Importance / Context

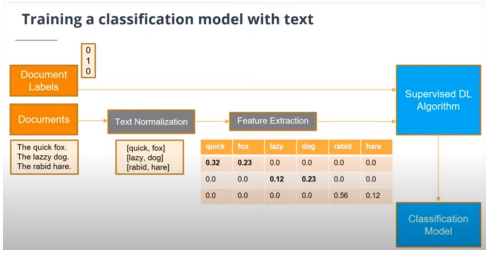
Thursday, July 9, 2020 11:06 PM

- What machine learning is and why it's so important in today's world
  - 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.
  - Take data and use it to make predictions or identify important relationships
  - Machine learning takes data and answers to determine rules; based on history
    - Limitations: we might not come up with best rules, ML will do its best to find the best rules possible based on data and answers
  - Traditional programming takes rules and data and finds answers
- Applications
  - Natural Language Processing
    - Text - summarization, topic detection, search
    - Speech - speech to text, text to speech, translation
  - Computer Vision
    - Self driving cars
    - Object detection / identification
    - LIDAR
  - Analytics
    - Regression, classification, forecasting, clustering
  - Decision Making
    - Sequential decision making
    - Recommendations
  - Examples:
    - Automating recognition of disease
      - Enables higher volume of patients seen
    - Recommend next best actions for individual care plans
      - Augment the diagnosis process
    - Enable personalized real-timing banking experiences with chatbots
    - Identifying the next best action for customer
      - assess the likelihood of a deal closing or the level of a customer's loyalty
      - can engage and delight customers with information and offers that are relevant to them
    - Capture, prioritize, and route service requests to the correct employee, and improve response times
      - capture incoming service requests, to route them to the correct employee in real-time, to refine prioritization, and improve response times
- The historical context of machine learning
  - AI emerging field in 50s
  - AI: broad term that refers to computers thinking like humans
  - Machine learning: subcategory of AI that involves learning from data without being explicitly programmed
  - Deep Learning: subcategory of machine learning that uses a layered neural network architecture originally inspired by the human brain

