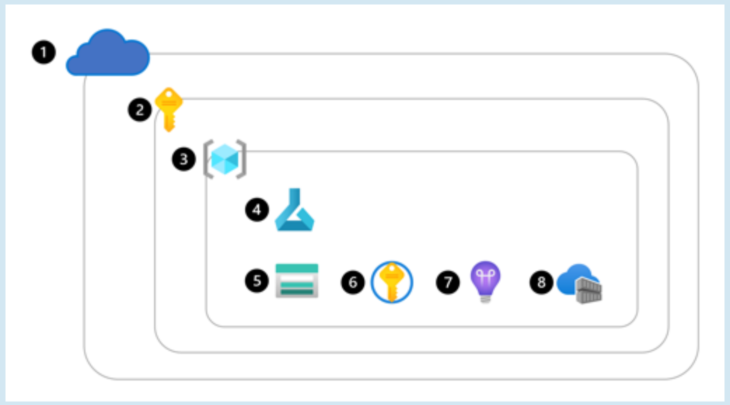
## Azure ML Service Structure



To create an Azure ML service, you need:

1. **Azure portal**
2. **Azure subscription**
3. **Resource group**
4. **Azure ML service** (create a workspace)
5. **Azure Storage Account**
6. **Azure Key Vault**
7. **Application Insights** (monitor services)
8. **Azure Container Registry** (storing images, is optional)

## Data

To find and access data in Azure ML, you'll use **Uniform Resource Identifiers (URIs)**

### Data assets

- Definition: Data assets are **references to where the data is stored**, **how to get access**, and any other relevant metadata.
- To simplify getting access to the data you want to work with, you can use data assets.
- There are three main types of data assets you can use:
  **URI file**: Points to a specific file.
  **URI folder**: Points to a folder.
  **MLTable**: Points to a folder or file and includes a schema to read as tabular data.

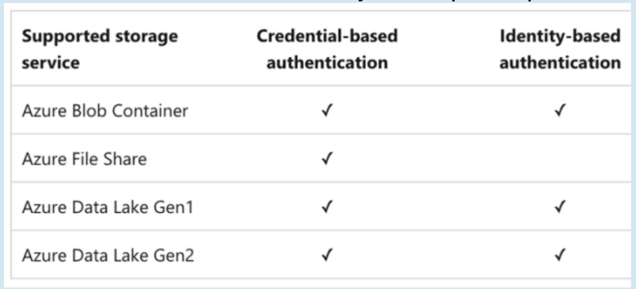
### Dataset

You can create the following types of datasets:

- **Tabular**: The data is read from the dataset as a table. You should use this type of dataset when your data is <u>consistently structured</u> and you want to work with it in common tabular data structures, such as Pandas dataframes.
- **File**: The dataset presents a list of file paths that can be read as though from the file system. Use this type of dataset when your <u>data is unstructured</u>, <u>or when you need to process the data at the file level</u> (for example, to train a convolutional neural network from a set of image files).

### Datastores

- Definition: Datastores are attached to the workspace and can be referred by name. They are used to store connection information to Azure storage services.
- Types:
  - Azure Storage (blob and file containers)
  - Azure Data Lake stores
  - Azure SQL Database
  - Azure Databricks file system (DBFS)

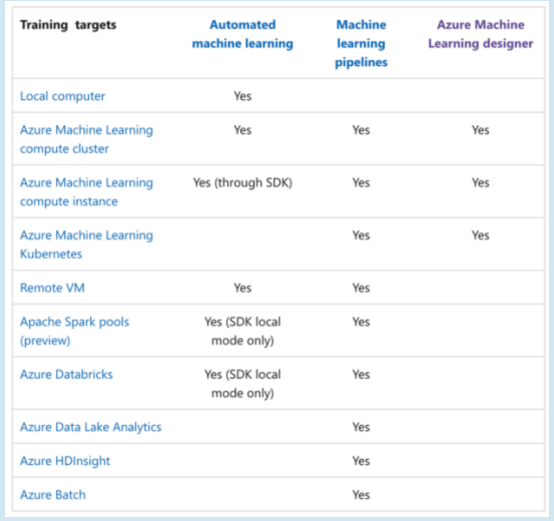


| Supported storage service | Credential-based authentication | Identity-based authentication |
|---|---|---|
| Azure Blob Container | ✓ | ✓ |
| Azure File Share | ✓ | |
| Azure Data Lake Gen1 | ✓ | ✓ |
| Azure Data Lake Gen2 | ✓ | ✓ |

- When a workspace is created, an Azure Storage account is created and automatically connected to the workspace. As a result, you'll have two datastores already added to your workspace:
  - **workspacefilestore**: Used to store files like Jupyter notebooks and Python scripts.
  - **workspaceblobstore**: Used to store metrics and output when tracking model training.

