Week 5 flow

- Supervised learning and unsupervised learning crash course 2
- Lab 1 and 2 discussion
- Romanian practice test discussion

Supervised and unsupervised learning crash course II

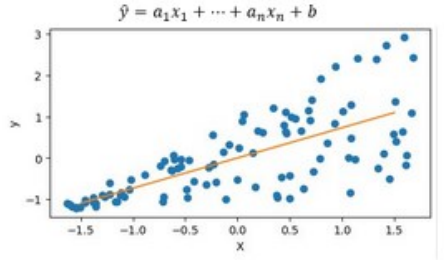
IOAI Training and Selection Programme 2025

Apr 19, 2025 Sat

Recap

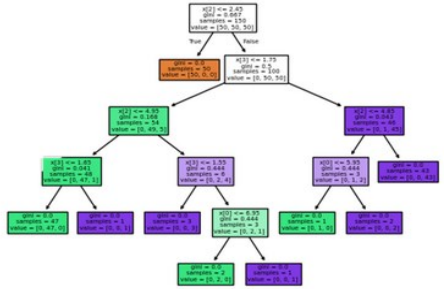
Linear Regression

$$\hat{y} = a_1 x_1 + \cdots + a_n x_n + b$$



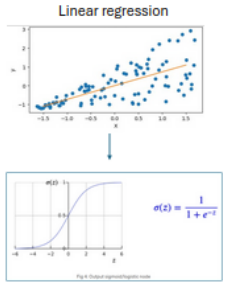
- MSE loss
- Use from sklearn.linear_model

Decision Trees



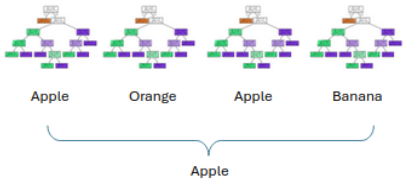
- Literally a bunch of "if-else"
- Decision trees have low bias, which is a useful property when ensembling
- Don't use this on its own!

Logistic regression



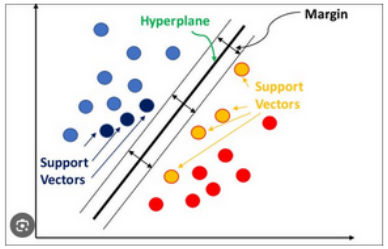
- Linear model
- Uses logistic loss (binary xent)
- Now a classifier!
- Use from sklearn.linear_model

Random Forest



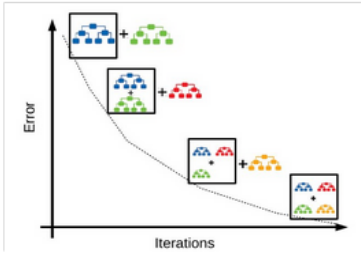
- Ensemble trees trained on different samples of data (bootstrap sampling)
- Trees only consider random subsets of features
- Majority vote for classification
- Average predictions for regression
- Non-parametric
- Use from sklearn.ensemble

Support Vector Machines



- Aims to find max margin by minimizing hinge loss
- Non-parametric by nature
- Commonly used for classification but works on regression
- Can model non-linear behaviour through kernel methods

Gradient Boosted Decision Trees



Imagine if your models keep making mistakes like underpredict. Can I just make another model that predicts the mistake, so that I can add a correction factor?

.. yes

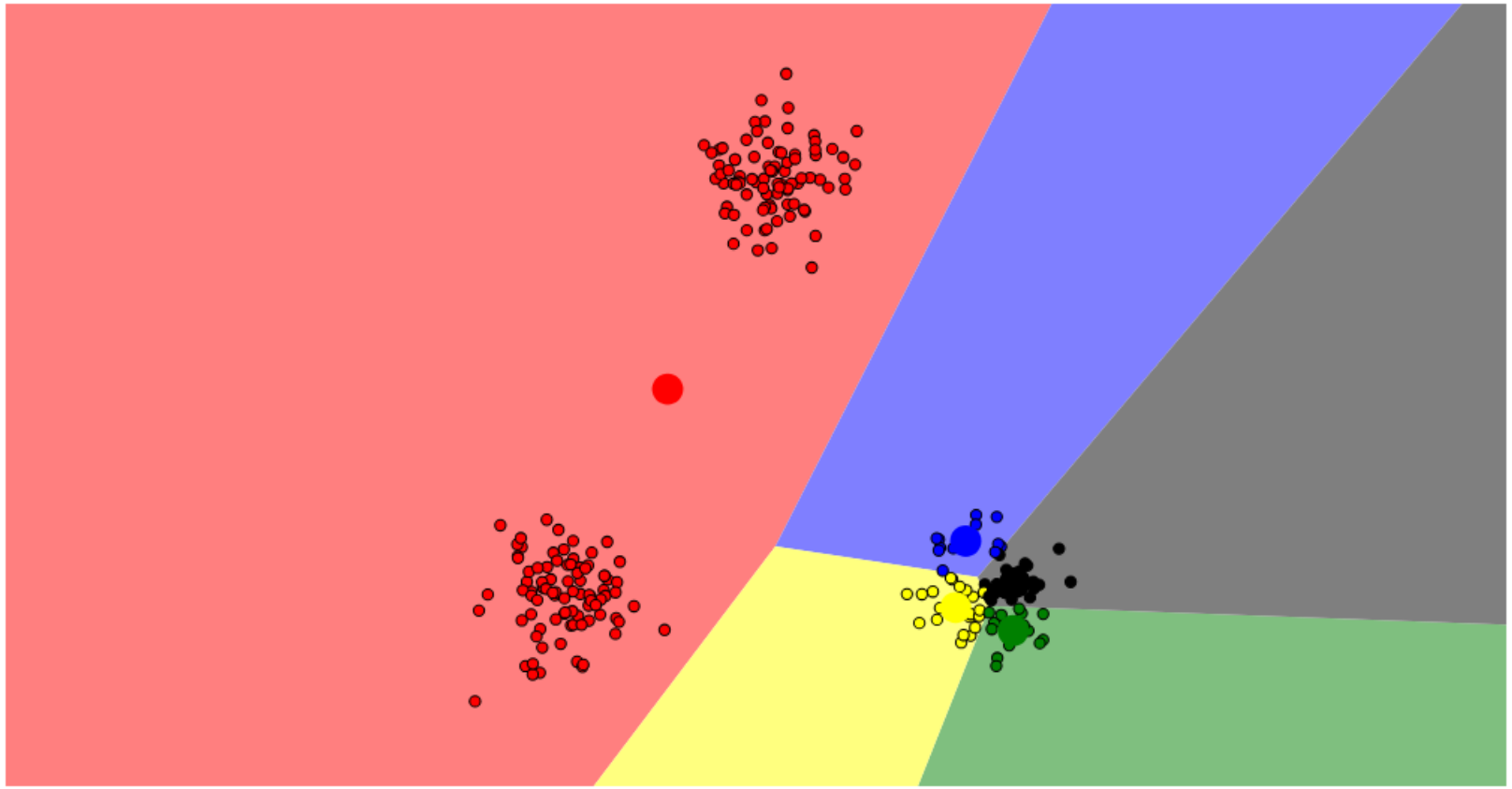
- Use from lightgbm or xgboost

K-means clustering

sklearn.cluster.k_means

https://www.researchgate.net/figure/llustration-of-K-means-Clustering-Note-The-estimation-routine-of-K-means-involves-i_fig2_371115662

