CSC311 Final Project Overview

UNIVERSITY OF TORONTO

- Background and Task
- Dataset and Starter Code
- Inspecting a Baseline Model
- Overview of Different Approaches

Background and Task

- **Massive Open Online Courses**: KhanAcademy, Coursera
- **Question:** How can we personalize education in MOOCs?
- **Idea:** Measure students' understanding of the material by introducing a personalized assessment component.

Background and Task

Why a personalized assessment component?

- Each question can be designed to highlight a misconception.
- Lets us adjust the level of difficulty.

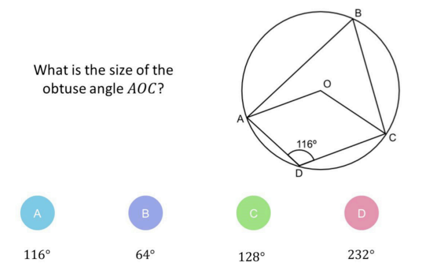


Figure 1: An example diagnostic question [1].

Background and Task

Goal: Build a predictive model to predict whether a student will answer a given question correctly, given answers to past questions, and other students' answer.

DATA:
(student_id, question_id, is_correct)
(1, 1, 1)
(1, 2, 0)
(2, 1, 0)
(2, 3, 1)
(3, 1, 1)

→

PREDICTIONS:
(student_id, question_id, ?)
(2, 2, ?)
(3, 3, ?)

Background and Task

- **Part A:** Try out established methods you've covered in class.
- **Part B:** Improve on the existing methods.

The project has an (ungraded) Kaggle-based competition component!

Background and Task

Lets switch to the Colab notebook.

- We’ll inspect the dataset and the starter code.
- We’ll build a baseline model and make a Kaggle submissions with it.

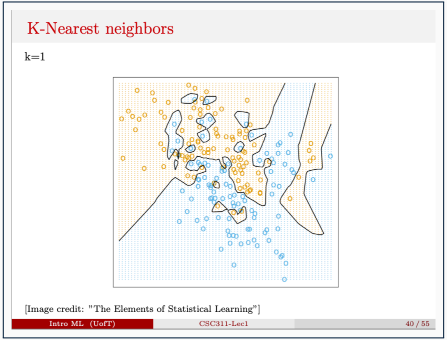
Meta Data

- The dataset also contains metadata including 1) date of birth 2) gender 3) eligibility for ”pupil premium”.
- Not used in part A, but might be relevant for part B.

Part A: Testing out various models, under the guidance of the project handout.

KNN

- Given a notion of similarity, classify a test example by looking at the most similar training examples to it.



- Similarity in terms of student, or similarity in terms of question?

KNN

What to analyze?

- **Notion of similarity:** Compare student-based similarity with item-based similarity.
- **Choice of hyperparameter:** In both cases, which value of k works better?
- **Limitations:** What are the limitations of using KNN in this context?

Item Response Theory

- **Goal:** Assign a probability that a student will answer a given question correctly.
- **Simplifying assumption 1:** Correct answer probability depends on two parameters:
  - θ_i: ith Student ability
  - β_j: jth question difficulty.
- **Simplifying assumption 2: Correct answer probability increases monotonically with θ_i and $-\beta_j$.**

Item Response Theory

- **Model:**

$$p(c_{ij}|\theta_i, \beta_j) = sigmoid(\theta_i - \beta_j) = \frac{\exp(\theta_i - \beta_j)}{1 + \exp(\theta_i - \beta_j)}$$

- **How to train:** Maximize data log likelihood under model parameters!
- **Connection to logistic regression:** Think about how this model relates to logistic regression!

- **Possible extensions**[1]

$$p(c_{ij}|\theta_i, \beta_j) = c + [1 - c] * sigmoid(k_j(\theta_i - \beta_j))$$

- c: Probability of getting question right via. random guess.
- k_j: How steep the sigmoid looks (i.e. how discriminative the question is")

[1] reference link

Can you think of other real-life problems where Item Response Theory can be applied?

- healthcare
- recommender systems
- ?

What to analyze?

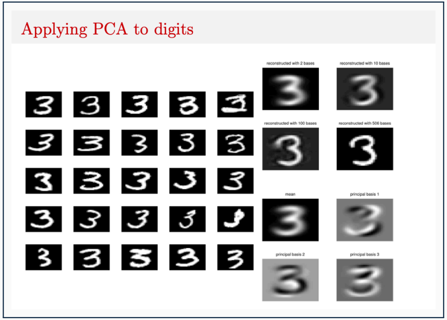
- **Log likelihood:** Derive the log likelihood and inspect it's form.
- **Inspecting the results:** Using the trained θ and β vectors, plot how the probability of a correct answer changes as "student ability" varies. Why does the plot look the way it does? What can we learn from the plot?

