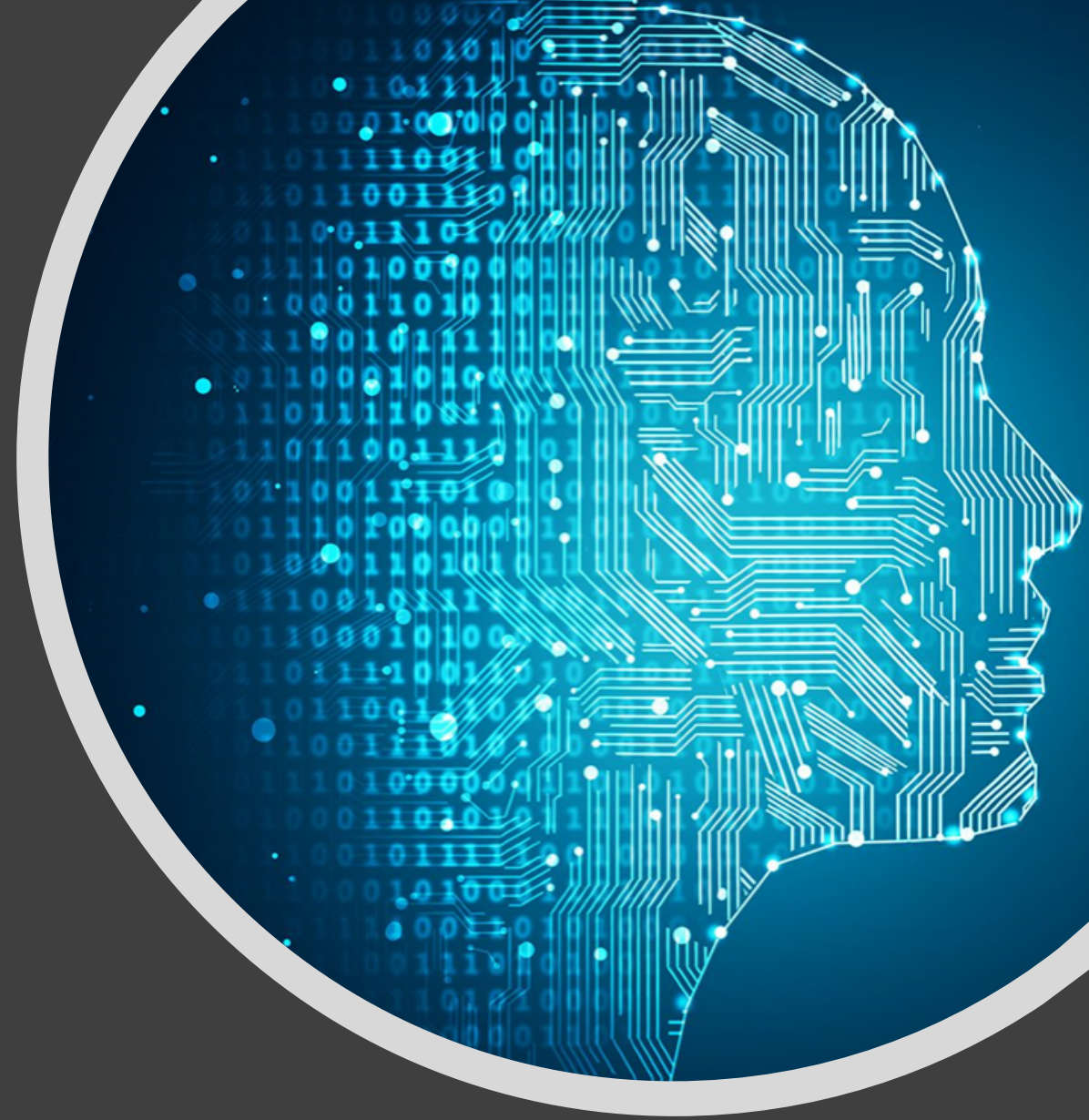


Introduction to Machine Learning

Nicolas Dupuis (d-Wise)

Kevin Lee (Clindata Insight)

