CIS8395 - Final Report
LANL EARTHQUAKE PREDICTION OF UPCOMING LABORATORY EARTHQUAKES

Submitted By
Blesson Thomas
Karen Zhang
Shekha Saxena
Shivam Namdeo

Introduction:

Earthquakes are one of the major catastrophe and their unpredictability causes even more destruction in terms of human life and financial losses. Earthquake prediction remained an unachieved objective due to several reasons. One of the reasons is the lack of technology in accurately monitoring the stress changes, pressure and temperature variations deep beneath the crust through scientific instruments, which eventually results in unavailability of comprehensive data about seismic features. The second probable cause is the gap between seismologists and computer scientist for exploring the various venues of technology to hunt this challenging task.

Forecasting earthquakes is one of the most important problems in Earth science. Current scientific studies related to earthquake forecasting focus on three key points: when the event will occur, where it will occur, and how large it will be.

In this project, we will address when the earthquake will take place. Specifically, we will predict the time remaining before laboratory earthquakes occur from real-time seismic data.

We try to achieve good prediction accuracy and the physics are ultimately shown to scale from the laboratory to the field, researchers will have the potential to improve earthquake hazard assessments that could save lives and billions of dollars in infrastructure.

Project Description

Predicting the timing and magnitude of an earthquake is a fundamental goal of geoscientists. In a laboratory setting, we show we can predict "labquakes" by applying new developments in machine learning (ML), which exploits computer programs that expand and revise themselves based on new data. What we did in this project is to use seismic signals to predict the timing of laboratory earthquakes. Here we show that by listening to the acoustic signal emitted by a laboratory fault, machine learning can predict the time remaining before it fails with great accuracy. We use ML to identify acoustic signal —much like a squeaky door— that predict when a quake will occur. The experiment closely mimics Earth timing, so the same approach may work in predicting timing, but not size, of an earthquake.

About the Data:

The goal of this project is to use seismic signals to predict the timing of laboratory earthquakes. The data comes from a well-known experimental setup used to study earthquake physics. The laboratory apparatus uses steel blocks to closely mimic the physical forces at work in a real earthquake, and also records the seismic signals and sounds that are emitted. The seismic signals come from grain fracture, rotation, and displacement, or brittle failure of adhesive grain contact junctions within the laboratory fault gouge. The acoustic_data input signal is used to predict the time remaining before the next laboratory earthquake (time_to_failure).

The primary source of our data is Kaggle:
https://www.kaggle.com/c/LANL-Earthquake-Prediction/data

Training Data:

The training data is a single sequence of signal and comes from one experiment alone. In total we have 600 million records with two variables the acoustic_data and time_to_failure.

• acoustic_data - the seismic signal

• time_to_failure - the time (in seconds) until the next laboratory earthquake

Sample training data Training Data:

acoustic_data	time_to_failure
12	1.469099983
6	1.469099982
8	1.469099981
5	1.46909998
8	1.469099979
8	1.469099978
9	1.469099977
7	1.469099976
-5	1.469099974

Before applying any machine learning models it is important to perform exploratory data analysis and visualization so that we can understand the dataset, remove outliers and clean the dataset.
In the below table we describe the Acoustic data variable, we get the mean, standard deviation,minimum and maximum value.

count 6.29145480e+08

mean 4.51946757e+00
std 1.07357072e+01
min -5.51500000e+03
25% 2.00000000e+00
50% 5.00000000e+00
75% 7.00000000e+00
max 5.44400000e+03
Name: acoustic_data, dtype: float64

The time to failure, is given in seconds with a max value of 16 seconds and minimum value close to zero (1e-5)

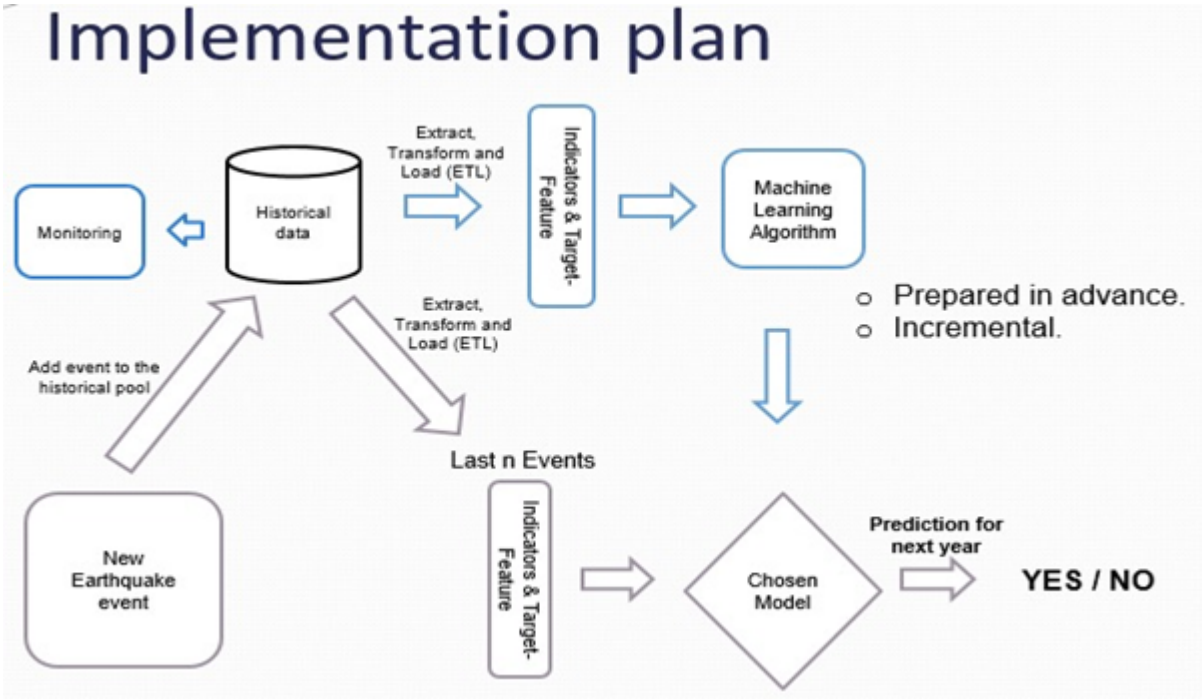
Test Data:

- The test data consists of several different sequences, called segments, that may correspond to different experiments.
- For each test data segment with its corresponding seg_id we will be predicting it's single time until the lab earthquake takes place
- seg_id - the test segment ids for which predictions should be made (one prediction per segment)

ETL Introduction

ETL is an abbreviation of Extract, Transform and Load. It is an important part of today's business intelligence (BI) processes and systems. It is the IT process from which data from disparate sources can be put in one place to programmatically analyze and discover business insights. An ETL tool extracts the data from different RDBMS source systems then transforms the data like applying calculations, concatenations, etc. and then load the data into the Data Warehouse system.