Matrix Factorization

We consider two options in the handout:

- Singular Value Decomposition
- Alternating Least Squares

- Using PCA (via. Singular Value Decomposition)



- **Goal:** Complete the matrix using the top principal components.
- **Question:** Using KNN to fill in missing values requires us to specify whether we're using question or student similarity. Is there such a distinction for SVD?

Matrix Factorization

- **Alternating Least Squares:** Assign each student and question a vector. Train the values of these vectors so that a high dot product between student *i* and question *j*'s vectors implies a correct answer.
- **Objective:**

$$\min_{U,Z} \frac{1}{2} \sum_{(n,m)\in\mathcal{O}} (C_{nm} - \mathbf{u}_n^T \mathbf{z}_m)^2 \tag{1}$$

- **How to train U and Z:** Loop over each $\boldsymbol{u}_n$ and $\boldsymbol{z}_m$, and solve (1) assuming all other terms are fixed. Repeat until convergence.

Matrix Factorization

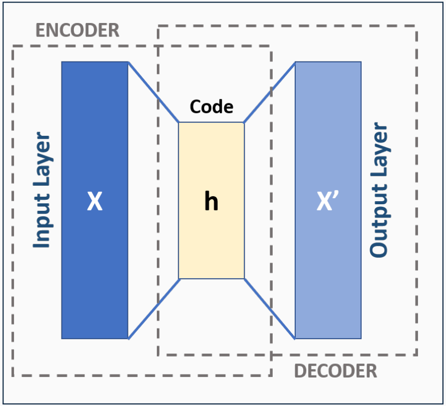
- How to train U and Z matrices:
  1. Initialize U and Z.
  2. repeat until “convergence”:
  3.   **for** $n = 1, \ldots, N$ **do**
  4. $\boldsymbol{u}_n = (\sum_{j:(n,j)\in\mathcal{O}} \boldsymbol{z}_j \boldsymbol{z}_j^\top)^{-1} \sum_{j:(n,j)\in\mathcal{O}} c_{nj} \boldsymbol{z}_j$
  5.   **for** $m = 1, \ldots, M$ **do**
  6. $\boldsymbol{z}_m = (\sum_{i:(i,m)\in\mathcal{O}} \boldsymbol{u}_i \boldsymbol{u}_i^\top)^{-1} \sum_{i:(i,m)\in\mathcal{O}} c_{im} \boldsymbol{u}_i$

What to analyze?

- **Limitations of SVD:** In what way is SVD limited in this context?
- **Affect of hyperparameters on ALS performance:** How does the choice of hyperparameters affect the training dynamics and the final accuracy?
- **Alternative objectives:** Can we change the loss function so that the problem is treated as a binary classification problem?

Neural Network

- **Learning a "student autoencoder":** Represent each student by a vector of length $N_{questions}$. Train an autoencoder to project the student vectors into a low dimensional space where *similar students are clustered together*.



Neural Network

- **Learning objective:**

$$\min_{\theta} \sum_{\mathbf{v} \in \mathcal{S}} ||\mathbf{v} - f(\mathbf{v}; \theta)||_2^2 \tag{2}$$

- **Network architecture:** Two layer, fully connected network.

Neural Network

What to analyze?

- **Bottleneck width:** How does the dimensionality of the bottleneck layer affect the results?
- **Effect of regularization:** How does regularizing the network weights by penalizing their Frobenius norm affect the results?

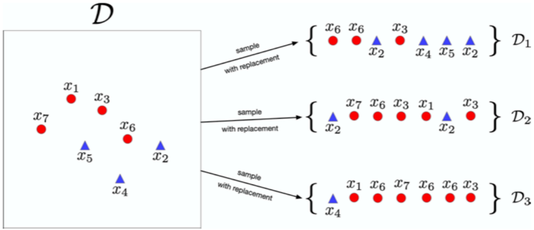
Ensemble

- Try to improve stability and accuracy by:
  1. Select 3 models (same or different).
  2. Generate three alternative datasets by bagging.
  3. Train the models on the corresponding bagged dataset.
  4. Pick the average of the 3 models as the final decision on the test set.

Ensemble

- Reminder about bagging:

Bagging



in this example $n = 7$, $m = 3$

Ensemble

What to analyze:

- How did using an ensemble affect the accuracy?
- How did it affect the stability of the model?

This part is more open ended - don't forget to explain your approach in enough detail that a reader of your report can faithfully reproduce your results.

Questions / Starter Code

If we have time remaining, we can either look deeper into the starter code, or answer student questions.