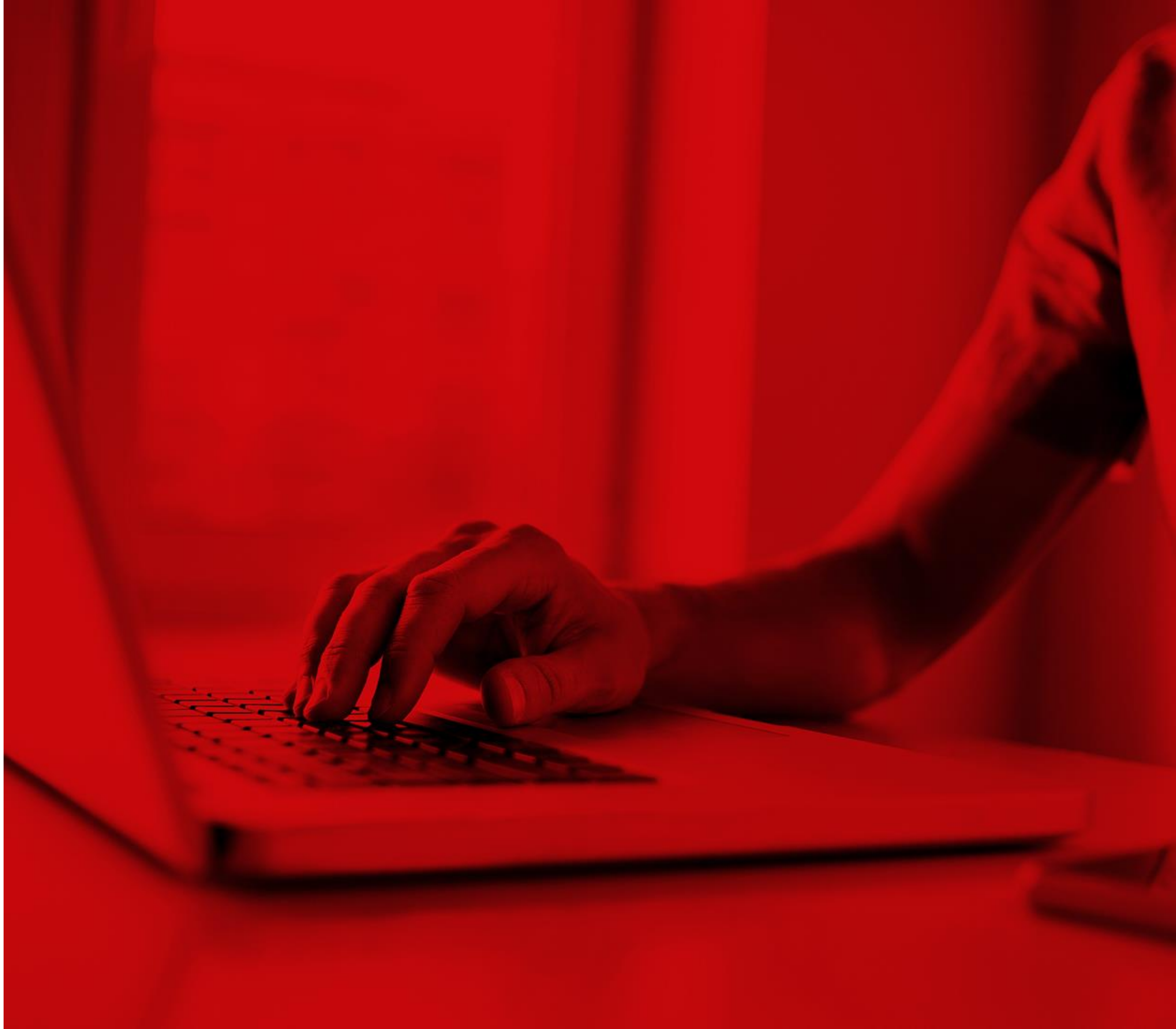
COURSE IN MACHINE LEARNING



AGENDA

Intro to ML

1. Data
2. Clustering
3. Classification
4. Regression

Tasks

1. Spam filter
2. Sentiment analysis

Solving the problem

1. Feature Extraction
2. Choosing a model
3. Splitting the data
4. Measuring the results

Your turn

1. SMS: Spam vs Ham
2. Rating prediction

WHO ARE WE?

- Joakim Myrvoll Johansen
- Consultant @ itera
- Works with Churn Prediction

- Håkon Dissen
- Consultant @ itera
- Works with recommendation systems at RiksTV

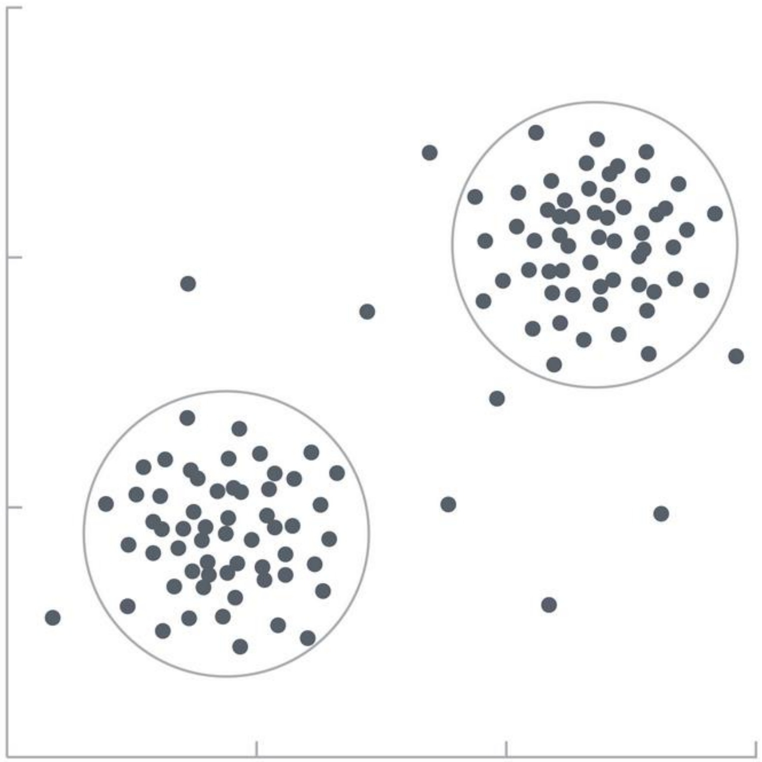
WHAT IS «DATA»?

