Week10
Dimensionality Reduction & Model Finetuning

PRESENTER: JENNA KIM

COURSE: IS597MLC-SP2024

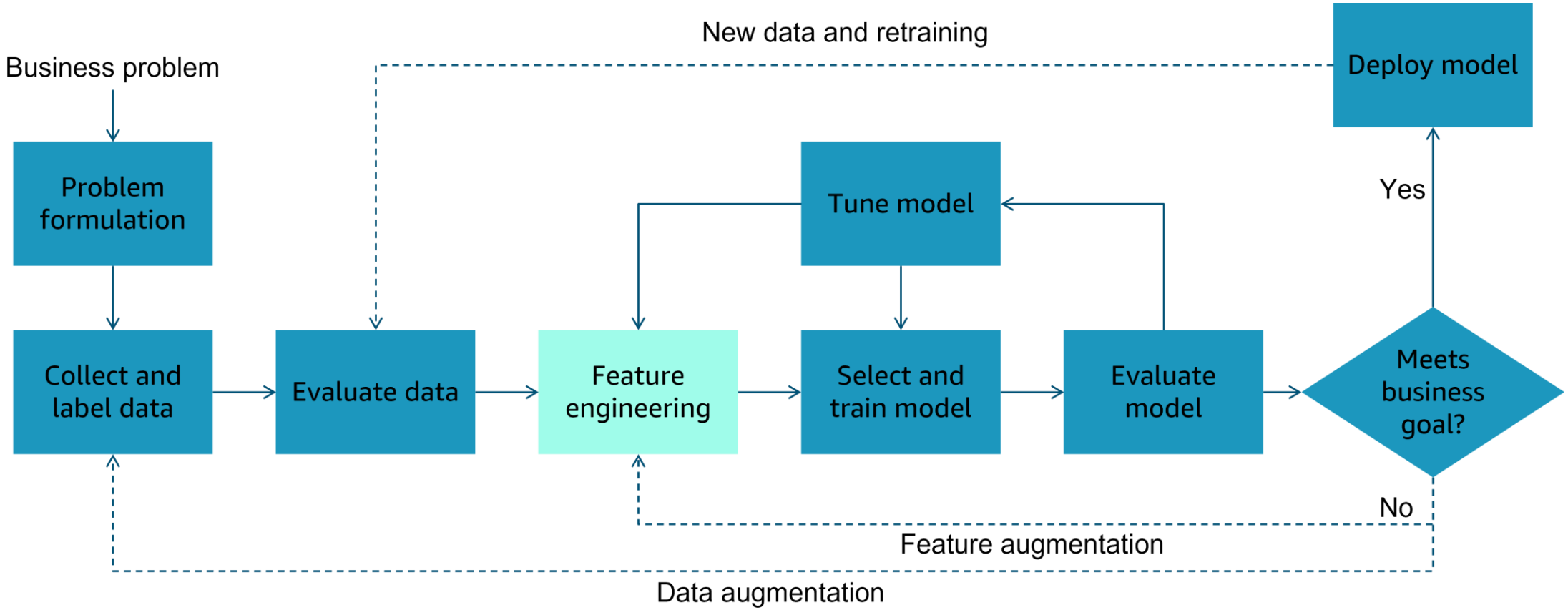
MARCH 25, 2024

Outline

- Dimensionality Reduction
- Model Finetuning

Part 1
Dimensionality Reduction

ML Workflow: Feature Engineering



Source: Amazon Web Services

Feature Engineering

- **Process of applying your knowledge of the data to create better features to train your model with**
- **What to consider:**
  - Which features should I use?
  - Do I need to transform these features in some way?
  - How do I handle missing data?
  - Do I need to create new features from the existing ones?
- **You cannot just give raw data to a model and expect good results.**
- **This is where expertise such as domain knowledge comes into play.**

The Curse of Dimensionality

- **Too many features can be problematic.**
- **Example:**
  - To predict how much money to make based on attributes of the people
  - Features: age of a person, height, weight, address, car a person drives, etc.
- **Feature engineering:**
  - Select the features most relevant to the problem at hand
  - Domain knowledge comes into play

Dimensionality Reduction

- **Attempts to distill higher dimensional data down to a smaller number of dimensions**
- **While preserving as much of the variance in the data as possible**
- **Reduce many features into fewer most important features**
- **Unsupervised dimensionality reduction techniques**
  - PCA (Principal Component Analysis)
  - K-Means
- **The goal is to distill many features into fewer features**