ML01 - PhUSE EU Connect 2018

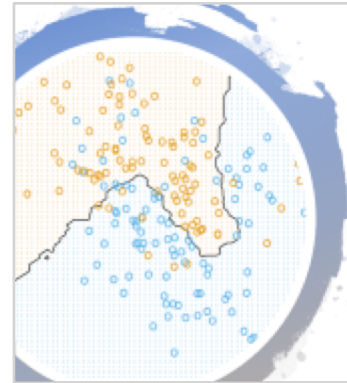


Introduction to Machine Learning

Basic definitions



A simple example: the
K-Nearest Neighbors
classifier



Ecosystems



Concrete
applications



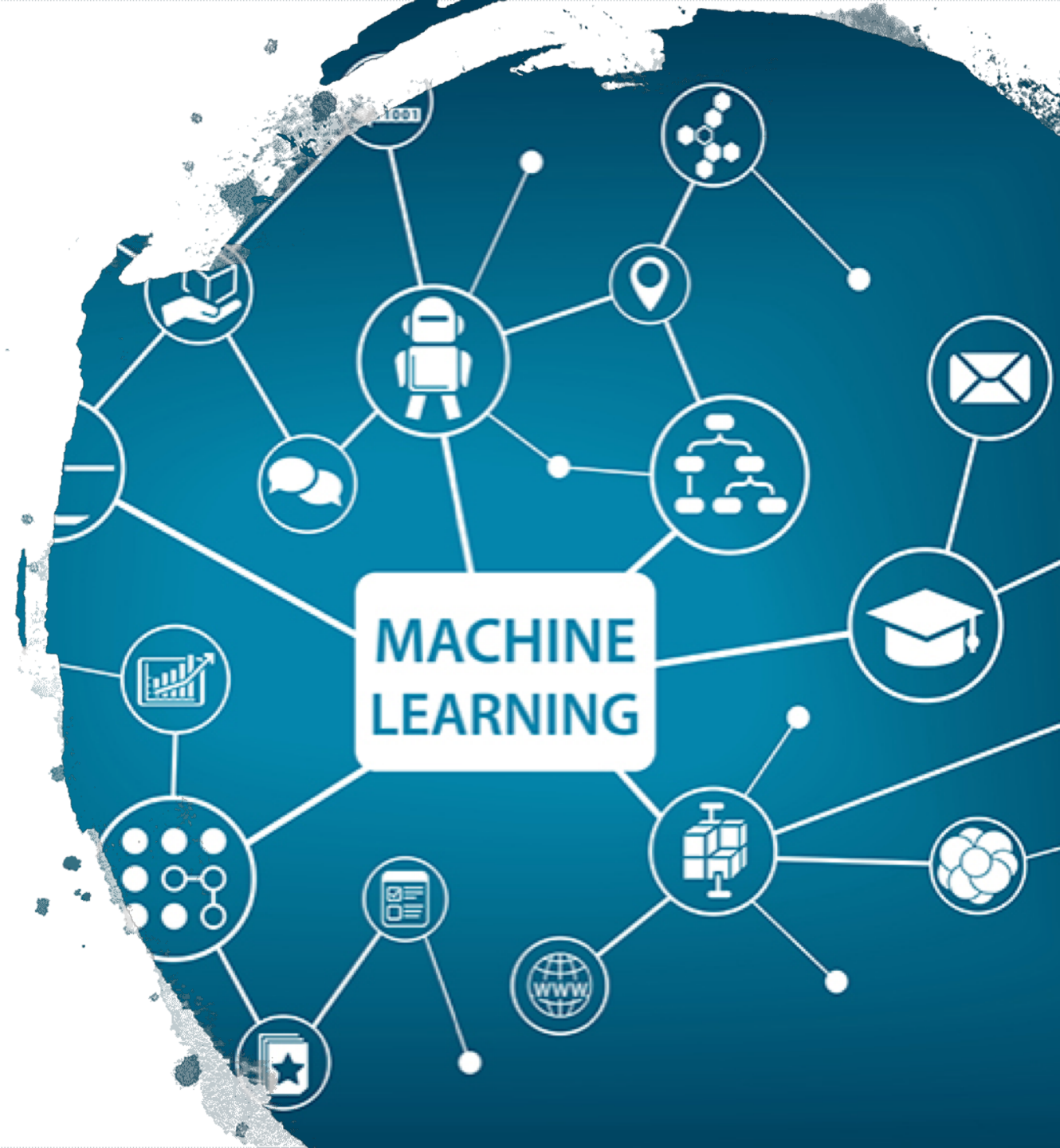
Educating
for
the
future



Q & A

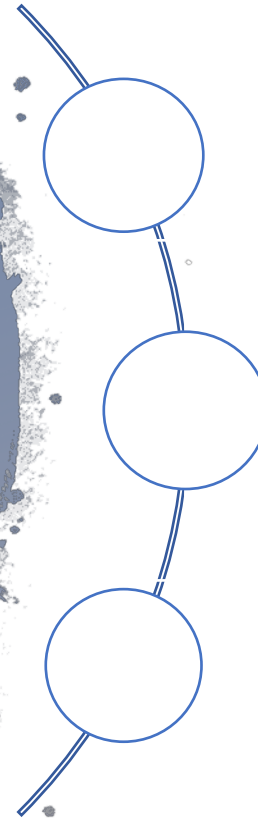


Basic definitions





Machine Learning?



Statistics
learn
explicit

0000000000000000
1111111111111111

ability to
without being

Tools
gain
my c

2222222222222222
3333333333333333
4444444444444444
5555555555555555

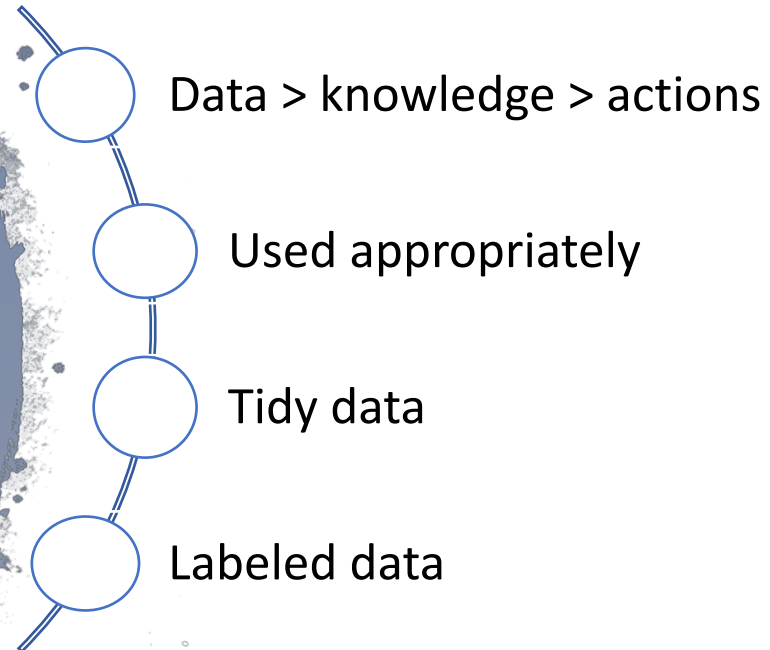
spam?"") or
learn from

Automated
complex
medical

6666666666666666
7777777777777777
8888888888888888
9999999999999999

solve many
option (e.g.

