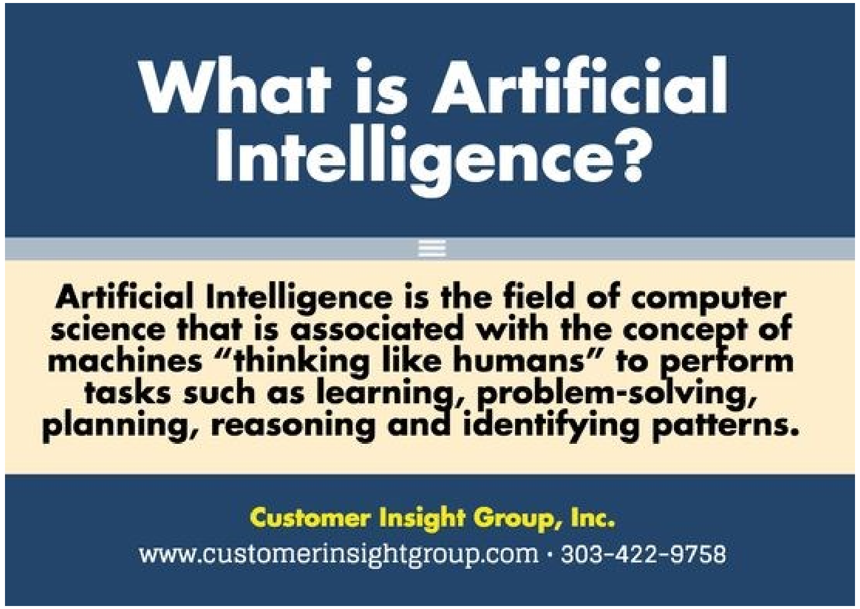
Introduction to Deep Learning

With PyTorch

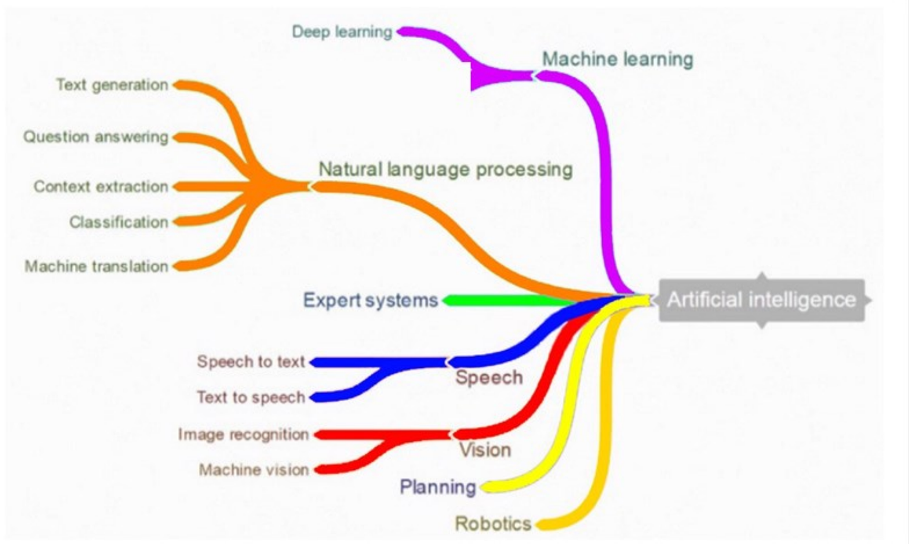
What is AI?



What is Artificial Intelligence?

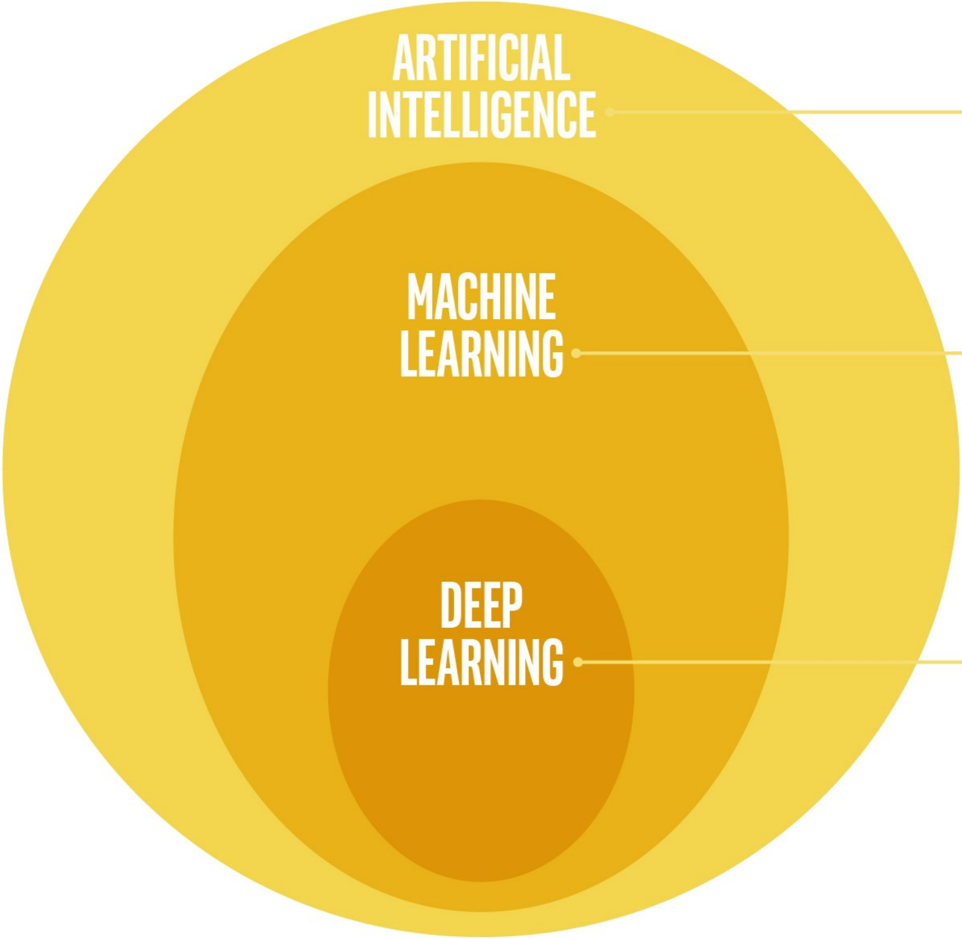
Artificial Intelligence is the field of computer science that is associated with the concept of machines "thinking like humans" to perform tasks such as learning, problem-solving, planning, reasoning and identifying patterns.

Customer Insight Group, Inc.
www.customerinsightgroup.com • 303-422-9758



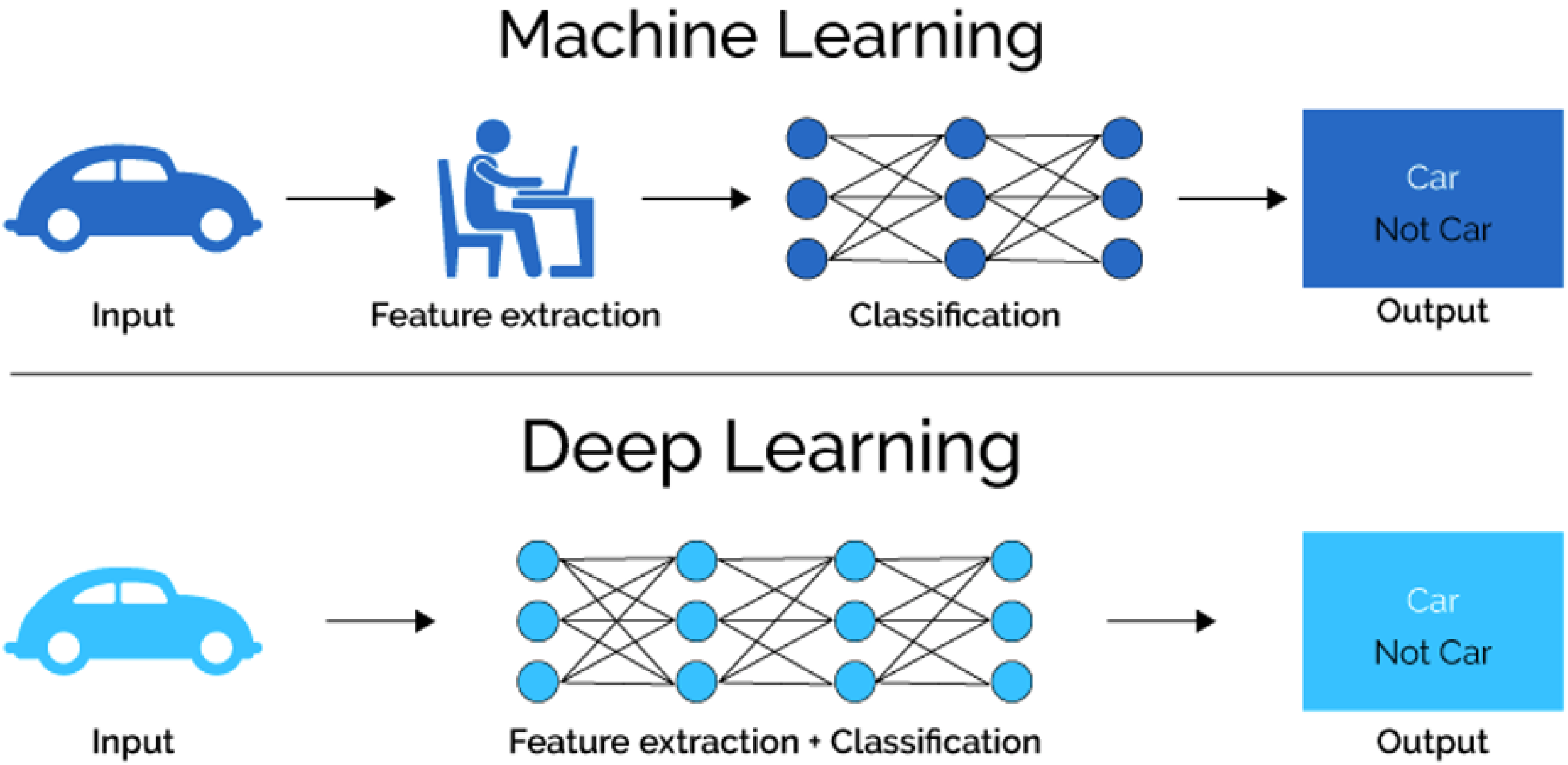
AI, ML, DL?

