# Feedback of project 1

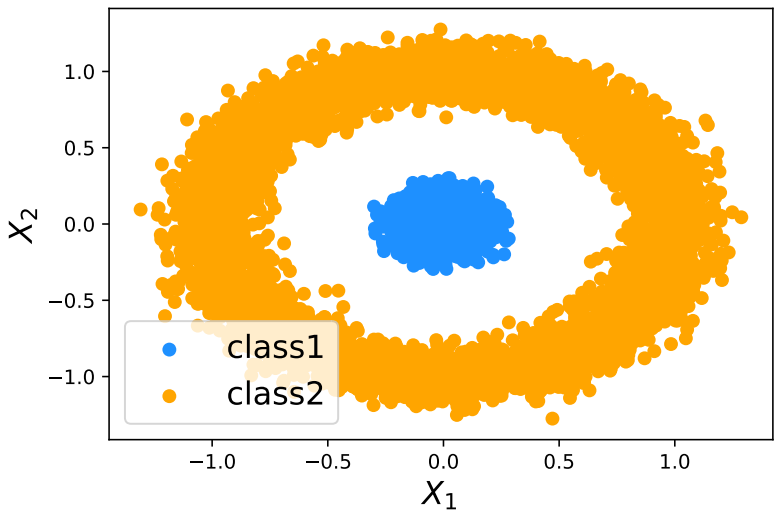
Antonio Sutera
a.sutera@uliege.be

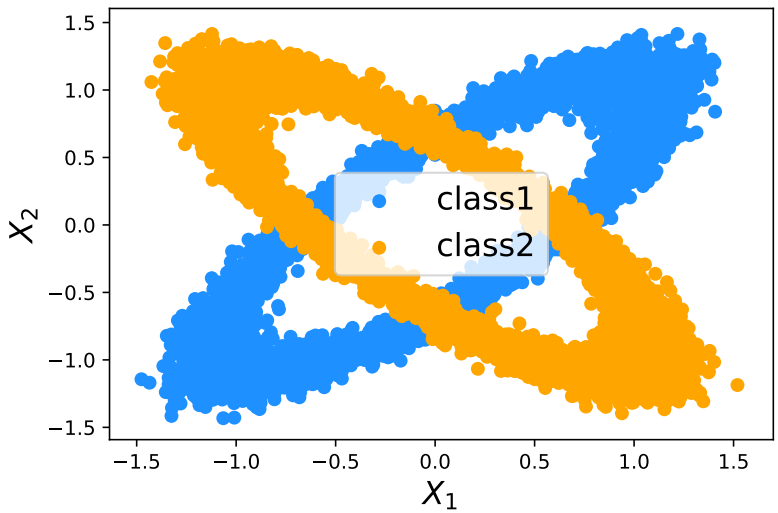
Institut Montefiore, University of Liège, Belgium


ELEN062-1
Introduction to Machine Learning
October 2020

# Introduction

Two datasets with two input variables, and two classes (output values).



# Q1 (DT): Questions

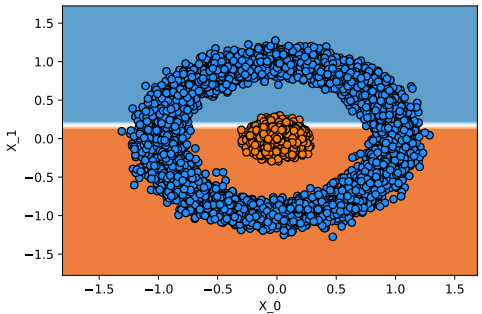
1. For both problems, observe how the decision boundary is affected by tree depth:
   (a) illustrate and explain the **decision boundary** for each depth;
   (b) discuss when the model is clearly **under- and over-fitting** and detail your evidence for each claim;
   (c) explain why the **model seems more confident when the depth is unconstrained**.
2. Report the **average test set accuracies (over five generations of the dataset)** along with the standard deviation for each depth. Briefly comment on them.
3. Based on both the decision boundaries and the test accuracies, discuss the differences between the two problems.

# Q1 (DT): Algorithm of the decision tree

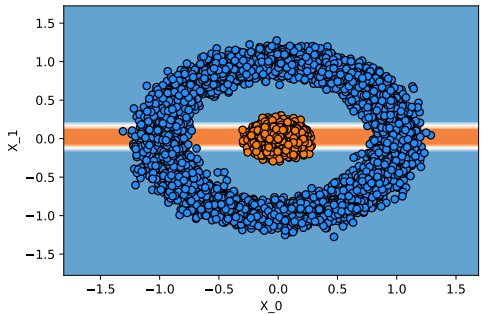
**Algorithm 1:** learn_dt($LS$)

**if** *all objects from* $LS$ *have the same class or if all objects have the same values for every attribute* **then**
  Create a leaf with a label corresponding to the majority class of the objects of $LS$;
**end if**
**else**
  Use $LS$ to find the best splitting attribute $A^*$ ;
  Create a test node for that attribute ;
  **forall** *different values* $a$ *of* $A^*$ **do**
    Build $LS_a = \{o \in LS \mid A^*(o) \text{ is } a\}$ ;
    Use learn_dt($LS_a$) to grow a subtree from $LS_a$
  **end forall**
**end if**

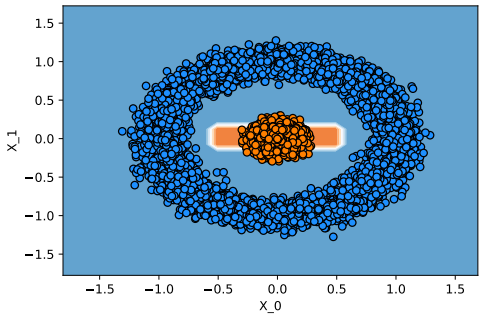
# Q1 (DT): decision boundaries (1)