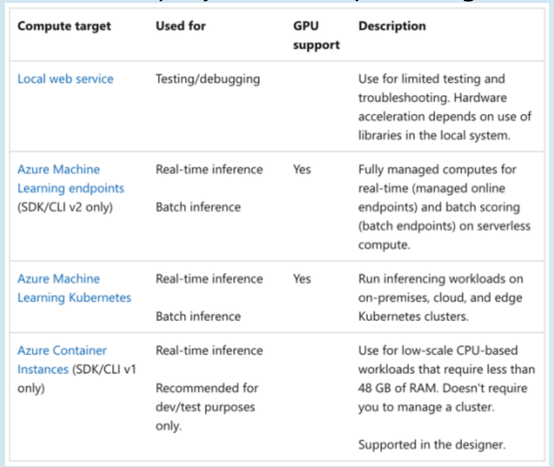
## Compute target

- Definition: Is a designated compute resource or environment where you run your training script or host your service deployment
- In a typical model development lifecycle, you might:
  - Start by developing and experimenting on a small amount of data. At this stage, use your local environment, such as a **local computer** or **cloud-based virtual machine** (VM), as your compute target.
  - Scale up to larger data, or do <u>**distributed GPU** training</u> by using one of these training compute targets:



| Training targets | Automated machine learning | Machine learning pipelines | Azure Machine Learning designer |
|---|---|---|---|
| Local computer | Yes | | |
| Azure Machine Learning compute cluster | Yes | Yes | Yes |
| Azure Machine Learning compute instance | Yes (through SDK) | Yes | Yes |
| Azure Machine Learning Kubernetes | | Yes | Yes |
| Remote VM | Yes | Yes | |
| Apache Spark pools (preview) | Yes (SDK local mode only) | Yes | |
| Azure Databricks | Yes (SDK local mode only) | Yes | |
| Azure Data Lake Analytics | | Yes | |
| Azure HDInsight | | Yes | |
| Azure Batch | | Yes | |

  - After your model is ready, deploy it to a web hosting environment with one of these deployment compute targets.



| Compute target | Used for | GPU support | Description |
|---|---|---|---|
| Local web service | Testing/debugging | | Use for limited testing and troubleshooting. Hardware acceleration depends on use of libraries in the local system. |
| Azure Machine Learning endpoints (SDK/CLI v2 only) | Real-time inference<br><br>Batch inference | Yes | Fully managed computes for real-time (managed online endpoints) and batch scoring (batch endpoints) on serverless compute. |
| Azure Machine Learning Kubernetes | Real-time inference<br><br>Batch inference | Yes | Run inferencing workloads on on-premises, cloud, and edge Kubernetes clusters. |
| Azure Container Instances (SDK/CLI v1 only) | Real-time inference<br><br>Recommended for dev/test purposes only. | | Use for low-scale CPU-based workloads that require less than 48 GB of RAM. Doesn't require you to manage a cluster.<br><br>Supported in the designer. |

## Train models

- **Azure Machine Learning SDK for Python**

| Training method | Description |
|---|---|
| command () | submit a command() that includes a training script, environment, and compute information |
| Auto ML | Automate algorithm selection and hyperparameter tuning. You don't have to worry about defining a job configuration when using automated ML |
| ML pipeline | Pipelines are not a different training method, but a way of defining a workflow using modular, reusable steps that can include training as part of the workflow |

- **Designer**: web-based UI
- **Azure CLI**: is often used for scripting and automating tasks.

