Exam Questions for the Machine Learning Part (L5) of "Introduction to Bioinformatics" (894.101)

Peter Sykacek
Machine Learning in Bioinformatics Research Group
BOKU University
peter.sykacek@boku.ac.at

November 22, 2012

The following exam questions concern the lecture "L5" in the "Introduction to Bioinformatics" (894.101) course. All references given in the sketched answers point to the respective slide numbers in the document bin_intro_L5_hdouts.pdf. For further references please consult my slides or for more information the books [1, 2, 3].

Note that you do not necessarily have to write assays. You may also sketch the answer with well chosen keywords.

1 What do you know about programming computers?

This is a question with two sub problems each worth half the total number of points for this question.

1.1 How does programming relate to instructions for humans and what are the differences?

1.1.1 Answer

A computer program works on "input data" to produce some desired "output data". This is equivalent to following instructions which are for example found in cooking recipes.
Example: produce beaten egg white
Ingredients (input): 1 chicken egg

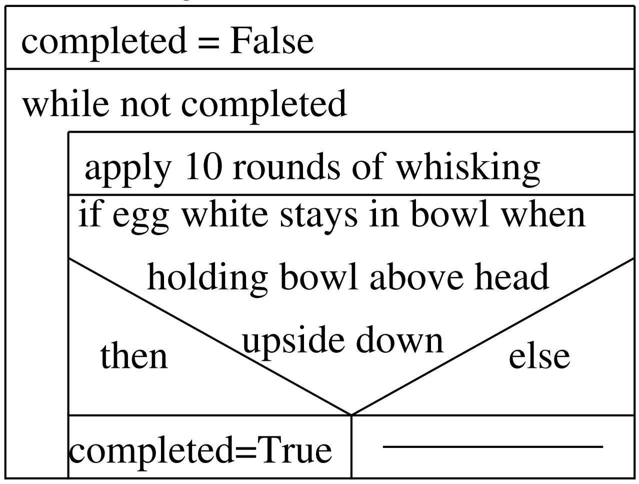
1. Separate egg white from yolk.

2. Use wire whisk to beat egg white until stiff.

3. Test of success: turn over mixing bowl and look from below into the bowl.

Result: (output): beaten egg white and one yolk.

In the context of programming computers, such instructions are for example represented as structograms.



The essential difference between a computer and a human following instructions is the human creativity and the human ability of applying common sense. Humans would not test the state of beaten egg white exactly as described in the structogram, because it is obvious that this would spill the contents of the bowl, as long as the egg white is not done. Computers on the other hand will do exactly as is specified in the program. This requires that programmers think carefully about all steps and the undesired side effects certain (often input data dependent) states can have. The difficulty of getting this right can be witnessed almost daily and manifests itself in crashing computer systems. A currently famous example are the recent problems in Bank Austria IT, which caused almost an entire month of problems with online banking.

1.1.2 Slide numbers

5,6 and 7.

1.2 Which programming paradigms do you know? Provide an example for at least two of three paradigms.

1.2.1 Answer

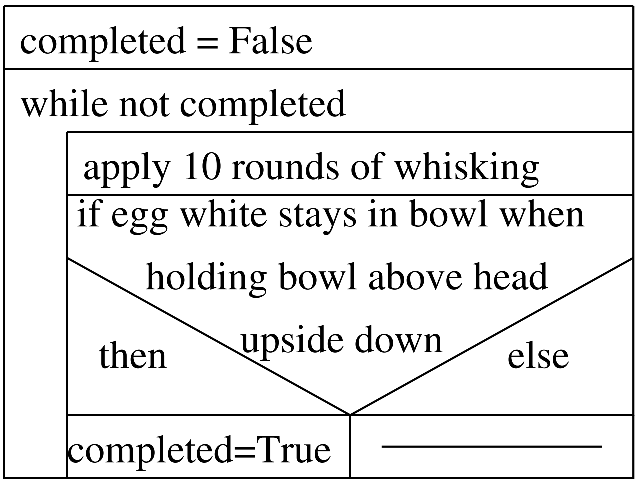
We know three programming paradigms.

