

Microsoft Azure Machine Learning

**Introduction to Machine Learning on
Azure**

What is machine learning?

- **Machine learning** is a data science technique used to extract patterns from data, allowing computers to identify related data, and forecast future outcomes, behaviors, and trends



01

**Automate the
recognition of
disease**

02

**Recommend next
best actions for
individual care
plans**

03

**Enable
personalized, real-
time banking
experiences with
chatbots**

04

**Identify the next
best action for the
customer**

05

**Capture, prioritize,
and route service
requests to the
correct employee,
and improve
response times**

Applications of Machine Learning

The data science Process



COLLECT
DATA



PREPARE
DATA



TRAIN
MODEL



EVALUATE
MODEL



DEPLOY
MODEL

After the model is deployed it is important to re-train model iteratively

Common Types of Data



Numerical data



Categorical data



Image data



Text data



Time series data



Non-numerical data forms are later encoded and transformed to numerical data and fed as input vector to train ML model

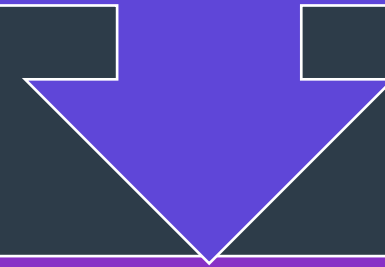
Tabular data

- The most common type of data that is encountered in ML would be Tabular data similar to that found in spreadsheets like Excel
- Row- An item or entity
- Cell- A single value
- Column- A property that the items or entities in the table can have.

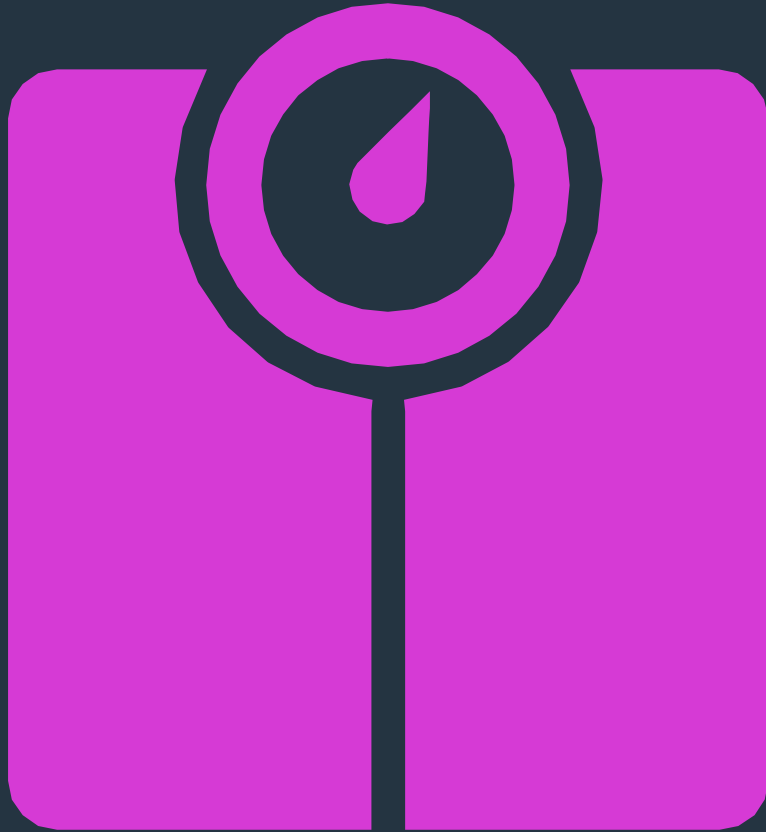


Vectors

In machine learning, the numerical representation will be in the form of an *array of numbers*—that is, a *vector*



A **vector** is simply an array of numbers, such as $(1,2,3)$ —or a nested array that contains other arrays of numbers, such as $(1,2(1,2,3))$



Scaling data

- **Scaling** data means transforming it so that the values fit within some range or scale, such as 0–100 or 0–1
- Two common approaches to scaling data include **standardization** and **normalization**.

Standardization

Standardization rescales data so that it has a mean of 0 and a standard deviation of 1

$$(x - \mu) / \sigma$$

Normalization

- **Normalization** rescales the data into the range $[0, 1]$.
- $(x - x_{min}) / (x_{max} - x_{min})$





Encoding categorical data

ML algorithms need to have data in numerical form.

