Recommendation System
Week 10

Group 3: Xing Hong, Mingyu Li, Dongming Li, Hantian Zhang, Ziyu Liu

- **Project introduction**
- Data (AutoEncoder)
- IPS (Logistic Regression)
- Joint Learning (Use Stochastic Gradient Descent for each)
  - FM(theta) -> Double robust loss
  - EIB(phi)   -> EIB loss

Spider-Man: No Way Home

PG-13 2021 · Action/Adventure · 2h 28m

Overview | Showtime

Reviews

94% Rotten Tomatoes | 8.8/10 IMDb | 4/5 Common Sense Media

Audience rating summary

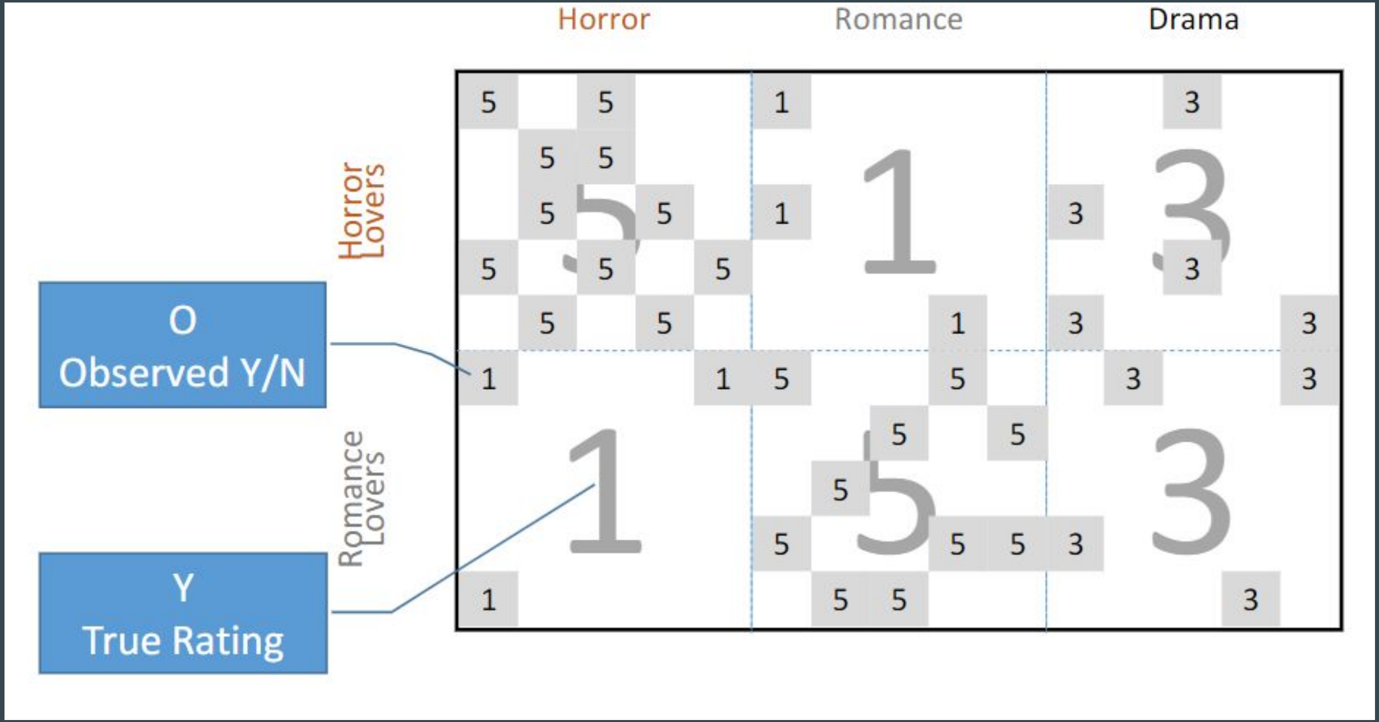
4.8

38099 ratings

Project introduction

- Research on methods of solving data scarcity caused by MNAR(missing not at random)
- Motivation: Creating a better unbiased estimator for recommendations
- Methods:
    - Imputation error (EIB)
    - Inverse Propensities Score (IPS)
    - Double Robust (DB)

