Data Science with Python: Exploratory Analysis of Movie Recommendation

STAT 535 Final Project

Bing Miu, Rong Zhang, Meilan Chen, Jinchao Feng

Department of Mathematics and Statistics
Department of Biostatistics and Epidemiology
University of Massachusetts Amherst

April 23, 2018

Outline

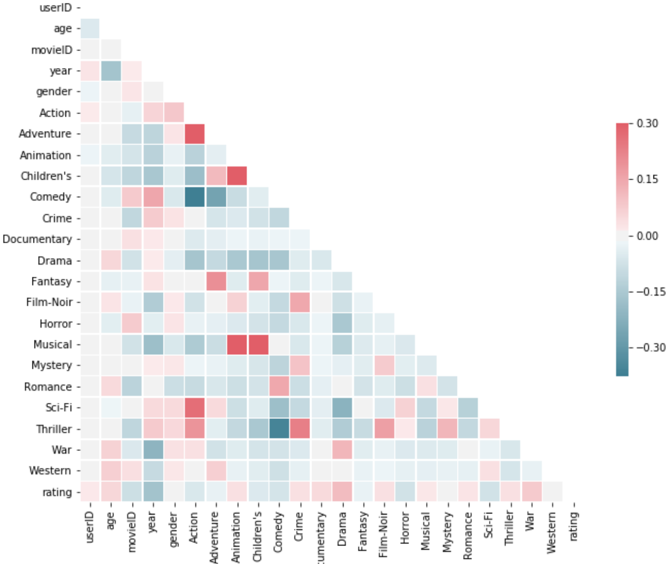
Goal and Objective

- Goal: Exploring the best model to predict future movie ratings
- Build different models:
  - Linear models: linear regression, lasso and ridge regression,
  - Other models: random forest, matrix factorization
- Cross-Validation: Train-Test Split with size 0.2
- Choose the best model based on MSE of validation set

Data Description

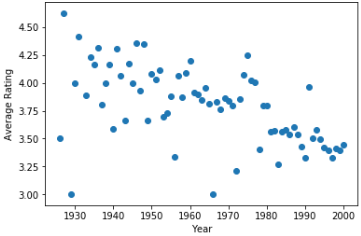
- 31620 observations, 10 variables in the dataset.
- Key variables:
  - Rating(response): 1, 2, 3, 4, 5
  - userID: 2353 users
  - movieID: 1465 movies
  - Age group: 1, 18, 25, 45, 35, 50, 56
  - Gender: M / F
  - Year: 1926-2000
  - name
  - genre1, genre2, genre3...
- Data transformation: transform gender, genre1, genre2, genre3 into dummy variable. (after that we have 25 variables)

Correlation

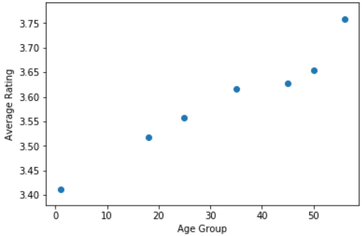


Data Visualization

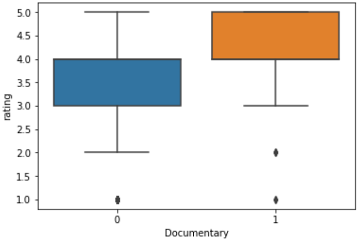
Average Rating vs Year



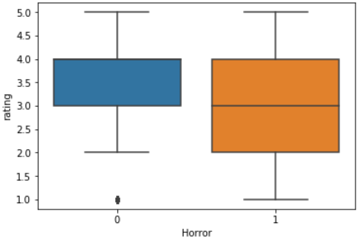
Average Rating vs Age Group



Documentary Movie Rating



Horror Movie Rating



Outline

Linear Regression

- Models the relationship between outcomes and predictors by fitting a linear equation to the observed data
- Linear Regression Formula:

$$\hat{Y}_i = \hat{\beta}_0 + \hat{\beta}_1 X_{i1} + \hat{\beta}_2 X_{i2} + \cdots + \hat{\beta}_{p-1} X_{i,p-1} + \hat{\epsilon}_i$$

