The Shake Love Model:
A New Class In Determining Earthquake Insurance Risk

Abstract

Measurable Objective: This model will classify a structure into one of 5 risk categories in order to predict the market rate earthquake insurance premium for a residential structure by weighing a number of relevant variables based on extensive research and placing that data into a KNN model machine learning algorithm that will use the data to correctly place and therefore predict a residential structures earthquake risk and thus insurance premium.

Before deciding upon a machine learning model, it was necessary to do research in the field where the model was to be applied. Thus I will begin this overview with the field of seismology, specifically earthquakes, the waveforms, and how they affect structures.

There are many factors that determine whether or not a building will survive a large magnitude earthquake. Factors such as building construction type, the height or number of stories a building has, local building and construction codes and regulations (referred to in this model as “age_eras”), structural enhancements or retrofitting, the composition type of the ground or soil, as well as the distance to the nearest fault line and lastly, and in many cases most importantly, the type of earthquake waveform that fault line produces.

This initial machine learning model has been designed for California. A second generation model would be needed for places like Taiwan or Japan. The reason for this is that earthquakes differ greatly around the world, and thus a model that works in California will most likely not work in Japan, due to the differences in their fundamental geology.

Seismic activity exists all around the world as a result of the geological phenomenon known as plate tectonics. The surface of the earth is made up of plates which are slowly but surely moving in some direction and manner in relation to the neighboring plate. This slow but steady movement is the underlying force behind volcanoes, tsunamis, and earthquakes, as well as other geological phenomena.

Some plates are moving underneath another plate; this is called subduction. As a result of this unique type of plate tectonic movement, a specific type of earthquake waveform is produced known as a Raleigh wave, such as in Japan. In other areas, the plates are trying to slide past one one another in opposite directions. This type of plate movement is called strike-slip. As a result of this unique type of plate tectonic movement, a specific type of earthquake waveform is produced known as a Love wave, such as is the case along the San Andreas fault line in California.

Raleigh waves, as in Japan, are big and long and produce earthquakes that are longer in duration and produce an undulating, rolling movement at the earth’s surface. This can be best

compared to an ocean wave. Smaller buildings can roll over these waves like a boat on the surface of the ocean while tall buildings like skyscrapers are most vulnerable and can be brought down. Converversely, a strike-slip fault, like the San Andreas, produces a much different style earthquake. These earthquakes tend to be much shorter in duration lasting from 20 seconds to a minute, roughly. These earthquakes are characterized by sharp, choppy movement and intense shaking back and forth. This creates a situation wherein skyscrapers are least prone to falling while 2-4 story buildings are most vulnerable.

This initial model is meant to determine earthquake risk and insurance premiums for structures that have been built in California which experiences sharp, choppy earthquakes, having greater effect on smaller buildings, due to the underlying plate tectonics described above. A second generation of this model would have to be modified to calculate insurance risk for structures in Japan or Taiwan where long, rolling, Raleigh style earthquakes affect buildings that are taller.

Generation One of this model, which I have named the “Shake Love” model, has been designed for the specific type of earthquake that results from California’s underlying geographical features. The model takes into consideration the following categories and weighs the risk of the individual features in each category in relation to each other.

Phases:

Planning & Research / Extract Transform Load / Data Analysis / Machine Learning Training and Testing / Machine Learning: Enhancing the Model / Visualizing the Results

The project was split into 6 phases:

Phase 1 Research: Phase 1 concentrated on planning the project, considering what measurable objectives could be achieved in the given time frame. Research into Machine Learning as well as seismology, structural engineering, and other related areas were concurrent and this approach has seemed useful. Admittedly, there has been some jumping between phases as the need to better understand machine learning made it necessary to take small sample sets of early data and run them through the ML process to better determine what was needed as well as what ML model to choose. Additionally, the project’s scope had to be narrowed from starting with California to choosing to major cities within the state. One must keep in mind that California is, both in terms of population and geography, the size of many European nations, and to attempt an accurate Machine Learning model on such scale given the time allowed would have been highly unlikely. Thus, San Francisco and Los Angeles were chosen to narrow the scope in the interest of obtaining more accurate Machine Learning results.

Phase 2 Extract, Transform, Load (ETL):

The 2nd phase was the extraction, transformation, and loading of numerical data into a server. Data from the USGS and its partnering institutions’ databases was extracted,

transformed, and loaded into a PostgreSQL server using PGAdmin. Here, the numerical data could be easily sorted and cleaned not only for my own research and understanding but also so that models and visualization of this data could be undertaken at a later stage. During the phase, the data was transformed in order to meet the goals of the project which included the dropping of extraneous columns, reformatting date and time when necessary, and creating new tables specifically designed to meet the objectives of this particular project.

Phase 4 Data Analysis: The truth is that the analysis of data was occurring in some manner at every stage leading up to Phase 4. However, it was in this phase that I was able to really nail down what it was that I had, and how it was to be used in the Shake Love ML model. The relationships between the variables of building type, building damage level, earthquake intensity, region, distance from epicenter, the role of retrofitting, soil type/ liquefaction, and building eras all came together in a way that made sense in this phase. Now that I could better understand the relationships, I positioned myself for better results with the Shake Love Machine Learning model. For me, all these steps were crucial in order to train and test the classification model I knew I needed to use and achieve accurate and meaningful results with real world applicability.

Phase 5 Machine Learning Training: The final phase, phase 5, consisted of two smaller phases, the first being the need to determine which machine learning model was best and the second part was the actual machine learning itself. With the research and data in hand, I knew I wanted to develop a prototype earthquake risk and damage casualty estimation model based on classifications. Thus, I tried both Random Forest and KNN models.

Phase 5 Machine Learning Improvement:

Phase 6 Data Visualization:

Methodology:

Determining Location, Weight, and Feature Variables in the "Shake Love" Risk Model

Weight Scale: The categories are ordered by weight with 1 being the most desirable or best score and 5 being least desirable or worst. Thus, as in golf, the lower the score the better. On this scale the par for the course is 6. Unlike in golf, it is not possible to score below par. So the best score is a 6. A building can, however, score well above par. The worst score on this scale is a 30. Most buildings score somewhere in between.

Features Used to Determine Weight Scale: These features have been determined by extensive research, given the time parameters, and are the result of the hard work and publication of said work by numerous individuals and professional organizations around the world. All research has been duly noted in the Citations section and I would like to thank the individuals and organizations that have done the work upon which I have based my own.

Weighted Features to Start 1-5 Top Down Best to Worst

Feature 1 Construction Type/ Materials Used	**Feature 2 Height of Structure / Stories or Floors / Residential** (best to worst)	**Feature 3 Era: Age of Structure (years)**	**Feature 4 Reinforced: Color Tag System**	**Feature 5 Soil Type**	**Feature 6**
wood	Residential 15-20 stories	BANR	Green	hardRock	Distance from Fault
metal	Residential 11-14 stories	NEHRP	Blue*	Rock	VeryFar
steel	Residential 6-10 stories	WERF	Yellow	Sandstone	Far
concrete	Residential 1 story	EERI	Orange*	Stiff	Average
masonry	Residential 2-5 stories	FIELD	Red	Soft	Close
					VeryClose

WHAT ARE THE CHANCES OF A MAJOR CALIFORNIA EARTHQUAKE?

California has **more than a 99% chance** of having one or more magnitude 6.7 or larger earthquakes within the next 30 years, **according to scientists**.

- For Northern California, the most likely source of such earthquakes is the Hayward-Rodgers Creek Fault (31% in the next 30 years). Such quakes can be deadly, as shown by the 1989 magnitude 6.9 Loma Prieta earthquake. The Loma Prieta quake caused 63 deaths, 3,757 injuries, and about $6 billion in damage.

- For **Southern California**, within the next 30 years the probability is 60% for an earthquake measuring magnitude 6.7. The southern San Andreas fault is much closer to Los Angeles, running at its closest just 25 miles northeast of downtown through the San Gabriel Mountains, and is capable of unleashing a magnitude 8.2 quake throughout all of Southern California. The 2019 Ridgecrest earthquake happened on a brand new fault line along the Eastern California Shear Zone.