1. Imperative programming: the solution is described using sequences of instructions, alternatives and loops as was used in the structogram for describing how to obtain beaten egg white.

2. Object oriented programming: solution and data are tied together. Objects are containers of data and “methods” which describe actions how to interact with the objects.

3. Functional programming: the solution is described by its properties and making use of induction.

1) Example of imperative programming:



2) Example of functional programming:
Expression of factorial of n (n is a natural number).

$$n! = \prod_{k=1}^{n} k \quad \text{(corresponds to} \;\; 1 * 2 * ... * n)$$

Factorial in Haskell (a purely functional language):

```
factorial :: Integer -> Integer -- type declaration
factorial 0 = 1                 -- pattern matching
factorial n = n * factorial (n - 1)
```

3) Example of object oriented programming:
Class for a linked list which allows keeping “words” ordered according to an “alphabet”. Note that the Node Class is missing here it can be found on slide 11 in your handouts.

```
class LinkList(object):
    def __init__(self):
        self.head=None
```

```
def add2list(self, strval):
    # first we generate a node
    newnode=Node(strval)
    # and now add it:
    if self.head:
        self.head = self.head.addnode(newnode)
    else:
        self.head = newnode
```

1.2.2 Slide numbers

8,9,10,11.

2 What do you know about probability theory?

This question consists of two parts each worth half the points of the overall question.

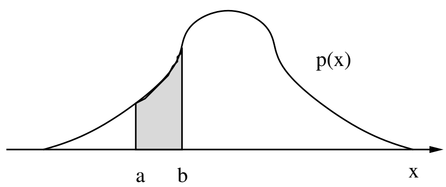
2.1 What is the difference between a variable and a random variable? Provide an example of a random variable.

2.1.1 Answer

Variables represent deterministic values.

Random variables represent "collections" of values. The density function (often $p(x)$) describes the relative occurrence of values. This is comparable to a sand heap which specifies the occurrence of grain positions.

Area of shaded region: probability observing sample in interval $P(x \in [a,b]) = \int_{x=a}^{b} p(x)dx$.



A dice is an example of a discrete random variable:

- When rolling a dice, we do not know which side will occur next.
- Occurrence of a particular outcome (e.g. rolling a 3) does not tell us anything about the next time we roll the dice.
- The dice as a random variable is fully described by the occurrence probability of the different sides.

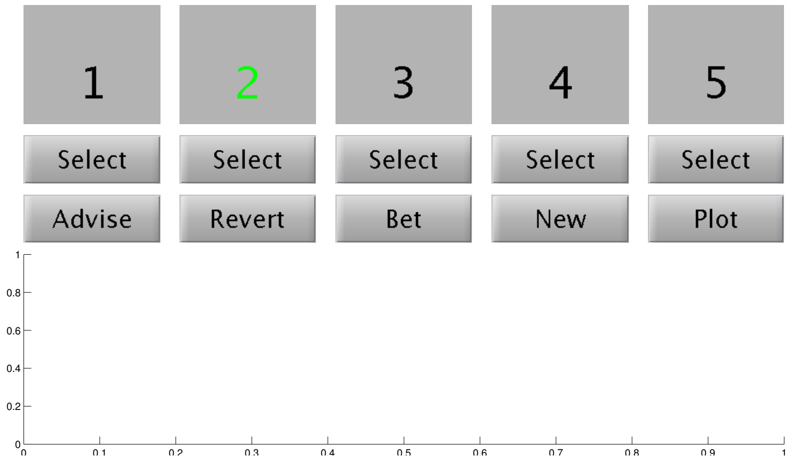
A fair dice would have $P_1 = P_2 = ...P_6 = 1/6$, other parametrisation are possible. Example for biased distribution in biology: distribution over amino acids where some might be preferred in a particular class of proteins.

