Take-home exam
DIT866/DAT340: Applied Machine Learning, March 14–15, 2021

Course responsible: Richard Johansson, CSE (richard.johansson@cse.gu.se, +46317721887)

Formatting instructions:

- You need to submit a PDF or Word document. If you prefer to write your solution on paper, it is OK to submit a scanned copy as long as you can get the scan into a PDF.

Please note:

- To keep grading anonymous, please do not include your name in the file you submit.
- If there is something you don't understand about a question, and which is not covered on the Canvas page *Frequently asked questions*, please contact Richard over email or phone (9 AM – 5 PM) as soon as possible. Do not use Canvas to ask questions. No answers guaranteed between 5 PM and 9 AM.
- If you find typos or errors, please let me know and I will post an updated version as soon as I can.
- Until the submission deadline, it is strictly prohibited to communicate with other students about the contents of the take-home exam.
- Standard plagiarism regulations apply and the submitted file will be checked by a plagiarism detection program. Your solutions need to be your own and you are not allowed to copy any material from any source. (Please get in touch if you are unsure.)

Statement

Please include a text where you declare that

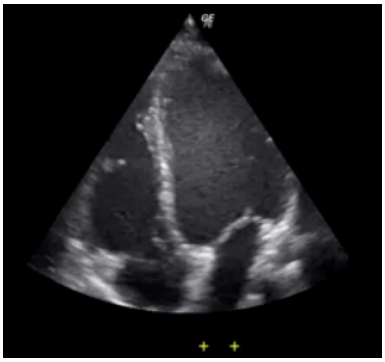
- you have not discussed with anyone (except possibly the course responsible) about the content of the exam;
- you have not copied any material from any source (printed or online).

Part 1: Basic questions

You need a score of at least **24** points in this part to receive a passing grade (*G/3*).

Question 1 of 12: Ultrasound images of the heart (12 points)

At a hospital, we would like to develop a machine learning system that processes ultrasound images of the heart in patients that are investigated for cardiovascular diseases. The figure below shows an example of such an image.[1]



The goal here is to develop a system that considers the *size* and *mobility* of the heart. In particular, with respect to the size aspect, the system should determine whether the heart has *normal size*, *slightly increased size*, *moderately increased size*, or *significantly increased size*. Similarly, with respect to the mobility, we'd like to determine whether the heart has *normal mobility*, *slightly reduced mobility*, *moderately reduced mobility*, or *significantly reduced mobility*.

The input to this system consists of raw pixel data. Although a wide range of measurements will typically be considered in a realistic medical scenario, we will only consider the image data here. Furthermore, in a real-world application we would consider a short "film" but here we will assume that we will just consider single static images.

(a, 6p) Explain how you would implement a machine learning model that can handle these prediction tasks. You don't need to show Python code or give exact values of hyperparameters, but please give a description of the system and explain all steps you would carry out when developing it.

(b, 2p) What metric(s) should we use to evaluate a system like this?

(c, 4p) Please express your opinion regarding any ethical issues you think are relevant in this context.

[1] `https://commons.wikimedia.org/wiki/File:Ultrasound_of_human_heart_apical_4-cahmber_view.gif`

Question 2 of 12: Rescaling a feature (6 points)

We are using machine learning for some classification task. We are considering two different classifiers, a random forest and a neural network. For instance, in scikit-learn we would write something like the following:

```
rf = RandomForestClassifier(max_depth=3, n_estimators=5, random_state=0)
nn = MLPClassifier(activation='tanh', hidden_layer_sizes=10, random_state=0)
```

One of the features is based on a length measurement in meters. Let's assume that we represent these measurements as millimeters instead: that is, we multiply the numbers by 1,000. We then retrain the classifiers on the same dataset.

(a, 2p) Why is the random forest not affected by this change?

(b, 2p) Why is the neural network affected by this change?

(c, 2p) When working with neural networks, how can we make the training more robust in this respect?

