Support Vector Machines — SVMs

By
Dr. Junaid Qazi, PhD
https://www.linkedin.com/in/jqazi/

Welcome to the Support Vector Machines lecture!

In this lecture, we will talk about another widely used supervised machine learning technique, Support Vector Machines (SVMs). SVMs deliver state-of-the-art performance in a range of real-world application and considered as one of the standard tool for machine learning. We will explore the theory behind SVMs and try to understand their working principle with practical examples.

✅ *Optional Readings and References:*
sklearn's Official Documentation
Introduction to Statistical Learning - Chapter 9
Machine Learning - A Probabilistic Perspective - Chapter 14

Data Science from Scratch — Part 2: Business Machine Learning

Support Vector Machines — SVMs

Support vector machines (SVMs) are a set of supervised learning methods used for both classification and regression analysis. SVMs have been shown to perform well in a variety of settings, and are often considered one of the best "out of the box" classifiers.

Support Vector Machines — SVMs

Support vector machines (SVMs) are a set of supervised learning methods used for both classification and regression analysis. SVMs have been shown to perform well in a variety of settings, and are often considered one of the best "out of the box" classifiers.

- Given a set of training examples, each marked as belonging to one or the other of two categories, SVM training algorithm builds a model that assigns new examples (test data-points) to one category or the other, making it a non-probabilistic binary linear classifier.
- SVM model is a representation of the examples as points in n-dimensional space (n is the number of features), mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible.
- New examples are then mapped into that same n-dimensional space and predictions are made for a category based on which side of the gap the new examples fall.

Support Vector Machines — SVMs

Support vector machines (SVMs) are a set of supervised learning methods used for both classification and regression analysis. SVMs have been shown to perform well in a variety of settings, and are often considered one of the best "out of the box" classifiers.

- Given a set of training examples, each marked as belonging to one or the other of two categories, SVM training algorithm builds a model that assigns new examples (test data-points) to one category or the other, making it a non-probabilistic binary linear classifier.
- SVM model is a representation of the examples as points in n-dimensional space (n is the number of features), mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible.
- New examples are then mapped into that same n-dimensional space and predictions are made for a category based on which side of the gap the new examples fall.

Support Vector Machines — SVMs

Support vector machines (SVMs) are a set of supervised learning methods used for both classification and regression analysis. SVMs have been shown to perform well in a variety of settings, and are often considered one of the best “out of the box” classifiers.

- Given a set of training examples, each marked as belonging to one or the other of two categories, SVM training algorithm builds a model that assigns new examples (test data-points) to one category or the other, making it a non-probabilistic binary linear classifier.
- SVM model is a representation of the examples as points in n-dimensional space (n is the number of features), mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible.
- New examples are then mapped into that same n-dimensional space and predictions are made for a category based on which side of the gap the new examples fall.

Support Vector Machines — SVMs

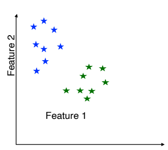
Support vector machines (SVMs) are a set of supervised learning methods used for both classification and regression analysis. SVMs have been shown to perform well in a variety of settings, and are often considered one of the best “out of the box” classifiers.

- Given a set of training examples, each marked as belonging to one or the other of two categories, SVM training algorithm builds a model that assigns new examples (test data-points) to one category or the other, making it a non-probabilistic binary linear classifier.
- SVM model is a representation of the examples as points in n-dimensional space (n is the number of features), mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible.
- New examples are then mapped into that same n-dimensional space and predictions are made for a category based on which side of the gap the new examples fall.

Let’s try to understand with some example training dataset.

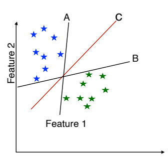
Support Vector Machines — SVMs

Consider, we have a labeled training data, blue and green with some feature 1 and 2. We want to predict the class for our new point in this binary classification problem.



Support Vector Machines — SVMs

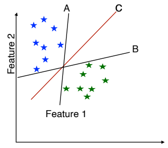
Consider, we have a labeled training data, blue and green with some feature 1 and 2. We want to predict the class for our new point in this binary classification problem.



We draw a hyperplane (simple line in this 2D data) between the two classes. To separate the classes perfectly, we have lots of options such as A, B and C in the diagram (we can draw more hyper planes!). The question is, which plane is the best to select!

Support Vector Machines — SVMs

Consider, we have a labeled training data, blue and green with some feature 1 and 2. We want to predict the class for our new point in this binary classification problem.