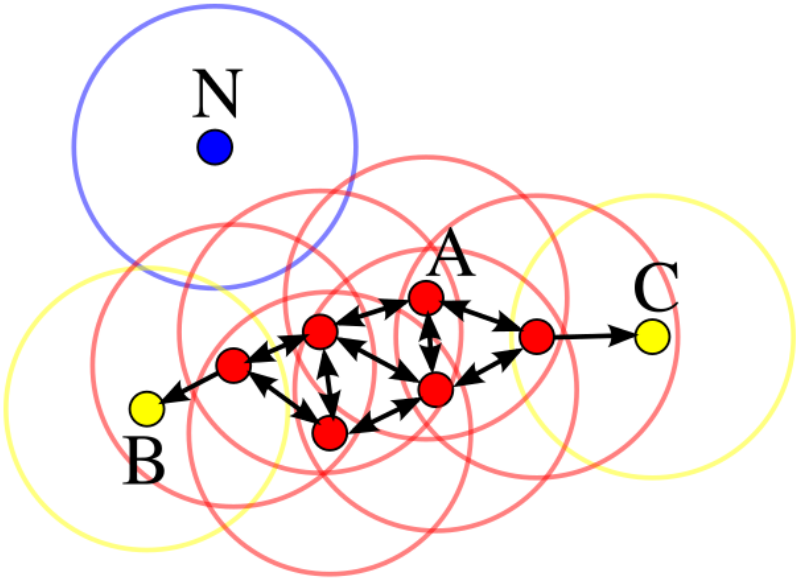
K-means clustering (cont)



https://www.naftaliharris.com/blog/visualizing-k-means-clustering/

DBSCAN

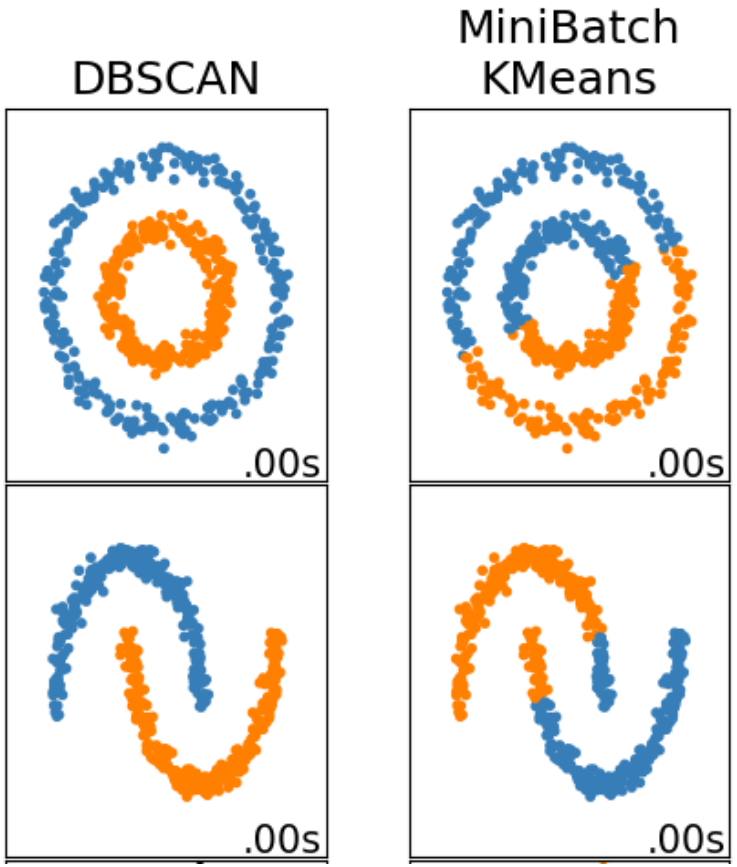
sklearn.cluster.DBSCAN



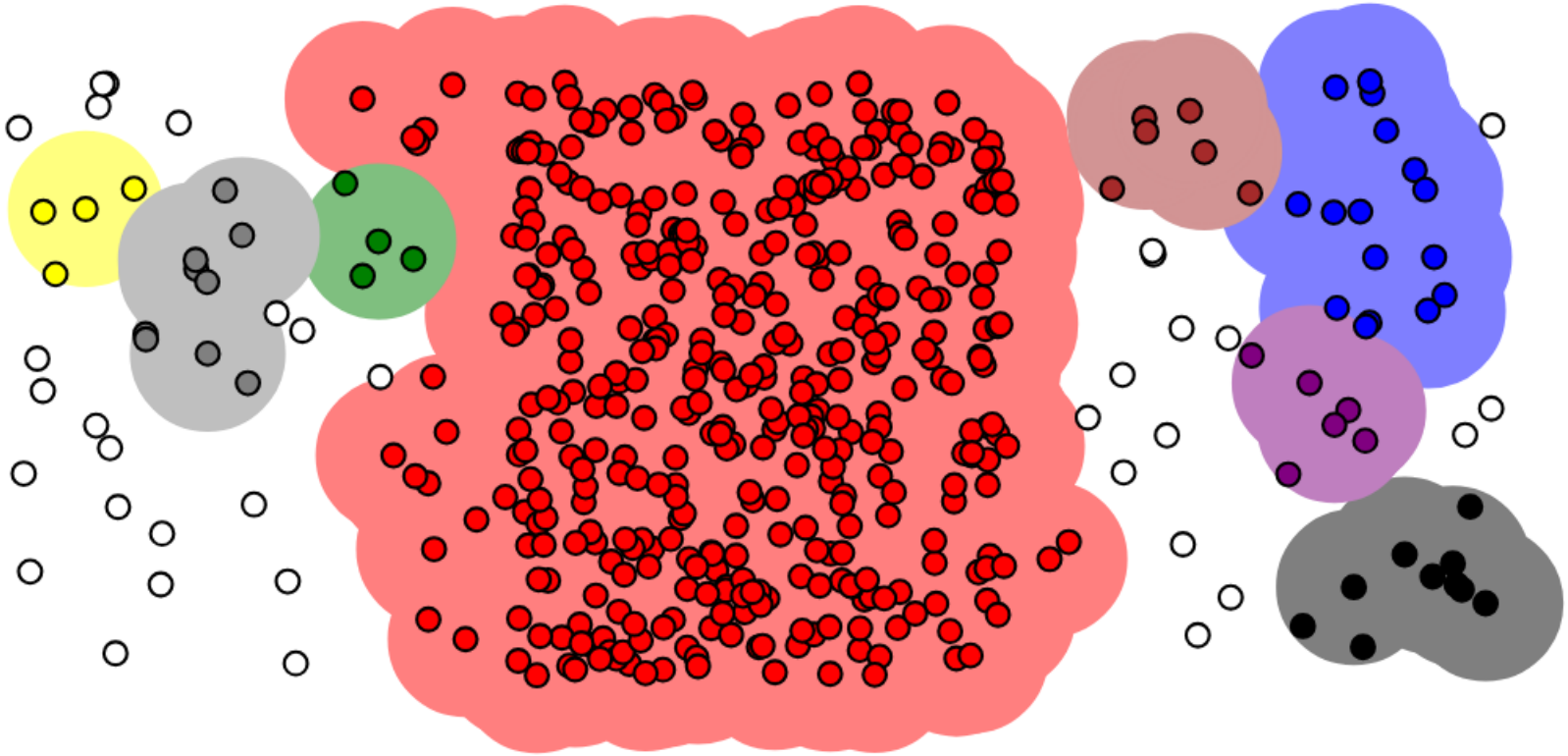
In this diagram, $\text{minPts} = 4$. Point A and the other red points are core points, because the area surrounding these points in an ε radius contain at least 4 points (including the point itself). Because they are all reachable from one another, they form a single cluster. Points B and C are not core points, but are reachable from A (via other core points) and thus belong to the cluster as well. Point N is a noise point that is neither a core point nor directly-reachable.

