MAD 2021/2022 Exam

Bulat Ibragimov, Kim Steenstrup Pedersen

Exam period: 17.01.2022 – 25.1.2022

This is the exam questions for the 8 day take-home exam on the course Modeling and Analysis of Data (MAD). The exam must be solved **individually**, i.e., you are **not allowed to work in teams or to discuss the exam questions with other students**. You have to submit your solution of the exam electronically via the **Digital Exam**[1] system. The submission deadline is **25 January 2022 at 22:00**.
Your solution should consist of

1. a pdf file `answers.pdf` containing your answers, and
2. a `code.zip` file containing the associated code (python files and / or Python Jupyter notebooks).

The grading will be done anonymously; **hence you should not mention your name anywhere in the answers or code**. Instead, you must **add your exam number** at the beginning of your `answers.pdf`.

WARNING: The goal of this exam is to evaluate your individual skills. We have to report any suspicion of cheating, in particular collaboration with other students, to the head of studies. Note that, if proven guilty, you may be expelled from the university. Do not put yourself and your fellow students at risk.

You are allowed to ask questions via the Discussions board in Absalon, but make sure that you do not reveal any significant parts of the solution. In doubt, just contact Bulat or Kim directly via e-mail! Any additional hints given by us will be made available to all of you via Absalon. Some further comments:

1. You **are allowed** to reuse the code that was made available to you via Absalon as well as the code you have developed in the course of the assignments. If you reuse code from the lectures or from the assignments, make sure to put a reference to this in your code, and if your code was developed as part of an assignment, in collaboration with a fellow student, add a corresponding comment to your answers, although keeping anonymity (i.e., just mention which parts stem from team work). In case you reuse code snippets you have found on the internet, please make sure that you provide a reference to this external source as well.
2. In case you notice any inaccuracies in the problem descriptions below, please let us know. If needed, we will provide updates and additional comments via Absalon. Thus, make sure that you check Absalon for announcements and discussions regularly!
3. For the coding tasks, you are given Python files/Jupyter notebook templates. You are supposed to complete these files and notebooks, but you are allowed to convert a Jupyter notebook to a python file or the opposite. Note that you are allowed to import additional Python packages and to make use of the functions provided by, e.g., the Numpy package. However, you should not use built-in functions if we ask you to implement a specific algorithm without using existing implementations (e.g., a single `kmeans` function from some other package that implements the K-means clustering approach). If in doubt, please ask us via the discussion board in Absalon!
4. All code templates and data can be found in the files `code.zip` and `data.zip`.
5. The deadline is hard (late submissions are not allowed), so make sure to submit in good time before the deadline. Up until the deadline it is possible to upload new versions of your solution several times. In the unlikely event that the Digital Exam system fails when you submit just at the deadline, then immediately send an e-mail to `uddannelse@diku.dk` and `bulat@di.ku.dk` with an explanation of what happened and attach your solution (the pdf and zip files) to the exam. Your solution will be assessed if you have a valid excuse and submitted on time.
6. Good luck! :-)

[1] `https://eksamen.ku.dk/`

Statistics

In this part, we will test your knowledge and skills in performing statistical analysis.

Question 1 (Maximum Likelihood Estimation, 1 points). Let X be a continuous random variable with the probability density function given below and having a parameter $\theta \in \mathbb{R}_+$ (the set of positive real numbers),

$$f(x;\theta) = \begin{cases} \frac{\theta}{x^2} \cdot \exp\left(-\frac{\theta}{x}\right) & \text{for } x \in \mathbb{R}_+ \\ 0 & \text{otherwise.} \end{cases}$$

Prove that the maximum likelihood estimate $\hat{\theta}$ for the parameter θ is given by

$$\hat{\theta} = \frac{N}{\frac{1}{x_1} + \frac{1}{x_2} + \ldots + \frac{1}{x_N}} \,.$$

Deliverables. Provide the essential steps and argumentation in your proof.
Instructions for grading:
0.25 points for correctly writing up the likelihood function. 0.25 points if takes derivatives with respect to θ and set equal to zero, but something goes wrong. A percentage of the remaining 0.4 points based on how correct the solution is. Give 0.1 point if the check for maximum (i.e. second derivative check) is considered.
Solution:
We will make the usual assumption of a i.i.id. dataset sampled from the distribution in question, and we can therefore write the following likelihood function

$$L(x_1, \ldots, x_N|\theta) = \prod_{n=1}^{N} f(x_n|\theta) \text{ for } x > 0,$$

otherwise, if $x_n \leq 0$ for any n, then $L(x_1, \ldots, x_N|\theta) = 0$. So we only need to consider the case of $x > 0$ in the following.

