Module overview

Sections

1. What is machine learning?
2. Business problems solved with machine learning
3. Machine learning process
4. Machine learning tools overview
5. Machine learning challenges

Demonstration

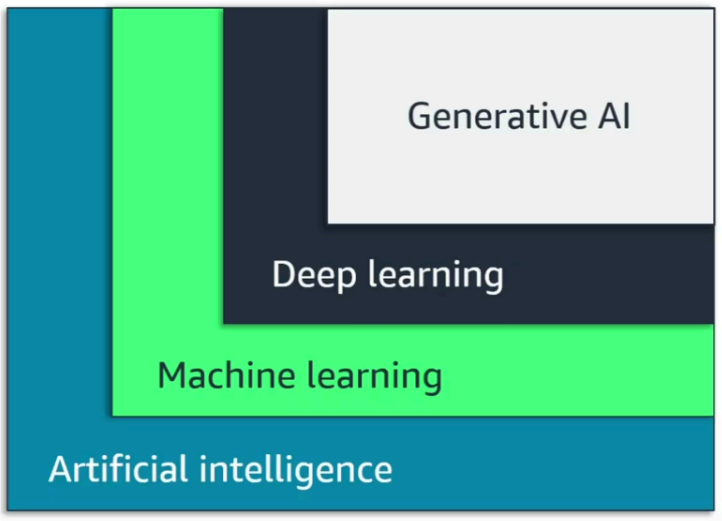
Introducing Amazon SageMaker

Module objectives

At the end of this module, you should be able to:

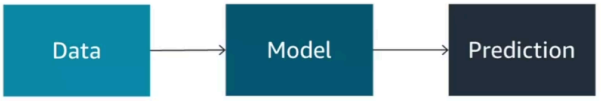
- Recognize how machine learning (ML), deep learning, and generative artificial intelligence (generative AI) are part of artificial intelligence (AI)
- Describe artificial intelligence and machine learning terminology
- Identify how machine learning can be used to solve a business problem
- Describe the machine learning process
- List the tools available to data scientists
- Identify when to use machine learning instead of traditional software development methods

AI, ML, deep learning, and generative AI



Machine learning

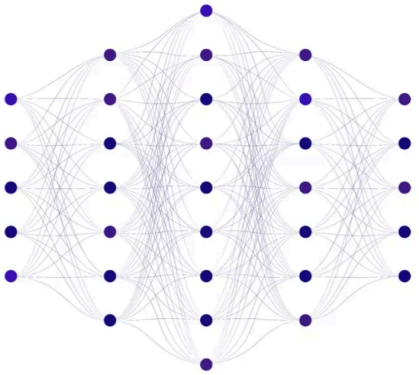
Machine learning is the scientific study of algorithms and statistical models to perform a task using inference instead of instructions.



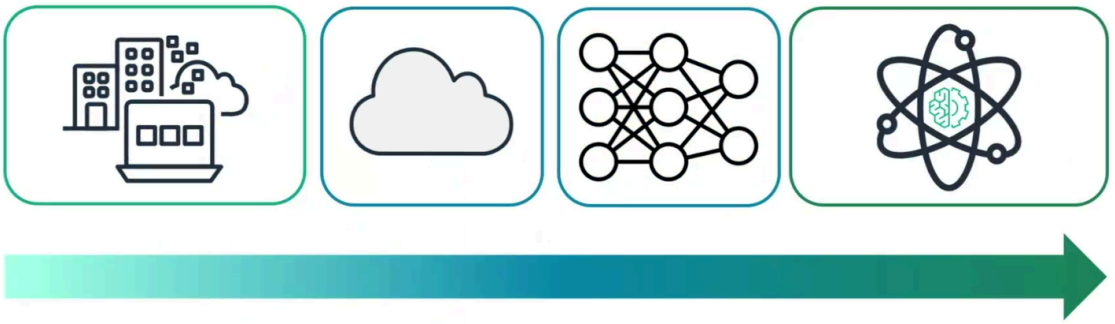
Machine learning flow

Deep learning

Artificial Neural Network



ML and technology advancements



Traditional computing

Cloud computing and Moore's law

Modern machine learning

Generative AI

Section 1 key takeaways

- Artificial intelligence
  - Machines performing human tasks
- Machine learning
  - Training models to make predictions
- Deep learning
  - Transformer-based neural networks
  - Neural Networks paved the way for generative AI
- Generative AI
  - Type of AI that can create new content
- Technology and economic advancements have made machine learning more accessible to individuals and organizations

aws

Section 2: Business problems solved with machine learning

Module 2: Introduction to Machine Learning

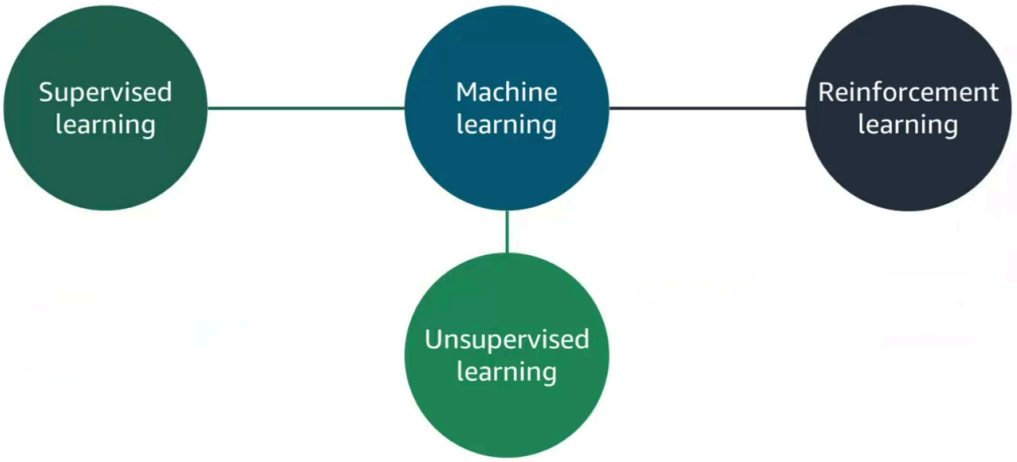
Common business use cases

Spam versus regular email

Recommendations

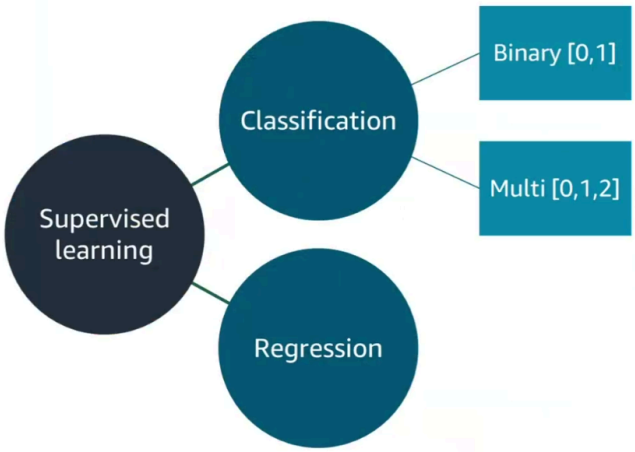
Fraud

Types of machine learning



Supervised learning

Learn by identifying patterns in data that is **already labeled**.

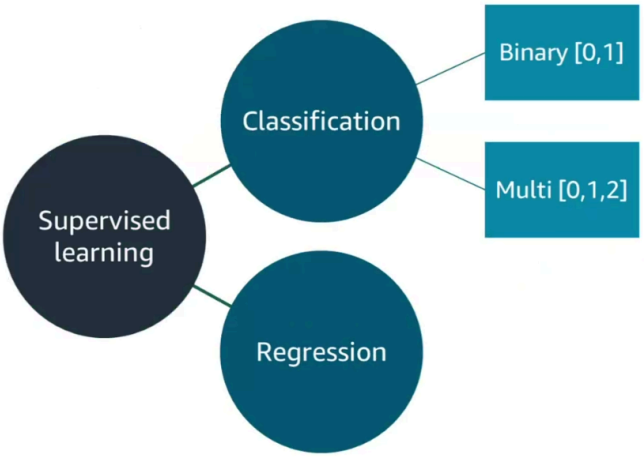


- Fraud detection
- Image recognition
- Customer retention
- Medical diagnostics
- Personalized advertising

- Product sales prediction
- Weather forecasting
- Market forecasting
- Population growth prediction

Supervised learning

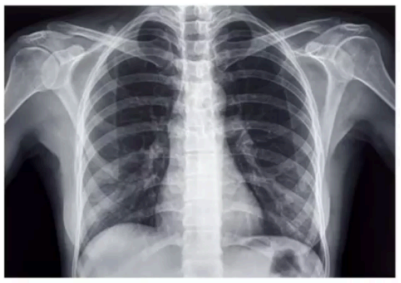
Learn by identifying patterns in data that is **already labeled**.

