ieeta instituto de engenharia electrónica e telemática de aveiro

universidade de aveiro

Departamento de Eletrónica, Telecomunicações e Informática

Machine Learning

Lecture 6: Model selection and validation – Bias vs. Variance

Petia Georgieva
(petia@ua.pt)

universidade de aveiro

Outline

- **Model selection: Bias vs. variance**
- **Learning curves**
- **K –fold Cross Validation**
- **Ensemble classifiers**
- **k-Nearest Neighbor (k-NN) classifier**

Deciding what to do next ?

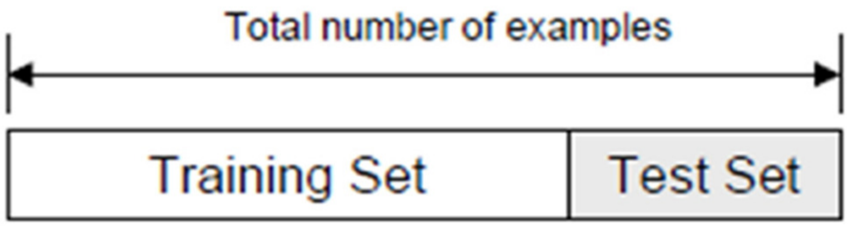
Suppose you have trained a ML model on some data. However, when you test your hypothesis on a new set of data, you find that it makes unacceptably large errors in its prediction . What should you do ?

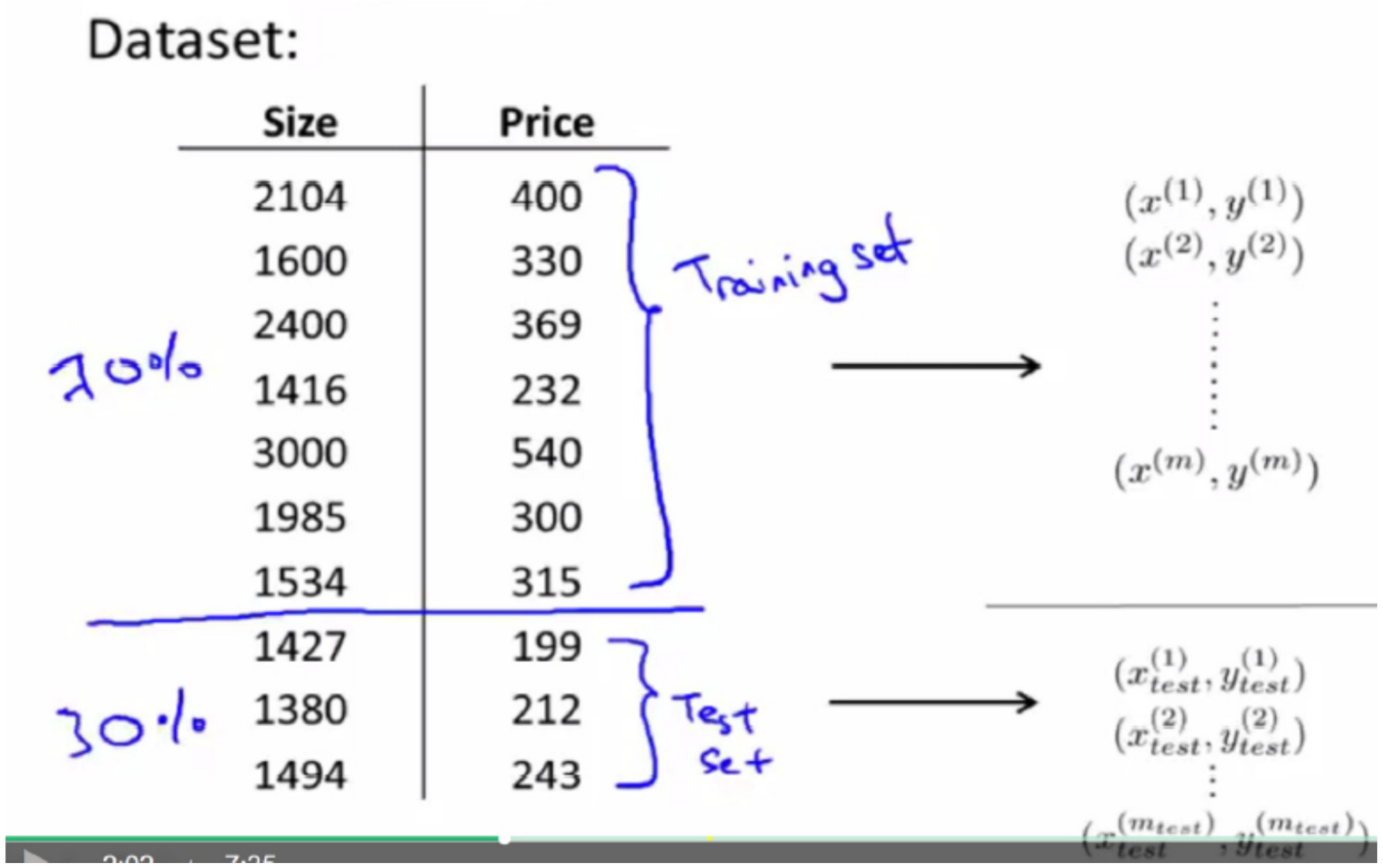
-- Get more training examples ?
-- Try smaller sets of features ?
-- Try getting additional features ?
-- Try adding polynomial features ?
-- Try decreasing/ increasing the regularization parameter lambda ?

Machine learning diagnostics:

You need to run tests to gain insight what isn't working with the learning algorithm and how to improve its performance.
Diagnostics can take time to implement, but can be a very good use of your time.

Train & Test subsets *(holdout method)*





Dataset:

Size	Price
2104	400
1600	330
2400	369
1416	232
3000	540
1985	300
1534	315
1427	199
1380	212
1494	243

$(x^{(1)}, y^{(1)})$
$(x^{(2)}, y^{(2)})$
$\vdots$
$(x^{(m)}, y^{(m)})$

$(x^{(1)}_{test}, y^{(1)}_{test})$
$(x^{(2)}_{test}, y^{(2)}_{test})$
$\vdots$
$(x^{(m_{test})}_{test}, y^{(m_{test})}_{test})$

Train & Test subsets (*one model*)

If you have chosen the model, split data into two sets:

- Training set : used to train the model
- Test set : used to test the trained model

- **Optimize the model parameters with training data**
  (minimize some cost (loss) function)

After the training stage is over (i.e. the cost function J converged)

- Compute the MSE on test data (for regression problems)

$$E_{test}(\theta) = \frac{1}{m_{test}} \left[\sum_{i=1}^{m_{test}} \left(h_\theta \left(x_{test}^{(i)} \right) - y_{test}^{(i)} \right)^2 \right]$$

or

- **Compute the model accuracy on test data (for classification problems)**

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Mean Squared Error (MSE) is not the same as the cost function!!!

