lecture1-introduction

November 14, 2021

1 Lecture 1: Introduction to Machine Learning

1.0.1 Applied Machine Learning

Volodymyr Kuleshov Cornell Tech

2 Welcome to Applied Machine Learning!

Machine learning is one of today's most exciting emerging technologies.

In this course, you will learn what machine learning is, what are the most important techniques in machine learning, and how to apply them to solve problems in the real world.

3 Part 1: What is Machine Learning?

We hear a lot about machine learning (or ML for short) in the news.

But what is it, really?

4 ML in Everyday Life: Search Engines

You use machine learninng every day when use a search engine.

5 ML in Everyday Life: Personal Assitants

Machine learning also powers the speech recognition, question answering and other intelligent capabilities of smartphone assistants like Apple Siri.

6 ML in Everyday Life: Spam/Fraud Detection

Machine learning is used in every spam filter, such as in Gmail.

ML systems are also used by credit card companies and banks to automatically detect fraudulent behavior.

7 ML in Everyday Life: Self-Driving Cars

One of the most exciting and cutting-edge uses of machine learning algorithms is in autonomous vehicles.

8 A Definition of Machine Learning

In 1959, Arthur Samuel defined machine learning as follows.

> Machine learning is a field of study that gives computers the ability to learn without being explicitly programmed.

What does “learn” and “explicitly programmed” mean here? Let’s look at an example.

9 An Example: Self Driving Cars

A self-driving car system uses dozens of components that include detection of cars, pedestrians, and other objects.

10 Self Driving Cars: A Rule-Based Algorithm

One way to build a detection system is to write down rules.

[2]:

```
# pseudocode example for a rule-based classification system
object = camera.get_object()
if object.has_wheels(): # does the object have wheels?
    if len(object.wheels) == 4: return "Car" # four wheels => car
    elif len(object.wheels) == 2:,
        if object.seen_from_back():
            return "Car" # viewed from back, car has 2 wheels
        else:
            return "Bicycle" # normally, 2 wheels => bicycle
return "Unknown" # no wheels? we don't know what it is
```

In practice, it’s almost impossible for a human to specify all the edge cases.

11 Self Driving Cars: An ML Approach

The machine learning approach is to teach a computer how to do detection by showing it many examples of different objects.

No manual programming is needed: the computer learns what defines a pedestrian or a car on its own!

12 Revisiting Our Definition of ML

> Machine learning is a field of study that gives computers the ability to learn without being explicitly programmed. (Arthur Samuel, 1959.)

This principle can be applied to countless domains: medical diagnosis, factory automation, machine translation, and many more!

13 Why Machine Learning?

Why is this approach to building software interesting?

- It lets us build practical systems for real-world applications for which other engineering approaches don't work.
- Learning is widely regarded as a key approach towards building general-purpose artificial intelligence systems.
- The science and engineering of machine learning offers insights into human intelligence.

# Part 2: Three Approaches to Machine Learning

Machine learning is broadly defined as the science of building software that has the ability to learn without being explicitly programmed.

How might we enable machines to learn? Let's look at a few examples.

14 Supervised Learning

The most common approach to machine learning is supervised learning.

1. First, we collect a dataset of labeled training examples.
2. We train a model to output accurate predictions on this dataset.
3. When the model sees new, similar data, it will also be accurate.

15 A Supervised Learning Dataset

Consider a simple dataset for supervised learning: house prices in Boston. * Each datapoint is a house. * We know its price, neighborhood, size, etc.

[13]:

```
# We will load the dataset from the sklearn ML library
from sklearn import datasets
boston = datasets.load_boston()
```

