Exam

August 19, 2025

1 Exam 21th of August 2025, 8.00-13.00 for the course 1MS041 (Introduction to Data Science / Introduktion till dataanalys)

1.1 Instructions:

1. Complete the problems by following instructions.
2. When done, submit this file with your solutions saved, following the instruction sheet.

This exam has 3 problems for a total of 40 points, to pass you need 20 points. The bonus will be added to the score of the exam and rounded afterwards.

1.2 Some general hints and information:

- Try to answer all questions even if you are uncertain.
- Comment your code, so that if you get the wrong answer I can understand how you thought this can give you some points even though the code does not run.
- Follow the instruction sheet rigorously.
- This exam is partially autograded, but your code and your free text answers are manually graded anonymously.
- If there are any questions, please ask the exam guards, they will escalate it to me if necessary.

1.3 Tips for free text answers

- Be VERY clear with your reasoning, there should be zero ambiguity in what you are referring to.
- If you want to include math, you can write LaTeX in the Markdown cells, for instance `$f(x)=x^2$` will be rendered as $f(x) = x^2$ and `$$f(x) = x^2$$` will become an equation line, as follows

  $$f(x) = x^2$$

  Another example is `$$f_{Y \mid X}(y,x) = P(Y = y \mid X = x) = \exp(\alpha \cdot x + \beta)$$` which renders as

  $$f_{Y \mid X}(y,x) = P(Y = y \mid X = x) = \exp(\alpha \cdot x + \beta)$$

1.4 Finally some rules:

- You may not communicate with others during the exam, for example:
  - You cannot ask for help in Stack-Overflow or other such help forums during the Exam.
  - You may not communicate with AI's, for instance ChatGPT.
  - Your on-line and off-line activity is being monitored according to the examination rules.

1.5 Good luck!

```
[ ]: # Insert your anonymous exam ID as a string in the variable below
     examID="XXX"
```

1.6 Exam vB, PROBLEM 1

Maximum Points = 14

This problem is about SVD and anomaly detection. In all the problems where you are asked to produce a matrix or vector, they should be **numpy arrays**.

1. [4p] Load the file `data/SVD.csv` as instructed in the code cell. Compute the Singular Value Decomposition, i.e. construct the three matrices U, D, V such that if X is the data matrix of shape `n_samples x n_dimensions` then $X = UDV^T$. Put the resulting matrices in their variables, check that the shapes align with the instructions in the code cell. Finally, extract the first right and left singular vectors and store those as 1-d arrays in the instructed variables. **Hint: make sure that the first right and left singular vectors are correct by using the matrix, also be careful about the shape!!**
2. [3p] The first goal is to calculate the explained variance, check the lecture notes for definition. Calculate the explained variance of using 1, 2,… number of singular vectors and select the smallest number of singular vectors that is needed in order to explain at least 90% of the variance.
3. [3p] With the number of components chosen in part 2, construct the best approximating matrix with the rank as the number of components. Explain geometrically what each row represents in the approximating matrix in terms of the original data, write your answer as free text in the Markdown cell below as instructed in the cells.
4. [4p] Create a vector which corresponds to the row-wise (Euclidean) distance between the original matrix `problem1_data`and the approximating matrix `problem1_approximation` and plot the empirical distribution function of that distance. Based on the empirical distribution function choose a threshold such that 10 samples are above it and the rest below. Store the 10 samples in the instructed variable.

```
[ ]: # Part 1: 4 points

     # Load the data from the file data/SVD.csv and store the data in a numpy␣
   ↪array called problem1_data below
     # Double check that the numbers have been parsed correctly by checking the␣
   ↪dtype of the array by calling problem1_data.dtype
     problem1_data = XXX # A numpy array of shape n_samples x n_dimensions

     problem1_U = XXX # The matrix of left singular vectors of problem1_data with␣
   ↪shape n_samples x n_dimensions
     problem1_D = XXX # The vector of singular values of problem1_data with shape␣
   ↪n_dimensions
     problem1_V = XXX # The matrix of right singular vectors of problem1_data␣
   ↪with shape n_dimensions x n_dimensions
```

```
problem1_first_right_singular_vector = XXX # The first right singular vector␣
↪of problem1_data with shape (n_dimensions,) hint sometimes one needs to invoke␣
↪flatten() to avoid having shape (n_dimensions, 1) or (1, n_dimensions)

problem1_first_left_singular_vector = XXX # The first left singular vector␣
↪of problem1_data with shape (n_samples,) hint sometimes one needs to invoke␣
↪flatten() to avoid having shape (n_samples, 1) or (1, n_samples)
```