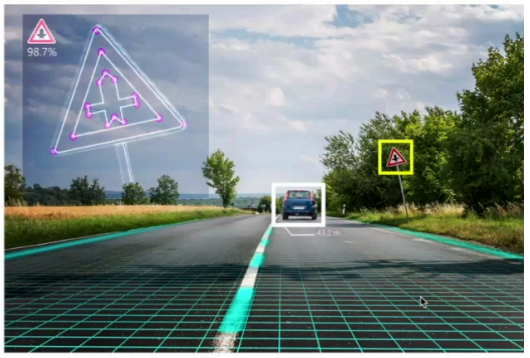


- Fraud detection
- Image recognition
- Customer retention
- Medical diagnostics
- Personalized advertising

- Product sales prediction
- Weather forecasting
- Market forecasting
- Population growth prediction

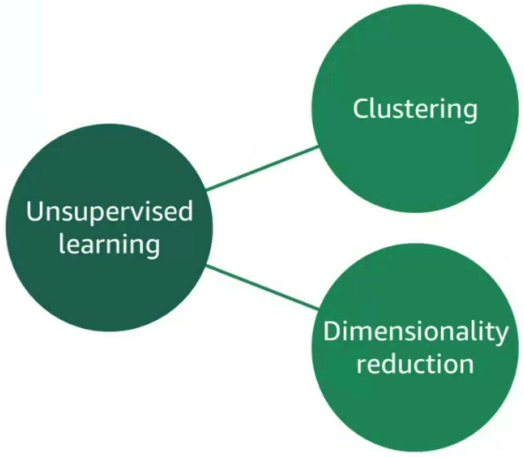
Computer vision





Unsupervised learning

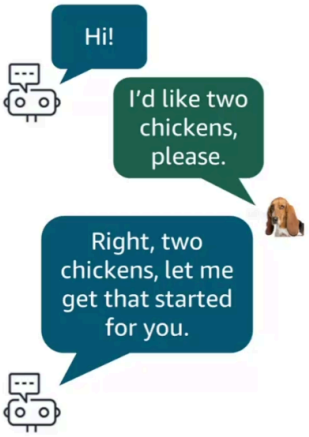
The machine must uncover and **create the labels** itself.



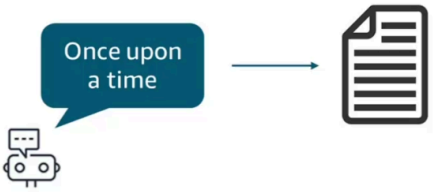
- Product recommendations
- Customer segmentation
- Targeted marketing
- Medical diagnostics

- Visualization
- Natural language processing
- Data structure discovery
- Gene sequencing

Natural language processing



gögn eru lykilatriði ⟷ los datos son clave

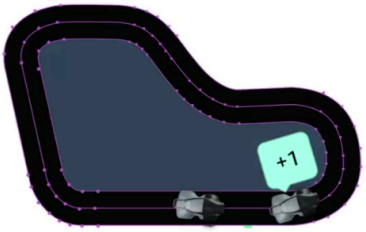


Poor story. Little character development. Jumps between scenes like you might get caught stealing. Unexplained bad guys appear with the thinnest of back story. Back to unlimited resources and let's not talk about the mechanics of building something so huge in such an inhospitable place...

Reinforcement learning

Learning through **trial and error**.

- Game AI
- Self-driving cars
- Robotics
- Customer service routing



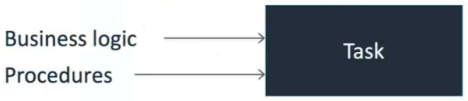
AWS DeepRacer

Best when the desired outcome is known but the exact path to achieving it is not known.

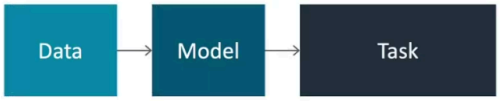
Self-driving vehicles

When to use machine learning?

Classical programming approach:



Machine learning approach:



Use machine learning when you have:

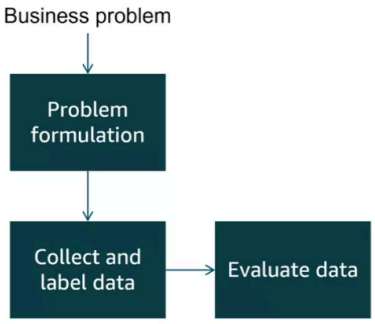
- Large datasets, large number of variables
- Lack of clear procedures to obtain the solution
- Existing machine learning expertise
- Infrastructure already in place to support ML
- Management support for ML

Section 2 key takeaways

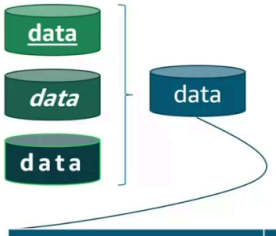
- Machine learning applications affect everyday life
- Machine learning can be grouped into –
  - Supervised learning
  - Unsupervised learning
  - Reinforcement learning
- Most problems are supervised learning

Section 3: Machine learning process

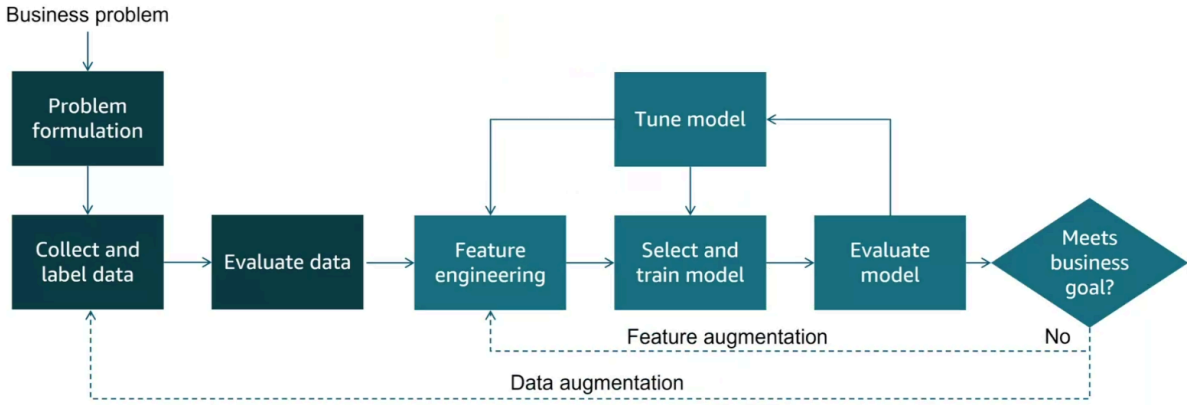
Module 2: Introduction to Machine Learning



Data handling and cleaning



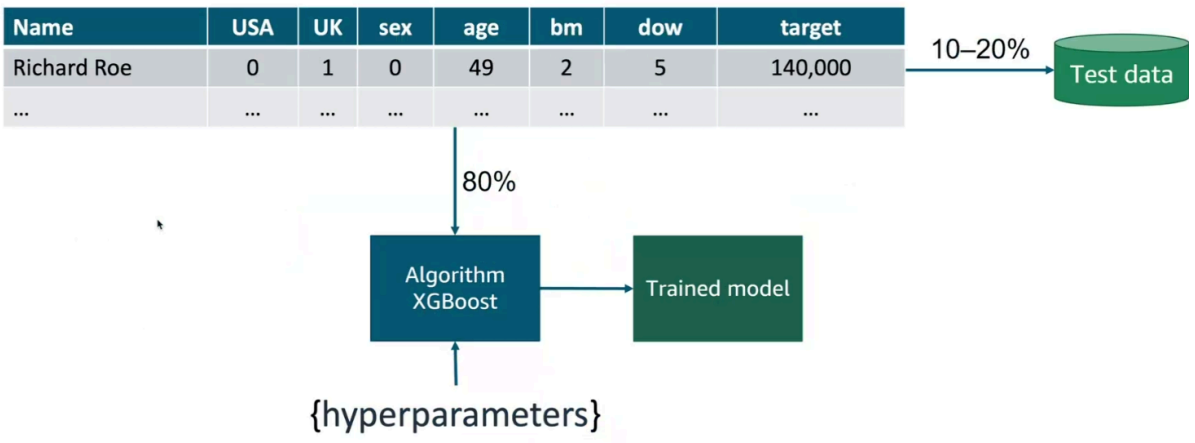
Name	Country	Sex	dob
Richard Roe	UK	Male	18/2/1972
Paulo Santos	Male		11/2/1969
Mrs. Mary Major	Denver	F	37
Desai, Arnav	USA	M	2/22/1962



