Machine Learning Project Report

Valentin Herrmann, Noah Wedlich
Applied Machine Learning in Python – LMU Munich

July 16, 2025

1 Task Overview

In this project we tested if common models like Kernel SVMs, perceptrons as well as decision tree ensembles suffer from the bias-variance tradeoff. For this purpose we trained and tested the models on simple datasets like concentric bands, interleaving half-moons, spirals and separated Gaussian clusters to produce signs of underfitting and especially of overfitting.

2 Methods

The implementation of this project can essentially be divided into two parts: the tunable models and the samplers, both of which we will shortly describe here. We chose an object-oriented approach to allow for easy extension and reuse of the code and to integrate with the Courselib.

Tunable Model: This class provides a convenient way to create and train models on all (or a subset of) the hyperparameter combinations specified. When initialized, it receives a subclass of the Courselib `TrainableModel` class and a dictionary mapping parameter names to their possible values. When training the model, it will instantiate a model for each combination of hyperparameters and train it on the given data. Optionally, one can also provide a validator function which can limit the training to a subset of the combinations. Since the different models are independent of each other, the training is parallelized using Python's `multiprocessing` module. We also utilize queues to communicate with the main process, which allows us to display the training progress in real-time. If an optimizer is used for training, we will further wrap it to provide granular progress updates.

Samplers: The samplers are used to generate the datasets on which the models are trained. The subclasses provided here will sample from a distribution on $S \times L \subseteq [-1, 1]^2 \times \mathbb{N}$ where S is some 2D shape and L is a set of labels. The Samplers also provide methods to apply pre- and postprocessing to the data, which can be used to transform the data into a suitable format for training and to apply transformations to the data after sampling. This allows to easily test the robustness of models using domain shifts or label noise.

3 Experiments and Results

In this project, we focused on Kernel SVMs with the RBF kernel, Perceptrons, and Random Forests. We chose a complexity-parameter for each model and repeatedly trained the models at different complexities. We then used several metrics to analyze the results for signs of over- and underfitting.

3.1 Perceptrons

We first considered (multilayer) Perceptrons, i.e. fully connected feedforward neural networks, with the ReLU activation function. We chose the number of hidden layers and the number of neurons per layer as the complexity parameters, as they control the capacity of the model to learn complex functions. We varied these parameters one-by-one to isolate their effects on the model's performance. First we fixed the number of neurons per layer to 30 and varied the number of hidden layers between 2 and 16, the results of which can be seen in Figure 1.

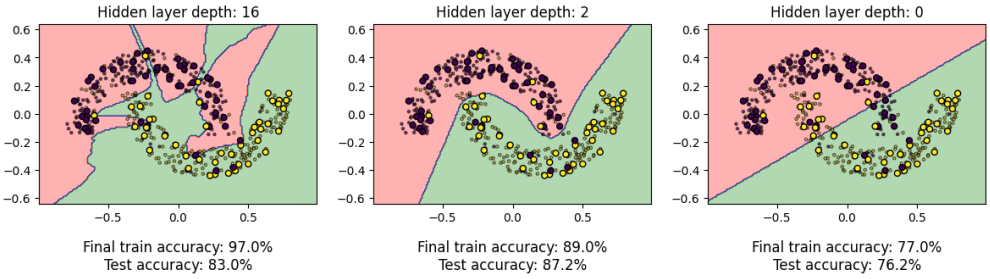

Figure 1: Perceptron results with varying number of hidden layers.

As can be seen, the training accuracy increases with the number of hidden layers as the model can learn more complex patterns. However, the test accuracy, and thus the generalization performance, starts to decrease after a certain point, indicating that the model is overfitting. This also becomes visible in the decision boundaries: the model is first too simple to separate the data, then finds the correct decision boundary, but then starts to overfit, leading to an unnecessarily complex decision boundary. This effect wasn't observed when varying the number of neurons per layer.

3.2 Kernel SVMs using the RBF kernel

Next we analyzed Kernel Support Vector Machines (SVMs) with the RBF kernel. We chose the kernel width σ as the complexity parameter, as it controls the sensitivity of the model to the data, i.e. how strongly the model will adapt to the training data. We varied σ between 0.02 and 1 and trained the model on the spirals. Some results of this are shown in Figure 2.

