PREDICTING PROBABILITY OF HAILSTORMS USING TIME SERIES FORECASTING

Jessica Manko

GEOG5990 Final Project

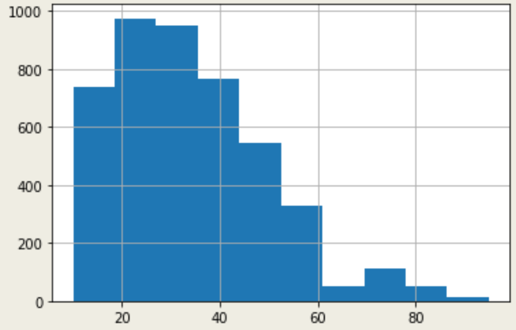
Spring 2020

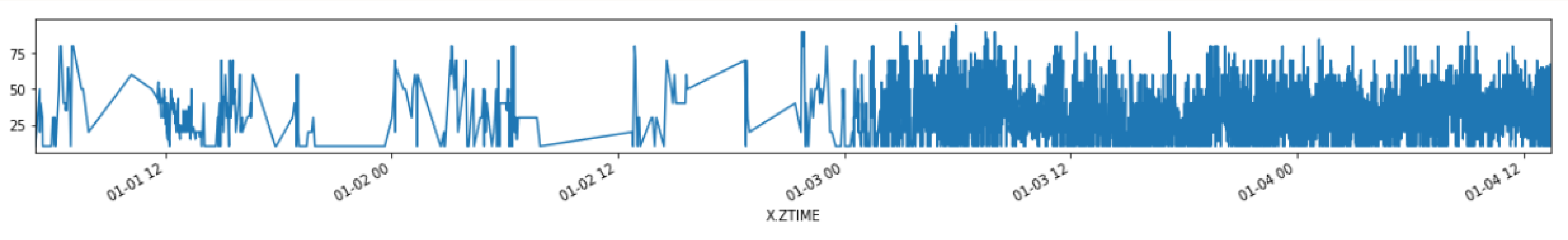
Introduction

- Severe weather is a meteorological phenomena with the potential to cause serious damage.
- Hail is a form of precipitation consisting of solid ice that forms inside thunderstorm updrafts and can cause billions of dollars of damage to structures, crops and livestock.
- It is useful to accurately forecast when these events will occur.
- The goal of this project is to run a time series prediction on the probability of hail events occurring using the Prophet and ARIMA models.

