Started on	Wednesday, 24 July 2024, 10:08 PM
State	Finished
Completed on	Wednesday, 24 July 2024, 10:17 PM
Time taken	9 mins 25 secs
Marks	21/21
Grade	**10** out of 10 (**100**%)

Question 1

Correct

Mark 1 out of 1

Which of the following holds true?

- ☐ a. Linear regression classifies input values into categorical output classes.
- [✓ b. Logistic regression classifies input values into categorical output classes.]
- ☐ c. Ridge regression enforces sparse parameters, i.e., most parameters being close to 0.
- [✓ d. Lasso regression enforces sparse parameters, i.e., most parameters being close to 0.] Yes. This is L1 regularization.
- [✓ e. Ridge and Lasso regularization are the same as L2 ($\|\theta\|_2^2$) and L1 ($\|\theta\|_1$) regularization.]
- [✓ f. Ridge and Lasso regularization apply to any model of the form $P(y|X = x) = \square(g_\theta(x), \sigma)$.]
- ☐ g. Ridge and Lasso regularization only apply to linear regression models.

Your answer is correct.

Correct

Marks for this submission: 1/1.

Question 2

Correct

Mark 4 out of 4

Besides having a desirable sigmoid shape and restricting its output to [0,1], the logistic function has some further quite nice properties that can come in handy for derivation etc.

Writing the logistic function as σ(x) = exp(x)/(1+exp(x)), sort the matching pairs of equal expressions.

- Rotational **symmetry**: σ(−a)= $\boxed{\text{1– σ(a)}}$
- **Inverse**: for l(y)= $\boxed{\text{log(y/(1 – y))}}$ holds l(σ(y))=y
- **Derivative**: d/dx σ = $\boxed{\text{σ(1-σ)}}$

Your answer is correct.

Correct

Marks for this submission: 4/4.

Question 3

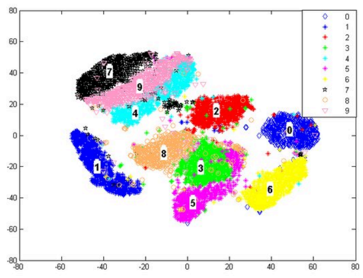
Correct

Mark 8 out of 8

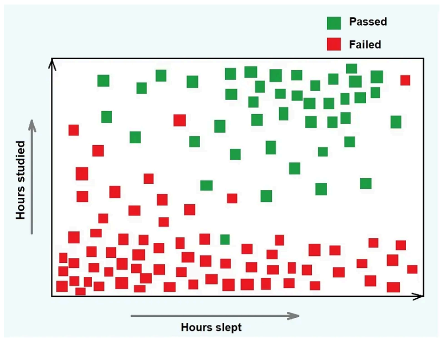
Bayesian networks are a typical example of generative models. A model is generative, if it describes the full joint distribution \(P(X,Y) \), and thus allows to sample new valid data points (x, y) (with or without any conditioning on x or also y). Other famous examples of generative models are deep neural network based language or image generators. In this lecture we had instead a closer look at discriminative models, i.e., models that only describe the conditional probability $P(Y|X)$, thus only allowing to discriminate between more(/most) and less likely outcomes y given an input x.

In the following, scatter plots of some simple classification problems with 2D inputs are given. Decide which of the so far discussed classification models (DNNs, logistic regression, QDA, LDA) are a good choice here.

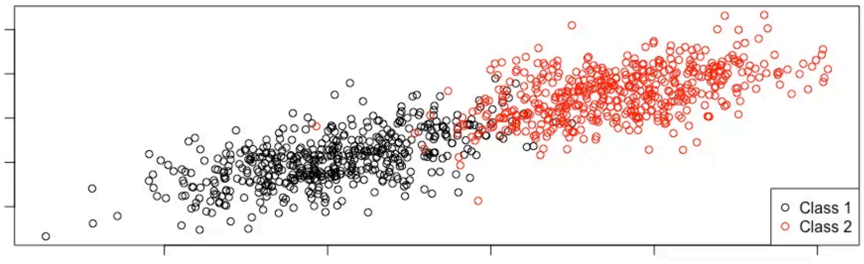
Note: Try to find the simplest model that might still be applicable with acceptable error rates (we will have a closer look at why this might be sensible in the chapter on the bias-variance tradeoff).



