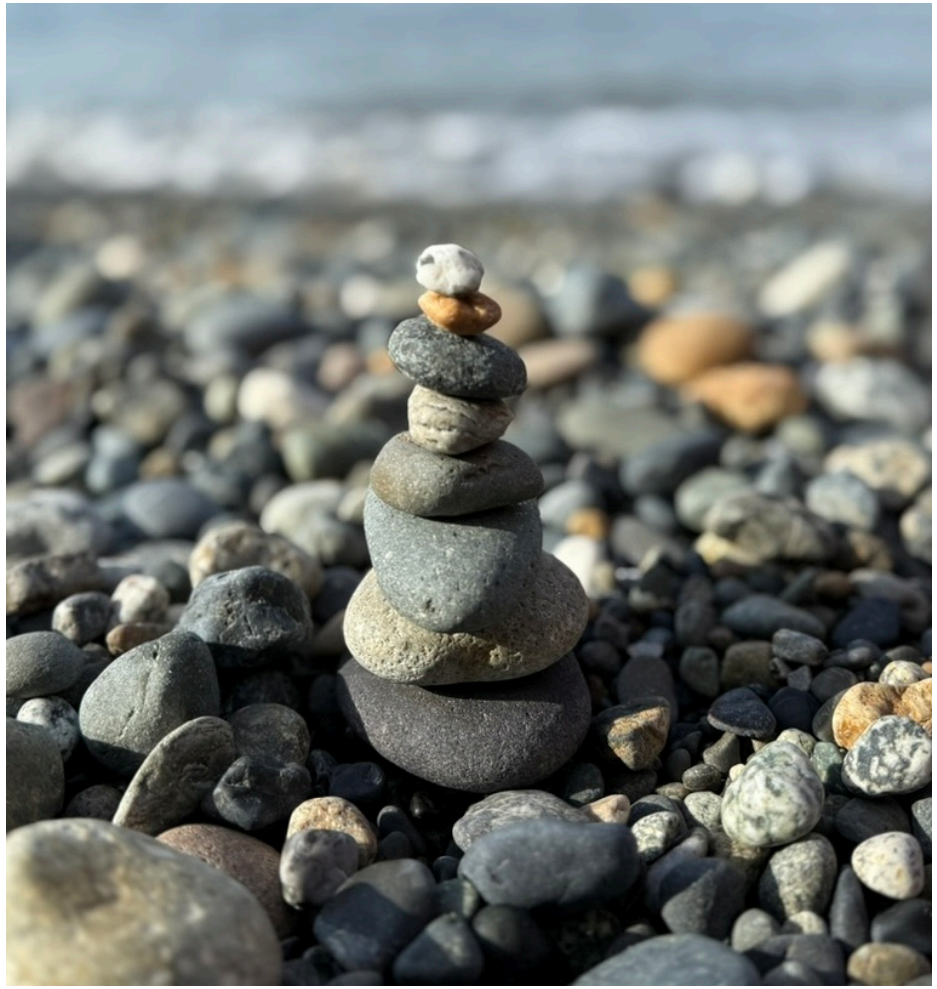
Lecture 4: k-nearest neighbours and SVM RBFs

Varada Kolhatkar

Focus on the breath!



Announcements

- HW2 was due yesterday
- Homework 3 (hw3) has been released (Due: Oct 5th, 11:59 pm)

Recap: iClicker overfitting 1

Link: https://join.iclicker.com/

Which scenarios, by themselves, do NOT provide sufficient evidence that a model is overfitting? Select all that apply.

- a. Training accuracy is 0.98, while validation accuracy is 0.60.
- b. A wildlife classifier learns that snow in the background is strongly associated with “wolf” in the training data. On validation images where wolves appear in a variety of backgrounds, its performance drops substantially.
- c. A classifier has a highly irregular decision boundary, but you are not given its training or validation performance.
- d. Training and validation accuracies are both approximately 0.88.
- e. A cancer-detection model learns that a ruler in the corner of an image is associated with positive cases. This association was present in the training data but not in validation data, where performance drops substantially.