We will use the usual trick of using the log-likelihood for finding the ML estimate $\hat{\theta}$ (because the PDF involves products and exponential functions of the parameter). First we compute

$$\log f(x|\theta) = \log\theta - \log x^2 - \frac{\theta}{x} \,.$$

The log-likelihood can then be written as

$$\log L(x_1, \ldots, x_N|\theta) = \sum_{n=1}^{N} \log f(x_n|\theta) = N \log\theta - \sum_{n=1}^{N} \left(\log x_n^2 + \frac{\theta}{x_n}\right) \,.$$

Now we just need to compute the first partial derivative of the log-likelihood function with respect to θ

$$\frac{\partial \log L}{\partial \theta} = -\frac{\partial}{\partial \theta}\left(N \log\theta - \sum_{n=1}^{N}\left(\log x_n^2 + \frac{\theta}{x_n}\right)\right) = \frac{N}{\theta} - \sum_{n=1}^{N} \frac{1}{x_n} \,.$$

Now we equate this derivative with 0 and isolate θ to obtain the ML estimate

$$\frac{\partial \log L}{\partial \theta} = 0 \quad \Rightarrow \quad \frac{N}{\theta} - \sum_{n=1}^{N} \frac{1}{x_n} = 0 \Rightarrow$$

$$\frac{N}{\theta} = \sum_{n=1}^{N} \frac{1}{x_n} \quad \Rightarrow \quad \theta = \frac{N}{\sum_{n=1}^{N} \frac{1}{x_n}} \,.$$

Therefore the ML estimate is

$$\hat{\theta} = \frac{N}{\frac{1}{x_1} + \frac{1}{x_2} + \ldots + \frac{1}{x_N}} \,.$$

To verify that it is indeed a maximum we need to compute the second derivative of the log-likelihood with respect to θ,

$$\frac{\partial^2 \log L}{\partial \theta^2} = \frac{\partial}{\partial \theta}\left(\frac{N}{\theta} - \sum_{n=1}^{N} \frac{1}{x_n}\right) = -\frac{N}{\theta^2} < 0 \,,$$

which is always negative for all θ, since $N > 0$ (recall also that $\theta > 0$, but is not important here), hence $\hat{\theta}$ is a maximum for the likelihood function. QED.

Question 2 (**Hypothesis testing, 2 points).** Assume we have a binary classification algorithm for which we know that the classification accuracy A (i.e. the probability of correct classification in percentage) is distributed as a normal distribution $A \sim \mathcal{N}(\mu_0 = 91, \sigma_0^2 = 4)$ (we obtained this knowledge by cross-validation). We are not satisfied with a mean accuracy of 91%, so we have implemented a new classification algorithm that we hope will provide a higher accuracy. We conduct 5-fold cross validation with the new algorithm and obtain the following classification accuracies for each split of the dataset

$$\{88, 93, 94, 94, 95\} \ .$$

In the following we will assume that the accuracy of the new algorithm is also normal distributed $A_{\text{new}} \sim \mathcal{N}(\mu, \sigma^2)$.

Perform a hypothesis test at 5%-significance level to investigate if the new algorithms mean accuracy μ is greater than our old algorithms accuracy of $\mu_0 = 91\%$. For the new algorithm, both the mean μ and variance σ^2 of the classification accuracy is unknown.

Deliverables. Provide the steps you follow in order to perform the test as well as your conclusion, i.e. is the new algorithm significantly more accurate than the old algorithm?

Instructions for grading:

2 points in total. Deduct 0.5 points if the definition of null and alternative hypoteses are wrong. Deduct 1 point if a t-test is not performed. Deduct fractional points depending on mistakes made in the 6 steps.

