# Orbital Insight: The Effect of Typhoons on Shanghai Shipments

## AI Studio Final Presentation

Break Through Tech New York @ Cornell Tech
December 7, 2022

# Introductions

# Our Team

**Kirsten Shyu**
Hunter College

**Ada Maldonado**
Brooklyn College

**Mariam Fondong**
Columbia University

**Seoeun Hong**
New York University

# Our AI Studio TA and Challenge Advisor

**Ananya Ganesh**
AI Studio TA

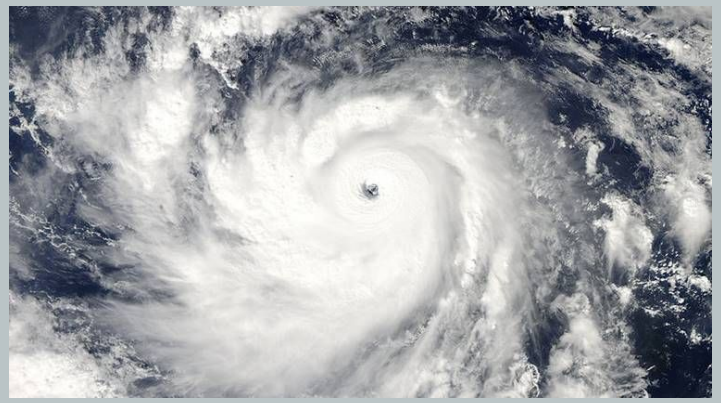
**Olivia Koski**
Advisor

# Presentation Agenda

1. Introduction
2. AI Studio Project Overview
3. Data Preprocessing
4. Model Selection and Evaluation
5. Final Thoughts
6. Questions

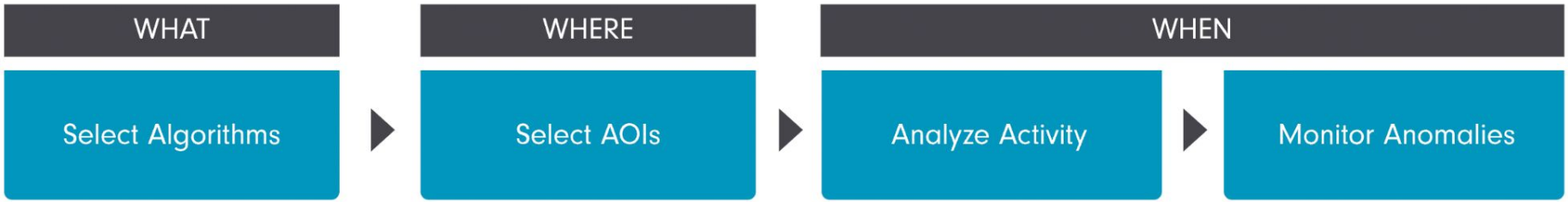
# AI Studio Project Overview

To build and test a model that will accurately predict dwell times of ships heading to Shanghai during and outside of typhoon season



# Business Impact

- Why we chose this goal:
  - Merchants and businesses need to know when shipments arrive
  - Shanghai is one of the busiest ports in the world
    - Delays here impact the global supply chain, making this important to understand
  - Ship dwell times are difficult to predict in typhoon-affected waters
  - See the effect that worsening typhoons (caused by climate change) could have on shipping
- Utilizes geo-spatial data rather than your typical dataset, AOI (drawing polygons, QGIS)
- Important economic impact, provides insight to businesses that can only be derived from AI



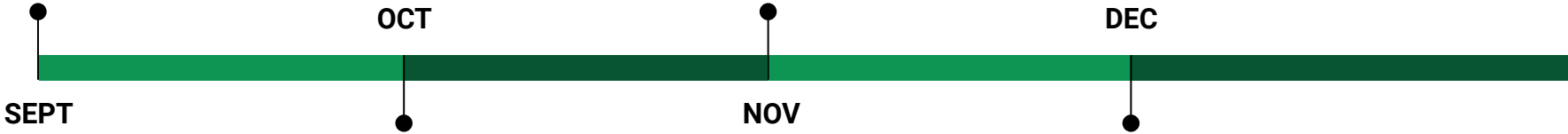
# Our Approach

Gather AIS data on the Shanghai Port

Dataset begins 2 months prior and 1 month following the typhoon

Use calculations to predict dwell times before and after the typhoon

Use dwell time predictions to make a general hypothesis of how storms will affect a port's average dwell time




