
NASA APPS
CODEX TEAM
CODEX
TEAM

CodeX program

Soler, Karol; Arevalo, Anderson; Ortiz, Numa

Seismic Detection Across the Solar System, Gimnasio Campestre, Bogota, Colombia

October 6, 2024

Challenge: Planetary seismology missions struggle with the power requirements necessary to send continuous seismic data back to Earth. But only a fraction of this data is scientifically useful! Instead of sending back all the data collected, what if we could program a lander to distinguish signals from noise, and send back only the data we care about? Your challenge is to write a computer program to analyze real data from the Apollo missions and the Mars InSight Lander to identify seismic quakes within the noise!

1. Codex program

Codex program is a supervised machine learning model using the K-Nearest Neighbors(KNN) algorithm to predict earthquakes in mars.

2. Metodology

2.1. Obtein the data

As part of the NASA challenge, we were provided with a dataframe consisting of folders for both the Moon and Mars. Each folder contained testing and training data, as well as a catalog of the most significant data. It was with this information that we began our work.

2.2. Preparation of the programming environment

We used Google Colab as testing platform due to its ease of sharing, after that, the extraction of the dataframe was done in ZIP format directly in Colab.

2.3. Data in csv

Several data organization measures were implemented, starting with the search for all CSV files in a specific directory and counting how many there are, storing the list of those files in a variable. This process is a common initial step in data preparation before loading and processing the files for further analysis.

2.4. Variables and Filter

The code is to prepare and filter seismic data from Mars for analysis, ensuring that only relevant and high-quality data are used for modeling and interpretation. Additionally, it includes the incorporation of new variables such as acceleration and frequency, which will be useful for subsequent analyses

2.5. Ideation process

The ideation process began, during which the proposal was made to train a model to improve precision and ensure continuous use. To achieve this, artificial intelligence was chosen. However, it was crucial to select the appropriate models. Neural networks and decision trees were discarded, and ultimately, K-Nearest Neighbors (KNN) was selected.

3. Variables

The acceleration was calculated with the goal of improving the model's precision by incorporating additional variables. This inclusion not only enhances the accuracy of the predictions but also enables a more comprehensive analysis of the seismic data. By understanding how the velocity changes over time, we can identify trends and patterns that may not be apparent from velocity data alone. This deeper insight allows for more effective modeling of seismic events, ultimately contributing to more reliable detection and interpretation of seismic activities on Mars. Additionally, the derived variables can help in refining the criteria for filtering out noise and irrelevant data, further enhancing the overall quality of the analysis.

$$a = \frac{\Delta v}{\Delta t} = \frac{v(t) - v(t-1)}{t - (t-1)}$$

```
df['acceleration'] = df['velocity(m/s)']
.diff() / df['time_rel(sec)'].diff()
 df['acceleration'] = df['acceleration']
 .fillna(0)
 \end{equation}
```

Incorporating frequency as a variable allows for a deeper understanding of the seismic events, as it reveals how often certain patterns or signals occur over time. This additional information helps in identifying significant changes in the data that may correlate with seismic activity.

$$f = \frac{|v(t)|}{\Delta t} = \frac{|v(t)|}{t - (t-1)}$$

```
# Calculate frequency
tr_times_filt = np.array(df
['time_rel(sec)'])
tr_data_filt = np.array(df
['velocity(m/s)'])
freq = [0]
for i in range(1, len(df['velocity
(m/s)'])):
    freq.append(abs(tr_data_filt[i])
    / (tr_times_filt[i] -
    tr_times_filt
    [i - 1]))
df['freq'] = freq
```

The differences in velocity and frequency were calculated to further enhance the model's precision and analytical capabilities. By examining how both velocity and frequency change over time, we gain critical insights into the dynamics of seismic events.

Calculating the difference in velocity allows us to understand how quickly the seismic signals are changing, which can indicate the intensity of an event. Similarly, the difference in frequency reveals shifts in the seismic signal patterns, helping to identify variations that may signify important geological activity.