2.1.2 Slide numbers

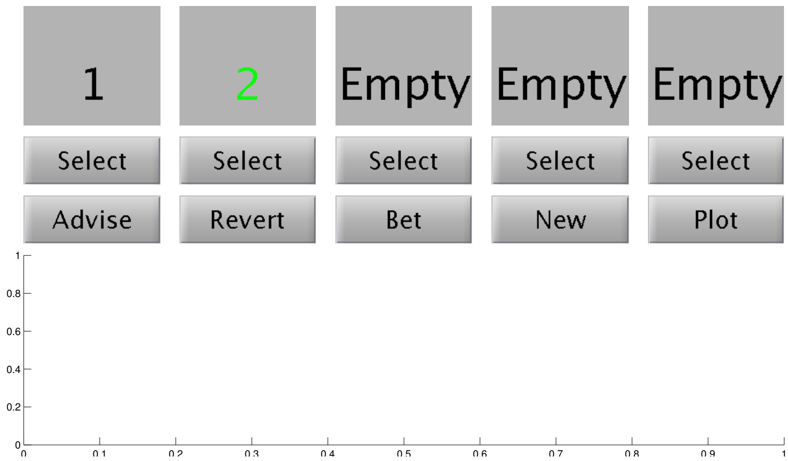
12, 13

2.2 Discuss the following quiz show game

A show master provides you the option of choosing one of five doors. Behind one door there is a prize all other doors lead to empty places.



After your choice (her door two), the show master reveals three bad choices that would have lead to "empty" boxes.



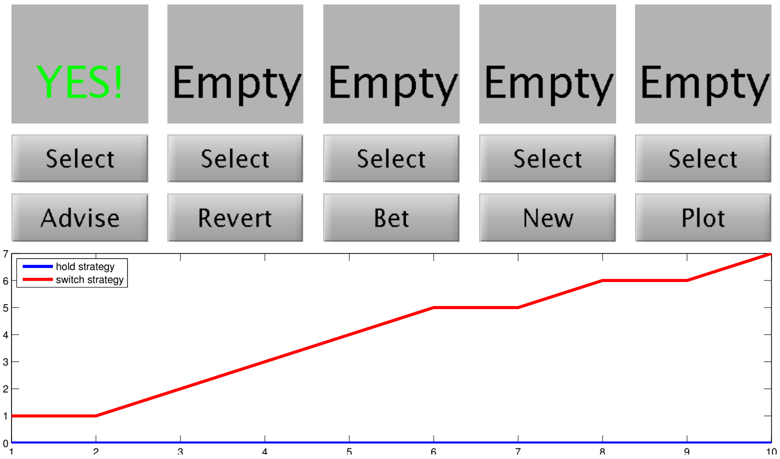
You have now two options: A) Keep selection B) Alter selection

What is the probability of winning the price for both strategies? Explain your answer.

2.2.1 Answer

As we selected the door *before* the three empty boxes were revealed by the quiz master, the probability of finding a prize behind our selection is 1/5. This probability of winning is not affected by knowing about the three bad choices. As the probability of winning is 1 (the price must be behind a closed door), the probability that the prize is behind the not selected closed door is thus $1 - 1/5$ pr $4/5$.

Applying strategy A) (stick to our selection), we have thus a probability of winning the prize of 1/5. Applying strategy B) (alter selection), gives us a probability of 4/5 of winning the price. We must therefore change our selection. Note that these are average expectations and every real experiment will deliver slightly different true rates of winning.



2.2.2 Slide numbers

14, 15 and 16.

3 Discuss the origin of noise and its implications on data analysis

3.1 Slide numbers

18 and 21.

3.2 Answer

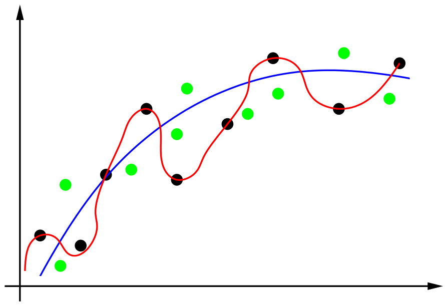
3.2.1 Origin of noise

Noise results from measurement errors (repeating an experiment leads for technical reasons always to different observations), missclassifications (experts get for example the phenotype wrong), and simplifications during modelling. The latter "modelling noise" refers to ignoring certain influence factors in the model which do however alter observations. Although this should be avoided if possible, sometimes these factors are very difficult to control and thus ignored.