Recap: iClicker overfitting 2

Link: https://join.iclicker.com/GHJB

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

- a. Overfitting makes the model more accurate on both training and unseen data.
- b. Overfitting means the model captures noise or irrelevant details from the training data.
- c. Overfitting is desirable because it reduces both training and test error.
- d. Overfitting can be identified by looking at training performance alone.

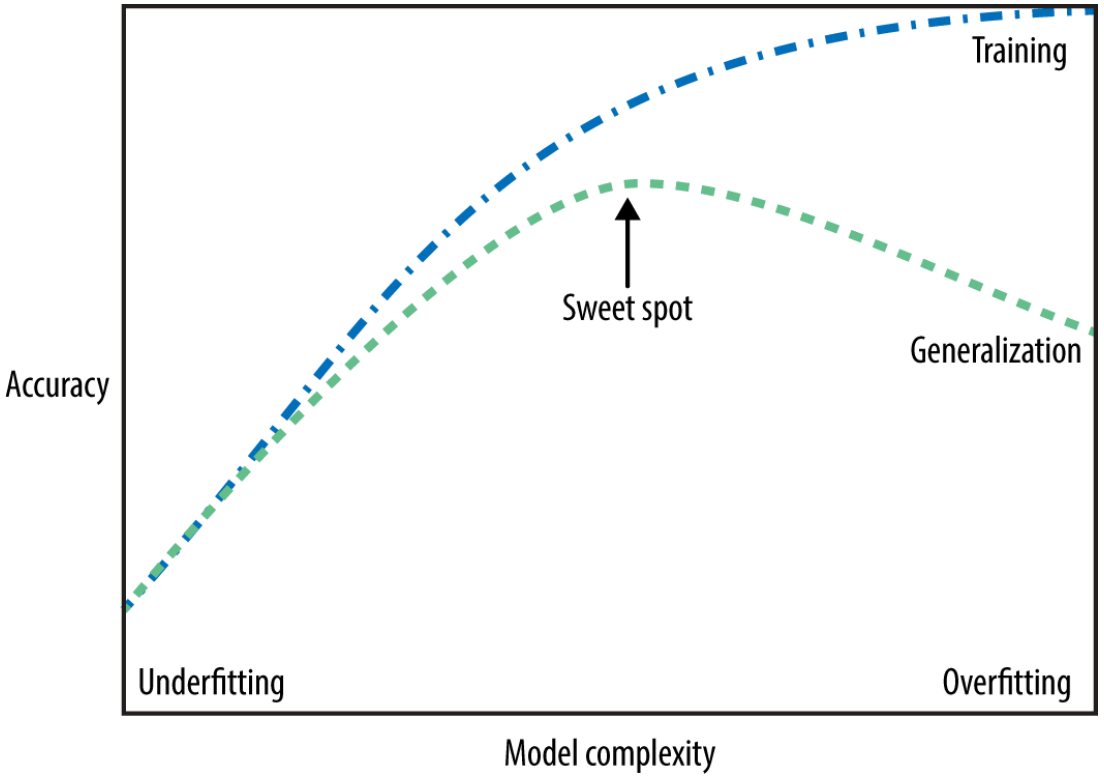
Recap: iClicker underfitting

Link: https://join.iclicker.com/GHJB

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 (e.g., more parameters, useful features, or deeper trees)
- d. Use a smaller dataset for training.

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.
- How to pick a model?
  - Compare validation scores using cross-validation on the training data.
  - Don’t choose based on training score alone.

Today's big questions

- How do **k-NN and RBF SVMs** use similarity to make predictions?
- How do **k, C, and γ** affect model complexity and generalization?
- Why do **feature scales and the number of features** matter for similarity-based models?

We'll explore these questions through **visual examples and prediction activities**.

