Course 2: Supervised Learning

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

2024-09-30

Course 2: Supervised Learning

Course 2: Supervised Learning

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

Summary

Last session

1. AI definition
2. Applications & Open Issues
3. Deep learning
4. Foundation models

Today's session

- Learning from labeled examples
- Challenges of supervised learning

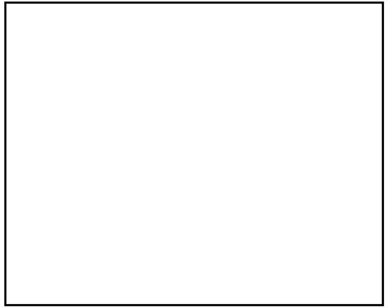
Summary

Last session
1 AI definition
2 Applications & Open Issues
3 Deep learning
4 Foundation models

Today's session
Learning from labeled examples
Challenges of supervised learning

Notations

Vector space ($\mathbb{R}^d$)

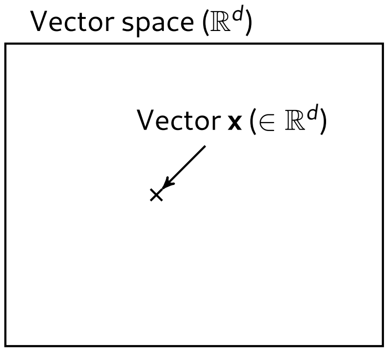


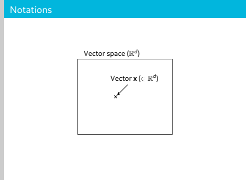
Notations

Vector space ($\mathbb{R}^d$)

We denote a vector space of real values in dimension d. We will consider vectors x in this space, and the set big X of all such vectors.

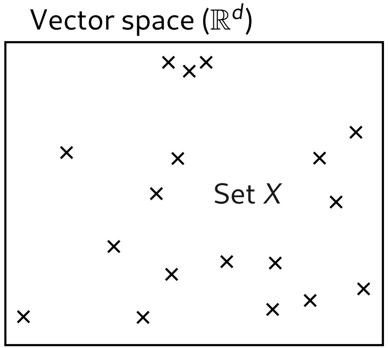
Notations

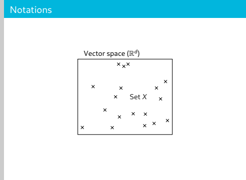




We denote a vector space of real values in dimension d. We will consider vectors x in this space, and the set big X of all such vectors.

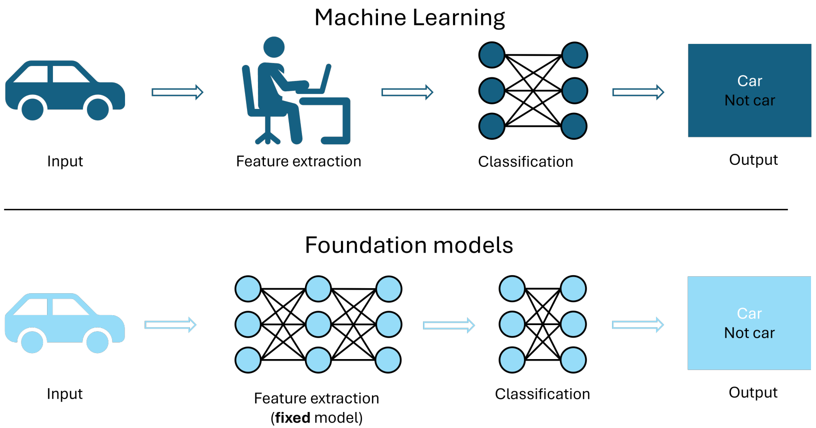
Notations





We denote a vector space of real values in dimension d. We will consider vectors x in this space, and the set big X of all such vectors.

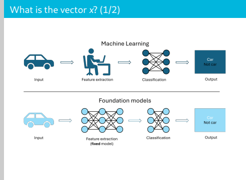
What is the vector x? (1/2)



2024-09-30

Course 2: Supervised Learning

└What is the vector x? (1/2)



What is the vector x? (2/2)

Traditional Machine Learning

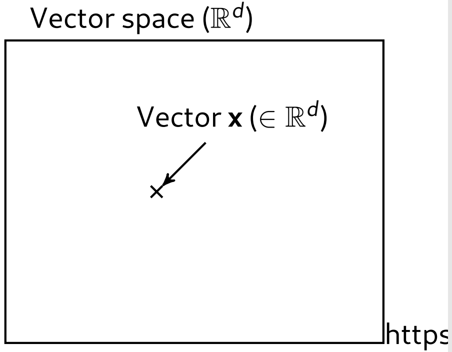
x is the data, or a small transformation of the data

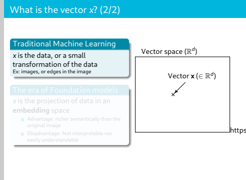
Ex: images, or edges in the image

The era of Foundation models

x is the projection of data in an **embedding** space

- Advantage: richer semantically than the original image
- Disadvantage: Not interpretable nor easily understandable





Embeddings are vectors in the latent space, i.e. the input vectors (image, text, ...) that have been mapped in a lower dimensional space by a function $f : R^d \to R^l$. Embeddings are usually richer semantically and easier to manipulate for a downstream task (e.g. classification). See also Lab 1 for more examples.

