SGN-41007 Pattern Recognition and Machine Learning
Exam 1.3.2017
Heikki Huttunen

- ▷ Use of calculator is allowed.
- ▷ Use of other materials is not allowed.
- ▷ The exam questions need not be returned after the exam.
- ▷ You may answer in English or Finnish.

1. Describe the following terms and concepts by a few sentences. (max. 6 p.)

   (a) Rectified linear unit

   (b) Linear classifier

   (c) Ensemble classifier

   (d) Multilabel classifier

   (e) Stratified cross-validation

   (f) L_1 regularization

2. The *Poisson distribution* is a discrete probability distribution that expresses the probability of a number of events $x \geq 0$ occurring in a fixed period of time:

$$p(x;\lambda) = \frac{e^{-\lambda}\lambda^x}{x!}$$

   We measure N samples: $x_0, x_1, \ldots, x_{N-1}$ and assume they are Poisson distributed and independent of each other.

   (a) Compute the probability $p(\mathbf{x};\lambda)$ of observing the samples $\mathbf{x} = (x_0, x_1, \ldots, x_{N-1})$. (1p)

   (b) Compute the natural logarithm of p, *i.e.*, $\log p(\mathbf{x};\lambda)$. (1p)

   (c) Differentiate the result with respect to λ. (2p)

   (d) Find the maximum of the function, *i.e.*, the value where $\frac{\partial}{\partial\lambda}\log p(\mathbf{x};\lambda) = 0$. (2p)

3. A dataset consists of two classes, containing four samples each. The samples are shown in Figure 1. The classes are linearly separable, and there are many linear decision boundaries that classify the training set with 100 % accuracy.

   (a) Find one such linear classifier. You can use whatever method you want (except the LDA), but justify your answer. Present the decision rule for sample $\mathbf{x} \in \mathbb{R}^2$ in the following format:

$$\text{Class}(\mathbf{x}) = \begin{cases} 1, & \text{if } \boxed{\textit{something}} \\ 2, & \text{otherwise} \end{cases}$$

   (b) Find the Linear Discriminant Analysis (LDA) classifier for this data. You can choose the threshold arbitrarily, but the projection vector has to be the LDA projection. Present the decision rule in the above format in this case as well.

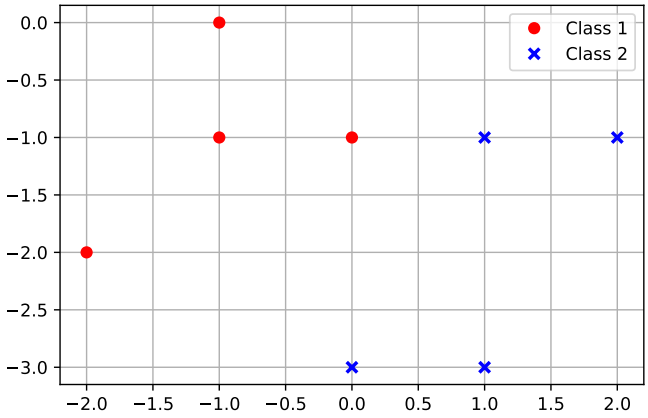

Figure 1: Training sample of question 3.

4. (a) (3 pts) In the lectures we defined the *logistic loss function*:

$$\ell(\mathbf{w}) = \sum_{n=0}^{N-1} \ln(1 + \exp(y_n \mathbf{w}^T \mathbf{x}_n)). \tag{1}$$

   i. Compute the formula for its gradient $\frac{\partial \ell(\mathbf{w})}{\partial \mathbf{w}}$.

   ii. There are two alternative strategies for using the gradient.

   - **Batch gradient:** Compute the gradient from all samples and then apply the gradient descent rule $\mathbf{w} \leftarrow \mathbf{w} - \eta \frac{\partial \ell(\mathbf{w})}{\partial \mathbf{w}}$.
   - **Stochastic gradient:** Compute the gradient from one sample and then apply the gradient descent rule. In other words, pretend $N = 1$ in formula 1.

   In the latter case, compute the next estimate for $\mathbf{w}$ when $\mathbf{x}_n = [-1, 1]^T$ and $y[n] = 1$ and $\mathbf{w} = [2, 1]^T$.

(b) (3 pts) Consider the Keras model defined in Listing 1. Inputs are 128×128 color images from 10 categories.

   i. Draw a diagram of the network.

   ii. Compute the number of parameters for each layer, and their total number over all layers.