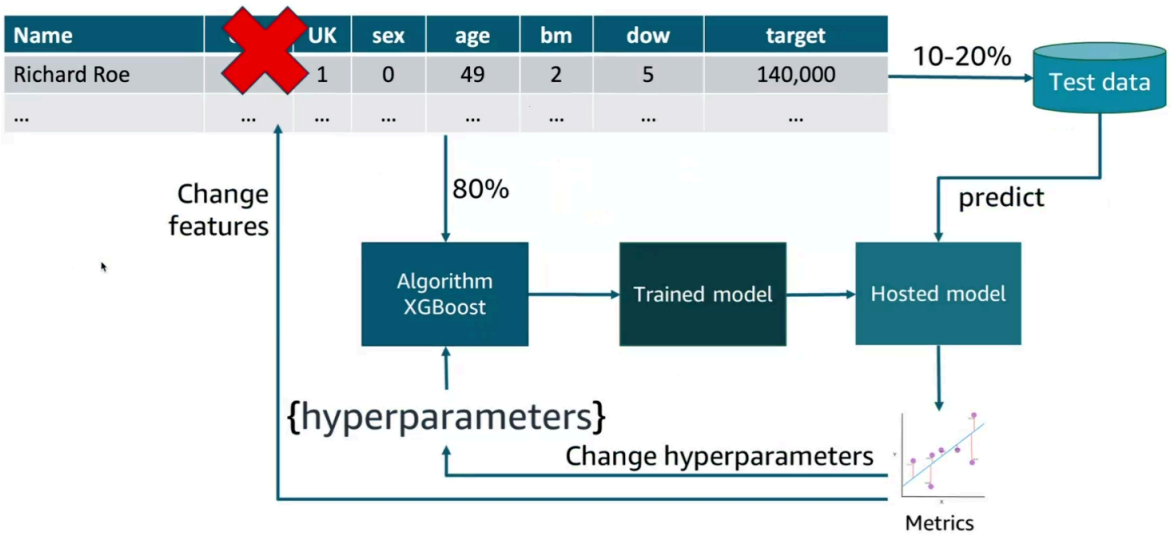
Name	Country	Sex	dob
Richard Roe	UK	Male	18/2/1972
Paulo Santos	Male		11/2/1969
Mrs. Mary Major	Denver	F	37
Desai, Arnav	USA	M	2/22/1962

?

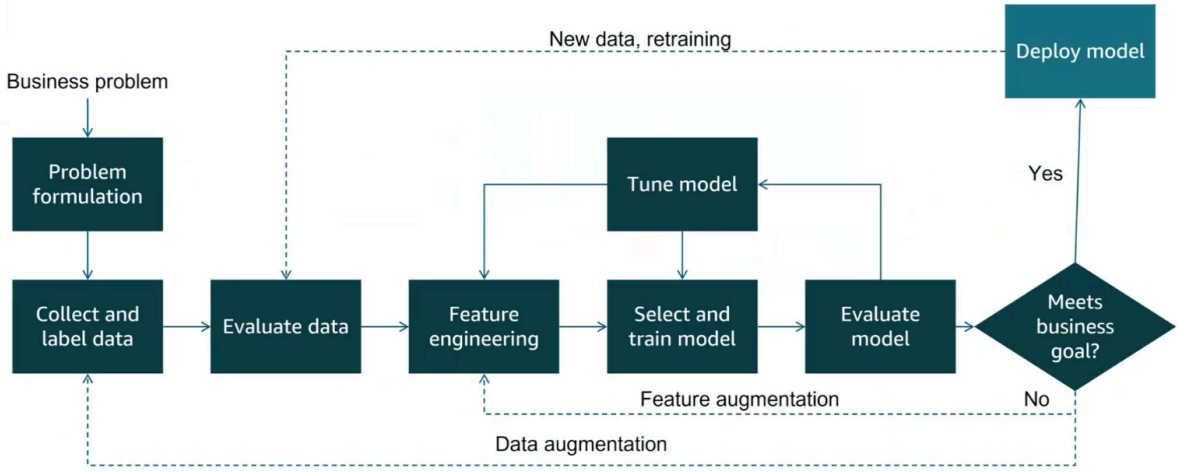
Name	USA	UK	sex	age	bm	dow	target
Richard Roe	0	1	0	49	2	5	140,000
Paulo Santos	1	0	0	51	11	7	78,000
Mary Major	1	0	1	37	NAN	0	167,000
Arnav Desai	1	0	0	58	2	4	100,000



ML pipeline: Evaluating and tuning the model



ML pipeline: Deployment



Section 3 key takeaways

- Machine learning pipeline guides you through the process of evaluating and training a model
- Iterative process of –
  - Data processing
  - Training
  - Evaluation

Python tools and libraries

- Jupyter Notebook
- JupyterLab
- pandas
- Matplotlib
- Seaborn
- NumPy
- scikit-learn

Machine learning frameworks and infrastructure

Machine learning **frameworks** provide tools and code libraries:

- Customized scripting
- Integration with AWS services
- Community of developers

Amazon **instances** that are designed for machine learning applications:

- AWS IoT Greengrass provides an infrastructure for building machine learning for IoT devices
- Amazon Elastic Inference reduces costs for running machine learning applications

PyTorch	Caffe2	Torch
TensorFlow	Gluon	Chainer
Keras	CNTK	Apache MXNet

EC2 P3 instances | EC2 C5 and C5n instances | AWS IoT Greengrass | Amazon Elastic Inference

Amazon SageMaker

Ground Truth

Set up and manage labeling jobs for highly accurate training datasets by using active learning and human labeling.

Notebook

Provide AWS and SageMaker SDKs and sample notebooks to create training jobs and deploy models.

Training

Train and tune models at any scale. Use high-performance AWS algorithms, or bring your own.

Inference

Create models from training jobs, or import external models for hosting so you can run inferences on new data.

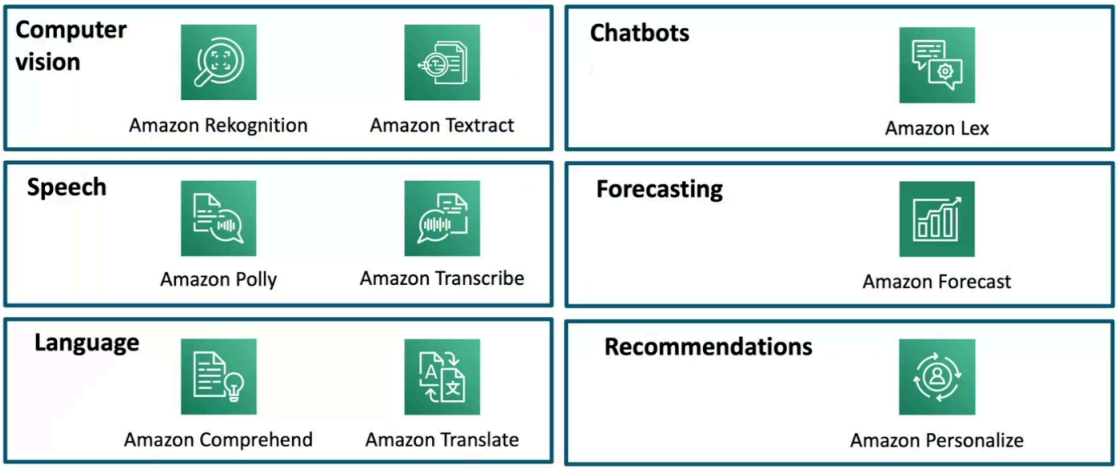
AWS Marketplace

Find, buy, and deploy ready-to-use model packages, algorithms, and data products in AWS Marketplace.

Demonstration: Introducing Amazon SageMaker

Machine learning managed services

These managed services don't require ML experience.



