Discussion 5:

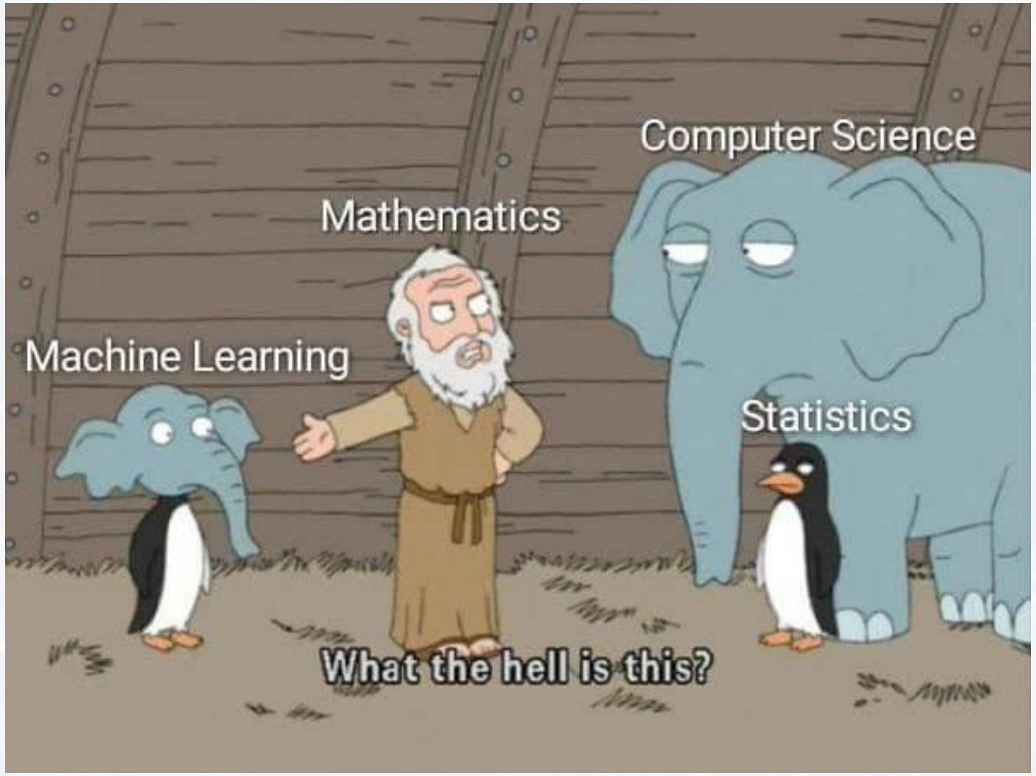
Machine Learning Overview & SVM

Markus Hohle

University California, Berkeley

Machine Learning Algorithms

MSSE 277B, 3 Units

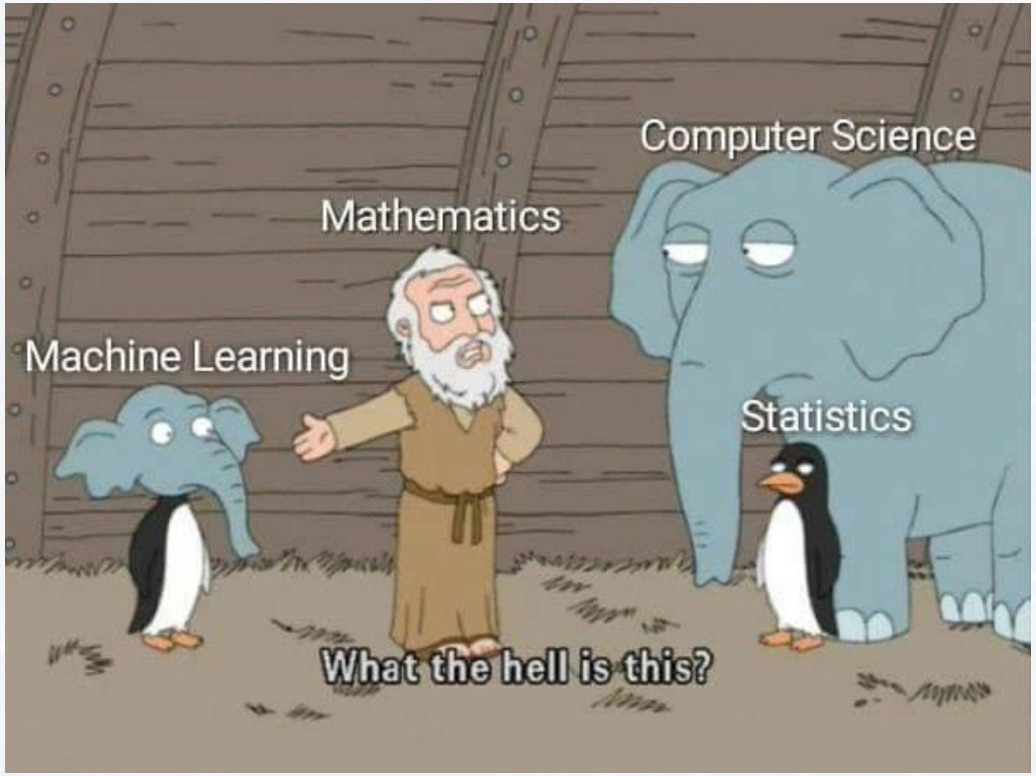


Outline

General Overview

Support Vector Machine
(Supervised Learning)

Machine Learning Overview & SVM



Outline

General Overview

Support Vector Machine
(Supervised Learning)

ML Overview

	supervised	unsupervised
regression	linear regression ✓ ANNs (soon)	
classification	logistic regression ✓ Naïve Bayes ✓ K-nearest **SVM (today)** ANNs (soon)	GMM ✓ K-means ✓ Leiden, DBSCAN HMM Trees ✓

ML Overview

supervised

Workflow

1) creating the model:

```
my_model = library.method(argument1 = 'arg1', … )
```

2) training the model

```
out = my_model.fit(xtrain, ytrain)
```

3) evaluation

```
ypred = out.predict(xeval)
accur = (ypred == yeval).sum()/len(yeval)
```

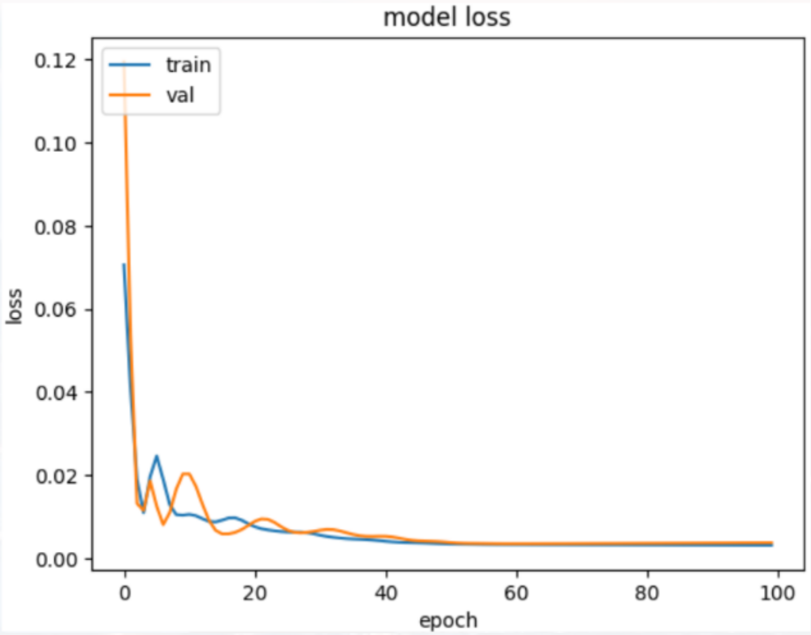
4) prediction (actual application)

```
ypred = out.predict(xnew)
```

How to split the data?

- 80% training **(60% actual training 20% evaluation)**
- 20% test

evaluation while training!



ML Overview

Workflow

supervised

1) creating the model:

```
my_model = library.method(argument1 = 'arg1', … )
```

2) training the model

```
out = my_model.fit(xtrain, ytrain)
```

3) evaluation

```
ypred = out.predict(xeval)
accur = (ypred == yeval).sum()/len(yeval)
```

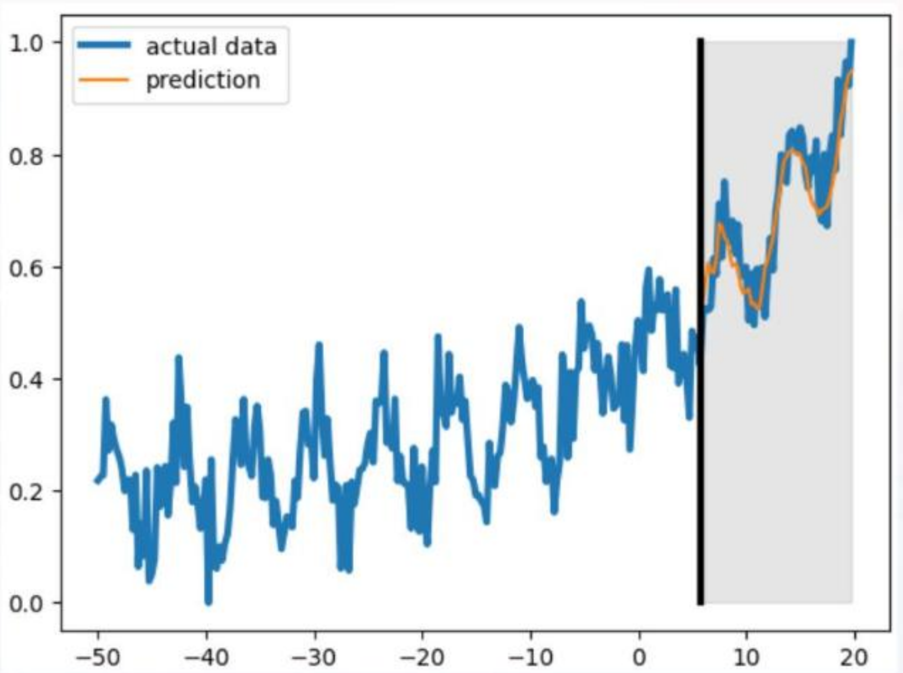
4) prediction (actual application)

```
ypred = out.predict(xnew)
```

How to split the data?