Data is the new oil, if...



4 → 4 2 → 2 3 → 3
4 → 4 9 → 9 0 → 0
5 → 5 7 → 7 1 → 1
9 → 9 0 → 0 3 → 3
6 → 6 7 → 7 4 → 4



Two major types of Machine Learning



	Supervised	Unsupervised
Data	Training data is already labeled . In another words, $Y = f(X)$ where Y and X are known in the training data.	Data is not labeled, the correct answer is not known
Goal	Approximate the mapping function to be able to predict Y for unseen X	Model the underlying structure or distribution to learn more about the data.
Usage	Classification: A classification problem is when the output variable is categorical (<i>e.g.</i> “cat” “dog” “puma”) Regression: A regression problem is when the output variable is a numeric value (<i>e.g.</i> height or cost).	Clustering: A clustering problem is where you want to discover the inherent groupings in the data (<i>e.g.</i> grouping customers by purchasing behavior) Association: An association rule learning problem to discover rules in the data (<i>e.g.</i> people that buy X also tend to buy Y)

Many different ML algorithms



K-Nearest Neighbors



Linear and Logistic Regression



Decision Trees



Random Forests



Support Vector Machine



Artificial Neural Network



Deep Neural Network



... constantly developing field ...



Shortcomings

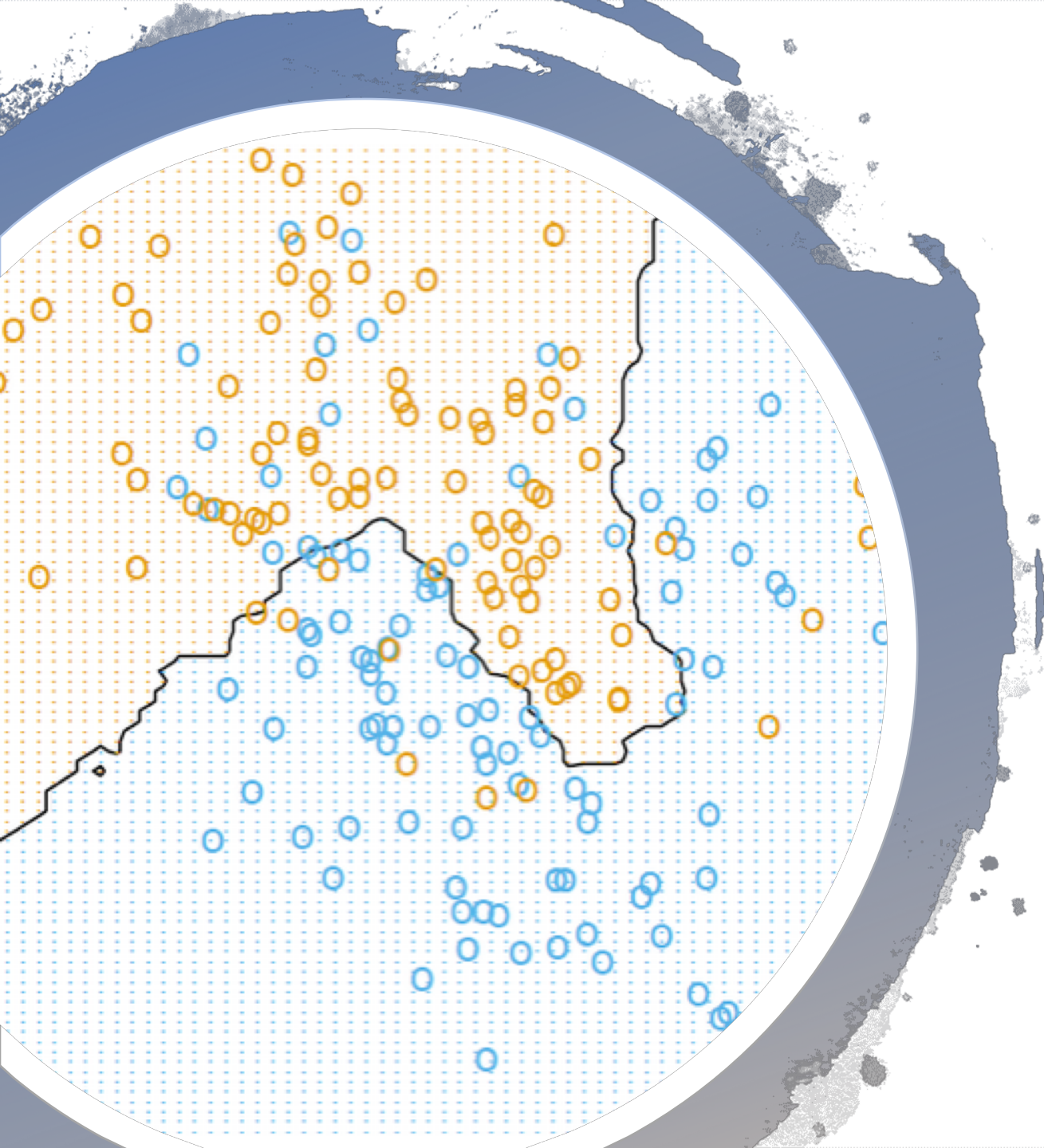


Effective machine learning remains difficult and often fails to deliver the expected value

