Machine Learning Introduction

Aline Kraemer
June 2020

Objective

Introduction to Machine Learning

Hands on: Examples in Python with Scikit-learn

About me

Control and Automation Engineering - UFSC

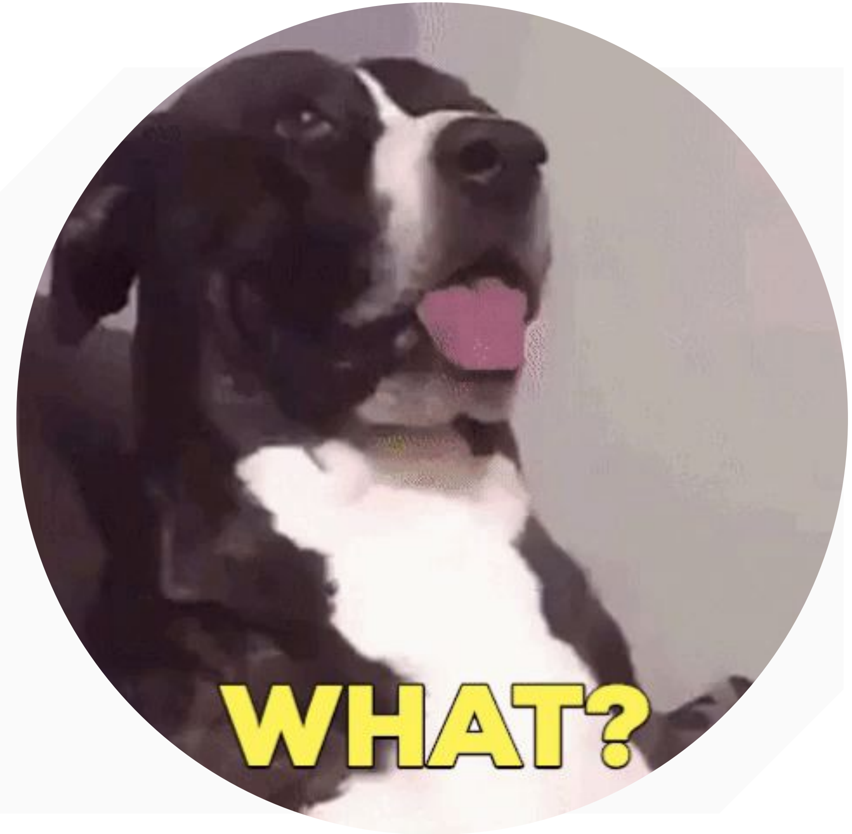
PhD student in Mechanical and Aeronautical Engineering - ITA

AceleraDev Python Women - Codenation

Backend Engineer - Loadsmart

Machine Learning?

"It is the practice of using ***algorithms*** *to* ***parse data***, ***learn from it***, *and then* ***make a determination or prediction*** *about something in the world"* - Nvidia



Machine Learning?

Applications

Speech Recognition

Weather Forecasting

Medical Diagnosis

Traffic Prediction

Image Recognition

Recommender Systems

Stock Market Trading

Autonomous Driving

Fraud Detection

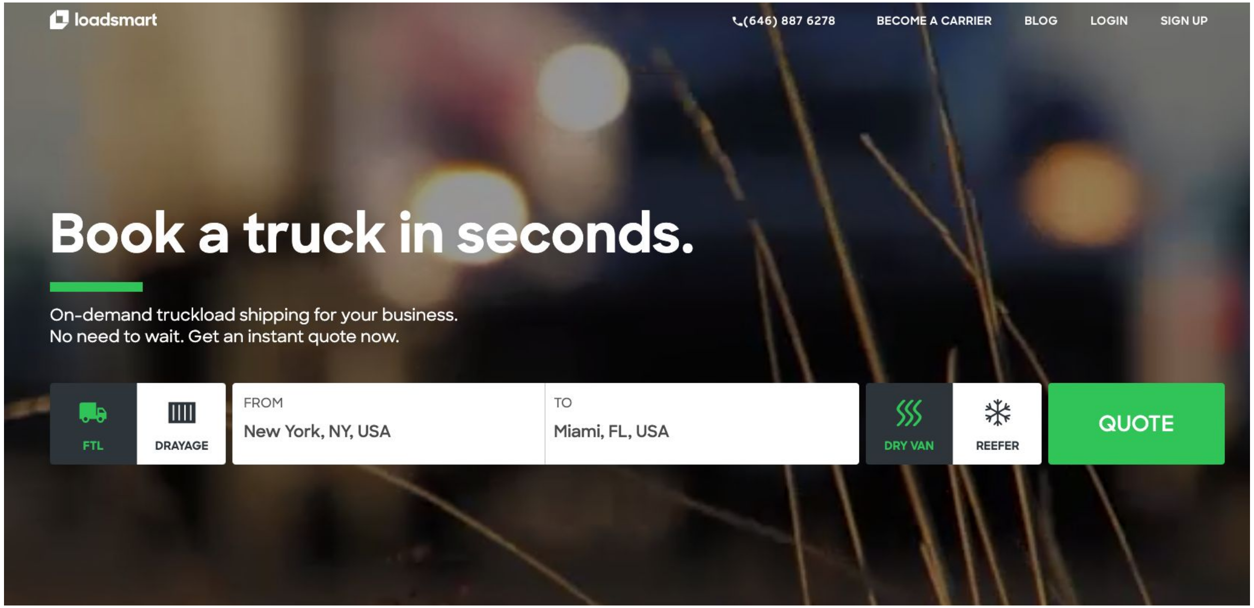
Search Engine

Face Recognition

Spam Detection

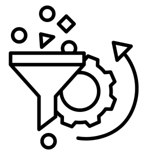
Machine Learning?

Applications



Machine Learning?

Important concepts

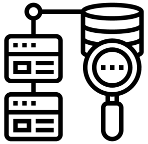


Data preprocessing

Feature Selection/Extraction

Model Selection

Hyperparameters Tuning



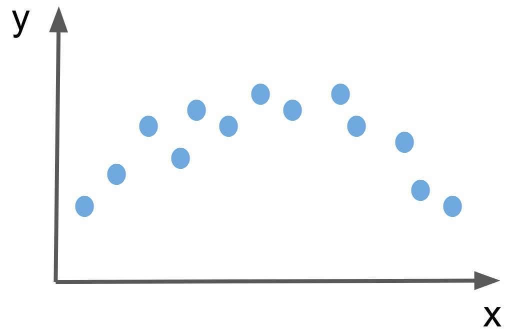
Model Evaluation

Performance Metrics

Cross-validation

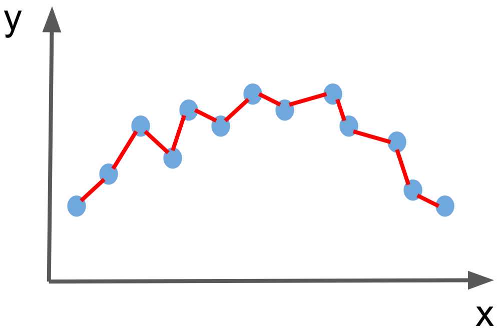
Machine Learning?

Overfitting vs Underfitting

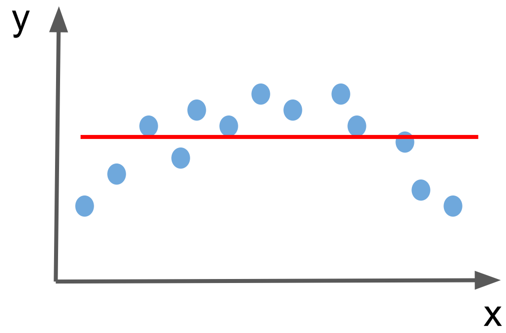


Machine Learning?

