AI Definition and History

Agenda

- Machine learning (ML) definitions
- History
- Trends
- Building blocks

Machine Learning Definitions

- Science of getting computers to learn from data and then act without being specifically programmed
- Deals with representation and generalization of models that can perform well on unseen data and reliably predict labels on unknown events
- Although it is a hot topic right now, it has a long and varied history
- Subset of artificial intelligence (AI)
    - AI deals with how to get machines to mimic human intelligence
    - ML deals with how to get machines to learn

History

- 1950s–60s
    - Perceptron
        - Binary classification algorithm modeled after a biological neuron
    - NNet
        - First feed-forward neural network implemented with a single hidden layer
        - Helped solve multinomial log-linear models
- 1970s–80s
    - ID3 algorithm for decision tree learning
    - COBWEB

History (cont.)

- 1990s–2000s
  - Development of support vector machines
    - Supervised learning models (predicting a labelled outcome)
    - Analyze input data and learn patterns
    - Predict label of new unseen data
  - Latest wave of deep learning (2006)
    - Apply layers of calculations to data to learn higher-level, more abstract concepts from previous simpler ones
    - Does very well on unstructured data such as pictures and audio

AI Definition and History

The End

Why Machine Learning?

Why Machine Learning?

Standard programming

1. Look for patterns
2. Write a program to detect patterns
3. In a real world problem, the rule set will get very complicated and hard to maintain

Machine learning

1. Let the machine determine what the patterns/predictors are by feeding it data
2. Provide it a new input and let it predict the outcome
3. As patterns change, feed it more data

Why Machine Learning?

The End

ML Trends

Trends

- Deep learning
  - Solve more complex problems by combining layers that start simple and get more complex
  - Specific subset of machine learning
- Big data
  - Distributed storage and parallel processing is bringing more data to the table
  - Cloud providers are now offering seamless data-center level distributed storage at high speeds
- Cloud-based machine learning resources
  - ML expertise not required
  - Offer trained and tuned models as an application programming interface (API) service
  - Automated machine learning solutions—you provide the data, they automatically train and tune a solution for you

Deep Learning

- Deep learning has seen waves of development
  - Cybernetics in the 1940s–60s
  - Connectionism in the 80s–90s
  - Current deep learning wave starting in 2006
- Biological neuroscience and the connection of neurons was an initial influence on designing neural nets but is not an attempt to simulate how the human brain works (we don’t know enough)
- Larger networks are able to achieve higher accuracy on more complex tasks
  - Deep nets have doubled in size every 2.4 years as hardware and software is optimized to use them and more data becomes available to feed them
- Rule of thumb
  - A supervised (labelled) deep learning algorithm will generally achieve acceptable performance with 5,000 labeled examples per category
  - And match or exceed human performance with a data set containing at least 10 million labelled examples

Exercise: answer some questions to test your understanding of ML

ML Trends

The End

Building Blocks

Building Blocks

- As we stated earlier machine learning is loosely based on how humans learn
- Algorithms we will be exploring have their basis in:
  - Statistics
  - Linear algebra
  - Calculus
  - Probability
  - Information theory
  - Numerical optimization
- We will cover these as we need to in this class

Statistics

- Statistics is a mathematical discipline that explores relationships between variables.
- Correlation and confidence are important in traditional statistics.
- Machine learning (ML) uses statistics as a foundation.
- Optimization and accuracy are emphasized in ML, though.

Linear Algebra

- The input to our ML models are going to be represented by numeric feature vectors.
- In the case of supervised learning, we will also have a label vector.
- Linear algebra helps us figure out how to convert our input into the right format for the ML model.
- This is an important aspect of preprocessing.

Calculus

- In machine learning, we have an objective function (y=f(x)).
- Our goal is to minimize, or optimize, f(x).
- For that reason, our objective function is quite often called a cost function.
- If we plot the value x on the horizontal axis and f(x) on the vertical axis, we can determine the slope of the line at any two points by taking the derivative.
- This is called gradient descent and is a very common optimization function in ML.

Probability

- Mathematical discipline that explores how to describe and deal with uncertainty
- Random variables can be:
  - Discrete, which is stateful
  - Continuous, which is real-valued
- As you experiment, you come up with a probability distribution of how likely a random variable is to be in a particular state
- A probability distribution of a discrete variable is described via a probability mass function
- A probability distribution of a continuous variable is described via a probability density function calculated by an integral, or area under the curve, to borrow from calculus again
- In ML, we usually have a lot of variables
- A probability distribution with many variables is called a joint probability distribution

Information Theory

- Applied mathematics dealing with how to pull a transmission from a noisy channel
- Important concepts borrowed from information theory include:
  - Entropy: a measure of information in a single random variable
  - Mutual information: a measure of information in common between two random variables
- These measures are derived from the probability distributions of the variables

Building Blocks

The End

Linear Algebra Basics

Breakout: Linear Algebra Basics

- How do you write a linear equation?
- How can you represent a linear equation in matrix notation?
- What is a scalar?
- What is a vector?
- What is a matrix?
- What is a tensor?