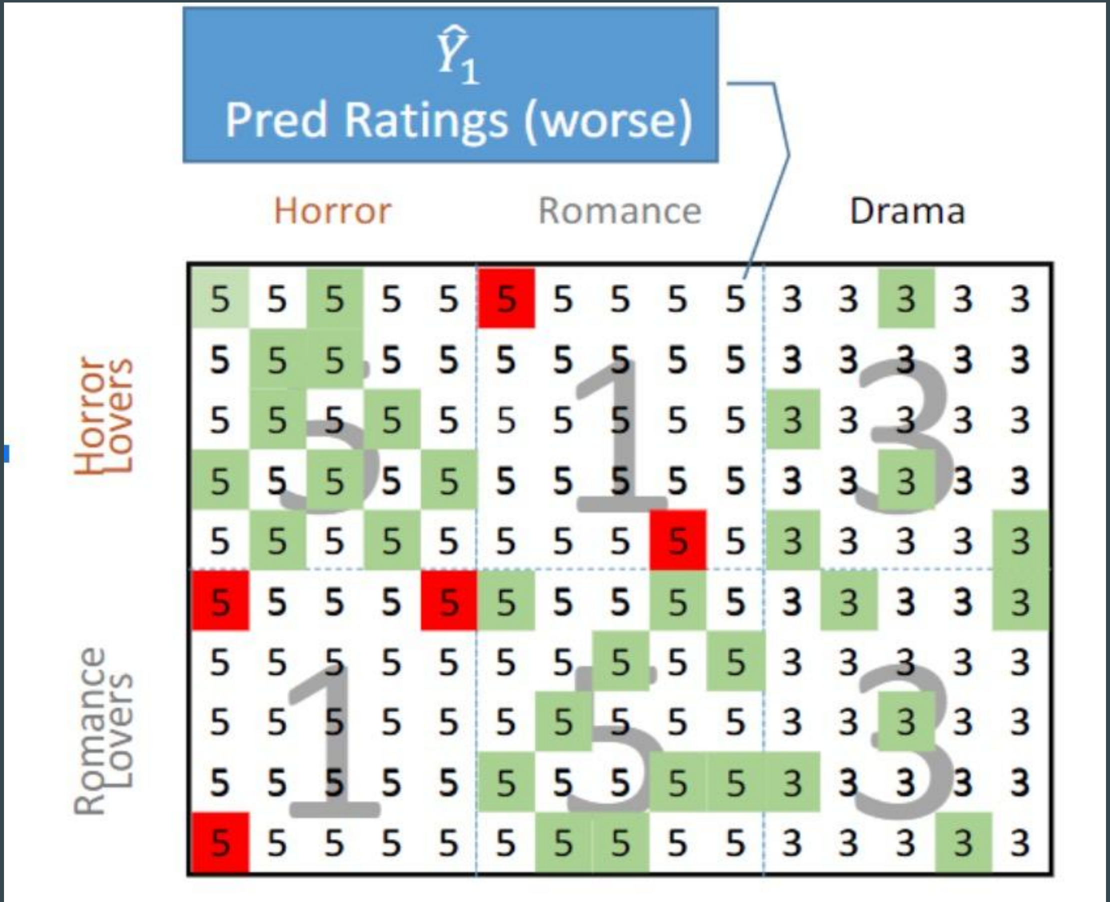
Observed losses are misleading due to selection bias



Example adapted from (Schnabel et al., 2016)

Due to Selection Bias, we would like to address recommendation issues based on MNAR(missing not at random) assumptions.

Observed losses are misleading due to selection bias



Naive method

The conventional practice is to estimate R(Y) using the average over only the observed entries

$$\hat{R}_{naive}(\hat{Y}) = \frac{1}{|\{(u,i): O_{u,i}=1\}|} \sum_{(u,i):O_{u,i}=1} \delta_{u,i}(Y,\hat{Y})$$

More generally, under selection bias, $\hat{R}_{naive}(\hat{Y})$ is not an unbiased estimate of the true performance.

$$\mathbb{E}_O\left[\hat{R}_{naive}(\hat{Y})\right] \neq R(\hat{Y})$$

Goal

Use a Doubly Robust estimator that has the ability to remain unbiased if either the imputed errors or propensities are accurate.

Doubly Robust Estimator

- Combine the EIB and IPS estimators (Wang et al. 2019)

$$\mathcal{E}_{\mathrm{DR}} = \mathcal{E}_{\mathrm{DR}}(\hat{\mathbf{R}}, \mathbf{R}^o) = \frac{1}{|\mathcal{D}|} \sum_{u,i \in \mathcal{D}} \left(\hat{e}_{u,i} + \frac{o_{u,i}\delta_{u,i}}{\hat{p}_{u,i}} \right)$$

- Given the imputed errors $\hat{\mathbf{E}} = \{\hat{e}_{u,i} | u, i \in \mathcal{D}\}$ and learned propensities $\hat{\mathbf{P}} = \{\hat{p}_{u,i} | u, i \in \mathcal{D}\}$, the DR estimator estimates the prediction inaccuracy P with above