- Advantages:
  - Easy to implement
  - Straightforward interpretations
- Disadvantages: Many assumptions
  - Linear relationship
  - Constant variance: $var(\epsilon_i) = \sigma^2$
  - Uncorrelated errors: $cov((\epsilon_i, \epsilon_j) = 0$
  - Probability distribution for the error: $\epsilon_i \sim N(0, \sigma^2)$
  - No or little multicollinearity

Linear Regression

Model summary : Backward Feature Selection

i	Predictors	BIC	MSE
1	age +year +F +Action +Adventure +Animation +Children's +Comedy +Crime + Documentary +Drama +Fantasy +Film-Noir +Horror +Musical +Mystery +Romance +Sci-Fi +Thriller +War +Western	75840	1.1500
2	- Musical	75840	1.1500
3	- Musical, -Mystery	75830	1.1499
4	- Musical, -Mystery, -Adventure	75820	1.1500
5	- Musical, -Mystery, -Adventure, -F	75810	1.1501
6	- Musical, -Mystery, -Adventure, -F, -age	75800	1.1502

$$\hat{Y}_i = 29.815 - 0.013 year - 0.143 Action + 0.467 Animation - 0.467 Children's - 0.087 Comedy + 0.175 Crime + 0.687 Documentary + 0.202 Drama + 0.090 Fantasy - 0.138 Film - Noir - 0.377 Horror + 0.088 Romance - 0.080 Sci - Fi + 0.167 Thriller + 0.145 War - 0.108 Western$$

Outline

Ridge and Lasso Regression

- Ridge Regression

$$min(\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 + \lambda \sum_{j=1}^{p} \beta^2)$$

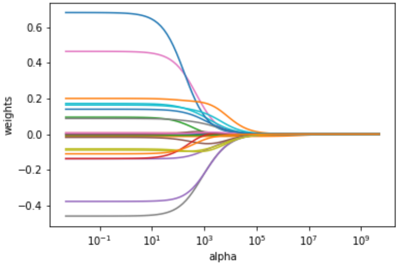
- Lasso Regression

$$min(\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 + \alpha \sum_{j=1}^{p} \mid \beta \mid)$$

Regularization and Parameters

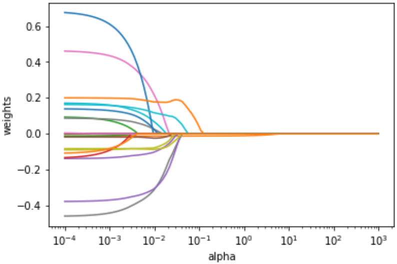
Ridge Regression

$$\lambda = 9.7$$



Lasso Regression

$$\alpha = 0.0003$$



Coefficients and Performance

Part of Lasso Coefficients

Documentary	0.639399
Drama	0.199288
Fantasy	0.0819556
Film-Noir	-0.110015
Horror	-0.37249
Musical	0
Mystery	0
Romance	0.0836925

Ridge MSE=1.150
Lasso MSE=1.149

Lasso drops the genres of Musical and Mystery and performs a little bit better than Ridge and OLS.

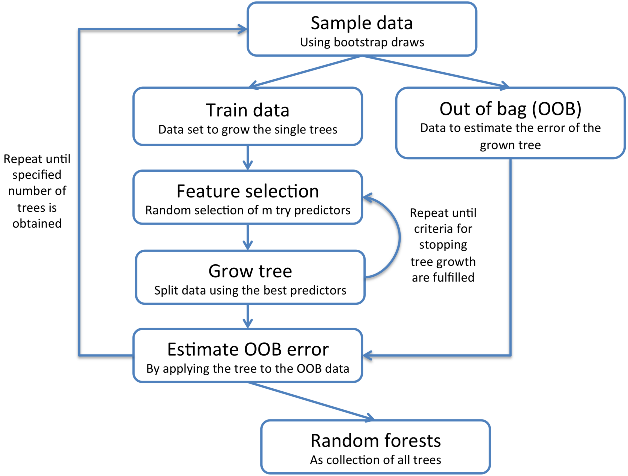
Generally, regularization is not very effective for this model because the coefficients are already very small.

