SAND2019-8558PE

Earthquake Forecasting On Lab Scale Induced Seismic Events

Daniel A. Lizama Molina

University of Puerto Rico- Mayaguez

Rachel Willis

University of North Carolina, Wilmington

Dr. Hongkyu Yoon – Org. 8864

2019 Summer Student Symposium

This work was supported by the Laboratory Directed Research and Development program at Sandia National Laboratories.

Personal Background

- **Education**
  BS. Mechanical Engineering- UPRM (2010-2015)
  MS.-PhD. Mechanical Engineering- UPRM (2015-Present)
- **Experience**
  MITRE Corp. (2014)
  Oak Ridge National Labs (2016)
  Sandia National Labs (2017)(2018)(2019-Present)
  UPRM(Present)

Outline

1. Problem Statements

2. Data Source and Methodology

3. Methodology Sections Overview
    - Preprocessing
    - Feature Extraction
    - Regression Machine Learning
    - Prediction

4. Optimization Discussion

5. Concluding Remarks

Introductory slide

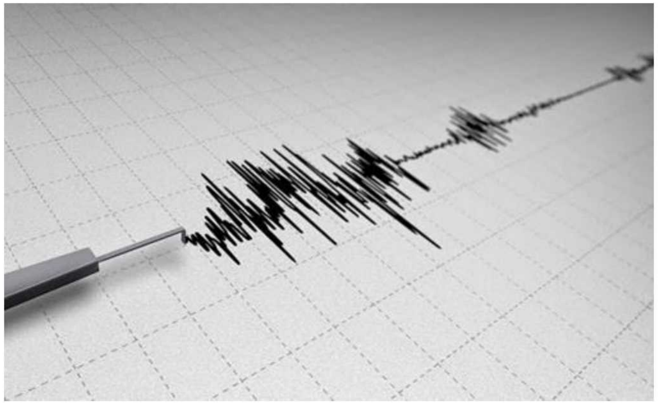
- Earthquake forecasting studies: **when**, where, and how large.

- **Kaggle: LANL Earthquake Prediction**

  LANL initial work:

  Machine Learning Predicts Laboratory Earthquakes[1]

  Use seismic signals (acoustic emissions), to predict the time remaining for the next earthquake to happen.



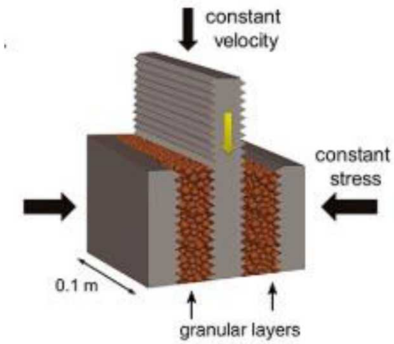
- **Research Questions:**

  What hidden signals preceding the earthquake onset that may help predict the event?
  What architecture of machine learning regressor is most suitable for this type of problem?

Data & Methodology

- Experimental data:

  Double direct shear geometry subjected to bi-axial loading.
  Aperiodic cycles of stick and slip (loading & failure).