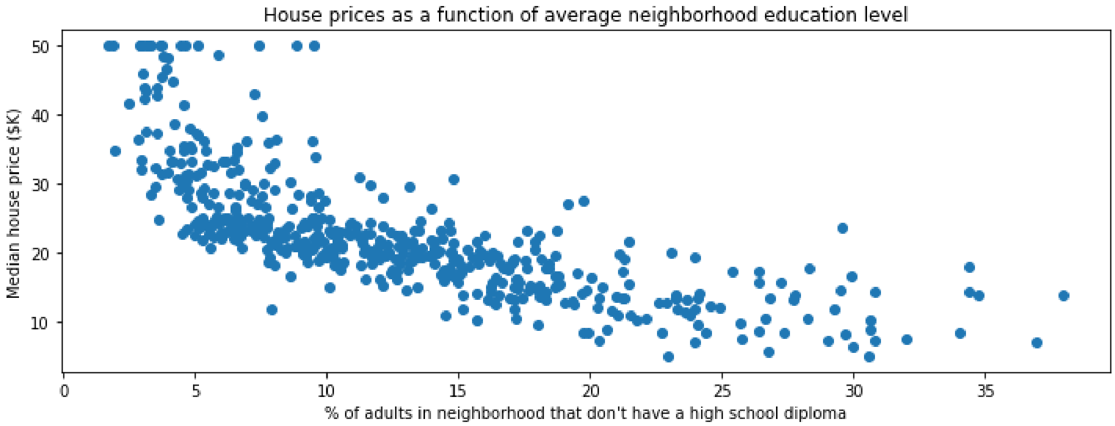
We will visualize two variables in this dataset: house price and the education level in the neighborhood.

[14]:

```
import matplotlib.pyplot as plt
plt.rcParams['figure.figsize'] = [12, 4]
plt.scatter(boston.data[:,12], boston.target)
plt.ylabel("Median house price ($K)")
plt.xlabel("% of adults in neighborhood that don't have a high school diploma")
plt.title("House prices as a function of average neighborhood education level")
```

[14]:

```
Text(0.5, 1.0, 'House prices as a function of average neighborhood education
level')
```



16 A Supervised Learning Algorithm

We can use this dataset of examples to fit a supervised learning model.

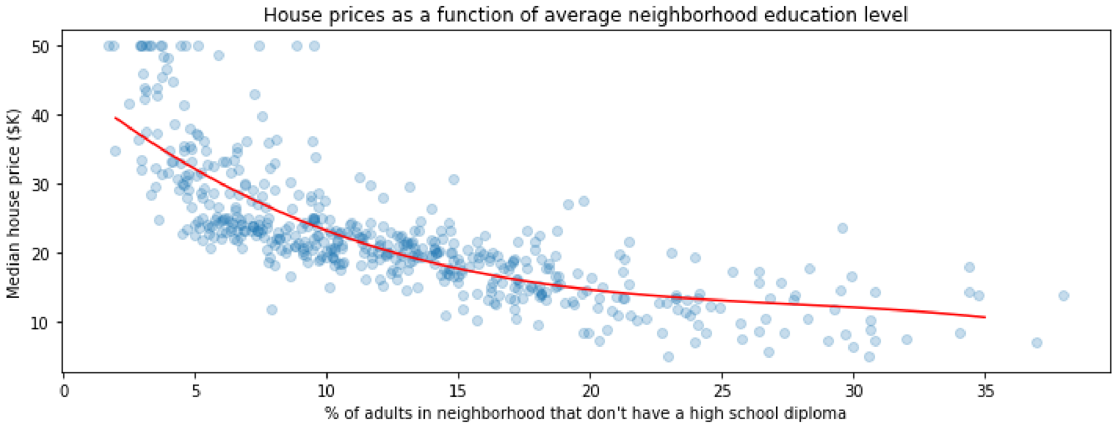
- The model maps input x (the education level) to output a y (the house price).
- It learns the mapping from our dataset of examples (x, y).

[15]:

```
import numpy as np
from sklearn.kernel_ridge import KernelRidge

# Apply a supervised learning algorithm
model = KernelRidge(alpha=1, kernel='poly')
model.fit(boston.data[:,[12]], boston.target.flatten())
predictions = model.predict(np.linspace(2, 35)[:, np.newaxis])

# Visualize the results
plt.scatter(boston.data[:,[12]], boston.target, alpha=0.25)
plt.plot(np.linspace(2, 35), predictions, c='red')
plt.ylabel("Median house price ($K)")
plt.xlabel("% of adults in neighborhood that don't have a high school diploma")
plt.title("House prices as a function of average neighborhood education level")
```

[15]:

```
Text(0.5, 1.0, 'House prices as a function of average neighborhood education
level')
```



17 Applications of Supervised Learning

Many of the most important applications of machine learning are supervised: * Classifying medical images. * Translating between pairs of languages. * Detecting objects in a self-driving car.

