**Week 1 – Notes**

**Setting up your Machine Learning Application**

**Train / Dev / Test sets**

A mistake made by professionals who transition from a domain to another (like from NLP to Computer Vision) is that they use the same intuition and the same space of hyperparameters, but this is wrong because for each domain you have to try something different

For small data sets (for example 10,000 examples) is ok to split the data into 60/20/20, but if the data set has like 1,000,000 examples, if enough to use 10,000 examples for development / test sets

For example, a split like 98 / 1 / 1 or 99.5 / .4 / .1 is enough

It’s extremely important to have at least for dev and test data sets the same distributions

Ex: train set – images from web, dev and test sets – images taken by users

Ideally, train / dev / test should have examples drawn from the same distribution

It’s alright to have only train and dev sets, as long as you acknowledge that the estimates made on the dev set are biased (probably you don’t need an unbiased estimate)

**Bias / Variance**

Bias (underfitting) and variance (overfitting)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Train set error | 1% | 15% | 15% | 0.5% |
| Dev set error | 11% | 16% | 30% | 1% |
| Bias / variance | High variance | High  bias | High bias &  High variance | Low bias &  Low variance |

These cases should be considered in relation with the optimal error and the quality of the data set; for example if the data is bad and even humans have a performance of 15%, then the case in which the train error is 15% and the dev error is 16% cannot be considered as a high bias case

When comparing the train and dev metrics, we don’t have to consider that 0.5% error on train and 1% error on dev is a case of overfitting only because the error is double; we have to consider the absolute values

A case of both high bias and variance is when we have non-linearly separable data and the classifier draw a barrier that is mostly linear, but with some irregularities

A drawing of a butterfly

Description automatically generated with medium confidence

**Basic Recipe for Machine Learning**

Initially you have to check it your network is biased (training data performance); solutions: bigger network, train longer, and eventually perform NN architecture search

Then you check for high variance (dev data performance); solutions: more data, regularization, and perhaps NN architecture search

Classical ML: there was a bias – variance tradeoff, because if you minimize one, the other one increases, but for DL there isn’t this tradeoff, because you can minimize one issue, w/o increasing the other one