SF Scenario based off historical data

https://docs.google.com/spreadsheets/d/13bS5MKT0q1zTMK8AFQRyERS5laKhvVng1of4Vs2wo-E/edit#gid=577737987

SF / Loma Prieta
50 miles away
7.0 earthquake
-will destroy or severely damage (red tag): 5%
-will damage (yellow tag) 10%
CURRENTLY half of all buildings in SF were built pre 1940; 184,175 units — were built before 1940
8% of SF residents live in wood framed, unreinforced, pre 1940 buildings

Features List: Features are listed from best to worst, thus “wood” is 1
Sources are in citations

Feature 1
Construction Type/ Materials Used
wood
metal
steel
concrete
masonry

Feature 2
Height of Structure / Stories or Floors / Residential (best to worst)
Residential 15-20 stories
Residential 11-14 stories
Residential 6-10 stories
Residential 1 story
Residential 2-5 stories

Feature 3

Era: Age of Structure (years)
BANR
NEHRP
WERF
EERI
FIELD

Feature 4
Reinforced: Color Tag System
Green
Blue*
Yellow
Orange*
Red

Feature 5 Soil Type
hardRock
Rock
Sandstone
Stiff
Soft

Feature 6
Distance from Fault
VeryFar
Far
Average
Close
VeryClose

EQ DATA NUMERICAL

USGS
Loma Prieta
https://earthquake.usgs.gov/earthquakes/eventpage/nc216859/impact
https://earthquake.usgs.gov/earthquakes/eventpage/nc216859/executive
https://earthquake.usgs.gov/earthquakes/eventpage/nc216859/origin/detail

Northridge
http://ds.iris.edu/wilber3/find_stations/350828

https://earthquake.usgs.gov/earthquakes/eventpage/ci3144585/executive
https://earthquake.usgs.gov/earthquakes/eventpage/ci3144585/origin/detail

California 30 years 5.0+ All Earthquakes
https://docs.google.com/spreadsheets/d/1iTUeJoM7Dp0ov6Br7ZQzE_T9kiAOuOmQmT04qHeFfoE/edit#gid=39685614

CHOOSING THE MACHINE LEARNING MODEL RESEARCH

(machine learning section is currently all research and articles with no original material, will put in original writing here once model works consistently)

Resources
https://modelzoo.co/
https://azure.microsoft.com/en-us/services/machine-learning

Data Lake House
https://us.nttdata.com/en/blog/tech-blog/what-is-a-data-lakehouse
A data lakehouse offers a metadata layer that sits over the top of a data lake architecture. It combines the best of data warehousing and data lakes by taking advantage of the cheaper cloud-based storage that data lakes use, and the data management and structure used by data warehouses. In doing so, the lakehouse allows organizations to **quickly query massive amounts of structured and unstructured data** that can sit in cloud stores like Azure Data Lake Store (ADLS).

Strengths:

Data Warehouse	- Fixed, structured data eases governance. - Viewed as a single source of truth. - Established ecosystem for extract, transform, load (ETL) tools and processes.

Data Lake	- Takes advantage of the least expensive storage types, allowing you to inexpensively store TBs of data. - Supports both structured and unstructured data, giving you the flexibility to store everything from JSON to images and blobs. - Ecosystem of tools that can read a wide variety of data quickly and inexpensively. - Easily scales.

Data Lakehouse	- Adds a metadata layer to provide structure to the data lake for manageability and governance. - Makes the data transactional for increased user confidence. - Low-cost interface allows you to take advantage of cheaper cloud storage options. - Connects with popular reporting tools. - Schema enforcement (and schema evolution).

Machine Learning Approach Possibilities
https://machinelearningmastery.com/types-of-classification-in-machine-learning/

- **k-Nearest Neighbors.**
- **Decision Trees.**
- **Naive Bayes.**
- **Random Forest.**
- **Gradient Boosting.**

https://machinelearningmastery.com/types-of-classification-in-machine-learning/

Understanding the Random Forest with an intuitive example

When learning a technical concept, I find it's better to start with a high-level overview and work your way down into the details rather than starting at the bottom and getting immediately lost. Along those lines, this post will use an intuitive example to provide a conceptual framework of the random forest, a powerful machine learning algorithm. After getting a basic idea down, I move on to a simple implementation to see how the technique works and if it will be useful to me before finally working out the details by digging deep into the theory. With that in mind, after understanding the overview of the random forest here, feel free to check out part two of this post, an end-to-end example worked out in Python code. Taken together, these two articles will help you conquer the first two steps in the learning process and leave you well prepared to dive as far into the random forest and machine learning as you want!

Decision Tree: The Building Block

To understand the random forest model, we must first learn about the decision tree, the basic building block of a random forest. We all use decision trees in our daily life, and even if you don't know it by that name, I'm sure you'll recognize the process. To illustrate the concept, we'll use an everyday example: predicting the tomorrow's maximum temperature for our city. To keep things straight, I'll use Seattle, Washington, but feel free to pick your own city.

In order to answer the single max temperature question, we actually need to work through an entire series of queries. We start by forming an initial reasonable range given our domain knowledge, which for this problem might be 30–70 degrees (Fahrenheit) if we do not know the time of year before we begin. Gradually, through a set of questions and answers we reduce this range until we are confident enough to make a single prediction.

What makes a good question to ask? Well, if we want to limit the range as much as is possible, it would be wise to think of queries that are relevant to the problem at hand. Since temperature is highly dependent on time of year, a decent place to start would be: what is the season? In this case, the season is winter, and so we can limit the prediction range to 30–50 degrees because we have an idea of what the general max temperatures are in the Pacific Northwest during the winter. This first question was a great choice because it has already cut our range in half. If we had asked something non-relevant, such as the day of the week, then we could not have reduced the extent of predictions at all and we would be back where we started. Nonetheless, this single question isn't quite enough to narrow down our estimate so we need to find out more

information. A good follow-up question is: what is the historical average max temperature on this day? For Seattle on December 27, the answer is 46 degrees. This allows us to further restrict our range of consideration to 40–50 degrees. Again, this was a high-value question because it greatly reduced the scope of our estimate.
Two questions are still not quite enough to make a prediction because this year might be warmer or colder than average. Therefore, we also would want to look at the max temperature today to get an idea if the year has been unusually warm or cold. Our question is simple: what is the maximum temperature today? If the max temperature today was 43 degrees, it might be colder this year and our estimate for tomorrow should be a little lower than the historical average. At this point, we can feel pretty confident in making a prediction of 44 degrees for the max temperature tomorrow. If we wanted to be even more sure, we could consult additional sources such as AccuWeather or Weather Underground to get information such as the forecast for the max temperature that we could incorporate into our mental model. However, at some point there is a diminishing return to asking more questions, and we cannot keep gathering more data for ever. For now, let us settle with using those three questions to make a prediction.
So, to arrive at an estimate, we used a series of questions, with each question narrowing our possible values until we were confident enough to make a single prediction. We repeat this decision process over and over again in our daily lives with only the questions and answers changing. At this point we are nearly ready to make the connection to the decision tree, but let's take just a minute to look at a graphical representation of the intuitive steps we took to find our answer:

Decision Process for Temperature Prediction
We start with an initial guess based on our knowledge of the world and refine our estimate as we gain more information. Eventually, we stop gathering data and make a decision, which in this case is the max temperature prediction. Our natural approach to the problem is what we might call a question-and-answer flowchart. In fact, this flowchart is also a rudimentary model of a decision tree! However, we have not quite built a full decision tree because as humans we take some shortcuts that make sense for us but are not so intuitive to a machine.
There are two main differences between our illustrated decision process and a real decision tree. First, we have neglected to list the alternative branches, that is the predictions we would have made if the answers to the questions had been different: for example, if the season had been summer instead of winter, our range of predictions would have been shifted higher. Moreover we phrased our questions such that they could take on any number of answers: when we asked 'what is the maximum temperature today?' the answer could be any real value. In contrast, a decision tree implemented in machine learning will list all possible alternatives to every question and will ask all questions in True/False form. This is a little tough to grasp because it is not

how humans naturally think, and perhaps the best way to show this difference is to create a real decision tree from our prediction process:

Machine Learning Decision Tree for Temperature Prediction

We notice right away that each question (the white blocks) has only two answers: True or False. Moreover, for each True and False answer there are separate branches. No matter the answers to the questions, we eventually reach a prediction (shown in the green blocks). This 'computer-friendly' version of the decision tree may look different than our intuitive model, but it works in the exact same way. Start at the node on the left, and progress through the tree answering the questions along the way. For our example, the season is winter so we take the True branch. We said the historical average was 46, so the second question is also True. Finally, the third answer is True as the max temperature today was 43. Therefore, the final prediction is 40 degrees for the max temperature tomorrow, close to our guess of 44.

This model here encompasses all the basic qualities of a decision tree. I purposefully left out all technical details, such as how the 'questions' are formed and how the thresholds are set, but these aren't really necessary to understand the model conceptually or even to implement it in Python code!

One aspect of the decision tree I should mention is how it actually learns. We refined our estimated range based on how the answers to the questions fit into our framework of the world. If the season is winter, our estimate is lower than if it is summer. However, a computer model of a decision tree has no prior knowledge and would never be able to make the connection 'winter = colder' on its own. It must learn everything about the problem from the data we provide it. We know how to translate answers from the flow-chart into reasonable predictions because our of daily experiences. In contrast, the model must be taught each of these relationships such as that if the temperature today is warmer than the historical average, the max temperature tomorrow will likely be warmer as well. As a supervised machine learning model, a random forest learns to map data (temperature today, historical average, etc.) to outputs (max temperature tomorrow) in what is called the training (or fitting) phase of model building.