- Project introduction
- **Data (AutoEncoder)**
- IPS (Logistic Regression)
- Joint Learning (Use Stochastic Gradient Descent for each)
    - FM(theta) -> Double robust loss
    - EIB(phi)   -> EIB loss

Dataset ml-100k

This data set consists of:

- 100,000 ratings (1-5) from 943 users on 1682 movies.
- Each user has rated at least 20 movies.
- Simple demographic info for the users (age, gender, occupation, zip)
- Types of movies

Rating Matrix

item_id	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
user_id																										
1	5.0	3.0	4.0	3.0	3.0	5.0	4.0	1.0	5.0	3.0	2.0	5.0	5.0	5.0	5.0	5.0	3.0	4.0	5.0	4.0	1.0	4.0	4.0	3.0	4.0	3.0
2	4.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2.0	NaN	NaN	4.0	4.0	NaN	NaN	NaN	NaN	3.0	NaN	NaN	NaN	NaN	NaN	4.0	NaN
3	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	4.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
5	4.0	3.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	4.0	NaN	NaN	NaN	3.0	NaN	NaN	4.0	3.0	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
939	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	5.0	NaN	NaN	NaN	NaN	NaN	5.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
940	NaN	NaN	NaN	2.0	NaN	NaN	4.0	5.0	3.0	NaN	NaN	4.0	NaN	3.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
941	5.0	NaN	NaN	NaN	NaN	NaN	4.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	4.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
942	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
943	NaN	5.0	NaN	NaN	NaN	NaN	NaN	NaN	3.0	NaN	4.0	5.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	4.0	4.0	4.0	NaN	NaN

943 rows × 1682 columns

Observed Matrix

user_id	1.0	2.0	3.0	4.0	5.0	6.0	7.0	8.0	9.0	10.0	11.0	12.0	13.0	14.0	15.0	16.0	17.0	18.0	19.0	20.0	21.0	22.0	23.0	24.0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	1	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0	0	1	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
5	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
939	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
940	0	0	0	1	0	0	1	1	1	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0
941	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
942	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
943	0	1	0	0	0	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0	1	1	1

943 rows × 1682 columns

Data Cleaning (AutoEncoder)

- User features: ['age', 'gender', 'occupation'] - 23 columns
- Item features: All sorts of movie types - 19 types

Adding up to 42 features!

Due to too much features and the sparse property of our data, dimension reduction is necessary.

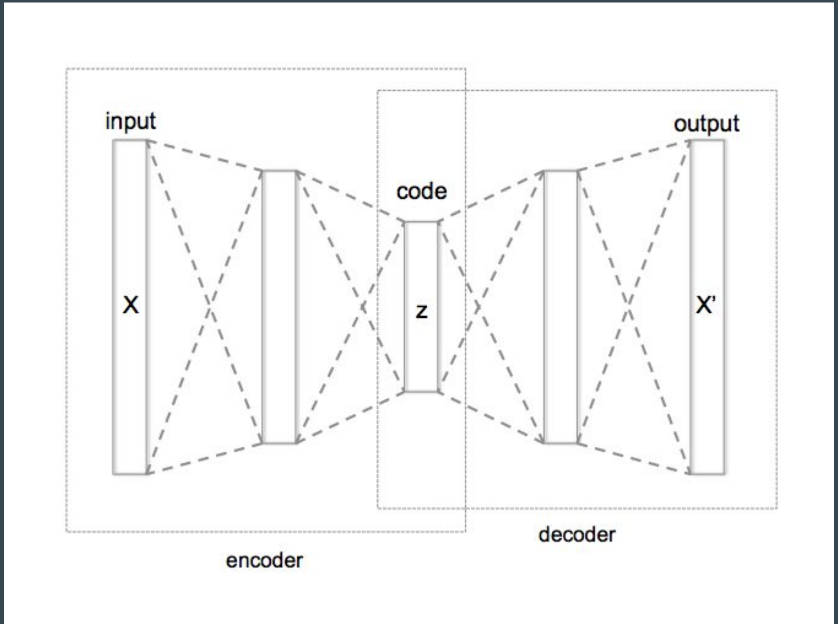
- Method: AutoEncoder
- Result: 2 user features(age, gender) + 10 occupation features + 8 movie types

AutoEncoder

An autoencoder is a type of artificial neural network used to learn efficient codings of unlabeled data

The encoding is validated and refined by attempting to regenerate the input from the encoding.