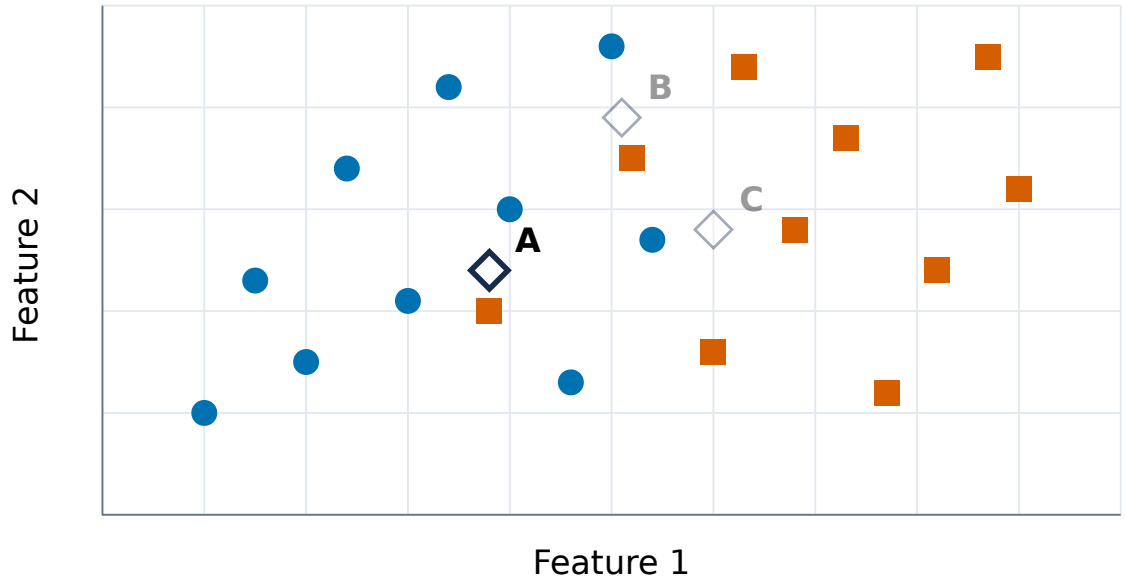
Similarity-based algorithms

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

k-nearest neighbours intuition

Predict the query point's class using a majority vote of its k nearest neighbours.

Query A Neighbours (k) 1 Reveal neighbours & prediction



Query A, k = 1: Which class do you predict?

Euclidean distance · Equal votes · Classes: Blue and Orange

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).

iClicker 4.1

Link: https://join.iclicker.com/GHJB

Select all of the following statements which are TRUE.