During training, we give the model any historical data that is relevant to the problem domain (the temperature the day before, the season of the year, and the historical average) and the true value we want the model to learn to predict, in this case the max temperature tomorrow. The model learns any relationships between the data (known as features in machine learning) and the values we want to predict (called the target). The decision tree forms the structure shown above, calculating the best questions to ask in order to make the most accurate estimates possible. When we ask the decision tree to make a prediction for tomorrow, we must give it the same data it used during training (the features) and it gives us an estimate based on the structure it has learned. Much as humans learn from examples, the decision tree also learns through experience, except

it does not have any previous knowledge it can incorporate into the problem. Before training, we are much ‘smarter’ than the tree in terms of our ability to make a reasonable estimate. However, after enough training with quality data, the decision tree will far surpass our prediction abilities. Keep in mind the decision tree does not have any conceptual understanding of the problem even after training. From the model’s ‘perspective’, it is simply receiving numbers as inputs and outputting different numbers that agree with those it saw during training. In other words, the tree has learned how to map a set of features to targets with no knowledge of anything about temperature. If we ask the decision tree another weather related question, it will have no clue how to respond because it has been trained for one specific task.

That is basically the entire high-level concept of a decision tree: a flowchart of questions leading to a prediction. Now, we take the mighty leap from a single decision tree to a random forest!

From Decision Tree to Random Forest

My prediction for the maximum temperature is probably wrong. And I hate to break it to you, but so is yours. There are too many factors to take into account, and chances are, each individual guess will be high or low. Every person comes to the problem with different background knowledge and may interpret the exact same answer to a question entirely differently. In technical terms, the predictions have variance because they will be widely spread around the right answer. Now, what if we take predictions from hundreds or thousands of individuals, some of which are high and some of which are low, and decided to average them together? Well, congratulations, we have created a random forest! The fundamental idea behind a random forest is to combine many decision trees into a single model. Individually, predictions made by decision trees (or humans) may not be accurate, but combined together, the predictions will be closer to the mark on average.

Why exactly is a random forest better than a single decision tree? We can think about it terms of having hundreds of humans make estimates for the max temperature problem: by pooling predictions, we can incorporate much more knowledge than from any one individual. Each individual brings their own background experience and information sources to the problem. Some people may swear by Accuweather, while others will only look at NOAA (National Oceanic and Atmospheric Administration) forecasts. Perhaps one person relies on a meteorologist friend for their predictions while another uses hundred of years of temperature data. If we only ask one individual, we would only take advantage of their limited scope of information, but by combining everyone’s predictions together, our net of information is much greater. Furthermore, the more diverse each person’s source of information, the more robust the random forest is because it will not be swayed by a single anomalous data source. If NOAA goes rogue and starts making predictions over 100 degrees and everyone relied on NOAA, then our entire model would be worthless. If instead, individuals in our ‘forest’ use a number of different

weather sources, then our model will not be greatly affected by a single source and we can continue to make reasonable predictions.

Why the name ‘random forest?’ Well, much as people might rely on different sources to make a prediction, each decision tree in the forest considers a random subset of features when forming questions and only has access to a random set of the training data points. This increases diversity in the forest leading to more robust overall predictions and the name ‘random forest.’ When it comes time to make a prediction, the random forest takes an average of all the individual decision tree estimates. (This is the case for a regression task, such as our problem where we are predicting a continuous value of temperature. The other class of problems is known as classification, where the targets are a discrete class label such as cloudy or sunny. In that case, the random forest will take a majority vote for the predicted class). With that in mind, we now have down all the conceptual parts of the random forest!

Wrap-Up

Machine learning may seem intimidating at first, but the entire field is just many simple ideas combined together to yield extremely accurate models that can ‘learn’ from past data. The random forest is no exception. There are two fundamental ideas behind a random forest, both of which are well known to us in our daily life:

Constructing a flowchart of questions and answers leading to a decision

The wisdom of the (random and diverse) crowd

It is the combination of these basic ideas that lead to the power of the random forest model.

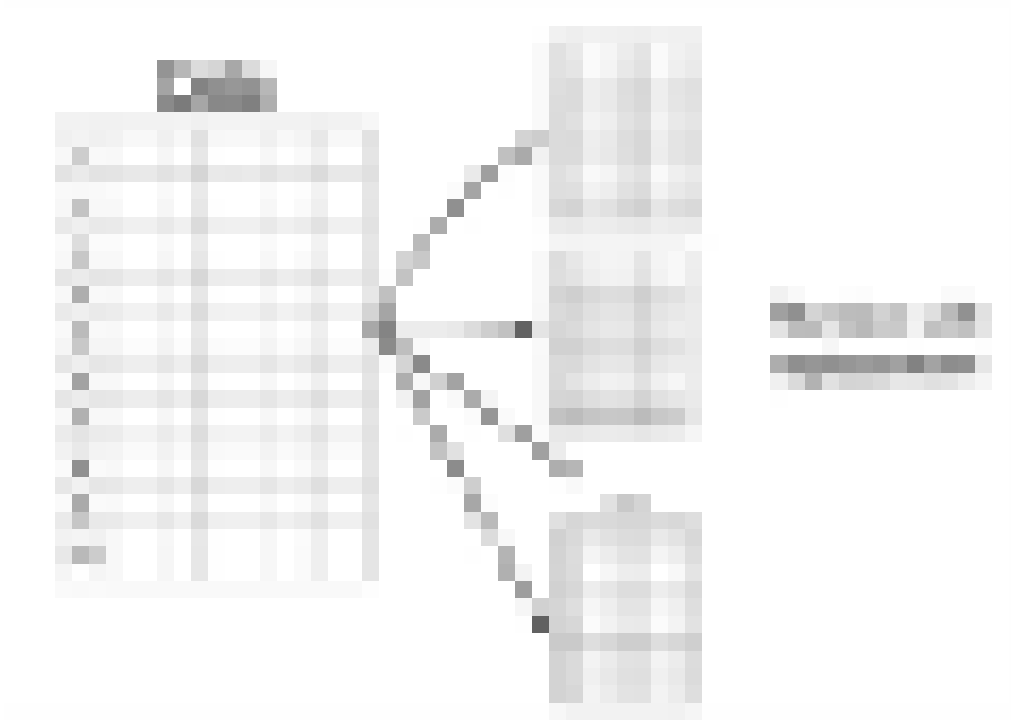
Now that you hopefully have the conceptual framework of a random forest, I recommend you take a look at the second part of this post, where we walk through the entire process of implementing a random forest for the max temperature problem in the Python programming language. Even if you have never programmed before, you should be able to follow along! If this post has sparked your curiosity, there are any number of resources out there to learn more. Sometimes too many resources can be overwhelming when trying to learn a new concept, so I’ll limit my recommendations to a great website made by the creator of the random forest (Leo Breiman), the free Coursera series on random forests, and the excellent book Introduction to Statistical Learning (by James, Witten, Hastie, and Tibshirani) which is available free online. With programming in general, and machine learning in particular, there are so many resources freely available online that match or even exceed any formal education so do not feel as if you are behind if you never saw this in college!

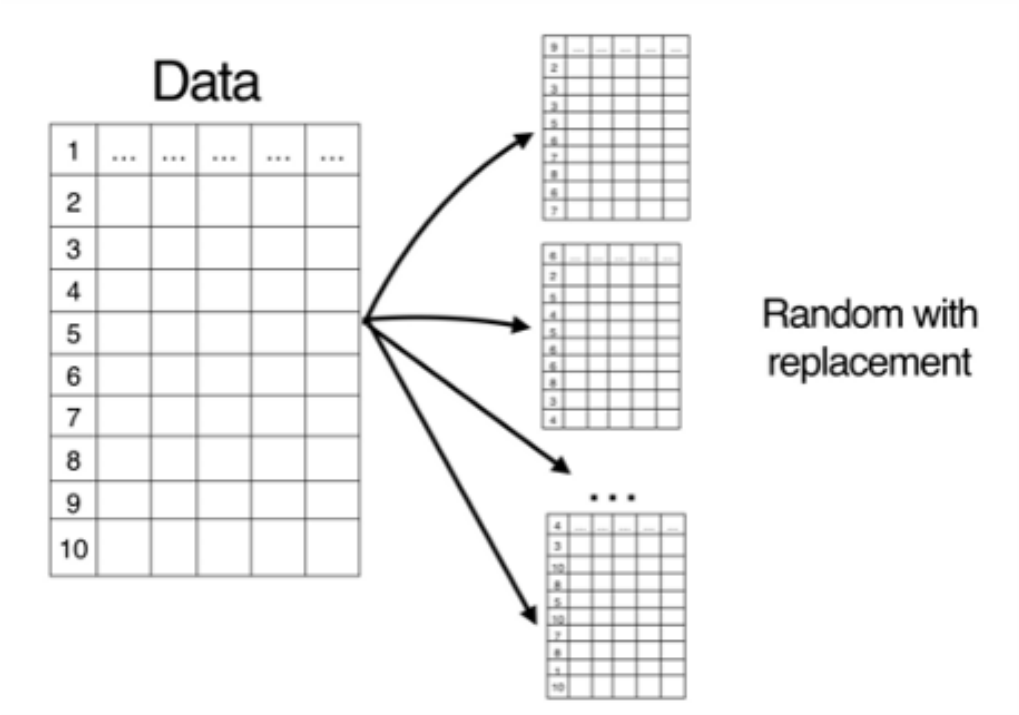
As always, I appreciate any feedback and constructive criticism. My email is wjk68@case.edu.