Source: https://en.m.wikipedia.org/wiki/DBSCAN
This is probably the best static illustration of DBSCAN that I have seen

Source: https://scikit-learn.org/stable/auto_examples/cluster/plot_cluster_comparison.htm



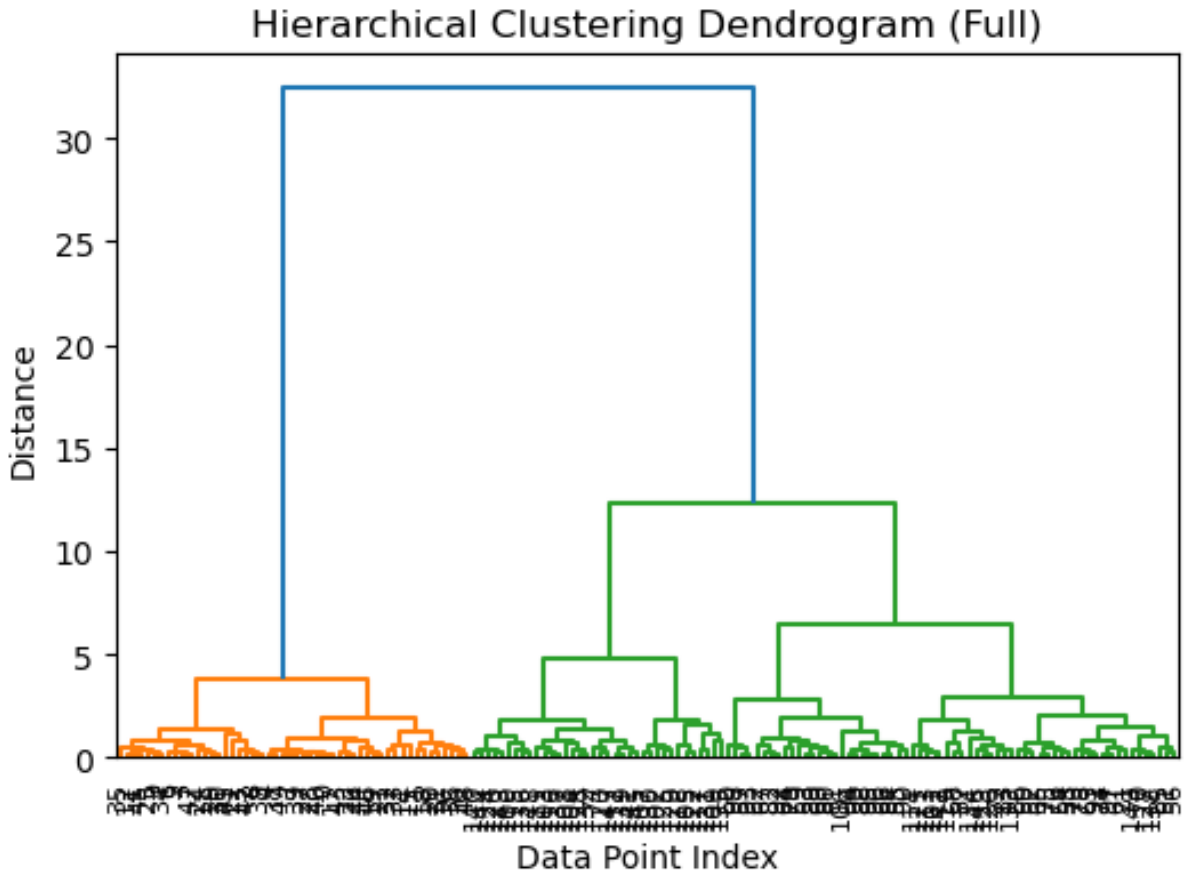
DBSCAN (cont)



