Tutorial on Conformal Predictors and Venn Predictors

Paolo Toccaceli

Computer Learning Research Centre
Royal Holloway, University of London

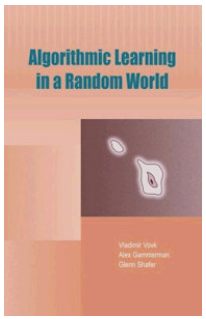
Egham, June 24, 2019

About the tutorial

- Informal presentation, with focus on concepts rather than on formalism.
  - What problems do Conformal Predictors and Venn Predictors solve?
  - What questions do they answer?
  - How do they work?
  - What do their guarantees really mean?
  - How can we use them?

Conformal Prediction and Venn Prediction

- Conformal Predictors and Venn Predictors
  - Probabilistic “distribution-free” ML techniques
  - Invented at RHUL CS Dept
    - Prof. Gammerman and Prof. Vovk are the fathers
    - Reference textbook:
      Gammerman, Vovk, Shafer,
      Algorithmic Learning in a Random World,
      Springer (2005)



    - Yearly symposium on the topic: COPA (2018, 7th edition)
    - Special issues on the topic in journals
  - Have theoretically proven guarantees, under minimal assumptions
  - Not just of theoretical interest:
    - CP and VP are currently used in drug development systems in a major pharmaceutical company

Hedging Prediction in Machine Learning: Conformal Predictors

Let's frame the problem

- Let's consider a specific Statistical Learning setting:
  Supervised Learning
  - We have a training set consisting of examples, each made up of an object and its label.
  - We are presented with an arbitrary object and we are asked to predict its label.
  - It's "supervised" because the algorithm is presented with the labels, as if provided by a supervisor.
  - In general, other settings are possible, e.g. unsupervised learning, reinforcement learning, etc.
- Often, practitioners content themselves in producing **bare predictions**, i.e. simply a label value, and overlook the **uncertainty** of the prediction.

The problem of prediction uncertainty

- Statistical learning theory (PAC, VC) provide bounds on the prediction errors.
- The bounds demonstrate it is possible to achieve arbitrarily good accuracy[1] with sufficiently large sizes of the training set.
- **Drawback**: these bounds are too loose to tell us anything interesting for training sets that we actually deal with in practice.
- A common practical and effective approach to estimate the accuracy is to use hold-out estimates: observe the rate of errors on a set separate (held-out) from that on which the learning algorithm is trained.
- By complementing the prediction value with a bound on the probability of error, we are effectively *hedging* the prediction.

[1] but of course within the limits of Bayes error

CP: a different approach to “hedging”

- Instead of producing a prediction and estimating its error, Conformal Predictors allow a different way of hedging predictions.
- CP can take almost any ML method and use it to output predictions with a **chosen** probability of error.
- The only assumption is that training and test data be i.i.d.
- However, to achieve that, there is a “price” to pay.
- The predictions are no longer single-valued: the prediction consists of a **set** (or region) of label values.
- This hedged prediction is considered correct when the prediction set contains the actual label (otherwise, it is considered an error).
- Note that predictions can also be empty.

Conformal Prediction as a framework

- Conformal Predictors are not a self-contained ML method, but should rather be viewed as a framework.
- Conformal Predictors operate on “top” of another ML algorithm (referred to as ’underlying’).
- Any classification or regression algorithm can be used, as long as one can extract a score from it (as opposed to just a “hard” prediction).
  - SVM, Decision Trees, kNN, Neural Network, Naïve Bayes,. . .

Conformal Predictors

- Two desiderata:

  Validity: the long run frequency of error does not exceed the **significance** level ϵ at each chosen **confidence** level $1 - \epsilon$.

  Efficiency: the prediction set should be as small as possible.

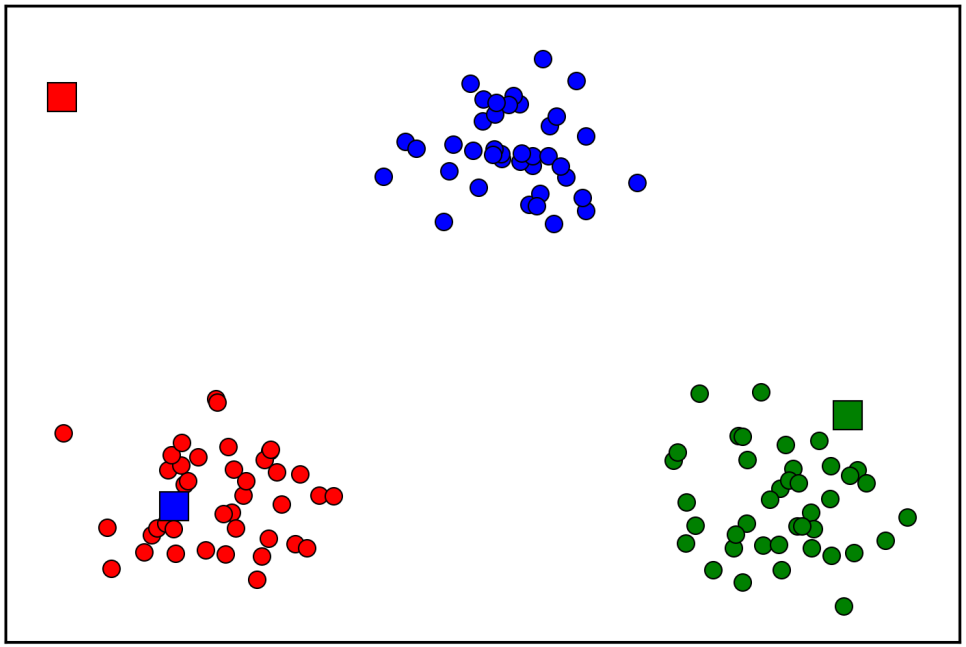
- As long as training and test data are independent and identically distributed (**i.i.d. assumption**), validity is guaranteed by CP and does not depend on the algorithm.

- The efficiency depends on the underlying algorithm.

- CP make it possible to avoid worrying about validity and to focus only on improving efficiency.

Why “conformal”

- The notion of **conformity** is at the heart of Conformal Prediction.
- How “likely” does it seem that this example comes from the same distribution that generated this training set?



- The **nonconformity measure** (NCM) quantifies (on an arbitrary but consistent scale) how random an example is, compared to a training set

Nonconformity measure

- The nonconformity measure is implemented using the underlying ML method.
- Example using nearest neighbours as underlying:

$$\alpha_i := \frac{\sum_{j=1}^{k} d_{ij}^{+}}{\sum_{j=1}^{k} d_{ij}^{-}}, \quad i = 1, \ldots, n,$$

with the elements (x_i, y_i) of our data set, where d_{ij}^{+} is the jth shortest distance from x_i to other objects labelled in the same way as x_i, and d_{ij}^{-} is the jth shortest distance from x_i to the objects labelled differently from x_i.

- Example: the NCMs for $k = 3$ for the examples in the previous page are: 12.14, 0.27, 1.19

An outline for the Classification case

- Conformal Prediction works by making hypotheses as to the value of the label y of the test object $\mathbf{x}_{test}$.
- The hypothesis we test is that the hypothetical example $(\mathbf{x}_{test}, y_{hyp})$ was drawn i.i.d from the same distribution as the training examples.
- We compute the *p-value* of this hypothesis
- We reject those hypotheses whose p-value is less than the significance level ϵ
- The labels of the hypotheses we could not reject constitute the prediction set.
- Gammerman and Vovk showed how to compute the p-values and proved that this procedure outputs predictions that have the validity property.

The details 1/2

Denoting the training set with $(x_1, y_1), \ldots, (x_\ell, y_\ell)$ and the test object with $x_{\ell+1}$, the Conformal Prediction is obtained as follows:

For every possible label $y \in Y$:

- we form the hypothetical "completion" $z_{\ell+1} = (x_{\ell+1}, y)$
- we compute $\alpha_1, \ldots, \alpha_\ell, \alpha_{\ell+1}$ as:

$$\alpha_i = \mathcal{A}(\{z_1, z_2, \ldots, z_{\ell+1}\} \setminus z_i, z_i)$$

where $\{z_1, z_2, \ldots, z_n\} \setminus z_i$ denotes the bag $\{z_1, z_2, \ldots, z_n\}$ with z_i removed.