Solution:

I apply the six steps to perform the hypothesis test:

1. Formulate a model of the data: This is already given in the question - we assume that $A \sim \mathcal{N}(\mu, \sigma^2)$ where μ and σ^2 are unknown.
2. The hypotheses: The null hypothesis $H_0 : \mu = \mu_0$ (the worst case) against the alternative hypothesis $H_A : \mu > \mu_0$ (what we hope for). This means that we have to perform a right-sided test.
3. Test statistics to use: We should consider the sample mean $\bar{A}$ as the test statistics which according to the model follows a normal distribution. Both the mean μ and variance σ^2 are unknown, so we should consider the sample mean $\bar{X}$, sample variance S^2 and using $\mu_0 = 91$ to compute the standardized test statistics which follows a t distribution with degree $N - 1 =$

$$T = \frac{\bar{X} - \mu_0}{S/\sqrt{n}} \sim t(N-1) \ .$$

4. Choose a significance level: This is already given in the text as 5%-significance levels, $\alpha = 5\%$.
5. Compute rejection region: Since we have realised that it is a right-sided test then we know from the alternative hypothesis that $P(\mu > \mu_0) = 1 - \alpha$ and we have the rejection region $R =]c; \infty[$ for the standardized test statistics and $P(T > c) = 1 - \alpha$, which gives by inverse look-up in the t distribution CDF $c = F^{-1}(1 - \alpha) = 2.13$ (computed using `c = scipy.stats.t.ppf(0.95, 4)`). The rejection region therefore is rejection region $R =]2.13; \infty[$
6. We now evaluate the test statistics on the dataset with $n = 5$, by first computing the sample mean to $\bar{a} = 92.8$, sample standard deviation $s = 2.48$, and the standardized statistics

$$t = \frac{\bar{x} - \mu_0}{S/\sqrt{n}} = \frac{92.8 - 91}{2.48/\sqrt{5}} = 1.62 \ .$$

   This means that we cannot reject the null hypothesis, because t is not in the rejection region, $t \notin]2.13; \infty[$. Therefore the new algorithm does not appear to be significantly better than the old algorithm (at least from the experiments we conducted).

Question 3 (**Confidence intervals, 2 points).** Lets assume we are given the following samples from a normal distributed random variable $X \sim \mathcal{N}(\mu, \sigma^2)$ where μ is unknown,

$$\{56.6, 59.0, 53.2, 66.1, 51.3, 50.4, 53.5, 44.5, 46.3, 60.3\} \ .$$

Find the 95%-confidence interval for the parameter μ, in the cases

a) we know that $\sigma = 5.0$,
b) σ is unknown.

Deliverables. For both a) and b), besides the confidence interval, provide the argumentation for and steps you follow in order to find the confidence intervals.

Instructions for grading:

a) 1 point, b) 1 point. In both cases deduct a fraction of points depending on severity of the mistakes made. For a), deduct points if a t-test is used instead of realising that the variance is known. The level of detail provided in the solution is acceptable (no need to prove the form of the final form of the interval since both were proven in lectures).

Solution:

In a) we know the standard deviation $\sigma = 5.0$ and the mean μ is unknown and this means that I can perform the 4 steps to compute the confidence interval in this case:

1. Choose a confidence level: This is already given in the question as $\gamma = 95\%$.
2. Since we know that the standardized test statistics is standard normal distributed

$$Z = \frac{\bar{X} - \mu_0}{\sigma/\sqrt{n}} \sim \mathcal{N}(0,1) \ ,$$

   we can compute the critical value by inverse lookup in the CDF of the standard normal distribution $c = \Phi(^{-1}(1+\gamma)/2) = 1.96$ (computed using `c = scipy.stats.norm.ppf(0.975)`).
3. Compute the sample mean $\bar{x} = 54.12$.
4. Since $n = 10$ for this data set we have the confidence interval $[\bar{x} - c\frac{\sigma}{\sqrt{n}}; \bar{x} + c\frac{\sigma}{\sqrt{n}}] = [51.0; 57.2]$ in which the true μ is with probability 95%. or formulated more compactly

$$P\left(\mu \in [51.0; 57.2]\right) = 95\% \ .$$

In b) both the mean μ and standard deviation σ is unknown. Again we perform the 4 steps:

1. Choose a confidence level: This is already given in the question as $\gamma = 95\%$.
2. In this case we need to use both the sample mean $\bar{X}$ and sample variance S to form the standardized test statistics which is t-distributed with degree $n - 1$

$$T = \frac{\bar{X} - \mu_0}{S/\sqrt{n}} \sim f_T(t; n-1) \ .$$

   This means that to find the critical value we need to do the inverse of the CDF of this t-distribution, $c = F^{-1}(1+\gamma)/2) = 2.26$ (computed using `c = scipy.stats.t.ppf(0.975, df=9)`).
3. Compute the sample mean $\bar{x} = 54.12$ of the data set as well as the sample standard deviation $s = 6.57$.
4. Since $n = 10$ for this data set we have the confidence interval $[\bar{x} - c\frac{s}{\sqrt{n}}; \bar{x} + c\frac{s}{\sqrt{n}}] = [49.4; 58.8]$ in which the true μ is with probability 95%. or formulated more compactly

$$P\left(\mu \in [49.4; 58.8]\right) = 95\% \ .$$

   Hence we see that the interval is wider in the case of unknown variance compared to the interval from a) - we are more uncertain about the interval the true μ falls in.

Regression

In this part, we will test your knowledge and skills in performing regression analysis of data.

Question 4 (**The Hot Air Balloon, 8 points).** A high school student is making a physics and mathematics project in which he must build a hot air balloon and investigate the relations between how hot the air inside the balloon must be in order for the balloon to lift off with different weights attached to it. He has performed several experiments with different weights and repeatedly measured the temperature inside the balloon when it lifts off the ground. He has noted down the temperatures in Kelvin and the lift ability (bouyancy) measured in grams. We will help him do the data analysis of his experimental data. For this, he has provided us with a file, `hot-balloon-data.csv`, which contains comma-separated lines for each measurement pairs, where the first column is the lift ability (in grams) and the second column is the temperature (in Kelvin). We provide a code template in the Python script `regression.py` that you can use as starting point for your solution.

The student is interested in modelling the temperature as a function of the lift ability and we will help him by performing regression. We will consider two non-linear response regression models of the data; a K'th order polynomial

$$f(x, \mathbf{w}) = \sum_{k=0}^{K} w_k x^k \tag{1}$$

and a logarithmic model

$$f(x, \mathbf{w}) = w_0 + w_1 \log x \ . \tag{2}$$

Both of these models provide a non-linear response, but are linear models in the parameters $\mathbf{w}$. This means that we can use the same approaches as for regression with linear models, if we write the data matrix $\mathbf{X}$ using basis functions $h_0(x), h_1(x), \ldots$ applied to the input data vectors x_n from the training dataset $\{(t_1, x_1), \ldots, (t_N, x_N)\}$.

We will assume that the noise in the temperature measurements (based on the quality of the thermometer and the way the student collected the measurements) is additive, independent, and identically Gaussian distributed with, $\varepsilon \sim \mathcal{N}(0, \sigma^2)$. Therefore, we will model the observed temperature by the generative model, $t = f(x, \mathbf{w}) + \varepsilon$.

Solve the following problems:

a) (1 point) Write and explain the mathematical expressions for the data matrix $\mathbf{X}$ for the two non-linear models, (1) and (2), such that we can use the same regression approaches as for linear models. Hint: Start by indentifying the basis functions $h_0(x), h_1(x), \ldots$ needed for the two models and then write up the expression for $\mathbf{X}$.

b) (1 point) Derive the mathematical expression for the likelihood function for the data set under our assumptions stated above. You can write the expression using the general notation $\mathbf{X}$ for the data matrix, so you do not need different likelihood functions for the two models (1) and (2).

c) (2 points) Write Python functions that implement your own maximum likelihood regression for both of the non-linear models (1) and (2). You must compute the estimated parameters $\hat{\mathbf{w}}$ as well as the estimated noise variance $\widehat{\sigma^2}$.

d) (1 point) Apply your implementation from c) to the dataset and report the estimated parameters $\hat{\mathbf{w}}$ and noise variance $\widehat{\sigma^2}$, as well as make a plot of the two estimated models together with the data points in the data set. For the polynomial model, you must choose order $K = 3$.

e) (2 point) Assume a Gaussian prior on the model parameters $p(\mathbf{w}|\mu_0, \sigma_0^2) = \mathcal{N}(\mu_0, \sigma_0^2\mathbf{I})$. In Python, implement Bayesian regression using this prior and the likelihood from b) for both of the non-linear models (1) and (2).

f) (1 point) Apply your implementation of Bayesian regression from e) to the data set. Use $\sigma^2 = 25$ in the likelihood, set the prior variance to $\sigma_0^2 = 10$ for both models, and the prior mean $\mu_0 = (268, 0, 0, 0)^T$ for the polynomial model, (1), and $\mu_0 = (133, 32)^T$ for the logarithmic model, (2). For each of the models, (1) and (2), make a plot of the mean of the predictive distribution and plus and minus its standard deviation curves together with the observations in the data set. We provide the Python function `predictiveplot` that makes such plots.