Lack of (suitable) data, data bias, badly chosen tasks and algorithms, wrong tools and people, garbage in/garbage out

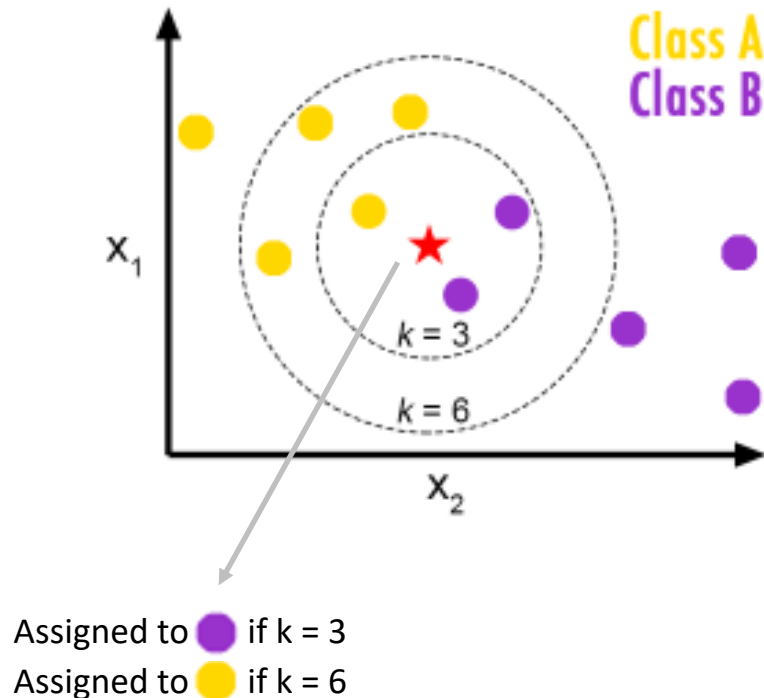
Mistakes happen (*e.g.* gorilla/black people image recognition failure from Google)

Black box: often difficult to understand or explain why a prediction was made



A simple example: the
K-Nearest Neighbors
classifier

The KNN classifier



Robust and versatile classifier that is often used as a benchmark for more complex classifiers (*e.g.* Neural Networks or SVM).

Simple and powerful, it is used in a variety of applications such as economic forecasting, data compression and genetics.

An object is classified by a majority vote of its nearest neighbors.

Before we start, let's discover the Iris dataset

```
In [1]: # Let's import all the libraries we need
import pandas as pd
import numpy as np
from sklearn.cross_validation import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score
%matplotlib inline
```

```
In [2]: # define column names
names = ['sepal_length', 'sepal_width', 'petal_length', 'petal_width', 'class']

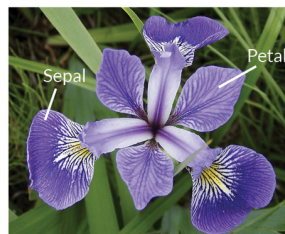
# load training data in a dataframe and print the first 5 records
df = pd.read_csv('iris.data', header=None, names=names)
df.head(5)
```

Out[2]:

	sepal_length	sepal_width	petal_length	petal_width	class
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

Features (input)

Class (output)



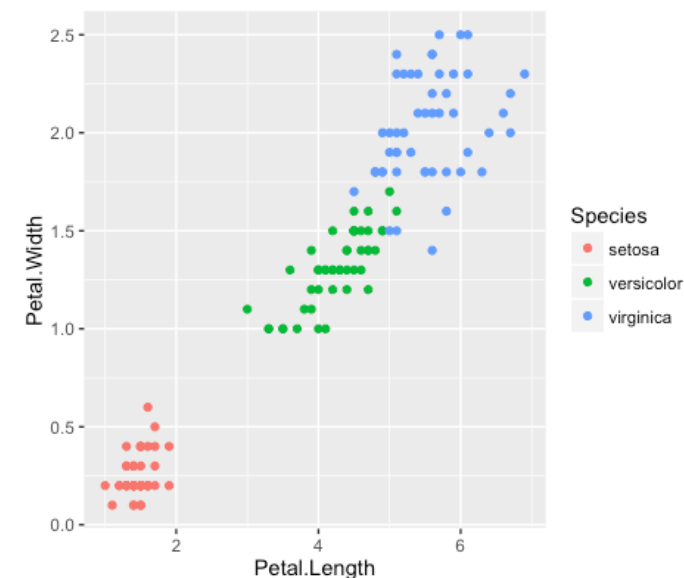
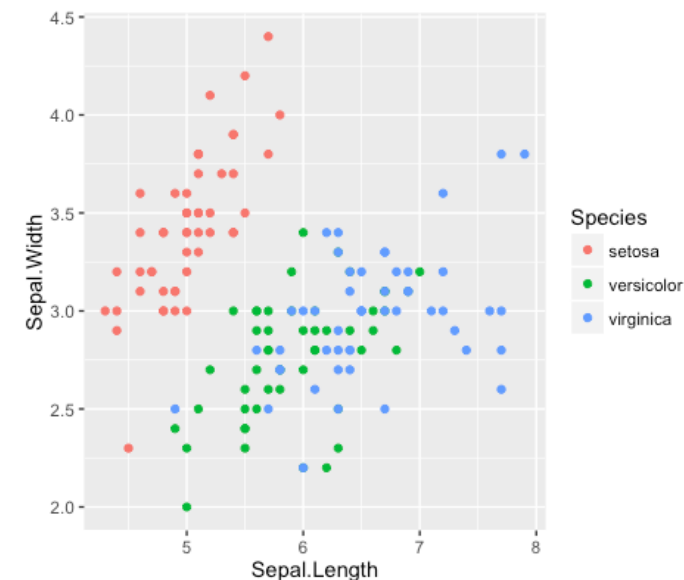
Iris Versicolor