Different Cost (Loss) Functions

Training data MSE

- **Linear Regression Cost Function with L2 Regularization**

$$J(\theta) = \frac{1}{2m}\left[\underbrace{\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2}_{\text{Training data MSE}} + \underbrace{\lambda\sum_{j=1}^{n}\theta_j^2}_{\text{Ridge Regression}}\right]$$

Ridge Regression

- **Logistic Regression Cost Function with L2 Regularization**

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left[-y^{(i)}\log(h_\theta(x^{(i)})) - (1-y^{(i)})\log(1-h_\theta(x^{(i)}))\right] + \boxed{\frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2}$$

- **SVM Cost Function with L2 Regularization**

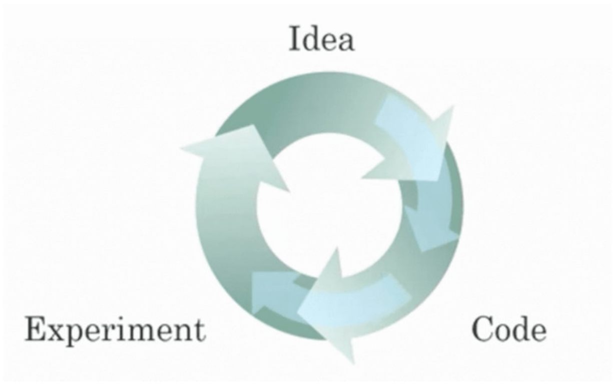
$$\min_\theta C\sum_{i=1}^{m}\left[y^{(i)}cost_1(\theta^T x^{(i)}) + (1-y^{(i)})cost_0(\theta^T x^{(i)})\right] + \boxed{\frac{1}{2}\sum_{i=1}^{n}\theta_j^2}$$

Mean Squared Error (MSE) is not the same as the cost function!!!

universidade de aveiro

3 way split: Train/Dev/Test Sets

Applied ML is a highly iterative process



Choose model architecture:

- # of layers ?
- # of hidden units (neurons) ?
- Activation functions ?
- Learning rates ?
- Polinomial degree
- Differrent ML algorithms: Log_Regr, NN, SVM, etc.
- ……

Devide dataset in 3 sub-sets:

- Training set
- Hold-out Cross Validation (CV) set /Development ('dev') set
- Test set (for unbiased final model evaluation !!!)

Traditional division for Small data set (100-1000-10000 exs.) :
60% - 20% - 20%

Big data (1000000 exs.): 98% - 1% - 1% (10000 exs.)

Mismatched train/dev/test distribution

Training set : Cat pictures from webpages or synthetic data from augmentation techniques

Dev and Test set: Cat pictures from users, using their cameras

Make sure dev and test data come from the same distribution.

Model selection

Step 1: Optimize parameters θ (to minimize the cost function *J*) using the same training set for all models. Compute the training error:

Training error:

$$E_{train}(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2\right]$$

Step 2: Test the optimized models from step 1 with the CV set and choose the model with the min CV error:

Cross validation (CV) error:

$$E_{cv}(\theta) = \frac{1}{2m_{cv}}\left[\sum_{i=1}^{m_{cv}}\left(h_\theta\left(x_{cv}^{(i)}\right) - y_{cv}^{(i)}\right)^2\right]$$

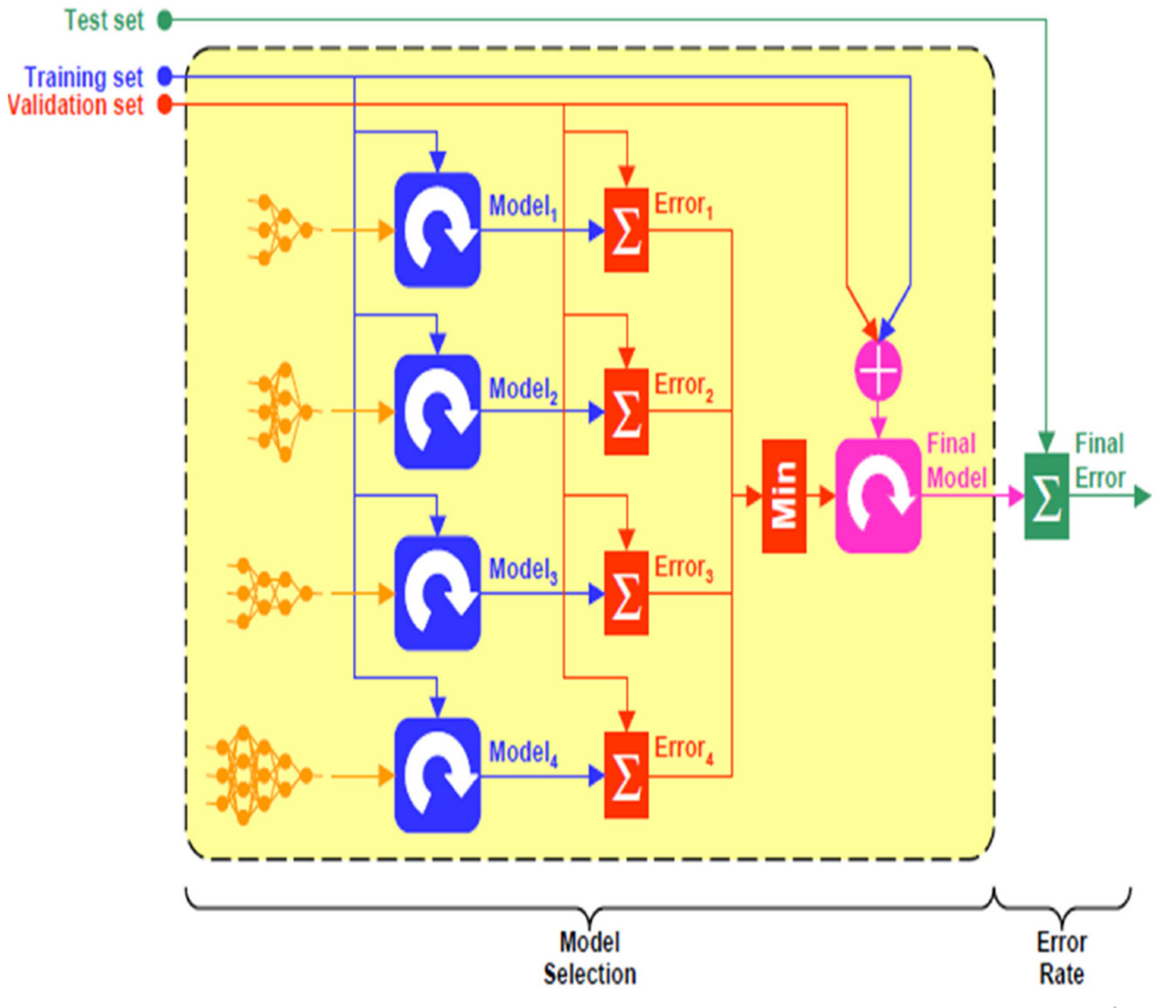
Step 3: Retrain the best model from step 2 with both train and CV sets starting from the parameters got at step 2. Test the retrained model with test set and compute test error (***this is the real model performance !!!***):