Question 3 of 12: Anomaly detection (6 points)

We develop a logistic regression classifier to detect anomalies in some equipment, and we collect a test set to evaluate this classifier. The table below shows 20 examples from this test set. The first column shows the gold-standard label and the second the probability of the class *abnormal* according to the classifier.

abnormal	0.58
normal	0.57
normal	0.32
abnormal	0.57
normal	0.23
normal	0.20
normal	0.58
normal	0.10
normal	0.19
normal	0.04
normal	0.01
abnormal	0.75
abnormal	0.80
normal	0.05
abnormal	0.97
abnormal	0.11
abnormal	0.54
abnormal	0.45
abnormal	0.92
normal	0.58

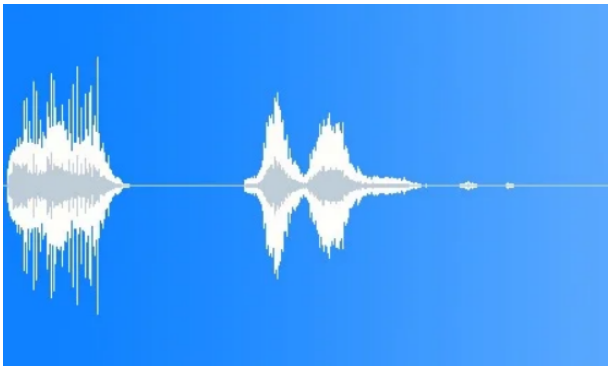
(a, 2p) Compute the precision and recall scores of this classifier with respect to the abnormal class.

(b, 2p) Describe a scenario where it could be important to have a recall close to 1.

(c, 2p) As described above, the classifier is a logistic regression model. It uses a collection of features based on various measurements in our equipment. How can we bring the recall of this classifier closer to 1? It's OK if this change leads to a drop in precision, but a trivial classifier that always outputs *abnormal* is not acceptable here.

Question 4 of 12: Events in sound recordings (4 points)

We would like to develop a system that detects different types of events in sound recordings of people sleeping. We collect a number of recordings at 44.1 kHz. Such recordings may be visualized, as exemplified in the figure below.



In this particular segment, the person first snores and then takes two breaths.

The goal for the machine learning system is to detect the occurrence of a number of event types such as snoring, breathing, coughing, etc. For each event detected in the audio signal, we'd like the system to determine the event type (e.g. snoring) and the start and end points of the event.

(a, 2p) What kind of manual annotation do we need to create in order to build a training set for a supervised machine learning system that carries out this event detection task?

(b, 2p) If we develop a system to find such events in the recordings, how should we evaluate this system?

Question 5 of 12: Feature selection (4 points)

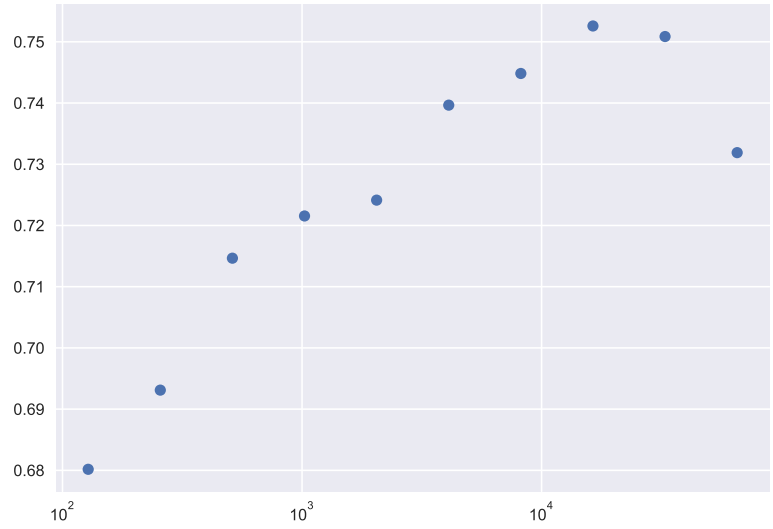
We have a training set for a supervised classification task. This training set consists of about 1,000 instances, where each input x consists of about 65,000 features and the output is a binary value. There is also a corresponding test set.