Calculate the dwell time (how long a ship remains anchored at a port) of each ship

Break the time period down into hours and days

Subtract the hour of a ship's final ping at the port from first ping

Model, visualize, and evaluate our predicted results

See what we could do to improve our models/feature selections

# Resources We Leveraged



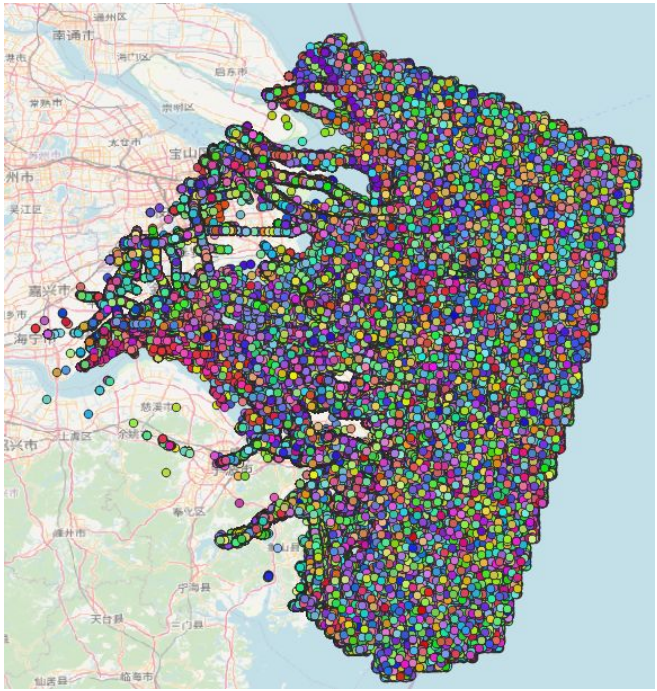
# Data Preprocessing

# Data Snapshot/Overview

- Polygon was manually drawn through QGIS
  - Demonstrates ship pings detected at Shanghai ports
  - Between June 1st to Sept 30th

- These pings are represented by dots
  - Each row in our AIS dataset represents one ping

- Color represents all the ships in this dataset
  - Assigned to each ping's unique device ID



# Data Transformation

- Converting Unix Time
- Adding Tonnage
  - (Height x Width x Drought)
- Calculating of Dwell Time

**Ship ID's: [Dwell Times]**

```
413816771: [62, 60, 5, 25],
413816801: [4, 11],
413817000: [52, 60],
413817288: [1, 1, 2, 3, 4, 3, 23, 22, 18, 37],
413817319: [1, 5],
413817368: [12],
413817675: [14, 22, 6, 61],
413817739: [10],
413817968: [8],
413818000: [45, 21],
413818939: [34, 8],
413818983: [22, 25, 64],
```

```python
sorted_arr = sorted_anchored['device_id'].unique() #unique device id
dwell_times = {}
for i in sorted_arr:
  current_time = sorted_anchored.loc[sorted_anchored['device_id'] == i][['hour']]
  array_hour = current_time.values.tolist()
  hour_len = len(array_hour)
  biggest_dif = 0
  trip_change = [0]
  list_hours = []
  for x in array_hour:
    list_hours.append(x[0])
  for j in list_hours:
    next_index = list_hours.index(j) + 1
    if(next_index < hour_len):
      biggest_dif = list_hours[next_index] - j

    if(biggest_dif > 12):
      trip_change.append(next_index)
  dwell = []
  for h in range(len(trip_change)):
    if((len(array_hour) != trip_change[h])):
      if(len(trip_change) == 2 and h == 1):
        dwell.append((list_hours[trip_change[h]-1] - list_hours[trip_change[h-1]]) + 1) #trip 1
        dwell.append(list_hours[-1] - list_hours[trip_change[h]] + 1) #trip 2
      elif(h + 1 == len(trip_change)):
        dwell.append(list_hours[-1] - list_hours[trip_change[h]] + 1)
      elif (h != 0):
        dwell.append((list_hours[trip_change[h]-1] - list_hours[trip_change[h-1]]) + 1)
  dwell_times[i] = dwell
print(dwell_times)
```

# Dwell Time

**What is the *Dwell time*?**

“Time spent in same position, area, stage of process”

