Sequential Data Analysis with Hidden Markov Models

1 Introduction

The hidden Markov model (HMM) is a generative probabilistic model of sequence data. It has similarities to clustering (covered on Day 2). Clustering assumes that the data are exchangeable, i.e., that the probability of a data set does not change if you change its order. Clearly data that have sequential structure, such as time series, are not exchangeable. The hidden Markov model can be thought of as a clustering method for non-exchangeable data.

In this project, we extend the mixture model to the hidden Markov model and explore inference algorithms, applications, further extensions and limitations.

Recall the objective function that K-means clustering maximizes with respect to cluster centers μ and assignments r, from Day 2,

$$\mathcal{L}(r, \mu) = \sum_{n=1}^{N} \sum_{k=1}^{K} -r_{nk} ||x_n - \mu_k||^2 \tag{1}$$

To introduce *soft* clustering, let $0 \leq r_{nk} \leq 1$ and $\sum_{k=1}^{K} r_{nk} = 1$ for all n. Soft clustering means that the assignments are not restricted to be binary one-hot vectors and instead the assignments are partial memberships of data points to clusters.

Now, here is an idea for adding first-order sequential structure to the soft clustering objective function: add an extra term to the objective function to encourage neighboring r's in the sequence to have the same cluster assignment,

$$\mathcal{L}(r, \mu, \pi) = \sum_{n=1}^{N} \sum_{k=1}^{K} -r_{nk} ||x_n - \mu_k||^2 + \sum_{k_1=1}^{K} \sum_{k_2}^{K} r_{n-1,k_1} r_{nk_2} \log \pi_{k_1,k_2} \tag{2}$$

where for all n, $0 \leq r_{nk} \leq 1$ and $\sum_{k=1}^{K} r_{nk} = 1$ and $\sum_{k_2=1}^{K} \pi_{k_1,k_2}$ for all $k_1 \in [1, K]$.

In Eq. 2, we introduced *transition matrix* π, a K-by-K matrix indicating the probability of transition between states. For simplicity, assume $r_{0,k} = \frac{1}{K}$, i.e., that we have no information about the starting cluster assignment, usually referred to as a *state* in HMMs.

Question: for $\pi = I$ (the K-by-K identity matrix), what happens to the objective function when data point $n-1$ is assigned to the same cluster as data point n? What happens when they are different? How do those answers change with different π?

2 Understanding the HMM as a Probabilistic Model

Question: [include your answer in the presentation on Day 5] I have presented the objective function for the simplest HMM in Eq. 2. In Day 3 we covered probabilistic models. Given what you now know, interpret Eq. 2 as a probabilistic model. Specifically, rewrite Eq. 2 as a sum of log probabilities using standard probability distributions (advanced: that is, those probability distributions belonging to the exponential family). Imagine that you observed a sequence of binary numbers instead of continuous numbers, how would you change the model (sum of log probabilities) and in turn, what new objective function would you obtain? *Optional bonus question:* draw the graphical model of the HMM.

3 Inference

Inference for the HMM can be done emphexpectation-maximization (EM) algorithm. You saw a variant of EM when we covered K-means on Day 2. The EM algorithm alternates between an E-step and an M-step until convergence. The M-step is partly the same for HMMs as for clustering (specifically, the update for μ is the same) and needs an additional simple step to update π, the probability of transitioning from one state (component) to another state (component). This is based on (smoothed) frequencies of transitions between the current estimates of the states. The E-step is more complicated because there is a joint dependence between all the unknown r_n's, and therefore uses an approach called the *forward-backward* algorithm to update efficiently.

4 Sequential Data Analysis of Stock Market Data

In the following tasks, you are going to apply the `hmmlearn` package to stock market data. You can obtain `hmmlearn` using the command `conda install hmmlearn`, using `pip` instead of `conda` and prepending with `sudo` if necessary.

4.1 Task 1: Setting Up

- Create a new iPython notebook for this project and give it a suitable name. Put the results of all the tasks into this notebook.
- Import the packages `numpy`, `hmmlearn`.
- Load the stock market data using the following code:[1]:

```
from matplotlib.finance import quotes_historical_yahoo_ochl
quotes = quotes_historical_yahoo_ochl(
    "INTC", datetime.date(1995, 1, 1), datetime.date(2012, 1, 6))

# Unpack quotes
dates = np.array([q[0] for q in quotes], dtype=int)
close_v = np.array([q[2] for q in quotes])
volume = np.array([q[5] for q in quotes])[1:]

# Take diff of close value. diff = np.diff(close_v)
dates = dates[1:]
close_v = close_v[1:]

# Pack diff and volume for training.
X = np.column_stack([diff, volume])
```

  X is an N-by-D numpy array, where N is the number of time steps and D is the dimensionality of the data. The first dimension gives the daily difference in closing price for the Intel stock. The second dimension gives the trading volume for Intel stock.

[1] Adapted from `http://hmmlearn.readthedocs.io/en/latest/auto_examples/plot_hmm_stock_analysis.html#sphx-glr-auto-examples-plot-hmm-stock-analysis-py`

4.2 Task 2: Fitting the Data

- Fit an HMM to X using the following code:

```
options = {"n_components":4,
 "covariance_type":"diag", "n_iter":1000}
model = hmmlearn.GaussianHMM(options).fit(X)
hidden_states = model.predict(X)
```

  In words: fit an HMM with 4 states, running the EM algorithm for a maximum of 1000 iterations. The Gaussian emissions are specified to have diagonal covariance meaning that the price difference and trading volume are assumed independent given the latent state of the HMM.

4.3 Task 3: Exploring the Fit

Let's explore the fit. Please refer to `https://hmmlearn.readthedocs.io/en/latest/tutorial.html` if you are unsure about any of the steps relating to `hmmlearn`.

- You now have the `hmmlearn.GaussianHMM` object fit to the stock data. Print the emissions parameters (means and variances for each state) and the hidden states of the data to the screen.
- Plot the data (price on the y-axis and time on the x-axis) coloured by the assignment to each state.
- Can you interpret what the states mean and why the trading on certain days were assigned those states?
- What is the predicted price differential and volume for the time step after the last time step in X?

4.4 Bonus Task 4: Extended HMM Models

Only do this task if you have extra time left over.

- As suggested in Section 2, the HMM is quite flexible and we are free to use a different emissions probability distribution (other than Gaussian). Compare the model you have been using, `GaussianHMM` to the other models possible in `hmmlearn`: `GMMHMM` and `MultinomialHMM`. How are they related? When would you use one over the other?

5 Conclusion

In this project you understood the HMM as a probabilistic model, ran inference in Python, and applied it to stock market data.