Fire Peril Loss Cost Prediction

Frank Lanfa Wang , Ruonan Ding

6/28/2016

Product Overview

Fire Peril is one type of coverages in Property Insurance.

Property insurance also Includes:

- fire insurance,
- flood insurance,
- earthquake insurance,
- home insurance,
- boiler insurance.

Outline

- **Dataset Overview**
- Feature Engineering
- Modeling:
    - Overall Approach
    - Measure Metrics
    - Model Fitting
- Final Result
- Takeaways

Dataset Overview

Competition Goal:
to predict the loss cost of total insured value of insurance policies.

Challenge:

Rare Event: 0.2%

- the total non-zero response is 1188 out of 450K records.

Many Features:

- 4 categories of features
- 300+ features
- categorical and numeric with missing values

Outline

- Dataset Overview
- **Feature Engineering**
- Modeling:
    - Overall Approach
    - Measure Metrics
    - Model Fitting
- Final Result
- Takeaways

Raw Features

Policy Characteristic

- **17 variables**
- A set of normalized variables representing policy characteristics.
- both categorical and Numeric

Crime Rate Variables

- **9 Variables**
- A set of normalized Crime Rate variables

Geodemographic Variables

- **37 variables**
- A set of normalized geodemographic variables

Weather Variables

- **236 Variables**
- A set of normalized weather station variables

Feature Engineering

Policy Characteristic

- Delete features that have more than 50% of missing value (7 features were deleted)
- Convert categorical features to dummies (var4, 7, 8, 9 need to have dummies)
- **9 origin features were kept**
- **74 features after making dummies.**

