Machine Learning Engineer Nanodegree

Capstone Proposal

Predictive Maintenance

Ross Green
March 16th, 2018

Domain Background

Predictive maintenance is a domain where data about an asset such as sensor data, its maintenance and operational history is collected over time to monitor the state of an asset with the goal of finding patterns in the degradation of the asset over its lifetime. The goal is then to use this to predict failure conditions of an in-service asset, which can be used to inform actionable insights such as improved planning about the maintenance of the asset or stopping the asset if an imminent failure is predicted.

This is based on the Azure predictive maintenance template https://gallery.azure.ai/Collection/Predictive-Maintenance-Template-3.

Motivation:

I work in the resources industry which is heavily reliant on a functional mobile fleet, where the impact of unscheduled downtime is very costly. The ability to predict the Remaining Useful Life of an asset could provide huge savings by reducing unscheduled downtime and better planning around the maintenance of an asset to avoid premature change-outs or running the asset until failure.

Problem Statement

The question I am trying to solve is “Given an engines operational and failure events history, can I predict when an in-service engine will fail”

This can be formulated as:

Predicting the Remaining Useful Life (RUL) of the asset. This is predicting how much longer an in-service asset has before it fails. Since this is a continuous variable this is a regression problem.

Predicting if an asset will fail within a certain timeframe. For example, will it fail within the next 30 cycles. Since this is predicting between 2 classes (will fail in 30 cycles, wont fail within 30 cycles) this is a binary classification problem.

Predicting if an asset will fail within different time windows. For example, will it fail within the next 15 cycles, will it fail within the next 15 to 30 cycles, or will it fail after 30 cycles. This is a multiclass classification problem.

Datasets and Inputs

I will use the publicly available Turbofan Engine Degradation Simulation Data Set available from:

- Training Data: http://azuremlsamples.azureml.net/templatedata/PM_train.txt
- Test Data: http://azuremlsamples.azureml.net/templatedata/PM_test.txt
- Ground Truth Data: http://azuremlsamples.azureml.net/templatedata/PM_truth.txt

The data contains engine run-to-failure operational time series data, with cycle as the time unit along with 3 operational settings and 21 sensor values for each cycle.

The dataset is made up of training, test and ground truth datasets. The training data contains around 20,000 rows, with data for 100 unique engine ids and provides data up until the engines failure, where the last cycle is the failure point. The test data contains around 13,000 rows for 100 unique engine ids and has the same schema, but provides no indication of the failure point of the engine, so the last cycle is not the failure point. The ground truth data gives the number of remaining cycles for each engine in the test data.

Sample training data

~20k rows,
100 unique engine id

id	cycle	setting1	setting2	setting3	s1	s2	s3	...	s19	s20	s21
1	1	-0.0007	-0.0004	100	518.67	641.82	1589.7		100	39.06	23.419
1	2	0.0019	-0.0003	100	518.67	642.15	1591.82		100	39	23.4236
1	3	-0.0043	0.0003	100	518.67	642.35	1587.99		100	38.95	23.3442
...	...										
1	191	0	-0.0004	100	518.67	643.34	1602.36		100	38.45	23.1295
1	192	0.0009	0	100	518.67	643.54	1601.41		100	38.48	22.9649
2	1	-0.0018	0.0006	100	518.67	641.89	1583.84		100	38.94	23.4585
2	2	0.0043	-0.0003	100	518.67	641.82	1587.05		100	39.06	23.4085
2	3	0.0018	0.0003	100	518.67	641.55	1588.32		100	39.11	23.425
...	...										
2	286	-0.001	-0.0003	100	518.67	643.44	1603.63		100	38.33	23.0169
2	287	-0.0005	0.0006	100	518.67	643.85	1608.5		100	38.43	23.0848

Sample testing data

~13k rows,
100 unique engine id

id	cycle	setting1	setting2	setting3	s1	s2	s3	...	s19	s20	s21
1	1	0.0023	0.0003	100	518.67	643.02	1585.29		100	38.86	23.3735
1	2	-0.0027	-0.0003	100	518.67	641.71	1588.45		100	39.02	23.3916
1	3	0.0003	0.0001	100	518.67	642.46	1586.94		100	39.08	23.4166
...	...										
1	30	-0.0025	0.0004	100	518.67	642.79	1585.72		100	39.09	23.4069
1	31	-0.0006	0.0004	100	518.67	642.58	1581.22		100	38.81	23.3552
2	1	-0.0009	0.0004	100	518.67	642.66	1589.3		100	39	23.3923
2	2	-0.0011	0.0002	100	518.67	642.51	1588.43		100	38.84	23.2902
2	3	0.0002	0.0003	100	518.67	642.58	1595.6		100	39.02	23.4064
...	...										
2	48	0.0011	-0.0001	100	518.67	642.64	1587.71		100	38.99	23.2918
2	49	0.0018	-0.0001	100	518.67	642.55	1586.59		100	38.81	23.2618
3	1	-0.0001	0.0001	100	518.67	642.03	1589.92		100	38.99	23.296
3	2	0.0039	-0.0003	100	518.67	642.23	1597.31		100	38.84	23.3191
3	3	0.0006	0.0003	100	518.67	642.98	1586.77		100	38.69	23.3774
...	...										
3	125	0.0014	0.0002	100	518.67	643.24	1588.64		100	38.56	23.227
3	126	-0.0016	0.0004	100	518.67	642.88	1589.75		100	38.93	23.274