These derived differences provide valuable context for interpreting seismic data. They enable us to detect significant fluctuations that may correspond to events of interest, such as tremors or shifts in geological formations. By integrating these differences into the model, we enhance its sensitivity to subtle yet critical changes, leading to more accurate predictions and a deeper understanding of the underlying processes affecting seismic activity on Mars. This comprehensive approach ultimately contributes to more reliable detection and interpretation of seismic events.

$$\Delta f = f(t) - f(t-1)$$

$$\Delta v = v(t) - v(t-1)$$

4. Model

First steps was labeling the catalog values. A default label of 0 was assigned to all data points, indicating the absence of seismic events. Then, values associated with seismic events were labeled with a 1, creating a clear distinction between seismic and non-seismic events and setting up the dataset for effective model training. This is achieved in the following lines:

```
df['etiquetas'] = 0
if file_key in fechas_dict.keys():
    fechas_validas = np.array
    (fechas_dict[file_key])
    .astype(int)
    df['etiquetas'] = df
    ['time_rel(sec)'].isin
    (fechas_validas).astype(int)
```

The code begins by defining the filter function, designed to identify and select relevant data points for seismic events in a DataFrame df. First, it calculates the mean and standard deviation of the velocity:

```
media = np.mean(df['velocity(m/s)'])
desviacion_estandar = np.std
(df['velocity(m/s)'])
```

Initially, all records are labeled as 0, and for records corresponding to a specific file within the dictionary fechas dict, labels are updated to 1 if the file contains a date matching a seismic event

```
file_key = filename[:-4]
if file_key in
```

```
fechas_dict.keys():
    fechas_validas = np.array
    (fechas_dict[file_key])
    .astype(int)
    df['etiquetas'] =
    df['time_rel(sec)']
    .isin(fechas_validas)
    .astype(int)
```

Non-seismic data (df_no_sismos) is further filtered to include only velocity values exceeding the mean plus one standard deviation:

```
df_no_sismos =
df[df['etiquetas'] == 0]
df_no_sismos = df_no_sismos
[df_no_sismos
['velocity(m/s)'] >
(media + desviacion_estandar)]
```

The mean and standard deviation of this filtered data are then recalculated, applying an additional threshold at the 98th percentile to capture only the highest velocity data in the absence of seismic events:

```
media_filtrada = np.mean
(df_no_sismos['velocity(m/s)'])
desviacion_estandar_filtrada = np.std
(df_no_sismos['velocity(m/s)'])
df_no_sismos = df_no_sismos
[df_no_sismos['velocity(m/s)']
> (media_filtrada +
desviacion_estandar_filtrada)]
umbral_percentil_98 = np.percentile
(df_no_sismos['velocity(m/s)'], 98)
df_no_sismos = df_no_sismos
[df_no_sismos
['velocity(m/s)'] > umbral_percentil_98]
```

Finally, seismic data labeled with a 1 is concatenated with the filtered non-seismic data, resulting in a clean, well-prepared DataFrame suitable for model training:

```
df_final = pd.concat([df[df['etiquetas']
== 1], df_no_sismos])
```

Finally, the code processes each CSV file in the LunarCatalogo folder, iterating through them and applying the DataTrain and filter functions to each file. This process extracts only the most relevant data according to the specified filters. The filtered DataFrames are stored in a list (DataframesLunar) and concatenated at the end to produce a single, consolidated DataFrame with all pertinent information, ready for analysis:

```
for filename in tqdm(csv_files):
    file_path =
    os.path.join(LunarCatalogo, filename)
    df =
    DataTrain(file_path)
    df_final =
    filtrar(df,fechas_dict)
    DataframesLunar.append(df_final)
df =
pd.concat(DataframesLunar, ignore_index=True)
```

The model was trained, and its metrics and accuracy are displayed in the results section. Finally, when a CSV file is input, the model accurately detects the seismic events present in the DataFrame.

5. Results

5.1. Variables and Distributions

Several graphs were made to observe the frequency with which the variables occurred. The following distribution plots were obtained:

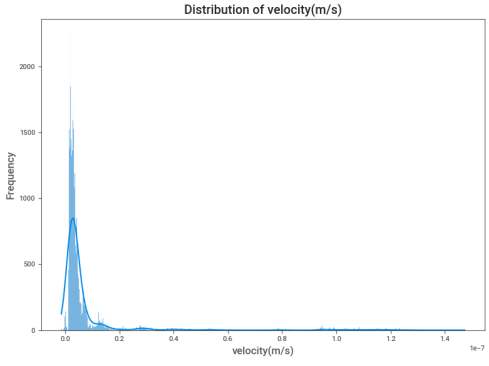

Figure 1: Distribution graphic of velocity (m/s)

