Machine Learning for Space Situational Awareness

AMOS 2018 – Short Course

Code Examples

Outline

- Tooling (< 5m): Software and libraries for ML/DL
- Dimensionality Reduction (10m): PCA, TSNE, UMAP
- Supervised Learning (60m):
  - Tabular Data: TLE example with fully-connected network (MLP)
  - Time-Series Data: 1D-CNN and Recurrent Neural Network (RNN)
- Break (10m)
- Unsupervised Techniques (40m)
  - Transfer Learning
  - Self-Supervision: Auto-Encoders and Auto-Encoder Style Algorithms
  - Domain Adaptation on Synthetic Data

Disclaimer

- Examples are intended to help convey the workflow of a machine learning approach to a problem. They certainly aren't intended as optimal solutions to real problems.

- All code can be found here:
  - https://github.com/PJ7668/ml-for-ssa

Tooling for Machine/Deep Learning

- Primary Deep Learning platforms:
  - TensorFlow (Google)
  - PyTorch / Caffe2 (Facebook with support from Nvidia)
  - MXNet (apache project with Amazon support)

- Use whatever environment you feel productive in:
  - R (keras interface)
  - MATLAB (Neural Networking Toolkit)
  - Python (keras+tensorflow or pytorch)

- Hardware:
  - There may come a time you need a GPU, but you don’t need it to get started.
  - When you do, you’ll almost certainly be wanting Nvidia hardware running their CUDA libraries, which are well optimized for deep learning on GPUs.

Dimensionality Reduction

- ML tasks can have many possible input features.
- Deep Learning Rule of Thumb:
  - Avoid hand-selecting or engineering features.
  - When practical, give neural network access to raw data to identify its own combinations of features.
- However, using “all the features” is not always practical. Dimensionality reduction techniques can be useful for:
  - Visualization
  - Reduce the feature space