https://williamkoehrsen.medium.com/random-forest-simple-explanation-377895a60d2d

https://medium.com/@harshdeepsingh_35448/understanding-random-forests-aa0ccecdbbbb

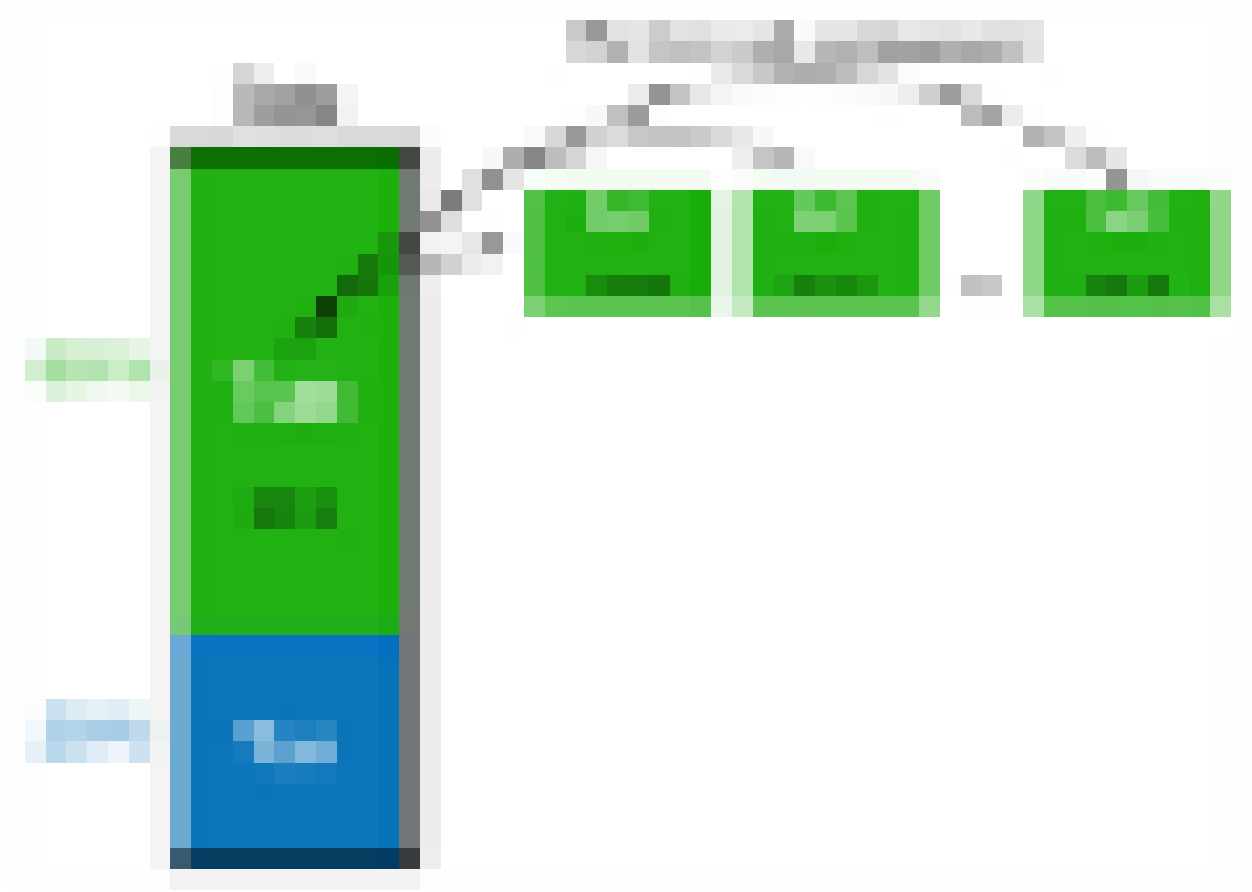
Before we start our discussion on random forests, we first need to understand Bagging. Bagging is a simple and a very powerful ensemble method. It is a general procedure that can be used to reduce our model's variance. A higher variance means that your model is overfitted. Certain algorithms such as decision trees usually suffer from high variance. In another way, decision trees are extremely sensitive to the data on which they have been trained. If the underlying data is changed even a little bit, then the resulting decision tree can be very different and as result our model's predictions will change drastically. Bagging offers a solution to the problem of high variance. It can systematically reduce overfitting by taking an average of several decision trees. Bagging uses bootstrap sampling and finally aggregates the individual models by averaging to get the ultimate predictions. Bootstrap sampling simply means sampling rows at random from the training dataset with replacement.

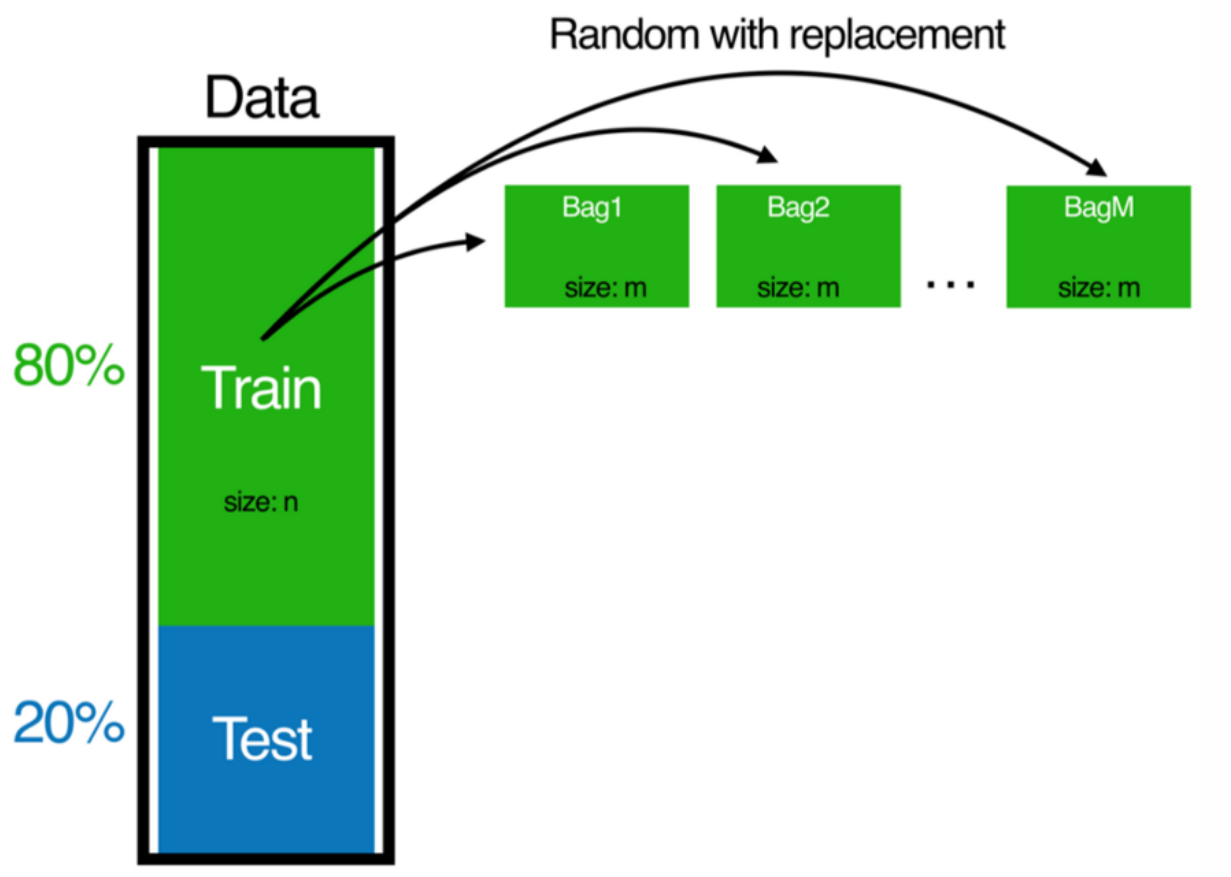




Bootstrap Sampling
With bagging, it is therefore possible that you draw a single training example more than once. This results in a modified version of the training set where some rows are represented multiple times and some are absent. This also lets you create new data, which is similar to the data you started with. By doing this, you can fit many different but similar models.
Specifically, the way bagging works is as follows:

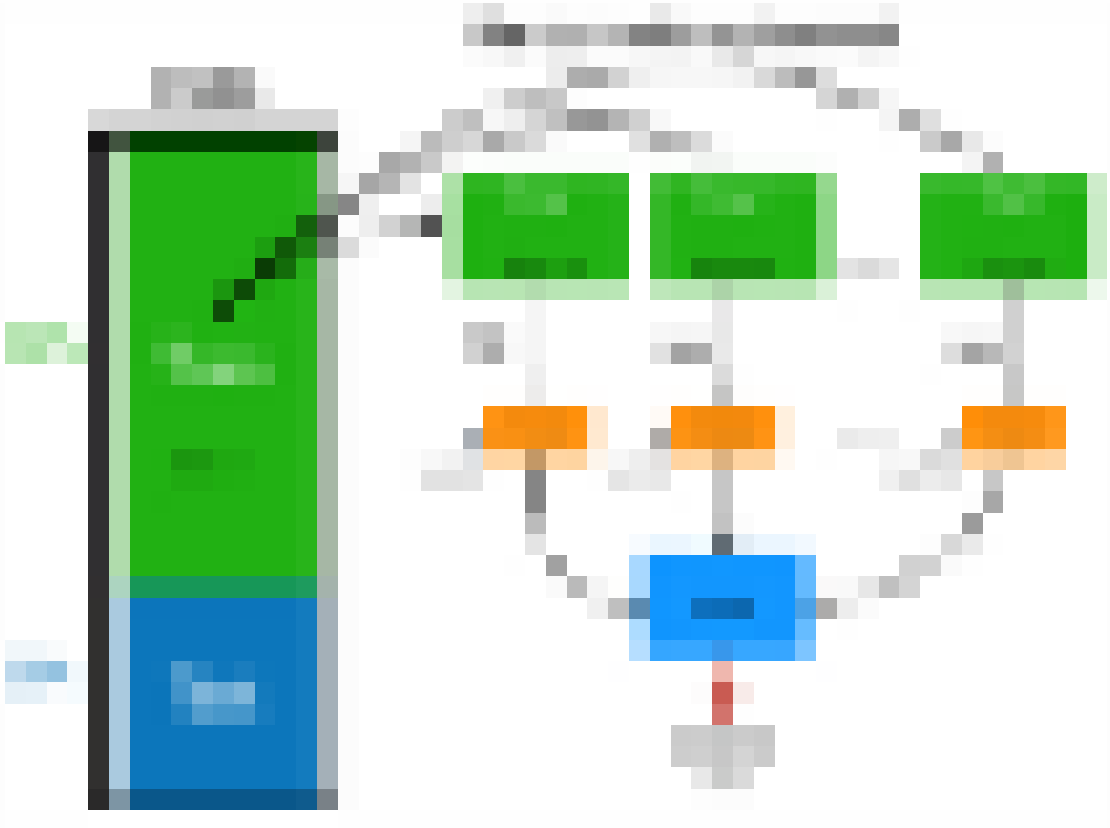
Step 1: You draw B samples with replacement from the original data set where B is a number less than or equal to n, the total number of samples in the training set.

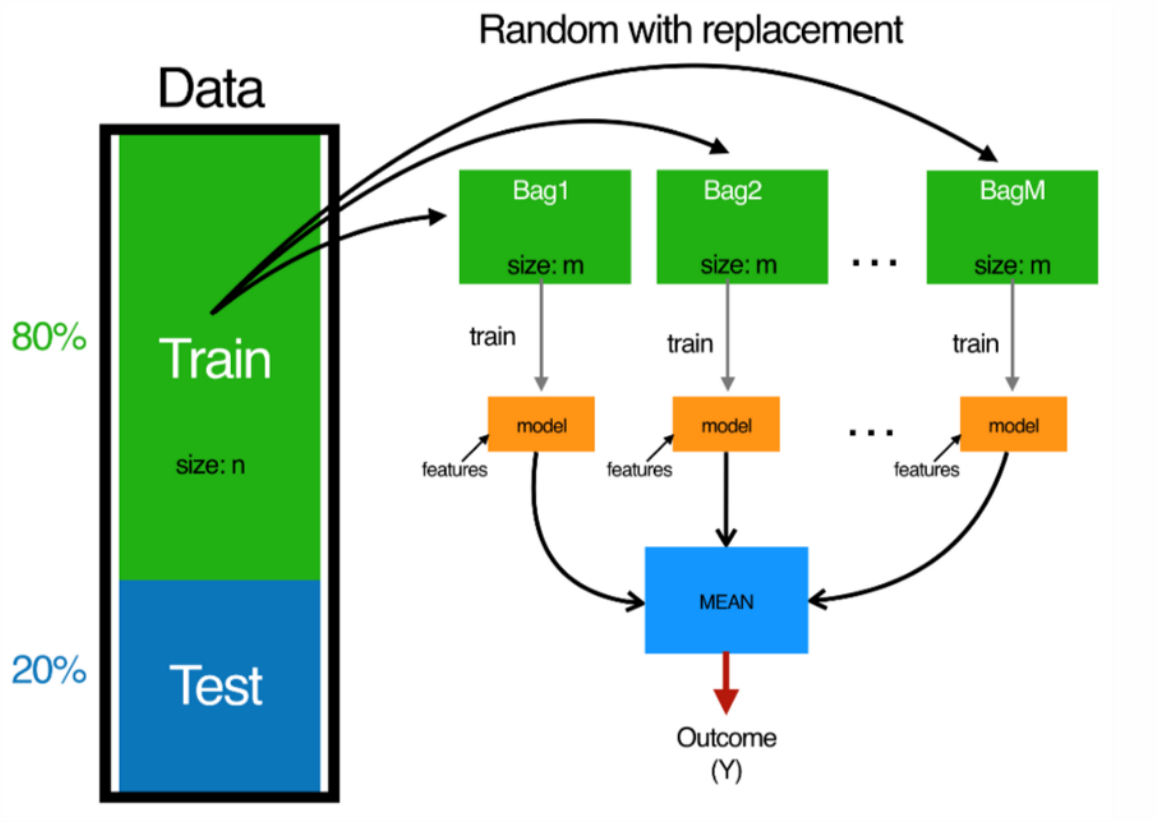




Step 1 : Understanding Bagging

Step 2: Train a decision trees on newly created bootstrapped samples. Repeat the Step1 and Step2 any number of times that you like. Generally, higher the number of trees, the better the model. But remember! Excess number of trees can make a model complicated and ultimately lead to overfitting as your model starts seeing relationships in the data that do not exist in the first place.





