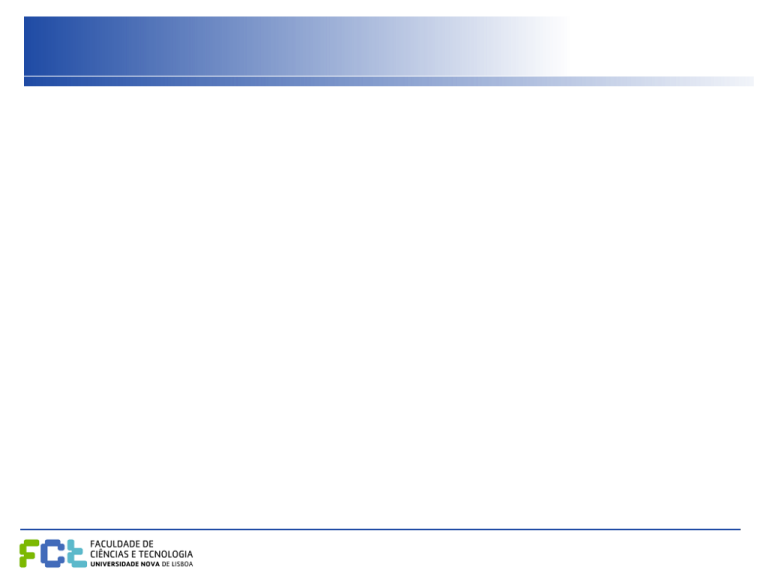
Aprendizagem Automática 

Ensemble Methods

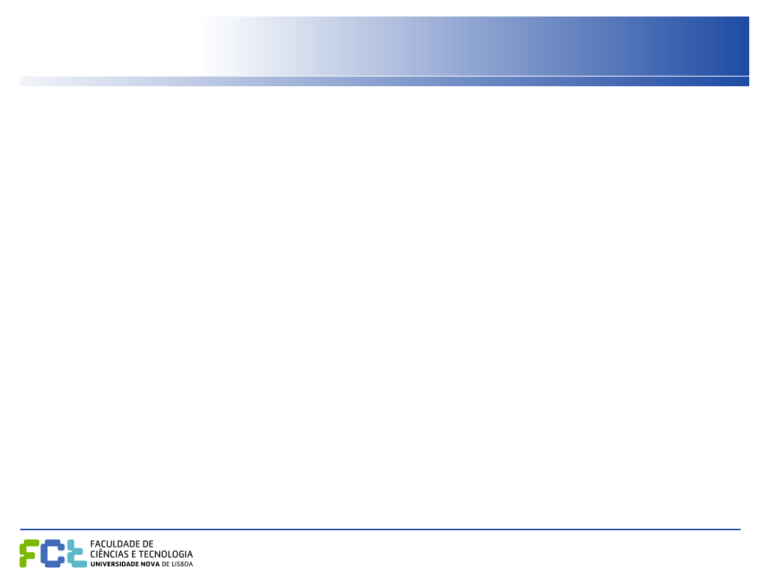
**Ludwig Krippahl**

Ensemble Methods 

**Summary**

■ Ensemble methods ■ Bagging and bragging ■ Boosting and stumping

1

Ensemble methods 

Ensemble Methods

2

Ensemble methods

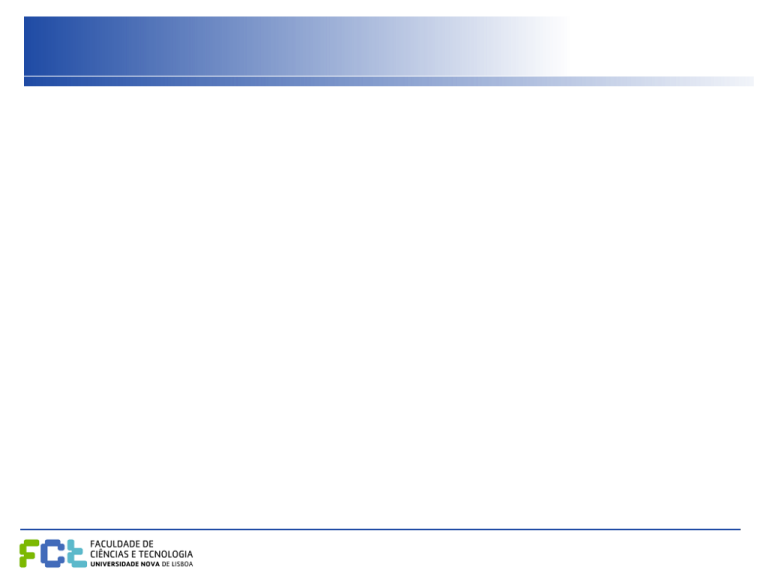
**Ensemble methods**

■ Combining groups of classifiers to improve classification **We'll focus on two different aproaches:**

