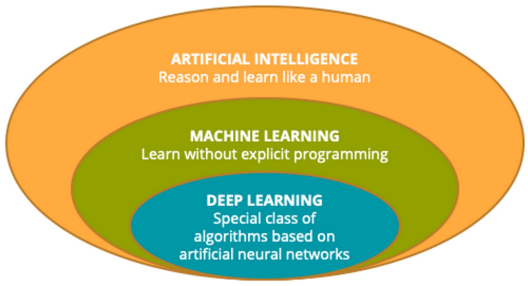
Introduction

Wednesday, July 22, 2020 5:10 PM

- Specialized cases of model training
  - *Similarity learning* and the basic features of a recommendation engine
    - Recommendations in ecommerce
  - *Text classification* and the fundamentals of processing text in machine learning
    - Understand and process natural language
  - *Feature learning*, an essential task in feature engineering
  - Anomaly detection
  - Time-series forecasting

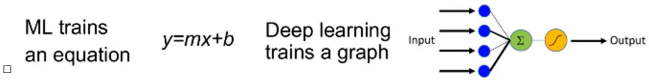
Classical ML vs Deep Learning

Wednesday, July 22, 2020 12:44 AM

- Deep Learning
  - A specialized type of machine learning algorithm
  - Not all ML is DL but all DL is ML
    - 

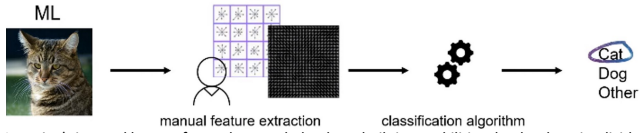
  - Usually more than one solution to a single problem
  - Ex. Solve a problem of binary classification
    - Iris dataset

Solving a problem of **binary classification**

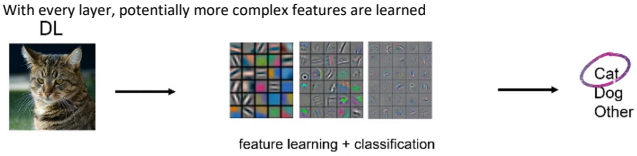


Ultimately both *can* predict a value of 0 (not iris-setosa) or 1 (is iris-setosa)

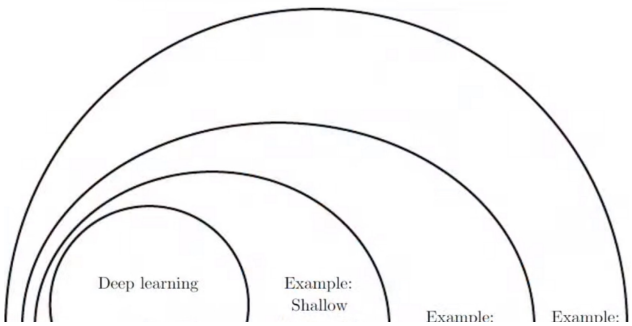
    - Classical ML trains a model based on regression that finds coefficients
    - Deep Learning trains a model (directed graph) that gets a result on classification
  - One of the biggest distinction between DL and CML is its inner capabilities of learning new features
    - Classical ML is a manual process to extract new features (feature engineering) then uses a selection of those features to train an algorithm to get results

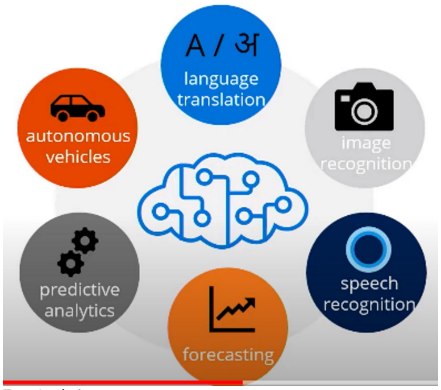


    - Deep Learning's internal layers of neural network that have built in capabilities that let them implicitly learn features without them having to be hard coded
      - With every layer, potentially more complex features are learned



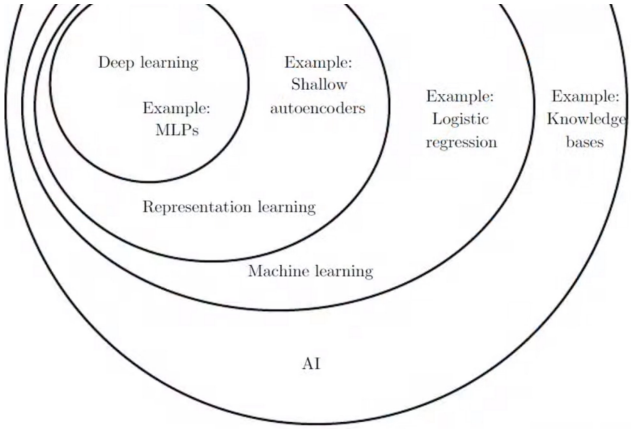
      - Deep learning has proven superior to CML in certain tasks but is more hardware intensive
  - Everything started with AI (highest level concept) with the goal of creating machines that act and think like humans
  - ML was the next subset of AI with the goal of solving the problem of learning functions from data without explicit programming
  - DL is another subset based on creating and utilizing models based on neural networks inspired by the human brain
    - Hardware resources have been limiting factors in the past for all three sets of AI
    - Exploded around 2003 when hardware advancements improved significantly\