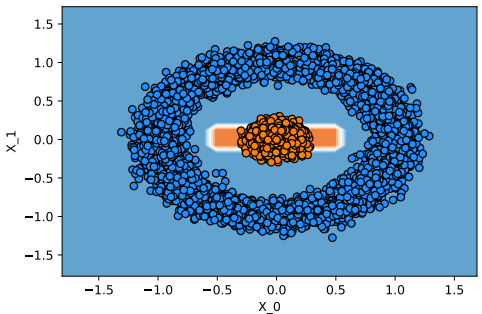

(a) Depth = 1



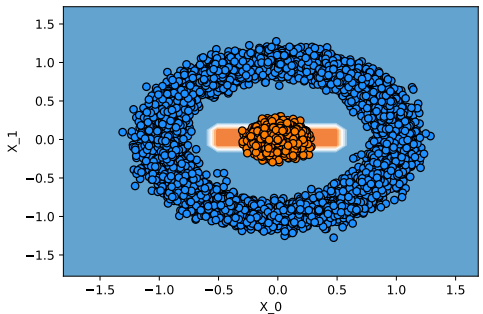

(b) Depth = 2




(c) Depth = 4

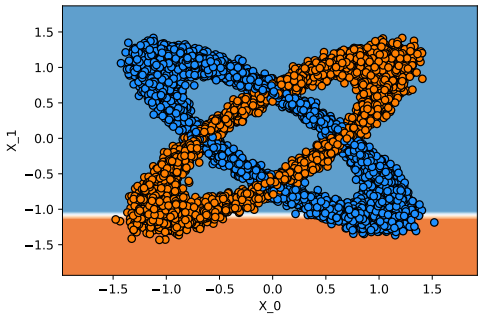



(d) Depth = 8

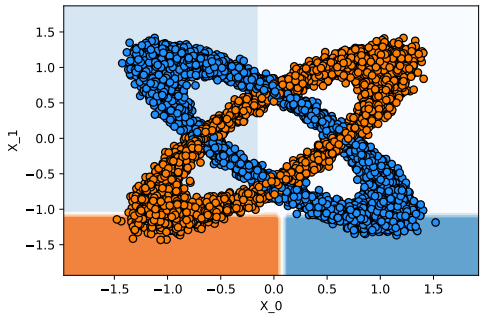



(e) Depth = None

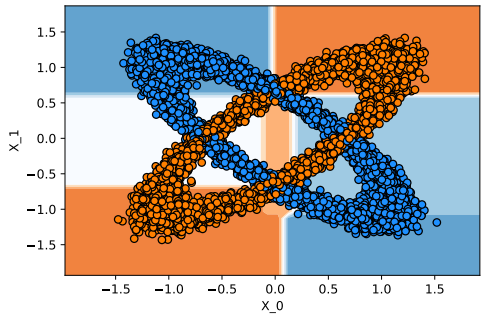
# Q1 (DT): decision boundaries (2)



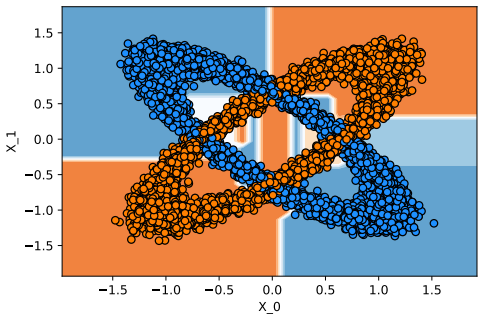

(a) Depth = 1



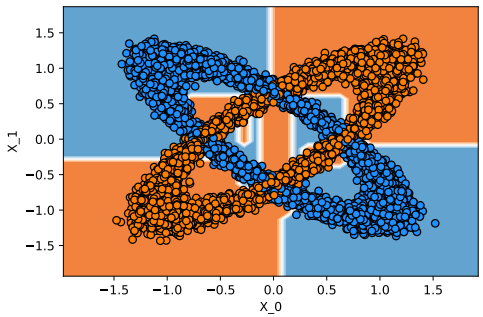

(b) Depth = 2




(c) Depth = 4



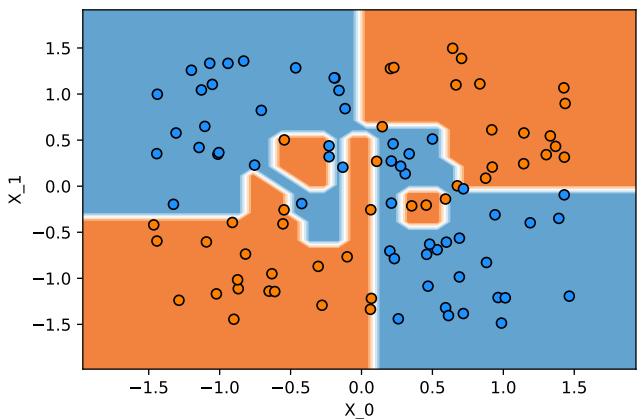

(d) Depth = 8




(e) Depth = None

# Q1 (DT): Why should we plot the TS instead of the LS?

What do you think in terms of under- or over-fitting?

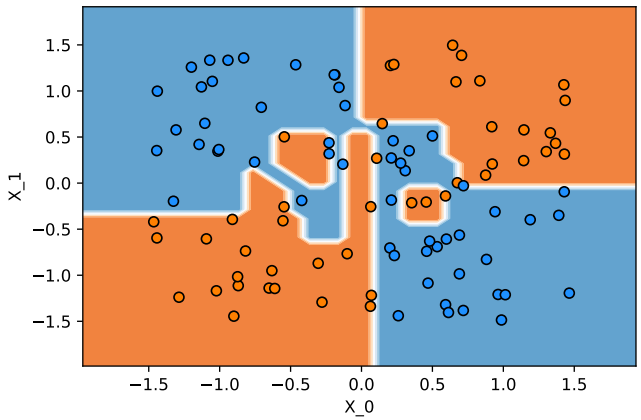



(a) Boundary decision with LS.

Can you say anything?

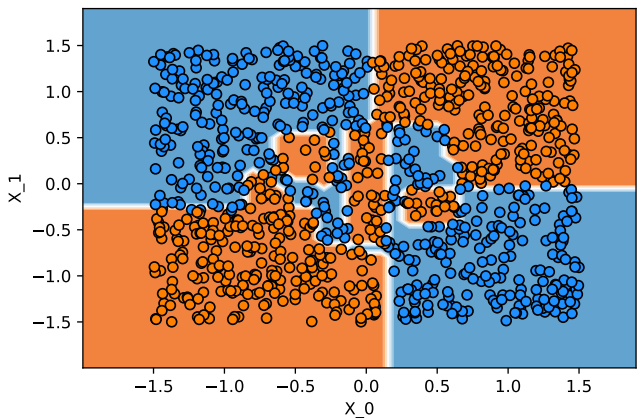
# Q1 (DT): Why should we plot the TS instead of the LS?

What do you think in terms of under- or over-fitting?




(a) Boundary decision with LS.

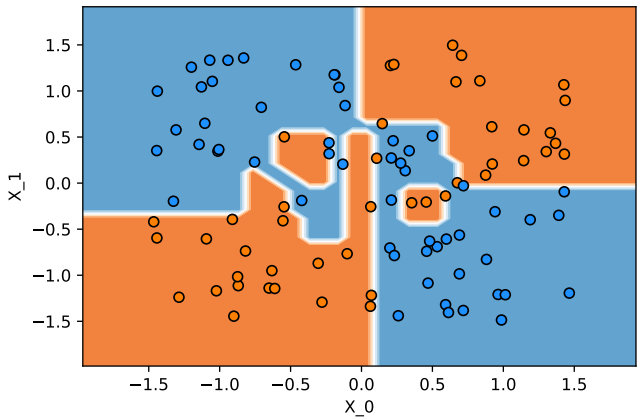
No over-fitting!