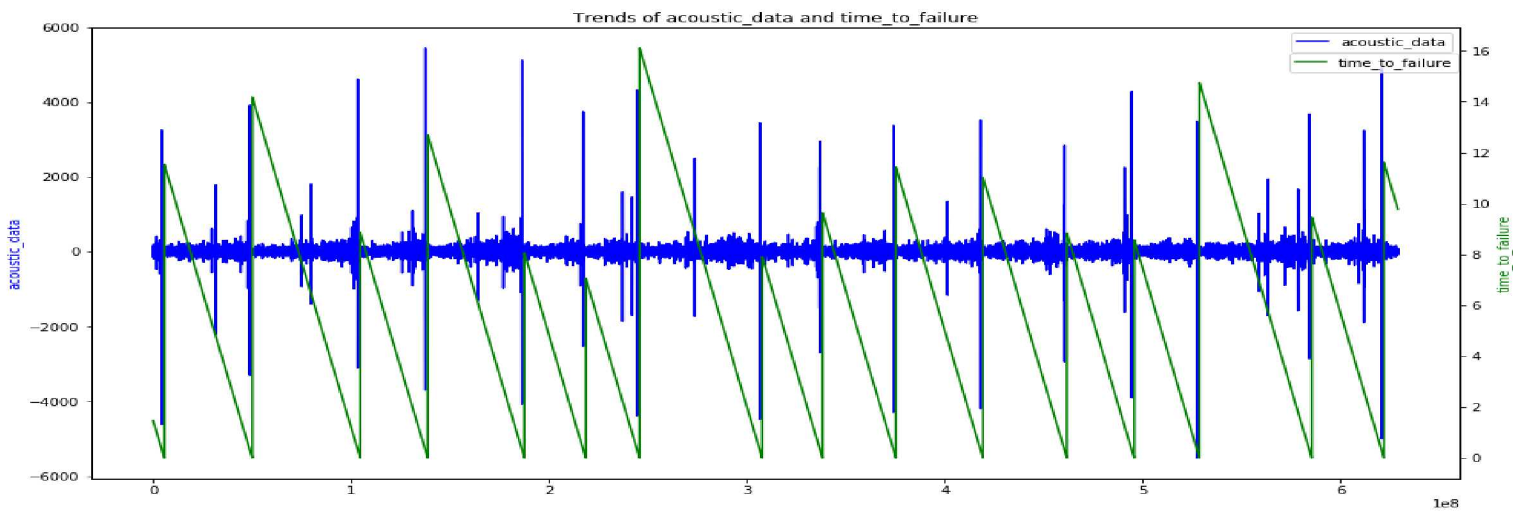


Data experimental setup [2]

- Training data: Continuous data containing 16 earthquakes.

- Testing data: Random earthquake cycle segments of 150,000 data-points

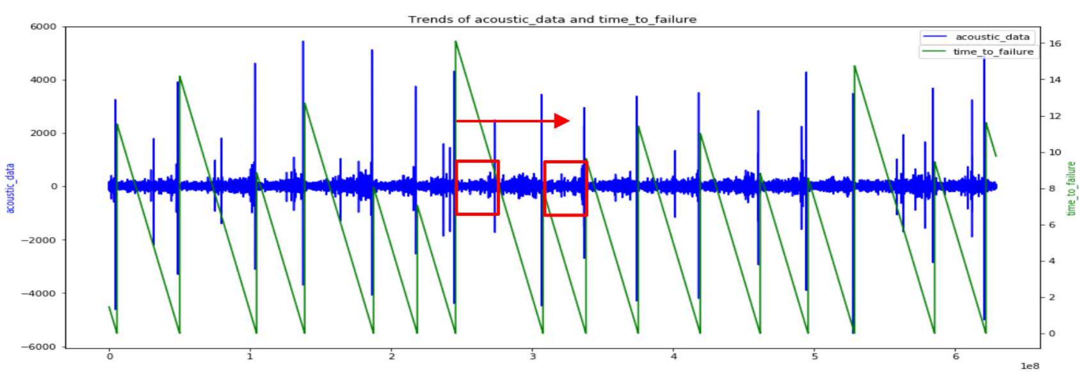
- Approach: Preprocess-> Feature Extraction-> Training->Predictions

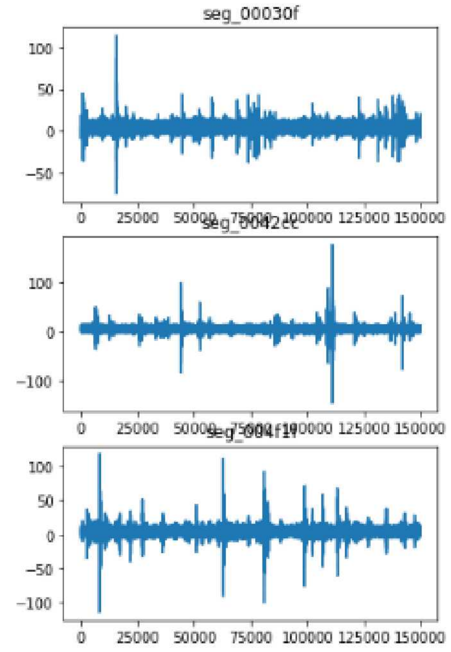


Training sample data file plotted.