## Automate ML model selection (AutoML)

- **primary_metric**: is the target performance metric for which the optimal model will be determined
- **set_limits()**: To minimize costs and time spent on training, you can set limits to an AutoML experiment or job:

  - **timeout_minutes**: Number of minutes after which the complete AutoML experiment is terminated.
  - **trial_timeout_minutes**: Maximum number of minutes one trial can take.
  - **max_trials**: Maximum number of trials, or models that will be trained.
  - **enable_early_termination**: Whether to end the experiment if the score isn't improving in the short term.

## Train models with scripts

- To create a production-ready script, you'll need to:
  - Remove nonessential code.
  - Refactor your code into functions. (Into multiple small functions rather than one large function)
  - Test your script in the terminal.
- MLflow
  is an **open-source** platform, designed to manage the complete machine learning lifecycle.

  There are two options to track machine learning jobs with MLflow:
  - Enable autologging using **mlflow.autolog()**
  - Use logging functions to track custom metrics using **mlflow.log_***
    mlflow.log_param(): For log **input** single key-value parameter
    mlflow.log_metric(): Log single key-value metric. Value must be a number. For log **output** metric

mlflow.log_artifact(): Log a file, include plot (save as image file first)

- Retriever runs.
  - retrieve the metrics of a specific run: mlflow.search_runs(exp.experiment_id)
  - search across all the experiments in the workspace: search_all_experiments=True

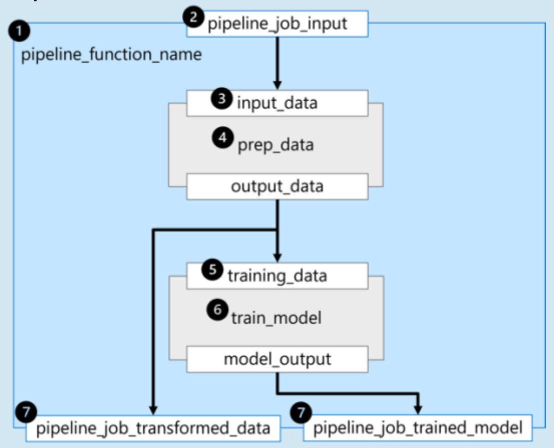
## Optimize model training

- Pipelines: in Azure ML, a pipeline is a workflow of machine learning tasks in which each task is defined as a component.

  Create components:
  - A component may consist of a Python script that normalizes your data, trains a machine learning model, or evaluates a model.
  - Components can be easily shared to other Azure ML users, who can reuse components in their own Azure ML pipelines within the same Azure ML workspace.

  Register a component.
  - To make the components accessible to other users in the workspace, you can also register components: prep =ml_client.components.create_or_update(prepare_data_component)

  Pipeline



  - Run a pipeline job:

```python
# submit job to workspace
pipeline_job = ml_client.jobs.create_or_update(
    pipeline_job, experiment_name="pipeline_job"
)
```

  - Schedule a pipeline job: use RecurrenceTrigger

- Hyperparameter tuning.
  - Discrete hyperparameters.
    **Python list** (Choice(values=[10,20,30])), **range** (Choice(values=range(1,10))), **comma-separated values** (Choice(values=(30,50,100)))
    **QUniform**(min_value, max_value, q): Uniform means you sample it from a set where drawing each element is equally probable
    **QLogUniform**(min_value, max_value, q): Returns a value like round(exp(Uniform(min_value, max_value)) / q) * q
    **QNormal**(mu, sigma, q): Returns a value like round(Normal(mu, sigma) / q) * q
    **QLogNormal**(mu, sigma, q): Returns a value like round(exp(Normal(mu, sigma)) / q) * q

  - Continuous hyperparameters.
    **Uniform**(min_value, max_value)
    **LogUniform(**min_value, max_value)
    **Normal**(mu, sigma)
    **LogNormal**(mu, sigma)

  - Configure a sampling method.
    **Grid sampling**
    **Random sampling** (**Sobol**: Adds a seed to random sampling to make the results reproducible.)
    hyperparameters.
    **Bayesian sampling**

  - Configure early termination.
    `evaluation_interval`: Specifies at which interval you want the policy to be evaluated.
    `delay_evaluation`: Specifies when to start evaluating the policy.