(b) Boundary decision with TS.

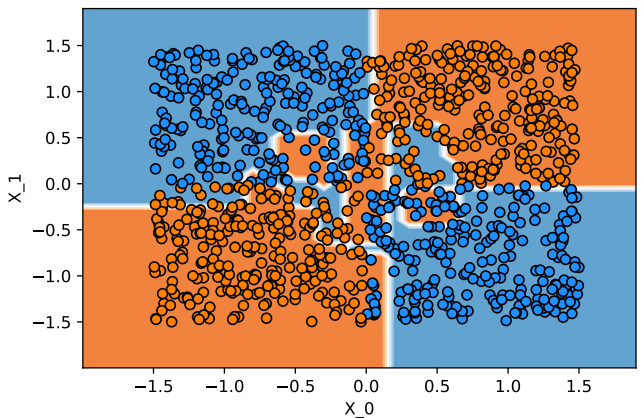
# Q1 (DT): Why should we plot the TS instead of the LS?

What do you think in terms of under- or over-fitting?



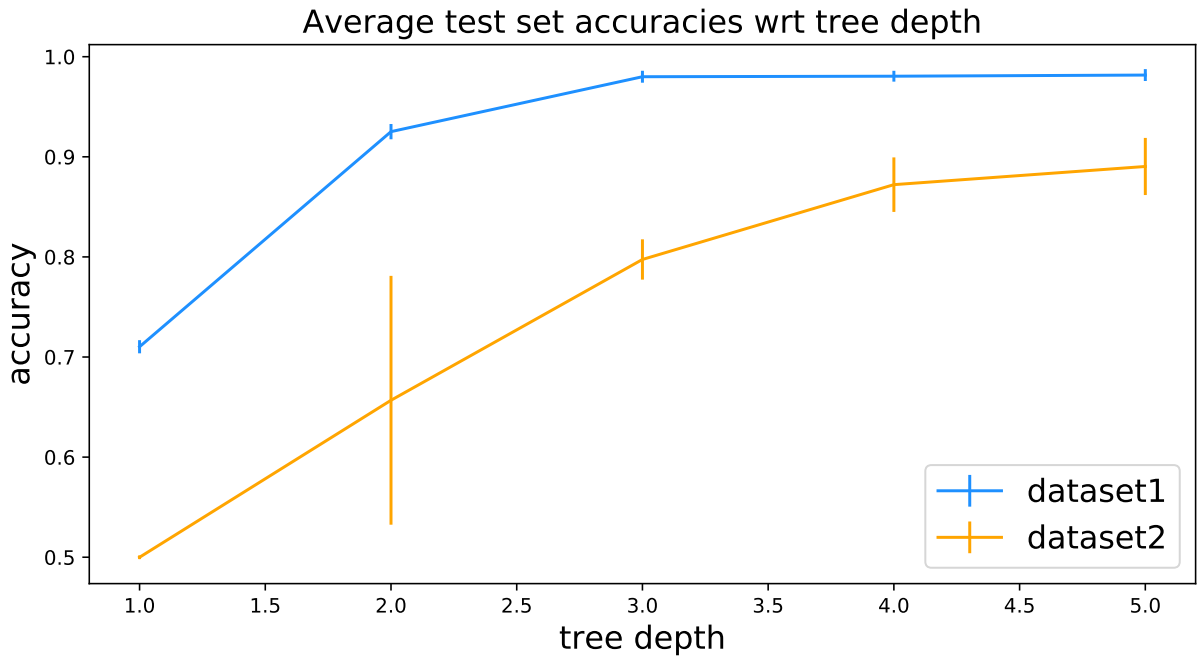

(a) Boundary decision with LS.

Over-fitting!



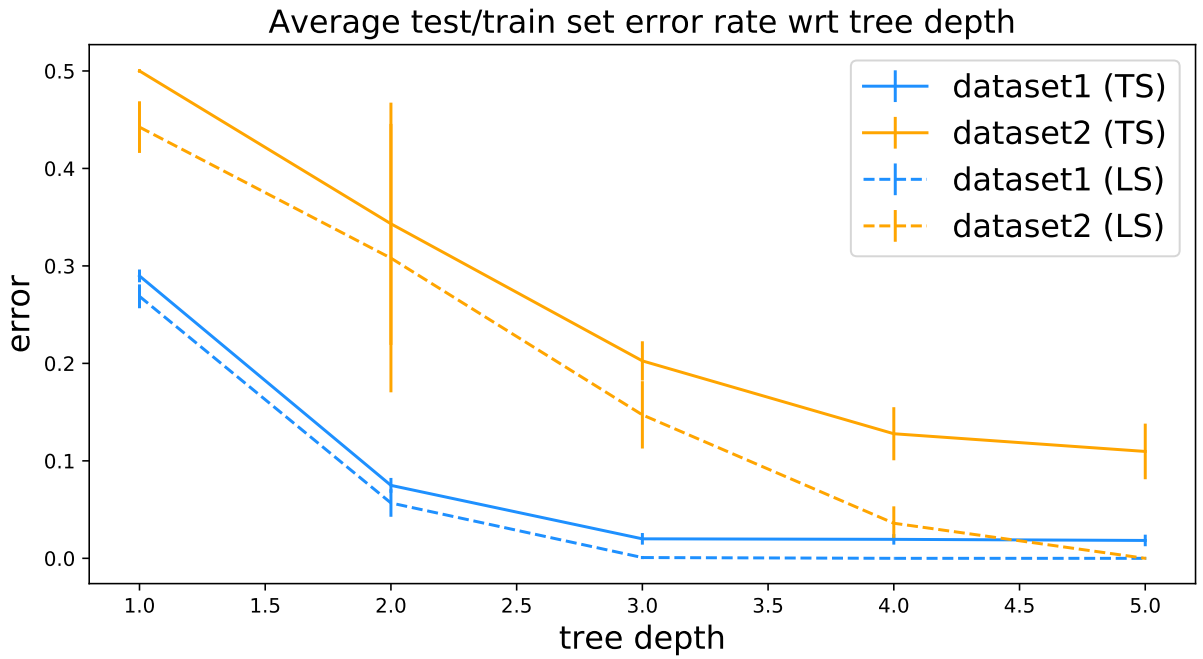

(b) Boundary decision with TS.