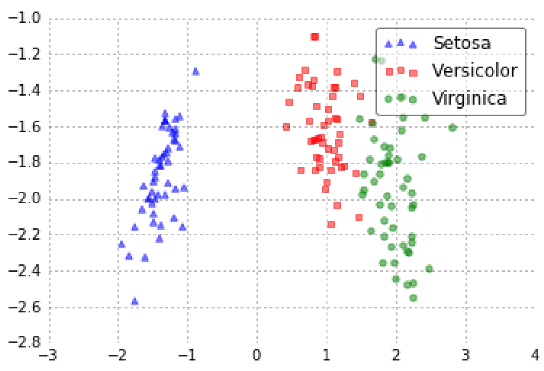
DNN (multi-class classification)



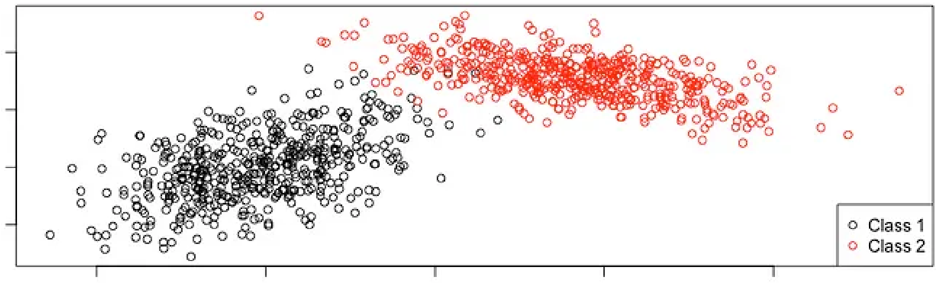
Logistic regression



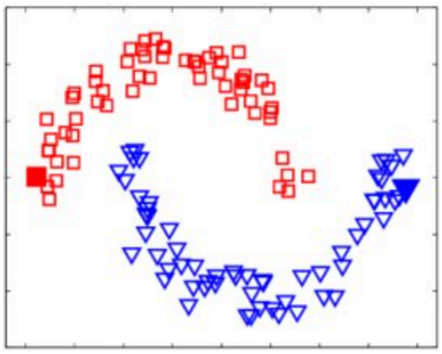
LDA (binary classification)



LDA (multi-class classification)



QDA



DNN (binary classification)

Your answer is correct.

Correct

Marks for this submission: 8/8.

Question 4

Correct

Mark 8 out of 8

As we have seen in the lecture, for regression problems linear models are one of the most favorable ones due to their simplicity and relatively few number of easy to determine parameters.

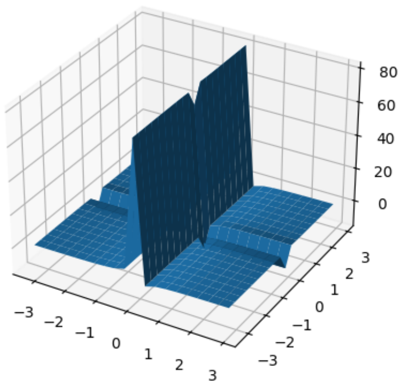
Hence, it can sometimes be desirable to simplify a given problem to a linear one. One trick for doing so is to apply a preprocessing to the data which turns the dependence of y on x into a linear one. In formulas: We assume one can write $y = \theta^T x_p = \theta^T p(x) = \sum_i \theta_i p(x)_i$ where p is a suitable preprocessing function. We have even seen that some problem shapes like polynomial regression or affine linear models can be turned into (non-affine) linear problems that way.

In the following, some plots of (simple) examplary (x, y)-datasets are provided which can be turned into linear regression problems using a suitable preprocessing function. Match the function to the plots.

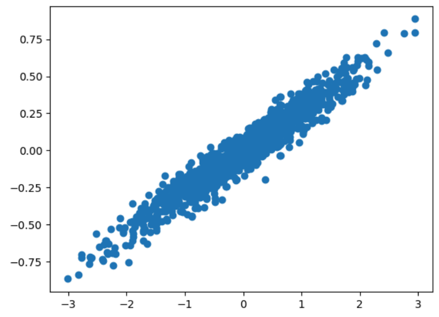
Note 1: The data is randomly sampled and plotted using this code. Feel free to play around with it to get a better feeling for the shapes one gets for different preprocessing.

Note 2: One can also use more complex and even parametrized preprocessing functions. E.g., a deep neural network $g_\theta : \mathbb{R}^n \to \mathbb{R}^m, x \mapsto x_p$ can be used for preprocessing! If employed as preprocessing for linear regression, this has little effect; we still have a DNN, and have to employ stochastic gradient descent to optimize its parameters (and that of the last linear layer). However, if applied as preprocessing for a softmax, we obtain a classifier that can be optimized using stochastic gradient descent by minimizing the log likelihood. Creating a classification DNN thus is as easy as adding a softmax output in the end (and now both the softmax and using negative log-likelihood/cross-entropy as loss makes sense from a mathematical perspective :-)).

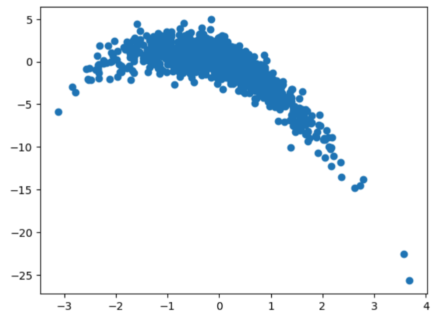
Note 3: Of course, this reformulation does not always work, and it might be hard to find a suitable data preprocessing function (where suitable means the performance of the predictor increases).



