Plan:

1. Octave programming language
    a. What is Octave
    b. Why Octave
    c. Setting up programming environment
2. Multiple linear regression
    a. Gradient descent with multiple variables
    b. Feature scaling
    c. Learning rate
    d. Polynomial regression
3. Computing parameters analytically
    a. Normal equation
    b. Comparing normal equation and gradient descent
4. Submitting programming assignments and tests

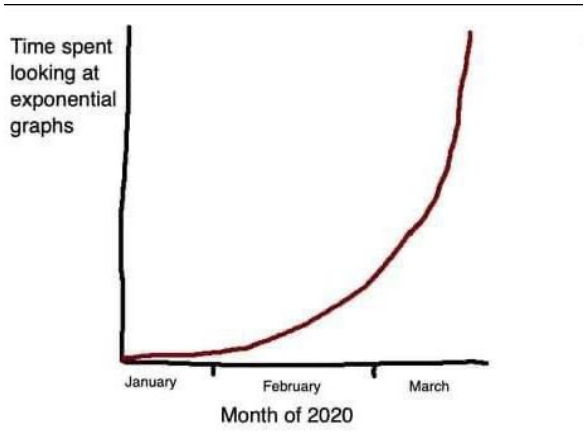
Questions:

1. What's the difference between Octave and Matlab?
2. Why it's important to set proper learning rate value and how can it affect our model?
3. What is feature scaling and when do we use it?
4. What's the difference between normal equation and gradient descent?

Glossary:

- Gradient descent - iterative algorithm used to minimize some function by moving in the direction of negative gradient which represents steepest descent - *Gradient descent is used to optimize model parameters*
- Normal equation - one-step algorithm which literally the value of this formula: $\theta = (X^T X)^{-1} X^T y$ - *We can use normal equation to optimize model parameters only if given matrix is not singular.*

P.S. There is a few memes as was asked in criteria.



Reflection:
Part 1.

What have I learned this week?

What was particularly useful and how can I use it in my studies and work?

What was redundant? (I already knew that)

What difficulties have I experienced?

What would I change in the content/quality of the week?

This week was even beneficial than the first one in terms of learning new stuff. First I've learned about a programming language I've never heard before - Octave. This is a high-level programming language and Andrew Ng - teacher of the course - highly recommends as an easy way to create models fast and effective. Also, Octave is very effective in prototyping, because you don't have to think about some low-level things (such as memory management in C/C++ for example). After learning Octave basics, we focused on multivariate linear regression, which is pretty much the same as linear regression with one variable, but in the case of multiple variables, we presenting datasets as matrixes instead of vectors. Then we learned about a feature scaling - preprocessing stage where you change your features order for it to have the same impact on the model as other features. For example, if feature A is in the interval of [-10000; 10000], features B and C in [-10; 10] we most likely want to change the order of feature A by dividing all values of A by 1000 and after we did that, our model (which looks like this M1A + M2B + M3C) is normalized and all features have a same-order impact on the model. I've also learned about a learning rate of gradient descent which defines the 'size' of every iteration of the algorithm. After that teacher talked about normal equation, vectorization, and at the end of the week, we had an assignment.

I think pretty much everything I learned this week is useful for me, because, as I mentioned in the last week's reflection, that's basics for creating regression models and I can't really remember anything redundant. So far I had no difficulties and everything feels pretty natural and easy to understand. This week I had everything - programming language, math, model-creating, and even practical assignment, so I wouldn't change anything.

What about Nikita's performance - even though I had some connection issues - I understood what he was trying to say. He also brought code samples which were helpful. But I didn't really saw the impact of the online course - he said that he restructured his project and tried to show all the improvements but because I didn't know what was there before I can't see the difference. Maybe he should have pointed at the exact function he modified. Or maybe he did that and I just missed it because of the bad internet quality. We will never find out...