Crime Rate Variables

- Reduced to 2 dimensions using PCAs.
- **2** new synthetic variables

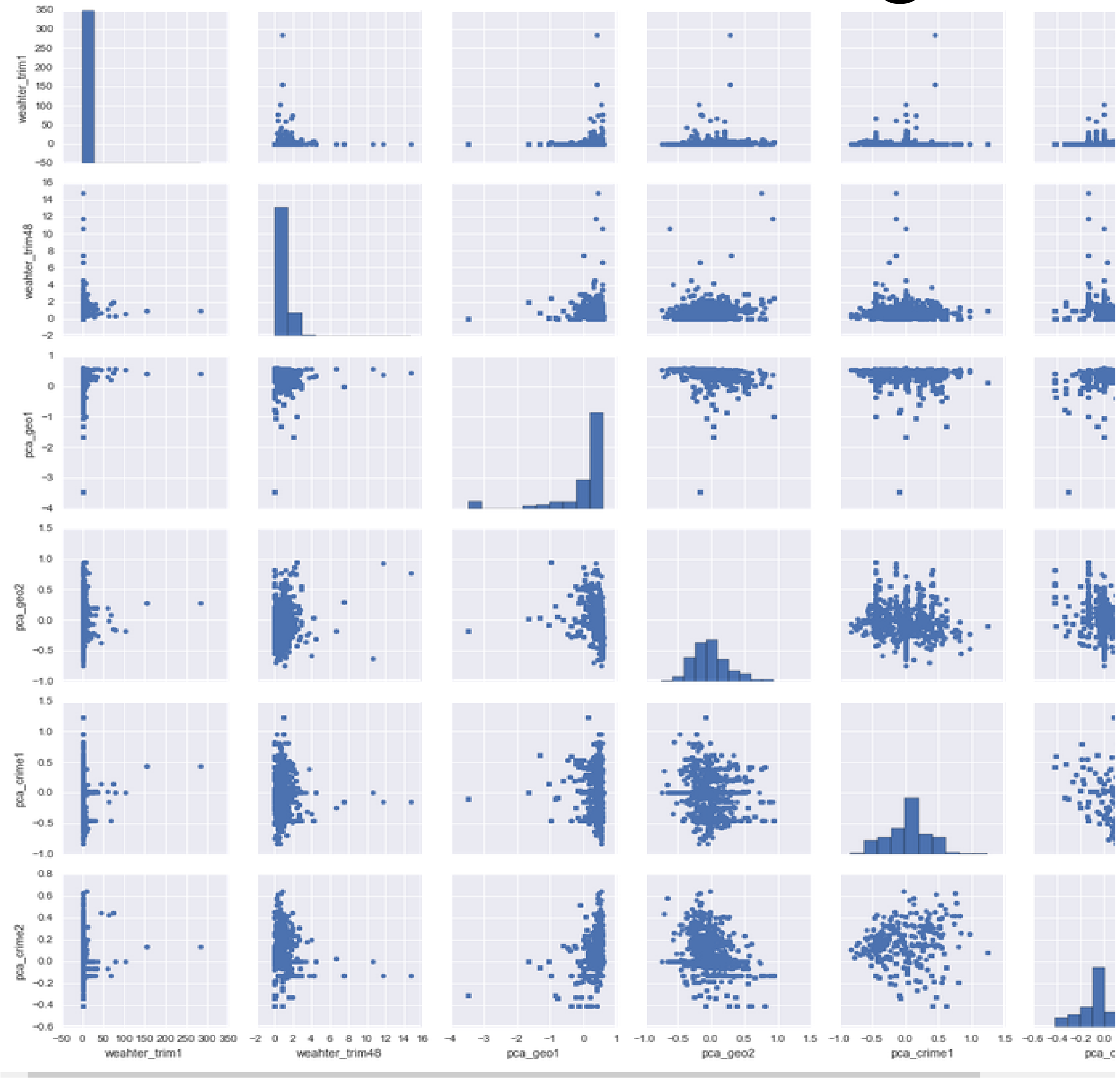