Example of the latter: gene expression changes in response to the metabolic state of the model organism (not all organisms are really equally fed at the same time). Such factors must never be confounded with the question at hand since that would lead to misleading results. If they can not be controlled perfectly they will inevitably add random variation to the observations and thus constitute part of the "noise".

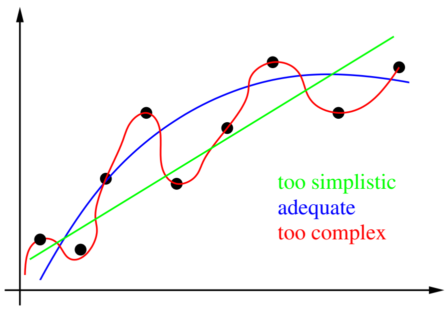
3.2.2 Implications on data analysis

Data analysis attempts finding an appropriate abstraction which explains some given observations (training data) reasonably well. An optimally chosen model should explain new observations (the green dots in the figure below) and avoid fitting the noise component in the measurements.



In the above sketch, the more complex (red) model fits the training data (black dots) better than the blue one. This improvement is though a result from poor averaging and trying to explain the noise.

A good fit is obtained by local averaging of observations and appropriately adjusting the model complexity (blue model).



In the above sketch the too complex and the too simplistic model are inadequate.

4 List the two data analysis strategies and link application scenarios to data analysis methods

4.1 Slide numbers

22 and 29.

4.2 Sketched answer

4.2.1 Analysis strategies (22)

There are two distinct strategies for data analysis: Situations which require modelling of a given target variable are in machine learning called "supervised problems". Solving such problems requires regression type data analysis methods. Situations which require exploring data for unknown structure are in machine learning called "unsupervised problems". These problems are approached with exploratory methods which either search for unknown groups in the data or for explanations of reduced dimension.

4.2.2 Data analysis applications and methods (29)

Task	$->$	Method
predict continuous y from input data	$->$	Regression
predict discrete y from input data	$->$	Classification
find unknown groups in input data	$->$	Clustering (e.g. k-means, mixture models)
find low dimensional representation for input data	$->$	Dimensionality reduction (PCA, ICA)

5 Provide a brief discussion of two important supervised learning tasks

5.1 Slide numbers

23, 24 and 25

5.2 Sketched answer

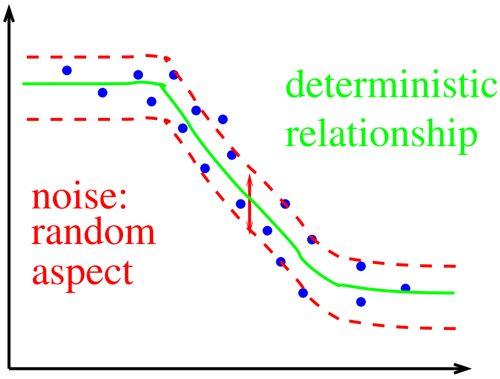
The two different regression tasks listed below do only differ in the nature of the response variable (i.e. the type of variable we want to predict): "Classical" regression is concerned with continuous valued responses. Classification is concerned with discrete valued response variables.

5.2.1 Regression with continuous response variable

The regression task uses a noisy data set $\mathcal{Z} = \{(y_1, \boldsymbol{x}_1), ..., (y_N, \boldsymbol{x}_N)\}$ from a life science experiment which allows separating the variables into one continuous valued target variable y which we would like to predict as good as possible and a set of independent variables $\boldsymbol{x}$ which are assumed providing information about the value of the target variable. Regression fits based on $\mathcal{Z}$ an "optimal" function relating $\boldsymbol{x}$ and y:

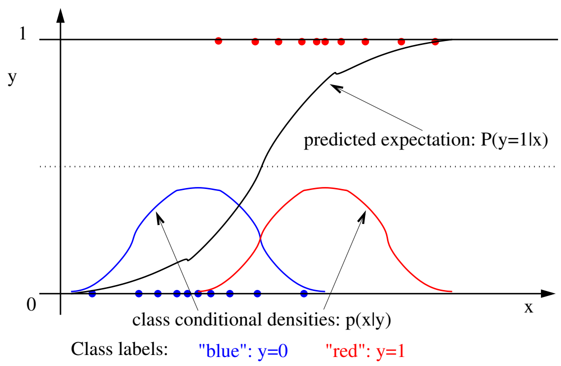
$$y = f(\boldsymbol{x}; \boldsymbol{\theta}) + \epsilon(\lambda)$$