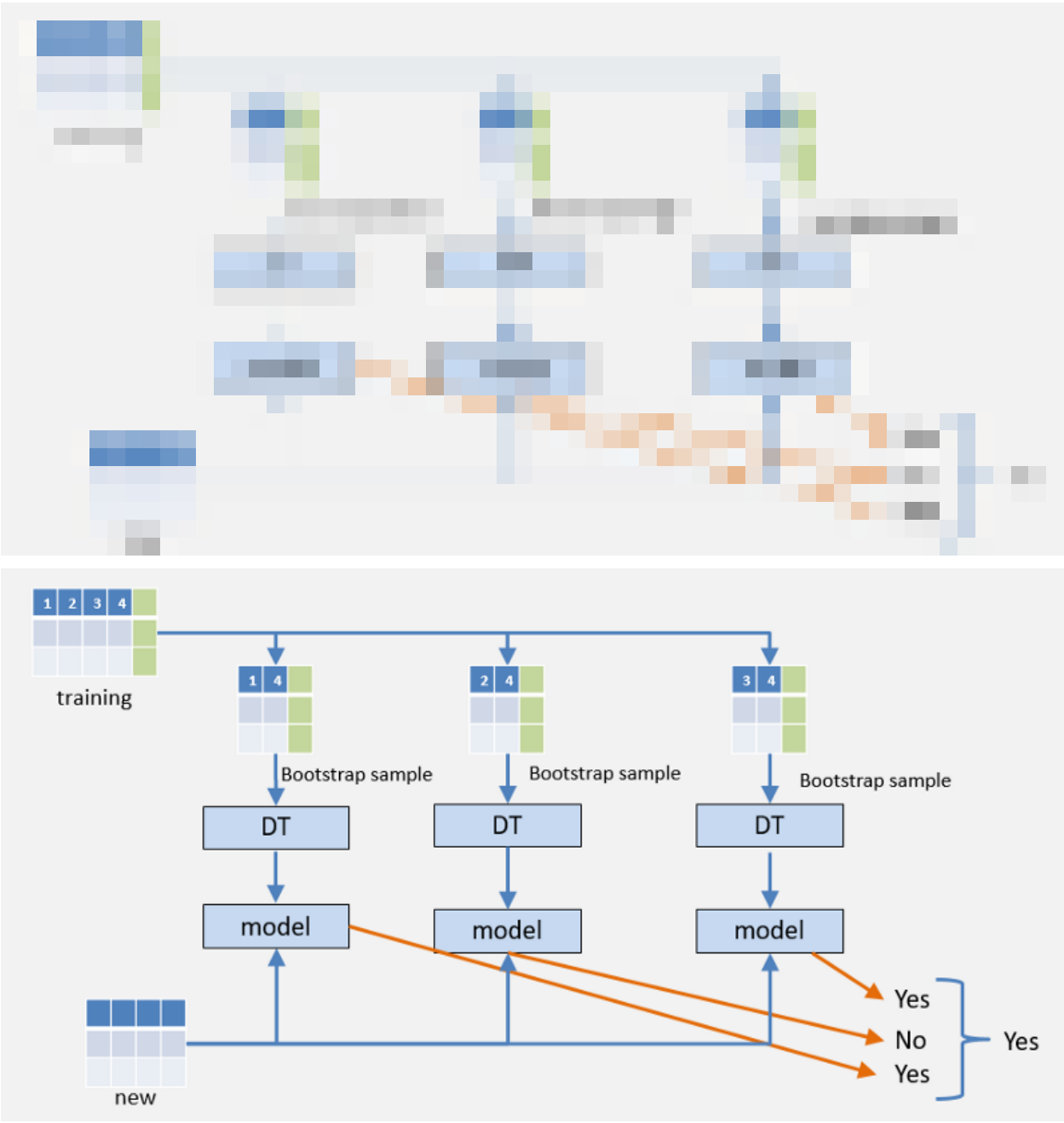
Step 2: Understanding Bagging

To generate a prediction using the bagged trees approach, you have to generate a prediction from each of the decision trees, and then simply average the predictions together to get a final prediction. Bagged or ensemble prediction is the average prediction across the sampled bootstrapped trees. Your bagged trees model works very similar to the council. Usually, when a council needs to take a decision, it simply considers a majority vote. The option that gets more votes (say- option A got 100 votes and option B got 90 votes), is the ultimately the council's final decision. Similarly, in bagging, when you are trying to solve a problem of classification, you are basically taking a majority vote of all your decision trees. And, in case of regression we simply take an average of all our decision tree predictions. The collective knowledge of a diverse set of decision trees typically beats the knowledge of any individual tree. Bagged trees therefore offer better predictive performance.

Random Forests

Random forest is different from the vanilla bagging in just one way. It uses a modified tree learning algorithm that inspects, at each split in the learning process, a random subset of the features. We do so to avoid the correlation between the trees. Suppose that we have a very strong predictor in the data set along with a number of other moderately strong predictors, then in the collection of bagged trees, most or all of our decision trees will use the very strong

predictor for the first split! All bagged trees will look similar. Hence all the predictions from the bagged trees will be highly correlated. Correlated
predictors cannot help in improving the accuracy of prediction. By taking a random subset of features, Random Forests systematically avoids correlation and improves model's performance. The example below illustrates how Random Forest algorithm works.



Understanding Random Forests
Let's look at a case when we are trying to solve a classification problem. As evident from the image above, our training data has four features- Feature1, Feature 2, Feature 3 and Feature 4. Now, each of our bootstrapped sample will be trained on a particular subset of features. For example, Decision Tree 1 will be trained on features 1 and 4 . DT2 will be trained on features 2

and 4, and finally DT3 will be trained on features 3 and 4. We will therefore have 3 different models, each trained on a different subset of features. We will finally feed in our new test data into each of these models, and get a unique prediction. The prediction that gets the maximum number of votes will be the ultimate decision of the random forest algorithm. For example, DT1 and DT3 predicted a positive class for a particular instance of our test data, while DT2 predicted a negative class. Since, the positive class got the majority number of votes(2), our random forest will ultimately classify this instance as positive. Again, I would like to stress on how the Random Forest algorithm uses a random subset of features to train several models, each model seeing only specific subset of the dataset.
Random forest is one of the most widely used ensemble learning algorithms. Why is it so effective? The reason is that by using multiple samples of the original dataset, we reduce the variance of the final model. Remember that the low variance means low overfitting. Overfitting happens when our model tries to explain small variations in the dataset because our dataset is just a small sample of the population of all possible examples of the phenomenon we try to model. If we were unlucky with how our training set was sampled, then it could contain some undesirable (but unavoidable) artifacts: noise, outliers and over- or underrepresented examples. By creating multiple random samples with replacement of our training set, we reduce the effect of these artifacts.

Random Forests in Python

In this section, we will see how to implement the Random Forest algorithm in Python. I have detailed all the necessary steps for anyone who is following along (including a couple of data pre-processing tasks). Here is a link to access my Jupyter Notebook.

HarshSingh16/Machine_Learning

Machine Learning Examples. Contribute to HarshSingh16/Machine_Learning development by creating an account on GitHub.

github.com

Data Pre-Processing

We will be using the bank-full-additional dataset.csv to perform this implementation. The data set can be accessed at my GitHub at the following link. The data is related with direct marketing campaigns of a Portuguese banking institution. The marketing campaigns were based on phone calls. Using features such as age, job, marital, education etc., I have tried to predict whether a given customer has subscribed to a term deposit or not. We will begin by importing the dataset and checking its information.

```
df=pd.read_csv("bank-additional-full.csv")
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 41188 entries, 0 to 41187
Data columns (total 20 columns):
age               41188 non-null int64
job               41188 non-null object
marital           41188 non-null object
education         41188 non-null object
default           41188 non-null object
housing           41188 non-null object
loan              41188 non-null object
contact           41188 non-null object
month             41188 non-null object
day_of_week       41188 non-null object
duration          41188 non-null int64
campaign          41188 non-null int64
pdays             41188 non-null int64
previous          41188 non-null int64
poutcome          41188 non-null object
emp.var.rate      41188 non-null float64
cons.price.idx    41188 non-null float64
cons.conf.idx     41188 non-null float64
euribor3m         41188 non-null float64
y                 41188 non-null object
dtypes: float64(4), int64(5), object(11)
memory usage: 6.3+ MB
```

Also, looking at our dataset, we see that some of our features such as job, education, default, housing, loan, contact, month, day_of_week, poutcome, are categorical in nature. That means we need to transform them using dummy variables so sklearn will be able to understand them. Let's do this in one clean step using pd.get_dummies.

```
df.head()
```

job	marital	education	default	housing	loan	contact	month	day_of_week	duration	campaign	pdays	previous	poutcome
housemaid	married	basic.4y	no	no	no	telephone	may	mon	261	1	999	0	nonexistent
services	married	high.school	unknown	no	no	telephone	may	mon	149	1	999	0	nonexistent
services	married	high.school	no	yes	no	telephone	may	mon	226	1	999	0	nonexistent
admin.	married	basic.6y	no	no	no	telephone	may	mon	151	1	999	0	nonexistent
services	married	high.school	no	no	yes	telephone	may	mon	307	1	999	0	nonexistent

```
#Converting string values to dummies
df_final=pd.get_dummies(df,columns=["day_of_week","job",
                                    "marital","education",
                                    "default","housing",
                                    "loan","contact","month",
                                    "poutcome"],drop_first=True)
```

Converting Categorical Columns to Dummy Variables

We are now ready to split our data into training and test set for ultimately running the Random Forest algorithm. The following lines of code will separate the y-labels from the data frame, and perform the required train-test split. We have kept 30% of the data for testing purposes.