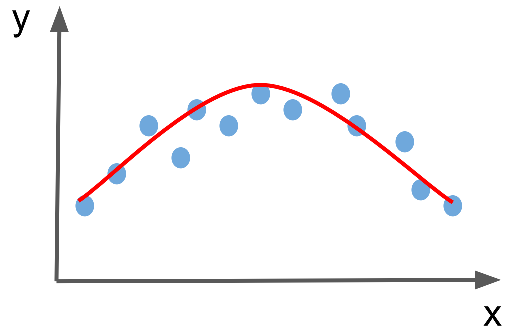
Overfitting vs Underfitting



Overfitting



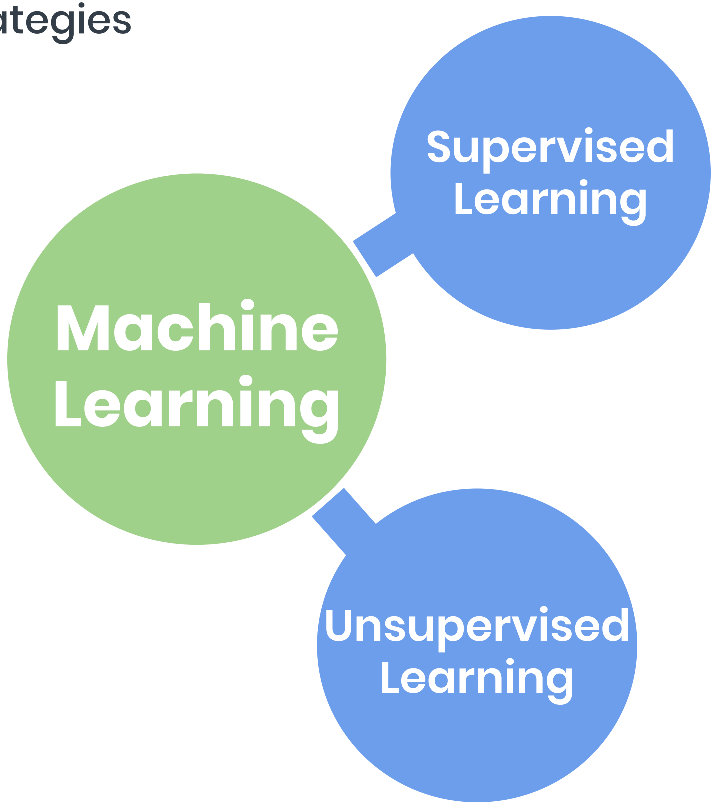
Underfitting



Good balance

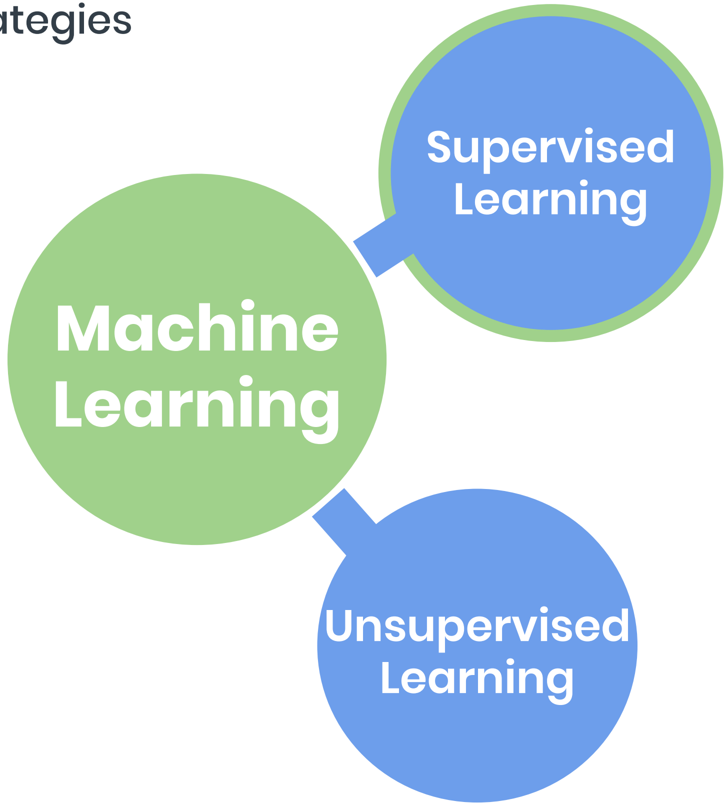
Types of Machine Learning

Learning strategies



Types of Machine Learning

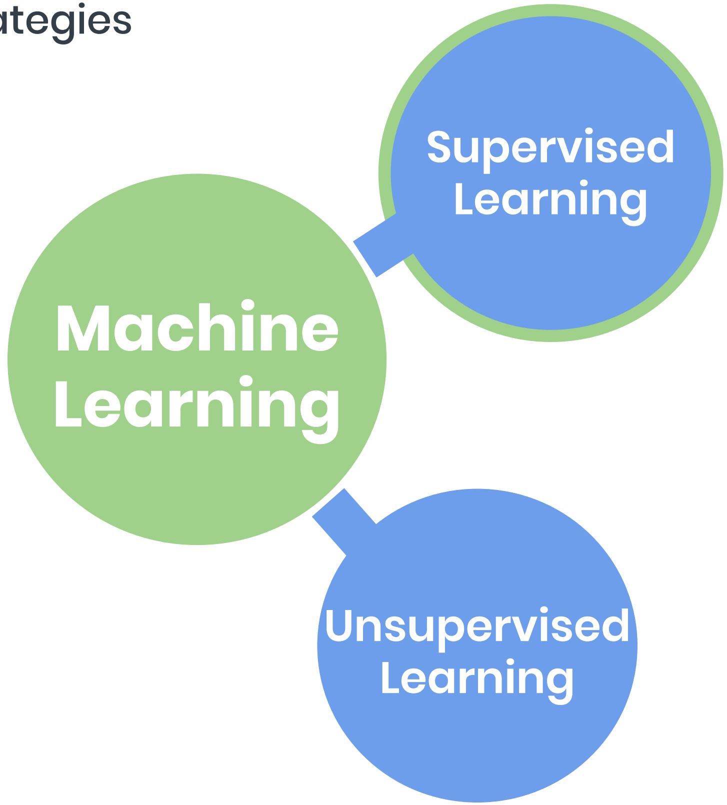
Learning strategies

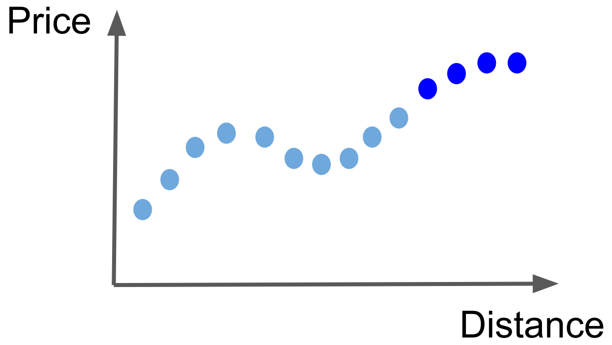


Labeled Data

Types of Machine Learning

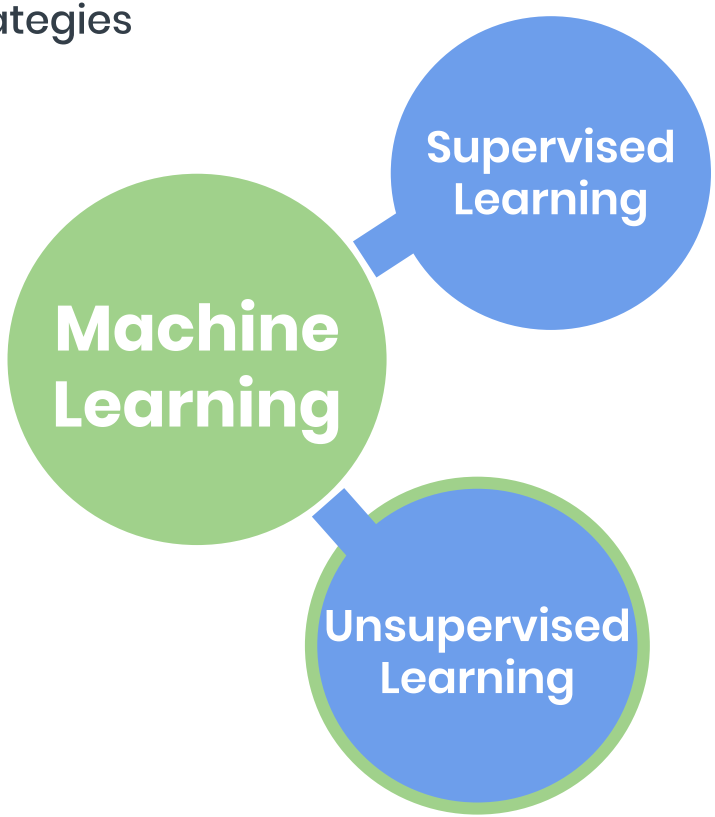
Learning strategies





Types of Machine Learning

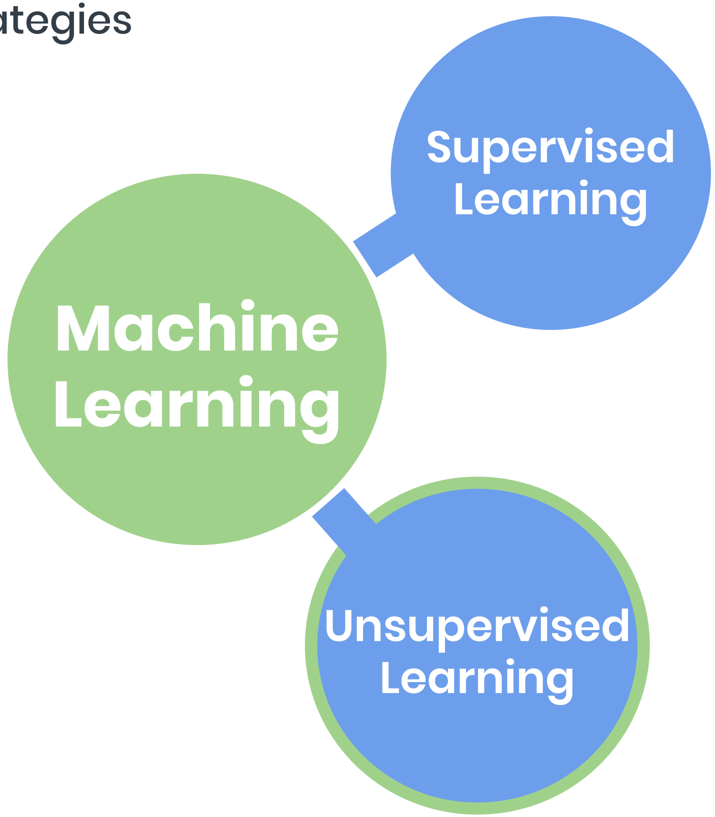
Learning strategies

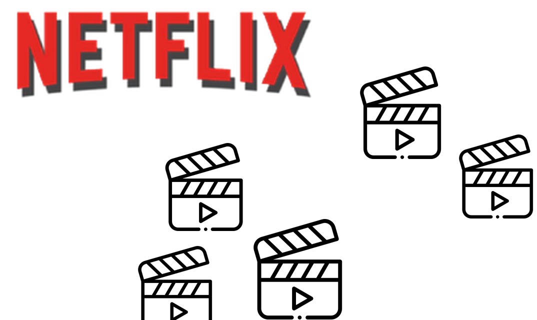


Unlabeled Data

Types of Machine Learning

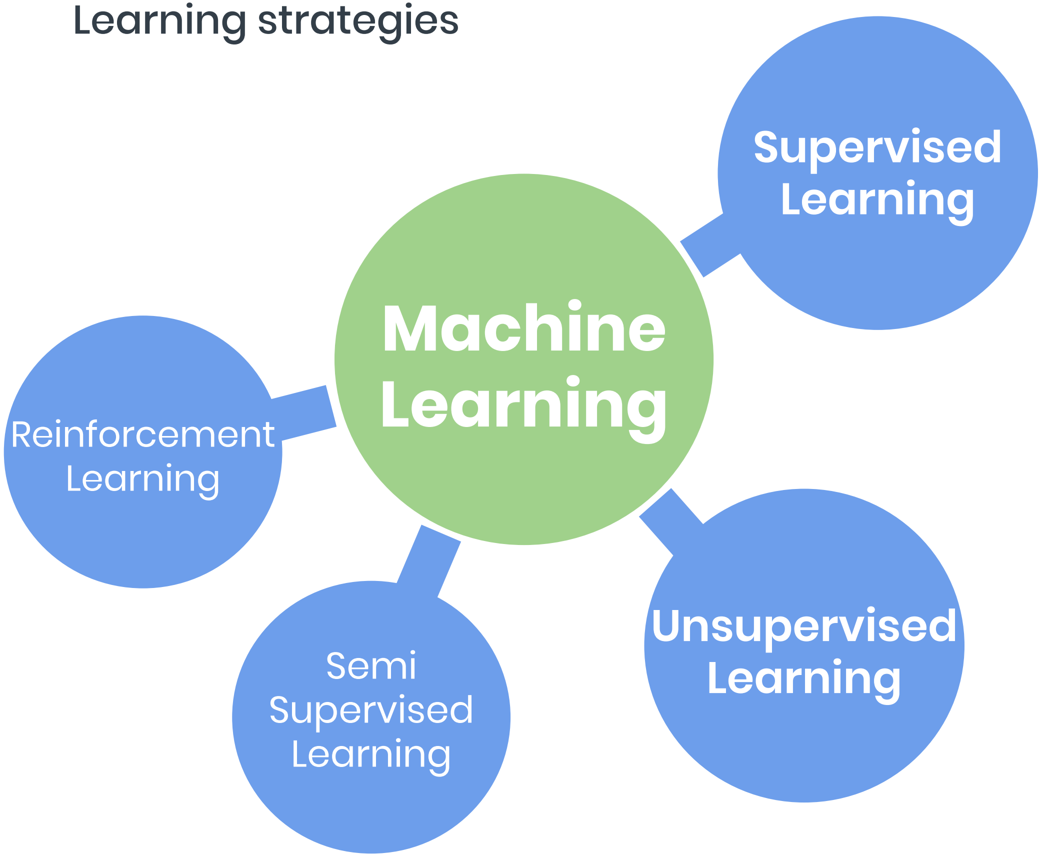
Learning strategies





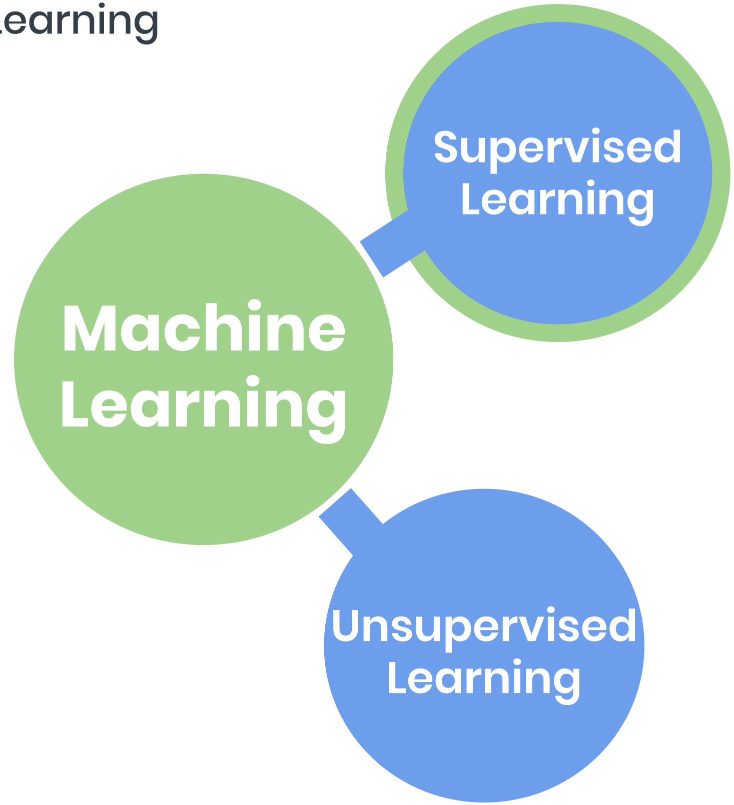
Types of Machine Learning

Learning strategies



Types of Machine Learning