https://www.naftaliharris.com/blog/visualizing-dbscan-clustering/

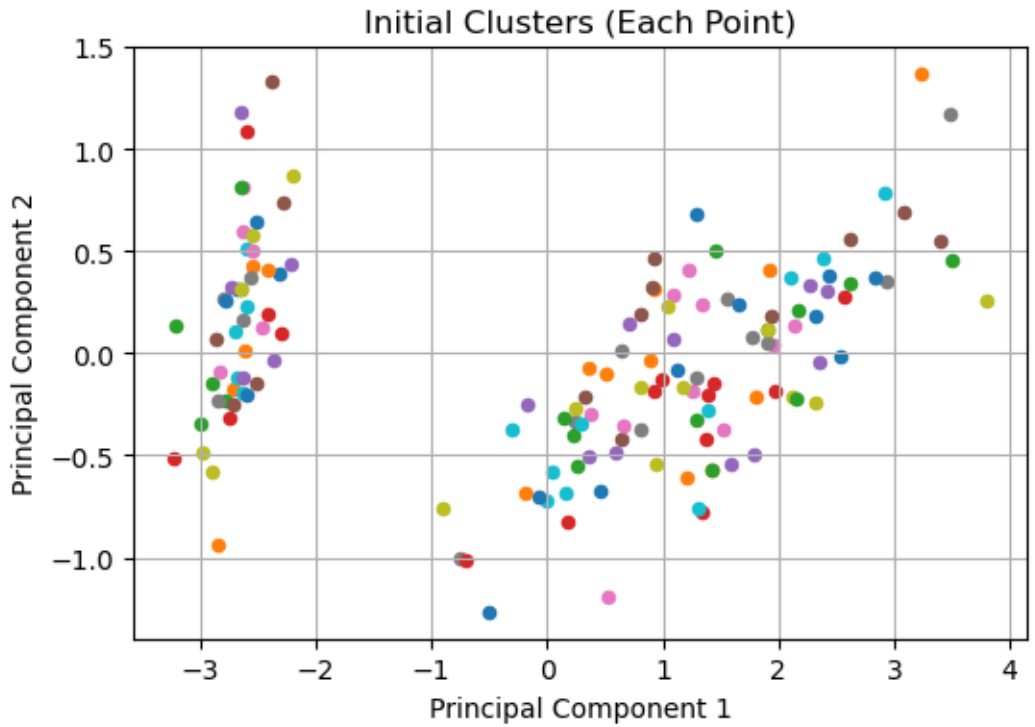
Hierarchical clustering

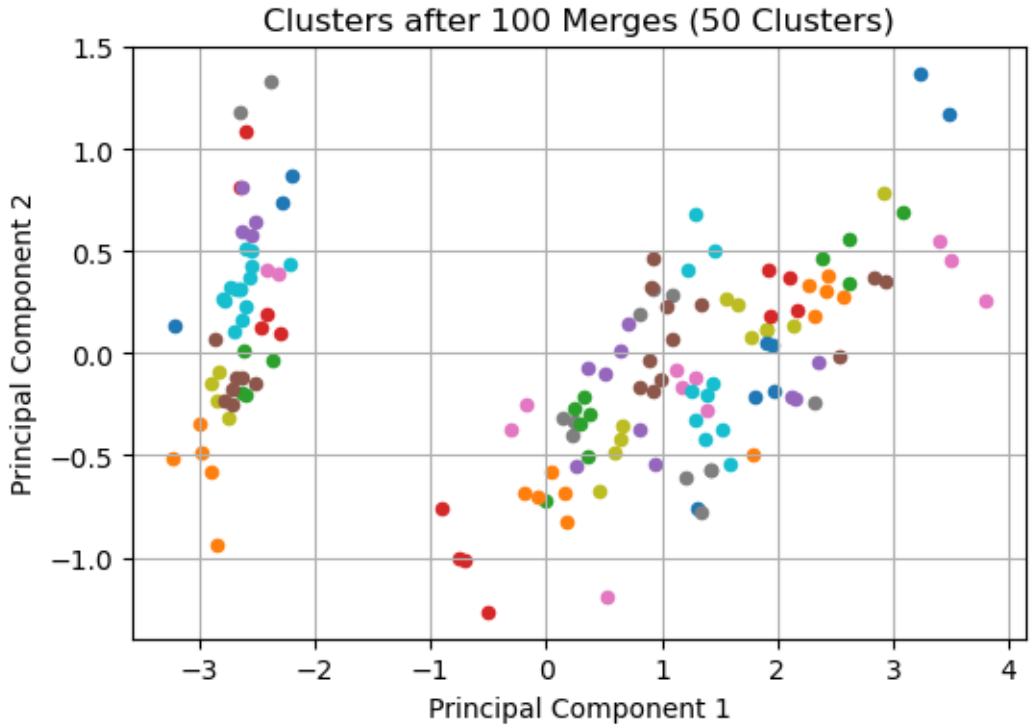
sklearn.cluster.AgglomerativeClustering



Hierarchical clustering (cont)

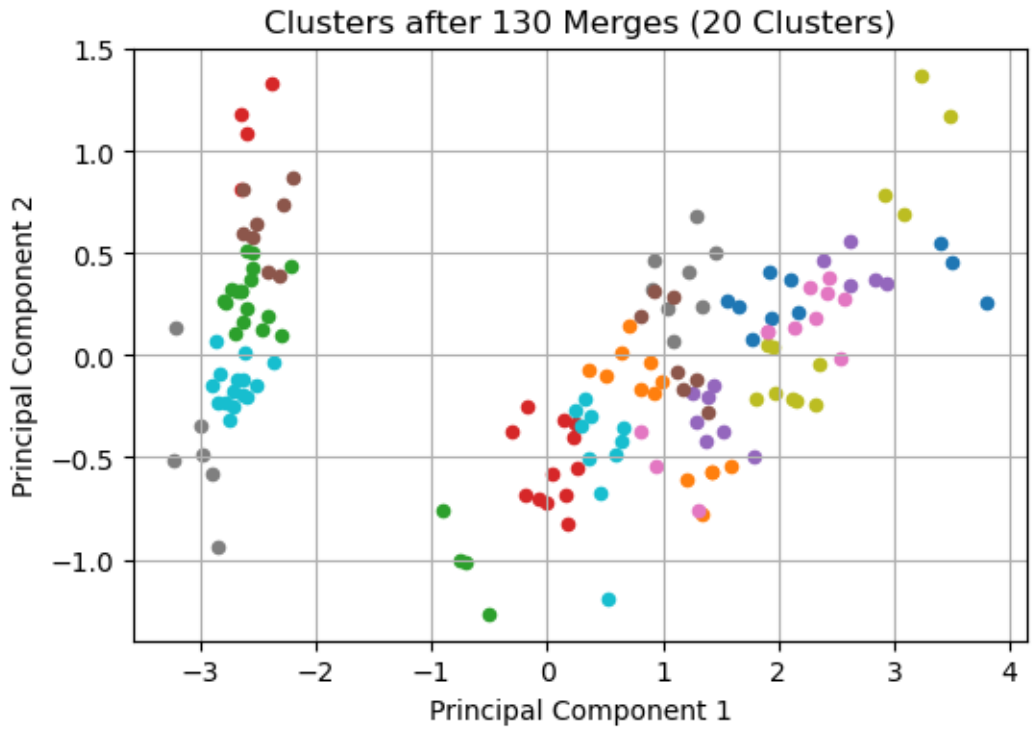
Iris dataset example: total 150 rows
Features are: Petal Length, Petal Width, Sepal Length, Sepal Width