Also, several intercorrelations were also carried out between the variables. Where it was observed how frequency, speed, acceleration and time interacted with each other. The results obtained were the following:

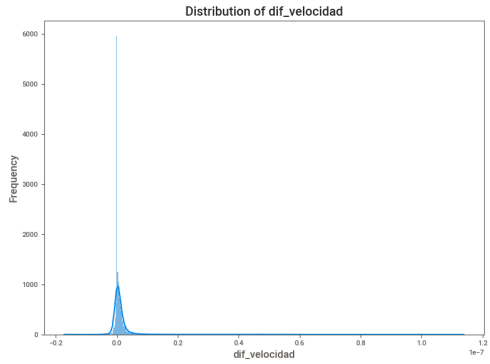

Figure 2: Distribution graphic of the velocity differences between two points.

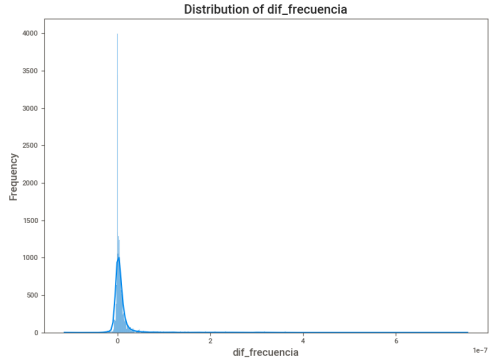

Figure 3: Distribution graphic of the frequency differences between two points.

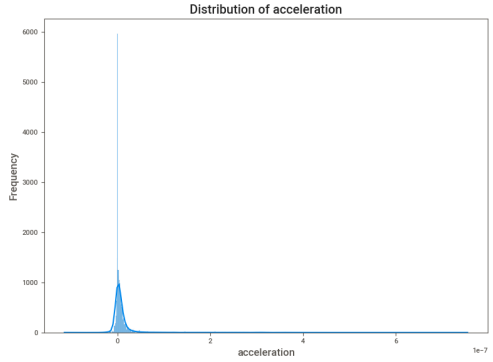

Figure 4: Distribution graphic of acceleration (m/s^2).

5.2. Report Model

It was necessary observe the learning process of the model, the indicators to know its certainty in relation to the data to know in advance how accurate the predictions will be. The following results were obtained by applying the algorithms shown previously.

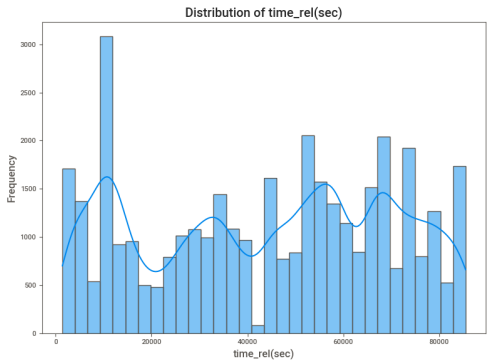

Figure 5: Distribution graphic of relative time (s)

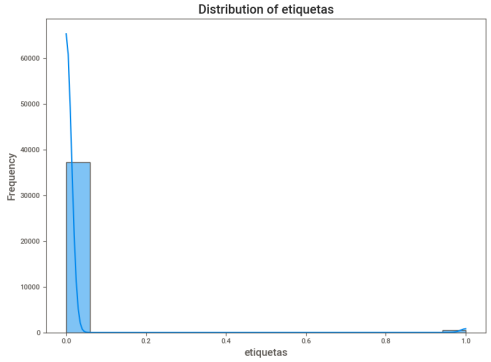

Figure 6: Distribution graphic of 0 or 1 tags

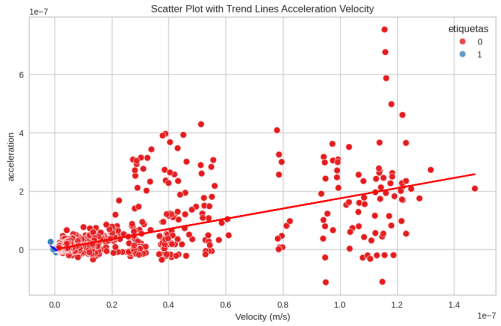

Figure 7: This plot shows the relationship between velocity and acceleration.

