CS181-S23

Assignment #1

Due: 11:59pm ET, February 9, 2023

Homework 1: Regression

Introduction

This homework is on different three different forms of regression: kernelized regression, nearest neighbors regression, and linear regression. We will discuss implementation and examine their tradeoffs by implementing them on the same dataset, which consists of temperature over the past 800,000 years taken from ice core samples.

The folder `data` contains the data you will use for this problem. There are two files:

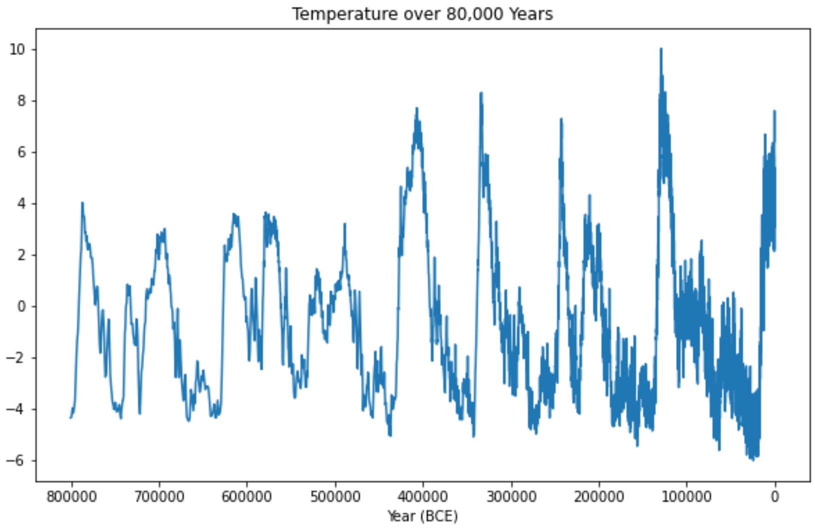
- `earth_temperature_sampled_train.csv`
- `earth_temperature_sampled_test.csv`

Each has two columns. The first column is the age of the ice core sample. For our purposes we can think of this column as the calendar year BC. The second column is the approximate difference in yearly temperature (K) from the mean over a 5000 year time window starting at the given age. The temperatures were retrieved from ice cores in Antarctica (Jouzel et al. 2007)[1].

The following is a snippet of the data file:

```
# Age, Temperature
3.999460000000000000e+05,5.090439218398755017e+00
4.099800000000000000e+05,6.150439218398755514e+00
```

Due to the large magnitude of the years, we will work in terms of thousands of years BCE in Problems 1-3. This is taken care of for you in the provided notebook.



[1] Retrieved from https://www.ncei.noaa.gov/pub/data/paleo/icecore/antarctica/epica_domec/edc3deuttemp2007.txt
Jouzel, J., Masson-Delmotte, V., Cattani, O., Dreyfus, G., Falourd, S., Hoffmann, G., … Wolff, E. W. (2007). Orbital and Millennial Antarctic Climate Variability over the Past 800,000 Years. *Science, 317*(5839), 793–796. doi:10.1126/science.1141038

If you find that you are having trouble with the first couple problems, we recommend going over the fundamentals of linear algebra and matrix calculus (see links on website). The relevant parts of the cs181-textbook notes are Sections 2.1 - 2.7. We strongly recommend reading the textbook before beginning the homework.

We also encourage you to first read the Bishop textbook, particularly: Section 2.3 (Properties of Gaussian Distributions), Section 3.1 (Linear Basis Regression), and Section 3.3 (Bayesian Linear Regression). (Note that our notation is slightly different but the underlying mathematics remains the same!).

Please type your solutions after the corresponding problems using this LaTeX template, and start each problem on a new page. You may find the following introductory resources on LaTeX useful: LaTeX Basics and LaTeX tutorial with exercises in Overleaf

Homeworks will be submitted through Gradescope. You will be added to the course Gradescope once you join the course Canvas page. If you haven't received an invitation, contact the course staff through Ed.

Please submit the writeup PDF to the Gradescope assignment 'HW1'. Remember to assign pages for each question.

Please submit your LaTeXfile and code files to the Gradescope assignment 'HW1 - Supplemental'. Your files should be named in the same way as we provide them in the repository, e.g. `hw1.pdf`, etc.

