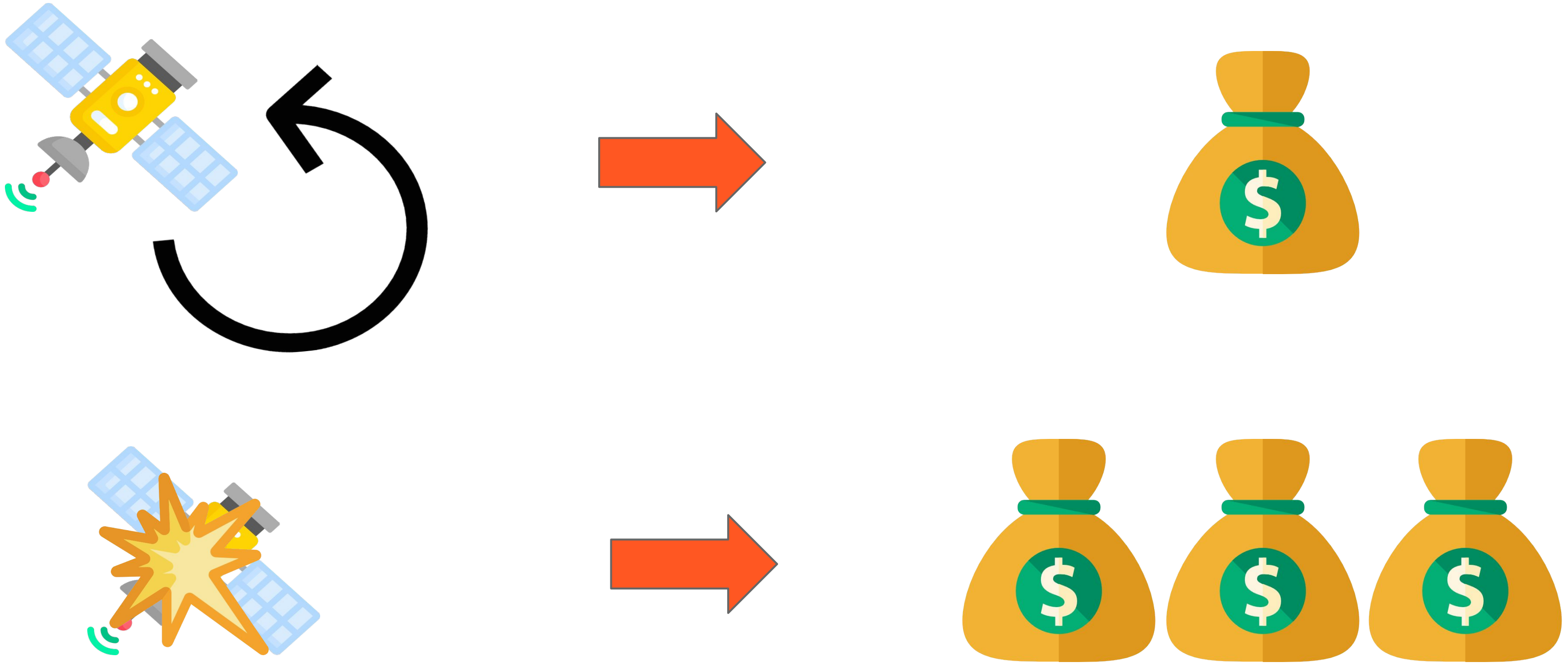
Satellite Collision Avoidance

Carlos Andrés Ramiro
Sergio Pérez Morillo
Jaime Pérez Sánchez

Description and Objectives

Space Debris Office

Description and Objectives



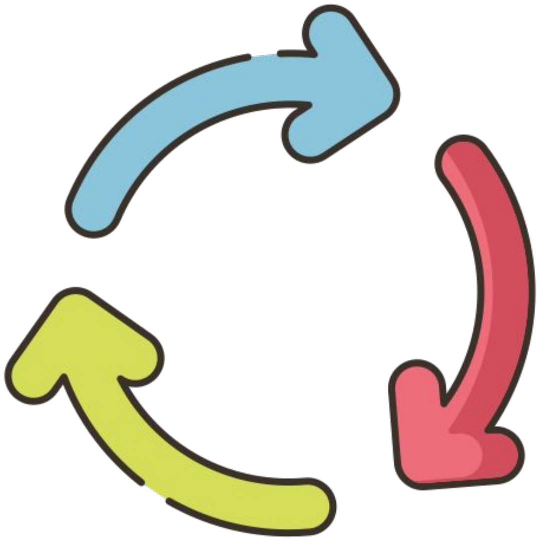
Description and Objectives

GOAL

Predict high-risk events two days in advance.