Test error:

$$E_{test}(\theta) = \frac{1}{2m_{test}}\left[\sum_{i=1}^{m_{test}}\left(h_\theta\left(x_{test}^{(i)}\right) - y_{test}^{(i)}\right)^2\right]$$

universidade de aveiro

Training/Valid (Dev)/Test subsets



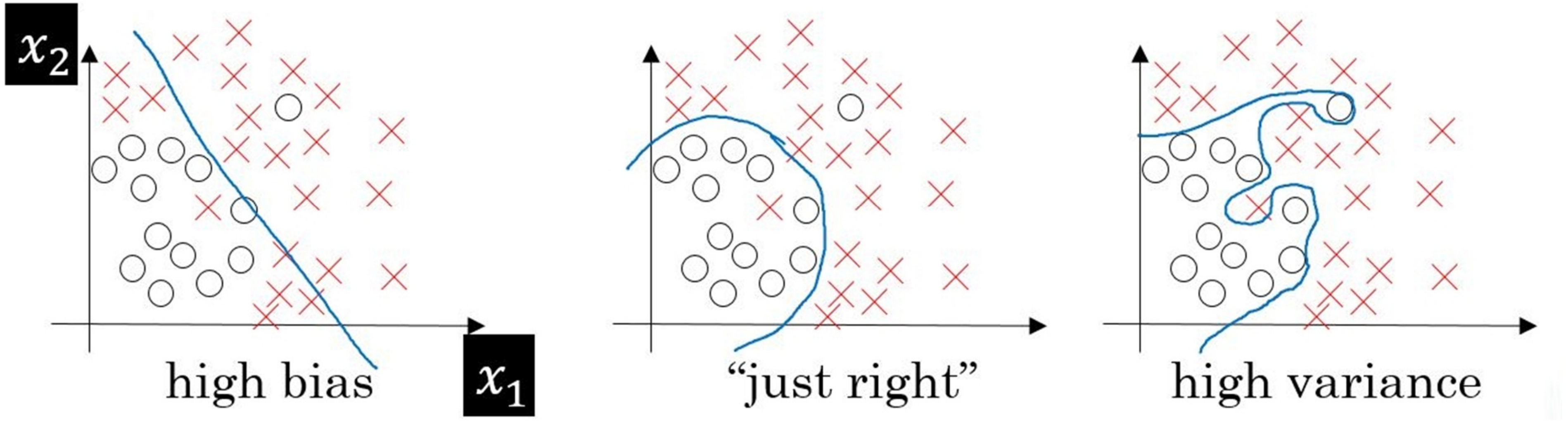
1

The most important and credible is the final error (obtained with the test set, not used for training or validation of the model)

universidade de aveiro

Bias vs. Variance

An important concept in machine learning is the bias-variance tradeoff. Models with high bias are not complex enough for the data and tend to under-fit, while models with high variance over-fit the training data.



underfiting data
(very simple model)

(good model)

overfiting data
(very complex model)

Diagnosing Bias vs. Variance

How to diagnose if we have a high bias problem or high variance problem ?

High Bias (underfiting) problem:

Training error (*Etrain)* and Validation/dev error (*Ecv)* are both high

High Variance (overfiting) problem:

Training error (*Etrain)* is low
and Validation/dev error (*Ecv)* is much higher than *Etrain*

Choose regularization parameter λ

For a given model, try different values of λ =[0, 0.01, 0.1, 1,….]

Step 1: For each λ, optimize parameters θ using the training set

$$E_{train}(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2\right]$$

Step 2: Test the optimized models from step 1 with the CV set and choose the model with λ that gets min CV error:

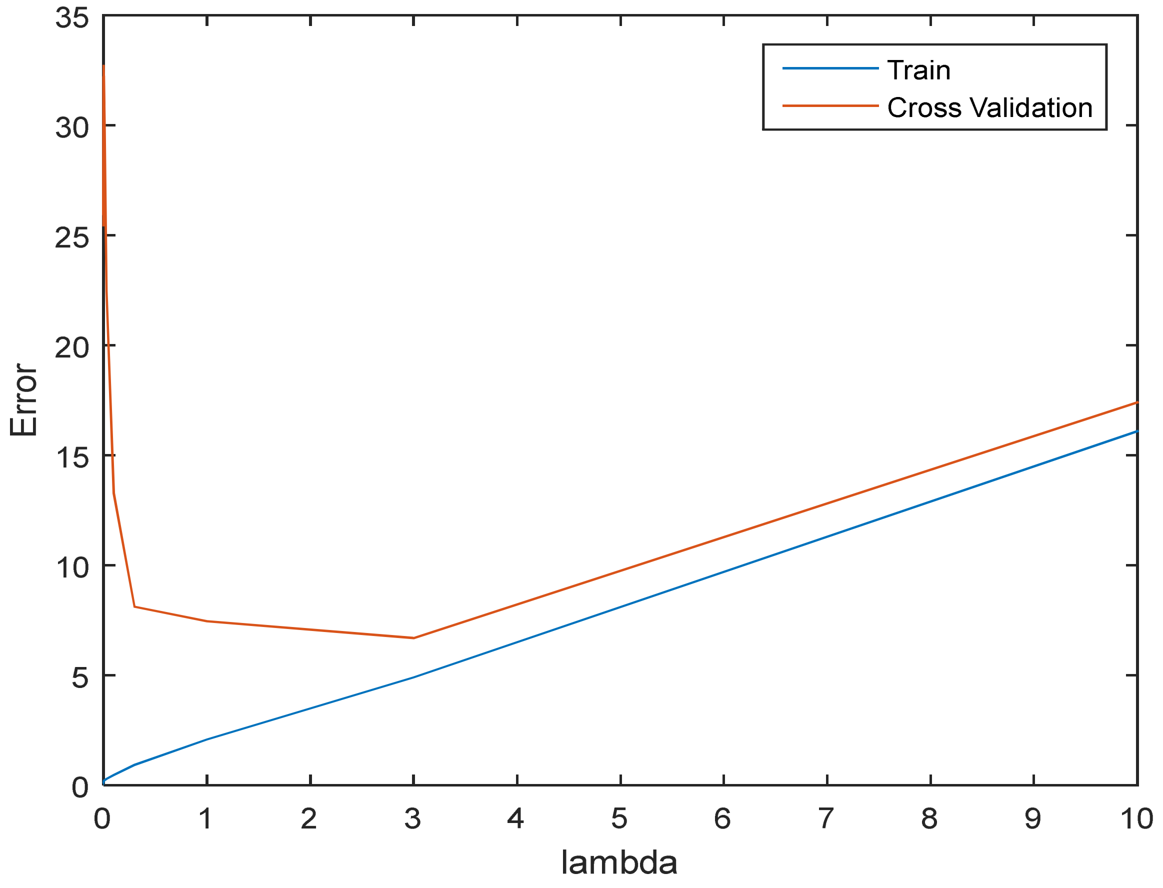
$$E_{cv}(\theta) = \frac{1}{2m_{cv}}\left[\sum_{i=1}^{m_{cv}}\left(h_\theta\left(x_{cv}^{(i)}\right) - y_{cv}^{(i)}\right)^2\right]$$

Step 3: Retrain the model with best λ from step 2 with both train and CV sets starting from the parameters θ got at step 2. Test the retrained model with the test set and compute the error:

$$E_{test}(\theta) = \frac{1}{2m_{test}}\left[\sum_{i=1}^{m_{test}}\left(h_\theta\left(x_{test}^{(i)}\right) - y_{test}^{(i)}\right)^2\right]$$

