Lecture 4: k-nearest neighbours and SVM RBFs

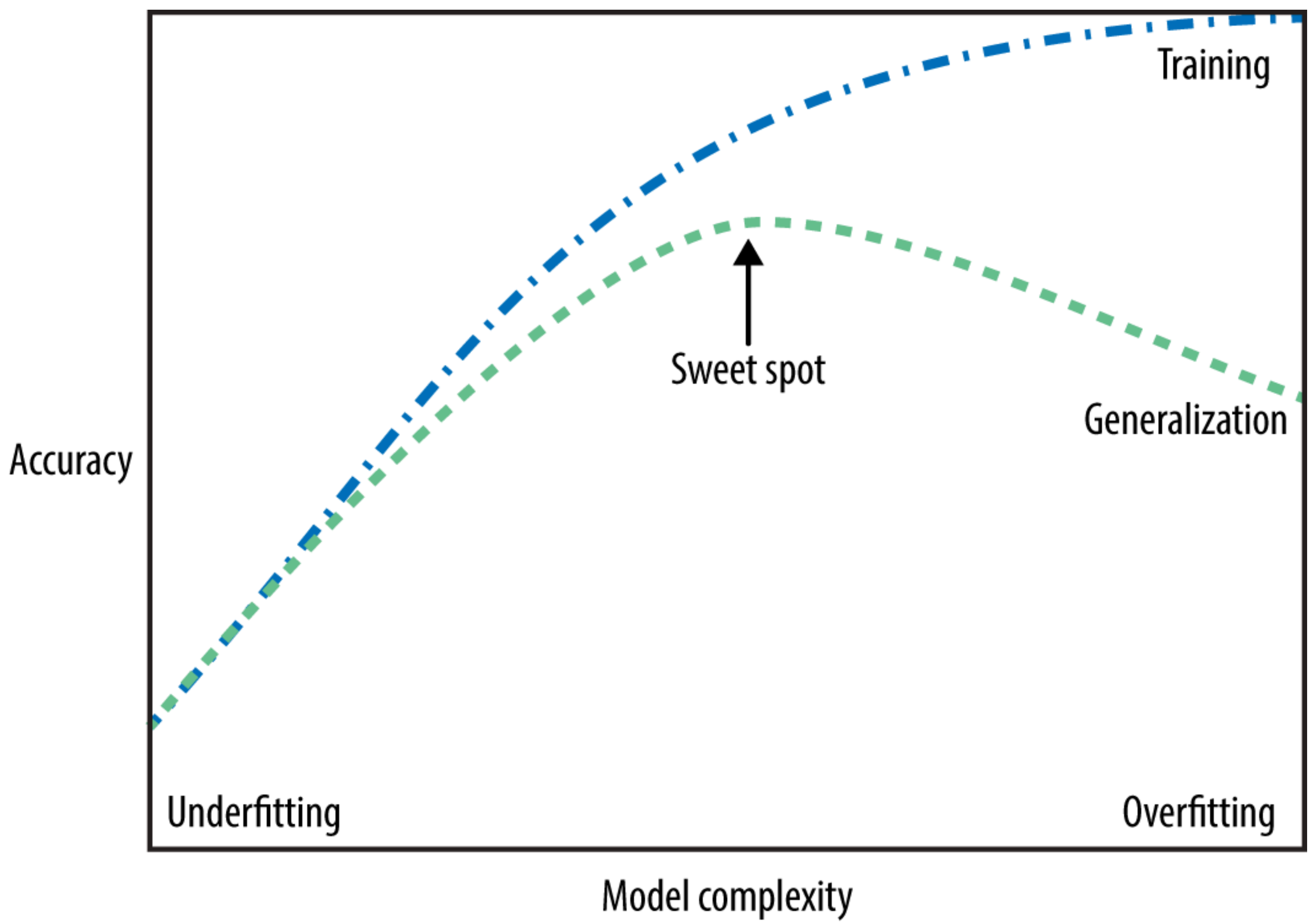
Firas Moosvi (Slides adapted from Varada Kolhatkar)

Pod Work: Discuss these questions

- Why do we split data?
- What are train/valid/test splits?
- What are the benefits of cross-validation?
- What's the fundamental trade-off in supervised machine learning?
- What is the golden rule of machine learning?

Recap: The fundamental tradeoff

As you increase the model complexity, training score tends to go up and the gap between train and validation scores tends to go up.



Pod Work: Discuss this question

Which of the following statements about **overfitting** is true?

- a. Overfitting is always beneficial for model performance on unseen data.
- b. Some degree of overfitting is common in most real-world problems.
- c. Overfitting ensures the model will perform well in real-world scenarios.
- d. Overfitting occurs when the model learns the training data too closely, including its noise and outliers.

Pod Work: Discuss this question

Which of the following scenarios do **NOT necessarily imply overfitting**?

- a. Training accuracy is 0.98 while validation accuracy is 0.60.
- b. The model is too specific to the training data.
- c. The decision boundary of a classifier is wiggly and highly irregular.
- d. Training and validation accuracies are both approximately 0.88.

