From different sources:
ADS course, Hesam Salavati
Hands on ML with Scikeat-learn, etc.

SVM & KNN For Classification

Introduction to Data Science
Spring 1403

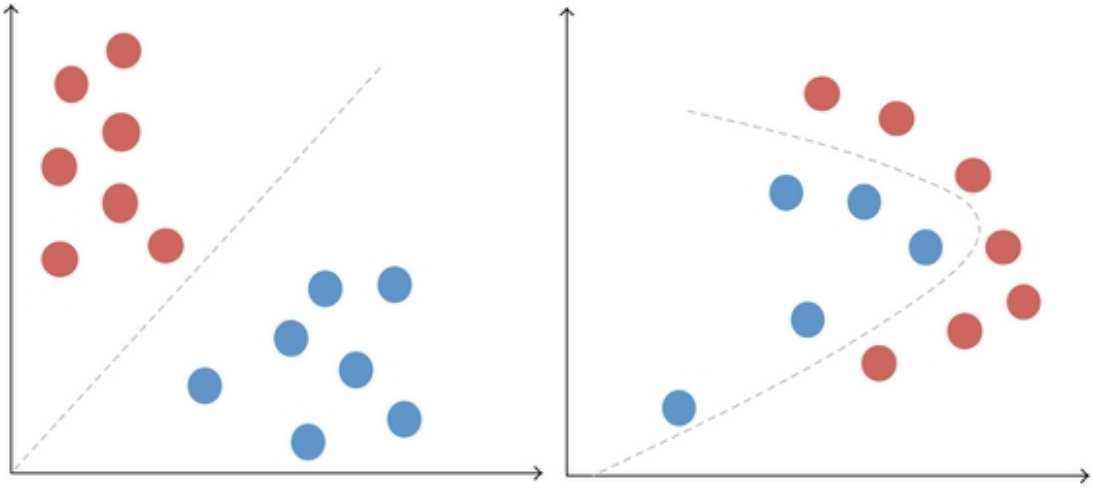
Yadollah Yaghoobzadeh

Agenda

- Review of classification and its applications
- SVM
- KNN

What is Classification?

- Predicting a *discrete* value, i.e. the class of the target variable, based on available data



Credit:https://www.researchgate.net/figure/Linear-versus-nonlinear-classification-problems_fig4_279274803

Classification Examples in Real Life

Fault Detection in Power Grid/Industrial Machinery
https://springerplus.springeropen.com/articles/10.1186/s40064-015-1080-x

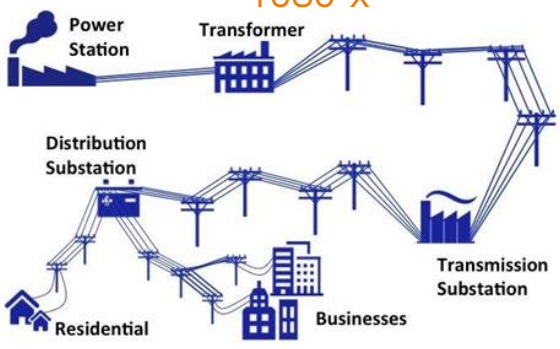


Image:researchgate.net/profile/Patrick-Hosein/publication/312562428/figure/fig3/AS:667722895618052@1536208952125/Power-Grid-Architecture.jpg

Stroke Risk Prediction
https://pdfs.semanticscholar.org/df5c/7d1bd7a59009dc51b9db903aa7f144241879.pdf

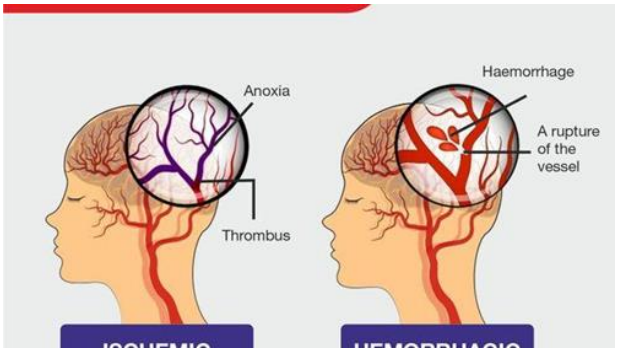


Image:pacehospital.com/brain-stroke-types-causes-symptoms-prevention-and-treatment