■ Bootstrap aggregating : bootstrapping to train, combine predictions to reduce variance

■ Boosting : training a linear combination of weak classifiers (mainly) to reduce bias

3

Ensemble methods 

**Bagging**

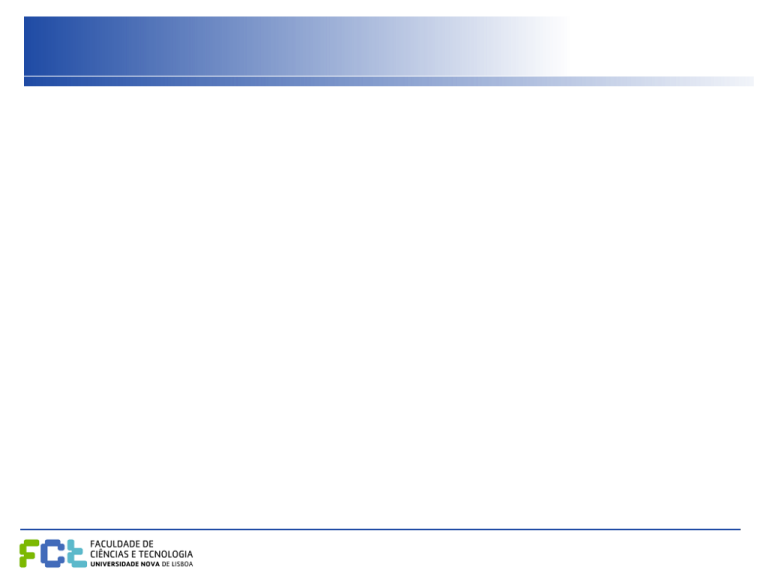
■ Bootstrap aggregating

• Use bootstrapping to generate replicas of training set

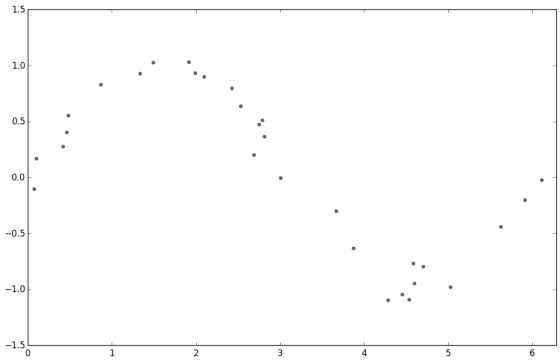
• Train model once per replica

• Aggregate the output of the hypotheses. Example: for regression, average the predictions