Our Approach

1. Data Description
2. Data Wrangling
3. Exploratory Data Analysis
4. Feature Selection
5. Feature engineering
6. Deep Learning Modeling



1. Data Description

Our Dataset

Our dataset has **162,634 rows** and **103 columns**.

We have **13,154 unique events**, each potential collision event is made of several rows.

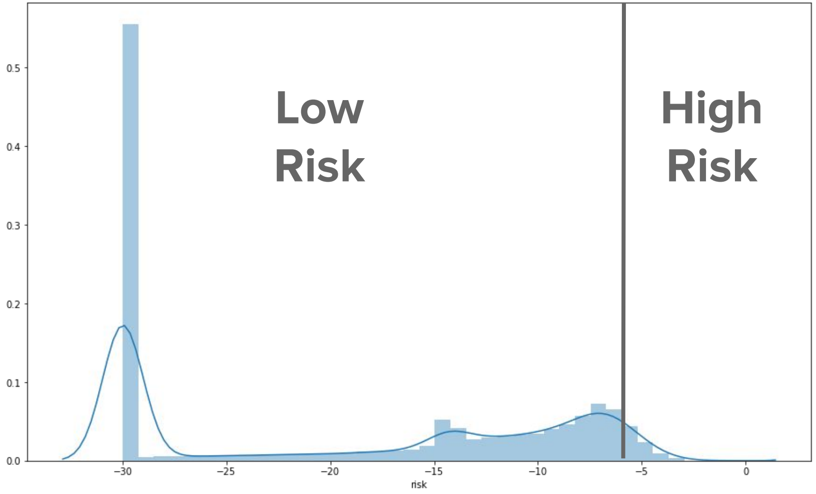
There are **3 categorical features** and the rest are numerical.

The features are related to the **movement** of objects in space.

The **target feature** is the **risk** of collision between the satellite and the chasing object.