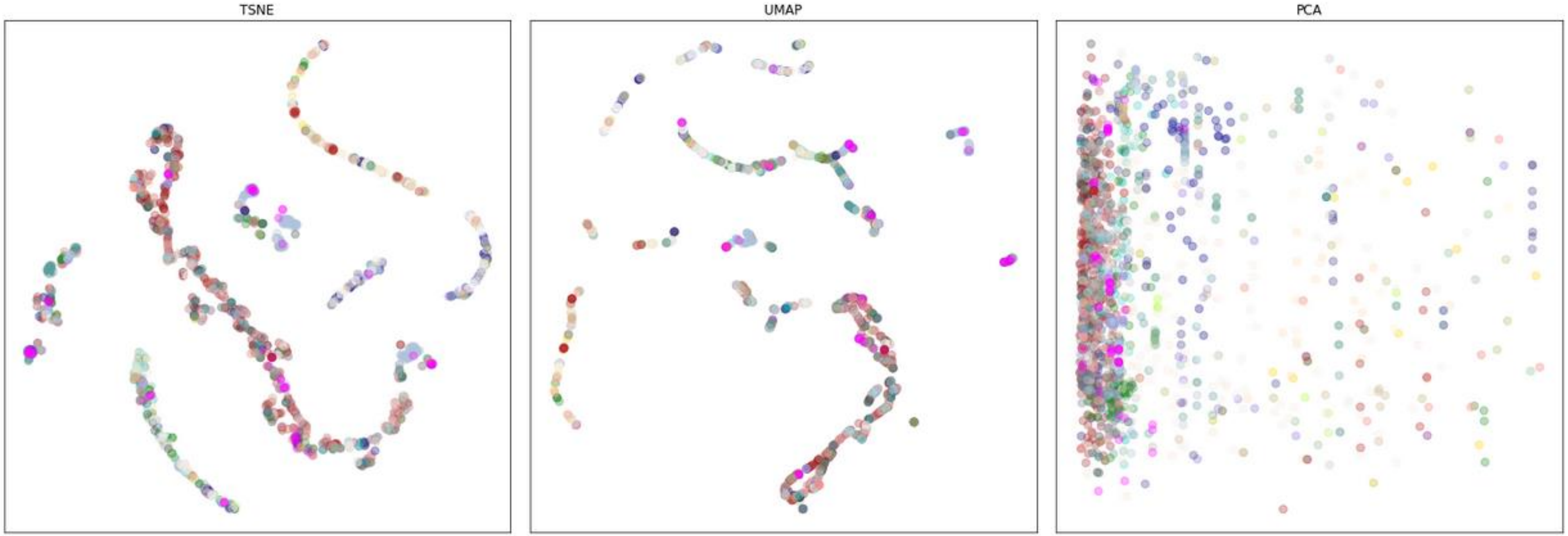
Dimensionality Reduction

arg_perigee	bstar	excentricity	group	inclination	mean_anomaly	mean_motion	mean_motion_derivative	orbit	right_ascension
344.9211	0.000000e+00	0.000220	ses	0.0117	192.3938	1.002720	-9.800000e-07	328	182.7004
240.3986	0.000000e+00	0.692405	science	70.7112	0.3731	0.378005	6.190000e-06	659	281.4812
47.2687	0.000000e+00	0.000032	intelsat	0.0269	160.3231	1.002700	3.900000e-07	5463	115.9600
174.5702	0.000000e+00	0.000298	intelsat	0.0329	192.7488	1.002723	-1.250000e-06	4866	352.6984
281.6462	0.000000e+00	0.000444	geo	0.0999	283.1222	1.002717	1.100000e-06	4522	275.0218
269.9895	2.424900e-05	0.001328	argos	99.1391	89.9755	14.123839	-2.000000e-08	68470	282.6112
307.8352	2.774000e-05	0.001298	education	97.5885	52.1692	14.908850	2.200000e-06	6182	144.7883
238.0707	0.000000e+00	0.000171	geo	0.0623	265.3074	1.002718	7.000000e-07	4721	280.2496
142.3965	3.386900e-05	0.001046	amateur	98.6875	217.7946	14.221306	3.100000e-07	78718	253.0090
14.1533	-3.579400e-08	0.001436	resource	6.0029	345.9173	14.765825	6.400000e-06	15832	324.8697
102.8129	4.843400e-05	0.000960	planet	97.4436	257.4181	15.246237	1.124000e-05	8380	309.6436
57.8069	0.000000e+00	0.006835	sarsat	56.4768	302.9085	2.005516	-5.100000e-07	8640	326.5501
83.0369	1.881400e-05	0.000260	iridium-NEXT	86.3968	277.1123	14.342186	7.200000e-07	2241	39.1703
129.1498	2.141600e-05	0.000184	planet	97.4114	230.9901	15.212770	4.100000e-06	4644	358.5258
63.5433	1.587800e-04	0.000271	cubesat	51.6345	296.5842	15.711176	1.988000e-04	6587	330.6930
209.0278	0.000000e+00	0.001739	glo-ops	64.1831	216.3147	2.130987	3.000000e-07	2923	289.8220
67.7021	0.000000e+00	0.000038	sarsat	3.0717	53.9275	1.002744	9.000000e-08	4655	71.6448
9.0699	2.485900e-05	0.001057	noaa	98.7677	351.0670	14.258733	1.400000e-07	5604	262.0679
307.4328	2.095800e-05	0.001307	engineering	97.5881	52.5701	14.908546	1.530000e-06	6181	144.6987
84.8071	4.285600e-05	0.000222	iridium	86.4018	275.3378	14.342184	1.400000e-06	7515	165.6108