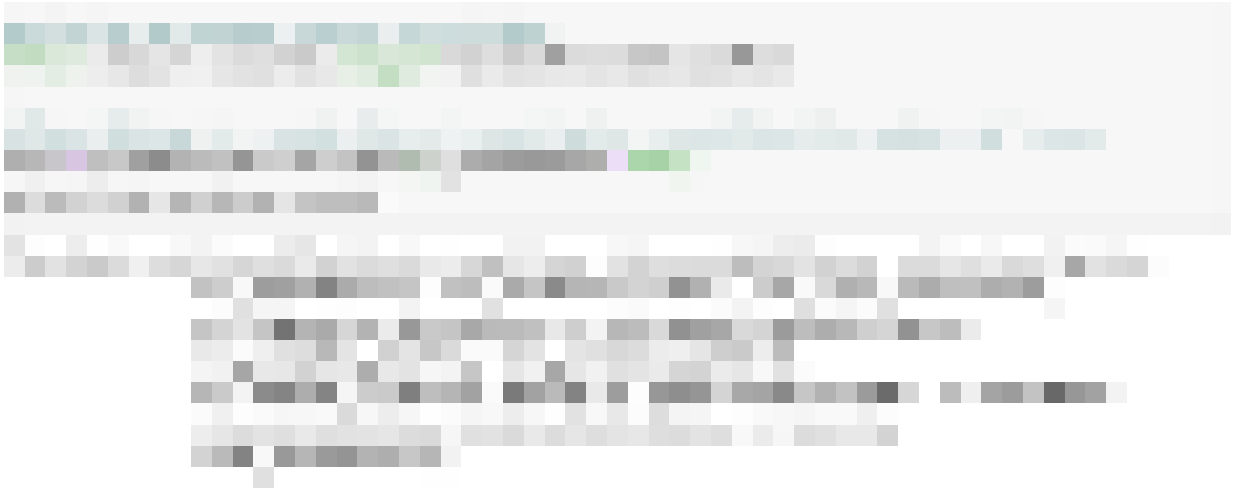
```
import sklearn
from sklearn.model_selection import train_test_split

#Separating the y-column from the rest of the data
df2=df_final.drop("y",axis=1)
X=df2
y=df_final["y"]

#Performing the Split
X_train, X_test, y_train, y_test = train_test_split(X,y , test_size=0.30)
```

Training and Testing our Random Forest Classifier

It is now time to fit our Random Forest Classifier. I have specified the n_estimators as 1000, which implies that our classifier will have a total of 1000 trees. For all the other parameters, I have used the default parameter settings. I highly recommend checking out the Random Forest's sklearn documentation if you want to learn more about the parameters RFC provides.



```
#Fitting a Random Forest Classifier
from sklearn.ensemble import RandomForestClassifier

#Sepcifying the random forest-n_estimator specifies the number of trees
rfc=RandomForestClassifier(n_estimators=1000)
rfc.fit(X_train,y_train)
```

```
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
            max_depth=None, max_features='auto', max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators=1000, n_jobs=None,
            oob_score=False, random_state=None, verbose=0,
            warm_start=False)
```

We will now predict our test data. I have also imported a few more modules to access the confusion matrix, accuracy score and the classification report.

```
#Getting our Predictions
y_hat=rfc.predict(X_test)


#Printing the Accuarcy, Confusion Matrix and the Classification Report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
from sklearn.metrics import accuracy_score


y_actu = pd.Series(y_test, name='Actual')
df_confusion = pd.crosstab(y_actu, y_hat)
acc = accuracy_score(y_test, y_hat, normalize=True)
print('Model Accuracy:%.2f'%acc)
print(df_confusion)
print(classification_report(y_test,y_hat))
```

Here are the results: Our model has an overall accuracy of 0.92. I have also printed out the Classification Report and the Confusion Matrix. The overall precision of our model is 0.66, and the recall is 0.50.

```
Model Accuracy:0.92
col_0       no  yes
Actual
no       10637  348
yes        690  682
              precision    recall  f1-score   support

          no       0.94      0.97      0.95     10985
         yes       0.66      0.50      0.57      1372

   micro avg       0.92      0.92      0.92     12357
   macro avg       0.80      0.73      0.76     12357
weighted avg       0.91      0.92      0.91     12357
```

Final Remarks:
I hope that this article helped you to get a basic understanding of how the gradient boosting algorithm works. Feel free to add me on LinkedIn and I look forward to hearing your comments and feedback.
Citations:
As mentioned above, I have used bank-full-additional dataset.csv for the purpose of the article. This dataset is publicly available for research. The details are described in [Moro et al., 2014].
[Moro et al., 2014] S. Moro, P. Cortez and P. Rita. A Data-Driven Approach to Predict the Success of Bank Telemarketing. Decision Support Systems, Elsevier, 62:22–31, June 2014

Harshdeep Singh

AI & Analytics Consultant | LinkedIn : https://ca.linkedin.com/in/harshanalytics

Research and Resources

Machine Learning Citations

UC Irvine Data Analysis Bootcamp
Instructor: Anthony Taylor
https://bootcamp.ce.uci.edu/data

Random Forest

https://medium.com/@harshdeepsingh_35448/understanding-random-forests-aa0ccecdbbbb

https://williamkoehrsen.medium.com/random-forest-simple-explanation-377895a60d2d

https://medium.com/@harshdeepsingh_35448/understanding-random-forests-aa0ccecdbbbb

https://medium.com/@Synced/how-random-forest-algorithm-works-in-machine-learning-3c0fe15b6674

https://medium.com/capital-one-tech/random-forest-algorithm-for-machine-learning-c4b2c8cc9feb

KNN

https://medium.com/capital-one-tech/k-means-clustering-algorithm-for-machine-learning-d1d7dc5de882

https://medium.com/capital-one-tech/k-nearest-neighbors-knn-algorithm-for-machine-learning-e883219c8f26

Naives Bayes
https://medium.com/capital-one-tech/naives-bayes-classifiers-for-machine-learning-2e548bfbd4a1

Artificial Neural Networks
https://medium.com/capital-one-tech/artificial-neural-networks-for-machine-learning-79c67d0681e9

Earthquake Research Citations

USGS
https://www.usgs.gov/faqs/how-will-my-house-hold-earthquake-can-usgs-send-someone-out-evaluate-my-property?qt-news_science_products=0#qt-news_science_products

NEHRP
https://www.nehrp.gov/

Earthquake Authority
https://www.earthquakeauthority.com/Blog/2020/Benefits-Seismic-Upgrades-Why-Retrofit-Your-Home

WHAT ARE THE CHANCES OF A MAJOR CALIFORNIA EARTHQUAKE?

California has **more than a 99% chance** of having one or more magnitude 6.7 or larger earthquakes within the next 30 years, **according to scientists**.

- For Northern California, the most likely source of such earthquakes is the Hayward-Rodgers Creek Fault (31% in the next 30 years). Such quakes can be deadly, as shown by the 1989 magnitude 6.9 Loma Prieta earthquake. The Loma Prieta quake caused 63 deaths, 3,757 injuries, and about $6 billion in damage.
- For **Southern California**, within the next 30 years the probability is 60% for an earthquake measuring magnitude 6.7. The southern San Andreas fault is much closer to Los Angeles, running at its closest just 25 miles northeast of downtown through the San Gabriel Mountains, and is capable of unleashing a magnitude 8.2 quake throughout all of

> Southern California. The 2019 Ridgecrest earthquake happened on a brand new fault line along the Eastern California Shear Zone.

The new study determined the probabilities that different parts of California—including higher-risk areas like the Bay Area and the city of Los Angeles—will experience earthquake ruptures of various magnitudes. The new statewide probabilities are the result of a model that comprehensively combines information from seismology, earthquake geology, and geodesy (measuring precise locations on the earth's surface).

https://www.stat.go.jp/english/data/jyutaku/pdf/25terms1.pdf

https://www.usgs.gov/faqs/how-will-my-house-hold-earthquake-can-usgs-send-someone-out-evaluate-my-property?qt-news_science_products=0#qt-news_science_products

Published maps will only provide generalized, uninterpreted information about specific areas. Every property consists of a unique combination of geologic and structural factors that must be considered to determine what might happen to a house during an earthquake. Therefore, an individual site study is necessary. Geologic factors include: type of underlying material, depth to bedrock, depth of groundwater, and slope of land. Structural factors include: materials used (wood or masonry) in construction, number of floors, design, and retrofitting present.

FEMA

https://www.fema.gov/emergency-managers/risk-management/building-science/building-codes/earthquakes

Short-Period Building Collapse Performance and Recommendations for Improving Seismic Design

Short-Period Building Collapse Series (ZIP)
FEMA P-2139 | December 2020
Recent analytical studies investigating a wide range of modern seismic-force-resisting systems have predicted collapse rates for short-period buildings that are significantly larger than those observed in earthquakes during the past 50 years. This gap between analytically predicted and historically observed collapse rates in known as the short-period building seismic performance paradox. Additionally, analytically predicted collapse rates for short-period buildings are generally larger than maximum collapse rates used in national model codes and standards to establish seismic design requirements.
The FEMA P-2139 series of reports documents a multi-year investigation of the response behavior and collapse performance of different structural systems to identify causes and develop solutions for the short-period building seismic performance paradox. Studies investigated three structural systems: wood light-frame, special reinforced masonry shear wall, and steel special concentrically braced frame systems. Volume 1 summarizes results, conclusions, and recommendations from the three-system.