# Q1 (DT): test accuracies over five generations

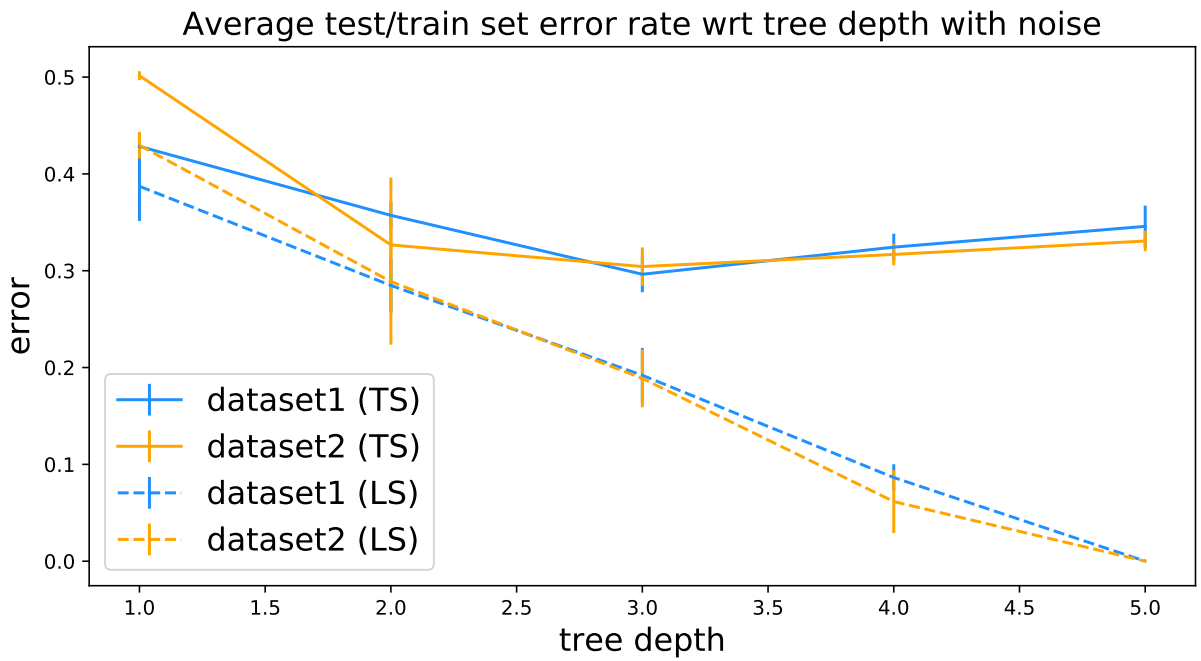


# Q1 (DT): test accuracies over five generations

# Q1 (DT): test accuracies over five generations



# Q1 (DT): test accuracies over five generations




For sake of illustration, same with **noise**.

# Q2: K-nearest neighbors

1. For both datasets, observe how the **decision boundary is affected by the number of neighbors**:

   [...]

2. Optimize the value of the `n_neighbors` parameter using a five-fold **cross validation strategy** and obtain an **unbiased estimate of the test accuracy** for the second dataset:

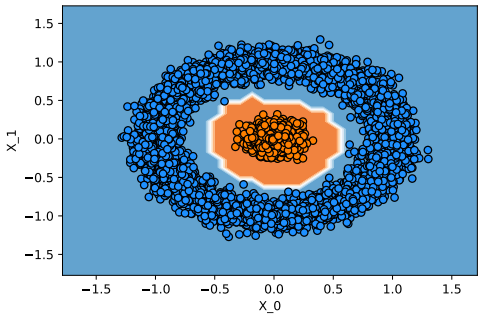
   [...]

3. For both datasets, observe the **evolution the optimal value** of the number of neighbors with respect to the size of the learning sample set. To do so:

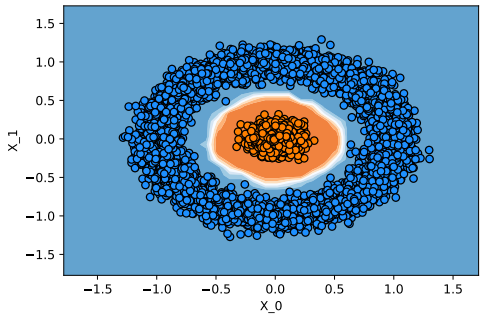
   [...]

4. Given the results of question 2.3 and a LS of size 250, **what do you think of using five-fold cross-validation to determine the optimal value of `n_neighbors` as you did in question 2.2**? Discuss.

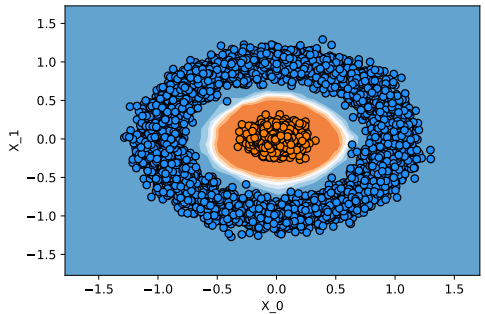
# Q2 (KNN): decision boundaries (1)