Traditional Machine Learning

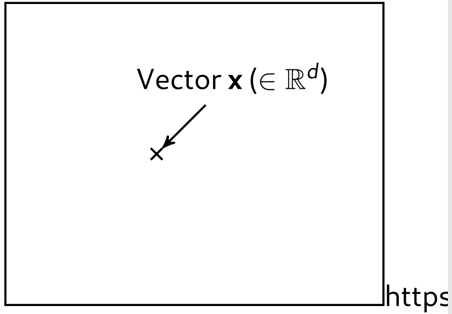
x is the data, or a small transformation of the data
Ex: images, or edges in the image

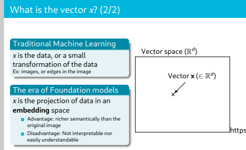
The era of Foundation models

x is the projection of data in an **embedding** space

- Advantage: richer semantically than the original image
- Disadvantage: Not interpretable nor easily understandable

Vector space ($\mathbb{R}^d$)





Embeddings are vectors in the latent space, i.e. the input vectors (image, text, ...) that have been mapped in a lower dimensional space by a function $f : R^d \rightarrow R^l$. Embeddings are usually richer semantically and easier to manipulate for a downstream task (e.g. classification). See also Lab 1 for more examples.

What is the vector x? (2/2)

Traditional Machine Learning

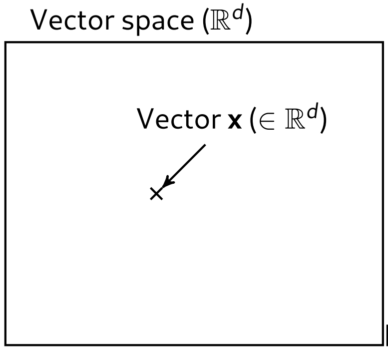
x is the data, or a small transformation of the data

Ex: images, or edges in the image

The era of Foundation models

x is the projection of data in an **embedding** space

- Advantage: richer semantically than the original image
- Disadvantage: Not interpretable nor easily understandable



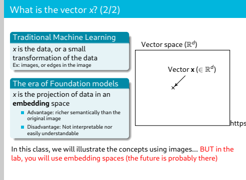
https

In this class, we will illustrate the concepts using images... BUT in the lab, you will use embedding spaces (the future is probably there)

2024-09-30

Course 2: Supervised Learning

└What is the vector x? (2/2)


What is the vector x? (2/2)
Traditional Machine Learning
x is the data, or a small transformation of the data
Ex: images, or edges in the image
The era of Foundation models
x is the projection of data in an embedding space
Advantage: richer semantically than the original image
Disadvantage: Not interpretable nor easily understandable
Vector space ($\mathbb{R}^d$)
Vector $\mathbf{x}$ ($\in \mathbb{R}^d$)
https
In this class, we will illustrate the concepts using images... BUT in the lab, you will use embedding spaces (the future is probably there)

Embeddings are vectors in the latent space, i.e. the input vectors (image, text, ...) that have been mapped in a lower dimensional space by a function $f : R^d \rightarrow R^l$. Embeddings are usually richer semantically and easier to manipulate for a downstream task (e.g. classification). See also Lab 1 for more examples.

Definition

Given:

- **x**: inputs (raw signals or feature vectors (e.g. embeddings))
- $\hat{\mathbf{y}}$: **labels** (annotated by humans)

Learn:

- a function *f*() such that $\hat{\mathbf{y}} \approx f(\mathbf{x})$
  $\Rightarrow$ *f*() is **learned** by the Machine Learning algorithm
- Ideally, *f*() should **generalize** ($\neq$ memorize) to unlabeled examples.

$f(\mathbf{x})$: 

$\hat{\mathbf{y}}$: "cat"

Supervised learning

Definition

Given:

- x: inputs (raw signals or feature vectors (e.g. embeddings))
- ŷ: labels (annotated by humans)

Learn:

- a function f() such that ŷ ≈ f(x)
  ⇒ f() is learned by the Machine Learning algorithm
- Ideally, f() should generalize (≠ memorize) to unlabeled examples

f(x): 

ŷ: "cat"

Definition

Given:

- **x**: inputs (raw signals or feature vectors (e.g. embeddings))
- $\hat{\mathbf{y}}$: **labels** (annotated by humans)

Learn:

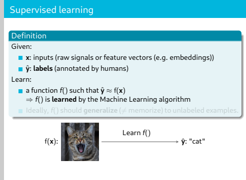
- a function $f()$ such that $\hat{\mathbf{y}} \approx \mathrm{f}(\mathbf{x})$
  $\Rightarrow$ $f()$ is **learned** by the Machine Learning algorithm
- Ideally, $f()$ should **generalize** ($\neq$ memorize) to unlabeled examples.

f(**x**):  Learn $f()$ → $\hat{\mathbf{y}}$: "cat"

2024-09-30

Course 2: Supervised Learning

└─Supervised learning



Supervised learning

Definition

Given:

- $\mathbf{x}$: inputs (raw signals or feature vectors (e.g. embeddings))
- $\hat{\mathbf{y}}$: **labels** (annotated by humans)

Learn:

- a function $f()$ such that $\hat{\mathbf{y}} \approx f(\mathbf{x})$
  $\Rightarrow$ $f()$ is **learned** by the Machine Learning algorithm
- Ideally, $f()$ should **generalize** ($\neq$ memorize) to unlabeled examples.

$f(\mathbf{x})$:  $\xrightarrow{\text{Learn } f()}$ $\hat{\mathbf{y}}$: "cat"

2024-09-30

Course 2: Supervised Learning

└Supervised learning

Supervised learning

Definition

Given:

- $\mathbf{x}$: inputs (raw signals or feature vectors (e.g. embeddings))
- $\hat{\mathbf{y}}$: **labels** (annotated by humans)

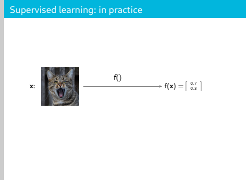
Learn:

- a function $f()$ such that $\hat{\mathbf{y}} \approx f(\mathbf{x})$
  $\Rightarrow$ $f()$ is **learned** by the Machine Learning algorithm
- Ideally, $f()$ should **generalize** ($\neq$ memorize) to unlabeled examples.