- 80% training **(60% actual training 20% evaluation)**
- 20% test

prediction after training/evaluation completed



ML Overview

unsupervised

Workflow

1) setting up the model

~~2) fitting the model: **X** are the **features** (sepal lengths/widths and petal lengths/widths),~~
Y are the **classes** (setosa, versicolor, virginica)

3) evaluating the model

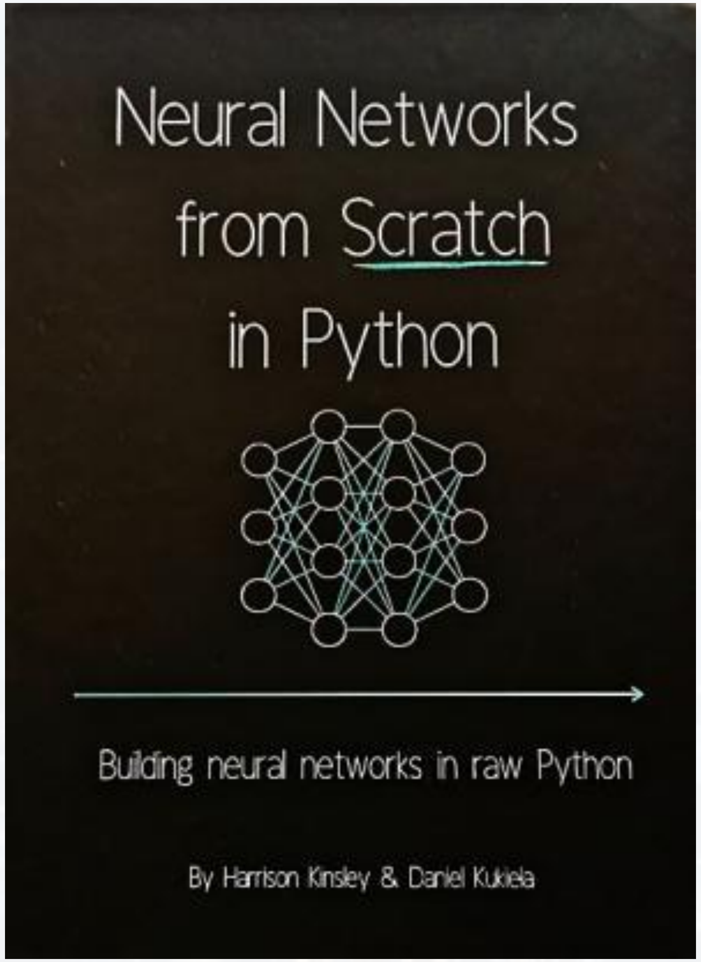
4) applying the model to a new data set

unsupervised learning:
- we don’t know the classes
- sometimes we even don’t know the **number *k* of classes**

ML Overview

resources for the next lectures to come:

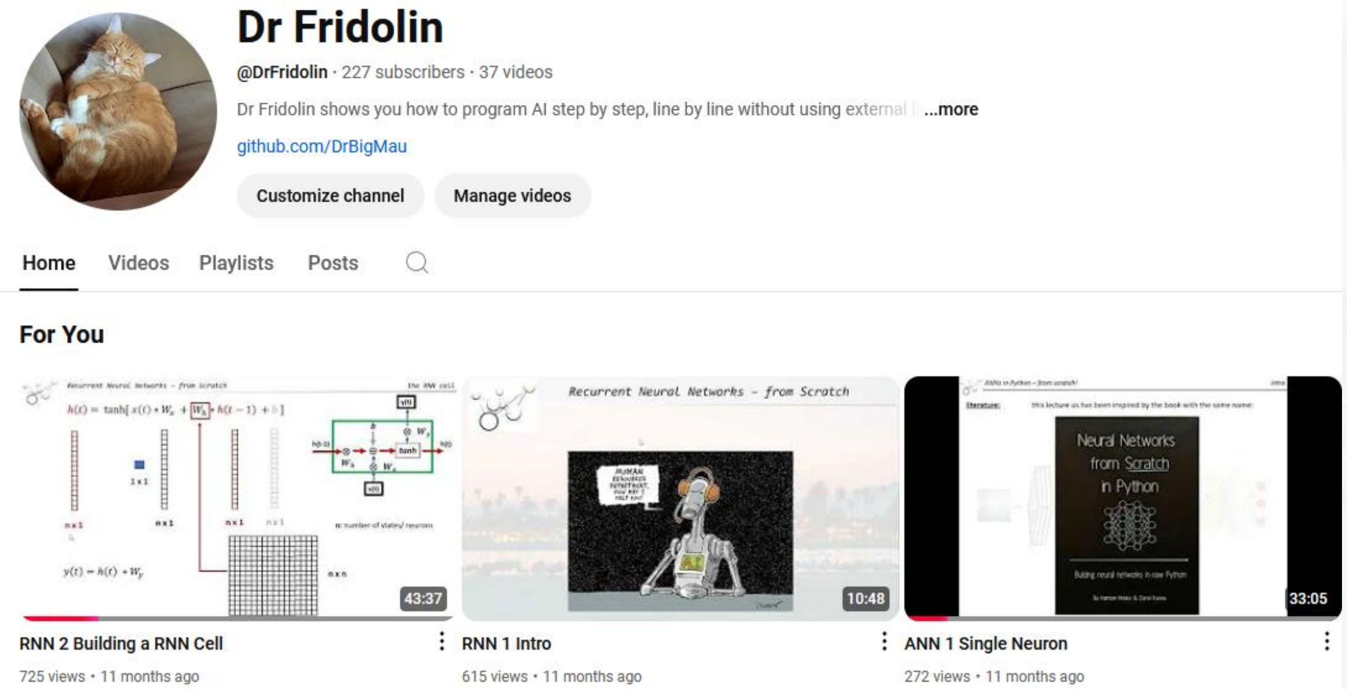
Tutorials and Books



Neural Networks from Scratch in Python

Building neural networks in raw Python

By Harrison Kinsley & Daniel Kukieła



Dr Fridolin

@DrFridolin · 227 subscribers · 37 videos

Dr Fridolin shows you how to program AI step by step, line by line without using external ...more

github.com/DrBigMau

RNN 2 Building a RNN Cell
725 views • 11 months ago

RNN 1 Intro
615 views • 11 months ago

ANN 1 Single Neuron
272 views • 11 months ago

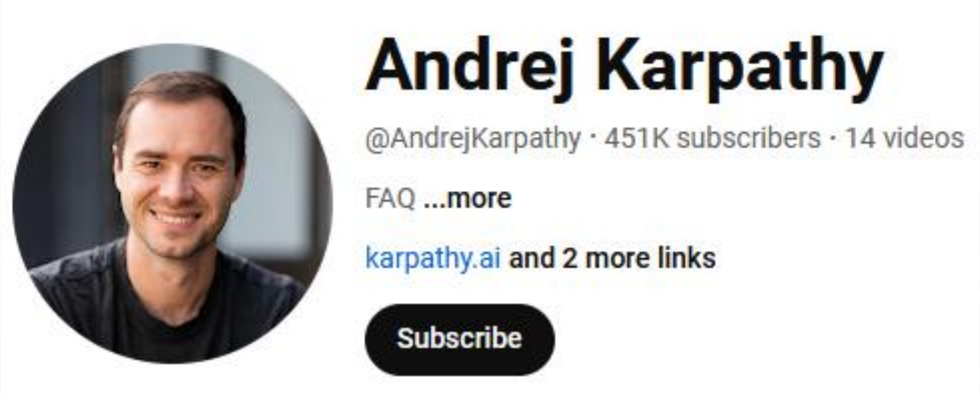
building ANNs, RNNs, LSTMs, CNNs **from scratch**
(only numpy and `matplotlib`)

Tutorials and Books

examples and **application** (CNN/RNN/LSTM and more):