4

Ensemble methods 

■ Example: regression

5

Ensemble methods

■ Example: regression, mean

**def bootstrap(samples,data):**

**train\_sets = np.zeros((samples,data.shape[0],data.shape[1])) for sample in range(samples):**

**ix = np.random.randint(data.shape[0],size=data.shape[0]) train\_sets[sample,:] = data[ix,:]**

**return train\_sets**

**train\_sets = bootstrap(replicas,data)**

**px = np.linspace(ax\_lims[0],ax\_lims[1],points)**

**preds = np.zeros((replicas,points))**

**for ix in range(replicas):**

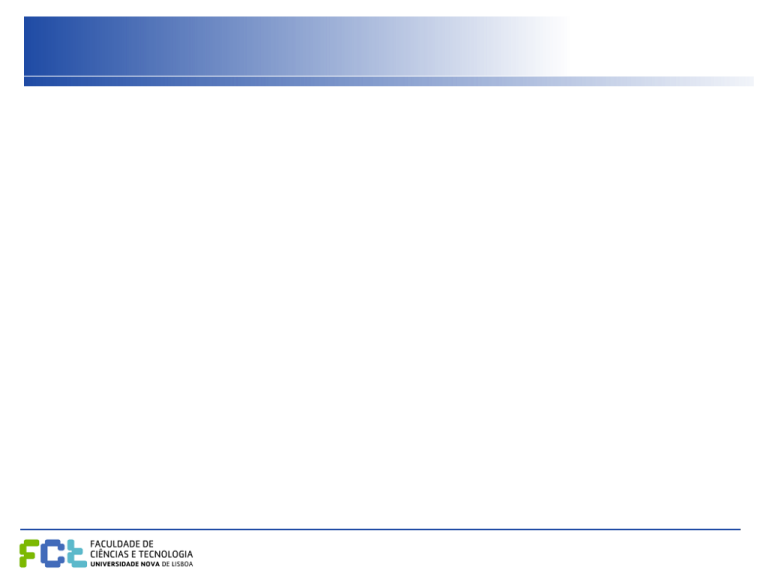
**coefs = np.polyfit(train\_sets[ix,:,0],**