The autoencoder learns a representation (encoding) for a set of data, typically for dimensionality reduction

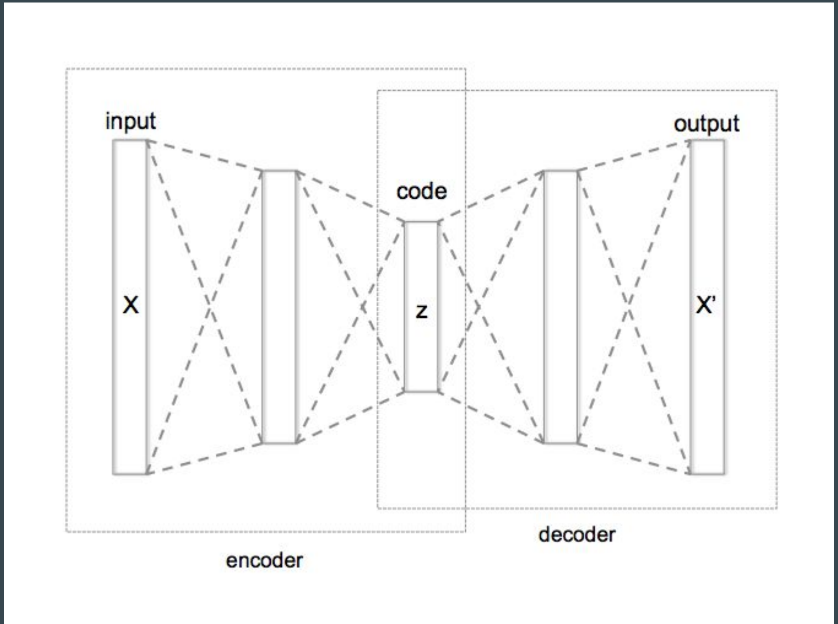


AutoEncoder

In this case, we reduce the dimension of users' occupation from 21 to 10,

Movie features from 19 to 8, separately

Adding up 20 dimensions for our input feature vectors for later modeling



2 Recent Approaches for MNAR

1. IPS (inverse propensity scoring)

   "Inversely weights the prediction error for each observed rating with the propensity of observing that rating". (Dudik et al., 2011)

   a. often suffers from the large variance issues
   b. When estimated propensity is very small, it creates a very large value

2. EIB (error imputation based)

   "Computes an estimated value of the prediction error for each missing rating". (Steck, 2013)

   a. Hard to accurately estimate the imputed errors
   b. it's almost as hard as predicting the original ratings

Each of them is not perfect

Double Robust Method

1. **IPS part - design propensity model to get learned propensity:** $\hat{p}_{u,i}$
2. Joint learning train:

   2.1. Imputation model - get imputed error(eui_hat) based on φ(fi)

   Then train Imputation model to update φ(fi) with loss function:

$$\mathcal{L}_{\mathrm{e}}(\theta, \phi) = \sum_{u,i \in \mathcal{O}} \frac{(\hat{e}_{u,i} - e_{u,i})^2}{\hat{p}_{u,i}} + \nu \|\phi\|_F^2$$

   2.2 Prediction model(FM model) - get predicted rating(rui_hat) based on θ(theta)

   Then train prediction model to update θ(theta) with loss function:

$$\mathcal{L}_{\mathrm{r}}(\theta, \phi) = \sum_{u,i \in \mathcal{D}} \left(\hat{e}_{u,i} + \frac{o_{u,i}(e_{u,i} - \hat{e}_{u,i})}{\hat{p}_{u,i}} \right) + v\|\theta\|_F^2$$

Doubly Robust Estimator

- Combine the EIB and IPS estimators (Wang et al. 2019)

$$\mathcal{E}_{\mathrm{DR}} = \mathcal{E}_{\mathrm{DR}}(\hat{\mathbf{R}}, \mathbf{R}^o) = \frac{1}{|\mathcal{D}|} \sum_{u,i \in \mathcal{D}} \left(\hat{e}_{u,i} + \frac{o_{u,i}\delta_{u,i}}{\hat{p}_{u,i}} \right)$$

- Given the imputed errors $\hat{\mathbf{E}} = \{\hat{e}_{u,i} | u, i \in \mathcal{D}\}$ and learned propensities $\hat{\mathbf{P}} = \{\hat{p}_{u,i} | u, i \in \mathcal{D}\}$, the DR estimator estimates the prediction inaccuracy P with above

Propensity

Defined as the probability of observing the true rating by,

$$P_{u,i} = P\left(O_{u,i} = 1\right)$$

If the estimated propensities are accurate, the prediction error is unbiased

Propensities P

	Horror	Romance	Drama
Horror Lovers	p	p/10	p/2
Romance Lovers	p/10	p	p/2

Example adapted from (Schnabel et al., 2016)

IPS (Inverse Propensity Scoring)

Use Inverse-Propensity-Scoring Estimator (IPS) to obtain unbiased estimate:

$$\hat{R}_{IPS}(\hat{Y}|P) = \frac{1}{U \cdot I} \sum_{(u,i):O_{u,i}=1} \frac{\delta_{u,i}(Y,\hat{Y})}{P_{u,i}} .$$