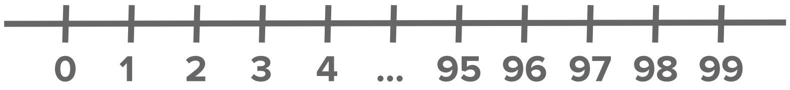
1. Data Description

Main Problems

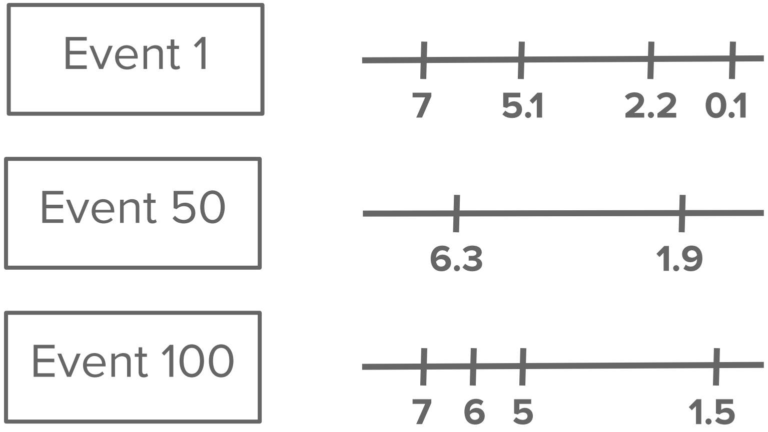


Target Feature Distribution

Regular time series



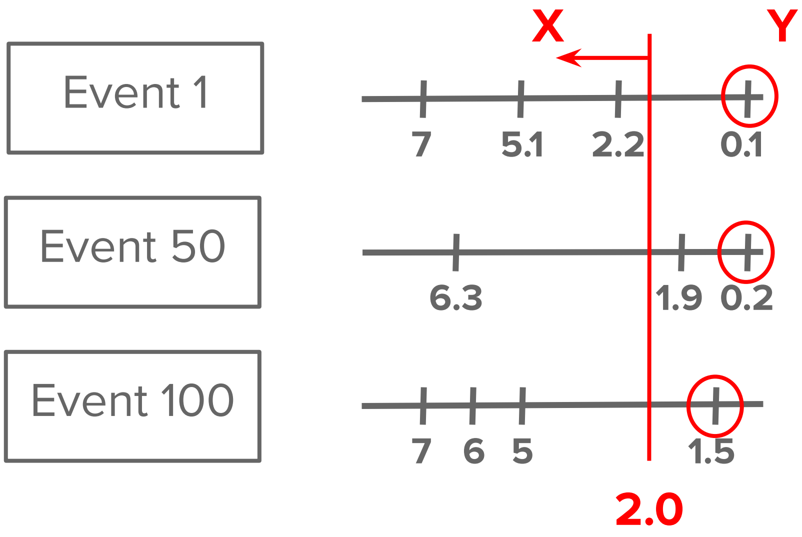
Our time series



2. Data Wrangling

Data for DL Models

X and Y arrays

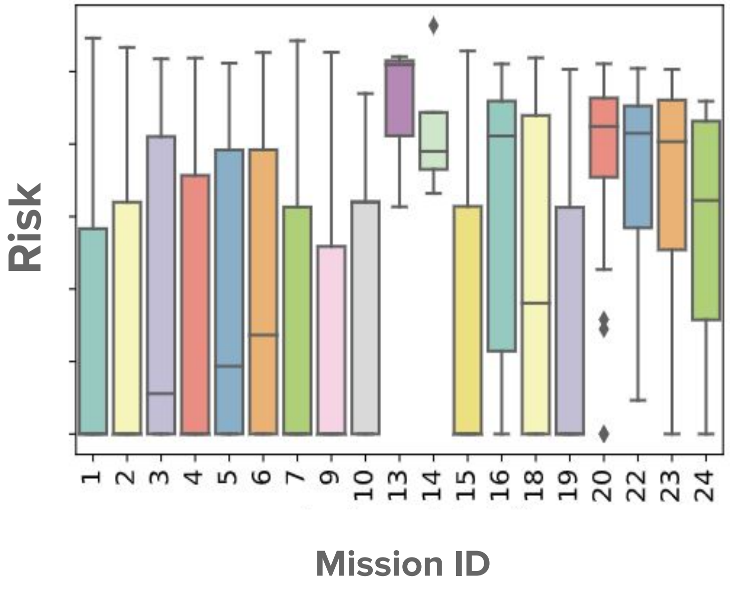


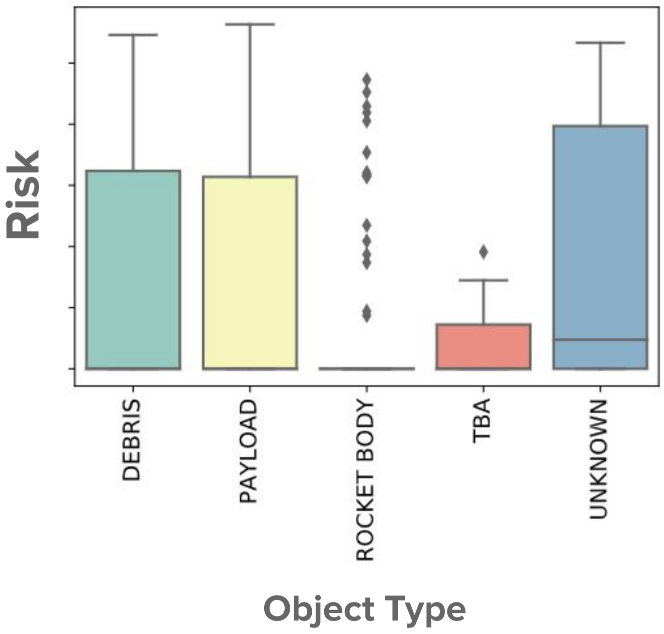
Data Augmentation

1. Imputing NaNs (kNN)
2. Padding Sequences
3. Overlapping Windows
4. Siamese Pairs
5. Oversampling (SMOTE)
6. Moving High-Risk Threshold

