Michigan Outdoor Recreation Search Interest Installment 8

This is my project to analyze and forecast Google search interest for 10 forms of oudoor recreation by people in Michigan. My data is daily data for each form of search interest from January 2021 to March 2025. The search interest terms for this project are atving, boating, camping, fishing, hiking, kayaking, rving, hunting, skiing, and snowmobiling. For more information on how this data is pulled, please see installment 5:

https://dataandoutdoors.com/michigan-outdoor-recreation-search-interest-installment-5/

This installment is my third of three attempts/methods to model and forecast search interest based on this data.

Long Short Term Memory Model

In previous installments, I attempted to make a non-stationary model and a stationary model. The nonstationary model accounts for trend and seasonality in the data. The stationary model accounts for idiosyncratic factors that may cause search interest to differ from what is predicted based on trend and seasonality. An example of this would be weather. In this installment, I employ a Long Short Term Memory (LSTM) neural network to the data series as a single model.

As with the previous installment, I consider 'exogenous' explanatory variables. These are four daily weather variables: precipitation, snow depth, maximum temperature, minimum temperature. These series are the average of the values taken from 105 different weather stations throughout Michigan. For more information on how these data series were created, please see installment 3:

https://dataandoutdoors.com/michigan-outdoor-recreation-search-interest-installment-iii/

Unlike with Installment 7 where I fit a feed forward neural network, I attempt to limit the number of variables considered in the LSTM model. For instance, I only use a lag of the same variable in each model, not the lags of the other nine forms of search interest. Then, I limit the weather variables such that snowfall is only considered for fishing, hiking, hunting, skiing, and snowmobiling. I also attempt to limit the holidays

considered to those that would be most relevant to the particular type of search interest. Please see table below for the holidays considered for each search interest. Also, see the link to Installment 7 which explains how I created variables in the previous installment.

https://dataandoutdoors.com/michigan-outdoor-recreation-installment-7/

Activity	Holidays	Snow Depth?
ATVing	Memorial Day, July 4, Labor Day	
Boating	Memorial Day, July 4, Labor Day	
Camping	Memorial Day, July 4, Labor Day	
Fishing	Memorial Day, July 4, Labor Day	Yes
Hiking	Memorial Day, July 4, Labor Day	Yes
Kayaking	Memorial Day, July 4, Labor Day	
Rving	Memorial Day, July 4, Labor Day	
Hunting	Nov 15, Thanksgiving, Christmas, New Year	Yes
Skiing	Christmas, New Year, Valentine	Yes
Snowmobiling	Christmas, New Year, Valentine	Yes

Additionally, for this installment I perform additional data cleaning steps on the variables. For instance, it's easier for neural networks to fit data when they are on a similar scale. While I do not attempt to scale the binary variabes, the other variables I attempt to make more similar. The weather variables are already scaled by the mean and standard deviation for that week, approximately placing them in a range of -2 to 2. The lagged values of search interest are also scaled, but from 0-100. Additionally, I have calendar day and day of the week. Finally, I limit the days impacted by many holidays from three full days before and after to only the weekend of the holiday. Below is a summary of the transformations I employ.

1. Lagged Search Interest: X/25-2, which transforms from 0 to 100 range to -2 to 2 range.

2. Day of the year: X/91-2, which transforms from 1 to 365 range to -2 to 2 range.

3. Day of the week: Make into 7 binary variables.

4. Memorial Day, Labor Day, and Thanksgiving Day: Only consider the long holiday weekend (including Friday) as part of the holiday.

Aspects of the LSTM model are much less well known than the feed forward neural network. LSTM is a type of recurrent neural network. Not only does a recurrent neural network transform underlying features with nonlinear transformations in nodes, but it also considers sequences of data. Essentially, the recurrent neural network completes the sequence. This makes it ideally suited for ordered time series data. The LSTM also considers a 'state' variable in addition to the current sequence when assigning output. In other words, the model will be different and transform the same input differently depending on where it occurs in

the timeline based on this state variable. The LSTM model changes this state in each step using update and forget processes or 'gates.'

A full discussion of the LSTM model is beyond that scope of this writing. However, a discussion of recurrent and LSTM neural networks can be found in the MIT EdX course "Machine Learning with Python". This course is availble for free on an audit basis.

https://www.edx.org/learn/machine-learning/massachusetts-institute-of-technology-machine-learning-with-python-from-linear-models-to-deep-learning

While the machine learning course discusses the concepts behind LSTM networks, it doesn't provide any details on implementation. I decided to implement the LSTM model using tensor flow and kera python packages. To do this, I heavily borrowed python program snippets from the following articles and video:

https://medium.com/@zargi.teddy7/lstm-long-short-term-memory-for-sales-forecasting-with-exogenous-features-a-deep-dive-into-fe922c283c85

https://dipanshu10.medium.com/implementing-lstm-networks-with-python-a-guide-using-tensorflow-and-keras-915b58f502ce

https://www.youtube.com/watch?v=94PlBzgeq90

There are a lot of hyperparameters in the LSTM model. Some were set by the example of others, still others I set by trial and error, and finally some were set using cross validation.

In my feed forward neural network, I didn't use many nodes per layer (up to five maximum). However, by example of others, I used many more nodes for the LSTM. In fact, based on trial and error I ended up using more nodes than what others were using in their projects. Specifically, I used two LSTM layers with 128 and 64 nodes respectively. Below is the snippet of my program:

```
LSTM(128, input_shape=(ts, X1.shape[2]),dropout=0,
recurrent_dropout=0, return_sequences=True),
    LSTM(64, dropout=do, recurrent_dropout=0),
    Dense(1)
    ])

model1.compile(optimizer='adam', loss='mean_squared_error', metrics=
['mean_absolute_error'])

history = model1.fit(
    X1, y1,
    epochs=ep,
    batch_size=32,
    verbose=verb)
```

In the above snippet, 'ts' stands for the time steps in each sequence for the reccurent neural network. In this case, I chose seven (7) due to the weekly structure of my daily data. I also tried one day, essentially no sequence at all, and it didn't work very well.