Pre-process

- Data correction:
  - Add normal distribution noise
  - Subtract the median
- Digital Filtering:
  - IIR Butterworth filter
- Data segmentation:
  - Segment training data into 150,000 points windows
  - Each window is assigned a 'time to failure' (target)



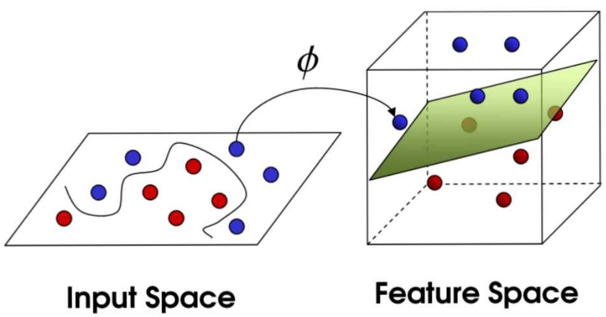


Sample test files.
https://www.kaggle.com/artgor/earthquakes-fe-more-features-and-samples

Signal Features

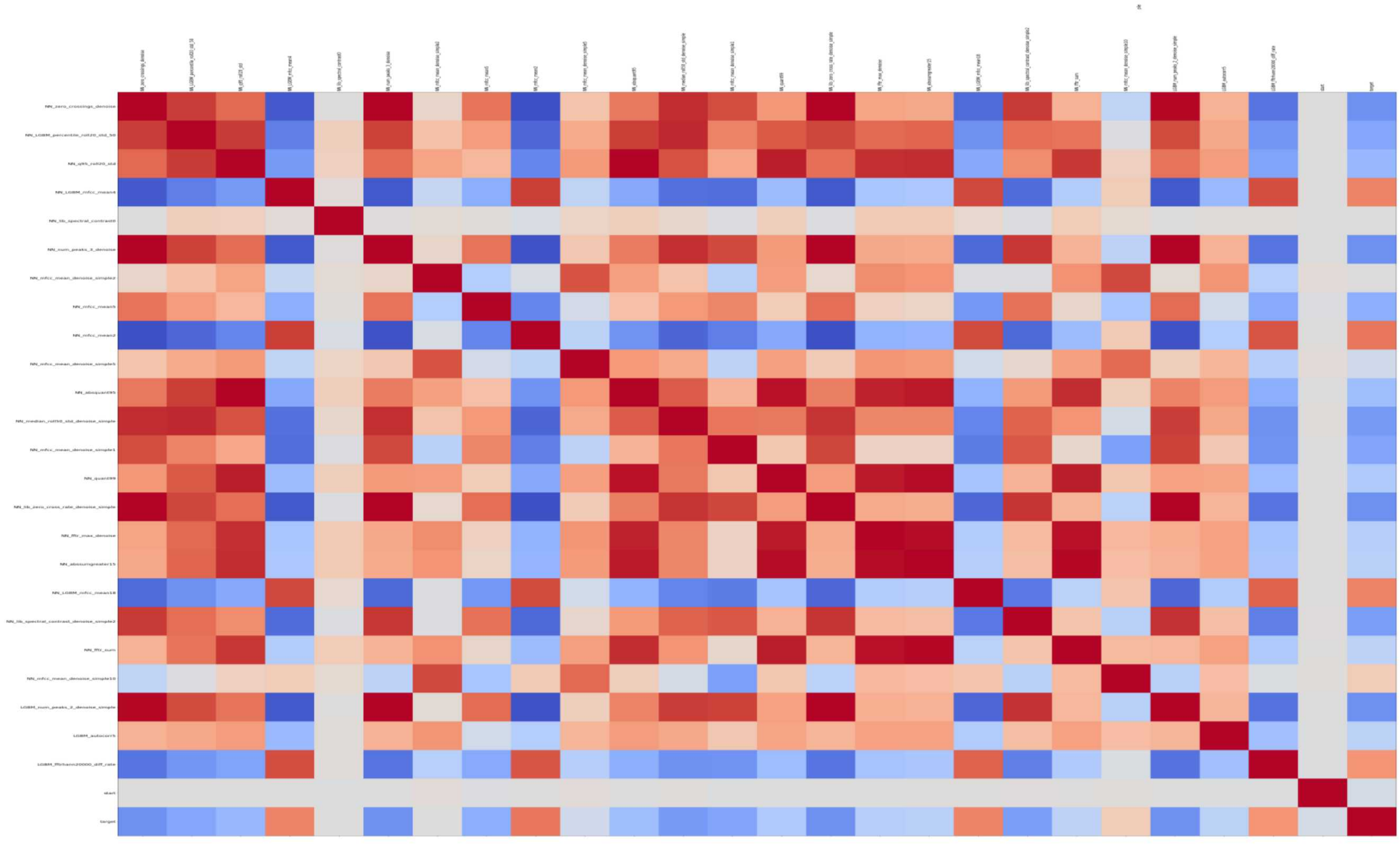
- Characterize the signal through various measurements
- Features are easily comparable to other signal's features (reduce overfitting)

