III. Model documentation and write-up

You can respond to these questions either in an e-mail or as an attached file (any common document format is acceptable such as plain text, PDF, DOCX, etc.) **Please number your responses.**

1. Who are you (mini-bio) and what do you do professionally?
   Senior Data Scientist and Director of Lose It! Labs at Lose It! My responsibilities are everything concerning predictive modeling, AI innovation, machine intelligence, etc. I am the creator of the Snap It food recognition algorithm released in 2016 as an industry first for food logging. I was formerly a research professor at Vanderbilt University where I directed novel methods development for machine-learning based computer-aided drug design.

   If you are on a team, please complete this block for each member of the team.

2. High level summary of your approach: what did you do and why?
   I created an ensemble of models using the standard approach from the blogpost and also another ensemble of models after adding some per row interpolation to the data. these two ensembles were then combined by averaging the predictions.

3. Copy and paste the 3 most impactful parts of your code and explain what each does and how it helped your model.

   the most impactful parts are the actual averaging of many methods (lasso, kernel ridge, linear, extra trees, gradient boosting, support vector regression, and random forest)

   second most important was the interpolation function for missing data:

```
def preprocess_timeseries_interpolation_perrow(timeseries, first_year, fillna_value=0):
  # column type
   timeseries.columns = timeseries.columns.astype(int)
   def interp(xxx):
      xx = xxx.values
      if np.isnan(xx).sum() == xx.shape[0]:
         xxx.values[:] = fillna_value
         return xxx
      mask = np.isnan(xx)
      xx[mask] = np.interp(np.flatnonzero(mask), np.flatnonzero(~mask), xx[~mask])
      xxx.values[:] = xx
      return xxx

   # subset to just data after first_year
   timeseries = timeseries.loc[:, timeseries.columns >= first_year]
   timeseries = timeseries.apply(lambda x: interp(x),axis=1)
   timeseries.fillna(fillna_value, inplace=True)
```

return timeseries

third most important is the lag search included in the original blog post.

4. What are some other things you tried that didn't necessarily make it into the final workflow (quick overview)?
   I tried bidirectional GRU networks which I suspect I gave up on too quickly. There may have been a bug in that code and I probably should have pushed on that a bit harder.

5. Did you use any tools for data preparation or exploratory data analysis that aren't listed in your code submission?
   no

6. How did you evaluate performance of the model other than the provided metric, if at all?
   just mape

7. Anything we should watch out for or be aware of in using your model (e.g. code quirks, memory requirements, numerical stability issues, etc.)?
   no

8. Do you have any useful charts, graphs, or visualizations from the process?
   no

9. If you were to continue working on this problem for the next year, what methods or techniques might you try in order to build on your work so far? Are there other fields or features you felt would have been very helpful to have?
   I think I was successful in creating a more generalizable model because I bagged so many different methods together. To this end, I feel like I should have added an additional bag of RNN-based methods as well.