# Assignment 2

**Assignment Policy**: Read all the instructions below carefully before you start working on the assignment, and before you make a submission.

- This assignment is divided into two parts: theoretical and programming. We strongly advise you to typeset your entries in Latex. With your submission, please include your name and student code. Late submissions will incur a 5% penalty per day, up to a maximum of three days, beyond which submissions will be rejected.
- For the programming questions, please include the plots required and answers to sub-parts in the same file which contains solutions to the theory problems. Use the name Colab_assignment2.ipynb for programming questions.

## Problem 1: Language models (5 points)

Assume you are given the following corpus of text:

```
I like dogs
You like some cats
I hate some cats
You like white dogs
I like you
```

Provide the equation for bigram probabilities and estimate all bigram probabilities for this corpus (ignore casing, e.g., treat you and You as the same token). Present the probabilities in a tabular format:

| **Bigram** | **Probability** |
|---|---|
| P(`like` \| `I`) | . |
| . | . |

## Programming 2: Sentiment analysis: 10 points

This problem will require programming in Python 3. The goal is to build a feedforward NN model that you learnt from the class on a real-world sentiment classification dataset. The dataset you will be using is collected from IMDB movie reviews datasetn(https://drive.google.com/file/d/13Rn9m_6dKPpImqeMGXpRwfQgBCrULL6i/view?usp=sharing). The dataset has been split into a training, a development and a test set.

First, implement and train a feedforward NN model with TF-IDF. And then train your model using word2vec embedding. Report both training and development accuracy on the dataset. *Try to use stochastic gradient descent or (mini-batch) stochastic gradient descent!*

Your implementation should be structured like this.

```
# load dataset  into memory
def load_data (filename):

# turn a dataset  into clean tokens
def clean_data(doc, vocab):

#  preprocess the dataset
def process_data():

# define the model
def define_model():

classify a review as negative or positive.
def predict_sentiment():
```