Data Science Process

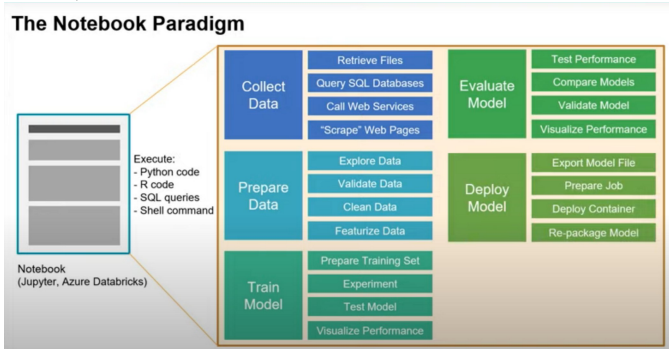
Monday, July 13, 2020 12:13 AM

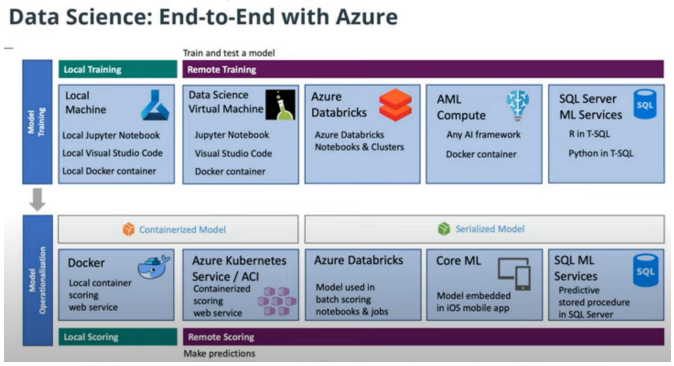
- The data science process
  - Collect Data
    - Write code to retrieve files / query databases etc.
  - Prepare Data : get data in format suitable for analysis, identify features etc.
    - Write queries and code to clean data, compute aggregate metrics etc.
    - Identify features useful in model training
  - Train Model : select algorithm, select training and test data sets, evaluate model to identify best working model
    - Write code, do math
    - Pick / tune ML algorithm, look at performance on validation datasets
  - Evaluate Model : test using validation data set to see how well it performs
    - Write Code, do math
    - Test performance of model by calculating evaluation metrics
  - Deploy Model : Implement then measure ongoing performance of model
    - Devops
    - Integrates training, evaluation and deployment steps in respective builds and release pipelines
  - Retrain Model : retrain on fresh data, iterative step
- The types of data that machine learning deals with
  - Types of data affect the parameters and hyperparameters
  - Numerical: integers or floats, all data ends up being numerical
  - Time-Series: numerical values that can be ordered based off date-time
    - Can have multiple series of data in a single set as long as its distinguished somehow
    - Stock prices over time etc.
  - Categorical: Discrete and limited set of values, meaning or category attached
    - Gender, ethnicity
  - Text
    - Words, sentences
    - Needs to be converted to a numerical representation which is hard
  - Image
    - Picture, snapshot of video
    - Transform it into a numerical representation is also a challenge
    - Pictures are not initially in numerical form but they will need to be transformed into RGB values, a set of numerical values ranging from 0 to 255, to be processed.
- Tabular Data
  - Data arranged in a data table
  - Rows describe single items while each column describes properties of item
  - Vectors
    - Array of numbers, used heavily in ML
    - All non-numerical data types must be eventually represented as numbers
      - In ML, numerical representation will be in the form of a vector
    - Vectorize: turn non-numerical data into vector form
- Scaling Data
  - Manipulating data so it fits within a certain range or scale
  - Scaling data affects training process, scaling 0-255 to 0-1 etc.
  - Two main approaches
    - Standardization
      - Rescales data to have mean = 0 and std. deviation = 1
      - Every value is modified: (x-mean)/variance
    - Normalization
      - Rescales the data into the range [0,1]
      - Every value is modified: (x-xmin)/(xmax-xmin)
- Encoding Categorical data
  - Ordinal Encoding: Converts categories into number of categories - 1, first category is 0, second is 1 etc.
    - Drawback: implicitly assumes order between categories (importance based on number)
  - One Hot Encoding: New columns are added for each original column, there is a new column for every potential category and values are replaced with either a 1 or a 0 to represent if an entity is part of that category
    - Drawback: large number of columns generated

- Image Data
  - Represented by pixels, three channels usually (RGB)
  - 3-D numeric vector size is [height]*[width]*[channel depth]
  - Each pixel in vector is represented by horizontal position, vertical position, color
  - Grayscale pictures have channel depth of 1
  - Preprocessing
    - Uniform aspect ratio: make sure input is a square
    - Normalized: subtract mean pixel value in a channel from each pixel value in that channel
    - Rotation, cropping, resizing, denoising, centering
- Text Data
  - Normalization: process of transforming a piece of text into a canonical (official) form
    - Difficult because there can be multiple forms that mean the same thing
      - To be, is, am, are
    - Can be multiple spellings of a word
      - Behavior, behaviour
  - Lemmatization: reduces multiple inflections to a single dictionary form
    - is, am are all become 'be'
  - Remove stop words (high freq. words that are unwanted during analysis)
    - The, which, a etc
  - Tokenize
    - Split each strong of text into a list of smaller parts or tokens
    - Split a sentence into separate keywords
  - Vectorization
    - Identify the particular features of the text that is relevant to the task
    - Get features extracted in a numerical form that is accessible to ML algorithm
    - Approaches include: TF-IDF, word embedding (Word2vec or Global Vectors GloVe)
      - TF-IDF gives less importance to words that contain less information and are common
      - More weight given to words that contain relevant information and are less frequent
	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**
  - Feature Extraction
    - Vectors can be visualized on a graph, the distance between two vectors is used to assess similarity in meaning or some connection
  - Pipeline for text data
    - Preprocessing and normalizing, tokenization, stop word removal etc
      -> feature extraction and vectorization (TF-IDF, GloVe, Word2Vec)
      -> feed vectorize document and labels into model and train model
    - 