Dimensionality Reduction Techniques

K-Means Clustering

- A dimensionality reduction algorithm
- Reduce data down to K dimensions

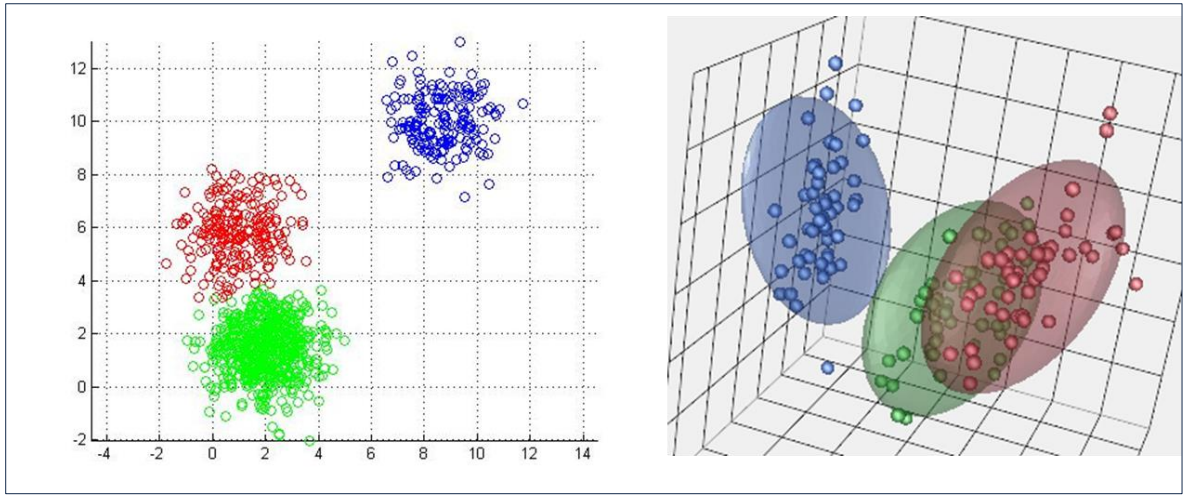


Image source: https://stanford.edu/~cpiech/cs221/handouts/kmeans.html

Dimensionality Reduction Techniques

Principal Component Analysis (PCA)

- **Finds eigenvectors in the higher dimensional data**
  - These define hyperplanes that split the data while preserving the most variance in it.
  - The data gets projected onto these hyperplanes, which represent the lower dimensions.
- **Also useful for image compression and facial recognition**

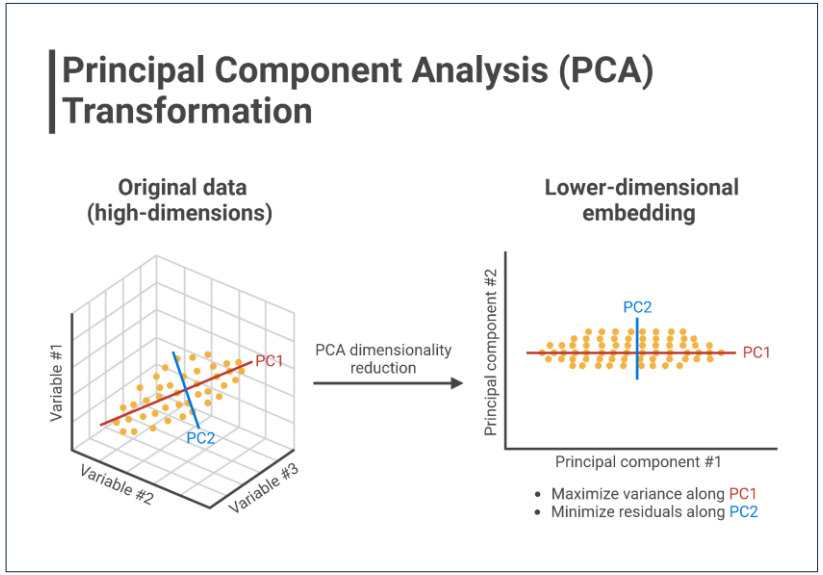


Image source: https://www.biorender.com/template/principal-component-analysis-pca-transformation