Implementation Plan



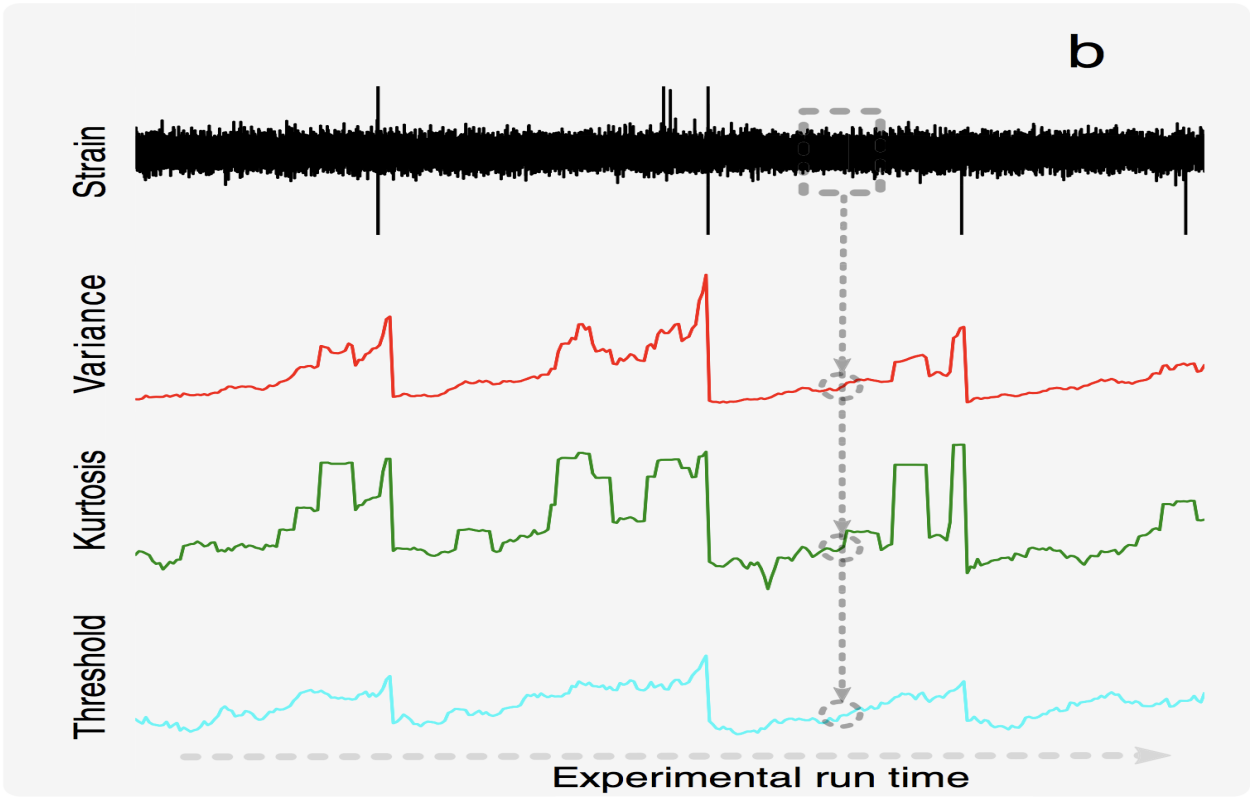
Transformation

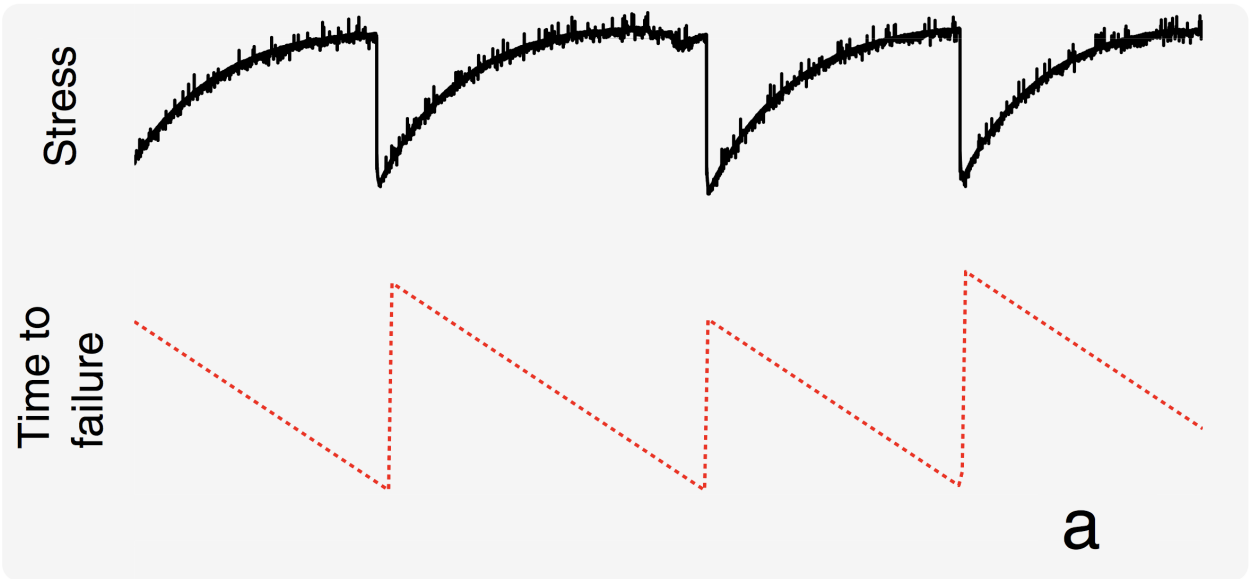
After the extraction step, we have around 600 million rows worth of data. Now our primary task is to clean, map and transform the data. We also have to remove the missing values and null values.

In our case, we are dealing a lot with the acoustic signals. We are analyzing these acoustic signals to predict the earthquakes. In each time window of the acoustic signal (also known as strain in terms of earthquake forces), we compute a set of approximately 100 potentially relevant statistical features (e.g., mean, variance, kurtosis, and autocorrelation). We use these statistical features to observe patterns, i.e. how does the mean or variance curve looks like of the acoustic signal when there is about to be a time of failure. In order to understand why we need to transform the acoustic signal (strain) and stress measures, let us understand what role do they play in an earthquake.

The data comes from a well-known experimental setup used to study earthquake physics. In total, we have 600 million records with two variables the acoustic_data and time_to_failure. The acoustic_data input signal is used to predict the time remaining before the next laboratory earthquake (time_to_failure). In the ETL step, we firstly exacted the data from Kaggle and

secondary sources from external contributor while created a changing table to track data changes and check timestamps. Based on the researches that we did on seismology, we also computed a group of potentially relevant statistical features, such as mean, variance, kurtosis, Amax, Amin, STD, autocorrelation and etc. After that, we loaded this data into our data warehouse.





Causes of Earthquakes

Within the Earth, rocks are constantly subjected to forces that tend to bend, twist, or fracture them. When rocks bend, twist or fracture they are said to deform. The strain is a change in shape, size, or volume. The forces that cause deformation are referred to as stresses. To understand the causes of earthquakes we must first explore stress and strain.

Stress and Strain

Recall that stress is a force applied over an area. Uniform stress is where the forces act equally from all directions. The pressure is uniform stress and is referred and is also called confining stress or hydrostatic stress. If stress is not equal from all directions then the stress is differential stress. There are three types of stress -

1> Tensional stress (or extensional stress), which stretches rock

2> Compressional stress, which squeezes rock and

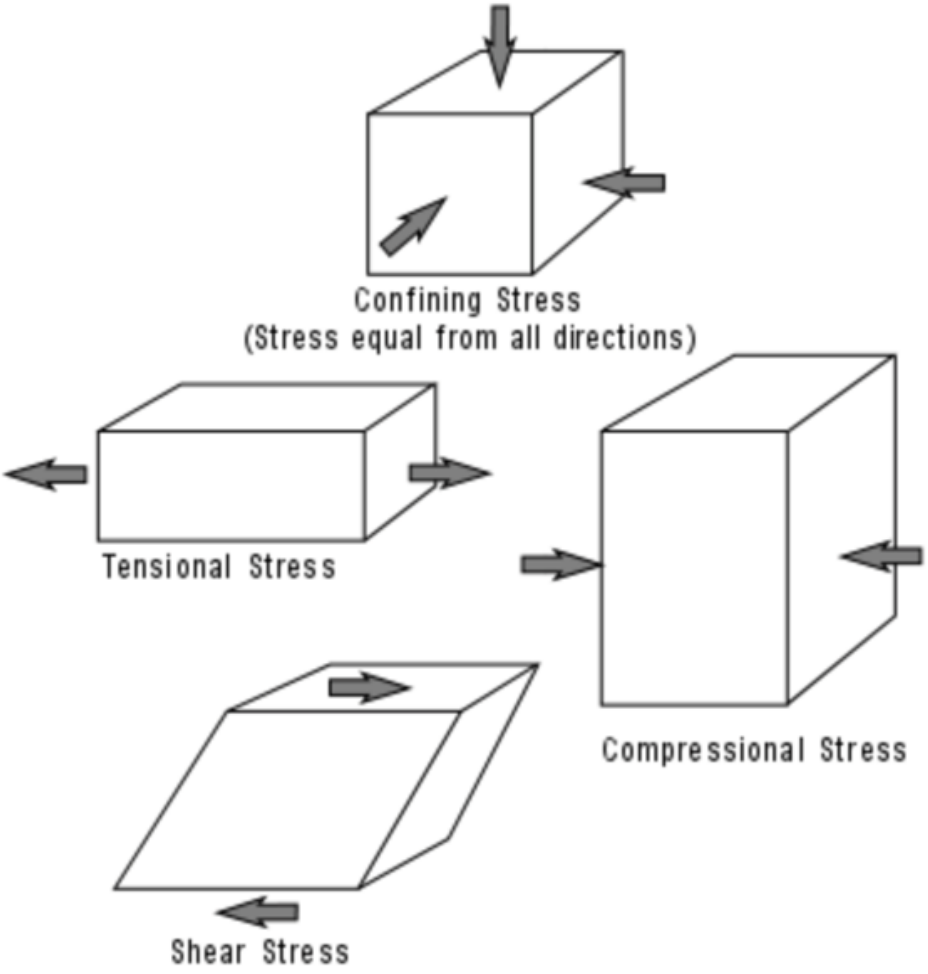
3> Shear stress, which results in slippage and translation.