ARTIFICIAL INTELLIGENCE TERMS

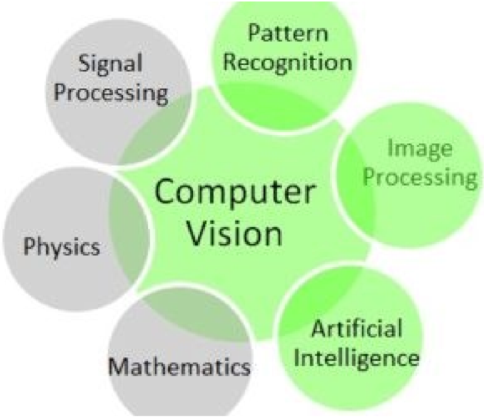


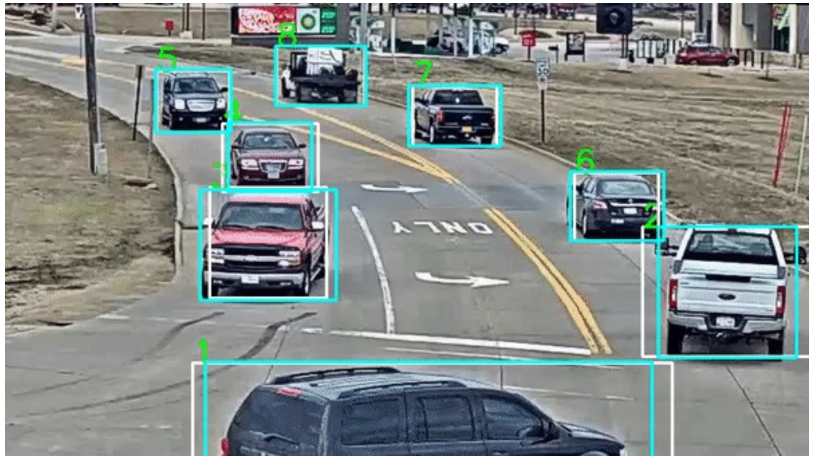
- **AI** is an umbrella term for machines capable of perception, logic, and learning.
- **Machine learning** employs algorithms that learn from data to make predictions or decisions, and whose performance improves when exposed to more data over time.
- **Deep learning** uses many-layered neural networks to build algorithms that find the best way to perform tasks on their own, based on vast sets of data.

ML and DL?

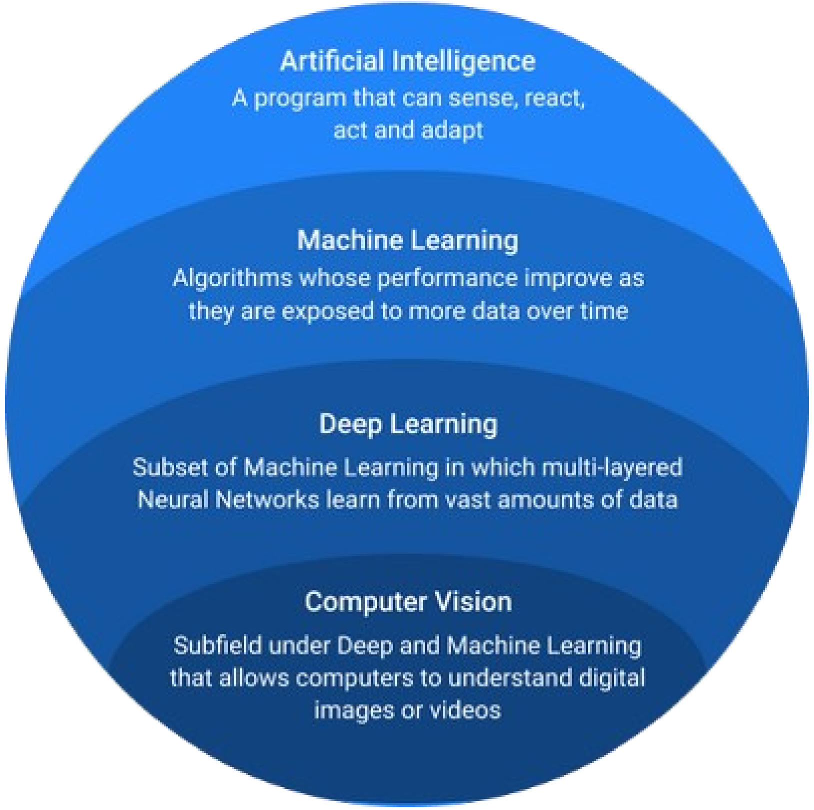

Machine Learning
Input
Feature extraction
Classification
Car
Not Car
Output
Deep Learning
Input
Feature extraction + Classification
Car
Not Car
Output

DL and CV?





Deep Learning vs. Computer Vision

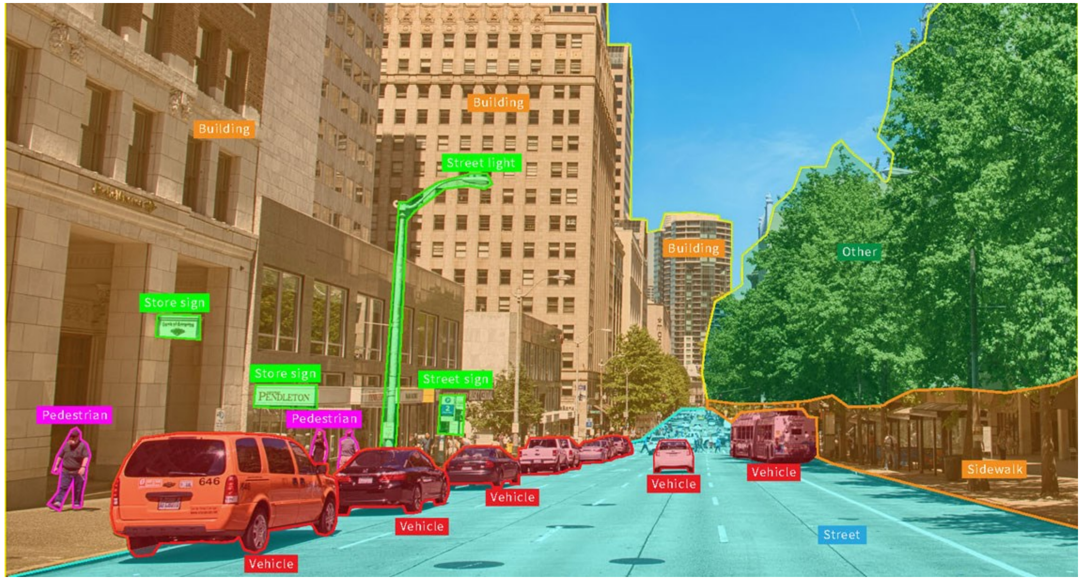


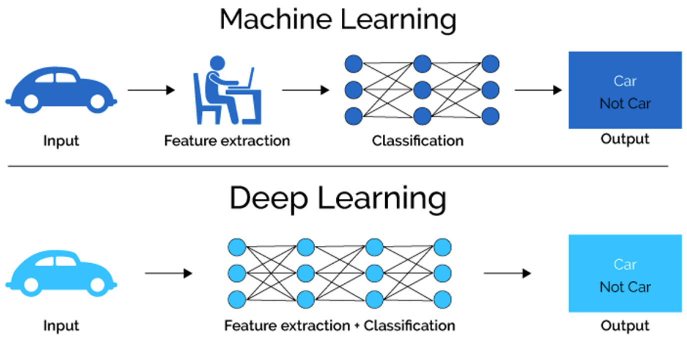
Artificial Intelligence
A program that can sense, react, act and adapt