The hyperparameter 'do' refers to dropout rate which is the percentage of random nodes dropped when making predictions. This is intended to prevent overfitting or reliance on a single node. I set the drop out using cross validation with each model using the options of 0.1, 0.3, and 0.5 dropout.

The final hyperparameter I wish to discuss is 'ep' or the epochs used during fitting. This is the number of times the model works through the entire training dataset to fit the model. More epochs leads to a better fit of the training data at the potential expense of overfitting the cross validation data. I set this hyperparameter using cross validation with choices of 60, 80, or 100 epochs. I chose these three options through trial and error as it seemed to work better in my case to use more epochs than others were using in their projects.

For training and cross validation, I held out the last year of data for cross validation just like in the last project. Just as in last project, I fit the model on the training data using one set of hyperparameters, and then use the model to make many 14 day forecasts in the year of holdout data and averge the MSE for all of these forecasts. In the last project, I made a 14 day forecast starting with each day. However, due to longer computation in this project I only made a 14 day forecast starting every 7 days. This approach is appropriate because the purpose of these models will be to produce a new 14 day forecast every week.

The below table shows the chosen number of epochs and drop out rate for each model.

Activity	Epochs	Dropout
ATVing	80	0.3
Boating	100	0.1
Camping	80	0.1
Fishing	60	0.5
Hiking	100	0.1
Kayaking	100	0.3
Rving	100	0.1
Hunting	100	0.3
Skiing	80	0.5
Snowmobiling	60	0.1

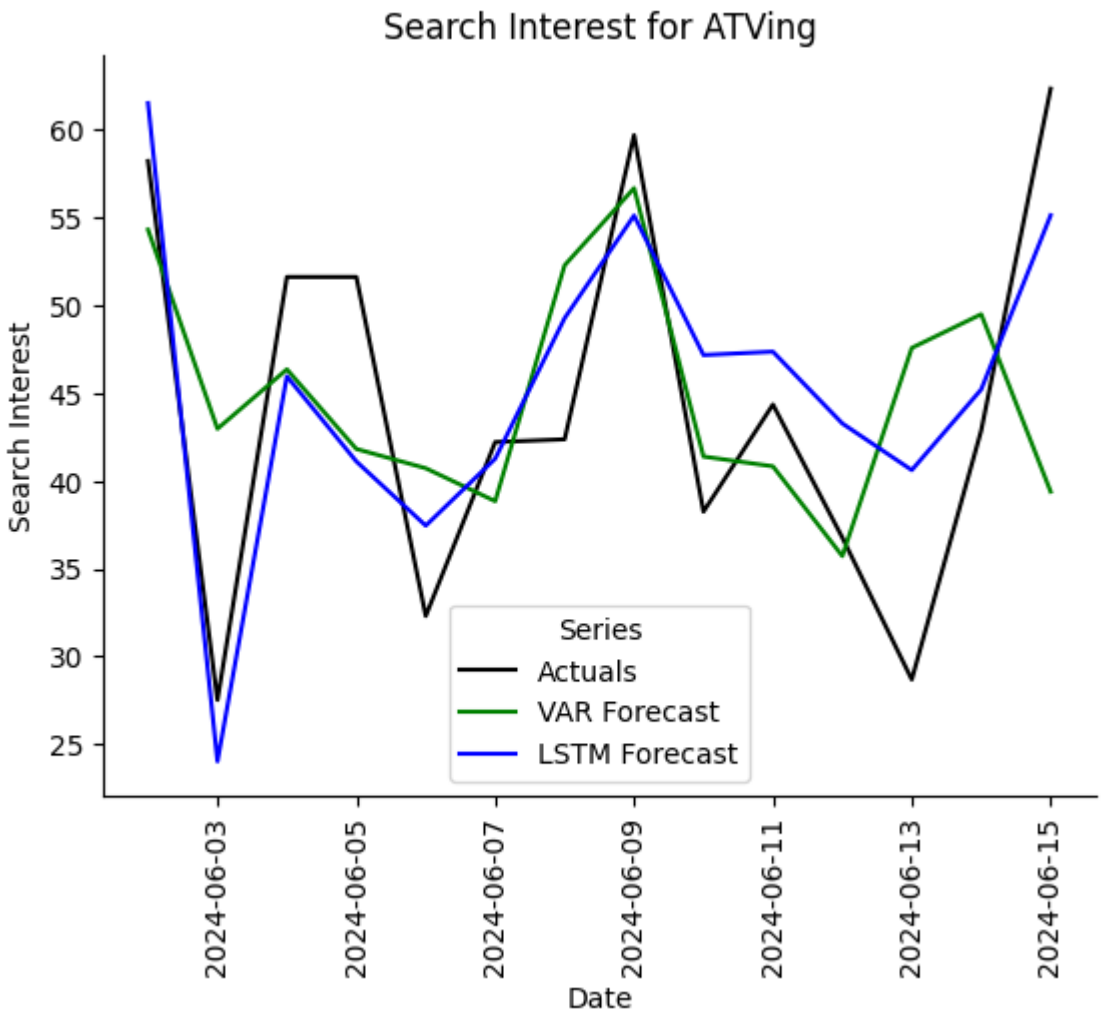
Selected Forecasts

Just as in Installment 6, I choose a different forecast period for each of the data series. In addition to plotting actuals and the results of Model 1, I now overlay the results from Model 3. (Note that I'm dropping Model 2 from consideration because the results were too close to Model 1.)

Please see Installment 6 for more details on these forecast periods. Note, that this isn't intended to be a comprehensive evaluation of how the models will perform over various time periods. Instead, these forecasts are intended for illustrative purposes only.