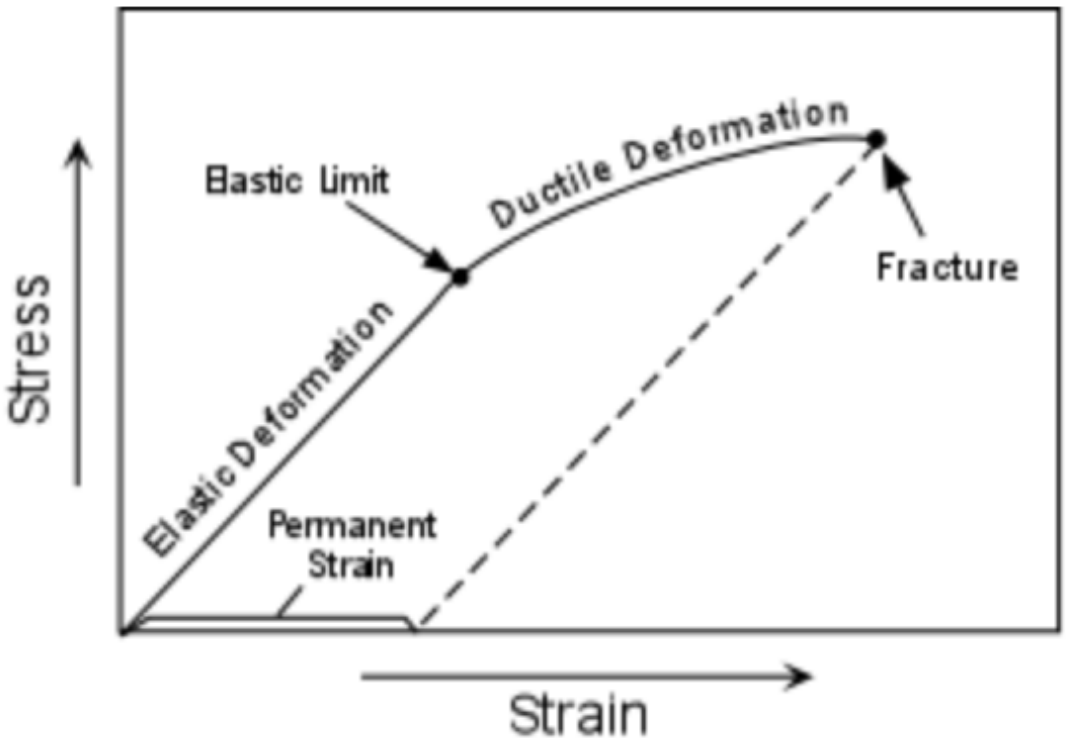
When a rock is subjected to increasing stress it changes its shape, size or volume. Such a change in shape, size or volume is referred to as a strain. When stress is applied to rock, the rock passes through 3 successive stages of deformation. There are three types of strain -

1>Elastic Deformation -- wherein the strain is reversible,

2> Ductile Deformation -- wherein the strain is irreversible and

3> Fracture -- irreversible strain wherein the material breaks.


Confining Stress
(Stress equal from all directions)
Tensional Stress
Compressional Stress
Shear Stress



Loading

Since our dataset is humungous, it was impossible for us to run any queries or perform manipulation on the data without using external sources. In our case we harnessed the power of EC2 instance on AWS and created a jupyter notebook there to work on our data and then ran machine learning models.

Our dataset was 9GB which made it difficult for us to choose a lower configuration EC2 instance of AWS. In our case we used t2.xlarge instance, providing us 4 vCPU's computational power and 16GB Memory. As shown in the image below was our estimated monthly cost, which we saved upon by following the standards mentioned below to save on AWS monthly cost.

Compute: Amazon EC2 Instances:

	Description	Instances	Usage	Type	Billing Option	Monthly Cost
⊖	earthquake	1	100 % Utilized/I	Linux on t2.xlarge	On-Demand (No Cor	$ 135.86
⊕	Add New Row					

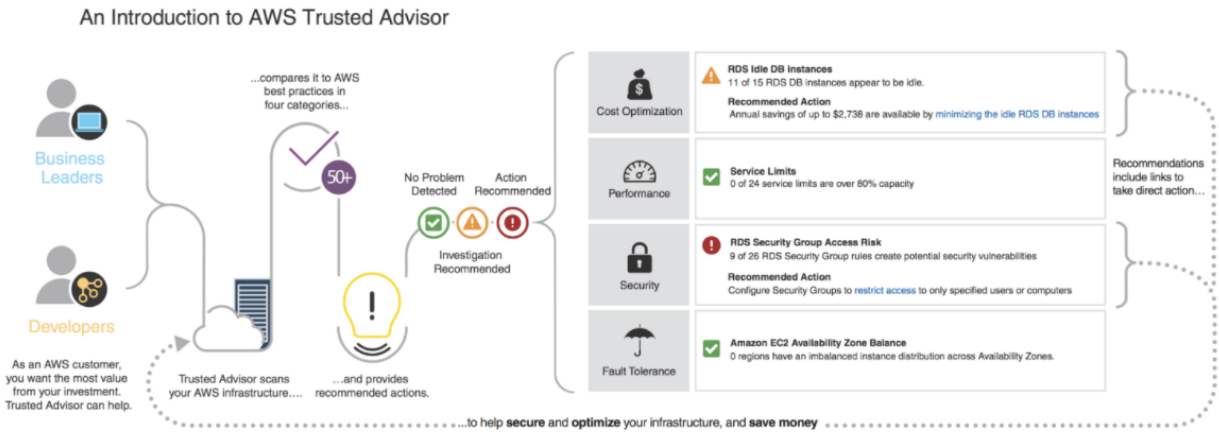
The advantage of using t2 is that it is an on demand instance and we can increase and decrease the configuration very easily depending upon the use. With On-Demand instances, you pay for compute capacity by per hour or per second depending on which instances you run. No longer-term commitments or upfront payments are needed. You can increase or decrease your compute capacity depending on the demands of your application and only pay the specified per hourly rates for the instance you use.

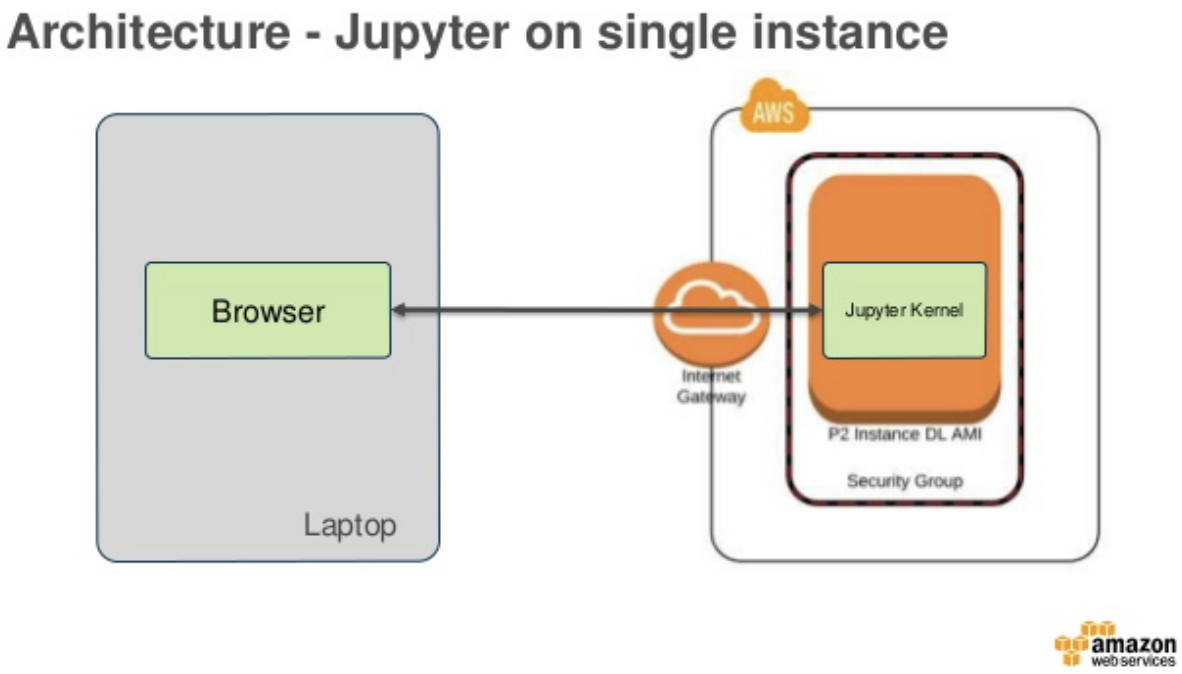
On-Demand instances are recommended for:

- Users that prefer the low cost and flexibility of Amazon EC2 without any up-front payment or long-term commitment
- Applications with short-term, spiky, or unpredictable workloads that cannot be interrupted
- Applications being developed or tested on Amazon EC2 for the first time

Steps we followed to save cost of our EC2 instances:

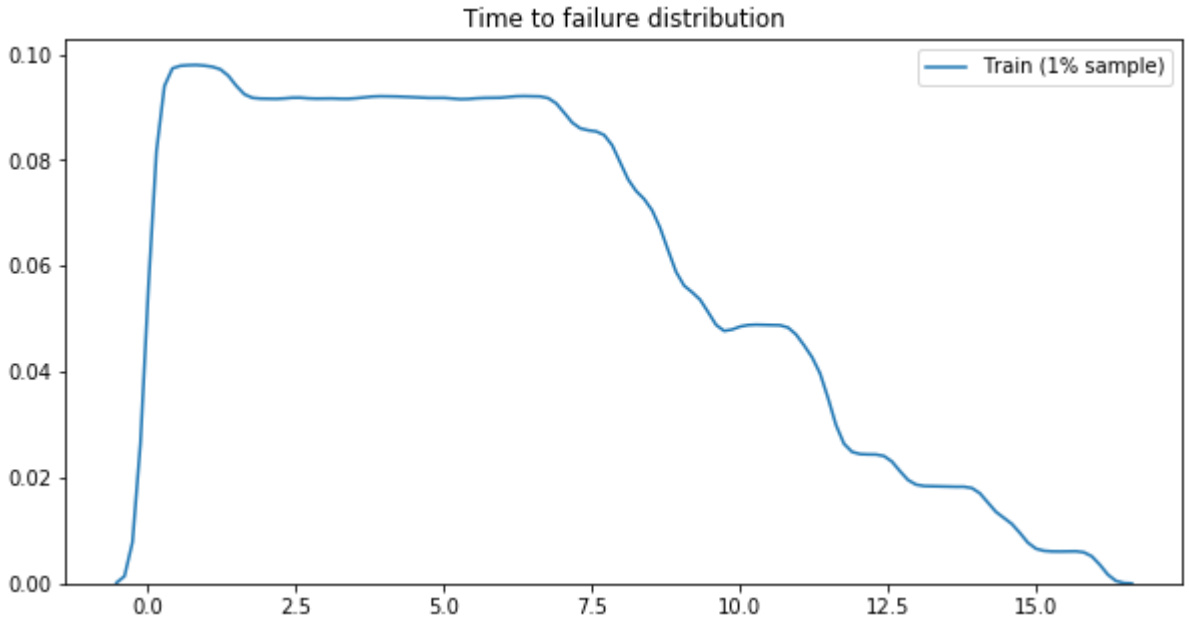
1. Shutdown unused instances
2. Select the right instance size for your workload
3. Select the appropriate S3 storage class
4. Use Cloudwatch and Trusted Advisor to monitor costs
5. Use Auto Scaling to align your resources with demand
6. Consolidated Billing provides cost savings
7. Take advantage of Reserved and Spot Instances





Data Visualization

For all the below plots we will be using a 1% random sample (~6M rows):
First we will plot a graph to determine the data distribution of acoustic signal and time to failure.



In the above graph the acoustic signal are represented by red color and time to failure in green. There are 16 earthquakes in the training data. We can see that signal peaks usually happen right before earthquakes. This is not a perfect rule though: there are some peaks far from earthquakes (e.g. between the 14º and 15º). Another interesting thing to check is the time between these high levels of seismic signal and the earthquakes.

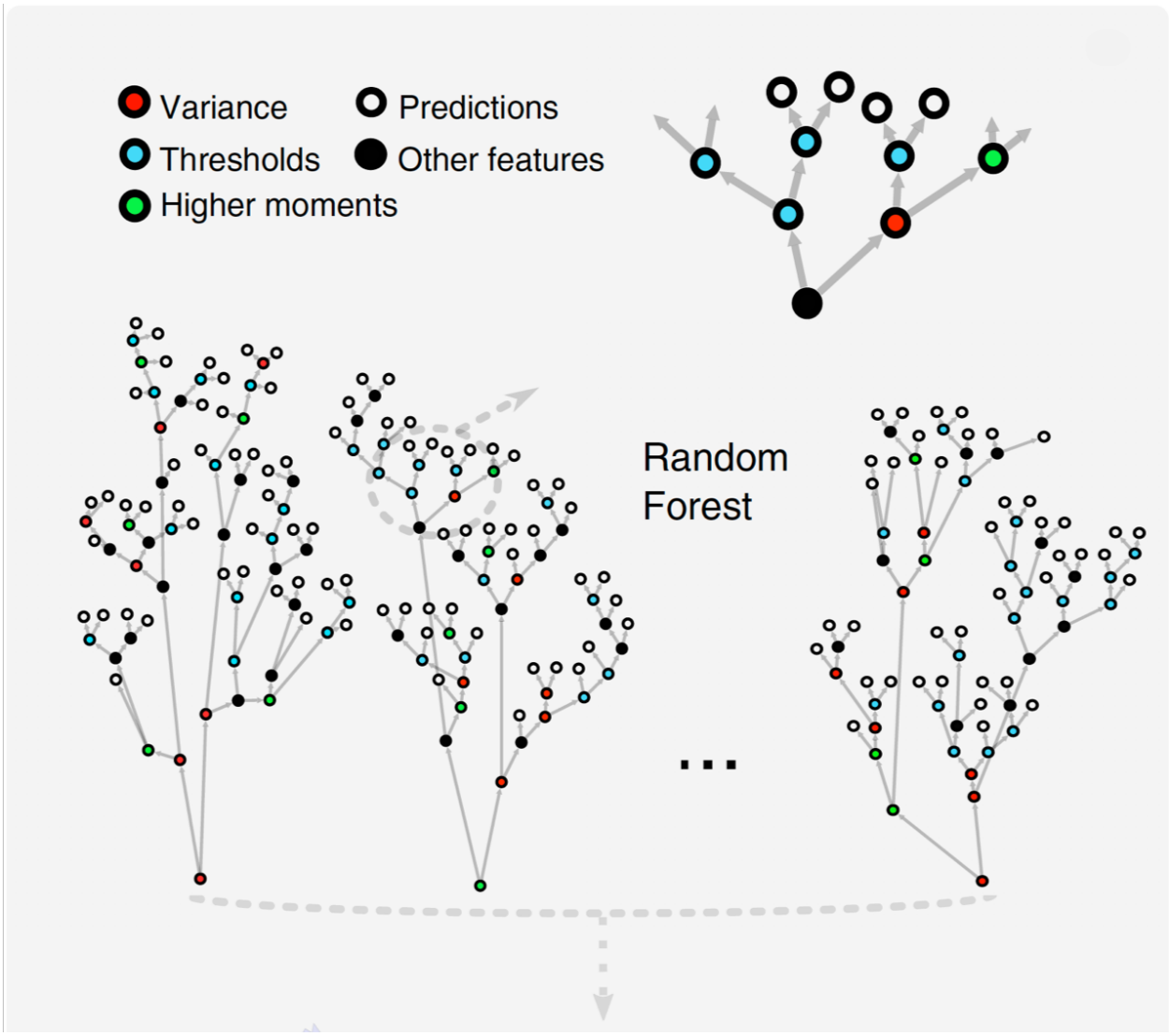
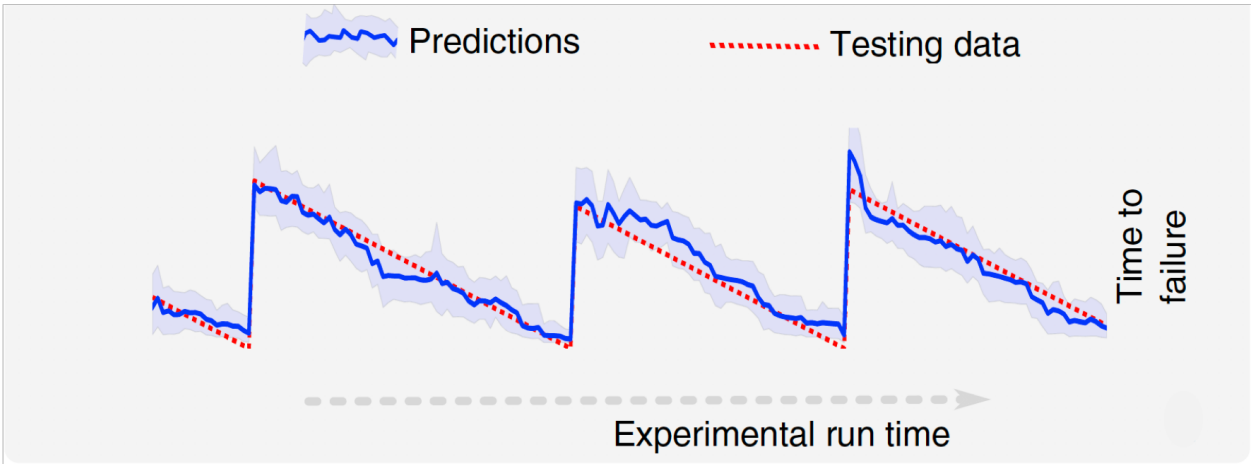