[ ]:
```
# Part 2: 3 points

# Calculate the explained variance of using 1,2,3,...,n_dimensions singular␣
↪values and store it as a numpy array called problem1_explained_variance below
problem1_explained_variance = XXX # A numpy array of shape (n_dimensions,),␣
↪it should be an increasing sequence of positive numbers and the last element␣
↪should be 1

# Store in the variable below the smallest number of singular values needed␣
↪to explain at least 90% of the variance
problem1_num_components = XXX # An integer
```

[ ]:
```
# Part 3: 3 points

# Calculate the approximating matrix of problem1_data using the first␣
↪problem1_num_components singular values and store it in the variable below
problem1_approximation = XXX # A numpy array of shape n_samples x␣
↪n_dimensions
```

1.7 Free text answer

Put the explanation for **part 3** of the rows of the approximating matrix below this line in this cell. In order to enter edit mode you can doubleclick this cell or select it and just press enter.

[ ]:
```
# Part 4: 4 points

# Calculate the reconstruction error of problem1_data using␣
↪problem1_approximation and store it in the variable below (should have shape␣
↪(n_samples,)) (row wise Euclidean distance)
problem1_reconstruction_error = XXX

# Put the code below to plot the empirical distribution function of the␣
↪reconstruction error
# XXX
# XXX
# XXX
```

```
    # Store the value of the selected threshold in the variable below
    problem1_threshold = XXX

    # Finally store the samples of problem1_data that have a reconstruction␣
↪error larger than problem1_threshold in the variable below, should have shape␣
↪(10, n_dimensions)
    problem1_outliers = XXX
```

1.8 Exam vB, PROBLEM 2

Maximum Points = 13

You are given the data-science job salaries dataset found in `data/salaries.csv`, which contains the salaries of jobs, their experience level and how much of the working hours are remote. Your task is to train a `linear regression` model to predict the salary of a job based on its attributes: * `work_year`, The year the salary was paid. * `experience_level`, The experience level in the job during the year with the following possible values: 0 Entry-level / Junior 1 Mid-level / Intermediate 2 Senior-level / Expert 3 Executive-level * `employment_type`, The type of employement for the role: Part-time, Full-time, Contract, Freelance * `salary_in_usd`, The total gross salary amount paid in US Dollars. * `remote_ratio`, The overall amount of work done remotely, possible values are as follows: 0 No remote work (less than 20%) 50 Partially remote 100 Fully remote (more than 80%)

To evaluate your model, you will split the dataset into a training set and a testing set. You will use the training set to train your model, and the testing set to evaluate its performance.

1. Load the data into a pandas dataframe `problem2_df`. Based on the column names, figure out what are the features and the target and fill in the answer in the correct cell below. [2p]
2. Split the data into train and test. [2p]
3. Train the model. [1p]
4. On the test set, evaluate the model by computing the mean absolute relative error and plot the empirical distribution function of the residual with confidence bands (i.e. using the DKW inequality and 99% confidence). Hint: you can use the function `plotEDF,makeEDF` combo from `Utils.py` that we have used numerous times, which also contains the option to have confidence bands. [3p]

$$\text{Absolute relative error} = \left|\frac{\text{true-predicted}}{\text{true}}\right|$$

5. Provide a scatter plot where the x-axis corresponds to the predicted value and the y-axis is the true value, do this over the test set. [2p]
6. Reason about the performance, for instance, is the value of the mean absolute relative error good/bad and what do you think about the scatter plot in point 5? [3p]

```
[ ]:    # Part 1
        # Let problem2_df be the pandas dataframe that contains the data from the␣
     ↪file
        # data/salaries.csv
```

```
problem2_df = XXX
```

[ ]:
```
# Part 1

# Fill in the features as a list of strings of the names of the columns

problem2_features = ["XXX"]

# Fill in the target as a string with the correct column name

problem2_target = "XXX"
```

[ ]:
```
# Part 2



# Split the data into train and test using train_test_split
# keep the train size as 0.8 and use random_state=42
problem2_X_train,problem2_X_test,problem2_y_train,problem2_y_test = XXX
```

[ ]:
```
# Part 3

# Include the necessary imports

# Initialize your linear regression model
problem2_model = XXX

# Train your model on the training data
```

[ ]:
```
# Part 4

# Evaluate the model by computing the mean absolute relative error
problem2_mare = XXX
```

[ ]:
```
# Part 4

# Write the code to plot the empirical distribution function of the residual
# with confidence bands with 99% confidence in this cell
```

[ ]:
```
# Part 5

# Write the code below to produce the scatter plot for part 5
```

1.9 Part 6

Double click this cell to enter edit mode and write your answer for part 6 below this line.