The details 2/2

The *p-value* for the completion is then obtained with:

$$p_Y := \frac{|\{i = 1, \ldots, \ell + 1 : \alpha_i \geq \alpha_{\ell+1}\}|}{\ell + 1},$$

In words, the *p-value* is the proportion of the αs which are at least as large as the last α , i.e. it is the proportion of examples that appear "stranger than" or "as strange as" the completion.
Given ϵ, the region prediction is then defined as:

$$\Gamma^\epsilon (x_1, y_1, \ldots, x_\ell, y_\ell, x_{\ell+1}) := \{y \in Y : p_y > \epsilon\}$$

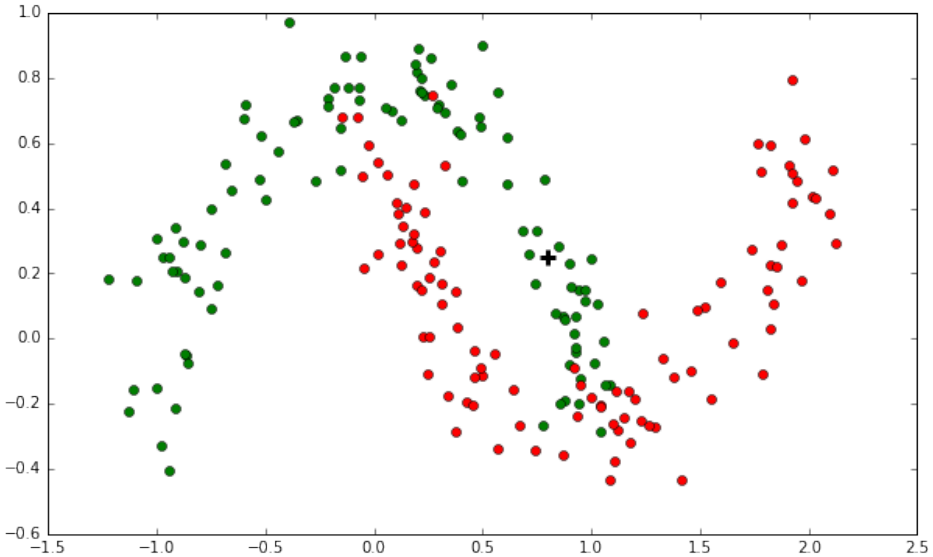
Practical Considerations

- Note that in the definition of α_i in slide 13 the bag varies from one α_i to the next.
- This means that the underlying algorithm is to be trained on a different training set for every α_i.
- If we have ℓ examples in the training set, we have to train ℓ instances of the underlying algorithm to calculate one *p-value* p_y.
- In addition, we need to repeat the calculation for every possible value that the label can take (2 in the simplest case).
- The resulting computational load can be prohibitive.

Just to be clear...

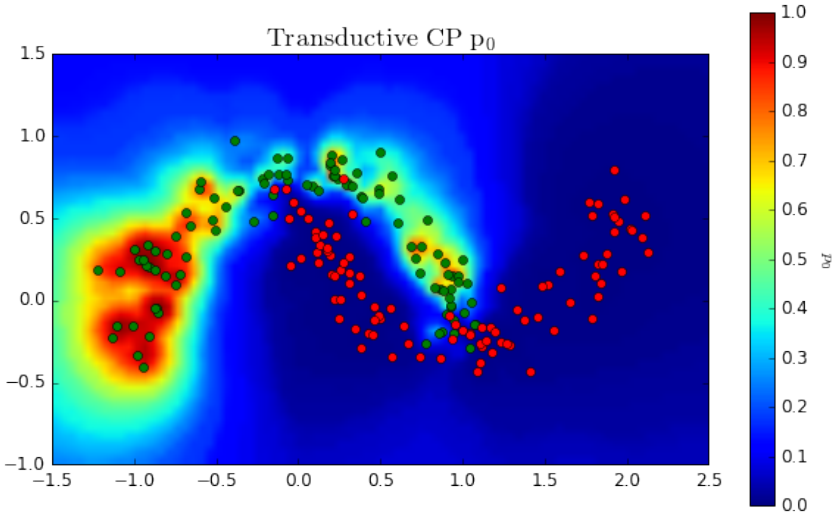
- CP computes **p-values** in order to produce a prediction set.
- p-values are **not posterior probabilities**!
  - They do not express the probability of object **x** having label y.
  - They express the probability of drawing, from the same distribution, an example that is as or more contrary to hypothesis of being drawn iid from the same distribution as the training set.
- Similar to p-value in classical Statistical Hypothesis Testing, i.e. the probability, under the Null Hypothesis, of an outcome as or more contrary to the Null Hypothesis.
- p_0 and p_1 do not have to sum to 1!

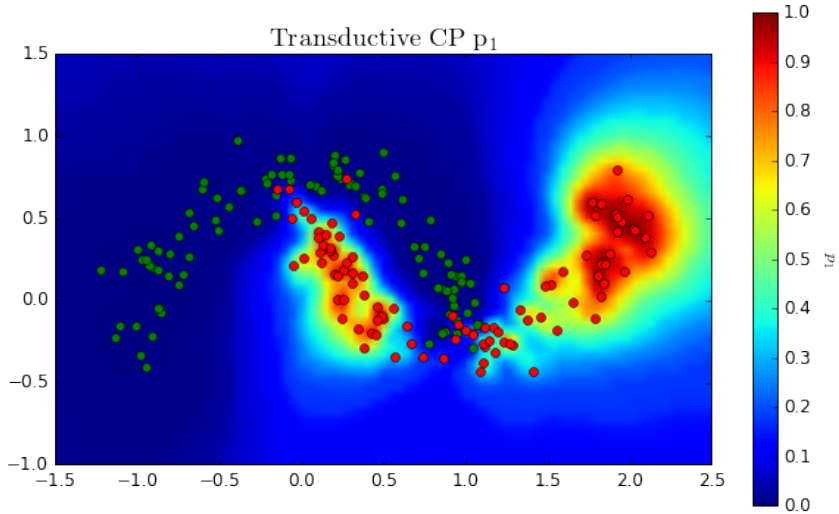
An example on synthetic data

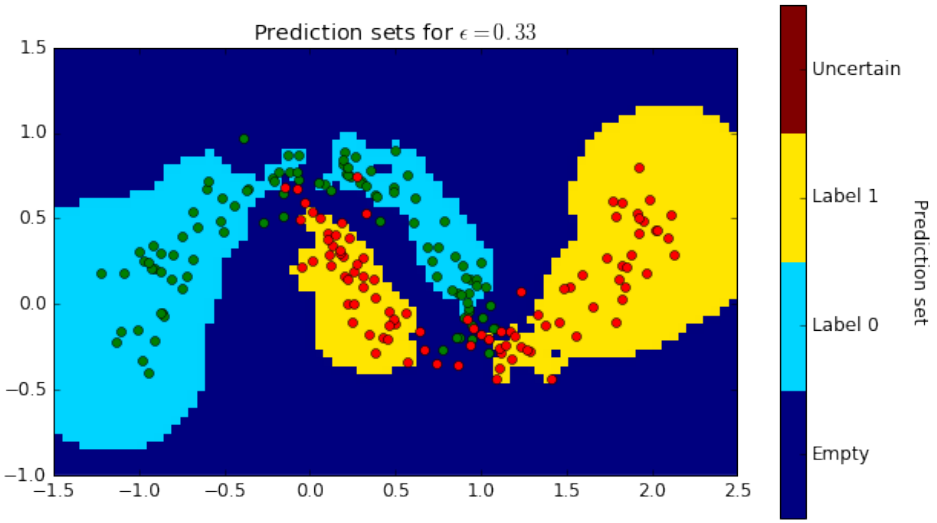


- Just a toy example with binary classification.
  - Two classes: 0 (green) and 1 (red)
  - 200 training examples
- kNN with $k = 3$ as underlying.
- At the point indicated with the black cross
  - $p_0 = 0.56$
  - $p_1 = 0.01$

An example on synthetic data

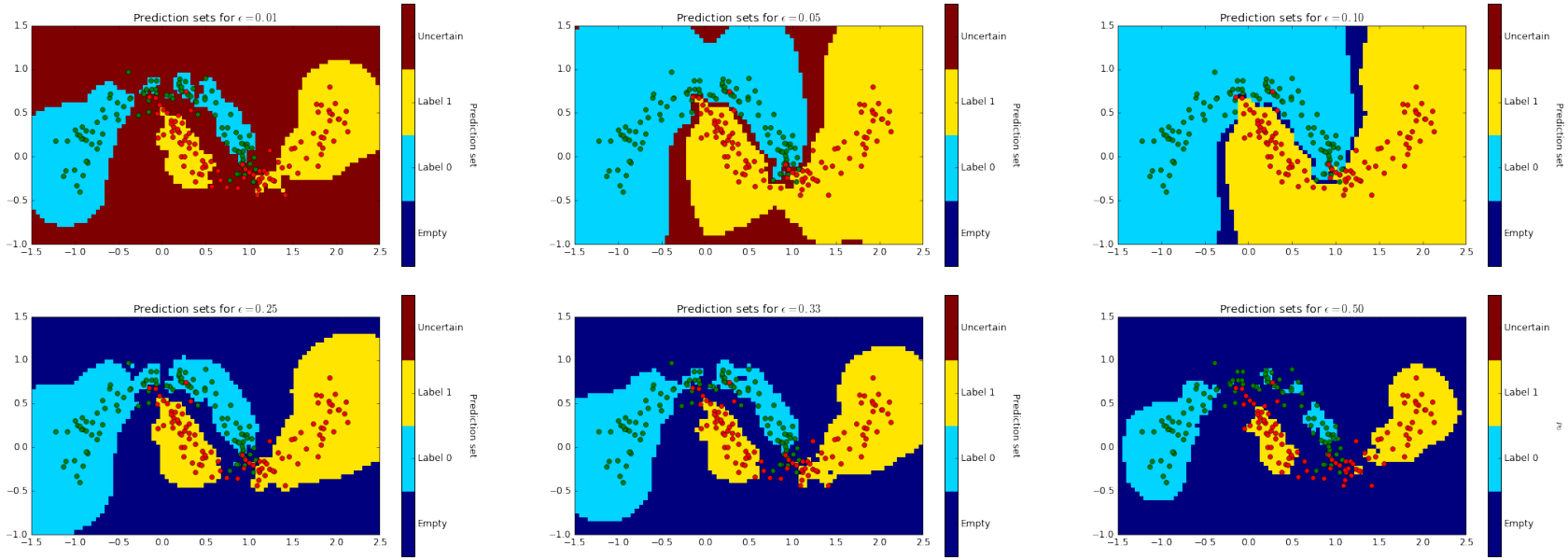






	Prediction set
$p_0 \leq \epsilon$, $p_1 \leq \epsilon$	$\emptyset$
$p_0 > \epsilon$, $p_1 \leq \epsilon$	$\{0\}$
$p_0 \leq \epsilon$, $p_1 > \epsilon$	$\{1\}$
$p_0 > \epsilon$, $p_1 > \epsilon$	$\{0, 1\}$ Uncertain

An example on synthetic data



- For low ϵ (error rate), the predictions are often hedged (brown areas). Both p values are larger than ϵ, hence both labels are in the prediction set.

