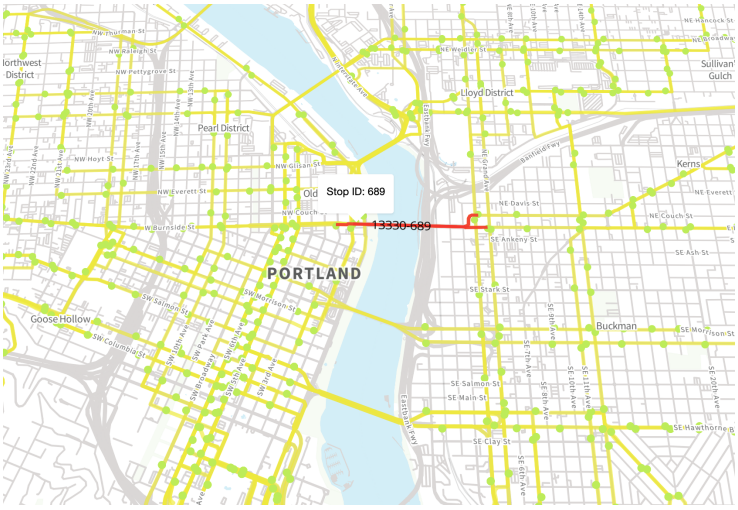
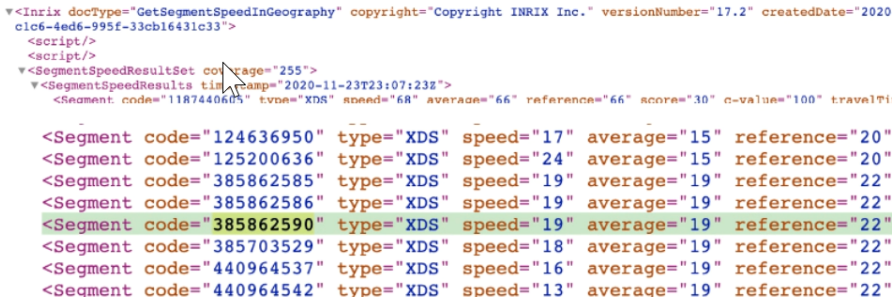
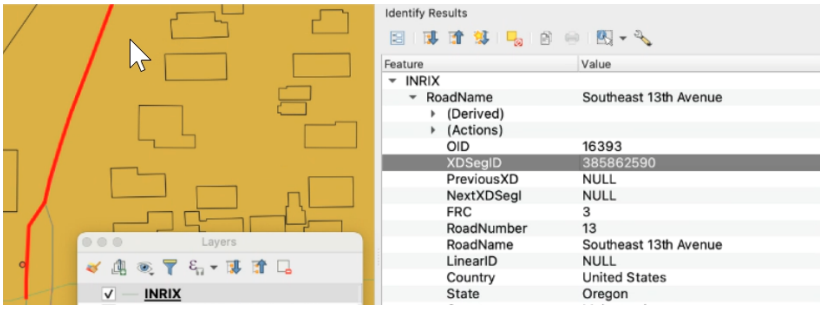
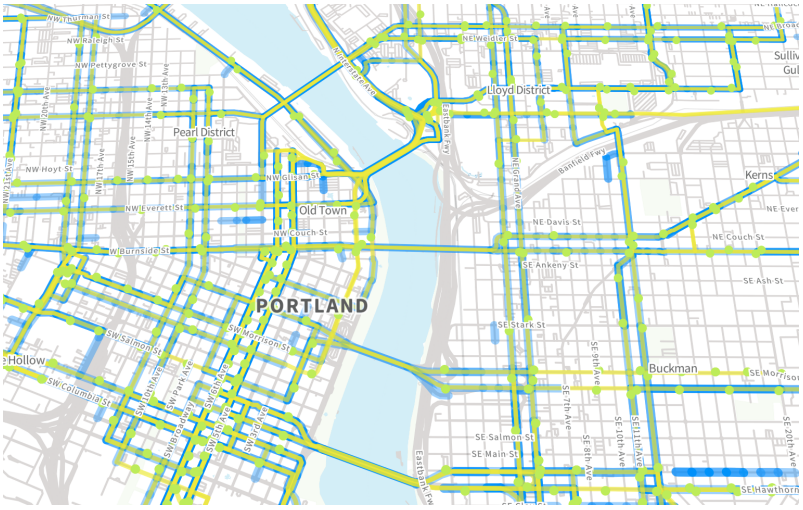
Collect Traffic Data

PRJ37-12 - Traffic / Speed data IN PROGRESS

Tasks	who, what, why, etc...

Use Machine Learning (ML) to help make better arrival predictions (e.g., *Tran sit Tracker*).	• Inspired by work done at TransLink in Vancouver B.C.
	TransLink ...ctions.pdf
	•
	• Guy is writing his own ML system, since the TransLink folks only published the above paper, and not the code or models used in the paper.
	• The stack, afaik:
	  • Postgres w/partitions (new partition each month ... millions of speed records imported ala 7k records every 5 minutes)
	  • scikit-learn, pandas and xgboost (and probably numpy, but not directly)
	• *The* missing input element from the model today is traffic and speed data.