Shows a curve that represents the balance between precision and recall. Shaded area represents variance/uncertainty. It reaches its maximum near the intersection point of precision and recall.

5.3. Predictions

Once the model was completed, predictions of earthquakes on Mars were made. The results provided by the model were the following:

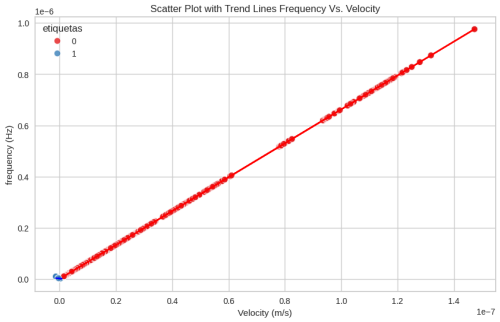

Figure 8: This plot shows the relationship between velocity and frequency.

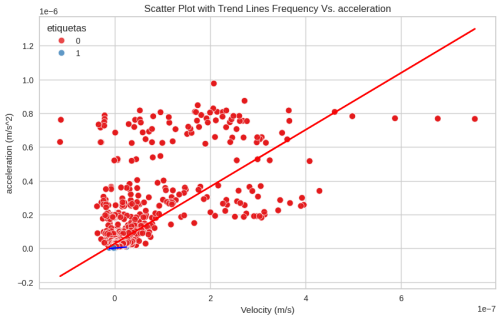

Figure 9: This plot shows more scatter compared to the frequency vs velocity.

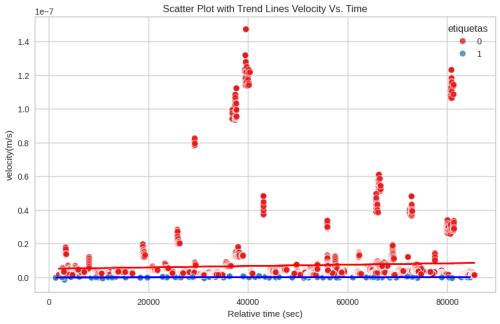

Figure 10: This plot shows more scatter compared to the velocity vs time.

6. Discussion

The velocity distribution (graph 1) is highly skewed towards lower values. There's a sharp peak near 0 m/s, indicating that most measurements show very low velocity. Also, the graph of $dif\_velocidad$ (graph 2) shown a roughly symmetrical but with a sharp peak distribution, indicating that most velocity changes are very small. It's roughly symmetrical but with a sharp peak, indicating that most velocity changes are very small. Similar to $dif\_velocidad$ the

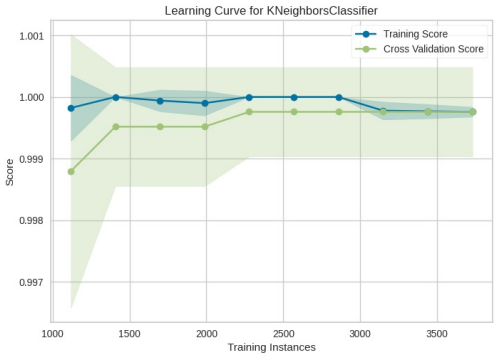

Figure 11: Graph with model's performance changes as the number of training instances increases.

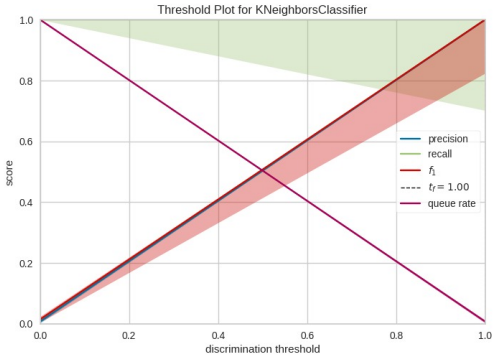

Figure 12: Graphic precision, recall, and F1 score change with different discrimination thresholds.

