Binary Classification of Imbalanced Data from Bosch Production Line

P-1

Liang Dong, Xi Yang & Yeo Jin Kim

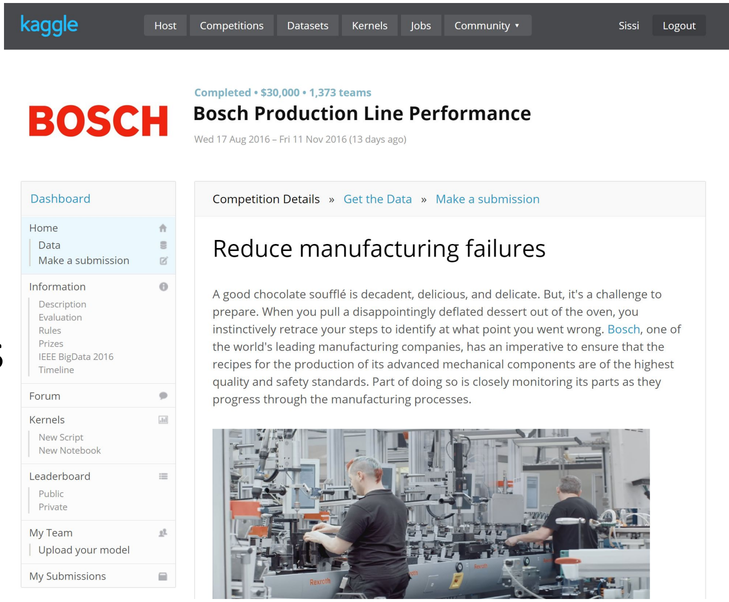
Outline

- Introduction
- Dataset
- Related Works
- Our Approach
- Experiments
- Conclusion

Introduction

- Problem Source (Kaggle Contest)
  - Bosch Production Line Performance[1] : measurements recorded in each step along assembly lines
  - Objective: predict internal failures
    - **Binary Classification** (Normal parts = 0; Failure parts = 1)
    - Improve manufacturing processes to bring quality product at lowest costs[2]

- Motivation
  - Challenge to analyze big data
  - Interested in imbalanced data
  - Contribute to detect manufacturer failures by this project



Dataset: Bosch Product Line Data

- Big Data
- Large & Complex Features
- Imbalanced
  – Ratio of Positive/Negative = 1:171 (6879: 1176868)
- High Missing Values Rates
  – Numeric (81%), Categorical (97%), Date (82%)