Outline

Random Forest

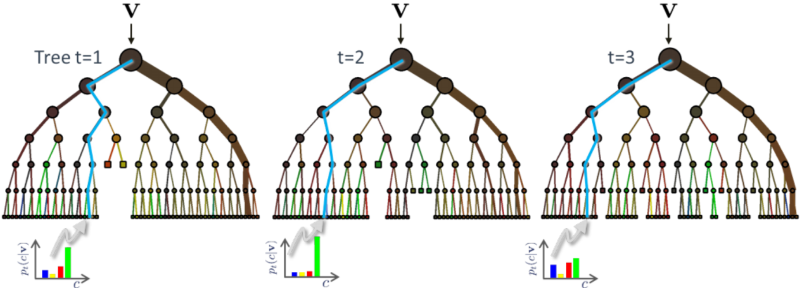
- A random forest is an ensemble of trees. Each tree consists of split nodes and leaf nodes.
- The random forest algorithm combines random decision trees with bagging (bootstrap) to achieve very high accuracy.
- Process
  - Bootstrap samples
  - At each split, bootstrap variables
  - Grow multiple trees and vote
- Tradeoff
  - Accuracy
  - Computation expensive.
  - Interpretation
  - Overfitting

Random Forest

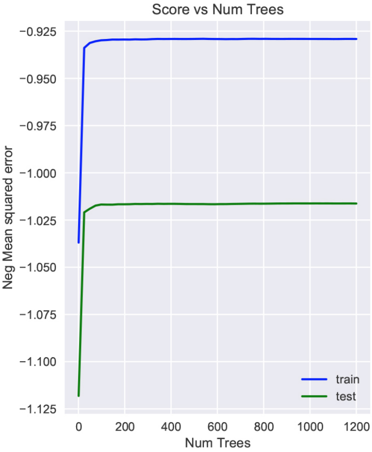


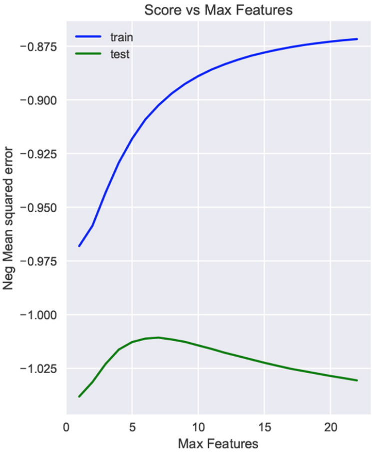
Random Forest

- The ensemble model
- Forest output probability: $P(c|v) = \frac{1}{T}\sum_t^T P_t(c|v)$



Hyperparamter Tuning





Outline

Basic Idea

userID \ movieID (rating)	1	2	3	4	5	6
4	4	1	4	5	3	2
5	3	3	5	3	4	3
6	1	5	2	4	2	4
15	5	4	3	1	3	5
17	2	3	4	4	4	5

Figure: A sample of matrix R

We want to approximate the matrix R as the product of two matrices:

$$R \approx PQ;$$

where P is an $N \times K$ and Q is a $K \times M$ matrix. This factorization gives a low dimensional numerical representation of both users and movies.