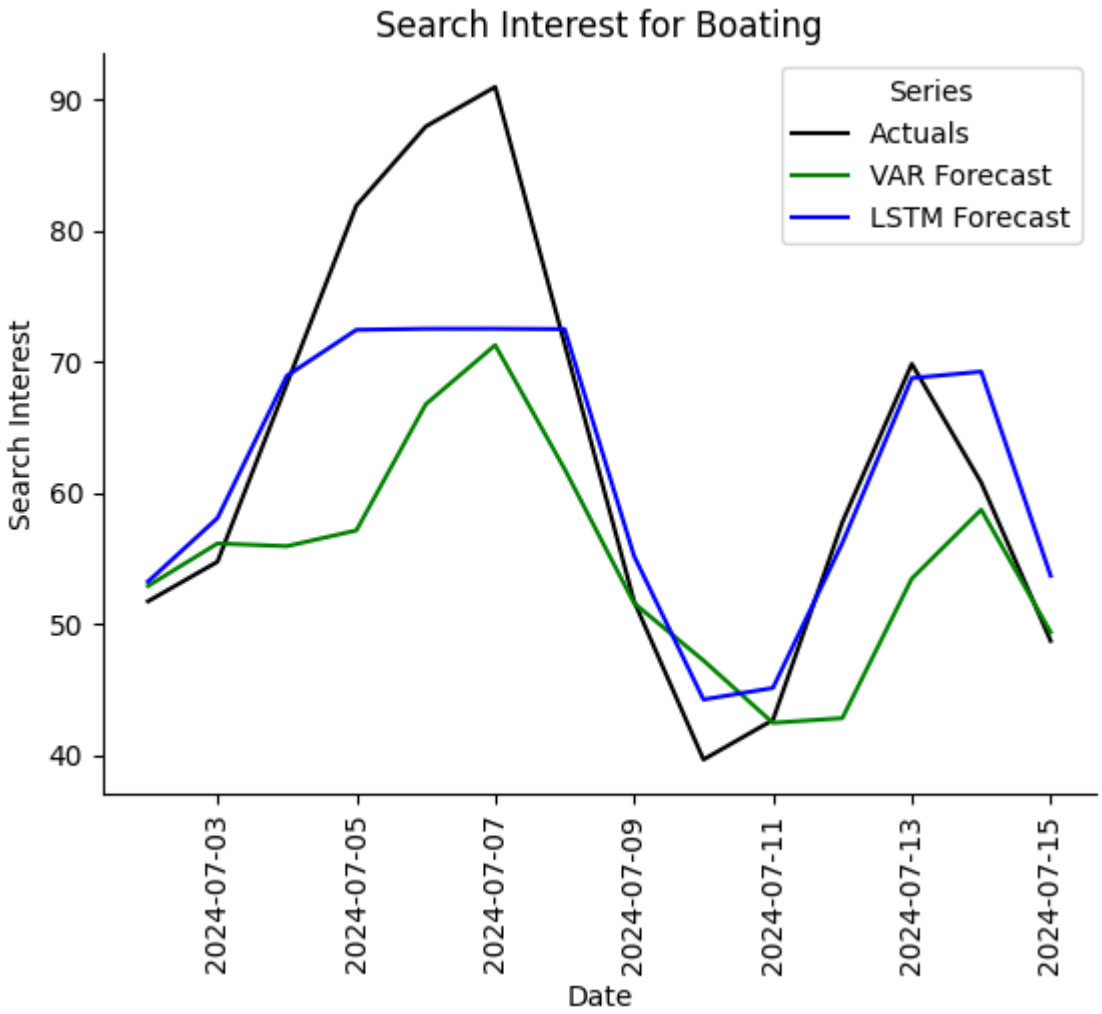
ATVing

For ATVing, I forecast the first week of June 2024. The actual series is highly variant. However, both models capture the general level of search interest and the peak seen on June 9. Given the actual series, this is the best I could hope for.