Jason Brownlee
Machine Learning Mastery

all about **transformers, LLM/NLP and way more**

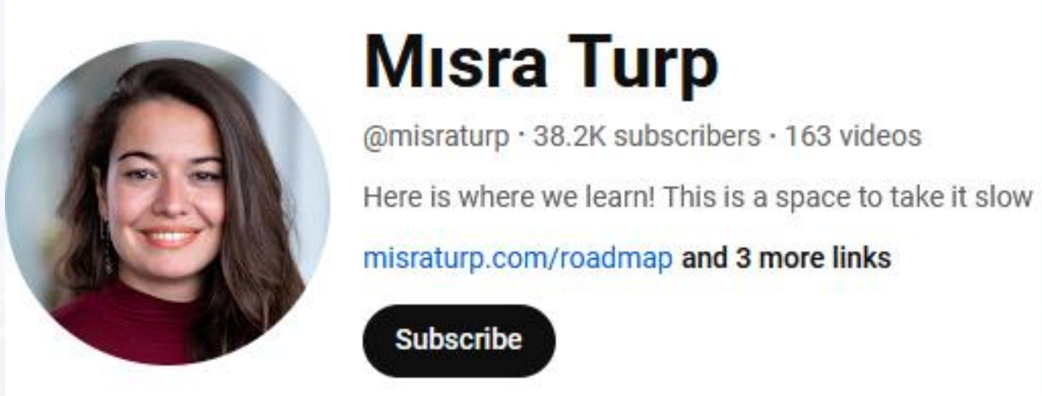


Andrej Karpathy

@AndrejKarpathy · 451K subscribers · 14 videos

FAQ **...more**

karpathy.ai **and 2 more links**



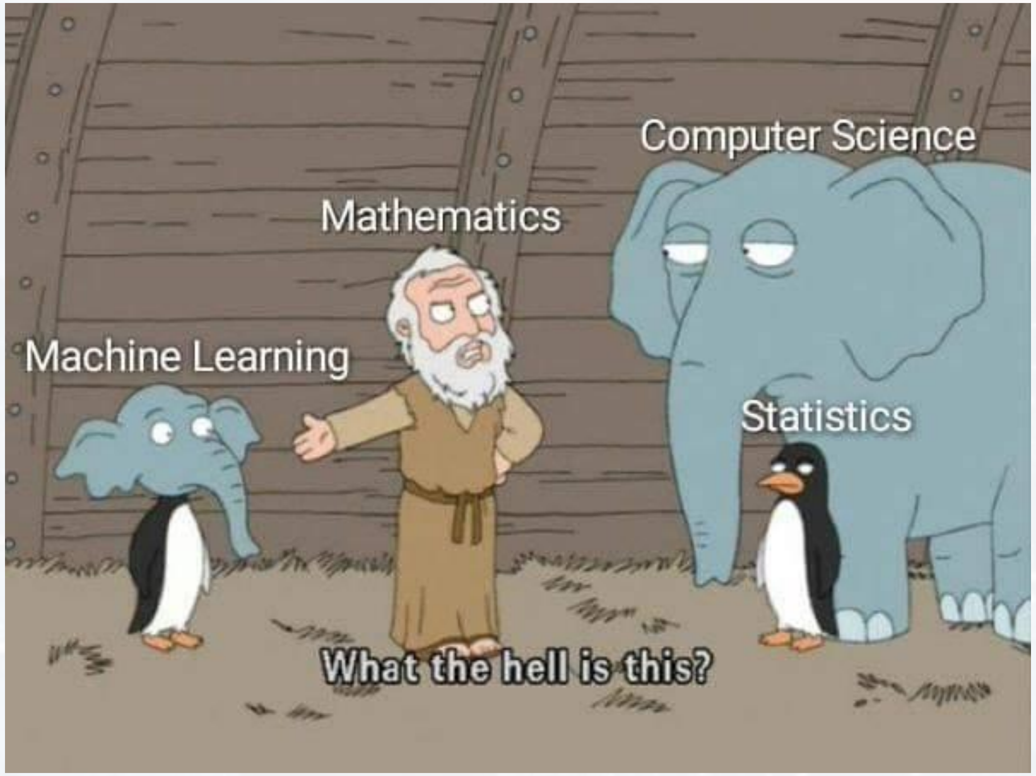
Mısra Turp

@misraturp · 38.2K subscribers · 163 videos

Here is where we learn! This is a space to take it slow

misraturp.com/roadmap **and 3 more links**

Machine Learning Overview & SVM



Outline

General Overview

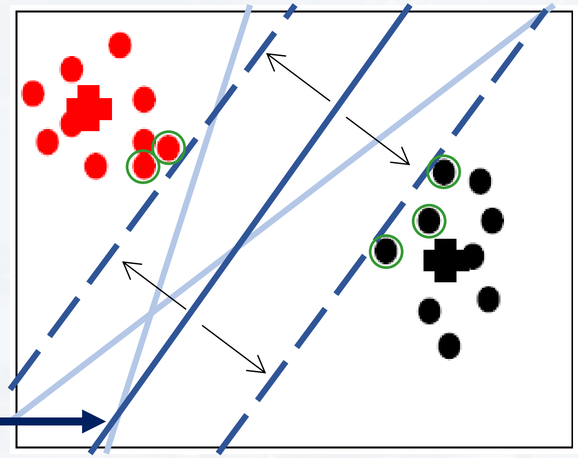
Support Vector Machine
(Supervised Learning)

SVM = **S**upport **V**ector **M**achine

idea:

1) finds best **linear** classifier for separating **two** classes by **maximizing margin** using **support vectors**
2) assign new data points to these categories
3) **supervised** learning
4) uses the “kernel trick”

○ support vectors (data points at the edge)



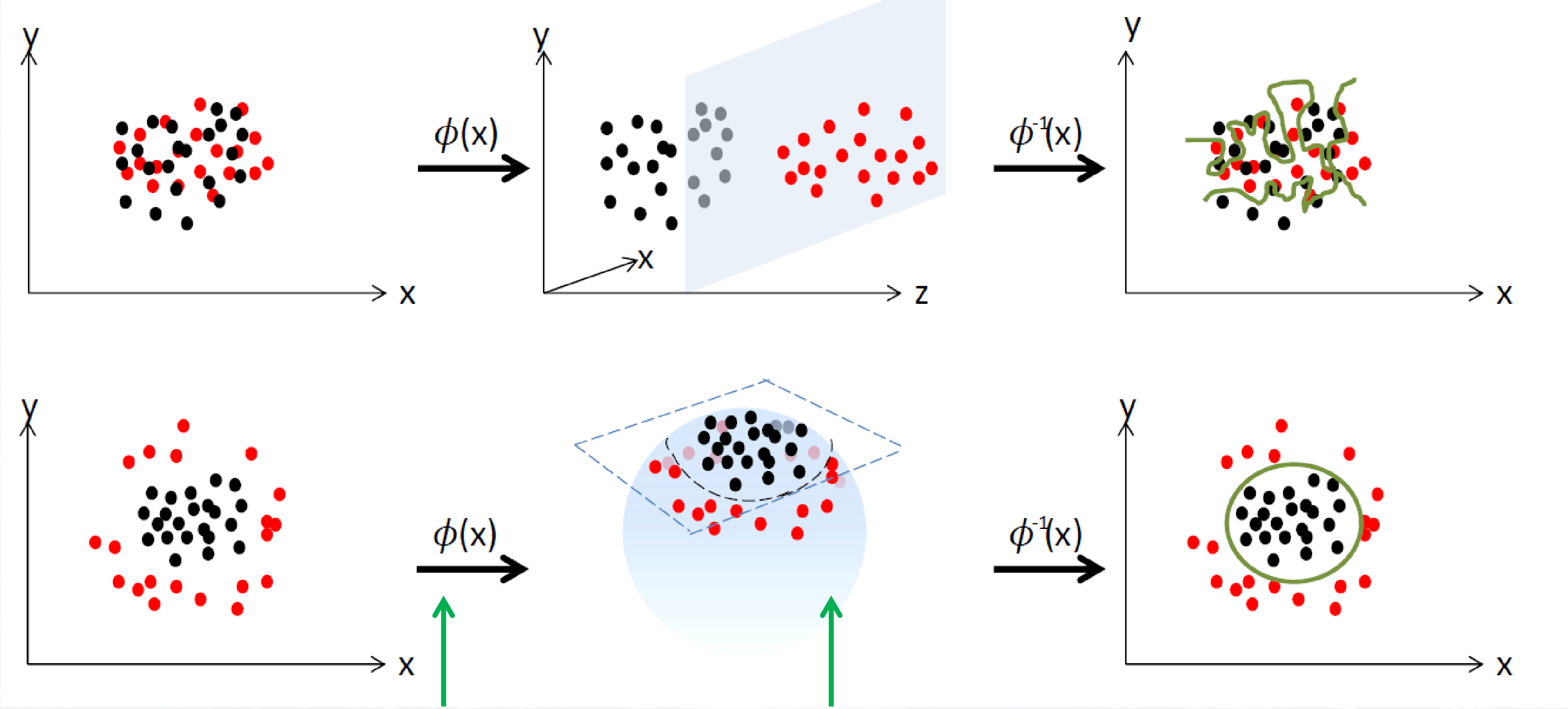
best separator

1) What if linear separation is not possible?
2) Is there a multiclass SVM?

ML Overview

1) **What if linear separation is not possible?**
2) Is there a multiclass SVM?

idea: mapping data to higher dimensional feature space



a) data space in N-D

b) a function ϕ maps the data into a M-D (M>N) feature space

c) find hyperplane separator in feature space

d) map back into data space (separator not linear)

1) **What if linear separation is not possible?**
2) Is there a multiclass SVM?

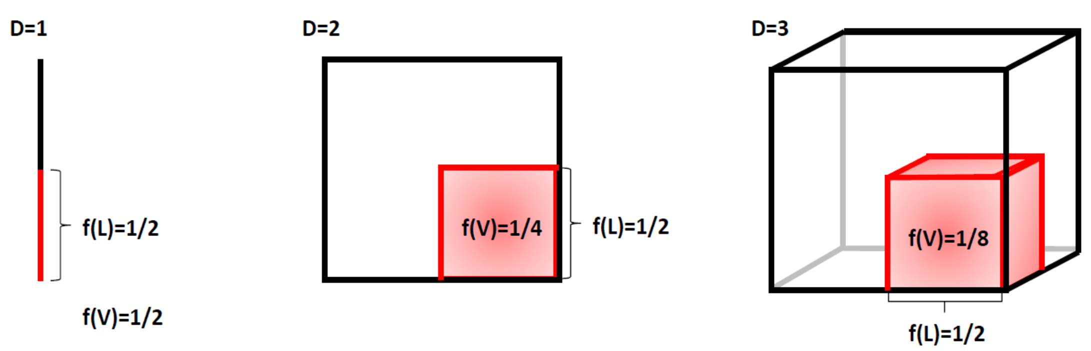
problems: - computationally intensive
- ϕ usually unknown

…and dimensionality!!

ML Overview

1) **What if linear separation is not possible?**
2) Is there a multiclass SVM?

What is the fraction of volume $f(V)$ covered by a certain fraction of length $f(L) = r/R$ for different dimensions D?



answer: $f(V) \sim f(L)^D$

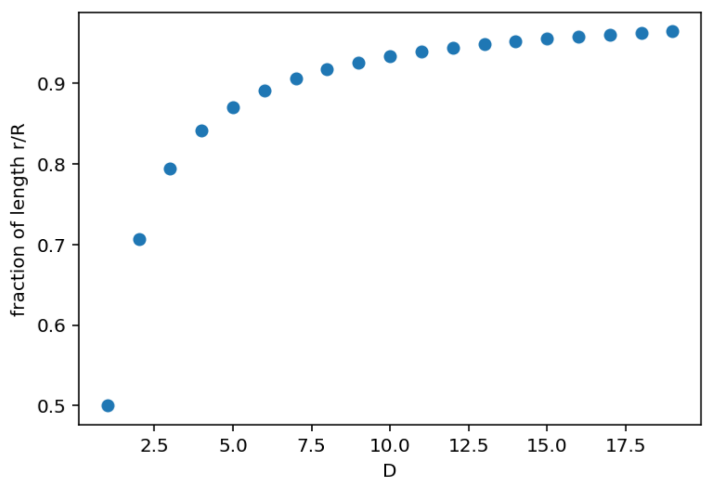
$$f(V) = C(D) f(L)^D$$

$C(D)$: constant that only depends on D

$$= C(D)\left(\frac{r}{R}\right)^D$$

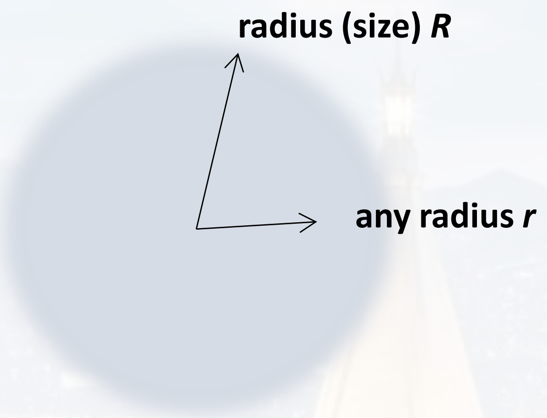
$$\left(\frac{r}{R}\right) \sim \sqrt[D]{f(V)}$$

fraction of length that we need to explore in order to find a particular fraction of datapoints **(uniformly distributed in V)**



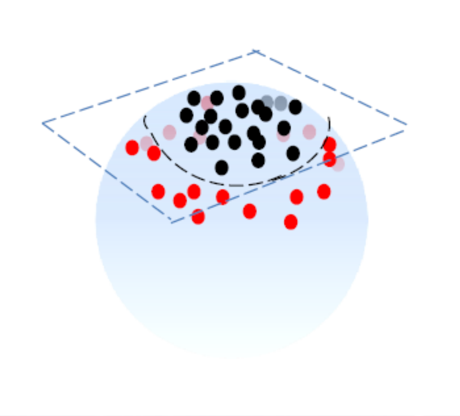
1) **What if linear separation is not possible?**
2) Is there a multiclass SVM?