18 Unsupervised Learning

Here, we have a dataset *without* labels. Our goal is to learn something interesting about the structure of the data: * Clusters hidden in the dataset. * Outliers: particularly unusual and/or interesting datapoints. * Useful signal hidden in noise, e.g. human speech over a noisy phone.

19 An Unsupervised Learning Dataset

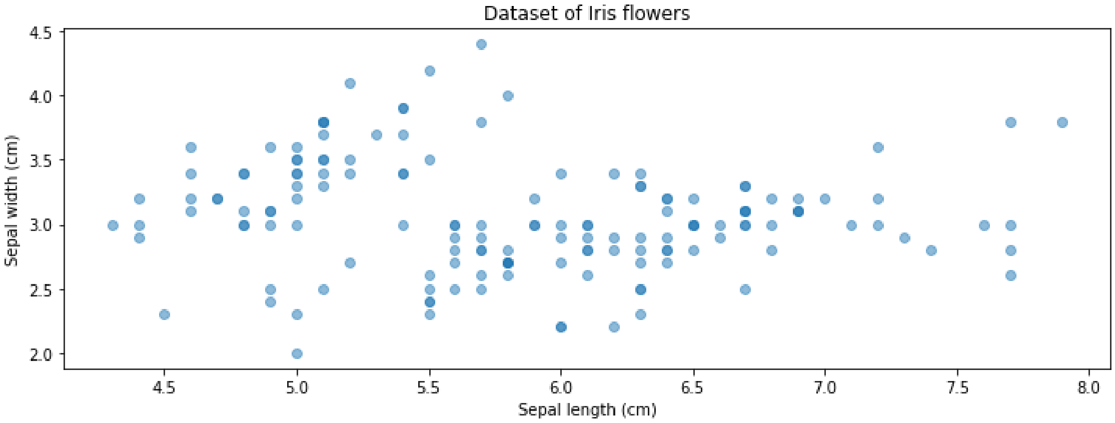
Here is a simple example of an unsupervised learning dataset: Iris flowers.

[20]:

```
# Load and visualize the Iris flower dataset
from sklearn import datasets

iris = datasets.load_iris()

plt.scatter(iris.data[:,0], iris.data[:,1], alpha=0.5)
plt.ylabel("Sepal width (cm)")
plt.xlabel("Sepal length (cm)")
plt.title("Dataset of Iris flowers")
```

[20]:

```
Text(0.5, 1.0, 'Dataset of Iris flowers')
```



20 An Unsupervised Learning Algorithm

We can use this dataset of examples to fit an unsupervised learning model. * The model defines a probability distribution over the inputs. * The probability distribution identifies multiple components (multiple peaks). * The components indicate structure in the data.

[21]:

```
# fit a Gaussian Mixture Model with three components
from sklearn import mixture
model = mixture.GaussianMixture(n_components=3, covariance_type='full')
model.fit(iris.data[:,[0,1]])
```

[21]:

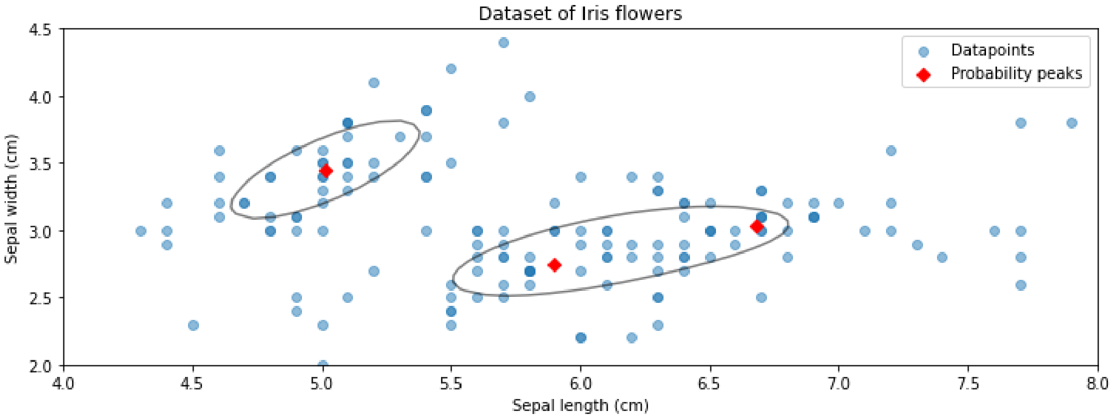
```
GaussianMixture(n_components=3)
```

[22]:

```
# display learned probabilities as a contour plot
x, y = np.linspace(4.0, 8.0), np.linspace(2.0, 4.5)
X, Y = np.meshgrid(x, y)
Z = -model.score_samples(np.array([X.ravel(), Y.ravel()]).T).reshape(X.shape)
plt.contour(X, Y, Z, levels=np.logspace(0, 10, 1), cmap="gray", alpha=0.5)
plt.scatter(iris.data[:,0], iris.data[:,1], alpha=0.5)
plt.scatter(model.means_[:,0], model.means_[:,1], marker='D', c='r')
plt.ylabel("Sepal width (cm)")
plt.xlabel("Sepal length (cm)")
plt.title("Dataset of Iris flowers")
plt.legend(['Datapoints', 'Probability peaks'])
```

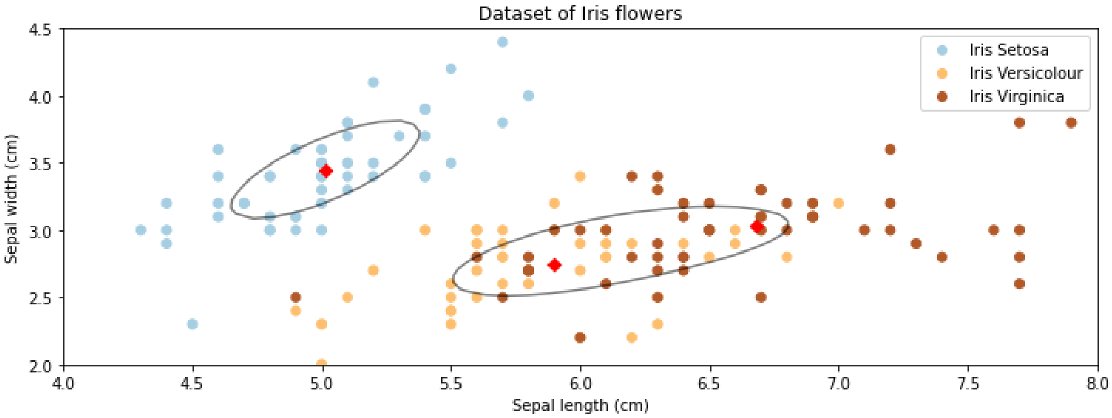
[22]:

```
<matplotlib.legend.Legend at 0x12293ad68>
```



```
[28]: CS = plt.contour(X, Y, Z, levels=np.logspace(0, 30, 1), cmap='gray', alpha=0.5)
p1 = plt.scatter(iris.data[:,0], iris.data[:,1], alpha=1, c=iris.target,␣
 ↪cmap='Paired')
plt.scatter(model.means_[:,0], model.means_[:,1], marker='D', c='r')
plt.ylabel("Sepal width (cm)")
plt.xlabel("Sepal length (cm)")
plt.title("Dataset of Iris flowers")
plt.legend(handles=p1.legend_elements()[0], labels=['Iris Setosa', 'Iris␣
 ↪Versicolour', 'Iris Virginica'])
```

```
[28]: <matplotlib.legend.Legend at 0x1229d3668>
```



21 Applications of Unsupervised Learning

Unsupervised learning also has numerous applications: * Recommendation systems: suggesting movies on Netflix. * Anomaly detection: identifying factory components that are likely to break soon. * Signal denoising: extracting human speech from a noisy recording.

22 Reinforcement Learning

In reinforcement learning, an agent is interacting with the world over time. We teach it good behavior by providing it with rewards.

Image by Lily Weng

23 Applications of Reinforcement Learning

Applications of reinforcement learning include: * Creating agents that play games such as Chess or Go. * Controling the cooling systems of datacenters to use energy more efficiently. * Designing new drug compounds.

24 Artificial Intelligence and Deep Learning

Machine learning is closely related to these two fields. * AI is about building machines that exhibit intelligence. * ML enables machines to learn from experience, a useful tool for AI. * Deep learning focuses on a family of learning algorithms loosely inspired by the brain.

Image source.

# Part 3: Logistics

We will now go over some practical bits of information.