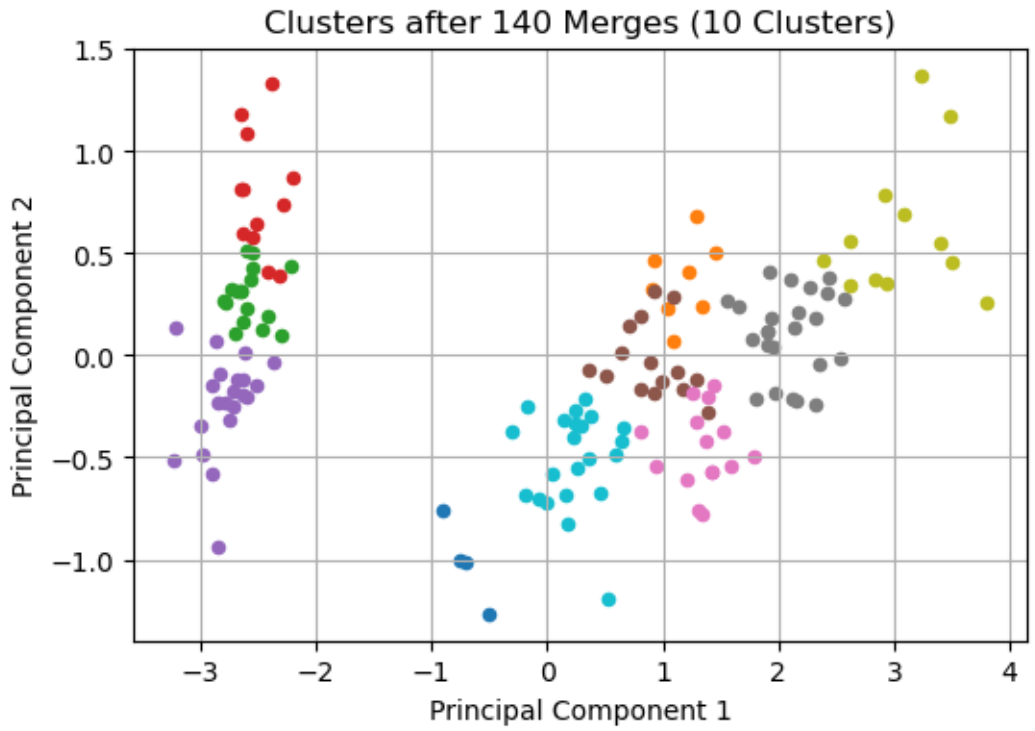




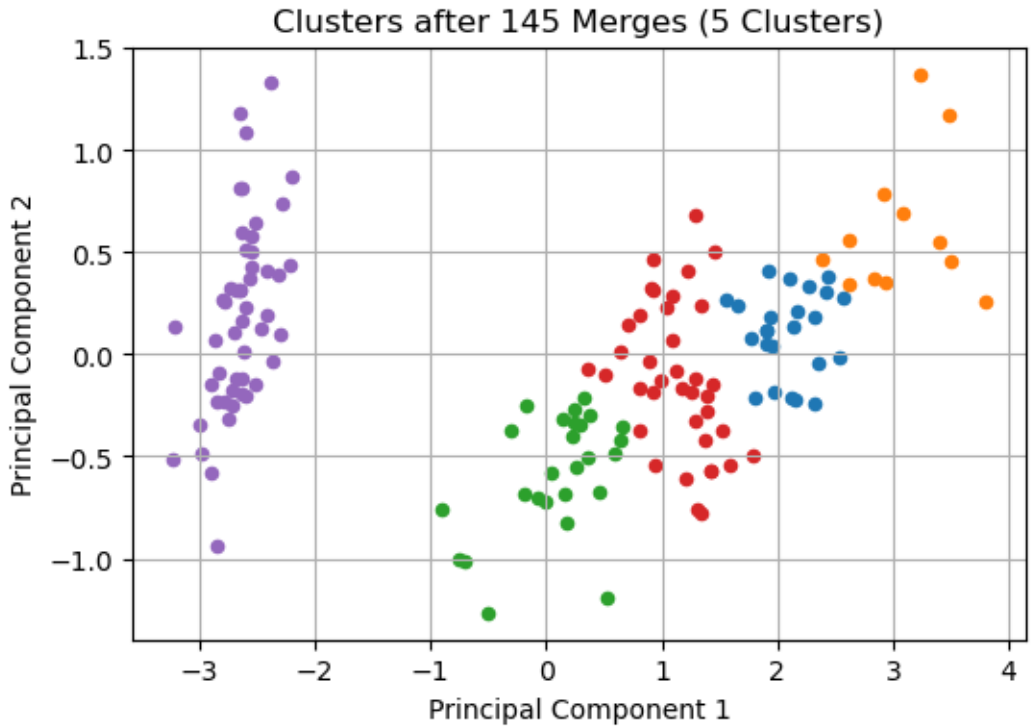
How did a 4d dataset fit on a 2d plot? (Foreshadowing)

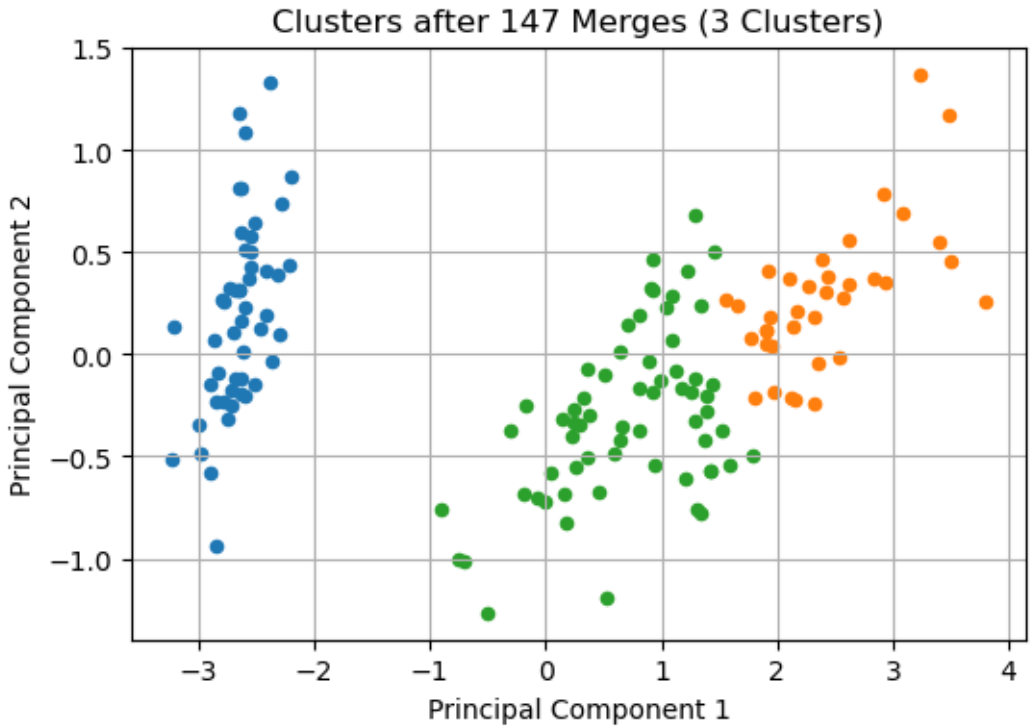
Hierarchical clustering (cont)