Iris Setosa



Iris Virginica



Classic workflow

1

Choose a classifier (KNN, SVM etc.)

2

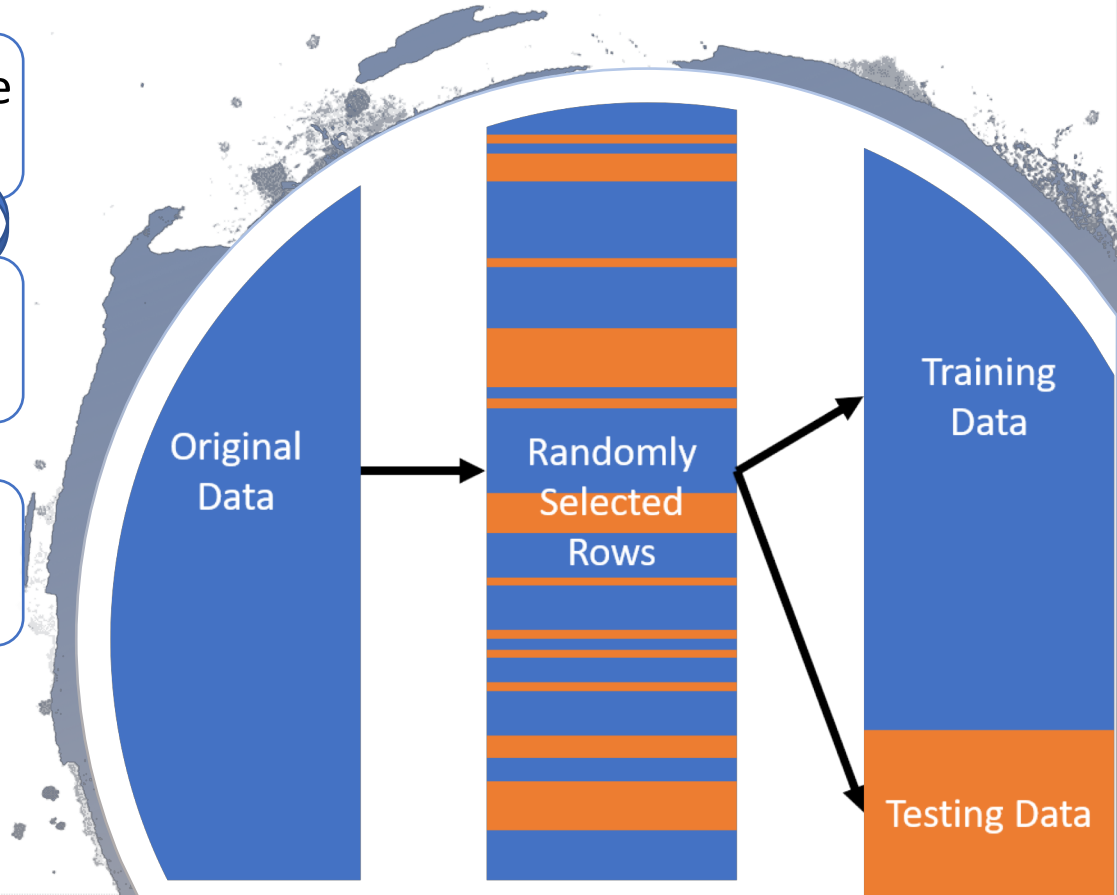
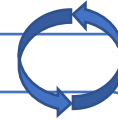
Split the labeled data into training and test. Train the classifier with the training data, tweak hyperparameters

3

Measure predictions accuracy with the test data

4

Make predictions on unseen data



Training the model with different values of

k

```
# split data into training and test sets
X = np.array(df.iloc[:, 0:2])
y = np.array(df['class'])

X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    test_size=0.33,
                                                    random_state=42)
```

```
# Instantiate the classifier into an object
knn = KNeighborsClassifier(n_neighbors = 1)

# Fit the model with training data
knn.fit(X_train, y_train)

# Now the classifier is ready to make predictions on the test data
y_pred = knn.predict(X_test)

# Print accuracy and some predictions
print(str(y_pred[:4]))

['Iris-versicolor' 'Iris-setosa' 'Iris-virginica' 'Iris-versicolor']
```