Recall: How Do You Write a Linear Equation?

What Is a Scalar?

What Is a Vector?

What Is a Matrix?

Tools

- We will be implementing machine learning in Python in this course
- Why Python?
  - Popular programming language
  - Lots of libraries for statistics, linear algebra, and scientific analysis
  - Object oriented and functional
  - Interpreted
- Specific libraries we will use in this course include:
  - NumPy
  - Pandas
  - MatplotLib
  - scikit-learn
  - TensorFlow
- We will introduce the classes and concepts of these libraries as you need them

Linear Algebra Basics

The End

Linear Algebra Operations

Linear Algebra Operations

- What is the transpose?
- How do you do matrix addition?
- How do you do scalar multiplication and addition?

What Is the Transpose of a Matrix?

How Does Matrix Addition Work?

Matrix Multiplication

Working With Arrays in NumPy

Breakout: NumPy and the ndarray

- https://docs.scipy.org/doc/numpy/index.html
- How do you call the NumPy library?
- How do you implement a vector in NumPy?
- What are some of the common array methods available in NumPy?
- How do you do conditional selection in a NumPy ndarray?
- How do you access elements of a vector? Of an array?
- How do you do matrix multiplication in NumPy?

Linear Algebra Operations

The End

Machine Learning Types

Machine Learning Types

- Most ML tasks can be categorized as either supervised or unsupervised based on the way the input data is structured
- Supervised ML
  - Takes a labelled set of features as input (the set of data we get these features and labels from is called a training set)
  - Goal is to predict label of a new feature set without a label
- Supervised ML tasks
  - Classification
  - Regression

Unsupervised ML

- Unsupervised learning:
  - Has no label
  - Looks for unknown patterns in the data
- Unsupervised ML tasks include:
  - Clustering
  - Outlier detection
  - Affinity analysis

Other Types of ML

- Reinforcement learning
  - Based on reward and punishment
  - The agent (model) needs to learn how to get the most reward over time
- Recommendation
  - Predict how a user would rate or prefer a given item
  - Common techniques include:
    - Item-based
    - User-based
    - Collaborative filtering
    - Content-based filtering

Machine Learning Types

The End

Supervised Machine Learning Tasks

Common ML Tasks: Classification

- Type: supervised
- Goal: classify new data into a set of categories based on existing data whose category is already known
- Input: examples consisting of numerical features and a discrete class label for them
- Common output: prediction of class based on a new example of features without a label
- Techniques and approaches to solve
  - Naïve Bayes
  - Support vector classifier
  - Logistic regression
  - TensorFlow 1.x—deep neural net classifier (tf.estimator.dnnclassifier)
  - TensorFlow 2.x—keras sequential model (tf.keras.sequential) with appropriate cross-entropy loss in the compile method
- Example
  - Is a flight late or not? Two answers—binary solution
  - Is a flight early, late, or on-time? More than two answers—multi-class solution

Common ML Tasks: Regression

- Type: supervised
- Goal: model and analyze correlation between two or more variables
- Input: examples with features/attributes and a continuous label
- Common output: prediction of continuous number given new example consisting of features/attributes with no label included
- Techniques and approaches to solve
  - Linear regression
  - Decision trees
  - Support vector regressor
  - TensorFlow 1.x—deep neural net regressor (tf.estimator.dnnregressor)
  - TensorFlow 2.x—keras sequential model (tf.keras.sequential) with appropriate loss (MSE) in the compile method
- Example
  - How late is the flight (5 minutes, 10 minutes late)?

Supervised Machine Learning Tasks

The End

Unsupervised Machine Learning Tasks

Common ML Tasks: Clustering

- Type: unsupervised
- Goal: find naturally occurring patterns in data
- Input: unlabeled data set
- Common outputs
  - Detecting natural groups
  - Preprocessing step to determine class labels (results can then be fed to a supervised task)
- Techniques and approaches to solve
  - K-means
  - Spectral clustering
  - Hierarchical clustering
  - Self organizing maps
- Example: document classification based on tags, topics, and content of document

Common ML Tasks: Outlier Detection

- Type: unsupervised
- Goal: determine what points don’t belong to a cluster
- Input: unlabeled data set
- Common output
  - Identified outliers (often used in preprocessing so outliers are eliminated before doing more analysis)
- Techniques and approaches to solve
  - Local outlier factor
  - Nearest neighbor
  - One-class support vector machine
- Example: What is a fraudulent credit card transaction?

Common ML Tasks: Affinity Analysis

- Type: unsupervised
- Goal: discover co-occurrence among actions of groups
- Input: unlabeled data where each example is a session or transaction
- Common output: association rules of when items appear together
- Techniques and approaches to solve
  - Apriori algorithm
  - Frequent pattern growth algorithm
- Example: market basket analysis in grocery stores (the diapers and beer conundrum—urban legend or valid co-occurrence?)

Unsupervised Machine Learning Tasks

The End

What Do We Mean by Learning?

What Is Learning?

"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, measured by P, improves with experience E."

