Introduction to Machine Learning (ML)

source: https://github.com/alexeygrigorev/mlbookcamp-code/tree/master/course-zoomcamp/cohorts/2022

Imagine a scenario where a customer wants to sell their car on a car classified website. This website requires to add description of your car type, model, mileage, and price etc. However, they can't decide on the price of the car. The reason being that the price should not be too low or high. Therefore, determining the optimal price of the car is challenging. One solution would be to predict the price manually, where the customer needs to do their own research on several similar car classified websites and spend time to search for similar cars and their prices. But the question is: How can we (as the owner of the car classified company) suggest prices to our customers? This can be achieved with the help of machine learning (ML).

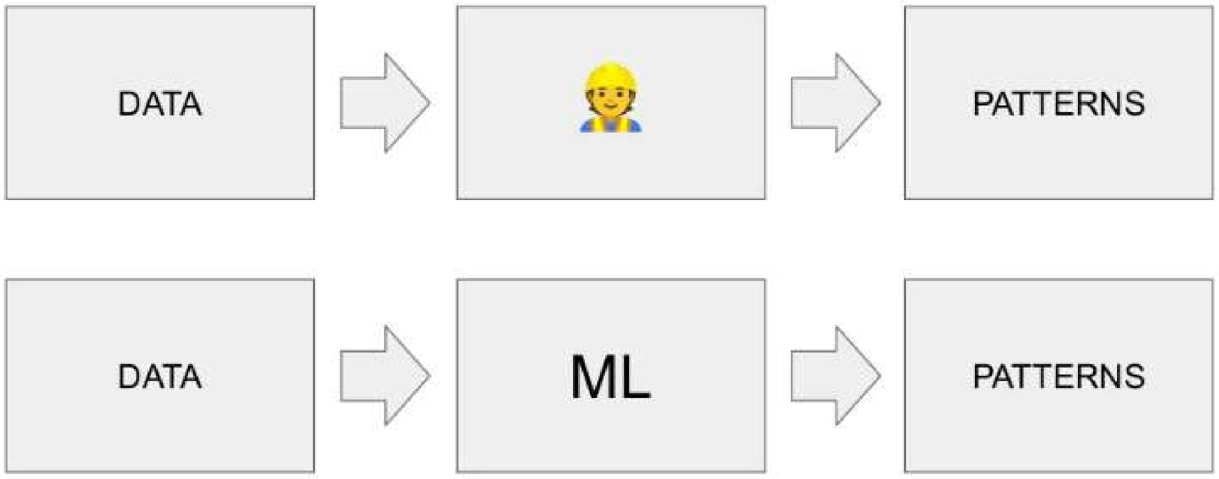
Before applying ML to predict the price of the car, we need to gather all the information provided by the user. For example, the following features/characteristics of the cars that are provided by the customers:

1. Year: the manufature year of the car (i.e.,older the car, lower is the price)
2. Price: estimated price of the car provided by the customers
3. Make: manufacturer of the car i.e. BMW, VW
4. Mileage: how many Kilometers, car has driven

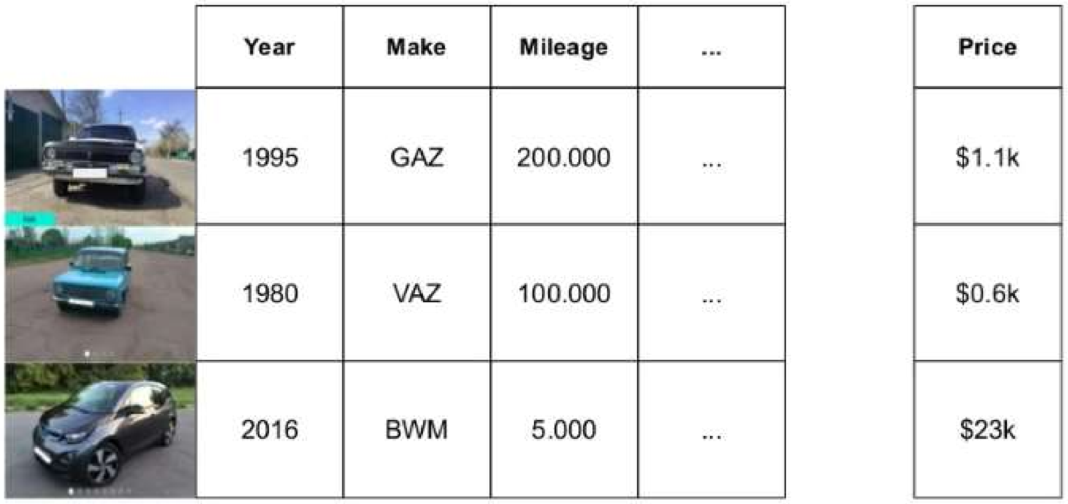
Thus, in ML world data is broadly of two types: features (all info about the data at hand), and target (this is what is going to be predicted).