Develop a system to collect traffic and speed data called *trafficdb*.	• INRIX was chosen as the traffic/speed data vendor, due to an existing ODOT license that gives TriMet full access. • The transit-speed data system we're developing needs to: a) ingest transit data, b) find the stop-to-stop transit segments and c) conflate those transit segments with the INRIX street segments. The gtfsdb fits the bill. • Thus, we're building a related project to gtfsdb called the trafficdb.
Start the *trafficdb* ingest process by finding all the stop-to-stop segments from the transit data (i.e., really the first thing is to load the transit data into trafficdb via existing functionality of the parent gtfsdb system)	This involves finding unique pairs of stops, when matched against the schedule's trip data: 

The transit data provider (INRIX in this case) has a set of street data with their segments.	INRIX publishes their street segments in GeoJSON format.  So we'll use ogr2ogr to load that data into trafficdb. 
Conflate the transit data with the transit stop segments above. Probably also need a map app to view the data at this point...	Since trafficdb is a spatially enabled PostGIS database, we can use routines (after a lot of experimentation and reading and hairpulling) to conflate the traffic segments with the stop segments. 

Develop code to authenticate with INRIX, and query their speed data.	1. sending down a uname & password to INRIX...
	
	`[inrix]`
	`traffic_url = http://na.api.inrix.com/traffic/Inrix.ashx`
	`vendorid = <your vid>`
	`consumerid = <your cid>`
	
	2. returns a temporary (~30 minutes) token
	
	`// http://na.api.inrix.com/traffic/Inrix.ashx?format=json&action=getsecuritytoken&ce6a9c3325fe&`
	
	`{`
	`"docType": "GetSecurityToken",`
	`"copyright": "Copyright INRIX Inc.",`
	`"versionNumber": "17.2",`
	`"createdDate": "2020-11-23T22:36:28Z",`
	`"statusId": 0,`
	`"statusText": "OK",`
	`"responseId": "41b75c8f-58fc-44b7-9b05-c5e7784fa160",`
	`"result": {`
	`"token": "0lLQnv3pDL0ryr7N4NpMgxU1FEaa9ebfiRqoIgQ46aU!",`
	`"tokenExpireDtUtc": "2020-11-23T23:35:00Z"`
	
	3. which then allows polling their speed data for the segments you're interested in
	
	`segments: 9981-8262 (SW 2383.6') -- 1236997384 (S 333.3'):`
	`2020-12-31 05:36am -- speed: 9.0 freeflow: 17.0 avg: 17.0 time: 0.439 (realtime is true with 90% confidence)`
	`2020-12-31 07:17am -- speed: 16.0 freeflow: 17.0 avg: 16.0 time: 0.237 (realtime is false with 0% confidence)`
	`2020-12-31 08:56am -- speed: 9.0 freeflow: 17.0 avg: 16.0 time: 0.410 (realtime is true with 90% confidence)`
	`2021-01-03 05:27am -- speed: 13.0 freeflow: 17.0 avg: 13.0 time: 0.293 (realtime is false with 0% confidence)`
	`2021-01-03 05:40am -- speed: 9.0 freeflow: 17.0 avg: 13.0 time: 0.410 (realtime is true with 90% confidence)`
	`2021-01-03 07:31am -- speed: 23.0 freeflow: 17.0 avg: 17.0 time: 0.166 (realtime is true with 0% confidence)`
	`2021-01-03 07:33am -- speed: 23.0 freeflow: 17.0 avg: 17.0 time: 0.166 (realtime is true with 0% confidence)`
	`2021-01-04 09:49am -- speed: 11.0 freeflow: 17.0 avg: 18.0 time: 0.362 (realtime is true with 90% confidence)`
	
	`segments: 9981-8262 (SW 2383.6') -- 440939848 (S 878.3'):`
	`2020-12-31 05:36am -- speed: 15.0 freeflow: 16.0 avg: 15.0 time: 0.653 (realtime is false with 0% confidence)`
	`2020-12-31 07:17am -- speed: 14.0 freeflow: 16.0 avg: 14.0 time: 0.713 (realtime is true with 90% confidence)`
	`2020-12-31 08:56am -- speed: 14.0 freeflow: 16.0 avg: 15.0 time: 0.682 (realtime is true with 90% confidence)`
	`2021-01-03 05:27am -- speed: 27.0 freeflow: 16.0 avg: 11.0 time: 0.365 (realtime is true with 0% confidence)`

We'll run our code every 5 minutes, and save off a historical set of stop-to-stop segment / speed data, which can now be fed into the ML models.	The stop-segment speed file we'll be creating every 5 minutes. 
Bonus Points: other agencies other traffic data sources (TBD) other uses?	Because the IMI project is funded by the FTA, the project must deliver **value** to other agencies and partners beyond TriMet. Thus, the trafficdb is able to load other agency's GTFS files, and absorb INRIX data for their region, just like we're doing for TriMet. Our code is both opensource, and also packaged via Docker to make deployment as light a lift as possible for others to operate. As time permits, I hope to at least have a minimal demo of another traffic vendor's data beyond INRIX. The speed data, conflated against transit data, has other uses beyond vehicle prediction models. Visually on a map, it could help to both show alongside our Realtime Vehicles that speed and traffic that vehicle is experiencing. Or maybe the speed data would help in animating those Realtime Vehicles smoothly on that map. Or maybe...


```
{
  "numSpeedRecords": 80,
  "speeds": [
    {
      "averageSpeed": 13.0,
      "beginStopId": "10751",
      "confidence": 100.0,
      "currentSpeed": 13.14,
      "date": 1610841526,
      "endStopId": "10752",
      "freeflowSpeed": 15.0,
      "hasSpeedData": true,
      "id": "10751-10752",
      "isRealtime": true,
      "lengthInFeet": 1275,
      "maxSpeed": 15.0,
      "minSpeed": 12.0,
      "numberOfLanes": 1.0,
      "travelTimeInMins": 0.395
    },
    {
      "beginStopId": "10752",
      "date": 1610841500,
      "endStopId": "10753",
      "hasSpeedData": false,
      "id": "10752-10753",
      "lengthInFeet": 1014
    },
    {
      "averageSpeed": 13.0,
      "beginStopId": "10752",
      "confidence": 100.0,
      "currentSpeed": 17.27,
```