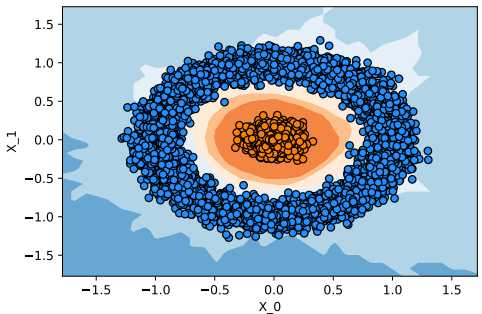

(a) Nb of neigh. = 1



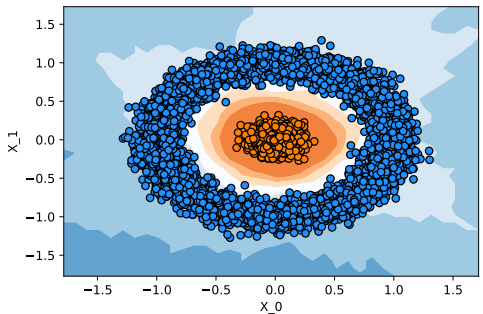

(b) Nb of neigh. = 5



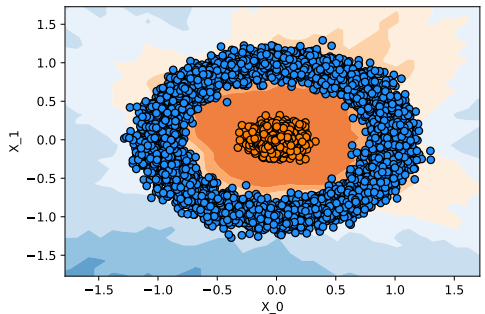

(c) Nb of neigh. = 10




(d) Nb of neigh. = 75

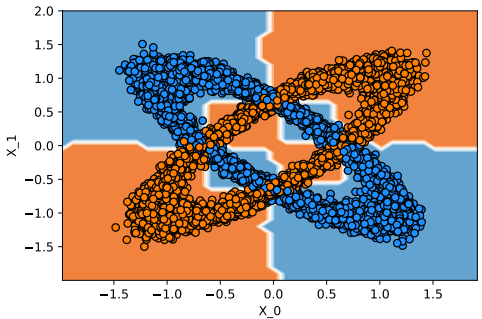



(e) Nb of neigh. = 100

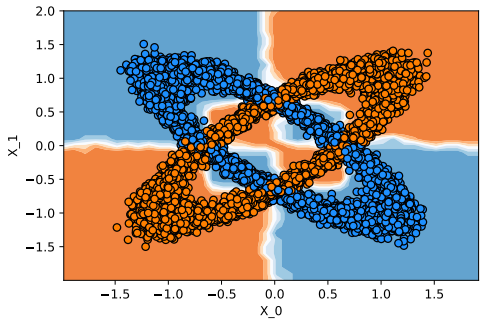



(f) Nb of neigh. = 150

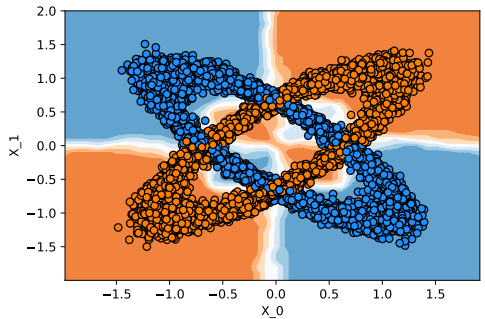
# Q2 (KNN): decision boundaries (2)



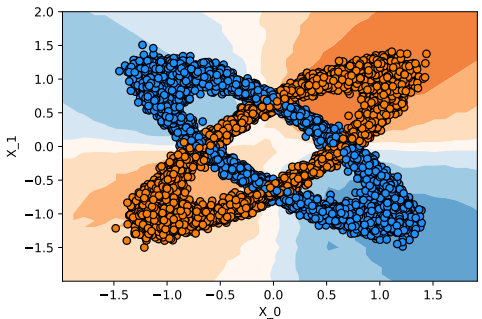

(a) Nb of neigh. = 1



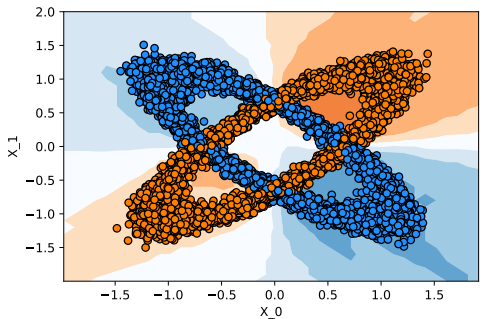

(b) Nb of neigh. = 5



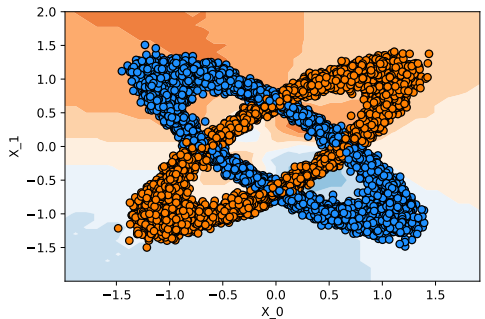

(c) Nb of neigh. = 10




(d) Nb of neigh. = 75




(e) Nb of neigh. = 100




(f) Nb of neigh. = 150

# Q2 (KNN): Estimating the performance of a model theory

**Given a model learned from some data set of size $N$, how to estimate its performance from this data set?**

What for?

- Model selection: choosing the best model among several models. **Example:** determining the right complexity of a model or choosing between different learning algorithms.
- Model assessment: having chosen a final model, it consists in estimating its performance on new data.

# Q2 (KNN): test set method theory

**Idea**: randomly divide the data set into two parts: a **learning set** and a **test set**.
**Example:** $70\% - 30\%$


Method:

1. Fit the model on the learning set
2. Test it on the test set

The resulting estimate is an estimate of the error of a model learned on the **whole** data set.

# Q2 (KNN): k-fold cross-validation theory

**Idea**: randomly divide the data set into $k$ subsets (*e.g.* $k = 10$).


Method:

- For each subset:
  1. Learn the model on the objects that are not in the subset.
  2. Compute a prediction with this model for the points in the subset.
- Report the mean error over these predictions.

When $k = N$, the method is called **leave-one-out** cross-validation.

# Q2 (KNN): using a cross validation strategy...

How do you obtain an **unbiased** estimate of the test accuracy for the second dataset?

Do you think it could be the mean test accuracy over the folds?

# Q2 (KNN): Model selection: typical scenario theory

Given a data set of $N$ objects (input-output pairs), how to best exploit this data set to obtain:

- The best possible model (*e.g.* among regression trees and k-NN) $\rightarrow$ **model selection**
- An estimate of its prediction error $\rightarrow$ **model assessment**