An example on synthetic data

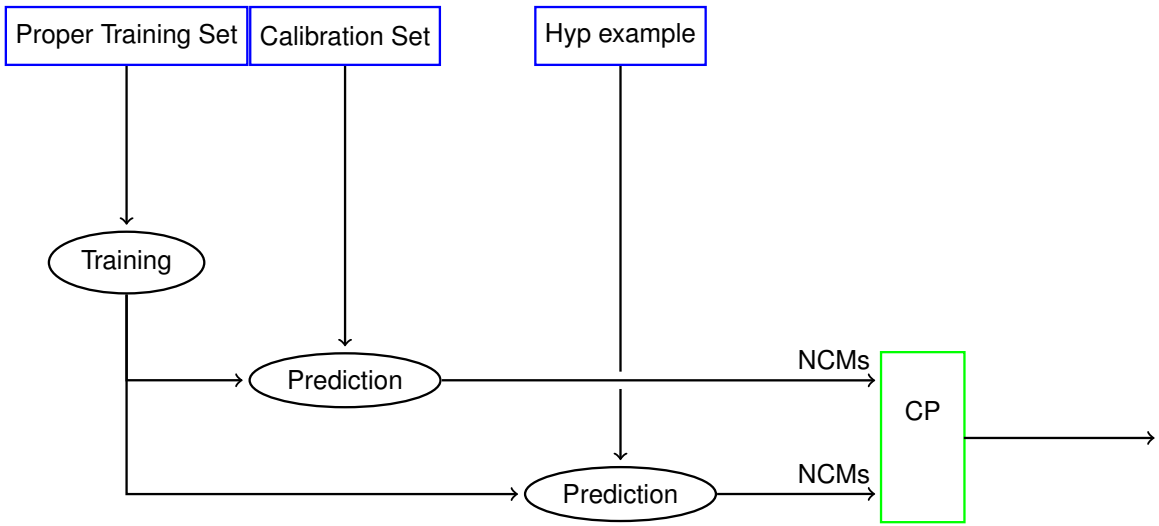
- Predictions on 1,000 test examples from the same distribution for varying significance level ϵ
- The error rate should be less than or equal to ϵ (barring statistical fluctuation)

ϵ	Positive predicted Positive	Positive predicted Negative	Negative predicted Negative	Negative predicted Positive	Empty preds	Uncertain preds	Error rate
0.01	344	3	345	2	0	306	0.005
0.05	427	33	462	15	0	63	0.048
0.10	440	39	464	21	36	0	0.096
0.15	415	25	455	9	96	0	0.130
0.20	398	13	430	6	153	0	0.172
0.25	393	8	398	3	198	0	0.209
0.50	275	2	265	0	458	0	0.460
0.75	138	0	127	0	735	0	0.735
0.80	121	0	111	0	768	0	0.768
0.85	88	0	90	0	822	0	0.822
0.90	53	0	59	0	888	0	0.888
0.95	28	0	28	0	944	0	0.944
0.99	4	0	4	0	992	0	0.992

Inductive Conformal Predictors

- The definition of CP discussed so far is referred to as "**Transductive** Conformal Prediction".
  - Excursus: Induction vs. Transduction (back-up slides)
  - The key aspect is that we use also the test object.
- A less compute-intensive form of CP is possible: **Inductive Conformal Prediction**.
- ICP is actually the recommended form of CP when there is more data available than needed to train the underlying with adequate performance.
- ICP too has the validity property.

Inductive Conformal Predictors



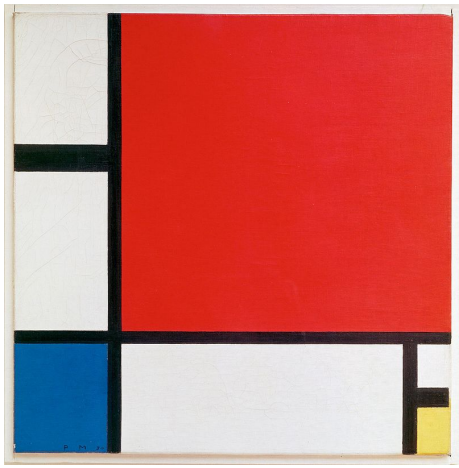
- The training set is split into *proper training set* and *calibration set*.
- The underlying algorithm is trained on the proper training set (once).
- The α_i are calculated applying the algorithm on the calibration set and the test object.
- Drawback: some data has to be used for calibration, rather than training

Mondrian (class-conditional) CP

- The validity guarantee of CP is not per-class.
  - The error rate for objects of one class might be higher than the target, but it may be compensated by a lower error rate for the other class: the guarantee is over **all** the classes.
  - It is a problem for **imbalanced** data sets, in particular.
- Class-conditional (Mondrian) CPs provide per-class validity guarantee.

Mondrian Conformal Predictors

Mondrian Conformal Predictors partition all examples (x_n, y_n) into categories and set a separate significance level ϵ_k for each category.



Called Mondrian because the categories resemble Mondrian paintings

"Composition II in Red, Blue, and Yellow", 1930 by Piet Mondrian (1872-1944)

Here we shall consider the so-called *label-conditional Conformal Predictors* where $k(n, x_n, y_n) = y_n$.

The fundamental advantage of a Mondrian Conformal Predictor is that the *validity* property holds separately on each category (label).

p-values for Mondrian CP

- The only change is in how p-values are computed.
- For the label-conditional CP the *p*-values are:

$$p(y) = \frac{|\{i = 1, \ldots, (\ell + 1) : y_i = y, \alpha_i \geq \alpha_{\ell+1}\}|}{|\{i = 1, \ldots, (\ell + 1) : y_i = y\}|}$$

- The difference with respect to the earlier definition of *p*-value is that the comparisons are restricted to the α_i associated with training examples with the **same** label as the hypothetical completion.

Using CP for point predictions

- In alternative to predictions sets, one can use CP to output a point prediction and hedge it with confidence and credibility.
- prediction: $\hat{y} = \arg\max_y p_y$
- confidence: $\{1 - \epsilon : |\Gamma^\epsilon| \leq 1\}$
- credibility: $\inf\{\epsilon : |\Gamma^\epsilon| = 0\}$
- In words, the confidence is 1 minus the 2^{nd} largest p-value, and the credibility is the largest p-value.
- Note: while we have validity guarantees for prediction sets, there is no guarantee on point predictions, confidence, and credibility.

Uses of CP

In addition to providing **valid** hedged predictions, CP can be used for:

- Ranking: order test objects e.g. by lowest p_0.
- Anomaly detection: declare an anomaly when you can reject all labels at the chosen significance level.
- Ensembling: p-values provide a common scale across disparate ML algorithms. p-value combination methods from classical Statistical Hypothesis Testing can be used to ensemble predictions.

Final considerations

- Conformal Predictors allow to produce predictions with a **chosen error rate** (significance level ϵ).
- The price to pay is multi-value predictions.
- CP is a flexible framework that can be applied to most ML algorithms.
- Inductive CP allows to scale the framework to large data sets.
- Mondrian CP provides per-class validity guarantees, essential for imbalanced data sets.

Calibrated Probabilistic Predictions: Venn Predictors

Confidence vs. Probability

- Confidence prediction: **p-values**

  "What is the probability of drawing an example that is as or more contrary than the test example to the hypothesis that it comes from the same distribution as the training set?"

- People generally think of the more direct question:

  "What is the probability of the label of the test object being L given the training set?"

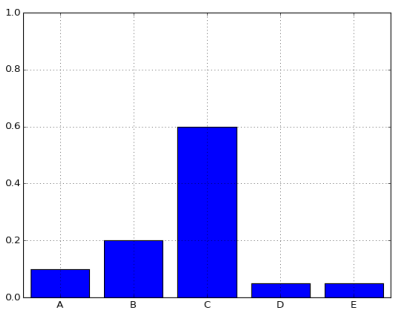
- **Probabilistic Prediction** answers this question

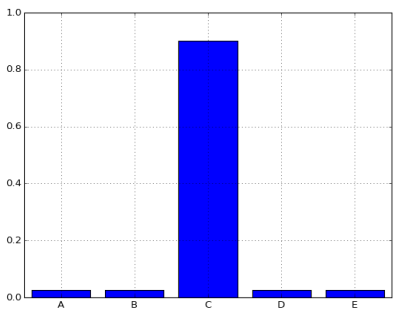
Why Probabilistic Prediction: a simple example

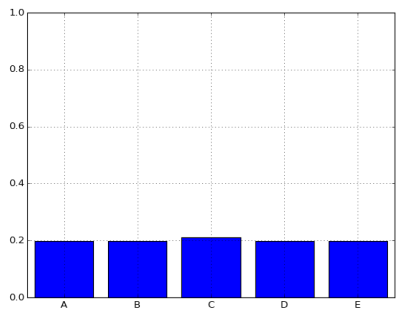
Suppose that, out of five possible labels A, B, C, D, and E, a classifier outputs C as prediction for an object x

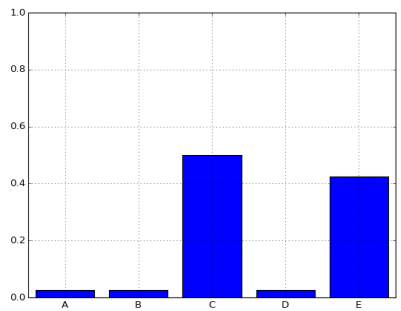
Generally, this means that C is the most probable label for x, given the training data.

