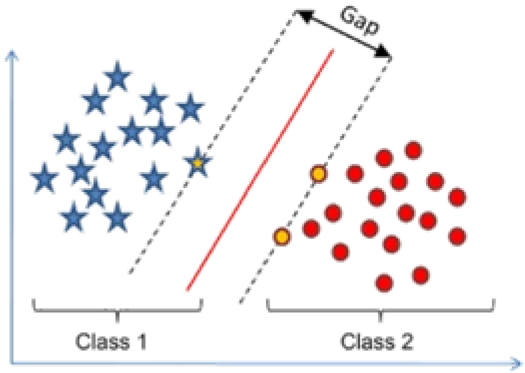
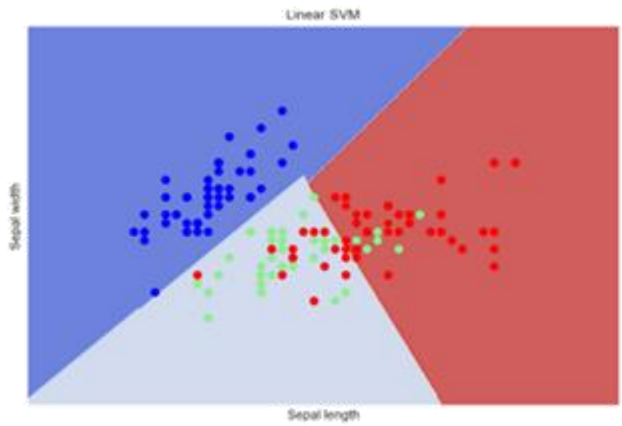
Material Summary: Support Vector Machines

1. Support Vector Machines

1.1 Support Vector Machines

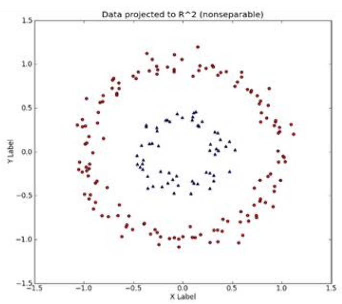
- Extreme(ly easy) case
  - Two linearly separable classes
    - Decision boundary: simple line (plane in many dimensions)
- Goal
  - Choose the line that best separates the classes
  - Maximum margin
  - The math formula for the objective function is a little complex because it involves matrix algebra
- Applications:
  - Mainly for classification
    - sklearn.svm.SVC, LinearSVC
  - Regression: sklearn.svm.SVR
- Difficult to separate classes ⇒ use regularization
  - C – penalty for misclassification (L2, $C = 1/\lambda$)
    - Smaller value = stricter (more regularization)
- Many classes
  - scikit-learn uses the "one-vs-one" approach
    - Trains $c(c-1)/2$ classifiers (c – number of classes)
- Considerations
  - Few datasets are linearly separable
  - High complexity: between $O(m.n^2)$ and $O(m.n^3)$
    - m – number of features, n – number of samples
    - Feasible for max $\sim 10^5$ samples

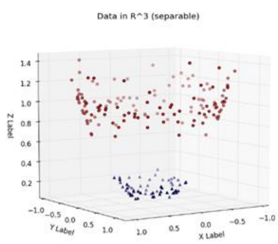


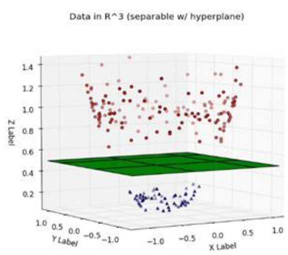


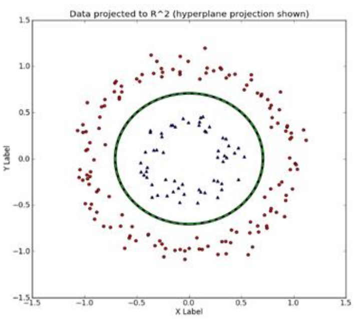
1.2 "Kernel Trick"

- Used when data is not linearly separable
- Algorithm
  - Create non-linear combinations of the features using a mapping function (**kernel**)
    - This projects them to a higher-dimensional space
- Most widely used: **R**adial **B**asis **F**unction (Gaussian) kernel
  - Hyperparameter γ – needs to be optimized (e.g. via grid search)

