Algorithmic Foundations of Data Science

Mock Exam With Solutions , whenever you want

Last Name: _____

First Name: _____

Student ID: _____

Study Program: _____

Remarks

- Write your name and student ID on **every** sheet of paper you use.
- Write your solution in the space provided on the problem sheets. If you need more paper, ask us. Do **not** use your **own paper** and hand in all paper you obtained (even scratch paper).
- You can answer either in German or in English but please do not mix languages.
- Only use **black** or **blue** pens. Do **not** use pencils.
- If you provide multiple solutions to a question, the worst of those counts.
- You have **120 minutes** to work on the exam. With **50 points** you have passed this exam.

I hereby declare that I have read the above guidelines and that I am healthy enough to take the exam.

_____

(Signature)

Do not write below this line.

	1	2	3	4	5
Points	/ 19	/ 24	/ 18	/ 17	/ 22
Sign.					

$\sum$	/ 100

Problem 1 (Questions on Learning Problems) 5+5+4+5 = 19 points

a) Briefly explain the concepts of supervised and unsupervised learning. What is the difference? For both concepts give a learning algorithm that fits to this concept.

Solution:

In supervised learning the input is a set of labeled data, e.g. training examples which are labeled according to some classification task. In unsupervised learning, the input is unlabeled and the task is to extract useful information.

An example for supervised learning is Decision Tree Learning whereas Clustering for example using the k-Means algorithm is a typical example of unsupervised learning.

b) Define the entropy of a probability distribution $\mathcal{P}$ on a finite sample space Ω. What is the intuitive meaning of the entropy?

Solution:

$H(\mathcal{P}) = \sum_{\omega \in \Omega} P(\{\omega\}) \cdot \log \frac{1}{\mathcal{P}(\{\omega\})}$.

Intuitively, the entropy measures the expected number of bits required to describe an elementary event $\{\omega\}$.

c) The PAC-learning framework allows to derive upper bounds on the number of training examples required for a learning problem. Which assumptions does the framework make on the training data?

Solution:

It is assumed that each training example is drawn **identically and idependently** according to some data generating distribution $\mathcal{D}$.

d) In the lecture we have discussed two methods for Dimension Reduction of high-dimensional data. Which ones?

Solution:

Random Projections and Principal Component Analysis.

Problem 2 (Eigenvalues and Eigenvectors) **6+8+4+6 = 24 points**

a) Consider the following matrix

$$M = \begin{pmatrix} 2 & -1 & 3 & 1 & -1 \\ 1 & -2 & -3 & 1 & 1 \end{pmatrix}.$$

What are the eigenvectors for all non-zero eigenvalues of $M^T M$?

Solution:

We first compute eigenvalue and eigenvectors for the matrix $MM^T \in \mathbb{R}^{2\times 2}$.

$$MM^T = \begin{pmatrix} 16 & -5 \\ -5 & 16 \end{pmatrix}.$$

We find that $(1,1)^T$ and $(1,-1)^T$ are the eigenvectors with eigenvalues 11 and 21.

Hence, the non-zero eigenvectors of $M^T M$ are $M^T(1,1)^T = (3,-3,0,2,0)$ and $M^T(1,-1)^T = (1,1,6,0,-2)$ with eigenvalues 11 and 21 .

b) Describe the Power Iteration Algorithm. What does it compute? What are the requirements on the input for the algorithm to work?

Solution:

The Power Iteration Algorithm takes as input a square matrix $A \in \mathbb{R}^{n\times n}$ and an initial vector $x_0 \in \mathbb{R}^n$. Then the algorithm iteratively computes $x_{k+1} = \frac{Ax_k}{\|Ax_k\|_2}$ and outputs the result as soon as it converges (or the result does not change significantly).

Assuming A is diagonsiable and has a unique absolute largest eigenvalue which is positive (i.e. $\lambda_1 > |\lambda|$ for all other eigenvalues $\lambda \neq \lambda_1$), and x_0 is not orthogonal to the eigenvector of the largest eigenvalue λ_1, the sequence converges to an eigenvector for the largest eigenvalue. Hence, the algorithm computes (an approximation) for an eigenvector to the largest absolute eigenvalue of A.

c) Give a square matrix A and an initial vector x_0 such that the Power Iteration algorithm does not converge on A with initial vector x_0. Explain why it does not converge.

Solution:

Consider the matrix $A = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$ and let $x_0 = (1,1)^T$. Then the Power Iteration algorithm alternates between vectors $x_{2i} = \frac{1}{\sqrt{2}}(1,1)^T$ and $x_{2i+1} = \frac{1}{\sqrt{2}}(1,-1)^T$. The eigenvalues of A are $\lambda_1 = 1$ and $\lambda_2 = -1$. The problem is that A has no dominant eigenvalue since $|\lambda_1| = |\lambda_2|$.

d) Suppose we are given ℓ points $x_1, \ldots, x_\ell \in \mathbb{R}^n$ with similarity measure $s\colon \mathbb{R}^n \times \mathbb{R}^n \to [0,1]$. We want to cluster the points into k clusters using Spectral Clustering. Give pseudocode for the Spectral Clustering Method that realizes this task. Briefly explain the steps of your algorithm.