Geodemographic Variables

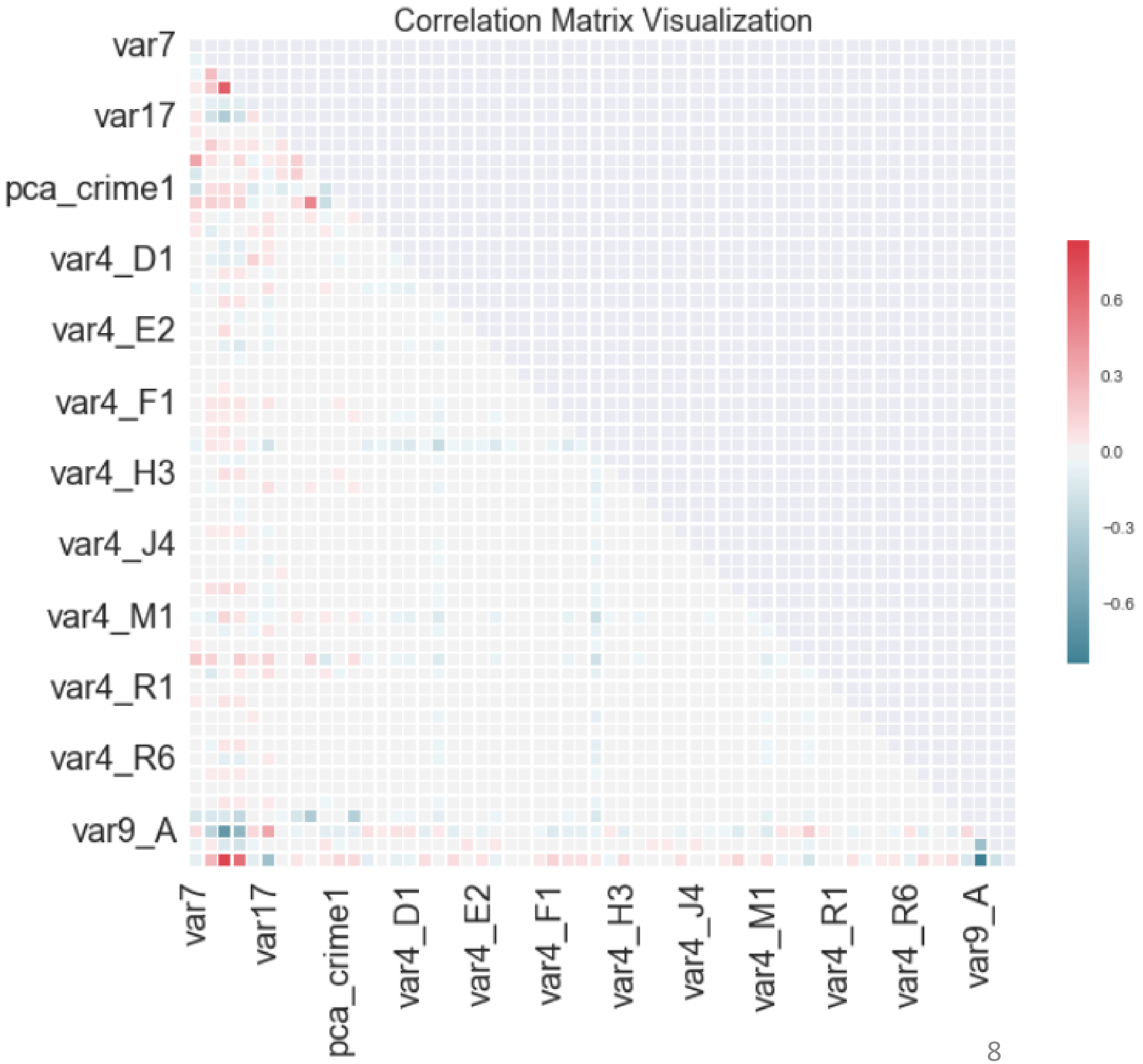
- Reduced to 2 dimensions using PCAs.
- **2** new synthetic variables