When we have *categorical* data, we need to encode it in some way so that it is represented numerically.

Two common approaches for encoding categorical data: **ordinal encoding** and **one hot encoding**.

Ordinal encoding

In **ordinal encoding**, we simply convert the categorical data into integer codes ranging from 0 to (number of categories-1)

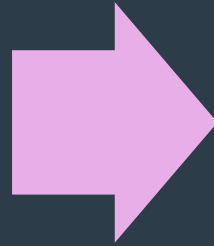
One of the potential drawbacks to this approach is that it implicitly assumes an order across the categories(i.e Values with higher number is assumed to have more importance)

One hot encoding

One-hot encoding is a very different approach. In one-hot encoding, we transform each categorical value into a column. If there are n categorical columns, n new columns are created

One drawback of one-hot encoding is that it can potentially generate a very large number of columns.

Images are another example of a data type that is commonly used as input in machine learning problems—but that isn't initially in numerical format



If you zoom in on an image far enough, you can see that it consists of small tiles, called *pixels*. The color of each pixel is represented with a set of values:

Image data

Image data

- In **grayscale images**, each pixel can be represented by a **single** number, which typically ranges from 0 to 255. This value determines how dark the pixel appears(e.g. 0 is black while 255 is white)
- In **colored images**, each pixel can be represented by a vector of **three** numbers (each ranging from 0 to 255) for the three primary color channels: red, green, and blue. These three red, green, and blue (RGB) values are used together to decide the color of that pixel.



Text data

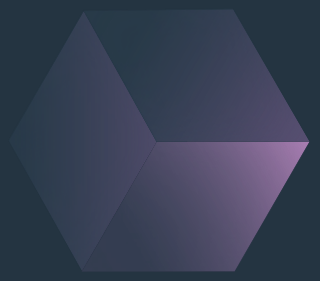
Text is another example of a data type that is initially non-numerical and that must be processed before it can be fed into a machine learning algorithm

*Text **normalization** is the process of transforming a piece of text into a canonical (official) form*

Text normalization

- *Lemmatization* is an example of normalization. A **lemma** is the dictionary form of a word and **lemmatization** is the process of reducing multiple inflections to that single dictionary form. For example, we can apply this to the is,am,are to make it to be form.

Original word	Lemmatized word
is	be
are	be
am	be



Stop Words

- **Stop words** are high-frequency words that are unnecessary (or unwanted) during the analysis. For example, when you enter a query like which cookbook has best pancake recipe

Original text	Normalized text
The quick fox.	[quick, fox]
The lazy dog.	[lazy, dog]
The rabid hare.	[rabid, hare]



Vectorization

- After we have normalized the text, we can take the next step of encoding it in a numerical form
- Typically this is done by text **vectorization**—that is, by turning a piece of text into a vector
- Common approaches include:
- TF-IDF vectorization
- Word embedding, as done with Word2Vec or GloVe

	quick	fox	lazy	dog	rabid	hare
[quick, fox]	0.32	0.23	0.0	0.0	0.0	0.0
[lazy, dog]	0.0	0.0	0.12	0.23	0.0	0.0
[rabid, hare]	0.0	0.0	0.0	0.0	0.56	0.12

TF-IDF vectorization

- The approach of TF-IDF is to give less importance to words that contain less information and are common in documents, such as "the" and "this"—and to give higher importance to words that contain relevant information and appear less frequently. Thus TF-IDF assigns weights to words that signify their relevance in the documents.




Two perspectives of ML

ML can be seen in two perspectives-The computer science perspective and statistical perspective

From computer science perspective we are using **input features** to create a **program** that can generate the desired **output**.

From statistical perspective we are trying to find a **mathematical function** that, given the values of the **independent variables** can predict the values of the **dependent variables**.