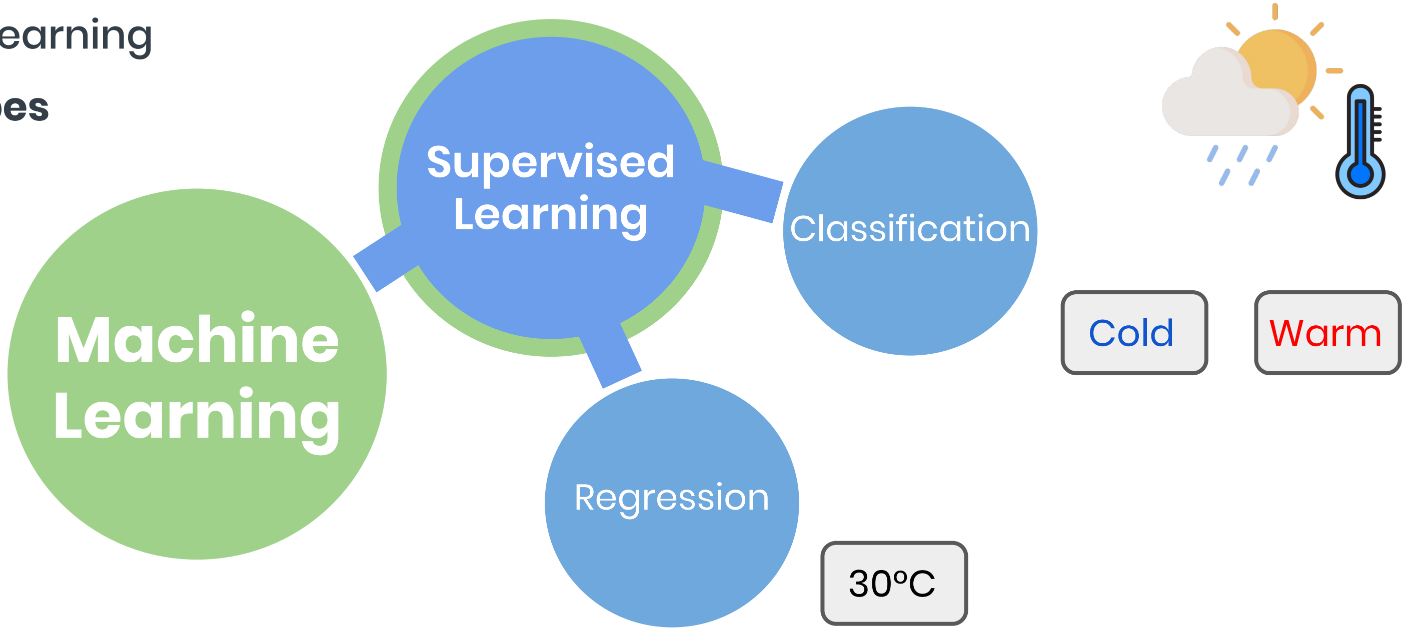
Supervised Learning



Types of Machine Learning

Supervised Learning

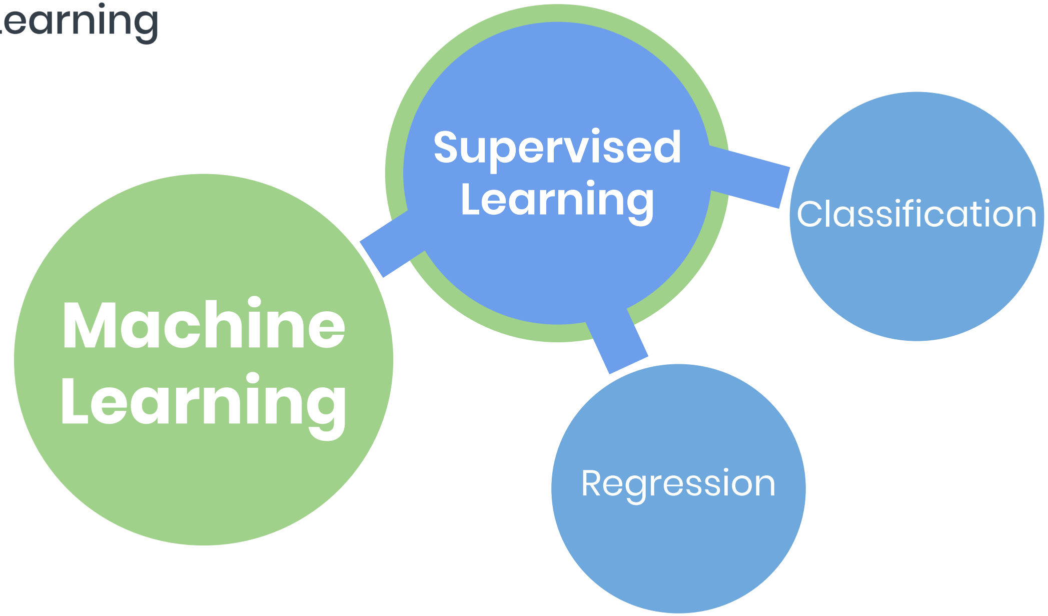
Problem Types



Types of Machine Learning

Supervised Learning

Algorithms



k-Nearest Neighbors

Support Vector Machine

Decision Trees

Random Forests

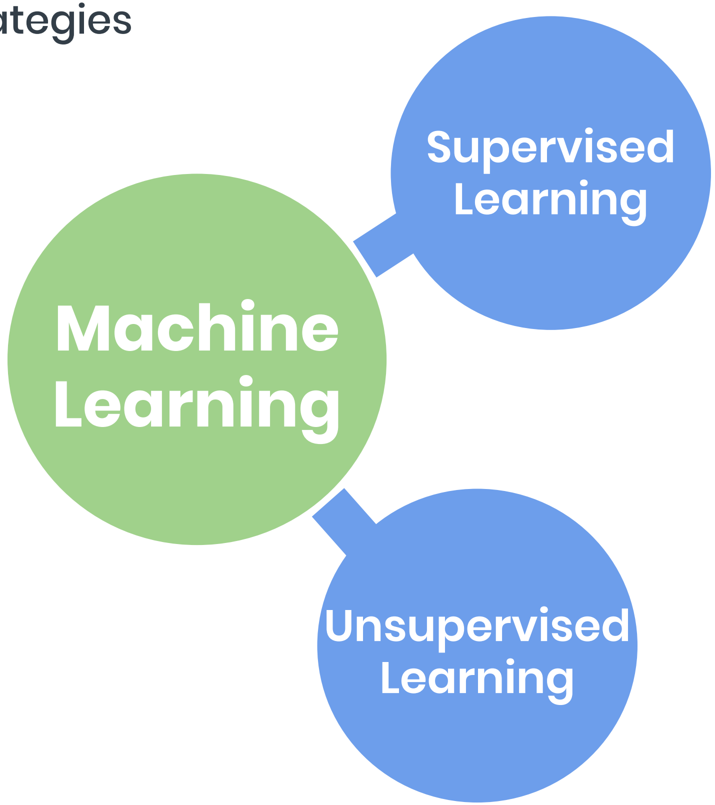
Logistic Regression

Linear Regression

Neural Networks

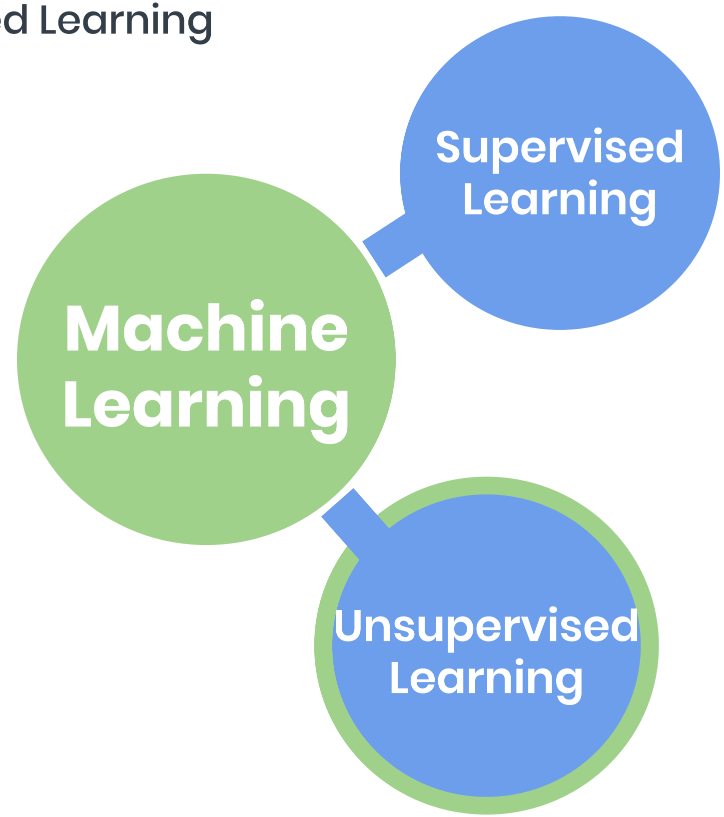
Types of Machine Learning

Learning strategies



Types of Machine Learning

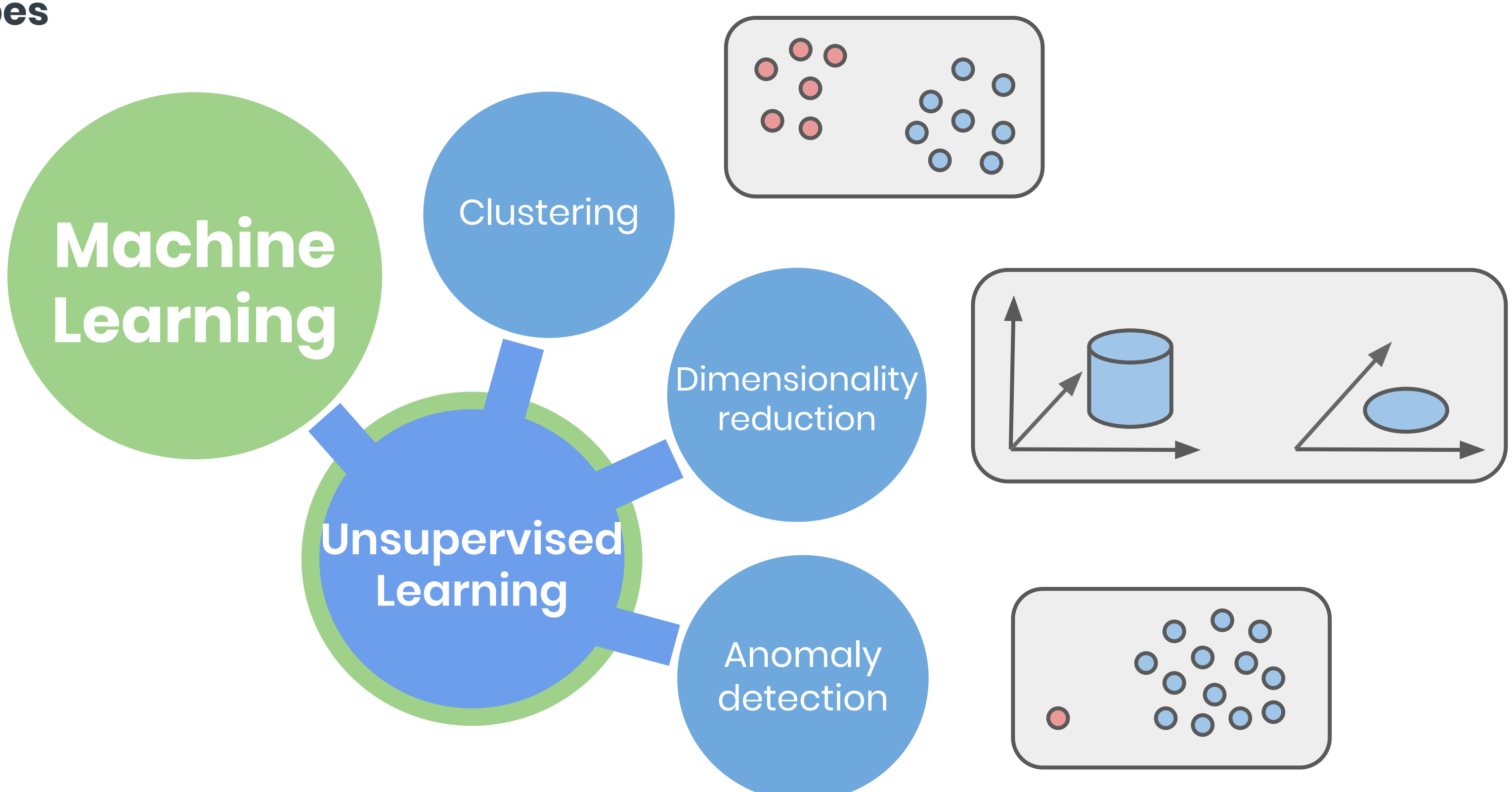
Unsupervised Learning



Types of Machine Learning

Unsupervised Learning

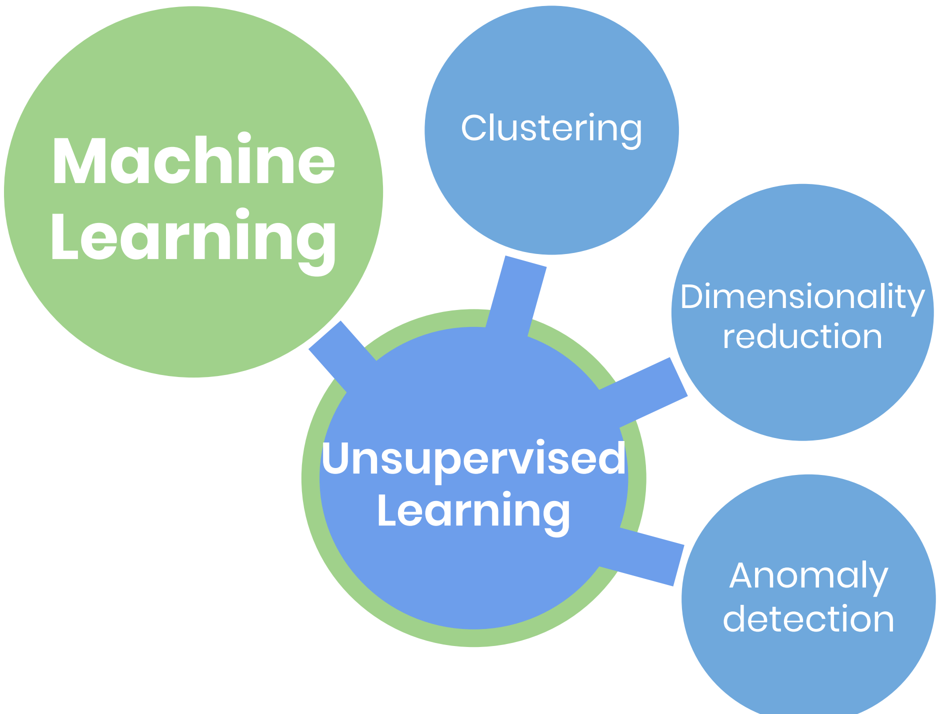
Problem Types



Types of Machine Learning

Unsupervised Learning

Algorithms



K-Means

K-Medoids

DBSCAN

Hierarchical Cluster Analysis

Principal Component Analysis

One-class SVM