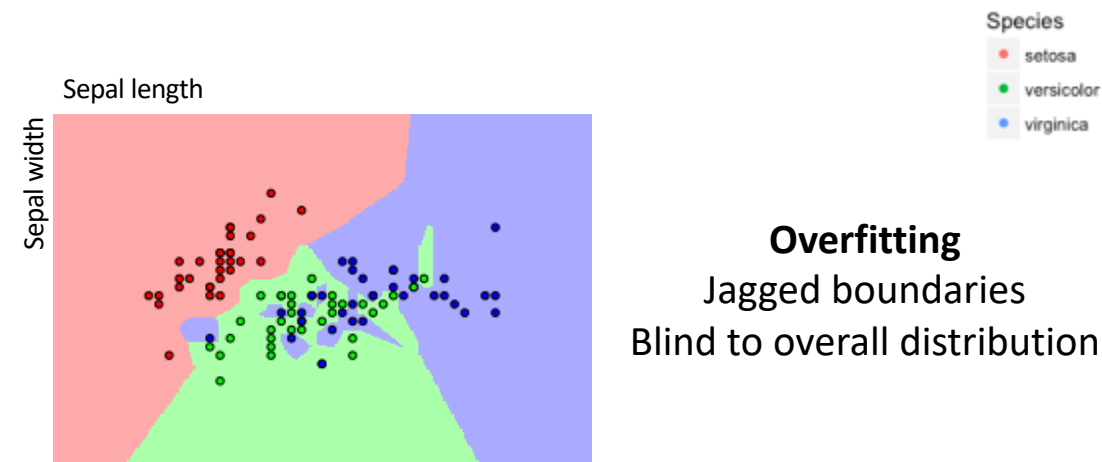
```
# How many predictions were correct?
print(y_pred == y_test)
```

```
[ True  True  True  True False  True  True  True  True  True  True  True
  True  True  True False  True  True  True False  True False  True False
  True False False  True  True  True  True  True False  True  True False
 False  True  True  True False False False  True  True  True  True  True
 False  True]
```

```
accuracy_score(y_pred, y_test)
```

0.72

1



Overfitting
Jagged boundaries
Blind to overall distribution

Training the model with different values of

k

```
# split data into training and test sets
X = np.array(df.iloc[:, 0:2])
y = np.array(df['class'])

X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    test_size=0.33,
                                                    random_state=42)
```

```
# Instantiate the classifier into an object
knn = KNeighborsClassifier(n_neighbors = 82)

# Fit the model with training data
knn.fit(X_train, y_train)

# Now the classifier is ready to make predictions on the test data
y_pred = knn.predict(X_test)

# Print accuracy and some predictions
print(str(y_pred[:4]))

['Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor']
```

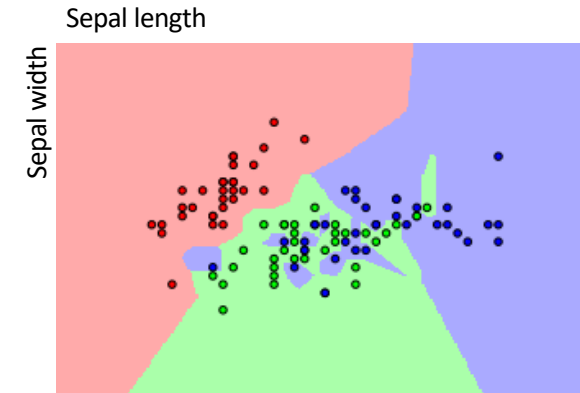
```
# How many predictions were correct?
print(y_pred == y_test)
```

```
[ True False False  True  True  True  True  True  True  True  True False
  True False  True  True  True  True  True False False False False False
  True  True False  True False False  True  True False False False False
 False  True  True  True False  True False  True  True  True  True False
 False  True]
```

```
accuracy_score(y_pred, y_test)
```

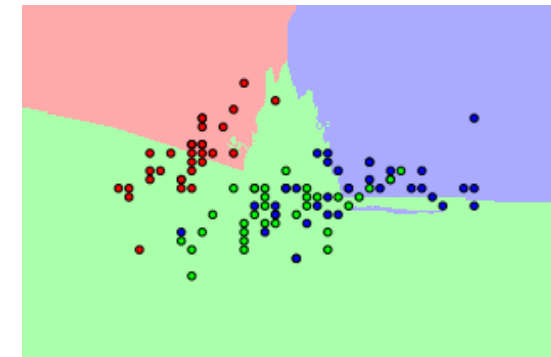
0.56

1



Overfitting
Jagged boundaries
Blind to overall distribution

82



Underfitting
Neither model the training
data nor generalize to new
data

Training the model with different values of

k

```
# split data into training and test sets
```

```
X = np.array(df.iloc[:, 0:2])
```

```
y = np.array(df['class'])
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y,  
                                                    test_size=0.33,  
                                                    random_state=42)
```

```
# Instantiate the classifier into an object
```

```
knn = KNeighborsClassifier(n_neighbors = 24)
```

```
# Fit the model with training data
```

```
knn.fit(X_train, y_train)
```

```
# Now the classifier is ready to make predictions on the test data
```

```
y_pred = knn.predict(X_test)
```

```
# Print accuracy and some predictions
```

```
print(str(y_pred[:4]))
```

```
['Iris-versicolor' 'Iris-setosa' 'Iris-virginica' 'Iris-versicolor']
```

```
# How many predictions were correct?
```