$f(\mathbf{x})$: $\xrightarrow{\text{Learn } f()}$ $\hat{\mathbf{y}}$: "cat"

Supervised learning: in practice

x:  $\xrightarrow{f()}$ $f(\mathbf{x}) = \begin{bmatrix} 0.7 \\ 0.3 \end{bmatrix}$



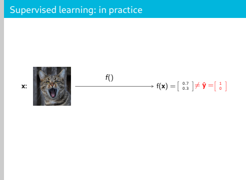
Supervised learning: in practice

x:  $\xrightarrow{\hspace{4em}f()\hspace{4em}}$ $f(\mathbf{x}) = \begin{bmatrix} 0.7 \\ 0.3 \end{bmatrix} \neq \hat{\mathbf{y}} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$

2024-09-30

Course 2: Supervised Learning

└Supervised learning: in practice



Supervised learning: in practice

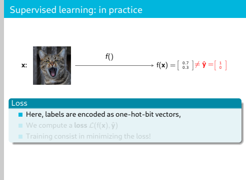
$\mathbf{x}$:

$$\mathbf{x} \xrightarrow{f()} f(\mathbf{x}) = \begin{bmatrix} 0.7 \\ 0.3 \end{bmatrix} \neq \hat{\mathbf{y}} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

Loss

- Here, labels are encoded as one-hot-bit vectors,
- We compute a **loss** $\mathcal{L}(f(\mathbf{x}), \hat{\mathbf{y}})$
- Training consist in minimizing the loss!

2024-09-30



Supervised learning: in practice

x: 

$$\mathbf{x} \xrightarrow{f()} \mathrm{f}(\mathbf{x}) = \begin{bmatrix} 0.7 \\ 0.3 \end{bmatrix} \neq \hat{\mathbf{y}} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

Loss

- Here, labels are encoded as one-hot-bit vectors,
- We compute a **loss** $\mathcal{L}(\mathrm{f}(\mathbf{x}), \hat{\mathbf{y}})$
  - Euclidean distance $\mathcal{L}(\mathrm{f}(\mathbf{x}), \hat{\mathbf{y}}) = \sum_{i=1}^{D} (\mathrm{f}(\mathbf{x})_i - \hat{\mathbf{y}}_i)^2$
  - Cross-entropy: $\mathcal{L}(\mathrm{f}(\mathbf{x}), \hat{\mathbf{y}}) = -\sum_{i=1}^{D} \hat{\mathbf{y}}_i \log(\mathrm{f}(\mathbf{x})_i)$
    $\Rightarrow$ To prevent the model to classify everything as one, outputs are **softmaxed**:
    $\mathrm{f}(\mathbf{x})_i = \frac{e^{\mathrm{f}(\mathbf{x})_i}}{\sum_{j=1}^{D} e^{\mathrm{f}(\mathbf{x})_j}}$
- Training consist in minimizing the loss!

2024-09-30

Course 2: Supervised Learning

└Supervised learning: in practice

x: 

$$\xrightarrow{f()} \quad f(\mathbf{x}) = \begin{bmatrix} 0.7 \\ 0.3 \end{bmatrix} \neq \hat{\mathbf{y}} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

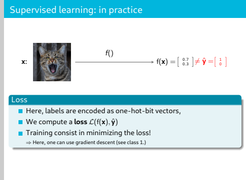
Loss

- Here, labels are encoded as one-hot-bit vectors,
- We compute a **loss** $\mathcal{L}(f(\mathbf{x}), \hat{\mathbf{y}})$
- Training consist in minimizing the loss!

  $\Rightarrow$ Here, one can use gradient descent (see class 1.)

2024-09-30 Course 2: Supervised Learning

└─Supervised learning: in practice



Supervised learning: in practice

x:  $\xrightarrow{f()}$ $\mathrm{f}(\mathbf{x}) = \begin{bmatrix} 0.2 \\ 0.8 \end{bmatrix} \color{red}{\neq \hat{\mathbf{y}} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}}$

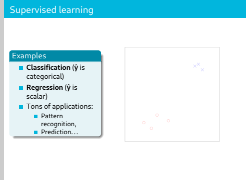
Loss

- Here, labels are encoded as one-hot-bit vectors,
- We compute a **loss** $\mathcal{L}(\mathrm{f}(\mathbf{x}), \hat{\mathbf{y}})$
- Training consist in minimizing the loss!

  $\Rightarrow$ Here, one can use gradient descent (see class 1.)

Examples

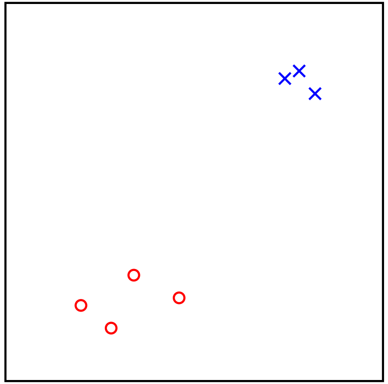
- **Classification** ($\hat{y}$ is categorical)
- **Regression** ($\hat{y}$ is scalar)
- Tons of applications:
  - Pattern recognition,
  - Prediction...

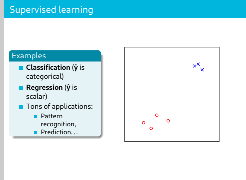


- We insist here one more time on the fact that learning is not memorizing. An expert is needed to provide the labels, that is why it is "supervised".
- Few examples of regression tasks (predicting the price of a product in the stock market, the age of a person based on his/her face, ...) and classification tasks (recognizing apples versus oranges).
- When the plot appears, say that for example if we have the points labeled in blue and the points labeled in red, a simple function could be learnt by just dividing the space in two regions.
- However if we present a new point (not part of training) that lies in the red region and is supposed to be "blue", then it means we are not generalizing.
- Finally, we present here another way to "learn", by defining the so-called Voronoi diagram , which are the regions of the space that are closer to one point than any other point.