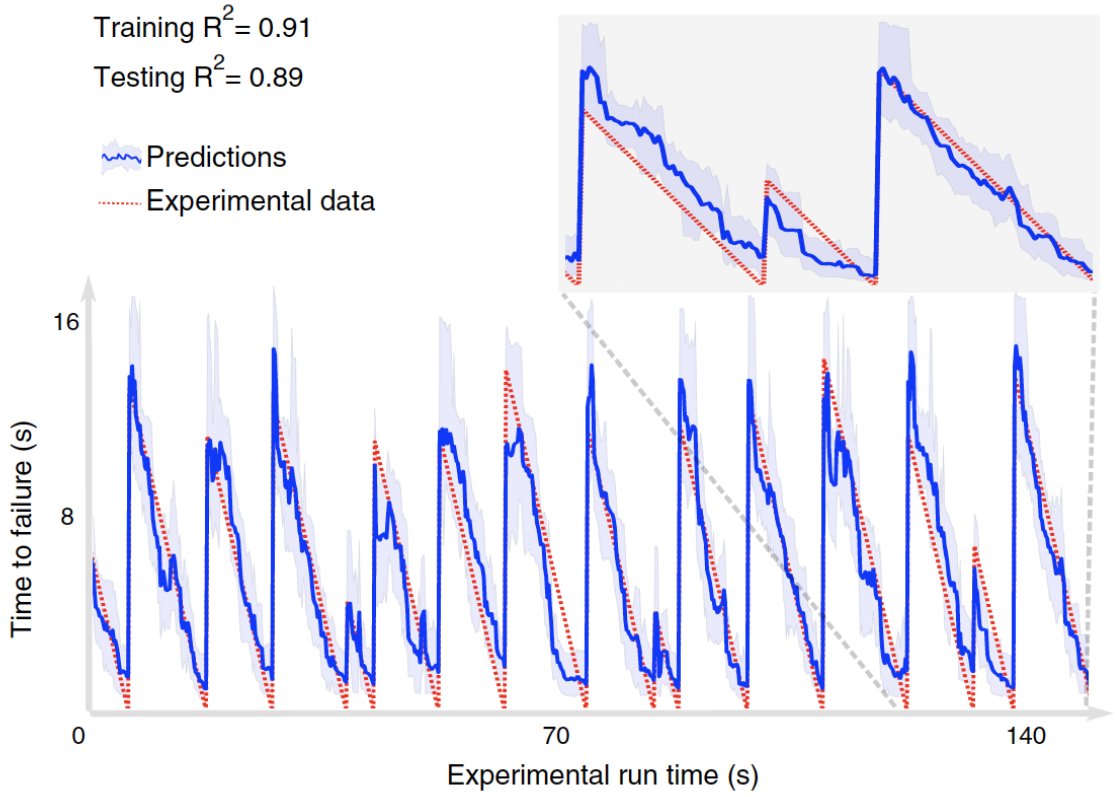
Figure. Random Forest (RF) approach for predicting time remaining before failure.

Our goal is to predict the time remaining before the next failure using only local, moving time windows of the AE data. We apply a machine learning technique, the random forest (RF), to the continuous acoustic time series data recorded from the fault. The RF model is an average over a set of decision trees. Each decision tree predicts the time remaining before the next failure using a sequence of decisions based on statistical features derived from the time windows shows the laboratory shear stress exhibiting multiple failure events during an experiment.



We wish to predict the time remaining before the next failure derived from the shear stress drops using the acoustic emission (dynamic strain) data.The RF model predicts the time remaining before the next failure by averaging the predictions of 1,000 decision trees for each time window.
Each tree makes its prediction (white leaf node), following a series of decisions (colored nodes) based on features of the acoustic signal during the current window. The RF prediction (blue line) on data it has never seen (testing data) with 90% confidence intervals (blue shaded region). The predictions agree remarkably well with the actual remaining times before failure (red curve). We emphasize that the testing data are entirely independent of the training data, and were not used to construct the model.

When making a prediction (blue curve): each prediction uses only the information within one single time window of the acoustic signal. Thus, by listening to the acoustic signal currently emitted by the system, we predict the time remaining before it fails—a “now” prediction based on the instantaneous physical characteristics of the system that does not make use of its history. We quantify the accuracy of our model using R2, the coefficient of determination. The time to failure predictions from the RF model are highly accurate, with an R2 value of 0.89.

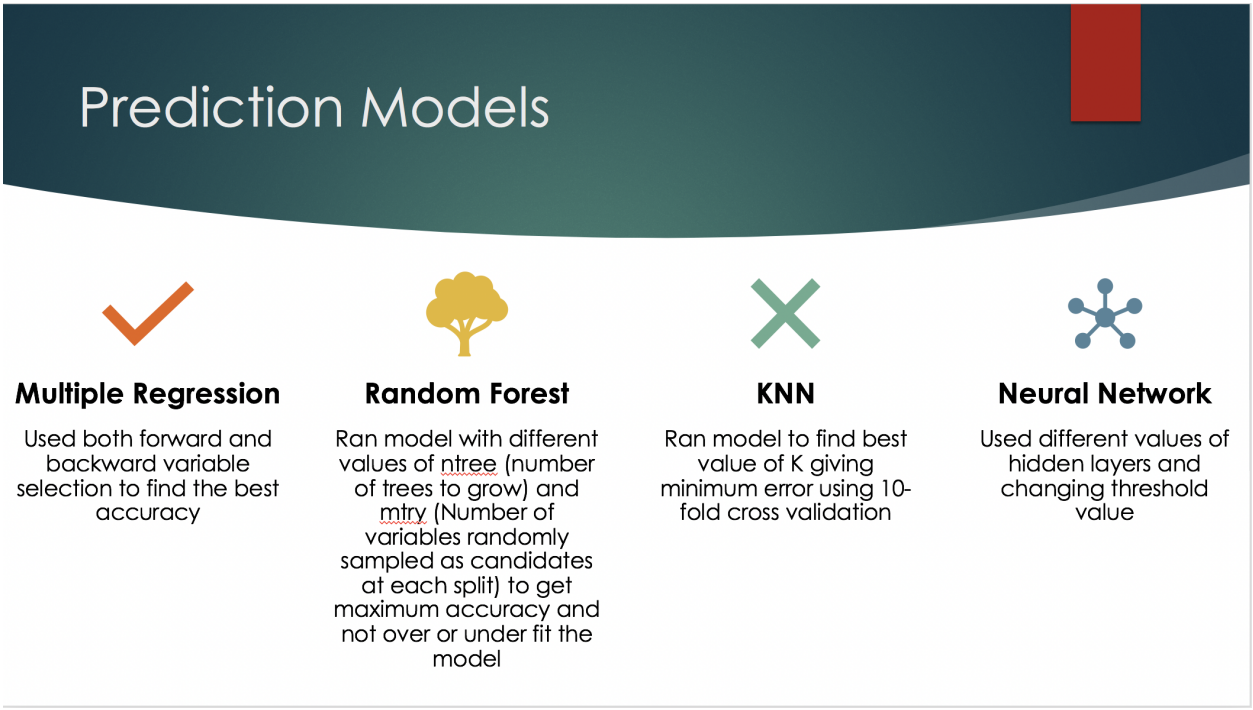


From each time window, we compute a set of approximately 100 potentially relevant statistical features (e.g., mean, variance, kurtosis, and autocorrelation). We find that statistics quantifying the signal amplitude distribution (e.g., its variance and higher-order moments) are highly effective at forecasting failure. The variance, which characterizes overall signal amplitude fluctuation, is the strongest single feature early in time. As the system nears failure, other outlier statistics such as the kurtosis and thresholds become predictive as well. These outlier statistics are responding to the impulsive precursor AE typically observed as a material approaches failure, including those under shear conditions in the laboratory and in Earth. These signals are due to
small, observable shear failures within the gouge immediately preceding the laboratory earthquake.

Machine learning Models:

We applied recent advances in machine learning to data using R Studio, Jupyter notebook at EC2 instance (AWS). We choose NN-Artificial neural network model, Multiple Regression , KNN, Random Forest to conduct analysis. We ran model to find best value of K giving minimum error using 10-fold cross validation. We ran Multiple regression, which is an extension of simple linear regression, to predict the value of a variable based on the value of the other variables. We also run

KNN model, which is a non-parametric, lazy learning algorithm, to use the database in which the data points are separated into several classes to predict the classification of a new sample point. The RF model is an average over a set of decision trees. Each decision tree predicts the time remaining before the next failure using a sequence of decisions based on statistical features derived from the time windows. Among those models, KNN has the best prediction accuracy.