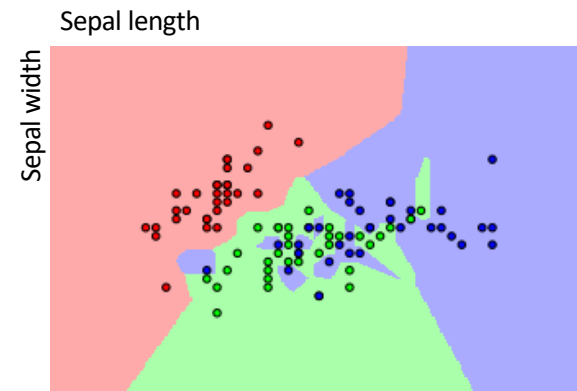
```
print(y_pred == y_test)
```

```
[ True  True  True  True False  True  True  True  True  True  True  True  
  True  True  True False  True  True  True  True  True  True  True  True  
  True  True  True  True  True  True  True  True False  True  True False  
 False  True  True  True False  True False  True  True  True False  True  
 False  True]
```

```
accuracy_score(y_pred, y_test)
```

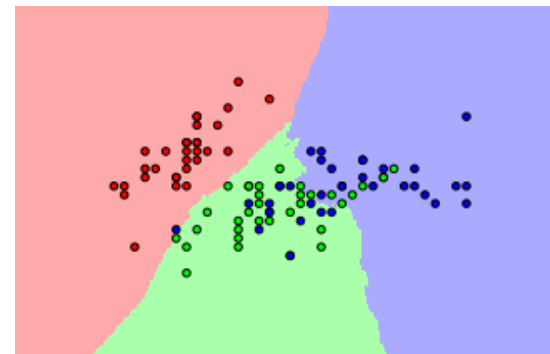
0.82

1



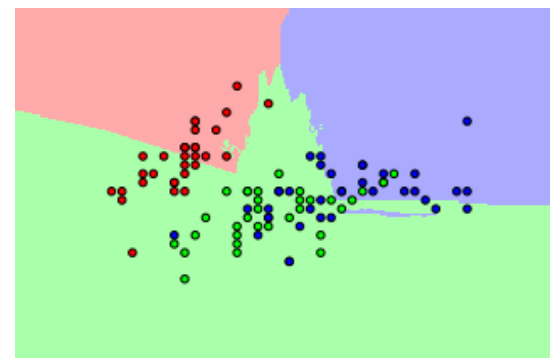
Overfitting
Jagged boundaries
Blind to overall distribution

24



Sweet spot
Smoother boundaries
Resilient to outliers

82

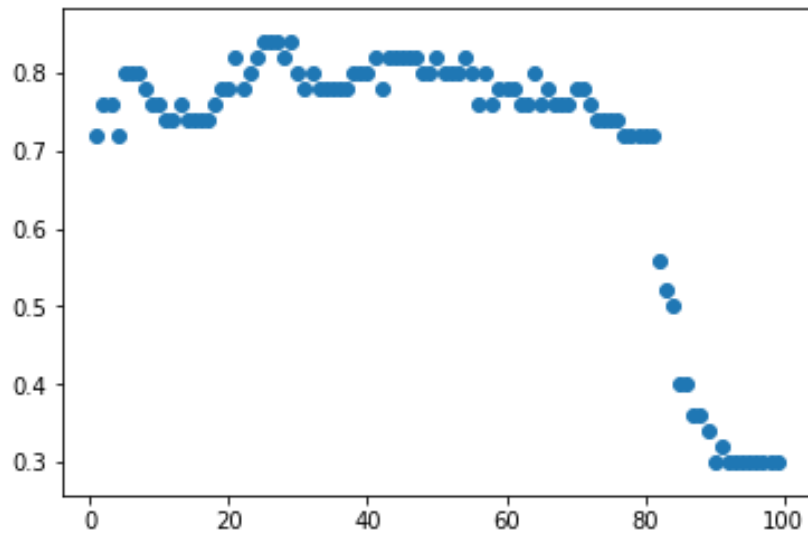


Underfitting
Neither model the training
data nor generalize to new
data

Training the model with different values of k

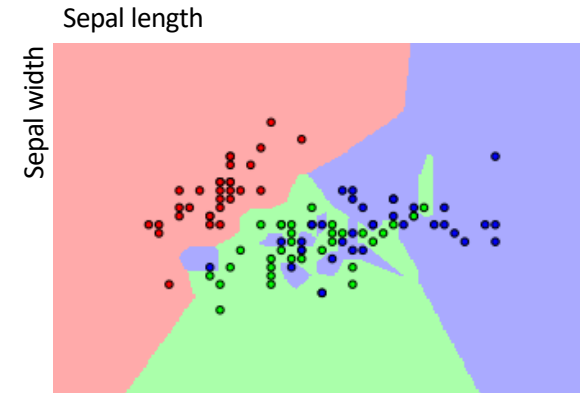
```
# Measure accuracy for k from 1 to 100
all_k = range(1, 100, 1)
scores = []
for k in all_k:
    knn = KNeighborsClassifier(n_neighbors = k)
    knn.fit(X_train, y_train)
    y_pred = knn.predict(X_test)
    scores.append(accuracy_score(y_pred, y_test) )

# Plot accuracy versus k
matplotlib.pyplot.scatter(all_k, scores)
matplotlib.pyplot.show()
```