- Features
  - Describes a single data-point
- Labels
  - Descibes the memebership of the data-point
  - Not always needed

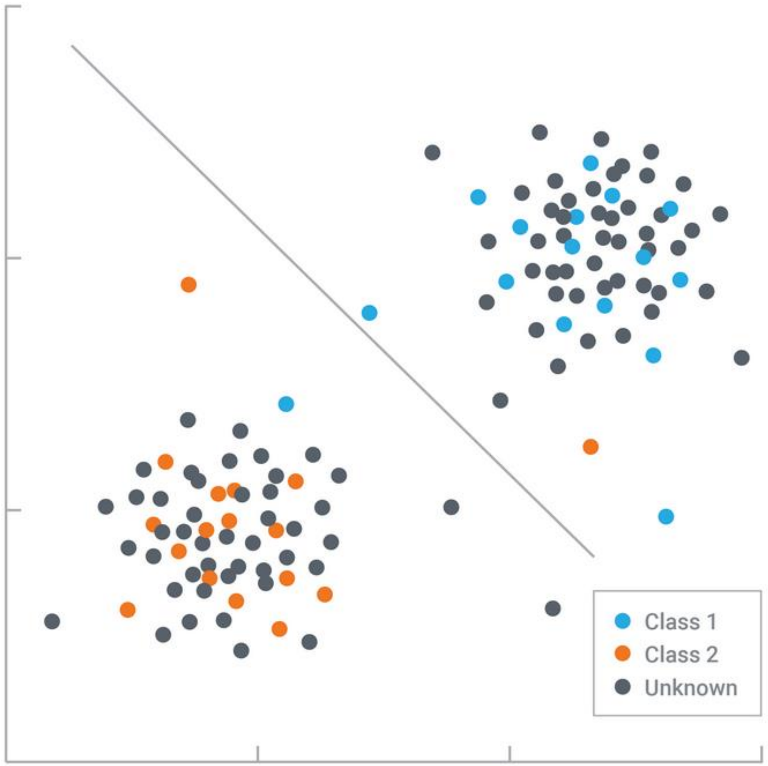
$$\mathbf{X} = \begin{pmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,n} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m,1} & x_{m,2} & \cdots & x_{m,n} \end{pmatrix} \qquad \mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix}$$

THE TWO SCHOOLS (TOOLS)

Unsupervised



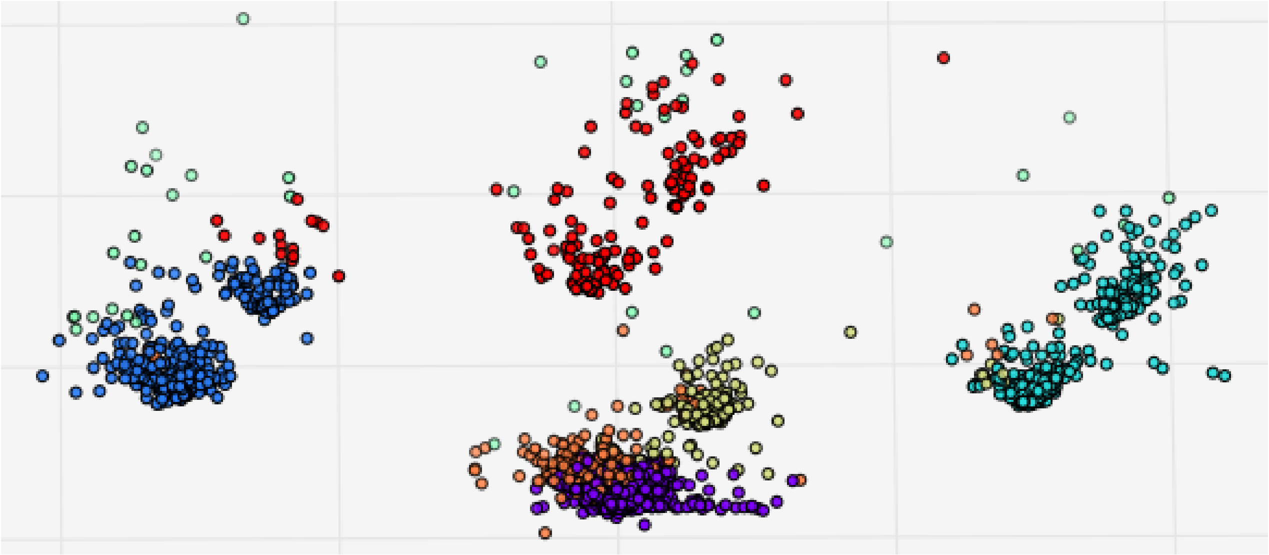
Supervised



Shamelessly ripped from: http://blog.sqrrl.com/an-introduction-to-machine-learning-for-cybersecurity-and-threat-hunting

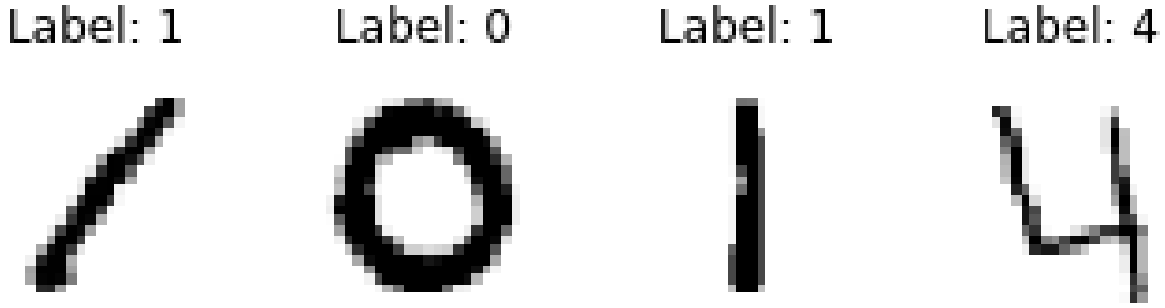
CLUSTERING

- Grouping similar datapoints
- Which datapoints belong together
- What users watch the same TV-shows?