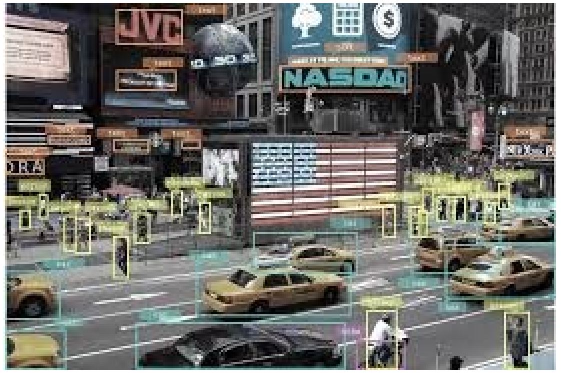
Machine Learning
Algorithms whose performance improve as they are exposed to more data over time

Deep Learning
Subset of Machine Learning in which multi-layered Neural Networks learn from vast amounts of data

Computer Vision
Subfield under Deep and Machine Learning that allows computers to understand digital images or videos

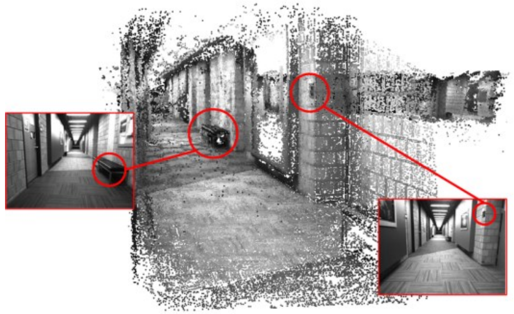


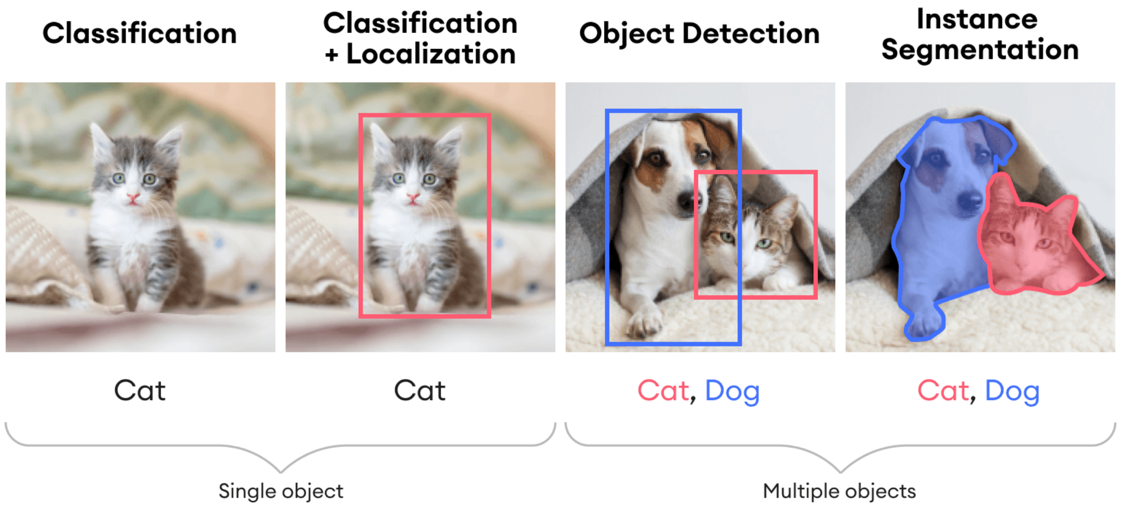


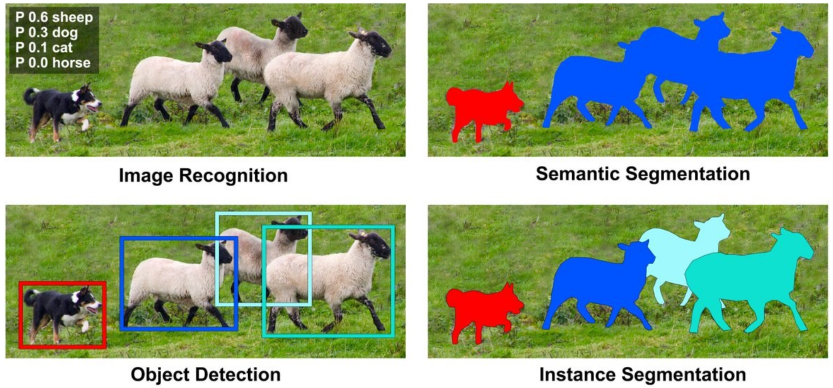


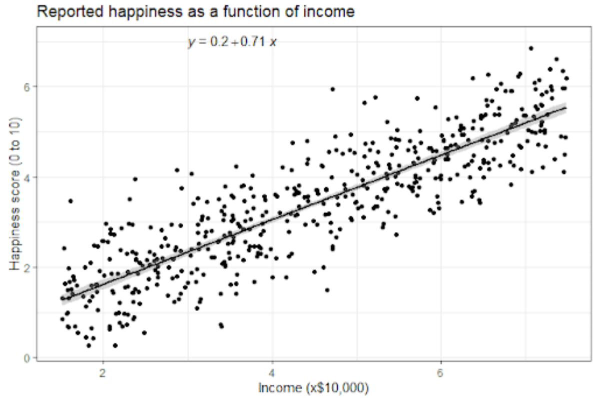
Tasks Related to DL/ CV

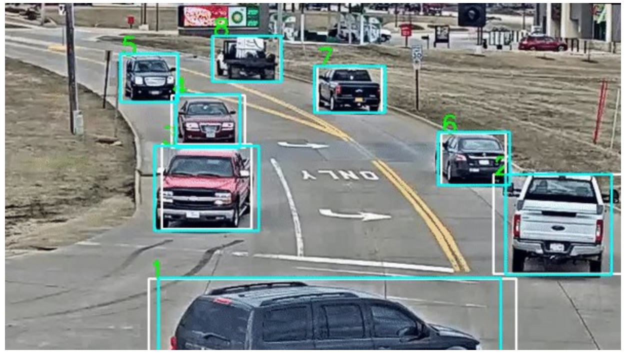
- Classification
- Regression
- Detection
- Tracking
- Segmentation
- Registration
- 3D Reconstruction
  - 3D Computer Vision





