Items-Legit
A Better Recommender
...

Presented by: Gretchen Riggs, Scott Sullivan, Jonathan Hersch

Overview

At Items-Legit, we've been using a Mean-of-Means Product Recommender for predicting what are the best products to introduce to customers for 5 years now.

Since the time of this Recommender implementation, great strides have been made in Recommender systems, which have been deployed by many of our competitors.

Deploying a Probabilistic Matrix Factorization (PMF) Recommender system, using Markov Chain Monte Carlo (MCMC) sampling, will provide our customers with more attractive recommendations, increasing their likelihood of purchase and thus increasing our revenue.

What is Collaborative Filtering using PMF?

Collaborative Filtering

Collaborative filtering is a method of making automatic predictions about the interests of a user by collecting preferences or taste information from many users, many which will have similar preferences or tastes.

Probabilistic Matrix Factorization (PMF)

PMF is a probabilistic perspective on collaborative filtering, which is based on the idea that a user's preferences are based on a small number of latent factors.

A user's preferences are modeled by linearly combining these latent factor vectors using user-specific coefficients.

This results in a more personalized approach to our customers' recommendations.

But first, we must use MAP Estimation:

Maximum a Posteriori (MAP) Estimation

We need to model our probability distribution using MAP Estimation. This entails using Bayes Theorem to estimate our model parameters given our data:

$$P(\theta_i \mid Y) \sim P(Y \mid \theta_i) * P(\theta_i),$$

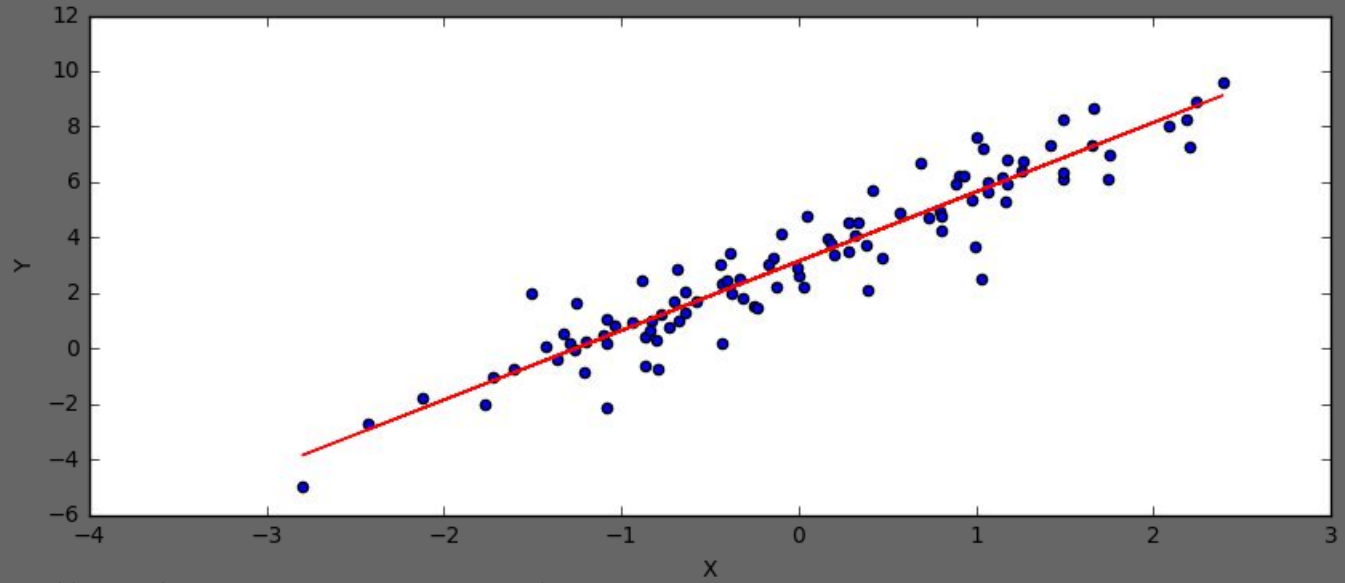
for parameters θ_i, i=1, 2, ..., n where n is the number of parameters

$P(\theta_i)$ is our prior belief in the values of our parameters

PyMC3 Basic Linear Example

- $Y \sim N(\mu, \sigma)$
- $\mu = \alpha + \beta X$

- $\alpha = 3.14$
- $\beta = 2.5$
- $\sigma = 1$



Here is a basic linear example of how the MAP Estimation works.

PyMC3's MAP Estimation models which values for α, β, σ best fit our data Y: $P(Y \mid \alpha, \beta, \sigma)$.
It then uses this model and the prior belief in the parameters α, β, σ, denoted $P(\alpha, \beta, \sigma)$, to give us our predicted probabilities of the parameters given our data, $P(\alpha, \beta, \sigma \mid Y)$.