# Q2 (KNN): Large data sets: test set method theory

**Idea**: randomly divide the data set into 3 parts:

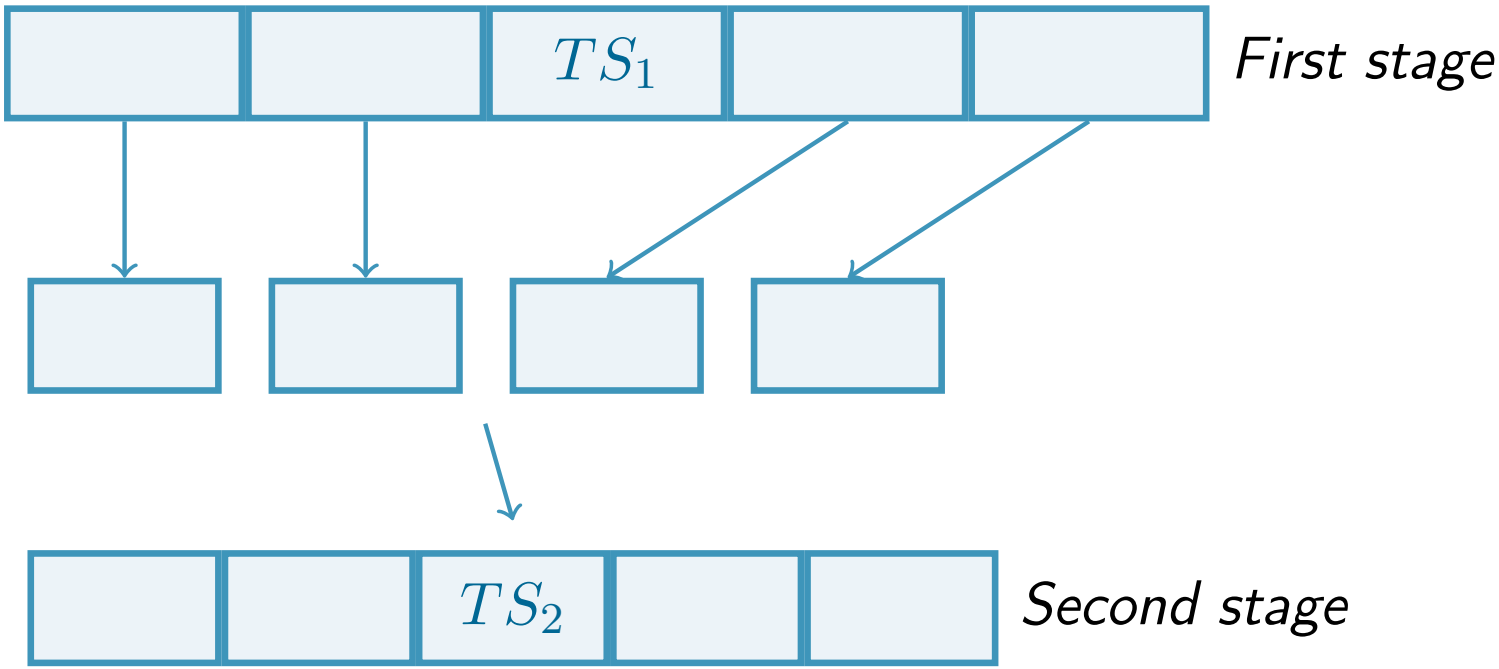
1. A learning set $LS$
2. A validation set $VS$
3. A test set $TS$

**Example:** $50\% - 25\% - 25\%$


1. Fit the models to compare on the learning set, using different algorithms or different complexity values.
2. Select the best one based on its performance on the validation set.
3. Retrain this model on $LS + VS$.
4. Test it on the test set $\rightarrow$ **performance estimate**.
5. Retrain this model on $LS + VS + TS$. This yields the finally chosen model.

# Q2 (KNN): Small data sets: cross-validation theory

**Idea**: use two stages of $k$-fold cross-validation.




The **first** stage is used for the **assessment** of the final model, while the **second** one is used for **model selection**.

***Note***: we could also combine test set and cross-validation.

# Q2 (KNN): unbiased estimate of the test accuracy

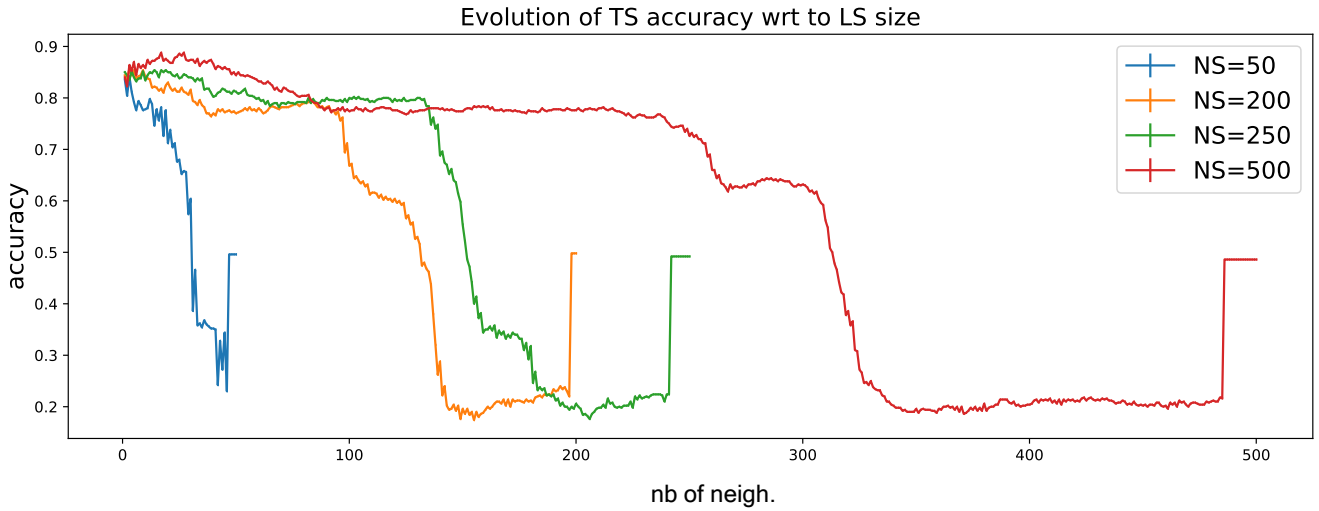
***Note***: we could also combine test set and cross-validation.

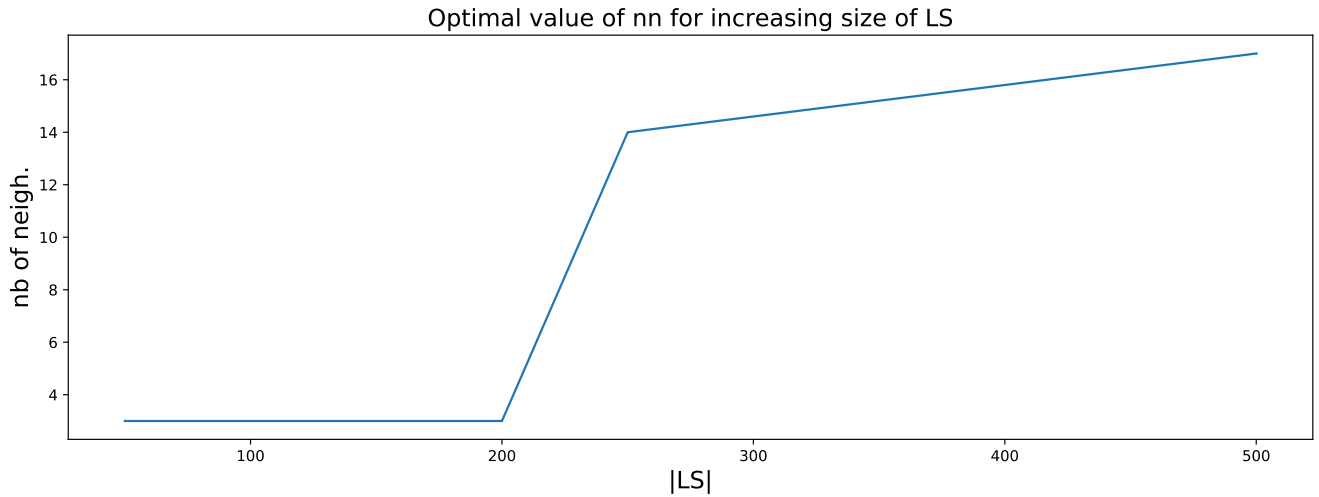
Test set method


Test set + CV method


CV (LS+VS in test set method)

# Q2 (KNN): optimal values



# Q3: Residual fitting

Residual fitting is a simple algorithm to fit iteratively a linear regression model **(see Lecture 3: linear regression, slide 20)**. We propose to implement this algorithm and to **use it here to address the two classification problems by encoding the two classes in numerical values 0/1**. Answer the following questions.