3. Exploratory Data Analysis

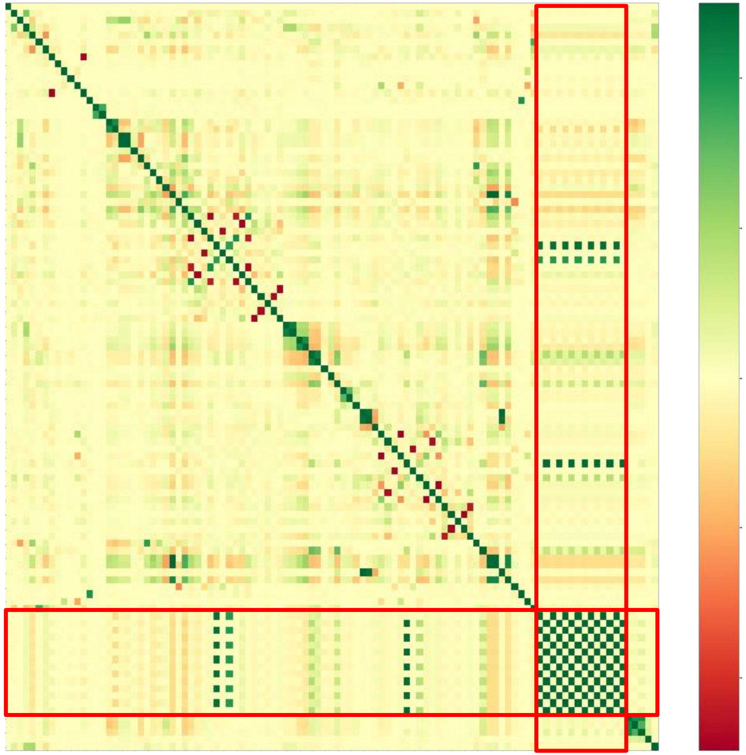
Categorical Features





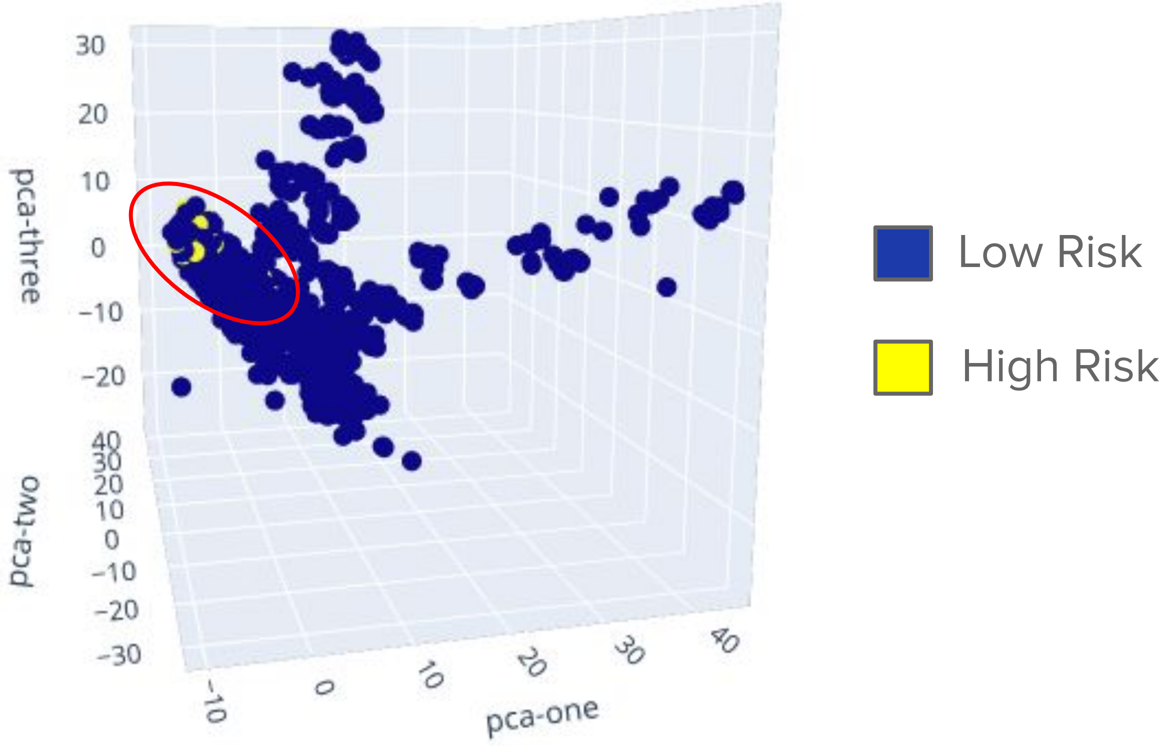
3. Exploratory Data Analysis

Correlations



3. Exploratory Data Analysis

PCA



4. Feature Selection

Filter
Univariate Selection

Wrapper
RFE and CV-RFE

Embedded
Feature Importance

Target Feature Analysis

+

Time Series Analysis

5. Feature Engineering

Brute Force

~2,000 new features transforming the best ones using primitives: +, -, *, /, x^2, log(x), etc.

Event Information

Event length

High risk timestamps

Mean time step length

Mean risk changes

Positive risk changes

Negative risk changes

Domain-Specific

Route of the chaser

6. Deep Learning Modeling

Previous Considerations

Data Splitting Train, Validation and Test Sets	**Data Scaling** Min-Max, Standardization and Quantiles
Problem Modeling Classification, Regression and Anomaly Detection	**Hyperparameter-Tuning** Random Search with Keras Tuner and Trial and Error

6. Deep Learning Modeling

Evaluation Tools

Evaluation Metric

$$L(r, \hat{r}) = \frac{1}{F_2} MSE(r, \hat{r})$$

Metric Configuration:

1. **F-beta** with β=2
2. **MSE** just for high-risk events

Baselines

Competition Baseline

Predicting always a constant value