  - Characteristics of Deep Learning
    - Highly effective in learning multi-dimensional nonlinear functions
    - Capable of handling massive amounts of training data
      - Ex. Hundreds of thousands of high res images
    - Excels with raw, unstructured data
      - Has intrinsic capabilities to learn new features at every hidden layer, can decipher raw unstructured data very well
    - Automatic feature extraction
      - Every layer can learn increasingly complex features based on its previous layers
    - Computationally expensive
      - Most complex models need lots of compute resources, specialized compute resources, GPUs, TPUs dedicated to it
  - Benefits and Applications of DL
    - Non-parametric approach taken by neural nets lets them learn arbitrarily complex functions
    - Most parametric ML models are limited by the form of the functions they can learn
    - Can learn complex patterns without explicitly seeing them
    - Ex. Image Recognition
      - Start with input data, hidden layers in deep neural network start to learn progressively and increasingly complex features from that image
      - ML model is able to do that without prior knowledge of what the image represents, it derives those patterns from the data itself
    - Effective across various "incarnations" of data (numbers, images, text)
      - Widely applicable to data of all kinds
    - Works on very large datasets
      - Other algorithms have intrinsic limitations to dataset size, prompts increased training times in some algorithms
    - Can be distributed for parallel training
      - Parallelization approach
        - Same machine, multiple cores
      - MPP approach
        - Multiple machines
    - Can learn the time-related patterns (RNN - recurrent neural networks)
      - Ex. Speech translated to text/data -> highly time related data
        - Occurs as a continuous stream of sounds /freq that are time based
    - Capable of reaching on-par performance with certain human activities
      - Ex. Speech recognition, language translation real-time
  - Applications of DL
    - Language Translation
      - DL models have proven to be equal or superior to humans
    - Image Recognition
      - Automatically extracting increasingly complex features out of images
    - Speech Recognition
    - Forecasting
    - Predicative analytics
    - Autonomous vehicles
      - 

    - Text Analytics
      - 

        - Semantics - detecting meaning and sentiment
        - Extraction of key phrases and entities and topics
        - Automate document summarizations
        - Apply clustering to identify similarity between terms and documents



- Within neural networks there is a separate distinction and DL is a subset of a wider set of neural net algorithms
  - Deep neural networks are distinct from general representation learning which may have limited numbers of layers and nodes
  - Deep learning neural networks have large numbers of hidden layers and large numbers of nodes in those layers
- Neural Networks
  - Artificial neural networks inspired by human brain but IS NOT A REPRESENTATION OF THE HUMAN BRAIN
  - Does not actually copy the human brain because of technology limitations, significantly simplifies the human brain to draw inspiration from it
  - Human brain has 86-88 billion neurons which each have a thousand trillion synaptic connections which transmits information via electrochemical signaling

- Semantics - detecting meaning and sentiment
- Extraction of key phrases and entities and topics
- Automate document summarizations
- Apply clustering to identify similarity between terms and documents

Lab: Train a Simple Neural Net

Wednesday, July 22, 2020 5:52 PM

https://introtomlsampledata.blob.core.windows.net/data/nyc-taxi/nyc-taxi-sample-data.csv

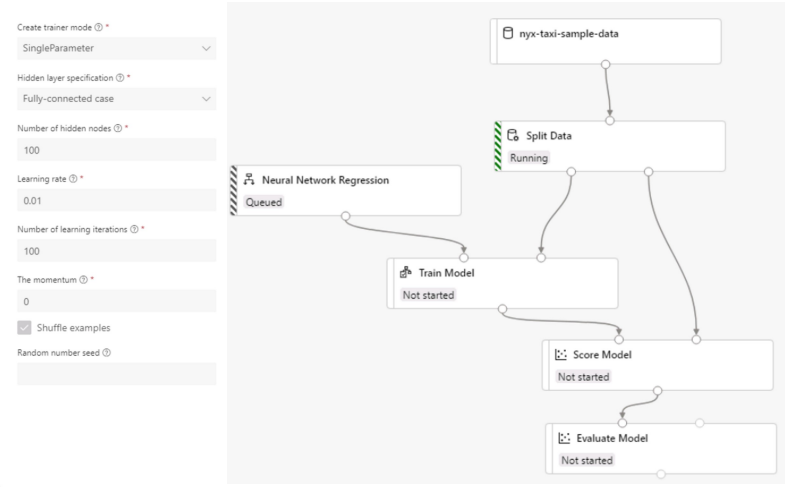
Train a simple neural net model

Although neural networks are widely known for use in deep learning and modeling complex problems such as image recognition, they are easily adapted to regression problems. Any class of statistical models can be termed a neural network if they use adaptive weights and can approximate non-linear functions of their inputs. Thus neural network regression is suited to problems where a more traditional regression model cannot fit a solution.

Neural network regression is a supervised learning method, and therefore requires a tagged dataset, which includes a label column. Because a regression model predicts a numerical value, the label column must be a numerical data type.

Lab Overview

In this lab we will be using a subset of NYC Taxi & Limousine Commission - green taxi trip records available from Azure Open Datasets. The data is enriched with holiday and weather data. Based on the enriched dataset, we will configure the prebuilt Neural Network Regression module to create a regression model using a customizable neural network algorithm. We will train the model by providing the model and the NYC taxi dataset as an input to Train Model. The trained model can then be used to predict NYC taxi fares. We will do all of this from the Azure Machine Learning designer without writing a single line of code.