**train\_sets[ix,:,1],degree)**

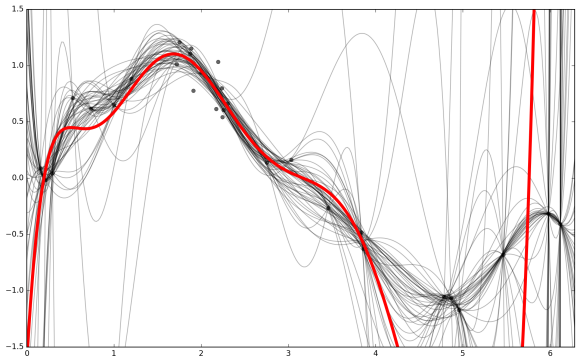
**preds[ix,:] = np.polyval(coefs,px)**

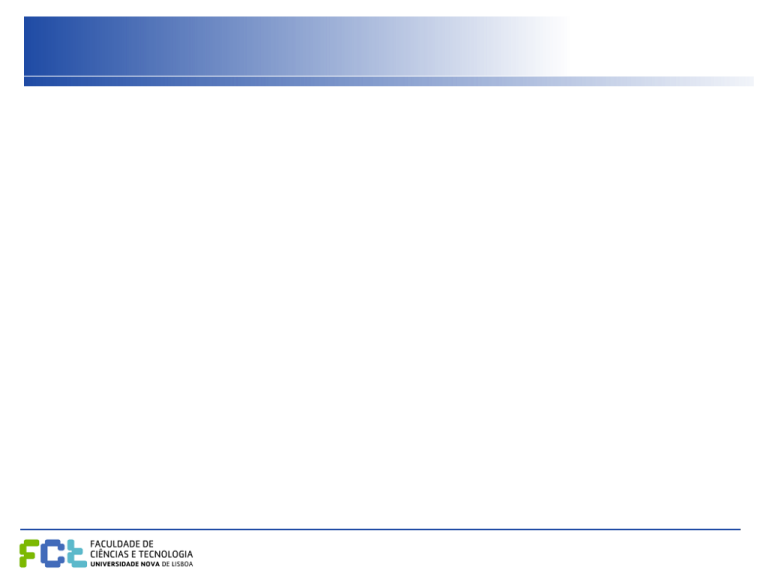
**mean = np.mean(preds,axis=0)**

6

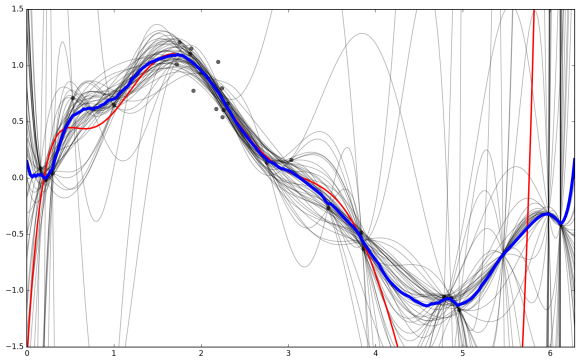
Ensemble methods 

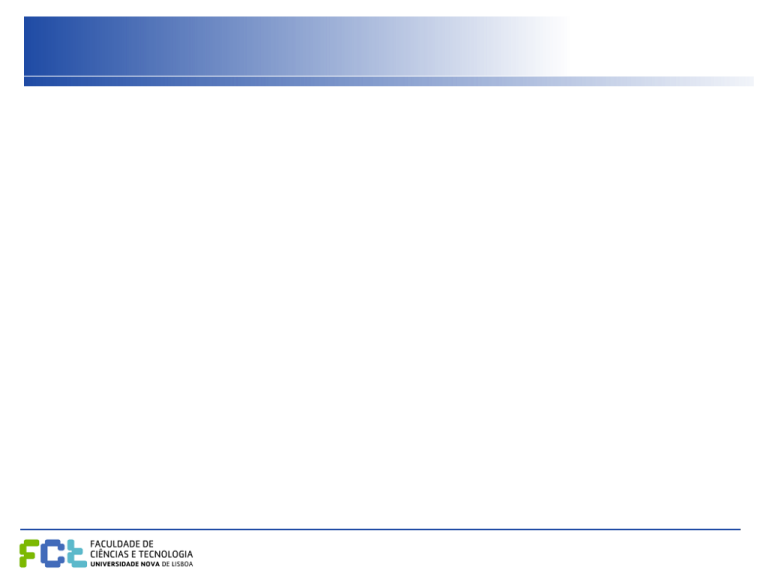
■ Example: regression, mean

7

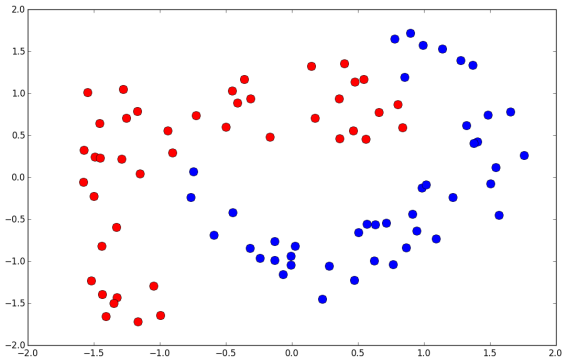
Ensemble methods 

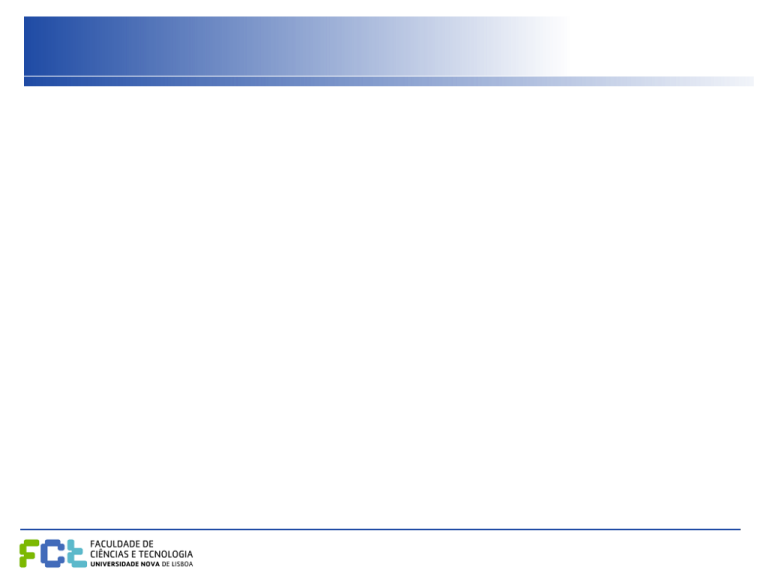
■ Variation: median instead of mean

8

Ensemble methods 

■ Classification example: SVM

9

Ensemble methods 

■ Classification example: SVM (majority vote)