$$\vec{F} = \begin{pmatrix} \text{Mean} & \text{STA/LTA} & \text{Maximum} \\ \text{Standard deviation} & \text{Correlation} & \text{Zero Crossing} \\ \text{Change rate} & \text{Kurtosis} & \text{Number of peaks} \\ \text{Percentile} & \text{Skew} & \text{Medians} \\ \text{Quantiles} & \text{Energy} & \text{Sum} \\ \text{Trend regression} & \text{Mel-frequencies} & \text{Autocorrelation} \\ \text{FFT} & \text{Minimum} & \text{Difference} \end{pmatrix}$$

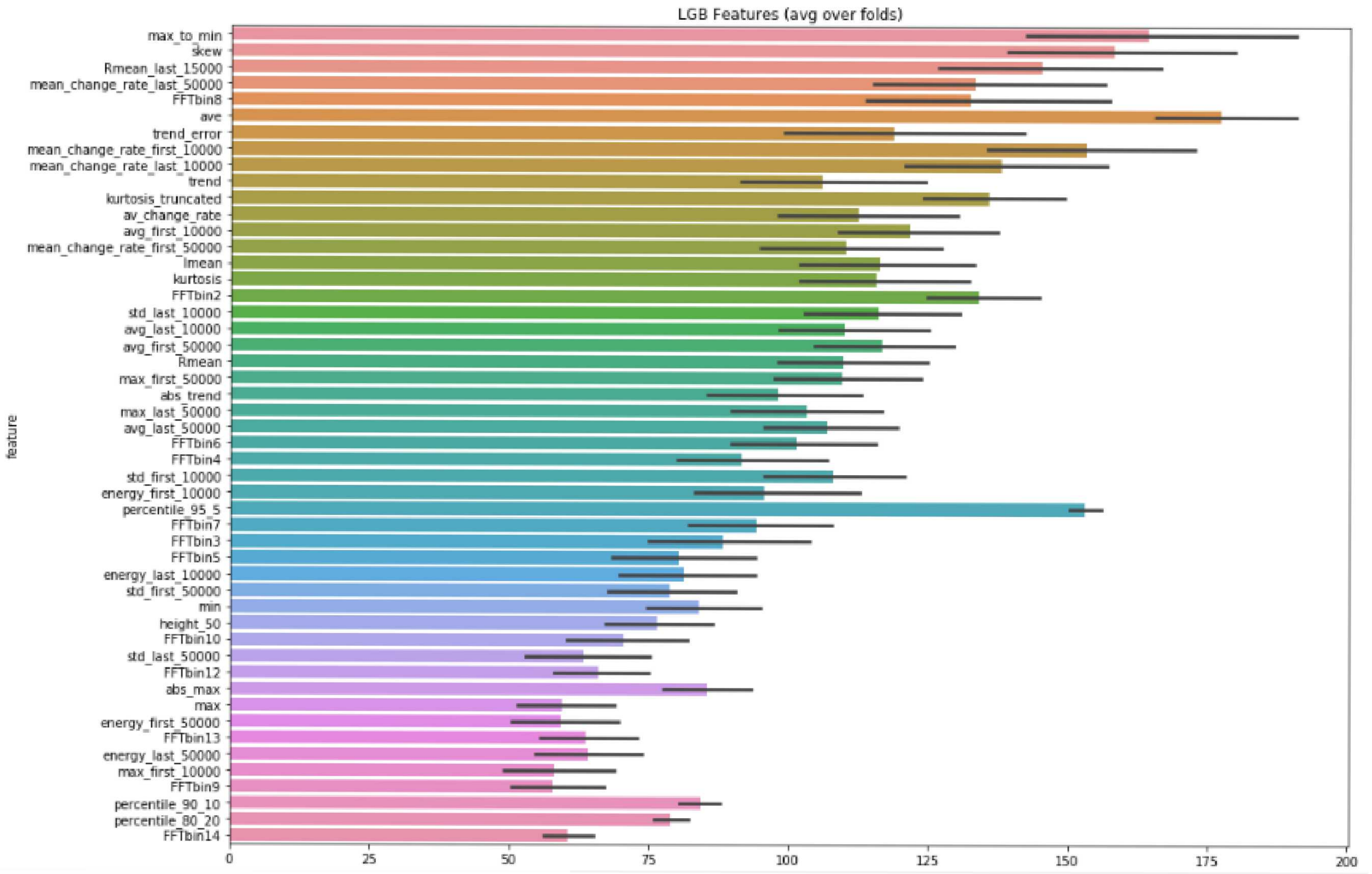


Features Correlation

Accuracy improved!



Feature Importance (LightGBM)



Forecast Machine Learning

- Data analysis method in which computers learn and autonomously build models based on data patterns.
- **Decision trees**

  Random Forest

  Boosting trees: LightGBM, XGBoost, CatBoost
- **Support Vectors:**

