Week 13:
Support Vector Machines

CMPS 320 : Machine Learning

Intro

- A Support Vector Machine (SVM) is a powerful and versatile Machine Learning model, capable of performing:
  - linear or nonlinear classification,
  - regression, and
  - outlier detection.
- It is one of the most popular models in Machine Learning, and anyone interested in Machine Learning should have it in their toolbox.
- They are particularly well suited for classification of complex small – or medium-sized datasets.

Linear SVM Classification

- Consider the figure given below:

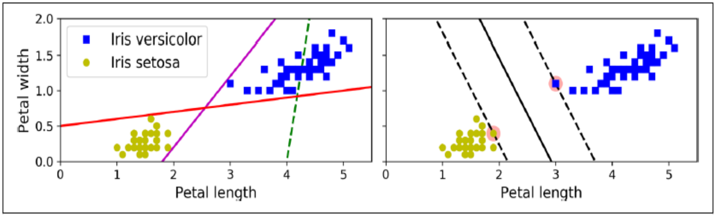

Figure 5-1. Large margin classification

- The two classes can clearly be separated easily with a straight line (they are linearly separable).
- The left plot shows the decision boundaries of three possible linear classifiers.
- The model whose decision boundary is represented by the dashed line is so bad that it does not even separate the classes properly.

Linear SVM Classification–cont.

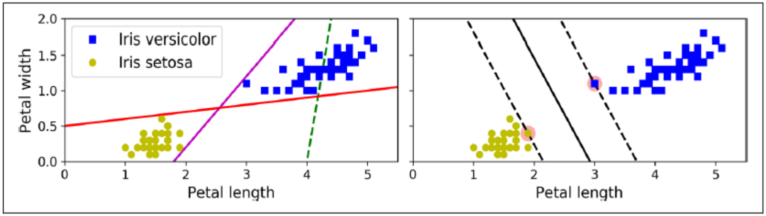

Figure 5-1. Large margin classification

- The other two models work perfectly on this training set, but their decision boundaries come so close to the instances that these models will probably not perform as well on new instances.
- In contrast, the solid line in the plot on the right represents the decision boundary of an SVM classifier;
  - this line not only separates the two classes but also stays as far away from the closest training instances as possible.

Linear SVM Classification–cont.

- You can think of an SVM classifier as fitting the widest possible street (represented by the parallel dashed lines) between the classes.
  - This is called **large margin classification**.
- Adding more training instances "off the street" will not affect the decision boundary:
  - It is fully determined (or "supported") by the instances located on the edge of the street.
  - These instances are called the **support vectors**.

Hard Margin Classification

- If we strictly impose that all instances must be off the street and on the right side, this is called **hard margin classification**.
- There are two main issues with hard margin classification:
  - First, it only works if the data is linearly separable.
  - Second, it is sensitive to outliers.

Soft Margin Classification

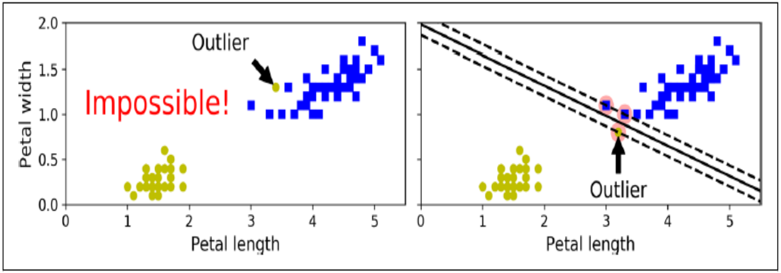

Figure 5-3. Hard margin sensitivity to outliers

- The figure above shows the iris dataset with just one additional outlier:
  - on the left, it is impossible to find a hard margin;
  - on the right, the decision boundary ends up very different from the one we saw in Figure 5-1 without the outlier, and it will probably not generalize as well.

Soft Margin Classification–cont.

- To avoid these issues, use a more flexible model.
- The objective is to find a good balance between keeping the street as large as possible and limiting the margin violations (i.e., instances that end up in the middle of the street or even on the wrong side).
- This is called **soft margin classification**.

Nonlinear SVM Classification

- Although linear SVM classifiers are efficient and work well in many cases, many datasets are not even close to being linearly separable.
- One approach to handling nonlinear datasets is to add more features, such as polynomial features:
  - in some cases this can result in a linearly separable dataset.

Nonlinear SVM Classification

- Consider the left plot in Figure 5-5: it represents a simple dataset with just one feature, x_1.
- This dataset is not linearly separable, as you can see.
- Adding a second feature $x_2 = (x_1)^2$, results in a 2D dataset which is perfectly linearly separable.

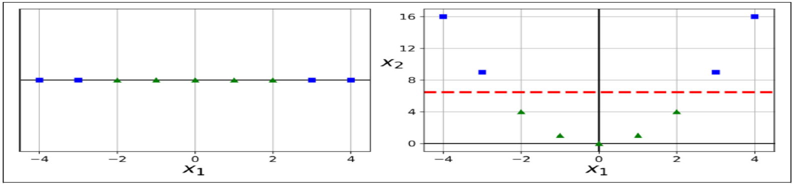

Figure 5-5. Adding features to make a dataset linearly separable

Polynomial Kernel

- When using SVMs we can apply a mathematical technique called **kernel**.
- The kernel trick makes it possible to get the same result as if you had added many polynomial features, even with very high-degree polynomials, without actually having to add them.
- There is no combinatorial explosion of the number of features because you don't actually add any features.
- If your model is overfitting, you might want to reduce the polynomial degree.
- Conversely, if it is underfitting, you can try increasing it.

Similarity Features–Gaussian RBF Kernel

- Another technique to tackle nonlinear problems is to add features computed using a similarity function, which measures how much each instance resembles a particular landmark.
- We define the similarity function to be the Gaussian Radial Basis Function (RBF) with $\gamma = 0.3$

Equation 5-1. Gaussian RBF

$$\phi_\gamma(\mathbf{x}, \ell) = \exp\left(-\gamma \| \mathbf{x} - \ell \|^2\right)$$

- This is a bell-shaped function varying from 0 (very far away from the landmark) to 1 (at the landmark).

Gaussian RBF Kernel

- Increasing gamma makes the bell-shaped curve narrower
- As a result, each instance's range of influence is smaller: the decision boundary ends up being more irregular, wiggling around individual instances.
- Conversely, a small gamma value makes the bell-shaped curve wider: instances have a larger range of influence, and the decision boundary ends up smoother.
- γ acts like a regularization hyperparameter: if your model is overfitting, you should reduce it; if it is underfitting, you should increase it.

Remarks

- Other kernels exist but are used much more rarely.
- Some kernels are specialized for specific data structures.
  - String kernels are sometimes used when classifying text documents or DNA sequences.
- Which Kernel should you choose??
- As a rule of thumb, you should always try the linear kernel first, especially if the training set is very large or if it has plenty of features.
- If the training set is not too large, you should also try the Gaussian RBF kernel; it works well in most cases.
- If you have spare time and computing power, you can experiment with a few other kernels, using cross-validation and grid search