The tools for Machine Learning

A typical machine learning ecosystem is made up of three main components:

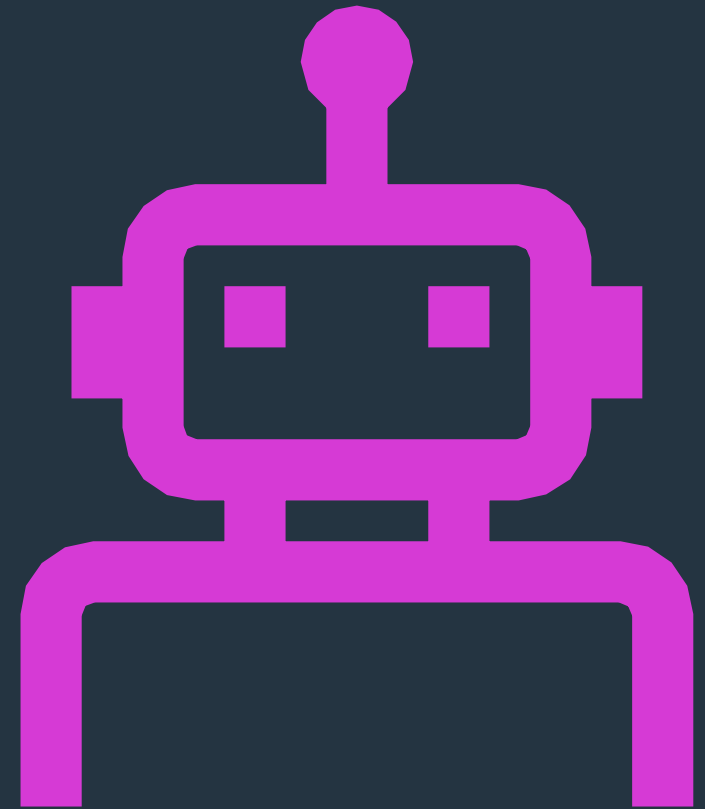
Libraries: A library is a collection of pre-written (and compiled) code that you can make use of in your own project. *NumPy* is an example of a library popularly used in data science, while TensorFlow is a library specifically designed for machine learning

Development environment: A *development environment* is a software application (or sometimes a group of applications) that provide a whole suite of tools designed to help you (as the developer or machine learning engineer) build out your projects (Jupyter notebook)

Cloud services: In the context of machine learning, you can use a cloud service to access a server that is likely far more powerful than your own machine, or that comes equipped with machine learning models that are ready for you to use

End to end ML services with Azure

- You can analyze and train a small amount of data with your local machine using Jupyter notebook, Visual studio, or other tools. But with very large amounts of data, or you need a faster processor, it's a better idea to train and test the model remotely using cloud services such as Microsoft Azure. You can use Azure Data Science Virtual Machine, Azure Databricks, Azure Machine Learning Compute, or SQL server ML services to train and test models and use Azure Kubernetes to deploy models



Libraries

Scikit learn

Apache
Spark

Pytorch

Tensorflow

Matplotlib

Seaborn

Numpy

Pandas

Datasets	Define, version, and monitor datasets used in machine learning runs.
Experiments / Runs	Organize machine learning workloads and keep track of each task executed through the service.
Pipelines	Structured flows of tasks to model complex machine learning flows.
Models	Model registry with support for versioning and deployment to production.
Endpoints	Expose real-time endpoints for scoring as well as pipelines for advanced automation.

Cloud services

- A typical cloud service for machine learning provides support for managing the core assets involved in machine learning projects. For your reference, you can see a table summarizing these main **assets**

Feature	Description
Compute	Manage compute resources used by machine learning tasks
Environments	Templates for standardized environments used to create ML tasks
Datastores	Data sources connected to the service environment (e.g. Data Lake stores, databases).



Cloud Services

- Machine learning cloud services also need to provide support for **managing** the resources required for running machine learning tasks

Brief Intro to Azure Machine Learning

Feature	Description
Automated ML	Automate intensive tasks that rapidly iterate over many combinations of algorithms, hyperparameters to find the best model based on the chosen metric.
Designer	A drag-and-drop tool that lets you create ML models without a single line of code.
Datasets	A place you can create datasets.
Experiments	A place that helps you organize your runs.
Models	A place to save all the models created in Azure ML or trained outside of Azure ML.
Endpoints	A place stores real-time endpoints for scoring and pipeline endpoints for advanced automation.
Compute	A designated compute resource where you run the training script or host the service deployment.
Datastores	An attached storage account in which you can store datasets.