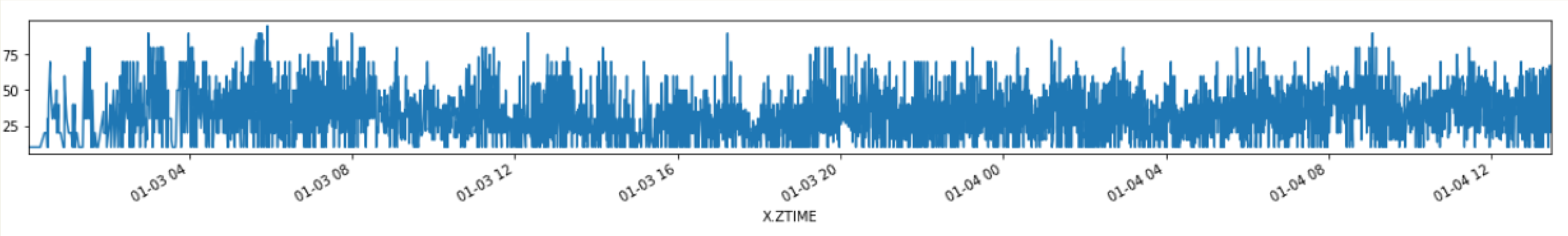
Data Preprocessing

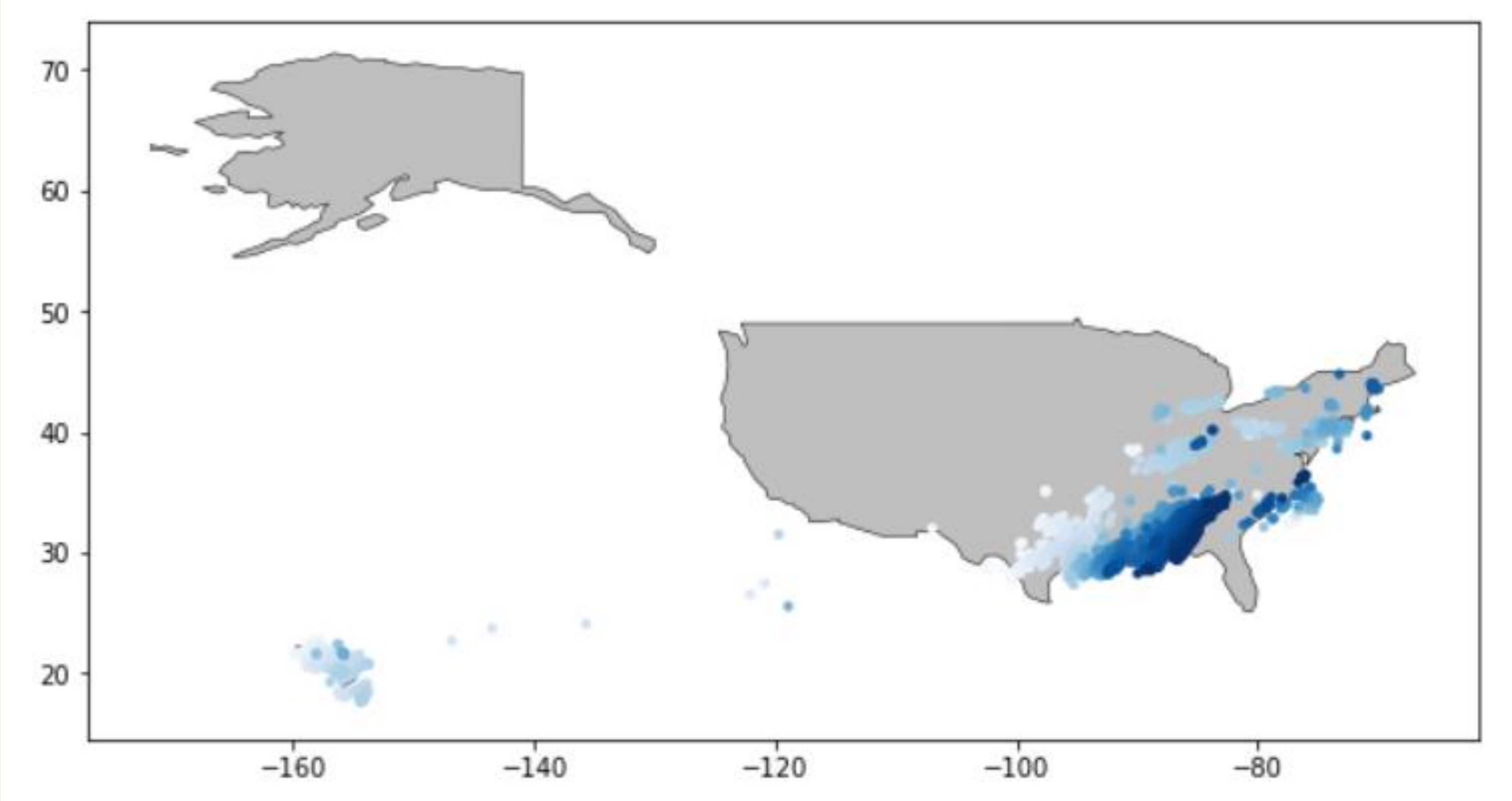
- The dataset is a product of NOAA from the Severe Weather Data Inventory representing detections of hail storm cells in the United States based on NEXRAD radar data during 2015.
- 10 rows: time, latitude, longitude, azimuth, range, wsr_id, cell_id, probability of hail, probability of severe hail, and maximum size of hail.
- 1,048,576 values; subset the first 20,000; clean up data by removing duplicates and null values.
- Create a datetime index to run the time series prediction.
- Subset the data frame to the columns needed to run analysis.
- Trim sparse data away and focus on events after 01-03-2015.