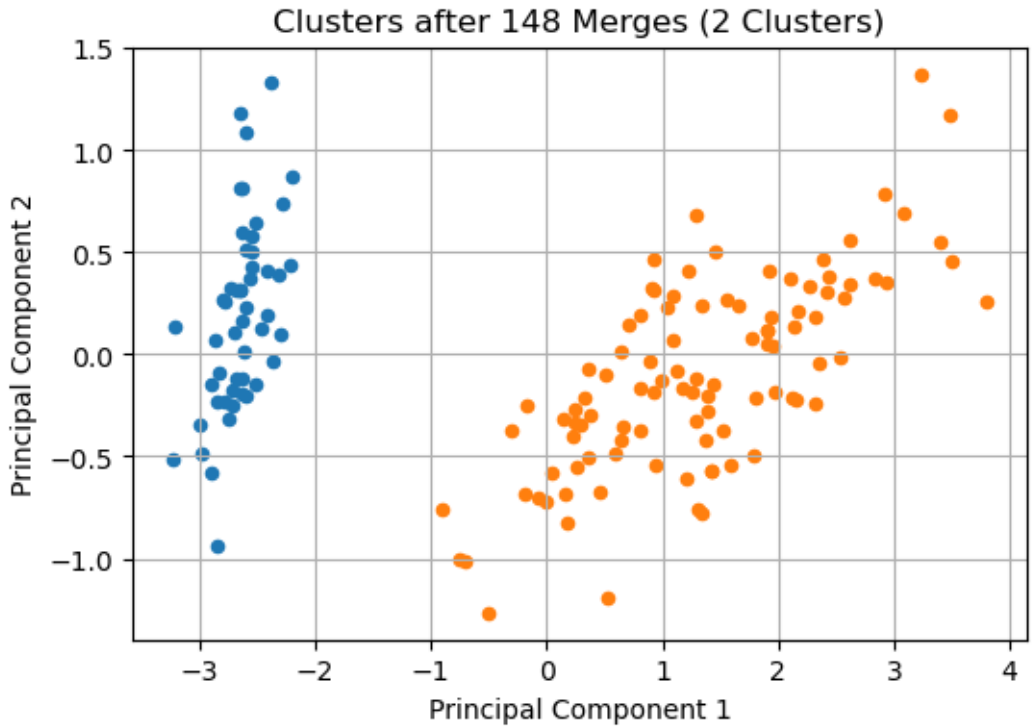


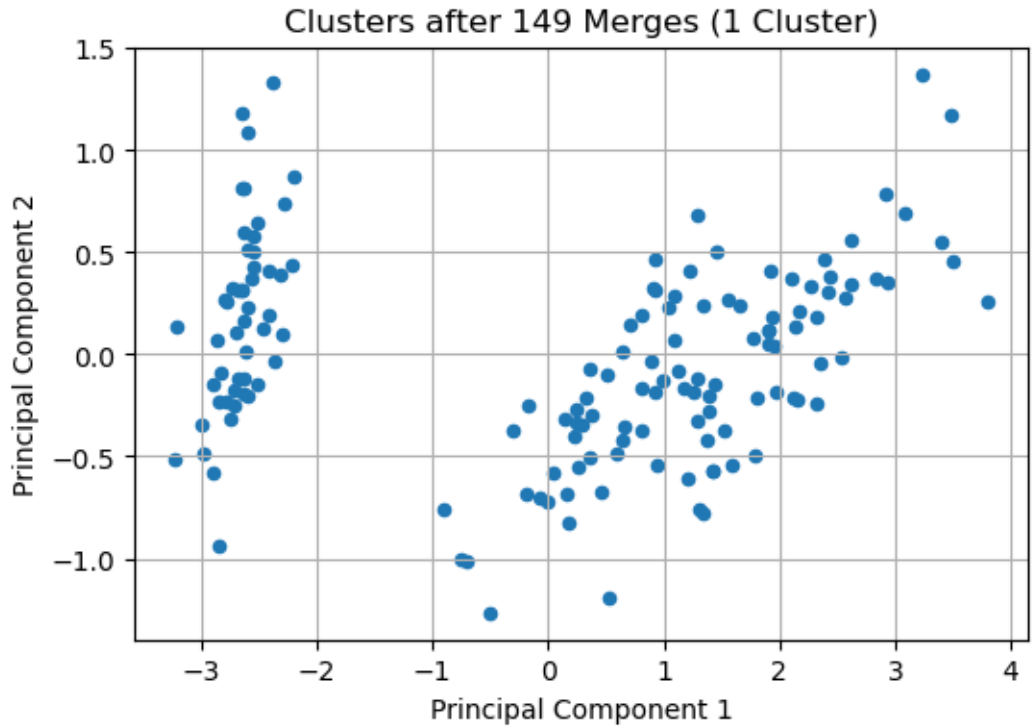
Hierarchical clustering (cont)





Hierarchical clustering (cont)





Clustering use cases tend to be more exploratory or are intermediary steps

- Note that when you want to accomplish these cases, sometimes you can do it with feature engineering which gives you more information
- General tip: think about the purpose of what you are trying to achieve
- You are doing clustering because you don’t have labels – think of it as you want to go to the second floor but there are no stairs so you have to bring your own ladder

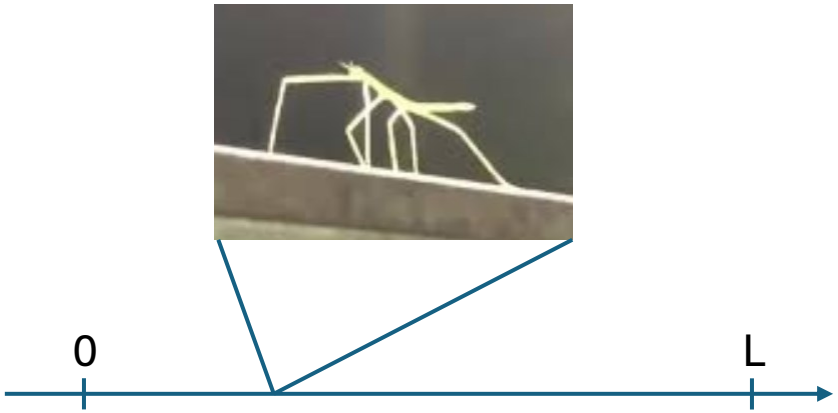
Principal Component Analysis

sklearn.decomposition.PCA

Coordinates of a bug travelling on this rope (3dims)

Source: Screenshot of Ikea website modified by GPT4o to add the rope

Coordinates of the same bug on the rope (1dim!)



PCA (cont)