Use IPS estimator that estimates the prediction inaccuracy:

$$\begin{aligned} \mathbb{E}_O\left[\hat{R}_{IPS}(\hat{Y}|P)\right] &= \frac{1}{U \cdot I} \sum_u \sum_i \mathbb{E}_{O_{u,i}}\left[\frac{\delta_{u,i}(Y,\hat{Y})}{P_{u,i}} O_{u,i}\right] \\ &= \frac{1}{U \cdot I} \sum_u \sum_i \delta_{u,i}(Y,\hat{Y}) = R(\hat{Y}) . \end{aligned}$$

Propensity Estimation

Experimental - Propensities are under control; known by design (e.g., ad placement)

Observational - Users self-select; need to estimate $P_{u,i}$

- Estimate parameter of binary random variables:

$$P_{u,i} = P\left(O_{u,i} = 1 \mid X, \tilde{Y}\right)$$

- Variety of models: Logistic Regression, Naïve Bayes, etc.

Debiasing Learning

- Empirical Risk Minimization(ERM)
- Training Objective of using ERM together with IPS :

$$\hat{Y}^{ERM} = \underset{\hat{Y} \in \mathcal{H}}{\operatorname{argmin}} \left\{ \hat{R}_{IPS}\left(\hat{Y} \mid P\right) \right\}$$

- Example: Matrix Factorization(MF) with MSE loss

$$\hat{Y}^{ERM} = \underset{V,W}{\operatorname{argmin}} \left\{ \sum_{O_{u,i}=1} \underbrace{\frac{1}{P_{u,i}}}_{\text{propensity weight}} \left(Y_{u,i} - V_u W_i\right)^2 + \lambda\left(\|V\|_F^2 + \|W\|_F^2\right) \right\}$$

Propensity Estimation via Naïve Bayes.

We assume that dependencies between covariates X, X_{hid} and other ratings are negligible.

$$P(O_{u,i} = 1 \mid Y_{u,i} = r) = \frac{P(Y = r \mid O = 1)P(O = 1)}{P(Y = r)} .$$

On the right side of the equation, we could obtain P(Y=r | O=1) and P(O=1) by simply counting the observed ratings, while P(Y=r) requires a sample of MCAR data.

However, the prediction is very unreliable

Logistic Regression

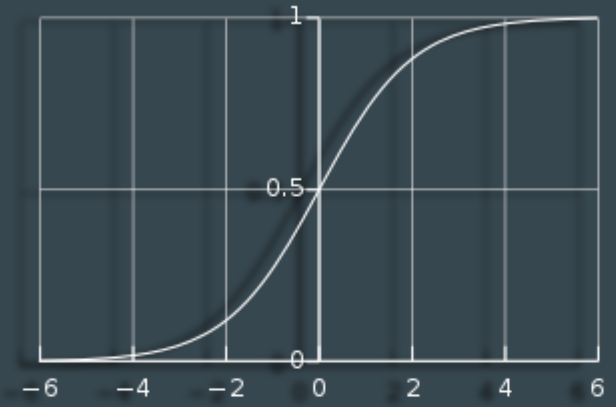
The main modeling assumption is that there exists a φ = (w, β, γ) such that

$$P_{u,i} = \sigma\left(w^T X_{u,i} + \beta_i + \gamma_u\right)$$

Here, $x_{u,i}$ is a vector encoding all observable information about a user-item pair (e.g., user demographics, category of a given movie, etc.)

$$w^T x + b = ln \frac{P(Y=1|x)}{1-P(Y=1|x)}$$

$$P(Y=1|x) = \frac{1}{1+e^{-(w^T x + b)}}$$



Observed Matrix

user_id	1.0	2.0	3.0	4.0	5.0	6.0	7.0	8.0	9.0	10.0	11.0	12.0	13.0	14.0	15.0	16.0	17.0	18.0	19.0	20.0	21.0	22.0	23.0	24.0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	1	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0	0	1	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
5	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
939	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
940	0	0	0	1	0	0	1	1	1	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0
941	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
942	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
943	0	1	0	0	0	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0	1	1	1

943 rows × 1682 columns

Propensities Estimation via. Logistic Regression

With the encoded Features, We used Logistic Regression to obtain our

propensity = (P_{u,i} = P(O_{u,i} = 1)

for all possible user-item pairs

```
[ ]  model.score(x, y)

     0.9369602414940553
```

