1 Project Summary:

Electricity retail prices are not fixed in the US; at some points in time, they are cheaper than at other times. We want to engineer any home appliance to autonomously turn on when the retail price of electricity is cheapest. This requires us to make a model that accurately predicts prices or that accurately classifies a single time, out of n times, as being the time when the retail rate is cheapest. We also need a pipeline that automates a progression:

- User tells an Alexa to turn on an appliance with some parameters I (e.g. what appliance mode to use and with what urgency (now or before time t)).
- Alexa sends I to a sub-pipeline that:
  - Decides the time t to run the appliance if not immediately (the model)
  - Sends time (t, I) to an agent that, in turn, instructs an embedded system (in some appliance) to turn on at t using state represented by I.

Concretely, this sub-pipeline is built using several technologies offered by Amazon AWS. In particular, these include: AWS Lambda, AWS DynamoDB, AWS Sagemaker, AWS Cloudwatch and AWS Pipeline.

Remark 1.1. Many of the modules to perform these tasks have been made by prior students working with Professor Sowers; they have not been tested comprehensively, documented or tested jointly with other modules (ie, in an integrated manner). These modules are not scalable: they only allow one user to turn on one appliance; ideally, we want a user to turn on any number of appliances belonging to her, and we want any number of users to be capable of doing this. Some of them also use bad coding practice (e.g. pyramid of doom in javascript).

In this regard, my senior thesis is about software development.

Remark 1.2. There are 5 active domains in which model development for EPF (electricity price forecasting takes place). These include artificial intelligience, time series analysis, economics-based models, game-theoretic models and stochastic models. A graduate student working on this project has explored AI models.

I know little about this project's model development and need to learn more.

In this regard, my senior thesis is also about model development.

2 Milestones

These milestones are listed in reverse chronological order from 2019 to November 2018:

- December 2019

  - Finish a 3rd Draft of the thesis and present it.

- November 2019
  - Finish a 1st Draft of the thesis
  - Finish a 2nd Draft of the thesis
- October 2019
  - Buffer
- September 2019
  - Learn and try 10 different models in domain C.
- August 2019
  - Learn and try 10 different models in domain C.
- July 2019
  - Buffer
- June 2019
  - Learn and try 5 different models in domain B.
  - Run fully integrated tests, having included the UI developed in May.
- May 2019
  - Learn and try 5 different models in domain B.
  - Develop or extend a UI that allows a user to add an appliance.
  - Run fully integrated tests on the existing infrastructure.
- April 2019
  - Learn and try 5 different models in domain A.
- March 2019
  - Build a pipeline incorporating AWS SageMaker
- February 2019
  - Finish developing tests for all components.
  - Finish documentation for all components.
  - Finish developing a pipeline \{Sagemaker}
  - Have a technical report of the differences between prediction models based in machine learning and prediction models based in time series analysis.
- January 2019
  - Buffer
- December 2018

  - Document half of the repository code.
  - Devise a local workflow for the development of the Alexa UI.
  - Refactor and test a function called RBS_Lambda.
  - Have a technical report of the prediction models tried by the group up until now.

- November 2018
  - Devise a local workflow for AWS lambda.. At present, we use AWS's interface for most of our programming. We need a local workflow to streamline testing and development.
  - Refactor and test a function called RBS_argmin.
  - Refactor and test a function called RBS_server.
  - Refactor and test a function called RBS_Alexa

3 Current Progress

In what follows, I use the word "we" intentionally: I am working with another student who is working on this project as an Independent Study project.

- Previous students working with Professor Sowers devised a way to turn on a dishwasher remotely. The infrastructure to turn it on began to rot after these students left, because the infrastructure was not maintained. We revived the infrastructure and can remotely turn on the dishwasher at will.
- We scaled the codebase to allow a single user to turn on multiple appliances remotely. This has not been tested at scale, however.
- We began documenting the codebase. This is a software project; for the sanity of its developers and future contributors, it needs clear documentation.
- At present, I am working to develop a local workflow for this project. Ideally, instead of programming in the AWS console, we should be able to develop locally on our personal machines and export whatever we write to AWS. This will allow parallel development between my partner and me; we can also use version control with this function, thereafter.