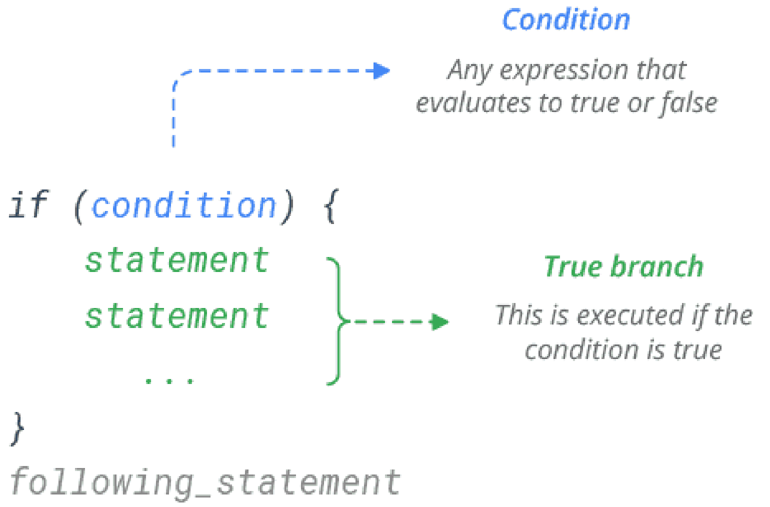






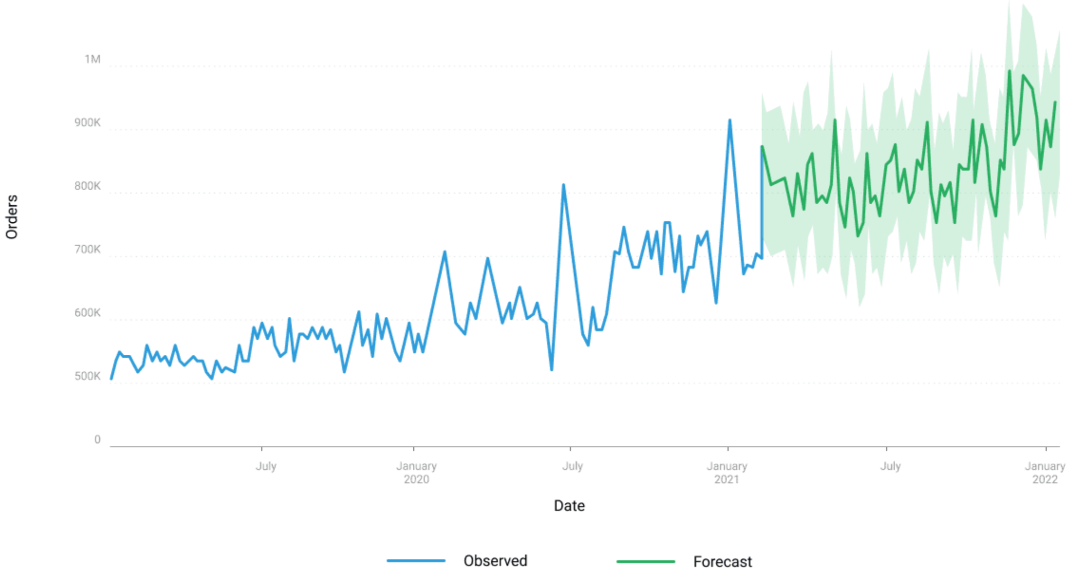
Is DL a Remedy for Everything?

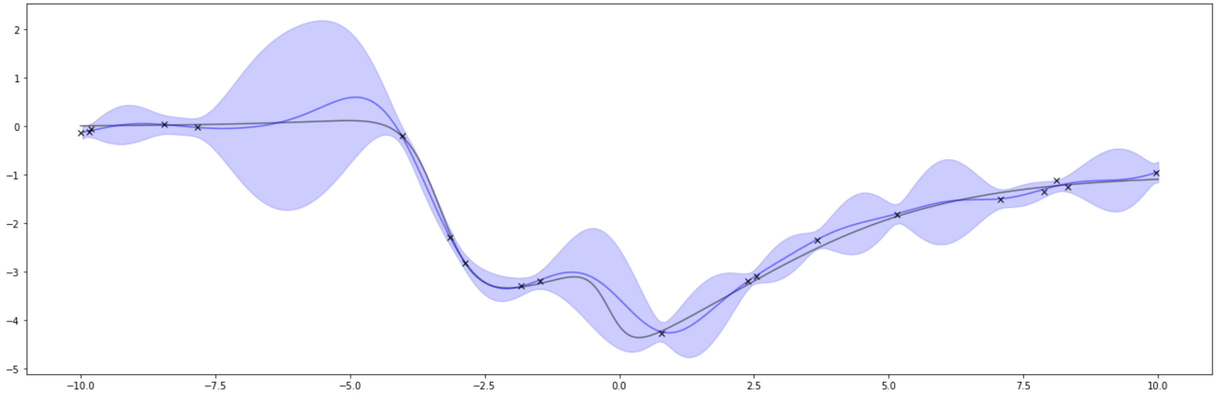
- Decision-making
- Forecasting
- Predictions/ Uncertainty Analysis



```
if (condition) {
    statement
    statement
    ...
}
following_statement
```

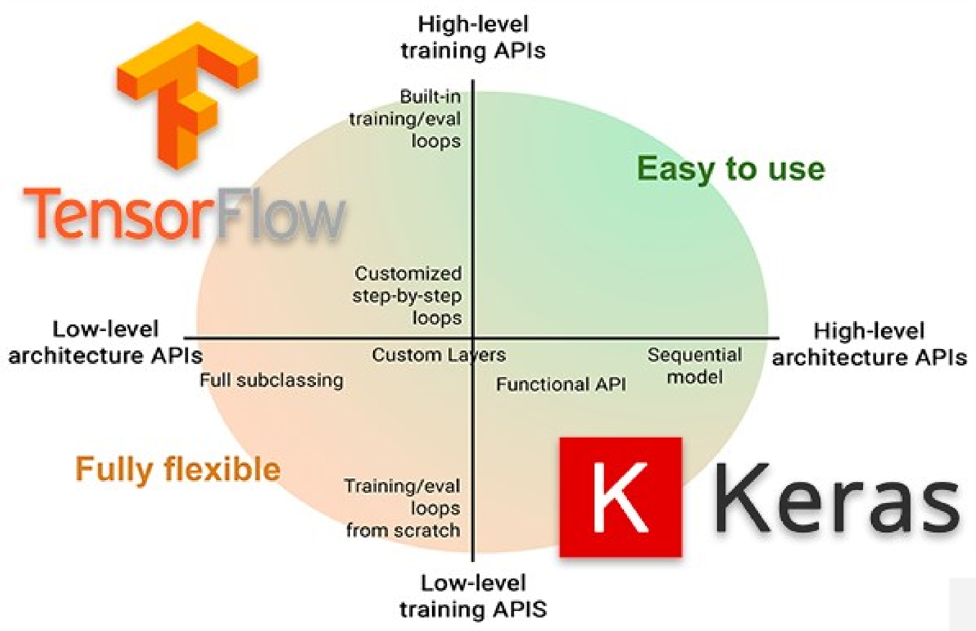
Condition
Any expression that evaluates to true or false
True branch
This is executed if the condition is true