Examples

- **Classification** ($\hat{y}$ is categorical)
- **Regression** ($\hat{y}$ is scalar)
- Tons of applications:
  - Pattern recognition,
  - Prediction...

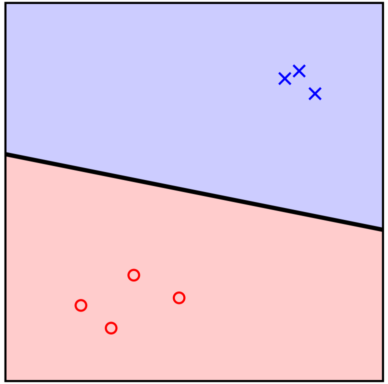


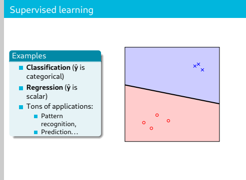


- We insist here one more time on the fact that learning is not memorizing. An expert is needed to provide the labels, that is why it is "supervised".
- Few examples of regression tasks (predicting the price of a product in the stock market, the age of a person based on his/her face, ...) and classification tasks (recognizing apples versus oranges).
- When the plot appears, say that for example if we have the points labeled in blue and the points labeled in red, a simple function could be learnt by just dividing the space in two regions.
- However if we present a new point (not part of training) that lies in the red region and is supposed to be "blue", then it means we are not generalizing.
- Finally, we present here another way to "learn", by defining the so-called Voronoi diagram , which are the regions of the space that are closer to one point than any other point.

Examples

- **Classification** ($\hat{y}$ is categorical)
- **Regression** ($\hat{y}$ is scalar)
- Tons of applications:
  - Pattern recognition,
  - Prediction...



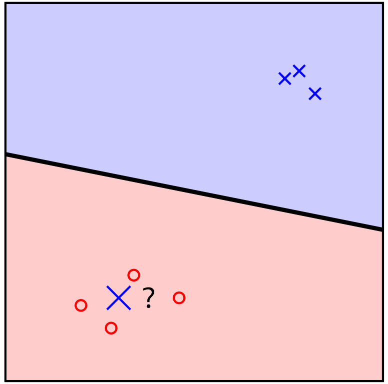


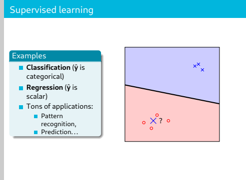
- We insist here one more time on the fact that learning is not memorizing. An expert is needed to provide the labels, that is why it is "supervised".
- Few examples of regression tasks (predicting the price of a product in the stock market, the age of a person based on his/her face, ...) and classification tasks (recognizing apples versus oranges).
- When the plot appears, say that for example if we have the points labeled in blue and the points labeled in red, a simple function could be learnt by just dividing the space in two regions.
- However if we present a new point (not part of training) that lies in the red region and is supposed to be "blue", then it means we are not generalizing.
- Finally, we present here another way to "learn", by defining the so-called Voronoi diagram , which are the regions of the space that are closer to one point than any other point.

Supervised learning

Examples

- **Classification** ($\hat{y}$ is categorical)
- **Regression** ($\hat{y}$ is scalar)
- Tons of applications:
  - Pattern recognition,
  - Prediction...

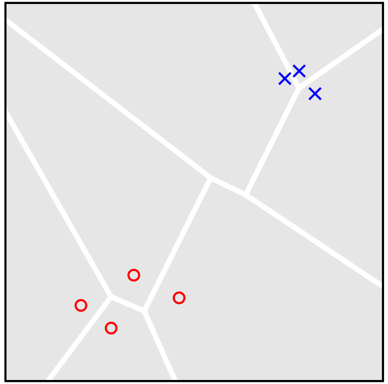


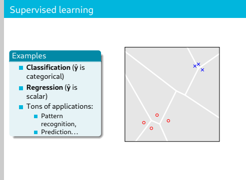


- We insist here one more time on the fact that learning is not memorizing. An expert is needed to provide the labels, that is why it is "supervised".
- Few examples of regression tasks (predicting the price of a product in the stock market, the age of a person based on his/her face, ...) and classification tasks (recognizing apples versus oranges).
- When the plot appears, say that for example if we have the points labeled in blue and the points labeled in red, a simple function could be learnt by just dividing the space in two regions.
- However if we present a new point (not part of training) that lies in the red region and is supposed to be "blue", then it means we are not generalizing.
- Finally, we present here another way to "learn", by defining the so-called Voronoi diagram , which are the regions of the space that are closer to one point than any other point.

Examples

- **Classification** ($\hat{y}$ is categorical)
- **Regression** ($\hat{y}$ is scalar)
- Tons of applications:
  - Pattern recognition,
  - Prediction...



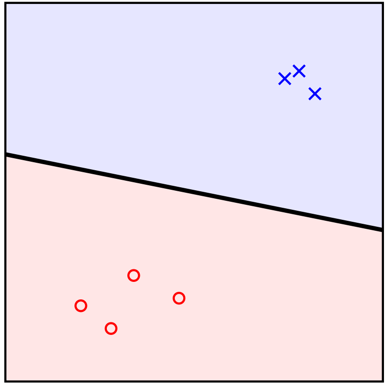


- We insist here one more time on the fact that learning is not memorizing. An expert is needed to provide the labels, that is why it is "supervised".
- Few examples of regression tasks (predicting the price of a product in the stock market, the age of a person based on his/her face, ...) and classification tasks (recognizing apples versus oranges).
- When the plot appears, say that for example if we have the points labeled in blue and the points labeled in red, a simple function could be learnt by just dividing the space in two regions.
- However if we present a new point (not part of training) that lies in the red region and is supposed to be "blue", then it means we are not generalizing.
- Finally, we present here another way to "learn", by defining the so-called Voronoi diagram , which are the regions of the space that are closer to one point than any other point.