N - D space



radius (size) *R*

any radius *r*



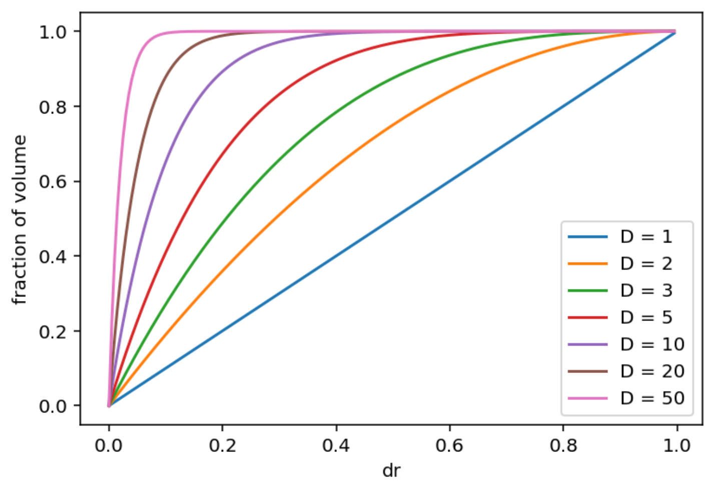
hypersphere:

$C(D)$: constant that only depends on D

$$V_D(r) = C(D)\, r^D$$

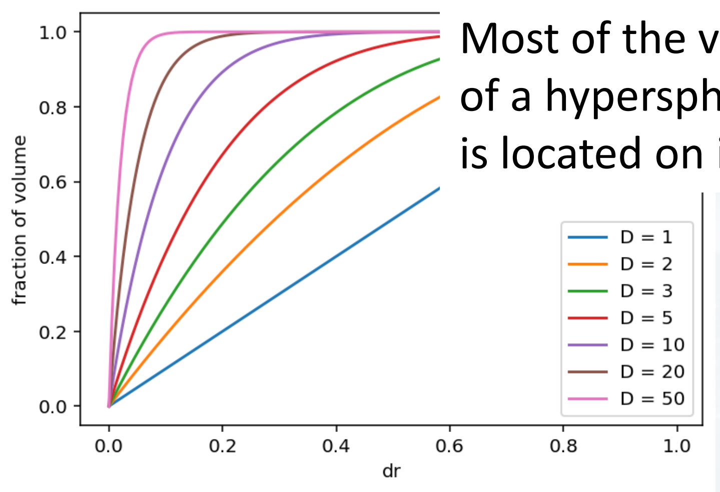
fraction of volume between r and $r - dr$

$$\frac{V_D(r) - V_D(r-dr)}{V_D(r)} = 1 - \frac{(r-dr)^D}{r^D}$$

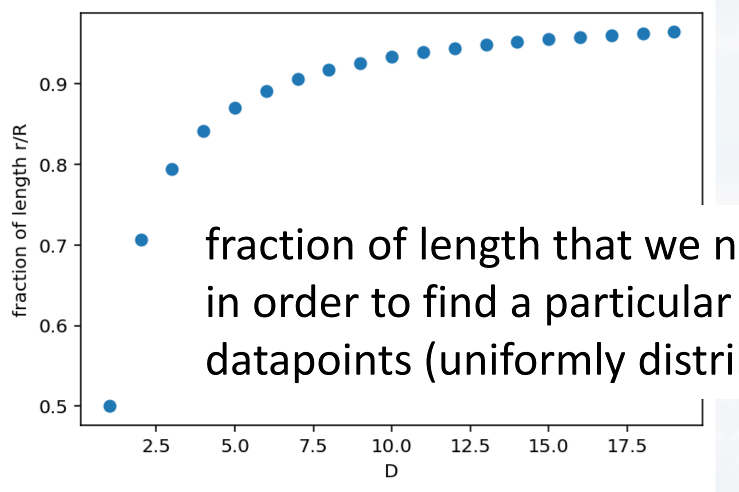


Most of the volume of a hypersphere is located on its surface!

1) **What if linear separation is not possible?**
2) Is there a multiclass SVM?



Most of the volume of a hypersphere is located on its surface!



fraction of length that we need to explore in order to find a particular fraction of datapoints (uniformly distributed in R)

- for **large D (= number of dimensions)**, one has to explore a **larger fraction $\frac{r}{R}$ of the data space** in order to get the **same fraction of data points**

- many algorithms get **less efficient** for large D

- for D→∞, the entire volume is located on the surface of the hyperspace

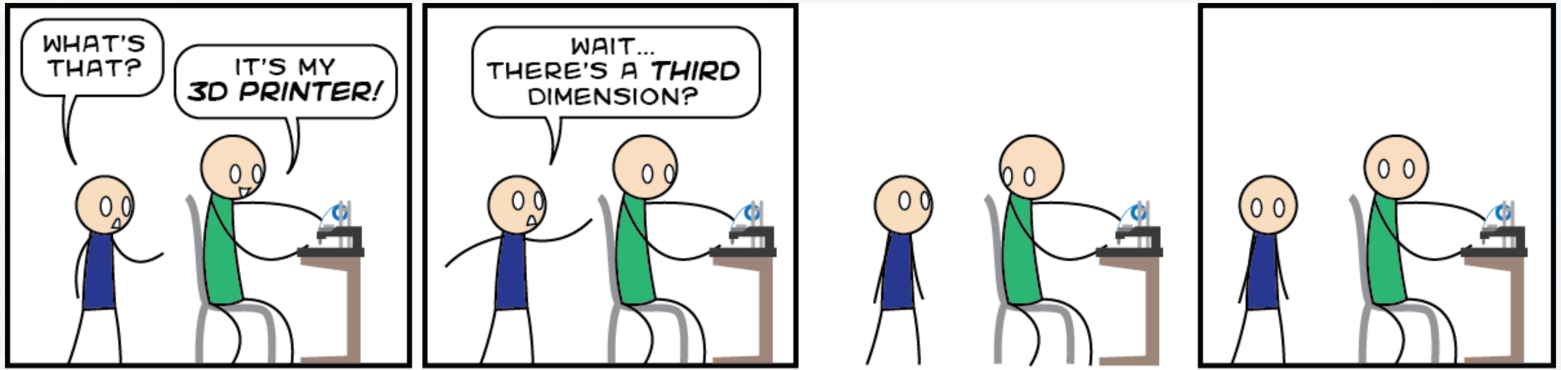
1) **What if linear separation is not possible?**
2) Is there a multiclass SVM?

- for **large D** **(= number of dimensions)**, one has to explore a **larger fraction** $\frac{r}{R}$ **of the data space** in order to get the **same fraction of data points**
- many algorithms get **less efficient** for large D
- for D→∞, the entire volume is located on the surface of the hyperspace

Makes many machine learning algorithms less efficient!

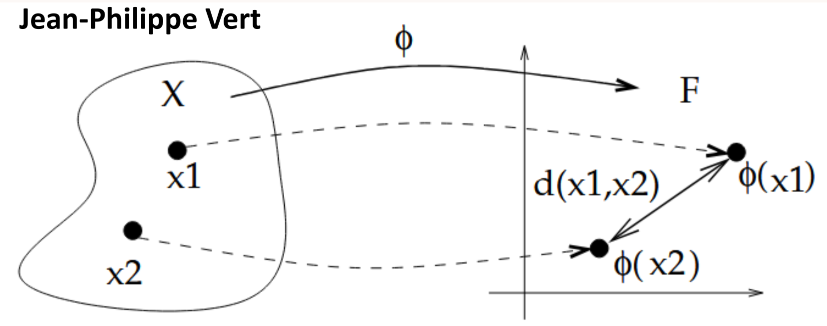
"Curse of Dimensionality"

Dimensionality reduction is important!



©2015 Awkward Turtle World

1) **What if linear separation is not possible?**
2) Is there a multiclass SVM?


Jean-Philippe Vert

problems: - computationally intensive
- ϕ usually unknown

idea: - entire mathematical framework not needed
- for separation: need distances *d* in data space and feature space

$$d^2(x,y) = \langle x-y, x-y\rangle$$

$$d_\phi^2\big(\phi(x),\phi(y)\big) = \langle \phi(x)-\phi(y), \phi(x)-\phi(y)\rangle$$

$$= \langle \phi(x),\phi(x)\rangle - 2\langle \phi(x),\phi(y)\rangle + \langle \phi(y),\phi(y)\rangle$$

kernel $\mathrm{K}(x,y) := \langle \phi(x),\phi(y)\rangle$

$$d_\phi^2\big(\phi(x),\phi(y)\big) = K(x,x) - 2K(x,y) + K(y,y)$$

kernel trick: - we don't know *K* either: **we guess it!**

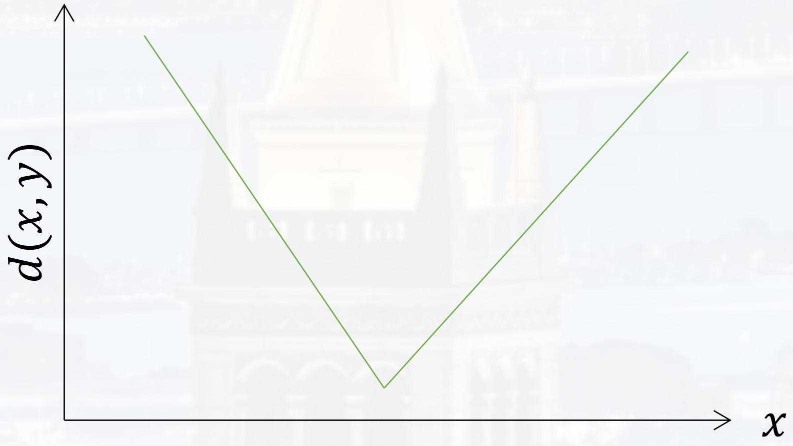
1) **What if linear separation is not possible?**
2) Is there a multiclass SVM?