Other examples

- Spam detection in Emails
- Handwritten digit recognition
- Sentiment analysis
- Facial identity recognition
- Product categorization

General Formulation

- We have a vector of discrete numbers: $y = [y_1, y_2, \ldots, y_n]$
- **Goal**: find a function to *explain* the target as best as you can
- **How:** run *classification* on the data (*observations*) that we have:

$$X_{n \times k} = \begin{bmatrix} x_{11} & x_{12} & \ldots & x_{1k} \\ x_{21} & x_{22} & \ldots & x_{2k} \\ \ldots & \ldots & \ldots & \ldots \\ x_{n1} & x_{n2} & \ldots & x_{nk} \end{bmatrix}$$

Linear Classification

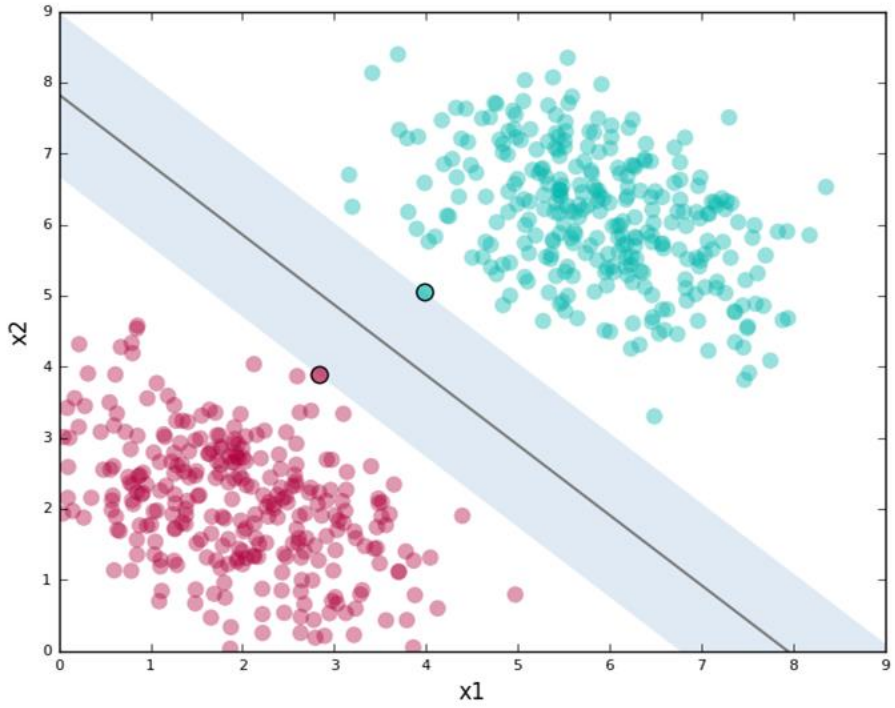
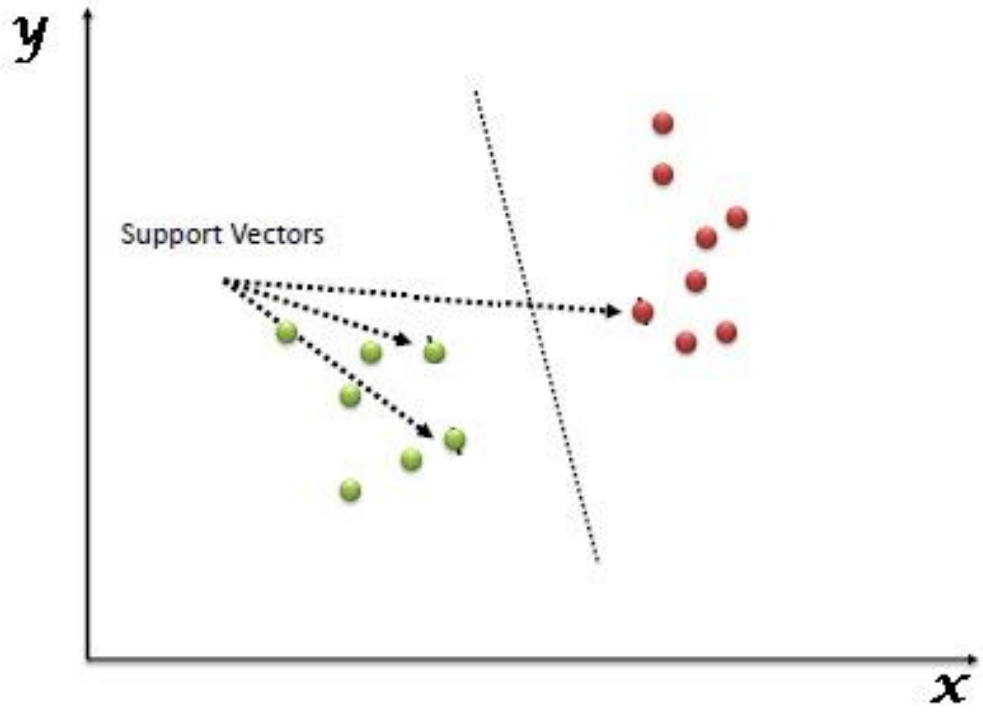