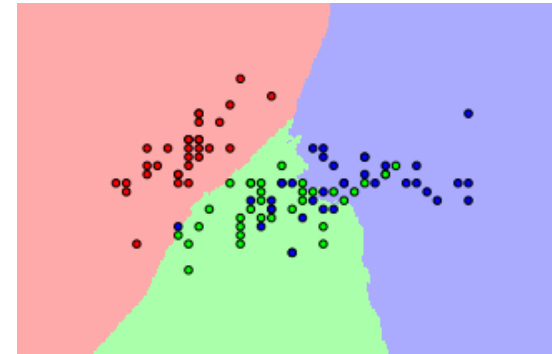
k

1



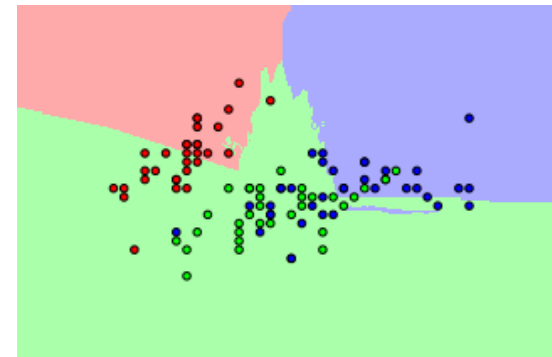
Overfitting
Jagged boundaries
Blind to overall distribution

24



Sweet spot
Smoother boundaries
Resilient to outliers

82



Underfitting
Neither model the training
data nor generalize to new
data

Ecosystems





Python

- Scikit-Learn is the most popular package for ML implementation
- Classification, Regression, SVM, Clustering, Dimensionality Reduction, Decision Trees, Random Forests.



R

- Has some ML packages. Often used for data manipulation and wrapping Python libraries
- Classification and Regression, Neural Network



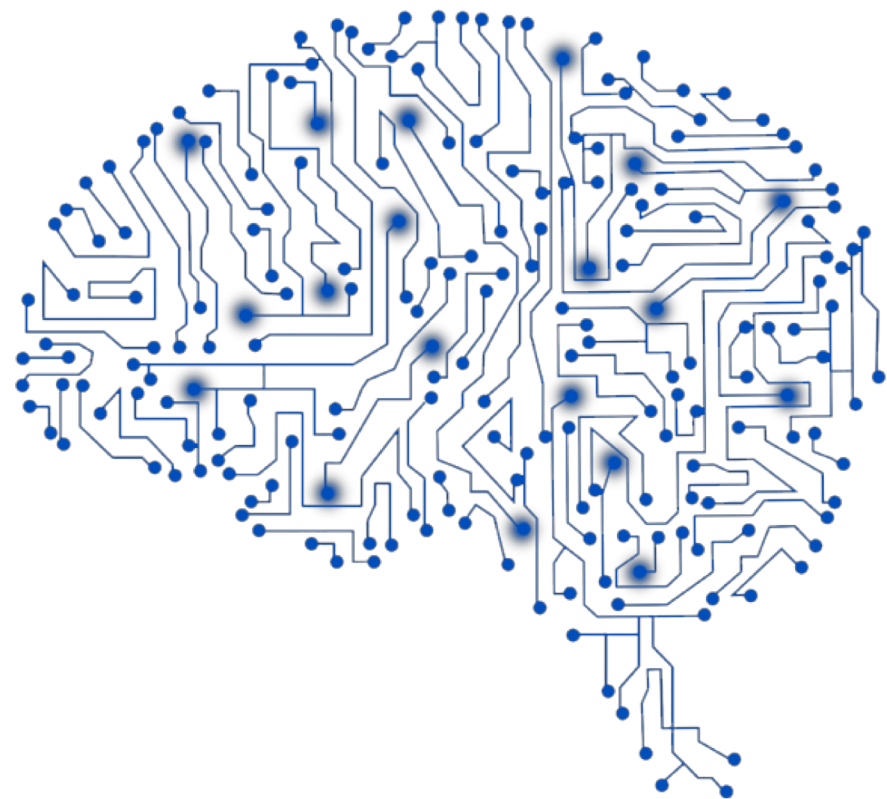
TensorFlow

- Open Source Python framework developed by Google
- ANN, CNN, RNN, Autoencoder, Reinforcement Learning



SAS

- 'SAS Visual Data Mining and Machine Learning'. Limited players, not free
- Linear Regression, Logistic Regression, SVM, DNN



Concrete
applications

Transformative technology in many fields



- Handwriting recognition
- Interface with human (chatbot, Siri, Alexa etc.)
- SPAM detection
- Customers recommendation (Amazon, Netflix, Spotify etc.)
- Sentiment analysis (Natural Language Processing)
- Alpha Go, the first AI to beat Go champions

Adoption in Pharma



- Drug discovery and candidate selection
- Medical image recognition, medical diagnosis
- Optimum site selection and recruitment
- Data anomaly detection
- Personalized medicine
- Operational data
- SDTM, ADaM, TFL?

Adoption in Pharma, cont.



- Slow
- Regulatory restriction
- Black box challenge, need to prove and validate models to explain its outcome
- Big investment in Healthcare and a lot of AI startups coming to Pharma
- Market grows 40% annually, 10 billion USD market in 2024
- Talents shortage, a great opportunities for us

Educating
for
the
future





Educating for the future

(Ian Fleming & James McDermott)

Raise
awareness and
educate our
industry

Gather and
curate
materials

White papers
&
presentations

Machine Learning - Sub teams

Engineering

(Kevin Lee & Nicolas Dupuis)

Thinking

Educational
materials

Ecosystems
Interfaces
and
Platforms

Pharma
Industry
Applications

Projects

White Papers
and PhUSE
presentations



Q & A