Challenges of supervised learning (1/5)

An ill-defined problem

- An infinity of potential solutions, one must be the “best one” but is unreachable,
- $\Rightarrow$ requires **priors or constraints**.



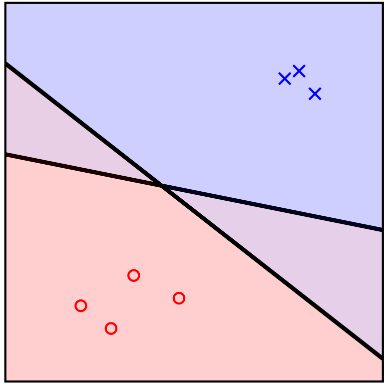
Challenges of supervised learning (1/5)
An ill-defined problem
An infinity of potential solutions, one must be the “best one” but is unreachable,
$\Rightarrow$ requires **priors or constraints**.

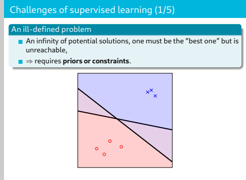
The point here is simply illustrate the fact that the solution is not unique. One way to find a solution that could be "better" than another one is to use prior knowledge or constraints of the problem at hand.

Challenges of supervised learning (1/5)

An ill-defined problem

- An infinity of potential solutions, one must be the “best one” but is unreachable,
- ⇒ requires **priors or constraints**.



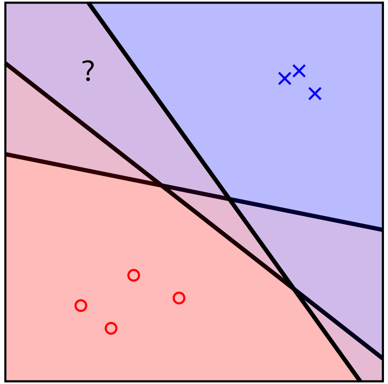


The point here is simply illustrate the fact that the solution is not unique. One way to find a solution that could be "better" than another one is to use prior knowledge or constraints of the problem at hand.

Challenges of supervised learning (1/5)

An ill-defined problem

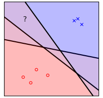
- An infinity of potential solutions, one must be the “best one” but is unreachable,
- $\Rightarrow$ requires **priors or constraints**.



Challenges of supervised learning (1/5)

An ill-defined problem

- An infinity of potential solutions, one must be the “best one” but is unreachable,
- ⇒ requires **priors or constraints**.

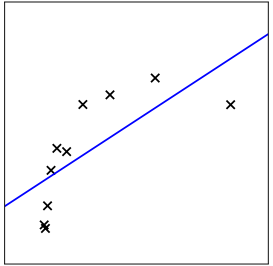


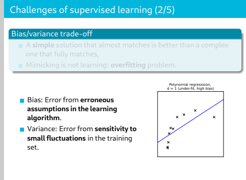
The point here is simply illustrate the fact that the solution is not unique. One way to find a solution that could be "better" than another one is to use prior knowledge or constraints of the problem at hand.

Bias/variance trade-off

- A **simple** solution that almost matches is better than a complex one that fully matches,
- Mimicking is not learning: **overfitting** problem.

- Bias: Error from **erroneous assumptions in the learning algorithm.**
- Variance: Error from **sensitivity to small fluctuations** in the training set.





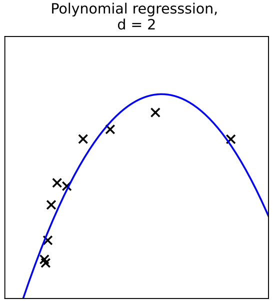
In the first part, the goal is to show what happens when trying to learn a polynomial function, with a polynomial regression of degree d (i.e. fitting points with a polynome of degree d). If the regression model is a polynomial of degree 1, it is not able to fit the points. If we take a polynomial of degree 2, it is able to fit the points, but not in a very good way. If we take a polynomial of degree 6, it fits the points very well, but it is not a good estimator, as it is not able to generalize to other points. This is the overfitting problem. Hence, a high bias indicates erroneus assumptions in the learning algorithm, and a high variance indicates that the algorithm is very sensitive to particularities in training data.
In the second part, learning curves are presented, with the goal to illustrate overfitting. The diagram on the left shows the error (in regression or classification). The X axis is illustrative, it doesn't correspond to something specific (although one could imagine it to correspond to order of a polynomial, epochs of training a neural net, ...) but it illustrates the situations of underfitting and overfitting.

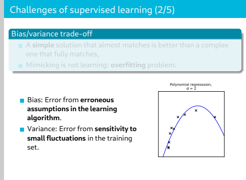
Cross-validation (https://en.wikipedia.org/wiki/Cross-validation_(statistics)).

Bias/variance trade-off

- A **simple** solution that almost matches is better than a complex one that fully matches,
- Mimicking is not learning: **overfitting** problem.

- Bias: Error from **erroneous assumptions in the learning algorithm**.
- Variance: Error from **sensitivity to small fluctuations** in the training set.