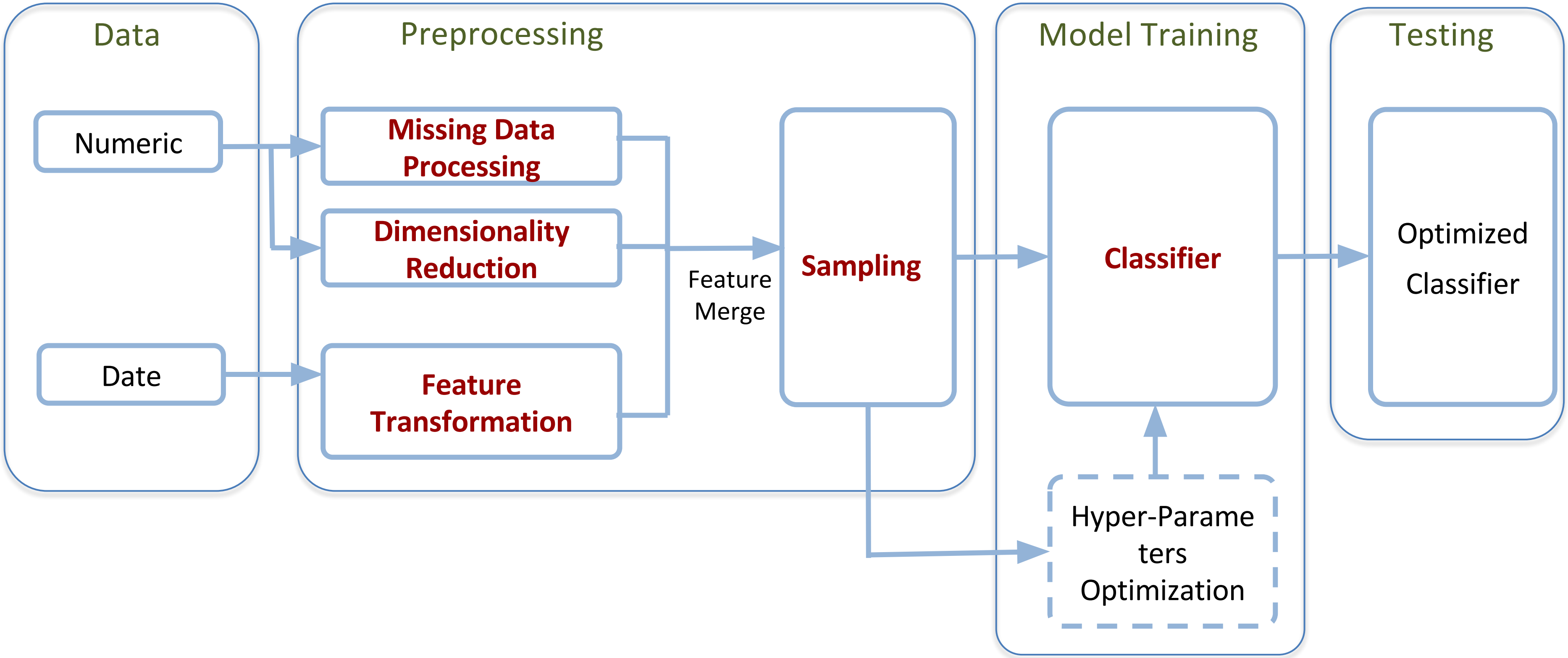
Data Type	Features	Training Instances	Training Size (GB)	Testing Instances	Testing Size (GB)
Numeric	986	118347	2.14	118348	2.14
Date	1157	118347	2.89	118348	2.89
Categorical	2140	118347	2.68	118348	2.68
Total	**4265**	118347	**7.71**	118348	**7.71**

Related Works

Aspects	Method	Pros	Cons	General Approaches
Missing Data (MD) [6]	Discarding	- Easy - Avoid to make noises	- Lose valuable instances	MD <0.1
	Imputation	- Keep instances with MD	- Risk to generate noises	0.1 < MD < 0.5
	Classifiers handling MD	- No missing data handling	- Limit to select classifiers	0.5 < MD
Big & Imbalanced → Sampling[3]	Downsampling	- Adjust the balance by removing the majority	- Lose important properties of the majority	Down: Big Data Up: Small Data * Balanced data is not always better than imbalanced
	Upsampling	- Adjust the balance by adding minority	- Overfitting: certain examples become “tied”	
Binary Classification	SVM[9]	- Find an optimal solution - Good to binary classifi. - Robust to overfit	- Slow - Less performance in high dimensionality	Gradient Boosting Tree suitable to big, complex data
	XGBoost[4,5]	- Parallel learning by Equal & quantile binning	- Need Pre-sorting, not easy to optimize	
	SAS Viya[7]	- Easy to implement - Parallel learning	- Need Pre-sorting, not easy to optimize - Only Equal binning - Commercial	
	Random Forest [8]	- Imbalanced, missing data handling	- Easy to overfitting	

Our Approach: Framework

- Feature Selection based on Missing Rate (FSMR) : Reduce the missing rate and features dimensions
- Downsampling: Reduce learning cost
- Gradient Boosting Tree: Fast learn on big data, Handle missing values



Our Approach: Preprocessing

- Dimensionality Reduction

Data Type	Method	Original Dimension	Reduced Dimension	Criteria
Numeric	FSMR (Feature Selection based on Missing Rate)	986	<0.1 : 104 <0.5 : 156	Missing value rate by feature
	EIPCA (Extreme Imputation + PCA)		126 198	Select from the range of the elbow
Date	Transformation	1157	6	Timestamp difference