Weather Variables

- Reduced to 2 variables using Lasso L1 penalty.
- **2** features remained

After Feature Engineering

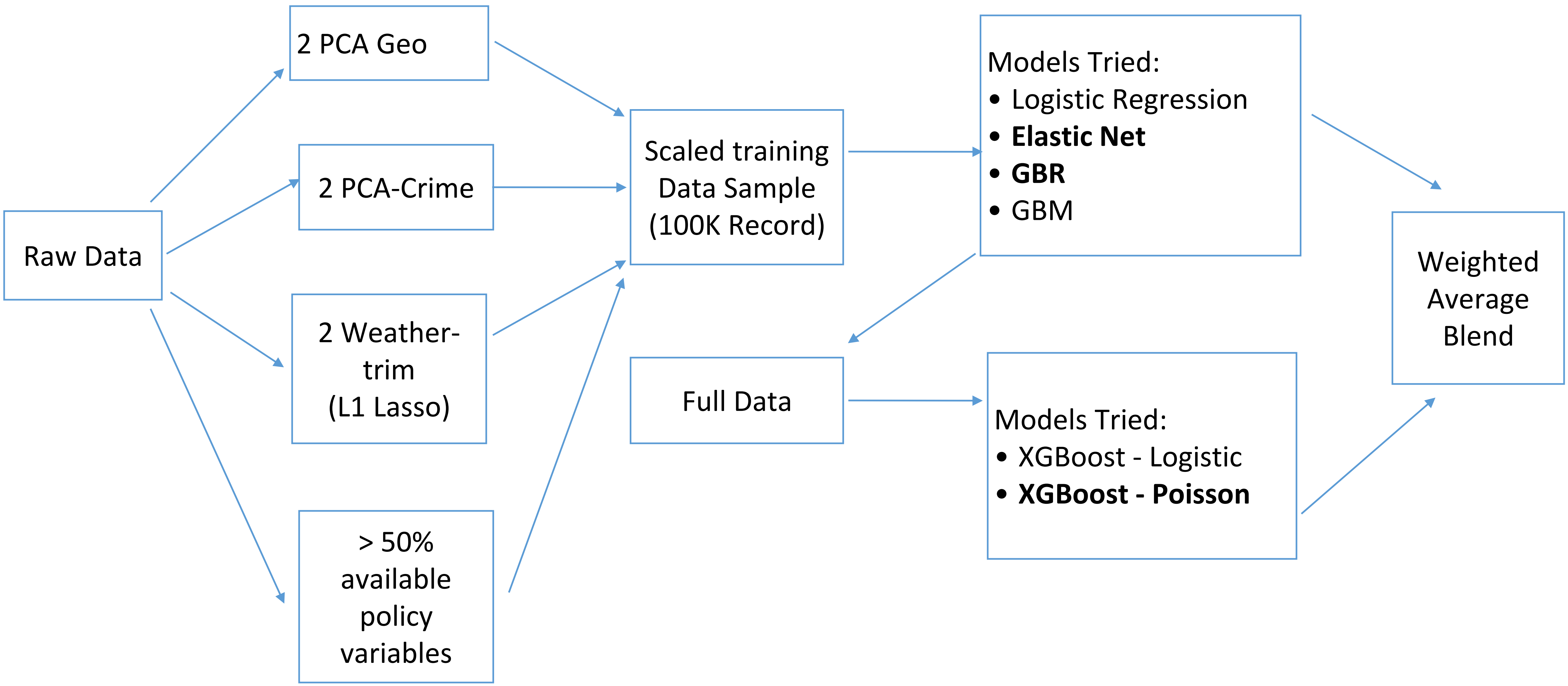




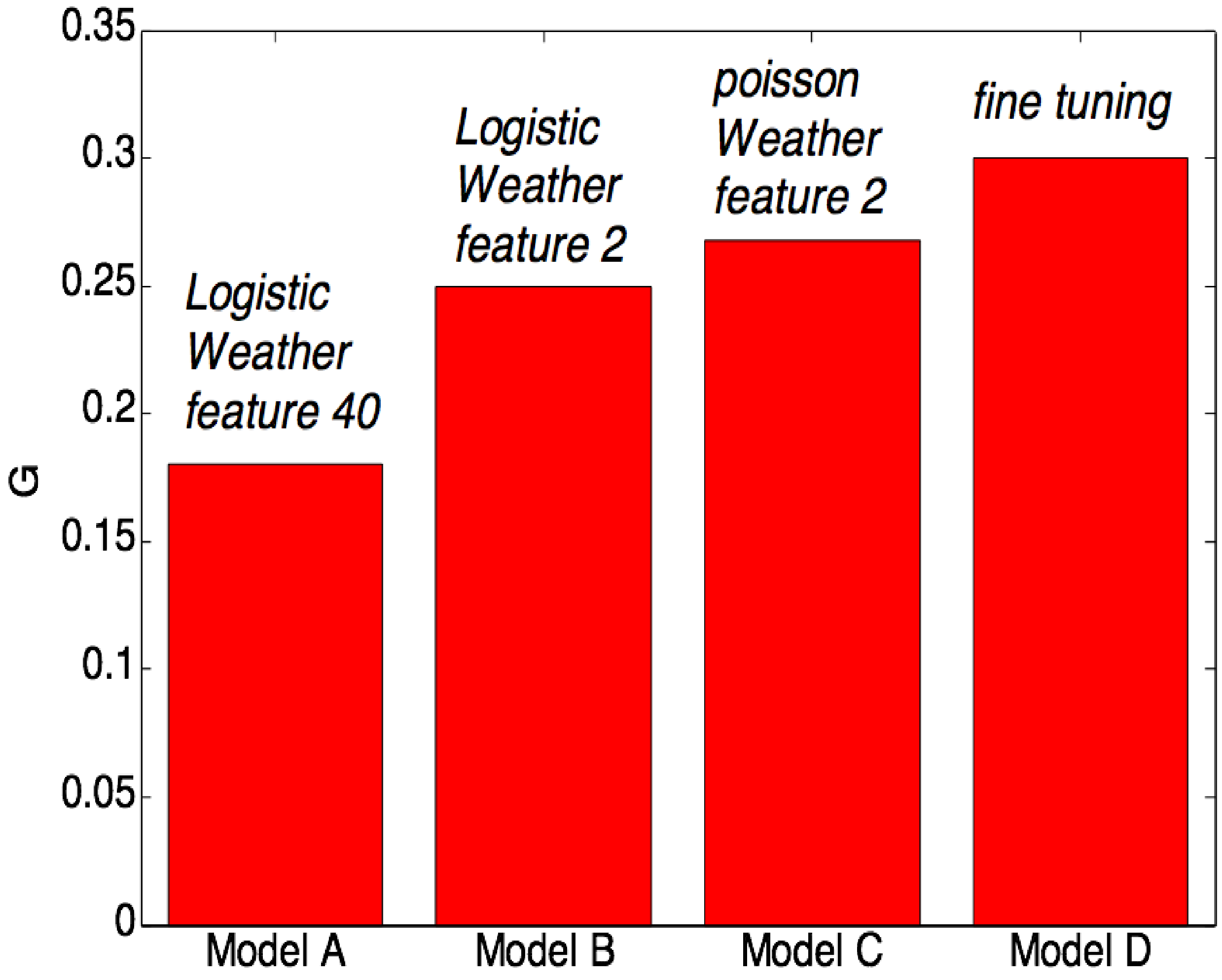
Outline

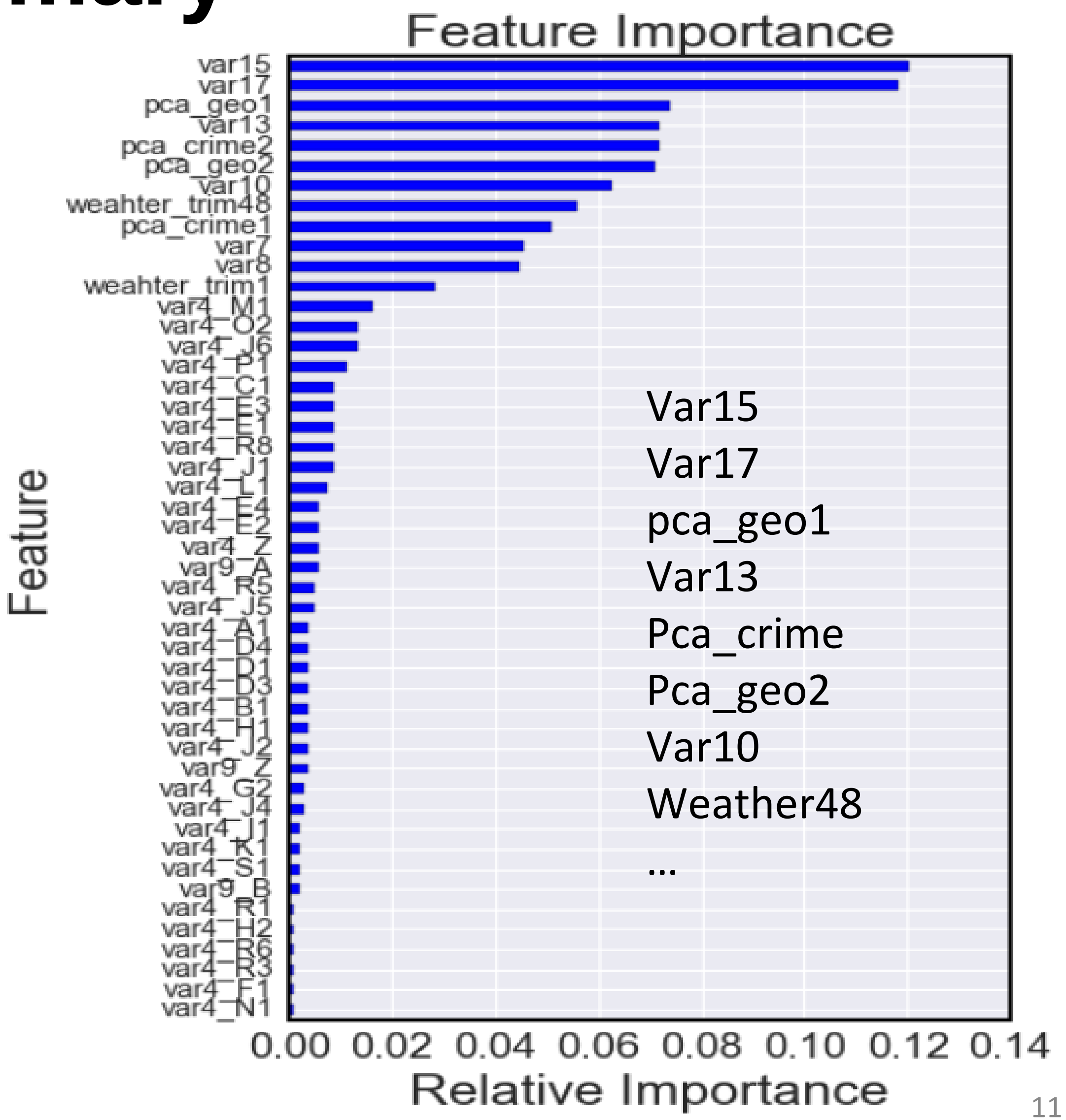
- Dataset Overview
- Feature Engineering
- **Modeling:**
    - **Overall Approach**
    - **Measure Metrics**
    - **Model Fitting**
- Final Result
- Takeaways