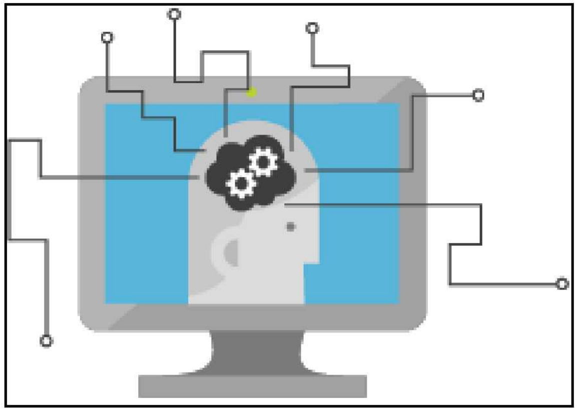
  Support Vector Regressor (SVR)

  Kernel Ridge Regression (KRR)
- **Neural Networks**

  Artificial Neural Networks (ANN)

  Short-Long Term Memmory (LSTM)

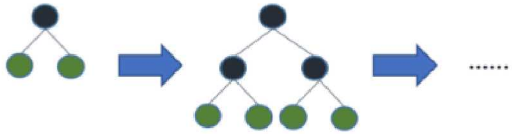
  Convolutional Neural Network (CNN)



Decision Trees: Boosting Algorithms

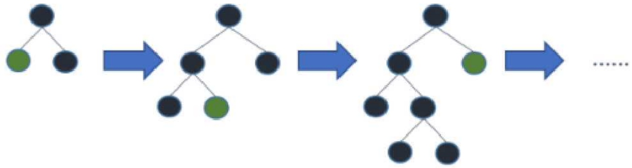
Boosting algorithms uses gradient boosting framework at its core. The default base learners are decision tree ensembles.

XGBoost grows horizontal (Level-Wise) decision trees.



Level-wise tree growth

LightGBM grows Leaf-Wise decision trees.



Leaf-wise tree growth

https://www.analyticsvidhya.com/blog/2017/06/which-algorithm-takes-the-crown-light-gbm-vs-xgboost/

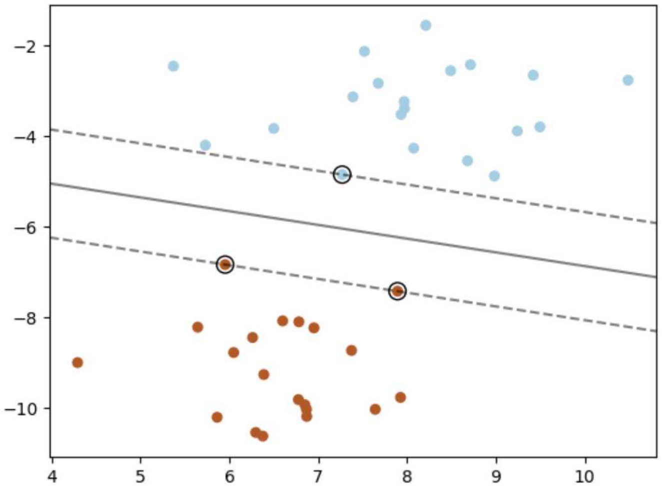
CatBoost uses a complex categorical boosting algorithm which is beyond this scope.