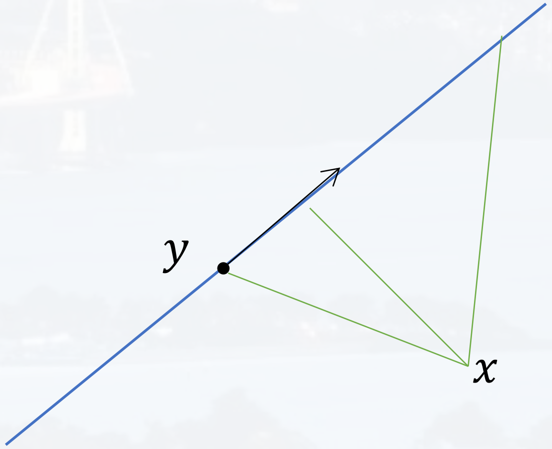
$$d^2\big(\phi(x),\phi(y)\big) = K(x,x) - 2K(x,y) + K(y,y)$$

example: linear kernel $\quad \phi\colon \mathrm{x} \mapsto \mathrm{x} \quad \phi(x) = x$

$$d_\phi^2(x,y) = \langle x,x\rangle - 2\langle x,y\rangle + \langle y,y\rangle = x^2 - 2xy + y^2 = (x-y)^2$$

$$d_\phi^2(x,y) = |x-y|$$

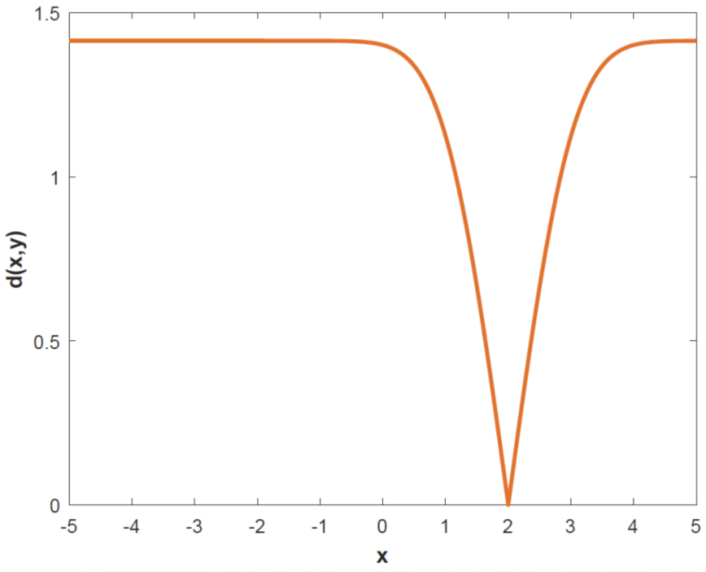


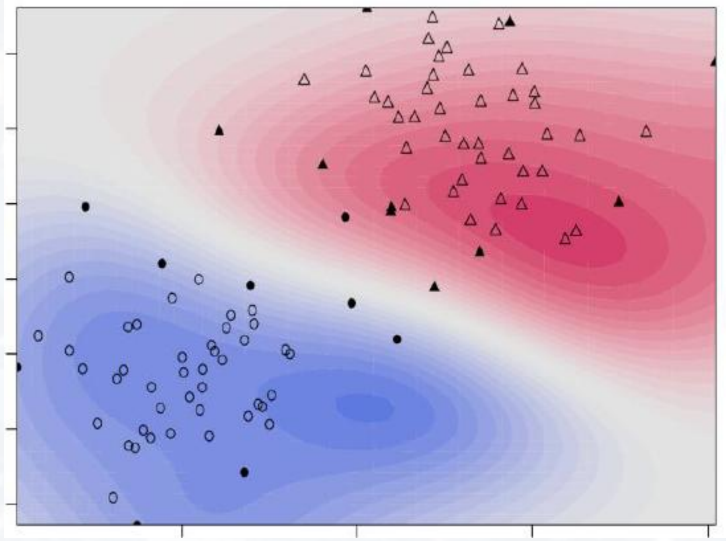


1) **What if linear separation is not possible?**
2) Is there a multiclass SVM?

example: : RBF (**r**adial **b**asis **f**unction) $\phi\colon \mathrm{x} \mapsto \varsigma$ $\phi(x) = \sim\exp(\sim x)$

$$K(x,y) = exp\left(-\frac{\|x-y\|^2}{2\sigma^2}\right) \qquad d_\phi^2(\phi(x),\phi(y)) = 2\left[1 - exp\left(-\frac{\|x-y\|^2}{2\sigma^2}\right)\right]$$





ML Overview

1) **What if linear separation is not possible?**
2) Is there a multiclass SVM?

kernels available in `sklearn`:

- linear: $K(x,y) = \|x-y\|$

- Gaussian aka RBF (**r**adial **b**asis **f**unction): $K(x,y) = exp\left(-\frac{\|x-y\|^2}{2\sigma^2}\right)$

  in `sklearn` we can adjust $\boldsymbol{\gamma} := \frac{\mathbf{1}}{\mathbf{2\sigma^2}}$

- polynomial: $K(x,y) = \sum_{n=1}^{N} \|x-y\|^n$

  in `sklearn` we can adjust $\boldsymbol{N}$

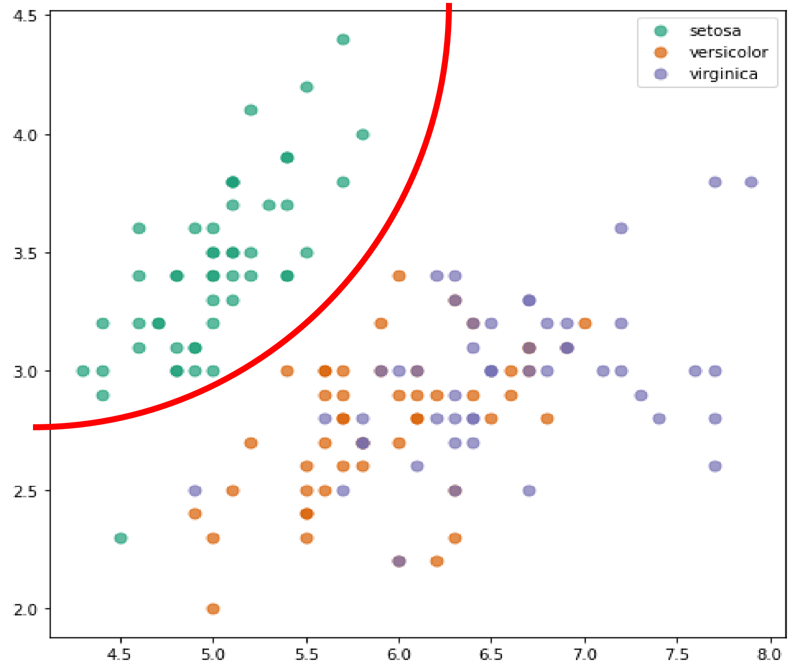
- sigmoidal: $K(x,y) = \frac{e^{\|x-y\|}}{1+e^{\|x-y\|}}$

1) What if linear separation is not possible?

2) Is there a multiclass SVM?

→ one vs rest / one vs one

green vs the rest
→ storing probabilities

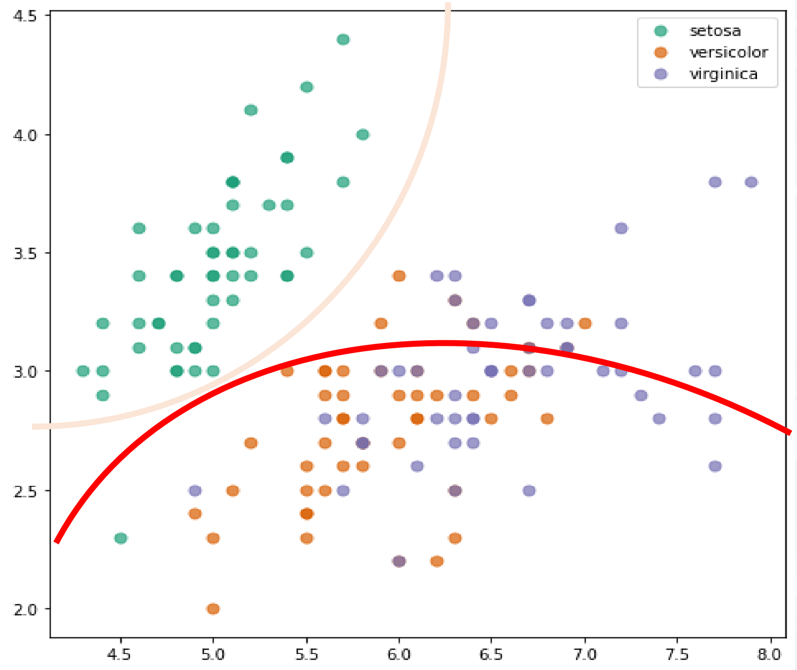


1) What if linear separation is not possible?