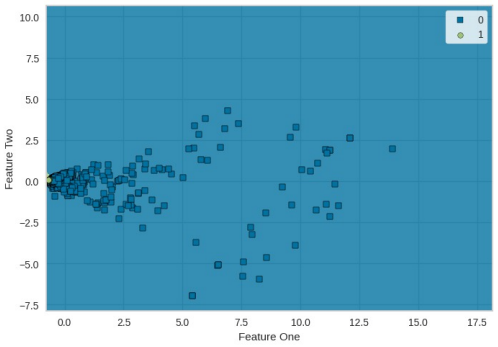

Figure 13: Distribution of data points across two features (Decision Boundary).

distribution graphs of dif_freq(graph 3) and `acceleration`(graph4), are centered around 0 with a sharp peak. Velocity, acceleration, and their changes ($dif\_velocidad$, $dif\_frequency$) all show similar patterns with sharp peaks near zero and long tails. This suggests that seismic events, which are likely represented in the tails, are rare but potentially distinguishable from the back-

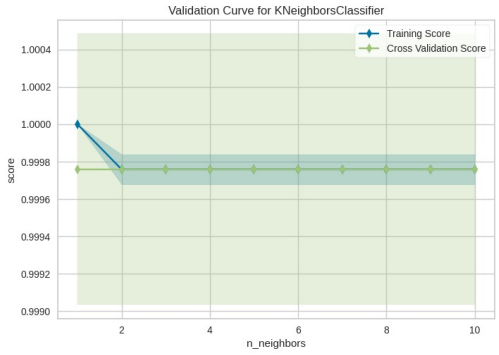

Figure 14: Graph with the model's performance changing with different numbers of neighbors (k).

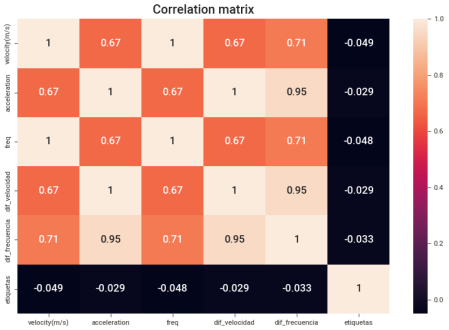

Figure 15: Heatmap with correlations between different features.

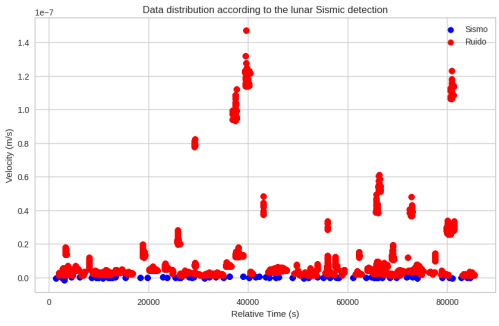

Figure 16: This plot shows velocity over time, distinguishing between seismic events (blue) and noise (red)

ground noise.

For the relative time distribution (graph 5) the distribution is somewhat uniform across the entire time range, with some fluctuations. There are several peaks, notably around 10,000 seconds, 50,000 seconds, and 70,000 seconds. The multiple peaks could represent different periods of seismic activity or changes in background noise

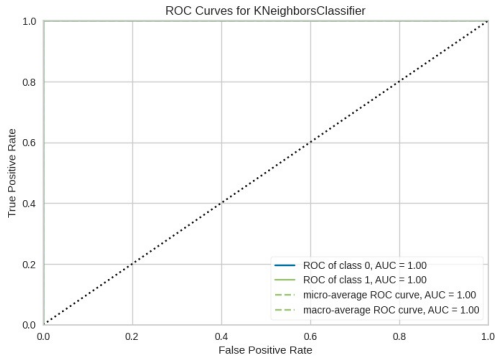

Figure 17: Caption

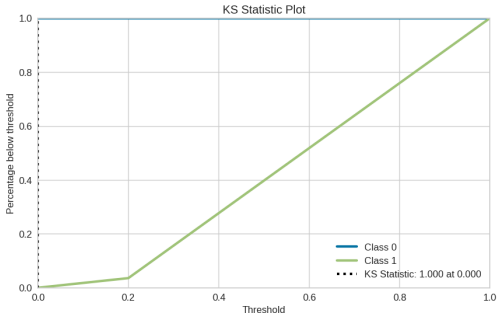

Figure 18: Kolmogorov-Smirnov (KS) statistic

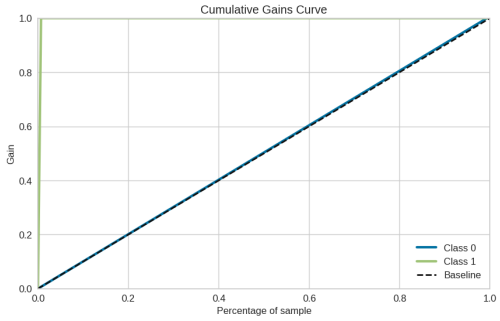

Figure 19: Cumulative Gains Curve.