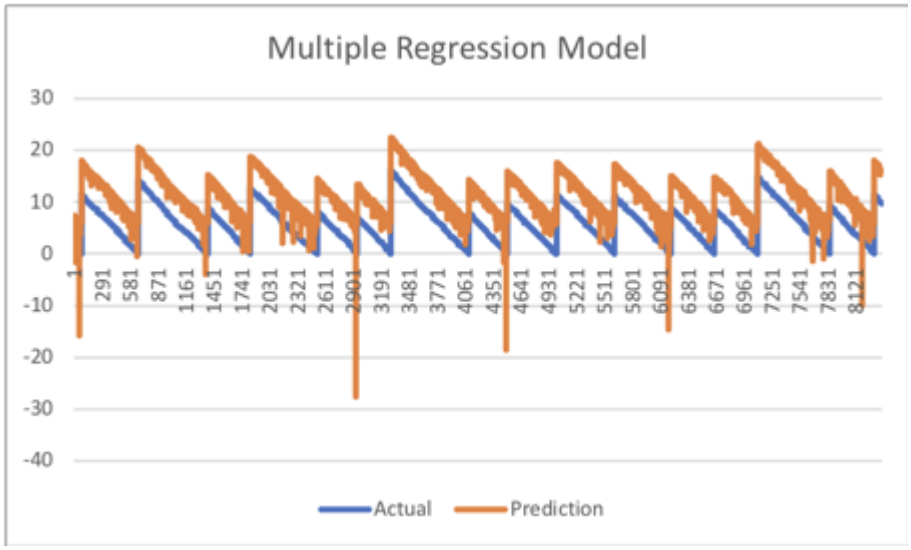


Multiple Regression :

```
# partition
mydata <- read.csv("newTrain2.csv")
set.seed(1)
train.index <- sample(c(1:dim(mydata)[1]), dim(mydata)[1]*0.8)
train <- mydata[train.index, ]
valid <- mydata[-train.index, ]

#train model
regmodel <- lm (time_to_failure ~.,data = train)
summary(regmodel)
```

```
library(forecast)
regpred <- predict(regmodel, valid)
accuracy(regpred, valid$time_to_failure)

#Variable Selection
stepPred <- step(regmodel, direction = "both")
summary(stepPred)  # Which variables did it drop?
stepPred.pred <- predict(stepPred, valid)
accuracy(stepPred.pred, valid$time_to_failure)

linearModel = data.frame(Actual = valid$time_to_failure, Prediction = regpred)
library(tidyverse)
write_csv(linearModel, path = "linear.csv")
```

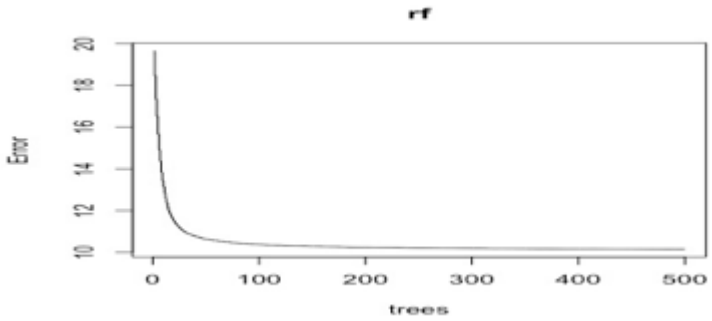
```
Call:
lm(formula = time_to_failure ~ ., data = train)

Residuals:
    Min      1Q  Median      3Q     Max
-20.141  -2.781  -0.329   2.312  31.879

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  8.996e+00  3.133e-01  28.715  < 2e-16 ***
Variance     7.132e-04  1.743e-05  40.918  < 2e-16 ***
Kurtosis    -8.079e-03  1.699e-03  -4.756 1.98e-06 ***
Mean        -3.813e-01  1.077e-01  -3.540 0.000401 ***
Median       2.555e-02  6.385e-02   0.400 0.689022
Percentile  -1.294e-02  7.062e-03  -1.832 0.066903 .
Amax         1.247e-02  9.612e-04  12.977  < 2e-16 ***
Amin        -4.948e-03  9.427e-04  -5.248 1.54e-07 ***
STD         -4.265e-01  3.098e-02 -13.769  < 2e-16 ***
Skew        -2.159e-02  7.339e-02  -0.294 0.768594
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.554 on 33545 degrees of freedom
Multiple R-squared:  0.07015,   Adjusted R-squared:  0.0699
F-statistic: 281.2 on 9 and 33545 DF,  p-value: < 2.2e-16
```

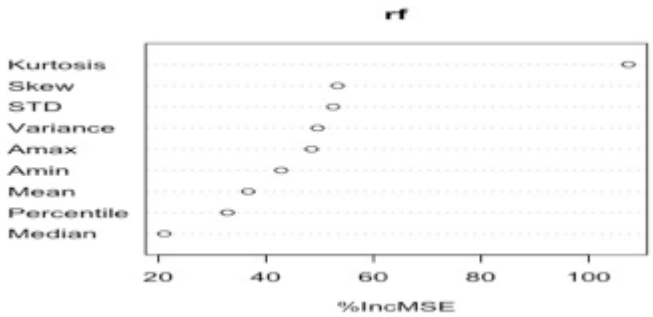


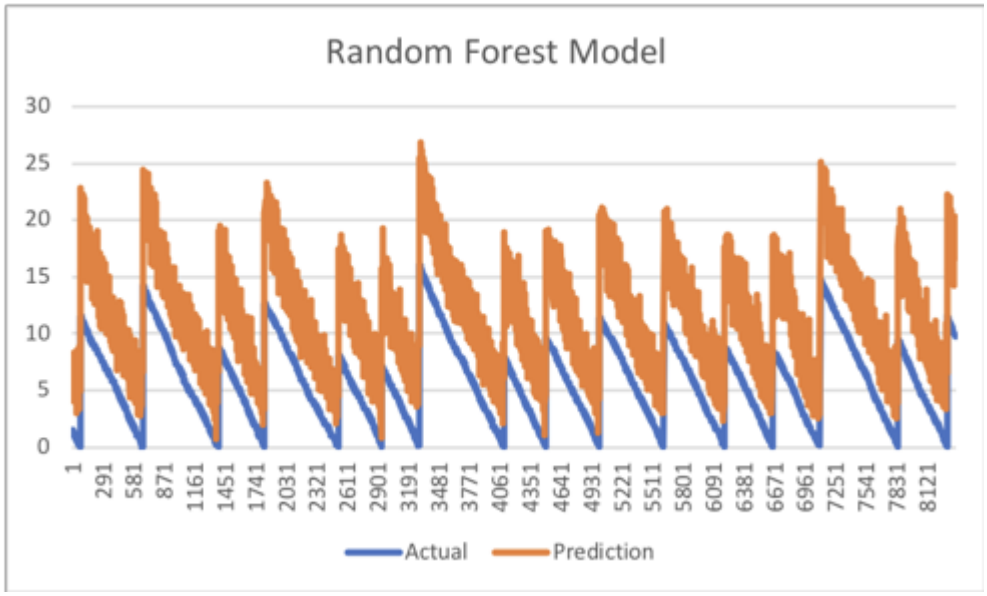
Random Forest :

```
# partition
mydata <- read.csv("newTrain2.csv")
set.seed(1)
train.index <- sample(c(1:dim(mydata)[1]), dim(mydata)[1]*0.8)
train <- mydata[train.index, ]
valid <- mydata[-train.index, ]
library(randomForest)
rf <- randomForest(time_to_failure ~ ., data = train, ntree = 500,
                   mtry = 4, nodesize = 5, importance = TRUE)

library(devtools)
library(reprtree)
tree <- getTree(rf, k=1, labelVar=TRUE)
realtree <- reprtree:::as.tree(tree, rf)

summary(rf)
summary(tree)
head(rf$votes,10)
```

```
## Plot forest by prediction errors
plot(rf, type = "simple")

## variable importance plot
varImpPlot(rf, type = 1)

## confusion matrix
rf.pred <- predict(rf, valid)
library(forecast)
accuracy(rf.pred, valid$time_to_failure)
randomForestModel = data.frame(Actual = valid$time_to_failure, Prediction = rf.pred)
library(tidyverse)
write_csv(randomForestModel, path = "rf.csv")
```





