Decision trees and linear regression with PYTHON

Objective: The objective of this exercise is to become familiar with fitting decision trees and linear regression models in Python.

Material: Lecture notes *"Introduction to Machine Learning and Data Mining"* as well as the files in the exercise 5 folder available from Campusnet.

Discussion forum: You can get help on our online discussion forum: https://learn.inside.dtu.dk/d2l/le/79650/discussions/List

Software installation: Extract the Python toolbox from DTU Inside. Start Spyder and add the toolbox directory (`<base-dir>/02450Toolbox_Python/Tools/`) to `PYTHONPATH` (Tools/PYTHONPATH manager in Spyder). Remember the purpose of the exercises is not to re-write the code from scratch but to work with the scripts provided in the directory `<base-dir>/02450Toolbox_Python/Scripts/` Make sure that you have `sklearn` package installed (we have already used it during the week 2 exercises). `Sklearn` package and documentation can be found here:
`http://www.scikit-learn.org/stable/install.html`
To visualize decision trees, we will need GraphViz editor, you can find it here: `http://www.graphviz.org/` Below you can see how graphviz is installed properly on your computer. Remember to restart Spyder before this takes into effect.

Linux

You just need to paste following snips in your terminal

```
sudo apt-get install graphviz
conda install graphviz
```

and you are done!

Mac

You need to install the python package. Do it from your terminal with following snip

```
conda install graphviz
```

and the graphviz executables also from your terminal

```
/usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/
    master/install)"
```

Windows

You need to download graphviz from graphviz.org (clickable). Unzip the folder and note the path where you store it—you need to input the path in the script later.

Representation of data in Python:

	Python var.	Type	Size	Description
	`X`	numpy.array	$N \times M$	Data matrix: The rows correspond to N data objects, each of which contains M attributes.
	`attributeNames`	list	$M \times 1$	Attribute names: Name (string) for each of the M attributes.
	`N`	integer	Scalar	Number of data objects.
	`M`	integer	Scalar	Number of attributes.
Regression	`y`	numpy.array	$N \times 1$	Dependent variable (output): For each data object, `y` contains an output value that we wish to predict.
Classification	`y`	numpy.array	$N \times 1$	Class index: For each data object, `y` contains a class index, $\mathtt{y}_n \in \{0, 1, \ldots, C-1\}$, where C is the total number of classes.
	`classNames`	list	$C \times 1$	Class names: Name (string) for each of the C classes.
	`C`	integer	Scalar	Number of classes.

5.1 Decision Trees

In this part of the exercise we will fit decision trees using the Python class `sklearn.tree.DecisionTreeClassifier`.

As a splitting criterion, the function uses one of the two impurity measures:

$$\text{gdi}(t) = 1 - \sum_{i=0}^{c-1} p(i|t)^2 \qquad \text{equivalent to Gini}(t),$$
$$\text{deviance}(t) = -2 \sum_{i=0}^{c-1} p(i|t) \log p(i|t) \quad \text{equivalent to Entropy}(t).$$

We will analyze the vertebrate data given in Table 4.1 of ”Introduction to Data Mining” [1] as well as the wine data we have used in the previous exercise (this data is similar to the data introduced in the chapter on “Tree-based methods” in the book “Introduction to Machine Learning and Data Mining”).

The vertebrate data set has the following attributes, which are all categorical:

Vertebrate dataset

#	Attribute	Unit
1	Body temperature	0:Cold blooded, 1:Warm blooded.
2	Skin cover	0:None, 1:Hair, 2:Scales, 3:Feathers, 4:Fur, 5:Quills
3	Gives birth	0:No, 1:Yes.
4	Aquatic creature	0:No, 1:Yes, 2:Semi.
5	Aerial creature	0:No, 1:Yes.
6	Has legs	0:No, 1:Yes.
7	Hibernates	0:No, 1:Yes.

The wine data set have the following attributes, where 1–11 are continuous and 12 is categorical:

Wine dataset

#	Attribute	Unit
1	Fixed acidity (tartaric)	g/dm^3
2	Volatile acidity (acetic)	g/dm^3
3	Citric acid	g/dm^3
4	Residual sugar	g/dm^3
5	Chlorides	g/dm^3
6	Free sulfur dioxide	mg/dm^3
7	Total sulfur dioxide	mg/dm^3
8	Density	g/cm^3
9	pH	pH
10	Sulphates	g/dm^3
11	Alcohol	% vol.
12	Quality score	0–10

5.1.1 Examine and run the script `ex5_1_1.py`, which contains the vertebrates data given in Table 4.1 of "Introduction to Data Mining".