Pod Work: Discuss this question

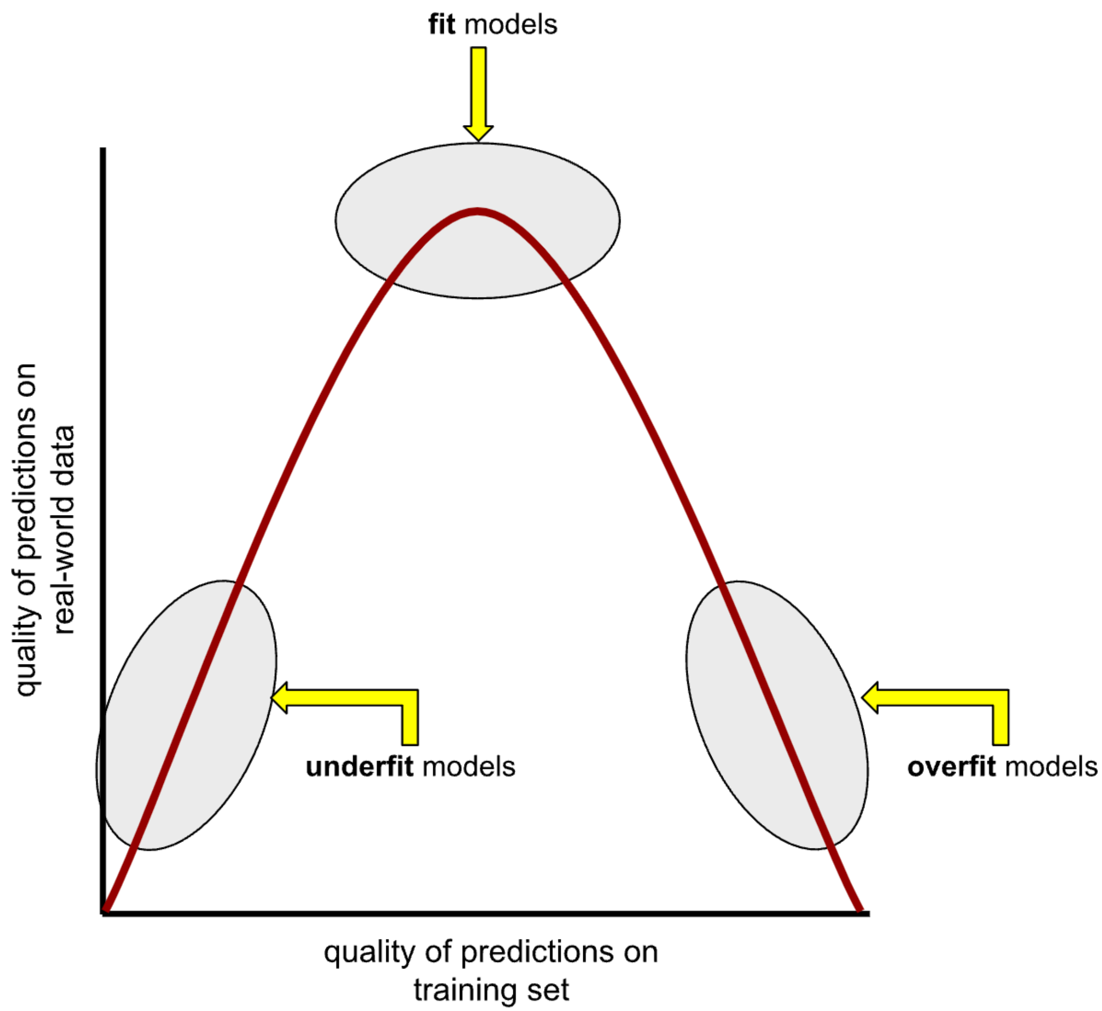
How might one address the issue of **underfitting** in a machine learning model.

- a. Introduce more noise to the training data.
- b. Remove features that might be relevant to the prediction.
- c. Increase the model's complexity, possibly by adding more parameter or features
- d. Use a smaller dataset for training.
- e. Use a larger dataset for training.

Overfitting and underfitting

- An **overfit model** matches the training set so closely that it fails to make correct predictions on new unseen data.
- An **underfit model** is too simple and does not even make good predictions on the training data

Overfitting and underfitting



Source

iClicker 4.1

iClicker cloud join link: https://join.iclicker.com/YJHS

Select all of the following statements which are TRUE.