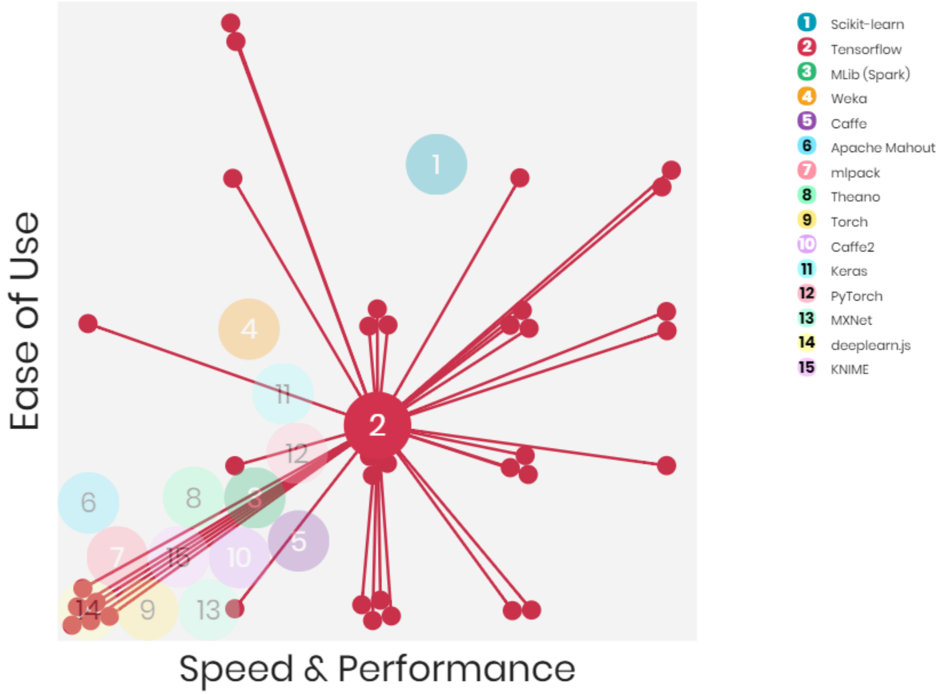
DL Frameworks

Keras



TensorFlow

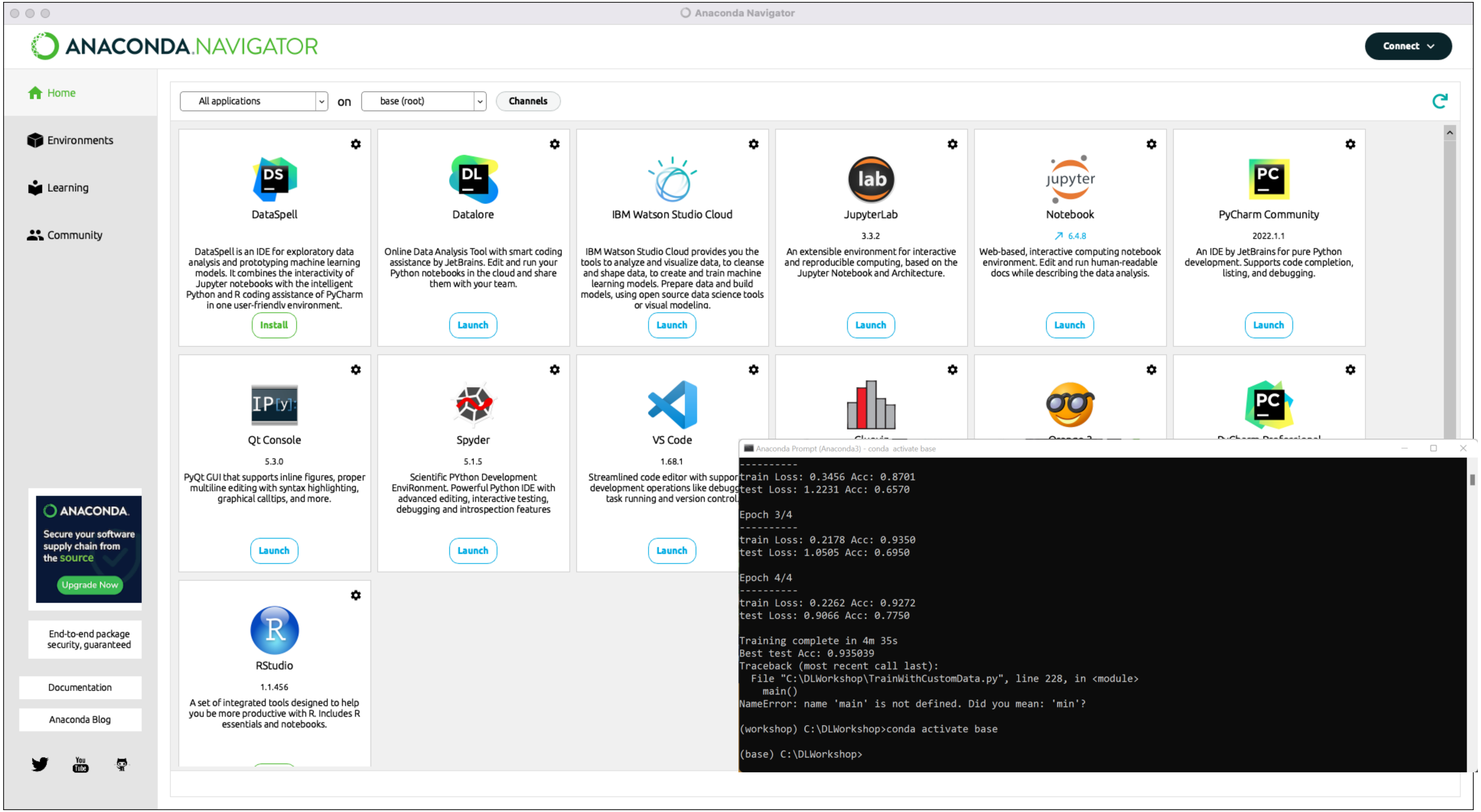
PyTorch



How To Start Solving DL Problems?

1. Get a Good GPU and RAM for model training (or use cloud)
2. Install OS, NVIDIA drivers, CUDA, cuDNN libs
3. Install Anaconda/ Python. Create Environment and Select DL Framework
   - OS: Windows/ Ubuntu (Recommended)
4. Setup development environment
   - Install libraries and their dependencies
   - Pay attention to versions
5. Understand the problem/ Prepare or download the datasets related to the task
   - For custom datasets, first acquire it and then annotate it
   - Annotation Tools: MATLAB ImageLabeler, LabelMe etc.
6. Model Training/ Testing
   - Model Evaluation

Anaconda



DataSpell

DataSpell is an IDE for exploratory data analysis and prototyping machine learning models. It combines the interactivity of Jupyter notebooks with the intelligent Python and R coding assistance of PyCharm in one user-friendly environment.

Datalore

Online Data Analysis Tool with smart coding assistance by JetBrains. Edit and run your Python notebooks in the cloud and share them with your team.

IBM Watson Studio Cloud

IBM Watson Studio Cloud provides you the tools to analyze and visualize data, to cleanse and shape data, to create and train machine learning models. Prepare data and build models, using open source data science tools or visual modeling.

JupyterLab

3.3.2

An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

Notebook

↗ 6.4.8

Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

PyCharm Community

2022.1.1

An IDE by JetBrains for pure Python development. Supports code completion, listing, and debugging.

Qt Console

5.3.0

PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.

Spyder

5.1.5

Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features

VS Code

1.68.1

Streamlined code editor with suppor development operations like debugg task running and version control.