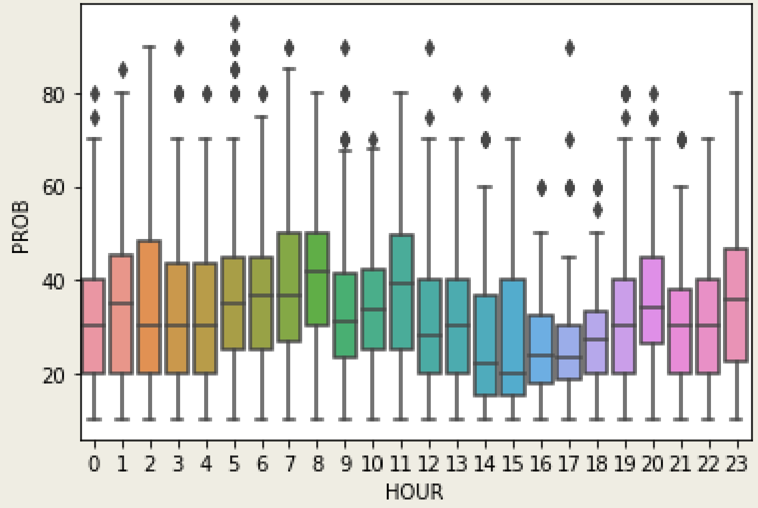




Data Visualization







Time Series Prediction

ARIMA (AutoregRessive Integrated Moving Average) is a model that can be fitted to time series data in order to predict future points in the series.

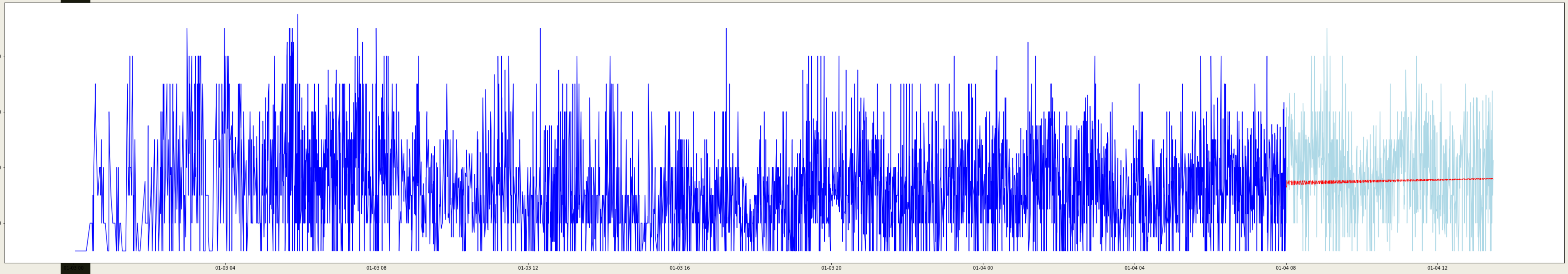
Prophet (created by Facebook) is a procedure for forecasting time series data based on an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality.

```
#separate training data before 01-04 and test data after 01-04
train = df.loc[: '2015-01-04 08:00:00']
test = df.loc['2015-01-04 08:00:00': ]

# ARIMA
model = ARIMA(train["PROB"], order=(4,1,7))
model_fit = model.fit(disp=0)
print(model_fit.summary())
predictions = pd.DataFrame(model_fit.forecast(len(test))[0])
predictions.index = test.index

#calculate rmse
rmse = np.sqrt(((predictions[0] - test['PROB'])**2).mean())
print(rmse)

#plot
ax = plt.axes([0, 0, 8, 2])
ax.plot(train["PROB"], color='blue')
ax.plot(test["PROB"], color='lightblue')
ax.plot(predictions, color='red', linewidth=1)

# plot residual errors
residuals = pd.DataFrame(model_fit.resid)
residuals.plot()
plt.show()
residuals.plot(kind='kde')
plt.show()
print(residuals.describe())
```

