Exercise Set 2

- Answer anonymously, i.e., do not write your name to the answer sheet.
- Submit the answer via Moodle at latest on 22 November 2020 (Moodle submission will open during the week starting on 16 November).
- Your answer will be peer-reviewed by other students and you will review your answer and answers of 2 random peers during the week starting on 23 November.
- The assignment should be completed by one person, but discussions with others are encouraged. Your final solution must be your own.
- The language of the assignments is English.
- The submitted report should be in a single Portable Document Format (pdf) file.
- Answer to the problems in the correct order.
- Read the general instructions in Moodle before starting with the problems.

Problem 9

[15% points]

Objective: Generative classifiers for normal distributions, linear discriminant analysis.

Read Section 4.4 of James et al.

Consider classification problem where the observations can belong to K classes, where $\pi_k \equiv P(Y = k)$ represents *a prior* probability that randomly chosen observation comes from class k, $f_k(x) \equiv P(X = x \mid Y = k)$ stands from the *probability density function* of X given that the observation belongs to class k. Assume that the conditional probability has a Gaussian shape for each of the classes:

$$f_k(x) = \frac{1}{\sqrt{2\pi}\sigma_k} \exp\left(-\frac{1}{2\sigma_k^2}\left(x - \mu_k\right)^2\right)$$

Task a

By using the Bayes rule, can write down the *posterior* probability that the observation x belongs to class k, or $p_k(x) \equiv P(Y = k \mid X = x)$. I.e., reproduce Eq. (4.12) of James et al., but for the case where σ_k^2 may differ.

Task b

The Bayes classifier will assign an observation $X = x$ to the class for which $p_k(x)$ is the largest. Now, assume that all the classes have the same variance term σ^2.

Show Eq. (4.13) of James et al., i.e., that the classification problem is equivalent to

$$\arg\max_k \left(x \cdot \frac{\mu_k}{\sigma^2} - \frac{\mu_k^2}{2\sigma^2} + \log(\pi_k)\right),$$

Notice that when the variances are equal the quadratic term in x vanishes and the classification boundary is linear in x.

Problem 10

[15% points]

Objective: naive Bayes classifier

Download the iris dataset or use the iris dataset already included, e.g., in R or SciPy. You task is to build a Naive Bayes (NB) classifier for a binary classification task of classifying the flower species as `virginica` or `notvirginica` (versicolor or setosa) based on the morphological measurements of the flowers, i.e., the four covariates; see the code below. The data set has 50 flowers of class `virginica` and 100 flowers of class `notvirginica`. You classifier should use normal distributions to model the covariates.

Task a

Split the dataset in random into training and test sets of equal sizes. Use *stratified sampling*: first sample into training set 25 items in random from class `virginica` and then 50 items from class `notvirginica`. The remaining 75 items (25 from `virginica` and 50 from `notvirginica`) form your test set. Use the training set to find your model and test set to compute predictions. (Stratified sampling helps to avoid extra variance due to class imbalances in the training set, which may be issue especially for small data sets.)

Task b

Compute and report the means and standard deviations of each of the attributes in the training set for the both classes separately. Estimate and report the class probabilities, using Laplace smoothing *with pseudocount* of 1 on the training set. I.e., you should produce a total of 18 numbers from this task.

Task c

Now you can find the class-specific expressions for $p(x \mid y)$ needed by the NB classifier. Remember that according to NB assumption the dimensions are independent, and hence, you can represent the class-specific $p(x \mid y)$ likelihoods as products of 4 1-dimensional normal distributions. Write down the formula needed to compute the posterior probability of the class being `virginica` $\hat{p}(y = \text{virginica} \mid x)$ as a function of the four morphological measurements in x and the statistics (means, standard deviations, class probabilities) you computed in the task b above.

Task d

Compute and report the classification accuracy for you classifier on the test set.

Hint: When computing classification accuracy you can use the following rule to obtain "hard" classes,

$$\hat{y} = \begin{cases} \text{virginica} & , \quad \hat{p}(y = \text{virginica} \mid x) \geq 0.5 \\ \text{notvirginica} & , \quad \hat{p}(y = \text{virginica} \mid x) < 0.5 \end{cases}$$

Task e

Repeat task d above, but with logistic regression. (Logistic regression may give you a warning about convergence. Can you explain why if this happens?) Make two plots: one for the training data and one for test data where the x-axis value is the linear response (you get this directly out of `predict`, if you use R; see the code below) and the y-axis value is 0 for the class `virginica` and 1 for the class `notvirginica`.

```
library(GGally)
## Create new data with Species in "virginica" or "notvirginica"
data2 <- iris
data2$Species <- factor("virginica",levels=c("virginica","notvirginica"))
data2$Species[iris$Species!="virginica"] <- "notvirginica"
ggpairs(data2,aes(color=Species,alpha=0.4),
        upper=list(continuous=wrap("cor", size=2)))+
  theme(text=element_text(size=7))
```