Basic Model Specs

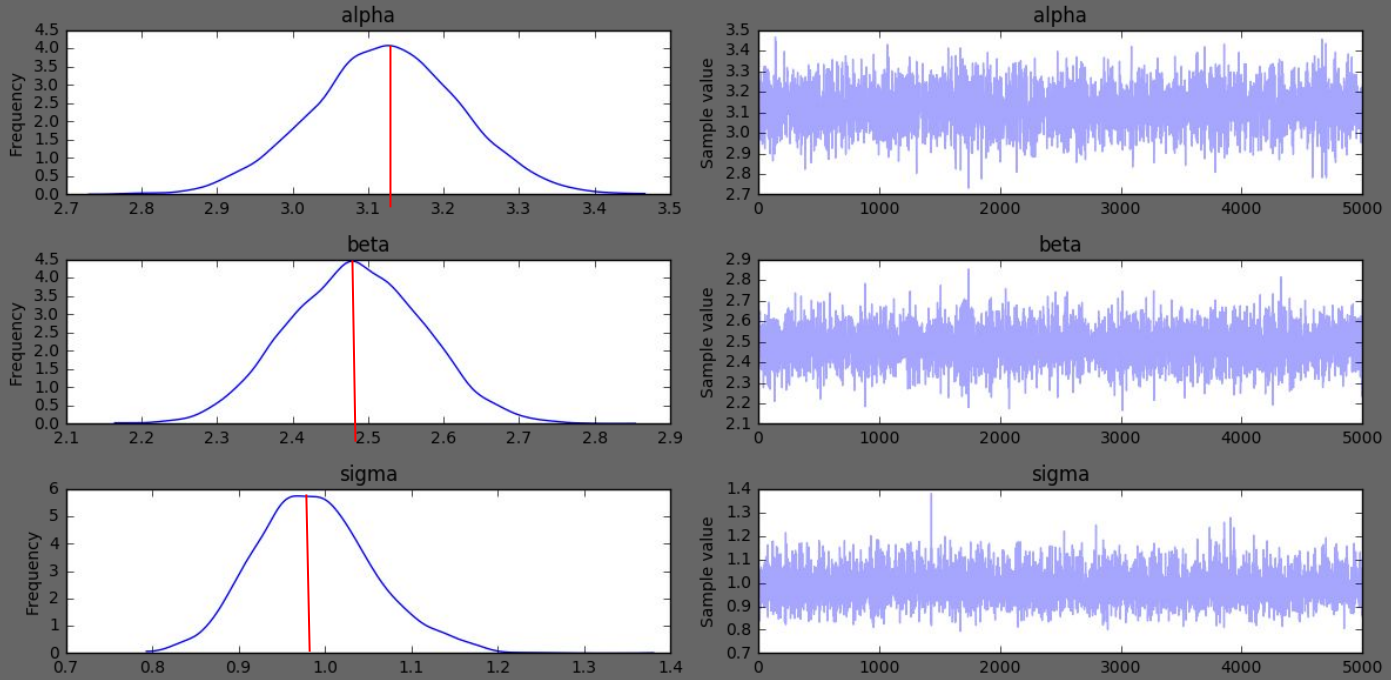
- Set prior distributions on α, β, σ
  - $\alpha \sim N(0,10)$
  - $\beta \sim N(0,10)$
  - $\sigma \sim |N(0,1)|$
- Set expected value of mean of Y, shown as μ here:
  - $\mu = \alpha + \beta X$
- These specifications allow for initial approximations of α, β, and σ to be calculated
- Model tells us $P(Y \mid \alpha, \beta, \sigma)$
- Goal is to determine $P(\alpha, \beta, \sigma \mid Y)$, using Bayes Theorem:
  - $P(\alpha, \beta, \sigma) \mid Y) \sim P(Y \mid \alpha, \beta, \sigma) * P(\alpha, \beta, \sigma)$

Maximum a Posteriori (MAP) Results

- $Y \sim N(\alpha + \beta X, \sigma)$
- Actual Parameter Values: $\alpha = 3.14$, $\beta = 2.5$, $\sigma = 1$
- Estimated Parameter Values: $\alpha^* = 3.12$, $\beta^* = 2.48$, $\sigma^* = 0.97$
- Good approximation for this example, but this is not always the case
- How can the MAP result be improved?

Markov Chain Monte Carlo (MCMC) Sampling

- Create distributions for α, β, and σ
- Choose values for α, β, and σ that maximize their probability



Prototype Input

The prototype test input was a popular joke ratings dataset, consisting of 1000 users rating 100 jokes.

To create a test dataset for validation, 20% of the ratings were randomly selected to be set to NaN so that once a model was created, we could assess if the predicted ratings matched the true user ratings and compute the error metric RMSE to judge the accuracy of our model.