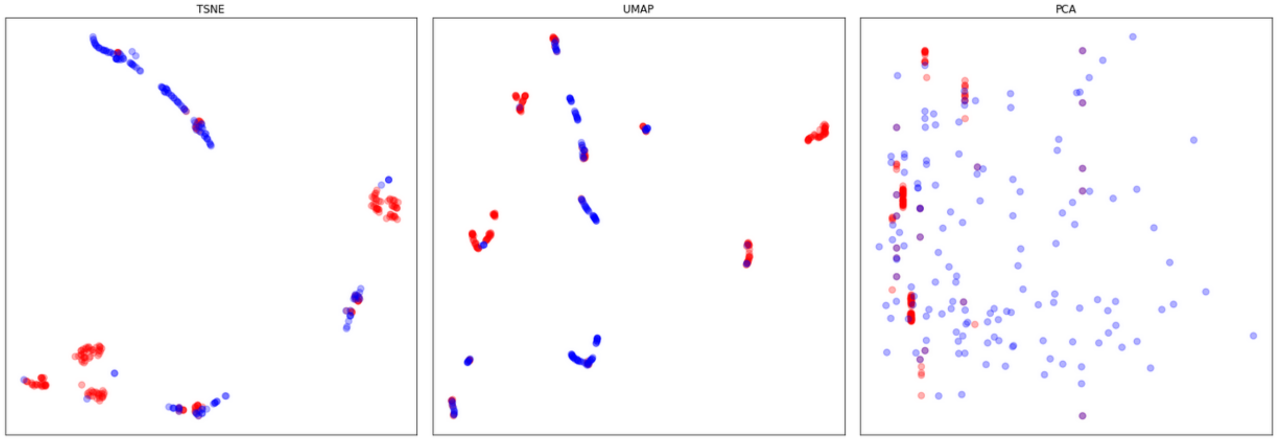
Sample TLE data.

9 real valued fields
1 categorical field (group)

“group” comes from the way CelesTrak categorized the given TLE.



- 2D embeddings of the 9D feature vectors
- Group field mapped to color
- (right) Just “planet” and “resource” groups



The visualizations suggest we should be able to make some better-than-random guesses about the group field based just on the numeric fields.

Classification

Goal: Predict group field based on 9 TLE derived, numeric fields.

Machine Learning Work Flow

1. Split data into train, test, and validation sets
2. Choose a metric (preferably a single, real-valued number)
3. Choose a model
    1. Train model on train set
    2. Compute metric on test set
    3. Adjust model and repeat
4. Evaluate model on validation set

Vocabulary

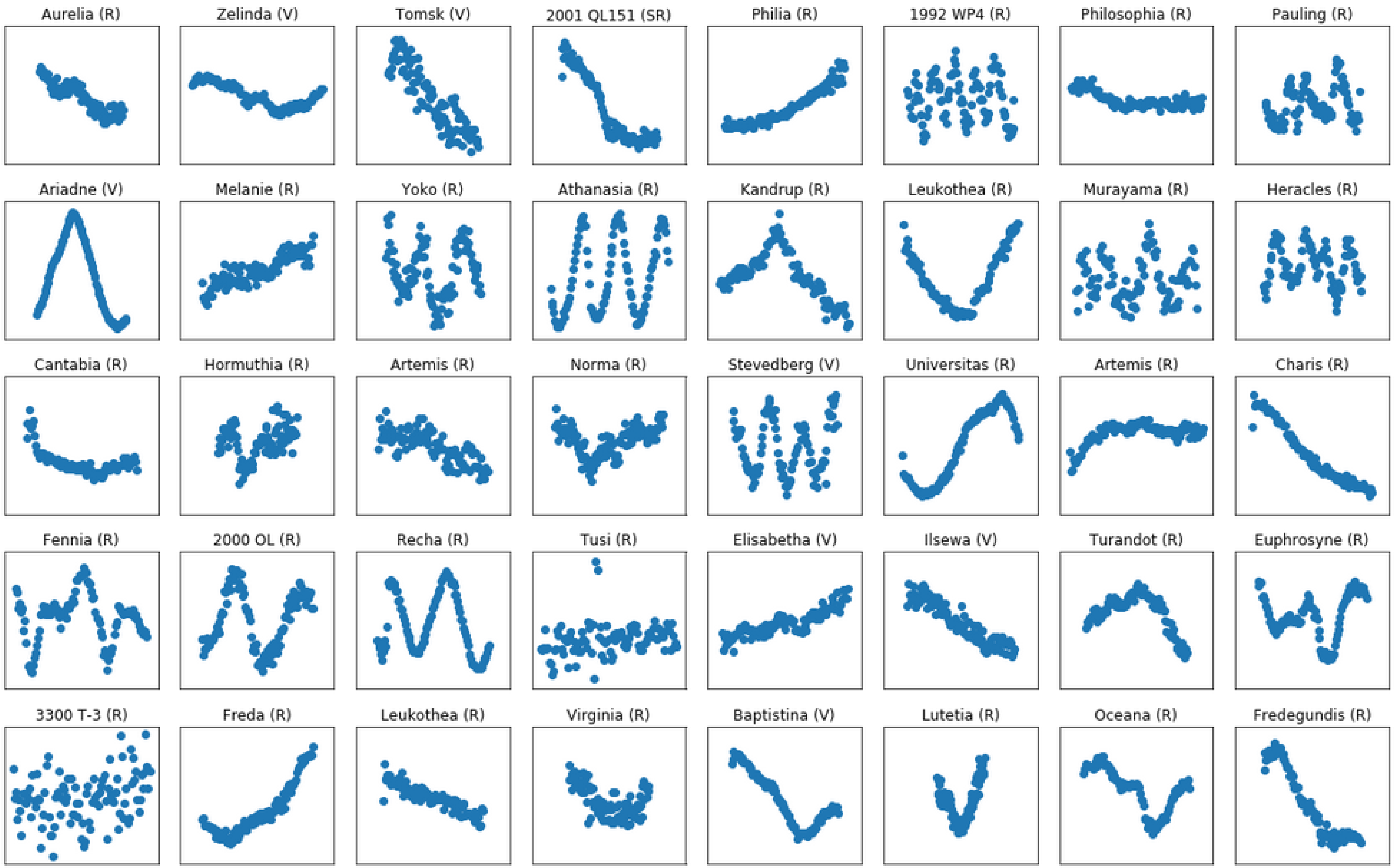
- Train/Test/Validation Sets:
  - Train: Data used to update model parameters.
  - Test: Data used to update model hyper-parmeters.
  - Validation: Data used to measure performance of selected model.
- Training Vocabulary:
  - Epoch – one full pass through test train set
  - Batch Size – how many examples to evaluate at once. Typically, a gradiant is computed on this data.
  - Loss – *differentiable* function comparing predictions and targets: model(x) vs y.
  - Stochastic Gradient Descent (SGD) – the algorithm for efficiently updating model parameters by computing partial derivatives of loss(model(x), y) over a batch of samples.