But this in itself does not tell us much!









In the 4 distributions above, C is the most probable label.

But the way we would act based on them might be very different!

Probabilistic Prediction

- We want to predict the posterior conditional probability (density) of the label, given the training set and the test object.

$$\mathbb{P}[Y = y | X = \mathbf{x}]$$

- There are several different approaches:
  - Bayesian Machine Learning
    - Requires a “prior”
  - Kernel-based Density Estimation (Parzen-Rosenblatt)
    - Limitation: curse of dimensionality
  - “Classical Statistics” methods
    - e.g. From ECDF, as regularized solution of Fredholm integral equations (Vapnik)
  - ...the approach of this tutorial

Our setting and goal

- Learning under unconstrained randomness:
  - we know the space of examples $(\mathbf{x}, y) \in \mathbf{X} \times \mathbf{Y}$
  - we know the examples are drawn independently from the same distribution Q
  - at the outset we know nothing about the distribution Q.
- We want the predictor to be **valid**
  - Validity is the property of a predictor that outputs probability distributions that perform well against statistical tests based on subsequent observation of the labels.
  - In particular, we are interested in **calibration**:

$$\mathbb{P}[Y = y \mid P_y = p] = p \qquad \text{a.s.}$$

- It can be proved that validity cannot be achieved for probabilistic prediction in a general sense.

Venn Predictors

- Venn Predictors are a form of multi-probabilistic predictors for which we can prove **validity** properties
- The impossibility result mentioned earlier is circumvented in two ways:
  - We output multiple probabilities for each label, one of which is the valid one.
  - We restrict the statistical test for validity to calibration, i.e. the property that probabilities are matched by observed frequencies (for example, a particular label should occur in about 25% of the instances in which we give it a probability of 0.25)

Venn Predictors

- We start from the usual method
  - divide the training set objects into categories.
  - use some method to classify the test object into one of the categories.
  - use the frequencies of labels in the category of the test object as predicted probabilities for the object's label.
- We introduce some key differences
  - We divide *examples* rather than objects into categories.
  - We create a test example from the test object by assigning a hypothetical label.
  - When we compute the frequencies of labels in the category containing the test example, we *include* the test example itself.
  - We repeat the category assignment and label frequency calculation, for each possible label value.

Venn Predictors in more formal terms

- Usual setting:
  - training examples $z_i = (x_i, y_i)$ forming a bag $\langle z_1, \ldots, z_\ell \rangle$
  - test object $x_{\ell+1}$
- Venn Predictor output:
  - For each possible label value $y \in \mathbf{Y}$:
    - a probability distribution on $|\mathbf{Y}|$
- NOTE: not a single probability for a label, but a probability distribution on the set of the labels. One of these probabilities is the calibrated one

Venn Predictors: some detail

- To create a Venn Predictor, one starts by defining a **Venn Taxonomy**.
  - Intuitively, the Venn Taxonomy is used to group examples that we consider sufficiently similar for the purposes of estimating label probabilities.
    More precisely, the taxonomy is a partition of the space $Z^\ell \times Z$
  - Let denote with $A(\{z_1, \ldots, z_\ell\}, z)$ the element of the partition that contains $(\{z_1, \ldots, z_\ell\}, z)$
  - Two examples z_i and z_j belong to the same category iff:

$$A(\{z_1, \ldots, z_{\ell+1}\} / z_i, z_i) = A(\{z_1, \ldots, z_{\ell+1}\} / z_j, z_j)$$

  - Example: Taxonomy based on Nearest Neighbour
    two examples are assigned to the same category if their nearest neighbours have the same label

Why Venn?

- John Venn [1834-1923] was a logician and philosopher.
- In 1866 he published *The Logic of Chance*, in which he laid the foundations for the frequentist notion of probability.
- He's credited to have been the first to formulate explicitly and study the **reference class problem**.

  "It is obvious that every individual thing or event have an indefinite number of properties or attributes observable in it, and might therefore be considered as belonging to an indefinite number of different classes of things"
- Which class do we take when calculating relative frequencies?
- Venn also thought that "the more special the statistics, the better". But this leads to a dilemma: the more specific the class is, the fewer the element in the class.
- The same dilemma applies when designing a taxonomy.

Venn Predictors: some detail

- Given the test object $x_{\ell+1}$
- For every possible value y of the label:
  - We form the bag $\wr z_1, \ldots, z_{\ell+1} \int$, with the hypothetical example $z_{\ell+1} = (x_{\ell+1}, y)$
  - Identify the category T to which the example $(x_{\ell+1}, y)$ belongs[2].
  - The empirical probability distribution p_y of the labels in category T is obtained as:

$$p_y(y') := \frac{|\{(x^*, y^*) \in T : y^* = y'\}|}{|T|}$$

  - In words: for every possible value y' of the label, we calculate the fraction of examples in category T that have label y'

[2] This is done by resorting to an underlying ML algorithm

One comment on calibration

- Calibration is desirable, but it is not the only property we seek in predictions
- Example: weather prediction
  - If we were asked to predict with what probability it will rain tomorrow, we can simply always respond with the long-term average probability of rain.
  - It would be a calibrated prediction, but it is hardly useful.
  - What we want to have a more **specific** prediction.
- Venn Predictors offer a theoretically-backed framework in which we no longer have to worry about calibration; we can focus only on making predictions more specific.

Venn Predictors: Nearest Neighbour example

- Let's create a taxonomy with this rule:
  "two examples are assigned to the same category if their nearest neighbours have the same label"

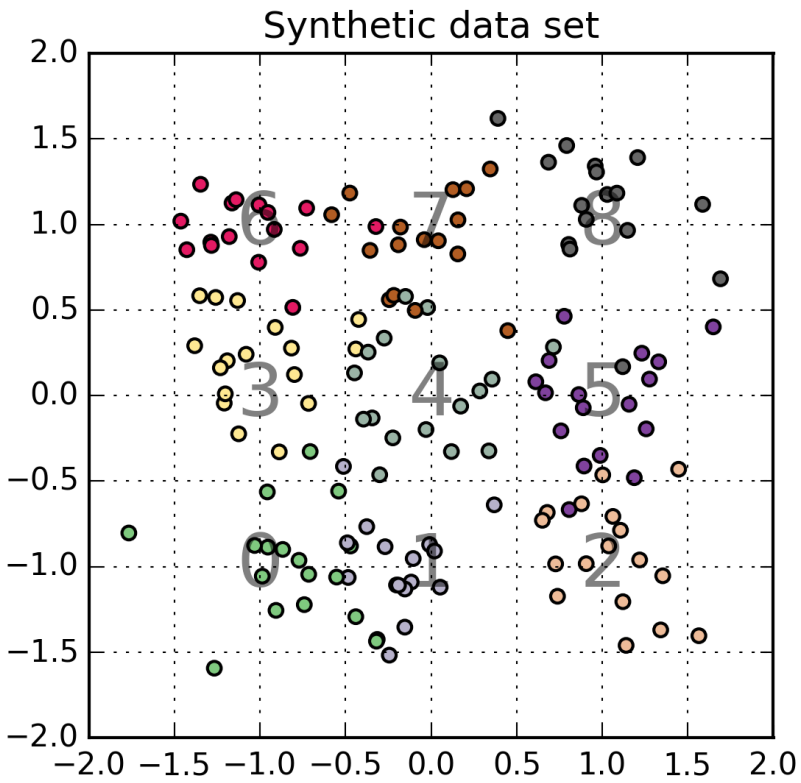
- Given a test object x_i
  - For each possible label value y
    - we create an example (x_i, y)
    - Identify the category T to which the hypothetical example (x_i, y) belongs.

      i.e. in this example find the label y_{NB} of the nearest neighbour of (x_i, y)
    - We compute the empirical probability distribution p_y of the labels in category T

      i.e. find all examples that have nearest neighbour with label y_{NB}; then for each possible label, count how many of those examples have that label; then normalize the counts to obtain relative frequencies.

- The result can be represented with a matrix, in which each row contains the probability distribution over the labels associated with the hypothetical label assignment.