Probabilistic Matrix Factorization (PMF) - Bayesian Model

- Ratings R_{ij} - users i, items j
- Users U_{ik} - users i, latent topics k
- Items V_{jk} - items j, latent topics k
- Fit probabilistic model:
  - $R_{ij} \sim N(\text{mean} = U_{ik} * V_{jk}, \sigma = 1/\alpha)$
  - $U_{ik} \sim N(\text{mean} = 0, \sigma_U = 1/\alpha_U)$
  - $V_{jk} \sim N(\text{mean} = 0, \sigma_V = 1/\alpha_V)$
- Each element of U_{ik}, V_{jk} is a parameter
- Model tells us $P(R_{ij} | U_{ik}, V_{jk})$
- Goal is to compute $P(U_{ik}, V_{jk} | R_{ij})$

Example User-Item Matrix (Item=Movies): R_{ij}

Users ↓ Movies ->	Xanadu	Love Actually	Warriors	Velvet Goldmine	Jesus Christ Superstar
Belle	5	1	3	4	5
Sebastian	1	1	3	1	2
Jeff	2	3	4	5	2
Tweedy	4	2	1	3	5
Robert	1	2	1	5	3
Smith	3	1	2	1	1

Example User-Item Matrix - Creating Training Dataset

Users ↓ Movies ->	Xanadu	Love Actually	Warriors	Velvet Goldmine	Jesus Christ Superstar
Belle	5	1	3	4	5
Sebastian	1	NaN	3	1	NaN
Jeff	NaN	3	4	5	2
Tweedy	4	2	NaN	3	5
Robert	1	NaN	1	5	3
Smith	3	1	NaN	1	1

Example Matrix Factorization - Matrix U_{ik}

Users ↓ / Latent Factors ->	λ_{U1}	λ_{U2}	λ_{U3}	...	λ_{Uk}
Belle	.19	.45	.33	...	.28
Sebastian	.35	.41	.28	...	.82
Jeff	.21	.81	.73	...	.55
Tweedy	.70	.90	.81	...	.33
Robert	.82	.20	.11	...	.90
Smith	.11	.38	.32	...	.71

Example Matrix Factorization - Matrix V_{jk}

Items ↓ / Latent Factors ->	λ_{V1}	λ_{V2}	λ_{V3}	...	λ_{Vk}
Xanadu	.23	.27	.48	...	.81
Love Actually	.10	.33	.87	...	.76
Warriors	.98	.13	.17	...	.75
Velvet Goldmine	.27	.84	.82	...	.13
Jesus Christ Superstar	.17	.46	.76	...	.76

The Computations Defining our PMF Model

$$P(R \mid U, V, \alpha^2) = \prod_{i=1}^{N} \prod_{j=1}^{M} \left[\mathcal{N}(R_{ij} \mid U_i V_j^T, \alpha^{-1}) \right]^{I_{ij}}$$

$$P(U \mid \alpha_U^2) = \prod_{i=1}^{N} \mathcal{N}(U_i \mid 0, \alpha_U^{-1} \boldsymbol{I})$$

$$P(V \mid \alpha_U^2) = \prod_{j=1}^{M} \mathcal{N}(V_j \mid 0, \alpha_V^{-1} \boldsymbol{I})$$

MAP Estimation

Minimizing this cost function to a local minima gives us our MAP Estimate:

$$E = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{M} I_{ij}(R_{ij} - U_i V_j^T)^2 + \frac{\lambda_U}{2}\sum_{i=1}^{N}\|U\|_{Fro}^2 + \frac{\lambda_V}{2}\sum_{j=1}^{M}\|V\|_{Fro}^2$$

where $\lambda_U = \alpha_U/\alpha, \lambda_V = \alpha_V/\alpha$, and $\|\cdot\|_{Fro}^2$ denotes the Frobenius norm

Markov Chain Monte Carlo Sampling

Next, we use our MAP Estimation as a starting point for our MCMC (Markov Chain Monte Carlo) Sampler. This sampling routine grabs random samples from our complex PMF model to create distributions for each of our modeling parameters. This sampling routine is run many times to get a large enough sample for each distribution.

The value with the highest probability in each parameter probability distribution is then chosen as the value for the parameter for our final model.

Baselines for Comparison

We created 3 Baseline models to use for rating prediction with the following methods:

- Uniform
  - If Ratings unknown, fill in with a Uniform Random value
- Global Mean
  - If Ratings unknown, fill in with the Global Mean: the mean of all the Ratings for all Users & Movies
- Mean of Means
  - If Ratings unknown, fill in with the Means of Means:
    Global Mean * Mean of each User's Ratings * Mean of each Movie's Ratings

The Recommender then takes the Z highest predicted ratings for each user as its recommendations for the item in question.

Model Performance on Joke Data

Baseline	RMSE
• Uniform	7.82
• Global Mean	5.25
• Mean of Means	4.81

Probabilistic Matrix Factorization

• MAP	4.01

Conclusion

As you can see, there is a great improvement in Recommender predictive power when using the more personalized PMF method for Product Recommendation.

Our prototype results show that there is great promise in using this new system, opening up more opportunities for Growth in Sales, joining our peers in putting forth the latest and greatest technology in growing our company.