HACKtheMACHINE

September 21-23rd

Team: #WestCoastIsTheBestCoast

Joe Ortiz, James Eitelberg, Chris Chen, Jessica Kimball, Chris Cirullo, Thuy Nguyen-Cirullo (SSC PAC); Richard Brereton (COMNAVSURFPAC);
Rock Pereira (Consultant)

Track 2: Data Science and The Seven Seas

#WestCoastIsBestCoast Approach #1

- Clean-Up and Prioritization
  - Remove unneeded data (8 fields, e.g. IMO, width)
  - Filter erroneous data (e.g. Tug Boats, negative COG, moored, anchored vessels)
  - Divide data into 10-minute periods
- Processing and Visualization
  - Determine distance between vessels using k-d trees
  - Import data into ArcGIS map over maritime layers

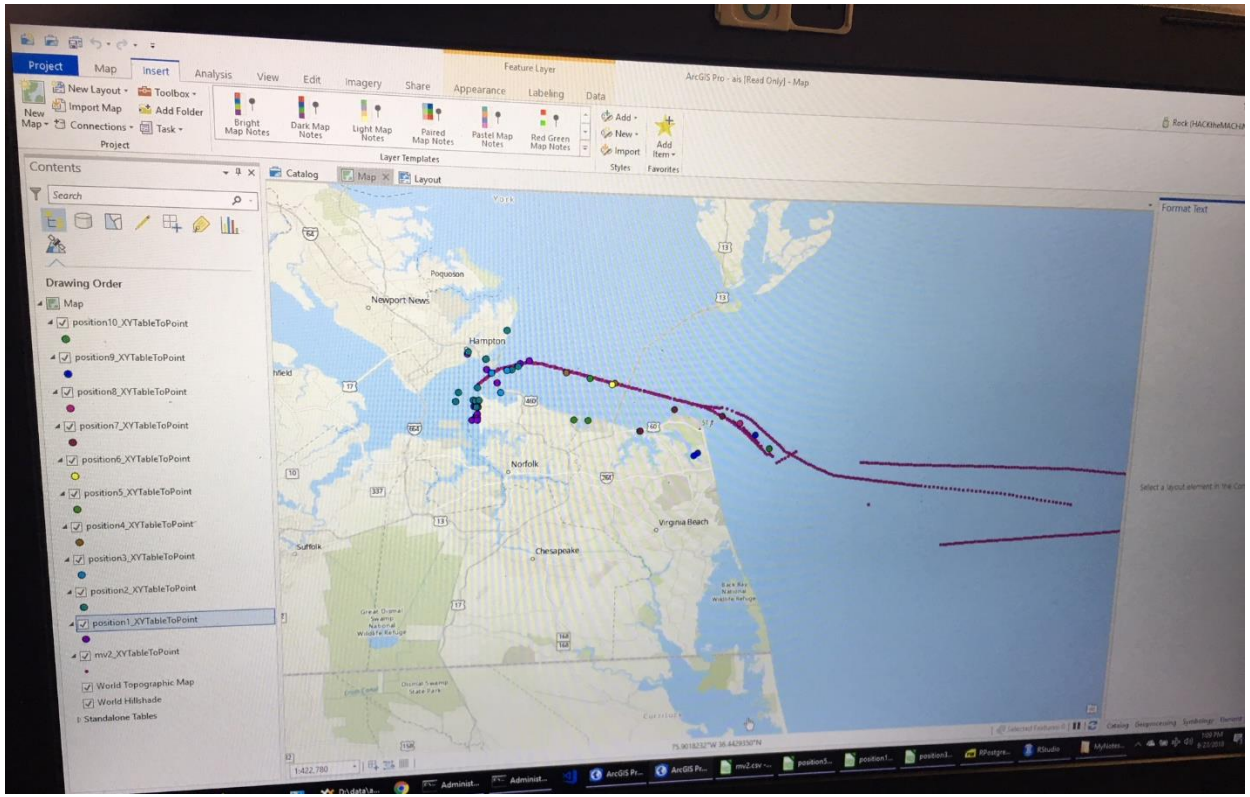
Results

- Computation time based on same data set:
  - Brute force: ~30/40 min
- Our machine learning based kd-tree algorithm we were able to improve performance to 10-15 min
  - > 50% improvement

#WestCoastIsBestCoast Approach #2

Simple geospatial calculation filtered by ship types

Follow that ship and create view frames



AIS data stored into AWS Cloud!