Score Model result visualization

Rows 3,520 Columns 15

snowDepth	precipTime	precipDepth	temperature	totalAmount	Scored Labels
0	6	0	-1.95	12.8	11.747262
0	6	0	24.638136	9.8	10.836934
0	24	51	21.695868	8.3	8.776441
0	1	0	6.591071	12.8	14.678989
3.571429	24	9999	7.844094	11.3	14.005844
0	24	43	12.368519	7.8	9.200625
0	6	0	11.371429	21.62	23.989247
0	1	0	6.845536	15.3	17.903069
0	24	28	4.400885	8.8	9.814176
54.411765	6	0	-0.308929	9.36	7.265617
0	24	48	8.551786	15.36	9.035709
0	1	0	16.794643	10.8	14.316396

Statistics

Mean	14.468
Median	10.9224
Min	5.2512
Max	105.2257
Standard deviation	9.9319
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_Absolute_Error	Coefficient_of_Determination
2.147296	3.847992	0.139729	0.296874	0.860271

Root_Mean_Squared_Error

Statistics

Mean	3.848
Median	3.848
Min	3.848
Max	3.848
Standard deviation	-
Unique values	1
Missing values	0
Feature type	Numeric Feature

Visualizations

Specialized Model Training Cases

Thursday, July 23, 2020 3:24 PM

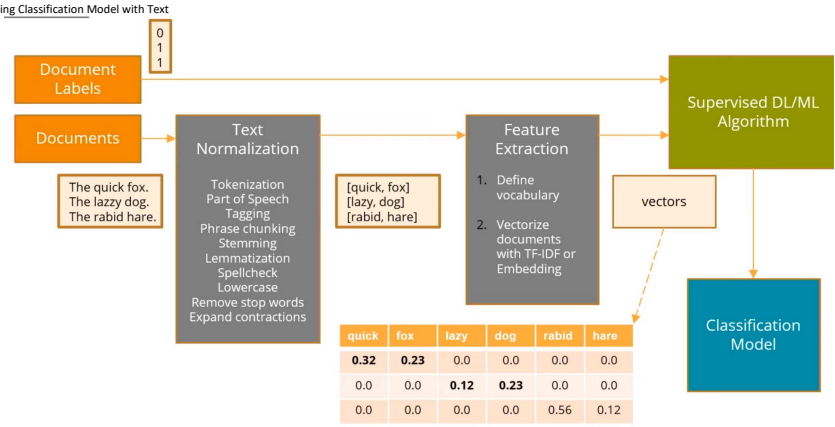
- Three Main Approaches to ML
  - Supervised Learning
    - Learns from both the inputs and expected outputs
    - Classification, Regression
      - Outputs are categorical, numerical respectively
    - Similarity Learning
      - Learns from examples using a similarity function
    - Feature Learning
    - Anomaly Detection
  - Unsupervised Learning
    - Learns from data that contains only inputs and finds hidden structures in data
    - Clustering
      - Assigns entities to clusters or groups
    - Feature Learning
    - Anomaly Detection
  - Reinforcement Learning
    - Learns how an agent should take actions in an environment to maximize a reward function
    - Ex. Intelligent roomba
      - It learns the surroundings and is capable of cleaning the entire floor because it uses reinforcement approach to map the space and then effectively cover the cleaning area
    - Markov Decision Process
      - Does not assume knowledge of an exact mathematical model
  - Semi-Supervised -> supervised + unsupervised
- Three Major Classes of Training
  - Classification, Regression, Clustering
- Specialized Cases of Model Training
  - Specific variations of generic classes that can be treated individually
  - Resulting combination of approaches is special enough that its treated as its own specific area of ML
  - Similarity Learning
    - Supervised approach that aims to learn based on a similarity function, widely used in recommendation systems
  - Text Classification
    - Special case of a supervised algorithm (classification)
    - Has a lot of applications in a lot of cases, warrants its own category
  - Feature Learning
    - Supervised (classification), unsupervised (clustering)
  - Anomaly Detection
    - Supervised (classification), unsupervised (clustering)
    - Binary classification that has a high imbalance between number of cases in each type
  - Forecasting
    - Supervised
    - Predicting future values in a time-series based scenario

Similarity Learning / Text Classification

Thursday, July 23, 2020 3:36 PM

- Similarity Learning
  - Related to classification and regression
  - Uses a different type of objective function
  - Often used in recommendation systems
    - Aim is to understand behavioral pattern of user and provide recommendation based on those patterns
  - Often used in solving verification problems (face, speech)
  - Supervised Learning Approach
    - Treated as a classification problem
      - Similarity function maps pairs of entities to finite number of similarity levels (ranging from 0-1 to anything)
      - Similarity function is discrete and outputs the level of similarity that is calculated for inputs
    - Treated as a regression problem
      - Maps pairs of entities (ex. User + product) to numerical values (similarity score)
      - Variation of this approach where an ordering measure is applied instead of an exact measure
        - Called ranking similarity learning
        - Better fit for large-scale real life problems
        - Relaxing requirement and moving towards an ordering measure significantly increases performance in situations where real-time analysis of large amounts of data is required
  - Recommender Systems
    - Application of similarity learning
    - Main aim is to recommend one or more items to users
    - User might be a person, group of persons, or any entity with item preferences
    - Approaches
      - Content-Based
        - Makes use of features for both users and items
        - User properties: age, gender, region etc.
        - Item properties: author, manufacturer etc.
      - Collaborative filtering
        - Only takes into account identifiers for users and items, don't care about properties
        - Calculate information from a matrix of ratings
          - Matrix is a preference chart between users and items
          - Ratings can be explicit (4 star rating of a movie) or implicit (takes into account history of purchases / browsing)
      -