Solution:

- compute similarity matrix S with entries $S_{i,j} = s(x_i, x_j)$
- compute Laplacian $L = S - D$ where D is the diagonal matrix containing the row-sums of S
- compute the eigenvectors $y_1, \ldots, y_k$ for the k smallest eigenvalues of L
- cluster points $z_1, \ldots, z_\ell$ where $z_i = ((y_1)_i, \ldots, (y_k)_i)$ using k-means
- let $C_1, \ldots, C_k$ be the output
- return $D_1, \ldots, D_k$ where $D_i = \{x_i \mid z_i \in C_i\}$

Problem 3 (Markov Chains) **4+3+7+4 = 18 points**

a) Suppose your are on a summer holiday and choose your daily activity only based on the activity of the previous day. There are only three possible activities: going to the beach **(B)**, hiking (H), or a trip to the nearby city **(C)**.

- If you went to the beach yesterday, then today you go to the beach again **(B)** with probability 0.4, you choose hiking **(H)** with probability 0.4 and the city **(C)** with probability 0.2.
- If you went hiking yesterday, then today you go to the beach **(B)** with probability 0.6, you choose hiking **(H)** with probability 0.1 and the city **(C)** with probability 0.3.
- If you went to the city yesterday, then today you go to the beach **(B)** with probability 0.4, you choose hiking **(H)** with probability 0.5 and the city **(C)** with probability 0.1.

Model your activities by a Markov chain. Give the corresponding transition matrix.

Solution:

We number the activities as follows: 1 - (B), 2 - (H), 3 - (C). The transition matrix is

$$Q = \begin{pmatrix} 0.4 & 0.4 & 0.2 \\ 0.6 & 0.1 & 0.3 \\ 0.4 & 0.5 & 0.1 \end{pmatrix}.$$

b) If you are hiking today, what is the probability to be on the beach on the day after tomorrow?

Solution:

Let $v = (0, 1, 0)$ which is the probability vector for today's activity. In order to compute the probabilities for the day after tomorrow, we need to compute vQ^2. We have $w := vQ = (0.6, 0.1, 0.3)$ and

$wQ = (0.24{+}0.06{+}0.12, 0.24{+}0.01{+}0.15, 0.12{+}0.03{+}0.03) = (0.42, 0.4, 0.18)$.

So the probability for being on the beach the day after tomorrow is 42%.

c) What fraction of the time do you spent in the city in the long run? Towards this end, compute the stationary distribution of the chain.

Solution:

In order to compute the stationary distribtion one needs to solve the equation $xQ = x$ which is equivalent to $(Q^T - I)x^T = 0$. We have

$$Q^T - I = \begin{pmatrix} -0.6 & 0.6 & 0.4 \\ 0.4 & -0.9 & 0.5 \\ 0.2 & 0.3 & -0.9 \end{pmatrix}.$$

Gaussian Elimination yields the matrix

$$\begin{pmatrix} -6 & 6 & 4 \\ 0 & -5 & \frac{23}{3} \\ 0 & 0 & 0 \end{pmatrix}.$$

Hence, $(33, 23, 15)^T$ is an eigenvector of Q^T with eigenvalue 1. The stationary distribtion is $\frac{1}{71}(33, 23, 15)$. So the fraction of the time spent in the city in the long run is $\frac{15}{71}$.

d) A Markov chain is said to be *symmetric* if its transition matrix is symmetric. What is the stationary distribution of a connected symmetric chain? Prove your answer.

Solution:

Let $Q \in \mathbb{R}^{n \times n}$ be the transition matrix of a symmetric Markov Chain, i.e. Q is symmetric and

$$\sum_{j \in [n]} q_{i,j} = 1$$

for all $i \in [n]$. Let $\pi = (1, \ldots, 1) \in \mathbb{R}^n$ be the all-ones vector. Then $\pi Q = \pi$ since, for all $j \in [n]$

$$\sum_{i \in [n]} 1 \cdot q_{i,j} = \sum_{i \in [n]} q_{j,i} = 1.$$

Normalizing the vector π we conclude that $(\frac{1}{n}, \ldots, \frac{1}{n})$ is the stationary distribution of the Markov chain.

Problem 4 (Map-Reduce Algorithms) **4+4+9 = 17 points**

Let G be the internet graph and suppose for simplicity the vertices (i.e. the websites) are numbered from 1 to n. In this exercise we consider MapRecude algorithms where the initial input of the first phase are tuples of the form $(siteA, linksA)$ where $siteA$ is a website and $linksA$ is a list of websites $siteB$ such that there is a link from $siteA$ to $siteB$.

a) Consider the following MapReduce algorithm for computing the set of all key-value pairs $(\{siteA, siteB\}, comAB)$ where $siteA$ and $siteB$ are websites and $comAB \neq \emptyset$ are the common links of the two websites.