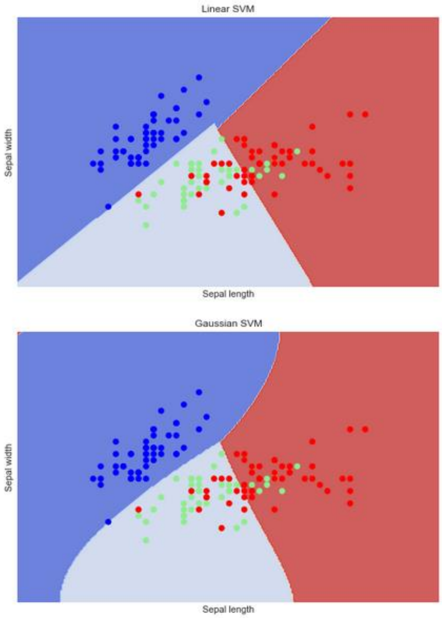








1.3 Example: Kernel SVM

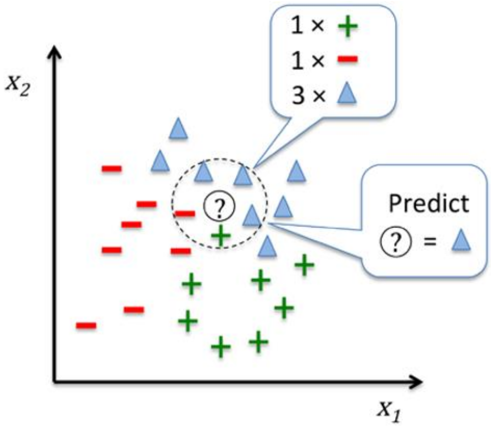


- Use a Gaussian SVM to predict Iris classes
  - Try to fine-tune the parameters (C, γ)
    - Using cross-validation
  - Print out-of-sample test scores for the model
  - Plot the decision regions
  - Plot a ROC curve
  - Perform model selection
    - Linear vs. RBF (Gaussian) kernel
- Some other explanations of the "kernel trick"
  - *Quora*
  - *Reddit*
  - *Medium* (a little more math)

2. k-Nearest Neighbors

2.1 k-Nearest Neighbors (kNN)

- "Lazy learner"
  - Doesn't learn a fitting function but memorizes the training data
- Algorithm
  - Choose a number k and a distance metric (e.g. Euclidean)
    - This choice provides bias / variance balance
    - **Minkowski distance**: generalized Euclidean distance
  - Find the k nearest neighbors of the current sample
  - Use majority vote to classify
- Advantage: easily adapts to new data
- Downside: computational complexity grows linearly with new samples
  - Efficient implementation: k-d trees



2.2 Example: k-Nearest Neighbors

- Perform kNN on Iris data
  - It can also be used for regression
    - We need to be extremely careful, especially in the case of extrapolation
- Display the decision regions

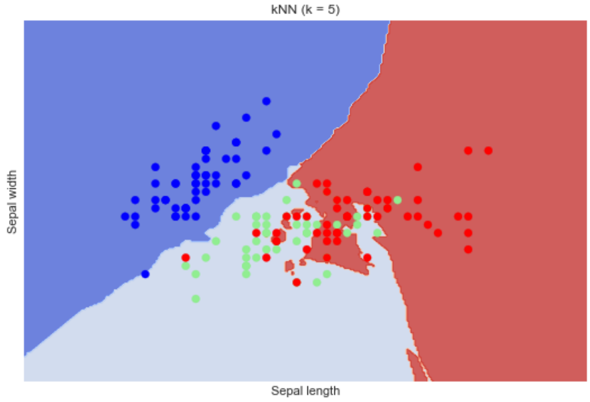
```
from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier(n_neighbors = 5)
knn.fit(iris.data, iris.target)
```

  - Voronoi tiling (tessellation)
    - Very useful in image processing and working with graphs

```
knn = KNeighborsClassifier(n_neighbors = 1)
```

  - Can also be used for regression
    - *Docs* (scikit-learn)



3. Anomaly Detection

3.1 One-Class SVM

- Anomaly / novelty detection
  - Given a dataset free of outliers, detect anomalies in new observations
- Outlier detection
  - Given a "polluted" dataset, filter out the outliers
    - We already know about RANSAC – this is one of many methods
- We can use a one-class SVM as an anomaly detector
  - *Docs and example*
  - Kernel: usually RBF
  - Parameters:
    - γ – kernel coefficient
    - ν – probability of finding a regular observation far from the others
      - $0 \leq \nu \leq 1$, $0{,}5$ by default
  - Works for outlier detection too, but not on all datasets

3.2 Example: Outlier Detection

- Use a one-class SVM to detect anomalies in the Boston housing dataset
  - Plot the anomalous observations
- * Optionally, compare different outlier detectors
  - E.g., RANSAC vs. one-class SVM
  - Follow the tutorial in the scikit-learn docs
    - Apply it to the Boston data
- Notes
  - Be extremely careful with the testing data
    - It must be properly stratified
  - You'll see that these algorithms don't accept a y parameter
    - Unsupervised learning