Example: Iris Flower Data

- **Iris dataset: comes with Scikit-learn**
- **An Iris flower has petals and sepals (the lower, supportive part)**
- **The length and width of the petals and sepal for many Iris specimens**
  - 4 dimensions for 3 different kinds of flowers
  - Subspecies classification of each flower
- **PCA allows us visualize this in 2 dimensions instead of 4, while still preserving the most info.**

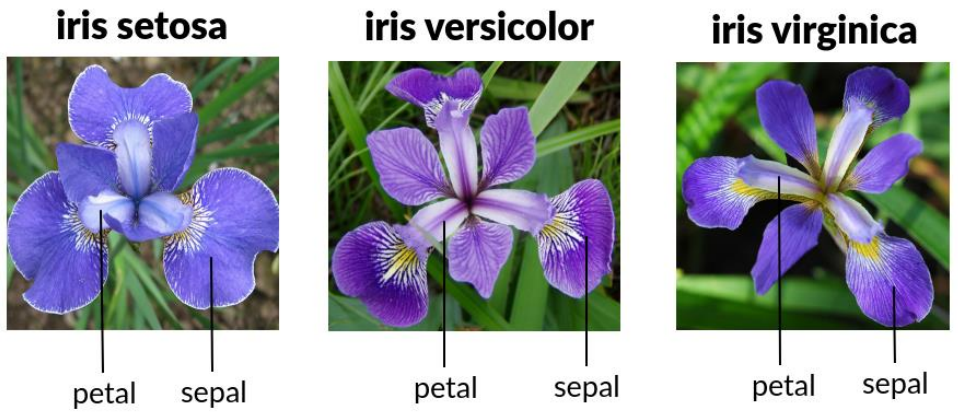


Image source: https://www.analyticsvidhya.com/blog/2022/06/iris-flowers-classification-using-machine-learning/

Code Example: PCA

Documentation:
https://scikit-learn.org/stable/auto_examples/datasets/plot_iris_dataset.html#sphx-glr-auto-examples-datasets-plot-iris-dataset-py

```
from sklearn.datasets import load_iris
from sklearn.decomposition import PCA
import pylab as pl
from itertools import cycle

iris = load_iris()
num_sample, num_feature = iris.data.shape

print(num_sample)
print(num_feature)
print(list(iris.target_names))
```

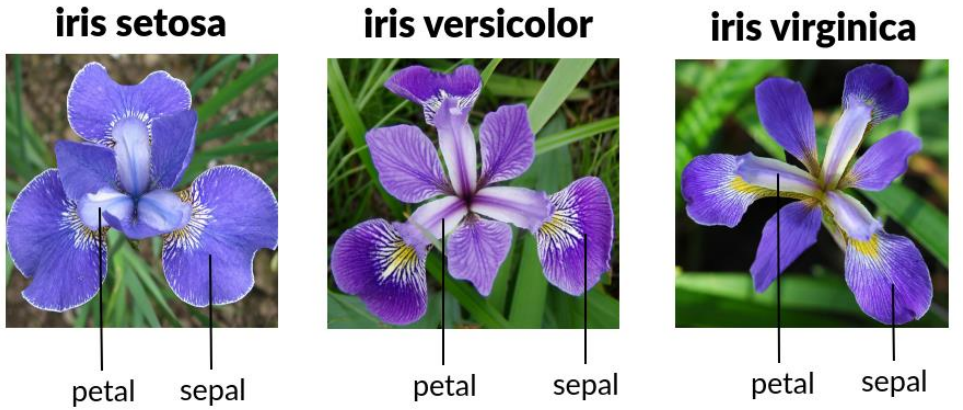


Image source: https://www.analyticsvidhya.com/blog/2022/06/iris-flowers-classification-using-machine-learning/

Code Example: PCA

- Let's take a look at the details by running the code in a Jupyter Notebook
- A video is prepared for demonstrating how to apply PCA on the Iris dataset.
- This video is provided in a Weekly Schedule.

