Machine learning summary

- We train a bunch of models with our training data
- We use cross-validation data to pick the best of these models
- We test it with the testing data to make sure our model is good.

Logistic regression model example

Let's say we have four candidates. We train a model of degree one, a model of degrees two, three, and four.

- We train them with the training data to find the slope and the coefficients of the polynomials et cetera.
- We use the cross-validation data to calculate, say, the F1 score of all these models.
- We pick the model with the highest F1 score.
- As a final step, we use our testing data to make sure our model is good.

The parameters of the algorithm in this case are the coefficients of the polynomial but the degree of the polynomial is called a *hyperparameter*.

Decision tree model example

What are the hyperparameters? Well, one of them is the depth with values of one, two, three, and four.

- We use the training data to train a bunch of trees of depth one, two, three, and four.
- The parameters here are the thresholds in the leaves and the nodes et cetera.
- Then we take the F1 score and calculate it on the cross-validation set on each of these models.
- Then we pick the one that did the best
- Finally, with the testing set, we make sure the model is good.

What happens if we have more than one hyperparameter?

Support vector machine model example

In an SVM, we have some hyperparameters like the kernel which can be linear or polynomial, and we also have the gamma parameter.
How do we pick the best combination of kernel and gamma?

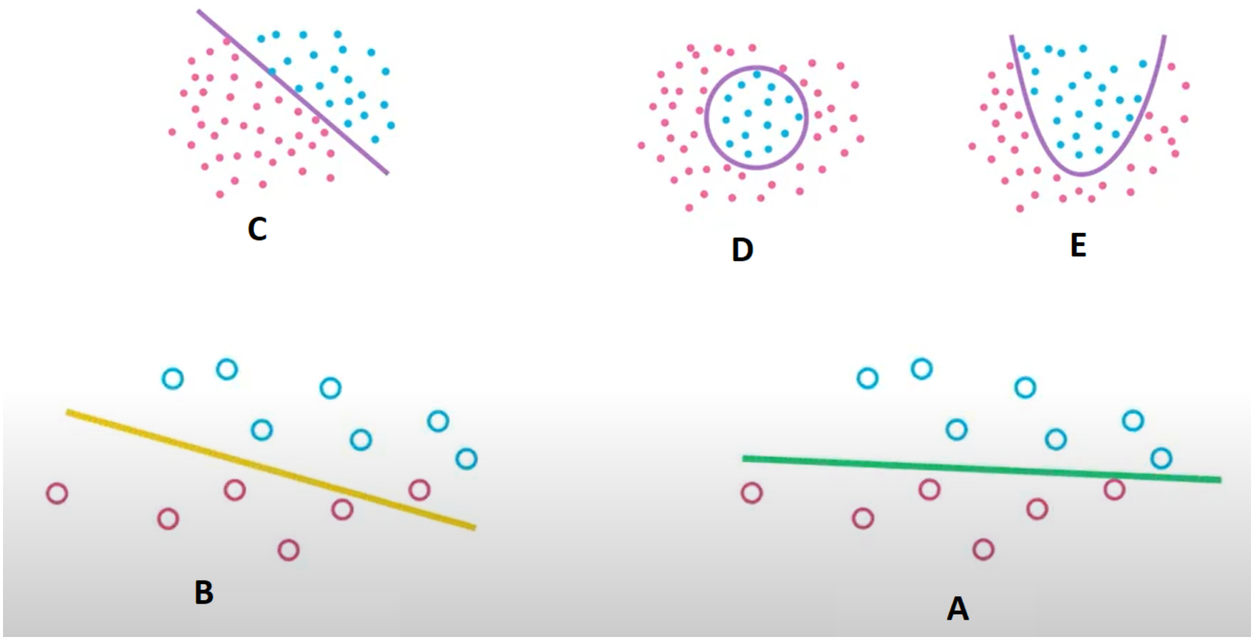
Grid search

Make a table with all the possibilities and pick the best one. Our columns here are the different kernels we can use; linear and polynomial. Our rows are the different values of gamma. It's recommended to take a few values that grow exponentially such as 0.1, 1, 10,100,1000 et cetera.

- We use a training set to train a bunch of linear models and polynomial models with different values of gamma
- We use the cross-validation set to calculate the F1 score in all of these models and then we simply pick the one with the highest F1 score.
- We used the testing set to make sure that what we did was good.