Image: https://medium.com/@paarthbir/image-classification-with-a-linear-classifier-cab02f7f8a30

Support Vector Machine

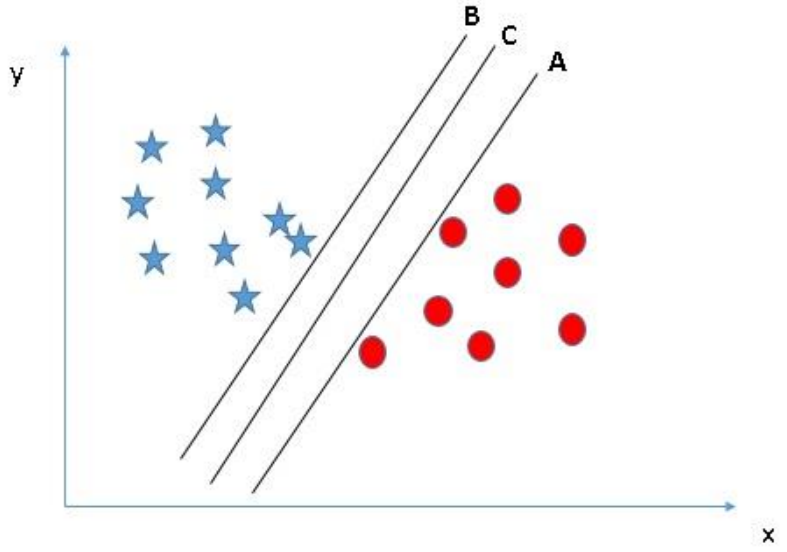
Support Vector Machine

- A *support vector machine* (SVM) is a powerful machine learning model, capable of performing linear or nonlinear classification and regression tasks.
- SVMs shine with small to medium-sized nonlinear datasets (i.e., hundreds to thousands of instances), especially for classification tasks.
- SVMs don't scale very well to very large datasets.