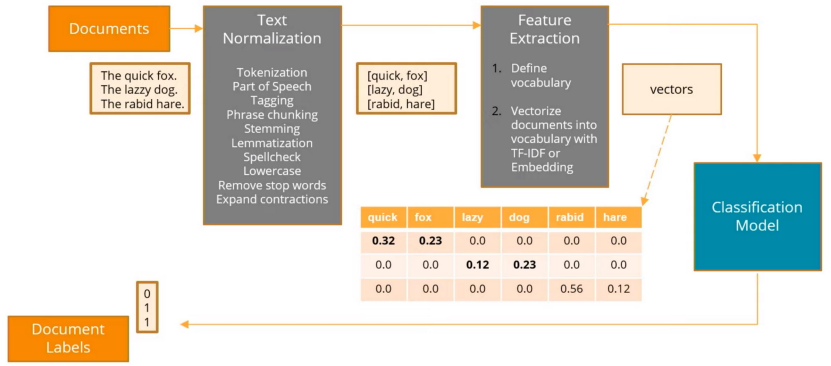
- Text Classification
  - First problem is to translate text into numerical format
    - Text embedding is that process
      - Two major forms, word embedding and scoring
      - Word Embedding -> transform every single word into a numerical vector with multiple features/dimensions and then use representation down the line
        - Enables deeper understanding of the text than scoring does
        - Twist: sentence embedding -> word embeddings combined to create a single embedding at the sentence level
      - Scoring -> aiming to calculate a score related to the importance of the particular word in the text
        - Enables understanding of the global properties such as sentiment but prevents more complex tasks like translation
  - Resulting numerical representations are then used as input to a wide range of classification algorithms
  - Training Classification Model with Text



quick	fox	lazy	dog	rabid	hare
0.32	**0.23**	0.0	0.0	0.0	0.0
0.0	0.0	**0.12**	**0.23**	0.0	0.0
0.0	0.0	0.0	0.0	0.56	0.12

    - Train model to classify a collection of "documents" as compliant or noncompliant with respect to a certain rule
    - Start with a set of "documents" and their associated labels (if its compliant or not to our 'rule')
    - Normalize documents to transform the words into a canonical form / remove stop words / etc.
    - Extract features in a 2-step process
      - Define vocabulary -> identify individual words based on frequency and record as "vocabulary"
      - Vectorize document using word embedding, TF-IDF, scoring based on your vocabulary to calculate relative importance of words in your text
      - Everything needs to be represented via numerical vectors
    - Use converted numerical dataset to train your ML/DL algorithm and end up with a classification model

  - Predicting Classification From New Text



quick	fox	lazy	dog	rabid	hare
0.32	**0.23**	0.0	0.0	0.0	0.0
0.0	0.0	**0.12**	**0.23**	0.0	0.0
0.0	0.0	0.0	0.0	0.56	0.12

    - Looking to predict document labels from document content
    - Same initial steps as previously
    - MUST use the same vocabulary set as the one used to train model
    - Results in document labels generated
  -

Lab: Train a Recommender

Thursday, July 23, 2020 3:44 PM

https://github.com/sidooms/MovieTweetings

Train a simple recommender

The main aim of a recommendation system is to recommend one or more items to users of the system. Examples of an item to be recommended, might be a movie, restaurant, book, or song. In general, the user is an entity with item preferences such as a person, a group of persons, or any other type of entity you can imagine.

There are two principal approaches to recommender systems:

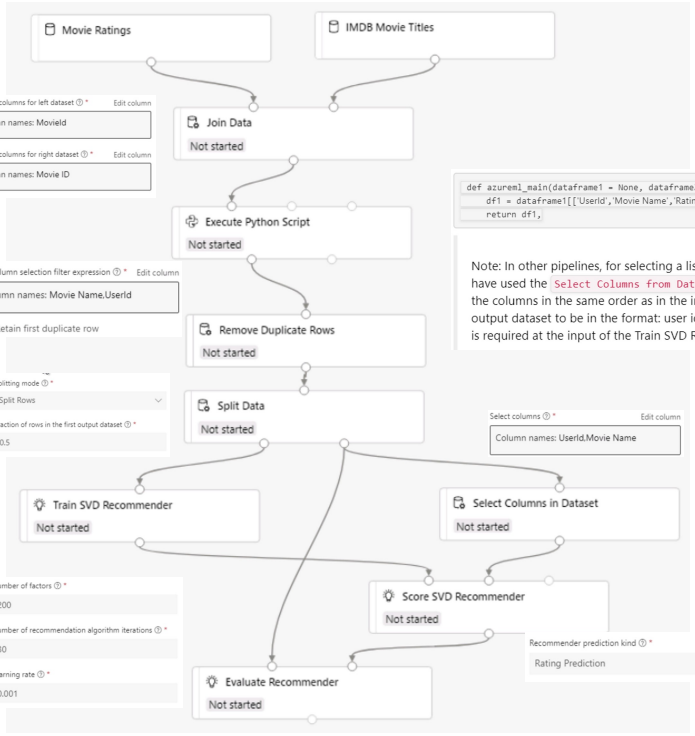
- The `content-based` approach, which makes use of features for both users and items. Users can be described by properties such as age or gender. Items can be described by properties such as the author or the manufacturer. Typical examples of content-based recommendation systems can be found on social matchmaking sites.
- The `Collaborative filtering` approach, which uses only identifiers of the users and the items. It is based on a matrix of ratings given by the users to the items. The main source of information about a user is the list the items they've rated and the similarity with other users who have rated the same items.

The SVD recommender module in Azure Machine Learning designer is based on the Single Value Decomposition algorithm. It uses identifiers of the users and the items, and a matrix of ratings given by the users to the items. It's a typical example of collaborative recommender.