Quiz

Use the image below for the following quiz



A: (C) Green line (almost) horizontally dividing the red and blue 100% correctly.
B: (C) Yellow line angled down the left to right dividing the red and blue, except for 1 red point
C: (Kernal) Purple line angled sharply down left to right dividing the red and blue 100% correctly
D: (Kernal) Purple circle dividing the red and blue 100% correctly
E: (Kernal) Purple parabola dividing the red and blue 100% correctly

Quiz Question

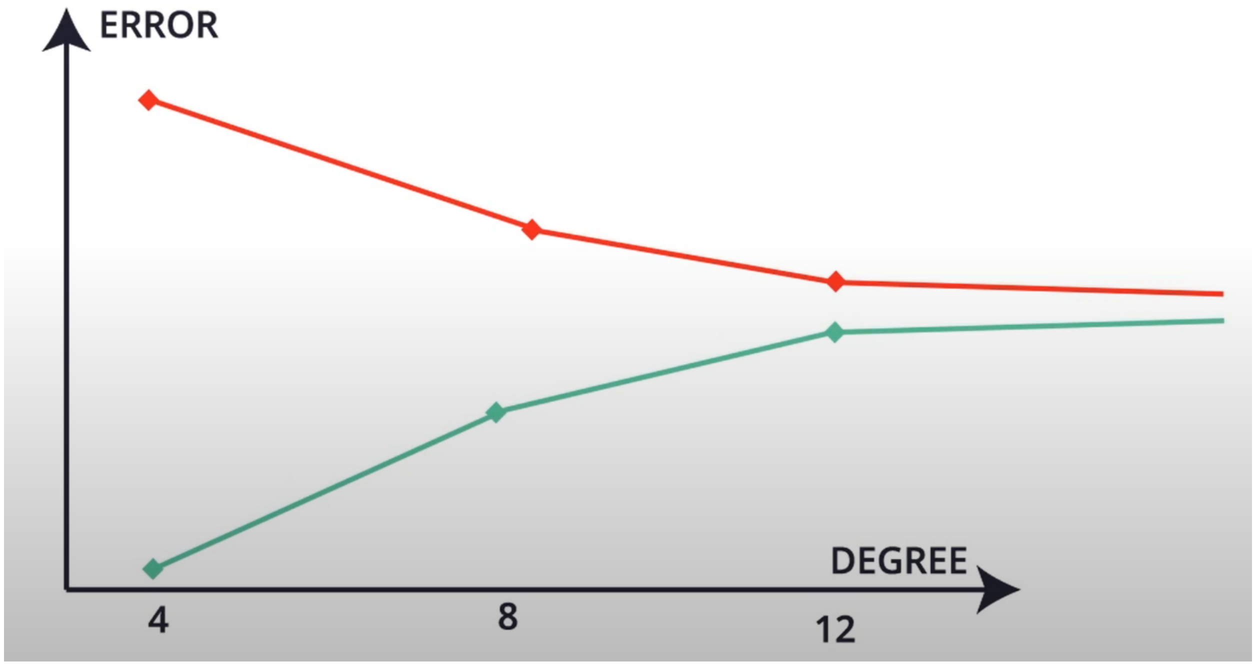
Connect hyperparameters with the correct graph A, B, C, D, E

Kernel-Linear	__C____
Kernel-Polynomial (blue surrounded entirely by red)	__D____
Kernel-Polynomial	___E___
Large C	__B____
Small C	__A____