Venn Predictors: Nearest Neighbour example



Venn Predictors: Nearest Neighbour example

Example: matrix for point (-1,-1)

	0	1	2	3	4	5	6	7	8
0	0.6470	0.2941	0.0000	0.0588	0.0000	0.0000	0.0000	0.0000	0.0000
1	0.5625	0.3750	0.0000	0.0625	0.0000	0.0000	0.0000	0.0000	0.0000
2	0.5625	0.3125	0.0625	0.0625	0.0000	0.0000	0.0000	0.0000	0.0000
3	0.5625	0.3125	0.0000	0.1250	0.0000	0.0000	0.0000	0.0000	0.0000
4	0.5625	0.3125	0.0000	0.0625	0.0625	0.0000	0.0000	0.0000	0.0000
5	0.5625	0.3125	0.0000	0.0625	0.0000	0.0625	0.0000	0.0000	0.0000
6	0.5625	0.3125	0.0000	0.0625	0.0000	0.0000	0.0625	0.0000	0.0000
7	0.5625	0.3125	0.0000	0.0625	0.0000	0.0000	0.0000	0.0625	0.0000
8	0.5625	0.3125	0.0000	0.0625	0.0000	0.0000	0.0000	0.0000	0.0625

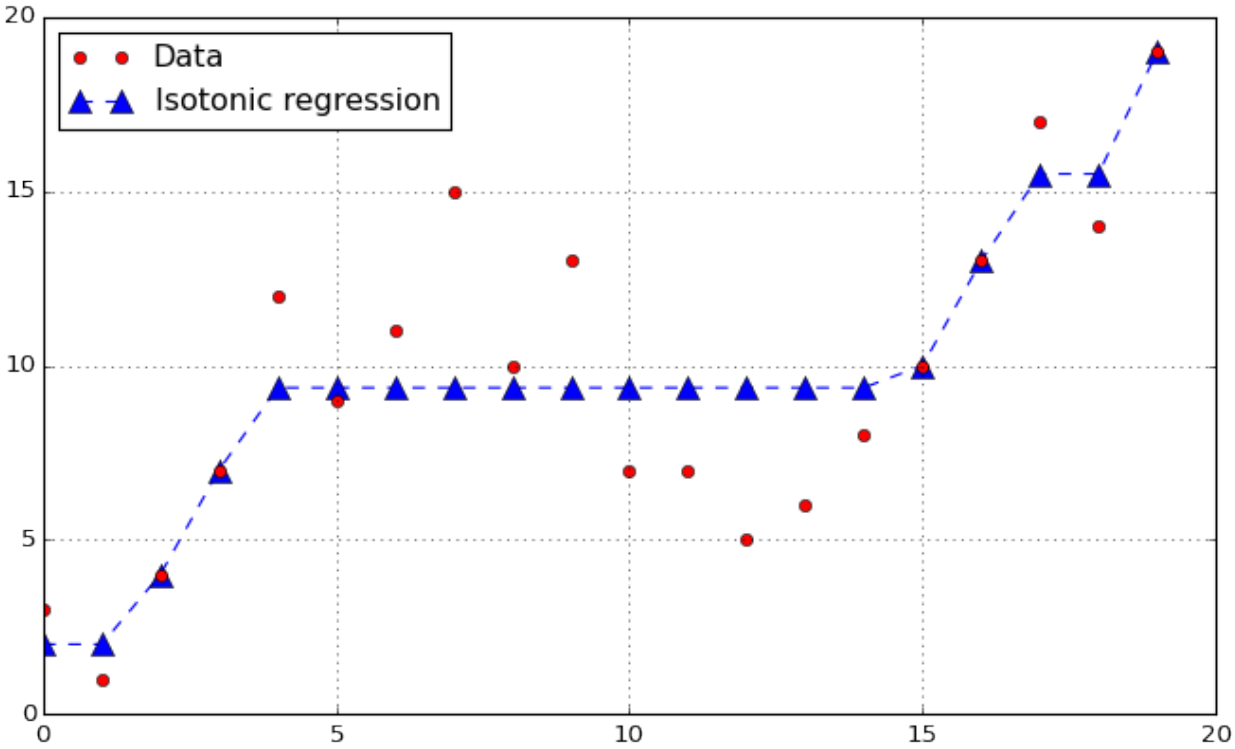
- Rows contain distributions
- Each row corresponds to a hypothetical assignment of a label to the test object

Venn-ABERS predictors

Calibration of scores into probabilities

- Let's restrict our attention to binary classification
- Many machine learning algorithms for classification are in fact *scoring classifiers*: they output a prediction score $s(x)$ and the prediction is obtained by comparing the score to a threshold.
- One could apply a function g to $s(x)$ to calibrate the scores so that $g(s(x))$ can be used as predicted probability.
  - Isotonic Regression: let's assume that $g()$ be an non-decreasing function.
  - Platt's scaling: let's fit a sigmoid

Isotonic Regression[3] example



Non-decreasing function that minimizes sum of square residues

[3] Monotonic: “one ordering”, either Isotonic (“order-preserving”) or Antitonic (“against the order”)

Isotonic calibrator

- The *isotonic calibrator* g for $((s(x_1), y_1), (s(x_2), y_2), \ldots, (s(x_\ell), y_\ell))$ is the non-decreasing function on $s(x_1), s(x_2), \ldots, s(x_\ell)$ that maximizes the likelihood

$$\prod_{i=1,2,\ldots,\ell} p_i$$

where:

$$p_i = \begin{cases} g(s(x_i)) & \text{if } y_i = 1 \\ 1 - g(s(x_i)) & \text{if } y_i = 0 \end{cases}$$

- The isotonic calibrator can be found as isotonic regression on $(s(x_1), y_1), (s(x_2), y_2), \ldots, (s(x_\ell), y_\ell))$.

Isotonic regression and Venn prediction

- Isotonic Regression is piecewise-constant.
- In each interval of s in which $g(s)$ is constant, the IR takes the average of the values of the training points in that interval[4]
- So, when the labels are encoded as 0 and 1, the value of the IR is the relative frequency of label 1 in the interval.
- It's what we need to use it as Venn predictor!
  - The categories of the Venn taxonomy are the intervals over which the IR is constant.

[4] theorem by Ayer, Brunk, Ewing, Reid, Silverman (1954)

Venn-ABERS predictors

- Let $s_0(x)$ be the scoring function for $(z_1, z_2, \ldots, z_\ell, (x, 0))$,
  $s_1(x)$ be the scoring function for $(z_1, z_2, \ldots, z_\ell, (x, 1))$,
  $g_0(x)$ be the isotonic calibrator for

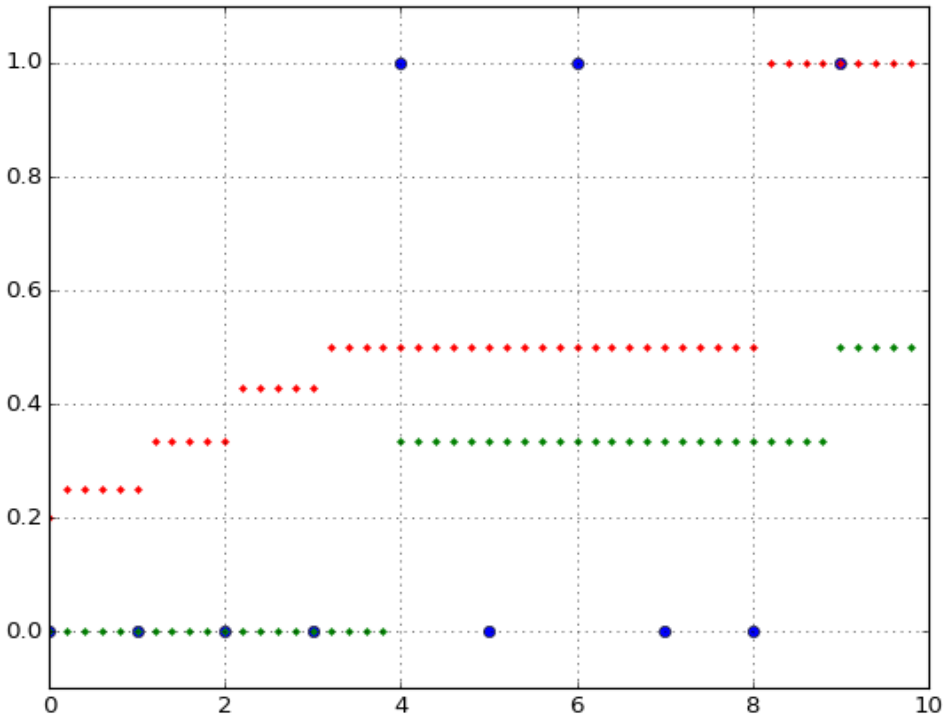
$$((s_0(x_1), y_1), (s_0(x_2), y_2), \ldots, (s_0(x_\ell), y_\ell), (s_0(x), 0))$$

  and $g_1(x)$ be the isotonic calibrator for

$$((s_1(x_1), y_1), (s_1(x_2), y_2), \ldots, (s_1(x_\ell), y_\ell), (s_1(x), 1))$$