Overall Approach



Model 1 - XGBoost Summary

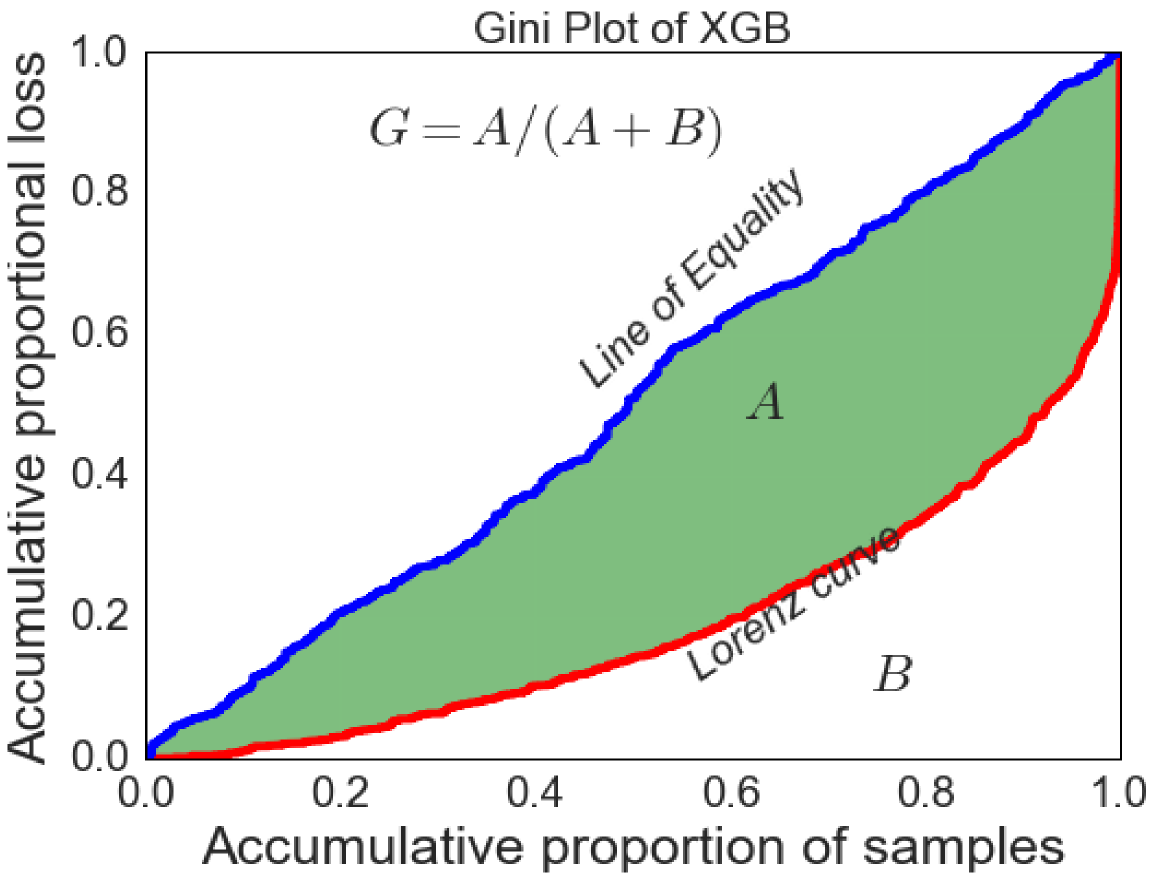


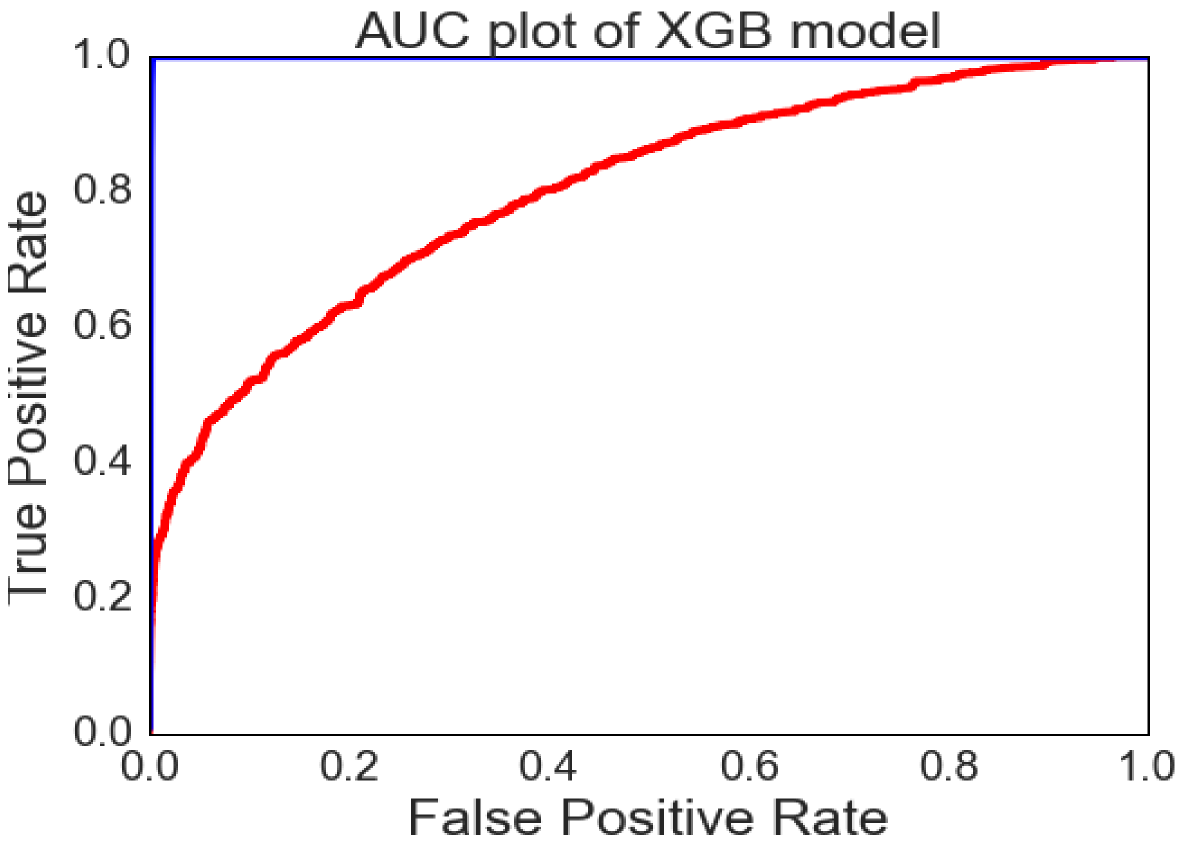


Var15
Var17
pca_geo1
Var13
Pca_crime
Pca_geo2
Var10