Isolation Forest

Hands On

Scikit-learn Library

Scikit-learn Library

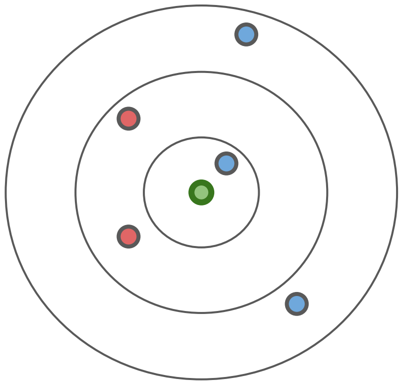
Open source machine learning library in Python

Supports supervised and unsupervised learning

Provides tools for:
- Model fitting
- Data preprocessing
- Model selection
- Model evaluation

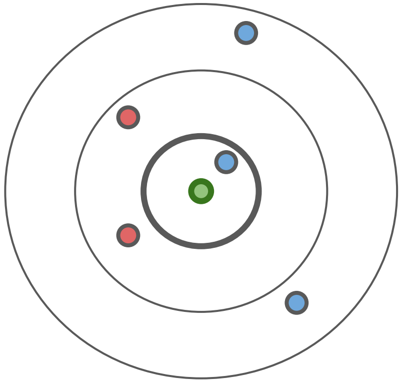
Supervised Learning Example

k-Nearest Neighbors Classification



Supervised Learning Example

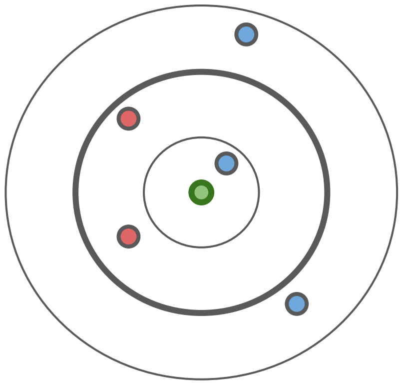
k-Nearest Neighbors Classification



k = 1 → Classified as

Supervised Learning Example

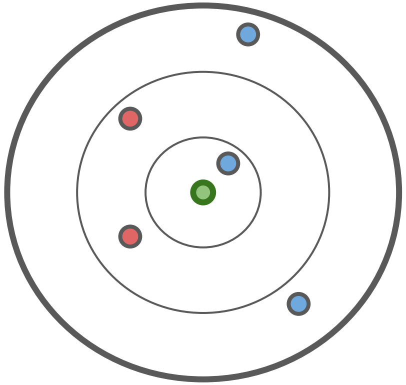
k-Nearest Neighbors Classification



k = 3 → Classified as

Supervised Learning Example

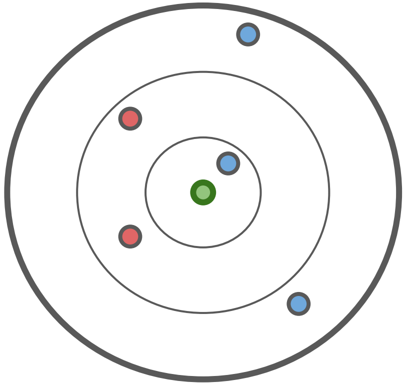
k-Nearest Neighbors Classification



k = 5 → Classified as 

Supervised Learning Example

k-Nearest Neighbors Classification