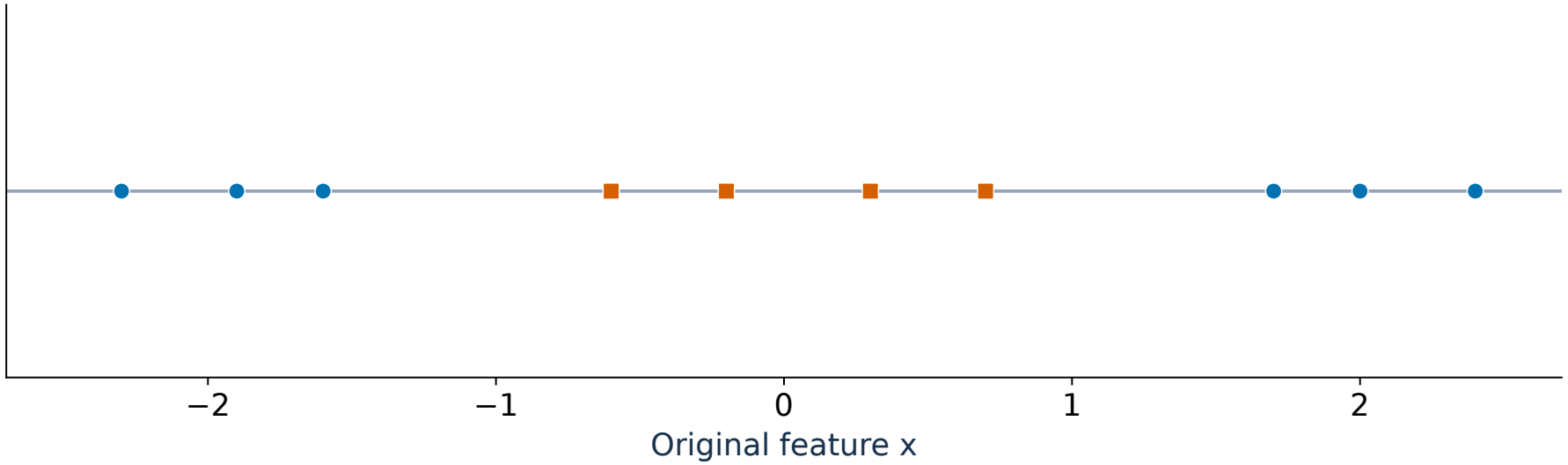
- a. Analogy-based models find examples from the test set that are most similar to the query example we are predicting.
- b. Euclidean distance will always have a non-negative value.
- c. With k-NN, setting the hyperparameter k to larger values typically reduces training error.
- d. Similar to decision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

Curse of dimensionality

- As dimensionality increases, the space gets much bigger and **data points become more spread out**.
- Distances become less informative:
  - Even the nearest neighbours may be far away.
  - Points start to look similarly distant.
- This makes it harder to generalize from limited data.
- How to deal with this?
  - Dimensionality reduction (e.g., PCA)
  - Feature selection

RBF SVM intuition

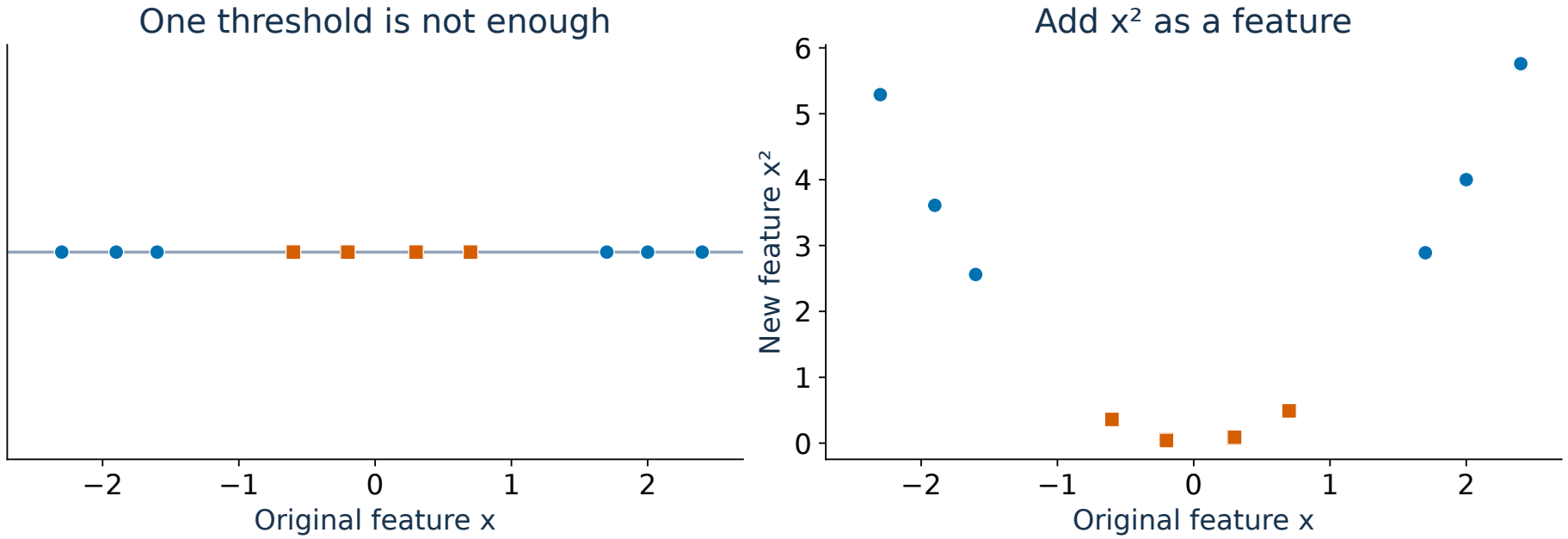
Can a decision stump classify these points perfectly?