universidade de aveiro

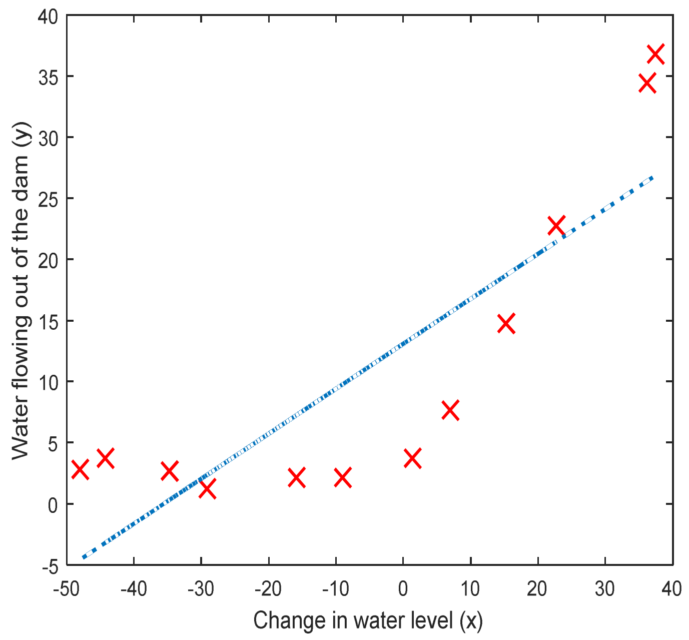
Select λ using CV(dev) set

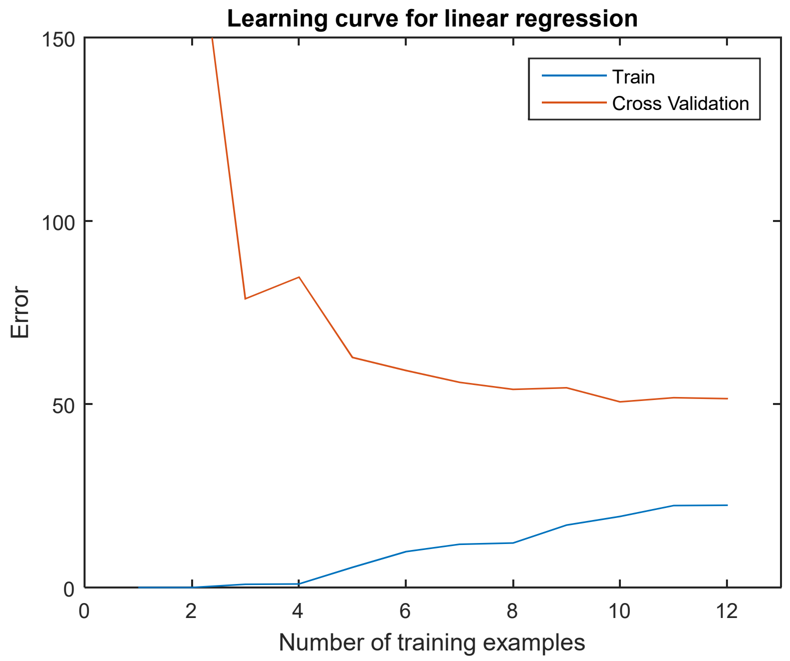


Best λ=3

Learning Curves

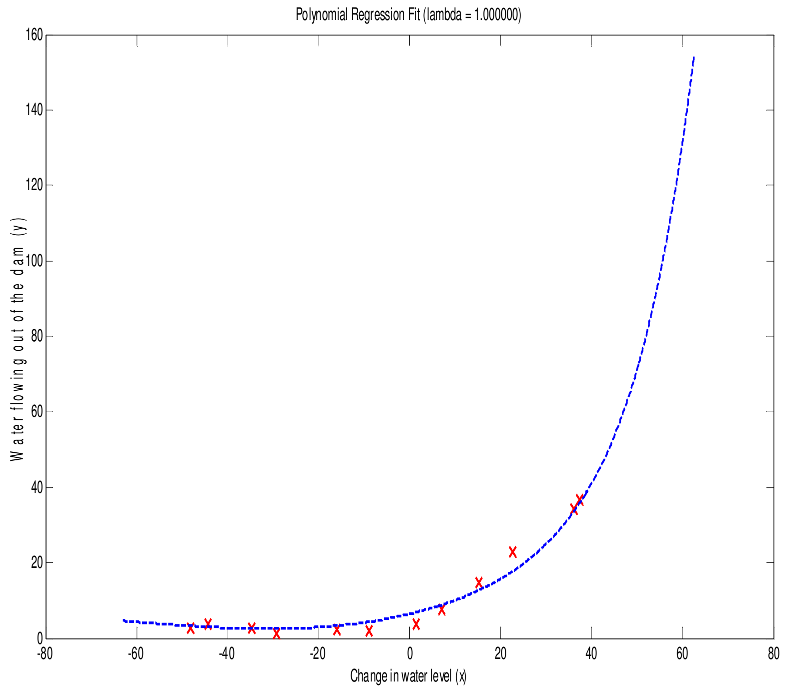
$$h_\theta(x) = \theta_0 + \theta_1 x$$

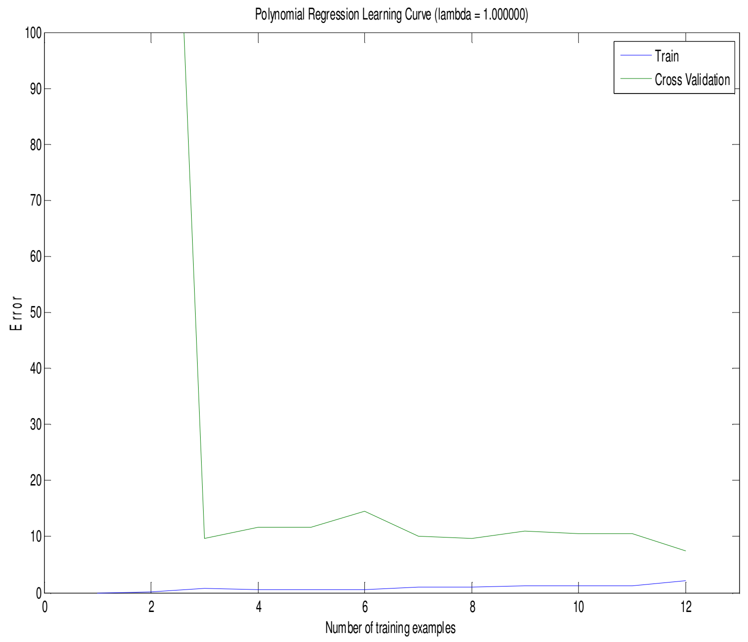




If a learning algorithm is suffering from high bias, getting more training data will not help much

Learning Curves

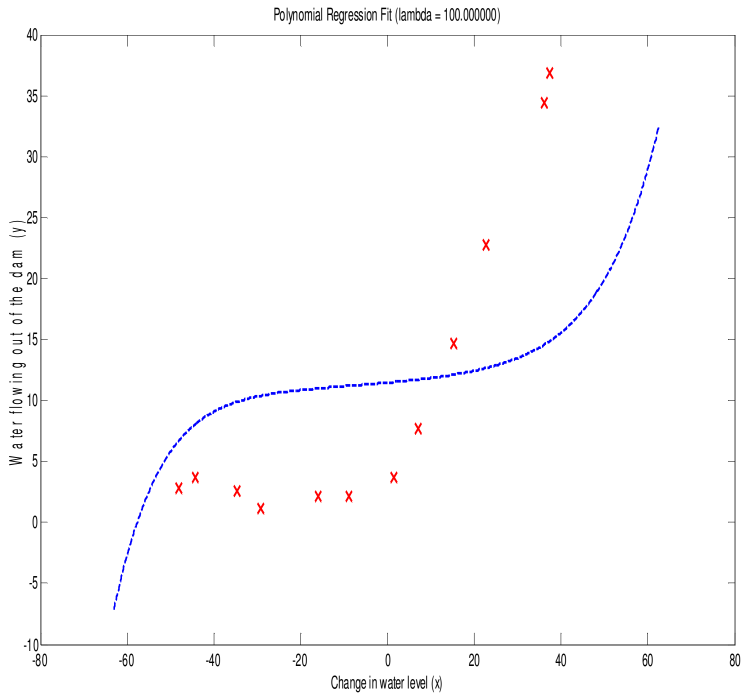


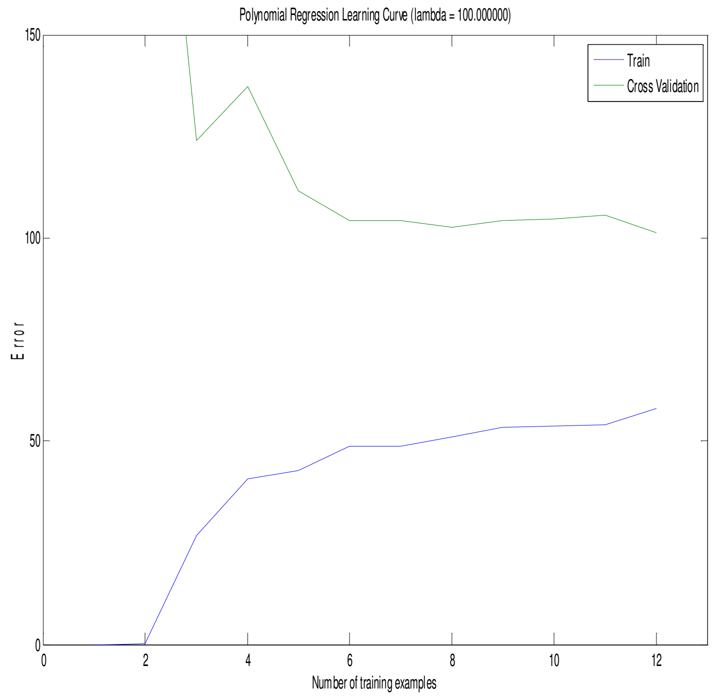


If a learning algorithm is suffering from high variance, getting more training data is likely to help

universidade de aveiro

Regularization and Learning Curves





Polynomial regression, λ=100 **Learning curve, λ=100**

universidade de aveiro

Hints to improve ML model

Suppose you have learned a data model (hypothesis). However, when you test your hypothesis on a new set of data, you find that it makes unacceptably large errors in its prediction (regression or classification). What should you try next?