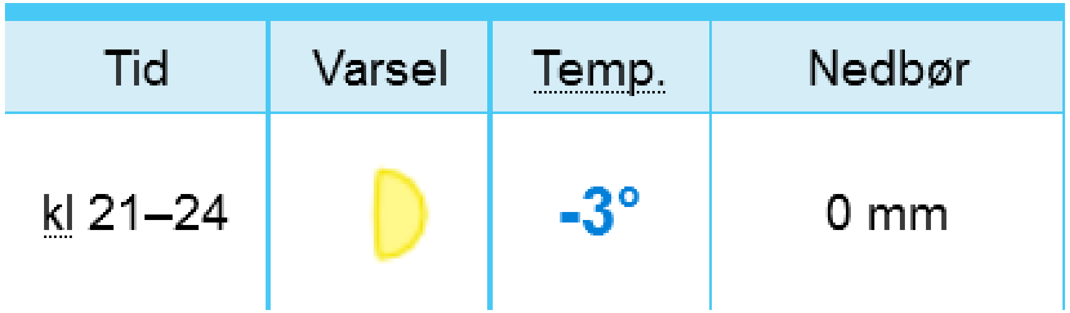
CLASSIFICATION

- Which *label* belongs to which observation
- We need *labeled* data
- Example: what number is this a picture of



REGRESSION

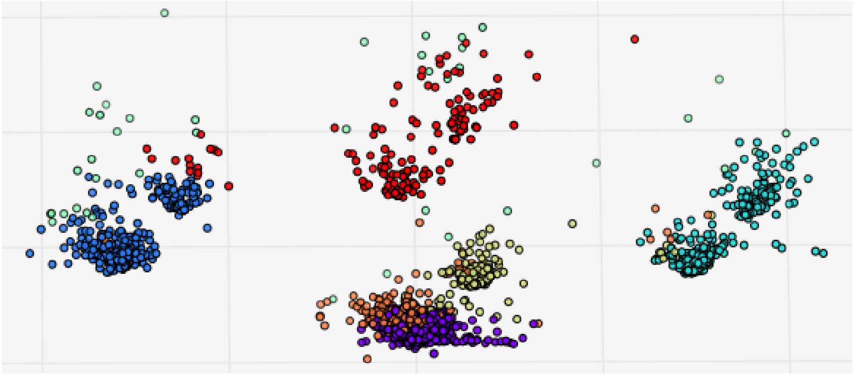
- Predict continuous values
- Weather forecasting
- Stock prices



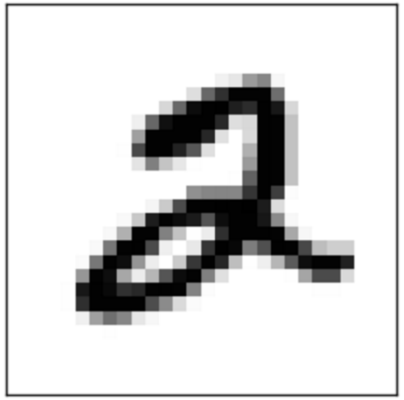
Tid	Varsel	Temp.	Nedbør
kl 21–24	◗	-3°	0 mm

MACHINE LEARNING BASICS

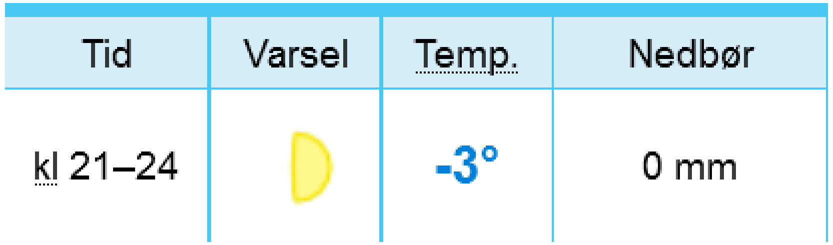
- Clustering
  - Find groups of observations



- Classification
  - Which *label* belongs to which observation?



- Regression
  - Continuous values
- …



Tid	Varsel	Temp.	Nedbør
kl 21–24	◗	-3°	0 mm

AGENDA

Intro to ML

1. Data
2. Clustering
3. Classification
4. Regression

Tasks

1. Spam filter
2. Sentiment analysis

Solving the problem

1. Feature Extraction
2. Choosing a model
3. Splitting the data
4. Measuring the results

Your turn

1. SMS: Spam vs Ham
2. Rating prediction

TASKS

- 1. Spam filter
- 2. Sentiment analysis
- Extra:
  - Number (bitmap) classification

SPAM VS HAM

- Many text messages with a *label* indicating it as *spam* or *ham.*

- Only text, no headers.

- A common data-set. (Sorry if someone here has seen it before)

- In this task we want to classify text messages as ham or spam.

THE PROBLEM

XXXMobileMovieClub:
To use your credit, click the WAP link
 in the next txt message or click here>>
 http://wap. xxxmobilemovieclub.
com?n=QJKGIGHJJGCBL

= ?

He like not v shock leh. Cos telling
shuhui is like telling leona also. Like
dat almost all know liao. He got ask me
abt ur reaction lor.

= ?

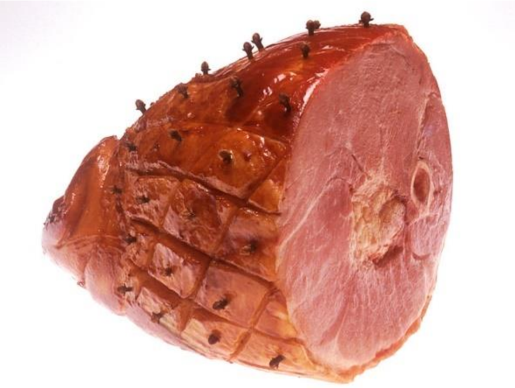
THE PROBLEM

XXXMobileMovieClub:
To use your credit, click the WAP link
in the next txt message or click here>>
http://wap. xxxmobilemovieclub.
com?n=QJKGIGHJJGCBL

=

He like not v shock leh. Cos telling
shuhui is like telling leona also. Like
dat almost all know liao. He got ask me
abt ur reaction lor.

=



SENTIMENT ANALYSIS

- Many film reviews with a *label* and a *score* indicating if the movie is good or not*.*

- Only text, no headers.

- In this task we want to use regression to predict what score a review has given.

AGENDA

Intro to ML

1. Data
2. Clustering
3. Classification
4. Regression

Tasks

1. Spam filter
2. Sentiment analysis

Solving the problem

1. Feature Extraction
2. Choosing a model
3. Splitting the data
4. Measuring the results

Your turn

1. SMS: Spam vs Ham
2. Rating prediction

TOOLS

- Python
- Scikit-Learn
  - Numpy
  - Scipy

MACHINE LEARNING STEPS

- Load data to memory
- Withhold some data for testing/verification
- Extract / Select Features
- Train model
- Test model

FEATURE EXTRACTION

-Bag of words

sklearn.feature_extraction.text.CountVectorizer

1. “John likes to watch movies. Mary likes movies too.”

