Prague Precipitation Forecasting using Hidden Markov Models

Matěj Štaif
CTU–FIT
staifmat@fit.cvut.cz

June 6, 2025

1 Introduction

This project work focuses on the implementation of Hidden Markov Models in Python using the *hmmlearn* (1) library.

The dataset is from the Prague-Ruzyně meteorological station (2) and contains very basic data, specifically: Avg. Temperature, Min. Temperature, Max. Temperature, Total Precipitation, Snow Depth, Wind Direction, Wind Speed, Peak Gust and Air Pressure.

The goal was to predict, based on daily historical data from 01/01/2000 to 31/12/2024, whether there would be precipitation the next day or not, and to appropriately backtest the model's performance.

1.1 Implemented Models

The following three models were evaluated as the most suitable choices for our predictive goal and were therefore implemented:

1. Hidden Markov Model with categorical (discrete) emissions (DHMM)
2. Hidden Markov Model with Gaussian mixture emissions (GMM HMM)
3. Hidden Markov Model with Multivariate Gaussian Emissions trained using Variational Inference (VGHMM)

2 Historical Data

The daily historical data was unevenly distributed, as there was no precipitation on 61% of the days and there was precipitation on 39% of the days. This fact complicated the model training, because there was a risk that the model would learn to predict every day as having no precipitation and subsequently achieve an accuracy of 0.61.

Even though the data is imbalanced, accuracy will be used as the main metric to better represent the model's performance and to ensure more natural comparability with the naive method described above.

3 Backtesting of the Performance of the Models

The model's performance was backtested using a sliding window approach, which is a method where a sliding window size n is selected, the model is trained on data of length n, and prediction for day $n+1$ is made regarding whether precipitation will occur or not. The sliding window then moves forward by one day, and the same steps are repeated until reaching the last record in the dataset. Subsequently, metrics such as accuracy, precision, recall, and F1 score were calculated.

3.1 Bayesian Optimization

At first, Grid Search was used to find best hyperparameters, but it was too slow. So, Bayesian Optimalization was used instead.

Bayesian optimization was implemented using the *optuna* (3) library for finding optimal hyperparameters. Optuna's built–in parallelization capabilities significantly reduced the computational time required for hyperparameter optimization.

3.2 Threshold Optimization

Since the models output rain probabilities instead of binary predictions, an optimal threshold for decision–making needed to be determined. This threshold optimization was performed using Bayesian optimization after the main hyperparameter tuning was completed.

4 Model Implementation and Algorithms

4.1 Training Algorithms

All HMM models used the Baum–Welch algorithm for parameter learning, except VGHMM which employed a variational algorithm for parameter learn-

ing. Both algorithms are fully implemented in the *hmmlearn* library.

4.2 Hidden Markov Model with discrete emissions (DHMM)

This model was implemented first, as it is the simplest to implement and debug. DHMM works only with categorical (discrete) emissions, while Gaussian HMM models can model more complex relationships in continuous data.

The model often achieved accuracy of only around 0.5 in early implementation phases. Initially, the author predicted whether there would be precipitation the next day using only a simple discrete rule: if the current state is X and the probability in that state is greater than threshold Y, then precipitation is predicted. This approach was very primitive.

Subsequently, the author improved the implementation to predict using marginalization. This improved correct predictions, but only slightly. Finally, the Forward Algorithm with marginalization was added, which is one of the most common methods for prediction in HMM.

The model was trained only on one type of data – the binary encoding of precipitation/no precipitation. The Author also attempted to train the model on the output from Linear Regression and experimented with adding new data columns for training, but ultimately abandoned this approach because the model achieved the best results when trained only on a single data column – the binary encoding of precipitation/no precipitation.

The implementation of this model took the most time because the author tried as many approaches as possible. This is considered a win, because the Forward Algorithm with Marginalization was subsequently used for precipitation prediction in all other models.

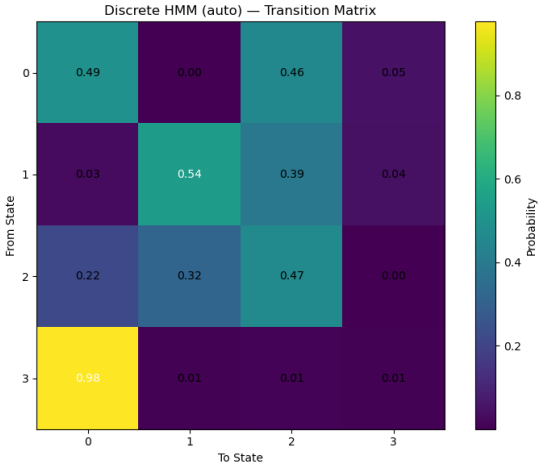

Figure 1: Transition Matrix

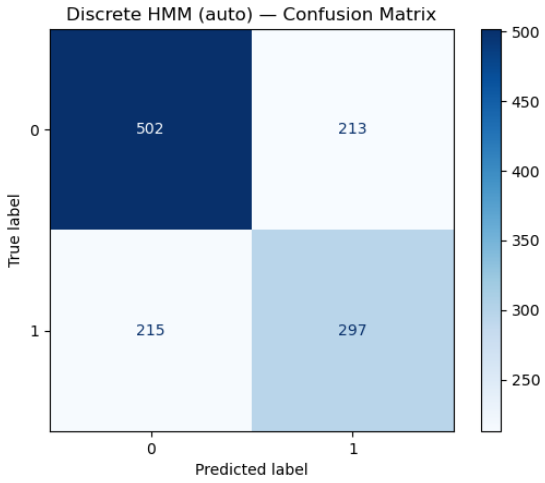

Figure 2: Confusion Matrix

The program automatically generates various graphs and visualizations during training. These particular figures were created by the DHMM model while searching for the best hyperparameters using Bayesian optimization.