Overfitting, underfitting, and just right

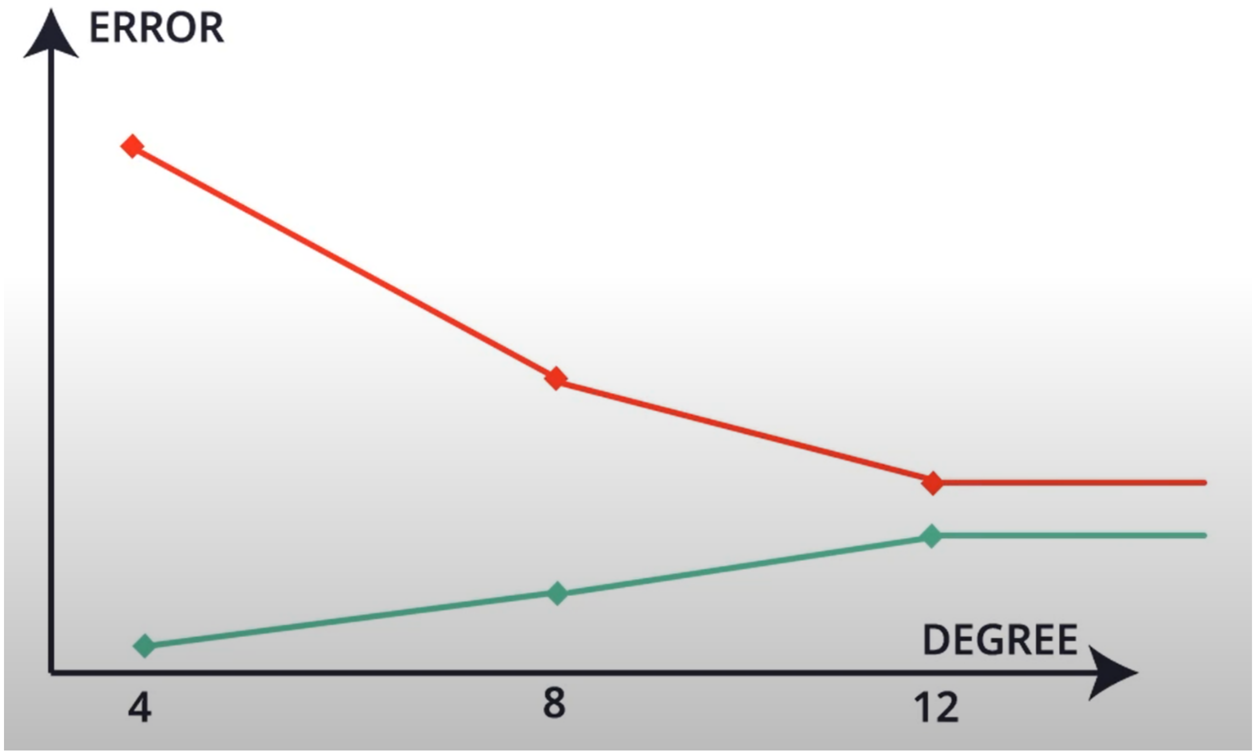
By looking at the shape of the learning curves, you can tell between underfitting, overfitting, and just right. We're going to try and fit three models.

- **Linear**: The first one is a linear or degree one model which doesn't do a very good job since it underfits the data. It's a high bias model. The two lines converge at a high point.



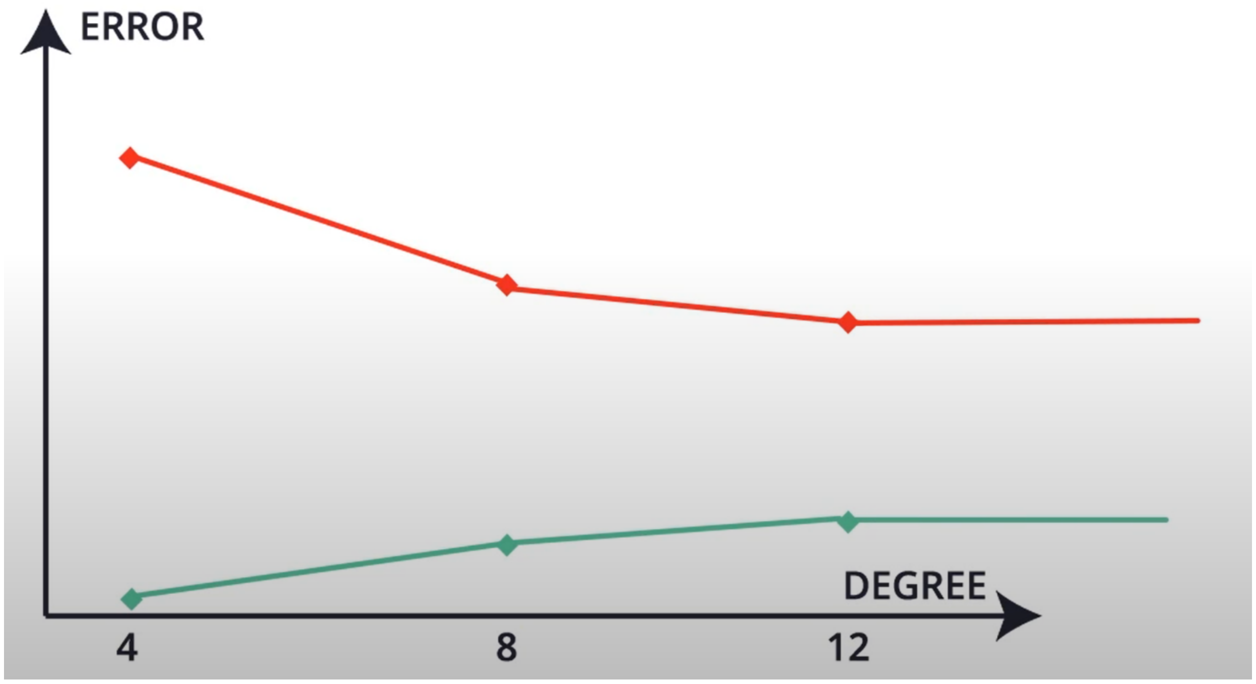
Linear model. Green is training error. Red is CV error.