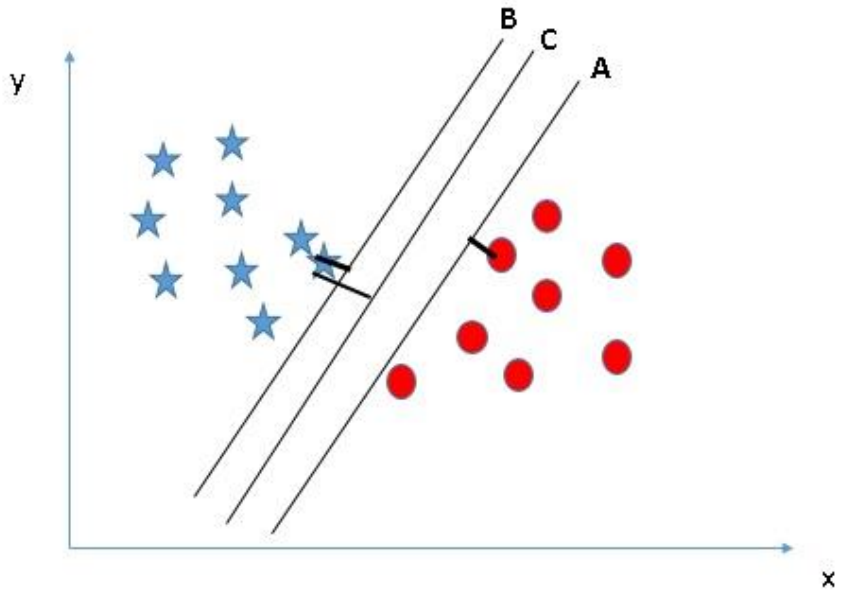


Support Vector Machines (SVM) in Data Science - Analytics Vidhya

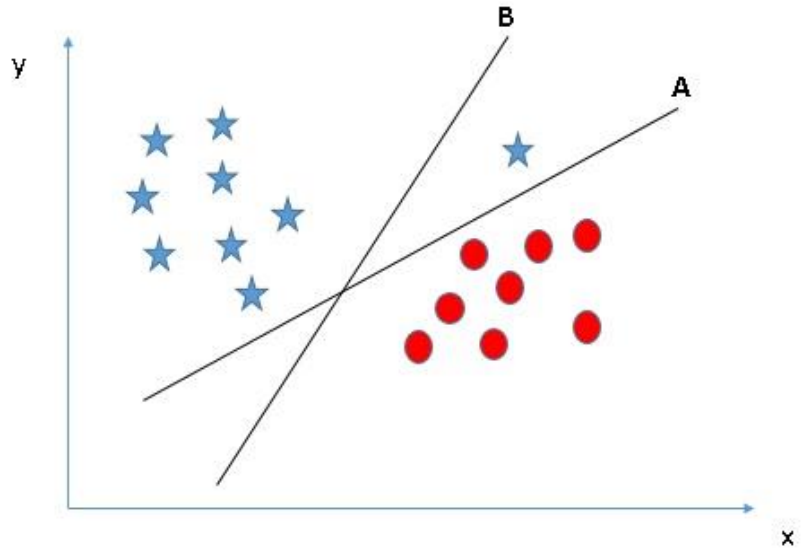
Finding the right hyperplane



Finding the right hyperplane



Quiz: A or B?

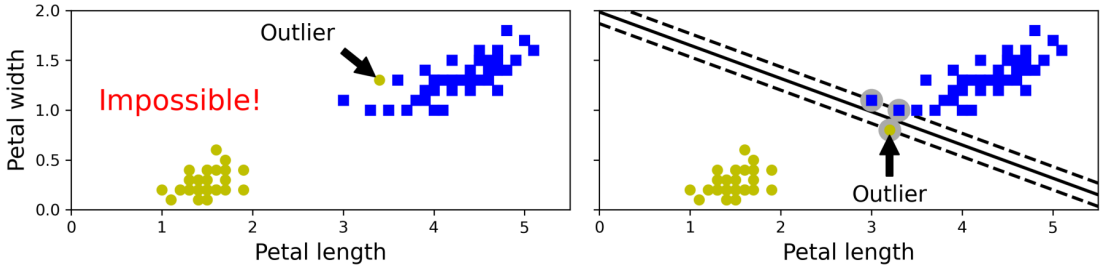


The Idea Behind SVM

- If classes are linearly separable, find a line that separates the two classes and stays far away from the closest training instances.
- You can think of an SVM classifier as fitting the widest possible street. This is called *large margin classification*.
- *Support vectors*: the instances located on the edge of the street.

Hard and Soft Margin Classification

- *Hard margin classification:* we strictly impose that all instances must be off the street and on the right side.
  - It only works if the data is linearly separable
  - It is sensitive to outliers.
- *Soft margin classification*: find a good balance between keeping the street as large as possible and limiting the *margin violations*.