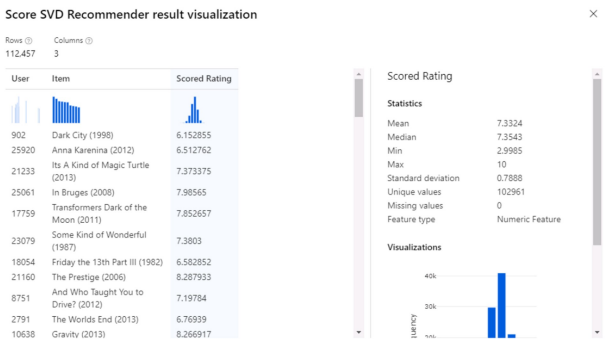
Lab Overview

In this lab, we make use of the Train SVD Recommender module available in Azure Machine Learning designer (preview), to train a movie recommender engine. We use the collaborative filtering approach: the model learns from a collection of ratings made by users on a subset of a catalog of movies. Two open datasets available in Azure Machine Learning designer are used the `IMDB Movie Titles` dataset joined on the movie identifier with the `Movie Ratings` dataset. The Movie Ratings data consists of approximately 225,000 ratings for 15,742 movies by 26,770 users, extracted from Twitter using techniques described in the original paper by Dooms, De Pessemier and Martens. The paper and data can be found on GitHub.

We will both train the engine and score new data, to demonstrate the different modes in which a recommender can be used and evaluated. The trained model will predict what rating a user will give to unseen movies, so we'll be able to recommend movies that the user is most likely to enjoy. We will do all of this from the Azure Machine Learning designer without writing a single line of code.



Note: In other pipelines, for selecting a list of columns from a dataset, we could have used the `Select Columns from Dataset` prebuilt module. This one returns the columns in the same order as in the input dataset. This time we need the output dataset to be in the format: user id, movie name, rating.This column order is required at the input of the Train SVD Recommender module.

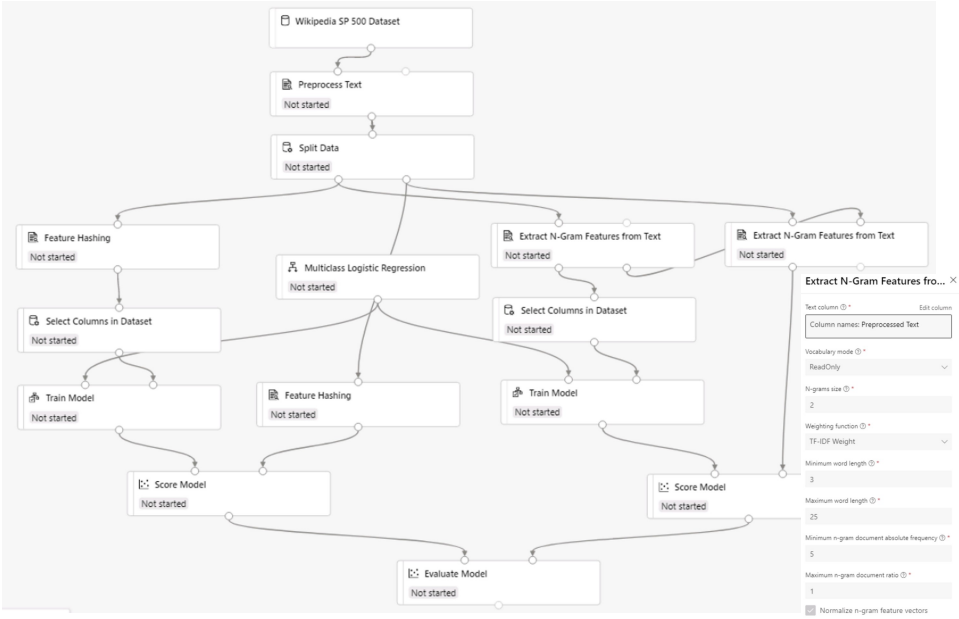


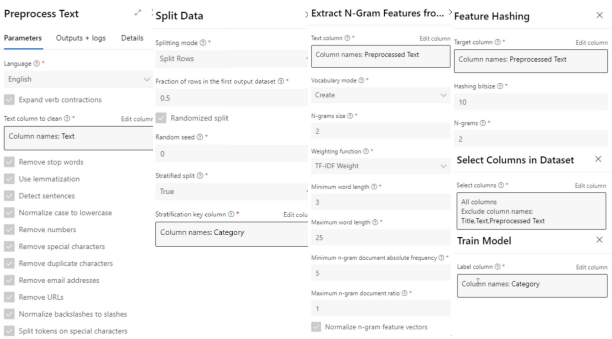
Train a simple text classifier

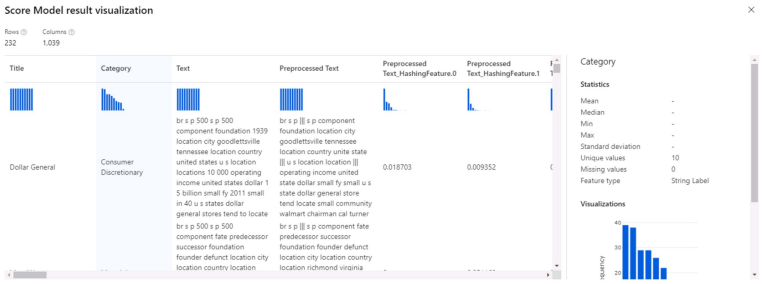
In text classification scenarios, the goal is to assign a piece of text, such as a document, a news article, a search query, an email, a tweet, support tickets, customer feedback, user product review, to predefined classes or categories. Some examples of text classification applications are: categorizing newspaper articles into topics, organizing web pages into hierarchical categories, spam email filtering, sentiment analysis, predicting user intent from search queries, support tickets routing, and customer feedback analysis.

