	-
LDA	0.472	**0.640**	0.329	**0.839**	-

Table 1. Summary of the obtained results with the proposed models and naive solution.

Table 1 presents the obtained results with the proposed models and the naive solution. The RMSE of BNN for predicting the modified target is close to the naive approach. It is a promising result that proves that one can use BNN to directly predict the risk at TCA, with the added value of providing the posterior distribution of the predictions, which is beneficial for space operators because they can evaluate the uncertainty of the predictions of the model.

The Siamese embedding model is a similarity-based learning model, and it succeeds at few-shot learning. Since it is stacked on top of the naive solution, it performs marginally better regarding F_1 metric and precision, and this is expected because the proposed model cleans the false positives.

Catboost is an efficient gradient boosting on decision trees algorithm. Gradient boosting trees can be sensitive to the variations in the input, yet they can provide insights regarding the importance of features at the same time. Although the recall of the Catboost model is better than the naive solution, the precision, F_1 and F_2 metrics are relatively inferior.

LDA is a simple model that learns the separation of classes by projecting the features on lower dimensional axes. LDA is also stacked on top of the naive solution, However, unlike the siamese-embedding models, it is trained to clean the false negatives, meaning that it succeeds at finding events that only become high-risk closer to tca, and therefore are not found by the current warning thresholds used by Space Operators. Its F_2 and recall metric is better than the naive solution.

5 CONCLUSION AND FUTURE WORK

The major goal of this paper is to benchmark the performances of ML models against the naive solution. From Section 4 it is possible to observe that Catboost and LDA models are the best to predict TN. Concerning the F_1 score and precision, the Siamese embedding model outperforms the naive solution. As the only regression model is the BNN it can only be compared to the naive solution, which is slightly better.

With the obtained results, it is possible to conclude that Siamese-embedding models and linear discriminant analysis are promising machine learning approaches for improving the naive classification.

The performance of all models can be improved by conducting comprehensive exploratory data analysis and fusing data from various resources. In addition, it is recommended to compare the performances of machine learning models against the naive approach to make sure developed models can be deployed to the real world.

For future work, several data-centric approaches can be performed. For instance, physics-aware feature engineering, and also the incorporation of the physical properties of the problem to indirectly predict the PoC by predicting its components. Regarding the data analysis, more exploratory investigation can be performed to leverage the data specifics and peculiarities.

6 ACKNOWLEDGEMENTS

The team would like to thank ESA/Space Debris Office for the unyielding support and for providing the data used in this paper.

7 REFERENCES

1 UCS Satellite Database, In-depth details on the 3,372 satellites currently orbiting Earth, including their country of origin, purpose, and other operational details. 2021 https://ucsusa.org/resources/satellite-database?_ga=2.206523283.1848871521.1598077135-464362950.1598077135

2 ESA, Space debris by the numbers. 2021 https://www.esa.int/Safety_Security/Space_Debris/Space_debris_by_the_numbers

3 Kessler, D.J. and Cour‐Palais, B.G., 1978. Collision frequency of artificial satellites: The creation of a debris belt. *Journal of Geophysical Research: Space Physics*, 83(A6), pp.2637-2646.

4 ESA Space Debris Office. ESA's Annual Space Environment Report, 2020.

5 Alfano, S., 2005, August. Collision avoidance maneuver planning tool. In *15th AAS/AIAA astrodynamics specialist conference* (pp. 7-11).

6 Flohrer, T., Krag, H., Merz, K. and Lemmens, S., 2019, September. CREAM-ESA's Proposal for Collision Risk Estimation and Automated Mitigation. in *Proceedings of the Advanced Maui Optical and Space Surveillance Technologies Conference (AMOS)*.

7 Uriot, T., Izzo, D., Simoes, L., Abay, R., Einecke, N., Rebhan, S., Martinez-Heras, J., Letizia, F., Siminski, J. and Merz, K., 2020. Spacecraft Collision Avoidance Challenge: design and results of a machine learning competition. arXiv preprint arXiv:2008.03069.

8 Metz, S., 2020. *Implementation and comparison of data-based methods for collision avoidance in satellite operations*. Technische Universität Darmstadt.

9 Acciarini, G., Pinto, F., Metz, S., Boufelja, S., Kaczmarek, S., Merz, K., Martinez-Heras, J.A., Letizia, F., Bridges, C. and Baydin, A.G., 2020. Spacecraft collision risk assessment with probabilistic programming. arXiv preprint arXiv:2012.10260.

10 Pinto, F., Acciarini, G., Metz, S., Boufelja, S., Kaczmarek, S., Merz, K., Martinez-Heras, J.A., Letizia, F., Bridges, C. and Baydin, A.G., 2020. Towards automated satellite conjunction management with Bayesian deep learning. arXiv preprint arXiv:2012.12450.

11 David J. C. MacKay. A practical Bayesian framework for backpropagation networks. *Neural Computation*, 4(3):448–472, 1992.

12 Laurent Valentin Jospin, Wray L. Buntine, Farid Boussaïd, Hamid Laga, Mohammed Bennamoun. *Hands-on Bayesian Neural Networks - a Tutorial for Deep Learning Users*. ACM Comput. Surv. 1, 2020.

13 Bromley, J., Guyon, I., LeCun, Y., Säckinger, E. and Shah, R., 1994. Signature verification using a" siamese" time delay neural network. *Advances in neural information processing systems*, pp.737-737.

14 Chicco, D., 2021 Siamese neural networks: An overview. *Artificial Neural Networks*, pp.73-94.

15 Abay, R. and Gupta, K. "Towards Automated Collision Risk Prediction for Spacecraft using Deep Learning," Submitted to *ACTA Astronautica*, 2020.

16 E. Alpaydin. Introduction to Machine Learning. *The MIT Press, second edition*, 2009. ISBN 9780262012430.

17 Tianqi Chen and Carlos Guestrin, XGBoost: A Scalable Tree Boosting System, In *22nd SIGKDD Conference on Knowledge Discovery and Data Mining*, 2016.

18 Kinnander, M. Predicting profitability of new customers using gradient boosting tree models : Evaluating the predictive capabilities of the XGBoost, LightGBM and CatBoost algorithms, 2020.

19 Anna Veronika Dorogush, Vasily Ershov, Andrey Gulin. Workshop on *ML Systems at NIPS 2017*

20 Bentéjac, C., Csörgő, A. & Martínez-Muñoz, G. A comparative analysis of gradient boosting algorithms. *Artif Intell Rev* 54, 1937–1967 (2021).

21 R. O. Duda, P. E. Hart, D. G. Stork. Pattern Classification (Second Edition), section 2.6.2.

22 Hastie T., Tibshirani R., Friedman J.*The Elements of Statistical Learning* , Section 4.3, p.106-119, 2008.