1. In the algorithm in the lecture slides, proof that the best weight $w_k$ for the attribute $a_k$ introduced in the model at step $k$ is $\rho_{a_k,\Delta_k y}\sigma_{\Delta_k y}$, where $\rho_{a_k,\Delta_k y}$ is the Pearson correlation between $a_k$ and $\Delta_k y$ and $\sigma_{\Delta_k y}$ is the standard deviation of $\Delta_k y$.
2. Implement the algorithm **as described in the slides**.
3. **Learn a residual fitting model on both datasets**:

   [...]
4. Learn a residual fitting model on a modified version of the second dataset that includes three new attributes corresponding to $X_1 * X_1$, $X_2 * X_2$ and $X_1 * X_2$, in addition to the two original ones $X_1$ and $X_2$:

   [...]

   **Comment on these results and compare them with those obtained in question 3.3**.

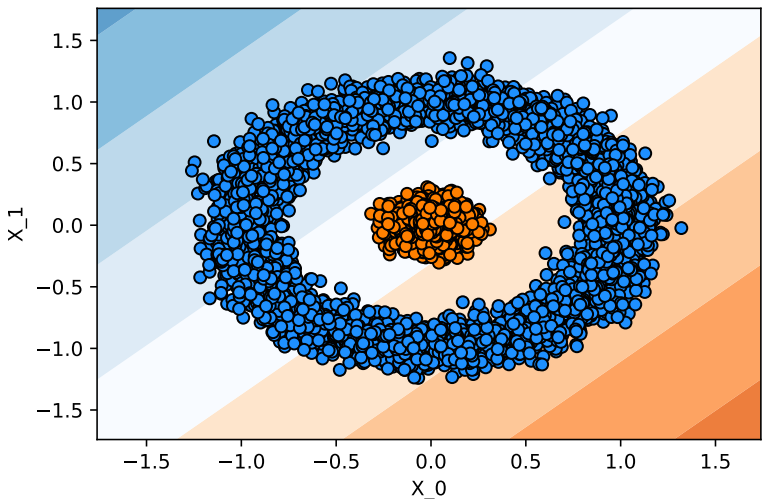
# Q3: Residual fitting (a.k.a. Forward-Stagewise Regression)

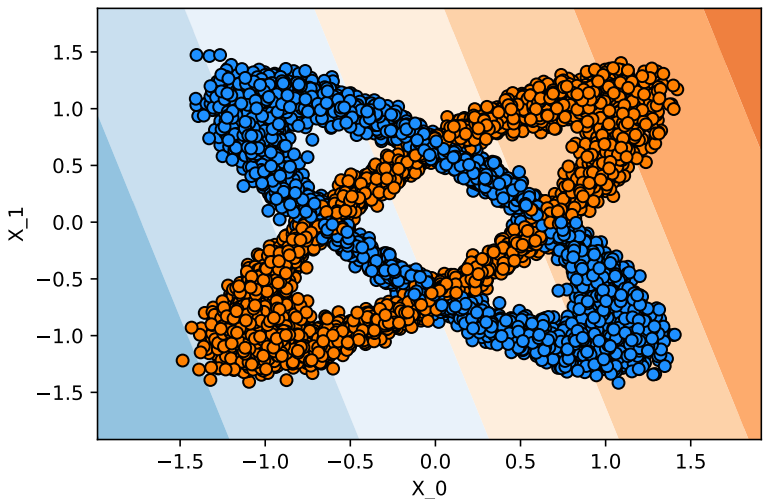
Residual fitting: alternative algorithm, of general interest

- Start by computing $w_0$ for the no-variable case: $w_0 = \overline{y}$
- Introduce attributes (assumed of zero mean, unit variance) progressively, one at the time
  - Define residual at step $k$ by
    $\Delta_k y(o) = y(o) - w_0 - \sum_{i=1}^{k-1} w_i a_i(o)$
  - Find best fit of residual with only attribute $a_k$:
    $w_k = \rho_{a_k, \Delta_k y} \sigma_{\Delta_k y}$.
    (since residuals have zero mean, and attributes are pre-whitened)

Note that this algorithm is in general suboptimal w.r.t. to the direct solution given previously, but it is linear in the number of attributes.

# Q3: residual fitting on datasets 1 and 2

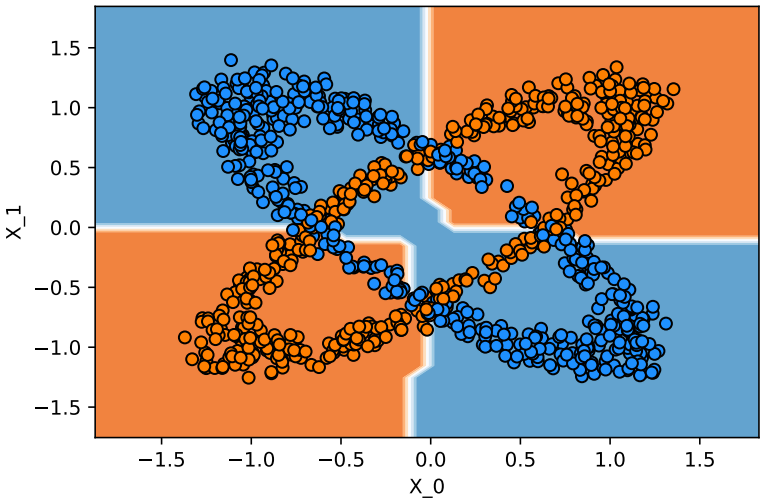


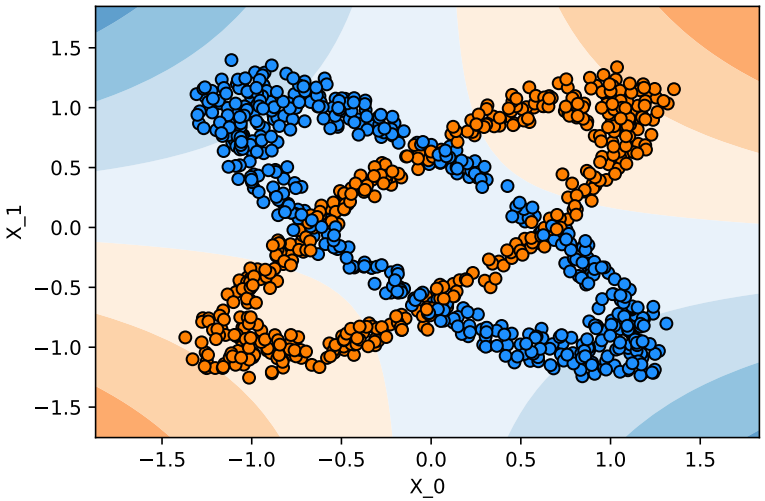




- Only $X_0$ and $X_1$.