Sparse Data

movieID / rating / userID	1	2	3	4	5	6
4	-	-	-	-	-	-
5	-	-	-	-	-	-
6	-	-	-	-	-	4
15	-	-	-	-	-	5
17	-	-	-	-	-	-

Figure: Part of matrix R for our dataset

31620:3447145

In our case, we have lots of unknown ratings of R which cannot be represented by zero.

$$\hat{r} = \sum_{k=1}^{K} p_{uk} q_{ki}$$

$$e_{ui} = r_{ui} - \hat{r}_{ui} \quad \text{for } (u,i) \in \mathcal{T}$$

$$MSE = \frac{1}{|\mathcal{T}|} \sum_{(u,i)\in\mathcal{T}} e_{ui}^2$$

$$(P^*, Q^*) = \underset{(P\geq 0, Q\geq 0)}{argmin} \quad MSE$$

Regularization and Gradient Decent

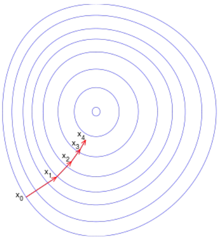
We want to introduce regularization to avoid overfitting and this is done by adding a parameter β and modify the squared error as follows:

$$e_{ui}^2 = (r_{ui} - \hat{r}_{ui})^2 + \frac{\beta}{2} \sum_{k=1}^{K} (p_{uk}^2 + q_{ki}^2)$$

Then we apply the gradient decent method to find a local minimum of MSE

$$p'_{uk} = p_{uk} + \alpha \frac{\partial}{\partial p_{uk}} e_{ui}^2 = p_{uk} + \alpha(2e_{ui}q_{ki} - \beta p_{uk})$$

$$q'_{ki} = q_{ki} + \alpha \frac{\partial}{\partial q_{ki}} e_{ui}^2 = q_{ki} + \alpha(2e_{ui}p_{uk} - \beta q_{ki})$$



Cross Validation

Since we want to optimize the *MSE* for the validation set, not for the training set, we separate the training set $\mathcal{T}$ into two part: $\mathcal{T}_1$, $\mathcal{T}_2$, then train with $\mathcal{T}_1$ and terminate when the *MSE* for $\mathcal{T}_2$ does not decrease during two epochs.

And we also select the dimensional parameter $K \in \{2, 5, 10, 15, 20\}$, and regularization parameter $\beta \in (0, 0.3)$ base on the MSE of $\mathcal{T}_2$.

Pros. and Cons.

Pros.

- do not need detailed information
- can achieve more precise prediction

Cons.

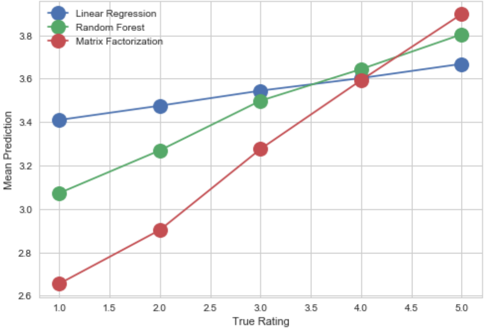
- not interpretable
- higher computational cost

Upgrade.

- regularization
- learning rate
- data processing

Results Comparison

Model	MSE
Linear Regression	1.1502
Ridge and Lasso	1.1499
Random Forest	0.9819
Matrix Factorization	0.9334



Summary

- Design and implement parametric and non-parametric statistical models on a large data set.
- Explore Python packages such as:
  - scikit-learn, pandas, seaborn, statsmodels, matplotlib, etc.
- Outlook
  - Current models are underperforming
  - Some models can still be improved
  - Can explore other models

For Further Reading I

Weisberg, S.
Applied Linear Regression, 3rd Edition.
John Wiley & Sons, Inc., 2005 3rd Edition.

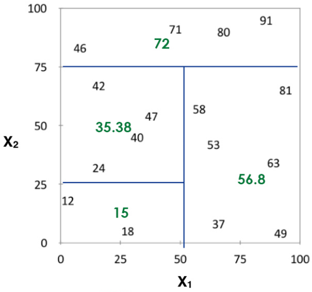
Trevor Hastie, Robert Tibshirani & Jerome Friedman
The Elements of Statistical Learning.
2010, Springer: New York

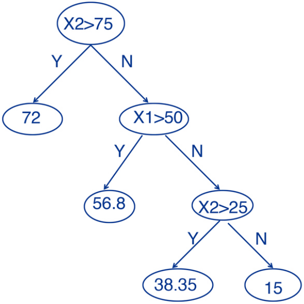
Takács, Gábor and Pilászy, István and Németh, Bottyán and Tikk, Domonkos
Matrix factorization and neighbor based algorithms for the netflix prize problem.
Proceedings of the 2008 ACM conference on Recommender systems, 267–274, 2008

Thank you!