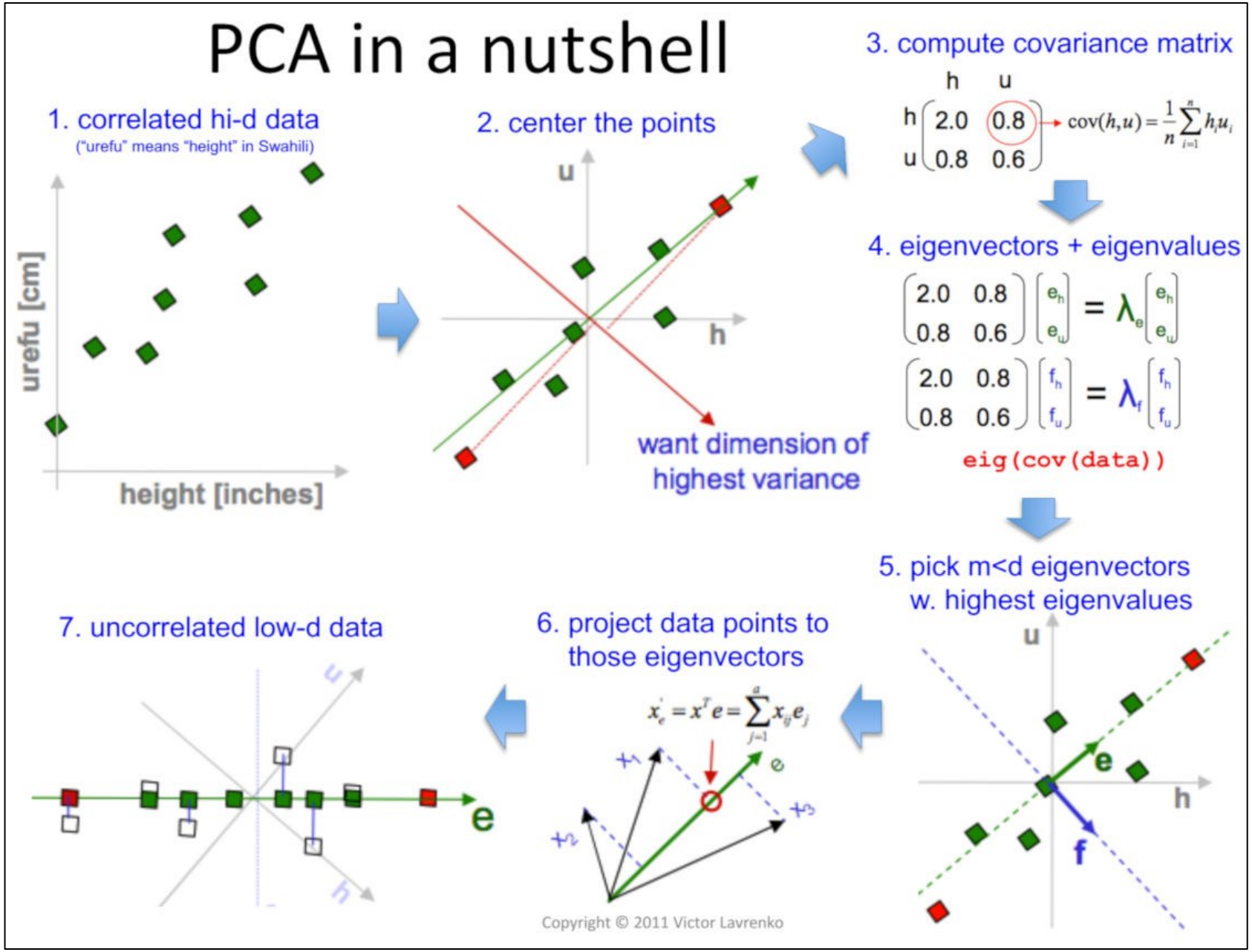
Can’t express this in the form of

$$Projected\ value\ on\ PC_n = a_1 d_1 + a_2 d_2 + \cdots + a_n d_n$$

PCA (cont)

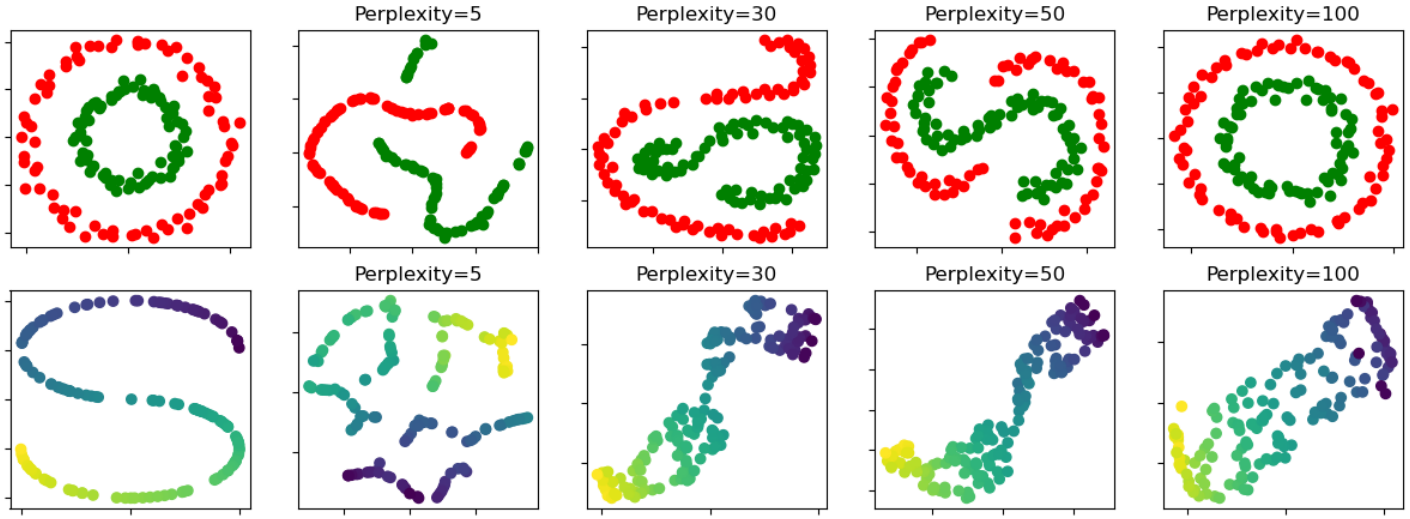
How are principal components determined?

Source: https://devopedia.org/principal-component-analysis



N for Neighbour

t-SNE



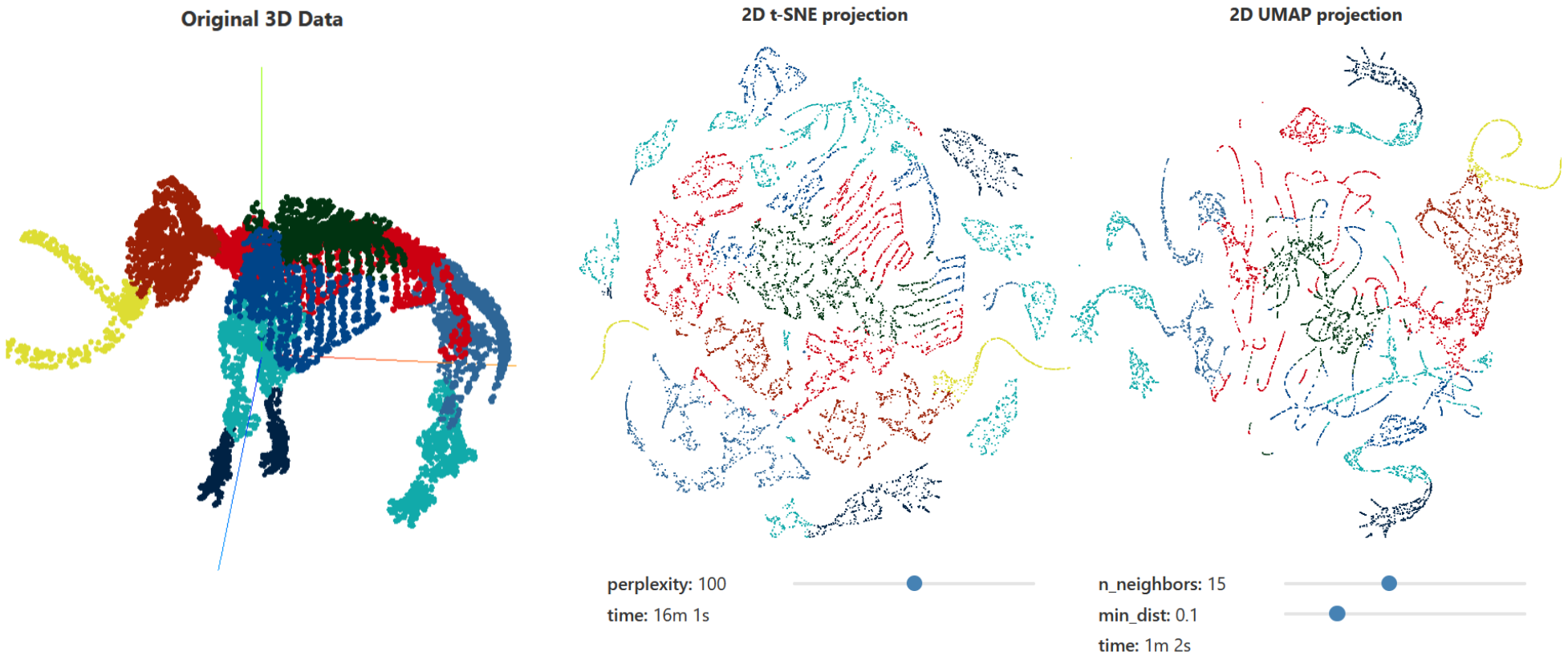
- Points are mapped to low-dim space from high-dim space
- Such that those points that are close in high-dim space stay close and v.v.
- Optimized by minimizing distance between 2 distributions: probability of neighboring in high-dim and low-dim space

Explanation: This party of travelers have a close friendship in their time (high-dimensional space). This same close relationship is preserved 40 thousand years into the future (low-dim space) in this comic.