Tools / Services

Thursday, July 9, 2020 11:12 PM

- The two main perspectives in ML: the *statistical* perspective and the *computer science* perspective
  - **Statistical**: output is dependent as a function of the input, looking for the function
    - trying to find a mathematical **function** that, given the values of the independent variables can predict the values of the dependent variables
    - Output variable = f(Input variables)
      - Dependent variables = f(Independent variables)
      - Y = f(X)
    - Input are independent, output is dependent
  - **Computer science**: data inputs (input features) are used to train a model to find the correct outputs, ML algorithm is trying to learn the function
    - using input features to create a program that can generate the desired output
    - Each row is an entity or observation about an entity, can also be called instance that is observed or generated by problem domain
    - Each column are attributes of the observation (features)
    - Input features and output features
      - **Input vector**: group of input variables
      - Output = Program(input Features)
- The essential tools needed for designing and training machine learning models
  - ML Ecosystem consists of three parts:
    - Libraries: Scikit-Learn, Pytorch, Tensorflow, Keras
      - Pre-written code that can be used publicly
    - Development Environment: Jupyter, Azure, Azure Databricks, VSCode, Visual Studio
      - Software application that provides a suite of tools designed to help develop projects
    - Cloud Services: Azure ML
      - Service that offers data storage or computing power over the internet
  - Notebooks: documenting tool that others can use to reproduce experiments
    - Combination of runnable code, output, formatted text and visualizations
    - Made up of one or more cells that allow execution of individual code snippets and chunks, output of each cell can be saved and viewed by others
    - Ex. Matlab, Wolfram Mathematica
    - 

  - Azure allows end to end model training with multiple services available in the suite
    - 

- Cloud Services
  - Provides support for managing core assets and resources in ML projects
  - Core Assets
    - 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
  - Resources
    - Compute
      - Manage compute resources used by machine learning tasks
    - Environments
      - Templates for standardized environments used to create compute resources
    - Datastores
      - Data sources connected to the service environment
        - Blob stores, file shares, data lake stores, databases

- Libraries
  - Core Framework and Tools
    - Python is a very popular high-level programming language that is great for data science. Its ease of use and wide support within popular machine learning platforms, coupled with a large catalog of ML libraries, has made it a leader in this space.
    - Pandas is an open-source Python library designed for analyzing and manipulating data. It is particularly good for working with tabular data and time-series data.
    - NumPy, like Pandas, is a Python library. NumPy provides support for large, multi-dimensional arrays of data, and has many high-level mathematical functions that can be used to perform operations on these arrays.
  - Machine Learning and Deep Learning
    - Scikit-Learn is a Python library designed specifically for machine learning. It is designed to be integrated with other scientific and data-analysis libraries, such as NumPy, SciPy, and matplotlib (described below).
    - Apache Spark is an open-source analytics engine that is designed for cluster-computing and that is often used for large-scale data processing and big data.
    - TensorFlow is a free, open-source software library for machine learning built by Google Brain.
    - Keras is a Python deep-learning library. It provide an Application Programming Interface (API) that can be used to interface with other libraries, such as TensorFlow, in order to program neural networks. Keras is designed for rapid development and experimentation.
    - PyTorch is an open source library for machine learning, developed in large part by Facebook's AI Research lab. It is known for being comparatively easy to use, especially for developers already familiar with Python and a Pythonic code style.
  - Data Visualization
    - Plotly is not itself a library, but rather a company that provides a number of different front-end tools for machine learning and data science—including an open source graphing library for Python.
    - Matplotlib is a Python library designed for plotting 2D visualizations. It can be used to produce graphs and other figures that are high quality and usable in professional publications. You'll see that the Matplotlib library is used by a number of other libraries and tools, such as SciKit Learn (above) and Seaborn (below). You can easily import Matplotlib for use in a Python script or to create visualizations within a Jupyter Notebook.
    - Seaborn is a Python library designed specifically for data visualization. It is based on matplotlib, but provides a more high-level interface and has additional features for making visualizations more attractive and informative.
    - Bokeh is an interactive data visualization library. In contrast to a library like matplotlib that generates a static image as its output, Bokeh generates visualizations in HTML and JavaScript. This allows for web-based visualizations that can have interactive features.
  - Will use a combination and variety of these tools to properly train models and visualize data