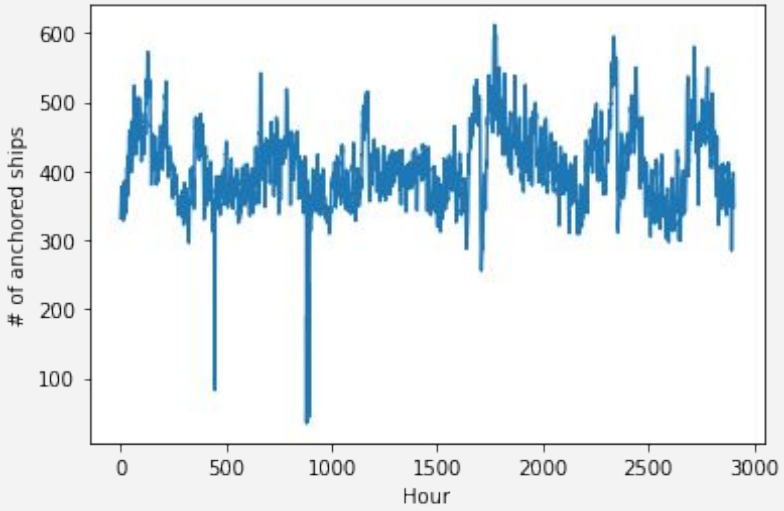
**How to Calculate It: (not that simple!)**

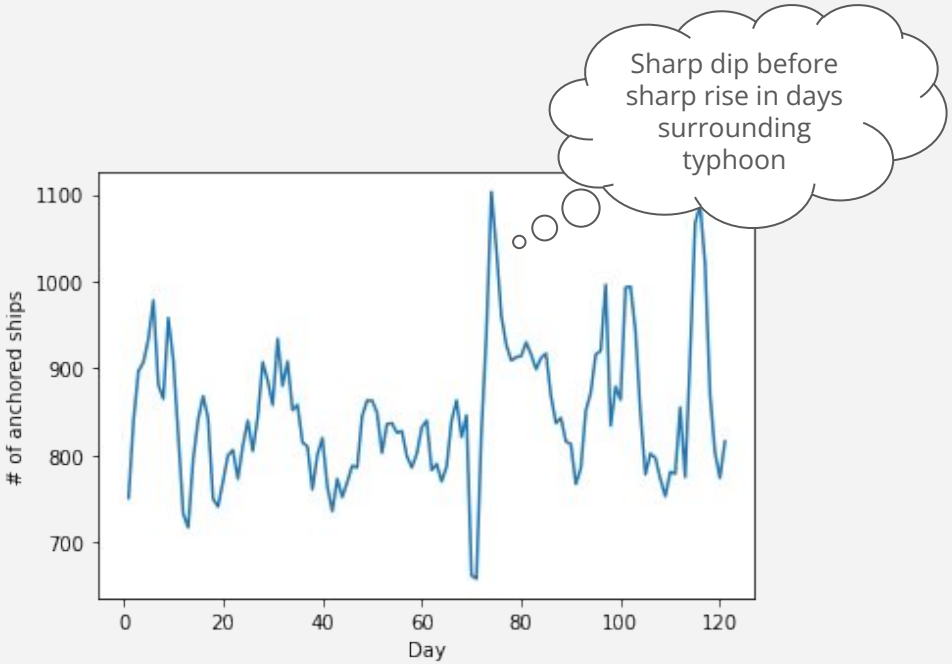
Last ping - First ping (for every trip)

**What we learned + things to consider:**

- Some ships had multiple trips
  - Differentiate between end of a first trip and start of another trip
- Creating a dictionary to store trip number and trip duration
- Nav status codes (anchored)

# Correlation Analysis




Typhoon Lekima

# Model Selection and Evaluation

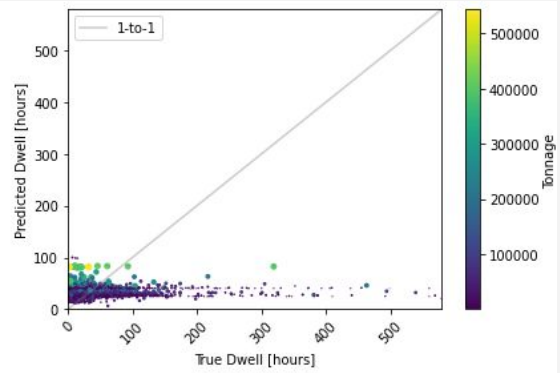
# Model Selection Process

- **Features:**
  - Vessel type code
  - Latitude & longitude
  - Typhoon Occurrence
  - Number of ships anchored
- **Label:** Dwell Time
- **Problem:** Regression
  - Supervised learning problem where the label is any real number
- **Our chosen models**:
  - linear regression
  - random forest
  - gradient boosting