2. “John also likes to watch football games.”

```
[
    "John",
    "likes",
    "to",
    "watch",
    "movies",
    "Mary",
    "too",
    "also",
    "football",
    "games"
]
```

1. [1, 2, 1, 1, 2, 1, 1, 0, 0, 0]

2. [1, 1, 1, 1, 0, 0, 0, 1, 1, 1]

https://en.wikipedia.org/wiki/Bag-of-words_model

TRANSFORMERS

- *Transform* data
- Used for:
  - Preprocessing
  - Feature selection
  - Feature extraction
  - Dimensionality reduction
- Implements *fit* and *transform*

```
documents = get_the_documents()

vectorizer = CountVectorizer(min_df=1)

dt_mat = vectorizer.fit_transform(documents)
```

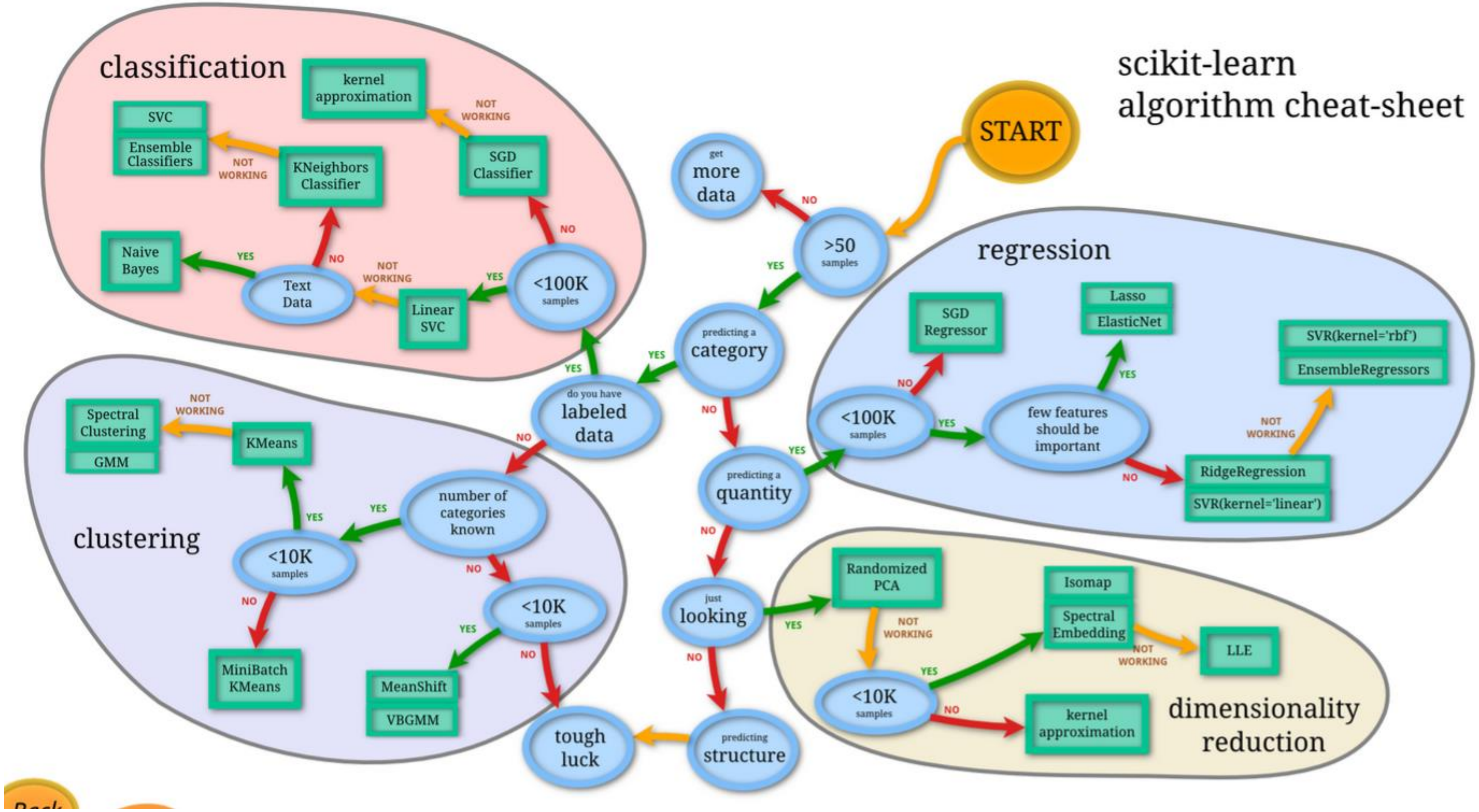
$$\mathbf{dt} = \begin{pmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,n} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m,1} & x_{m,2} & \cdots & x_{m,n} \end{pmatrix}$$

SCIKIT ESTIMATORS

- All learning algorithms
  - Here: a Support Vector Machine
- *Fits* the data, and becomes a *model*
- Implements the *fit* method
- Implements the *score* method