25 What Is the Course About?

The focus of this course is on applied machine learning. * We cover a *broad* set of algorithms from different subfields of ML * In the lectures, we explain how they work using *math* * In the homeworks, you *code* them up using `numpy` or `sklearn`

We will additionally cover practical aspects of applying machine learning.

26 Syllabus

The course will have, roughly speaking, three parts. 1. Basic ML * Supervised and unsupervised learning * Regression, classification, SVMs, K-means, PCA, etc. 2. Advanced ML * Kernel methods, boosted decision trees, deep learning 3. How to apply machine learning * Overfitting, error analyis, learning curves, etc.

27 Syllabus

The course will have, roughly speaking, four parts. 1. Supervised learning * Regression, Classification, Naive Bayes, SVMs 2. Unsupervised learning * K-Means clustering, Kernel density estimation, PCA 3. Advanced supervised learning and deep learning * Neural networks, boosted decision trees, kernel metohds 4. How to apply machine learning * Overfitting, error analyis, learning curves, etc.

28 Prerequisites: Is This Course For You?

This course is aimed at a general technical audience. Main requirements are: * Programming experience (at least 1 year), preferably in Python. * College-level linear algebra. Matrix operations, the SVD decomposition, etc. * College-level probability. Probability distributions, random variables, Bayes' rule, etc.

29 Tutorials: Reviewing the Prerequisites

We will review the prerequisites in the first few weeks: * Linear algebra and probability (covered by Nathan Kallus's team) * Python and Numpy tutorial (by our team)

These will be held outside class; location/date/time is TBD.

30 Logistics

We meet Tue/Thu 1pm-2:15pm in Bloomberg 131.

- Class Webpage: https://canvas.cornell.edu/courses/33314
- Teaching Assistants: Zheng Li, Yiran Zhao, Yin Li
- Graders: Sri Chakra Kumar, Grace Le, Eric Zhewen Li, Yujie Shao, Ziyu Song, Sanika Bapat, Rashmi Sinha
- Office hours
  - Volodymyr: After class
  - See website for all the hours

31 More Logistics

We will use Canvas and Gradescope for the course. * Gradescope: download and submit all assignments there * Click the "Gradescope" tab in Canvas * Use the Canvas Discussion forums to ask us questions online * Click the "Discussions" tab in Canvas * Make sure to regularly watch Canvas for announcements * Click the "Announcements" tab in Canvas

32 Remote/Zoom Logistics

Following Cornell Tech policies, we will make course content available online at least until the end of September. * Classes will be streamed on Zoom. Use the "Zoom" tab in Canvas to access these. * We will also have Zoom links for office hours. You may come see us in person or over Zoom.

33 Recorded Lecture Videos

Lecture videos from 2020 are available online

34 COVID-19 Policies

Your health and safety is our priority. * Make sure to wear a mask indoors * Do not come to class if you are experiencing any Covid-19 symptoms

35 Grading

We will have homeworks, a prelim, and a project. * Four homeworks: 10% each * Mix of conceptual and programming questions * In-class prelim (Oct 19): 15% * Course project: 45% * Proposal (2-3 paragraphs) : 5% * Progress report (3-5 pages): 15% * Final report (5 pages): 25% * You have **six late days** with a **max of two** per deliverable

36 Project

Course projects will be done in groups of up to 3 students and can fall into one or more of the following categories: * Application of machine learning on a novel task/dataset * Algorithmic improvements into the representation, learning, or evaluation of machine learning models * Theoretical analysis of any aspect of existing machine learning methods

You are encouraged to choose your project, but we will make suggestions.

37 In-Class Feedback System

We encourage you to submit (anonymous) feedback about how the course is going throughout the semester. * Use our Google form * Tell us what you think about assignments, the pace of the lectures, the due dates, project deliverables, etc.

37.0.1 Again, Welcome to Applied Machine Learning!

38 Software You Will Use

You will use Python and popular machine learning libraries such as: * `scikit-learn`. It implements most classical machine learning algorithms. * `tensorflow`, `keras`, `pytorch`. Standard libraries for modern deep learning. * `numpy`, `pandas`. Linear algebra and data processing libraries used to implement algorithms from scratch.