Support Vector Regressor (SVR) & Kernel Ridge Regression (KRR)

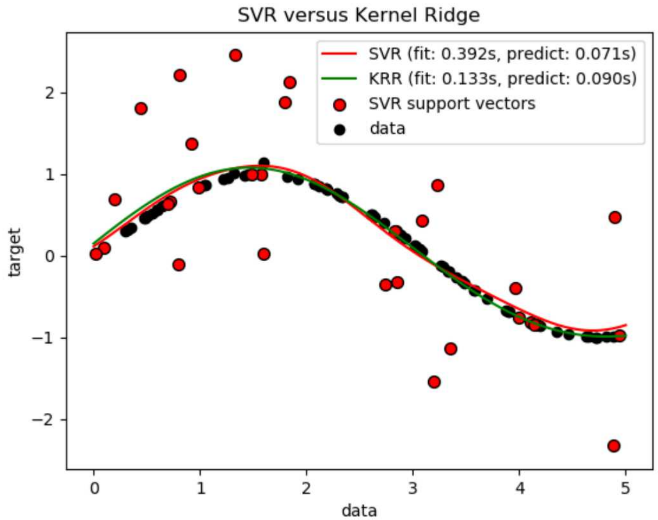
Support vector machine(SVM) algorithms can be modified for regression analysis, known as SVR.

Kernel ridge regression (KRR) [M2012] combines Ridge Regression (linear least squares with l2-norm regularization) with the kernel trick.

Different loss functions are used: KRR uses squared error loss while support vector regression uses e-insensitive loss, both combined with l2 regularization.



Support Vector machine learning. https://scikit-learn.org/stable/modules/svm.html#svm-regression



KRR and SVR comparison. https://scikit-learn.org/stable/modules/kernel_ridge.html

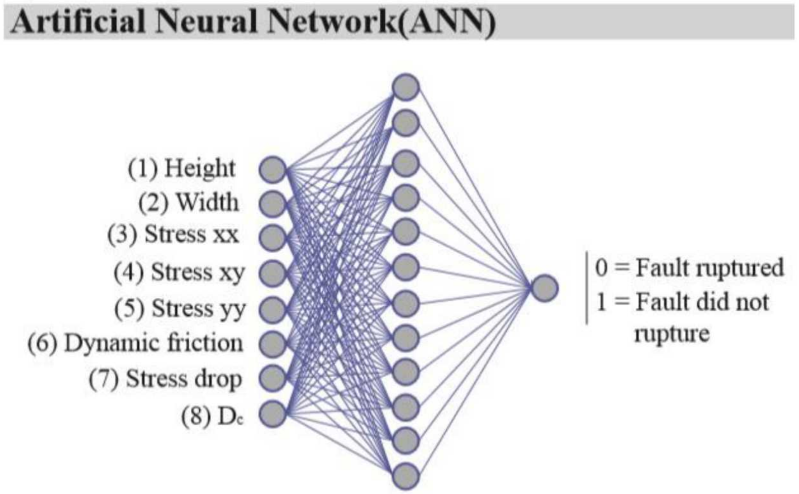
Neural Networks (NN)

Inspired in the behavior of animal's neurons communication. Each layer thresholds the information to the next node using specific activation functions.

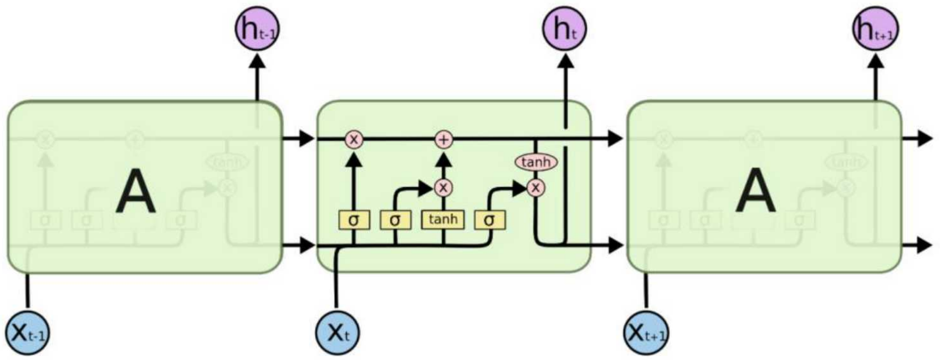
NN consists of various layers: 1 input, N hidden layers, 1 output layer