5.1.2 Examine and run the script `ex5_1_2.py`. The script fits a decision tree to the vertebrates data using the Gini splitting criterion and stops splitting only when nodes are pure.

Script details:

- *Import module* `tree` *from* `sklearn` *and type* `help(tree.DecisionTreeClassifier)` *to learn how to fit a decision tree in Python.*
- *The parameter* `criterion` *can be used to choose the splitting criterion (Gini or Entropy).*
- *The parameters* `min_samples_split`, `min_samples_leaf`, *and* `max_depth` *influence the stopping criterion.*
- *After fitting the tree, export its structure to GraphViz format. The generated file (here:* `tree_gini.gvz`*) can be visualized externally with i.e.* `Graphviz` *editor or* `dot` *command-line tool or automatically rendered in Spyder as done in the script.*
- *The parameter* `feature_names` *can be used during the exporting to supply the names of the attributes.*

Did you get a similar result to the figure 1? Try to understand the information provided by the tree.

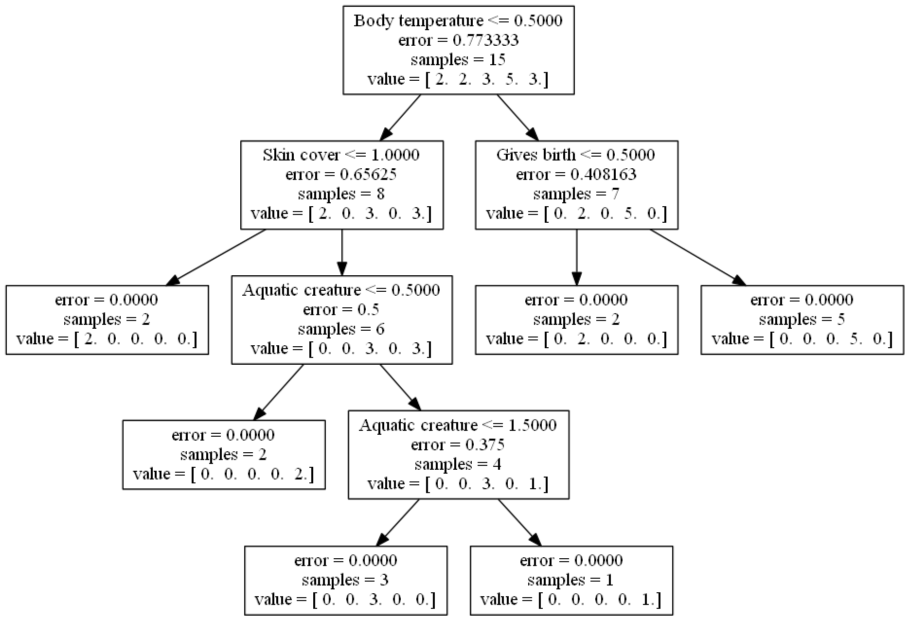
Body temperature <= 0.5000
error = 0.773333
samples = 15
value = [2. 2. 3. 5. 3.]
Skin cover <= 1.0000
error = 0.65625
samples = 8
value = [2. 0. 3. 0. 3.]
Gives birth <= 0.5000
error = 0.408163
samples = 7
value = [0. 2. 0. 5. 0.]
error = 0.0000
samples = 2
value = [2. 0. 0. 0. 0.]
Aquatic creature <= 0.5000
error = 0.5
samples = 6
value = [0. 0. 3. 0. 3.]
error = 0.0000
samples = 2
value = [0. 2. 0. 0. 0.]
error = 0.0000
samples = 5
value = [0. 0. 0. 5. 0.]
error = 0.0000
samples = 2
value = [0. 0. 0. 0. 2.]
Aquatic creature <= 1.5000
error = 0.375
samples = 4
value = [0. 0. 3. 0. 1.]
error = 0.0000
samples = 3
value = [0. 0. 3. 0. 0.]
error = 0.0000
samples = 1
value = [0. 0. 0. 0. 1.]