Section 4 key takeaways

- Python is the most popular ML language
- Jupyter Notebooks
- Many open-source tools
- Frameworks and services for all requirements
  - Low-level frameworks
  - Amazon SageMaker
  - Managed ML services

Machine learning challenges

Data

- Poor quality
- Non-representative
- Insufficient
- Overfitting and underfitting

Users

- Lack of data science expertise
- Cost of staffing with data scientists
- Lack of management support

Business

- Complexity in formulating questions
- Explaining models to the business
- Cost of building systems

Technology

- Data privacy issues
- Tool selection can be complicated
- Integration with other systems

Using existing models and services

Amazon ML managed services

- Amazon ML managed services
- No ML experience needed

You Only Look Once (YOLO)

- Use existing trained and tuned models
- Enhance with domain-specific instances

AWS Marketplace

- Over 250 ML model packages and algorithms
- Over 14 industry segments

Section 5 key takeaways

- Machine learning challenges
  - Data
  - People
  - Business
  - Technology
- Managed services simplify machine learning

Module takeaways

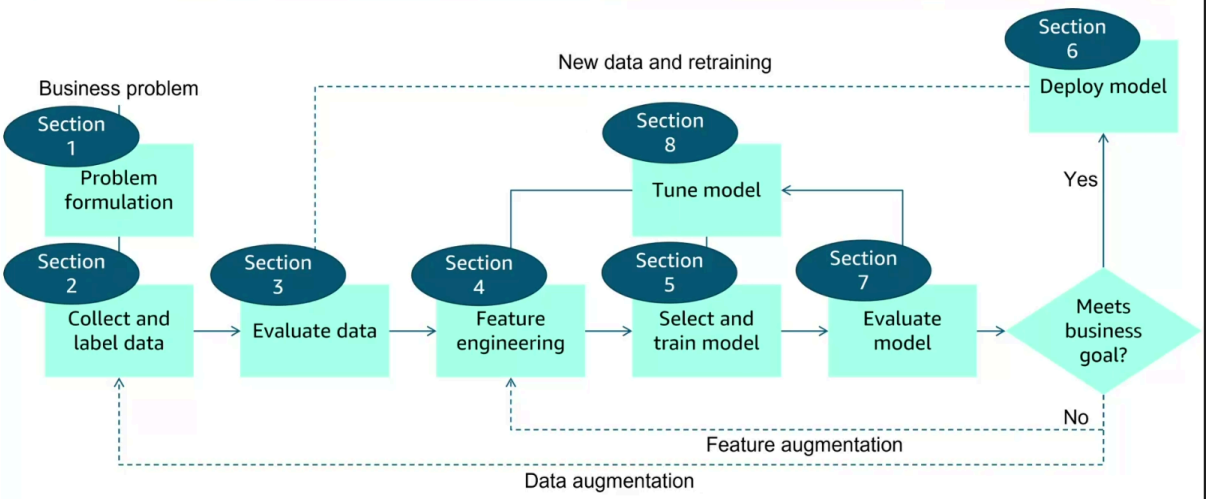
- Machine learning is a subset of artificial intelligence
  - Machine learning applies learning algorithms to develop models from large datasets
- The machine learning pipeline describes the different stages for developing a machine learning application
- The Amazon Machine Learning stack has three key layers
  - Managed services, machine learning services, machine learning frameworks
- Machine learning development is different from traditional development
  - Training algorithm is applied to data to create a model for making predictions

Module summary

In summary, in this module, you learned how to:

- Recognize how machine learning and deep learning are part of artificial intelligence
- Describe artificial intelligence and machine learning terminology
- Identify how machine learning can be used to solve a business problem
- Describe the machine learning process
- List the tools available to data scientists
- Identify when to use machine learning instead of traditional software development methods

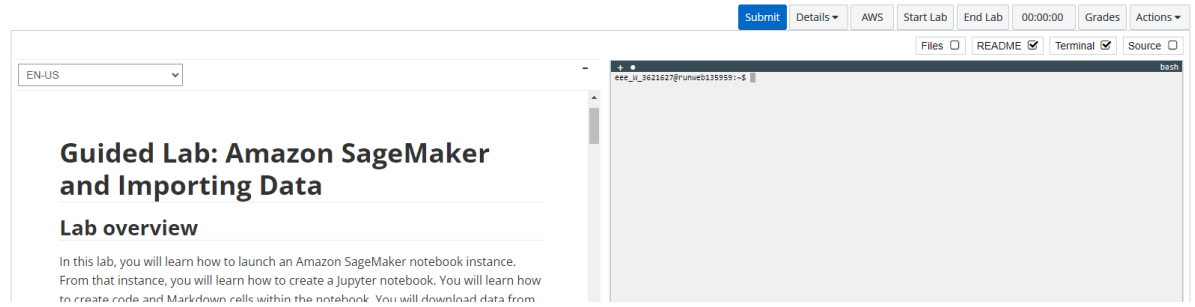
Machine learning pipeline

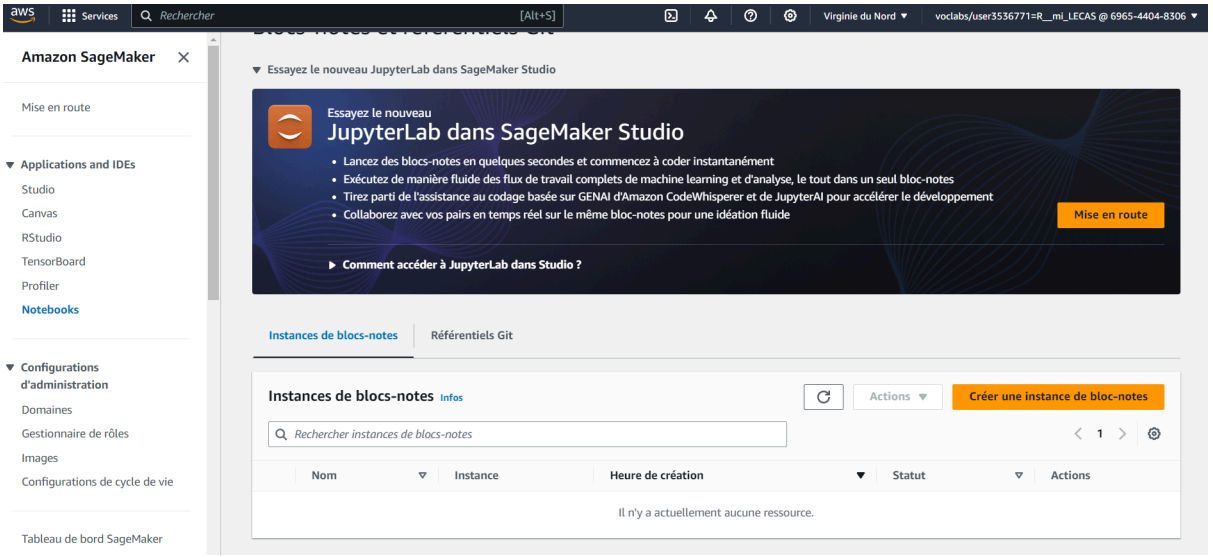


Section 3 key takeaways

- The first step in evaluating data is to make sure that it's in the right format
- pandas is a popular Python library for working with data
- Use descriptive statistics to learn about the dataset
- Create visualizations with pandas to examine the dataset in more detail

LABS 3.1





Paramètres d'instances de blocs-notes

Nom de l'instance de bloc-notes

Mynotebook

Maximum de 63 caractères alphanumériques. Puis incluent les traits d'union (-), mais pas d'espaces. Doit être unique au sein de votre compte dans une région AWS.

Type d'instance de bloc-notes

ml.m4.xlarge

Identifiant de plateforme **En savoir plus**

Amazon Linux 2, Jupyter Lab 1

▼ **Configuration supplémentaire**

Configuration du cycle de vie - *facultatif*

Personnalisez votre environnement de bloc-notes avec des plug-ins et des scripts par défaut.

ml-pipeline-c135731a3455556l7704372t1w696544048306

Taille de volume en Go - *facultatif*

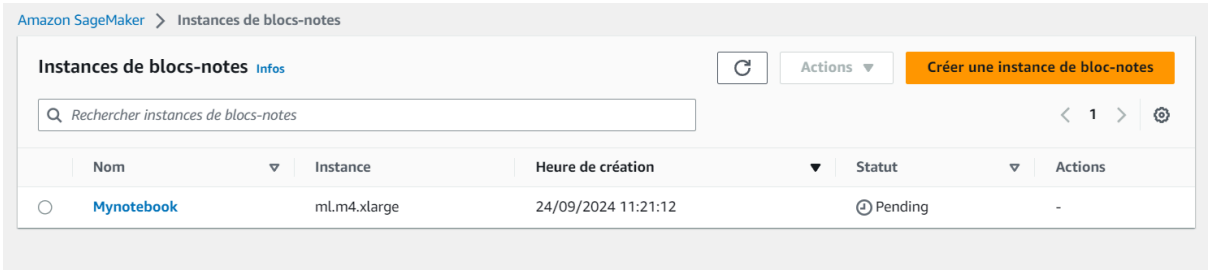
Saisissez la taille du volume de l'instance de bloc-notes en Go. La taille du volume doit être comprise entre 5 Go et 16 384 Go (16 To).