Problem 1 (Optimizing a Kernel)

Kernel-based regression techniques are similar to nearest-neighbor regressors: rather than fit a parametric model, they predict values for new data points by interpolating values from existing points in the training set. In this problem, we will consider a kernel-based regressor of the form:

$$f_\tau(x^*) = \frac{\sum_n K_\tau(x_n, x^*) y_n}{\sum_n K_\tau(x_n, x^*)}$$

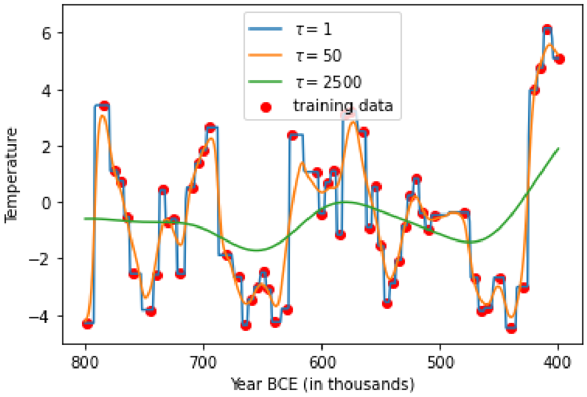
where $\{(x_n, y_n)\}_{n=1}^N$ are the training data points, and $K_\tau(x, x')$ is a kernel function that defines the similarity between two inputs x and x'. A popular choice of kernel is a function that decays as the distance between the two points increases, such as

$$K_\tau(x, x') = \exp\left\{-\frac{(x - x')^2}{\tau}\right\}$$

where τ represents the square of the lengthscale (a scalar value that dictates how quickly the kernel decays). In this problem, we will consider optimizing what that (squared) lengthscale should be. *Make sure to include all required plots in your PDF.*

1. Let's first take a look at the behavior of the fitted model for different values of τ. Plot your model for years in the range $800,000$ BC to $400,000$ BC at 1000 year intervals for the following three values of τ: $1, 50, 2500$. Since we're working in terms of thousands of years, this means you should plot $(x, f_\tau(x))$ for $x = 400, 401, \ldots, 800$. The plotting has been set up for you in the notebook already.

   Include your plot in your solution PDF.



   In no more than 5 sentences, describe what happens in each of the three cases. How well do the models interpolate? If you were to choose one of these models to use for predicting the temperature at some year in this range, which would you use?

   In the case for $\tau = 1$, the model over-fitted to the training data, predicting the exact value of the training data for each corresponding domain value in the training set. This could lead to the model not being very generalizable. In the case of $\tau = 50$, the regression predicted close to the training data with some variance. It was a very smooth model which seemed to better interpolate taking into account more surrounding points. When $\tau = 2500$, the regression under-fitted creating a more simple model which is less generalizable. I would choose the model with $\tau = 50$ in order to predict the temperature at some year in this range since the model is general enough to follow the trend of the data without over-fitting to the training data.

Problem 1 (cont.)

2. Say we instead wanted to empirically evaluate which value of τ to choose. One option is to evaluate the mean squared error (MSE) for f_τ on the training set and simply choose the value of τ that gives the lowest loss. Why is this a bad idea?
   If we were choose τ based on MSE, the model will be very over-fitted. If τ is very small, for values closer to the current point, $K_\tau(x, x')$ will be proportionally larger than for values farther from the current point. Ideally, we would want τ to be 0 because that would lower the train MSE to zero, but then the model will be overfitted to the training data and that would make the model not generalizable.

   Hint: consider what value of τ would be optimal, for τ ranging in $(0, \infty)$. We can consider $f_\tau(x^*)$ as a weighted average of the training responses, where the weights are proportional to the distance to x^*, and the distance is computed via the kernel. What happens to $K_\tau(x, x')$ as τ becomes very small, when $x = x'$? What about when $x \neq x'$?

3. We will evaluate the models by computing their MSE on the test set.

   Let $\{(x'_m, y'_m)\}_{m=1}^{M}$ denote the test set. Write down the form of the MSE of f_τ over the test set as a function of the training set and test set. Your answer may include $\{(x'_m, y'_m)\}_{m=1}^{M}$, $\{(x_n, y_n)\}_{n=1}^{N}$, and K_τ, but not f_τ.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y})^2$$

$$MSE = \frac{1}{M}\sum_{i=1}^{M}(y'_i - \frac{\sum_n K_\tau(x_n, x'_i) y_n}{\sum_n K_\tau(x_n, x'_i)})^2$$

4. We now compute the MSE on the provided training set. Write Python code to compute the MSE with respect to the same lengthscales as in Part 1. Which model yields the lowest test set MSE? Is this consistent with what you observed in Part 1?