We use this data to train and evaluate a classifier using scikit-learn. Here is the code:

```
from sklearn.feature_selection import SelectKBest
from sklearn.svm import LinearSVC
pipeline = make_pipeline(SelectKBest(k=...), LinearSVC())
pipeline.fit(X, Y)
```

```
from sklearn.metrics import accuracy_score
acc = accuracy_score(Ytest, pipeline.predict(Xtest))
```

We explore some different values of the hyperparameter `k` in `SelectKBest` and compute the accuracy for each value. The figure below shows the result.



Explain the shape of the curve in this figure.

Question 6 of 12: Data augmentation for images (4 points)

When training machine learning models for images, it is common to apply *data augmentation* techniques where we carry out some sort of modification to the images we have in the training set. For instance, we might increase the contrast of an image, mirror it along the x axis, or rotate it. In the Keras library, there are several types of data augmentation techniques available in the utility `ImageDataGenerator`, as shown below.

```
tf.keras.preprocessing.image.ImageDataGenerator(
    featurewise_center=False, samplewise_center=False,
    featurewise_std_normalization=False, samplewise_std_normalization=False,
    zca_whitening=False, zca_epsilon=1e-06, rotation_range=0, width_shift_range=0.0,
    height_shift_range=0.0, brightness_range=None, shear_range=0.0, zoom_range=0.0,
    channel_shift_range=0.0, fill_mode='nearest', cval=0.0,
    horizontal_flip=False, vertical_flip=False, rescale=None,
    preprocessing_function=None, data_format=None, validation_split=0.0, dtype=None
)
```

(a, 1p) What is the reason we use data augmentation for image-based ML tasks?

(b, 3p) What are the tradeoffs we need to think of when using this technique? For instance, what happens if we set the `rotation_range` in Keras to a value that is too low? What if it is too high?

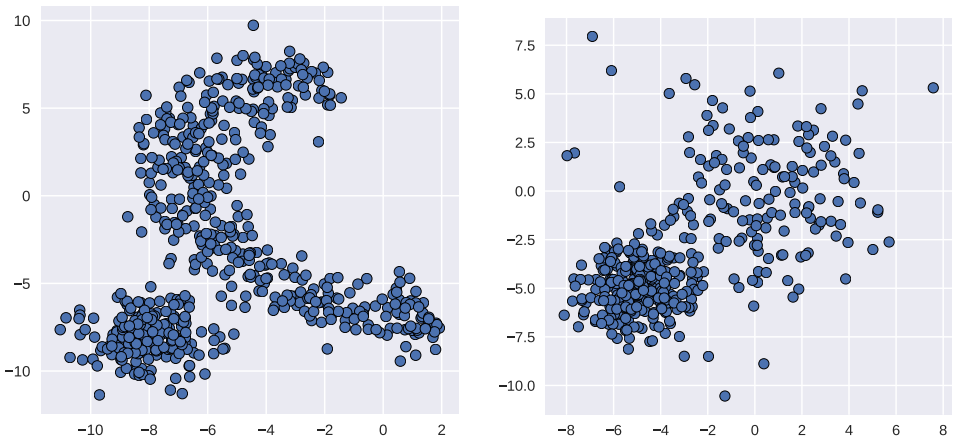
Part 2: Questions for the high grades

DIT866: You need a total score of **62** points to receive the grade *VG*.

DAT340: You need a total score of **48** for the grade *4*, and **62** for the grade *5*.

Question 7 of 12: Clustering (8 points)

(a, 1p) We have a couple of datasets and we want to apply some sort of clustering to find natural groups in the data. These datasets are two-dimensional and here are the scatterplots.



What do you think an ideal clustering algorithm should produce in these two cases? That is, what are the natural groups in the data in your view?