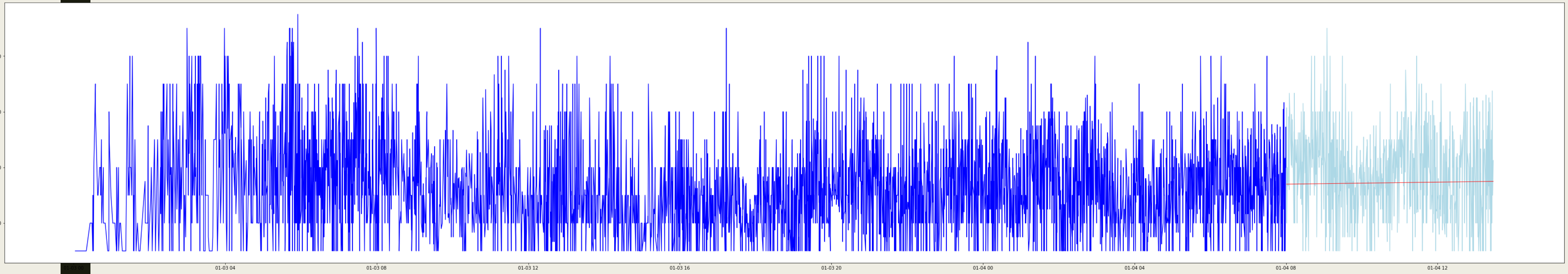
```
# Run fbprophet for prediction
# Prophet requires columns ds (Date) and y (probability)
train['ds'], train['y'] = train['DATE'],  train['PROB']
test['ds'], test['y'] = test['DATE'],  test['PROB']

# Make the prophet model and fit on the data
model2 = fbprophet.Prophet(changepoint_prior_scale=0.15)
model2_fit = model2.fit(train)

# Make predictions
predictions2 = model2.predict(test)['yhat']
predictions2.index = test.index

#calculate rmse
rmse = np.sqrt(((predictions2[0] - test['PROB'])**2).mean())
print(rmse)

#plot
ax2 = plt.axes([0, 0, 8, 2])
ax2.plot(train["PROB"], color='blue')
ax2.plot(test["PROB"], color='lightblue')
ax2.plot(predictions2, color='red', linewidth=1)
```