e.g.: $y_1 = -5$, $y_2 = -5$, $y_3 = -5$

Time Series Baseline

Predicting always the last risk value in X for each event

e.g.: X = -1.5, → y = -1.5

6. Deep Learning Modeling

FeedForward Neural Networks

Layers: 5 Dense

Neurons: 256, 128, 64, 16, 4

FFN configuration:

1. Activation: ReLu
2. L2 Regularizer: 0.01

Optimizer: RAdam

Batch: 32, **Epochs:** 40

Class Weights: 45/2

	L	MSE	f-beta
FFN	**1.16**	0.555	0.479
Competition	9.07	0.461	0.050
Time Series	0.86	0.423	0.487

6. Deep Learning Modeling

Recurrent Neural Networks

Layers: 4 LSTM + 1 Dense

Neurons: 50, 25, 12, 5, 1

LSTM configuration:

1. Dropout: 0.1, R-Dropout: 0.2
2. Batch normalization
3. L2: 0.001, Stateless

Optimizer: Adam, **Timesteps:** 8

Batch: 256, **Epochs:** 100

Class Weights: 300/1

	L	MSE	f-beta
LSTM	**0.429**	0.186	0.433
Competition	6.819	0.340	0.050
Time Series	0.600	0.372	0.620

6. Deep Learning Modeling

Convolutional Neural Networks

Layers: 2 VGG16-like Conv1D blocks
+ 2 Dense

Total params: ~10K

Optimizer: Nadam, **Timesteps:** 8

Batch: 64, **Epochs:** 50

Class Weights: 100/1

Input shape approaches:

1. (events, features*timestep, 1)
2. (events, timestep, features)

	L	MSE	f-beta
Conv1D	**0.534**	0.231	0.432
Competition	6.819	0.340	0.050
Time Series	0.600	0.372	0.620