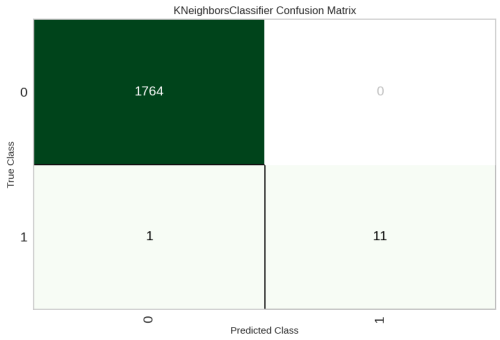

Figure 20: Confusion Matrix.

throughout the day. The relatively uniform distribution suggests continuous data collection with-

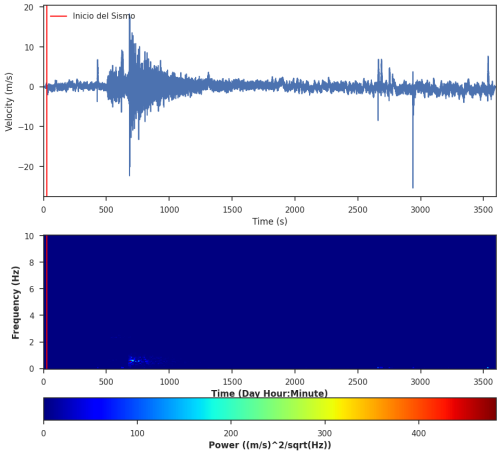

Figure 21: Seismogram and spectrogram for the prediction of a seismic event on Mars.

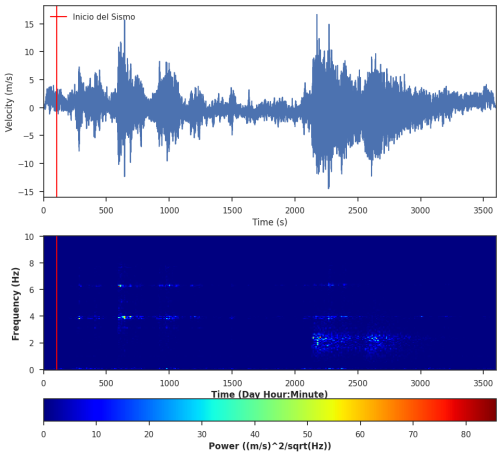

Figure 22: Seismogram and spectrogram for the prediction of a seismic event on Mars.

out significant gaps. Finaly, for tags distribution (graph 6) there's a very small number of data points for class 1 (likely seismic events), visible as a tiny spike near 1.0 on the x-axis.

In the same way, the existing correlations between the variables were observed for subsequent analysis.

In the graph 7 there's a positive correlation between velocity and acceleration. The majority of data points (both seismic and noise) are clustered near the origin. Noise events (red) show higher variability and reach higher values for both velocity and acceleration. Very few seismic events (blue) are visible, reinforcing the class imbalance issue. The scatter plots reveal that seismic events generally have lower velocity and acceleration compared to some high-magnitude noise events.

In the graph 8 there's a strong linear relationship between frequency and velocity. Both seismic and noise events follow this linear trend closely. The relationship appears consistent across the entire range of values.

In the graph 9 there's still a positive correlation between frequency and acceleration, but it's less pronounced. Noise events (red) show higher variability, especially at lower frequencies. Seismic events (blue) are clustered at the lower end of both frequency and acceleration.

In the graph 10 shows the sporadic nature of high-velocity events (likely noise). Seismic events (blue) consistently have lower velocities throughout the time period. The velocity vs. time scatter plot shows that high-magnitude events (likely noise) occur sporadically, while seismic events have a more consistent, low-magnitude presence over time.