Generally, a human expert can determine the price of the car by using the above described information or features. The question is, how an expert decides on the price: they usually have gathered information from other dealerships, learnt about the general price of the cars based on their available features. Therefore, an expert has spent enough time to learn and extract some patterns about the price based on the their experience.

If an expert can determine the price of the car so can a ML model! If a dataset is available to us with features and price, we can feed a ML model with datasets and the ML model will learn from the provided patterns.



If an expert can, so can a model!



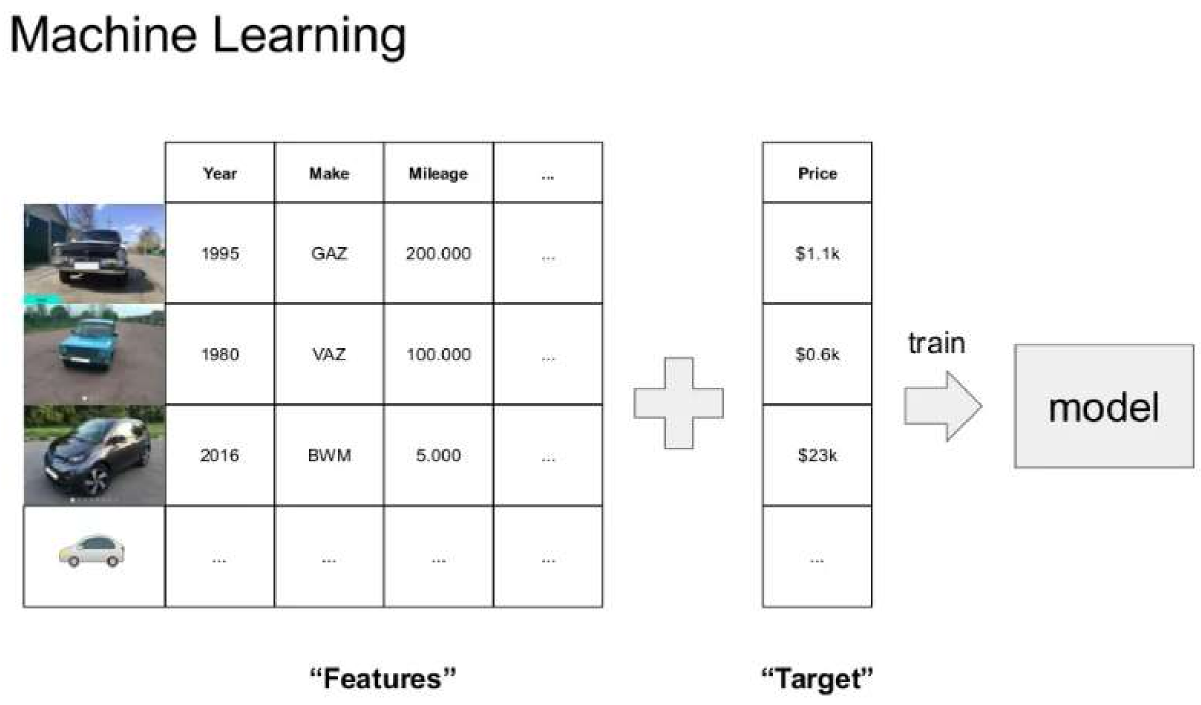
Year	Make	Mileage	...	Price
1995	GAZ	200.000	...	$1.1k
1980	VAZ	100.000	...	$0.6k
2016	BWM	5.000	...	$23k

“Features”
what we know about cars

“Target”
what we want to predict

In the current car price prediction example, we will:

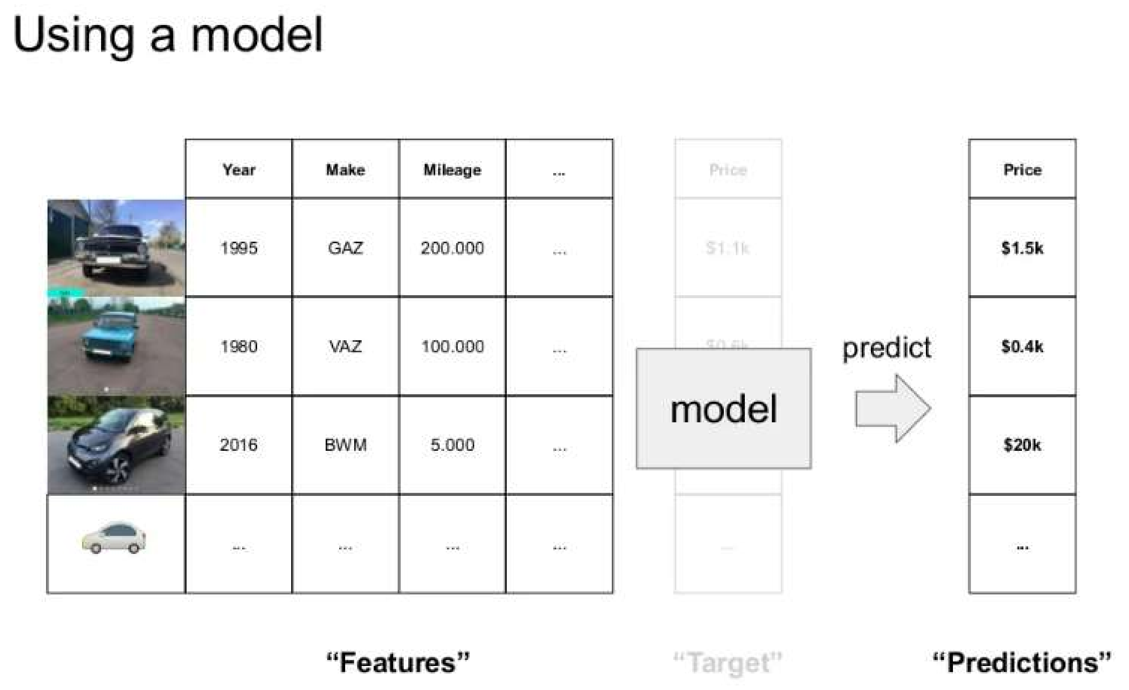
- provide all the available information i.e., features and the target to the ML model and train it (An ML model encapsulate all the patterns that it learns from the data)



	Year	Make	Mileage	...
	1995	GAZ	200.000	...
	1980	VAZ	100.000	...
	2016	BWM	5.000	...
	...	...	...	...

Price
$1.1k
$0.6k
$23k
...

- then we use the trained model and apply it to the features (EXCEPT THE target) to make price prediction
- the model will not provide the correct prediction but it will give an average



	Year	Make	Mileage	...
	1995	GAZ	200.000	...
	1980	VAZ	100.000	...
	2016	BWM	5.000	...
	...	...	...	...

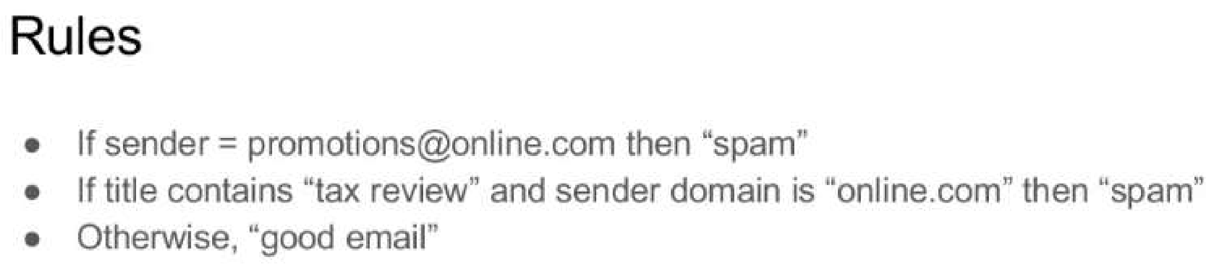
Price
$1.5k
$0.4k
$20k
...

Thus, ML is the process of extracting patterns from the data. We can summarize the ML process in the following two steps:

- Step 1: Feed in the features/characteristics and the target data to the ML model which will output in a trained ML model
- Step 2: apply the trained ML model to the features EXCEPT THE TARGET which will output in a predicted target feature (e.g., car price in this case)