**train\_sets = bootstrap(replicas,data)**

**gamma = 2**

**C=10000**

**svs = []**

**pX,pY = np.meshgrid(pxs,pys)**

**pZ = np.zeros((len(pxs),len(pys)))**

**for ix in range(replicas):**

**sv = svm.SVC(kernel='rbf', gamma=gamma,C=C)**

**sv.fit(train\_sets[ix,:,:-1],train\_sets[ix,:,-1])**

**svs.append(sv)**

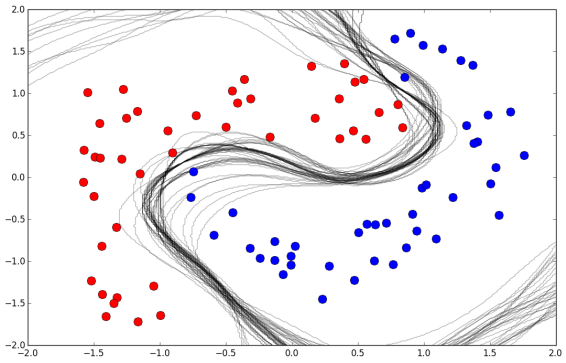
**preds = sv.predict(np.c\_[pX.ravel(),pY.ravel()]).reshape(pZ.shape) pZ = pZ + preds**

**pZ = np.round(pZ/float(replicas))**

10

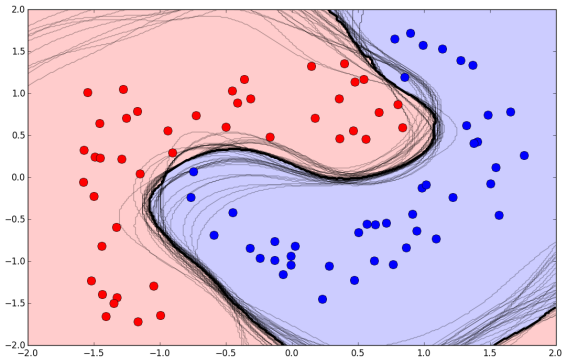
Ensemble methods

■ 50 SVM, trained with bootstrapping

11

Ensemble methods

■ Majority class of 50 SVM

12

Ensemble methods

**Bagging (Bootstrap aggregating)**

■ Averaging reduces variance and overfitting, increasing probability of correct classification as number of classifiers increases T T

kpk)T−k

( ) ∑ (1 − p

k=T/2+1



13

Ensemble methods 

**Bagging (Bootstrap aggregating)**

■ Bootstrap aggregating classifiers

T T

kpk)T−k

( ) ∑ (1 − p

k=T/2+1

■ This assumes classifiers are independent ■ If classifiers are correlated, this does not work so well ■ Bagging is best for unstable algorithms • (susceptible to input variations)

14

Ensemble methods 

Boosting

15

Boosting 

**Boosting**

■ Learn a linear combination of weak classifiers

■ Individual classifiers must have error rate below 0.5

■ Combination of classifiers has a lower bias and better classification power

16

Ensemble methods

**AdaBoost**

wn = 1/N

■ Initialize sample weights: 

ym(x)

■ Fit classifier by minimizing weighted error

N

wnm ym xntn

Jm = ∑ I( ( ) ≠ )

n=1

■ Compute weighted error on training set: Nwnm ym xntn

∑ I( ( ) ≠ )

ϵm =

n=1

Nwnm

∑

n=1

17

Ensemble methods

■ Compute classifier weight:

αm = ln1 − ϵm

ϵm

1

αm = ln

1−ϵm

• Original (Freund and Schapire,2003):

2

ϵm

■ Compute new sample weights (and normalize): nm+1 wnm αm ym xntn 

w = exp( I( ( ) ≠ ))

■ Increases weight of misclassified points

ϵm

■ Stop when is zero or greater than 0.5

■ Output of the boosted classifier is weighted sum of classifiers: M

f(x) = sign ∑ (x)

αmym

m=1

18

(Decision Tree) 

**Decision tree algorithm**

■ Split data into 2 subsets according to some feature and rule x1 ≤ 1

• e.g.