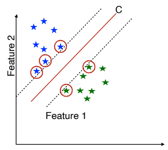


We draw a hyperplane (simple line in this 2D data) between the two classes. To separate the classes perfectly, we have lots of options such as A, B and C in the diagram (we can draw more hyper planes!). The question is, which plane is the best to select!

The plane which maximize the margin between the classes is the best option, in this case C separates the classes with the maximum margin. A & B also separate the classes, but only with a small margin.

Support Vector Machines — SVMs

Consider, we have a labeled training data, blue and green with some feature 1 and 2. We want to predict the class for our new point in this binary classification problem.



We draw a hyperplane (simple line in this 2D data) between the two classes. To separate the classes perfectly, we have lots of options such as A, B and C in the diagram (we can draw more hyper planes!). The question is, which plane is the best to select!

The plane which maximize the margin between the classes is the best option, in this case C separates the classes with the maximum margin. A & B also separate the classes, but only with a small margin.

In the diagram, dotted lines are the margin lines that extend out from the hyperplane. The vector point (training points), in circles, that are touching the the dotted (margin) lines are called Support Vectors and this is where the name Support Vector Machines comes from.

Support Vector Machines — SVMs

In addition to performing linear classification, SVMs can efficiently perform a **non-linear classification** using what is called the kernel trick, implicitly mapping their inputs into higher-dimensional feature spaces.

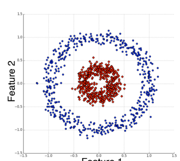


Figure Source, fair policy use

non-linearly separable data, the input space in this case cannot be separated well by a linear classifier and yet we can obviously see a clear separation between the two classes.

Support Vector Machines — SVMs

In addition to performing linear classification, SVMs can efficiently perform a **non-linear classification** using what is called the kernel trick, implicitly mapping their inputs into higher-dimensional feature spaces.

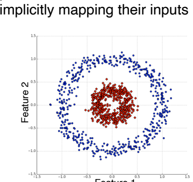
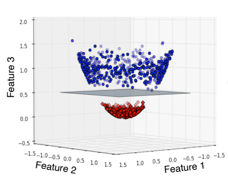


Figure Source, fair policy use

non-linearly separable data, the input space in this case cannot be separated well by a linear classifier and yet we can obviously see a clear separation between the two classes.



The kernel trick (viewing the data in higher dimensional feature space) transforms the data in a feature space where the instances from the two classes may be linearly separable.

Support Vector Machines — SVMs

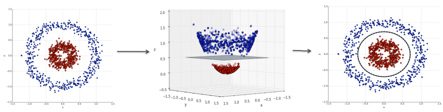


Figure Source, fair policy use

So, using the kernel trick, we end up separating the non-linearly separable data by viewing it in higher dimensional feature space.

SVM - Important Parameters

```
svm_model = SVC()

Init signature: SVC(C=1.0, kernel='rbf', degree=3, gamma='auto', coef0=0.0,
shrinking=True, probability=False, tol=0.001, cache_size=200, class_weight=None,
verbose=False, max_iter=-1, decision_function_shape=None, random_state=None)
Docstring:
C-Support Vector Classification.
```

Different Kernel functions can be specified for the decision function. Common kernels are linear, polynomial, rbf or sigmoid. It is also possible to specify our own custom kernels such as giving the kernel as a Python function etc.

```
kernel : string, optional (default='rbf')
    Specifies the kernel type to be used in the algorithm.
    It must be one of 'linear', 'poly', 'rbf', 'sigmoid', 'precomputed' or
    a callable.
    If none is given, 'rbf' will be used. If a callable is given it is
    used to pre-compute the kernel matrix from data matrices; that matrix
    should be an array of shape ``(n_samples, n_samples)``.
```


SVM - Important Parameters

```
svm_model = SVC()

Init signature: SVC(C=1.0, kernel='rbf', degree=3, gamma='auto', coef0=0.0,
shrinking=True, probability=False, tol=0.001, cache_size=200, class_weight=None,
verbose=False, max_iter=-1, decision_function_shape=None, random_state=None)
Docstring:
C-Support Vector Classification.
```

```
C : float, optional (default=1.0)
    Penalty parameter C of the error term.
```

```
degree : int, optional (default=3)
    Degree of the polynomial kernel function ('poly').
    Ignored by all other kernels.
```

```
gamma : float, optional (default='auto')
    Kernel coefficient for 'rbf', 'poly' and 'sigmoid'.
    If gamma is 'auto' then 1/n_features will be used instead.
```


