http://terra-ai.sg
https://ai4impact.org
hi@ai4impact.org

AI Workshop – Lab 4

Feature Engineering with Neural Networks

Copyright (2019) by Terra Weather Pte Ltd. All rights reserved.

Lab Summary

- What is Feature Engineering?
- Creating the features and training the models.
- Discussion of results.

Introduction

"Feature Engineering" is a significantly different approach to prediction. It requires more work and is often trial-and-error. But when it works, it can be very effective. This method can also be used with non-NN methods like decision trees.

The key difference between NNs with feature engineering is in the input stage, not the NN models themselves. Previously, our "pipeline" was:

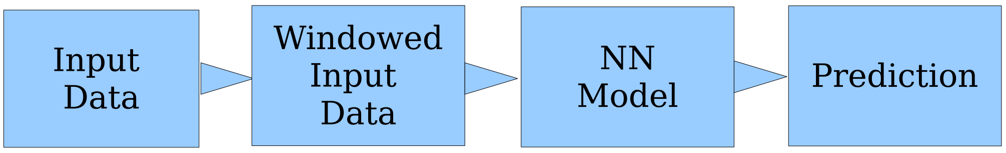

Figure 4.1: Prediction pipeline using "windowed" input.

Previously, as shown in Figure 4.1, we simply "windowed" the incoming data:

... y(-2), y(-1), y(0), y(1), y(2) ...

to get pairs of an "input" window of size W (ie, a vector {y(0), y(-1)....y(-W+1)}) and a "label" y(t+lead-time). We've used lead-time = 5 steps, so the labels are y(t+5). So, for t=0,1,2, the input & labels:

time step (t)	**input window (W=3)**	**label y(t+5)**
0	y(0), y(-1), y(-2)	y(5)
1	y(1), y(0), y(-1)	y(6)
2	y(2), y(1), y(0)	y(7)

and so on.

In the Feature Engineering approach (Figure 4.2), we **transform** this window of input into a new set of inputs, before feeding it into the network (for both training/testing and also in deployment).

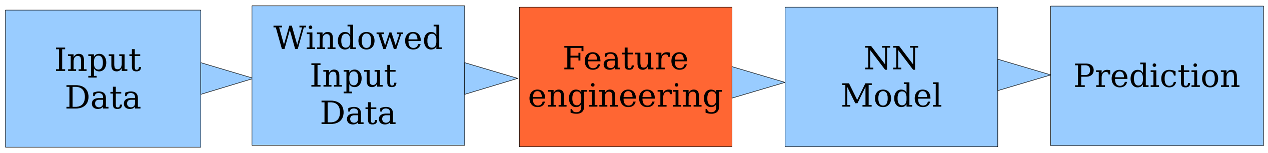

Figure 4.2: Prediction pipeline with feature engineering stage.

In Lab 3, the window size was fixed to W=16, so the previous 16 values prior to the present (ie, t=0) were used to predict the value at t+5.

In the feature engineering approach, we transform the input window into a **set** of "features". This could be anything – the average, variance, maximum/minimum etc. The more the features, the better.

As a rule of thumb, the features should **not** be derivable as a linear combination of other features. For example, we should **not** have a features f,g & h related as:

$$f = 3.5g \text{ (bad!)}$$

$$h = 2f + 7.4g \text{ (bad!)}$$

the reason being that such combinations slow down learning for the neural network. Just some common features to try are:

- min / max / max – min ("range") of the window,
- moments – mean, variance, skewness, kurtosis, etc.,
- percentile values (eg, 5%, 10%, 50% – median, etc).
- non-linear transformations (eg thresholds, log, exp, powers),
- the number of values above/below a threshold,
- the largest "run" – consecutive values above a threshold,
- autocorrelations at fixed lags,
- power spectra (eg from a Fourier transformation of the window),
- output of an auto-encoder,
- Combining these features (eg, average values above/below a threshold)

Copyright (2019) by Terra Weather Pte Ltd. All rights reserved.

The goal is to try using the simplest features and incrementally add features to help improve forecast accuracy. Don't try to do it all at once, because NN performance degrades with “noise” – signals that don't contribute to the prediction.