—Tom Mitchell, 1997

Let’s Look at Experience (E)

- In the previous video, we covered ML types and tasks
- Each ML task had a set of inputs
  - Classification inputs—example: features/attributes and an identified discrete label
  - Regression inputs—example: features/attributes and an identified continuous label
  - Clustering inputs—example: features/attributes and no label
- Let’s map this, our new definition
  - The set of inputs is our experience (E)

Let’s Look at Our Task (T)

- The output (T) is based on a function applied to our inputs (E):

  output = f(inputs)

- Our ML model applies a function to the inputs to approximate the output
- Great! We have approximated the output based on the inputs; we are done, right?
  - Wait, what about improving with experience?
  - Don’t forget performance (P)

Determining Performance (P)

- Since the result we want in classification is to come up with a good approximation of what class it belongs to, we have to *train the model* with a bunch of examples (feature inputs with a known label).
- The process of feeding a model data from the training set is also called *fitting the model.*
- As we feed the model more examples, we want it to minimize the error between the predicted value of the outcome and the true value of the outcome provided in the training data.
- This is our performance measure (P).
- The definition of the error measure is called a *cost function.*
- The way we minimize the cost function is via an *optimization algorithm.*

Practical Example

- In the case of linear regression, the formula to draw a straight line is:

  y = wx + b
- x here is our matrix of feature inputs
- w and b are parameters of the model and are what we change to come up with our best estimate based on minimizing error
- The cost function in this case is mean squared error (MSE)

  $1/n\sum(truth - predicted)^2$
- One of the most common optimizers to use is called stochastic gradient descent

Model Evaluation

- When do we stop training?
- Up to this point, we have been tweaking the model based on the training data.
- What we want to do, however, is see how it generalizes to unseen data.
- To do this, we use a test set.
- Instead of using our performance measure to tweak the parameters of the model, we now use it to see how well our model fits to the new data we haven’t seen before.

Prediction

- Once we have trained our model, we can use it to predict the outcome based on a feature input without a label
- For classification, our prediction is:
  - Is the flight going to be late: late or not late, yes or no?
- For regression, our prediction is:
  - 15 minutes late, 20 minutes late, −5 minutes late (equals early)

Comparing ML Models

- Quite often there are different approaches to solving a problem
    - Classification approaches: logistics regression, naive Bayes, support vectors
- The approaches we listed are models that take the input examples and estimate the outcome
- Since the model is estimating the outcome, or inferring the outcome from what it has learned, models are often called estimators
- Different models have pros and cons and performance considerations based on the input data

Summary

- ML models are given a particular input.
- Based on the input, they estimate a particular output.
- The output is compared against a known label as part of the objective function.
- An optimization algorithm is used to minimize parameters of the model to improve the prediction.

What Do We Mean by Learning?

The End

What Could Go Wrong?

What Can Go Wrong?

- Data issues
    - Not enough data
    - Nonrepresentative data
    - Poor quality data
    - Not the right features
- Algorithm issues
    - Underfitting/overfitting
    - No free lunch theorem

Not Enough Data

- With modern ML, the more data you have the better your model will perform
- As shown by:
  - Banko and Brill (2001), “Learning Curves for Confusion Set Disambiguation”
  - Peter Norvig et al. (2009), “The Unreasonable Effectiveness of Data”
- They were able to show accuracy improved across the board just by providing more data to several different types of models

Nonrepresentative Data

- You need enough examples of classes and cases that you want to generalize to.
- Your method of sampling could be the issue.
- Sometimes picking too simple of a model will keep you from seeing patterns in the data that the model cannot differentiate.

Poor Quality Data

- It is important to look at your data to take care of:
  - Errors
  - Outliers
  - Noise
- Get rid of outliers that might skew your results
- Use an imputer to come up with missing values

Irrelevant Features

- You need to be smart about the features you feed to your dataset
- Feature engineering involves:
  - Selecting the most useful features
  - Combining features to produce more useful ones
  - Creating new features/gathering new data if your model is stuck

Underfitting and Overfitting

- Underfitting and overfitting
  - When we do model evaluation a couple of things can happen
    1. Our cost function is really low on our training data but really high on our unseen data
       - We’ve learned too much, or **overfit** on our training data
       - Overfitting tends to happen when your model is too complex
       - Regularization is how you can constrain your model and make sure it doesn’t get too complex
    2. Our cost function is really high on our training data
       - We haven’t learned enough, or **underfit** on our training data
       - Underfitting tends to happen when your model is not complex enough
       - You might want to pick a more powerful model or do some feature engineering to combat underfitting

No Free Lunch Theorem: David Wolpert (1996)

- Most of our examples in this lecture have been about linear models
- When you use a linear model you assume the data is fundamentally linear
- There is no model guaranteed to work better than any other if you make no assumptions
- This is why studying your input data becomes very important

Tuning Your Model With Hyper-Parameters

- We’ve covered parameters that the model modifies to minimize the cost function
- There are also parameters that set how the algorithm is structured
- These are called hyper-parameters
- They are set as you initialize your model, before training
- A lot of time can be spent tuning your hyper-parameters to increase performance and avoid underfitting

What Could Go Wrong?

The End