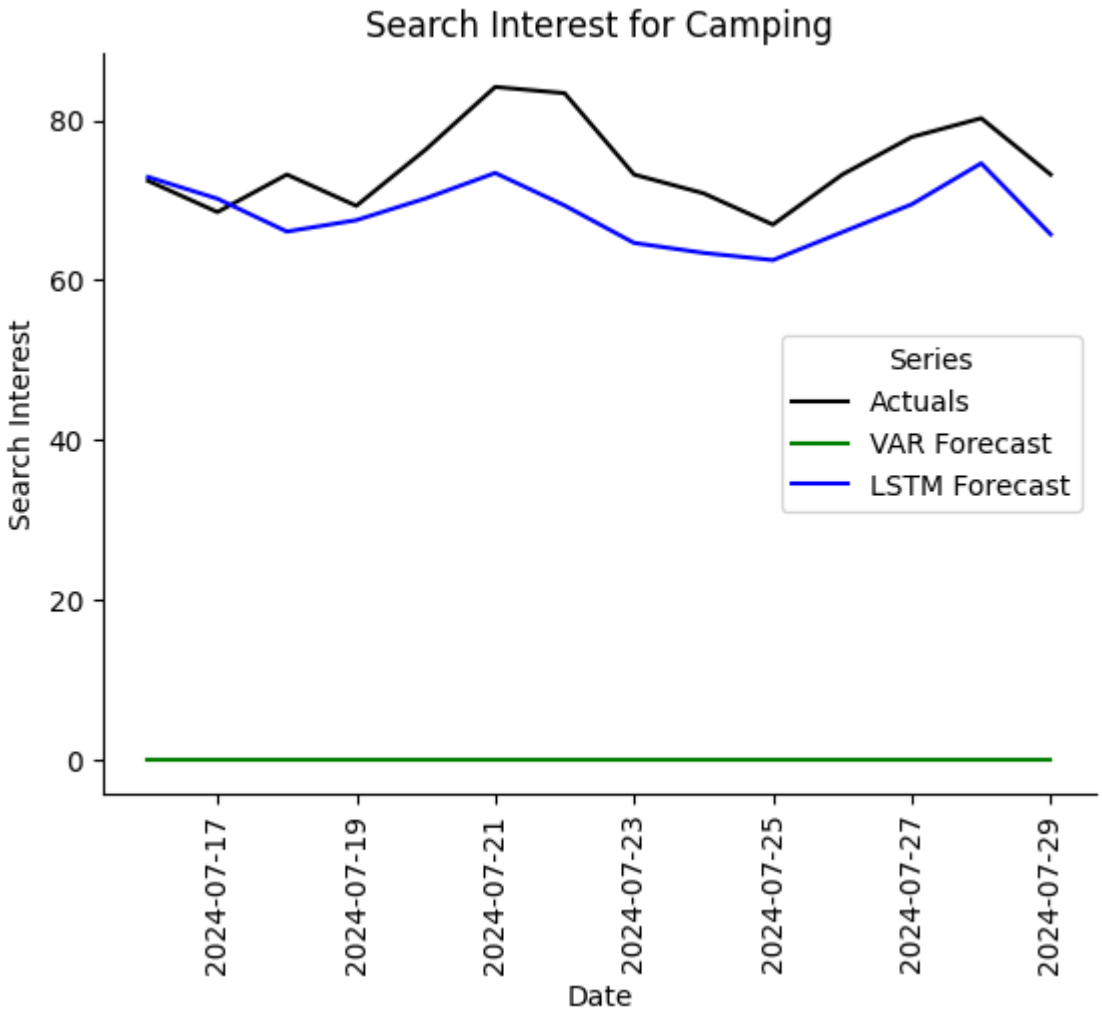
Boating

For boating, I forecast the first week of July 2024, which includes the 4th of July holiday. Both models capture the general pattern in the actuals, but the LSTM model is closest. See the holiday effect in the LSTM model from July 4-9.



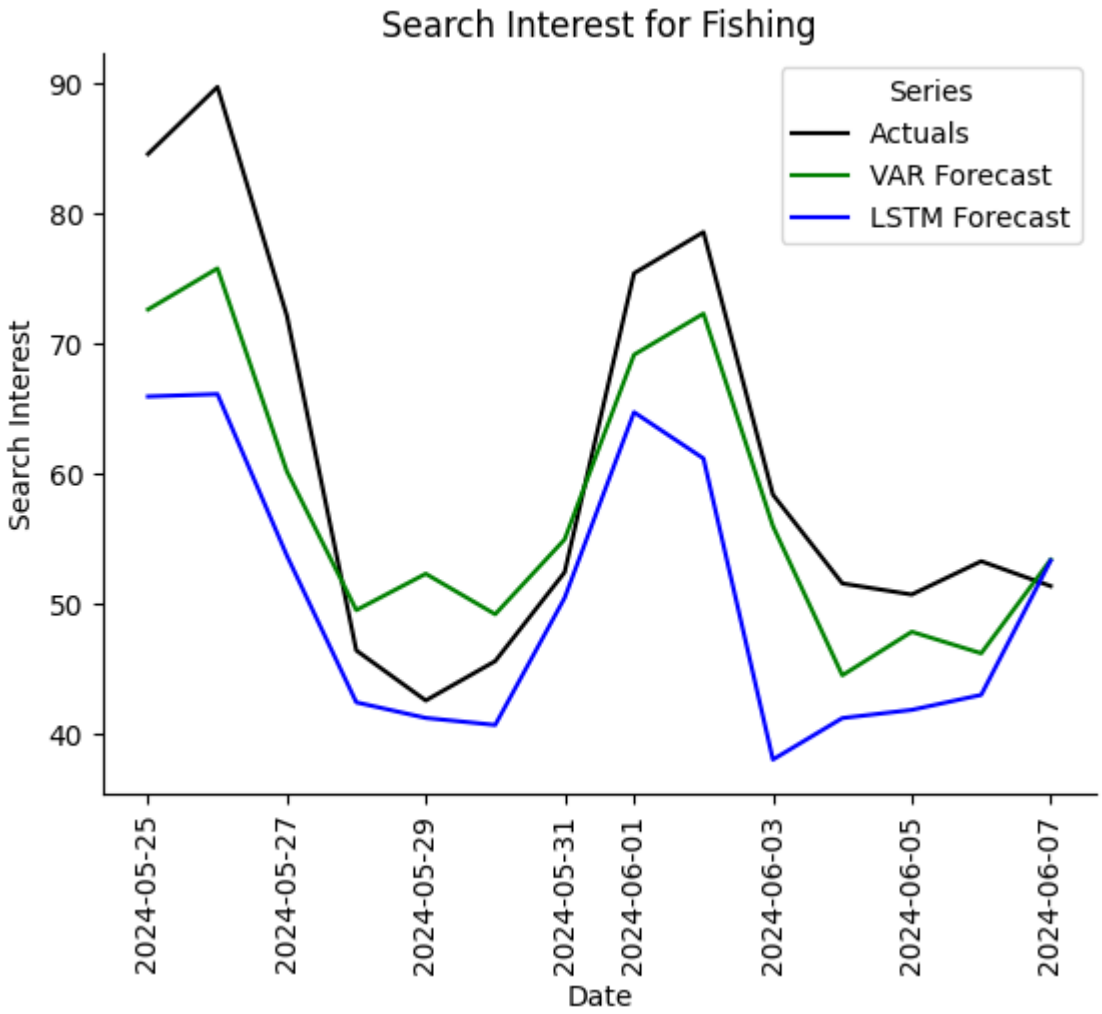
Camping

Unfortunately, for camping I wasn't able to produce a forecast with the VAR method (see constant green line where I zeroed out the forecast). However, the LSTM model works and works fairly well. This forecast is for the second half of July 2024.