■ Use some measure of information gain to evaluate the split

• Classification error: assuming most common class in each subset G = 1 − ∑c p2c

• Gini Index:

Entropy = ∑ log c pc pc

• Information Entropy:

■ Choose one feature and rule that optimizes information gain ■ Repeat for each subset with mixed classes

19

(Decision Tree) 

**Decision tree algorithm**

■ Example:

20

(Decision Tree) 

**Decision tree algorithm**

■ Example:

21

(Decision Tree)

**Decision tree algorithm**

■ Example:

22

Boosting

■ Stumping: AdaBoost with decision stumps (level 1 decision tree) • Choose one feature, split at one point

• Use DecisionTreeClassifier

**from sklearn.tree import DecisionTreeClassifier**

**hyps = []**

**hyp\_ws = []**

**point\_ws = np.ones(data.shape[0])/float(data.shape[0])**

**max\_hyp = 50**

**for ix in range(max\_hyp):**

**stump = DecisionTreeClassifier(max\_depth=1)**

**stump.fit(data[:,:-1], data[:,-1], sample\_weight = point\_ws) pred = stump.predict(data[:,:-1])**

**errs = (pred != data[:,-1]).astype(int)**

**err = np.sum(errs\*point\_ws)**

**alpha = np.log((1-err)/err)**

**point\_ws = point\_ws\*np.exp(alpha\*errs)**

**point\_ws = point\_ws/np.sum(point\_ws)**

**hyps.append(stump)**

**hyp\_ws.append(alpha)**

23

Boosting

■ Stumping: AdaBoost with decision stumps (level 1 decision tree)24

Boosting

■ Stumping: AdaBoost with decision stumps (level 1 decision tree)25

Boosting

■ Stumping: AdaBoost with decision stumps (level 1 decision tree)26

Boosting

■ Stumping: AdaBoost with decision stumps (level 1 decision tree) ■ Classifying data and computing error

**net\_pred = np.zeros(data.shape[0])**

**for ix in range(len(hyps)):**

**pred\_n = hyps[ix].predict(data[:,:-1])**

**preds = preds+pred\_n\*hyp\_ws[ix]**

**net\_pred[preds<0] = -1**

**net\_pred[preds>=0] = 1**

**errors = np.sum((net\_pred !=data[:,-1]).astype(int))**

27

Boosting 

**AdaBoost, derivation**

■ We can see AdaBoost as a sequential mimization of the exponential error function:

N

E = ∑exp(− ( ))

tn fm xn

n=1

fm(x) m

■ Where is the weighted classification of the classifiers:

1

m

fm(x) = (x)

αjyj

2 ∑

j=1

f1. . . fm−1

■ All are assumed constant αmym(x)

■ Minimize only for the last one,

28

Boosting

■ We can decompose the error in correctly and incorrectly classified:

N

wnm12tnαmym xn e−αm/2 ∑

wnm eαm/2

wnm

E = exp(−( )) ∑ = +

n∑

n=1

N

−αm/2 ∑

n∈

N

∈

wnm eαm/2 e−αm/2 ∑

wnm ym xntn

= e + ( − ) I( ( ) ≠ )

n=1

ym

n=1

• Minimizing with respect to : 

N

wnm ym xntn

Jm = ∑ I( ( ) ≠ )

n=1

αm

• Minimizing with respect to :

1 − ϵm

N

N

wnm ym xntn ∑

αm = ln = I( ( ) ≠ ) /

ϵmϵm ∑

n=1

wnm

n=1

■ AdaBoost minimizes the exponential error of the linear combination

of the base classifiers with a sequential optimization. 29

Ensemble methods 

■ Two examples, to illustrate solutions to different problems. **Bagging**

■ Averages predictions based on different datasets (bootstrapping) ■ Good for models with low bias and high variance (overfitting) **Boosting**

■ Computes linear combination of weak classifiers (changing example weights)

■ Good for models with high bias and low variance (underfitting)

30

Ensemble methods 

First test

31

First Test 

**First test**

■ Lectures 1-12 (this one).

■ Next 2 sessions (lectures 13-16) not for first test. ■ Session of November 17 for questions and revisions ■ The test will be online. I will post a demo first.

■ Exam will be scored in two independent parts

• The test score for each part is guaranteed

■ Test will not include questions on Assignment 1 (unlike previous year)

32

Ensemble methods 

Summary

33

Ensemble methods 

**Summary**

■ Bagging: reduce variance by averaging

• Useful for models with large variance

• Useful for unstable models, otherwise there is too much correlation ■ Boosting: reduce bias by linear combination of classifiers • Useful for combining weak classifiers (large bias)

• Note: must be able to weigh samples

**Further reading**

■ Alpaydin, Sections 17.6, 17.7

■ Marsland, Chapter 7

■ Bishop, Sections 14.2, 14.3

34