k = 1 → Classified as

k = 3 → Classified as

k = 5 → Classified as

Supervised Learning Example

k-Nearest Neighbors Classification
Iris Dataset

[1]

```python
from sklearn import datasets
iris = datasets.load_iris()
```

[2]

```python
iris.data[:5]
```

```
array([[5.1, 3.5, 1.4, 0.2],
       [4.9, 3. , 1.4, 0.2],
       [4.7, 3.2, 1.3, 0.2],
       [4.6, 3.1, 1.5, 0.2],
       [5. , 3.6, 1.4, 0.2]])
```

[3]

```python
iris.feature_names
```

```
['sepal length (cm)',
 'sepal width (cm)',
 'petal length (cm)',
 'petal width (cm)']
```

Supervised Learning Example

k-Nearest Neighbors Classification
Iris Dataset

[4]

```
iris.target[:5]
```

```
array([0, 0, 0, 0, 0])
```

[5]

```
iris.target_names
```

```
array(['setosa', 'versicolor', 'virginica'], dtype='<U10')
```

[6]

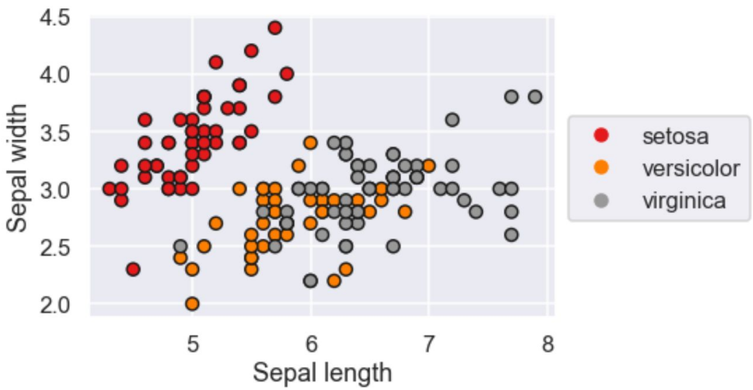
```