Figure 1: This transition matrix shows the probabilities of switching between 4 hidden weather states, where the dark purple areas represent very low transition probabilities (under 0.05) and the bright yellow shows a state that almost always stays the same (0.98 probability).

Figure 2: This confusion matrix shows the model's precipitation predictions, correctly identifying 502 no–precipitation days and 297 precipitation days, with 213 false positives and 215 false negatives.

4.3 Hidden Markov Model with Gaussian mixture emissions (GMM HMM)

To improve seasonal predictions, new columns were also added specifically: month_sin, month_cos, day_sin, day_cos. These new dataset columns derived from dates were intended to help the model better identify summer, as it rains more frequently in summer than in other seasons. However, this feature engineering was unsuccessful and the model responded poorly, likely because it gave too much weight to these columns and subsequently created poor predictions. Therefore, this approach was abandoned and a static method was added that increases the probability of rain by 1.2 times during selected summer months.

It turned out that training the model only on dataset columns: Air Pressure, Total Precipitation, Snow Depth – was the most successful choice.

The author also tried to add other features such as average Air Pressure over the last 10 days or changes in pressure to help the model better find hidden states, but this method again did not bring

any improvement – rather the opposite.

The implementation had many complications, mainly because of model stability issues. Finally, the problem was solved, but the implementation had to be done more simply using *tied* covariance matrix instead of *full*.

4.4 Hidden Markov Model with Multivariate Gaussian Emissions trained using Variational Inference (VGHMM)

This model used the most advanced *full* covariance matrix type, which is typically less stable. However, thanks to Variational Inference training, no stability problems occurred. The model runs significantly faster than GMM HMM but has slightly lower accuracy.

This was the only model where experimentation with more than 5 hidden states was conducted. Testing was performed with up to 10 states successfully, and only at 12 states did the model become unstable. This represents a significant improvement in stability compared to other models.

5 Results

Model	Accuracy	F1-Score	Recall
DHMM	0.6453	0.4945	0.4448
GMM HMM	0.6491	0.4705	0.3988
VGHMM	0.6447	0.4433	0.3632

Table 1: Best model performance via Bayesian Optimization, maximizing accuracy.

Model	Accuracy	F1-Score	Recall
Approach A	0.6127	0	0
Approach B	0.3873	0.5583	1.0000

Table 2: Approach A: Every day predicted as no-precipitation day. Approach B: Every day predicted as precipitation day.

5.1 Which model is best?

It really depends on the use case, but in this case, the optimization was performed for accuracy.

A short story can be considered to represent model performance. Two friends, Emma and James. Emma makes a bet with James – he has to predict whether there will be precipitation each day for 25 years (our dataset length). For each correct prediction, Emma gives James 10 dollars.

Model	Accuracy	Earnings
Approach A	0.6127	\$55,908
Approach B	0.3873	\$35,341
DHMM	0.6453	\$58,883
GMM HMM	0.6491	\$59,230
VGHMM	0.6447	\$58,828

Table 3: The calculation of earnings is: *days per year * 25 years * \$10 * accuracy*

For this case, the best choice for James is the GMM HMM model, which would earn 5.22% more than the naive Approach A and 66.4% more than the naive Approach B.

6 Conclusion

The paper published by Zhang et al. (4) presents arguments that Long Short–Term Memory (LSTM) models using HMM are better for time series prediction than standalone models, which is a fascinating direction for future extension of this work. Try to extend the work with an ensemble model using LSTM and HMM together, with HMM for discovering hidden states and regimes of the time series and LSTM for subsequent enhancement and prediction.

The precipitation data was highly stochastic and challenging to predict. Unlike professional meteorologists who use satellite imagery, multi–altitude measurements, and radar data, this project relied on basic historical measurements from a single weather station. Despite these limitations, the GMM HMM model achieved 0.6491 accuracy, successfully exceeding the baseline.

I was hoping for even better results by breaking the accuracy barrier of 0.7, but I didn't manage to achieve that. However, I am still extremely happy that I chose this topic because I learned a lot. Since this was my first time working with Hidden Markov Models, everything was completely new to me, but that made the experience even more valuable.

References

[1] Gael Varoquaux Sergei Lebedev Antony Lee Matthew Danielson David Cournapeau, Fabian Pedregosa. hmmlearn: Documentation for implementating hmm in python. online, 2010. `https://hmmlearn.readthedocs.io/en/latest/`.

[2] Meteostat. Weather data for prague from meteostat. `https://meteostat.net/en/place/cz/prague?s=11518&t=2025-05-08/2025-05-15`, 2025.

[3] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. *optuna: Documentation for Bayesian Optimalization in Python*. 2019. `https://optuna.org`.

[4] Junhuan Zhang, Jiaqi Wen, and Zhen Yang. China's gdp forecasting using long short term memory recurrent neural network and hidden markov model. *PLOS ONE*, 17(6):e0269529, 2022.

[5] Walter Zucchini, Iain L. MacDonald, and Roland Langrock. *Hidden Markov Models for Time Series: An Introduction Using R*. Chapman & Hall/CRC Monographs on Statistics and Applied Probability. Chapman and Hall/CRC, Boca Raton, FL, 2 edition, 2016. ISBN 978-1482253832. Second Edition.