τ	1	50	2500
Test MSE	1.9473	1.858	8.334
Train MSE	1.0520e-15	0.5785	5.006

   This is consistent with what I observed in part 1. When $\tau = 2500$, the test and train MSE is the highest as the regression produced a model that was too simple and did not predict well. The model with $\tau = 1$ did surprisingly well with a test MSE only slightly higher than the model with $\tau = 50$ as it overfitted on the training data, leading to the train MSE of very close to zero. As expected the model with $\tau = 50$ did the best on the test data, with the lowest test MSE, because it wasn't too simple and did not overfit.

Problem 1 (cont.)

5. Say you would like to send your friend your kernelized regressor, so that they can reproduce the same exact predictions as you. You of course will tell them the value of τ you selected, but what other information would they need, assuming they don't currently have any of your data or code? If our training set has size N, how does this amount of information grow as a function of N—that is, what is the space complexity of storing our model?
   What is the time complexity of your implementation, when computing your model on a new datapoint? I would also need to send my friend my training dataset as long as my value of τ. Thus, the space complexity of our model is $O(N)$, thus the amount of information grows **linearly** with N. In order to calculate $K_\tau(x, x')$, we need to perform operations that take constant time such as division, subtraction, squaring, and exponential. Thus, the kernel function takes constant time. Since we need to perform this constant time operation for all the N elements in the dataset, along with a constant time multiplication of y_n, the time complexity of this regression is $O(N)$ also **linear**.

Problem 2 (Kernels and kNN)

Now, let us compare the kernel-based approach to an approach based on nearest-neighbors. Recall that kNN uses a predictor of the form

$$f(x^*) = \frac{1}{k}\sum_n y_n \mathbb{I}(x_n \texttt{ is one of k-closest to } x^*)$$

where $\mathbb{I}$ is an indicator variable. For this problem, you will use the **same dataset as in Problem 1**.

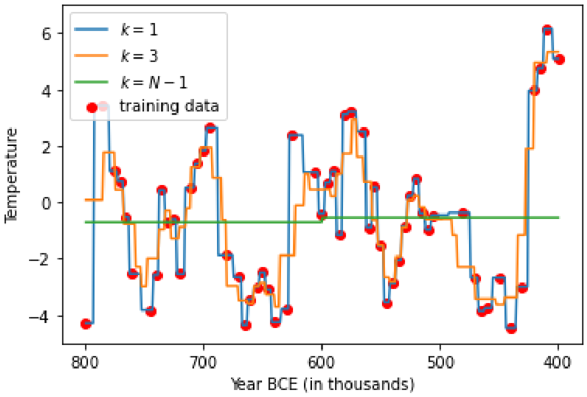
Note that our set of test cases is not comprehensive: just because you pass does not mean your solution is correct! We strongly encourage you to write your own test cases and read more about ours in the comments of the Python script.

Make sure to include all required plots in your PDF.

1. Implement kNN for $k = \{1, 3, N-1\}$ where N is the size of the dataset, then plot the results for each k. To find the distance between points, use the kernel function from Problem 1 with lengthscale $\tau = 2500$.

   You will plot x^* on the year-axis and the prediction $f(x^*)$ on the temperature-axis. For the test inputs x^*, you should use an even grid spacing of 1 between $x^* = 800$ and $x^* = 400$. (Like in Problem 1, if a test point lies on top of a training input, use the formula without excluding that training input.) Again, this has been set up for you already.

   Please **write your own implementation of kNN** for full credit. Do not use external libraries to find nearest neighbors.



2. Describe what you see: what is the behavior of the functions in these three plots? How does it compare to the behavior of the functions in the three plots from Problem 1? In particular, which of the plots from Problem 1 look most similar to each in Problem 2? Are there situations in which kNN and kernel-based regression interpolate similarly? In general, knn and the kernel-based regression interpolate similarly using data on nearby data points to predict y-values in between the given x-values. This is especially true for $k = 1$, the plot is similar to when we did a kernel based regression with $\tau = 1$ as the function is very non-smooth with jumps to fit training data. As both functions gave the nearest neighbor the highest weight for prediction it makes sense the functions are very similar. For $k = 3$, the function is less complex as when we plotted the function for $\tau = 50$. However, the KNN function is never going to smooth as even when $k = 50$ there are jumps and vertical lines as apposed to in the kernel regression for $\tau = 50$ which was smooth. For $k = N - 1$, it was a straight line which makes sense since knn is just averaging across all the other datapoints which is quite different from the kernel based regression for $\tau = 2500$ which had a prediction curve but similar in that these two models gave higher weights to points farther away, as compared to the other models, both creating simpler models.