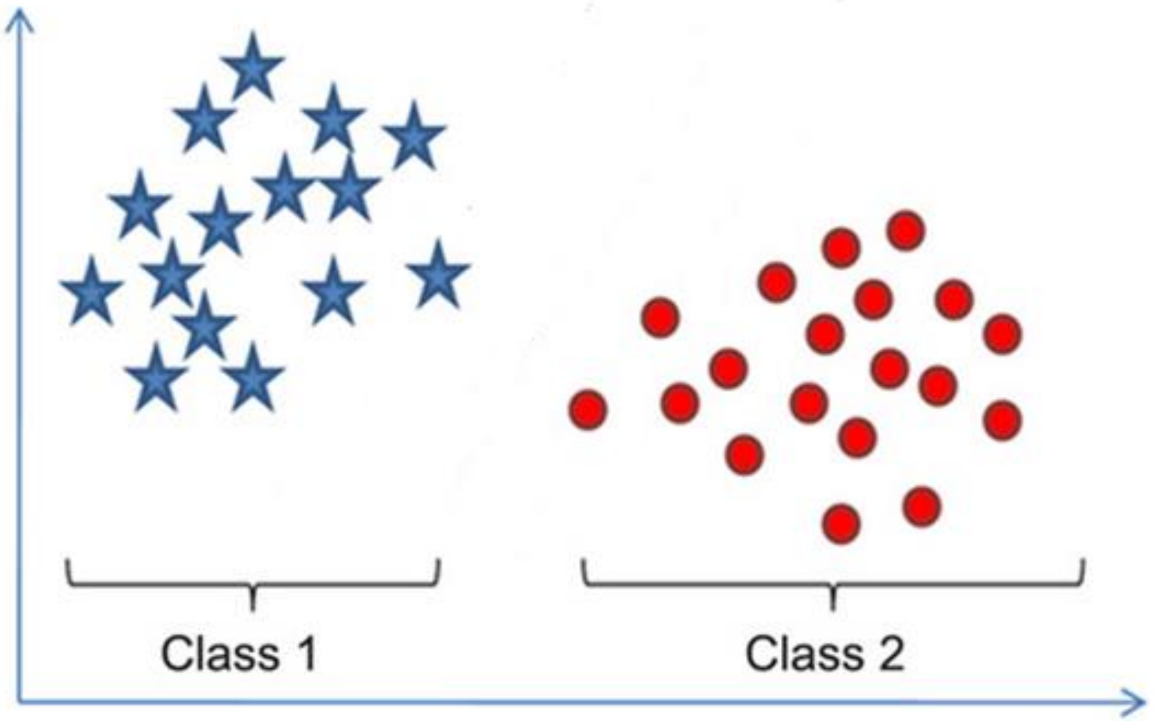
```
estimator = SVC(kernel=‘rbf’)

trained_model = estimator.fit(t_data, t_labels)

predictions = estimator.predict(unseen_data)
```

CHOOSING AN ESTIMATOR



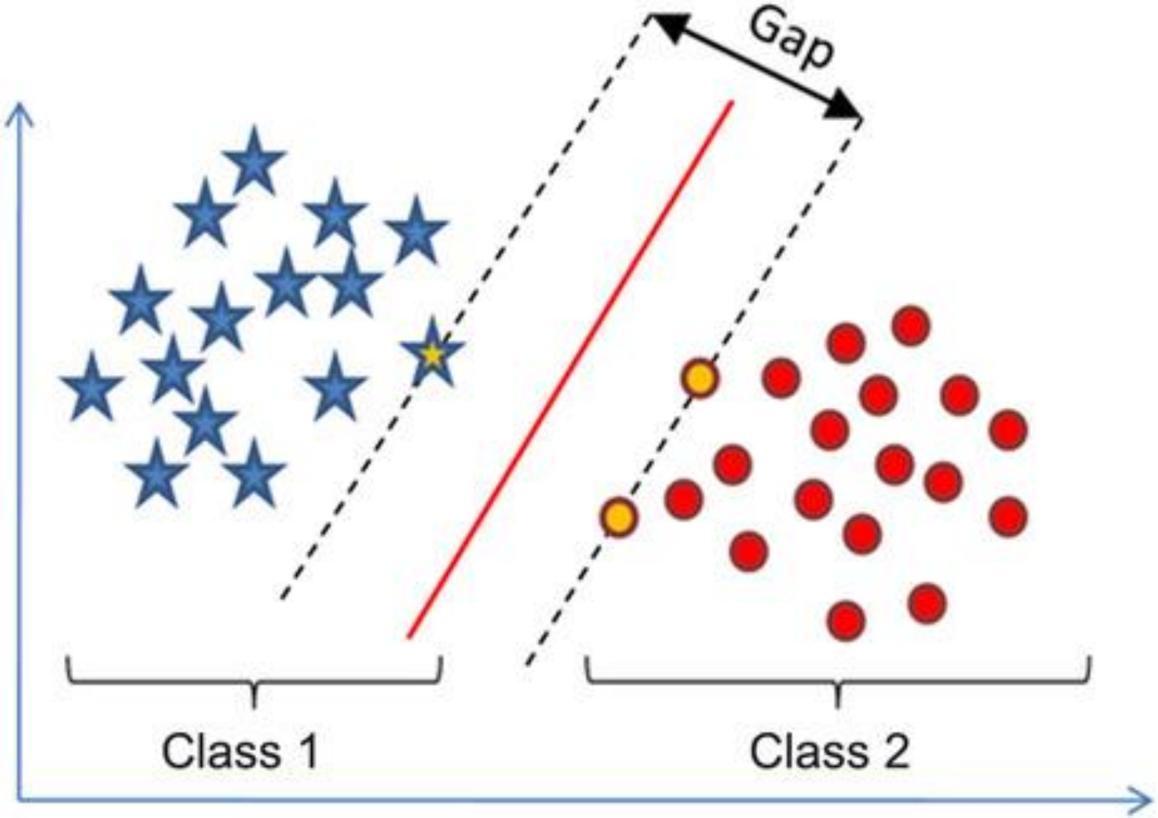
Source: http://scikit-learn.org/stable/tutorial/machine_learning_map/

SUPPORT VECTOR MACHINE (BINARY)