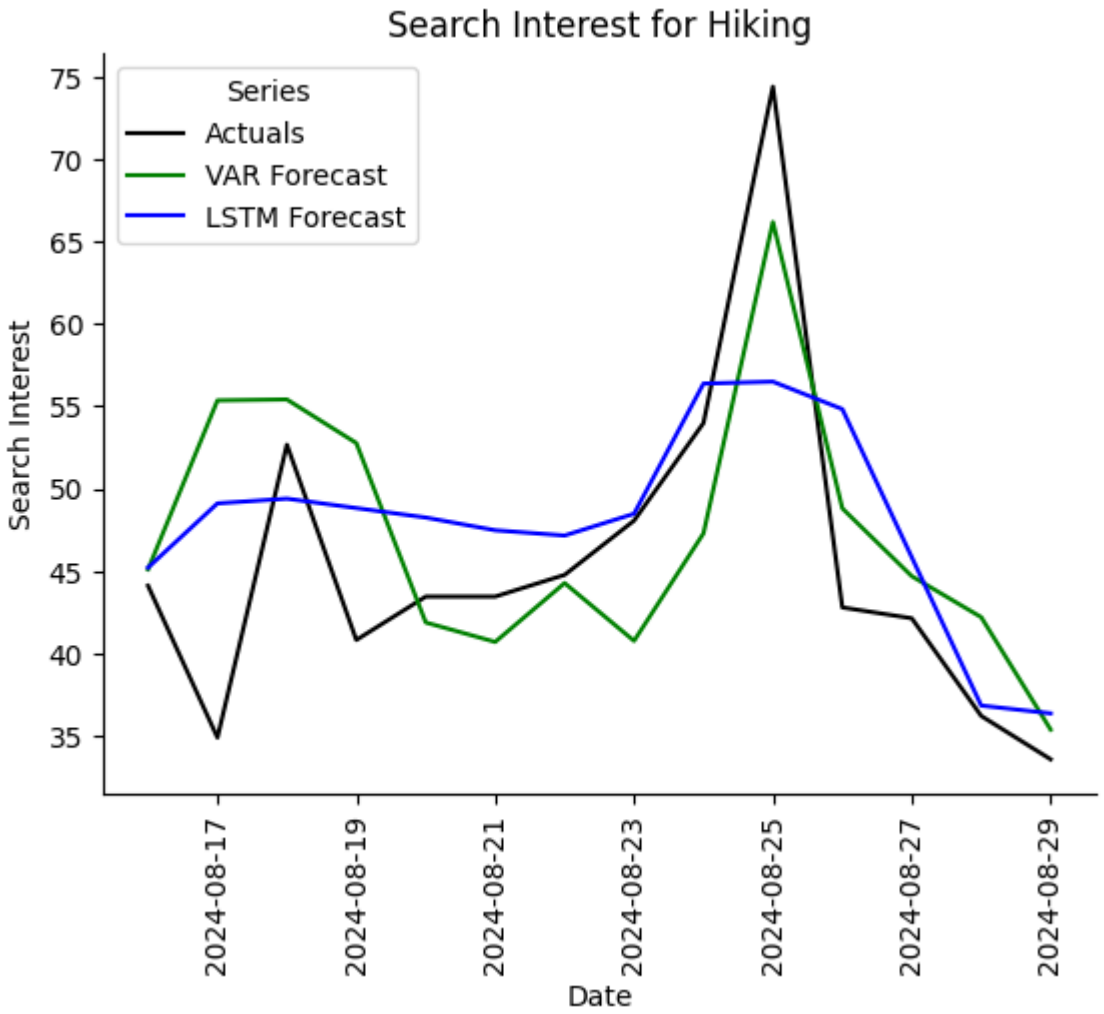
Fishing

For fishing I look at a forecast starting before Memorial Day and the start of bass fishing season. Both the VAR and LSTM model capture the general pattern of the data but the VAR model is closer than the LSTM.