Deliverables. a) Provide argumentation and mathematical expressions for the data matrices, b) argumentation and a mathematical expression, c) code in python file and provide code snippets in the report highlighting the essential parts of your solution, d) add your estimated values and the plot to the report and comment on the results, e) code in python file and provide code snippets in the report highlighting the essential parts of your solution, f) two plots and your comments to the results.

Instructions for grading:

a) 1 point in total, each model is worth 0.5 points, b) 1 point and must include some arguments and steps for how the arrive at the likelihood function - deduct up to 0.5 points if this is missing but the likelihood function is written correctly, c) 2 points for a correct implementation for both models, deduct 0.5 points if they forget to include the noise variance estimate $\widehat{\sigma^2}$ otherwise some fraction based on how correct it is, d) 1 point if they answer this also if the implementation from c) is wrong, deduct up to 0.25 points if comments are missing, e) 2 points for a correct implementation and deduct a fraction if mistakes, f) 1 point if they answer this also if the implementation from e) is wrong, deduct up to 0.25 points if comments are missing. There was some miscommunication on my part on whether or not to pass predictive variance or standard deviation into `predictiveplot`, so both solutions are acceptable.

Solution:

We are given a data set $\{(t_1, x_1), \ldots, (t_N, x_N)\}$ where x is lift and t is temperature.

For a), we realize that we can write the data matrices for both models in terms of non-linear basis functions $h(x)$. The data matrix for the polynomial model $f(x, \mathbf{w}) = \sum_{k=0}^{K} w_k x^k$ can be written using the monomial

basis $h_k(x) = x^k$ as

$$\mathbf{X} = \begin{bmatrix} x_1^0 & x_1^1 & \cdots & x_1^K \\ x_2^0 & x_2^1 & \cdots & x_2^K \\ \cdots & \cdots & \cdots & \cdots \\ x_N^0 & x_N^1 & \cdots & x_N^K \end{bmatrix} = \begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^K \\ 1 & x_2 & x_2^2 & \cdots & x_2^K \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 1 & x_N & x_N^2 & \cdots & x_N^K \end{bmatrix} \in \mathbb{R}^{N \times K} ,$$

such that we can write the model as $f(x, \mathbf{w}) = \mathbf{X}\mathbf{w}$. Similarly for the logarithmic model $f(x, \mathbf{w}) = w_0 + w_1 \log x$,

$$\mathbf{X} = \begin{bmatrix} 1 & \log(x_1) \\ 1 & \log(x_2) \\ \cdots & \cdots \\ 1 & \log(x_N) \end{bmatrix} \in \mathbb{R}^{N \times 2} ,$$

and again the model can be written as $f(x, \mathbf{w}) = \mathbf{X}\mathbf{w}$.

b) The since we assume that the noise is i.i.d. zero mean Gaussian distributed, $\varepsilon \sim \mathcal{N}(0, \sigma^2)$, then the likelihood of a single data point (t, x) is Gaussian distributed

$$p(t|x, \sigma^2) = \mathcal{N}(\mathbf{w}^T\mathbf{x}, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{(t - \mathbf{w}^T\mathbf{x})^2}{2\sigma^2}\right) .$$

The likelihood function for the data set formulated in terms of the data matrix $\mathbf{X}$ and vector of observed temperatures (targets) $\mathbf{t}$ is for both non-linear models given by

$$p(\mathbf{t}|\mathbf{X}, \sigma^2) = \prod_{n=1}^{N} p(t_n|\mathbf{x}_n, \sigma^2) ,$$

where $\mathbf{x}_n$ denotes the n'th row of the data matrix $\mathbf{X}$. Using the expression for the likelihood of one data point we write

$$p(\mathbf{t}|\mathbf{X}, \sigma^2) = \prod_{n=1}^{N} \mathcal{N}(\mathbf{w}^T\mathbf{x}_n, \sigma^2) = \prod_{n=1}^{N} \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{(t_n - \mathbf{w}^T\mathbf{x}_n)^2}{2\sigma^2}\right) = \frac{1}{(\sqrt{2\pi\sigma})^N} \exp\left(-\sum_{n=1}^{N} \frac{(t_n - \mathbf{w}^T\mathbf{x}_n)^2}{2\sigma^2}\right) .$$

Which is a multivariate Gaussian distribution (its ok to skip lightly over this part and refer to the book), so we can simplify and write,

$$p(\mathbf{t}|\mathbf{X}, \sigma^2) = \mathcal{N}(\mathbf{X}\mathbf{w}, \sigma^2\mathbf{I}) ,$$

where $\mathbf{I} \in \mathbb{R}^{N \times N}$ is the identity matrix.