Predicting Differences

For this type of forecasting, it may help to predict the difference between y(t) and y(t+lead-time), rather than predicting y(t+lead-time) directly as we've done before. To recover y(t+lead-time) we simply add the predicted difference to y(t). We'll take this approach in this lab.

Quiz: *Why predict differences rather than the value y(t+lead-time) directly? Could this method also benefit the previous “direct” extrapolation technique we used in Labs 2 & 3?

Open up Lab 4's **pre.m** file.

IMPORTANT: The function of **pre.m** is to create new features for your model to use as input. You should edit this file to add or remove new features. Review the commented section in pre.m that tells you all the feature words available for use.

We're no longer using sine wave data, but actual weekly average temperatures over Singapore. This is a live data stream updated monthly.

Step 1: Without making any changes to pre.m, save the data, train the model and plot the losses as usual. What was the best test loss obtained?

Step 2: Take a close look at pre.m. Add in the “range” feature into the **transform** word there. Save the as usual data and retrain. Do your models improve their performance? Should the ordering of the features make a difference to performance? Remember if you are using an SGD solver, you may need to do multiple tries. You can do this with the **repeats** word in config.m. For example, adding the line:

```
5 repeats
```

Will repeat each configuration 5 times.

Copyright (2019) by Terra Weather Pte Ltd. All rights reserved.

Step 3: *From the list of possible features given on page 3, try to implement one which you are familiar with. Your instructors will help if you have questions. Take advantage of Smojo's help for syntax.

Step 4: *Try to create a new data source with no features, just the raw input. (HINT: Read the commented help in pre.m Which word would you use?). Save the data and re-run your experiment. Do your "direct extrapolation" models perform better or worse compared to feature engineering?

Step 5: *Create a new network file. Call it "arima.m". It is recommended you copy-n-paste the existing networks.m and edit accordingly. Make this file run just an ARIMA model. Use the data from Step 4. Train the models and get the best test losses.

Step 6: Repeat steps 3,4,& 5 using a lead times of 1 week and 10 weeks. Be sure to carefully collect your best test losses and the best performing configuration. You can change the lead time of your forecast by adding the line (eg, to 7 weeks ahead):

```
7 lead-time!
```

in your lab-4.m file **before** the 2 lines that generate your train/test datasets.

Class Discussion

- What are the best test losses for ARIMA vs simple NN vs feature engineered NN?
- Discuss your findings with the class.

Getting the Best Network for Deployment

Step 0: Open up lab-4.m Generate the train & test data, **ensuring that the lead time in lab-4 is 5 weeks**. Failure to do so will mean your network will be judged using the wrong lead time. Also note that in deployment, the window size is fixed as **16 weeks**. You can discard or use as much of this as you need to in your pre.m.

Step 1: Select the best performing network from Lab 4, them determine its configuration using the "config" word. For example,

Copyright (2019) by Terra Weather Pte Ltd. All rights reserved.

```
3 config
```

Will display the details of Config #3.

Step 2: Edit Lab 4's config.m, using ONLY the settings determined in Step 1. We want to repeat the experiment a number of times and get the best performing version of the model. The line:

```
5 repeats
```

in the Lab 4's config.m does this, creating 5 copies with the same configuration. This still creates different models since the initial weights are stochastically assigned by default. Stochastic solvers like SGD can generate considerable variability in the final trained models.

Step 3: Train the model. The same configuration is repeated (because of the "repeats" word used in Lab 4's config.m).

Step 4: Determine the best performing repeat.

Step 5: Deploy the model using the "deploy" word as you did in Lab 3. For example, if your best model (in Step 4) is configuration #2,

```
2 deploy
```

Step 6: Once your model is deployed, you can see how you're doing by running the word:

```
leaderboard
```

Because the dataset is "live" and updated monthly, the losses (and winners!) will change with time. Keep updating your models!

Copyright (2019) by Terra Weather Pte Ltd. All rights reserved.

Step 7: You can update your deploy model at any time, using the steps described in this Lab.

Step 8: As in Lab 3, use a nickname to change your name displayed in the Leaderboard.

Class Discussion

1. Are there other objectives of deployment besides the one outlined in the introduction?
2. Get the top 3 networks on the leaderboard to share their solutions with the class. Did they expect to top the competition?

Copyright (2019) by Terra Weather Pte Ltd. All rights reserved.