A decision stump makes **one split**. Where would you split on x?

No single threshold works: the Orange points lie between two Blue groups.

What if we add x^2 as a feature?



Can a **decision stump** classify these points perfectly now?

Yes! Split on $x^2 \leq 1.5$: **Orange** below the threshold, **Blue** above it.

From new features to kernels

Changing the representation can make classification easier.

- A tree needed two splits on x, but only **one split on x^2**.
- A **support vector machine (SVM)** also learns a decision boundary.
- **Kernels** let SVMs use a new representation without explicitly building all its features.

RBF kernel: distance into similarity

The **radial basis function (RBF)** kernel gives nearby points higher similarity.

$$K(x, z) = \exp\left(-\gamma \|x - z\|^2\right)$$

Same point	Nearby points	Far-apart points
Similarity **1**	Higher similarity	Similarity approaches **0**

- $\|x - z\|$ is the Euclidean distance.
- $\gamma > 0$ controls **how quickly similarity falls with distance**.

Intuition for gamma

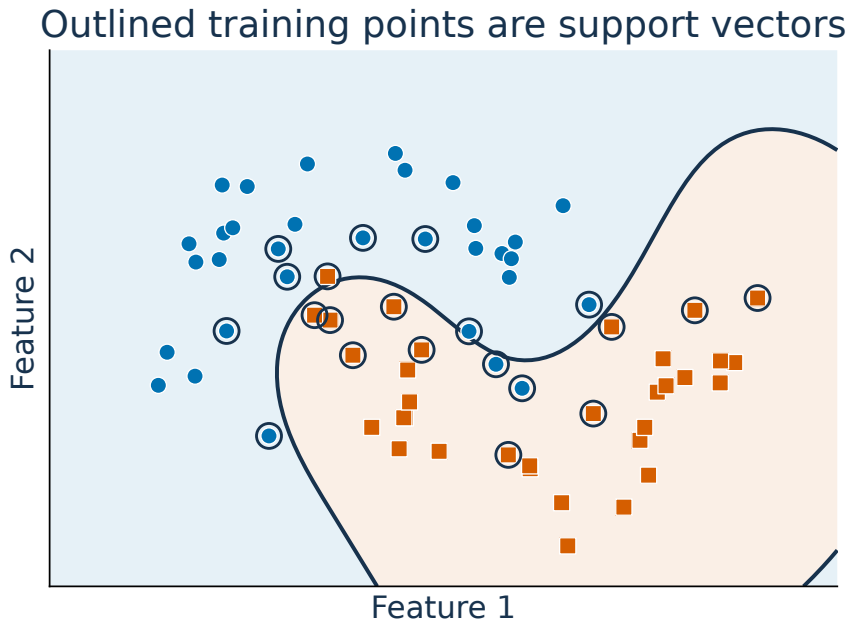
$$K(x, z) = \exp\left(-\gamma \|x - z\|^2\right)$$

Distance r	$\gamma = 0.1$	$\gamma = 1$	$\gamma = 10$
0	1	1	1
0.5	0.975	0.779	0.0821
1	0.905	0.368	0.0000454
2	0.670	0.0183	4.25×10^{-18}

$\gamma > 0$ controls **how quickly similarity falls with distance**.