- Data for 3 years in Zone 10
- All zones for 1 month (DEC 2017)

Our Approach Challenge #2 - Continue

- Stage 2:
  - Preprocess each of the features:
    - For continuous variables, normalize it by subtract it by the means and divided by standard deviation
    - For categorical variables, encode it to a number
  - PCA and/or Factorial Analysis
    - To determine which features are the most important ones.
  - Multiclass Classification Algorithms.
    - ie: K-means Clustering, and etc.

UNCLASSIFIED

ArcGIS Pro - HACKTheMachine_Sample - output_5k.csv

Table View | Map Time | Standalone Table Data

Project | Map | Insert | Analysis | View | Edit | Imagery | Share

Jessica (SPAWAR)

Global | Local | Map | Convert | Link Views | Catalog Pane | Catalog View | Contents | Python | Tasks | Reviewer Rules | Workflow Manager | Create | Import | Add | Remove | Depth Priority | Drawing Mode | Navigator

View | Link | Windows | Thumbnail | Animation | Scene | Navigation

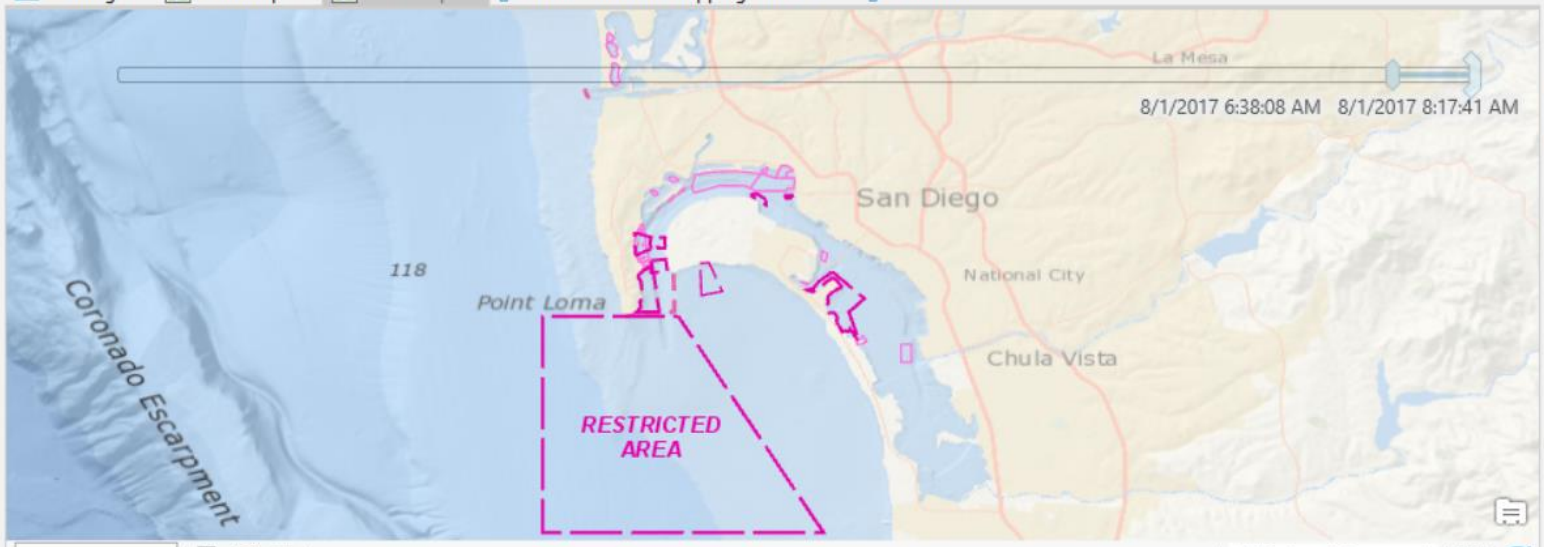
Contents

Search

Drawing Order

- OutliersInShippingSchemeLoc
- Outliers In Shipping Scheme
- AIS
  - AIS_Data_Zone16_Aug_2017_FC
- AIS_Data_Zone16_Aug_2017_FC_
- AOI
- Chart
- Chart Military
- World Ocean Base
- Standalone Tables
  - AIS_Data_Zone16_Aug_2017.csv
  - AIS_LA_SD_Jan_1_to_15_2016_Filtere
  - AIS_LA_SD_Jan_1_to_15_2016_Filtere
  - output_5k.csv
  - Results_5000
  - AIS_2016
  - AIS_2017_01_Zone11.csv

Catalog | Start Map | Final Map | Find Outliers In Shipping Scheme | Model 1



1:284,638 | 117.1269185°W 32.6065205°N | Selected Features: 1

output_5k.csv

Field: Add Delete Calculate | Selection: Zoom To Switch Clear Delete Copy

Field1	BaseDateTime	Ship1	Ship2	Distance
0	1/18/2017	338003172	338143881	195.78312
1	1/18/2017	338003172	367360430	2187.663557
2	1/18/2017	338003172	311000415	2145.102009
3	1/18/2017	338003172	367198110	1501.814615
4	1/18/2017 12:10:00 AM	338003172	338143881	196.999164
5	1/18/2017 12:10:00 AM	338003172	367373970	2173.071026
6	1/18/2017 12:10:00 AM	338003172	367360430	2360.341836