2) **Is there a multiclass SVM?**

→ one vs rest / one vs one

green vs the rest
→ storing probabilities



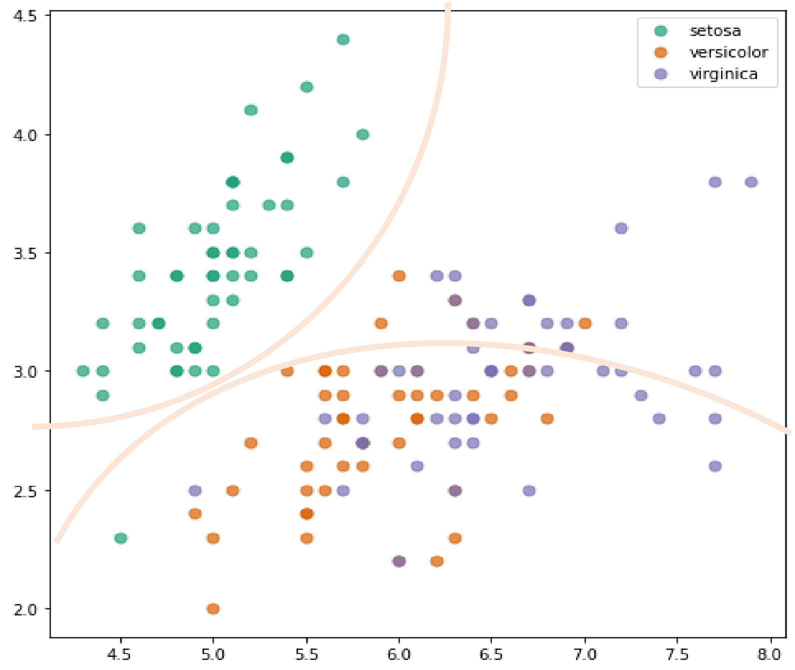
orange vs the rest
→ storing probabilities

ML Overview

1) What if linear separation is not possible?
2) **Is there a multiclass SVM?**

→ one vs rest / one vs one

green vs the rest
→ storing probabilities



blue vs the rest
→ storing probabilities

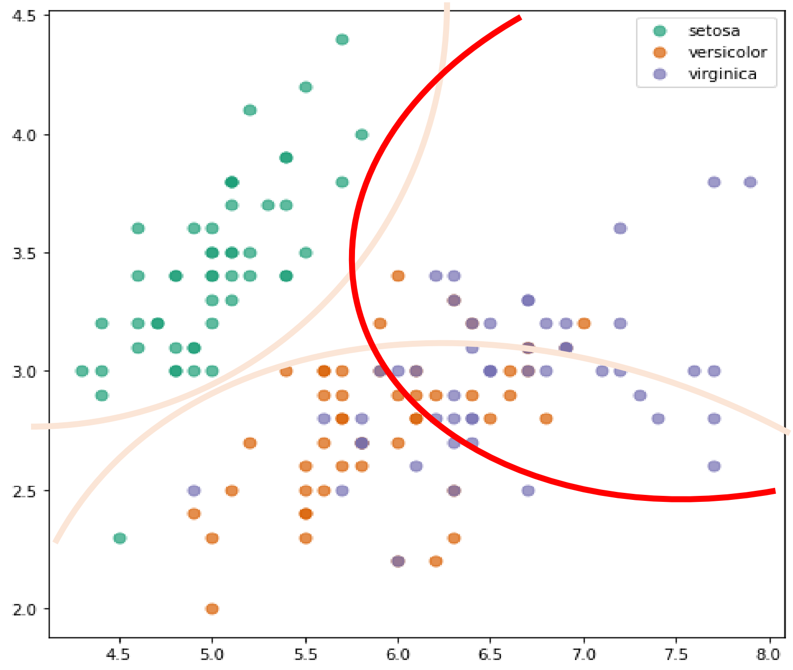
orange vs the rest
→ storing probabilities

ML Overview

1) What if linear separation is not possible?
2) **Is there a multiclass SVM?**

→ one vs rest / one vs one

green vs the rest
→ storing probabilities



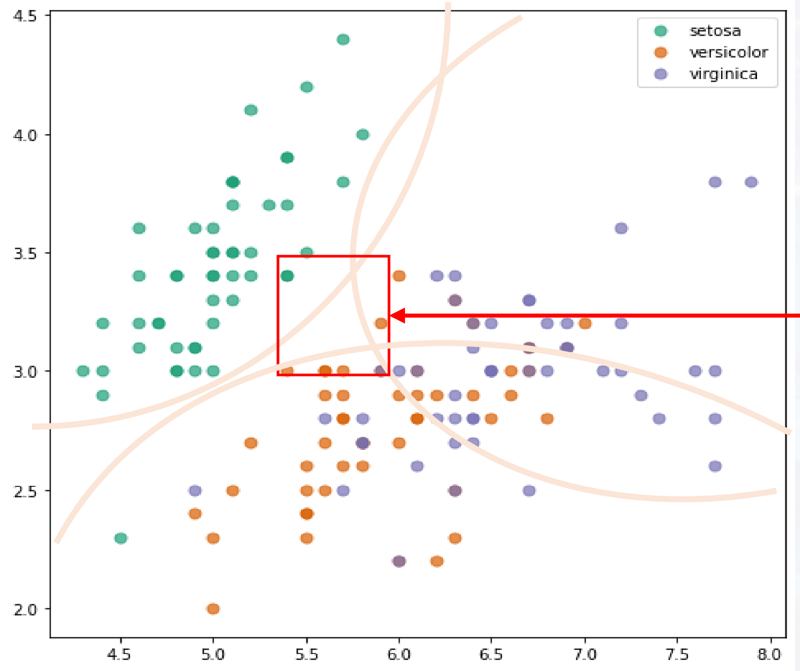
blue vs the rest
→ storing probabilities

orange vs the rest
→ storing probabilities

1) What if linear separation is not possible?
2) **Is there a multiclass SVM?**

→ one vs rest / one vs one

green vs the rest
→ storing probabilities



three probabilities for each data point
→ assign class to most probable value

k class classification with two-class discriminant functions is ambiguous!

ML Overview

SVM

```
from sklearn import svm
```

see **Walk_Through_SVM.ipynb**

1) setting up the model & 2) fitting the model

running analysis with different kernel

```
outlinear = svm.SVC(kernel = 'linear', C = 1, decision_function_shape = 'ovr')
linear    = outlinear.fit(X2D, Y)
```

one **v**ersus **r**est

L2 regularization parameter for error tolerance when calculating the classifier

```
outrbf    = svm.SVC(kernel = 'rbf', gamma = 1, C = 1, \
                            decision_function_shape = 'ovr')

rbf       = outrbf.fit(X2D, Y)
```

$$\gamma := \frac{1}{2\sigma^2}$$

```
outpoly   = svm.SVC(kernel = 'poly', degree = 3, C = 1,\
                            decision_function_shape = 'ovr')

poly      = outpoly.fit(X2D, Y)
```

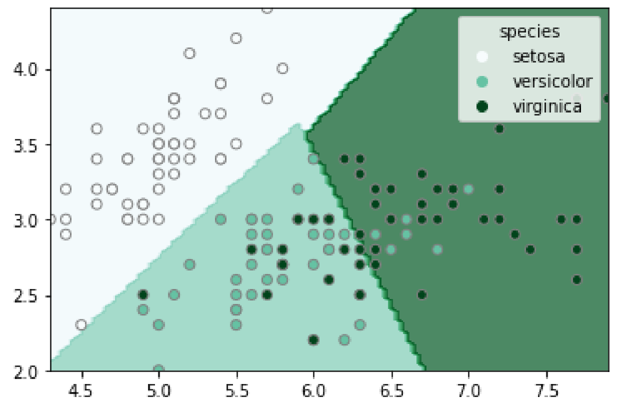
refers to *N* in $\sum_{n=1}^{N} \|x-y\|^n$

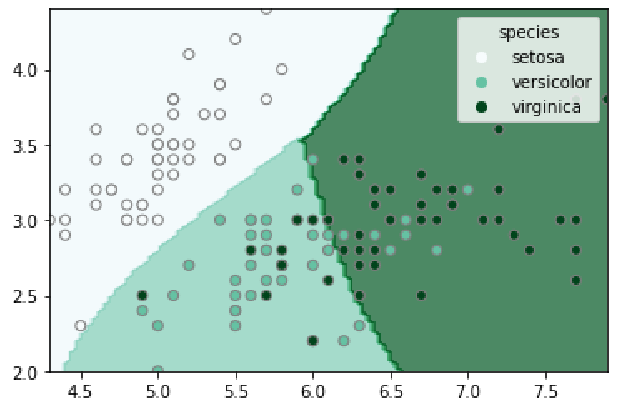
```
outsig    = svm.SVC(kernel = 'sigmoid', C = 1, decision_function_shape = 'ovr')
sig       = outsig.fit(X2D, Y)
```

```
from sklearn import svm
```

see `Walk_Through_SVM.ipynb`

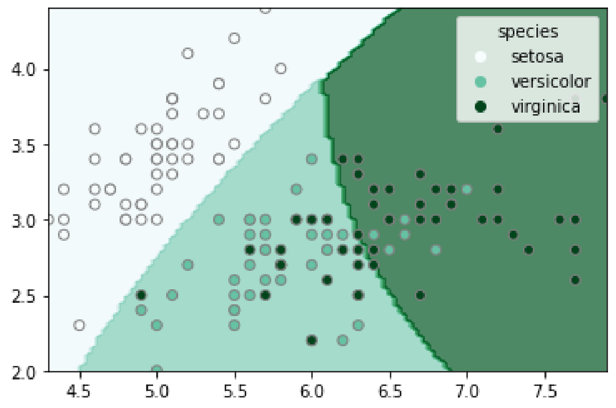
$$K(x, y) = \|x - y\|$$

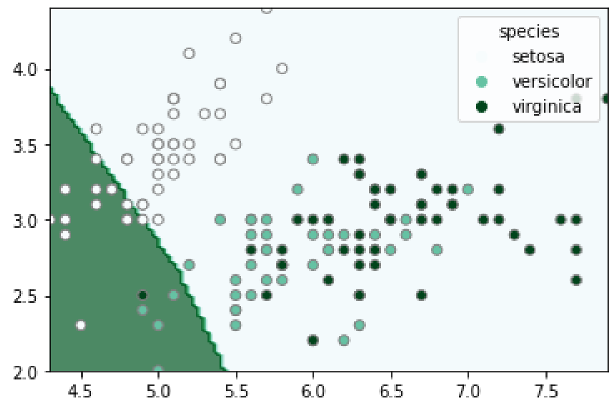




$$K(x, y) = exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right)$$

$$K(x, y) = \sum_{n=1}^{N} \|x - y\|^n$$





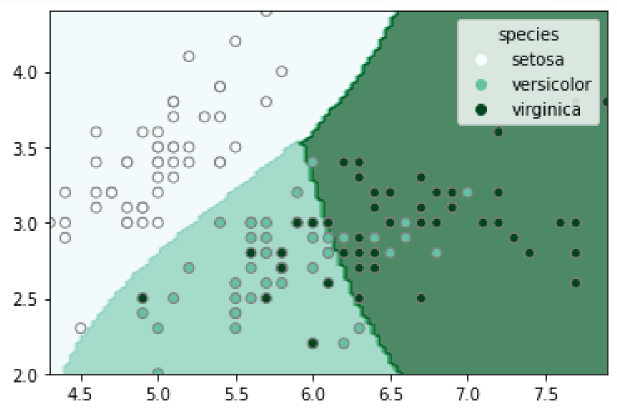
$$K(x, y) = \frac{e^{\|x-y\|}}{1 + e^{\|x-y\|}}$$