With $\gamma = 0.1$, similarity is about **0.9**; with $\gamma = 10$, it is almost **zero**.

Training identifies the support vectors



- Training learns **which points contribute** to the decision function and **their weights**.
- These are the **support vectors**; they can be a large fraction of the training set.

How does an RBF SVM predict?

For a new query point:

1. Compute its **RBF similarity to each support vector**.
2. Multiply each similarity by its **learned weight and class sign**.
3. Add these contributions and a **bias**; use the score's sign to choose the class.

SVM learns how to combine similarities during training.

(Optional) From similarities to a score

For binary classification, encode **Blue as** -1 and **Orange as** $+1$:

$$f(x_{\text{new}}) = \sum_{i \in \text{SV}} \underbrace{\alpha_i}_{\text{learned weight}} \underbrace{y_i}_{\text{class sign}} \underbrace{K(x_i, x_{\text{new}})}_{\text{similarity}} + b$$

- x_i: a support vector; $\alpha_i > 0$: its learned weight; b: learned bias.
- **Positive score:** Orange. **Negative score:** Blue. Zero is the decision boundary.

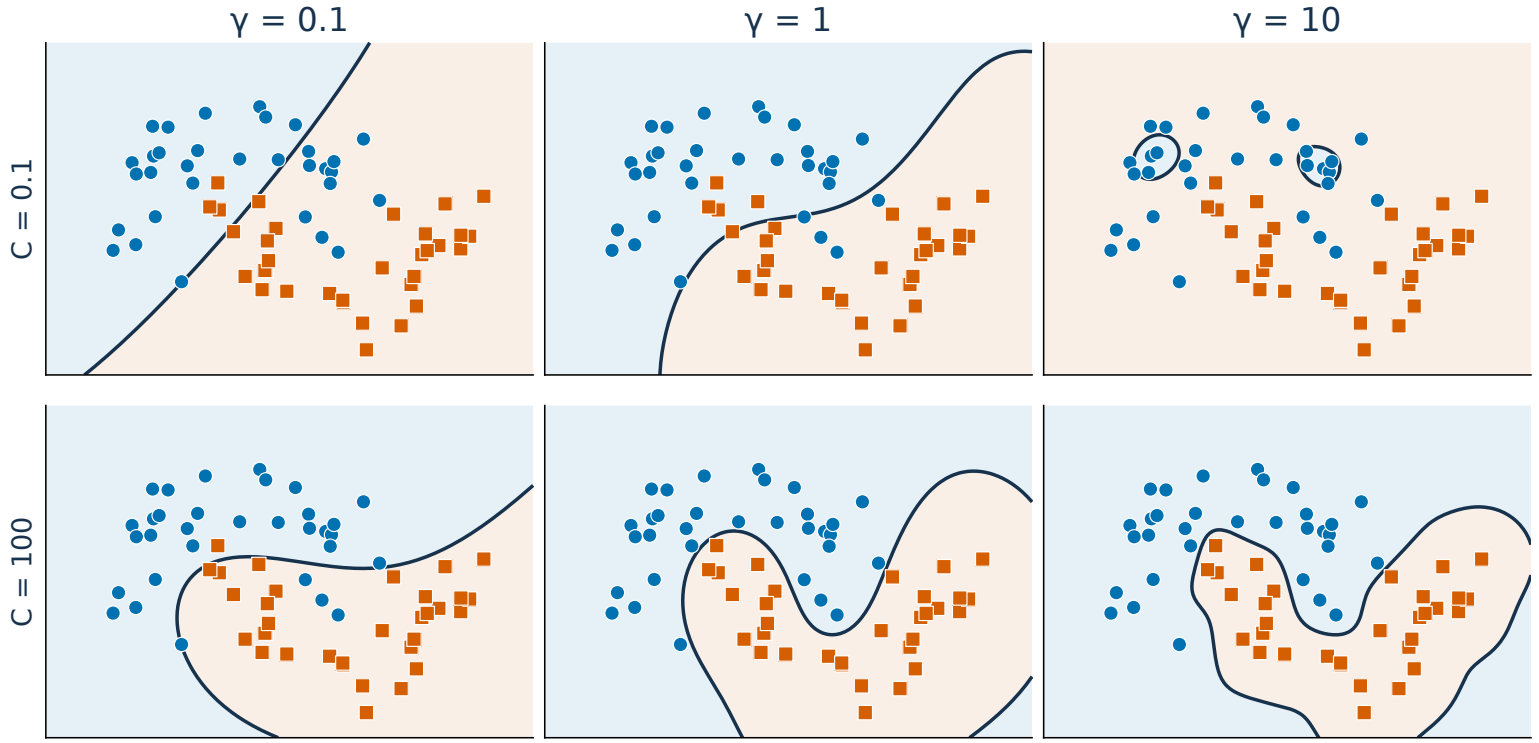
For example: $\underbrace{1.2(0.8)}_{\text{Orange}} - \underbrace{0.7(0.5)}_{\text{Blue}} - \underbrace{0.1}_{\text{bias}} = 0.51 \quad \Rightarrow \quad$ **Orange**