Sample ground truth data

100 rows

RUL
112
98
69
82
91

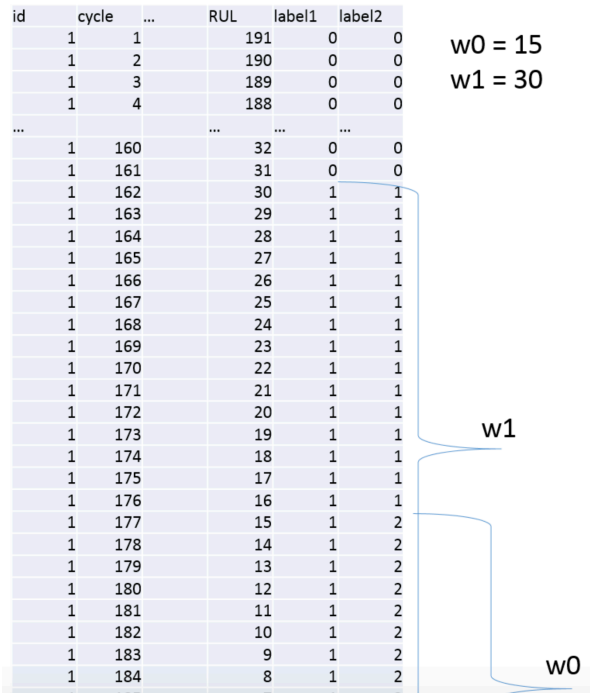
Solution Statement

I will be formulating a solution which will predict the 3 problems as stated above:

- Predict Remaining Useful Life (RUL): Regression
- Predict failure within time period: Binary Classification
- Predict failure in multiple time windows: Multiclass Classification

This will first involve creating the labels I am trying to predict based on the assets current cycle, including:

- RUL: The number of cycles remaining until the engine fails
- Label1: Is the current cycle within w1 cycles of failing
- Label2: Is the current cycle within w0, [w0, w1], >w1 cycles of failing



Using the input data for each cycle (sensor information, sensor aggregations, etc.) I will train a Regression model against the RUL label, and Classification Models against Label1 and Label2 labels. I can then use these models to predict the RUL, Label1 and Label2 of the test data for the last cycle of each engine, and compare the predictions to the Ground Truth to measure the models accuracy.

Since the data is time series based, I will use a Long Short Term Memory (LSTM) deep neural network, which are proven to be good at learning from sequences of data, since the model is able to look back at a period of data and find failure patterns within the data.

I will compare the performance of LSTM network to another type of model such as a Random Forest Classifier / Regressor.

Benchmark Model

A common problem in predictive maintenance classification problem is the large imbalance of failure events, since generally an in service asset will not fail within a time period. This makes testing the accuracy of the model tricky, as simply predicting that it wont fail every time, regardless of the input data, will likely give a large accuracy. Therefor it is important to measure the precision and recall of the model to measure the accuracy of predictions when there is an imminent failure.

- Regression: Simple Linear Regression Model
- Classification: Model that always predicts 0 (won’t fail)

Evaluation Metrics

A common problem in predictive maintenance classification problem is the large imbalance of failure events, since generally an in service asset will not fail within a time period. This makes testing the accuracy of the model tricky, as simply predicting that it wont fail every time, regardless of the input data, will likely give a high accuracy. Therefor it is important to measure the precision and recall of the model to measure the accuracy of predictions when there is an imminent failure.

For this type of scenario, it is more important that any imminent failure events are correctly predicted, than for a non imminent failure event to be incorrectly classified as a failure event (false alarm) since the money lost in running an asset until failure is worse than the inconvenience of investigating a failure when its actually OK.

The accuracy of the RUL prediction will be based on the difference between the number of remaining cycles predicted, and the actual number of remaining cycles.

- RUL: R2 Score, Root Mean Square Error, Mean Absolute Error
- Label 1: Accuracy, Precision, Recall, F1 Score
- Label 2: Accuracy

Project Design

1. Data Ingestion and Exploration:

Visualise and understand the data, look for patterns in the sensor data that could indicate degradation of the asset.

2. Date Pre-processing and Feature Engineering:

Create the labels (RUL, Label1, Label2), create aggregate features which summarise the sensor data such as rolling mean, rolling standard deviation, differences between sensor data, etc.

Normalise the data, using MinMaxScaler so all values are between 0 and 1

3. Model Building

Assess which models will be most suitable.

- A model such as Random Forrest to predict RUL based on the current cycles input data including the aggregated features. Will probably use Scikit-Learn libraries to implement this.
- An LSTM network to predict the RUL based on a given a sequence of cycle data, e.g. the previous 30 cycles of sensor data. This might have the advantage of creating its own features and aggregations over the cycle window which could be more effective then the ones created in the feature engineering stage. Will probably use Keras with TensorFlow backend to implement this.

Experiment with hyperparameters to find the best performing model. Use GridSearchCV where applicable.

4. Evaluation

Assess the performance of the models on the test data.

- Do they accurately predict the RUL of an in-service asset?
- Would they be useful in an organisation to predict issues that lead to failures and assist in maintenance planning?