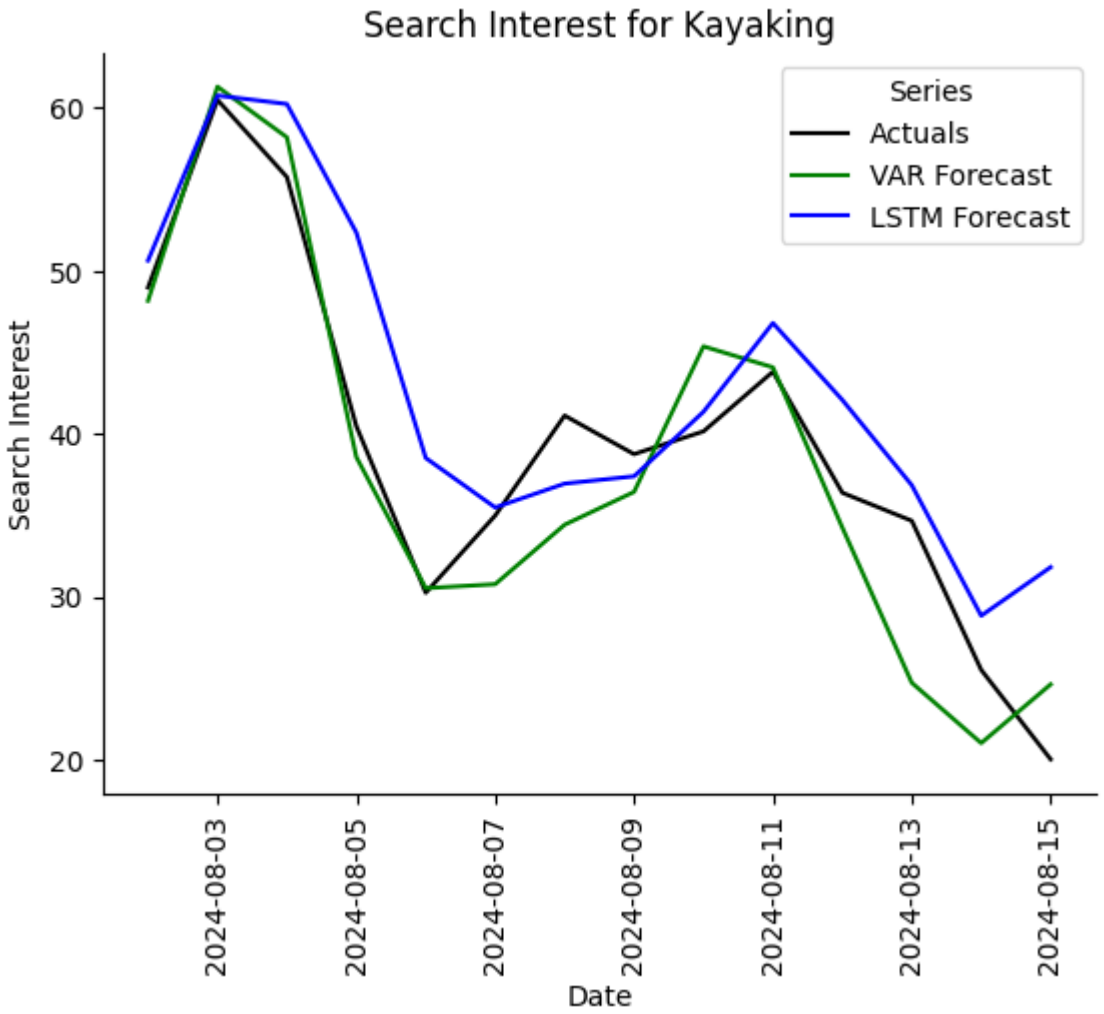
Hiking

For the forecast starting July 15, The VAR model seems to perform better than the LSTM network.



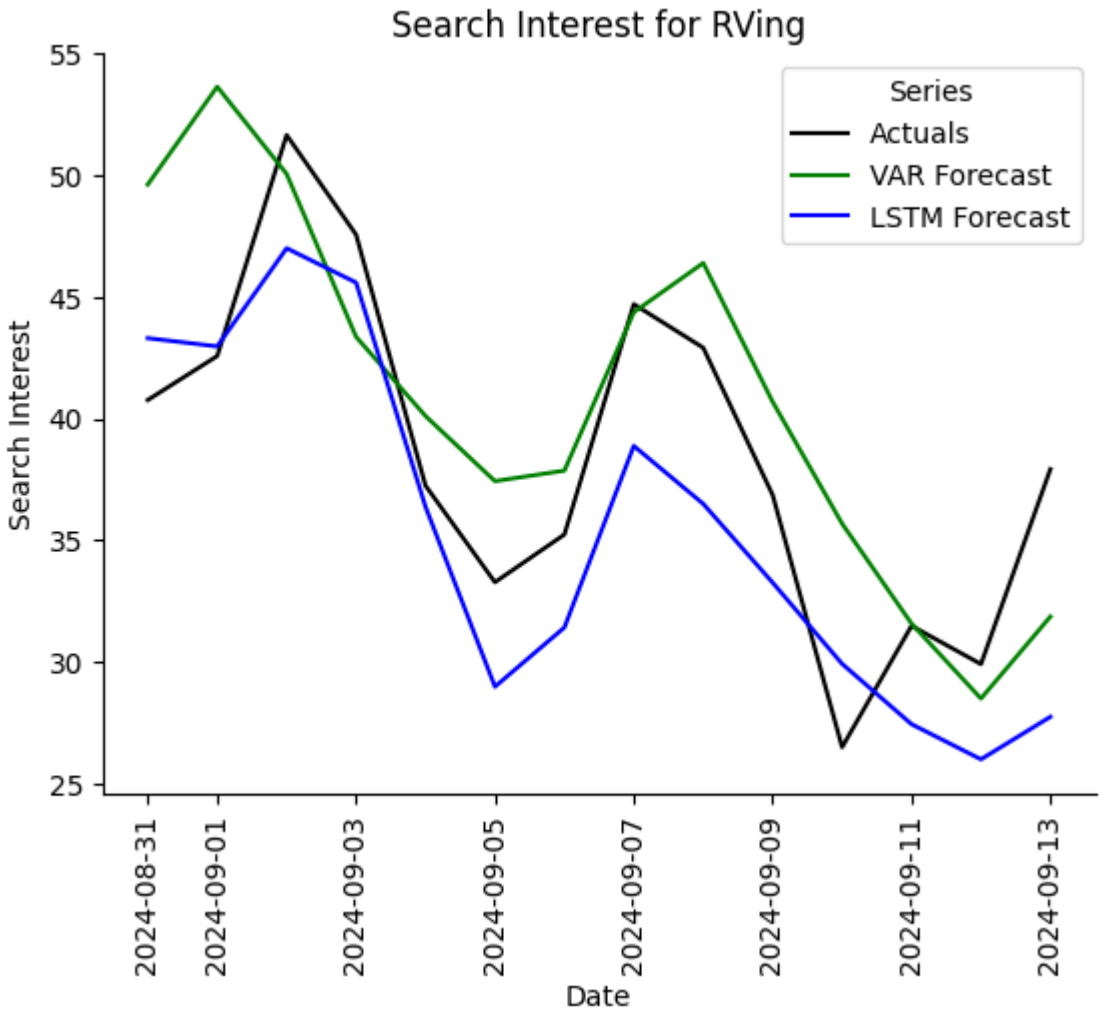
Kayaking

I provide a forecast for Kayaking starting August 1. The VAR network appears to perform better than the LSTM network.