Two controls: `gamma` and `C`

Hyperparameter	What does it control?	Increasing it…
γ	How local similarity is	Makes each point's influence more local
C	How strongly training errors are penalized	Pushes harder to fit the training data

- Larger γ or C can produce a more intricate boundary and increase overfitting risk.
- Smaller values can underfit. **Neither direction guarantees better validation performance.**

See how `gamma` and `C` interact



Compare **across a row**, then **down a column**. Choose both using **cross-validation**.

KNNs vs RBF SVM

k-NN (uniform weights)	SVM with RBF
Find the query's nearest k training points	Compare the query with the learned support vectors
Each neighbour gets one vote	Each similarity gets a learned, signed weight
Neighbours change with the query	Support vectors stay fixed; similarities change

More comments on RBF SVM

- A useful candidate when classes need a **nonlinear decision boundary**.
- Requires choosing both C **and** γ using validation data.
- Distances depend on feature scales: **preprocessing matters** (coming up).
- Training can be expensive with many examples.

iClicker 4.2

Select all of the following statements which are TRUE.

- a. In `sklearn`'s RBF `SVC`, increasing `gamma` typically increases the training score, but it does not necessarily increase the validation score.
- b. If we increase `gamma` but decrease `C`, we can't be certain whether the model becomes more or less complex.

Class demo

Models

Supervised models we have seen

- Decision trees: Split data into subsets based on feature values to create decision rules
- k-NNs: Classify based on the majority vote from k nearest neighbors
- SVM RBFs: Create a boundary using an RBF kernel to separate classes

Comparison of models (activity)

Model	Parameters and hyperparameters	Strengths	Weaknesses
Decision Trees			
KNNs			
SVM RBF			

Our workflow so far

Split → Explore → **Compare** → Refit → Evaluate

- **Compare model types and hyperparameters** using cross-validation on training data.
- Keep the test set untouched until the final evaluation.

Preprocessing motivation: example

You're trying to find a suitable date based on:

- Age (closer to yours is better).
- Number of Facebook Friends (closer to your social circle is ideal).

Preprocessing motivation: example

- You are 30 years old and have 250 Facebook friends.

Person	Age	#FB Friends	Euclidean Distance Calculation	Distance
A	25	400	$\sqrt{5^2 + 150^2}$	150.08
B	27	300	$\sqrt{3^2 + 50^2}$	50.09
C	30	500	$\sqrt{0^2 + 250^2}$	250.00
D	60	250	$\sqrt{30^2 + 0^2}$	30.00

Based on the distances, the two nearest neighbors (2-NN) are:

- **Person D** (Distance: 30.00)
- **Person B** (Distance: 50.09)

What's the problem here?