Module 21
Machine Learning – Part 1 (Amazon Translate)

This and the next few modules cover a few of the AWS Machine Learning (ML) services (Figure 1).

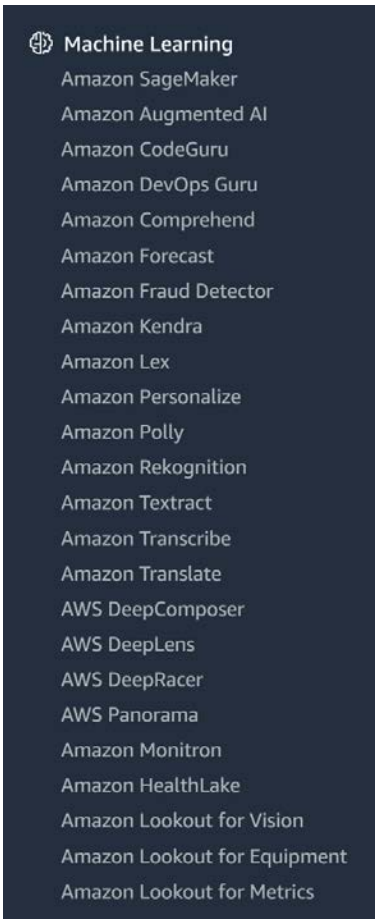


Figure 1: AWS machine learning services (Source: the AWS console)

It is OK if you haven't taken an ML course before because all the services listed in Figure 1 are exposed as APIs where you do not need to provide any algorithms-related settings or parameters. This is unlike some other ML libraries like *TensorFlow* or *PyTorch* where the developer needs to know Machine Learning to effectively use them. This is deliberately designed that way by AWS to allow application developers to incorporate ML capabilities in their applications without necessarily having expertise in the field. Figure 2 shows how AWS ML services compare in terms of ease of use.

	Need to know all the details of the internals of the algorithms	Need to know how the algorithms work and the meaning of the different parameters that affect learning.
Write ML algorithms from scratch	Yes	Yes
Use ML libraries such as TensorFlow and PyTorch	No	Yes
AWS Machine Learning services	No	No

Figure 2: Developers can use AWS ML services without having expert knowledge of ML algorithms.

For those who did not take an ML course I will briefly describe what ML algorithms try to achieve without going much in details. Machine Learning is suitable to tackle problems where it is not clearly understood how the output of an event is affected by the attributes of its input. As a concrete example, it is known that some people cheat on their taxes. But there is no formula or a definitive set of rules that, if applied, finds if someone is likely to cheat or not. This is an example where the output (cheat on tax or not) cannot easily be determined from the input (the attributes of the tax filer: age, gender, ethnicity, education, address, income, etc.).

Another ML example is to say differentiate among the different species of the cats' family (Figure 3). Although it is easy for our eyes to distinguish anatomical characteristics, it is not clear how one can encode these differences in a computer program. It is another example where the output (type of cat) cannot easily be determined from the input (pixels that make up the picture of the animal).

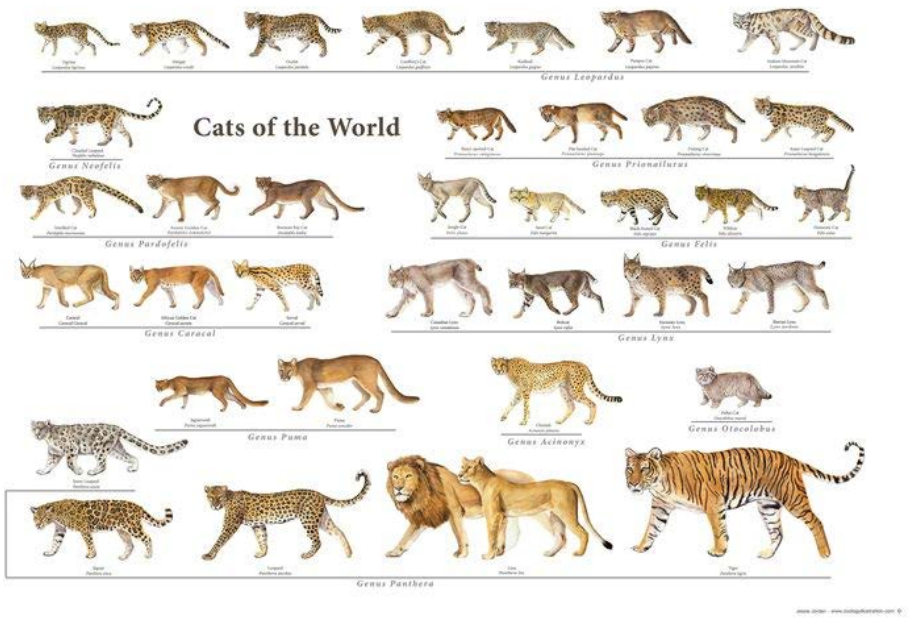


Figure 3: Various animals of the cats' family.