Implementation in Scikit-Learn

```
from sklearn.datasets import load_iris
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.svm import LinearSVC

iris = load_iris(as_frame=True)
X = iris.data[["petal length (cm)", "petal width (cm)"]].values
y = (iris.target == 2)  # Iris virginica

svm_clf = make_pipeline(StandardScaler(),
                        LinearSVC(C=1, random_state=42))
svm_clf.fit(X, y)
```

```
X_new = [[5.5, 1.7], [5.0, 1.5]]
svm_clf.predict(X_new)
```

```
array([ True, False])
```

Support Vector Machines: Formulation

- Margin is *inversely* related to the norm of the weights.
- So instead of maximizing the margin, minimize the weights:

$$\min \|w\|_2^2$$

Subject to:

$$w^T X_i - b \geq 1, \forall y_i = 1$$
$$w^T X_i - b \leq -1, \forall y_i = -1$$

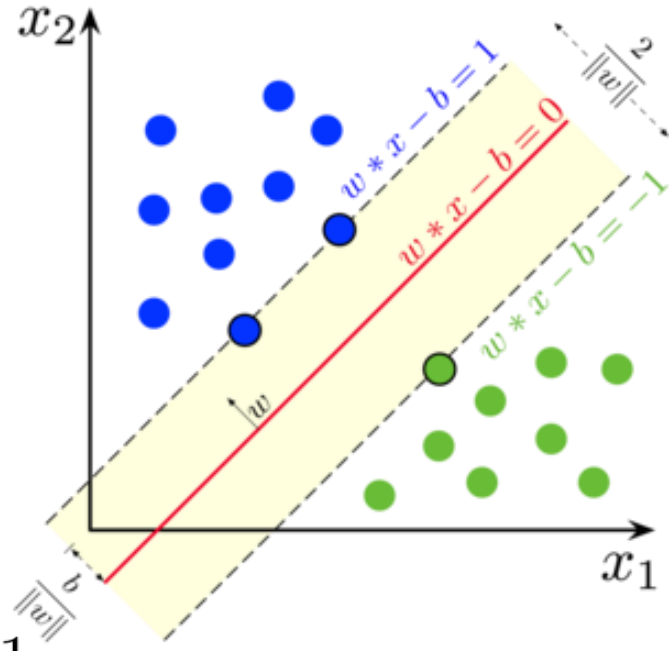


Image: https://en.wikipedia.org/wiki/Support-vector_machine

Testing a linear SVM

The separator is defined as the set of points for which:

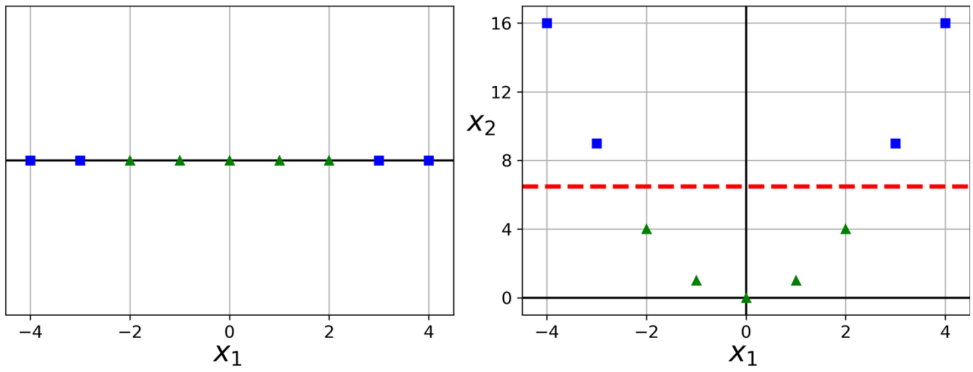
$$\mathbf{w}.\mathbf{x} + b = 0$$

$$\textit{so if} \quad \mathbf{w}.\mathbf{x}^c + b > 0 \quad \textit{say its a positive case}$$

$$\textit{and if} \quad \mathbf{w}.\mathbf{x}^c + b < 0 \quad \textit{say its a negative case}$$

Nonlinear classification

- 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.
  - ➢ This may result in a linearly separable dataset.