M for Manifold

UMAP

Preserves global and local structure better than TSNE



https://pair-code.github.io/understanding-umap/

Lab 1 discussion

IOAI Training and Selection Programme 2025

Apr 19, 2025 Sat

Do transforms

```
transform = transforms.Compose([
    transforms.Resize(256),
    transforms.CenterCrop(224),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])
])

test_dataset = datasets.ImageFolder(root=test_dir, transform=transform)
test_loader = DataLoader(test_dataset, batch_size=32, shuffle=False)
```

Replace the final classification layer

```
# Load a pre-trained ResNet model
model = models.resnet34(pretrained=True)
model.fc = torch.nn.Linear(model.fc.in_features, 2)
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
model.to(device)
```

Log every 10 minibatches

```python
# Fine-tuning loop
for epoch in range(num_epochs):
    model.train()  # set model to training mode
    running_loss = 0.0
    train_preds = []
    train_labels = []

    # Iterate over the training data.
    for i, (inputs, labels) in enumerate(train_loader):
        inputs, labels = inputs.to(device), labels.to(device)

        optimizer.zero_grad()      # Clear gradients for this batch
        outputs = model(inputs)      # Forward pass
        loss = criterion(outputs, labels)  # Compute loss
        loss.backward()              # Backward pass (compute gradients)
        optimizer.step()             # Update weights

        running_loss += loss.item()
        preds = outputs.argmax(dim=1)
        train_preds.extend(preds.cpu().numpy())
        train_labels.extend(labels.cpu().numpy())

        # Every 10 minibatches, log the training and testing metrics.
        if (i + 1) % log_interval == 0:
            # Compute training metrics so far.
            train_acc = accuracy_score(train_labels, train_preds)
            train_prec = precision_score(train_labels, train_preds, average='macro', zero_division=0)
            train_rec = recall_score(train_labels, train_preds, average='macro', zero_division=0)

            # Evaluate on the test set.
            test_loss, test_acc, test_prec, test_rec = evaluate(model, test_loader)

            print(f"Epoch [{epoch+1}/{num_epochs}], Batch [{i+1}/{len(train_loader)}]:")
            print(f"Train Loss: {running_loss / log_interval:.4f}, "
                  f"Train Acc: {train_acc:.4f}, Train Prec: {train_prec:.4f}, Train Rec: {train_rec:.4f}")
            print(f"Test Loss: {test_loss:.4f}, "
                  f"Test Acc: {test_acc:.4f}, Test Prec: {test_prec:.4f}, Test Rec: {test_rec:.4f}")
```

Lab 2 discussion

IOAI Training and Selection Programme 2025

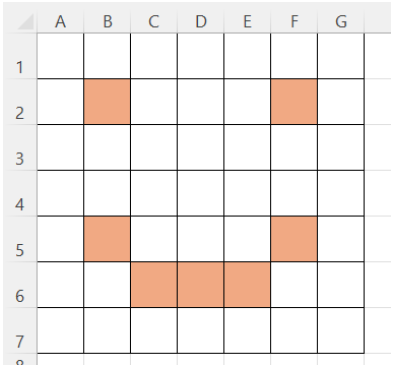
Apr 19, 2025 Sat

Why are predictions not good

A lot of information is lost when the image reaches the segmentation head

```
head = nn.Sequential(
    nn.Conv2d(512, num_classes, kernel_size=1),
    nn.ConvTranspose2d(
        in_channels=num_classes,
        out_channels=num_classes,
        kernel_size=64,
        stride=32,  # Upsample by 32×
        padding=16,
        bias=False,
    ),  # Transpose conv for upsampling
)
```

After the nn.Conv2d, an 224x224 image is reduced to 7x7 with 512 channels!



This is how small 7x7 is

plant0134_fg.png
plant0134_rgb.png
plant0135_fg.png
plant0135_rgb.png

The train and test dataset are slightly different!

plantA006_fg.png
plantA006_rgb.png
plantA007_fg.png
plantA007_rgb.png

plantB034_fg.png
plantB034_rgb.png
plantB036_fg.png
plantB036_rgb.png

plantC024_fg.png
plantC024_rgb.png
plantC025_fg.png
plantC025_rgb.png

Residual connections akin to U-Net

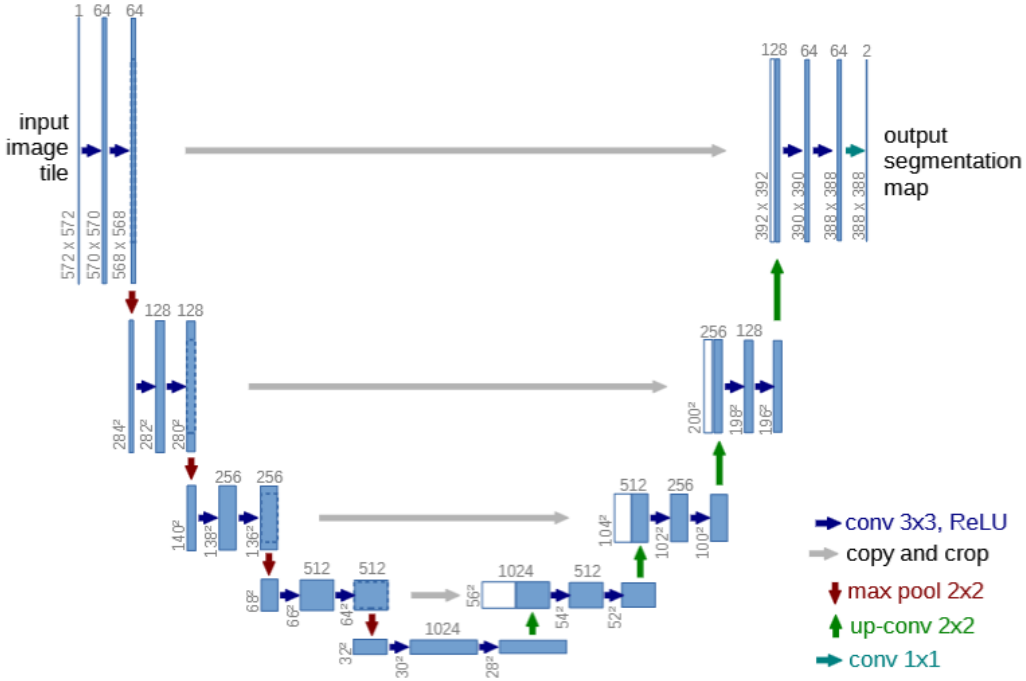


Fig. 1. U-net architecture (example for 32x32 pixels in the lowest resolution). Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations.

Selection 1 announcement

- No class next week, will have Selection 1 test instead.
- Will be same format and timing as MAIO using the same platform Fri 8pm – Sun 8pm
- Score higher than a passing score + turn in Lab 2 to continue
- Not meant to be difficult if you have been learning along as we go!

Romania Easter Round Discussion

IOAI Training and Selection Programme 2025

Apr 19, 2025 Sat