import matplotlib.pyplot as plt
import seaborn as sns

sns.set(context='talk')
```

Supervised Learning Example

k-Nearest Neighbors Classification
Iris Dataset

[7]

```
scatter = plt.scatter(x=iris.data[:,0], y=iris.data[:,1], c=iris.target,
    cmap=plt.cm.Set1, edgecolor='k')

plt.xlabel('Sepal length')
plt.ylabel('Sepal width')
plt.legend(handles=scatter.legend_elements()[0], labels=list(iris.target_names),
    bbox_to_anchor=(1, 0.7))
```

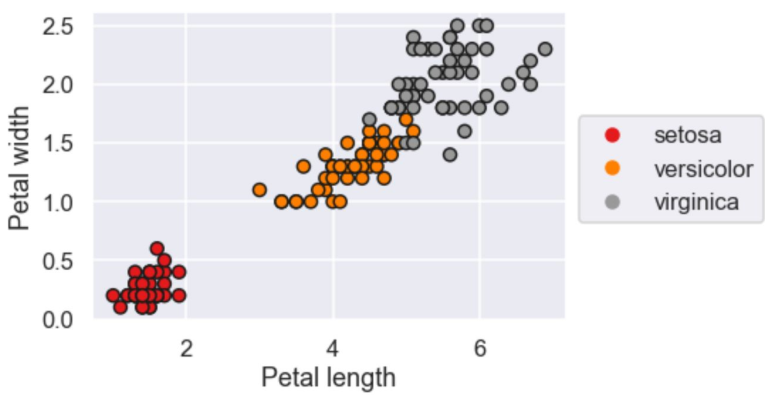


Supervised Learning Example

k-Nearest Neighbors Classification
Iris Dataset

[8]

```
scatter = plt.scatter(x=iris.data[:,2], y=iris.data[:,3], c=iris.target,
    cmap=plt.cm.Set1, edgecolor='k')

plt.xlabel('Petal length')
plt.ylabel('Petal width')
plt.legend(handles=scatter.legend_elements()[0], labels=list(iris.target_names),
    bbox_to_anchor=(1, 0.7))
```