	Prediction	True label
Sample 1	0.8	1
Sample 2	0.5	1
Sample 3	0.6	0
Sample 4	0.2	0

Table 1: Results on test data for question 5a.

Listing 1: A CNN model defined in Keras

```
model = Sequential()

w, h = 3, 3
sh = (3, 128, 128)

model.add(Convolution2D(32, w, h, input_shape=sh, border_mode='same'))
model.add(MaxPooling2D(pool_size=(4, 4)))
model.add(Activation('relu'))

model.add(Convolution2D(32, w, h, input_shape=sh, border_mode='same'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Activation('relu'))

model.add(Flatten())
model.add(Dense(100))
model.add(Activation('relu'))

model.add(Dense(10, activation = 'softmax'))
```

5. (a) (3p) A random forest classifier is trained on training data set and the `predict_proba` method is applied on the test data of only four samples. The predictions and true labels are in Table 1. Draw the receiver operating characteristic curve. What is the Area Under Curve (AUC) score?

   (b) (3p) The following code trains a list of classifiers with a fixed training data set and estimates their accuracy on a fixed test data set. What are the missing lines of code in listing 2?

   i. Define a list of classifiers: Logistic Regression, SVM and Random Forest.

   ii. Insert code for computing the accuracy scores.

Listing 2: Training and error estimation of classifiers.

```
import numpy as np
from sklearn.neighbors import KNeighborsClassifier
from sklearn.lda import LDA
from sklearn.svm import SVC, LinearSVC
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from data_provider import load_training_data, load_test_data
from sklearn.metrics import accuracy_score
# The above function has signature: accuracy_score(y_true, y_pred)

# Load data:
X_train, y_train = load_training_data()
X_test, y_test   = load_test_data()

# Define classifier list:
classifiers = # <insert code 1 here>

# Test each item in list:
for clf in classifiers:
    # <insert code 2 here>
    # 1) Train clf using the training data
    # 2) Predict target values for the test data
    # 3) Compute accuracy of prediction

    print ("Accuracy: %.2f" % (score))
```

Related Wikipedia pages

Inversion of 2 × 2 matrices [edit]

The *cofactor equation* listed above yields the following result for 2 × 2 matrices. Inversion of these matrices can be done as follows:[6]

$$\mathbf{A}^{-1} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \frac{1}{\det \mathbf{A}} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix} = \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}.$$

ROC space [edit]

The contingency table can derive several evaluation "metrics" (see infobox). To draw a ROC curve, only the true positive rate (TPR) and false positive rate (FPR) are needed (as functions of some classifier parameter). The TPR defines how many correct positive results occur among all positive samples available during the test. FPR, on the other hand, defines how many incorrect positive results occur among all negative samples available during the test.

A ROC space is defined by FPR and TPR as *x* and *y* axes respectively, which depicts relative trade-offs between true positive (benefits) and false positive (costs). Since TPR is equivalent to sensitivity and FPR is equal to 1 − specificity, the ROC graph is sometimes called the sensitivity vs (1 − specificity) plot. Each prediction result or instance of a confusion matrix represents one point in the ROC space.

If the entries in the column vector

$$\mathbf{X} = \begin{bmatrix} X_1 \\ \vdots \\ X_n \end{bmatrix}$$

are random variables, each with finite variance, then the covariance matrix Σ is the matrix whose (*i*, *j*) entry is the covariance

$$\Sigma_{ij} = \operatorname{cov}(X_i, X_j) = \operatorname{E}[\,(X_i - \mu_i)(X_j - \mu_j)\,]$$

where

$$\mu_i = \operatorname{E}(X_i)$$

is the expected value of the *i* th entry in the vector **X**. In other words,

$$\Sigma = \begin{bmatrix} \operatorname{E}[(X_1 - \mu_1)(X_1 - \mu_1)] & \operatorname{E}[(X_1 - \mu_1)(X_2 - \mu_2)] & \cdots & \operatorname{E}[(X_1 - \mu_1)(X_n - \mu_n)] \\ \operatorname{E}[(X_2 - \mu_2)(X_1 - \mu_1)] & \operatorname{E}[(X_2 - \mu_2)(X_2 - \mu_2)] & \cdots & \operatorname{E}[(X_2 - \mu_2)(X_n - \mu_n)] \\ \vdots & \vdots & \ddots & \vdots \\ \operatorname{E}[(X_n - \mu_n)(X_1 - \mu_1)] & \operatorname{E}[(X_n - \mu_n)(X_2 - \mu_2)] & \cdots & \operatorname{E}[(X_n - \mu_n)(X_n - \mu_n)] \end{bmatrix}.$$