Long-short term memory(LSTM) nodes: Cell state, forget gate, input gate, tanh layer

Convolutional Neural Network (CNN): not covered here



Example Neural Network. arXiv:1906.06250 [physics.geo-ph]



LSTM nodes model. https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Prediction

Output corresponds to estimated predictions of time to failure for each of the testing data

Accuracy of the global predictions is given by mean average error (MAE)

Lower score is better

$$\mathrm{MAE} = \frac{1}{n}\sum_{j=1}^{n}|y_j - \hat{y}_j|$$

Most models received and MAE score between ~2.3 (winner) and ~2.7

	time_to_failure
seg_id	
seg_00030f	3.421094
seg_0012b5	5.758909
seg_00184e	7.369895
seg_003339	9.946596
seg_0042cc	7.786591

Example submission file contents.

Submission Evaluation

*Blend= averaged prediction

1) All features (64)

Submissions	CV mean	STD	Public Score	Private Score
LGB	2.0746	0.0924	1.52744	2.66190
XGB	2.0988	0.0921	1.5311	2.67000
Blended	——	——	1.51931	2.65868

2)2nd place winner features

Submissions	CV mean	STD	Public Score	Private Score
LGB	2.1054	0.0955	1.58427	2.70104
XGB	2.1225	0.0932	1.56686	2.70317
Blended	——	——	1.56699	2.69513

3)1st and 2nd place features

Submissions	CV mean	STD	Public Score	Private Score
LGB	2.0543	0.1198	1.62295	2.65173
XGB	2.0715	0.1196	1.55728	2.64105
KRR	2.0906	0.1078	1.56615	2.52527
Blend KRR XGB	——	——	1.53121	2.56981

4)Top features from 1st and 2nd

Submissions	CV mean	STD	Public Score	Private Score
KRR	2.0560	0.1142	1.52040	2.52111

Discussion

- Features
  - None significant feature like min and max reduce score significantly. Should include only the top relevant.
  - More features doesn't mean better. Only keep top ones.
- Models
  - More training slightly increase prediction accuracy.
  - Using samples with the broader time range generalize better.
  - Averaging predictions results in better score most of the time.
  - Less tuning parameter regressors like SVR generalize better than the LGB.
- Results
  - Predictions of time to failure over 10 seconds long can be made with just an average error of 2.5 seconds

Conclusions

- Several signal characteristics preceding an earthquake proved useful to accurately estimate its arrival over 10 seconds prior.
- Developing such an accurate model requires extensive fine tuning in exploring characteristics and tuning the predictor. (problem specific)
- Similar approach may be applied to solve other researches of interest (material fault, avalanche, land slides, health, BCI, etc.)

References

Rouet-Leduc, B., Hulbert, C., Lubbers, N., Barros, K., Humphreys, C. J., & Johnson, P. A. (2017). Machine learning predicts laboratory earthquakes. *Geophysical Research Letters*, 44, 9276– 9282

Inc., K. (2019). Kaggle. Retrieved from LANL-Earthquake-Prediction: https://www.kaggle.com/c/LANL-Earthquake-Prediction/

CatBoost

CatBoost has the flexibility of giving indices of categorical columns so that it can be encoded as one-hot encoding using one_hot_max_size (Use one-hot encoding for all features with number of different values less than or equal to the given parameter value).

If you don't pass any anything in cat_features argument, CatBoost will treat all the columns as numerical variables.

For remaining categorical columns which have unique number of categories greater than one_hot_max_size, CatBoost uses an efficient method of encoding which is similar to mean encoding but reduces overfitting. The process goes like this—

- Permuting the set of input observations in a random order. Multiple random permutations are generated
- Converting the label value from a floating point or category to an integer
- All categorical feature values are transformed to numeric values using the following formula:

$$avg\_target = \frac{countInClass + prior}{totalCount + 1}$$

Top Solution

Add normal distribution noise.

Subtract the median

21 features for NN, and 6 features for Lightgbm

Note: tested out the features using p-values

Only used training samples from the longest 10 sequences

3 folds training repeated: 10 times for Lightgbm and 8 for NN

- Discarted bad scoring CV resultant models.
- NN consists of LSTM (128), conv(128), conv(84), conv(64), dense (64), dense(32), dense(1) and 3 loss parameters (TTF, TSF, Binary TTF)

MAE for both predictors was good, but averaged prediction score was better.