-- Get more training examples – fixes high variance

-- Try smaller sets of features – fixes high variance

-- Try getting additional features – fixes high bias

-- Try adding polynomial features - fixes high bias

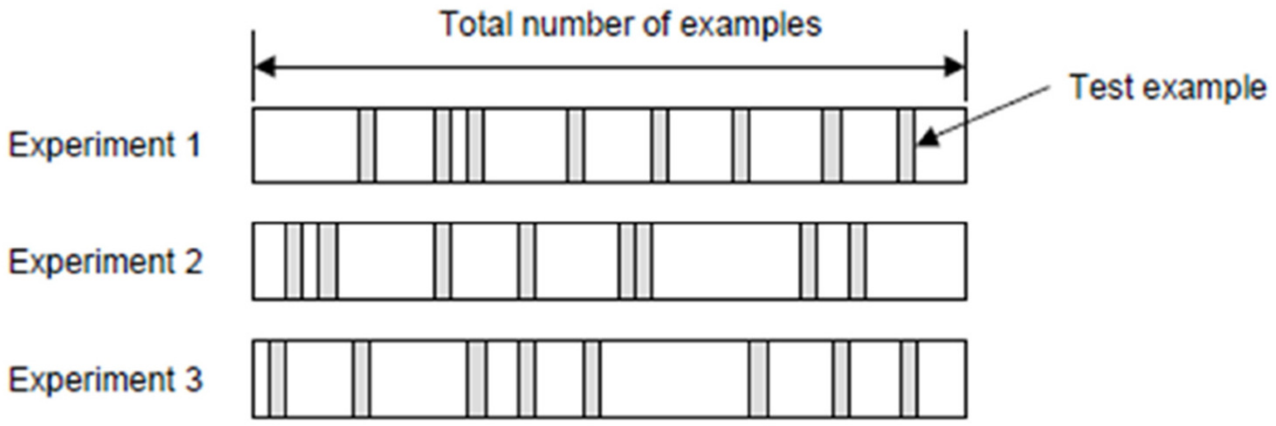
-- Try decreasing λ – fixes high bias

- Try increasing λ – fixes high variance

Statistical approach – Random Subsampling

- Make K training experiments, where each time randomly select some of the examples for training (70%) and the rest for CV without replacement.
- For each data split retrain the model from scratch with the training examples and estimate the error *Ecv* with the CV examples.
- The final validation error is obtained as the average of the CV errors.
- The estimate is better than the holdout method.

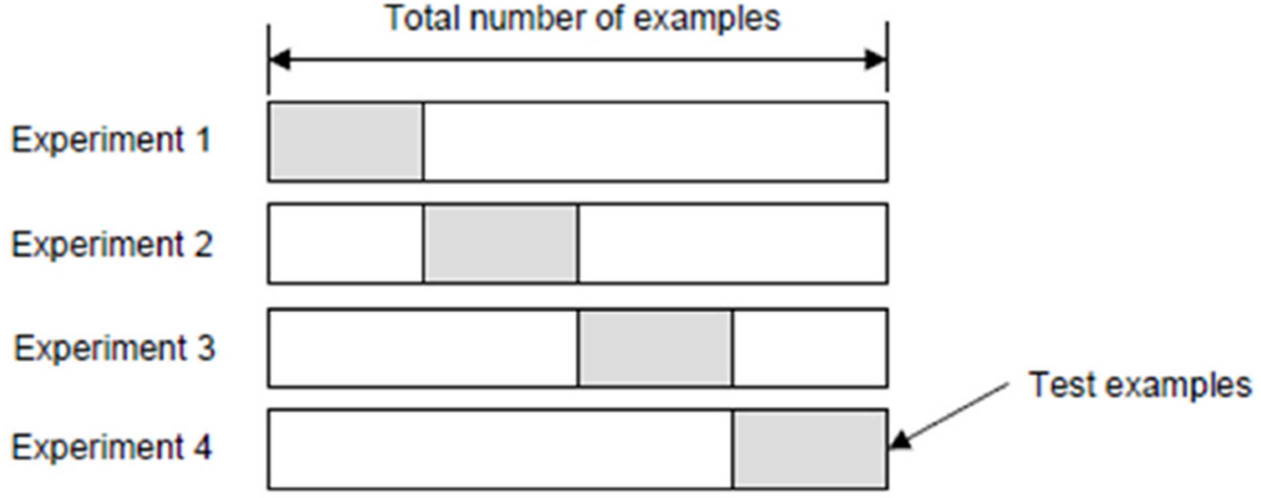
$$E_{cv} = \frac{1}{K}\sum_{i=1}^{K} E_{testi}$$



K –fold Cross Validation

- Divide data into Training and Test subsets. From here on work only with training data.
- Divide (training) data into K subsets (K-fold).
- Use K-1 subsets for training and the remaining subset for CV.
- The advantage of K-fold CV is that all examples in the dataset are used for both training and validation.
- As before the final validation error is estimated as the average CV error.