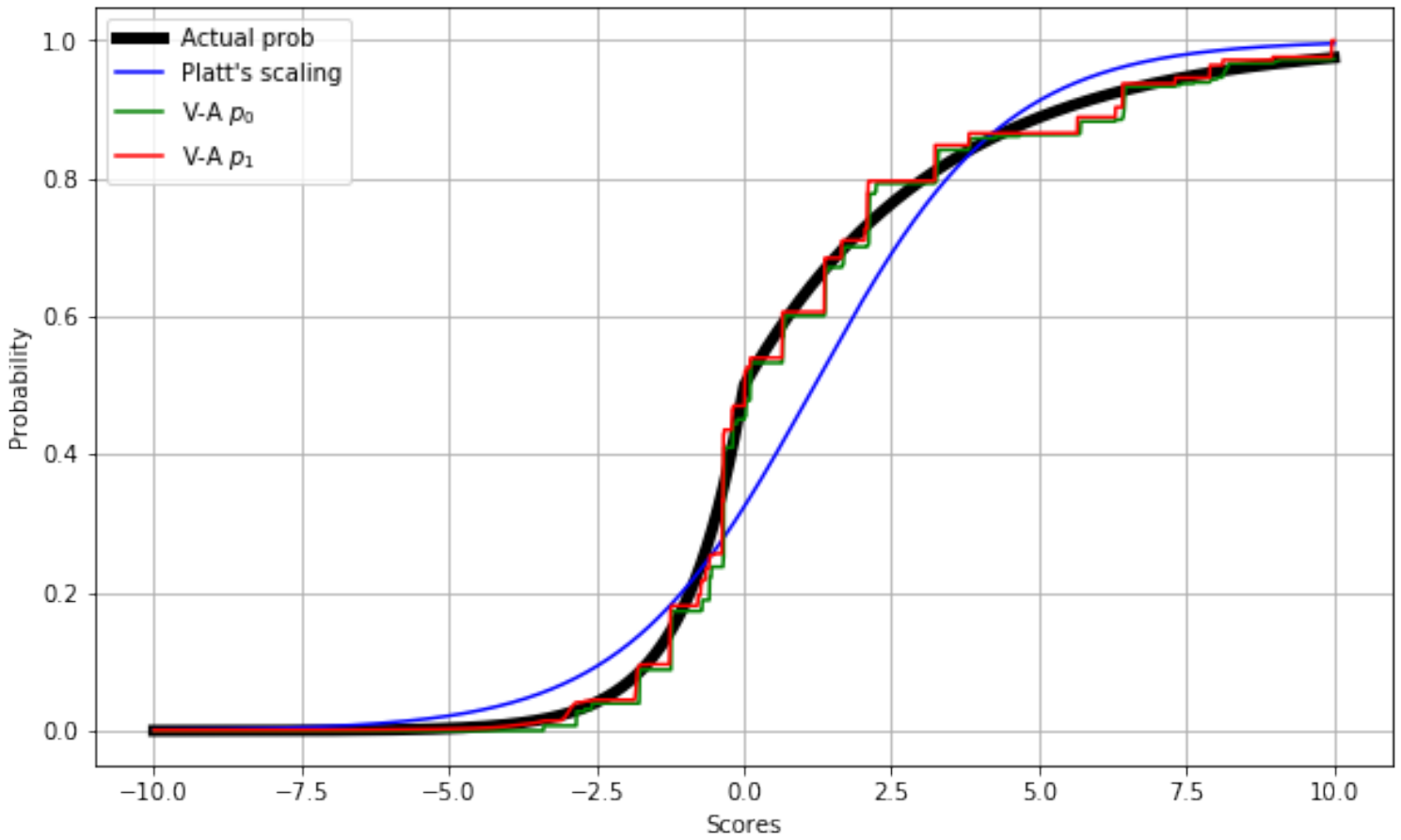
- The multiprobabilistic prediction output by the Venn-ABERS predictor is: (p_0, p_1), where $p_0 := g_0(s_0(x))$ and $p_1 := g_1(s_1(x))$

Toy example



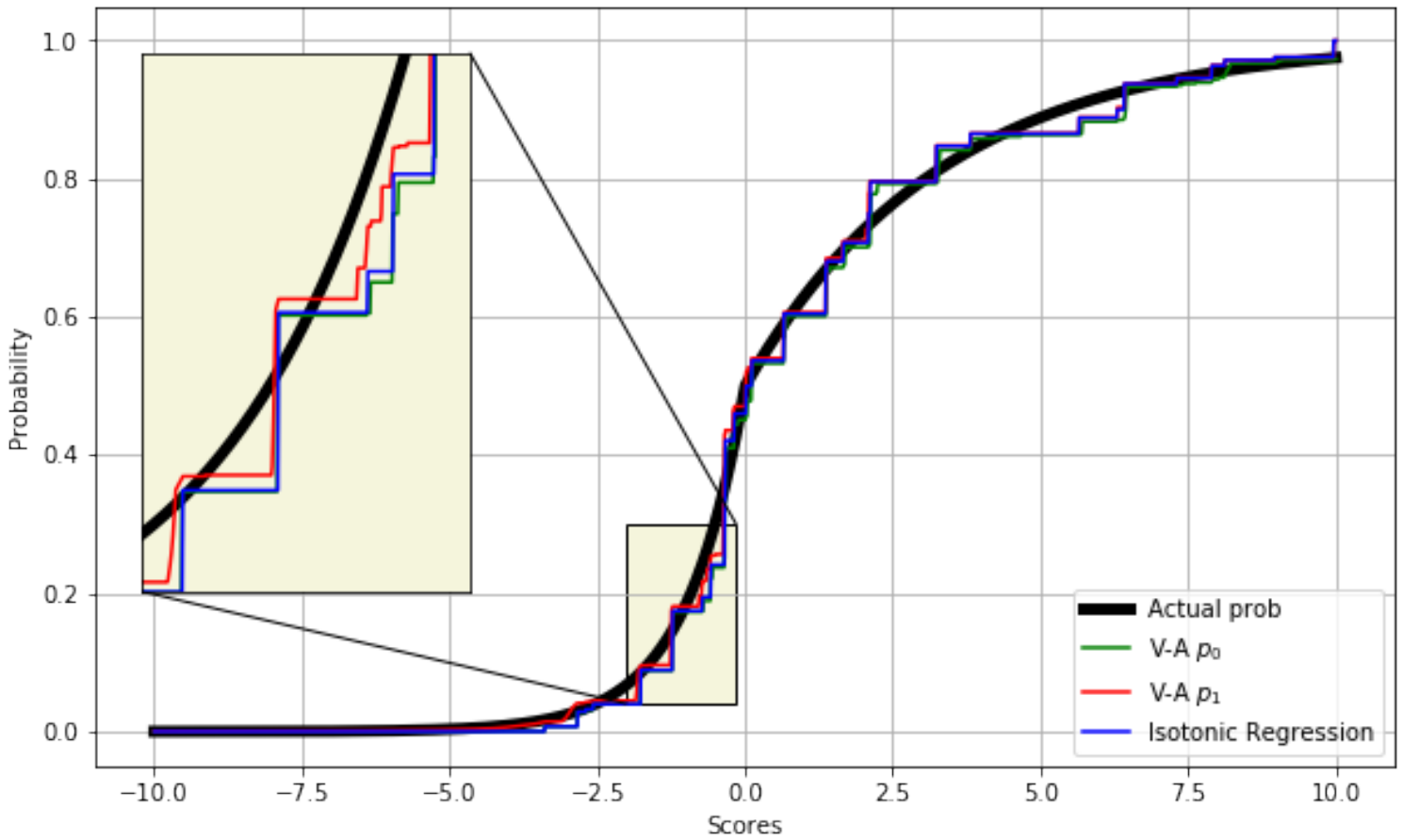
- Example of the two Isotonic Regressions in Venn Prediction
  - Blue dots: data (+1 or 0)
  - green dots: $g_0(s)$, red dots: $g_1(s)$

Synthetic data: V-A vs. Platt's



- Data set with deliberate departure from sigmoid
  - Venn-ABERS calibrator manages to recover actual score-probability relationship
  - Platt's scaling is not as accurate

Synthetic data: V-A vs. Isotonic Regression



- IR too recovers the score-probability relationship
  - However, the probability estimates are not as fine-grained
  - IR: 31 different probability levels
  - Venn-ABERS: 211 for p_0, 1823 for p_1

Further observations

- Calibration via Isotonic regression is known to "overfit"
- Venn-ABERS predictors lessen this tendency to overfit and inherit the validity guarantee of Venn predictors.
- Compared with bare Isotonic Regression, the multi-probabilistic output also provides an indication of the reliability of the probability estimates.
  - If the probabilities differ, this can be taken as an indication of the uncertainty on the probability estimate itself.
- Compared with Platt's Scaling (fitting a sigmoid as calibrator), Venn-ABERS predictors do not make any assumption on the shape (functional form) of the calibrator.

Single value prediction

- Can we deal with a single-valued probability instead?
- One approach is the following:
  - Assume a loss function $L(y, p)$ and minimize the expected loss with respect to it
  - E.g. log loss:

$$\begin{cases} -\log p & \text{if } y = 1 \\ -\log(1-p) & \text{if } y = 0 \end{cases}$$

  - The optimal p for log loss is

$$p = \frac{p_1}{1 - p_0 + p_1}$$

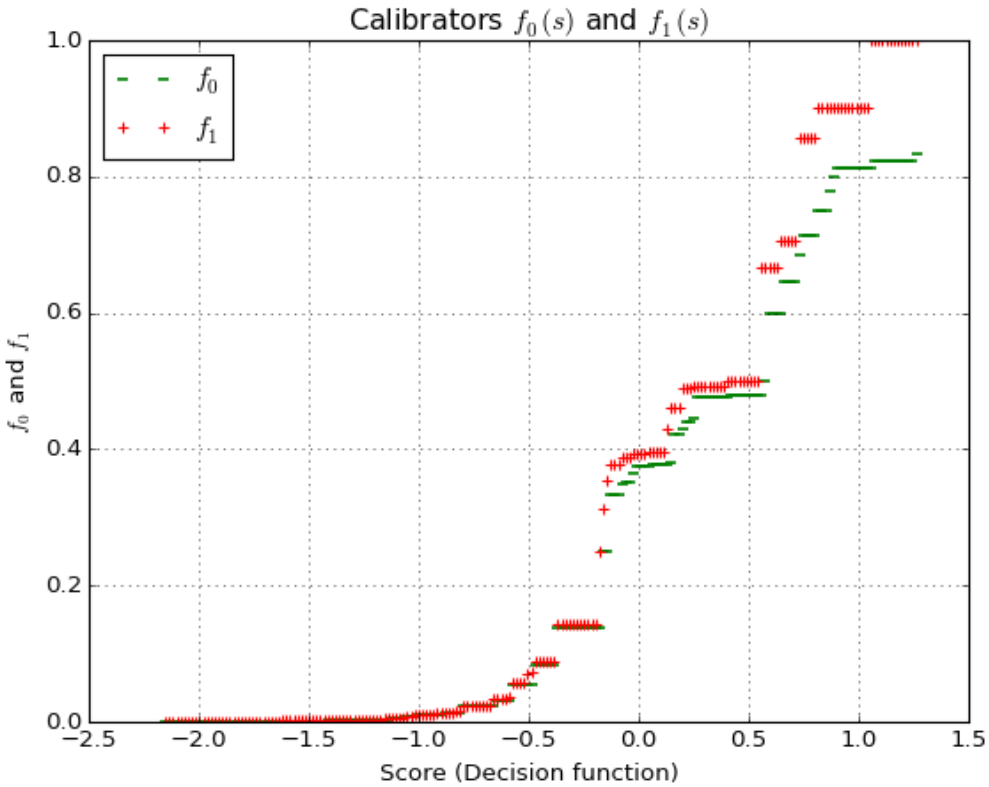
  - There is also an optimal p for Brier score (aka RMSE).