- **Quadratic**: The second one below, a quadratic model of degree two, is just right. The two lines converge at a lower point than with the linear model



Quadratic model. Green is a training error. Red is a CV error.

- **Polynomial** the third one is a higher degree polynomial of degree six which overfits. It's a high variance model. The two curves get closer but do not converge at the same point.



Polynomial model. Green is training error. Red is CV error.

Quiz

Please use the image below for the following quiz

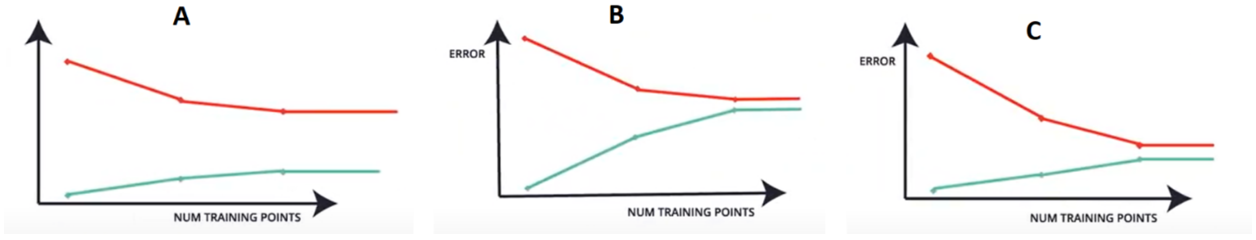


Image descriptions

A: Degree 12. As the number of training points increases, the cross-validation error stays large, and the training error increases very slowly. They would not converge at any point, even if you continue the graph

B: Degree 4. As the number of training points increases, the cross-validation error decreases from large to medium slowly, and the training error increases rather slowly. They would converge at some point if you continue the graph. This point is higher on the graph

C: Degree 8. As the number of training points increases, the cross-validation error decreases from large to medium to small, and the training error increases slowly. They would converge at some point if you continue the graph. This point is lower on the graph

Quiz Question

Connect the correct graph with the type of error

High variance	____overfit_____
High bias	______underfit___
No error	____well fit_____

Recycling our data

Testing is done by separating our data into a training and a testing set, but this is not always ideal as we throw away some data that could be useful for training our algorithm. Is there anything we can do to **not throw away** this data, but at the same time not cheat?

K buckets

With K-Fold Cross Validation,

- Break our data into K buckets.
- Then, train our model K times. Each time using a different bucket as our testing set, and the remaining points as our training set
- Then, average the results to get a final model

To do this in SKLearn we have to create a K-fold object where the parameters are the size of the data and the size of the testing set. For example, kf = KFold(12, 3)

It is always recommended to randomize our data to remove any hint of bias.

Split data into buckets chosen randomly instead of in order. Randomizing is also easily done in SKLearn by setting the shuffled parameter to True when we initialize our K-fold object. For example, kf = KFold(12, 3, shuffle = True)

Quiz Question

True or false: K-buckets split up the data into smaller chunks, then train each bucket, averaging all those training results together.

[✓ a. True]

b. False

Model complexity graph

Let's look carefully at the model complexity graph:

- The model on the left gives us high training and testing errors. This shows underfitting or high bias error.
- The model on the right gives us a very low training error, but a high testing error. This shows overfitting or high variance error.
- The model in the middle gives us relatively low training and testing errors. This is the model we should pick.

Thus, we decide that the best model for our data is a polynomial of degree 2.

Breaking the golden rule: never use your testing data for training

How do we make a good decision about our model without using the testing data?

We can break down our dataset even more. Instead of having a training and a testing set, we'll add a cross-validation set. The training set will be used for training the parameters. The cross-validation set will be used for making decisions about the model such as the degree of the polynomial, and the testing set will be used for the final testing of the model.

Summary

Here we have the model complexity graph and the example altogether.

- On the left, we see *underfitting* or an oversimplification of the problem. This is bad on both the training and the cross-validation set because our model simply does not capture the complexity of our data.

- On the right, we see *overfitting* or an over-complication of the problem. This is great on the training set because we're memorizing it but bad on the cross-validation set because the model does not generalize well.

- In the middle, we have a model which is good on both the training and the testing set.

In general, we want to pick the model that appears closest to where our cross-validation and training data are close together before they start to distance themselves from one another and the model starts to overfit the training data.