39 Executable Course Materials

The core materials for this course (including the slides!) are created using Jupyter notebooks. * We are going to embed an execute code directly in the slides and use that to demonstrate algorithms. * These slides can be downloaded locally and all the code can be reproduced.

[29]:

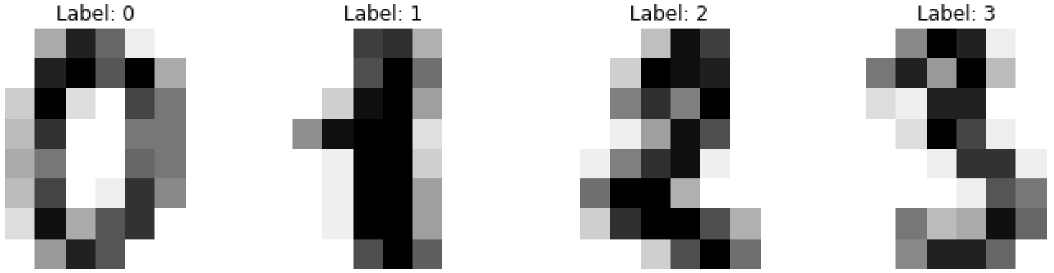
```python
import numpy as np
import matplotlib.pyplot as plt
from sklearn import datasets, neural_network
plt.rcParams['figure.figsize'] = [12, 4]
```

We can use these libraries to load a simple datasets of handwritten digits.

[7]:

```python
# https://scikit-learn.org/stable/auto_examples/classification/
 ↪plot_digits_classification.html
# load the digits dataset
digits = datasets.load_digits()
```

```
# The data that we are interested in is made of 8x8 images of digits, let's
# have a look at the first 4 images.
_, axes = plt.subplots(1, 4)
images_and_labels = list(zip(digits.images, digits.target))
for ax, (image, label) in zip(axes, images_and_labels[:4]):
    ax.set_axis_off()
    ax.imshow(image, cmap=plt.cm.gray_r, interpolation='nearest')
    ax.set_title('Label: %i' % label)
```

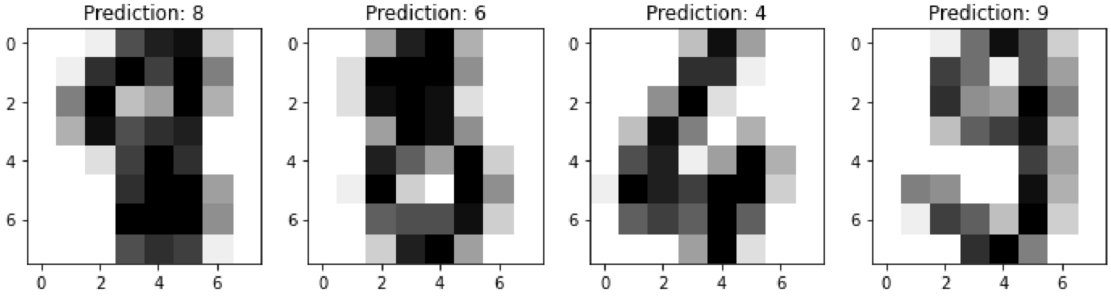


We can now load and train this algorithm inside the slides.

[30]:
```
np.random.seed(0)
# To apply a classifier on this data, we need to flatten the image, to
# turn the data in a (samples, feature) matrix:
data = digits.images.reshape((len(digits.images), -1))

# create a small neural network classifier
from sklearn.neural_network import MLPClassifier
classifier = MLPClassifier(alpha=1e-3)

# Split data into train and test subsets
X_train, X_test, y_train, y_test = sk.model_selection.train_test_split(data,␣
 ↪digits.target, test_size=0.5, shuffle=False)

# We learn the digits on the first half of the digits
classifier.fit(X_train, y_train)

# Now predict the value of the digit on the second half:
predicted = classifier.predict(X_test)
```

We can now visualize the results.

[31]:
```
_, axes = plt.subplots(1, 4)
images_and_predictions = list(zip(digits.images[n_samples // 2:], predicted))
for ax, (image, prediction) in zip(axes, images_and_predictions[:4]):
    ax.imshow(image, cmap=plt.cm.gray_r, interpolation='nearest')
    ax.set_title('Prediction: %i' % prediction)
```


Prediction: 8
Prediction: 6
Prediction: 4
Prediction: 9