ML Overview

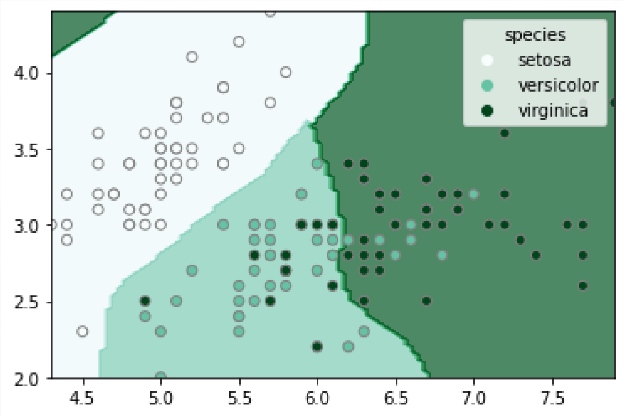
```
from sklearn import svm
```

see `Walk_Through_SVM.ipynb`

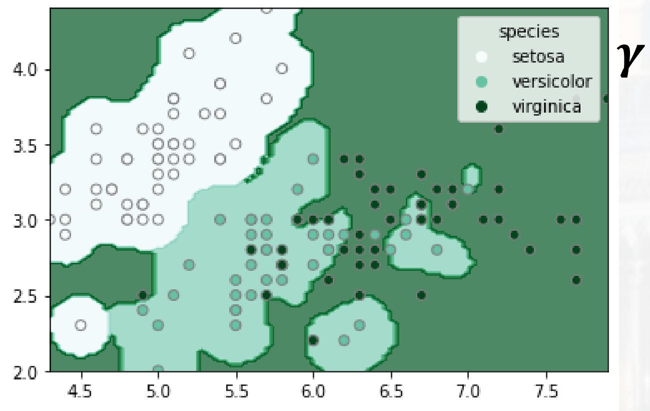
$$K(x,y) = exp\left(-\frac{1}{2\sigma^2}\|x-y\|^2\right)$$



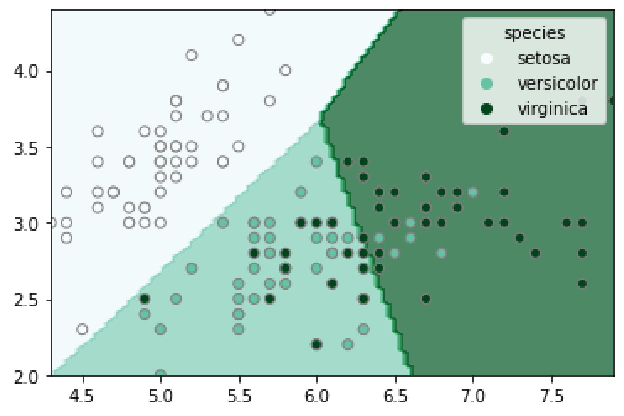
$\gamma := \frac{1}{2\sigma^2} = 1$



$\gamma := \frac{1}{2\sigma^2} = 5$



$\gamma := \frac{1}{2\sigma^2} = 50$



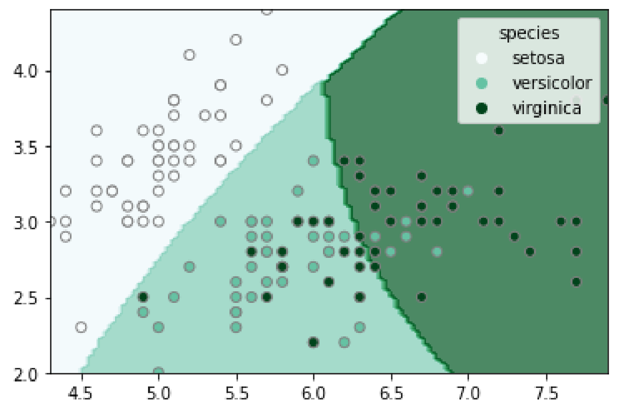
$\gamma := \frac{1}{2\sigma^2} = 0.1$

ML Overview

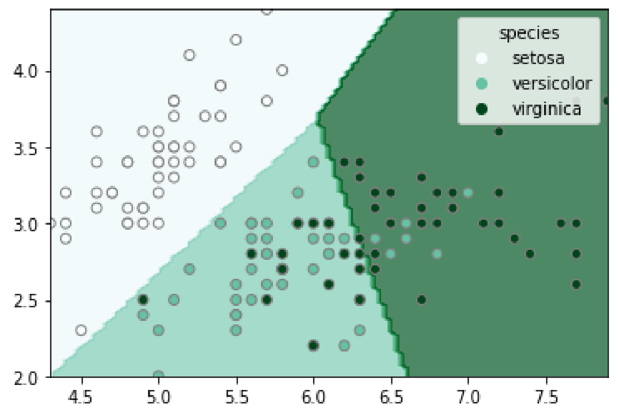
```
from sklearn import svm
```

see `Walk_Through_SVM.ipynb`

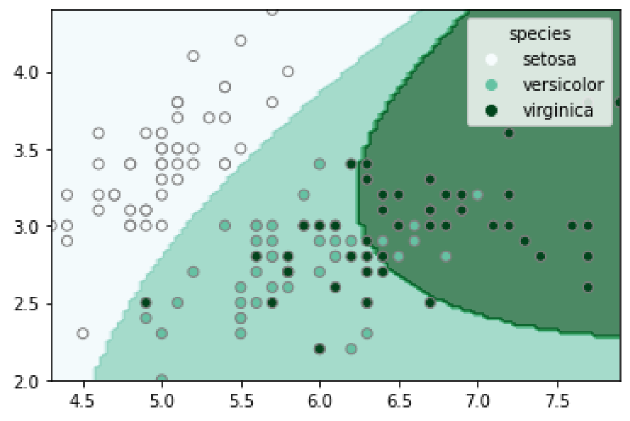
$$K(x, y) = \sum_{n=1}^{N} \|x - y\|^n$$



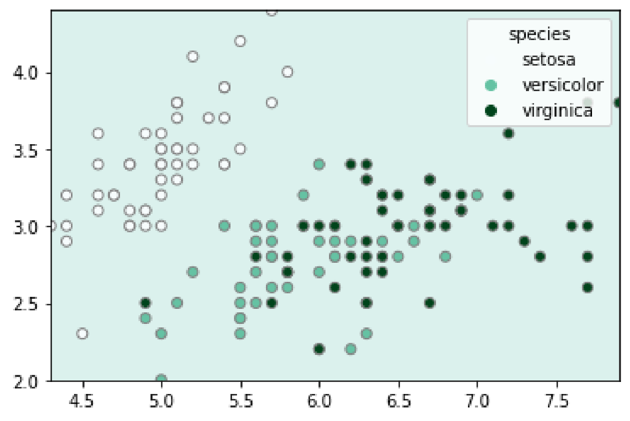
N = 3



N = 1



N = 5



N = 0

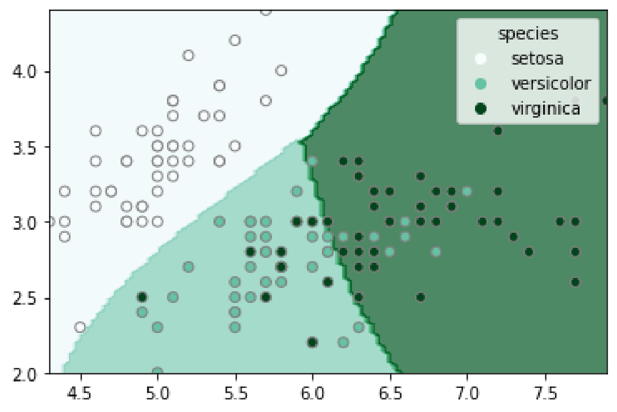
ML Overview

```
from sklearn import svm
```