Figure 1: Decision tree of the vertebrates data

5.1.3 Examine and run the script `ex5_1_3.py`. The script shows how the tree changes when you use the Entropy (i.e. `deviance`) splitting criterion instead of the Gini (`gdi`).

Script details:

- *The parameter* `criterion` *can be used to choose the splitting criterion used.*

5.1.4 Use the script `ex5_1_4.py` to show that a dragon, which is cold-blooded, has scales, gives birth, is semi aquatic, is an aerial creature, has legs, and hibernates, will be classified as a reptile.

Script details:

- *After fitting decision tree, you can use the method* `predict()` *of the tree object to classify a new data object.*

5.1.5 Load the wine data set into Python using `ex5_1_5.py`.

The script removes outliers from the data set. Outliers are defined as data objects where Volatile acidity > 20 g/dm^3, Density > 10 g/cm^3, or Alcohol $> 200\%$.

The script also removes attribute 12, Quality score, from the data set, in order to use only the quantitative measurements for the prediction.

Script details:

- *You have worked with the wine data before in exercise 4.2.1 You can use your solution to that exercise as a starting point.*
- *To select a subset of data objects from the data matrix, you can write* `X=X[n,:]` *where* `n` *is either a vector of indices of the data objects to be selected, or a binary mask. Remember to update the class index vector accordingly* `y=y[n,:]`*, and to recompute* `N`*.*
- *Similarly you can select a subset of the attributes from the data matrix* `X=X[:,m]` *where* `m` *is either a vector of indices to the attributes, or a binary mask. Remember to update the list of attribute names accordingly, and to recompute the number of attributes* `M`*.*

You should end up with a data matrix of size $N \times M = 6304 \times 11$.

5.1.6 Inspect and run the script `ex5_1_6.py`. The script fits a decision tree to the wine data in order to estimate if the wine is red or white. The script uses the Gini splitting criterion and the following parameter value as stopping criterion `min_samples_split=100`.

Script details:

- *The solution to exercise 5.1.2 is used as a starting point.*

Explain what happens when you change the values of the parameter `min_samples_spli`

Observe the changes in the tree size after reducing the parameter.

5.1.7 Explain how the script `ex5_1_7.py` shows that a wine with the following attribute values would be classified as White.

#	Attribute	Value
1	Fixed acidity (tartaric)	6.9
2	Volatile acidity (acetic)	1.09
3	Citric acid	0.06
4	Residual sugar	2.1
5	Chlorides	0.0061
6	Free sulfur dioxide	12
7	Total sulfur dioxide	31
8	Density	0.99
9	pH	3.5
10	Sulphates	0.44
11	Alcohol	12

5.2 Linear and logistic regression

Regression is widely used in datamining and machine learning for predicting an output (dependent variable) for a data object given its attributes. Regression can be defined as the task of learning a target function that maps a set of attributes onto an output.

We will initially consider a linear regression problem where we will model the outputs, $\boldsymbol{y} = [y_1, y_2, \ldots, y_N]^\top$, based on a single attribute for each data object, $\boldsymbol{X} = [x_1, x_2, \ldots, x_N]^\top$, using the model

$$y_n = w_0 + w_1 x_n + \epsilon_n,$$

where ϵ_n is residual noise, w_0 a constant offset and w_1 the linear coefficient for x_n. We can write an equation for each data object,

$$\begin{aligned} y_1 &= w_0 + w_1 x_1 + \epsilon_1 \\ y_2 &= w_0 + w_1 x_2 + \epsilon_2 \\ &\vdots \\ y_N &= w_0 + w_1 x_N + \epsilon_N. \end{aligned}$$

Introducing a vector of all ones $\mathbf{1} = [1, 1, \ldots, 1]^\top$ we can write the N equation compactly in matrix-vector form

$$\boldsymbol{y} = w_0 \mathbf{1} + w_1 \boldsymbol{x} + \boldsymbol{\epsilon} = [\mathbf{1}\ \boldsymbol{x}] \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} + \boldsymbol{\epsilon}.$$

In order to estimate w_0 and w_1 we minimize the least squares error function, i.e.