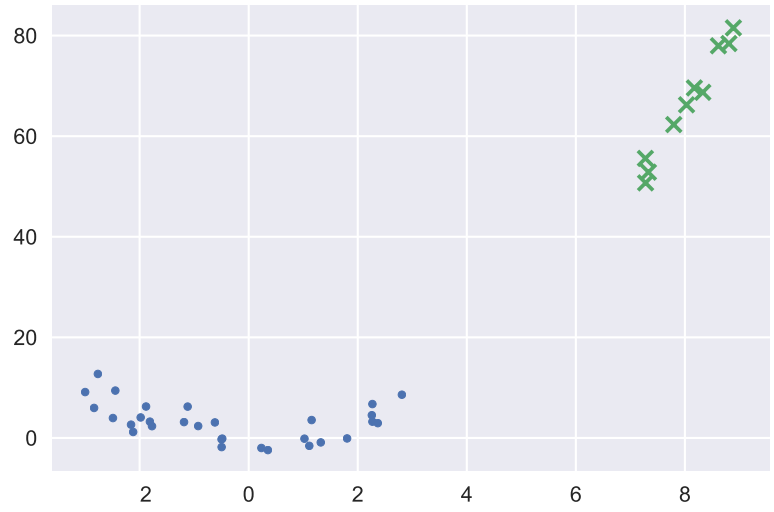
(b, 4p) Roughly, what do you think that *K*-means and DBSCAN would produce for these datasets? You can assume that we use the "best" hyperparameter tuning in both cases.

(c, 1p) How could you use a clustering algorithm such as *K*-means to cluster images? Describe what you would try to do if faced with this task.

(d, 2p) We apply a clustering algorithm to a set of images of cars and bicycles, and now we wonder whether the clustering has automatically discovered how to distinguish the cars from the bicycles. How would you measure to what extent this is true?

Question 8 of 12: Extrapolation (8 points)

The figure below shows a dataset for a regression task. Here, we would like to predict the output (the y axis) as a function of the input (the x axis).



The training set is represented by the blue dots in this figure. We train four different regression models using scikit-learn:

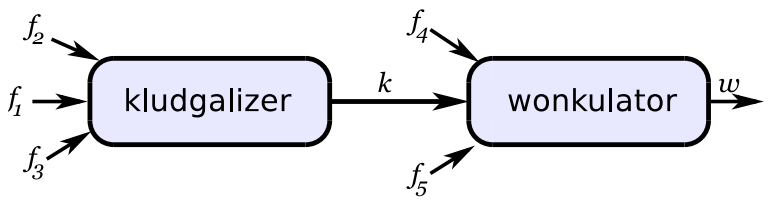
```
mlp1 = MLPRegressor(activation='relu', hidden_layer_sizes=16)
mlp2 = MLPRegressor(activation='tanh', hidden_layer_sizes=16)
rf = RandomForestRegressor(max_depth=5, n_estimators=10)
lin = LinearRegression()
```

The test set (green crosses in the figure) is sampled in a region that is completely distinct from the training set. Describe the potential behavior of each of these models when applied to the test set. We're writing "potential" here because there will be some variation in the models due to initialization, tweaking and so on; the idea here is that you should discuss and explain some ways that these models might possibly behave.

Question 9 of 12: Connected machine learning systems (6 points)

In an application at a Gothenburg industry, we want to develop two subsystems: the *kludgalizer* and the *wonkulator*. We want to use machine learning to implement both of these systems.

The kludgalizer depends on three input features f_1, f_2, and f_3 that are generated from specialized sensor hardware, and it produces a discrete output k. The wonkulator uses the features f_4 and f_5, also from hardware, in addition to a kludgalizer output k.



To develop these systems, we have a training set that is structured as a tabular file. Here, k and w are the desired kludgalizer and wonkulator outputs, respectively.

f_1	f_2	f_3	f_4	f_5	k	w
⋮	⋮	⋮	⋮	⋮	⋮	⋮

(a, 3p) If we train the wonkulator directly on this data, using the inputs f_4, f_5, and k from the training file, it is likely to perform quite poorly when we run the system "live." Explain why.

(b, 3p) Propose some way to mitigate this problem.

Question 10 of 12: Multiclass linear classification (6 points)

We want to build a classifier for multiclass problems: that is, when we have more than two classes. We use a linear classifier for this.

The model's parameters are stored in a weight matrix $\boldsymbol{W}$ with m rows and n columns, where m is the number of classes and n the number of features. Intuitively, each row is like a binary classifier that detects one of the classes. For an instance $\boldsymbol{x}$ (a feature vector), we can compute a score for the class y by taking the row $\boldsymbol{w}_y$ from $\boldsymbol{W}$ and then computing the dot product $\boldsymbol{w}_y \cdot \boldsymbol{x}$.