    **Bandit policy**: Uses a slack_factor (relative) or slack_amount(absolute). Any new model must perform within the slack range of the best performing model.
    early_termination_policy = BanditPolicy(slack_amount = 0.35, evaluation_interval=1, delay_evaluation=5)
    **Median stopping policy**: Uses the median of the averages of the primary metric. Any new model must perform better than the median.
    early_termination_policy = MedianStoppingPolicy(evaluation_interval=1, delay_evaluation=5)
    **Truncation selection policy**: Uses a truncation_percentage, which is the percentage of lowest performing trials. Any new model must perform better than the lowest performing trials.
    early_termination_policy = TruncationSelectionPolicy(truncation_percentage=10, evaluation_interval=1, delay_evaluation=5)

- more

## Deploy a model

- Deploy models to a managed online endpoint for real-time inferencing.

  - An endpoint is an HTTPS endpoint to which you can send data, and which will return a response (almost) immediately.

  - Any data you send to the endpoint will serve as the input for the scoring script hosted on the endpoint. The scoring script loads the trained model to predict the label for the new input data, which is also called inferencing.

  - Within Azure Machine Learning, there are two types of online endpoints:
    **Managed online endpoints**: Azure Machine Learning manages all the underlying infrastructure.
    **Kubernetes online endpoints**: Users manage the Kubernetes cluster which provides the necessary infrastructure.

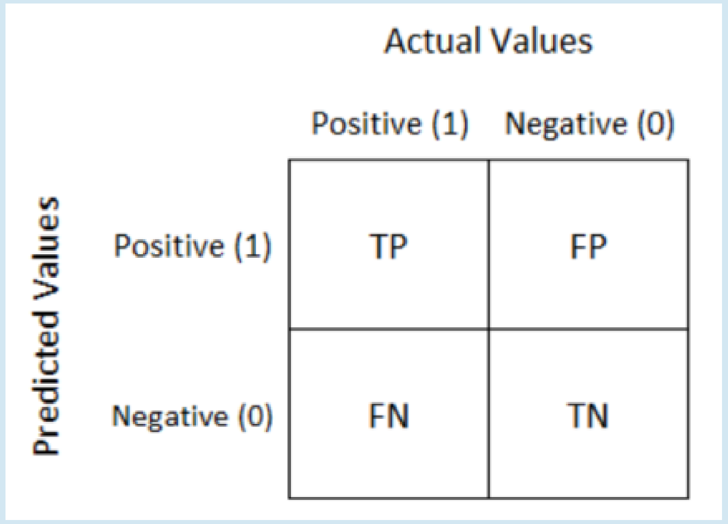
  - To deploy your model to a managed online endpoint, you need to specify four things:

    **Model assets** like the model pickle file, or a registered model in the Azure Machine Learning workspace.
    **Scoring script** that loads the model.
    **Environment** which lists all necessary packages that need to be installed on the compute of the endpoint.
    **Compute configuration** including the needed compute size and scale settings.

  - Blue/green deployment
    Blue/green deployment allows for multiple models to be deployed to an endpoint. You can decide how much traffic to forward to each deployed model. This way, you can switch to a new version of the model without interrupting service to the consumer.

## Evaluate automated machine learning experiment results

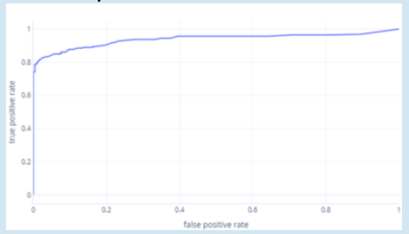
Classification



| | | Actual Values | |
|---|---|---|---|
| | | Positive (1) | Negative (0) |
| Predicted Values | Positive (1) | TP | FP |
| | Negative (0) | FN | TN |

- Confusion Matrix

- **Accuracy** = (TP + TN)/Number of samples
  Objective: Closer to 1 the better
  Range: [0, 1]
  Usage: used for **balanced** datasets. A high accuracy model indicates very few incorrect predictions. However, this doesn't consider the business cost of those incorrect predictions.
- **Precision** = TP / (TP + FP)
  Objective: Closer to 1 the better
  Range: [0, 1]
- **Recall** (True Positive Rate) = TP/(TP+FN)
  Objective: Closer to 1 the better
  Range: [0, 1]
- **F1 Score** = **harmonic mean** of Precision and Recall = 2*Precision*Recall / (Precision + Recall)
  Objective: Closer to 1 the better
  Range: [0, 1]
- **FPR** (False Positive Rate) = FP / (TN + FP)
- **Specificity** = TN / (TN + FP)
- **False Positive Rate** + **Specificity = 1**
- **ROC** (Receiver Operator Characteristic Curves)