$$(w_0, w_1) = \arg\min_{w_0,w_1} \sum_{n=1}^{N} (y_n - (w_0 + w_1 x_n))^2 = \arg\min_{w_0,w_1} \left\| \boldsymbol{y} - \begin{bmatrix} \mathbf{1} & \boldsymbol{x} \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} \right\|_F^2 .$$

Based on our linear regression model we can predict the value of an unobserved output y^* from an observed measurement x^* as

$$y^* = w_0 + w_1 x^*.$$

5.2.1 Inspect and run the script `ex5_2_1.py`. The script generates 100 data objects according to the linear regression model,

$$y_n = w_0 + w_1 x_n + \epsilon_n,$$

Each of the n observations (having one attribute value x_n) should be set to $x_1 = 0, x_2 = 1, x_3 = 2, \ldots, x_N = N - 1$, and ϵ_n is generated at random from a Normal distribution with mean zero and standard deviation 0.1. Inspect the script and verify the parameters of the model are $w_0 = -0.5$ and $w_1 = 0.01$.

Script details:

- *You can use functions* `range()` *or* `linspace()` *to make a sequence of integer or real numbers.*
- *You can generate Normal distributed random numbers using the function* `random.randn()`.

Make a scatter plot of your data. Does it look like the plot in figure 2?

5.2.2 Inspect and run the script `ex5_2_2.py`. Explain how the script estimates the model parameters, w_0 and w_1, from the data you have generated. Show that the estimated parameters are close but not exactly equal to the parameters used to generate the data.

Notice how the script plots predictions of the model as well as the predictions you get using the parameters used to generate the data in the same graph as the scatter plot you made in exercise 5.2.1.

Script details:

- *You can use the class* `sklearn.linear_model.LinearRegression` *as your model. Set* `fit_intercept` *parameter to True, unless you have added column of "1"s to your data matrix X.*

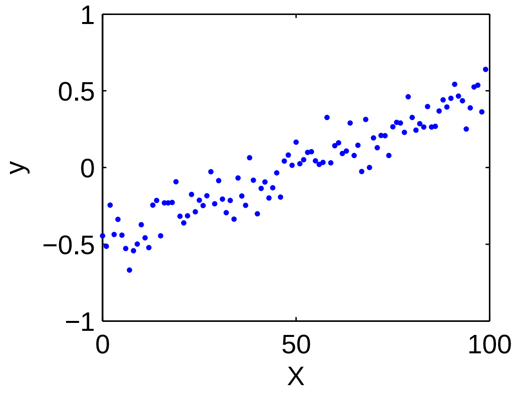

Figure 2: Scatter plot of data

· *Use method* `fit()` *to estimate parameters in the linear model, and method* `predict()` *to predict values for new data points.*

Show that the estimated model parameters get closer to the parameters used to generate the data if you include more data objects or reduce the noise variance.

Sometimes there are more complex relationships between the attributes and output. In order to capture more complex relationships we can include attributes that are specified transformations of existing attributes such as $\sqrt{x}$, $\log(x)$, x^2 and x^3. We will presently consider the model

$$y_n = w_0 + w_1 x_n + w_2 x_n^2 + \epsilon_n$$

The above model can again be written for all N observations simultaneously in terms of vectors and matrices, i.e.

$$\boldsymbol{y} = [\mathbf{1}\ \boldsymbol{x}_1\ \boldsymbol{x}_2] \begin{bmatrix} w_0 \\ w_1 \\ w_2 \end{bmatrix} + \boldsymbol{\epsilon} = [\mathbf{1}\ \boldsymbol{X}]\boldsymbol{w} + \boldsymbol{\epsilon}$$

where $\boldsymbol{x}_1 = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}, \quad \boldsymbol{x}_2 = \begin{bmatrix} x_1^2 \\ x_2^2 \\ \vdots \\ x_N^2 \end{bmatrix}$ and $\boldsymbol{w} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \end{bmatrix}$.

5.2.3 Inspect and run the script `ex5_2_3.py` which generates a data set according to the linear regression model including the squared term, fits the model

and makes a plot as in the previous exercise. The script allows you to specify separately how many terms to include (squared, cubed, etc.) for generating the data and fitting the model. Illustrate what happens when fitting a model with more higher order terms? How well is the model able to fit the data?

