CSCI 5922 - Problem Set 2

Karan Praharaj

January 19th, 2022

1. Model Parameters versus Hyperparameters

A **model parameter** is a configuration variable that is internal to the model and whose value can be estimated from data. These values are required by the model when making predictions. These values are the building blocks of the final decision made by the model at the inference stage. These parameters are learned ("fitted") from the data during the training process, and are not set by the model designer.

On the other hand, **model hyperparameter** is a configuration that is external to the model and whose value *cannot* be estimated from data. Conventionally, these configurations are pre-defined by the model designer. The model hyperparameters need to be tuned to the right configuration for the most optimal model performance. Examples of hyperparameters are number of epochs, batch size and learning rate etc.

2. Dataset Splits

The validation set is a section of the dataset that is used during training to get a sense of how well the model is doing on data that are not being used in the training. The examples in the validation data set are used to tune the hyperparameters of the machine learning model. It is also called the "development set" (or "dev set"). The performance on the validation set can also be used to determine whether training needs to bes stopped. For example, if the candidate models are successive iterations of the same network, and training stops when the error on the validation set grows, choosing the previous model saved at the last checkpoint.

A stratified split is used to ensure that the train and test sets have approximately the same percentage of samples of each target class as the complete set.

As a result, if the data set has a large amount of each class, stratified sampling is pretty much the same as random sampling. But if one class isn't much represented in the data set, which may be the case in your dataset since you plan to oversample the minority class, then stratified sampling may yield a different target class distribution in the train and test sets than what random sampling may yield.

Note that the stratified sampling may also be designed to equally distribute some features in the next train and test sets. For example, if each sample represents one individual, and one feature is age, it is sometimes useful to have the same age distribution in both the train and test set.

3. Overfitting versus Underfitting

(a) If your model performs poorly on the training data and poorly on new examples, the model is **underfitting**. This is because the model is unable to capture the relationship between the input examples and the target values.

(b) If your model performs well on the training data and generalizes poorly to new examples, the model is **overfitting**. This is because the model is memorizing the data it has seen and is unable to generalize to examples outside of this training data that it has not seen.

4. Model Size

Training sets are used to make models learn from the "experience" of the past. In supervised learning, the data is collected as a set of features and a label. Each feature encodes a different aspect or property of the data, and each training data point (or training example) has varying values for all these features. To make the machine "see" the examples, with their labels, and have it deduce the underlying patterns and intra-dependencies in the features is why we need training data.

The test set, like the training set is also split from the same dataset, although one must ensure that examples that are in the test set do not also occur in the training set. In other words, your ML model should not be tested on data that it has already seen. This is because the entire purpose of the testing process is to *test* the generalization ability of the model to unseen data points, after it has learned from the training data.

5. Convolutional Neural Networks

(a) Two advantages of using convolutional layers instead of fully connected layers are as follows:

- Convolutional Neural Networks have the ability to extract meaningful features by also preserving spatial relations within the image. On the other hand, fully connected neural networks, despite of being universal approximators, remain poor at identifying and generalizing the raw image pixels. Thus, when it comes to data where the spatial characteristic is crucial, convolutional networks have a significant advantage over fully connected neural nets.
- The other advantage CNNs have over is FCNNs is the model size. Fully connected neural networks have a larger number of weights or parameters and are thus highly prone to overfitting, whereas a single convolution operation reduces the number of parameters quite significantly which makes it less prone to overfitting. This also entails that convolutional networks train faster, and need lesser training data relatively.

(b) Two advantages of using pooling layers instead of fully connected layers are as follows:

- Unlike fully connected networks, pooling builds in invariance and resilience towards small translations of the input.
- Reduces memory requirements by down-sampling the input slice. This is not done by fully connected layers, which are much heavier in terms of memory requirements.

- By virtue of the above, computational requirements are also reduced as a result, in comparison to fully connected layers.