Problem 2 (cont.)

3. Choose the kNN model you most prefer among the three. Which model did you choose and why? What is its mean squared error on the test set?

k	1	3	N-1
Train MSE	1.9473	1.858	8.334
Test MSE	1.7406	3.8908	9.6640

I would prefer to use the model for $\tau = 1$ because that has the lowest train and test MSE. With $k = 1$, the model is very sensitive to noise and if we were going to use this model for larger scale predictions for more data I would prder the model for $k = 3$ because it is more generalizable. s

4. As before, say you wanted to send your friend your kNN, so that they can reproduce the same exact predictions as you. You will again tell them the value of the k you selected, but what other information would they need, assuming they do not currently have any of your data or code, and how does this information grow as a function of the size of the training set, N? Again worded more formally, what is the space complexity of storing your model?

   What is the time complexity of your implementation, when computing your model on a new datapoint? Give a brief overview of your implementation when you justify your answers.
   They would need my training data which would make the space complexity O(N). For a new datapoint, I would have to calculate the distance between points using the kernel function which takes constant time. Then, I have to sort those distances. I use *np.argsort* which uses the *QuickSort* with O(nlog(n)) time complexity. I then look up the last k elements of the list which is constant time. Calculating the mean is constant then appending to my list of predictions $[f(x_1), f(x_2), ...]$ is constant time. Thus, the time complexity is **O(Nlog(N))**

Problem 3 (Modeling Climate Change 800,000 Years Ago)

The objective of this problem is to learn about different forms of linear regression with basis functions.

Make sure to include all required plots in your PDF.

1. Recall that in *Ordinary Least Squares* (OLS) regression, we have data $\{(\mathbf{x}_i, y_i)\}_{i=1}^N = \{\mathbf{X}, \mathbf{y}\}$ where $\mathbf{X} \in \mathbb{R}^{N \times D}$. The goal is to find the weights $\mathbf{w} \in \mathbb{R}^D$ for a model $\hat{\mathbf{y}} = \mathbf{X}\mathbf{w}$ such that the MSE

$$\frac{1}{N}\|\mathbf{y} - \hat{\mathbf{y}}\|^2 = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

is minimized.

Without any novel bases, we have merely a single feature $D = 1$, the year, which is not enough to model our data. Hence, in this problem you will improve the expressivity of our regression model by implementing different bases functions $\phi = (\phi_1, \ldots, \phi_D)$. In order to avoid numerical instability, we must transform the data first. Let this transformation be f, which has been introduced in the code for you in the notebook.

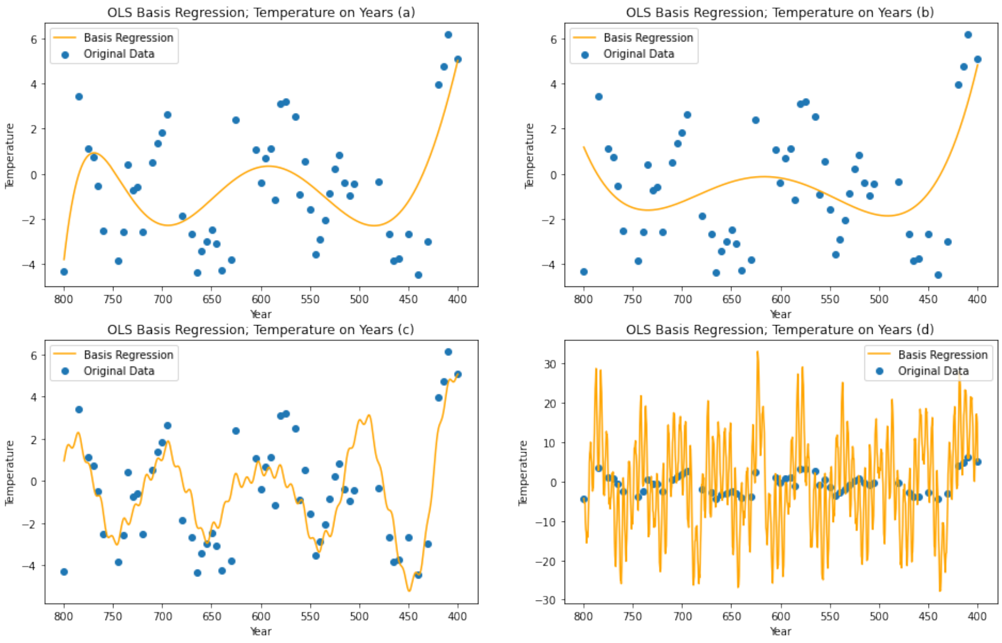
(a) $\phi_j(x) = f(x)^j$ for $j = 1, \ldots, 9$. $f(x) = \frac{x}{1.81 \cdot 10^2}$.