Some Additional Slides

Single decision tree





The basic idea is to to calculate all possible splits in every node, and select the the split that has minimum MAE

The MAE (f. D) = $\frac{1}{N} | y_i - f(x_i) |$, wher $f(x_i)$ = mean value of that

① The first node

	MAE
$x_1 = 25$	15.75
$x_1 = 50$	14.81
$x_1 = 75$	15.13
$x_2 = 25$	15.5
$x_2 = 50$	15.25
$x_2 = 75$	14.5

→ choose $x_2 = 75$ as the first node

② The second node:

The second node is to split the data $x_2 < 75$ into two parts folowing the same processing as the first node.

	MAE
$x_1 = 25$	12.13
$x_1 = 50$	11.5
$x_1 = 75$	12.17
$x_2 = 25$	13.17
$x_2 = 50$	13.67

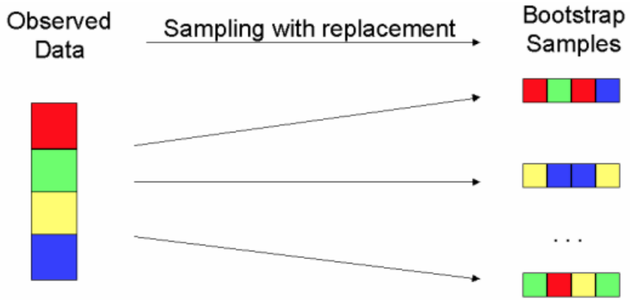
③ The third node:

$X_1 < 50$		$X_1 > 50$	
$x_1 = 25$	11	$x_1 = 75$	9.67
$x_2 = 25$	5.75	$x_2 = 25$	7.75
$x_2 = 50$	6.5	$x_2 = 50$	8.83

Thus, we get the three nodes, then we average the group value as each leaf value.

Bootstrap

- Bootstrap aggregation or bagging is an approximation that takes a single training set T_r and randomly sub-samples from it K times (with replacement) to form K training sets $T_{r1}, \cdots, T_{rK}$.
- Each of these training sets is used to train a different instance producing K regression functions $f_1(x), ..., f_k(x)$.

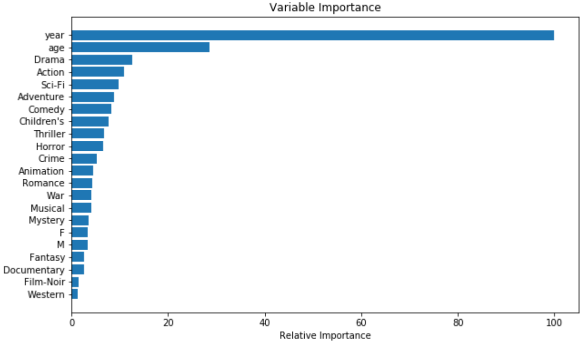


Hyperparamter tuning

- Hyperparameters is the settings of an algorithm that can be adjusted to optimize model performance.
- While parameter is a numeric or categorical value that partially determines the predictions made by a model.
- In random forest, hyperparameters include the number of decision trees in the forest and the number of features considered by each tree when splitting a node, and maximum depth of the tree etc.
- For hyperparameter tuning, we perform many iterations of the entire K-Fold CV process, each time using different model settings. We then compare all of the models, select the best one, train it on the full training set, and then evaluate on the testing set.

Variable importances

- Random forest can be used to rank the importance of variables in a regression or classification problem in a natural way.
- The importance score for the j_{th} feature is computed by averaging the difference in out-of-bag error before and after the permutation over all trees.



Residual Plot