Double Robust Method

1. IPS part - design propensity model to get learnt propensity: Pui_hat
2. Joint learning train:

2.1. Imputation model - get imputed error(eui_hat) based on φ(fi)

Then train Imputation model to update φ(fi) with loss function:

$$\mathcal{L}_{\mathrm{e}}(\theta, \phi) = \sum_{u,i \in \mathcal{O}} \frac{(\hat{e}_{u,i} - e_{u,i})^2}{\hat{p}_{u,i}} + \nu \|\phi\|_F^2$$

2.2 Prediction model(FM model) - get predicted rating(rui_hat) based on θ(theta)

Then train prediction model to update θ(theta) with loss function:

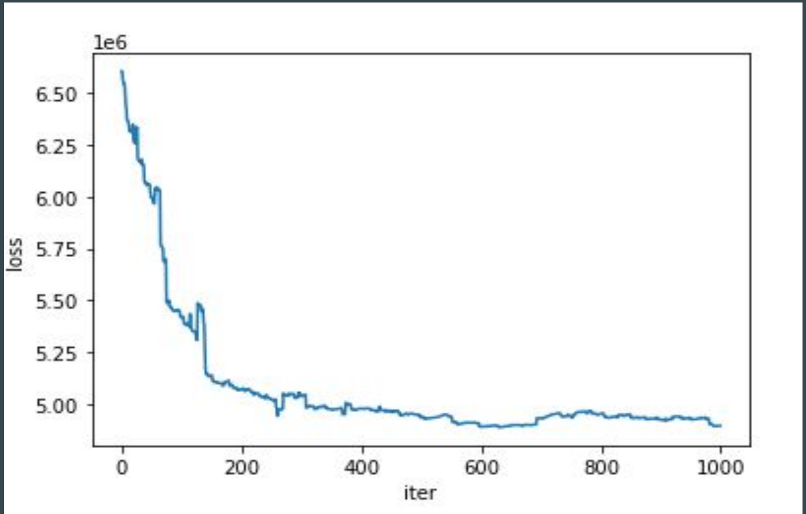
$$\mathcal{L}_{\mathrm{r}}(\theta, \phi) = \sum_{u,i \in \mathcal{D}} \left(\hat{e}_{u,i} + \frac{o_{u,i}(e_{u,i} - \hat{e}_{u,i})}{\hat{p}_{u,i}} \right) + \upsilon \|\theta\|_F^2$$

Imputation Part

- Compute imputed error for MSE: $\hat{e}_{u,i} = \omega(\hat{r}_{u,i} - \gamma)^2$ where ω and γ are hyper-parameters φ(phi).
- Train model with loss function $\mathcal{L}_{\mathrm{e}}(\theta, \phi) = \sum_{u,i \in \mathcal{O}} \frac{(\hat{e}_{u,i} - e_{u,i})^2}{\hat{p}_{u,i}} + \nu \|\phi\|_F^2$

  and stochastic gradient descent(SGD) method to update the φ(phi)

- Loss train result:
  - loss in a downward trend
  - training converges



Imputation partial derivative and SGD

As $\mathcal{L}_e(\theta, \phi) = \sum_{u,i \in \mathcal{O}} \frac{(\hat{e}_{u,i} - e_{u,i})^2}{\hat{p}_{u,i}} + \nu \|\phi\|_F^2$, and $\hat{e}_{u,i} = \omega(\hat{r}_{u,i} - \gamma)^2$, the partial derivative are:

$$\frac{1}{p} \cdot \frac{\partial}{\partial w} \left(\left(w(r-y)^2 - e \right)^2 \right) = 1/p * 2\left(wr^2 - 2wry + wy^2 - e \right)\left(r^2 - 2ry + y^2 \right)$$

$$\frac{1}{p} \cdot \frac{\partial}{\partial y} \left(\left(w(r-y)^2 - e \right)^2 \right) = 1/p * 2\left(wr^2 - 2wry + wy^2 - e \right)\left(2wy - 2wr \right)$$

- Stochastic gradient descent (SGD): Each iteration can update parameters with only one training data, therefore, provide a fast and smooth optimization result for training imputation error model

- Data (AutoEncoder)
- **Double Robust Method**
    - IPS (Logistic Regression)
    - Joint Learning (Use Stochastic Gradient Descent for each)
        - FM(theta) -> Double robust loss
        - EIB(phi) -> EIB loss

Doubly Robust Estimator

- Combine the EIB and IPS estimators (Wang et al. 2019)

$$\mathcal{E}_{\mathrm{DR}} = \mathcal{E}_{\mathrm{DR}}(\hat{\mathbf{R}}, \mathbf{R}^o) = \frac{1}{|\mathcal{D}|} \sum_{u,i \in \mathcal{D}} \left(\hat{e}_{u,i} + \frac{o_{u,i}\delta_{u,i}}{\hat{p}_{u,i}} \right)$$