Classification

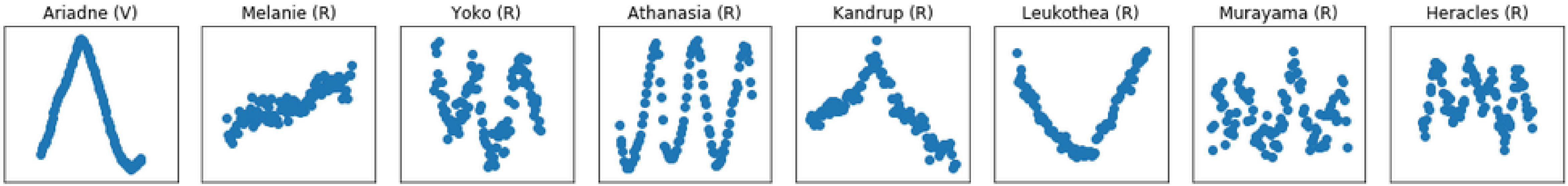
Rules of Thumb:

- Avoid hand-selected/engineered features (at least initially)
- Split train/test data based on some deterministic criteria
  - For example, test samples are precisely those for which SAT_ID % 5 == 0. This helps to avoid accidently letting the same or equivalent examples fall into your train and test sets.
- Be very careful with metrics and class imbalance.
  - For example, if 95% of your data belongs to the same class, reporting a 96% accuracy needs some qualifications.
- Identify a single-number metric to help simplify model comparisons.
- Normalize your feature data.
- Focus on the simplest version of a problem first and avoid extensive hyperparameter optimization until you can demonstrate that the most basic thing is working.
- Make sure the model is learning the training data before even worrying about the test data. If your model can't even memorize the training data well, either your problem is going to work or something else is broken.

Convolutional Neural Networks



Example light curves from alcdef.org: Asteroid Lightcurve Photometry Database



- Goal: Recognize asteroids based just on their light curves.
- Our features are translationally invariant in the sense that:
  - If we happened to start collecting the light curve a few seconds earlier or later, then we would still be looking at the same object.
  - Likewise, if there was some property near the beginning of the light curve that was important for asteroid recognition, the same property located in the middle or near the end, should be just as important.
- This is the motivation behind convolutional neural networks.

1D – CNN Diagram Goes Here