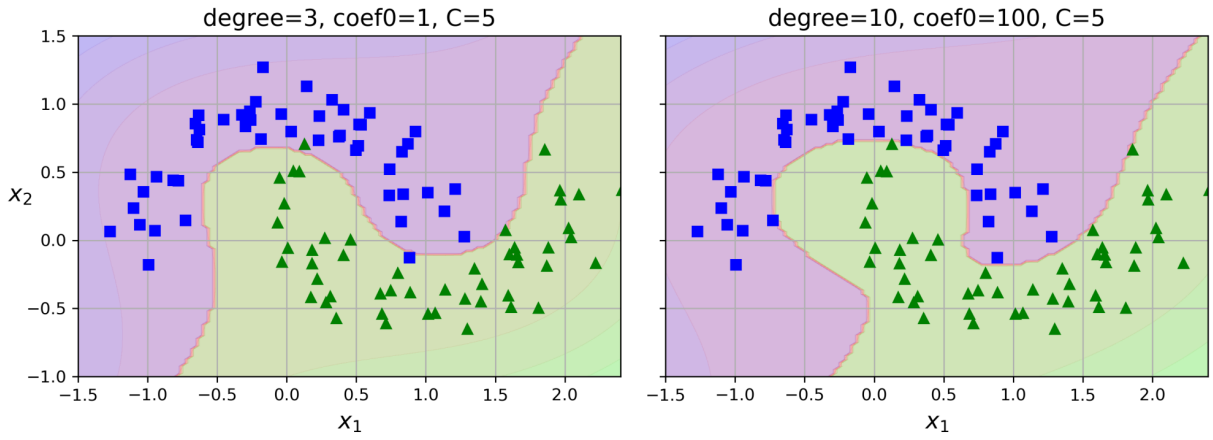
Polynomial Kernel

- ➢ Adding polynomial features is simple to implement and work with all sorts of ML algorithms.
  - ➢ Low polynomial degree: cannot deal with very complex datasets.
  - ➢ High polynomial degree: huge number of features make a slow model.

- ➢ 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.

Polynomial Kernel

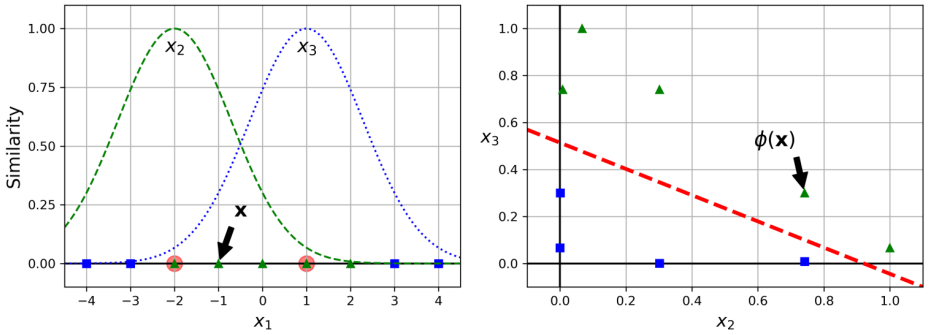
```
from sklearn.svm import SVC

poly_kernel_svm_clf = make_pipeline(StandardScaler(),
                                    SVC(kernel="poly", degree=3, coef0=1, C=5))
poly_kernel_svm_clf.fit(X, y)
```



Similarity Features

- Another way to tackle nonlinear problems is to add features computed using a *similarity function*, which measures how much each instance resembles a particular *landmark*.
- Example. Gaussian *Radial Basis Function* (RBF):

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

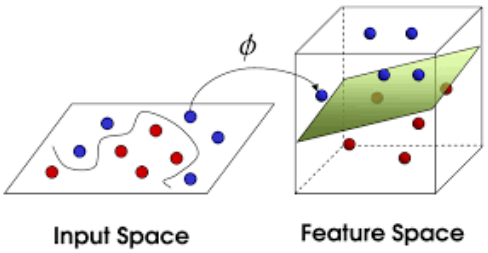


Landmarks

- How to select the landmarks?
  - The simplest approach is to create a landmark at the location of each and every instance in the dataset.
  - Doing that creates many dimensions and thus increases the chances that the transformed training set will be linearly separable.
  - The downside is that a training set with m instances and n features gets transformed into a training set with m instances and m features (assuming you drop the original features).
  - If your training set is very large, you end up with an equally large number of features.