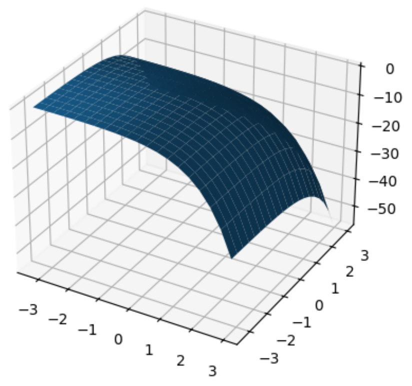
x_p = (1/x_1, 1/x_2) for 2D inputs (outliers at x_i=0 axes)



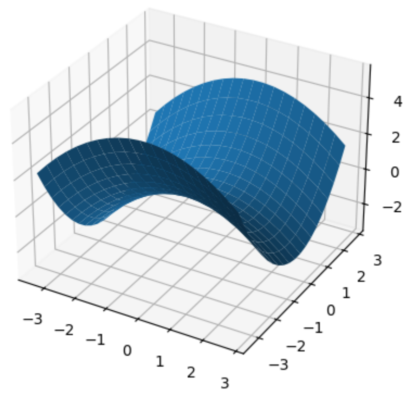
No preprocessing, x_p=x



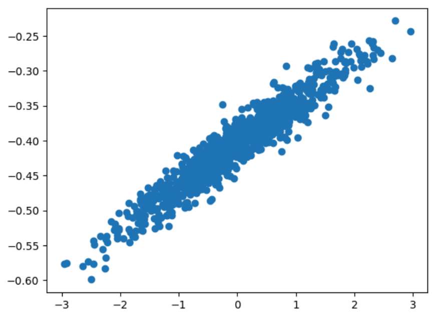
Polynomial regression: x_p = (1, x, x^2)



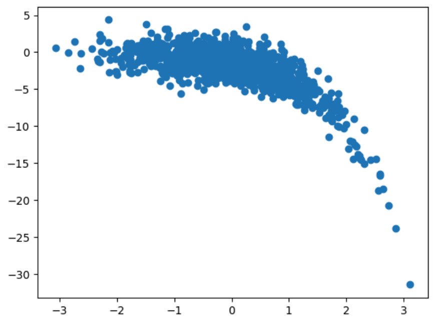
Exponential: x_p = exp(x) for 2D inputs



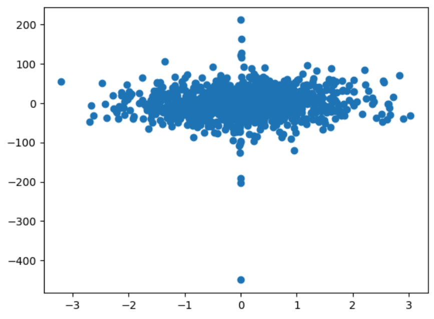
Parabola / Bent surface: x_p = x^2 for 2D inputs



Bias trick: x_p = (1, x)



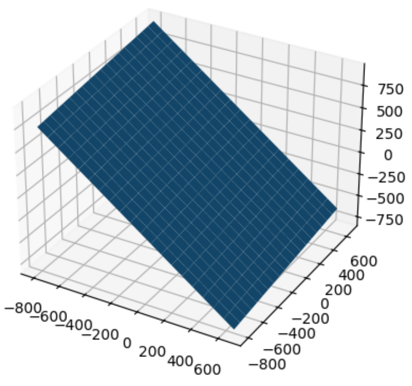
Exponential: x_p = exp(x) for 1D inputs



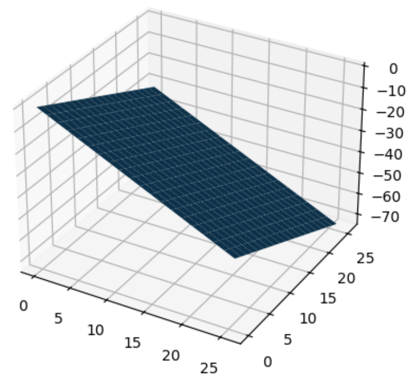
x_p = 1/x for 1D inputs (outliers at x=0)

Your answer is correct.

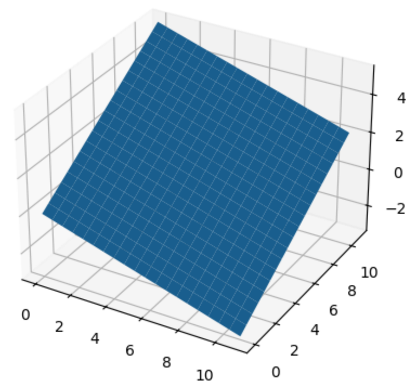
After applying the preprocessing, one gets, e.g., for the 1/x example:



for the exp example:



and for the bent surface:



Correct

Marks for this submission: 8/8.

◄ 05. Quiz - Bayesian Learning

Jump to...

07. Quiz - Support Vector Machines ►