Models Vs Algorithms



Models are the **specific representations** learned from data



Algorithms are the processes of **learning**



$\text{Model} = \text{Algorithm}(\text{Data})$

Linear Regression

- **Linear regression** is an algorithm that uses a straight line (or plane) to describe relationships between variables
- It assumes there is a linear relationship between input and output variables
- Simple linear regression is a type of regression where there is only one independent variable (X) to arrive at dependent variable(Y)
- It can be represented as $Y = B_0 + B_1 * X$
- B_0 -Intercept(Decides where the line intercepts Y-axis)
- B_1 -Defines slope of the line



Linear regression

In more complex cases where there is *more than one* input variable, we might see something like this:

$$y = B_0 + B_1 * x_1 + B_2 * x_2 + B_3 * x_3 \dots$$

The main aim of Linear regression model is to minimize the error which is measured by cost function(RMSE)

Assumptions of Linear Regression

X and Y is assumed to have linear relation. If the data does not prove to show linear relation transformation techniques like log transformation is done to make it linear.

It assumes Input data is noiseless so it is necessary to remove outliers before training the model

Remove collinearity in order to prevent overfitting of highly correlated data

Make sure all the variables follow Gaussian distribution

Prediction is more reliable if you rescale input using scaling techniques

Irreducible error

$$Y = F(X) + e$$

E = This error can never be removed

It may be due to data collection problem or due to lack of features

It is different from model error

Parametric Vs Non- Parametric

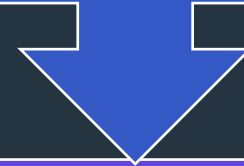
Parametric ML Algorithms simplifies assumptions about the mapping function and have a fixed number of parameters. No matter how much data is used to learn the model, this will not change how many parameters the algorithm has

Example of Parametric ML Algorithm would be Linear Regression

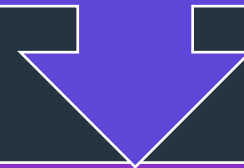


Benefits of Parametric Algorithms

Simpler and **easier** to understand



Faster when talking about learning from data



Less training data required to learn the mapping function, working well even if the fit to data is not perfect

Limitations of Parametric Algorithms



Highly constrained to the specified form of the simplified function



Limited complexity of the problems they are suitable for




Poor fit in practice, unlikely to match the underlying mapping function.

Non- Parametric Algorithms

It has no assumptions regarding the form of the mapping between input and output variables

Few of the Non-Parametric Algorithms are K Nearest Neighbours and Decision Tree



Benefits of Non- parametric Algorithms

High flexibility, in the sense that they are capable of fitting a large number of functional forms

Power by making weak or no assumptions on the underlying function

High performance in the prediction models that are produced



01

More training data is required to estimate the mapping function

02

Slower to train, generally having far more parameters to train

03

Overfitting the training data is a risk; overfitting makes it harder to explain the resulting predictions

Limitations of Non-Parametric Algorithms

Approaches to Machine Learning

Supervised learning

**Unsupervised
learning**

**Reinforcement
learning**

Supervised Learning

Learns from data that contains both the inputs and expected outputs (e.g., labeled data).
Common types are:

Classification: Outputs are categorical.

Regression: Outputs are continuous and numerical.

Similarity learning:
Learns from examples using a similarity function that measures how similar two objects are.

Feature learning:
Learns to automatically discover the representations or features from raw data.

Anomaly detection: A special form of classification, which learns from data labeled as normal/abnormal.

Unsupervised Learning

Learns from input data only; finds hidden structure in input data.

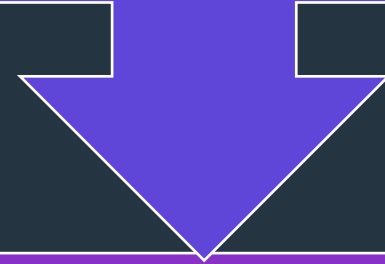
Clustering: Assigns entities to clusters or groups.

Feature learning: Features are learned from unlabeled data.

Anomaly detection: Learns from unlabeled data, using the assumption that the majority of entities are normal.

Reinforcement Learning

Learns how an agent should take action in an environment in order to maximize a reward function.



Markov decision process: A mathematical process to model decision-making in situations where outcomes are partly random and partly under the control of a decision-maker. Does not assume knowledge of an exact mathematical model.



The Tradeoffs

Bias Vs Variance
Tradeoff

Overfitting Vs
Underfitting

Bias Vs Variance Tradeoff

Bias measures how *inaccurate* the model prediction is in comparison with the true output. It is due to *erroneous assumptions* made in the machine learning process to simplify the model and make the target function easier to learn. High model complexity tends to have a low bias.