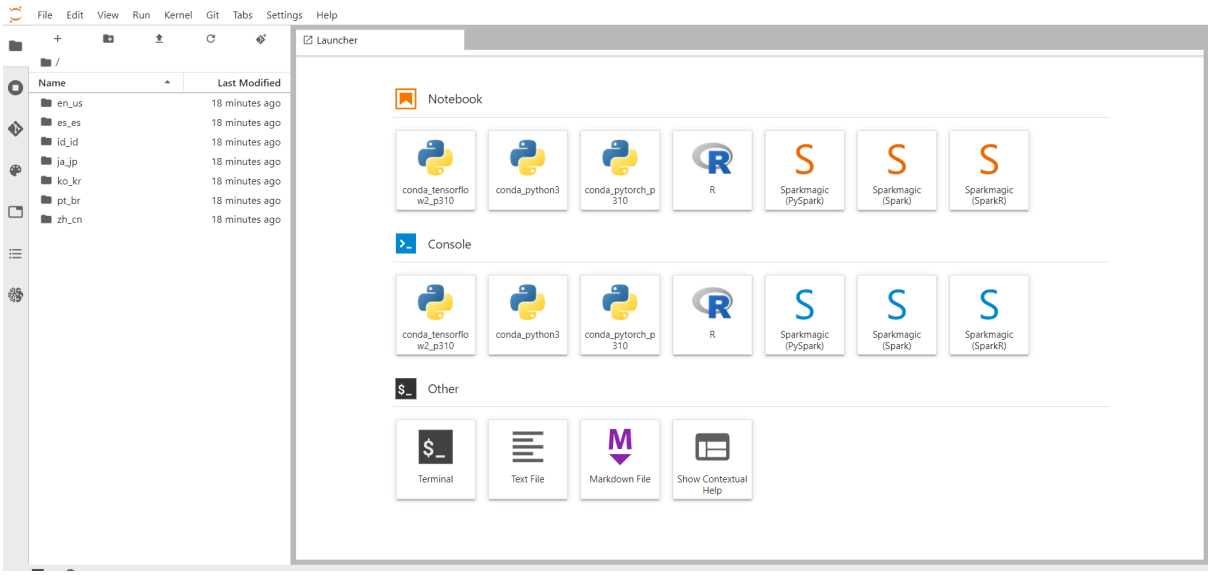
5

Version IMDS minimale - *facultatif*

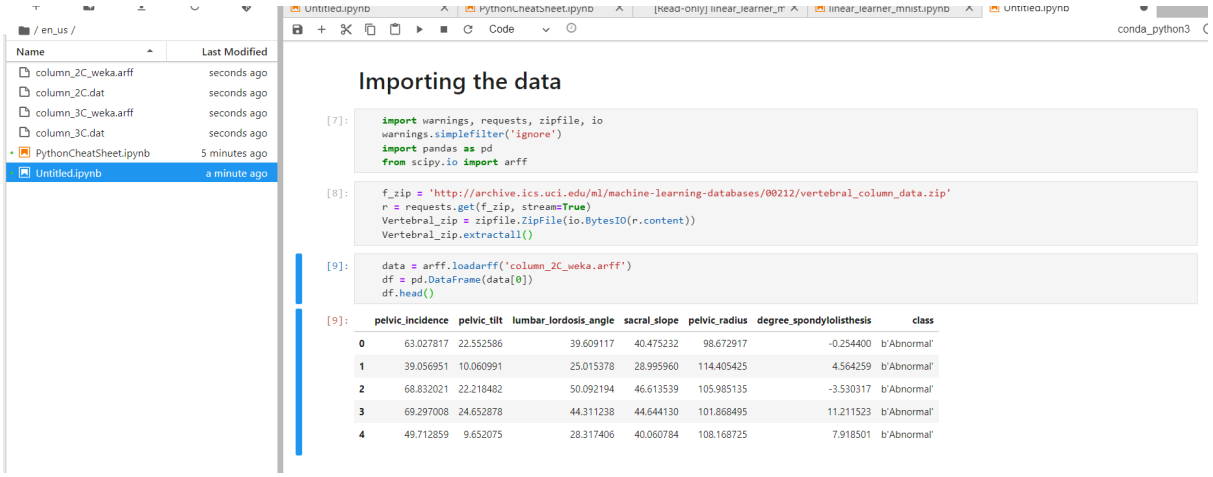
Sélectionnez la version IMDS minimale qui peut être utilisée dans votre instance

2


Amazon SageMaker > Instances de blocs-notes
Instances de blocs-notes Infos
Actions
Créer une instance de bloc-notes
Rechercher instances de blocs-notes
1
Nom
Instance
Heure de création
Statut
Actions
Mynotebook
ml.m4.xlarge
24/09/2024 11:21:12
Pending
-


File Edit View Run Kernel Git Tabs Settings Help
Launcher
Name
Last Modified
en_us 18 minutes ago
es_es 18 minutes ago
id_id 18 minutes ago
ja_jp 18 minutes ago
ko_kr 18 minutes ago
pt_br 18 minutes ago
zh_cn 18 minutes ago
Notebook
conda_tensorflow2_p310
conda_python3
conda_pytorch_p310
R
Sparkmagic (PySpark)
Sparkmagic (Spark)
Sparkmagic (SparkR)
Console
Other
Terminal
Text File
Markdown File
Show Contextual Help


/ en_us /
Name
Last Modified
PythonCheatSheet.ipynb 18 minutes ago
Markdown
Python Cheatsheet
Contents
1. Syntax and whitespace
2. Comments
3. Numbers and operations
4. String manipulation
5. Lists, tuples, and dictionaries
6. JSON
7. Loops
8. File handling
9. Functions
10. Working with datetime
11. NumPy
12. Pandas
To run a cell, press Shift+Enter or click Run at the top of the page.
1. Syntax and whitespace
Python uses indented space to indicate the level of statements. The following cell is an example where 'if' and 'else' are in same level, while 'print' is se
space to a different level. Spacing should be the same for items that are on the same level.
[ ]: student_number = input("Enter your student number:")



LABS 3.2

Challenge Task: Try updating the code in the previous cell to view the statistics of other features. Which features have outliers that you might want to examine?

```
[11]: df[['pelvic_incidence', 'pelvic_tilt', 'lumbar_lordosis_angle', 'sacral_slope', 'pelvic_radius', 'degree_spondylolisthesis']].describe()
```

[11]:

	pelvic_incidence	pelvic_tilt	lumbar_lordosis_angle	sacral_slope	pelvic_radius	degree_spondylolisthesis
count	310.000000	310.000000	310.000000	310.000000	310.000000	310.000000
mean	60.496653	17.542822	51.930930	42.953831	117.920655	26.296694
std	17.236520	10.008330	18.554064	13.423102	13.317377	37.559027
min	26.147921	-6.554948	14.000000	13.366931	70.082575	-11.058179
25%	46.430294	10.667069	37.000000	33.347122	110.709196	1.603727
50%	58.691038	16.357689	49.562398	42.404912	118.268178	11.767934
75%	72.877696	22.120395	63.000000	52.695888	125.467674	41.287352
max	129.834041	49.431864	125.742385	121.429566	163.071041	418.543082

Question: Are there any features that aren't well-distributed? Are there any features with outliers that you want to look at? Does it look like there might be any correlations between features?

Le pelvic tilt avec une valeur négative et le degree_spondylolisthesis a une large plage de valeur, cela montre que certaines caractéristiques ne sont pas bien réparties.

Dans un contexte de classification de patients comme normal ou anormal, les valeurs extrêmes observées dans pelvic tilt et degree of spondylolisthesis sont des points d'attention à examiner plus en détail. Ces valeurs pourraient influencer la précision du modèle de machine learning.

Le pelvic incidence, sacral slope, et lumbar lordosis angle sont tous liés à la géométrie du bassin et de la colonne vertébrale, et sont susceptibles d'être corrélés.