Quiz Question

True or False. Very low training error, but a high testing error is an indication of overfitting.

a. True

b. False

Quiz Question

True or False. High training error and high testing errors is an indication of overfitting.

a. True

b. False.

Detecting errors

Let's recall the types of errors learning the last section. Here we can see our data on the model which fits it properly, which is a curve or a polynomial of degree 2.

- It seems to fit our data well and it will generalize well. Then there's the high bias or *underfitting* error obtained when we oversimplify our model. In this case, it's trying to fit a line or a polynomial of degree 1 through our data, this model won't fit our data well.

- Then there's a high variance or *overfitting* error obtained when we over-complicate our model. In this case, it's trying to fit a polynomial of degree 6 through our data. This model will fit our data perfectly, but it won't generalize well to the testing set.

Split the data

Let's first split our data into training and testing sets. So the set of points for the colored center are the training set and the set of points for the white center is the testing set.
Three possible models.

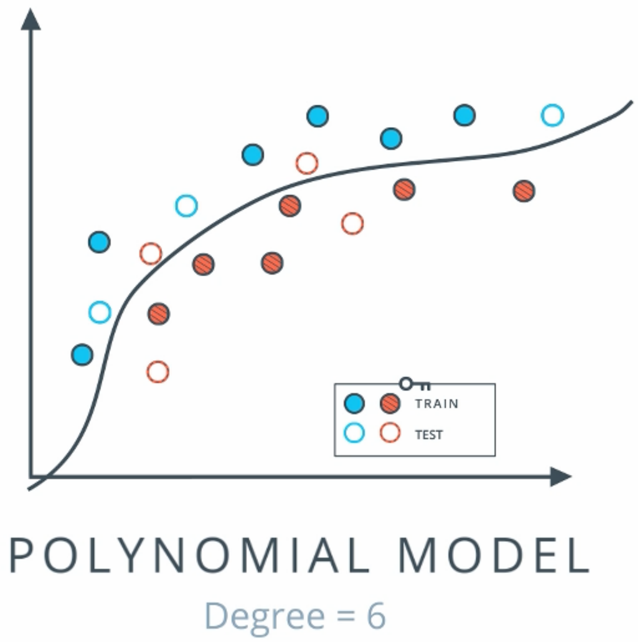
- Linear model of degree 1
    - The linear model gives us this line over here. Now how are we doing in terms of the training set and the testing set? Well, we count the number of training errors and we count three of them.
        - We conclude that the training error is 3
        - We conclude the testing errors is also 3

- Quadratic model of degree 2
    - Repeat this process but in the quadratic model, this model gives us a parabola over here. What is our training error? It's 1 because we've misclassified one of the training points. The testing error is also 1, since we misclassified one testing point. Thus, this model gave us a training error of 1 and a testing error of 1.

- Polynomial model of degree 6
    - Repeat this process with a polynomial of degree 6. The model gives us this curve.

Quiz

What are the training and testing errors for this model?



Model Complexity Graph Quiz

In the model above, how many training and testing errors are there?

For example, if you find 1 training error and 4 testing errors, your answer should be 1, 4.

____0 ,2___________________

Underfitting and overfitting

- Underfitting: When we oversimplify the problem
- Overfitting: when we overcomplicate the problem

In machine learning, these two types of errors are very easy to make.

Oversimplification

If our model is too simple it is called *underfitting*.

- One characteristic is that it doesn't do well on the training set
- We call this type of error an "error due to *bias*."

Overcomplication

If our model is too specific to the training set, this error is called *overfitting*.

- One characteristic of it is that it does well on the training set, but will not do well on the testing set.
- We call this type of error an "error due to *variance*."

Summary

- High bias (underfitting): Oversimplify the problem; the model is too simple to capture the complexity of our data.
- High variance (overfitting): Overcomplicate the problem; the model is too complex. It ends up memorizing the data instead of learning it.

A good model tends to fit the training data well. When it comes to the testing data, the high bias model and the high variance model both tend to perform poorly.

A good model is the one that performs well in the training data as well as the testing data.

Quiz Question

Check all that are true about types of errors and what they mean: (There is more than one correct answer)

a. Underfitting: oversimplify the problem
   Overfitting: overcomplicate the problem
b. Overfitting:, oversimplify the problem
   Underfitting: overcomplicate the problem
c. High bias: underfitting
   High variance: overfitting
d. High variance: underfitting
   High bias: overfitting

Quiz Question

Which one of the following most accurately describes the consequences of the errors

a. High variance, underfitting
   High bias , overfitting

b. High variance, overfitting
   High bias, underfitting

c. Low bias, underfitting
   Low variance, overfitting