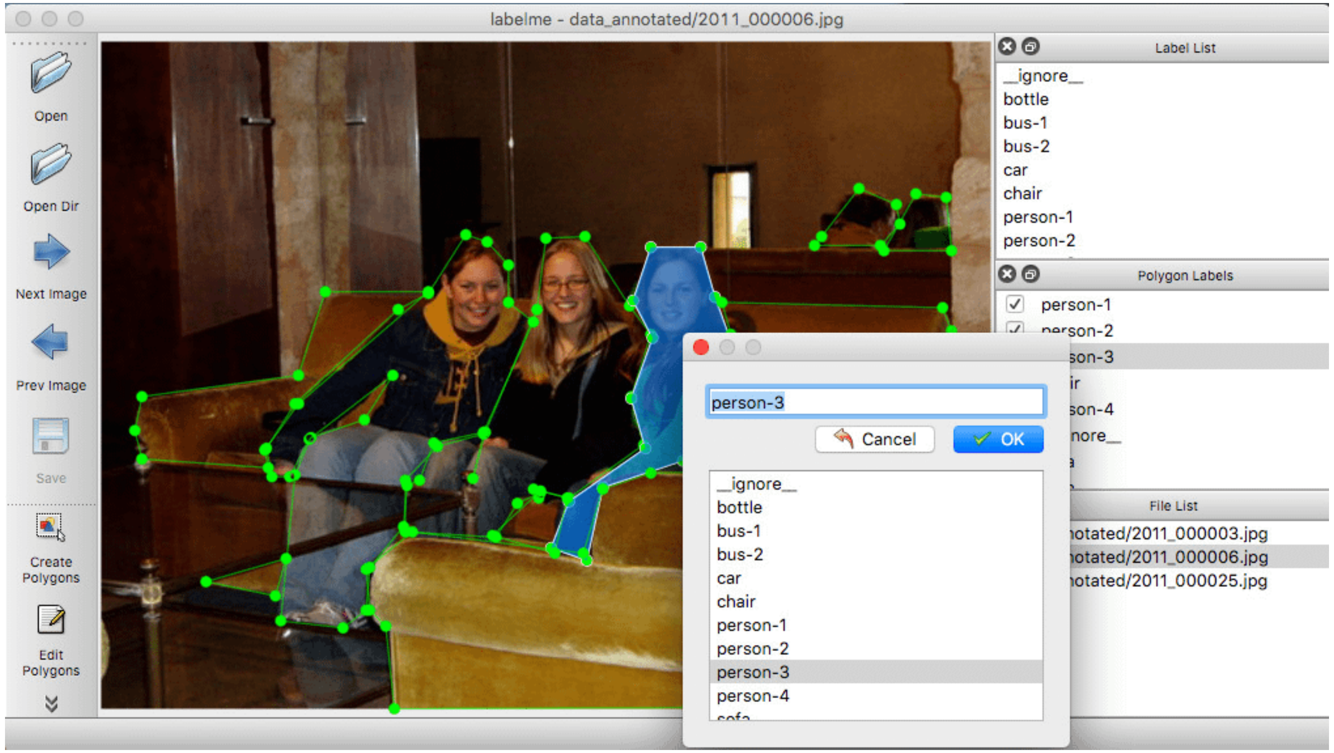
RStudio

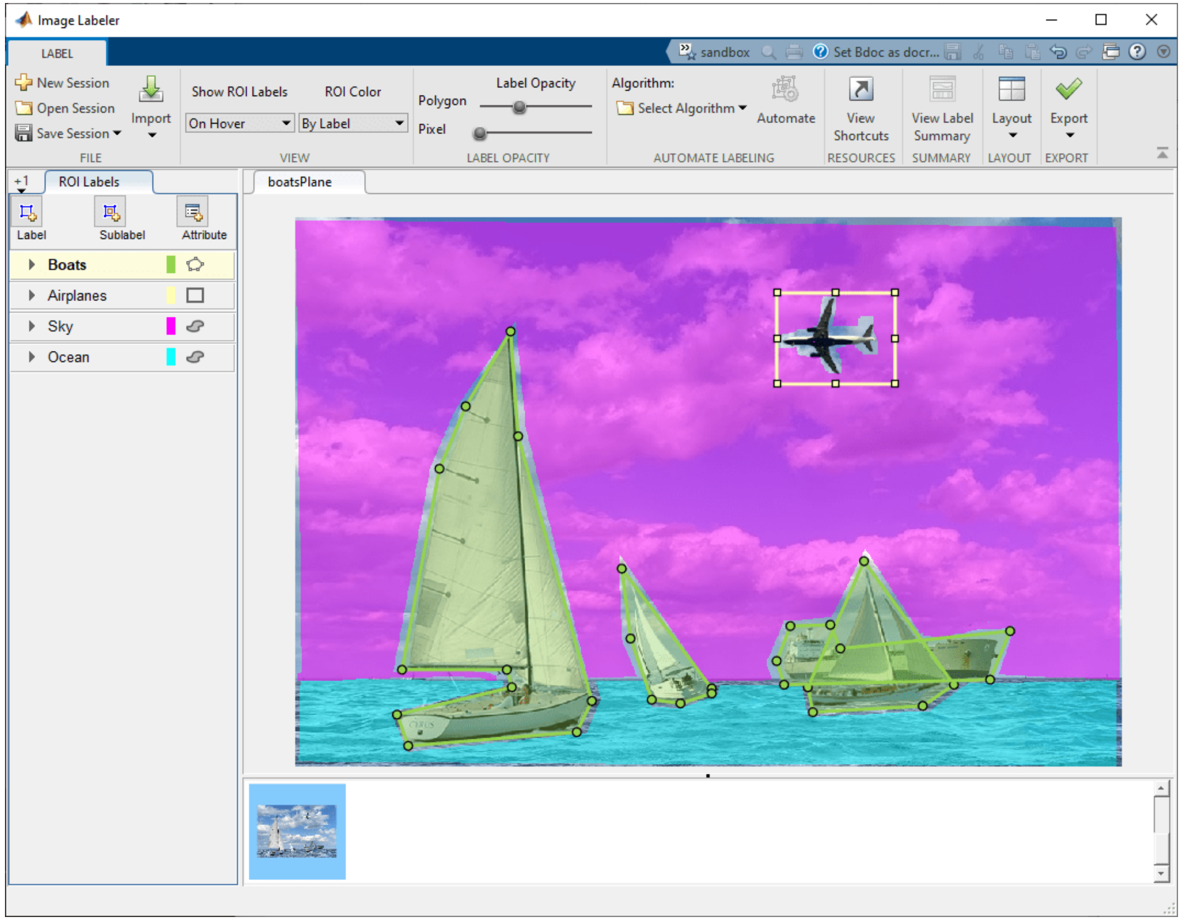
1.1.456

A set of integrated tools designed to help you be more productive with R. Includes R essentials and notebooks.

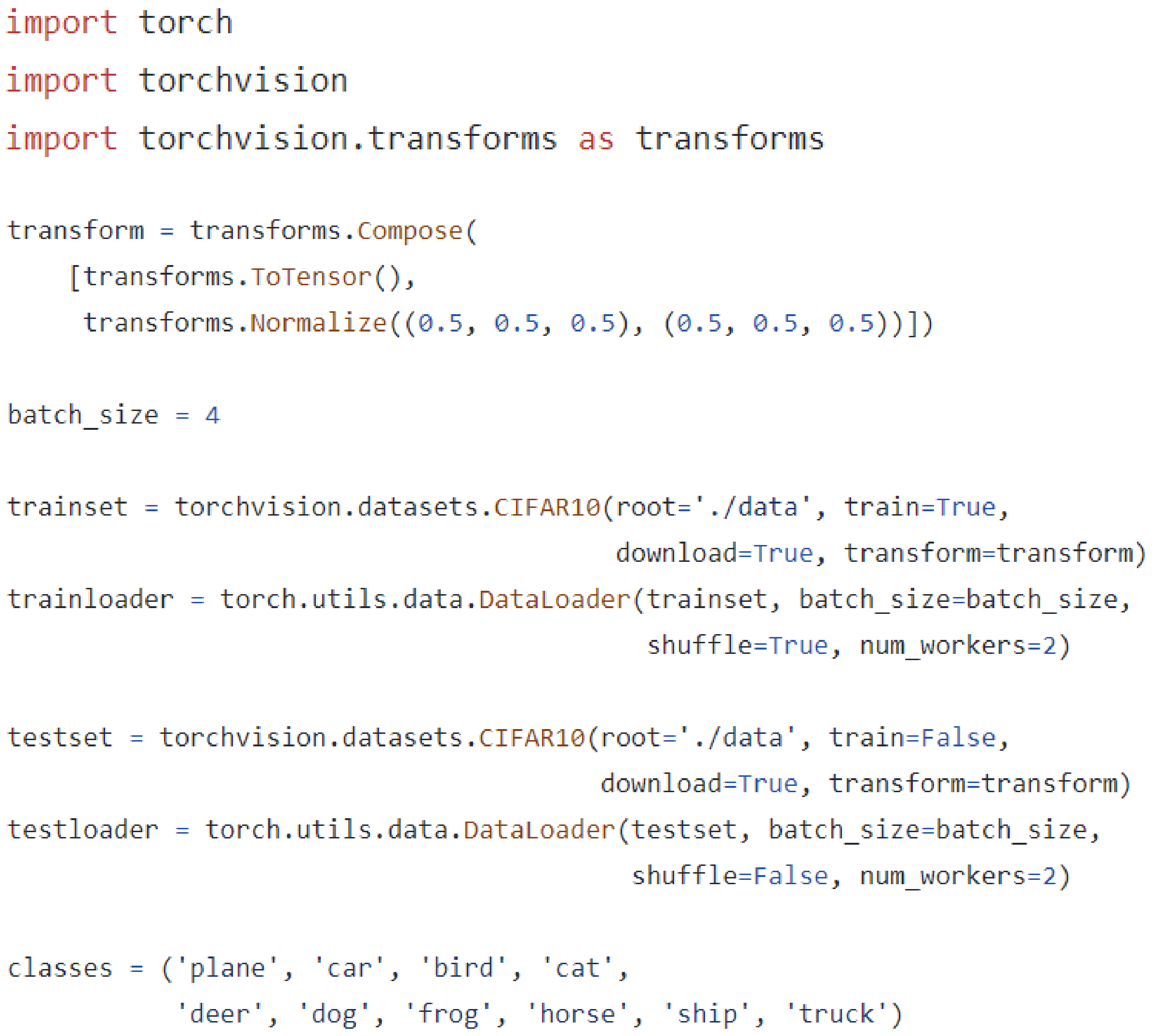
```
----------
train Loss: 0.3456 Acc: 0.8701
test Loss: 1.2231 Acc: 0.6570

Epoch 3/4
----------
train Loss: 0.2178 Acc: 0.9350
test Loss: 1.0505 Acc: 0.6950

Epoch 4/4
----------
train Loss: 0.2262 Acc: 0.9272
test Loss: 0.9066 Acc: 0.7750

Training complete in 4m 35s
Best test Acc: 0.935039
Traceback (most recent call last):
  File "C:\DLWorkshop\TrainWithCustomData.py", line 228, in <module>
    main()
NameError: name 'main' is not defined. Did you mean: 'min'?

(workshop) C:\DLWorkshop>conda activate base

(base) C:\DLWorkshop>
```

Ground Truth Annotation Tools





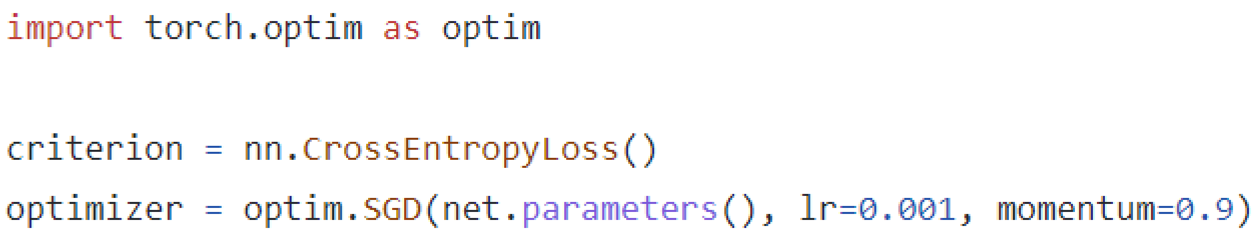
Example: A Simple Classification Network
Step # 1: Load Dataset