- a. Analogy-based models find examples from the test set that are most similar to the query example we are predicting.
- b. Euclidean distance will always have a non-negative value.
- c. With k-NN, setting the hyperparameter k to larger values typically reduces training error.
- d. Similar to decision trees, k-NNs finds a small set of good features.
- e. In k-NN, with $k > 1$, the classification of the closest neighbour to the test example always contributes the most to the prediction.

iClicker 4.2

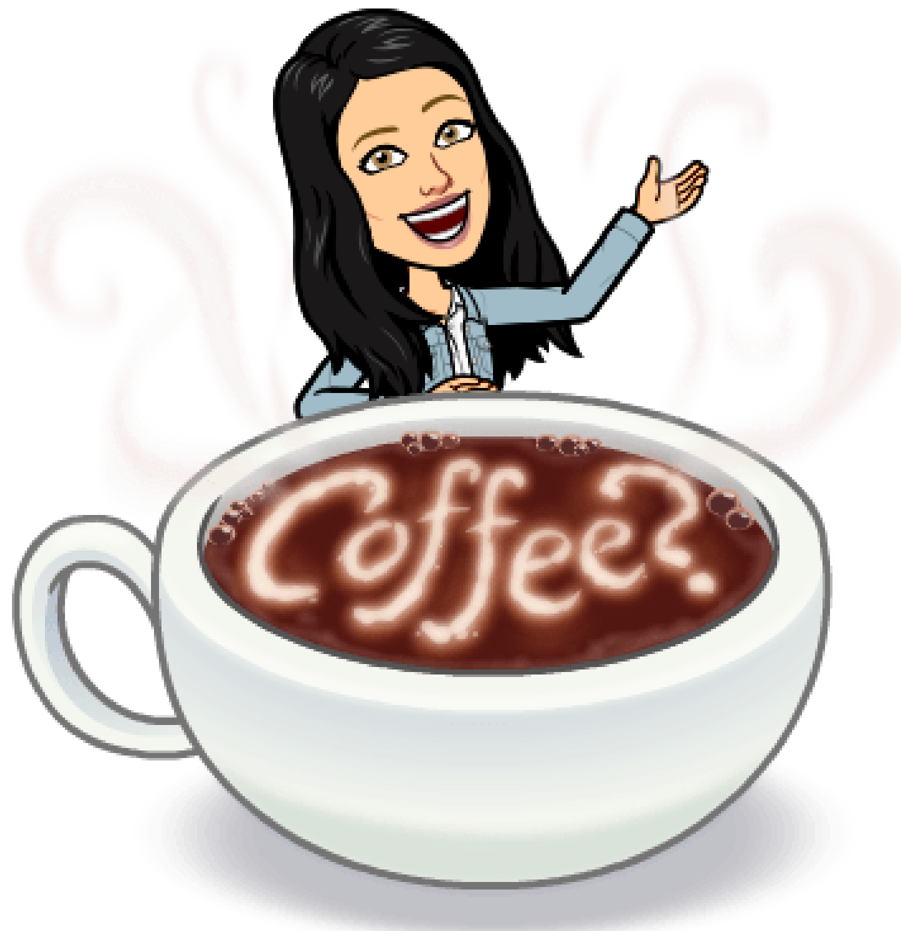
iClicker cloud join link: https://join.iclicker.com/YJHS

Select all of the following statements which are TRUE.

- a. k-NN may perform poorly in high-dimensional space (say, $d > 1000$).
- b. In sklearn's SVC classifier, large values of `gamma` tend to result in higher training score but probably lower validation score.
- c. If we increase both `gamma` and `C`, we can't be certain if the model becomes more complex or less complex.

Break

Let’s take a break!



Similarity-based algorithms

- Use similarity or distance metrics to predict targets.
- Examples: k-nearest neighbors, Support Vector Machines (SVMs) with RBF Kernel.

k-nearest neighbours

- Classifies an object based on the majority label among its k closest neighbors.
- Main hyperparameter: k or `n_neighbors` in `sklearn`
- Distance Metrics: Euclidean
- Strengths: simple and intuitive, can learn complex decision boundaries
- Challenges: Sensitive to the choice of distance metric and **scaling** (coming up).

Curse of dimensionality

- As dimensionality increases, the volume of the space increases exponentially, making the data sparse.
- Distance metrics lose meaning
  - Accidental similarity swamps out meaningful similarity
  - All points become almost equidistant.
- Overfitting becomes likely: Harder to generalize with high-dimensional data.
- How to deal with this?
  - Dimensionality reduction (PCA) (not covered in this course)
  - Feature selection techniques.

SVMs with RBF kernel

- RBF Kernel: Radial Basis Function, a way to transform data into higher dimensions implicitly.
- Strengths
  - Effective in high-dimensional and sparse data
  - Good performance on non-linear problems.
- Hyperparameters:
  - C: Regularization parameter (trade-off between correct classification of training examples and maximization of the decision margin).
  - γ (Gamma): Defines how far the influence of a single training example reaches.

Intuition of `C` and `gamma` in SVM RBF

- `C` (Regularization): Controls the trade-off between perfect training accuracy and having a simpler decision boundary.
  - High C: Strict, complex boundary (overfitting risk).
  - Low C: More errors allowed, smoother boundary (generalizes better).
- `Gamma` (Kernel Width): Controls the influence of individual data points.
  - High Gamma: Points have local impact, complex boundary.
  - Low Gamma: Points affect broader areas, smoother boundary.
- Key trade-off: Proper balance between `C` and `gamma` is crucial for avoiding overfitting or underfitting.