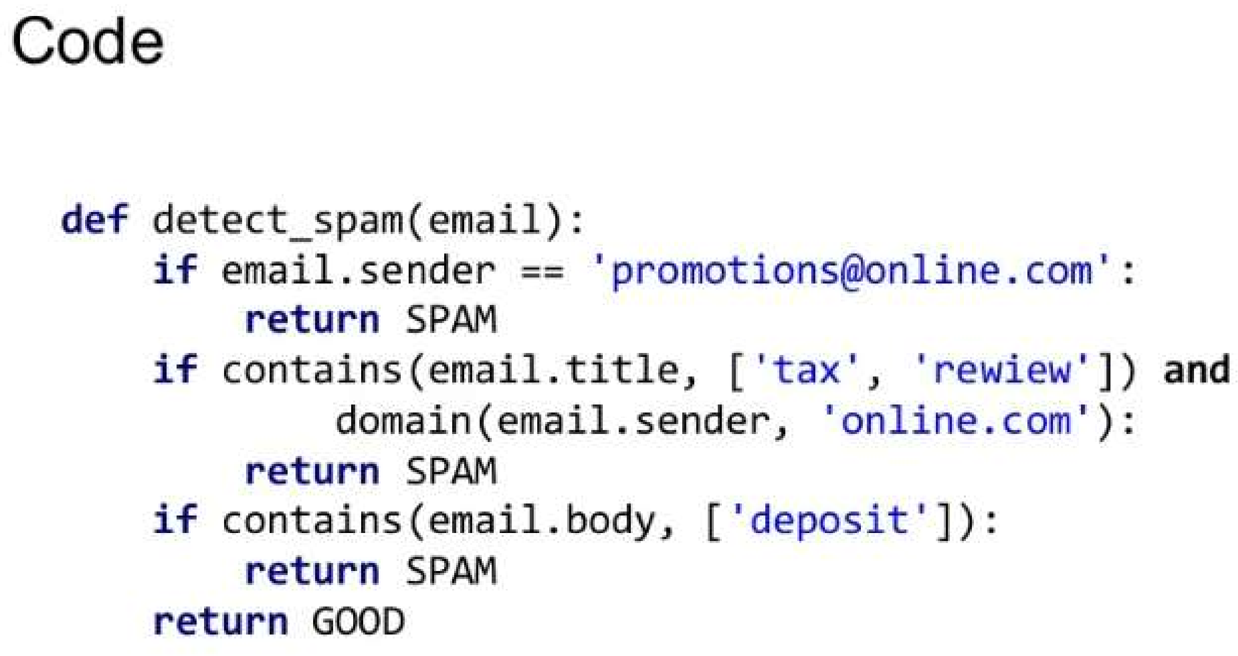
Machine Learning (ML) vs Rule-based Systems

- Imagine a scenario in which user complaints about getting many spam emails
- As a data scientist, we would like to build a classifier. This classifier will classifies the incoming are spams or not
- The goal of the classifier will be to detect a pattern and find out what features make an email a spam message
- Old rule-based system will use few rules and write few lines of code based on them to return output if the email is good or spam:



Rules

- If sender = promotions@online.com then "spam"
- If title contains "tax review" and sender domain is "online.com" then "spam"
- Otherwise, "good email"

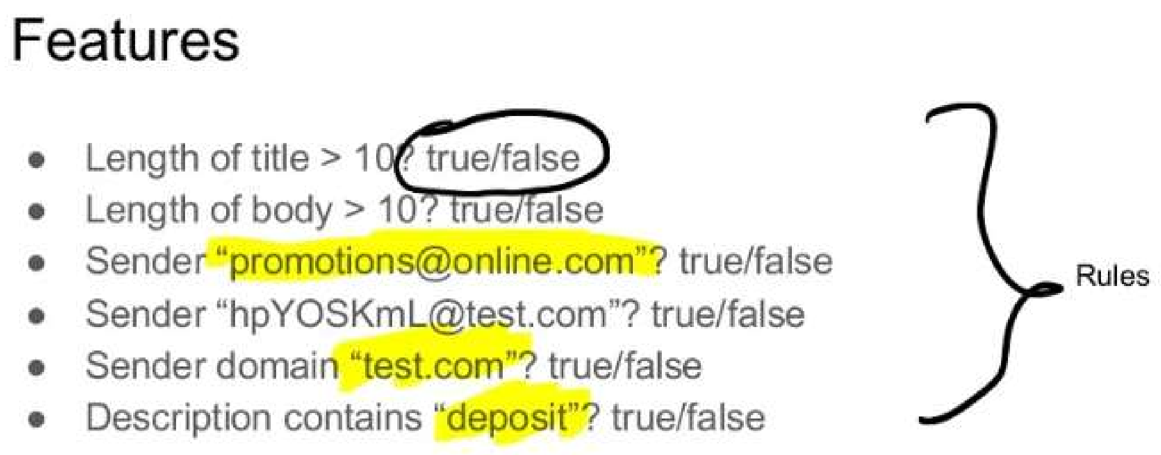


Code

```
def detect_spam(email):
    if email.sender == 'promotions@online.com':
        return SPAM
    if contains(email.title, ['tax', 'rewiew']) and
            domain(email.sender, 'online.com'):
        return SPAM
    if contains(email.body, ['deposit']):
        return SPAM
    return GOOD
```