Supervised Learning Example

k-Nearest Neighbors Classification
Iris Dataset

[9]

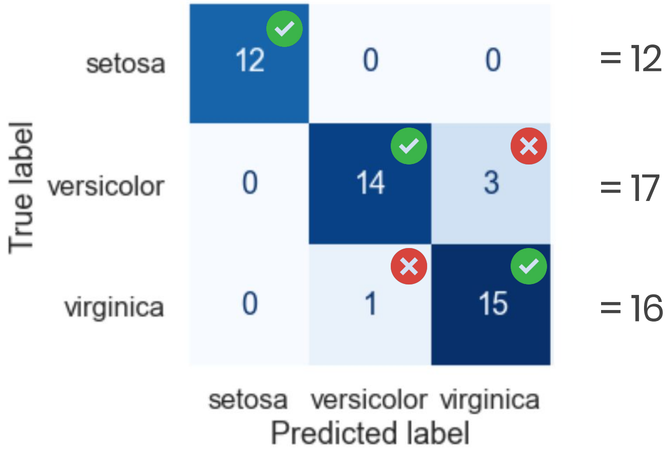
```
from sklearn.model_selection import train_test_split
from sklearn import neighbors

X = iris.data
y = iris.target
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.3)

n_neighbors = 5
clf = neighbors.KNeighborsClassifier(n_neighbors)
clf.fit(X_train, y_train)
```

Supervised Learning Example

k-Nearest Neighbors Classification
Iris Dataset

[10]

```
from sklearn.metrics import plot_confusion_matrix

plot_confusion_matrix(clf, X_test, y_test,
    display_labels=iris.target_names, cmap=plt.cm.Blues)
plt.grid(False)
```



Supervised Learning Example

k-Nearest Neighbors Classification
Iris Dataset

[11]

```
# Using only 2 features
X = iris.data[:,:2]
y = iris.target

n_neighbors = 5
clf = neighbors.KNeighborsClassifier(n_neighbors)
clf.fit(X, y)
```

[12]

```
from matplotlib.colors import ListedColormap

# Create color maps
cmap_light = ListedColormap(['#ed5e5f', '#ffa54d', '#bfbfbf'])
cmap_bold = ListedColormap(['#e41a1c', '#ff7f00', '#999999'])
```

Supervised Learning Example

k-Nearest Neighbors Classification
Iris Dataset

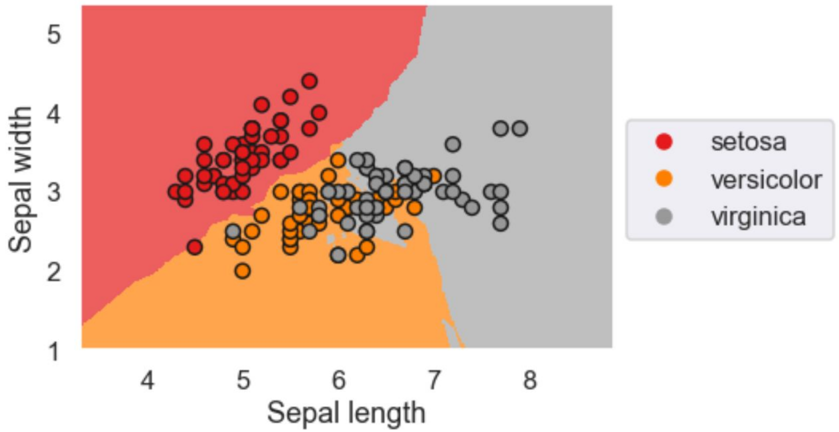
[13]

```
import numpy as np

# Create a mesh
h = 0.02  # step size
x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, h),
    np.arange(y_min, y_max, h))

Z = clf.predict(np.c_[xx.ravel(), yy.ravel()])

# Plot the decision boundary
Z = Z.reshape(xx.shape)
plt.figure()
plt.pcolormesh(xx, yy, Z, cmap=cmap_light)
```

Supervised Learning Example

k-Nearest Neighbors Classification
Iris Dataset

```
# Plot the training points
plt.scatter(X[:, 0], X[:, 1], c=y, cmap=cmap_bold, edgecolor='k')
plt.xlim(xx.min(), xx.max())
plt.ylim(yy.min(), yy.max())

plt.xlabel('Sepal length')
plt.ylabel('Sepal width')
plt.legend(handles=scatter.legend_elements()[0], labels=list(iris.target_names),
    bbox_to_anchor=(1, 0.7))
```



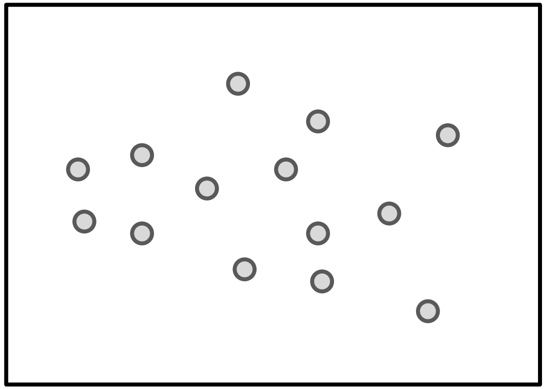
Unsupervised Learning Example

k-Means Clustering

Unsupervised Learning Example

k-Means Clustering

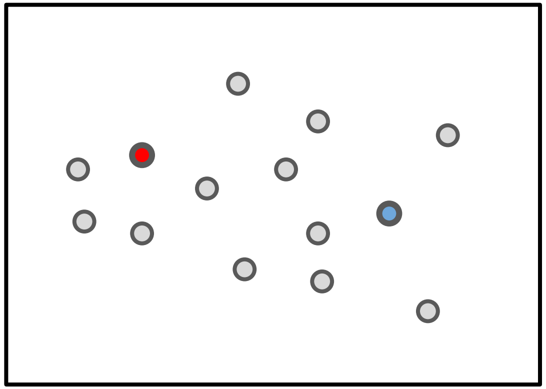
K = 2 (2 clusters)



Unsupervised Learning Example

K-Means Clustering

K = 2 (2 clusters)

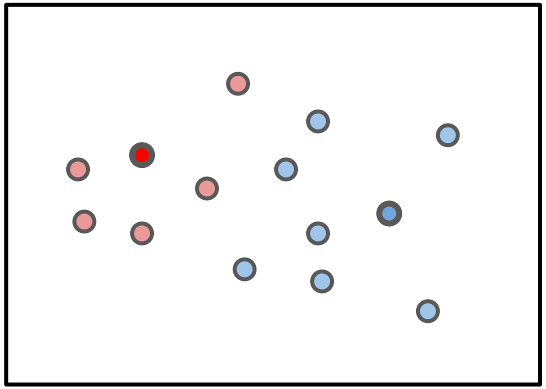


1. Randomly pick K centroids from the sample points as initial cluster centers

Unsupervised Learning Example

K-Means Clustering

K = 2 (2 clusters)