Hint to task e: If you use R: You can do logistic regression with `glm`. Also see examples in James et al.!

```
## Assume your training data is in data2_train and test data
## in data2_test.

## First train a logistic regression model m.
m <- glm(Species ~ .,data2_train,family=binomial)

## "link" (the default output of predict, see R help page: ?predict.glm)
## is the linear response (the term \beta^Tx). The predicted
## probability would be sigmoid of this response or \sigma(\beta^Tx).
link <- predict(m,data2_test)

## Predictions (probability of 0.5 is at link==0):
## (PS. I hope I got the classes the right way...)
yhat <- ifelse(link<0,"virginica","notvirginica")

## Summary of results:
summary(m)
```

Problem 11

[15% points]

Objective: Understanding discriminative vs. generative learning.

Download Ng et al. (2001). You **do not need to read the full paper** in detail or understand all of the details! Rather try to find the answers to the following questions.

Reference: Ng, Jordan (2001) On discriminative vs. generative classifiers: A comparison of logistic regression and naive Bayes. NIPS. http://papers.nips.cc/paper/2020-on-discriminative-vs-generative-classifiers-a-comparison-of-logistic-regression-and-naive-bayes.pdf

Task a

Read the *abstract* and the *Introduction* (Sec. 1). Is discriminative learning better than generative learning, according to the authors? Justify your answer.

Task b

By a "parametric family of probabilistic models", the authors mean a set of distributions, where each distribution is defined by a set of parameters. An example of such a family is our friend, the family of normal distributions where the parameters are μ and Σ. Ng and Jordan denote by h_{Gen} and h_{Dis} two models chosen by optimizing different things. Find an explanation of what these "things" are that are being optimized in the paper, i.e., what characterizes these two models. Which two families do the authors discuss, and what are the (h_{Gen}, h_{Dis}) pairs for those models?

Task c

Study Figure 1 in the paper. Explain what it suggests (see the last paragraph of the Introduction). Reflect this on the previous item.

Problem 12

[20% points]

Objective: comparing classifiers on synthetic data, application of different classifiers

Consider a toy data with a binary class variable $y \in \{0, 1\}$ and with two discrete features $x_1 \in \{0, 1\}$ and $x_2 \in \{0, 1, 2\}$. The true distribution from which the data is sampled is such that $P(y = 1) = 1 - P(y = 0) = 0.6$ and the class-conditioned distributions for (x_1, x_2) are specified as follows:

$$P(x_1, x_2 \mid y = 0) = \begin{cases} 0.2 & , \quad x_1 = 0 \wedge x_2 = 0 \\ 0.4 & , \quad x_1 = 0 \wedge x_2 = 1 \\ 0 & , \quad x_1 = 0 \wedge x_2 = 2 \\ 0.1 & , \quad x_1 = 1 \wedge x_2 = 0 \\ 0.2 & , \quad x_1 = 1 \wedge x_2 = 1 \\ 0.1 & , \quad x_1 = 1 \wedge x_2 = 2 \end{cases}$$

and

$$P(x_1, x_2 \mid y = 1) = \begin{cases} 0.6 & , \quad x_1 = 0 \wedge x_2 = 0 \\ 0.1 & , \quad x_1 = 0 \wedge x_2 = 1 \\ 0.1 & , \quad x_1 = 0 \wedge x_2 = 2 \\ 0.1 & , \quad x_1 = 1 \wedge x_2 = 0 \\ 0.1 & , \quad x_1 = 1 \wedge x_2 = 1 \\ 0 & , \quad x_1 = 1 \wedge x_2 = 2 \end{cases}$$

Task a

Is the naive Bayes (NB) assumption valid for data generated by this procedure? Explain why or why not.

Task b

Generate a test data set of 10000 points and 10 training data sets of sizes $n \in \{2^4, 2^5, \ldots, 2^{13}\}$, respectively.

For each of the training set, train the following classifiers that output probabilities.

- Naive Bayes (NB) (e.g., `naiveBayes` from library `e1071`)

- Logistic regression without interaction term (e.g., `glm`)
- Logistic regression with interaction term (e.g., `glm`)
- Support Vector Machine (SVM) with radial basis function (e.g., `svm` from library `e1071`)
- Bayes classifier that uses the true class conditional probabilities (that you know in this case!) - no probabilistic classifier can do better than this
- "Dummy classifier" that does not depend of x and that always outputs probability $\hat{p}(y = 1 \mid x_1, x_2) = 0.6$ ("dummy" means here that the classifier output does not depend on the covariates, it is always a good idea to include dummy classifier in your comparison!)

