Detecting Prime Numbers Using Machine Learning Techniques

# Brandon Sams Winter 2020 https://brandonsams.github.io

# Which Domain?

What domain is this data going to come from? Please list 10 references (with a brief annotation) to use to make sense of what you’re doing with these data.

This data will come from a purely mathematical domain, although some applications do exist in the field of cryptography. The following citations are examples of the work that has been done in this problem space already.

Egri, László and Thomas R. Schultz. “A Compositional Neural-Network Solution to Prime-Number Testing.” Proceedings of the Annual Meeting of the Cognitive Science Society, vol. 28, 2006, <https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.560.802&rep=rep1&type=pdf>.

*Describes an implementation of knowledge-based cascade-correlation (KBCC) to detect if a number is prime or composite. Implementation is based on the traditional method of testing factors of n up to √n, with optimizations coming from neural networks.*

Keltner, Luke. LukeKeltner/Primes-and-ML. 2018. 2018. GitHub, <https://github.com/LukeKeltner/Primes-and-ML>.

*Uses a random forest classifier to train a machine learning model that can identify prime numbers. Encodes the digits of a number as fields in a dataframe. Performs tests on odds and evens in the dataset.*

Machine Learning Classification of Prime Decompositions. http://marcharper.codes/2016-08-08/Machine+Learning+Classification+of+Prime+Decompositions.html. Accessed 6 Dec. 2020.

*Contains sample code that performs such techniques as Principal Component Analysis, K-Nearest Neighbors, Neural Networks, and Confusion Matrices.*

Michael N. Vrahatis. A First Study of the Neural Network Approach to the RSA Cryptosystem. 2002, doi:10.13140/2.1.2495.3284.

*The RSA Cryptosystem relies on keys which are products of two very large primes, making them difficult to factor. Some of the techniques used to factor these numbers may also be useful in identifying primes.*

Posted by Vincent Granville on April 6, 2017 at 8:00am, and View Blog. Factoring Massive Numbers with Machine Learning Techniques. https://www.datasciencecentral.com/profiles/blogs/factoring-massive-numbers-a-new-machine-learning-approach. Accessed 6 Dec. 2020.

*Describes a related problem of factoring large numbers. Knowing if a large number is a prime can help to determine if it can be a prime factor of a larger number.*

Posted by Vincent Granville on September 20, 2017 at 10:00am, and View Blog. Data Science Method to Discover Large Prime Numbers. https://www.datasciencecentral.com/profiles/blogs/data-science-method-to-discover-large-prime-numbers. Accessed 6 Dec. 2020.

*Examines patterns of numbers which appear to have larger than normal amounts of primes distributed throughout them. Shows techniques for determining if a number is a “probable” prime.*

“Prediction - Could a Neural Network Detect Primes?” Artificial Intelligence Stack Exchange, https://ai.stackexchange.com/questions/3389/could-a-neural-network-detect-primes. Accessed 6 Dec. 2020.

*In an answer to the question of if a neural network could detect primes, a link to the Egri, László and Thomas R. Schultz paper is mentioned. This post also contains some analysis of the sub-questions contained in the original question, such as:*

1. *Can it by simply memorizing the primes over the range of integers?*
2. *Can it by learning to factor and apply the definition of a prime?*
3. *Can it by learning a known algorithm?*
4. *Can it by developing a novel algorithm of its own during training?*

Stekel, Avigail, et al. “Goldbach’s Function Approximation Using Deep Learning.” ArXiv:1803.09237 [Cs, Stat], Mar. 2018. arXiv.org, <http://arxiv.org/abs/1803.09237>.

*It is helpful to know how Deep Learning is being used in related mathematical questions to know how it is likely best to use similar techniques in detecting prime numbers. Goldbach’s conjecture is an example of this type of research, which is benefitting from Deep Learning.*

Supe Arun. “Careless Learner: A Heuristic Algorithm for Factoring Larger Numbers into Their Composite Primes Using Machine Learning.” Careless Learner, 22 Jan. 2015, <http://carelesslearner.blogspot.com/2015/01/a-heuristic-algorithm-for-factoring.html>.