- Sampling

Method	Implementation	Ratio (pos : neg)
Up-sampling	Copy positive instances	{1, 10, 20, 50, 171} :171
Down-sampling	Randomly sample from negative instances	1: {171, 50, 20, 10, 1}

Our Approach: Model Training

- Classifiers Comparison
  - Gradient Boosting Tree: Xgboost, SAS Viya
  - Random Forest
  - SVM

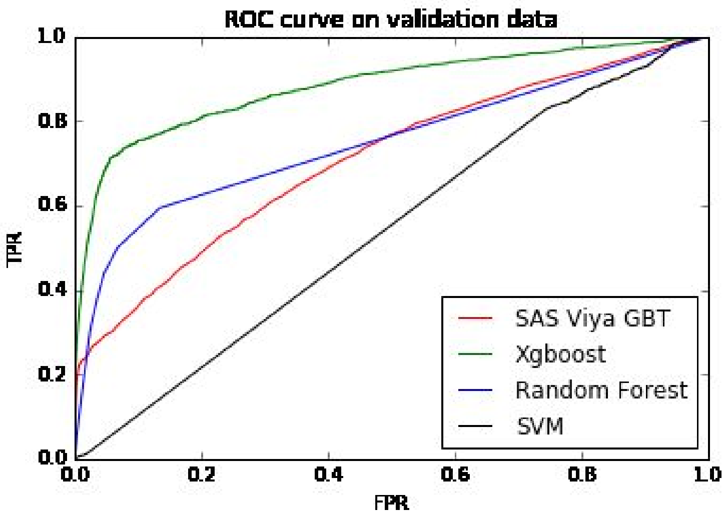
- Experiment Environment
  - Only use numerical & date datasets from Training Dataset
  - Randomly divide training dataset into training/validation/testing sets

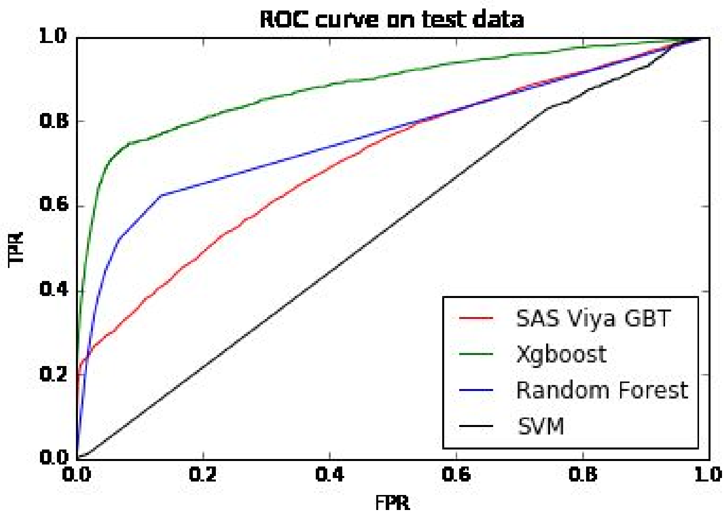
Data	Training	Validation	Testing
Percentage of size	50%	25%	25%
Num of Positives	3440	1720	1719
Num of Negative	588434	294217	294217

Experiments: Classifiers

Instances Sampling	Features Reduction	Classifiers	Parameter Selection
None	None	Xgboost, SAS Viya, Random Forest, SVM	Grid Search

- Analysis
  1) Good generalization performance (Similar results in validation and test data)
  2) Xgboost got best ROC comparing to other methods
  3) SVM could not deal with imbalanced data well (ROC near to diagonal)