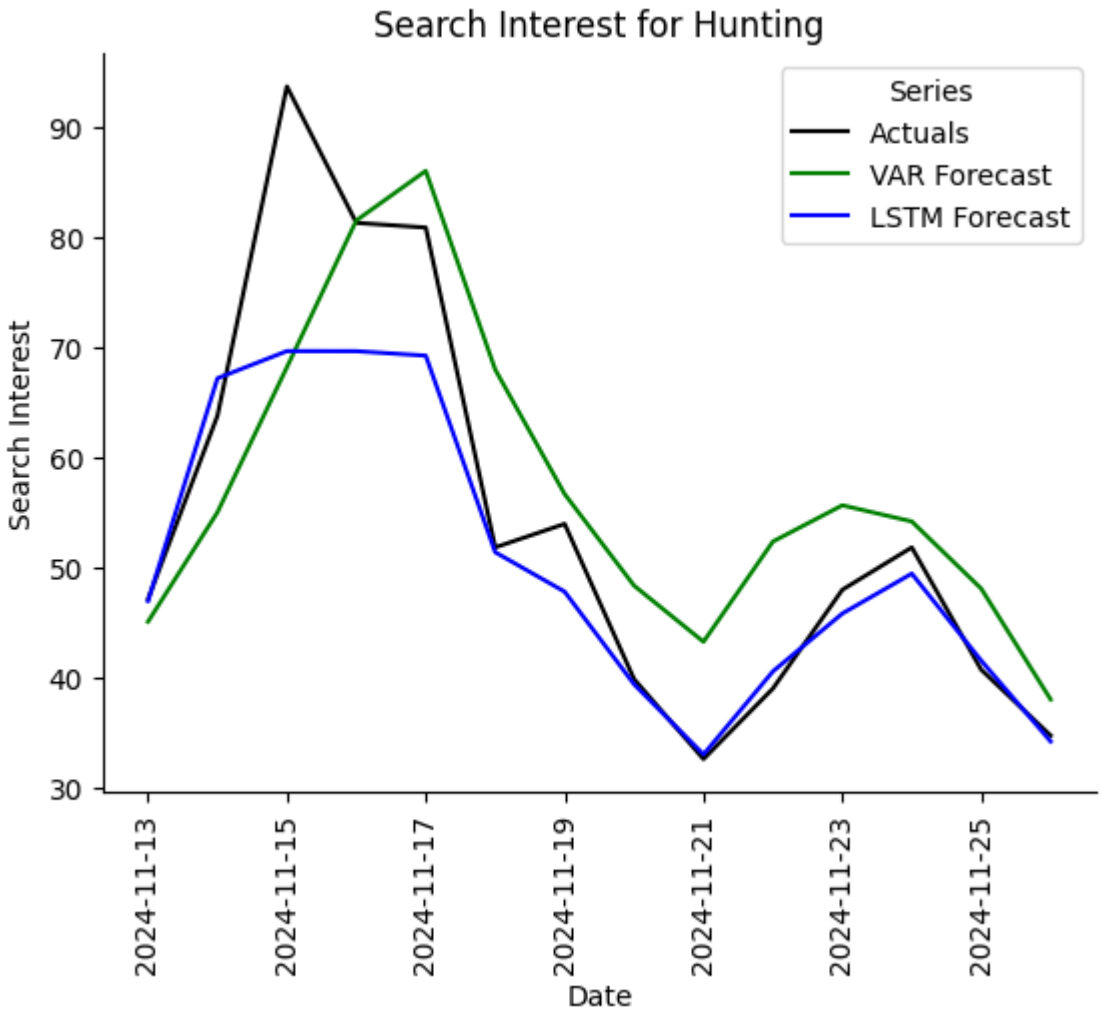
RVing

My forecast for RVing is over Labor Day weekend. Both the models capture the general pattern of the actuals with the LSTM undershooting actuals and the VAR model overshooting actuals.