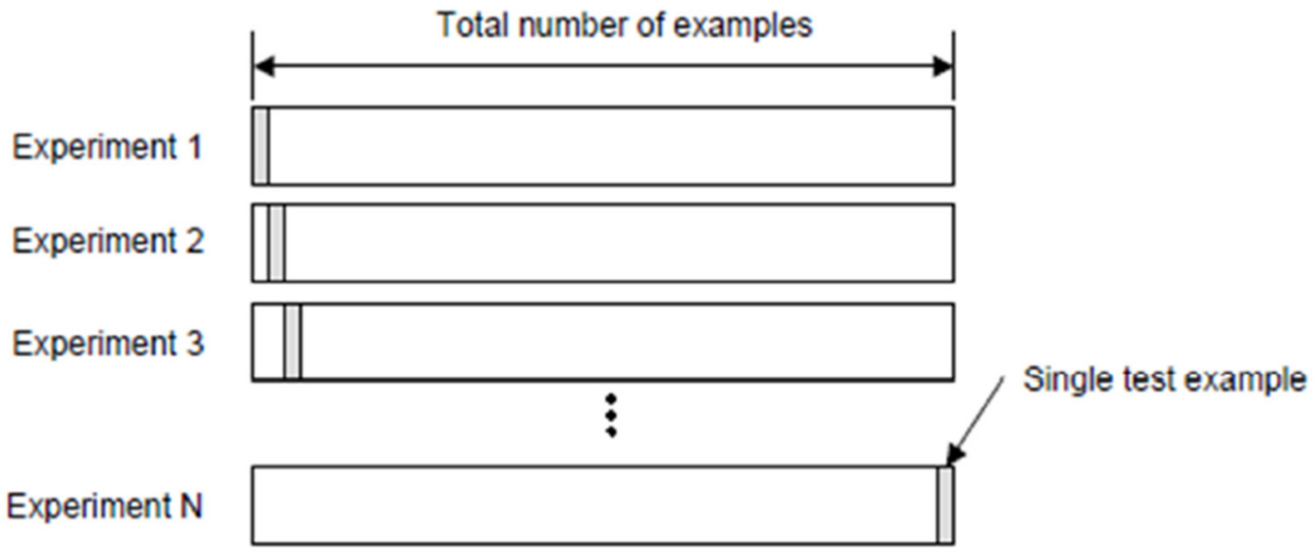
$$E_{cv} = \frac{1}{K}\sum_{i=1}^{K} E_{testi}$$



Leave-one-out Cross Validation

- Leave-one-out is the degenerate case of K-fold CV, where K is chosen as the total number of examples.
- For a dataset with *N* examples, perform m experiments.
- For each experiment use *N-1* examples for training and the remaining example for CV.
- As before the final validation error is estimated as the average error on CV examples.
- Useful for small data sets.

$$E_{cv} = \frac{1}{K}\sum_{i=1}^{K} E_{testi}$$



Ensemble (Committee) Classifiers

● Learn a set of classifiers (ensemble, committee) and combine (somehow) their predictions.

Build "weak" classifiers, do not try too hard to find the best classifier. Choose classifiers with different nature (deterministic, probabilistic, linear, nonlinear)

● **MOTIVATION:**

- reduce the variance (results are less dependent on specificity of training data)

- reduce the bias (multiple classifiers means different models)

universidade de aveiro

Combining Predictions

1. Voting
- each ensemble classifier votes for one of the classes
- predict the class with the majority of votes (e.g., bagging)

2. Weighted voting
- make a *weighted* sum of the votes of the ensemble classifiers
- weights depend on the classifier error (e.g., boosting)

3. Stacking
- Use a higher level (meta) classifier to make the final decision.
- the meta classifier has as inputs the predictions of the ensemble classifiers and the outputs are the class labels

Ensemble classification- Bagging

a) **Bagging:** take some of the examples from the original training data (with replacement) but keep the size equal to the original data set.

b) Train M classifiers (preferably different type of models e.g. Log Regr., SVM, NN etc.) with different Bags.

c) Test all classifiers with the test examples.

d) Assign the class that receives majority of votes.

Variations are possible -e.g., size of subset, sampling w/o replacement, etc.

Original Data	1	2	3	4	5	6	7	8	9	10
Bagging (Round 1)	7	8	10	8	2	5	10	10	5	9
Bagging (Round 2)	1	4	9	1	2	3	2	7	3	2
Bagging (Round 3)	1	8	5	10	5	5	9	6	3	7

Ensemble classification- Boosting

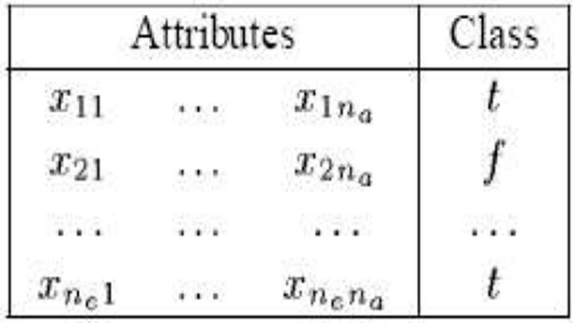
- **Basic Idea:**

- next classifiers focus on examples that were misclassified by earlier classifiers

- increase the weight of incorrectly classified examples, this ensures that they will become more important

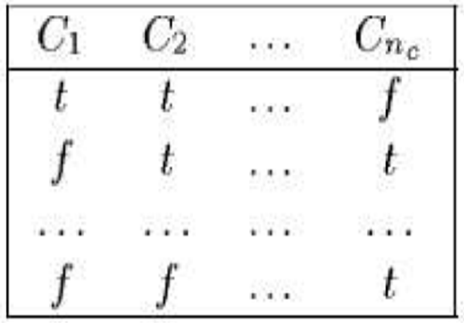
- weight the predictions of the classifiers with their error

STACKING (hierarchical) - Example



Attributes			Class
x_{11}	$\dots$	x_{1n_a}	t
x_{21}	$\dots$	x_{2n_a}	f
$\dots$	$\dots$	$\dots$	$\dots$
x_{n_e1}	$\dots$	$x_{n_en_a}$	t

(a) training set

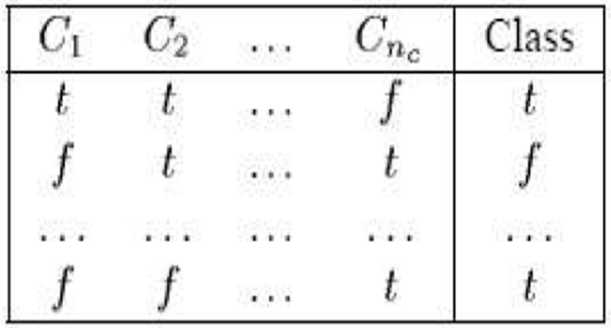


C_1	C_2	$\dots$	C_{n_c}
t	t	$\dots$	f
f	t	$\dots$	t
$\dots$	$\dots$	$\dots$	$\dots$
f	f	$\dots$	t