We will return to the wine data and attempt to predict the alcohol content using linear regression. The linear regression model we consider is

$$y_n = w_0 + \sum_k w_k x_{n,k} + \epsilon_n,$$

where y_n is the observed alcohol content, ϵ_n the model residual, w_k the value of the estimated model coefficient for the k^{th} attribute whereas $x_{n,k}$ denotes the n^{th} observation of the k^{th} attribute. As such, $x_{2,1}$ denotes the second observation of the first attribute (Fixed acidity) and $x_{5,2}$ the fifth observation of the second attribute (Volatile acidity).

5.2.4 Inspect and run the script `ex5_2_4.py`. The script loads the Wine data into Python using the solution to exercise 5.1.5. The script also fits a linear model that predicts the Alcohol attribute based on the 10 remaining attributes (again excluding Quality score) and makes a scatter plot of the true versus the predicted alcohol content. Explain how the visualizations show that the alcohol content can be predicted with an accuracy of approximately ± 1 percentage point.

Script details:

· *You can take similar approach as in exercises 5.2.2 and 5.2.3 to fit linear regression model and use it to predict values.*

Show from the estimated parameters of the linear regression model that wines with higher alcohol content in general have lower density.

5.2.5 Inspect and run the script `ex5_2_5.py`. The script introduces more variables in the regression by combining or transforming existing variables. Explain how that is done in the script and which transformations are used. Try to introduce your own transformations.

Script details:

- *For example you can include the square of the Fixed acidity in the model and the square of the Volatile as well as a multiplicative interaction between Fixed acidity and Volatile acidity. You can use* `numpy.power()` *and* `numpy.multiply()` *functions on appropriate columns of data matrix.*
- *Function* `numpy.bmat()` *is useful to concatenate matrices and/or vectors, for instance:* `X=np.bmat('X, Xfa2, Xva2, Xfava')`.
- *Plots of attributes versus the model residuals can reveal if there is structure in the output that can be explained by transformations of the attributes.*

How low can you get the mean squared error?

Until now we have discussed the most simple linear model based on the squared error cost function. In the generalized linear model, we can also use other cost functions as well as non-linear link functions. For instance, for outputs that are purely binary, $y \in \{0, 1\}$, the logistic function is very suitable, logistic$(s) = \frac{1}{1+exp(-s)}$, which maps the output in the regression analysis to the range (0,1). In figure 3 there is a plot of the logistic link function.

It can be very useful to limit the possible outputs to the range (0,1) if the outputs we wish to predict are binary. Predicting binary outputs is what we have previously referred to as classification, and generalized linear models can thus both be used for regression and classification. Classification with a binomial cost function and logistic link function is called *logistic regression*, and in Python is implemented in module: `sklearn.linear_model.logistic.LogisticRegression`

We will return to the wine data and predict the type of wine using logistic regression. The model we consider is

$$y_n = \text{logistic}\left(w_0 + \sum_k w_k x_{n,k}\right) + \epsilon_n$$

where y_n is the observed type of wine (Red: $y_n = 0$ and White: $y_n = 1$). We will say that a wine is White with probability p and Red with probability $1 - p$. where $p = \text{logistic}(w_0 + \sum_k w_k x_{n,k})$.

5.2.6 Inspect and run the script `ex5_2_6.py`. The script loads the Wine data into Python using the solution to exercise 5.1.5 and fits a logistic regression model that predicts the type of wine (Red or White) based on the attributes 1-11 (again excluding Quality score). Explain how this is done.

Show that the wine defined in exercise 5.1.7 would be classified as White with more than 90% probability.