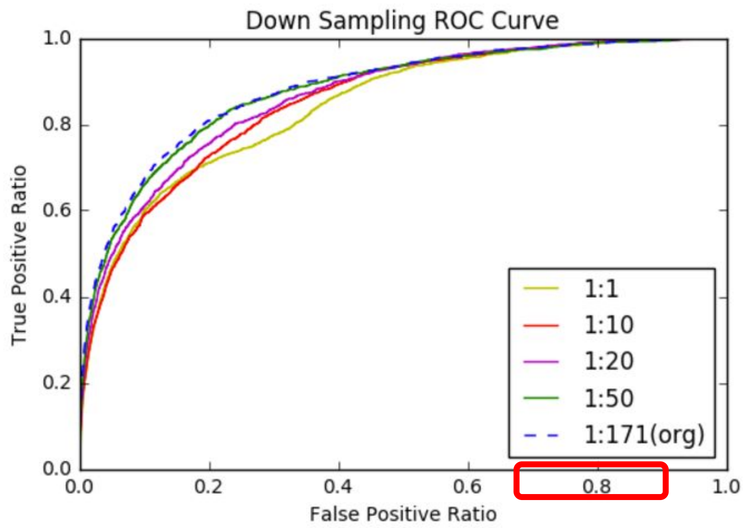


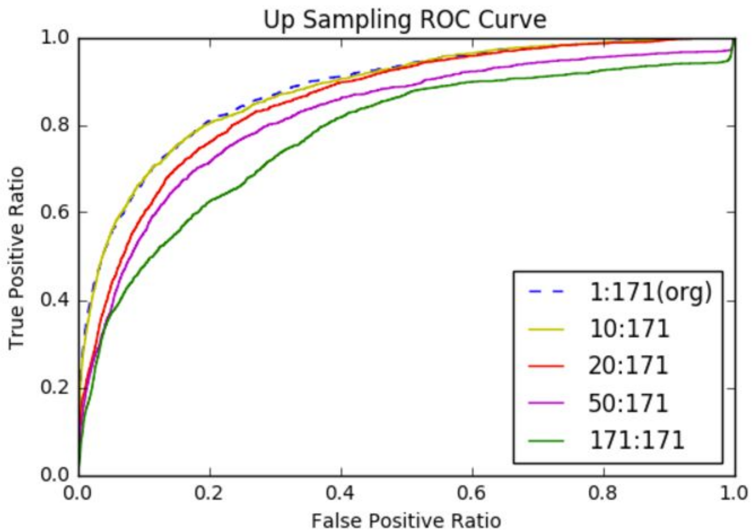


Experiments: Sampling

Instances Sampling	Features Reduction	Classifier
Downsampling	Feature Selection based on Missing rate < 0.1	Xgboost
Upsampling		

- Analysis
  1) Downsampling: could substitute original dataset (achieves comparative results)
  2) Upsampling: not restore result of original dataset well when the data size is large