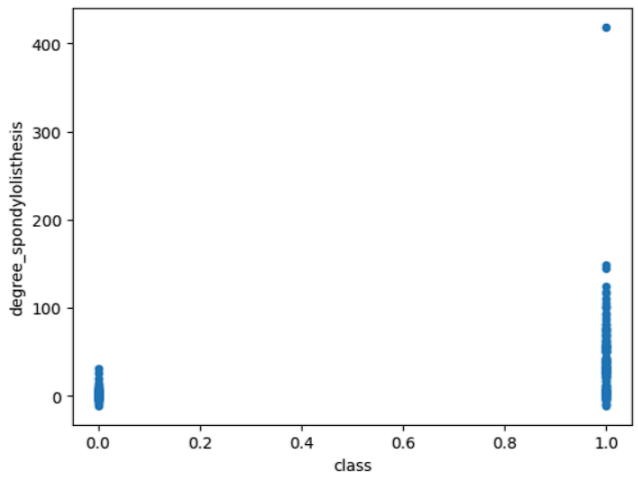
Do any of the visualizations stand out?

Pelvic_tilt et degree_spondylolisthesis ont un pic à 0 comparé aux autres, donc les données sont pas bonnes

Challenge Task: By using the previous cells, determine how the values of other features correspond against the target.

```
[23]: df.plot.scatter(y='degree_spondylolisthesis',x='class')
```

```
[23]: <Axes: xlabel='class', ylabel='degree_spondylolisthesis'>
```

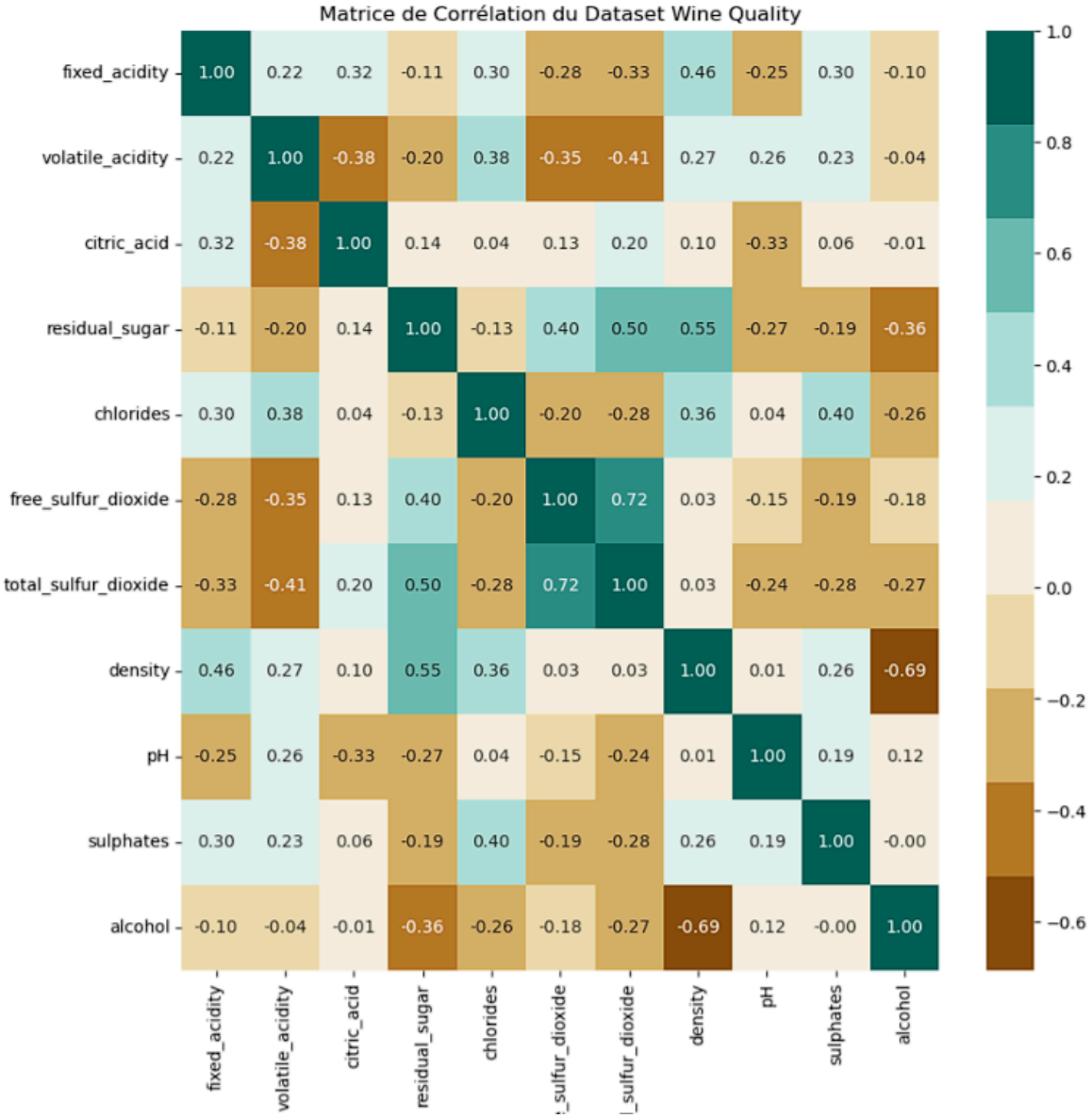


Il y a beaucoup de valeur dans vers 1.0, donc cela confirmerait que cette caractéristique est un indicateur fort pour la détection d'anomalies.

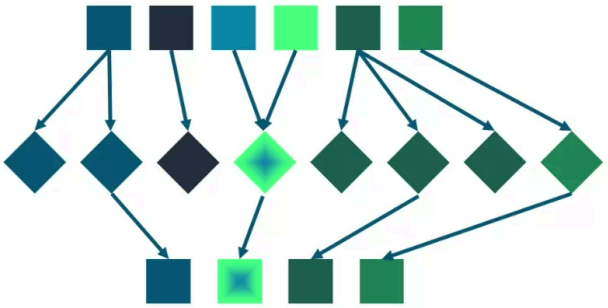
```
from ucimlrepo import fetch_ucirepo
import seaborn as sns
import matplotlib.pyplot as plt

wine_quality = fetch_ucirepo(id=186)  # ID 186 correspond au dataset Wine Quality

X = wine_quality.data.features
corr_matrix = X.corr()
fig, ax = plt.subplots(figsize=(10, 10))  # Définir la taille de la figure
colormap = sns.color_palette("BrBG", 10)
sns.heatmap(corr_matrix, cmap=colormap, annot=True, fmt=".2f", ax=ax)
plt.title("Matrice de Corrélation du Dataset Wine Quality")
plt.show()
```

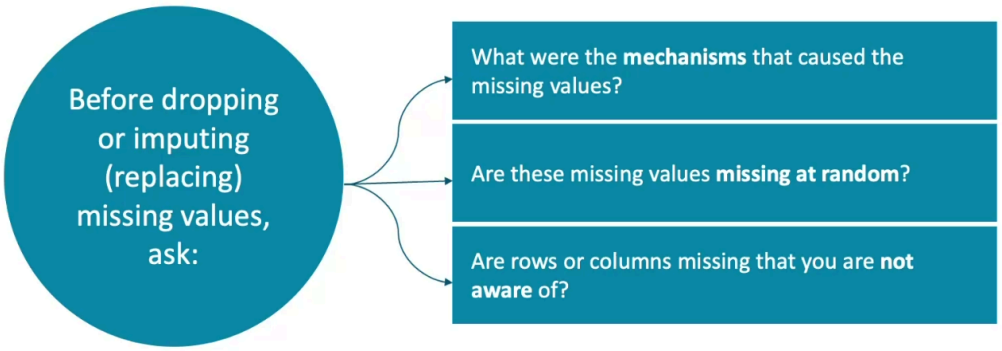

Matrice de Corrélation du Dataset Wine Quality
fixed_acidity 1.00 0.22 0.32 -0.11 0.30 -0.28 -0.33 0.46 -0.25 0.30 -0.10
volatile_acidity 0.22 1.00 -0.38 -0.20 0.38 -0.35 -0.41 0.27 0.26 0.23 -0.04
citric_acid 0.32 -0.38 1.00 0.14 0.04 0.13 0.20 0.10 -0.33 0.06 -0.01
residual_sugar -0.11 -0.20 0.14 1.00 -0.13 0.40 0.50 0.55 -0.27 -0.19 -0.36
chlorides 0.30 0.38 0.04 -0.13 1.00 -0.20 -0.28 0.36 0.04 0.40 -0.26
free_sulfur_dioxide -0.28 -0.35 0.13 0.40 -0.20 1.00 0.72 0.03 -0.15 -0.19 -0.18
total_sulfur_dioxide -0.33 -0.41 0.20 0.50 -0.28 0.72 1.00 0.03 -0.24 -0.28 -0.27
density 0.46 0.27 0.10 0.55 0.36 0.03 0.03 1.00 0.01 0.26 -0.69
pH -0.25 0.26 -0.33 -0.27 0.04 -0.15 -0.24 0.01 1.00 0.19 0.12
sulphates 0.30 0.23 0.06 -0.19 0.40 -0.19 -0.28 0.26 0.19 1.00 -0.00
alcohol -0.10 -0.04 -0.01 -0.36 -0.26 -0.18 -0.27 -0.69 0.12 -0.00 1.00
fixed_acidity
volatile_acidity
citric_acid
residual_sugar
chlorides
density
pH
sulphates
alcohol
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6