Variance measures how much the target function will *change* if different training data is used. Variance can be caused by modeling the *random noise* in the training data. High model complexity tends to have a high variance.

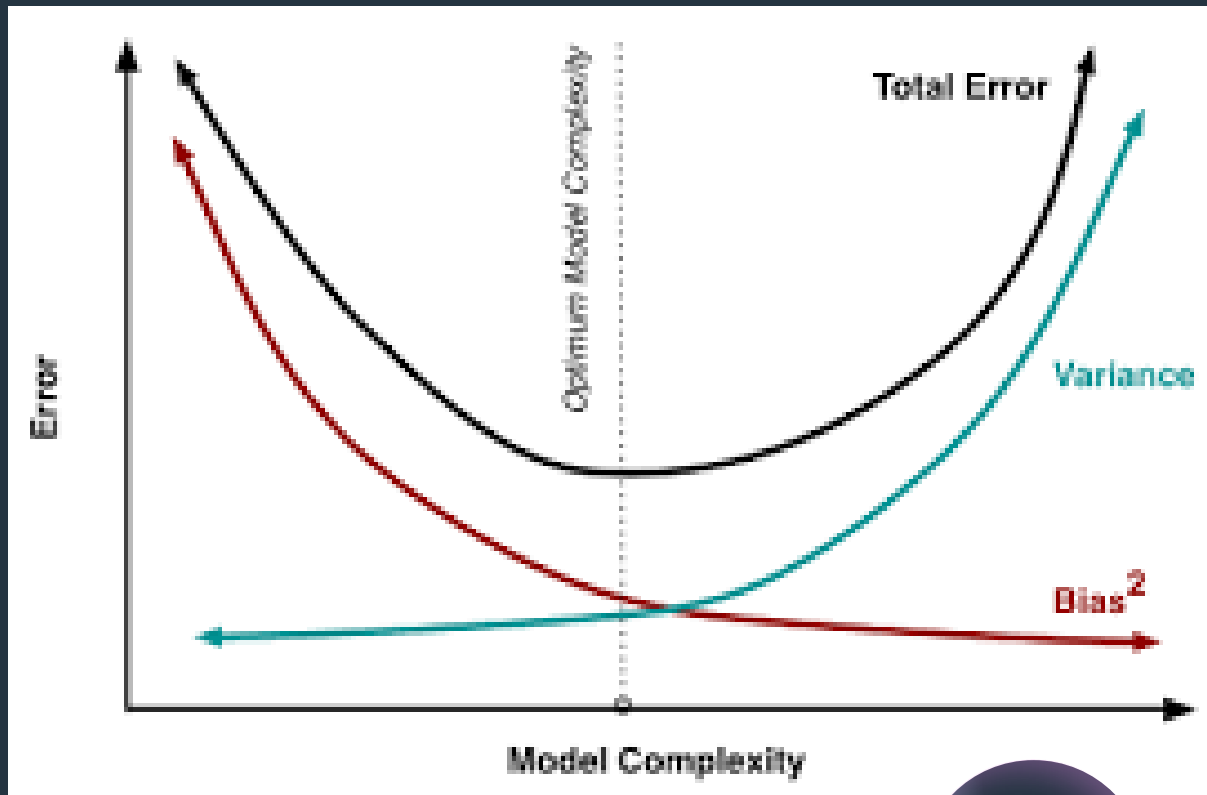
Parametric and linear algorithms often have high bias and low variance, whereas non-parametric and non-linear algorithms often have low bias and high variance

Bias Vs Variance Tradeoff

- **Lower bias correlates to few assumptions and high bias suggests more assumptions made about the form of the target functions.**
- **Variance error is caused by Variance**
- **High variance means it is strongly influenced by the specifics of the training data being used**
- **Low bias-KNN,Decision Tree**
- **High bias-Linear regression,Logistic Regression**
- **Low Variance:Linear regression,Linear Discriminant Analysis**
- **High Variance:Decision Tree,Support Vector Machine**

Bias Vs Variance Tradeoff

- Goal of Supervised Machine Learning Algorithms is to achieve low bias and low variance





Overfitting Vs Underfitting

Overfitting refers to the situation in which models fit the training data very well, but fail to generalize to new data.

Underfitting refers to the situation in which models neither fit the training data nor generalize to new data.

How to limit overfitting?