6. Deep Learning Modeling

Autoencoders

Modeling: Anomaly detection

Train set: Low risk events

Test set: Entire Dataset

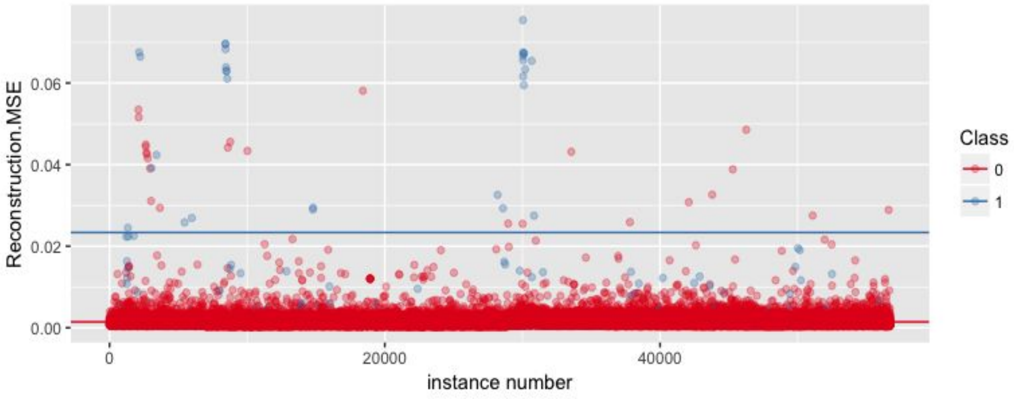
Layers: 4 LSTM

Neurons: 32, 16, 16, 32

LSTM-Activation: SELU

Optimizer: Nadam, **Timesteps**: 10

Batch: 64, **Epochs**: 50



It did not work!

6. Deep Learning Modeling

Siamese Neural Networks

Layers: 2 LSTM + 1 Manhattan distance

Neurons: 32, 16

LSTM configuration:

1. Dropout: 0.25, R-Dropout: 0.5
2. Batch normalization
3. Stateless neurons

Optimizer: Nadam w/ Gradient clipping

Timesteps: 17, **Batch**: 512, **Epochs**: 20

Two approaches:

1. Few Shots Learning
2. Data augmentation

	L	MSE	f-beta
MaLSTM	**1.535**	0.266	0.173
Competition	6.375	0.253	0.040
Time Series	0.429	0.221	0.516

6. Deep Learning Modeling

Deep Ensembles

Input:

5 FFN pre-trained models

Stack model:

Random Forest Classifier

	L	MSE	f-beta
Ensemble	**1.13**	0.514	0.454
Competition	9.07	0.461	0.050
Time Series	0.86	0.423	0.487

Conclusions

The **dataset was a mess**: raw data, variable timestep, unbalanced classes, lots of missing values, useless features, etc.

We explored **many different approaches**: imputation techniques, padding sequences, oversampling, feature selection, feature engineering, etc.

We used **many DL models and configurations**: FeedForward NNs, Convolutional NNs, Recurrent NNs, Autoencoders, Deep Ensembles, and Siamese NNs.

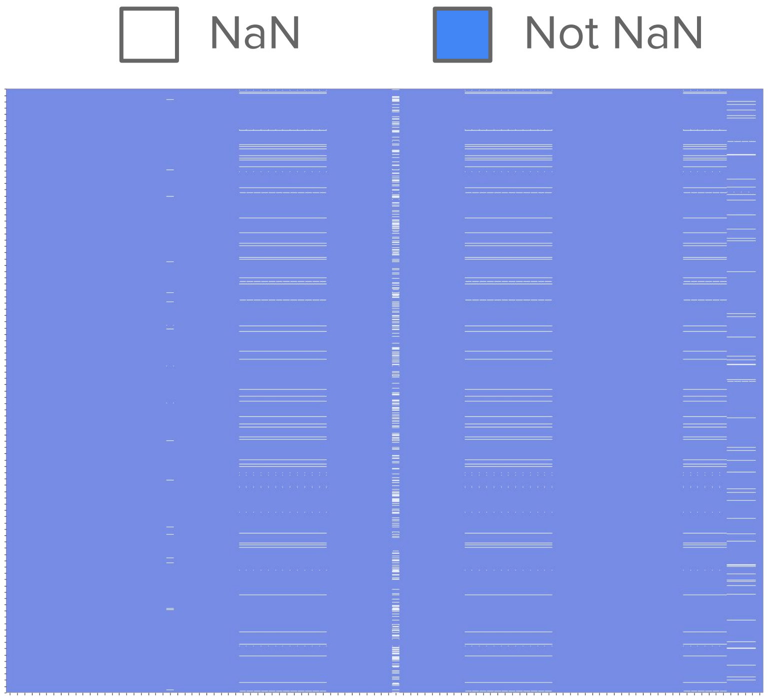
Although we beat the two baselines with RNN and CNN models and ranked among the best in the competition, we think that **the evaluation metric is flawed and might not produce the best models**.

Thanks!

EXTRA

2. Data Wrangling

Missing Values & Imputation



Missing Values Distribution

Simplified Dataset Sample

Event	Time to collision	Risk
Event1 - t1	1.5	-30
Event2 - t1	5.2	-25
Event2 - t2	1.1	-6
Event3 - t1	7	-1