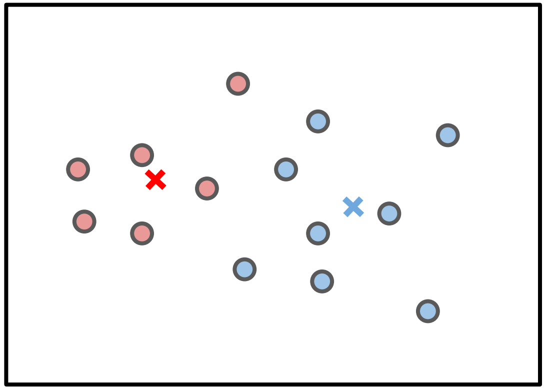


2. Assign each sample to the nearest centroid

Unsupervised Learning Example

K-Means Clustering

K = 2 (2 clusters)

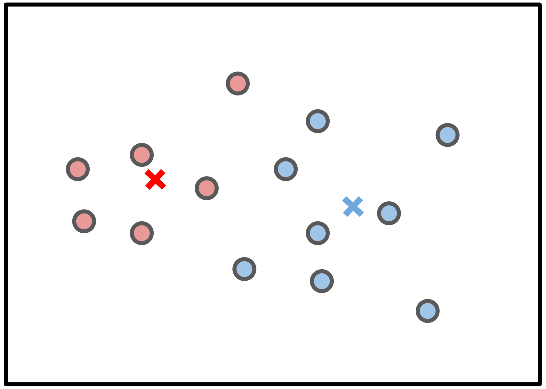


3. Move the centroids to the center of the samples that were assigned to it

Unsupervised Learning Example

K-Means Clustering

K = 2 (2 clusters)



4. Repeat steps 2 and 3 until the cluster assignments do not change or a user-defined tolerance or maximum number of iterations is reached

Unsupervised Learning Example

K-Means Clustering

[1]

```python
from sklearn.datasets import make_blobs

# Create dataset
X, y = make_blobs(n_samples=150, n_features=2, centers=3,
    cluster_std=0.5, shuffle=True, random_state=0)
```

[2]

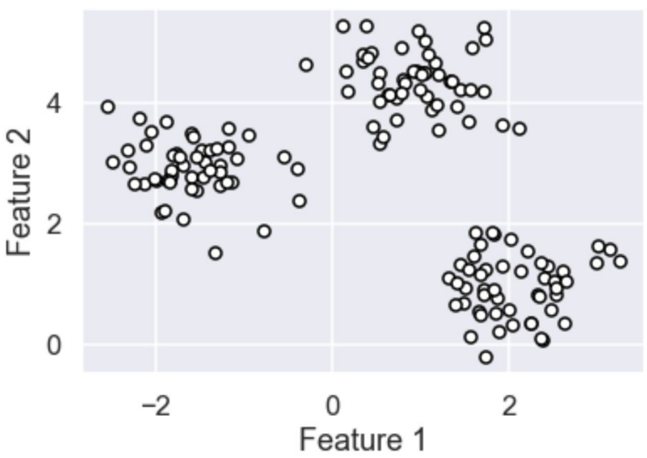
```python
import matplotlib.pyplot as plt
import seaborn as sns

sns.set(context='talk')
```

Unsupervised Learning Example

K-Means Clustering

[3]

```python
# Plot
plt.scatter(X[:, 0], X[:, 1], c='white', marker='o',
    edgecolor='black', s=50)

plt.xlabel('Feature 1')
plt.ylabel('Feature 2')
```



Unsupervised Learning Example

K-Means Clustering

[4]

```
from sklearn.cluster import KMeans

km = KMeans(n_clusters=3, init='random', n_init=10,
    max_iter=300, tol=1e-04, random_state=0)

y_km = km.fit_predict(X)
```

Unsupervised Learning Example

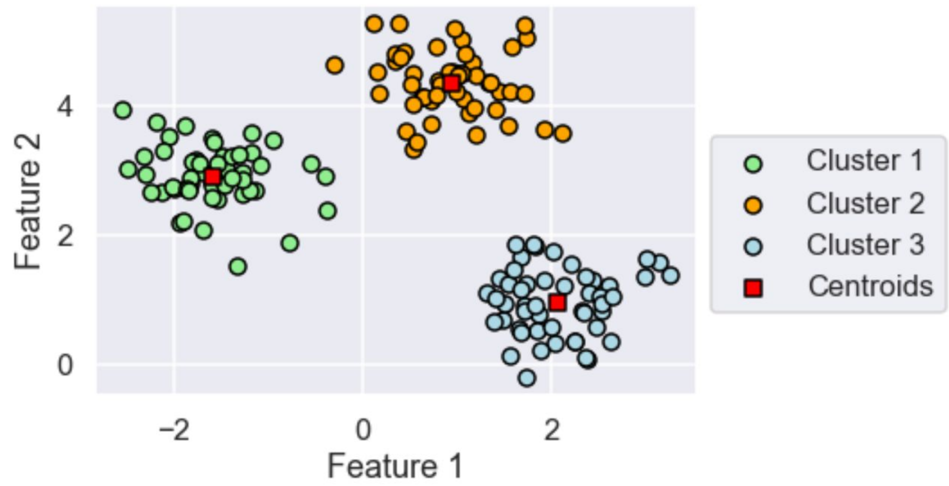
K-Means Clustering

[5]

```
# Plot the 3 clusters
plt.scatter(X[y_km == 0, 0], X[y_km == 0, 1],
    c='lightgreen', edgecolor='black', label='Cluster 1')

plt.scatter(X[y_km == 1, 0], X[y_km == 1, 1],
    c='orange', edgecolor='black', label='Cluster 2')

plt.scatter(X[y_km == 2, 0], X[y_km == 2, 1],
    c='lightblue', edgecolor='black', label='Cluster 3')

# Plot the centroids
plt.scatter(km.cluster_centers_[:, 0], km.cluster_centers_[:, 1],
    marker='s', c='red', edgecolor='black', label='Centroids')

plt.legend(bbox_to_anchor=(1, 0.7))
plt.xlabel('Feature 1')
plt.ylabel('Feature 2')
```

Unsupervised Learning Example

K-Means Clustering

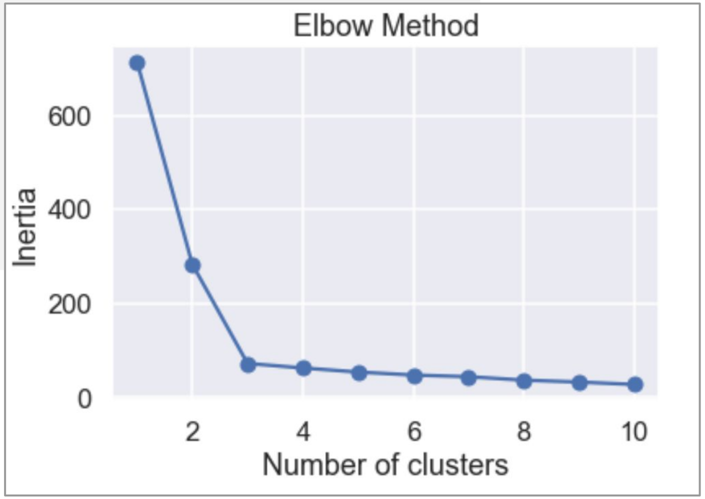


Unsupervised Learning Example

K-Means Clustering

[6]

```python
# Given a range of number of cluster, calculate the sum of
# squared distances of samples to their closest centroid.
inertia = []
for i in range(1, 11):
    km = KMeans(n_clusters=i, init='random', n_init=10, max_iter=300,
        tol=1e-04, random_state=0)
    km.fit(X)
    inertia.append(km.inertia_)

# Plot
plt.plot(range(1, 11), inertia, marker='o')
plt.xlabel('Number of clusters')
plt.ylabel('Inertia')
plt.title('Elbow Method')
```



Final Remarks

A machine learning **model**
is only **as good as** the **data** it is fed.

Thank you!

loadsmart

We are hiring!