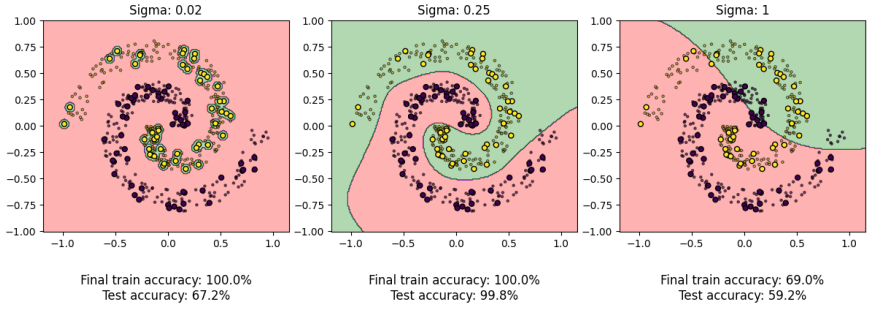

Figure 2: Kernel SVM results with varying kernel width σ.

These three plots nicely show the three stages of the bias-variance tradeoff. The model on the right is too simple to separate the data, leading to underfitting. The model in the middle is able to separate the data correctly, but the model on the left is too complex and starts to overfit, leading to a drastically worse generalization performance. This also becomes apparent in the decision boundaries, as the model on the left essentially memorizes each training point.

3.3 Random Forests

Finally, we considered Random Forests using average voting. We chose both the amount of trees in the ensemble and the maximum tree depth as the complexity parameters. The number of trees controls the amount of noise in the model, while the maximum tree depth controls how complex the individual trees can be. In Figure 3 we show the results of varying the depth, while the effects of varying the number of trees can be seen in the supplementary material.

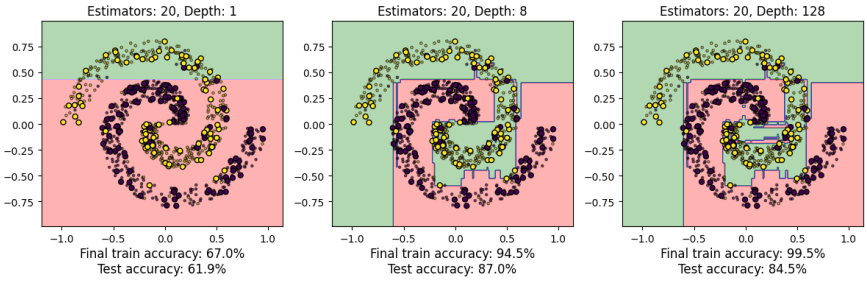

Figure 3: Random Forest results with varying maximum tree depth.

These models also exhibit the bias-variance tradeoff, however the overfitting is less pronounced than in the previous models, with the difference in training accuracy being only 2.5%.

3.4 Final Comparison

Finally we compared the three models on the spirals dataset, using the same complexity parameters as before. The results can be seen in Figure 4. As can be seen, most models follow the same trend, where the training accuracy increases with the complexity, while the test accuracy first increases and then decreases again. The Kernel SVMs are the most sensitive to this effect, as they are less reliant on huge models than the Perceptrons and Random Forests.

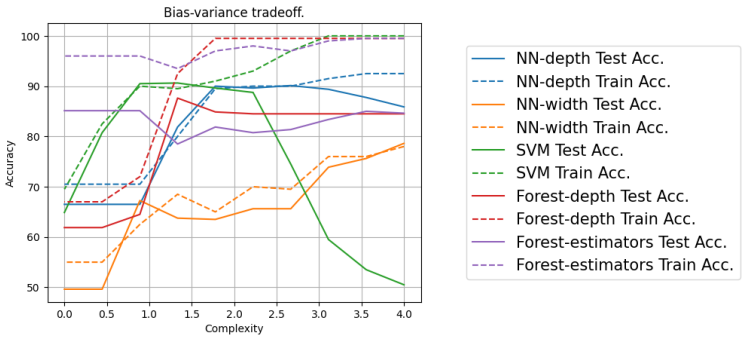

Figure 4: Final comparison of the three models on the noisy spirals dataset.

4 Discussion

We have analyzed the effects of model complexity on the performance of different models and found that all models exhibit signs of overfitting, indicating that the bias-variance tradeoff is a fundamental phenomenon in machine learning. The Kernel SVMs were the most sensitive to this effect. However, we believe that the perceptrons would also show more pronounced overfitting with larger model architectures, something we plan to investigate in the future.