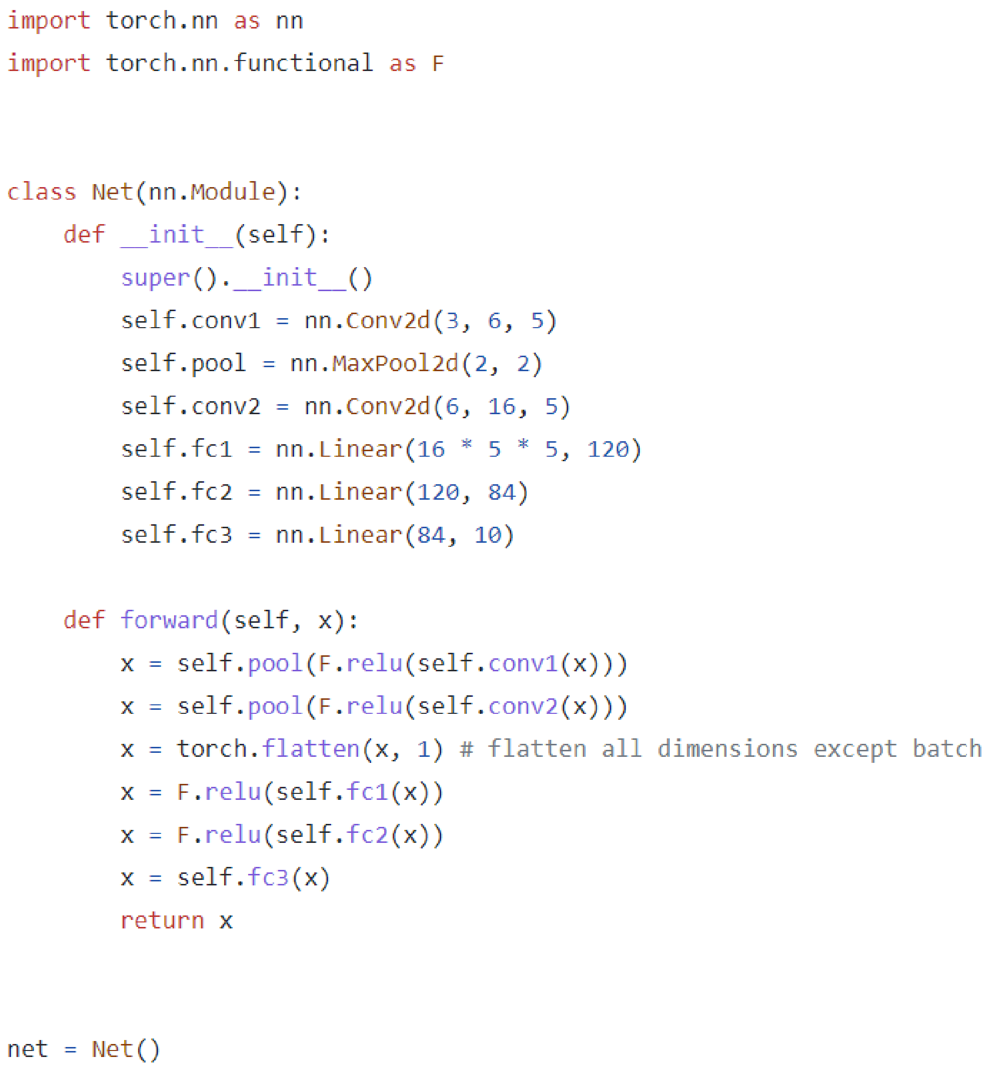
```
import torch
import torchvision
import torchvision.transforms as transforms

transform = transforms.Compose(
    [transforms.ToTensor(),
     transforms.Normalize((0.5, 0.5, 0.5), (0.5, 0.5, 0.5))])

batch_size = 4

trainset = torchvision.datasets.CIFAR10(root='./data', train=True,
                                        download=True, transform=transform)
trainloader = torch.utils.data.DataLoader(trainset, batch_size=batch_size,
                                          shuffle=True, num_workers=2)

testset = torchvision.datasets.CIFAR10(root='./data', train=False,
                                       download=True, transform=transform)
testloader = torch.utils.data.DataLoader(testset, batch_size=batch_size,
                                         shuffle=False, num_workers=2)

classes = ('plane', 'car', 'bird', 'cat',
           'deer', 'dog', 'frog', 'horse', 'ship', 'truck')
```

https://github.com/taimurhassan/DLTutorials

Example: A Simple Classification Network
Step # 2: Create the Network and Optimizer



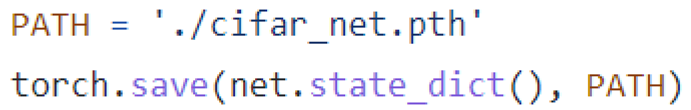
```python
import torch.optim as optim

criterion = nn.CrossEntropyLoss()
optimizer = optim.SGD(net.parameters(), lr=0.001, momentum=0.9)
```



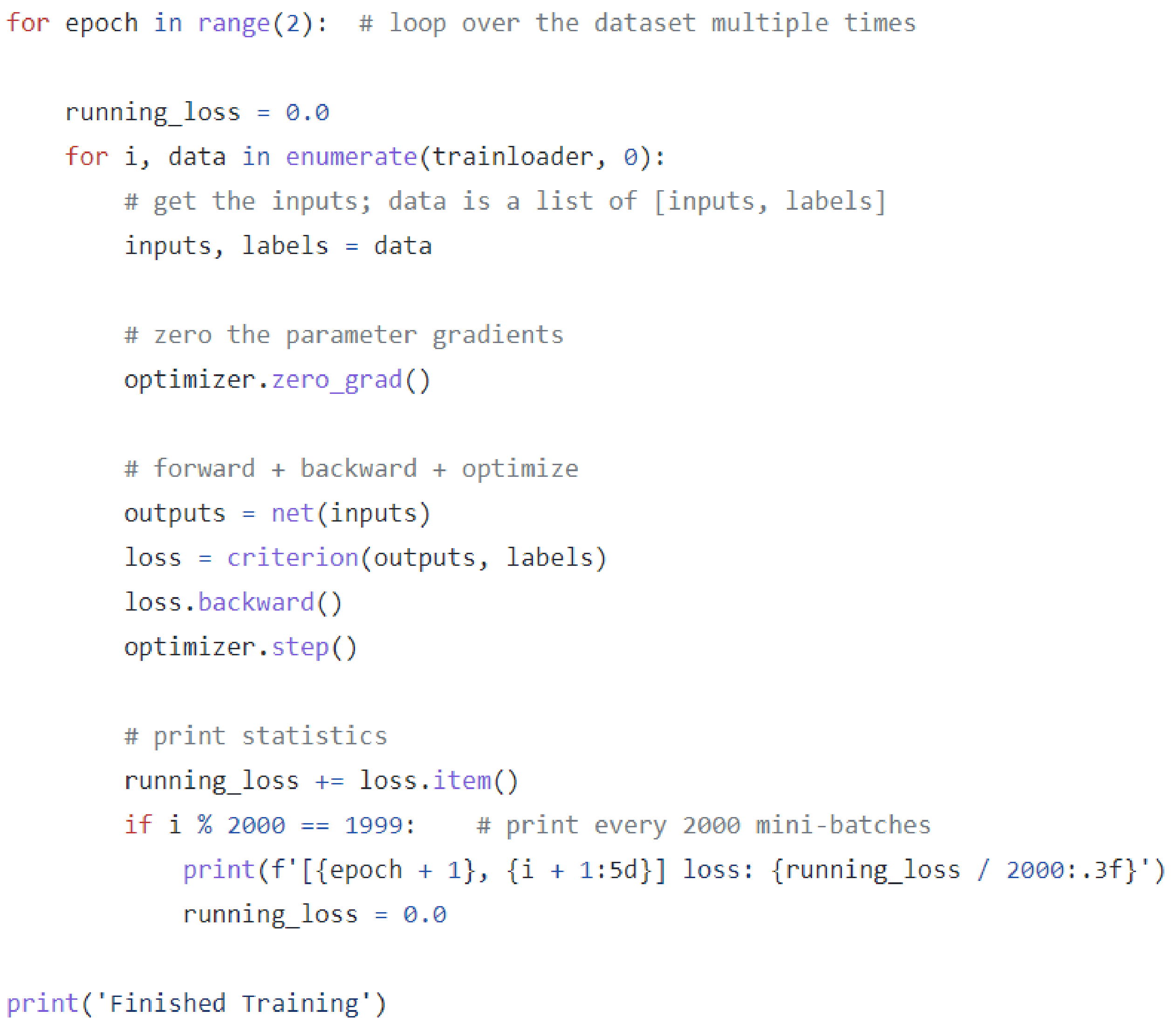
```python
import torch.nn as nn
import torch.nn.functional as F


class Net(nn.Module):
    def __init__(self):
        super().__init__()
        self.conv1 = nn.Conv2d(3, 6, 5)
        self.pool = nn.MaxPool2d(2, 2)
        self.conv2 = nn.Conv2d(6, 16, 5)
        self.fc1 = nn.Linear(16 * 5 * 5, 120)
        self.fc2 = nn.Linear(120, 84)
        self.fc3 = nn.Linear(84, 10)

    def forward(self, x):
        x = self.pool(F.relu(self.conv1(x)))
        x = self.pool(F.relu(self.conv2(x)))
        x = torch.flatten(x, 1) # flatten all dimensions except batch
        x = F.relu(self.fc1(x))
        x = F.relu(self.fc2(x))
        x = self.fc3(x)
        return x


net = Net()
```