Feature extraction



- Invalid values
- Wrong formats
- Misspelling
- Duplicates
- Consistency
- Rescale
- Encode categories
- Remove outliers
- Reassign outliers
- Bucketing
- Decomposition
- Aggregation
- Combination
- Transformation
- Normalization
- Dimensionality reduction

Plan for missing values



LABS 3.3:

```
20]: sv"
ng','normalized-losses','fuel-type','aspiration','num-of-doors','body-style','drive-wheels','engine-location','wheel-base',
                'length','width','height','curb-weight','engine-type','num-of-cylinders','engine-size',
                'fuel-system','bore','stroke','compression-ratio','horsepower','peak-rpm','city-mpg','highway-mpg','price','other']

v(url,sep=',',names = col_names ,na_values="?",  header=None)
```

```
df_car['other'] = 'vroum-vroum'
```

ıeel-base	length	width	height	curb-weight	engine-type	num-of-cylinders	engine-size	fuel-system	bore	stroke	compression-ratio	horsepower	peak-rpm	city-mpg	highway-mpg	price	other
88.6	168.8	64.1	48.8	2548	dohc	four	130	mpfi	3.47	2.68	9.0	111.0	5000.0	21	27	13495.0	vroum-vroum
88.6	168.8	64.1	48.8	2548	dohc	four	130	mpfi	3.47	2.68	9.0	111.0	5000.0	21	27	16500.0	vroum-vroum
94.5	171.2	65.5	52.4	2823	ohcv	six	152	mpfi	2.68	3.47	9.0	154.0	5000.0	19	26	16500.0	vroum-vroum
99.8	176.6	66.2	54.3	2337	ohc	four	109	mpfi	3.19	3.40	10.0	102.0	5500.0	24	30	13950.0	vroum-vroum
99.4	176.6	66.4	54.3	2824	ohc	five	136	mpfi	3.19	3.40	8.0	115.0	5500.0	18	22	17450.0	vroum-vroum
99.8	177.3	66.3	53.1	2507	ohc	five	136	mpfi	3.19	3.40	8.5	110.0	5500.0	19	25	15250.0	vroum-vroum

Target -> Feature qu'on a créer et qu'on souhaite tester

LABS: 3.5
Challenge task: Update the previous code to send the second row of the dataset. Are those predictions correct? Try this task with a few other rows.

The **iloc** function takes parameters of [*rows,cols*]

To only get the first row, use `0:1`. If you want to get row 2, you could use `1:2`.

To get all columns *except* the first column (*col 0*), use `1:`

[52]:
```
row = test.iloc[1:2, 1:]
row.head()
```

[52]:

	pelvic_incidence	pelvic_tilt	lumbar_lordosis_angle	sacral_slope	pelvic_radius	degree_spondylolisthesis
230	65.611802	23.137919	62.582179	42.473883	124.128001	-4.083298

You can convert this to a comma-separated values (CSV) file, and store it in a string buffer.

[53]:
```
batch_X_csv_buffer = io.StringIO()
row.to_csv(batch_X_csv_buffer, header=False, index=False)
test_row = batch_X_csv_buffer.getvalue()
print(test_row)
```

```
65.61180231,23.13791922,62.58217893,42.47388309,124.1280012,-4.083298414
```

Now, you can use the data to perform a prediction.

[54]:
```
xgb_predictor.predict(test_row)
```

[54]:
```
b'0.777283251285553'
```

The result you get isn't a *0* or a *1*. Instead, you get a *probability score*. You can apply some conditional logic to the probability score to dete
this process when you do batch predictions.

For now, compare the result with the test data.

[55]:
```
test.head(5)
```

[55]:

	class	pelvic_incidence	pelvic_tilt	lumbar_lordosis_angle	sacral_slope	pelvic_radius	degree_spondylolisthesis
136	1	88.024499	39.844669	81.774473	48.179830	116.601538	56.766083
230	0	65.611802	23.137919	62.582179	42.473883	124.128001	-4.083298
134	1	52.204693	17.212673	78.094969	34.992020	136.972517	54.939134
130	1	50.066786	9.120340	32.168463	40.946446	99.712453	26.766697
47	1	41.352504	16.577364	30.706191	24.775141	113.266675	-4.497958

Après avoir passer le treshold de 0.65 à 0.80

[63]:
```
def binary_convert(x):
    threshold = 0.80
    if x > threshold:
        return 1
    else:
        return 0

target_predicted['binary'] = target_predicted['class'].apply(binary_convert)

print(target_predicted.head(10))
test.head(10)
```

```
      class  binary
0  0.996607       1
1  0.777283       0
2  0.994641       1
3  0.993690       1
4  0.939139       1
5  0.997396       1
6  0.991977       1
7  0.987518       1
8  0.993334       1
9  0.682776       0
```

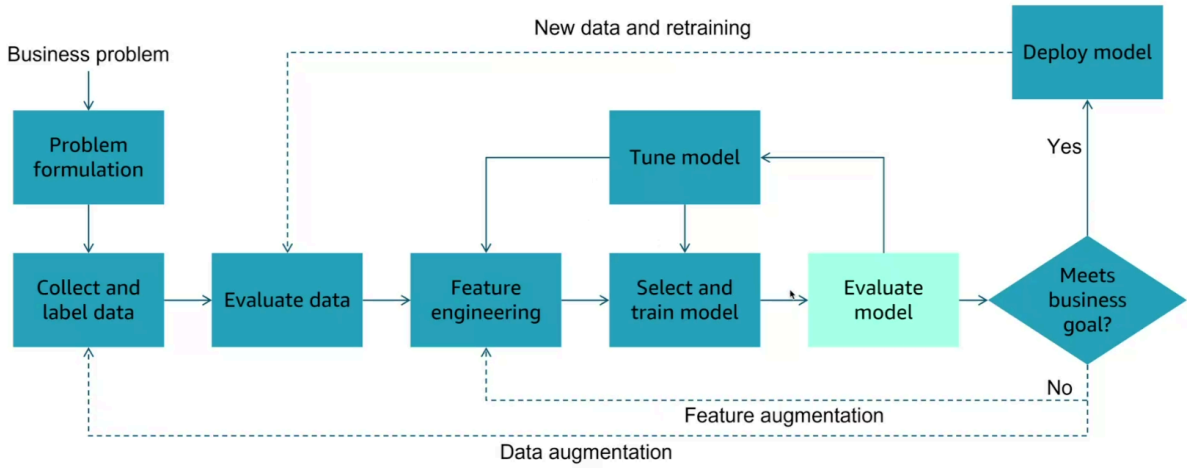
[63]:

	class	pelvic_incidence	pelvic_tilt	lumbar_lordosis_angle	sacral_slope	pelvic_radius	degree_spondylolisthesis
136	1	88.024499	39.844669	81.774473	48.179830	116.601538	56.766083
230	0	65.611802	23.137919	62.582179	42.473883	124.128001	-4.083298
134	1	52.204693	17.212673	78.094969	34.992020	136.972517	54.939134
130	1	50.066786	9.120340	32.168463	40.946446	99.712453	26.766697
47	1	41.352504	16.577364	30.706191	24.775141	113.266675	-4.497958
135	1	77.121344	30.349874	77.481083	46.771470	110.611148	82.093607
100	1	84.585607	30.361685	65.479486	54.223922	108.010218	25.118478
89	1	71.186811	23.896201	43.696665	47.290610	119.864938	27.283985
297	0	45.575482	18.759135	33.774143	26.816347	116.797007	3.131910
4	1	49.712859	9.652075	28.317406	40.060784	108.168725	7.918501

Note: The *threshold* in the **binary convert** function is set to .65.