The noise which affects measurements implies that y is a random variable and that we do best when predicting expected y values from x (local averages). The complete description of the regression model includes noise characteristics. The red error bars represent the standard deviation around the expected y value. This is a complete description in the case of Gaussian noise.



5.2.2 Classification

Although it appears to be different, classification is just a special case of regression with the target values being discrete. The general case allows for multiple class labels. Binary classification is a special case, where $y = \{0, 1\}$. In this case the predicted expectations are between zero and one and correspond to the posterior probability for class label 1, $P(y = 1|x)$.



6 Discuss two important instances of unsupervised learning

6.1 Slide numbers

26, 27 and 28

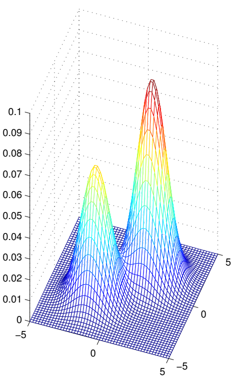
6.2 Sketched answer

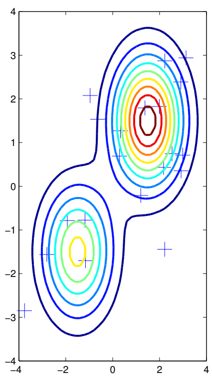
Both method classes for unsupervised learning attempt providing a simplified view of a data set. These approaches are useful for exploring data when fitting a model for a dedicated variable is not intended. *These approaches should not be used if we want to answer a question about relations between different variables!! In this case we have to use regression type models.* The two approaches discussed below differ in the nature of the summary they provide about a data set: Clustering and mixture density models provide a discrete summary which tells us which component of the model did most likely “generate” a particular observation. Continuous latent variable models provide a continuous summary which provides most of the information in a data set and is used for visualising data in a lower than original dimension.

The problem statement in unsupervised learning or exploratory data analysis is finding unknown structure in a data set $\mathcal{X} = \{x_1, x_2, ..., x_N\}$, x_n distributed according to unknown pdf $p(x)$. The learning task is summarising x by an unobserved variable t.

6.2.1 Mixture density models and clustering

Mixture density modelling represents the data generating density as $p(x) = \sum_k P(t = k)p(x|t = k)$, with $t \in \{1, .., K\}$ being a discrete variable. The term $p(x|t = k)$ represents the component density and $P(t = k)$ the prior probability that kernel k generates samples. An important example is the so called mixture of Gaussians model which uses $p(x|t = k) = \mathcal{N}(x; \mu_k, \lambda_k)$, i.e. Gaussians as component densities.



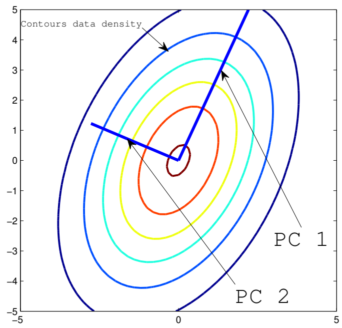


The figure above visualises this graphically. The left hand side shows a mesh plot of a resulting density over a two dimensional data set. The left hand plot shows the contour lines and the training data $\mathcal{X}$ which was used for model fitting. As a summary we obtain from a mixture of Gaussians model the component number $t = k$ which was most likely responsible for generating a data point $\boldsymbol{x}$.

6.2.2 Continuous latent variable models and dimensionality reduction

Continuous latent variable models use a similar representation of the data generating density. The summary provided about the data is captured by a continuous valued latent (unobserved) variable t and the summation in the mixture model is replaced by an integral $p(x) = \int_t p(t)p(x|t)dt$, $x \in \Re^k$. The latent variable is multivariate and continuous $t \in \Re^d$ and typically lower dimensional than the data itself, $k > d$.