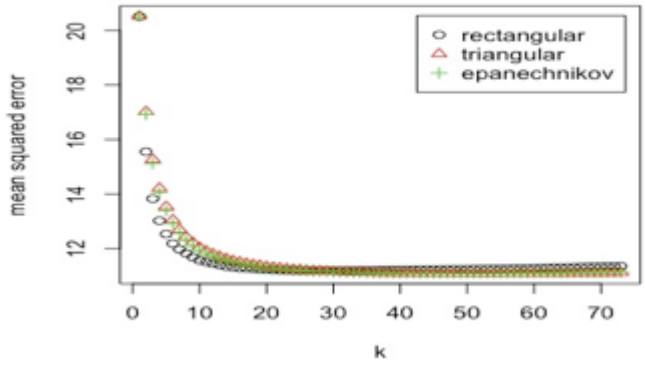


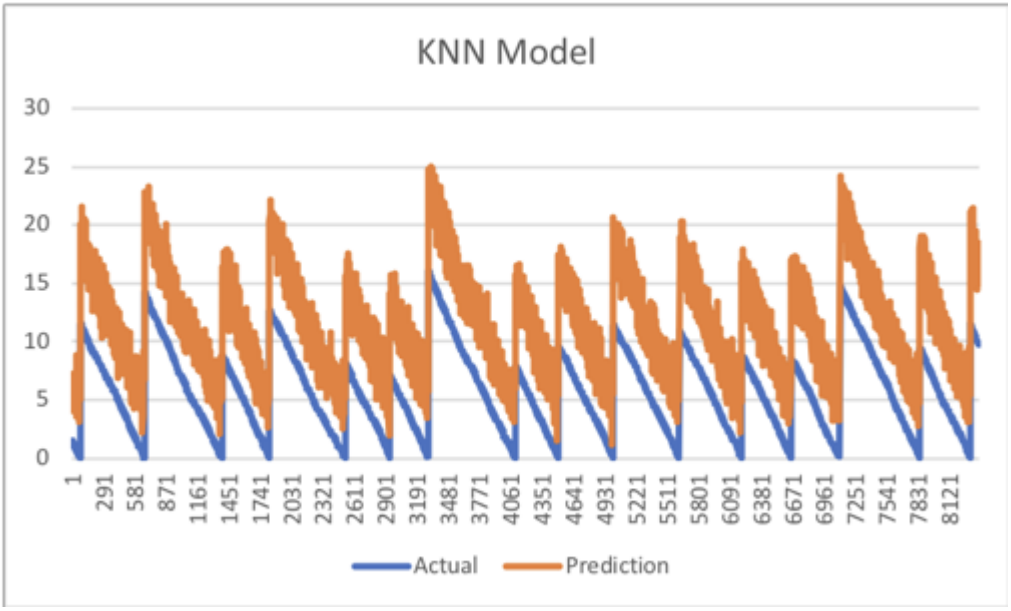
KNN Model :

```
# partition
mydata <- read.csv("newTrain2.csv")
set.seed(1)
train.index <- sample(c(1:dim(mydata)[1]), dim(mydata)[1]*0.8)
train <- mydata[train.index, ]
valid <- mydata[-train.index, ]

library(DMwR)
set.seed(502)
grid1 <- expand.grid(.k = seq(2, 150, by = 10))
control <- trainControl(method = "cv")
knn.train <- train(time_to_failure ~ ., data = train,
                   method = "knn",
                   trControl = control,
                   tuneGrid = grid1)
knn.train
```

```
knn.pred <- predict(knn.train, newdata = valid)

library(kknn)
set.seed(123)
kknn.train <- train.kknn(time_to_failure ~ ., data = train, kmax = 80,
                         distance = 10,
                         kernel = c("rectangular", "triangular", "epanechnikov"))
plot(kknn.train)

kknn.train
kNN.pred <- kNN(time_to_failure ~ .,train,valid,norm=TRUE,k=73)

kknn.pred <- predict(kknn.train, newdata = valid)

KnnModel = data.frame(Actual = valid$time_to_failure, Prediction = kknn.pred)
accuracy(kknn.pred, valid$time_to_failure)
library(tidyverse)
write_csv(KnnModel, path = "knn.csv")
```

```
Type of response variable: continuous
minimal mean absolute error: 2.673926
Minimal mean squared error: 11.08715
Best kernel: triangular
Best k: 53
>
```

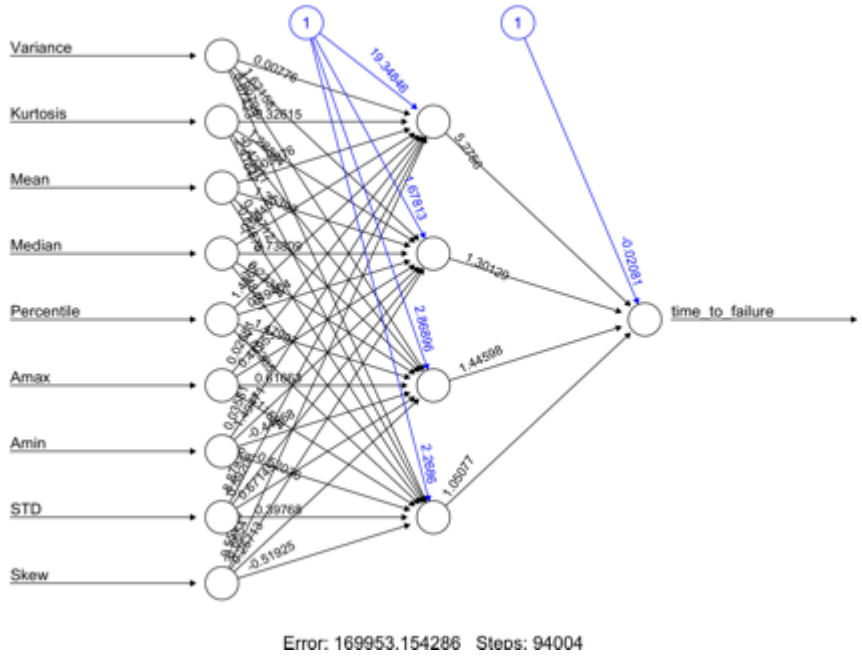


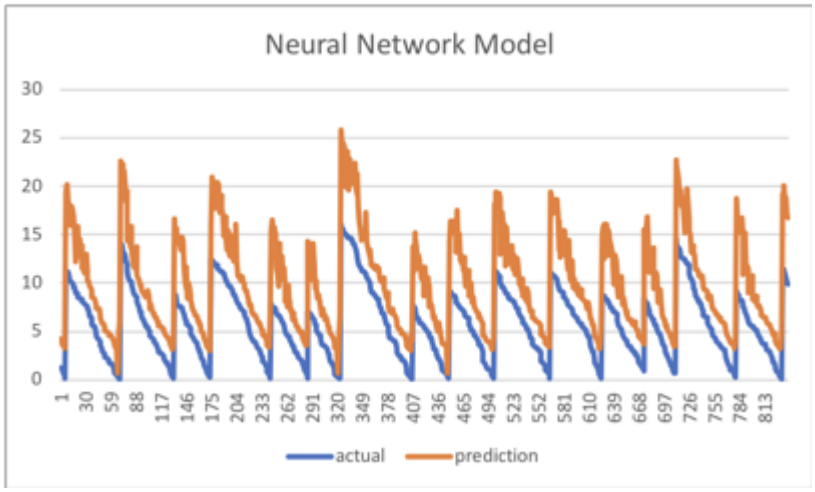


Neural Network Model :

```
# partition
mydata <- read.csv("newTrain2.csv")
set.seed(1)
train.index <- sample(c(1:dim(mydata)[1]), dim(mydata)[1]*0.8)
train <- mydata[train.index, ]
valid <- mydata[-train.index, ]
library(neuralnet)
library(nnet)
library(caret)
```

```
nn <- neuralnet(time_to_failure ~ ., data = train, hidden = 4, linear.output = TRUE, threshold = 0.1)
nn$result.matrix
plot(nn)

nn.results <- neuralnet::compute(nn, valid[, -c(8)])

results <- data.frame(actual = valid$time_to_failure, prediction = nn.results$net.result)

accuracy(results$prediction, valid$time_to_failure)

library(tidyverse)
write_csv(results, path = "nn.csv")
```





Comparison of Error values of all models :

```
> accuracy(regpred, valid$time_to_failure)
                    ME     RMSE      MAE       MPE     MAPE
Test set -0.02572933 3.49162 2.857612 -348.9805 385.2563
>
```

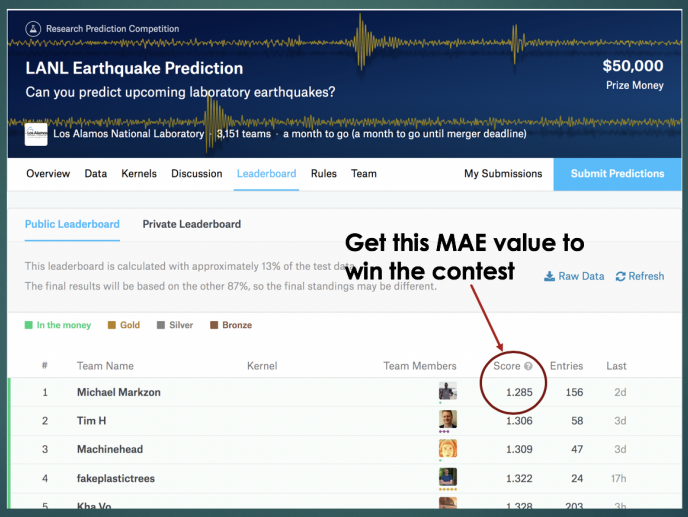
```
> accuracy(rf.pred, valid$time_to_failure)
                    ME     RMSE      MAE       MPE     MAPE
Test set -0.04710793 3.130435 2.486949 -414.3376 440.607
>
```

```
> accuracy(kknn.pred, valid$time_to_failure)
                    ME     RMSE      MAE       MPE     MAPE
Test set -0.009574229 3.263729 2.625079 -389.4562 416.4801
```