Map	on input $(siteA, linksA)$ emit $(\{siteA, siteB\}, linksA)$ for all sites $siteB$
Reduce	on input $(\{siteA, siteB\}, val)$ emit $(\{siteA, siteB\}, comAB)$ if $comAB \neq \emptyset$ where $comAB$ is the intersection of all sets $links \in val$.

Determine the communication cost of the algorithm by counting the number of input key-value pairs for both phases (in the worst-case).

Solution:

The communication cost is n for the Map phase and $\binom{n}{2}$ for the Reduce phase. So in total the communication cost is $n + \binom{n}{2}$.

b) Give a MapReduce algorithm that outputs all pairs $(siteA, inLinksA)$ where $siteA$ is a website and $inLinksA$ is a list of all incoming links, i.e. websites $siteB$ such that there is a link from $siteB$ to $siteA$.

Solution:

Map	on input $(siteA, linksA)$ emit $(siteB, siteA)$ for all sites $siteB \in linksA$
Reduce	on input $(siteA, val)$ emit $(siteA, val)$.

c) Let d be the maximum degree of the internet graph (counting incoming and outgoing edges). It is reasonable to assume that d is much smaller than n, the total number of websites. Use part b) to give an improved algorithm for the problem from part a) taking the maximum degree d into account. Again, analyse the communication cost. Compare it to your results from part a).

Solution:

Map	on input $(siteA, linksA)$ emit $(siteB, siteA)$ for all sites $siteB \in linksA$
Reduce	on input $(siteA, val)$ emit $(siteA, val)$
Map	on input $(siteA, inLinksA)$ emit $(\{siteB, siteC\}, siteA)$ for all distinct sites $siteB, siteC \in inLinksA$
Reduce	on input $(\{siteA, siteB\}, val)$ emit $(\{siteA, siteB\}, val)$

To analyse the communication cost we need to count the number of input key-value pairs for all four phases. The first Map-phase receives n key-value pairs and similarly, first Reduce-phase receives n key-value pairs (in both cases a pair for each website). Since the first Reduce-phase does not do anything the second Map-phase also receives n key-value pairs. In the second Map-phase each task emits at most $\binom{d}{2}$ key-value pairs which means each task contributes at most $\binom{d}{2}$ different keys. So in total the second Reduce-phase receives at most $\binom{d}{2} \cdot n \leq d^2 n$ many key-value pairs. So in total the communication cost is bounded by $(3 + d^2)n$. If d is significantly smaller than $\sqrt{n}$ this is much better than the algorithm from part a).

Problem 5 (Streaming Algorithms) 4+6+4+3+5 = 22 points

a) Consider the stream $a = 1, 4, 4, 1, 5, 3, 8, 2, 2, 1, 5$ over the universe $\mathbb{U} = \{1, \ldots, 10\}$. Compute the p-th frequency moment for $p = 0, 1, 2$, i.e. give $F_0(a)$, $F_1(a)$ and $F_2(a)$.

Solution:

We have $F_0(a) = 6$, $F_1(a) = 11$ and $F_2(a) = 9+4+1+4+4+0+0+1+0+0 = 23$.

b) Apply the Tug-Of-War estimator to the stream of part a) using the hash $h\colon \mathbb{U} \to \{-1, 1\}$ given below. Give the result as well as some some intermediate steps showing how the estimator is computed from a.

u	1	2	3	4	5	6	7	8	9	10
$h(u)$	1	1	-1	1	-1	1	1	1	-1	-1

Solution:

We first need to compute $x = \sum_{i\in[11]} h(a_i) = 1+1+1+1-1-1+1+1+1+1-1 = 5$. The the Tug-Of-War estimator return $x^2 = 25$.

c) What are the requirements on the family of hash functions for the Tug-Of-War estimator? What guarantee on the output does the Tug-Of-War estimator give?

Solution:

The Tug-Of-War Estimator requires a strongly 4-universal family of hash functions. Let B be the estimator returned by the algorithm. Then $E(B) = F_2(a)$.

d) The Tug-Of-War estimator itself typically does not give very good results. What is the problem with the Tug-Of-War estimator?

Solution:

The problem of the algorithm is that the output has a rather high variance which means that, in many cases, the output is relatively far away from the actual frequency moment.

e) In the lecture we have discussed an extension of the Tug-Of-War estimator which avoids the problems discussed in part d). How does this extension work? What additional guarantee can we get for the extended version?

Solution:

In the lecture we saw an extension of the algorithm using averaging. We draw k hash functions independently and return the average of the estimators computed by the Tug-Of-War for each hash function k.

By choosing k sufficiently large one can guarantee that the estimator is arbitrarily close (up to an ε-factor) to the true frequency moment with high probability.