Azure ML - Models/Algorithms

Monday, July 13, 2020 12:22 AM

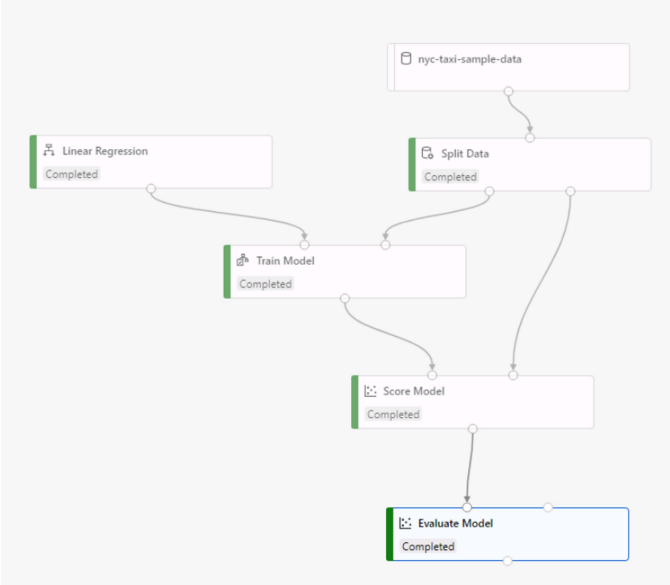
- The basics of Azure ML
  - Workspace: centralized place to work with all the artifacts you create
  - Features
    - Notebooks
      - Sample notebooks and user files loaded inside of compute instances
    - Automated ML
      - Can automate intensive tasks that rapidly iterate over many combinations of algorithms, hyperparameters to find the best model based on the chosen metric
      - Create new runs and view previous runs in the Automated ML tab
    - Designer
      - Drag-and-drop tool that lets you create ML models without any code
      - Has templates and can view drafts
    - Datasets
      - Create datasets from local files, datastores, etc
    - Experiments
      - Helps organize runs
      - All runs must be associated with an experiment, can view all runs related to an experiment
    - Models
      - Models are produced by runs in Azure ML, all models created in Azure or trained outside of Azure are accessible here
      - Must register models trained outside of Azure ML
    - Endpoints
      - Exposes real-time endpoints for scoring or pipelines for advanced automation
      - Each endpoint deployed shows up here and can be delved into
    - Compute
      - Designated compute resource where you run training script or host service deployment
      - Manage compute instance, training cluster, inference cluster, attached compute
    - Datastores
      - Attached storage account in which you can store datasets

- The distinction between models and algorithms
  - Models: specific representations learned from data
    - Model = Algorithm(Data)
    - Models represent what is learned by a machine learning algorithm on the data
    - Ex. y = Wx + b was solved for W = 1, b = 0
      - y = 1*x + 0 is the **model** that was obtained from running the **algorithm** y=Wx+b on training data
      - Models can be written in a set of weights or coefficients instead of the full equation
        - The algorithm is known so the coefficients/weights are all that's needed
  - Algorithms: process of learning
    - Prescriptive recipes that allow you to convert data into trained models
    - Choice of algorithm influences the internal structure of models
    - Standardized algorithm neutral representations of models
      - ONYX (Open Neural Net Exchange Format)
    - Ex. y = Wx+b
      - Based off different values of y and x, W and B can be calculated (learned), the learning process is the algorithm
      - "an algorithm was run on the data and learned the values for W and b"