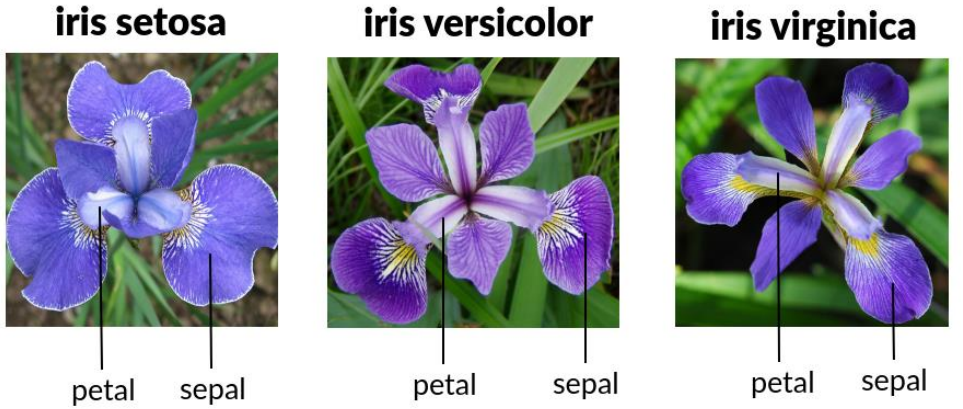


Image source: https://www.analyticsvidhya.com/blog/2022/06/iris-flowers-classification-using-machine-learning/

Part 2
Model Finetuning

Cross Validation (CV)

- An advance methods for splitting data into training and testing sets.
- The goal of train test split is to fairly evaluate a model's performance on unseen data.
- Not able to tune hyperparameters to the entire dataset.
- There is a way to train all the data and evaluate all the data.
- We can achieve this with cross validation.

Cross Validation (CV): Reasons & Steps

- Reasons for using cross validation during ML process:
  - Tuning model hyperparameters
  - Testing different properties of the overall datasets
  - Iterating the training process
  - In case where your training dataset is small
  - Splitting them into 3 subsets may significantly affect training accuracy.
- Two steps
  - Splitting the data into subsets (called folds)
  - Rotating the training and test (validation) among them

Cross Validation (CV): Properties

- Each fold with approximately the same size.
- Randomly selected data in each fold or stratified.
- All folds are used to train the model except one for validation.
- The validation fold should be rotated until all folds have become a validation fold only once.
- Each example is recommended to be contained in one and only one fold.
- K-fold and CV are interchangeably used.
- K-fold describes how many folds you want to split your dataset into.

  - e.g., if k=10, represeting 90% (training set) & 10% (validation set)

Cross Validation (CV): Process

	Fold-1	Fold-2	Fold-3	Fold-4	Fold-5	Fold-6	Fold-7	Fold-8	Fold-9	Fold-10
Step-1	Train	Train	Train	Train	Train	Train	Train	Train	Train	Test
Step-2	Train	Train	Train	Train	Train	Train	Train	Train	Test	Train
Step-3	Train	Train	Train	Train	Train	Train	Train	Test	Train	Train
Step-4	Train	Train	Train	Train	Train	Train	Test	Train	Train	Train
Step-5	Train	Train	Train	Train	Train	Test	Train	Train	Train	Train
Step-6	Train	Train	Train	Train	Test	Train	Train	Train	Train	Train
Step-7	Train	Train	Train	Test	Train	Train	Train	Train	Train	Train
Step-8	Train	Train	Test	Train	Train	Train	Train	Train	Train	Train
Step-9	Train	Test	Train	Train	Train	Train	Train	Train	Train	Train
Step-10	Test	Train	Train	Train	Train	Train	Train	Train	Train	Train

Figure 2: A 10-fold representation of how each fold is used in the cross-validation process.

Image source: https://towardsdatascience.com/what-is-cross-validation-60c01f9d9e75

Scikit-learn

- **scikit-learn**

  from sklearn.model_selection import cross_validate

- **Cross Validation Documentation**
  https://scikit-learn.org/stable/modules/cross_validation.html

Grid Search

- Complex models often have multiple adjustable hyperparameters.
- A grid search is a way of training and validating a model on every possible combination of multiple hyperparameter options.
- Scikit-learn includes a GridSearchCV class
- It is capable of testing a dictionary of multiple hyperparameter options through cross-validation.
- This allows for both cross-validation and a grid search to be performed in a generalized way for any model.

Grid Search

- **Scikit-learn**

```
from sklearn.model_selection import GridSearchCV
```

- **Grid Search Documentation**

  https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

- **Let's watch a video demonstrating how to use Grid Search with Cross Validation.**

Questions or Comments?

Thank You!