(b) $\phi_j(x) = \exp\left\{-\dfrac{(f(x) - \mu_j)^2}{5}\right\}$ for $\mu_j = \frac{j+7}{8}$ with $j = 1, \ldots, 9$. $f(x) = \frac{x}{4.00 \cdot 10^2}$.

(c) $\phi_j(x) = \cos(f(x)/j)$ for $j = 1, \ldots, 9$. $f(x) = \frac{x}{1.81}$.

(d) $\phi_j(x) = \cos(f(x)/j)$ for $j = 1, \ldots, 49$. $f(x) = \frac{x}{1.81 \cdot 10^{-1}}$. [a]

All you need to include in your writeup for 4.1 are these four plots.



[a] For the trigonometric bases (c) and (d), the periodic nature of cosine requires us to transform the data such that the lengthscale is within the periods of each element of our basis.

Problem 3 (cont.)

2. We now have four different models to evaluate. Our models had no prior knowledge of any of the testing data, thus evaluating on the test set allows us to make stronger (but not definitive!) claims on the generalizability of our model.

   Observe that there is never an objectively "good" value of MSE or negative log likelihood - we can use them to compare models, but without context, they don't tell us whether or not our model performs well.

   For each basis function, complete three tasks and include the results in your writeup:

   - Compute the MSE on the train and test set.
   - Assume that the data is distributed as $y_i = \mathbf{w}^\top \mathbf{x}_i + \varepsilon$ where $\varepsilon \sim \mathcal{N}(0, \sigma^2)$, we roll in the bias $\mathbf{x}_i = \begin{bmatrix} 1 \\ x_i \end{bmatrix}$, and each data point is drawn independently. Find σ_{MLE} and $\mathbf{w}_{\text{MLE}}$ (recall the formulas from class!) and use these to compute the negative log-likelihood of a model with parameters $\sigma_{\text{MLE}}, \mathbf{w}_{\text{MLE}}$ on your train and test sets. The following derives the likelihood.

$$p(\mathbf{y} \mid \mathbf{X}, \mathbf{w}, \sigma_{\text{MLE}}) = \prod_{i=1}^{N} \mathcal{N}(\mathbf{y}_i \mid \mathbf{w}^\top \mathbf{x_i}, \sigma^2_{\text{MLE}})$$
$$= \prod_{i=1}^{N} \frac{1}{\sigma_{\text{MLE}}\sqrt{2\pi}} \exp\left(-\frac{(y_i - \mathbf{w}^\top \mathbf{x}_i)^2}{2\sigma^2_{\text{MLE}}}\right)$$

   - Make a claim regarding whether this basis overfits, underfits, or fits well. Write 1-2 sentences explaining your claim using the train and test negative log-likelihood and MSE.

   ### Negative Log Likelihood (NLL)

Basis	Train MSE	Test MSE	Train NLL	Test NLL
Basis (a)	4.83	7.96	125.768	63.256
Basis(b)	5.53	8.71	129.620	64.035
Basis(c)	2.88	5.97	111.018	62.098
Basis(d)	0.64	58.86	68.30	1161.307

   - Basis (a) is underfitting with the second highest train MSE and NLL and third highest test MSE and NLL
   - Basis (b) underfits with the highest Train MSE and NLL. Additionally, the second highest test MSE and NLL. This basis is more underfit than basis A because it has higher MSE and NLL indicating that it was a worse fit overall.
   - Basis (c) does the best out of the three basis with the overall. It has the second lowest MSE and NLL for the train data and lowest MSE and NLL for the test data.
   - Basis (d) overfits the training data with a train MSE close to zero but with the highest test MSE. Similarily, the train NLL is the lowest but the test NLE is extremely high compared to the other NLL

Problem 3 (cont.)