- The problem with such system will start to appear when there are same keywords for spam and good emails. The spam constantly keep changing and it will end up in many lines of code which will be difficult to maintain
- The solution is to use ML! We can first use all types of emails as our data. Then, define and calculate specific features of the emails to feed to the ML model for training. At the end we can use the trained ML model as a classifier to detect good vs spam emails. The features will be based on the old rule based system. Therefore, we can start with the rules and then use these

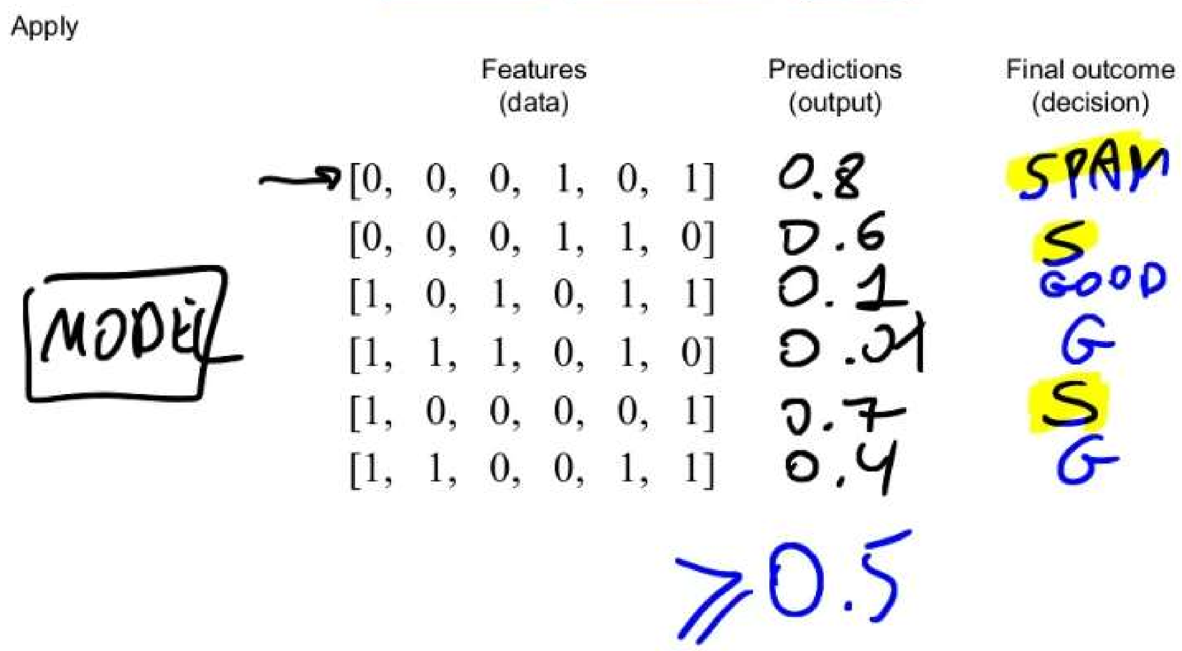
rules as features



Features

- Length of title > 10? true/false
- Length of body > 10? true/false
- Sender "promotions@online.com"? true/false
- Sender "hpYOSKmL@test.com"? true/false
- Sender domain "test.com"? true/false
- Description contains "deposit"? true/false

- For example, in an email, we can use the features to identify true (1) or false (0) and feed the true and false features to the ML model
- Based on the true/false predictions, we can decide if the prediction is >50% then it is spam and if it's < 50% then identify as a good email



Apply

Features (data)	Predictions (output)	Final outcome (decision)
[0, 0, 0, 1, 0, 1]	0.8	SPAM
[0, 0, 0, 1, 1, 0]	0.6	S
[1, 0, 1, 0, 1, 1]	0.1	GOOD
[1, 1, 1, 0, 1, 0]	0.21	G
[1, 0, 0, 0, 0, 1]	0.7	S
[1, 1, 0, 0, 1, 1]	0.4	G

MODEL
$\geqslant 0.5$

To summarize, in a rule-based system we will write some code and provide data to the software and predictions will be the outcome (Data + code -> fed into the software -> Outcome (e.g., spam or good email)). However this system becomes difficult to maintain after a certain time. While ML based system outcome is the input to the ML model. We know if an email is spam and feed it to the ML system. Then output is the trained model which can be used in those cases when we do not know the outcome or to make the prediction if the email is spam or not (Data+ trained ML model -> predictions).