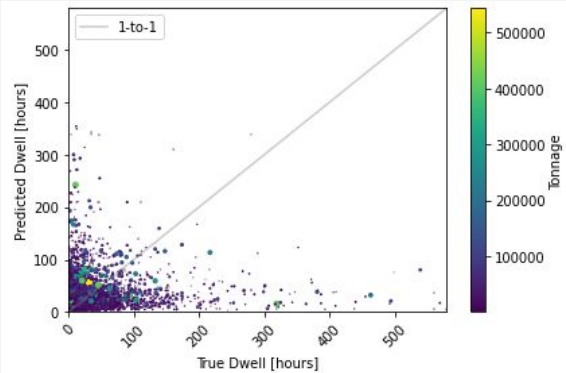
# Model Comparison

| Model Name | Description | Results | Pros | Cons |
|---|---|---|---|---|
| Linear regression | Model where straight line represents the data and proximity of points to line shows relationship between feature and label | Mean absolute error: **26.36<**<br>R^2: **0.01** | • Faster, more efficient, and simple<br>• Overfitting can be reduced | • Prone to underfitting<br>• Sensitive to outliers<br>• Assumes data is independent |
| Random forest regressor | Regression model that involves multiple decision trees contained within one set and combined into an ensemble method | Mean absolute error: **29.55**<br>R^2: **-0.34** | • High accuracy<br>• Easy data preparation<br>• Can handle larger data | • Biased towards more complex variables<br>• Little control over the model |
| Gradient boosting regressor | Iterative model that progressively refines its predictions by combining multiple decision trees | Mean absolute error: **26.34**<br>R^2: **0.00** | • Generally more accurate compare to other models<br>• Lots of flexibility | • Prone to overfitting<br>• Hard to interpret the final models |

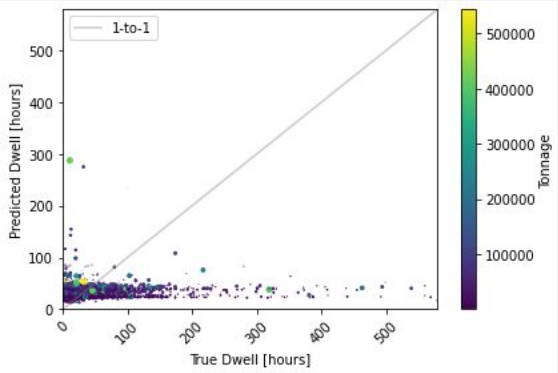
# Model Visualization




Linear Regression




Random Forest Regressor




Gradient Boosting Regressor

# Insights and Key Findings

- Final Model Selection -
  - Gradient Boosting Regressor
    - Lowest mean absolute error (only by a negligible amount)
- Mean Absolute Error is very high
  - Overfitted model
  - Features may not lead to accurate predictions
- Potential Solutions:
  - Spend more tinkering with features
  - Try using different/more datasets
  - Potentially look at approaching as classification problem

# Final Thoughts

# Trial and Error

**Lessons Learned:**

- How to decide our reasonable thesis and scale of our project
- How to deal with AIS data
- Deeper understanding of python libraries (Numpy, Pandas, Scikit-Learn) & ML tools (Google Colab)
- How important the data preparation process is
- How to format the data to be suitable for our Machine Learning model
- Why correct understanding of input and output feature is important
- How to work as a team
- Importance of understanding the business/economic impact of our project

**Takeaways:**

- Exposure to a new side of machine learning
- Gaining hands on experience for Python in a quick-paced learning environment
- To not be afraid to ask questions!

# Obstacles & Potential Next Steps

- Scheduling a time when we are all available
- Limited in-person meetings
- Beginner knowledge in Python
- New exposure to machine learning and data science
- Never done any industrial scale project before
- Geospatial data vs clean data

**Next Steps:**

- Scale up to different region or different typhoon
- Apply this model to other natural disaster
- Apply this model to different transportation such as truck, airplane, train, and so on
- Focus on a specific company? (Amazon, Ebay, etc)


Questions?