  Objective: Closer to 1 the better
  Range: [0, 1]
- AUC (Area Under the ROC Curve)
  Usage: used for **imbalanced** datasets. AUC is the go-to metric in such scenarios as it calibrates the trade-off between sensitivity and specificity at the best-chosen threshold.
- log_loss

$$Logloss = \frac{1}{N}\sum_{i=1}^{N} logloss_i$$

$$Logloss = -\frac{1}{N}\sum_{i=1}^{N}[y_i \ln p_i + (1-y_i)\ln(1-p_i)]$$

  Objective: Closer to 0 the better
  Range: [0, inf)
- more

Regression
- **MAE**: Mean Absolute Error, the average of the difference between the actual value and the predicted one.
  The lower the MAE, the better the model

$$MAE = (\frac{1}{n})\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

- **MSE**: Mean squared error.

$$MSE = (\frac{1}{n})\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

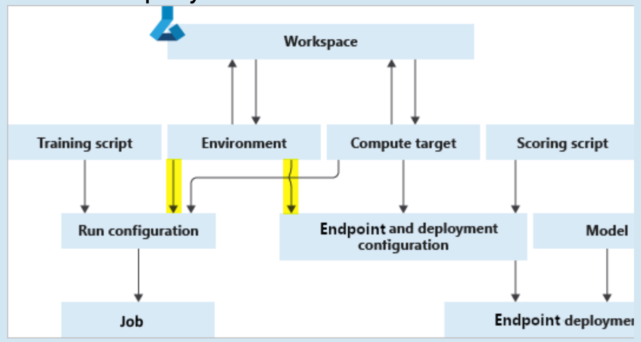
- **RMSE**: Root mean squared error.

$$RMSE = \sqrt{(\frac{1}{n})\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

- **r2_score**: R Square (also named coefficient of Determination) measures the proportional reduction in mean squared error (MSE) relative to the total variance of the observed data.
  Objective: Closer to 1 the better
  Range: [-1, 1]
- more

## Environment

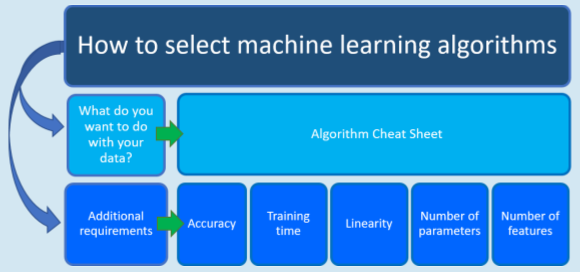
- The following diagram illustrates how you can use a single Environment object in both your job configuration (for training) and your inference and deployment configuration (for web service deployments



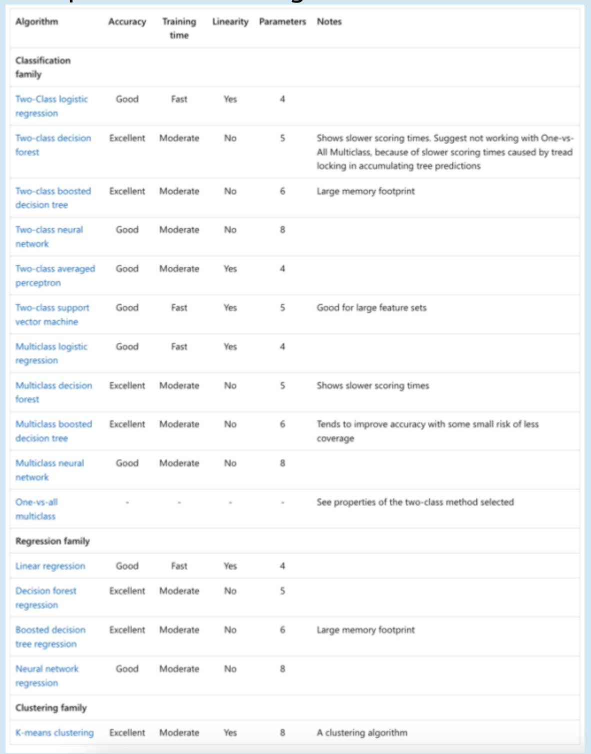
- Type of environments
  - **Curated**
  - **User-managed**
  - **System-managed**