- Given the imputed errors $\hat{\mathbf{E}} = \{\hat{e}_{u,i} | u, i \in \mathcal{D}\}$ and learned propensities $\hat{\mathbf{P}} = \{\hat{p}_{u,i} | u, i \in \mathcal{D}\}$, the DR estimator estimates the prediction inaccuracy P with above

Proof of the effectiveness of Doubly Robust Estimator

By definition, the bias of the DR estimator is

$$\mathrm{Bias}(\mathcal{E}_{\mathrm{DR}}) = \left|\mathcal{P} - \mathbb{E}_{\mathbf{O}}[\mathcal{E}_{\mathrm{DR}}]\right|.$$

$$\mathcal{P} = \mathcal{P}(\hat{\mathbf{R}}, \mathbf{R}^f) = \frac{1}{|\mathcal{D}|}\sum_{u,i\in\mathcal{D}} e_{u,i}\,,$$

Expectation of O(u,i) equals the propensity

$$\mathbb{E}_{\mathbf{O}}[\mathcal{E}_{\mathrm{DR}}] = \frac{1}{|\mathcal{D}|}\sum_{u,i\in\mathcal{D}}\left(\hat{e}_{u,i} + \frac{p_{u,i}\delta_{u,i}}{\hat{p}_{u,i}}\right).$$

$$\mathrm{Bias}(\mathcal{E}_{\mathrm{DR}}) = \frac{1}{|\mathcal{D}|}\left|\sum_{u,i\in\mathcal{D}}\Delta_{u,i}\delta_{u,i}\right|.$$

Bias of different estimators

$\mathcal{E}_{\mathrm{EIB}}$	$\mathcal{E}_{\mathrm{IPS}}$	$\mathcal{E}_{\mathrm{DR}}$
$\left\lvert\sum_{u,i\in\mathcal{D}}\frac{(1-p_{u,i})\delta_{u,i}}{\lvert\mathcal{D}\rvert}\right\rvert$	$\left\lvert\sum_{u,i\in\mathcal{D}}\frac{\Delta_{u,i}e_{u,i}}{\lvert\mathcal{D}\rvert}\right\rvert$	$\left\lvert\sum_{u,i\in\mathcal{D}}\frac{\Delta_{u,i}\delta_{u,i}}{\lvert\mathcal{D}\rvert}\right\rvert$

Double Robust Method

1. IPS part - design propensity model to get learnt propensity: Pui_hat
2. Joint learning train:

   2.1. Imputation model - get imputed error(eui_hat) based on φ(fi)

   Then train Imputation model to update φ(fi) with loss function:

$$\mathcal{L}_{\mathrm{e}}(\theta, \phi) = \sum_{u,i \in \mathcal{O}} \frac{(\hat{e}_{u,i} - e_{u,i})^2}{\hat{p}_{u,i}} + \nu \|\phi\|_F^2$$

   2.2 Prediction model(FM model) - get predicted rating(rui_hat) based on θ(theta)

   Then train prediction model to update θ(theta) with loss function:

$$\mathcal{L}_{\mathrm{r}}(\theta, \phi) = \sum_{u,i \in \mathcal{D}} \left(\hat{e}_{u,i} + \frac{o_{u,i}(e_{u,i} - \hat{e}_{u,i})}{\hat{p}_{u,i}} \right) + \upsilon \|\theta\|_F^2$$

Rating Prediction: Prevalent Model vs Factorization Machine

Prevalent Model

$$y = \omega_0 + \sum_{i=1}^{n} \omega_i x_i$$

One of the most common models is probably linear regression. For those prevalent models, they ignore the relationship among different features.

Factorization Machine

$$\hat{y}(\mathbf{x}) := w_0 + \sum_{i=1}^{n} w_i \, x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \langle \mathbf{v}_i, \mathbf{v}_j \rangle \, x_i \, x_j$$

With the last term, this new equation could reflect how different features would influence each other.