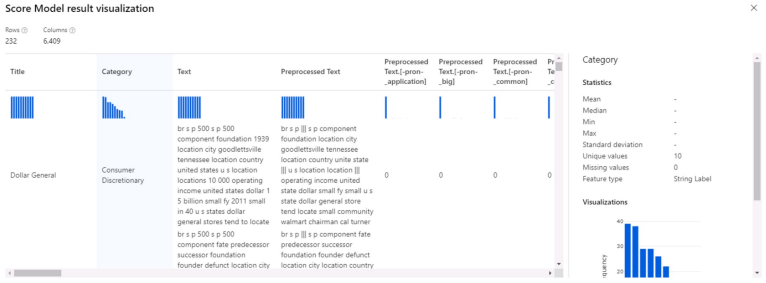
Lab Overview

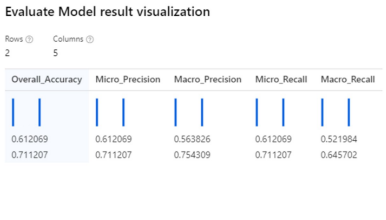
In this lab we demonstrate how to use text analytics modules available in Azure Machine Learning designer (preview) to build a simple text classification pipeline. We will create a training pipeline and initialize a **multiclass logistic regression classifier** to predict the company category with Wikipedia SP 500 dataset derived from Wikipedia. The dataset manages articles of each S&P 500 company. Before uploading to Azure Machine Learning designer, the dataset was processed as follows: extracted text content for each specific company, removed wiki formatting, removed non-alphanumeric characters, converted all text to lowercase, known company categories added. Articles could not be found for some companies, so that's why the number of records is less than 500.

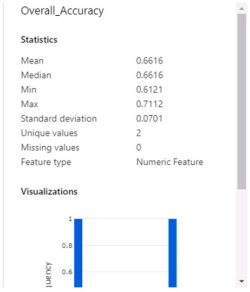












Feature Learning / Anomaly Detection / Forecasting

Friday, July 24, 2020 2:07 PM

- Feature Learning
  - Technique that derives/calculates new features in your dataset
    - Used to transform inputs into other inputs that are more useful
  - Having correct input features is a prerequisite to having high quality ML models
  - Supervised Approaches
    - New features learned using data that has been labeled
    - Ex. Datasets with multiple categorical features with high cardinality
      - If you embed category (1,2,3) you introduce bias implicitly because it's ordered
      - Create new column for each category then use 0,1 if it fits but doesn't work for large numbers of features with large number of options
    - Ex. Image Classification
      - Form of data is not suited for classification task
      - Feature learning occurs automatically within hidden layers of neural net
  - Unsupervised Approaches
    - Learning features without labeled input data
    - Clustering = form of feature learning
      - split input data into clusters and assign cluster identifier, natural feature creation
    - PCA (Principle Component Analysis)
      - Statistical approach more than a ML approach
    - Independent component analysis
    - Autoencoder (Deep Learning)
    - Matrix Factorization
  - Applications of Feature Learning
    - Image Classification
      - DL models are capable of automatically learning new features in hidden layers
    - Image Search
      - Same approach as above but for search, not classification
    - Feature Embedding (categorical features with high cardinality)
      - Lots of options per category column
    - Convolutional Neural Networks (CNN)
      - Special case of DL neural network with a combination of hidden layers that have specific abilities in learning hidden features
      - Typical CNN has two types of hidden layers
        - Convolutional layers
        - Max Pooling layer
        - Output layer is a dense layer that combines outputs for final solution
      - Image Classification
        - Step 1: learn local patterns
          - Translation invariant: once model learns to recognize a pattern locally in the image, it can recognize it elsewhere too
          - Different from DCNs (Densely Connected Neural Networks) who learns global patterns which requires retraining for application elsewhere
          - Can learn hierarchies of patterns
          - Typical CNNs have multiple sets of Conv/Max-Pool layer stacks which allow for increasingly complex pattern recognition and learning
          - Ex. Human,
            - Learning local patterns like eyes, ears, nose
            - Learning facial structure is a more complex pattern than above
            - Learning human face patterns is a more complex pattern than above
            - 1000 human faces in a picture is a local image that can be applied elsewhere
          - Learns local patterns by using a matrix applied over input data
            - 3x3 matrix called a kernel
            - Well known kernels use for edge detection, blur, sharpening etc.
        - Step 2: down-sample and make translation invariant
          - Max-Pooling layer handles step 2
          - Max-Pooling layer uses a special kind of kernel and outputs a representation that is translation invariant
          - Allows object / face recognition even if image is taken from different angles
        - Step 3: Densely connect all outputs to learn classification
      - Multiple stacks of conv and m-p layers with tons of nodes in each layer would exist in a real CNN
    - Autoencoder
      - Special type of neural network used for unsupervised learning
      - Most important feature: trained to reproduce its own inputs as accurately as possible
      - Don't need labels on data since autoencoder seeks to reproduce its inputs
      - Typical shape has a large number of inputs to start but each layer decreases in size until min-width is reached then layers increases in size until it reaches same size as input
      - Middle layer with smallest width has special significance
        - It marks border between encoder (left of layer) and the decoder (right of layer)
        - Produces the **feature vector** (compressed representation of the inputs)
          - Compressed is dimensionally reduced, not reduced in size
          - Shines in image translation to N-dimensional vectors, not size compression
      - Left half (encoder) can be used to embed inputs into feature vectors
        - Distance measure is used to identify similar input images
      - Ex. Set of images that contain atoms
        - As the autoencoder is trained, you get an encoder that can encoder images with atoms in it
        - Feed new images into encoder to get its feature vector(s)
        - Calculate distance between the feature vector and the feature vectors of training images
          - If distance is below certain threshold, there is a high probability that your new image contains atoms