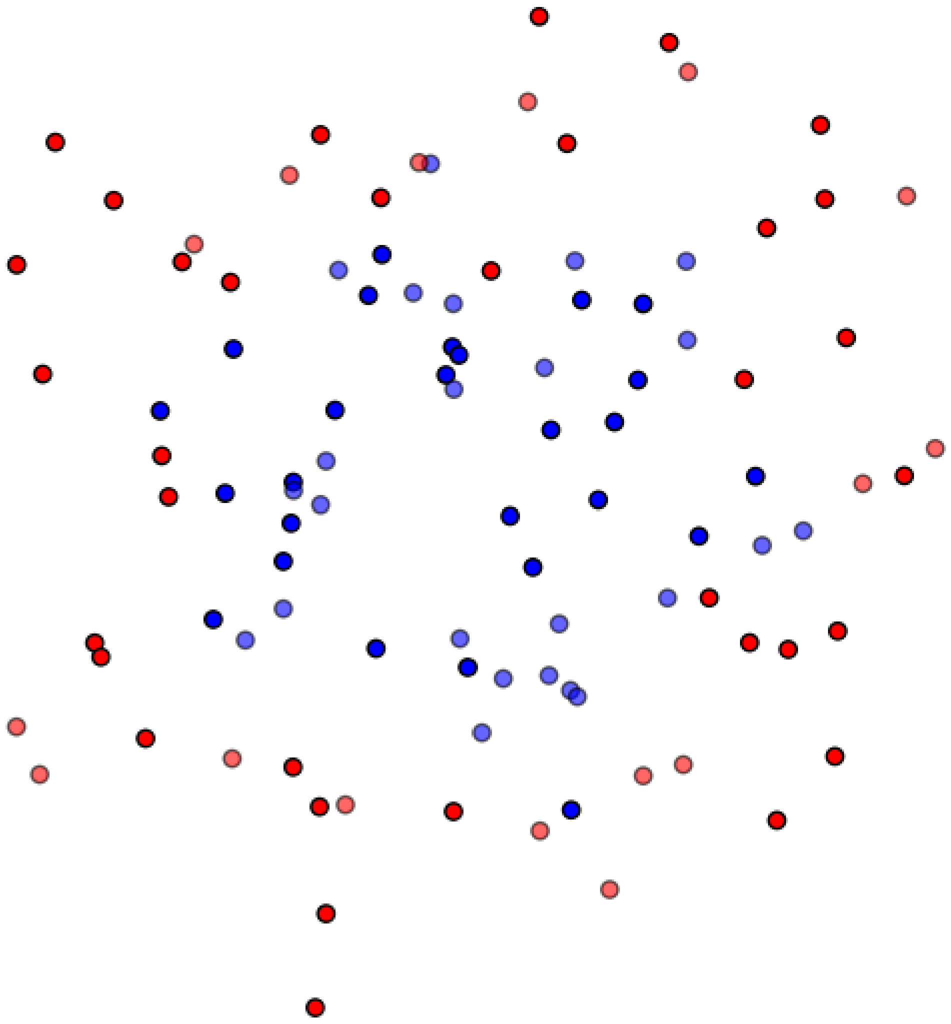
Source: http://diggdata.in/post/94066544971/support-vector-machine-without-tears

SUPPORT VECTOR MACHINE (BINARY)