So to classify the instance $\boldsymbol{x}$, we search over all possible classes and select the one that gives the highest score:

$$\hat{y} = \arg\max_y \boldsymbol{w}_y \cdot \boldsymbol{x}$$

There are many different loss functions we might use to train such a model. In our case, the loss function looks like this:

$$\text{Loss}(\boldsymbol{W}, \boldsymbol{x}, y) = \boldsymbol{w}_{\hat{y}} \cdot \boldsymbol{x} - \boldsymbol{w}_y \cdot \boldsymbol{x}$$

That is: for an instance $\boldsymbol{x}$ with the gold-standard class label y, we first compute the prediction $\hat{y}$ as defined above, and then the difference in scores between the predicted class and the gold-standard class.

(a, 1p) What are we trying to achieve when using this loss function? Explain in words.

(b, 3p) The subgradient of the loss function with respect to the weight matrix $\boldsymbol{W}$ is a matrix of the same size as $\boldsymbol{W}$. To express the subgradient, let's introduce the utility function $\phi(\boldsymbol{x}, y)$. For a feature vector $\boldsymbol{x}$ and a class y, this function returns an m-by-n matrix where the row corresponding to y is equal to $\boldsymbol{x}$, and the other rows are all zero.

The subgradient of the loss now simply becomes

$$\nabla_{\boldsymbol{W}} \text{Loss} = \phi(\boldsymbol{x}, \hat{y}) - \phi(\boldsymbol{x}, y)$$

Write the pseudocode (or Python approximation) for a stochastic gradient descent algorithm (with a minibatch size of 1) to train a classifier by minimizing the loss function on a training set. You can assume that the learning rate is a user-defined constant and that we don't use a regularizer.

(c, 2p) If we are not careful, a simple implementation of our training algorithm may be quite inefficient, in particular if the number of classes is quite large. Modify your pseudocode so

that it is more efficient and avoids redundant computations. Alternatively, motivate why your previous solution is already efficient enough.

Question 11 of 12: Approximation (5 points)

We have some sort of black-box system and we want to use a machine learning approach to train a new system that replicates the behavior of the original system as faithfully as possible. For instance, the black-box system might be a neural network that is too large to fit onto a mobile device, or some sort of proprietary system that we want to mimic.

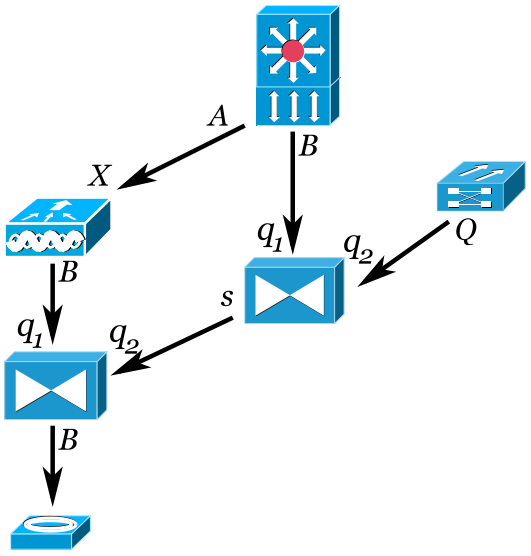
How would you solve a task like this? Describe all relevant steps and what kind of resources you would use in the ideal case. How do you know you have succeeded?

To make the discussion more specific, we can assume that the black-box system is a classifier that gives a probabilistic output (similar to e.g. a neural network with a softmax output). Apart from this, we make no assumptions about the inner workings of this system.

Question 12 of 12: Classifying networks (5 points)

A large Swedish telecom company manufactures different types of components that can be assembled as a network structure. Each component belongs to a certain type, and has a number of inputs and outputs.

The following figure shows an example of such a network.



In some scenarios, we would like to apply a classification or regression model to the network as a whole. For instance, we might want to develop a classifier that determines whether the network is connected correctly or not.

(a, 4p) How would you develop such a classifier?

(b, 1p) Can you think of an application that does *not* involve a network of components, but resembles this type of problem on a structural level?