*Uses Logistic Regression and Random Forest techniques to factor large numbers into primes. Sample code is given that trains and tests the first 50 million prime numbers.*

“The Mysterious Primes.” Machine Learning Medium, 1 Nov. 2017, <https://machinelearningmedium.com/2017/11/01/prime-numbers/>.

*Contains some examples of proofs that can help to serve as heuristics to limit the search space a bit when testing larger numbers. For example, even numbers greater than 2 are never prime. So no need to test them. (And no need to train a model on them?)*

# Which Data?

What is the dataset you’ll be examining? Please provide a codebook if there is one or a link to the dataset as well as a detailed description.

I will be using a list of prime numbers to train the model. There are downloadable sets of these numbers available at links such as this one: <https://primes.utm.edu/lists/small/millions/>

However, it is fairly straightforward to generate a complete list of numbers using python code such as this, and looping through the integers. The data can then be saved to a csv, for example.

import math

def isPrime(x):

if x < 2:

return(False)

if x == 2:

return(True)

for i in range(2, int(math.ceil(math.sqrt(x)) + 1)):

if x % i == 0:

return(False)

return(True)

# Research Questions? Benefits? Why analyze these data?

How are you proposing to analyze this dataset? This is about your approach. Here, you’ll be proposing your research questions as well as justifications for why you’d offer these data in this way.

How plausible is it to detect if a number is prime or not, based on machine learning techniques. How can a search space be limited, in order to make the algorithm faster? Can a neural network rediscover any patterns or heuristics that are already used to detect primes? What is the best way to encode numbers that will serve as the input? Are certain encodings (such as binary) better than categorical?

This research aims to determine numbers which are probably primes, using machine learning techniques. Prime numbers are notably important to fields such as number theory, but there is not a known easy way to determine if a number is a prime or not. (Although some tests exist, they are not known for being easy tests to run). As an additional benefit, it would be fascinating to know what numbers are surprisingly prime, even if the model does not predict them to be so.

This data has applications in the field of cryptography. There are plenty of secret keys that rely on numbers being difficult to factor into smaller sections, and being able to quickly identify likely candidates for factors could improve the speed at which keys could be factored.

# What Method?

What methods will you be using? What will those methods provide in terms of analysis? How is this useful?

I anticipate that it would be good to convert a number to its binary representation, and feed it into a neural network of some configuration. A single bit at the end would be an indicator of whether or not it is a prime number. The general configuration plan is to have lots of perceptrons funnel down to a smaller number as the network proceeds.

# Potential Issues?

What challenges do you anticipate having? What could cause this project to go off schedule?

I anticipate that I will need to handle imbalanced classes. Most numbers are not prime, and as n gets larger, the probability that a given number is prime decreases. Some resampling will likely need to take place. I also anticipate that the number of perceptrons in the input layer will need to be a large number, and the network will lose its predictiveness for larger numbers that do not “fit” into the network as trained.

I am not confident that this technique will work at all. I hope it does, of course. And neural networks have shown plenty of promise in finding hidden patterns that are not always obvious to humans, but if there is not a pattern in the first place, then these efforts may be in vain. However, based on the literature that exists already, I am cautiously optimistic.

# Concluding Remarks

Tie it all together. Think of this section as your final report’s abstract.

This research aims to create a neural network that is able to detect if a given number is a prime number or not. Numbers are converted into binary, as each bit in the number serves as input for the neural network. The training data is balanced to include just as many prime numbers as composite numbers, to prevent the network from correctly learning that any given number is not likely to be a prime. This project aims to see if this technique is even possible, and if it is, where the limitations lie. Some positive results have been shown in the field such as those from Egri and Shultz, but this paper aims to build on their research.