Make a plot - or table - where you show for different models the classification accuracy and perplexity on the test set as a function of training data set size.

Hint: Perplexity is another and perhaps easier-to-interpret way to express log-likelihood. Perplexity on the test set of size $n = 10000$ is $\exp(-L/n)$, where L is the log-likelihood given by $L = \sum_{i=1}^{n} \log p(y_i \mid x_i)$ and (x_i, y_i) are the data items in the test set. The perplexity is 2 for coin flipping for which $p(y_i \mid x_i) = 0.5$ and 1 for a "perfect classifier" that always gives $p(y_i \mid x_i) = 1$.

Hint: Most of the code needed is given below in the instructions.

Task c

Discuss your observations and what you can conclude. Which of the models above are probabilistic, discriminative, and generative? How do accuracy and perplexity (log-likelihood) compare? Is there a relation to the insights from Problem 11 above? Why does logistic regression with the interaction term perform so well for larger datasets? Does your dummy classifier ever outperform other classifiers, or do other classifiers ever outperform the Bayes classifier?

Instructions

The following R function should provide you the needed datasets:

```
set.seed(42)
makedata <- function(n) {
  a <- data.frame(y =factor(c(0,0,0,0,0,0,1,1,1,1,1,1)),
                  x1=factor(c(0,0,0,1,1,1,0,0,0,1,1,1)),
                  x2=factor(c(0,1,2,0,1,2,0,1,2,0,1,2)))
  p <- c(0.4*c(0.2,0.4,0.0,0.1,0.2,0.1),
         0.6*c(0.6,0.1,0.1,0.1,0.1,0.0))
  a[sample.int(12,n,replace=TRUE,prob=p),,drop=FALSE]
}
data_test <- makedata(10000) # test data set
data <- lapply(2^(4:13),makedata) # training data sets
```

You can include the interaction term in logistic regression by writing, e.g.,:

```
model <- glm(y ~ x1*x2,data[[4]],family=binomial)
```

"*" implies that an interaction should be included in the model, instead of "+" which assumes only additive effects.

Here, it is useful to make a function `phat` that takes the training and testing data as input and outputs the the probabilities $p(y = 1 \mid x_1, x_2)$ and then computes accuracy and perplexity by using theses probabilities. Below, you can find one way to accomplish this, if you use R. You can of course do the problem in some other way or language as well.

```
library(e1071)
## estimate phat. The idea is to make a function that outputs the predicted
## probabilities and the parameters can be modified "easily" for different
```

```
## models.
phat <- function(data.tr, # training data
                 data.te=data_test, # test data - here data_test
                 form=y ~ x1+x2, # formula
                 model=function(f,d) naiveBayes(f,d,laplace=1), # model family
                 m=model(form,data.tr), # model trained on data.tr
                 pred=function(m,x) predict(m,x,type="raw")[,2]) { # function to make the predictions b
  pred(m,data.te)
}

## accuracy
acc <- function(p,y=data_test$y) mean(ifelse(p>=0.5,1,0)==y)

## perplexity
perp <- function(p,y=data_test$y) exp(-mean(log(ifelse(y==1,p,1-p))))
```

Then you can for example collect the requested numbers for NB into dataframe `res`:

```
# collect results to data frames
res <-  data.frame(n=sapply(data,nrow))

phat_nb <- lapply(data,phat)
res$nb_acc  <- sapply(phat_nb,acc)
res$nb_perp <- sapply(phat_nb,perp)
```

Now you have your results for the NB. Then it remains only to do the same for all other classifiers, which is easy because you just use different libraries (but unfortunately, there are slight differences in model options) - the SciPy sklearn is a bit nicer in this respect.

Probabilities for logistic regression:

```
phat_glm <- lapply(data,function(d) {
  phat(d,
       model=function(f,d) glm(f,d,family=binomial),
       pred=function(m,x) predict(m,x,type="response"))
})
```

You can compute the accuracies and perplexities the same way you did for the NB above by using `phat_glm` instead of `phat_nb`. Logistic regression with interaction:

```
phat_glmx <- lapply(data,function(d) {
  phat(d,
       form=y ~ x1*x2,
       model=function(f,d) glm(f,d,family=binomial),
       pred=function(m,x) predict(m,x,type="response"))
})
```

SVM:

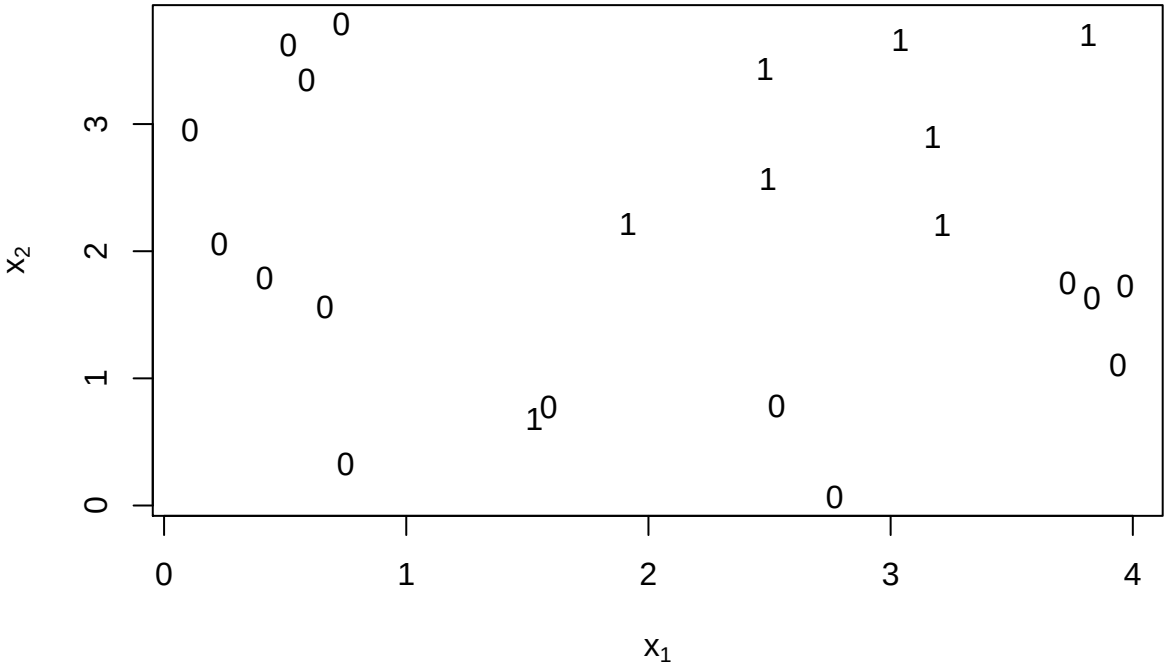
```
phat_svm <- lapply(data,function(d) {
  phat(d,
       model=function(...) svm(...,probability=TRUE),
       pred=function(m,x) attr(predict(m,x,probability=TRUE),"probabilities")[,"1"])
})
```

Problem 13

[15% points]

Objectives: basic principles of decision trees

Familiarise yourself with Section 8.1 of James et al. Pick one of the impurity measures presented in Equations (8.5), (8.6), or (8.7). Then simulate the tree building algorithm by hand.



Task a

Sketch a run of the classification tree algorithm with the toy data set in the figure above(binary classification task in $\mathbb{R}^2$) and draw the resulting classification tree. Report the values of the chosen impurity measure for each split and try to choose the splits that obtain the best impurity measure. You do not need to worry about over-fitting here: the resulting classification tree should fit the training data with no error. Don't worry that you don't count the classes exactly right or that your results are not super-accurate as long as they are "in the ballpark".

Problem 14

[15% points]

Learning objectives: basics of k-NN method.

Consider the problem of applying *k-nearest neighbour* (k-NN) classifier to the training dataset $D = \{(x_i, c_i)\}_{i=1}^{14}$, where the covariates $x_i \in \mathbb{R}$ and the classes $c_i \in \{-1, +1\}$ are given in below. You should be able to do this by pen and paper.

i	x_i	c_i
1	0	$+1$
2	2	$+1$
3	3	$+1$
4	5	-1
5	6	$+1$
6	8	$+1$
7	9	$+1$
8	12	-1
9	13	-1
10	15	-1
11	16	$+1$
12	18	-1
13	19	-1
14	21	-1

Task a

Where are the classification boundaries for the 1-NN and 3-NN classifiers? What are the respective classification errors on the training dataset?

Task b

How does the choice of k in k-NN affect the classification boundary (not in the above example but in general)? Give examples of the behaviour for extreme choices.

Problem 15

[5% points]

Objectives: self-reflection, giving feedback of the course

Tasks

- Write a learning diary of the topics of lectures 5-8 and this exercise set.

Instructions

The length of your reply should be 1-3 paragraphs of text. Guiding questions: What did I learn? What did I not understand? Was there something relevant for other studies or (future) work?