(b) predictions of the classfiers

- Train the lower level classifiers *C1...Cnc*

- Form a feature vector consisting of their predictions

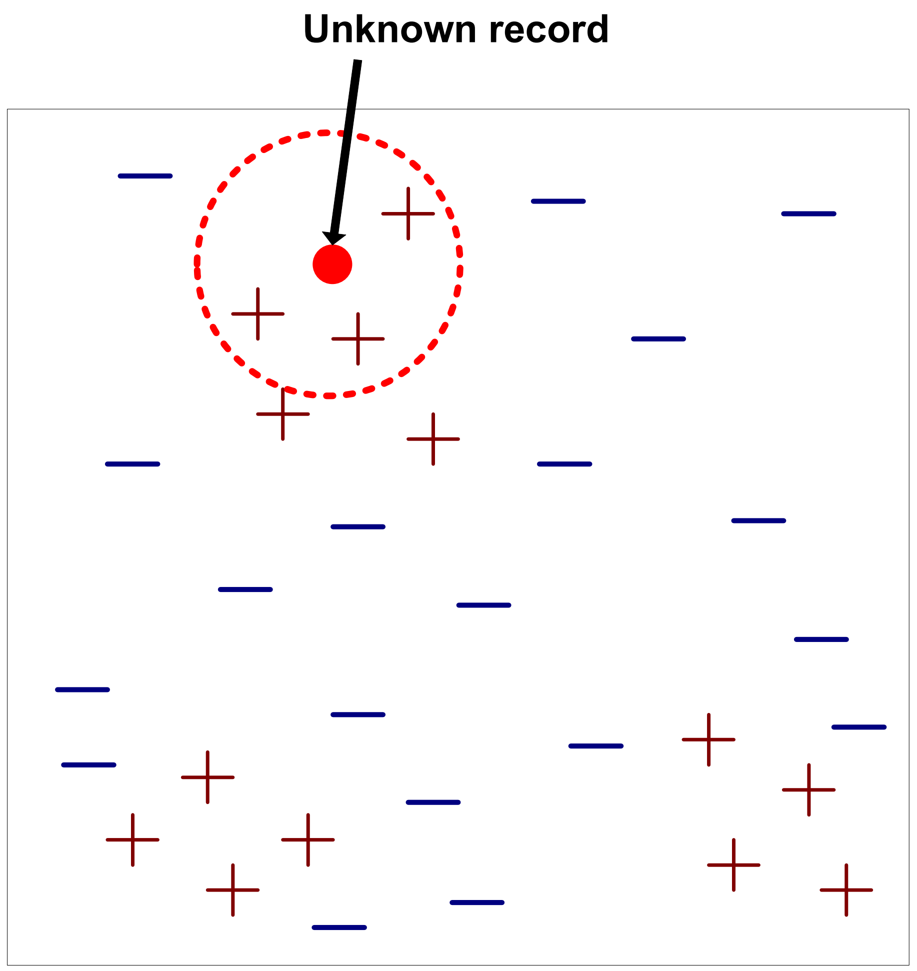


C_1	C_2	$\dots$	C_{n_c}	Class
t	t	$\dots$	f	t
f	t	$\dots$	t	f
$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
f	f	$\dots$	t	t

(d) training set for stacking

- Train the meta classifier with These feature vectors

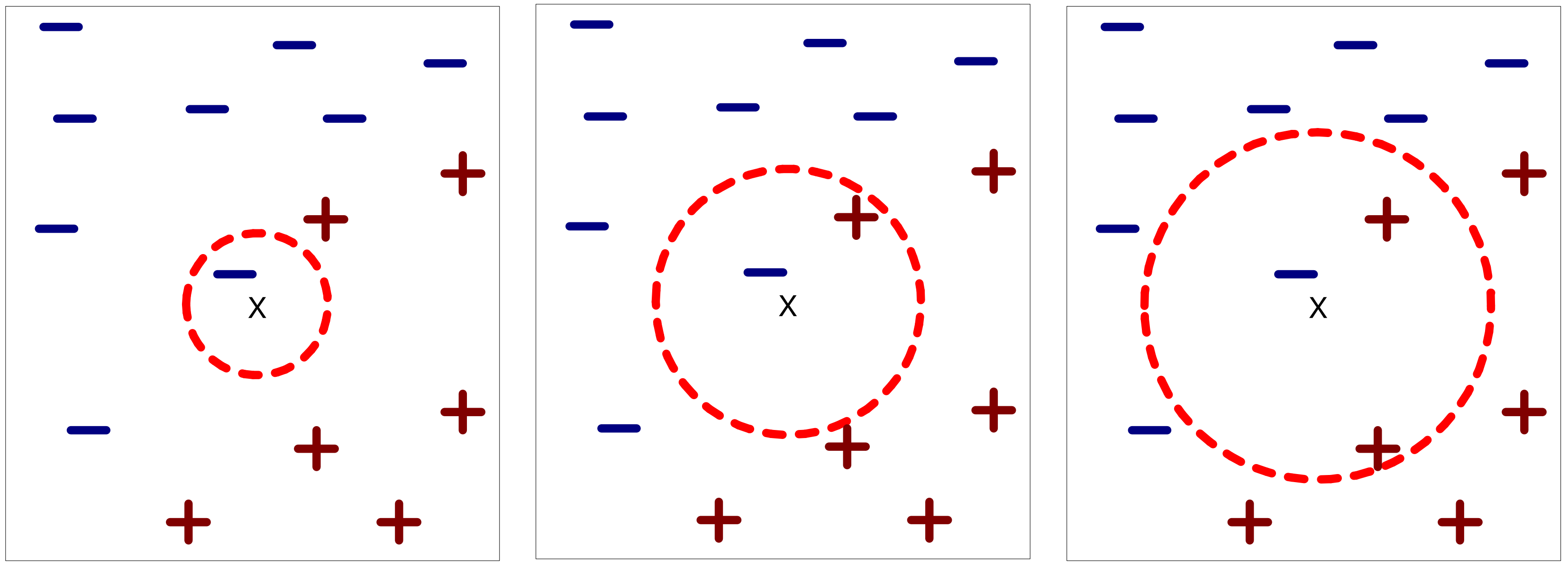
K- Nearest-Neighbor (kNN) Classifier



- KNN requires:
  - Set of labeled records.
  - Measure to compute distance (similarity) between records.
  - *K* is the number of nearest neighbors (the closest points).

- To classify a new (unlabeled) record:
  - Compute its distance to all labeled records.
  - Identify *k* nearest neighbors.
  - The class label of the new record is the label of the majority of the nearest neighbors.

kNN- choice of k



(a) 1-nearest neighbor (b) 2-nearest neighbor (c) 3-nearest neighbor

K-nearest neighbors of the new point x are the points that have the smallest distance to x

universidade de aveiro