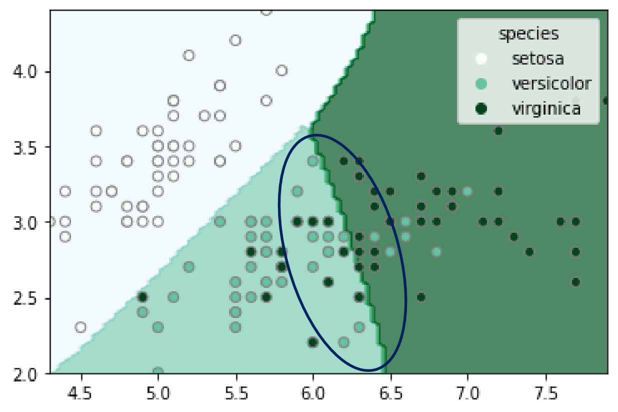
see `Walk_Through_SVM.ipynb`

$$K(x,y) = exp\left(-\frac{1}{2\sigma^2}\|x-y\|^2\right)$$

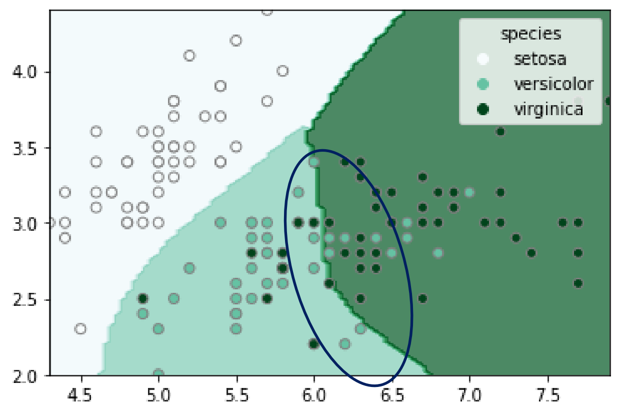
C is a L2 regularization parameter



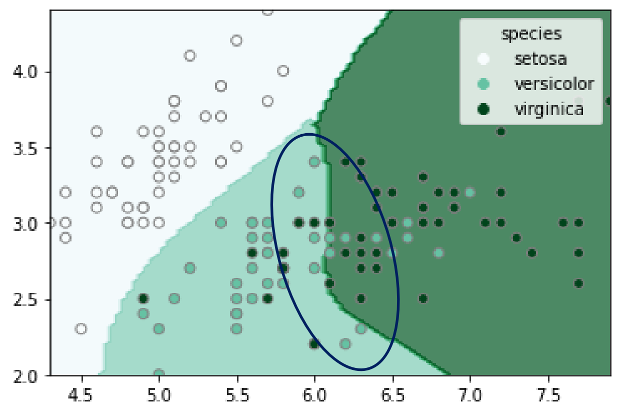
C = 1



C = 0.1



C = 10



C = 20

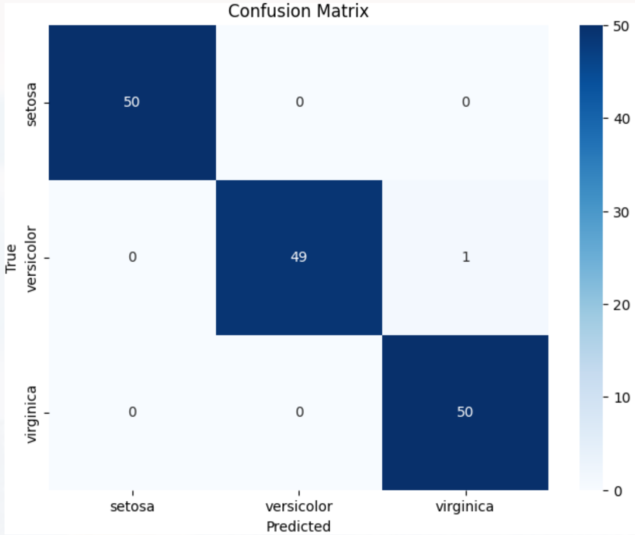
ML Overview

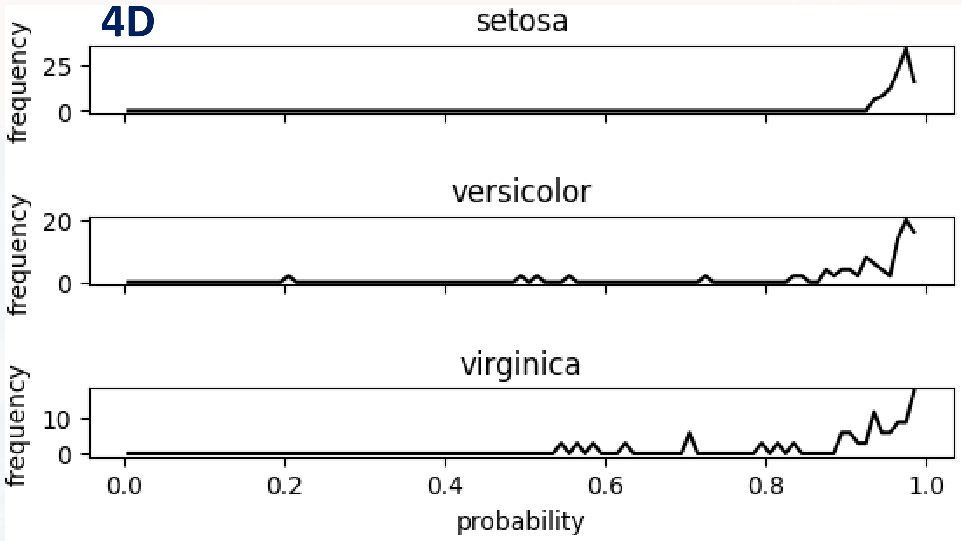
full **4D** data set

3) evaluating the model

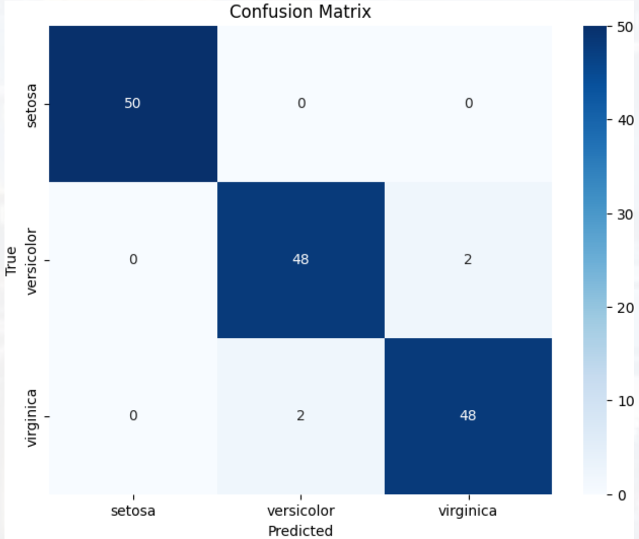
see **`Walk_Through_SVM.ipynb`**

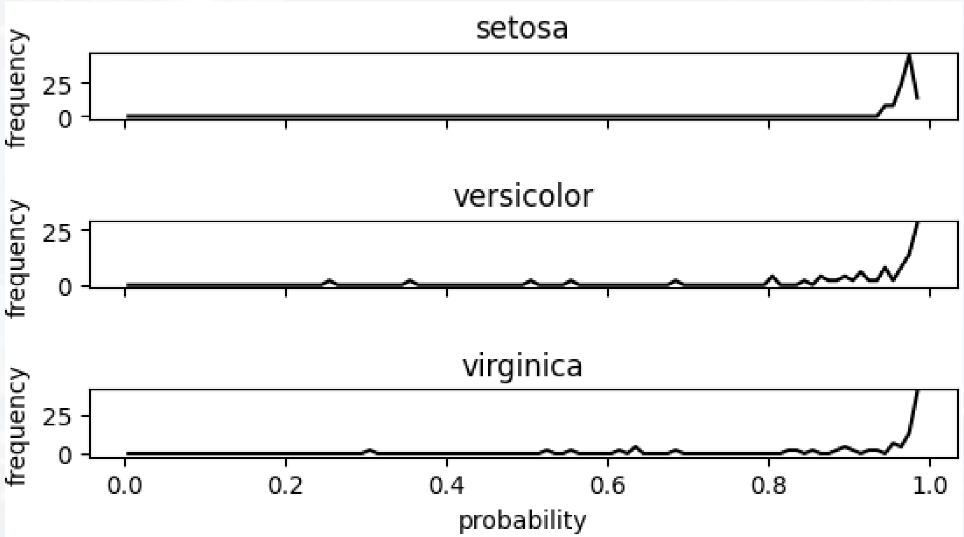
linear
accuracy: 99.3%





Gaussian ($\gamma = 1$)
accuracy: 97.3%





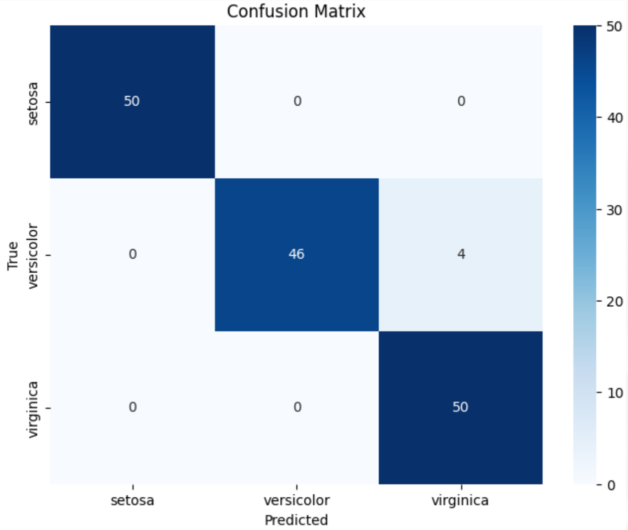
ML Overview

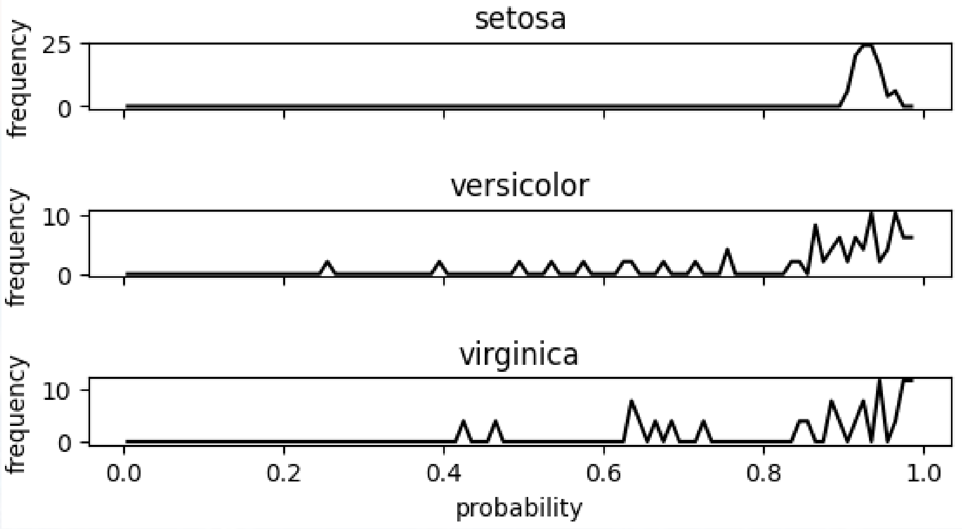
full **4D** data set

3) evaluating the model

see **`Walk_Through_SVM.ipynb`**

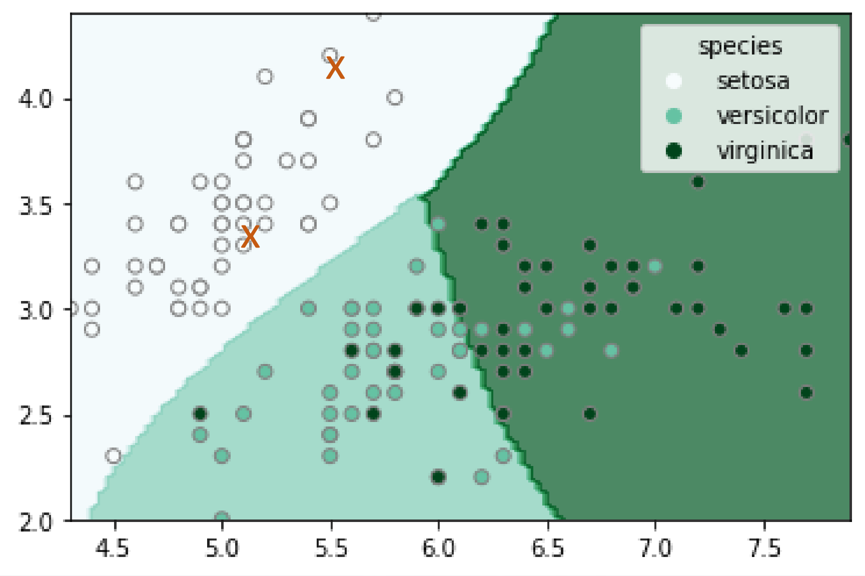
polynomial (n = 3)
accuracy: 97.3%





4) applying the model to a new data set

```
ypred = outpoly.predict([[6, 3.5],[6.3, 4.5]])
```



Machine Learning Overview & SVM

Thank you very much for your attention!