Discussion on the value of the MARE

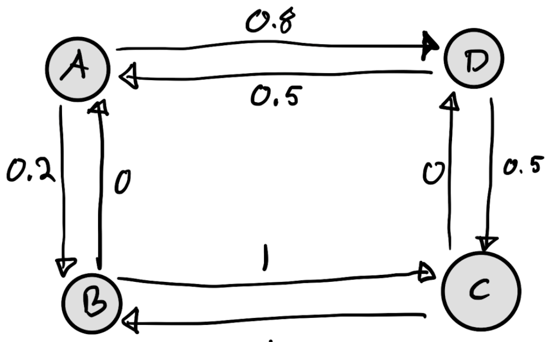
Discussion on the predicted vs. true scatterplot

Discussion

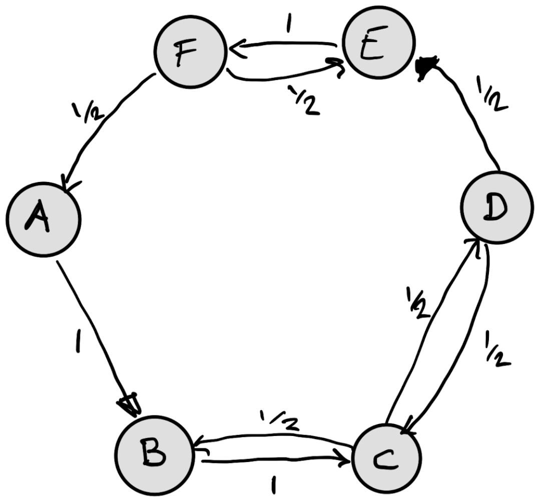
1.10 Exam vB, PROBLEM 3

Maximum Points = 13

Consider the following two Markov chains:



Markov Chain A



Markov Chain B

Answer each question for all chains:

1. [2p] What is the transition matrix?
2. [1p] Is the Markov chain irreducible?
3. [4p] Is the Markov chain aperiodic? What is the period for each state? Hint: Recall our definition of period; Let $\mathbb{T} := \{t \in \mathbb{N} : P^t(x, x) > 0\}$ and the greatest common divisor of $\mathbb{T}$ is the period.
4. [2p] Being in state A at time 0 what is the probability of being in state B at time 5 (after 5 steps)
5. [4p] Define T as the first time being in state D starting in state A. That is, if $X_0, X_1, \ldots$ is the Markov chain then define for $X_0 = "A"$

$$T(\omega) = \inf_{t \in \mathbb{N}} \{t : X_t(\omega) = "D"\}$$

where the infimum over the empty set is ∞. Calculate $\mathbb{P}(T = 1)$, $\mathbb{P}(T = 2)$, $\mathbb{P}(T = 3)$, $\mathbb{P}(T = 4)$, $\mathbb{P}(T = 5)$, $\mathbb{P}(T = \infty)$.

```
[ ]:    # PART 1

        #------------------------TRANSITION MATRIX -------------------------------
```

```
# Answer each one by supplying the transition matrix as a numpy array
# of shape (n_states,n_states), where state (A,B,...) corresponds to index␣
↪(0,1,...)

problem3_A    = XXX
problem3_B    = XXX
```

[ ]:
```
# PART 2
#------------------------REDUCIBLE ------------------------------
# Answer each one with a True or False

problem3_A_irreducible = XXX
problem3_B_irreducible = XXX
```

[ ]:
```
# PART 3
#------------------------APERIODIC------------------------------
# Answer each one with a True or False

problem3_A_is_aperiodic = XXX
problem3_B_is_aperiodic = XXX

# Answer the following with the period of the states as a numpy array
# of shape (n_states,)

problem3_A_periods = XXX
problem3_B_periods = XXX
```

[ ]:
```
# PART 4

# Answer the following with the probability of being in state B at time 5␣
↪for the two problems
problem3_A_PB5 = XXX
problem3_B_PB5 = XXX
```

[ ]:
```
# PART 5

# Answer the following probabilities for T=1,2,3,4,5 and infinity
problem3_A_PT1 = XXX
problem3_A_PT2 = XXX
problem3_A_PT3 = XXX
problem3_A_PT4 = XXX
problem3_A_PT5 = XXX
problem3_A_PT_inf = XXX

problem3_B_PT1 = XXX
problem3_B_PT2 = XXX
problem3_B_PT3 = XXX
```

```
    problem3_B_PT4 = XXX
    problem3_B_PT5 = XXX
    problem3_B_PT_inf = XXX
```