In general, the learning happens as shown in Figure 4 (using an example of a learning system that learns to classify a mortgage borrower as likely to pay off a loan in full or default on the loan): In the learning phase, historical data of mortgage borrowers is used to develop a model. The model is then used to answer queries.

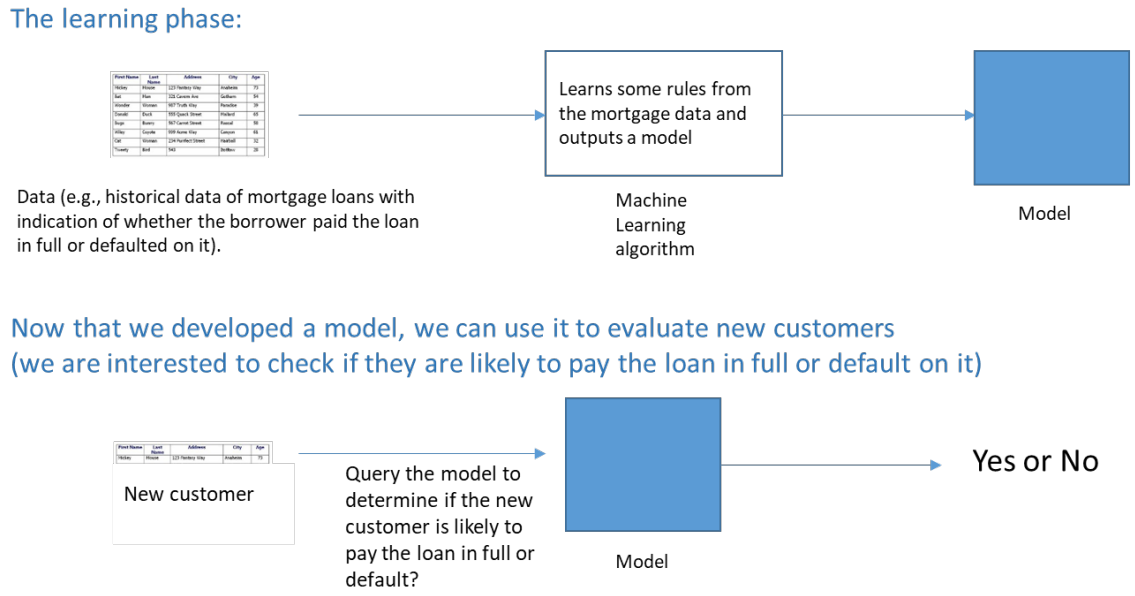


Figure 4: An example learning model.

Figure 4 shows the data cleaned in a tabular format. In practice, much data pre-processing needs to happen to get the data in a shape that can be fed into the learning algorithm. Figure 5 shows a more realistic view of the series of steps involved.

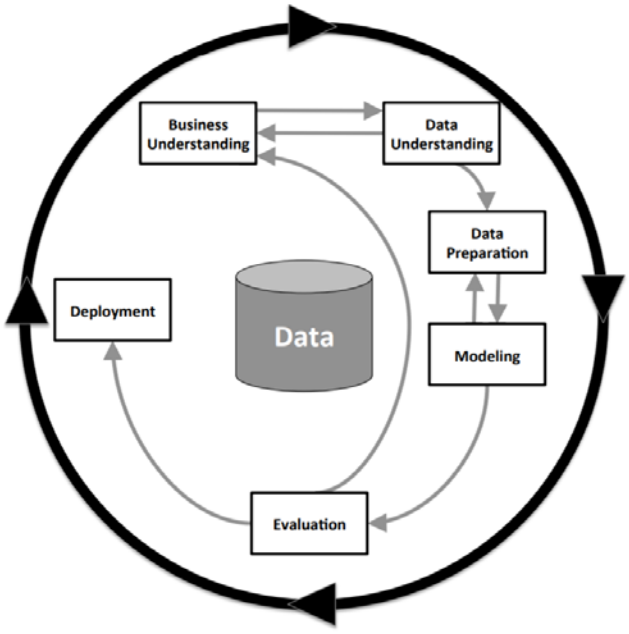


Figure 5: The various phases of developing a Machine Learning model (Source: *Fundamentals of Machine Learning for Predictive Data Analytics - Kelleher et al*).