Model Results Based on Seconds

ARIMA

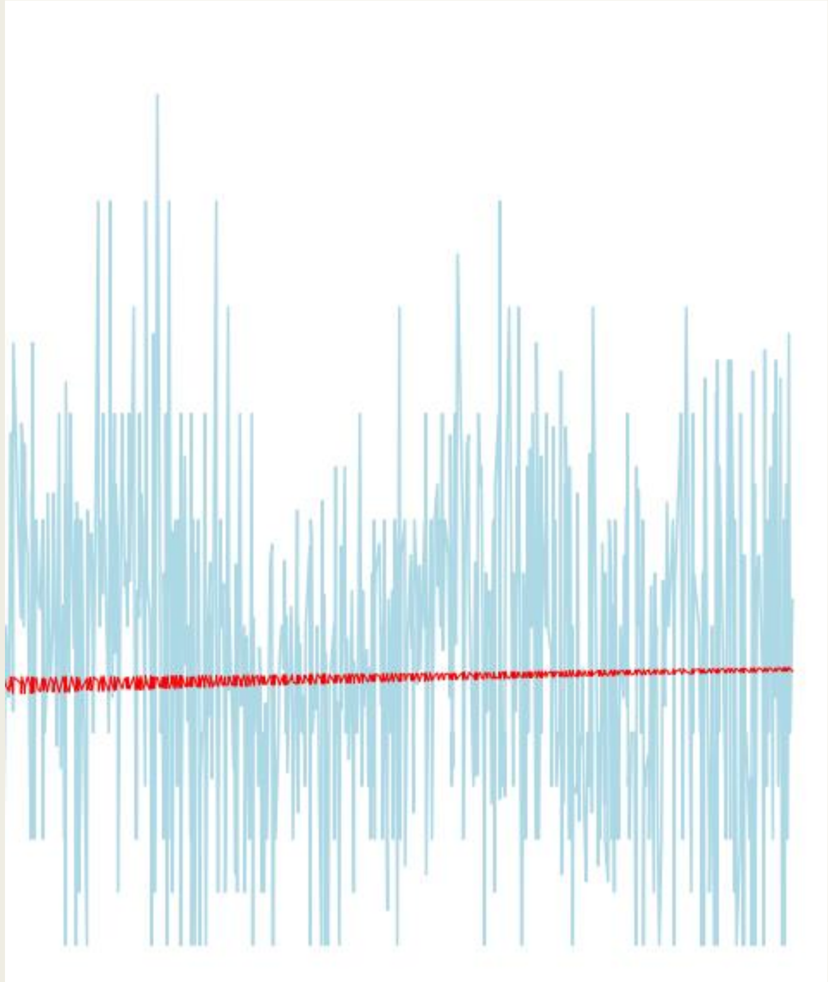


Prophet

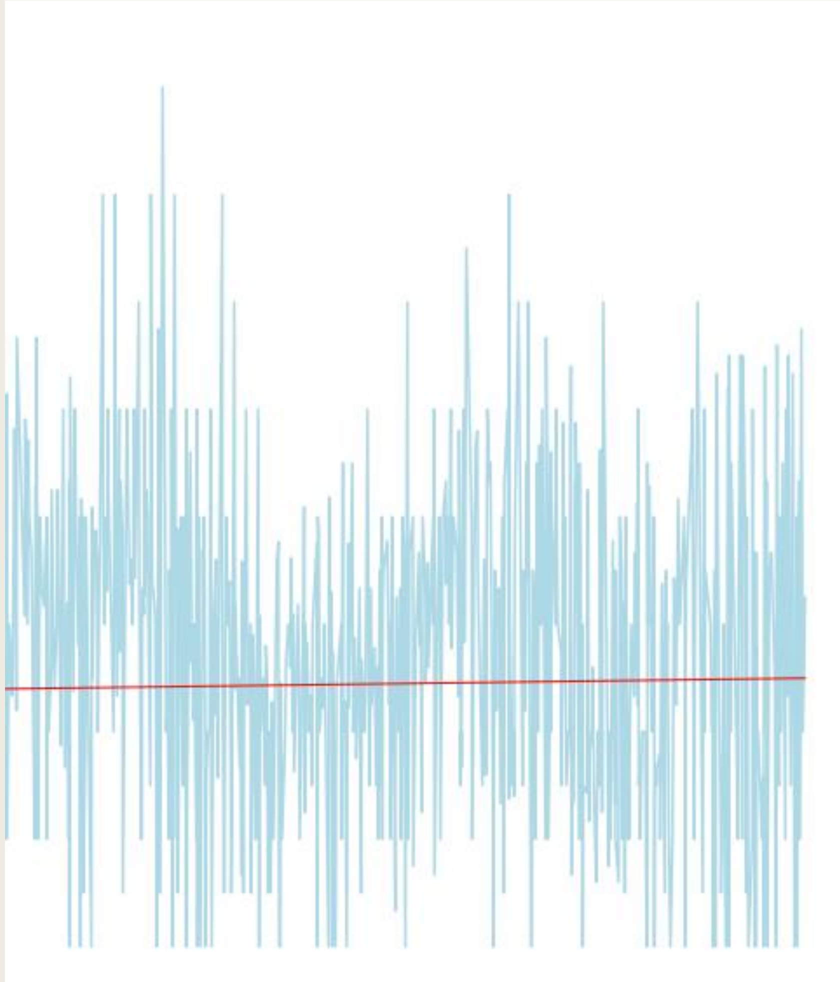


Model Results Based on Seconds

ARIMA



Prophet



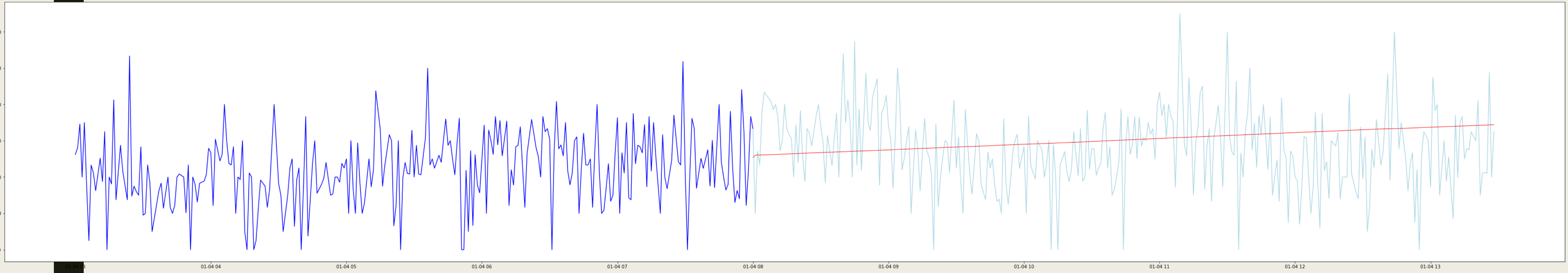
Analysis

Statistics	Value
Mean	33.5%
Standard Deviation	16.2%
Range	10%-95%
ARIMA RMSE	15.17%
Prophet RMSE	15.33%

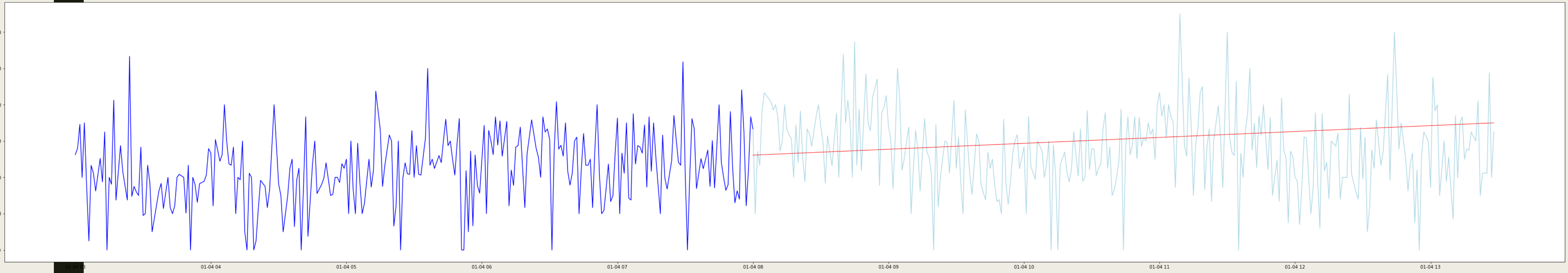
- There was approximately a ~15% error for both models
- Errors of models are only ~0.8% better than the error of a horizontal line at the mean

Model Results Based on Minutes

ARIMA RMSE: 11.61%



Prophet RMSE: 10.88%



Conclusion

- Limitations
  - *Total spatial consolidation - ie. hail data is being averaged for the entire US.*
  - *Data is only for January-April; subset is only for a couple of days in January.*
  - *Results may be more accurate for a longer sequence.*
  - *Data is noisy, time stamps for records are by the second.*
- Recommendations
  - *Smooth out prediction and resample data to hours or days.*
  - *Change parameters of models.*
  - *Change training and test date periods.*
  - *Run models for longer time period.*
  - *Future work: these methods could be duplicated to perform time series forecasting on different datasets.*

References

- Brownlee, Jason. “How to Create an ARIMA Model for Time Series Forecasting in Python.” *Machine Learning Mastery*, 3 May 2020, machinelearningmastery.com/arima-for-time-series-forecasting-with-python/.
- Brownlee, Jason. “What Is Time Series Forecasting?” *Machine Learning Mastery*, 21 Aug. 2019, machinelearningmastery.com/time-series-forecasting/.
- “Hail Basics.” *NOAA National Severe Storms Laboratory*, www.nssl.noaa.gov/education/svrwx101/hail/.
- NOAA. “Severe Weather Data Inventory.” *Kaggle*, 24 Oct. 2016, www.kaggle.com/noaa/severe-weather-data-inventory.
- NOAA's Severe Weather Data Inventory, www.ncdc.noaa.gov/swdi/.
- Lyla, Yang. “A Quick Start of Time Series Forecasting with a Practical Example Using FB Prophet.” *Medium*, Towards Data Science, 10 Jan. 2019, towardsdatascience.com/a-quick-start-of-time-series-forecasting-with-a-practical-example-using-fb-prophet-31c4447a2274.