- The basics of a linear regression model
  - Predicts a variable y from an input variable x
  - Assumes a linear relationship
  - Simple Linear Regression
    - Y = B0 + B1*X
    - B0 is the offset
  - Multiple Linear Regression
    - Y = B0 + B1*X1 + B2*X2…+Bn*Xn
    - Visualized as a **plane** of multiple dimensions
  - Minimize error between the line and each data point
  - Training linear regression model is to find the coefficients that best represent the input variables
    - Done by minimizing the error between data and the line of best fit
  - Algorithm
    - Preparing data
      - Linear assumption
        - May need to transform data to make data linear
        - Ex. if your data has an exponential relationship, you can use log transformation
      - Remove noise
        - Linear regression assumes input and output are not noisy so clean data and remove outliers especially in output
      - Remove collinearity
        - When two variables are collinear, this means they can be modeled by the same line or are at least highly correlated; in other words, one input variable can be accurately predicted by the other
        - Will overfit when highly correlated variables are input, consider removing the most co-related
        - highly correlated input variables will make the model less consistent, so it's important to perform a correlation check among input variables and remove highly correlated input variables
      - Gaussian distribution
        - Linear regression will make better predictions if variables have a gaussian distribution
        - Linear regression assumes that the distance between output variables and real data (called residual) is normally distributed
      - Rescale Inputs
        - More reliable predictions if you rescale inputs using standardization or normalization
        - Linear regression is very sensitive to the distance among data points
  - Model
    - Y = B0 + B1*X
    - Can estimate coefficients from the data
      - Calculate mean from variables using training data
      - Slope:
        - $$B1 = \frac{\sum_{i=1}^{n}(x_i - mean(x)) \times (y_i - mean(y))}{\sum_{i=1}^{n}(x_i - mean(x))^2}$$
      - Intercept:
        - $$B0 = mean(y) - B1 \times mean(x)$$
      - Estimating Error: Root mean squared error RMSE
        - $$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(p_i - y_i)^2}{n}}$$
  - Training a model
    - Learn the coefficients and bias that best fir the data
    - Cost Function
      - The process of finding the coefficients and bias that minimize error.
      - RMSE is the most common

Lab: Train a Linear Regression Model

Monday, July 13, 2020 3:48 PM



https://docs.microsoft.com/en-us/azure/machine-learning/algorithm-module-reference/linear-regression

Score Model result visualization

Rows 3,520 Columns 15

snowDepth	precipTime	precipDepth	temperature	totalAmount	Scored Labels
0	6	0	-1.95	12.8	11.752907
0	6	0	24.638136	9.8	11.405143
0	24	51	21.695868	8.3	8.796228
0	1	0	6.591071	12.8	14.62984
3.571429	24	9999	7.844094	11.3	14.286417
0	24	43	12.368519	7.8	9.034839
0	6	0	11.371429	21.62	24.4253
0	1	0	6.845536	15.3	17.997363
0	24	28	4.400885	8.8	10.153361
54.411765	6	0	-0.308929	9.36	6.510378
0	24	48	8.551786	15.36	9.432731
0	1	0	16.794643	10.8	14.408524
0	6	0	-2.06875	37	37.630629

Scored Labels

Statistics

Mean	14.7943
Median	11.285
Min	3.9313
Max	99.6638
Standard deviation	10.1191
Unique values	3516
Missing values	0
Feature type	Numeric Score

Visualizations

Evaluate Model result visualization

Rows 1 Columns 5

Mean_Absolute_Error	Root_Mean_Squared_Error	Relative_Squared_Error	Relative_
2.131346	3.805547	0.136663	0.294669

Root_Mean_Squared_Error

Statistics

Mean	3.8055
Median	3.8055
Min	3.8055
Max	3.8055
Standard deviation	-
Unique values	1
Missing values	0
Feature type	Numeric Feature

Visualizations

Learning Functions et al.