In the first part, the goal is to show what happens when trying to learn a polynomial function, with a polynomial regression of degree d (i.e. fitting points with a polynome of degree d). If the regression model is a polynomial of degree 1, it is not able to fit the points. If we take a polynomial of degree 2, it is able to fit the points, but not in a very good way. If we take a polynomial of degree 6, it fits the points very well, but it is not a good estimator, as it is not able to generalize to other points. This is the overfitting problem. Hence, a high bias indicates erroneus assumptions in the learning algorithm, and a high variance indicates that the algorithm is very sensitive to particularities in training data.

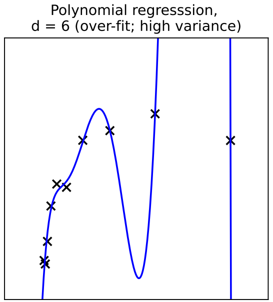
In the second part, learning curves are presented, with the goal to illustrate overfitting. The diagram on the left shows the error (in regression or classification). The X axis is illustrative, it doesn't correspond to something specific (although one could imagine it to correspond to order of a polynomial, epochs of training a neural net, ...) but it illustrates the situations of underfitting and overfitting.

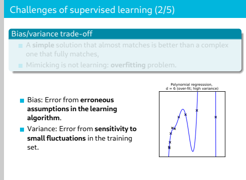
Cross-validation (https://en.wikipedia.org/wiki/Cross-validation_(statistics)).

Bias/variance trade-off

- A **simple** solution that almost matches is better than a complex one that fully matches,
- Mimicking is not learning: **overfitting** problem.

- Bias: Error from **erroneous assumptions in the learning algorithm**.
- Variance: Error from **sensitivity to small fluctuations** in the training set.





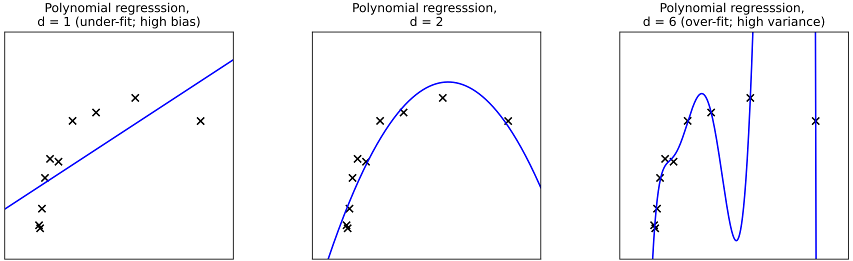
In the first part, the goal is to show what happens when trying to learn a polynomial function, with a polynomial regression of degree d (i.e. fitting points with a polynome of degree d). If the regression model is a polynomial of degree 1, it is not able to fit the points. If we take a polynomial of degree 2, it is able to fit the points, but not in a very good way. If we take a polynomial of degree 6, it fits the points very well, but it is not a good estimator, as it is not able to generalize to other points. This is the overfitting problem. Hence, a high bias indicates erroneus assumptions in the learning algorithm, and a high variance indicates that the algorithm is very sensitive to particularities in training data.
In the second part, learning curves are presented, with the goal to illustrate overfitting. The diagram on the left shows the error (in regression or classification). The X axis is illustrative, it doesn't correspond to something specific (although one could imagine it to correspond to order of a polynomial, epochs of training a neural net, ...) but it illustrates the situations of underfitting and overfitting.

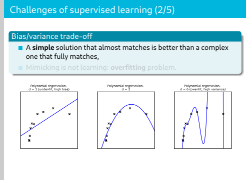
Cross-validation (https://en.wikipedia.org/wiki/Cross-validation_(statistics)).

Challenges of supervised learning (2/5)

Bias/variance trade-off

- A **simple** solution that almost matches is better than a complex one that fully matches,
- Mimicking is not learning: **overfitting** problem.





In the first part, the goal is to show what happens when trying to learn a polynomial function, with a polynomial regression of degree d (i.e. fitting points with a polynome of degree d). If the regression model is a polynomial of degree 1, it is not able to fit the points. If we take a polynomial of degree 2, it is able to fit the points, but not in a very good way. If we take a polynomial of degree 6, it fits the points very well, but it is not a good estimator, as it is not able to generalize to other points. This is the overfitting problem. Hence, a high bias indicates erroneus assumptions in the learning algorithm, and a high variance indicates that the algorithm is very sensitive to particularities in training data.

In the second part, learning curves are presented, with the goal to illustrate overfitting. The diagram on the left shows the error (in regression or classification). The X axis is illustrative, it doesn't correspond to something specific (although one could imagine it to correspond to order of a polynomial, epochs of training a neural net, ...) but it illustrates the situations of underfitting and overfitting.

Cross-validation (https://en.wikipedia.org/wiki/Cross-validation_(statistics)).

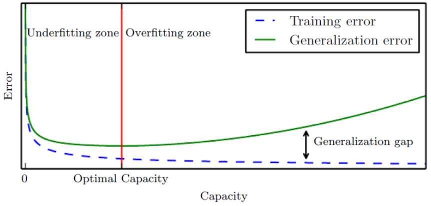
Challenges of supervised learning (2/5)

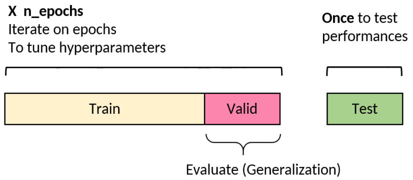
Bias/variance trade-off

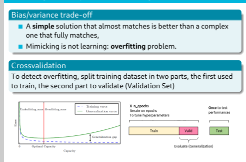
- A **simple** solution that almost matches is better than a complex one that fully matches,
- Mimicking is not learning: **overfitting** problem.