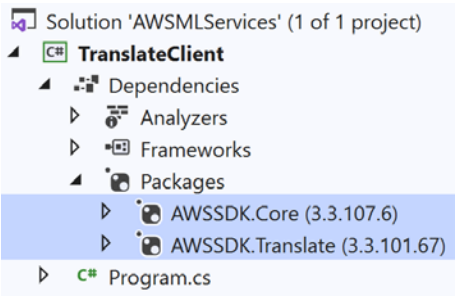
In this module and all future modules on Machine Learning services, we will not discuss the internal functioning of whatever algorithm is used by the ML service. This is beyond the scope of the class. We will just use them as APIs

and get an idea of how to incorporate them in a larger application.

Let's first look at AWS Translate, one ML service that has a wide use in many applications. This translate service is based on an artificial neural network (ANN). But as you will soon see, there is no need to know anything about the topologies of a neural network, how it works, and what parameters influence its learning.

1. Create a Visual Studio solution and name it AWSMLServices.
2. Under the above solution create a Console application project of type **Console App (.NET Core)**:

3. Give the project the name **TranslateClient**.
4. Add references to packages AWSSDK.Core and AWSSDK.Translate.



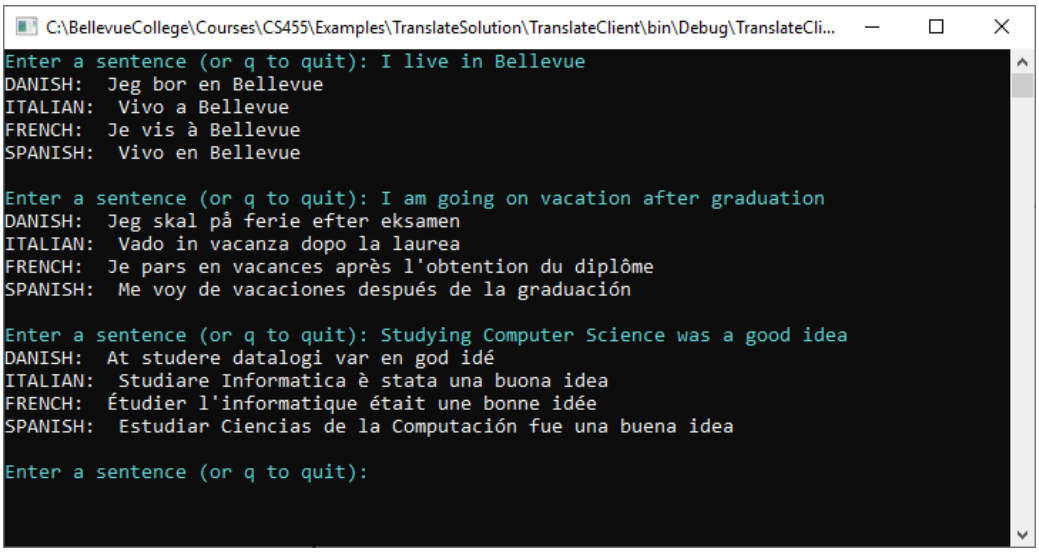
5. Add the following using statements:

```
using Amazon;
using Amazon.Runtime;
using Amazon.Runtime.CredentialManagement;
using Amazon.Translate;
using Amazon.Translate.Model;
```

6. Write a program that accepts a sentence in English, and translates it to the following languages:

   Danish
   Italian
   French
   Spanish

   The program repeatedly asks the user for a sentence and prints the translated versions of the sentence in the above 4 languages. The program should quit when the user types q or Q. Below is an example output:



IMPLEMENTATION HINTS:

Some classes you need to use: `AmazonTranslateClient`, `TranslateTextRequest`, `TranslateTextResponse`

When you create a `TranslateTextRequest` object, you need to specify a `SourceLanguageCode` and a `TargetLanguageCode`. You can get these codes from here.

7. If you know some language (other than Danish, Italian, French, or Spanish) and the language is supported, experiment translating to this language and check how accurate the translation is.

What to Submit:

There is nothing to submit for this module.