Script details:

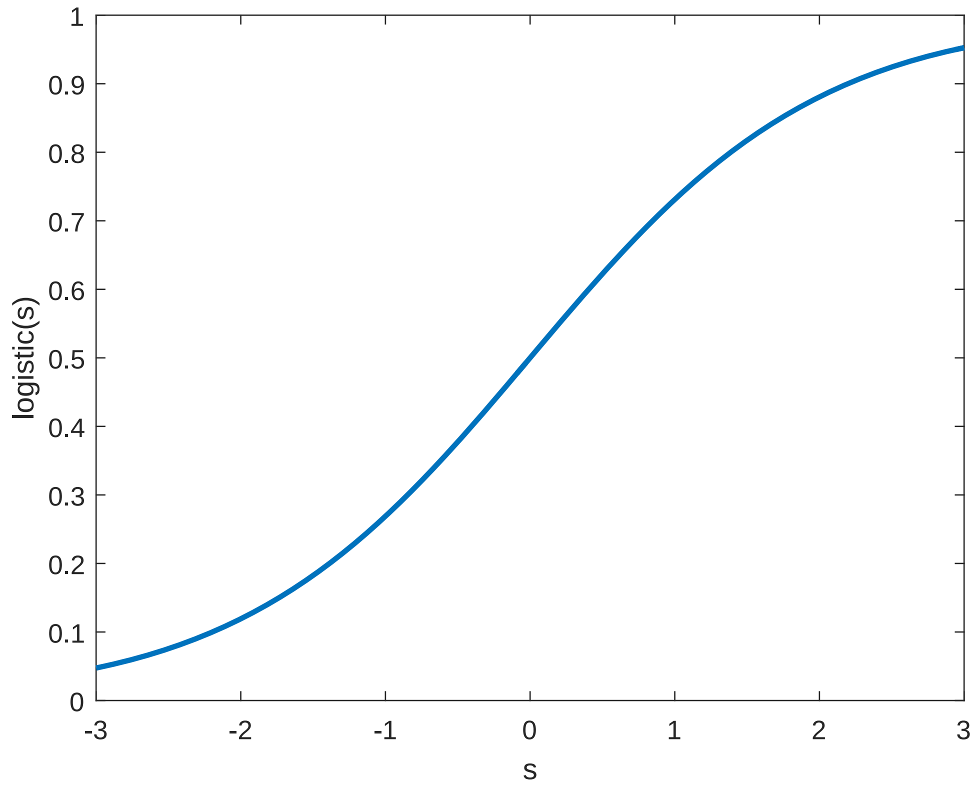

Figure 3: Plot of the logistic link function.

- *To estimate a logistic regression model, use the class:* `sklearn.linear_model.logistic.LogisticRegression`*.*
- *To make a prediction from the estimated model, use the function* `predict()`*. If you want to get probabilities of new data object belonging to given class use* `predict_proba()`*.*

5.3 Tasks for the report

The report will contain three sections, two about regression and one about classification. For the regression section, you will now be able to address this question:

Regression, part a: In this section, you are to solve a relevant regression problem for your data and statistically evaluate the result. We will begin by examining the most elementary model, namely linear regression.

- Explain what variable is predicted based on which other variables and what you hope to accomplish by the regression. Mention your feature transformation choices such as one-of-K coding. Since we will use regularization momentarily, apply a feature transformation to your data matrix $\mathbf{X}$ such that each column has mean 0 and standard deviation 1[1].

Meanwhile, for the **classification** section:

- Explain which classification problem you have chosen to solve. Is it a multi-class or binary classification problem?

1 Homework problems for this week

[1]We treat feature transformations and linear regression in a very condensed manner in this course. Note for real-life applications, it may be a good idea to consider interaction terms and the last category in a one-of-K coding is redundant (you can perhaps convince yourself why). We consider this out of the scope for this report

Problems

Question 1. Fall 2013 question 14: We consider a dataset on survival of breast cancer taken from `http://archive.ics.uci.edu/ml/machine-learning-databases/haberman/haberman.names`. The data has been binarized as outlined in Table 1. The dataset contain a total of 306 observations. In the dataset 81 survived after 5 years (i.e., $y = 1$) whereas 225 died (i.e., $y = 0$). We would like to build a decision tree and consider using positive axillary nodes detected (PAN) as an attribute condition at the root of the tree. We thereby split according to whether positive axillary nodes were detected and find:

- For the 170 subjects that had positive axillary nodes detected 62 survived.
- For the 136 subjects that did not have positive axillary nodes 19 survived.

What is the gain, Δ, of splitting according to whether a subject had positive axillary nodes (PAN) using the Gini as impurity measure $I(t)$, (i.e., $I(t) = 1 - \sum_{i=0}^{C-1} p(i|t)^2)$?