0 of *2000 selected | Load All | Filters: | 100 %

Geocode Table

Geocode Table

Guided Workflow Complete
Review your inputs below and run the toc

Input Table: AIS_2016

Input Locator: ArcGIS World Geocoding Service

Input Address Fields: Multiple Field

Locator Field	Data Field
Address or Place	LAT_1
Address2	LON_1
Address3	<None>
Neighborhood	<None>
City	<None>
Subregion	<None>
State	<None>
ZIP	<None>
ZIP4	<None>
Country	<None>

Output: AIS_2016_Geocoded

☑ Add output to map after completion

Preferred Location Type: Address Location

Catalog | Geocode Table

1:16 PM 9/23/2018

Jupyter Notebook Example

UNCLASSIFIED

GitHub, Inc. [US] | https://github.com/jessielulu/htm/blob/master/AnalysisAndClassification-Stage1.ipynb

In [129]:

```
# Perform KD-Tree Search on 10-minutes bucket
from sklearn.neighbors import KDTree
from collections import defaultdict


d = defaultdict(list)
for i in day_cleaned['BaseDateTime_Rounded'].unique():
    time_rounded = day_cleaned[day_cleaned['BaseDateTime_Rounded']== i]
    #df_latlon = time_rounded[['LAT', 'LON', 'MMSI']]


    tree = KDTree(time_rounded[['LAT', 'LON']])
    dist, ind = tree.query(time_rounded[['LAT', 'LON']], k=5)

#    print(time_rounded.iloc[0]['MMSI'])
#    break




    for k, v in enumerate(ind):
        ship1 = time_rounded.iloc[k]['MMSI']
        for j in v:
            if(k!=j):
                ship2 = time_rounded.iloc[j]['MMSI']

#                 print(k, j)
                #if (df_latlon.iloc[k][2] == df_latlon.iloc[j][2]):
                #    count += 1
                d[ship1].append([ship2, great_circle(time_rounded.iloc[k][2:3], time_rounded.iloc[j][2:3])
.feet/3, i ])


#a = dict(zip(pd.date_range('2010-01-01', '2010-01-01 23:59:39', freq='30T').time, range(48)))

#convert time column to times and map by dict
#df['a'] = df['time'].dt.time.map(a)
#print (df)
```

HACKtheMACHINE.pptx | AIS_2017_01_Zone11.zip | R-3.5.1-win.exe | ml_ai_workshop.zip.html Canceled | python-3.0.amd64.msi | neo4j-desktop-offl....exe | Show all

1:18 PM 9/23/2018