- Anomaly Detection
  - Anomalies: data that deviates significantly from the norm
  - Can be result of bad data, unusual behavior or important exceptions to typical trends
  - Anomaly detection is ML technique concerned with finding these data points
    - Given a set of entities, train a model to detect anomalies
    - Usually, # of abnormal entities << normal entities
    - Anomaly detection difficult because typical ML approaches / algorithms used for classification, can't deliver results because not enough instances of fraud
    - Ex. Fraud detection -> millions of valid transactions, much less fraudulent transactions
  - Supervised approaches
    - Binary classification problem -> either normal or anomaly
    - Uses a labeled training set with normal and anomaly data
      - Difficult because you don't always have a track record of abnormal data
  - Unsupervised approaches
    - Binary clustering problem -> either normal or anomaly
    - Relies on a dataset with no labels available, have no idea what is normal and what is abnormal
    - Model needs to be able to define 'normal' and 'abnormal' then detect that pattern in new items
  - Applications of Anomaly Detection
    - Condition monitoring and failure prevention
      - Ex. Industrial maintenance
    - Fraud detection
      - Ex. Credit card charges
    - Intrusion detection
      - Ex. Network communication, detecting attacks
    - Anti-virus and anti-malware protection
    - Data preparation
      - Outlier detection - remove outliers that may skew model
    - Example: machinery maintenance
      - Approach: train autoencoder to recognize 'normality'
      - Training process
        - Given a set of machine parameters (readings from sensors etc)
        - Pass inputs through autoencoder (will try to reproduce inputs)
        - Train autoencoder with sensor readings of normal operations
        - Apply autoencoder to real stream of data
          - Will be able to identify failure when it occurs
          - Threshold value to detect anomalies is the highest acceptable value for normality (mean average error in ex.)
          - Goal is to look for signs of failure before it happens (the diminishing and then spike in graph)

- Forecasting
  - Predicting the next occurrences in a series of time-based events (or any orderable data)
  - Handles the class of problems that deals with predictions in the context of orderable datasets
  - More than one property can exist for data points
    - One (numerical) will be set as target property
    - Other properties are used to enhance the accuracy of the model in other ways (ensemble learning)
  - Algorithms an approaches
    - ARIMA: AutoRegressive Integrated Moving Average
      - Evolution of ARMA, designed to provide description for random time-based processes
    - Multi-variate Regression
      - Task is to predict a numerical value of next iteration of time series
      - Can take into account multiple properties of entities in time-series
        - Location etc.
      - Ex. Values for a sensor reading once a day-> predict reading for next day
  - Prophet
    - Developed by Facebook
    - Works best with time-series that have strong seasonal effects
      - Ex. Sales for brick-and-mortar pharmacy
      - Highly seasonal products (flu meds)
    - Capable of taking seasonal effects into account
  - Temporal Convolutional Network (TCN)
    - ForecastTCN
      - Developed by Microsoft
    - Special kind of convolutional network
      - One-dimensional because time-series
    - Capable of exhibiting longer memory than other approaches
  - Recurrent Neural Networks (RNNs)
    - Class of networks that have additional connections between nodes
      - Classical structures is a feed-forward network
      - Adding backwards-facing connections creates cycles
    - Can effectively learn time-based patterns
    - Long-short term memory, gated recurring unit etc.
    - Revolutionized speech recognition, machine translation, fields highly related to time-based data

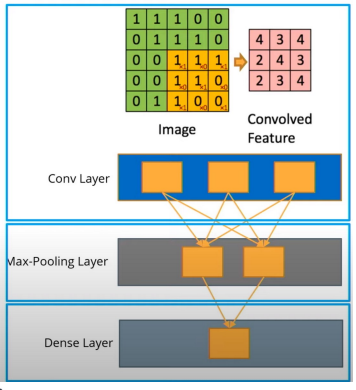
Image classification with CNNs

CNN =
Convolutional Neural Network

Step 1 – learn local patterns

Step 2 – down-sample and make translation invariant

Step 3 – densely connect all outputs to learn classification



Train an autoencoder to recognize 'normality'

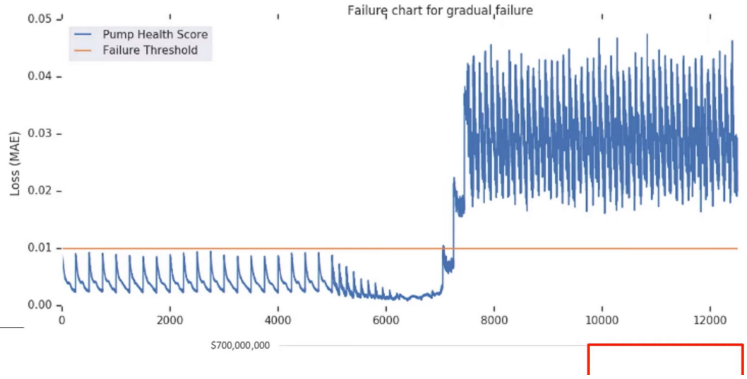


Image search with autoencoders

Image search with autoencoders

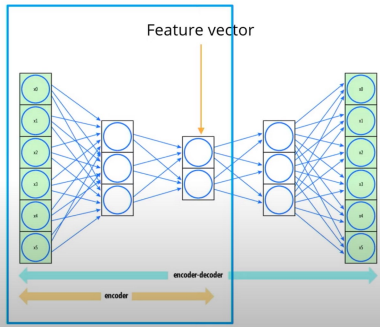
Autoencoder =
neural net for unsupervised learning

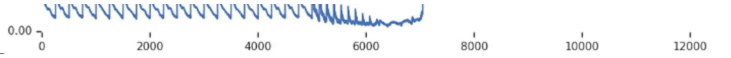
Trains to reproduce its inputs

Produces the feature vector

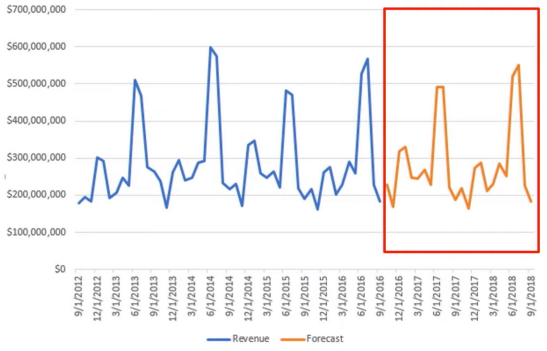
The left half (the encoder) can be used to embed inputs into feature vectors