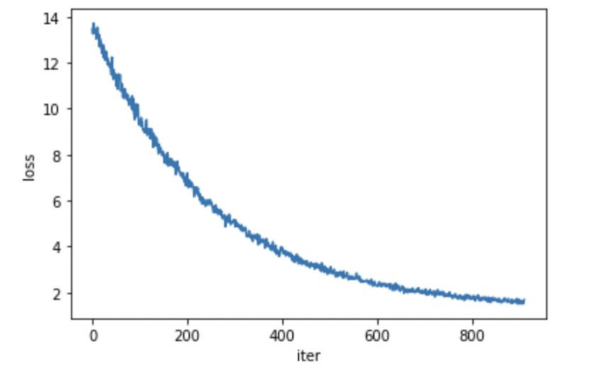
Factorization Machine

We use FM to build up a prediction model in order to obtain prediction rating

Error: e(u,i) $\left(f_\theta(\boldsymbol{x}_{u,i}) - r_{u,i}\right)^2$

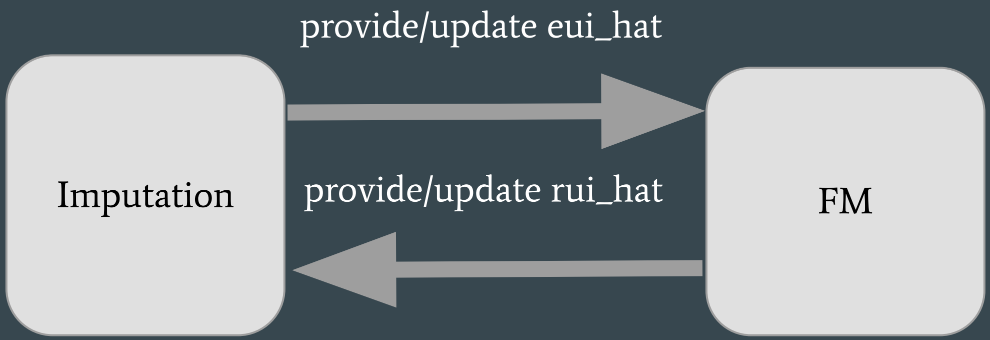
Loss Function for Doubly Robust(used for joint learning)

$$\mathcal{L}_{\mathrm{r}}(\theta, \phi) = \sum_{u,i \in \mathcal{D}} \left(\hat{e}_{u,i} + \frac{o_{u,i}(e_{u,i} - \hat{e}_{u,i})}{\hat{p}_{u,i}} \right) + v\|\theta\|_F^2$$




```
imputed_error = tf.reduce_sum(tf.multiply(theta[0],tf.square(y_hat - theta[1])))
inner = tf.subtract(tf.subtract(y, y_hat), imputed_error)
error = tf.divide(tf.add(imputed_error, tf.multiply(O_ui, inner)), P_ui)
```

Joint Learning between Imputation model and FM



Algorithm 1 Alternating Training for Joint Learning

input: observed ratings $\mathbf{R}^o$ and learned propensities $\hat{\mathbf{P}}$
while *stopping criteria is not satisfied* **do**
  for *number of steps for training the imputation model* **do**
    Sample a batch of user-item pairs $\{(u_j, i_j)\}_{j=1}^{J}$ from $\mathcal{O}$
    Update ϕ by descending along the gradient $\nabla_\phi \mathcal{L}_e(\theta, \phi)$
  end for
  for *number of steps for training the prediction model* **do**
    Sample a batch of user-item pairs $\{(u_k, i_k)\}_{k=1}^{K}$ from $\mathcal{D}$ [2]
    Update θ by descending along the gradient $\nabla_\theta \mathcal{L}_r(\theta, \phi)$
  end for
end while

where $e_{u,i} = (f_\theta(\boldsymbol{x}_{u,i}) - r_{u,i})^2$, $\hat{e}_{u,i} = (f_\theta(\boldsymbol{x}_{u,i}) - g_\phi(\boldsymbol{x}_{u,i}) - \perp(f_\theta(\boldsymbol{x}_{u,i})))^2$ with $\perp$ the operator that sets the gradient of the operand to zero so $\nabla_\theta \perp(f_\theta(\boldsymbol{x}_{u,i})) = 0$ and $\perp(f_\theta(\boldsymbol{x}_{u,i})) = f_\theta(\boldsymbol{x}_{u,i})$ [1], $\upsilon \geq 0$, and $\|\cdot\|_F^2$ is the Frobenius norm. Meanwhile, we also learn the parameters of

Before entering this "while loop", we need to initialize those hyper-parameters

Conclusion

- We are dealing with data MNAR recommendation
- Propose a doubly robust estimator that uses imputed errors and propensities to estimate the prediction inaccuracy.
- Propose a joint learning algorithm that "learns rating prediction and error imputation jointly to guarantee a low prediction inaccuracy". (Wang)
- Bias of estimating the prediction inaccuracy is effectively and reliably reduced with the utilization of real world data in our model to achieve a greater purpose:

A Better Recommendation System

References

Dud´ık, M., Langford, J., and Li, L. Doubly robust policy evaluation and learning. In *ICML*, pp. 1097–1104, 2011.

Schnabel, Tobias. Recommendations as Treatments: Debiasing Learning and Evaluation.

Steck, Harald. Evaluation of recommendations: rating- prediction and ranking. In *RecSys*, pp. 213–220, 2013.

Wang, Xiaojie. Doubly Robust Joint Learning for Recommendation on Data Missing Not at Random. 2019.

Thank you