Crossvalidation

To detect overfitting, split training dataset in two parts, the first used to train, the second part to validate (Validation Set)







In the first part, the goal is to show what happens when trying to learn a polynomial function, with a polynomial regression of degree d (i.e. fitting points with a polynome of degree d). If the regression model is a polynomial of degree 1, it is not able to fit the points. If we take a polynomial of degree 2, it is able to fit the points, but not in a very good way. If we take a polynomial of degree 6, it fits the points very well, but it is not a good estimator, as it is not able to generalize to other points. This is the overfitting problem. Hence, a high bias indicates erroneus assumptions in the learning algorithm, and a high variance indicates that the algorithm is very sensitive to particularities in training data.

In the second part, learning curves are presented, with the goal to illustrate overfitting. The diagram on the left shows the error (in regression or classification). The X axis is illustrative, it doesn't correspond to something specific (although one could imagine it to correspond to order of a polynomial, epochs of training a neural net, ...) but it illustrates the situations of underfitting and overfitting.

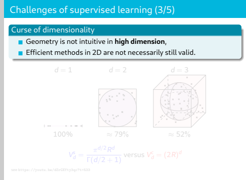
Cross-validation (https://en.wikipedia.org/wiki/Cross-validation_(statistics)).

Curse of dimensionality

- Geometry is not intuitive in **high dimension**,
- Efficient methods in 2D are not necessarily still valid.

$$V_d^s = \frac{\pi^{d/2} R^d}{\Gamma(d/2+1)} \text{ versus } V_d^c = (2R)^d$$

see https://youtu.be/dZrGXYty3qc?t=533

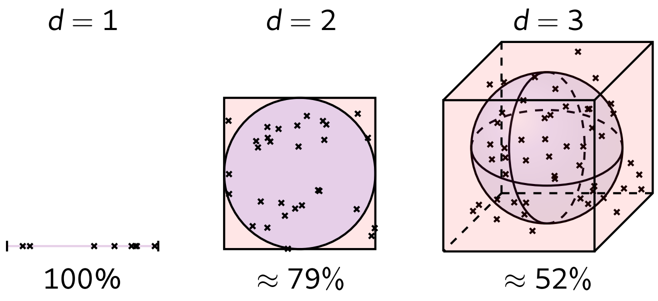


The point here is to show that when the dimension increases, the space tends to be more and more "empty". V_d^s is the volume of the hypersphere, and V_d^c is the volume of the hypercube. The crosses in the different figures are generated by each coordinates following a uniform distribution $\mathcal{U}(0, R)$ (so on average they have a value of $R/2$). When d increases, the ratio between the hypersphere and the hypercube becomes smaller and smaller, so that the majority of the volume of the hypercube lies in the corners, this means that the majority of crosses will be equally far from the center of the hypersphere (for instance a nearest neighbors algorithm would not work at all!). The intuitions we have easily in 2D are not valid anymore, so we can imagine why it is difficult to build good classifiers in high dimensions.

Challenges of supervised learning (3/5)

Curse of dimensionality

- Geometry is not intuitive in **high dimension**,
- Efficient methods in 2D are not necessarily still valid.



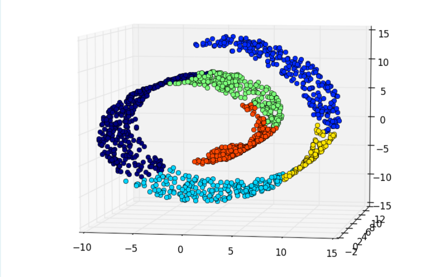
$$V_d^s = \frac{\pi^{d/2} R^d}{\Gamma(d/2+1)} \text{ versus } V_d^c = (2R)^d$$

see `https://youtu.be/dZrGXYty3qc?t=533`

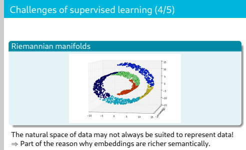
Challenges of supervised learning (3/5)
Curse of dimensionality
Geometry is not intuitive in **high dimension**,
Efficient methods in 2D are not necessarily still valid.

The point here is to show that when the dimension increases, the space tends to be more and more "empty". V_d^s is the volume of the hypersphere, and V_d^c is the volume of the hypercube. The crosses in the different figures are generated by each coordinates following a uniform distribution $\mathcal{U}(0, R)$ (so on average they have a value of $R/2$). When d increases, the ratio between the hypersphere and the hypercube becomes smaller and smaller, so that the majority of the volume of the hypercube lies in the corners, this means that the majority of crosses will be equally far from the center of the hypersphere (for instance a nearest neighbors algorithm would not work at all!). The intuitions we have easily in 2D are not valid anymore, so we can imagine why it is difficult to build good classifiers in high dimensions.

Riemannian manifolds



The natural space of data may not always be suited to represent data!
⇒ Part of the reason why embeddings are richer semantically.



Top part : the point here is to show an example of a dataset in 3D, which is actually much simpler because it is 1D. A nice example to explain the swiss roll is to explain how to roll the cake to make it !

Computation time

Example on ImageNet, simply going through all images:

- $n = 10.000.000$, $d \approx 1.000.000$,
- $\approx 10^{13}$ elementary operations,
- $\approx$ 2h45 on a modern processor.

Scalability

- Finding the best solution to a problem would be feasible with unlimited computation time,
- But searching through the space of possible functions is often **untractable**,
- Solutions must be computationally reasonable, which is the true challenge today.

Challenges of supervised learning (5/5)
Computation time
Example on ImageNet, simply going through all images:
$n = 10.000.000$, $d \approx 1.000.000$,
$\approx 10^{13}$ elementary operations,
$\approx$ 2h45 on a modern processor.
Scalability
Finding the best solution to a problem would be feasible with unlimited computation time,
But searching through the space of possible functions is often untractable,
Solutions must be computationally reasonable, which is the true challenge today.

This slide is pretty much self-explanatory. First, the goal is to show that just going through each image is very costly. Second, it is easy to explain why the space of possible functions quickly become so huge that it's not possible to search through it.