See reference solution in `regression/solution.py`.

Classification

In this part, we will test your knowledge and skills in performing classification of data.

Question 5 (Classification & Random Forests, 2 points). This question tests your understanding of random forest construction, in particular the information gain based on entropy and Gini index calculation. Let's say you are given the original set of samples that looks the following: 12 class1, 5 class2, 8 class3, 9 class4.

Let's say the original set of samples is split into two subset, where the 1st subset consists of 6 class1, 1 class2, 6 class3, 6 class4; and the 2nd subset consists of 6 class1, 4 class2, 2 class3, 3 class4. Please compute the information gain based on entropy and Gini index from the proposed split of samples.

a) (1 point) Gain based on Entropy:
b) (1 point) Gini:

Deliverables. For both a) and b), include your computed values as well as explanations on how you reach these values.

Instructions for grading:
Please award 2 points if the student correctly calculated *Gain*, *Gini*(*orig*), and *Index* for Gini. Subtract 0.5 points if the student forgot about *Index*. Subtract 0.5 points the student made some errors in calculations but correctly followed the order of operations for calculating the indices.

Solution:
Information Gain Step 1. Compute entropy of the original set and subsets:

$$H(orig) = -\frac{12}{34}log_2(\frac{12}{34}) - \frac{5}{34}log_2(\frac{5}{34}) - \frac{8}{34}log_2(\frac{8}{34}) - \frac{9}{34}log_2(\frac{9}{34}) = 1.9357$$

$$H(sub1) = -\frac{6}{19}log_2(\frac{6}{19}) - \frac{1}{19}log_2(\frac{1}{19}) - \frac{6}{19}log_2(\frac{6}{19}) - \frac{6}{19}log_2(\frac{6}{19}) = 1.7990$$

$$H(sub2) = -\frac{6}{15}log_2(\frac{6}{15}) - \frac{4}{15}log_2(\frac{4}{15}) - \frac{2}{15}log_2(\frac{2}{15}) - \frac{3}{15}log_2(\frac{3}{15}) = 1.8893$$

Information Gain Step 2. Compute the information gain for the complete separation:

$$Gain = H(orig) - \frac{19}{34}H(sub1) - \frac{15}{34}H(sub2) = 1.9357 - 1.0053 - 0.8335 = 0.0969$$

Gini Step 1. Compute Gini for subsets:

$$Gini(sub1) = 1 - \frac{6}{19}^2 - \frac{1}{19}^2 - \frac{6}{19}^2 - \frac{6}{19}^2 = 0.6981$$

$$Gini(sub2) = 1 - \frac{6}{15}^2 - \frac{4}{15}^2 - \frac{2}{15}^2 - \frac{3}{15}^2 = 0.7111$$

Gini Step 2. Compute Gini for the complete separation:

$$Index = \frac{19}{34}Gini(sub1) - \frac{15}{34}Gini(sub2) = 0.7038$$

Question 6 (Classification & Validation, 4 points). This task aims to test your understanding on the training/validation data separation and classifier parameter tuning. Use training data from `accent-mfcc-data_shuffled_train.txt` and validation data from `accent-mfcc-data_shuffled_validation.txt`. The data corresponds to samples of human speech performed for individuals with different mother languages, e.g. English, Spanish, German etc. Both files are in comma- separated format and can be read by modifying the `loaddata` function from `regression.py` to accommedate the columns of these files. Each line in the file represents one speech sample. The first value of a line is the mother language label. The remaining 12 values denoted X1-X12 are numerical features of the speech sample.