$\Rightarrow$ Not very good.

# Q3: residual fitting on extended dataset 2

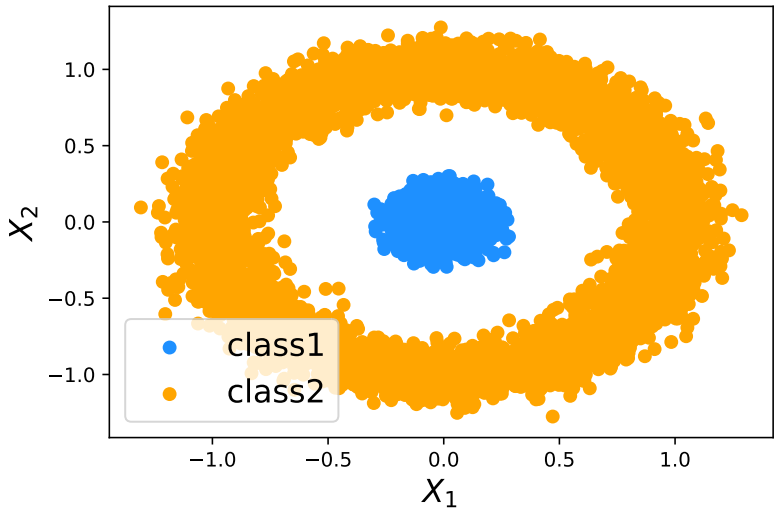



- $X_0$, $X_1$ AND $X_0^2$, $X_1^2$, $X_0 * X_1$.

$\Rightarrow$ Very good. Why?

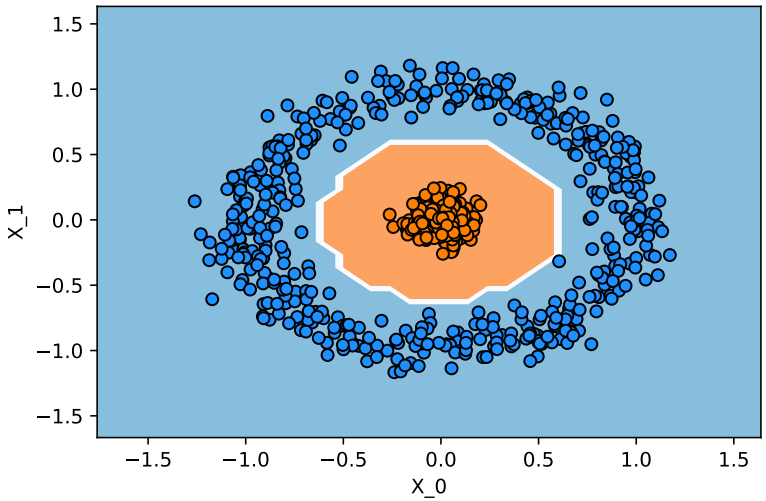
# Q3: residual fitting on extended dataset 2

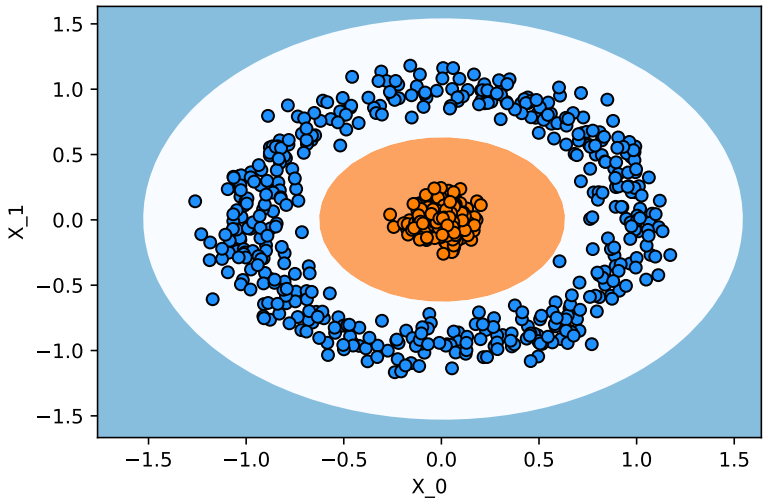



- $X_0$, $X_1$ AND $X_0^2$, $X_1^2$, $X_0 * X_1$.

- Is it supposed to work on the first dataset?
- Can we *create* other features that may be very useful for the first dataset?

# Q3: residual fitting on extended dataset 2




- $X_0$, $X_1$ AND $X_0^2$, $X_1^2$, $X_0 * X_1$.

- Is it supposed to work on the first dataset?
- Can we *create* other features that may be very useful for the first dataset?
- For instance $\sqrt{X_0^2 + X_1^2}$ and $\tan^{-1}\left(\frac{X_0}{X_1}\right)$

# Project 2: Reminder: bias/variance decomposition theory

$$E_{LS}\left\{E_{y|\underline{x}}\left\{(y-\hat{y}(\underline{x}))^2\right\}\right\} = \text{noise}(\underline{x}) + \text{bias}^2(\underline{x}) + \text{variance}(\underline{x})$$

- $\text{noise}(\underline{x}) = E_{y|\underline{x}}\left\{(y-h_B(\underline{x}))^2\right\}$:
  Quantifies how much $y$ varies from $h_B(\underline{x}) = E_{y|\underline{x}}\{y\}$ (the Bayes model).
- $\text{bias}^2(\underline{x}) = (h_B(\underline{x}) - E_{LS}\{\hat{y}(\underline{x})\})^2$:
  Measures the error between the Bayes model and the average model.
- $\text{variance}(\underline{x}) = E_{LS}\left\{(\hat{y} - E_{LS}\{\hat{y}(\underline{x})\})^2\right\}$:
  Quantifies how much $\hat{y}(\underline{x})$ varies from one learning sample to another.

# Project 2

- On bias and variance analysis
- Two parts: analytical derivations and empirical analyses
- By 17 November
- Concise report & codes must be submitted.

## By next week

- ☐ Register on the Submission Platform.
- ☐ Fill in the second form (Project 2) for updating Slack.