## Machine Learning Algorithm

- How to select algorithms

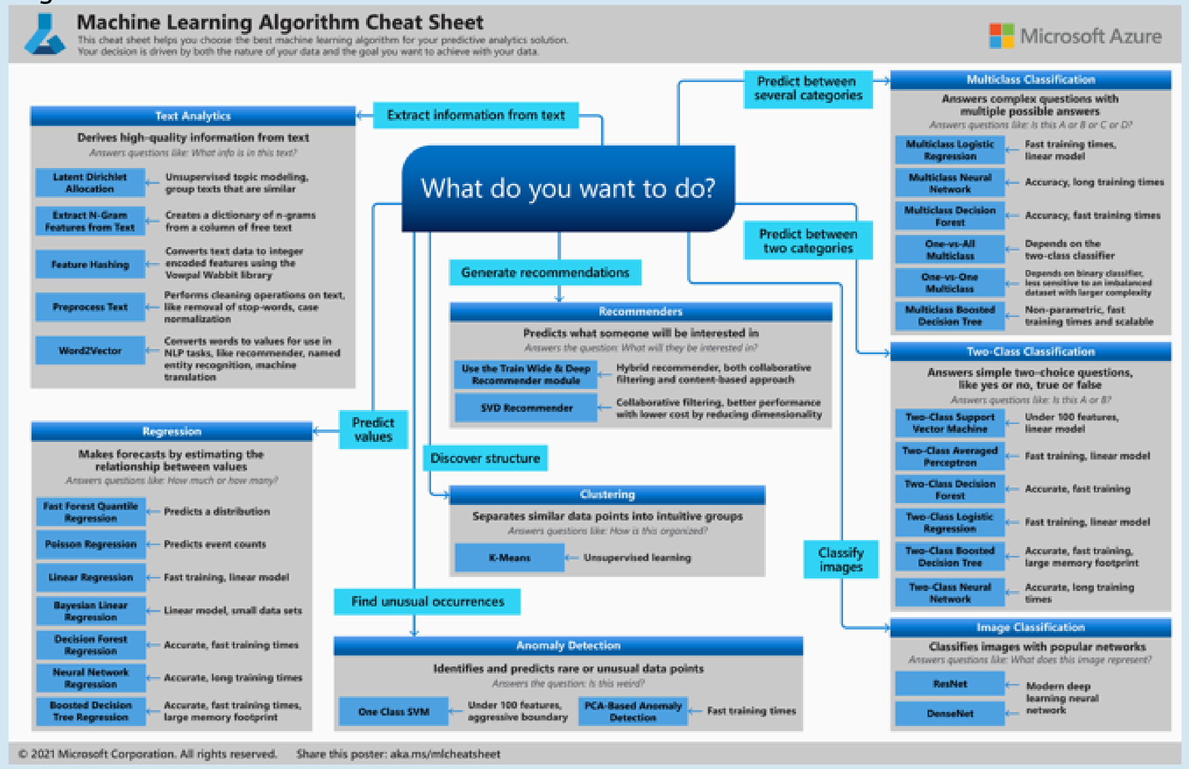


- Comparison of ML algorithms.