Synthetic data example: Loss

- Losses on test data (average over 20 sets of 5000 test examples)

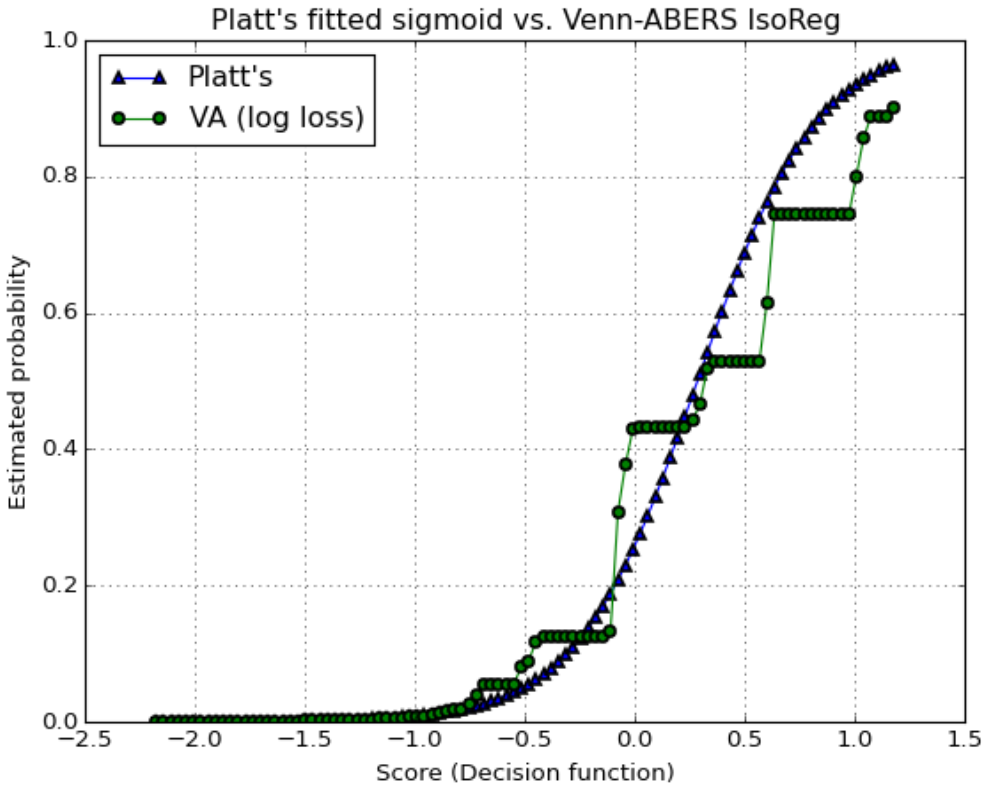
Probability estimate	Log loss	Brier
Isotonic Regression	0.380691	0.123435
Platt scaling	0.400399	0.131967
Venn-ABERS	0.368595	0.123379
VA p_0	0.375235	0.123389
VA p_1	0.375019	0.123461

Real-life example



- Venn-ABERS Calibrators for Compound Activity Prediction
  - Applied to SVM decision function
  - green dots: $g_0(s)$, red dots: $g_1(s)$
- Imbalanced data set (class 1 was $\approx 1\%$)

Platt scaling vs. Venn-ABERS



- Platt scaling vs. (log-loss) Venn-ABERS
  - Platt's scaling is possibly less accurate for high probs

Large-scale Venn-ABERS predictors

- Venn-ABERS appear much more computationally demanding than Isotonic Regression or Platt's Scaling.
  - For every evaluation, we have to retrain the underlying machine learning algorithm and recompute Isotonic Regressions
  - This would not scale to large data sets.
- Inductive Venn-ABERS Predictors
  - The training set is split into a proper training set and a calibration set.
  - The proper training set is used to train the underlying ML algorithm once.
  - The Isotonic Regression is calculated only on the calibration set.
  - This method retains the theoretical validity guarantee.

Large-scale Venn-ABERS predictors

- Inductive Venn-ABERS predictors are a step in the right direction but they still are prohibitive for large data sets.
  - For every evaluation, it seems we have to recompute 2 Isotonic Regressions on the calibration set.
- In actual fact, it can be computed very efficiently
  - It is possible to exploit the fact that only one data point is added to an otherwise fixed calibration set
  - Most computation occurs once for $g_0()$ and once for $g_1()$.
  - The evaluation requires only a binary search in a pre-computed data structure

Conclusions

- Complementing a prediction with its probability can enable better decision making
- Venn Predictors are (multi)probabilistic predictors with validity guarantee
- Venn-ABERS Predictors are Venn Predictors that can be applied on top of a Scoring Classifier
- VAP do not assume a functional form (e.g. sigmoid) for the relationship between score and probability

Acknowledgements

- Many thanks to Prof. Alexander Gammerman, Prof. Vladimir Vovk, Prof. Zhiyuan Luo, and Dr. Ilia Nouretdinov for useful advice and insightful discussions.
- This work was made possible by the AstraZeneca grant "Machine Learning for Chemical Synthesis" (R10911) and by the ExCAPE H2020 project.

References

- Vladimir Vovk, Alex Gammerman, and Glenn Shafer.
  Algorithmic Learning in a Random World.
  Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005.

- Vladimir Vovk, Ivan Petej.
  Venn-Abers Predictors
  In *Proceedings of 30th Conference on Uncertainty in Artificial Intelligence*, 2014
  `http://alrw.net/articles/07.pdf`

- Vladimir Vovk, Ivan Petej, and Valentina Fedorova.
  Large-scale probabilistic predictors with and without guarantees of validity.
  Technical Report `arXiv:1511.00213` [cs.LG], `arXiv.org` e-Print archive, November 2015.

- Python implementation at: `https://github.com/ptocca/VennABERS`

Back-up material

Ranking example

Biological activity of chemical compounds (threshold: 3.81%). The table on the left is order by lowest $p_{inactive}$, the one the right by highest p_{active}.

Rank	Compound tag	Viability	$p_{inactive}$
1	79813	1.76	3.483e-10
2	129543	4.57	9.419e-10
3	115173	1.48	1.593e-09
4	108813	15.69	2.372e-09
5	100523	0.85	4.316e-09
6	116614	39.05	2.161e-08
7	94529	3.57	2.312e-08
8	104764	1.47	3.455e-08
9	62991	25.27	4.058e-08
10	64246	4.44	4.743e-08
11	84878	1.77	4.755e-08
12	127825	1.67	5.238e-08
13	52454	2.95	5.885e-08
14	74599	3.84	6.941e-08
15	75236	74.03	9.263e-08
16	91399	2.05	1.138e-07
17	121411	1.69	1.929e-07
18	6106	2.27	2.118e-07
19	104197	1.78	2.127e-07
20	12551	1.08	2.363e-07
21	85895	2.03	2.412e-07
22	128112	1.96	2.579e-07
23	96373	1.16	2.599e-07
24	74016	2.37	2.820e-07
25	130880	3.36	3.077e-07

Rank	Compound tag	Viability	p_{active}
1	115173	1.48	1.000
2	116614	39.05	1.000
3	129543	4.57	1.000
4	79813	1.76	1.000
5	100523	0.85	0.998
6	108813	15.69	0.998
7	94529	3.57	0.997
8	62991	25.27	0.994
9	64246	4.44	0.992
10	84878	1.77	0.990
11	104764	1.47	0.988
12	127825	1.67	0.985
13	52454	2.95	0.984
14	74599	3.84	0.982
15	75236	74.03	0.978
16	115494	83.84	0.977
17	121411	1.69	0.977
18	91399	2.05	0.977
19	119648	80.08	0.973
20	128112	1.96	0.964
21	85895	2.03	0.961
22	129514	50.91	0.960
23	130880	3.36	0.958
24	6106	2.27	0.958
25	104197	1.78	0.957

From Toccaceli et al. *Combination of Conformal Predictors for Classification*, COPA, 2017