Use the implementation of Random forests from the `sklearn` Python package to do the following:

a) (1 point) Implement random forest training using the data from `accent-mfcc-data_shuffled_train.txt`. Use features X1-X12 to predict the "language" label. Report the accuracy of the random forest model on the training data. The accuracy is the number of correctly predicted languages divided to the total number of samples (accuracy lies between 0 to 1).

b) (2 points) Use the validation data to find the optimal set of random forest classifier parameters. The parameters to test: 1) criterion (entropy or gini); 2) maximal tree depth (values to test 2, 5, 7, 10, 15); 3) the number of features to consider for each split (*sqrt* and log_2 of the total number of features). To find the optimal set of parameters, perform an exhaustive search over all possible combinations of parameters from 1), 2) and 3). Calculate and print two metrics during the optimal parameter search. The first metric is the number of correctly classified validation samples. The second metric is the average probability assigned to the correct class for all validation samples. Use the first metric to select the optimal set of parameters. If two sets of parameters have the same value of correctly classified validation samples, i.e. the first metric is the same for both sets, the set with the highest second metric is considered more optimal. Add into report the code fragment that calculates both metrics and updates the optimal parameter set if these metrics are superior to the metrics for the previously found optimal set.

c) (1 point) Add into the report the accuracy improvements during the optimal parameter search. Every time you find a more optimal set of parameters, print a statement "criterion = ? ; max_depth = ? ; max_features = ? ; accuracy on validation data = ? ; number of correctly classified validation samples = ?". Replace question marks with the appropriate values.

Deliverables. a) Provide a code snippet in the report of the essential steps, b) code snippet in the report of the essential steps, c) provide the obtained results in the report as well as your reflections on these.

Instructions for grading:

a) award 1 point if the student correctly used existing or their own implementation of random forests for data classification. Subtract 0.5 points if the student figured out how to use random forest implementation messed up the data (mixed features with labels, columns with rows etc); b-c) Full points if the parameter search is exhaustive and printing is more or less in agreement with what was requested. Subtract 0.5 point if situation when the first metric is equal for two parameter sets is not handled. Subtract 1 points if their parameter search is not exhaustive and they do not properly arrange for loops. Using existing parameter optimization tools is OK, but they should have correct printing statements. Subtract 1 point if printing statements are not there or not complete because of the use of some existing parameter optimization tools.

Solution:

Please check the corresponding python code with the solution example.

Clustering

In this part, we will test your knowledge and skills in performing clustering of data.

Question 7 (K-means Clustering & Principal Component Analysis, 7 points). This task aims to test your understanding of clustering and principal component analysis. The task includes the following:

a) (1 point) Read and normalize data from the comma-separated data file `seedsDataset.txt`. You can read the file by modifying the `loaddata` function from `regression.py` to accommodate the columns of this file. The dataset consist of 200 samples. Each line in the file represent one sample. Each sample is defined with 8 features X1-X8. Normalize the data using the mean and standard deviation values computed for each feature. That is, for the i'th sample, the normalized X1 feature value is $X1_i^{norm} = (X1_i - \bar{X1})/(\sigma_{X1})$, where $X1_i$ is the original X1 feature value for the i'th sample, $\bar{X1}$ is the mean value of feature X1 computed from all samples, and σ_{X1} is the standard deviation of feature X1 computed from all samples.

b) (2.5 points) Implement K-means clustering without using any existing implementation. Test your K-means implementation using the normalized data from a) and set number of clusters to $K = 3$. Run K-means 5 times using random samples from the dataset as the initial cluster centers. Select the solution with the smallest intra-cluster distance.

c) (0.5 point) Compute and print the number of samples in each cluster in the final solution you obtained in b).

d) (1 point) Compute the principal components of the data. You are allowed to use existing implementations of the principal component analysis.

e) (2 points) Transform the data using the two largest components, i.e. eigenvectors with the largest eigenvalues. Plot the clustering results including the transformed cluster centers and transformed data. Your plot should contain 200 colored dots representing data samples and 3 black dots representing cluster centers. The dot associated with i'th sample should have coordinates $(y, z) = PCA_2(X1_i, X2_i, ..., X8_i)$, where PCA_2 is the transformation defined by projection onto the two largest components, and the color defined by the cluster label of i'th sample obtained in b). In figure 1, you find an example of such a plot. Note that you should run clustering on the original 8-dimensional data, save the clustering labels in some array, transform the data to 2-dimensional space and then plot the data samples colored according to the saved clustering labels.