- Can use two options for deployment
  - Amazon SageMaker hosting
  - Batch transform
- Deploy only after you have tested your model
  - Goal is to generate predictions for client applications
- Create an endpoint
  - Single-model endpoint for simple use cases
  - Multi-model endpoint to support multiple use cases

Machine learning pipeline 8



Confusion matrix terminology

		Actual	
Predicted		Cat	Not cat
	Cat	TP	FP
	Not cat	FN	TN

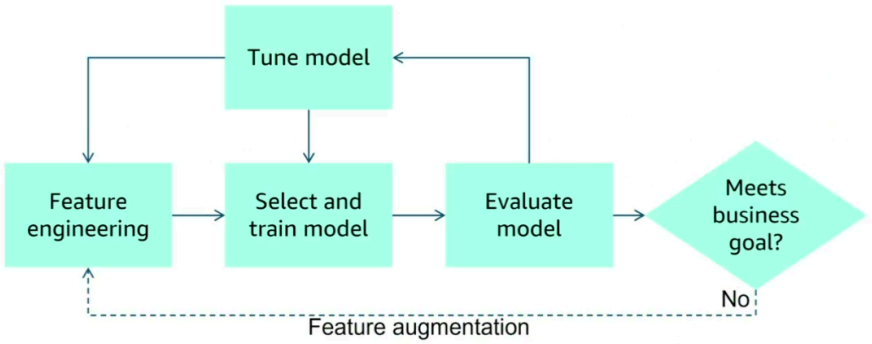
True positive (TP)
Predicted a cat and it was a cat

False positive (FP)
Predicted *not* a cat and it was a cat

False negative (FN)
Predicted a cat and it was *not* a cat

True negative (TN)
Predicted *not* a cat and it was *not* a cat

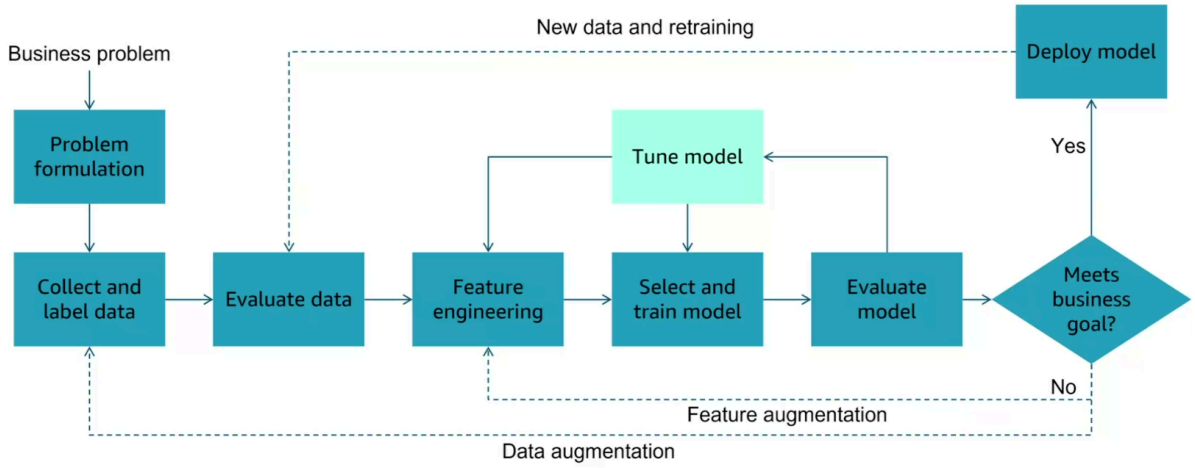
ML tuning process



Section 7 key takeaways

- Several ways to validate the model
  - Hold-out
  - K-fold cross validation
- Two types of model evaluation
  - Classification
    - Confusion matrix
    - AUC-ROC
  - Regression testing
    - Mean squared

Machine learning pipeline 9



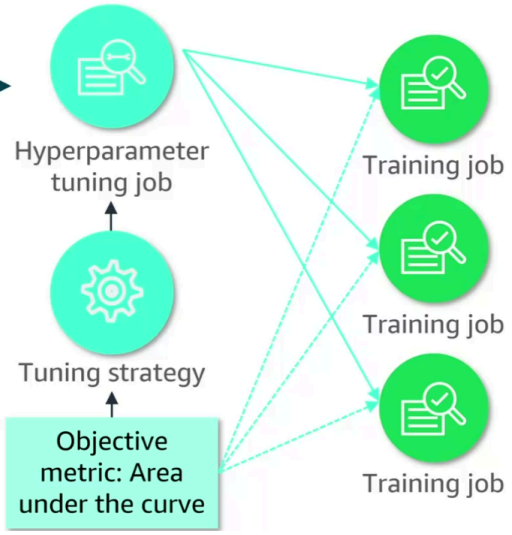
Hyperparameter categories

Model	Optimizer	Data
Help define the model	How the model learns patterns on data	Define attributes of the data itself
Filter size, pooling, stride, padding	Gradient descent, stochastic gradient descent	Useful for small or homogenous datasets

Automated hyperparameter tuning

Clients
(console, notebook, IDEs, AWS CLI)

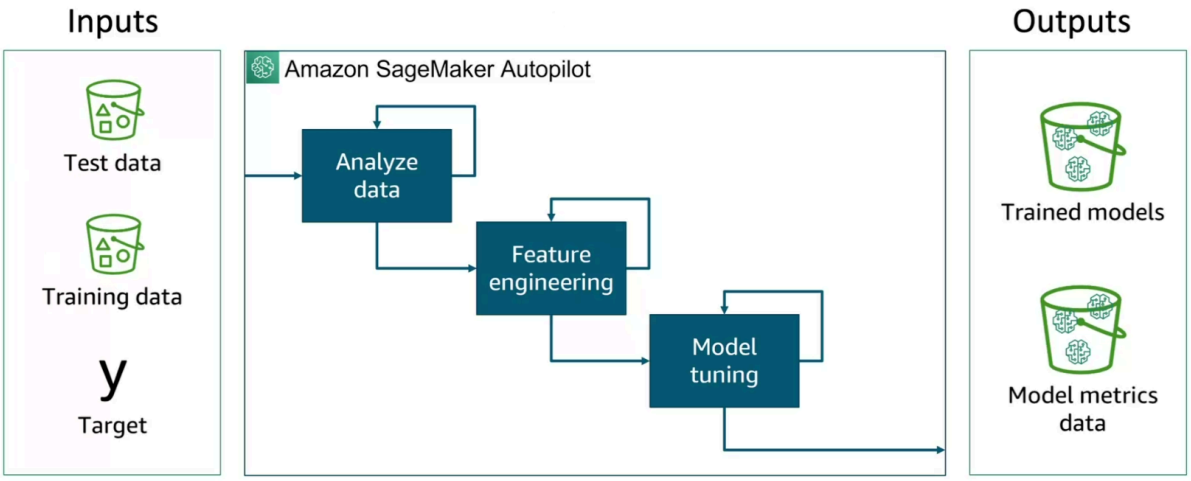
Model Name	Objective Metric	Eta	Max_depth	…
Model3	0.8	0.7	6	…
Model1	0.75	0.09	5	…
…	…	…	…	…



Tuning best practices

- Don’t adjust every hyperparameter
- Limit your range of values to what’s most effective
- Run one training job at a time instead of multiple jobs in parallel
- In distributed training jobs, make sure that the objective metric that you want is the one that is reported back
- With Amazon SageMaker, convert log-scaled hyperparameters to linear-scaled when possible

Amazon SageMaker Autopilot



Section 8 key takeaways

- Model tuning helps find the best solution
- Hyperparameters
  - Model
  - Optimizer
  - Data
  - Tuning
- Use Amazon SageMaker to help tune hyperparameters
- Use Autopilot for faster development

Module summary

In this module, you learned how to:

- Formulate a problem from a business request
- Obtain and secure data for machine learning (ML)
- Build a Jupyter Notebook by using Amazon SageMaker
- Outline the process for evaluating data
- Explain why data must be preprocessed
- Use open source tools to examine and preprocess data
- Use Amazon SageMaker to train and host an ML model
- Use cross-validation to test the performance of an ML model
- Use a hosted model for inference
- Create an Amazon SageMaker hyperparameter tuning job to optimize a model's effectiveness