Challenges of supervised learning (5/5)

Computation time

Example on ImageNet, simply going through all images:

- $n = 10.000.000$, $d \approx 1.000.000$,
- $\approx 10^{13}$ elementary operations,
- $\approx$ 2h45 on a modern processor.

Scalability

- Finding the best solution to a problem would be feasible with unlimited computation time,
- But searching through the space of possible functions is often **untractable**,
- Solutions must be computationally reasonable, which is the true challenge today.

Challenges of supervised learning (5/5)

Computation time

Example on ImageNet, simply going through all images:

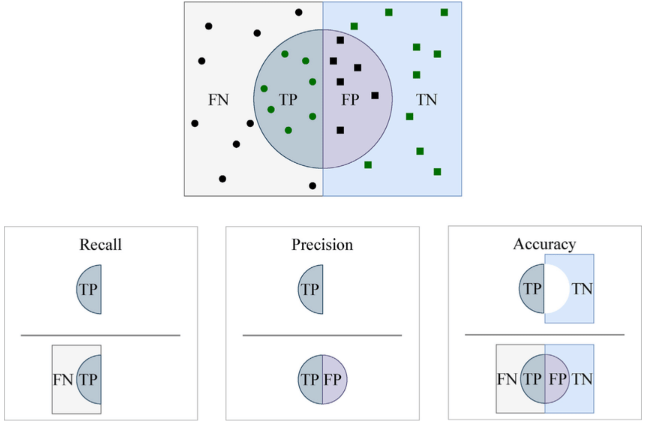
- $n = 10.000.000$, $d \approx 1.000.000$,
- $\approx 10^{13}$ elementary operations,
- $\approx$ 2h45 on a modern processor.

Scalability

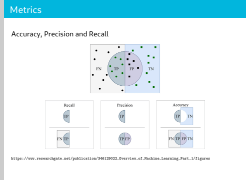
- Finding the best solution to a problem would be feasible with unlimited computation time,
- But searching through the space of possible functions is often **untractable**,
- Solutions must be computationally reasonable, which is the true challenge today.

This slide is pretty much self-explanatory. First, the goal is to show that just going through each image is very costly. Second, it is easy to explain why the space of possible functions quickly become so huge that it's not possible to search through it.

Accuracy, Precision and Recall

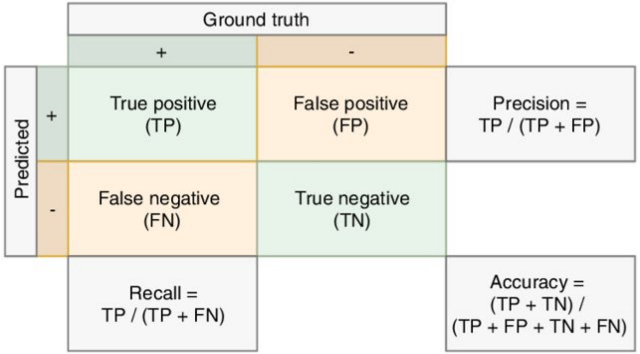


https://www.researchgate.net/publication/346129022_Overview_of_Machine_Learning_Part_1/figures

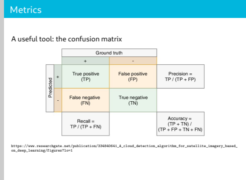


n.b. The picture is relative to a one class problem.
Accuracy: fraction of correctly classified instances over all instances (can be misleading for imbalanced classes)
Recall: fraction of positive (correctly retrieved) instances among relevant items
Precision: fraction of positive (correctly retrieved) instances among the retrieved instancesThe confusion matrix is useful to visualize the results of a supervised learning algorithm. It compares the instances of the ground truth (actual class) and the predicted class. The diagonal elements indicate the instances that are correctly predicted and the off diagonal elements the instances that are misclassified.

A useful tool: the confusion matrix



https://www.researchgate.net/publication/334840641_A_cloud_detection_algorithm_for_satellite_imagery_based_on_deep_learning/figures?lo=1



The confusion matrix is useful to visualize the results of a supervised learning algorithm. It compares the instances of the ground truth (actual class) and the predicted class. The diagonal elements indicate the instances that are correctly predicted and the off diagonal elements the instances that are misclassified.

Lab Session 2 and assignments for Session 3

Lab Supervised Learning

- Basics of machine learning using sklearn (including new definitions / concepts)
- Tests on the modality chosen in Lab 1 (text, vision or audio), based on the same foundation model than in Lab 1.

Project 1 (P1)

You will choose a supervised learning method among those available (see Lab 2). You will present

- A brief description of the theory behind the method,
- Basic tests on this technique for your modality.

During Session 3 you will have 7 minutes to present.

Lab Session 2 and assignments for Session 3
Lab Supervised Learning
Basics of machine learning using sklearn (including new definitions / concepts)
Tests on the modality chosen in Lab 1 (text, vision or audio), based on the same foundation model than in Lab 1.
Project 1 (P1)
You will choose a supervised learning method among those available (see Lab 2). You will present
A brief description of the theory behind the method,
Basic tests on this technique for your modality.
During Session 3 you will have 7 minutes to present.

Lab Session 2 and assignments for Session 3

List of Supervised Learning Methods

- Support Vector Machines (SVM)
- Decision Trees
- Multi-layer Perceptrons (MLP)
- Random Forest classifiers
- Logistic Regression
- Adaboost

2024-09-30

Course 2: Supervised Learning

Lab Session 2 and assignments for Session 3

Lab Session 2 and assignments for Session 3

List of Supervised Learning Methods

- Support Vector Machines (SVM)
- Decision Trees
- Multi-layer Perceptrons (MLP)
- Random Forest classifiers
- Logistic Regression
- Adaboost