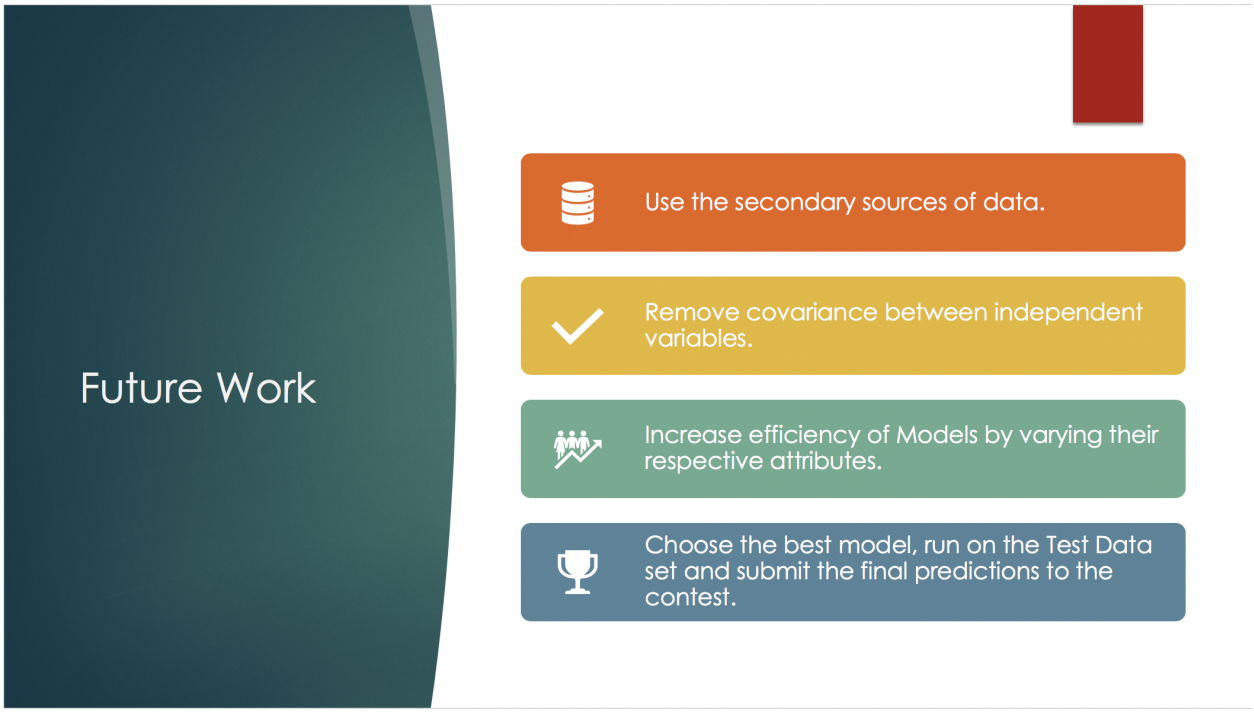
```
> accuracy(results$prediction, valid$time_to_failure)
                ME     RMSE      MAE       MPE     MAPE
Test set 1.626368 3.177401 2.437937 -148.2669 199.5417
```

Our Target :

MAE Value required for contest



Future Work :



Conclusion:

To summarize, we show that ML applied to this experiment provides accurate failure forecasts based on the instantaneous analysis of the acoustic signal at any time in the slip cycle and reveals a signal previously unidentified. These results should suffice to encourage ML analysis of seismic signals in Earth. In particular, ML-based approaches mitigate human bias by automatically searching for patterns in a large space of potentially relevant variables. Our current approach is to progressively scale from the laboratory to the Earth by applying this approach to Earth problems that most resemble the laboratory system. An interesting analogy to the laboratory may be faults that exhibit small repeating earthquakes. For instance, fault patches located Repeaters at these fault patches may be emitting chattering in analogy to the laboratory. If so, can this signal be recorded by borehole and surface instruments? We are currently studying this problem. Whether ML approaches applied to continuous seismic or other geophysical data succeed in providing information on timing of earthquakes (not to mention the challenge of predicting earthquake magnitude), this approach may reveal unidentified signals associated with undiscovered fault physics. Furthermore, this method may be useful for failure prediction in a broad spectrum of industrial and natural materials. Technology is at a confluence of dramatic advances in instrumentation, machine learning, the ability to handle massive data sets and faster computers. Thus, the stage has been set for potentially marked advances in earthquake science.

Check next page for my experience…

My Experience in this Project:

Introduction:

I had a whole different experience in this project. I still remember in the first class, we were asked about what we would do in our final project. Although we were not sure at that time and explained about something which we would want to implement that has the power to change the world or any new kind of innovative process which we would like to see around us. In the end, we did implement something but slightly different from what we discussed in our first class. This actually made me realize one thing that on the go, there's a high possibility that we will have to change our approach multiple times. Eventually we did a project called Earthquake prediction because we strongly believed that this product has the potential to save human lives. And that started our journey in this area.

Team formation and Finalizing the project:

The second stage in our project was team formation. We tended to keep our team diverse. We are all four from different backgrounds and really good at what we do. This was kind of once in a while experience where nobody was a data scientist in their previous role and we went on a journey to do highly data reliable project.

Project approach:

We worked in a Software development life cycle (SDLC) kind of approach. The first step was to gather the requirements. So, we checked a lot of the requirements given by you and finalized a project. During the whole journey, we got a lot of feedback from you, which kind of made this project an agile project as well. I believe that the industry has gone agile and we have to keep ourselves updated with the new technology around us. Requirements changes every time and the gap between different teams need to be narrowed down. We have to accept change in requirements no matter how early or late in the project we are. To me it was a part academic and part industrial project. You clearly motivated us to build a project by creating an environment of appreciation and constant feedback.

Our strategy was to collect data from different sources, merge them, run different kind of models to find the best model and conclude which model could predict better. Upon your feedback, we tried to increase the complexity in our models by adding different features such as AWS, Tableau etc. to make our predictions better and visualize the outputs. The cost for running some of those models on cloud were incurred by the PHD students from Nevada State University as the collaboration also helped them out in their research.

Implementation:

a) **Data Collection**

There were several challenges during implementation of this project. First was finding the right data set. After analyzing data from several sources, we finalized on the one which was the most complex of all. Our data was huge in size (approximately 10 Gb) and it had around 660 million rows.

b) Data Cleaning

We had to further clean the data to remove inconsistencies and errors in order to make our predictions better. We also had to extract the data.

c) **Learning Earthquake terms**

Learning key terms in Earthquake was very important since we were doing a project in Earthquake prediction. We had several sessions with the students from Nevada State University to understand how different variables used are correlated. Basically, our main goal was to find out what are all the factors which come into play when we predict an Earthquake. These sessions gave us a clear understanding of what we want to achieve. For example, we tried to understand what time to failure is, what stress and strain is.

d) Loading of data

Once we have got our data, we didn't run our predictions directly on our systems. Even if we wanted to, we could not do that due to the size of the data. Our systems did not have sufficient power to process that much information. That's how Amazon cloud came into picture. We tried to leverage the power of AWS to run our queries. For this purpose, we connected our Jupyter notebook with an EC2 instance. This itself was a task since none of us had any experience in merging a Jupyter console with an EC2 server.

We verified several plans and which plan will suit us best. We decided on using an On demand t3.xlarge instance since our dataset was huge and we had more than 9 Gb's of data. That was the cheapest option with the highest CPU power of 4. Once decided, we created an EC2 instance and ran several commands to download the Jupyter notebook in that instance and later upload all our data onto that instance. Our EBS volume had a size of 16Gbs. This was a perfect way to get our hands on on AWS and learn something new.

e) ETL

One of the tasks deliverables you gave us was to understand the ETL process and implement that in our project. Since our data was static, we could not do that in our project. But we had to come up with a hypothetical solution and architecture as to how would we implement that in our project. We learned about kafka and its usage and had to learn about dataware houses and how would kafka fit inoto our model. It was once again an amazing experience from architectural point of view since it broadens my horizon into a different level. It's definitely going to help me in my internship as well since I had to think as an architect and learn about the problems and outcomes.

f) Running models on data

Since we ran four different models on data, I can say that we got pretty good hands-on on several approaches and how to get best predictions. We had to research about several different models and their implementation approaches. Some of the models were new to us this gave us a pretty decent opportunity to flex our coding skills. It also helped us with our troubleshooting skills and gave us an exceptional chance to predict the outcomes on a dataset as huge as an organization can provide. I firmly believe that what we did in our project would definitely help for Data Engineer jobs as well as Data Analyst/Scientist jobs. We have seen every phase in an analysis whether it is collection, cleaning, loading and making predictions. These are the must skills for the today's agile demands.

Conclusion:

This project clearly improved our collaboration skills, presentation skills and prediction skills. It taught us about how to formulate a strategy and its implementation. And the idea to improve the complexity was fantastic since it taught us about integration of different tools and technologies and how to leverage those tools to make our predictions and interpretation better. I would also like to say that this project has a clear impact on my personality.