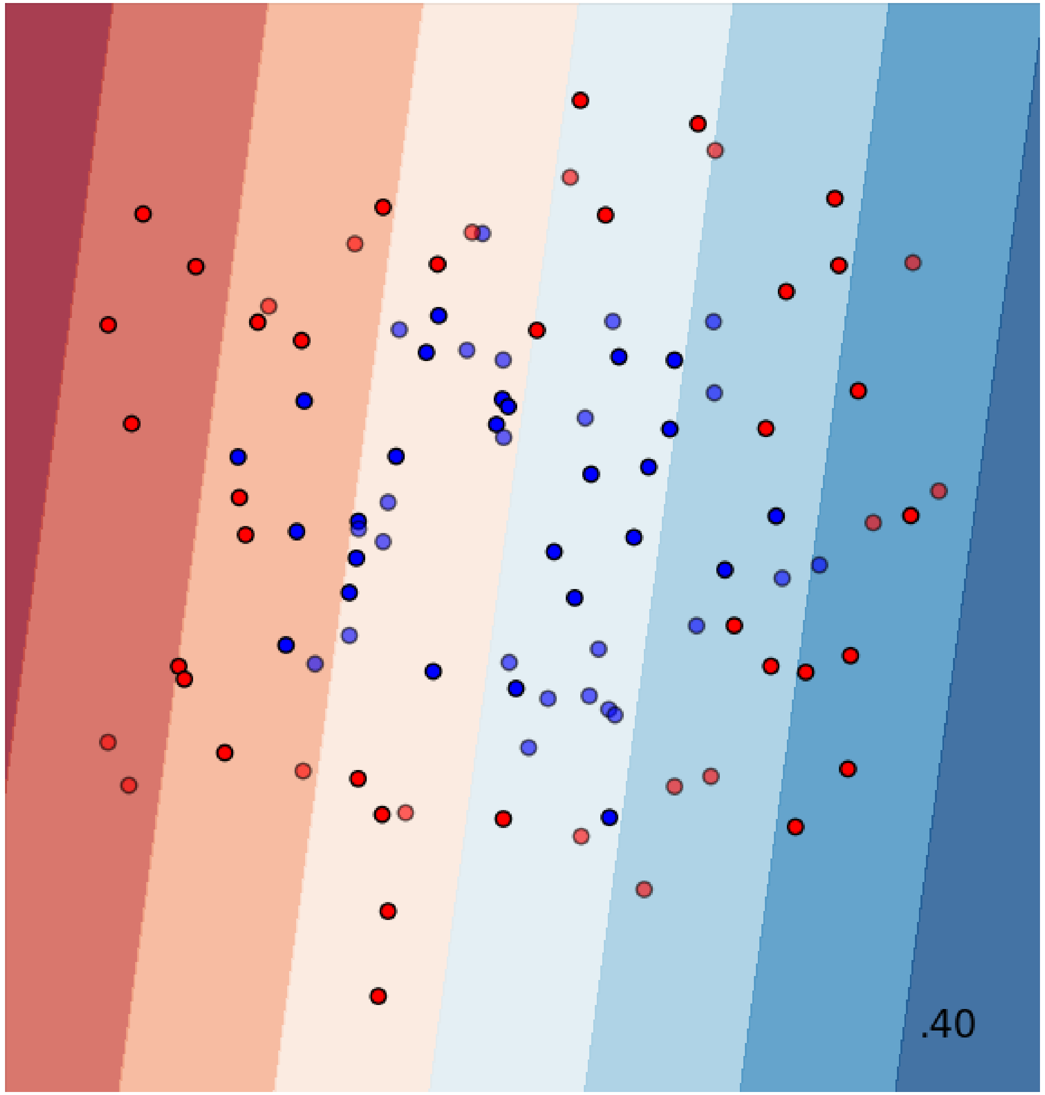


Source: http://diggdata.in/post/94066544971/support-vector-machine-without-tears

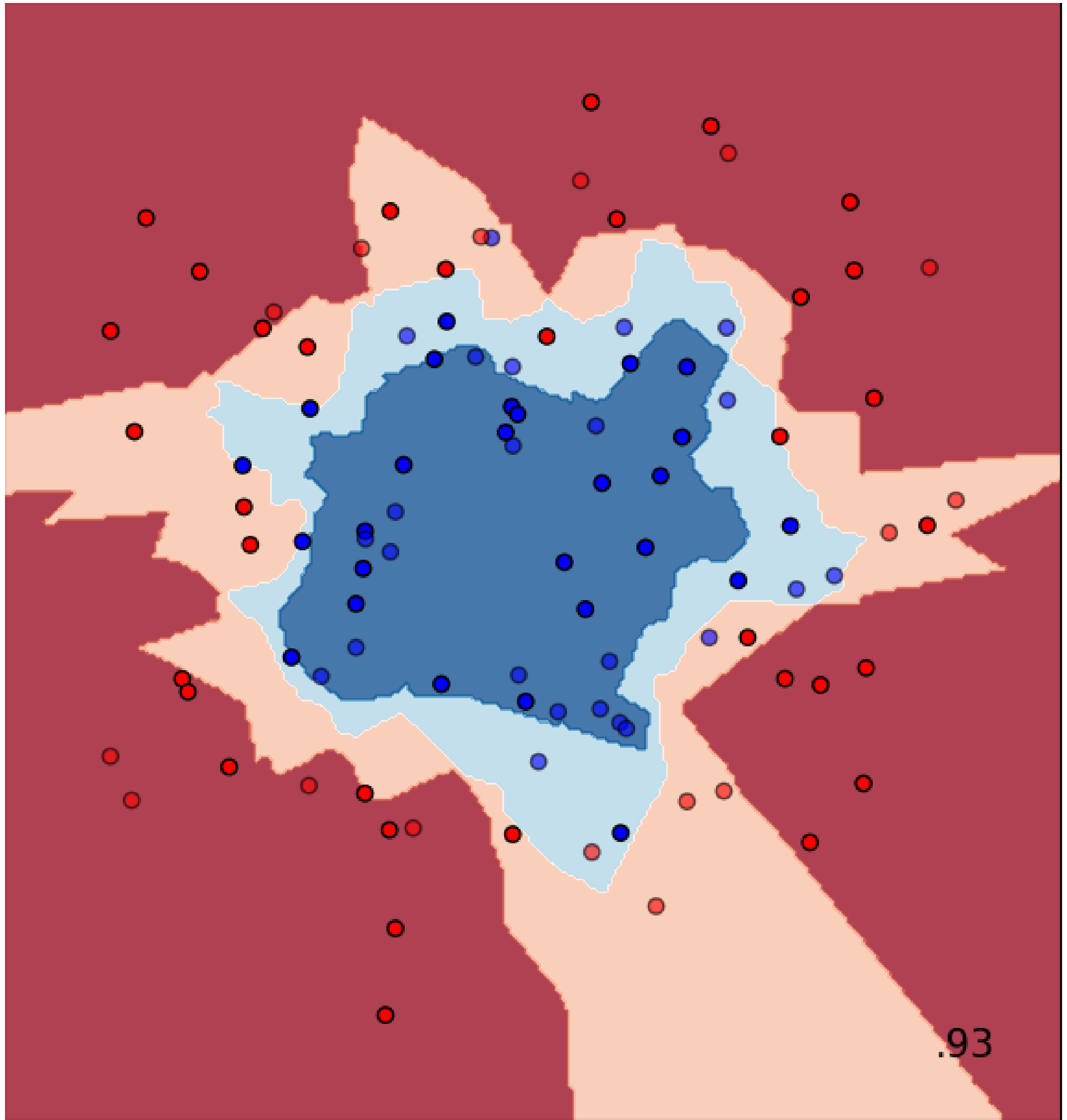
NON-LINEARLY SEPARABLE



LINEAR SVM ON NON-LINEAR DATA

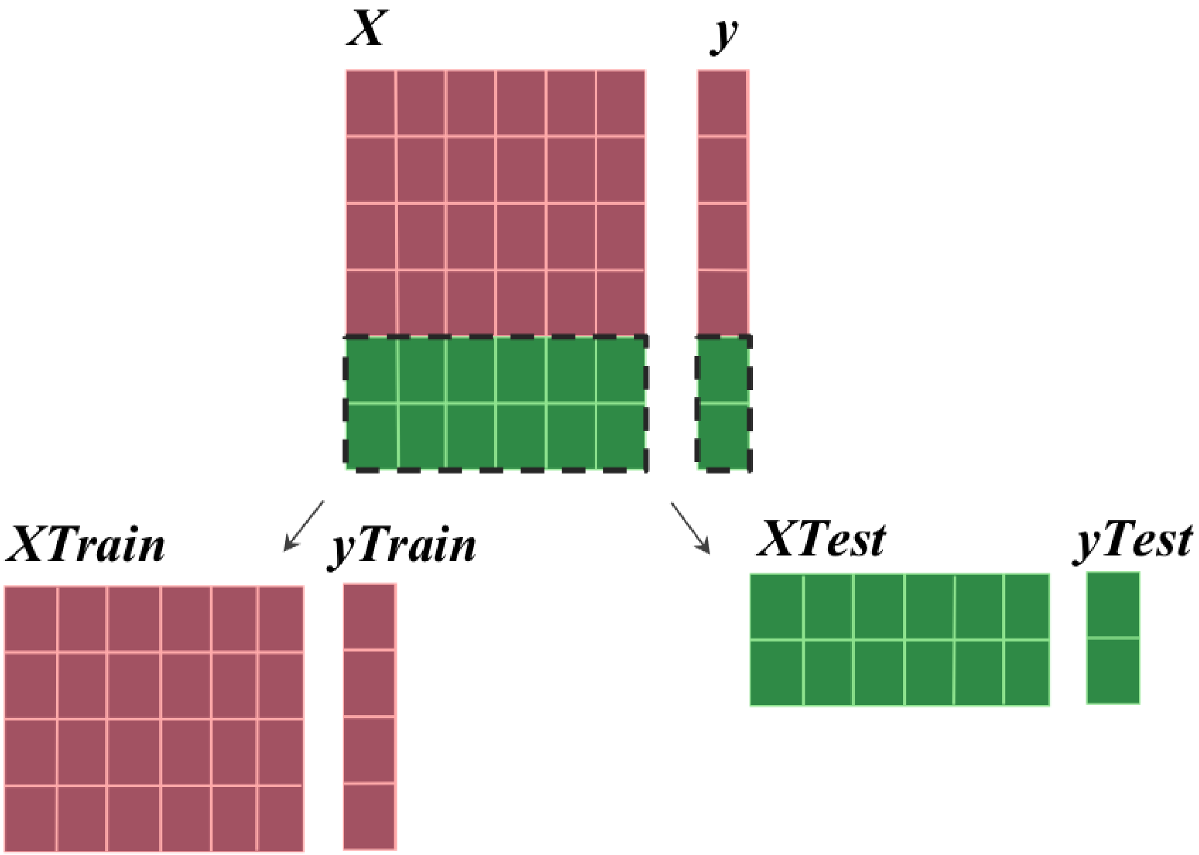


K NEAREST NEIGHBOURS



VALIDATION

- How well is our model fitting to the training data?
  - Not really useful
- Does the model *generalize* well?
  - Super useful
- Split the data in two parts:
  - Training data
  - Test data