Monday, July 13, 2020 4:12 PM

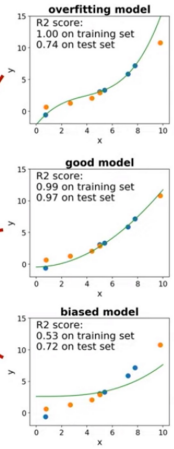
- Learning a function
  - Core goal is to learn a useful transformation of the input data that gets us closer to expected output via target function f
    - Y = f(X)
    - Process extrapolates from a limited set of values so there will always be an error e called **irreducible error** because the target function will always be estimated
      - Y = f(X) + e
    - Irreducible error is different from model error, it is caused by data collection process
      - Not enough data
      - Not enough data features
      - Model error is how different the predictions are from the true output, can be reduced by refining the model learning process

- The distinction between parametric vs. non-parametric functions
  - ML algorithms can be classified into two groups
    - Parametric
      - Simplify mapping to a known functional form, ex. Linear regression
      - Regression simplifies the learning process because it knows the general form, then just computes for the coefficients and constants
      - Benefits
        - Simpler, easier to understand, easier to interpret results
        - Faster when talking about learning from data
        - Less training data required to learn the mapping function, works well even if the fit to data is not perfect
      - Limitations
        - Highly constrained to the specific 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
    - Nonparametric
      - Not making assumptions regarding the form of the mapping between input data and output
      - Allows free form relationships to form, any functional form
      - Ex. K-nearest neighbors
      - Benefits
        - High flexibility, 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
      - Limitations
        - More training data needed to estimate the mapping function
        - Slower to train, generally having far more parameters
        - Overfitting the training data is a risk

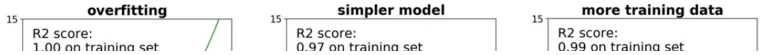
- The distinction between classical machine learning vs. deep learning
  - Deep learning is a specific subset of ML algorithms
    - All deep learning is ML, not all ML is deep learning
  - Classical ML
    - Based on classical mathematical algorithms
    - More suitable for small data
    - Easier to interpret outcomes
    - Cheaper to perform
    - Can run on low-end machines
    - Does not require large computational power
    - Difficult to learn large datasets
    - Requires feature engineering
    - Difficult to learn complex functions
  - Deep Learning
    - Based on neural networks
    - Suitable for high complexity problems
    - Better accuracy, compared to classical ML
    - Better support for big data
    - Complex features can be learned
    - Difficult to explain trained data
    - Requires significant computational power

Concept: Overfitting & Underfitting

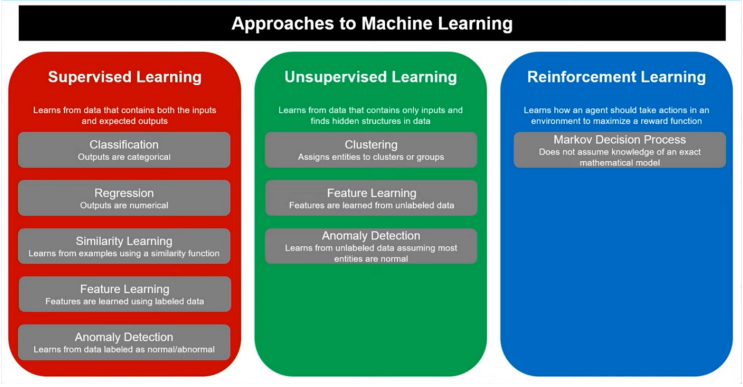
		Model performance on training data	
		Poor	Good
Model performance on new data	Poor	Underfitting (high bias)	Overfitting (high variance)
	Good	Very unlikely	Optimal