Although non probabilistic (i.e. the model does not really use the density formulation above), principle component analysis (PCA) is an example of latent variable modelling. PCA represents a data point x as $x = m + W_1 t_1 + \ldots + W_d t_d$ with $x \in \Re^k$ and $t = [t_1, \ldots, t_d] \in \Re^d$ and $W_d : [k \times 1]$ denoting the d-th eigenvector of the sample covariance matrix. The underlying assumption is thus that the data set $\mathcal{X}$ was generated by a d dimensional Gaussian density.



The above figure illustrates the mapping we obtain when assuming that the contour lines represent curves of equal density under the data generating Gaussian. The summary provided by PCA are the projections on the principal component directions. We typically use fewer principal components than data dimensions and reduce thus the dimensionality of the data.

7 Formulate an objective function which can be used for model fitting and discuss its most important limitation

7.1 Slide numbers

30, 31 and 32

7.2 Sketched answer

7.2.1 An objective function for model fitting

Assuming that we may influence the behaviour of the model by changing the model coefficients, the objective of model fitting is minimising the discrepancy between the predictions we obtain with the chosen model and the given training data.

The goal of model fitting is thus tuning $\boldsymbol{\theta}$ such that $f(\boldsymbol{x}_n; \boldsymbol{\theta})$ represents all $(y_n \boldsymbol{x}_n)$ pairs well.

In order to achieve this goal, we need an expression we may optimise (maximise, minimise) for fitting all n "training" samples well.

One possible choice is using the so called sum of squared errors (SSE). The idea of the SSE is subtracting the deterministic part from all y_n. We get thus $\epsilon_n = y_n - f(\boldsymbol{x}_n; \boldsymbol{\theta})$. To penalise deviations for all data points in both directions equally, we sum over the squared difference.

$$\text{SSE} = \sum_n \epsilon_n^2 = \sum_n (y_n - f(\boldsymbol{x}_n; \boldsymbol{\theta}))^2$$

There are other objective functions as well. One of them is the so called (log)-likelihood.

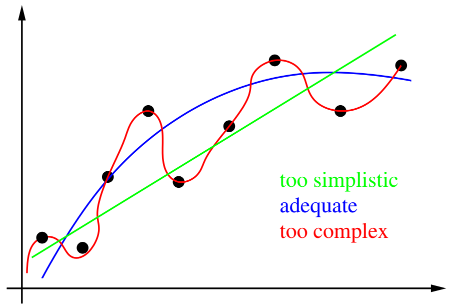
7.2.2 The most important limitation of the SSE

The most important limitation of the SSE is apparent, by analysing the behaviour of the SSE when changing the complexity of the model. Assuming that we have a linear model, the true relation in the data is captured by

$$y_n = \boldsymbol{x}_n^T \boldsymbol{\theta} + \epsilon_n.$$

This corresponds to a situation, where each sample can only be explained up to a small but finite ϵ_n, which must be there due to the noise component we have in all data sets. The SSE of the optimal model is thus larger than zero.

In general the optimal model is unknown and finding the appropriate complexity part of model fitting. To allow in the above example for models of varying complexity we can for example introduce nonlinearities (e.g. by allowing for arbitrary powers of x) and increase the number of parameters (here implicit in the dimension of the vector $\boldsymbol{\theta}$.



Doing so, we will eventually get curves like the red one in the above illustration which pass through all points in the data set exactly. This corresponds to having all ϵ_n equal to zero and thus an SSE of zero, which is the smallest value the SSE can attain. Adapting the model complexity during fitting can thus not be obtained by applying simple objective criteria like SSE since this will inevitably lead to too complex models. This property of the SSE and other objective functions such as likelihoods is also the reason, why over complex model are much more of a problem than too simplistic models.

References

[1] C. M. Bishop. *Pattern Recognition and Machine Learning.* Springer, New York, 2006.

[2] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification, Second Edition.* Wiley, New York, 2000.

[3] D. J. C. MacKay. *Information Theory, Inference and Learning Algorithms.* Cambridge University Press, Cambridge UK, 2003.