3. For the third time, you wish to send your friend your model. Lets say you fitted some weight vector of dimension D. What information would you need to share with your friend for them to perform the same predictions as you? Do you need to share your entire training set with them this time? Again, what is the space complexity of storing your model?

   Given an arbitrary datapoint, what is the time complexity of computing the predicted value for this data point?

   How do these complexities compare to those of the kNN and kernelized regressor?
   The space complexity of this model is O(D) because you only need to give your friend your basis function and your weights. Given an arbitrary datapoint, the time complexity is **O(D)** because you need to transform the new point based on your basis function, which consists of arithmetic operations and fixed values(i.e arrays of fixed length and constants), and then do the dot product with w transpose. All of this will take **O(D)** which is better compared to he kNN and kernelized regressor.

   Your response should be no longer than 5 sentences.

Note: Recall that we are using a different set of inputs **X** for each basis (a)-(d). Although it may seem as though this prevents us from being able to directly compare the MSE since we are using different data, each transformation can be considered as being a part of our model. Contrast this with transformations (such as standardization) that cause the variance of the target **y** to be different; in these cases the MSE can no longer be directly compared.

Problem 4 (Impact question: Building a descriptive (explanatory) linear regression model to understand the drivers of US energy consumption, to inform national policy decisions by the US President.)

Prompt: You are leading the machine learning team that is advising the US president. The US president is concerned about 3 things - climate change, the energy crisis in Europe and sustainable energy security in the US and asks you to help him understand what the driving factors of annual US energy consumption might be.

How would you build a regression model that can be used to explain the driving factors of the annual US energy consumption? Please answer the questions below by using concise language (350 - 700 words). Bullet points are appropriate.

1. **Target variable:** What target variable would you choose and what would be its unit? I would target monthly energy use per capita in Kilowatt-Hours.

2. **Features:** List 5 possible features and explain your assumption why you think they might impact the target variable.
   The five possible features I would look at are temperature, energy prices, age, number of electronics, and occupation. Firstly, temperature in the local area is an important feature because energy consumption will vary by month based on where the person lives. According to the US Energy Information Administration **(EIA)**, Americans, on average, tend to use more energy in the summer months so want to know how much the change in energy consumption is due to temperature. Additionally, I care about energy prices compared to cost of living in the area in order to see how energy consumption may be driven by what people can afford.I would also take into account age because different age groups need energy for different reasons (i.e someone in their thirties who has a family versus a person in their 20s). Number of electronics significantly affect energy usage because electronics need energy. I also want to know the occupation of the person using the energy because some jobs may require larger energy consumption at home due to working from home or other factors.

3. **Dataset size:** What should be the size of your dataset / covered time period? Why?
   My covered time period would be 1960-2022. According to the **(EIA)**, nuclear energy consumption started in 1960. Nuclear makes up 18.9% and renewable energy make up 19.8% of of total energy consumption in the US in 2021 (**(EIA)**). I want data that takes into account the variety of energy options available in order to create a model that is modern enough to predict the current energy situation in the U.S. In order to make my modeling easier, instead of sampling the entire population, I would want to take a sample of around 10 million or more people in order to get a representative sample of the population. I am assuming I have access to all of this data that already exists. If I were to use the data of 10 million random people per month per year, I would have 7.44 billion datapoints.

4. **Performance metric:** What metric would you use to assess the model's performance? I would use MSE to assess my model's performance because it has very useful properties, as discussed in class, and it is will be more easily interpretable if I have to explain it to politicians since MSE preserves information about the target variable.

5. **Policy decision:** Explain one policy decision the US president could make based on your model. Based on my model, the president could decide to allocate more money to educational program targeted towards certain age groups based on if the model finds age as a strong predictor of energy consumption or if certain age groups seem to have exorbitant energy consumption.

6. **Trust:** What could be barriers for the US president to trust your model? List two possible barriers. Since I am not experienced in the field, the president may be worried that I have not chosen the best model for the trend in the data. Perhaps he doesn't trust that I choose the right basis function or how I choose my parameters when making my function. Secondly, he may not trust my choice of features and may fear omitted variable bias or covariance among the variables chosen.

7. **Risk:** What happens if your model is wrong/inaccurate? List one real-world consequence. If my model inaccurately says that a certain occupation and energy usage is positively correlated, there may be unfair regulations put in place towards that occupation or hateful media coverage.

Name : Lily Nguyen

Collaborators and Resources

Angelika Antsmae

Calibration

Approximately how long did this homework take you to complete (in hours)? 10 hours