SVM - Important Parameters

```
svm_model = SVC()

Init signature: SVC(C=1.0, kernel='rbf', degree=3, gamma='auto', coef0=0.0,
shrinking=True, probability=False, tol=0.001, cache_size=200, class_weight=None,
verbose=False, max_iter=-1, decision_function_shape=None, random_state=None)
Docstring:
C-Support Vector Classification.
```

The rbf kernel on two samples **x** and **x'**, represented as feature vector in some input space, is defined by the equation:

$$K(\mathbf{x}, \mathbf{x}') = \exp(-\gamma\|\mathbf{x} - \mathbf{x}'\|^2)$$

where, $\|\mathbf{x} - \mathbf{x}'\|^2$ may be recognized as the square Euclidean distance between the two feature vectors, **x** and **x'**.

and gamma, γ, must be greater than 0.

SVM - Important Parameters

```
svm_model = SVC()
```

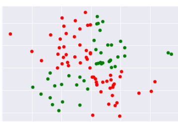
```
Init signature: SVC(C=1.0, kernel='rbf', degree=3, gamma='auto', coef0=0.0,
shrinking=True, probability=False, tol=0.001, cache_size=200, class_weight=None,
verbose=False, max_iter=-1, decision_function_shape=None, random_state=None)
Docstring:
C-Support Vector Classification.
```

When training an SVM with the Radial Basis Function (RBF) kernel, two parameters must be considered:

C is common to all SVM kernels, trades off misclassification of training examples against simplicity of the decision surface. A low C makes the decision surface smooth, while a high C aims at classifying all training examples correctly.

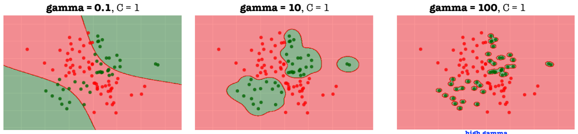
gamma defines, how far the influence of a single training example reaches, with low values meaning 'far' and high values meaning 'close'.

SVM - Important Parameters



SVM - Important Parameters

gamma = 0.1, C = 1 | **gamma = 10**, C = 1 | **gamma = 100**, C = 1



low gamma
decision regions are not good

high gamma
islands of decision-boundaries around data points

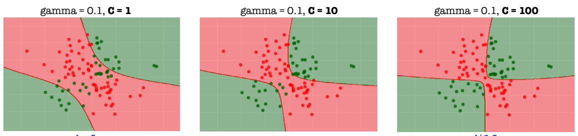
SVM - Important Parameters

gamma = 0.1, C = 1 | gamma = 10, C = 1 | gamma = 100, C = 1

low gamma
decision regions are not good

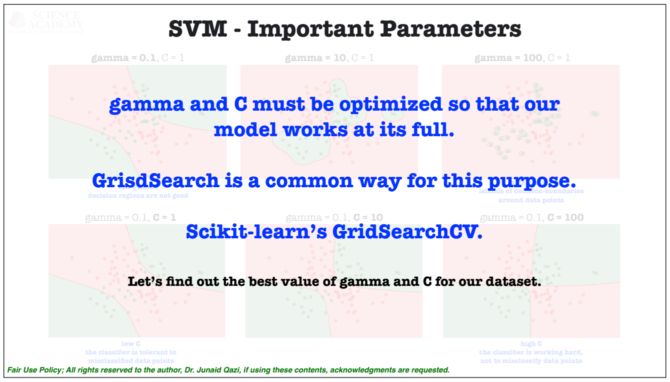
high gamma
islands of decision-boundaries around data points

gamma = 0.1, **C = 1** | gamma = 0.1, **C = 10** | gamma = 0.1, **C = 100**



low C
the classifier is tolerant to misclassified data points

high C
the classifier is working hard, not to misclassify data points


SVM - Important Parameters
gamma = 0.1, C = 1
gamma = 10, C = 1
gamma = 100, C = 1
gamma and C must be optimized so that our model works at its full.
GrisdSearch is a common way for this purpose.
decision regions are not good
around data points
gamma = 0.1, C = 1
gamma = 0.1, C = 10
gamma = 0.1, C = 100
Scikit-learn's GridSearchCV.
Let's find out the best value of gamma and C for our dataset.
low C
the classifier is tolerant to
misclassified data points
high C
the classifier is working hard,
not to misclassify data points
Fair Use Policy; All rights reserved to the author, Dr. Junaid Qazi, if using these contents, acknowledgments are requested.