FEMA
https://www.fema.gov/emergency-managers/risk-management/building-science/building-codes/earthquakes

Faultline: Seismic Science at the Epicenter
https://www.exploratorium.edu/faultline/damage/building.html
P and S waves may seem like shaky business, but the waves that really do damage are the ones that occur when the energy of the quake reaches the surface of the earth. Rayleigh waves churn over and under like rolling ocean waves; Love waves shake the earth from side to side. Love and Rayleigh waves, named after their discovers, are the ones often responsible for making buildings collapse. Imagine yourself trying to remain standing while the earth was going through such contortions! To make matters worse, these waves can travel at different speeds through different types of rock, bouncing back or changing direction. In places with certain rock compositions, this bouncing will amplify the waves, which will then cause more damage.

https://www.procrewschedule.com/building-materials-types-and-uses-in-construction/

https://oregonstate.edu/instruct/oer/earthquake/13%20chapter%2011_color.html#:~:text=Earthquake%20loads%20are%20both%20vertical,applied%20suddenly%2C%20with%20high%20acceleration.
Wood or Steel?

What is the best material to build on to withstand an earthquake?
Wood and steel have more give than stucco, unreinforced concrete, or masonry, and they are favored **materials for building** in fault zones. Skyscrapers everywhere must be reinforced to **withstand** strong forces from high winds, but in **quake** zones, there are additional considerations.

Faultline: Earthquake Engineering | Exploratorium
https://www.exploratorium.edu › damage › building

More flexible than concrete and other **building materials**, steel **is** more likely to bend instead of break when experiencing seismic force. Because of these attributes, **buildings** constructed primarily from steel require less **earthquake** proofing than those made from other **materials**.Apr 3, 2019

This Material Is Key to Designing Earthquake-Resistant ...

Construction Type/ Materials Used
https://www1.wsrb.com/blog/earthquake-building-classifications
https://oregonstate.edu/instruct/oer/earthquake/13%20chapter%2011_color.html#:~:text=Earthquake%20loads%20are%20both%20vertical,applied%20suddenly%2C%20with%20high%20acceleration.

https://skyciv.com/technical/commonely-used-materials-in-structural-engineering/

Materials

Summary

To better describe steel, concrete and wood. Let us summarize their major characteristics that would highlight each material.

Steel is very strong in both tension and compression and therefore has high compressive and tensile strengths. Steel has an ultimate strength of about 400 to 500 MPa (58 – 72.5 ksi). It is also a ductile material that yields or deflects before failure. Steel stands out for its speed and efficiency in construction. Its relative light weight and ease of construction allows for a workforce about 10 to 20% smaller compared to a similar concrete-based structure being built. Steel structures also have excellent durability.

Concrete is extremely strong in compression and therefore has high compressive strength of about 17MPa to 28MPa. With higher strengths up to or exceeding 70 MPa. Concrete makes it possible to design very robust and durable buildings, and taking advantage of its thermal mass by keeping it inside the building envelope can help regulate interior temperatures. There is also an increasing use of precast concrete in the building industry, which offers advantages in terms of environmental impact, cost and speed of construction.

Wood is resistant to electrical currents, making it an optimal material for electrical insulation. Tensile strength is also one of the main reasons for choosing timber as a building material; its remarkably strong qualities make it the perfect choice for heavy-duty building materials such as structural beams. Wood is much lighter by volume than both concrete and steel, it is easy to work with and very adaptable on site. It is durable, results in less thermal bridging than its counterparts and easily incorporates prefabricated elements. Its structural performance is very high and its compressive strength is similar to that of concrete. Despite all these, timber is used more widely for residential and low-rise structures. It is rarely used as the main material for highrise structures.

These are the most common construction materials used for building. Each material has its own unique set of benefits and drawbacks. Eventually these may be superceded by materials that have very little to no limitations with the technological advancements in the future. Regardless, our current building materials will remain relevant for many decades to come.

Buildings cannot be made **earthquake**-proof, only **earthquake**-resistant. Because the majority of **old** houses are built with wood frames, a relatively flexible **construction** method, they can sway in an **earthquake** like a palm tree in a stiff breeze.Apr 18, 2013

Seismic Upgrades for Old Houses - Old House Journal Magazine

https://www.oldhouseonline.com/repairs-and-how-to/seismic-upgrades-for-old-houses/

To be **earthquake proof**, buildings, **structures** and their foundations need to be built to be **resistant** to sideways loads. ... They must be strong enough to take the loads. They must be tied in to any framing, and reinforced to take load in their weakest direction.

Earthquake Proof and Resistant Building Structures | REIDsteel

Earthquake Engineering Began in 1948 in California
https://theconstructor.org/earthquake/performance-buildings-types-earthquake/2224/

- Performance of Various Types of Buildings during Earthquake
  - 1. Mud and Adobe Houses during Earthquakes
  - **2. Masonry Buildings during Earthquakes*****
  - 3. Reinforced Masonry Buildings during Earthquakes
  - 4. Brick-Reinforced Concrete Frame Buildings during Earthquake
  - **5. Wooden Buildings during Earthquakes*****
  - 6. Reinforced Concrete Buildings during Earthquakes
  - **7. Steel Skeleton Building Performance during Earthquakes*****
  - **8. Steel and Reinforced Concrete Composite Buildings during Earthquakes*****

How do earthquake underwriters think about risk?
Source: Daniel Wallace, President Modus Underwriting Earthquake Insurance **https://www.modusunderwriting.com)**
Floors
Waveform (Japan v CA)

Seismic Wave Motions—4 waves animated- Incorporated Research ...

Seismic Wave Motions—4 waves animated

- Body **Waves** - Primary (P) & Secondary (S) **Waves**.
- Surface **Waves** - Rayleigh & Love **Waves**.

Rayleigh-wave Motion

Rayleigh Waves—surface waves that move in an elliptical motion, producing both a vertical and horizontal component of motion in the direction of wave propagation.

Love-wave Motion

Love Waves—surface waves that move parallel to the Earth's surface and perpendicular to the direction of wave propagation.

Seismic Wave Motions—4 waves animated- Incorporated Research ...

Seismic waves can be classified into two basic types: body **waves** which travel through the Earth and surface **waves**, which travel along the Earth's surface. Those **waves** that are the **most** destructive are the surface **waves** which generally have the **strongest** vibration.

Earthquake | Seismic waves as body waves and surface waves

Earthquake Insurance Underwriting

Source: Daniel Wallace, President Modus Underwriting Earthquake Insurance
https://www.modusunderwriting.com/

Two different types of earthquake to be concerned about in the industry
Strike Slip v Subduction (Love v. Raleigh Waves)

Japan big, long waves (high rises go down)
California can go over the big waves (sharp choppy waves) (20 seconds to a minute) (side by side) (1-3 story buildings) 6.7-7.something shorter and more intense) (High rises okay in SF and 2 story is fucked)

Renaissance Earthquake, IRIS non profit. Help us thinking about high frequency waves
2-4 story buildings are most vulnerable in California.
94 89

Color tagging structure
Red tagged (knocked down and completely rebuild)
Yellow tagged
Green tagged

Life Safety Construction Code: California has a bad construction code (accepts 1 collapse out of 10 new construction)

Japan is better they just had a major 9.0

FLOORS (NUMBER OF STORIES) 2-4 are the worst

Prop 93 / CONDOS are most vulnerable

Height of Structure / Stories or Floors / Residential (best to worst)
Residential 15-20 stories
Residential 11-14 stories
Residential 6-10 stories
Residential 1 story
Residential 2-5 stories

Source: Daniel Wallace, President Modus Underwriting Earthquake Insurance
https://www.modusunderwriting.com
1994 construction- important transition year

Not required is main reason don't buy it

On a high rise rollers would add 50 mil in California

Source: https://www.modusunderwriting.com/

---

https://en.wikipedia.org/wiki/Color-tagged_structure

Color-tagged structure

From Wikipedia, the free encyclopedia
Jump to navigation
Jump to search

A **color-tagged structure** is a structure in the United States which has been classified by a color to represent the severity of damage or the overall condition of the building. The exact definition for each color may be different at local levels.[1]

A "red-tagged" structure has been severely damaged to the degree that the structure is too dangerous to inhabit. Similarly, a structure is "yellow-tagged" if it has been moderately damaged to the degree that its habitability is limited (only during the day, for example). A "green-tagged" structure may mean the building is either undamaged or has suffered slight damage, although differences exist at local levels when to use a green tag.

Tagging is performed by government building officials, or, occasionally during disasters, by engineers deputized by the building official. Natural disasters such as earthquakes, floods and mudslides are among the most common causes of a building being red-, yellow- or green-tagged. Usually, after such incidents, the local government body responsible for enforcing the building safety code examines the affected structures and tags them as appropriate.

In some areas of the United States, buildings are marked with a rectangular sign that is red with a white border and a white "X". Such signs provide the same information as "red-tagging" a building. Tagging structures in these ways can warn firefighters and others about hazardous buildings before the buildings are entered.