The aim was to observe the efficiency of the model, from the results obtained it is possible to observe the learning curve of the model. The graph 11 shows how the model's performance changes as the number of training instances increases. Both training and cross-validation scores are consistently high (around 0.999-1.000), indicating excellent performance. The close alignment of training and cross-validation scores suggests the model is not overfitting. The steady performance across different training sizes indicates that the model is stable and generalizes well.

The thresholds of the model were also analyzed. The graph 12 shows how precision, recall, and F1 score change with different discrimination thresholds. The precision and recall curves intersect around a threshold of 0.5, which is often a good balance. The F1 score (harmonic mean of precision and recall) peaks near this intersection point, suggesting it might be an optimal threshold for classification.

It is also possible observe the distribution data (graph 13), the plot shows points across two features. The data appears to be separable, with class 0 (likely non-seismic events) clustered near the origin and class 1 (likely seismic events) more spread out. The data points for Class 1 are widely scattered and sparse, especially compared to the dense cluster of Class 0 points. This sparsity suggests that we don't have enough samples to fully characterize the distribution of seismic events.

More data points for Class 1 would provide better statistical robustness, allowing for more reliable model training and evaluation.

The graph 14 shows how the model's performance changes with different numbers of neighbors (k). Both training and cross-validation scores are consistently high across all k values, with only a slight decrease in training score as k increases. This suggests that the model is robust to changes in this hyperparameter and performs well regardless of the chosen k value.

The heatmap in the graph 15 shows correlations between different features. There are strong positive correlations between many features (e.g., acceleration and $dft_velocidad$ at 0.95), indicating potential redundancy in the feature set. The $'etiquetas'$ feature (likely the target variable) has weak negative correlations with other features, which is interesting and might warrant further investigation.

In the graph 16 there's a clear class imbalance, with many more noise data points than seismic events. Seismic events generally have lower velocity amplitudes compared to noise. Several noise spikes are visible, particularly around 40,000 and 80,000 seconds.

The ROC Curves for KNeighborsClassifier (graph 17) appears to be performing exceptionally well, achieving perfect separation between classes. It could indicate a highly effective model and feature set. In the other hand, it might suggest potential data leakage or overfitting. There could be a feature that perfectly separates the classes, which might not generalize to new data.

For the KS Statistic plot (graph 18) shows the maximun distance between the cumulatica distribution functions of two classes. The blue line (Class 0) is perfectly aligned with the top of the plot, while the green line (Class 1) follows the diagonal. The KS statistic value is 1.000 at 0.000, indicating a perfect separation between the two classes at the lowest threshold.This suggests that the model can distinguish between the two classes (likely seismic and non-seismic events) with high accuracy.

The Cumulative Gains Curve (graph 19) shows the model's performance in identifying positive cases (likely seismic events) as the classification threshold varies.The green line (Class 1) rises very steeply at the beginning, indicating that the model identifies most positive cases with high confidence.The blue line (Class 0) closely follows the baseline, suggesting good performance for negative cases as well. The large area between the green line and the baseline indicates a highly effective model.

The Confusion Matrix (graph 20) shows the performance of the *KNN classifier* in predicting seismic events.

- True Negatives (top-left): 1764 cases correctly identified as non-seismic events.
- False Positives (top-right): 0 cases incorrectly identified as seismic events.
- False Negatives (bottom-left): 1 case incorrectly identified as a non-seismic event.
- True Positives (bottom-right): 11 cases correctly identified as seismic events.

The results indicate that the KNN model performs exceptionally well in distinguishing between seismic and non-seismic events on Mars. The model shows high accuracy, with only one misclassification out of 1776 cases. The perfect separation in the KS plot and the steep curve in the Cumulative Gains plot further support the model's effectiveness. However, the small number of positive cases (12 seismic events) suggests that more data might be beneficial for improving the model's robustness, especially for detecting rare seismic events.

Finally, the images obtained by the predictions (21 and 22) show seismograms and spectrograms for two different seismic events on Mars. Both seismograms show clear spikes in amplitude after the earthquake start, indicating seismic activity. The second prediction shows more prolonged and intense activity compared to the first one. The spectograms in those images show the frequency content of the signal over time. The color scale represents power, with warmer colors indicating higher power. The second prediction also shows more diverse frequency content and higher power across various frequencies compared to first prediction.