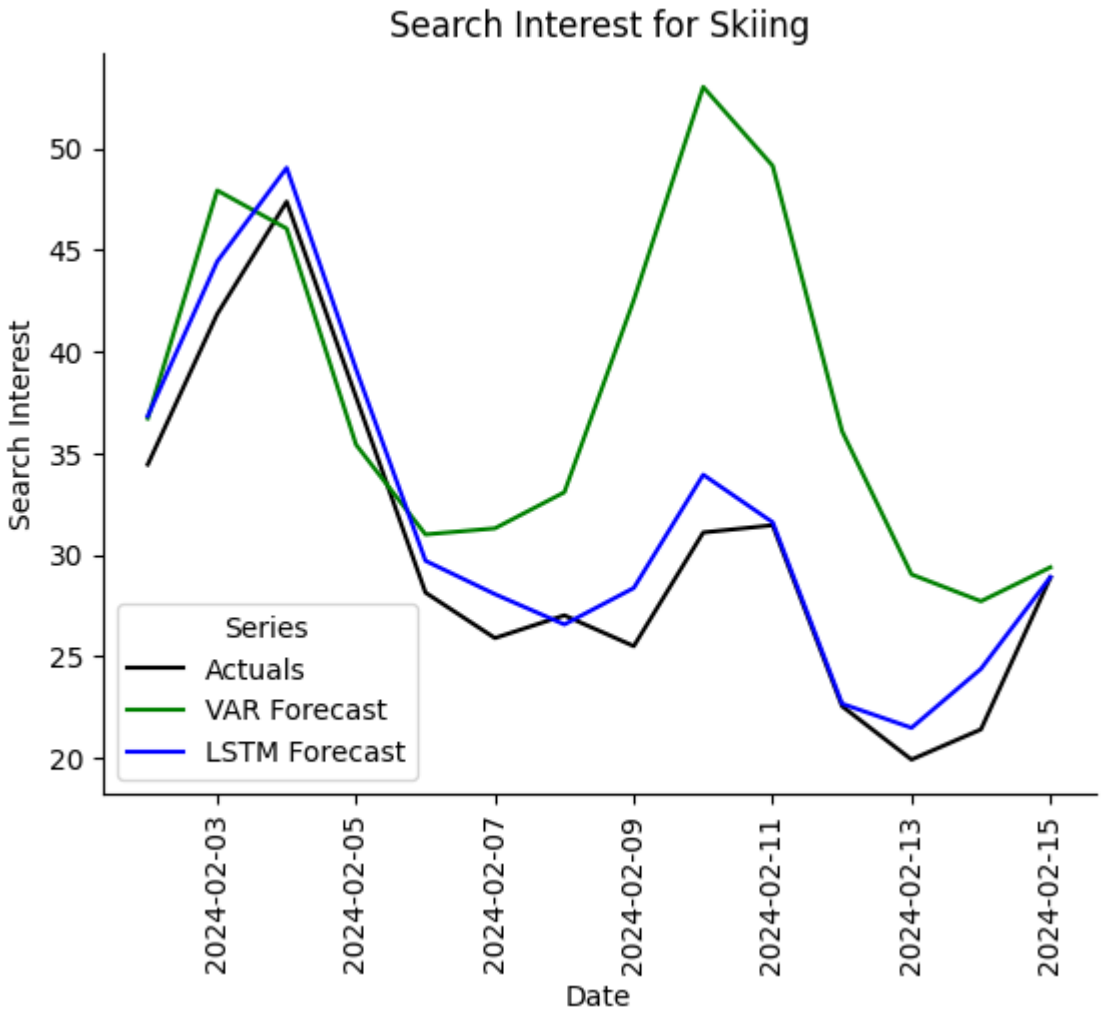
Hunting

I provide a forecast for hunting over November 15th, which is the opening of firearms deer season in Michigan. The VAR model seems to fit better the first week of data capturing the peak during Nov 15th weekend better. However, the LSTM model fits better the second half of the forecast period. Note the VAR model can better account for the high values from the previous year due to the hankel matrix nonstationary model that considers the previous year's value. The LSTM model attempts to fit annual seasonality with a transformation of day of year so it isn't able to capture this peak. However, it does have a holiday effect that can be seen in the series from Nov 14-17th.



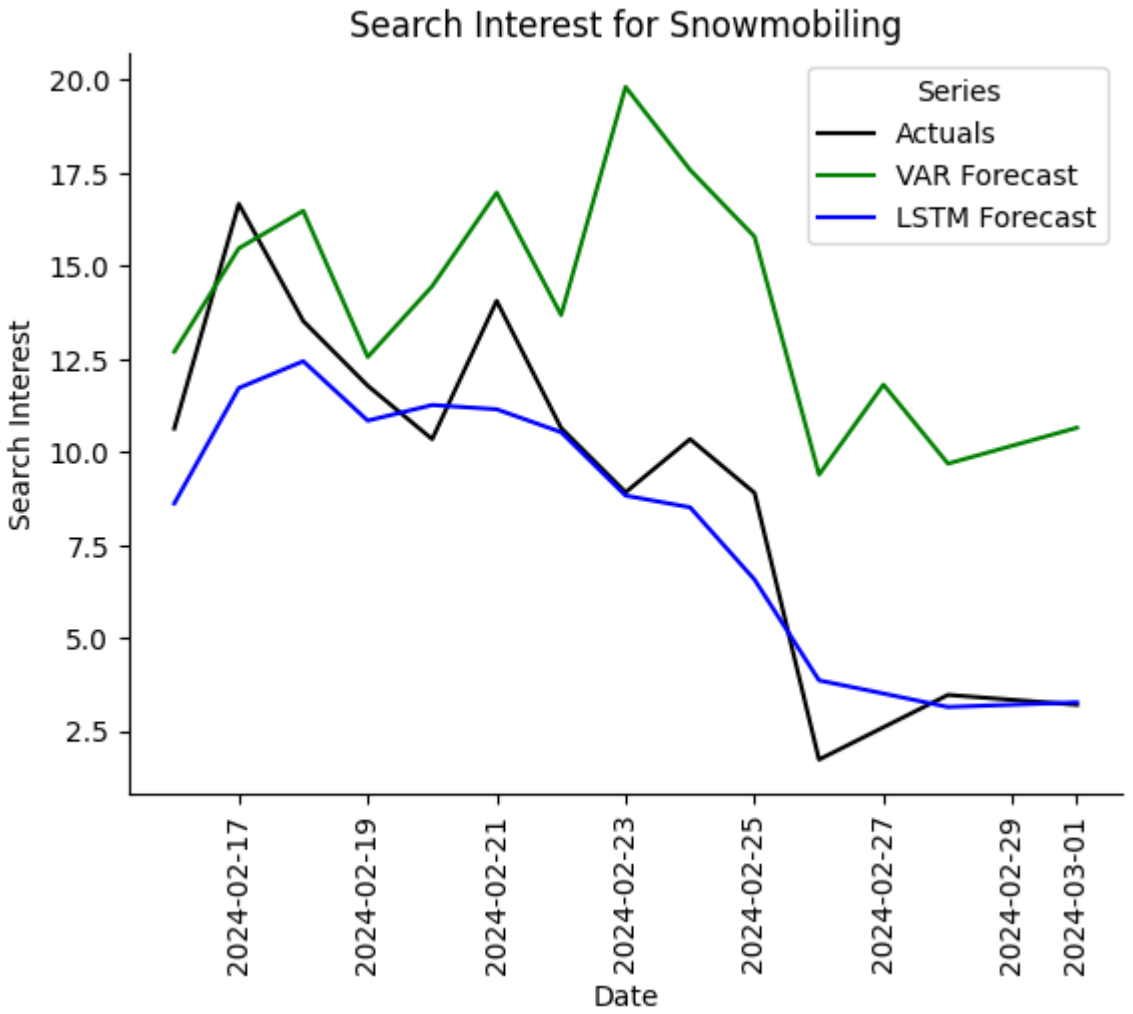
Skiing

I provide a forecast for skiing starting February 1st. The LSTM model fits this series much better, especially in the second half of the forecast.



Snowmobiling

I provide a forecast for snowmobiling starting on February 15th. The LSTM model fits this forecast period much better.



Conclusion

So far I've tested three options.

The first was a hankel matrix based nonstationary model paired with a stationary VAR model. This model performed fairly well in some series but very poorly for others, especially camping, skiing, and snowmobiling. For camping, I couldn't even get a model.

The second model was to replace the stationary model above with a feed forward neural network. This failed to provide forecasts significantly different from the first model. This model is now dropped from consideration.

The third model is the LSTM model. In many cases, this model was a great improvement upon the first VAR model, especially for camping, skiing, and snowmobiling. For the other seven series, it was a toss-up between the VAR and LSTM models. In many cases the LSTM model performed relatively poorly, I was using all 100 epochs to fit the model. This makes me wonder if more epochs would have done better.

However, these forecasts only analyzed a single 14 day forecast period. For the next installment, I will make a final cross validation of many forecast periods comparing the first and third models. My guess is that the best model will be different between the ten series. I will most likely allow for this flexibility in the final version.