Best practice to limit overfitting

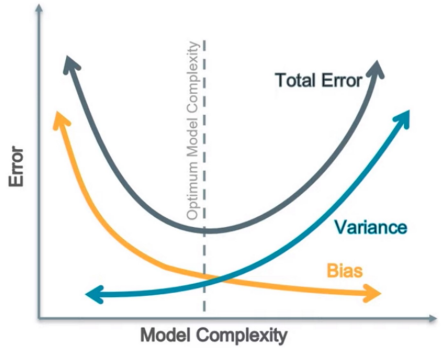


- The main approaches to machine learning
  - Supervised Learning
    - Learns from data that contains both inputs and expected outputs
    - **Classification**: outputs are categorical
    - **Regression**: outputs are continuous and numerical
    - **Similarity Learning**: learns from examples using a similarity function, used in ranking or recommendations
    - **Feature Learning**: Learns to automatically discover the representations or features from raw data
    - **Anomaly Detection**: special form of classification, learns from data labeled as normal/abnormal
      - Generally a much smaller dataset than normal classification problems, makes it harder to have accurate / high quality models
  - Unsupervised Learning
    - Learns from data that only contains inputs, finds hidden structures in input data
    - **Clustering**: find inherent groups or clusters in the data, assigns entities to each cluster/group
    - **Feature Learning**: Features are learned from unlabeled data
    - **Anomaly Detection:** Learns from unlabeled data, assumes that the majority of the entities are normal
  - Reinforcement Learning
    - Learns how an agent should take action in an environment to maximize a reward function
    - **Markov Decision Process:** Mathematical process to model decision making in situations where out comes are partly random and partly under the control of a decision maker.
      - Does not assume knowledge of an exact mathematical model
  - Reinforcement learning is an active process where the actions of the agent influence the data observed in the future, hence influencing its own potential future states
  - Supervised and unsupervised learning approaches are passive processes where learning is performed without any actions that could influence the data

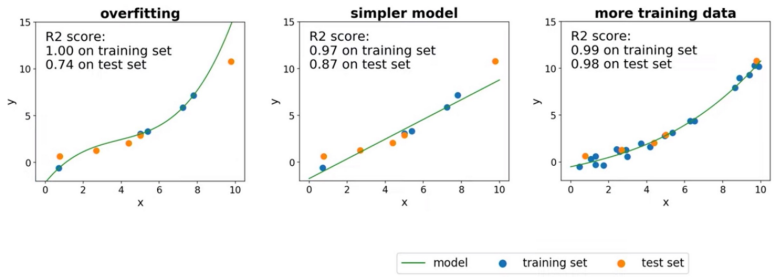


- The trade-offs that come up when making decisions about how to design and training machine learning models
  - Bias vs Variance
    - **Bias**: simplifying assumptions made by a model to make the target function easier to learn
      - Measures how inaccurate the model prediction is in comparison to true output
      - Due to erroneous assumptions made in the ML process to simplify the model and make the target function easier to learn
      - High model complexity usually has low bias
      - Low bias means fewer assumptions about the target function
      - High bias = more assumptions + can potentially miss important relations between features and outputs and cause underfitting
    - **Variance**: amount the estimate of the target function will change if different training data is used
      - Can be caused by modeling the random noise in the training data
      - High model complexity usually has high variance
      - Low variance indicates changes in training data would result in similar target functions
      - High variance suggests that the algorithm learns the random noise instead of the output and causes overfitting
    - Parametric and linear algorithms usually have high bias, low variance
    - Non-parametric and non-linear algorithms usually have low bias, high variance
    - Prediction error: sum of modeling error and the irreducible error
      - Prediction error = bias error + variance error + irreducible error
    - Generally, increasing model complexity would decrease bias error since the model has more capacity to learn from the training data. But the variance error would increase if the model complexity increases, as the model may begin to learn from noise in the training data
    - Optimal complexity is where bias error crosses with variance error

Bias-Variance Trade-Off



- Overfitting vs Underfitting
  - Overfitting
    - Situation in which the model fits the training data very well but fails to generalize new data
    - "memorizing" the data and not adapting well to new data
  - Underfitting
    - Situation in which the models neither fit the training data nor generalize to new data
    - Doesn't model training data well, doesn't generalize new data well either



- Situation in which the model fits the training data very well but fails to generalize new data
- "memorizing" the data and not adapting well to new data

- Underfitting
  - Situation in which the models neither fit the training data nor generalize to new data
  - Doesn't model training data well, doesn't generalize new data well either
- Limiting Overfitting
  - K-fold cross-validation
    - Splits initial training data into k subsets and trains the model k times
  - Hold back a validation dataset from the initial training data to estimate how well the model generalizes on new data
  - Simplify the model
  - Use more data
  - Reduce dimensionality
    - PCA
  - Stop the training early when performance has not improved after a number of training iterations