https://github.com/taimurhassan/DLTutorials

Example: A Simple Classification Network
Step # 3: Write Training Routine



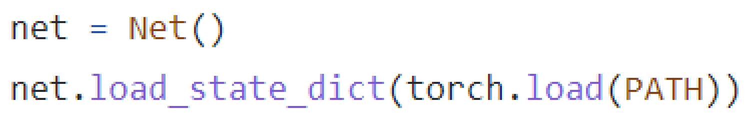
```python
PATH = './cifar_net.pth'
torch.save(net.state_dict(), PATH)
```



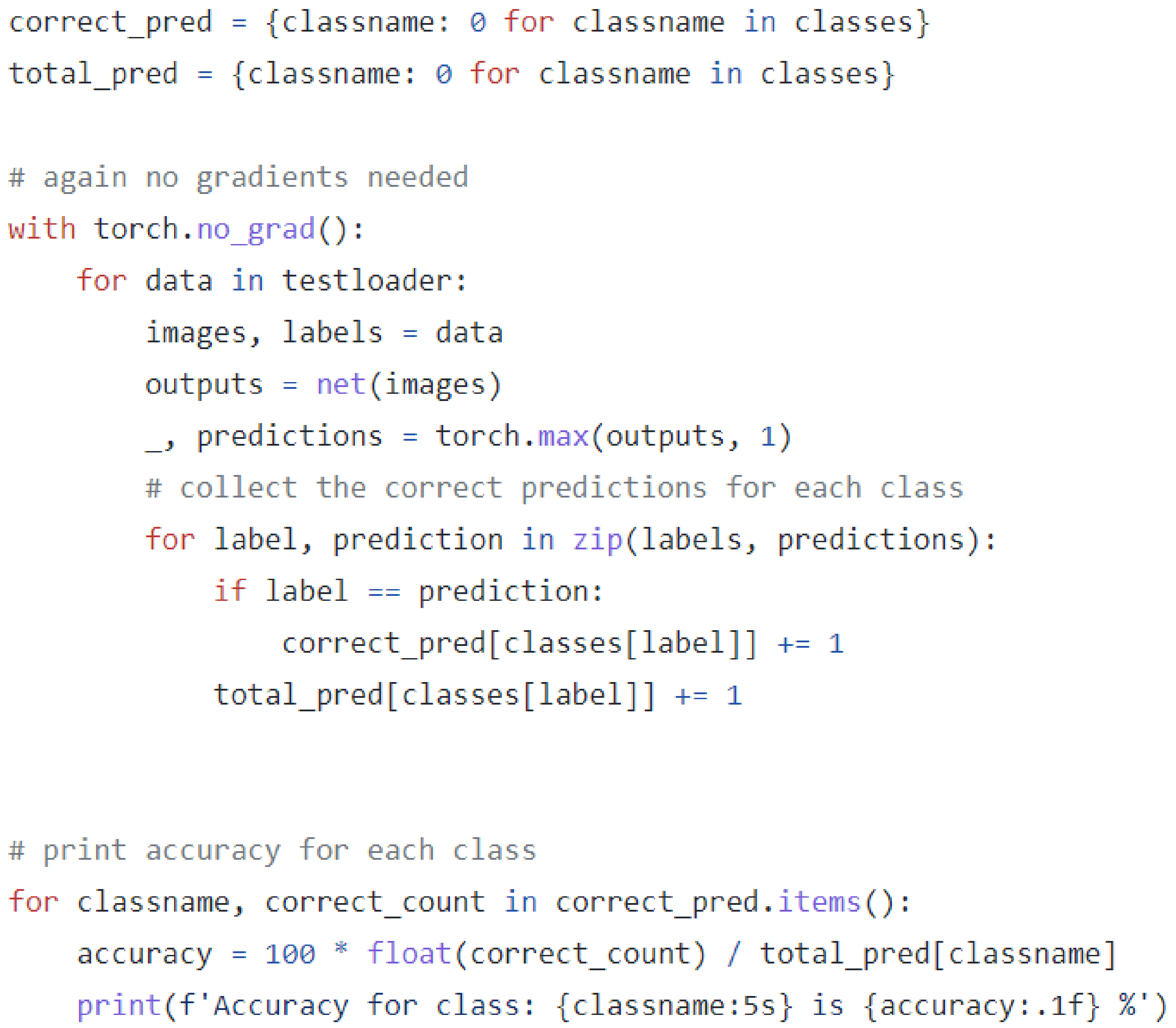
```python
for epoch in range(2):  # loop over the dataset multiple times

    running_loss = 0.0
    for i, data in enumerate(trainloader, 0):
        # get the inputs; data is a list of [inputs, labels]
        inputs, labels = data

        # zero the parameter gradients
        optimizer.zero_grad()

        # forward + backward + optimize
        outputs = net(inputs)
        loss = criterion(outputs, labels)
        loss.backward()
        optimizer.step()

        # print statistics
        running_loss += loss.item()
        if i % 2000 == 1999:    # print every 2000 mini-batches
            print(f'[{epoch + 1}, {i + 1:5d}] loss: {running_loss / 2000:.3f}')
            running_loss = 0.0

print('Finished Training')
```

https://github.com/taimurhassan/DLTutorials

Example: A Simple Classification Network
Step # 4: Test and Evaluate the Trained Model



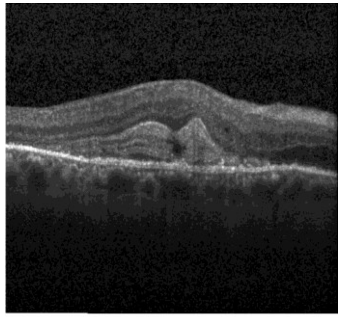
```
net = Net()
net.load_state_dict(torch.load(PATH))
```

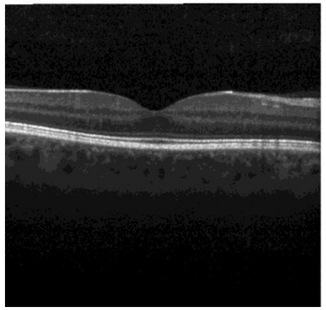


```
correct_pred = {classname: 0 for classname in classes}
total_pred = {classname: 0 for classname in classes}

# again no gradients needed
with torch.no_grad():
    for data in testloader:
        images, labels = data
        outputs = net(images)
        _, predictions = torch.max(outputs, 1)
        # collect the correct predictions for each class
        for label, prediction in zip(labels, predictions):
            if label == prediction:
                correct_pred[classes[label]] += 1
            total_pred[classes[label]] += 1


# print accuracy for each class
for classname, correct_count in correct_pred.items():
    accuracy = 100 * float(correct_count) / total_pred[classname]
    print(f'Accuracy for class: {classname:5s} is {accuracy:.1f} %')
```

https://github.com/taimurhassan/DLTutorials

Exercise (Image Classification): Training Model on Custom Dataset





- Use the given dataset to predict retinal diseases.
    - Link

Steps:

- Install OS, Anaconda, Python
- Load Anaconda and Create Environment
- Install Pip, PyTorch, Torchvision, pandas, scikit-image, matplotlib and all the dependencies
- Download the code file ‘TrainWithCustomData.py’ from the GitHub
- Run it and compute the results (in terms of accuracy)
- Understand the code and explain it in the next session
- Use to same code to classify digits in MNIST dataset

https://github.com/taimurhassan/DLTutorials