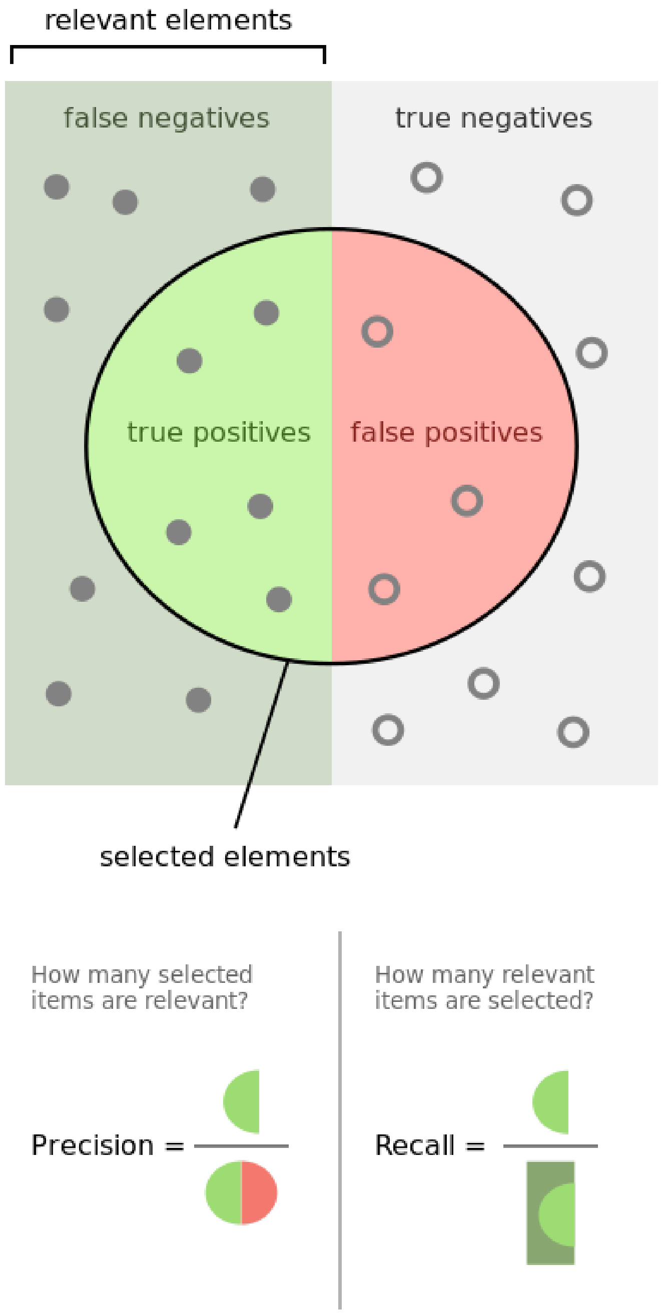
SCORING - CLASS

$$Precsion = \frac{tp}{tp + fp} \qquad Recall = \frac{tp}{tp + fn}$$

$tp =$ something we said **is** spam, that **is** spam

$fp =$ something we said **is** spam, that **is not** spam

$fn =$ something we said **is not** spam, that **is** spam



SCORING - REGRESSION

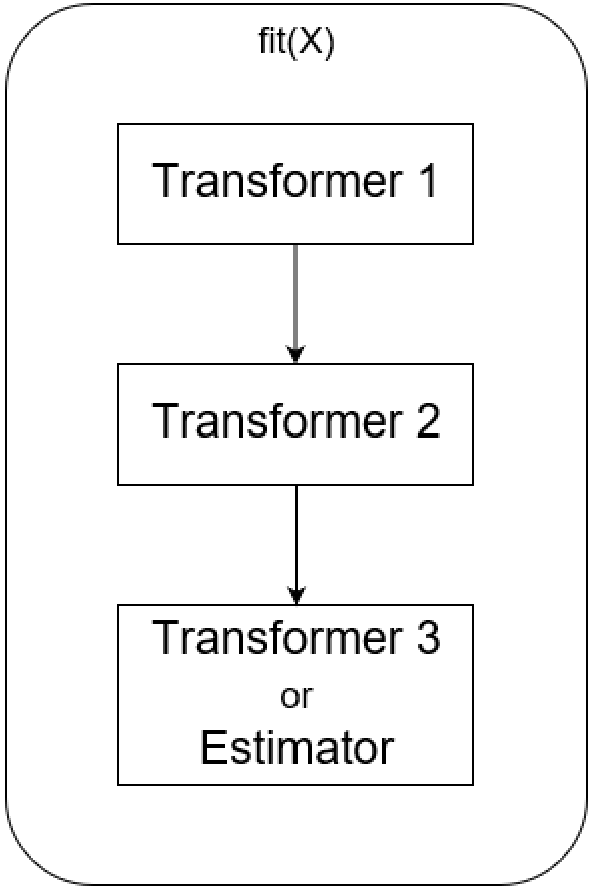
$$\mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix} \qquad \hat{\mathbf{y}} = \begin{pmatrix} \hat{y_1} \\ \hat{y_2} \\ \vdots \\ \hat{y_m} \end{pmatrix}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

Root mean squared error: how much are you missing on average.

PIPELINES

- An easy way to chain together transformers
- Performance is about more than just choosing the right estimator



HYPER PARAMETERS

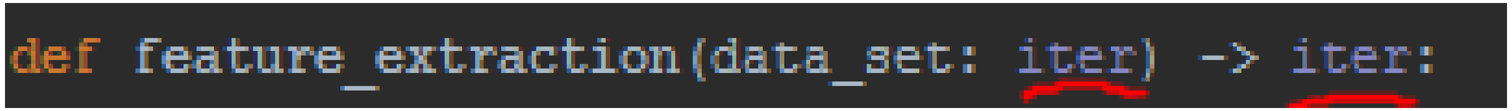
- Parameters that define how the Estimator is trained
- Can make or break your performance
- Scikit-learn provides *reasonable* defaults
- Scikit-learn can help us find the best combination
  - Grid search

Adapted from: https://arxiv.org/pdf/1309.0238.pdf

YOUR TURN

https://github.com/Itera/ml-scikit-intro

Any questions? Just ask!



```
def feature_extraction(data_set: iter) -> iter:
```

itera

MAKE A DIFFERENCE