Combination example

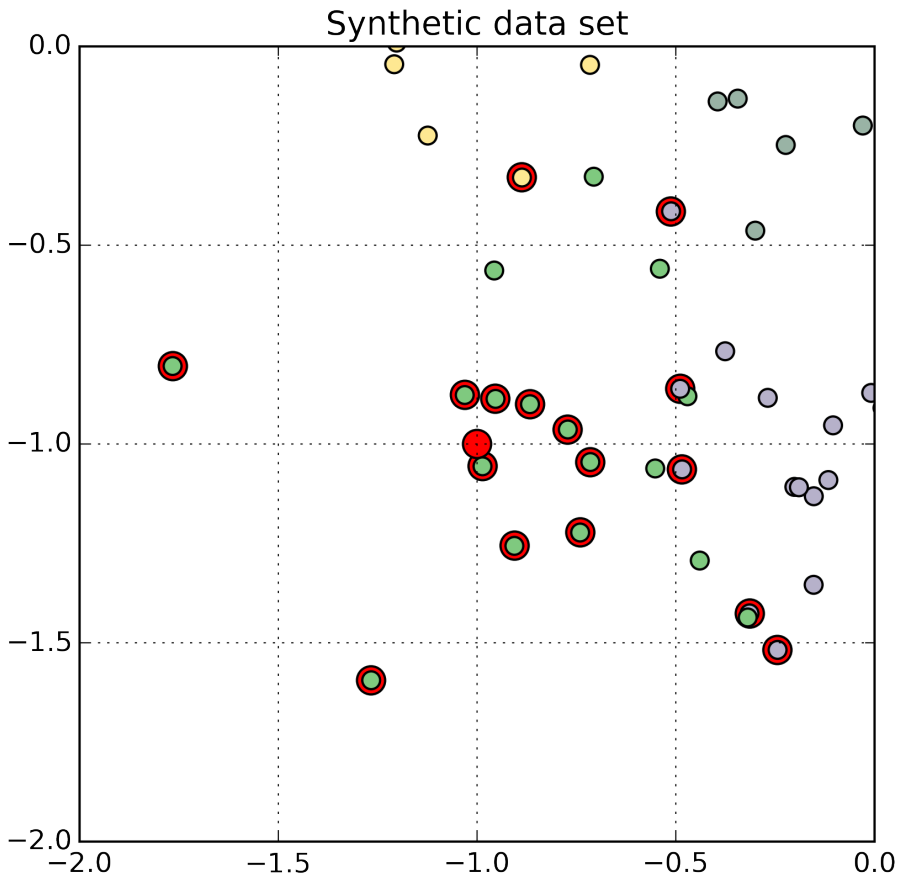
p-value combination of SVC, XGB, kNN. F1 score for precise predictions for various significance levels (averages over 50 runs). The best values are highlighted in bold.

	F_1 for the Active class				F_1 for the Inactive class			
epsilon	0.01	0.05	0.10	0.15	0.01	0.05	0.10	0.15
SVC	0.269	0.176	0.122	0.096	0.193	0.459	0.617	0.716
XGB	0.258	0.173	0.121	0.096	0.287	0.501	0.645	0.736
kNN	0.161	0.102	0.077	0.066	0.253	0.401	0.510	0.593
min	0.232	0.122	0.091	0.085	0.367	0.637	0.745	0.771
max	0.166	0.220	**0.209**	**0.177**	0.110	0.230	0.334	0.429
mean	0.198	0.235	0.187	0.142	0.159	0.317	0.477	0.603
Fisher	0.217	0.133	0.102	0.087	0.468	0.653	0.742	0.793
min ECDF	0.261	0.177	0.128	0.106	0.289	0.480	0.613	0.693
max ECDF	0.218	0.143	0.106	0.088	0.279	0.493	0.630	0.711
mean ECDF	0.236	0.163	0.120	0.097	0.294	0.514	0.656	0.743
Fisher ECDF	**0.280**	0.183	0.127	0.100	0.310	0.523	0.658	0.743
weighted soft	0.196	0.235	0.192	0.148	0.161	0.325	0.496	0.634
weighted hard	0.240	0.212	0.169	0.140	0.312	0.567	0.712	0.794
reduced soft	0.199	**0.239**	0.190	0.141	0.161	0.325	0.495	0.630
reduced hard	0.248	0.155	0.114	0.099	0.326	0.578	0.710	0.773
weighted soft ECDF	0.235	0.158	0.116	0.094	0.353	0.633	0.782	0.856
weighted hard ECDF	0.237	0.155	0.126	0.134	**0.643**	**0.839**	**0.897**	**0.896**
reduced soft ECDF	0.240	0.169	0.123	0.099	0.295	0.517	0.661	0.749
reduced hard ECDF	0.251	0.172	0.125	0.104	0.279	0.477	0.610	0.692

From Toccaceli et al. *Combination of Inductive Mondrian Conformal Predictors*, Machine Learning, 2018

Nearest Neighbour example

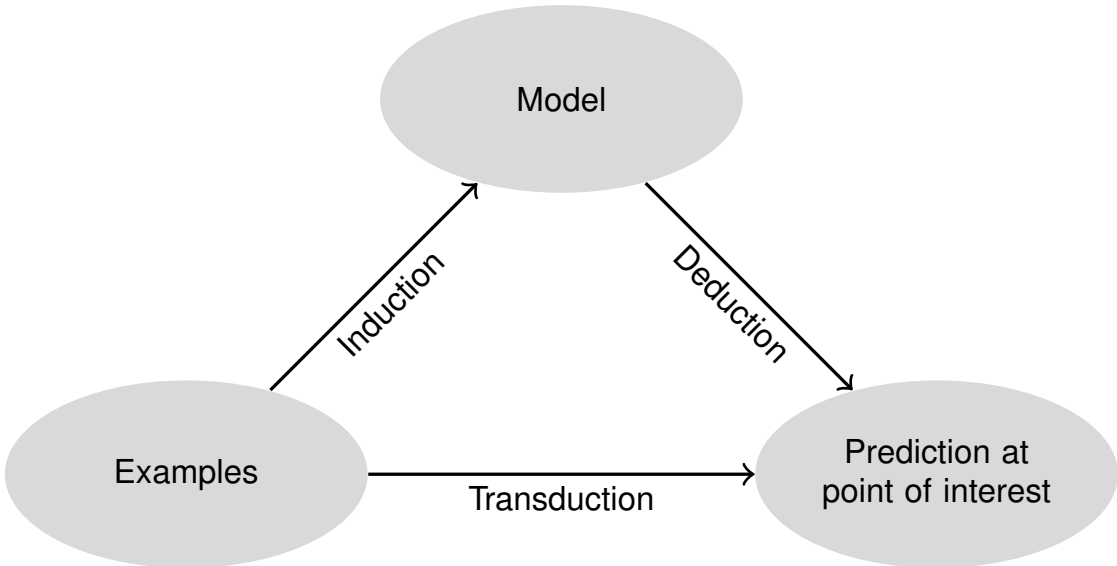
Zoom on the lower left quadrant



Test object at (-1,-1) with hypothetical label 0 (pale green, not drawn).
Large red dot denotes examples belonging to category 0.

Induction, Deduction,... Transduction! 1/2

- Classical philosophy usually considers two types of inference:
  - induction: from particular to general
  - deduction: from general to particular
- V. Vapnik proposed a new type[5,6]:
  - **transduction**: from particular to particular (*direct inference* instead of *generalization*)



[5] V. Vapnik, The nature of statistical learning theory, p.293, 2nd ed., Springer 1999

[6] V. Vapnik, Estimation of Dependencies Based on Empirical Data, Chap.3, , Springer 2006

Induction, Deduction,... Transduction! 2/2

- Rationale: "*If you are limited to a restricted amount of information, do not solve a particular problem by solving a more general problem*."
- The more general problem refers to creating a model that predicts well for all possible values; in other words, it refers to the problem of estimating a **function**.
- The transductive approach is instead to create a model that predicts well at the point(s) of interest, not everywhere; it estimate the **values of a function** at point(s) of interest.
- The implication is that at training time you also use the test objects.
- It can be of benefit when you have a (relatively) small amount of data.

Validity guarantees

P: exchangeable probability distribution on Z^∞
$\omega = (\mathbf{x}_1, y_1, \mathbf{x}_2, y_2, \dots)$ drawn from P

$$\mathrm{Err}_n^{(\epsilon)}(\Gamma, \omega) := \begin{cases} 1 & \text{if } y_n \notin \Gamma^\epsilon(\mathbf{x}_1, y_1, \dots, \mathbf{x}_n, y_n, \mathbf{x}_{n+1}) \\ 0 & \text{otherwise} \end{cases}$$

$\mathrm{Err}_n^{(\epsilon)}(\Gamma, P)$: random variable of which $\mathrm{Err}_n^{(\epsilon)}(\Gamma, \omega)$ is a realization

- Asymptotic conservative validity:

$$\mathbb{P}\left[\limsup_{n\to\infty} \frac{\mathrm{Err}_n^{(\epsilon)}(\Gamma, P)}{n} \leq \epsilon\right] = 1$$

- A stronger version applying the law of iterated logarithms:

$$\mathbb{P}\left[\limsup_{n\to\infty} \frac{\mathrm{Err}_n^{(\epsilon)}(\Gamma, \omega)}{\sqrt{2\epsilon(1-\epsilon)n \ln\ln n}} \leq 1\right] = 1$$

- Finite sample guarantee using Hoeffding's inequality:

$$\forall N > 0,\ \forall \delta > 0 \qquad \mathbb{P}\left[\omega : \mathrm{Err}_N^{(\epsilon)}(\Gamma, \omega) \geq (\epsilon + \delta)N\right] \leq e^{-2N\delta^2}$$