A distance measure is used to identify similar input images





Given N time-ordered single-valued data points, predict the next M points



Lab: Forecasting

Monday, July 27, 2020 10:14 PM

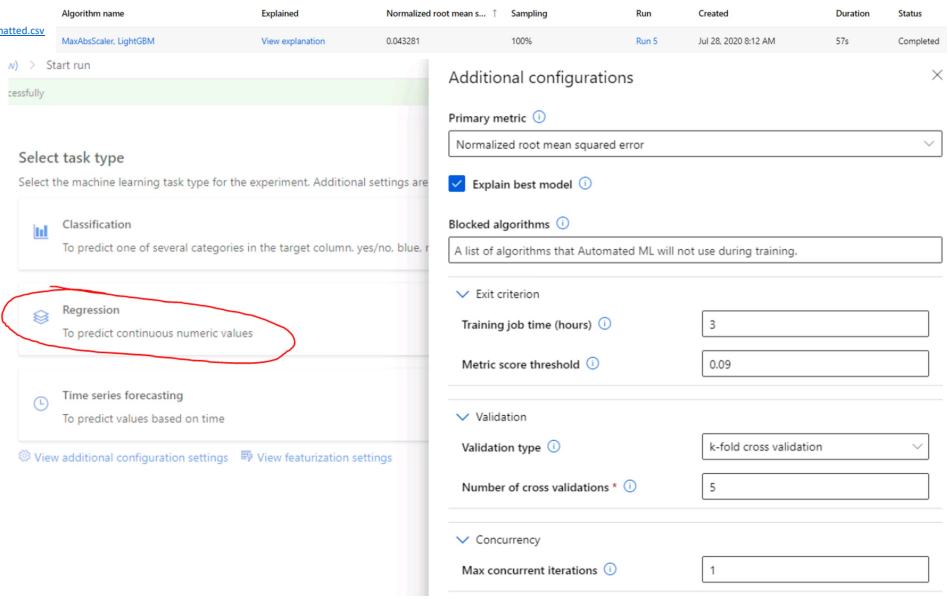
https://introtomlsampledata.blob.core.windows.net/data/battery-lifetime/training-formatted.csv

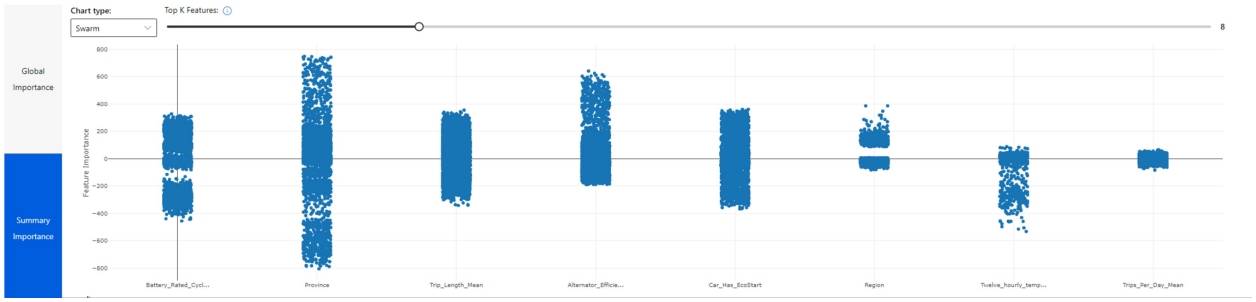
Train a time-series forecasting model using Automated Machine Learning

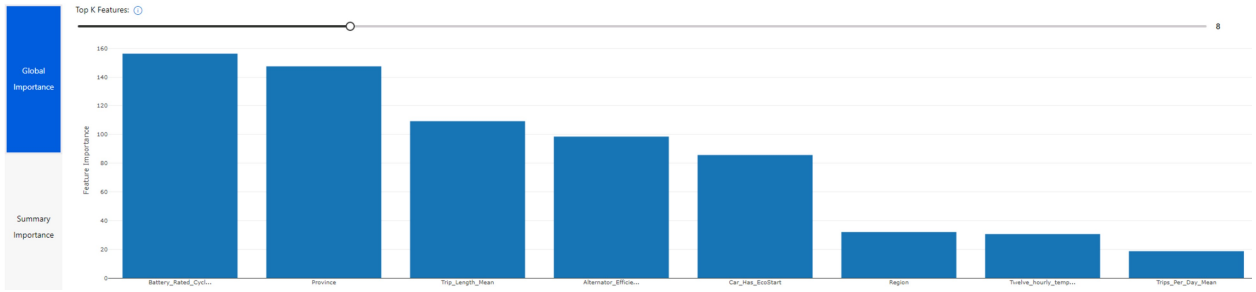
Lab Overview

In this lab you will learn how the Automated Machine Learning capability in Azure Machine Learning (AML) can be used for the life cycle management of the manufactured vehicles and how AML helps in creation of better vehicle maintenance plans. To accomplish this, you will train a Linear Regression model to predict the number of days until battery failure using Automated Machine Learning available in AML studio.

In the models list, notice at the top the iteration with the best **normalized root mean square error** score. Note that the normalized root mean square error measures the error between the predicted value and actual value. In this case, the model with the lowest normalized root mean square error is the best model.







automlregression