| Algorithm | Accuracy | Training time | Linearity | Parameters | Notes |
|---|---|---|---|---|---|
| **Classification family** | | | | | |
| Two-Class logistic regression | Good | Fast | Yes | 4 | |
| Two-class decision forest | Excellent | Moderate | No | 5 | Shows slower scoring times. Suggest not working with One-vs-All Multiclass, because of slower scoring times caused by tread locking in accumulating tree predictions |
| Two-class boosted decision tree | Excellent | Moderate | No | 6 | Large memory footprint |
| Two-class neural network | Good | Moderate | No | 8 | |
| Two-class averaged perceptron | Good | Moderate | Yes | 4 | |
| Two-class support vector machine | Good | Fast | Yes | 5 | Good for large feature sets |
| Multiclass logistic regression | Good | Fast | Yes | 4 | |
| Multiclass decision forest | Excellent | Moderate | No | 5 | Shows slower scoring times |
| Multiclass boosted decision tree | Excellent | Moderate | No | 6 | Tends to improve accuracy with some small risk of less coverage |
| Multiclass neural network | Good | Moderate | No | 8 | |
| One-vs-all multiclass | - | - | - | - | See properties of the two-class method selected |
| **Regression family** | | | | | |
| Linear regression | Good | Fast | Yes | 4 | |
| Decision forest regression | Excellent | Moderate | No | 5 | |
| Boosted decision tree regression | Excellent | Moderate | No | 6 | Large memory footprint |
| Neural network regression | Good | Moderate | No | 8 | |
| **Clustering family** | | | | | |
| K-means clustering | Excellent | Moderate | Yes | 8 | A clustering algorithm |

- Algorithm Cheat Sheet



Machine Learning Algorithm Cheat Sheet
This cheat sheet helps you choose the best machine learning algorithm for your predictive analytics solution. Your decision is driven by both the nature of your data and the goal you want to achieve with your data.
Microsoft Azure
What do you want to do?
Extract information from text
Text Analytics
Derives high-quality information from text
Answers questions like: What info is in this text?
Latent Dirichlet Allocation
Unsupervised topic modeling, group texts that are similar
Extract N-Gram Features from Text
Creates a dictionary of n-grams from a column of free text
Feature Hashing
Converts text data to integer encoded features using the Vowpal Wabbit library
Preprocess Text
Performs cleaning operations on text, like removal of stop-words, case normalization
Word2Vector
Converts words to values for use in NLP tasks, like recommender, named entity recognition, machine translation
Predict values
Regression
Makes forecasts by estimating the relationship between values
Answers questions like: How much or how many?
Fast Forest Quantile Regression
Predicts a distribution
Poisson Regression
Predicts event counts
Linear Regression
Fast training, linear model
Bayesian Linear Regression
Linear model, small data sets
Decision Forest Regression
Accurate, fast training times
Neural Network Regression
Accurate, long training times
Boosted Decision Tree Regression
Accurate, fast training times, large memory footprint
Generate recommendations
Recommenders
Predicts what someone will be interested in
Answers the question: What will they be interested in?
Use the Train Wide & Deep Recommender module
Hybrid recommender, both collaborative filtering and content-based approach
SVD Recommender
Collaborative filtering, better performance with lower cost by reducing dimensionality
Discover structure
Clustering
Separates similar data points into intuitive groups
Answers questions like: How is this organized?
K-Means
Unsupervised learning
Find unusual occurrences
Anomaly Detection
Identifies and predicts rare or unusual data points
Answers the question: Is this weird?
One Class SVM
Under 100 features, aggressive boundary
PCA-Based Anomaly Detection
Fast training times
Predict between several categories
Multiclass Classification
Answers complex questions with multiple possible answers
Answers questions like: Is this A or B or C or D?
Multiclass Logistic Regression
Fast training times, linear model
Multiclass Neural Network
Accuracy, long training times
Multiclass Decision Forest
Accuracy, fast training times
One-vs-All Multiclass
Depends on the two-class classifier
One-vs-One Multiclass
Depends on binary classifier, less sensitive to an imbalanced dataset with larger complexity
Multiclass Boosted Decision Tree
Non-parametric, fast training times and scalable
Predict between two categories
Two-Class Classification
Answers simple two-choice questions, like yes or no, true or false
Answers questions like: Is this A or B?
Two-Class Support Vector Machine
Under 100 features, linear model
Two-Class Averaged Perceptron
Fast training, linear model
Two-Class Decision Forest
Accurate, fast training
Two-Class Logistic Regression
Fast training, linear model
Two-Class Boosted Decision Tree
Accurate, fast training, large memory footprint
Two-Class Neural Network
Accurate, long training times
Classify images
Image Classification
Classifies images with popular networks
Answers questions like: What does this image represent?
ResNet
DenseNet
Modern deep learning neural network
© 2021 Microsoft Corporation. All rights reserved.
Share this poster: aka.ms/mlcheatsheet