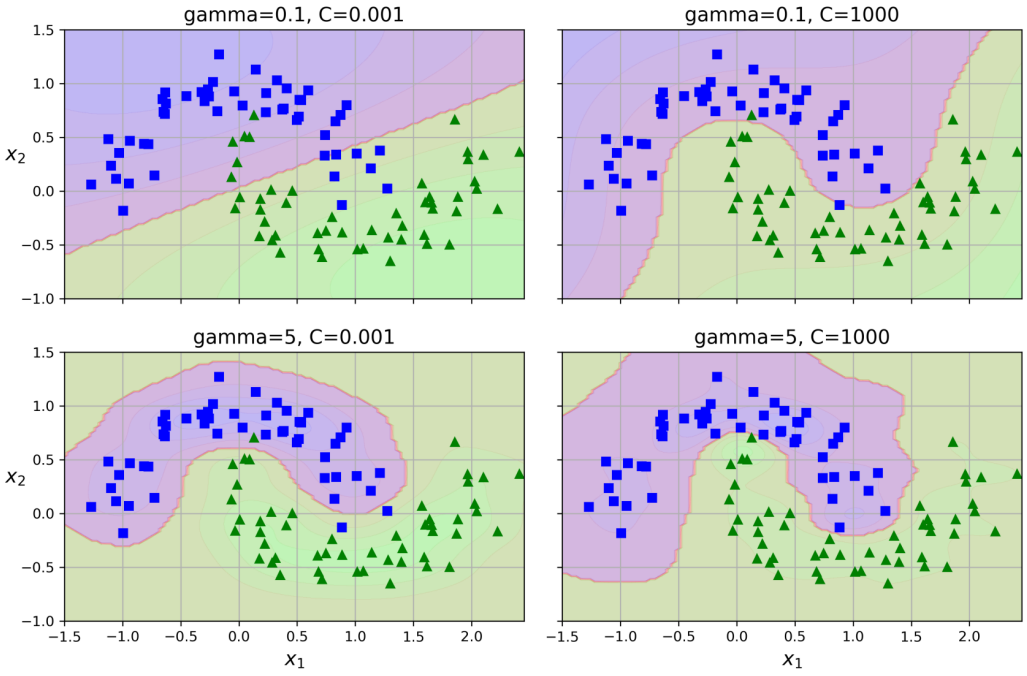
Gaussian RBF Kernel

- The similarity features method can be useful with any machine learning algorithm, but it is computationally expensive.
  - For each landmark, we have a new feature.
- The kernel trick makes it possible to obtain a similar result as if you had added many similarity features.




```
rbf_kernel_svm_clf = make_pipeline(StandardScaler(),
                                   SVC(kernel="rbf", gamma=5, C=0.001))
rbf_kernel_svm_clf.fit(X, y)
```

SVM Classifiers with RBF Kernel



K-Nearest Neighbors (KNN)

- Assign the class based on the labels of the object's **k closest** neighbors.
- Need a metric to measure distance from "neighbors":
  - Euclidean distance
  - Hamming distance
- *K* is something we need to tune.