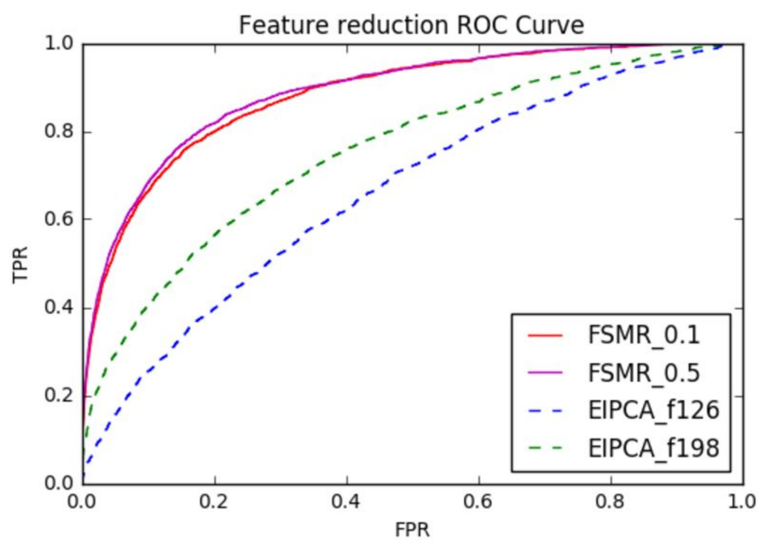


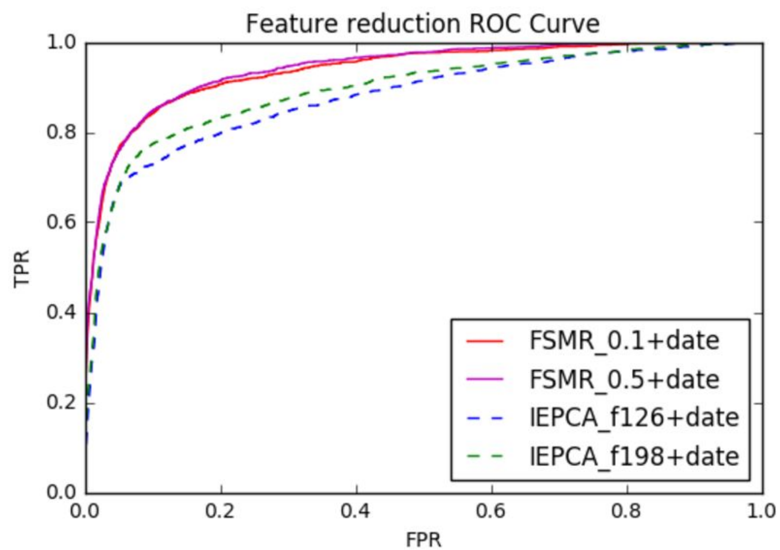


Experiments: PCA vs. Feature Selection

Features Reduction	Instances Sampling	Classifier
FSMR (Feature Selection based on Missing rate) MR < 0.1 (104 features), MR < 0.5 (156 features)	Downsampling (1:50)	Xgboost
EIPCA (Extreme Imputation + PCA) PC = 126, PC = 198		

- Analysis
  1) FSMR: Effective (discard the higher missing rate features, avoid to overfit)
  2) EIPCA: Not effective (feed the high missing rate features into PCA)





Conclusions

- Contributions
  - Big, Missing data, Large & Complex features
    - FSMR shows better performance than EIPCA, when using a classifier handling missing data
    - XGBoost achieves fast speed learning and better performance than SVM, SAS Viya, Random Forest
  - Big & Imbalanced Data
    - Down-Sampling achieves comparative performance to the original data with less time & space cost for a big dataset

- Limitations
  - Need more experiments
    - Better imputation methods
    - Better up-sampling methods
  - Only use 2 types of feature sets (excludes categorical)

References

- [1] https://www.kaggle.com/c/bosch-production-line-performance
- [2] Choudhary, A.K., Harding, J.A. & Tiwari, M.K. (2009) Data mining in manufacturing: a review based on the kind of knowledge. Journal of Intelligent Manufacturing, 20:501-521.
- [3] He, H., & Garcia, E. (2009) Learning from Imbalanced Data. IEEE Transactions of Knowledge and Data Engineering, 21(9):1263-1284
- [4] Tianqi Chen, & Carlos Guestrin. (2016) XGBoost: A Scalable Tree Boosting System. KDD.
- [5] Keck T. (2016) A speed-optimized and cache-friendly implementation of stochastic gradientboosted decision trees for multivariate classification arXiv preprint arXiv: 1609.06119.
- [6] Cismondi, F, Fialho, A.S., Vieria, S.M., Reti, S.R., Sousa, J.M.C., & Finkelstein, S.N. (2013) Missing data in medical databases: Impute, delete or classify?, Artificial Intelligence in Medicine 58:63-72
- [7] http://www.sas.com/en_us/software/viya.html
- [8] Breiman, L. (2001) "Random forests." *Machine learning* 45(1): 5-32.
- [9] http://www.csie.ntu.edu.tw/~cjlin/libsvm/

Thanks