Deliverables. a) Provide a code snippet in the report showing how you perform the normalization, b) provide a code snippet in the report showing your K-means implementation, c) include the number of samples in each cluster in the report, d) provide a code snippet and explanation of what you did in the report, e) include a code snippet, the plot and your reflections on the results in the report.

Instructions for grading:

a) 1 point if reading and normalization is performed. It is OK if they used some other normalization, as long as they justify their decision. subtract 0.5 points if no normalization is done; b) Subtract 1.5 points if they did not implement k-means but used some build-in implementation from sklearn, or similar. Do not subtract any points if they found and adapted some implementation from the internet, but subtract 0.5 points if they did not completely adapt it and left some operations that are irrelevant to the problem they face. Subtract 0.5 points if K-means does not run on normalized data. Subtract 1 point if they did not do 5 tries to find the optimal set of parameters. Subtract 0.5 points if they did 5 tries but something was wrong in selection of initial centers. Subtract 0.5 points if the rule for selection of the optimal solution among 5 tries does not work correctly. For example, if using the sum of intra-cluster distances or the mean of intra-cluster distances is equally OK. Do not subtract points if they used something else than intra-cluster distance, which however does the optimal solution selection adequately. c) Obvious; d) 1 point if PCA correctly applied on the data. Please be sure that the signs of principal components do not matter. Subtract 0.5 points if everything is more or less correct, but eigenvalues or principal components are wrong. e) Subtract 1 point if transformation is not correct, i.e. the points are not visually separated on 2D space. Subtract 0.5 points if they used clustering label during normalization of PCA. The label should be used as a color.

Solution:

Please check the corresponding python code with the solution example.

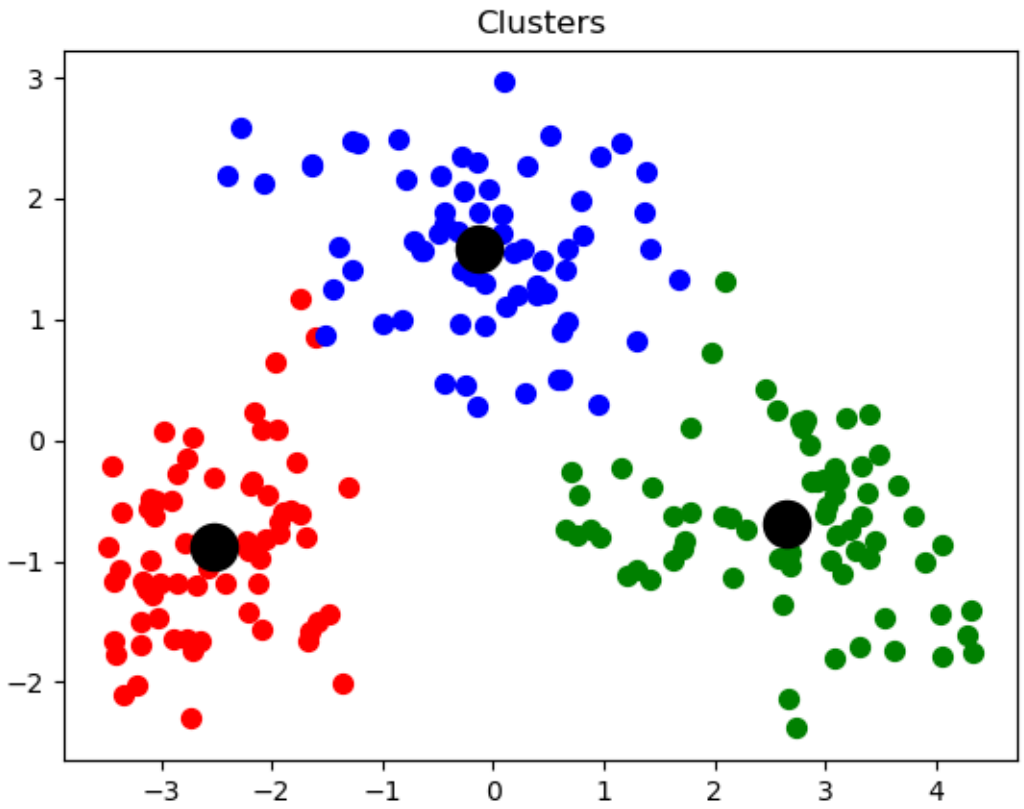

Figure 1: Example of plot we ask for in question 7(e). The black dots represents the cluster centers.