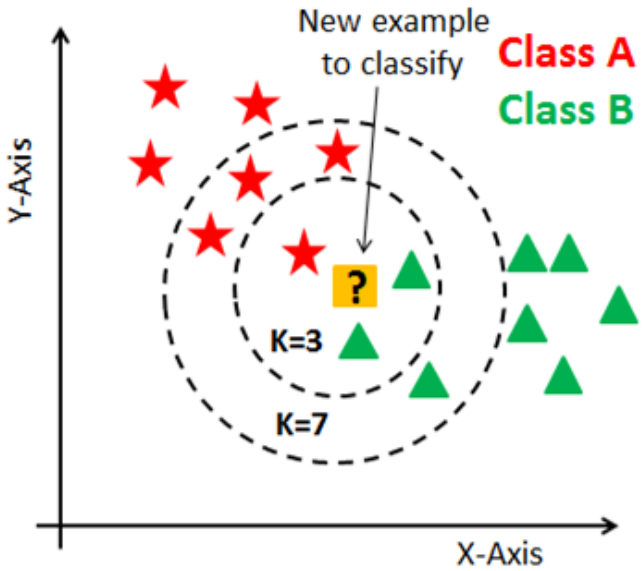


Image: https://github.com/Robots-Vision/knn-examples

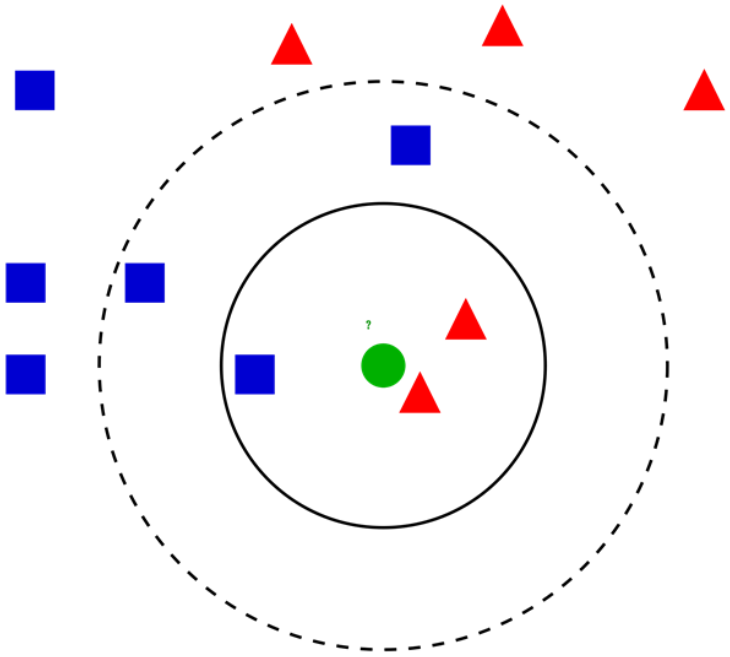
This Looks Like Majority Voting. Any Drawbacks?

K-Nearest Neighbors (KNN): Pros and Cons

- Simple majority voting drawback: **imbalanced** datasets
- Solution: **weighted voting** (weights proportional to the distance from neighbor)
- Advantages of KNN:
  - Easy to implement (no need for a model!)
  - Versatile: both for classification and **regression!**
- Disadvantages of KNN:
  - Becomes very slow for large datasets (computationally expensive)

KNN in Action

- What is the label for the green point according to KNN algorithm for:
    - K=3
    - K=5
    - K=10

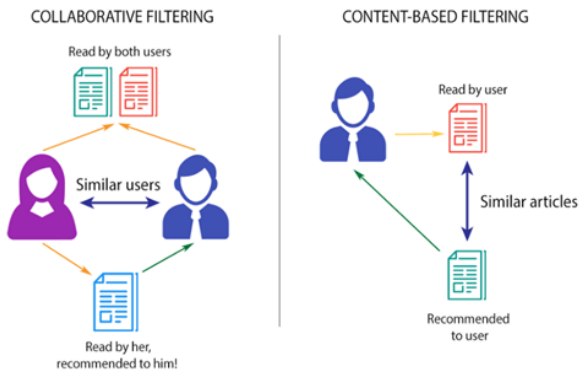


https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/

KNN Real Life Applications

Recommendation Systems

https://www.analyticsvidhya.com/blog/2020/08/recommendation-system-k-nearest-neighbors/



Text Categorization

pdf.sciencedirectassets.com/278653/1-s2.0-S1877705814X00020/1-s2.0-S1877705814003750/main.pdf

Image:towardsdatascience.com/machine-learning-nlp-text-classification-using-scikit-learn-python-and-nltk-c52b92a7c73a