No.	Attribute description	Abbrev.
x_1	Young (< 60 years), $x_1 = 0$ or Old (≥ 60 years), $x_1 = 1$	Age
x_2	Operated before, $x_2 = 0$ or after 1960, $x_2 = 1$	OpT
x_3	Positive axillary nodes detected No, $x_3 = 0$ or Yes, $x_3 = 1$	PAN
y	Lived after 5 years No, $y = 0$ or Yes, $y = 1$	Surv

Table 1: A modified version of Haberman's Survival Data taken from http://archive.ics.uci.edu/ml/machine-learning-databases/haberman/haberman.names. The attributes x_1-x_3 denoting the age, operation time and cancer size as well as the output denoting survival after five years are binary. The data contains a total of $N = 306$ observations.

A -0.025

B 0

C 0.025

D 0.036

E Don't know.

Question 2. Spring 2013 question 12: We fit a linear regression model to the PM10 data shown in Table 2. The input attributes are standardized (i.e., we have subtracted the mean of each input attribute, x_1–x_7, and divided by their standard deviations) whereas the output logPM10 is kept in its original format. We obtain the following model:

$$f(\boldsymbol{x}) = 3.27 + 0.36x_1 - 0.01x_2 - 0.19x_3 + 0.01x_4 + 0.05x_7.$$

Which one of the following statements about the model is *incorrect*?

No.	Attribute description	Abbrev.
x_1	Logarithm of number of cars per hour	logCAR
x_2	Temperature 2 meter above ground (degree Celsius)	TEMP
x_3	Wind speed (meters/second)	WIND
x_4	Temperature difference between 25 and 2 meters (degree Celsius)	TEMPDIF
x_5	Wind direction (degrees between 0 and 360)	WINDDIR
x_6	Whole hour of the day	HOUR
x_7	Day number from October 1. 2001	DAY
y	Logarithm of PM10 concentration	logPM10

Table 2: The attributes of the PM10 data. The output is given by the hourly values of the logarithm of the concentration of PM10 particles (logPM10).

A According to the model WINDDIR and HOUR are not relevant for predicting the pollution level.

B According to the model fewer cars and more wind will result in lower pollution levels.

C According to the model it seems that pollution is decreasing over time.

D According to the model higher temperatures will result in lower pollution levels.

E Don't know.

Question 3. Spring 2014 question 5: We consider the Wholesale dataset shown in Table 3 and wish to predict whether a consumer is from Lisbon (y=0) or Oporto (y=1) by discarding observations from the Other region included in the wholesale data. After discarding the observations pertaining to the Other region we standardize the attributes x_1–x_6 (i.e., for each attribute subtract the mean and divide by the standard deviation) and fit a logistic regression model. We obtain the following model for the prediction of the origin of the consumer:

$$f(x_1,x_2,x_3,x_4,x_5,x_6) = \mathrm{logit}(-0.51 - 0.11x_1 - 0.36x_2 + 0.44x_3 + 0.39x_4 + 0.09x_5 - 0.28x_6),$$

where $\mathrm{logit}(w) = \frac{1}{1+\exp(-w)}$ is the logit function. Which one of the following statements about the model is *correct*?

No.	Attribute description	Abbrev.
x_1	Fresh products	FRESH
x_2	Milk products	MILK
x_3	Grocery products	GROCERY
x_4	Frozen products	FROZEN
x_5	Detergents and paper products	PAPER
x_6	Delicatessen products	DELI
y	Region	REGION

Table 3: The six input attributes x_1–x_6 denoting the annual consumption in monetary units of customers as well as the output y denoting which of the three regions; Lisbon, Oporto, and one additional region denoted Other, the customers came from in the wholesale customer data.

A According to the model it seems that people in Lisbon buy more FRESH products, MILK products and DELI products than people in Oporto.

B According to the model if a costumer after the standardization has $x_1 = x_2 = x_3 = x_4 = x_5 = x_6 = 0$ the customer is more likely to come from Oporto than Lisbon.

C The logit function will return the probability a person is from Lisbon.

D From the model it can be seen that FRESH and PAPER are unimportant and should be removed in order to avoid overfitting.

E Don't know.

References

[1] Pang-Ning Tan et al. *Introduction to data mining*. Pearson Education India, 2006.