Weather48
…

Model 1 - XGBoost Summary

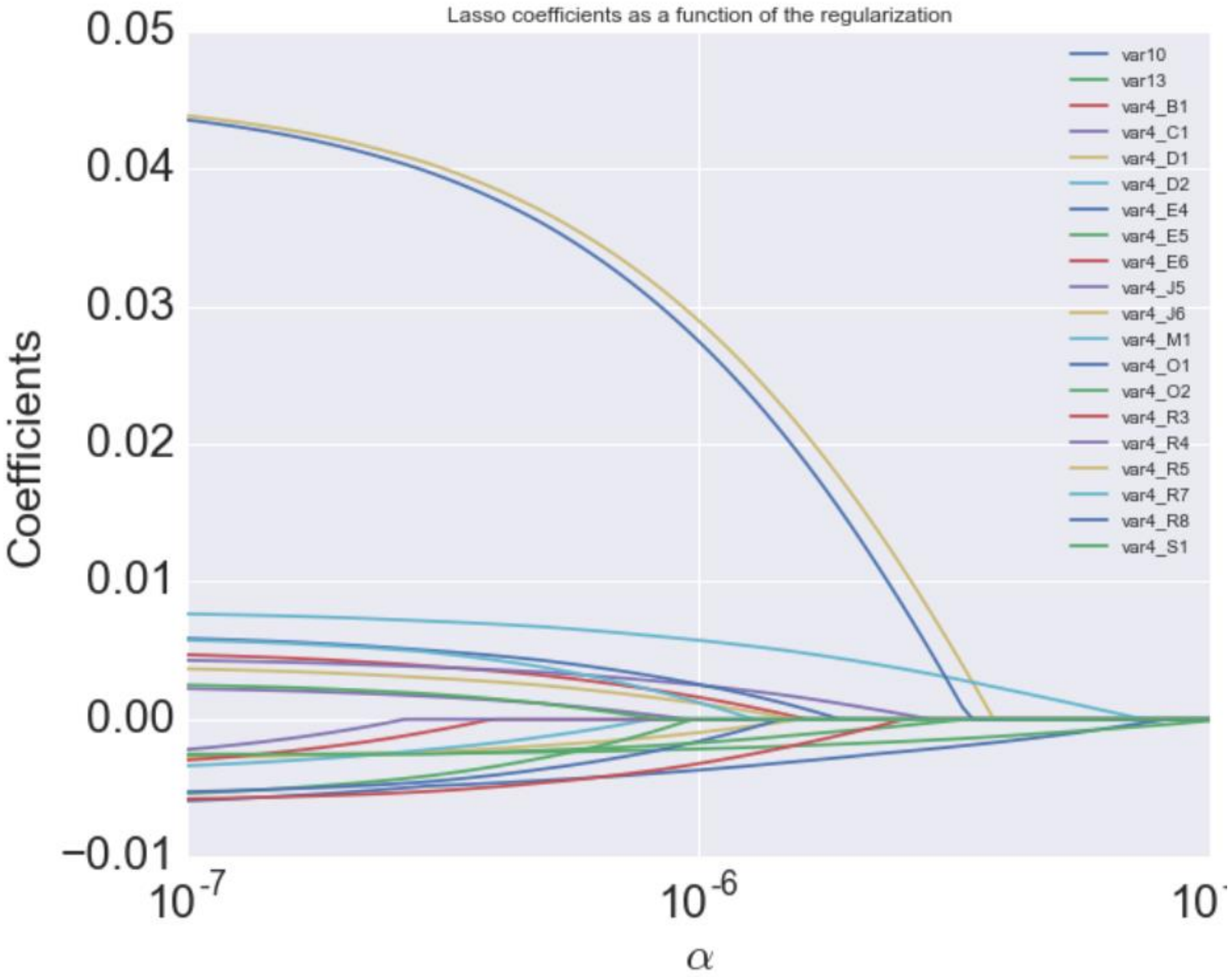
Final Model:
Objective = count:poisson;
Learning rate=0.05;   Max_depth=6; Gamma=5;   Num_round=24

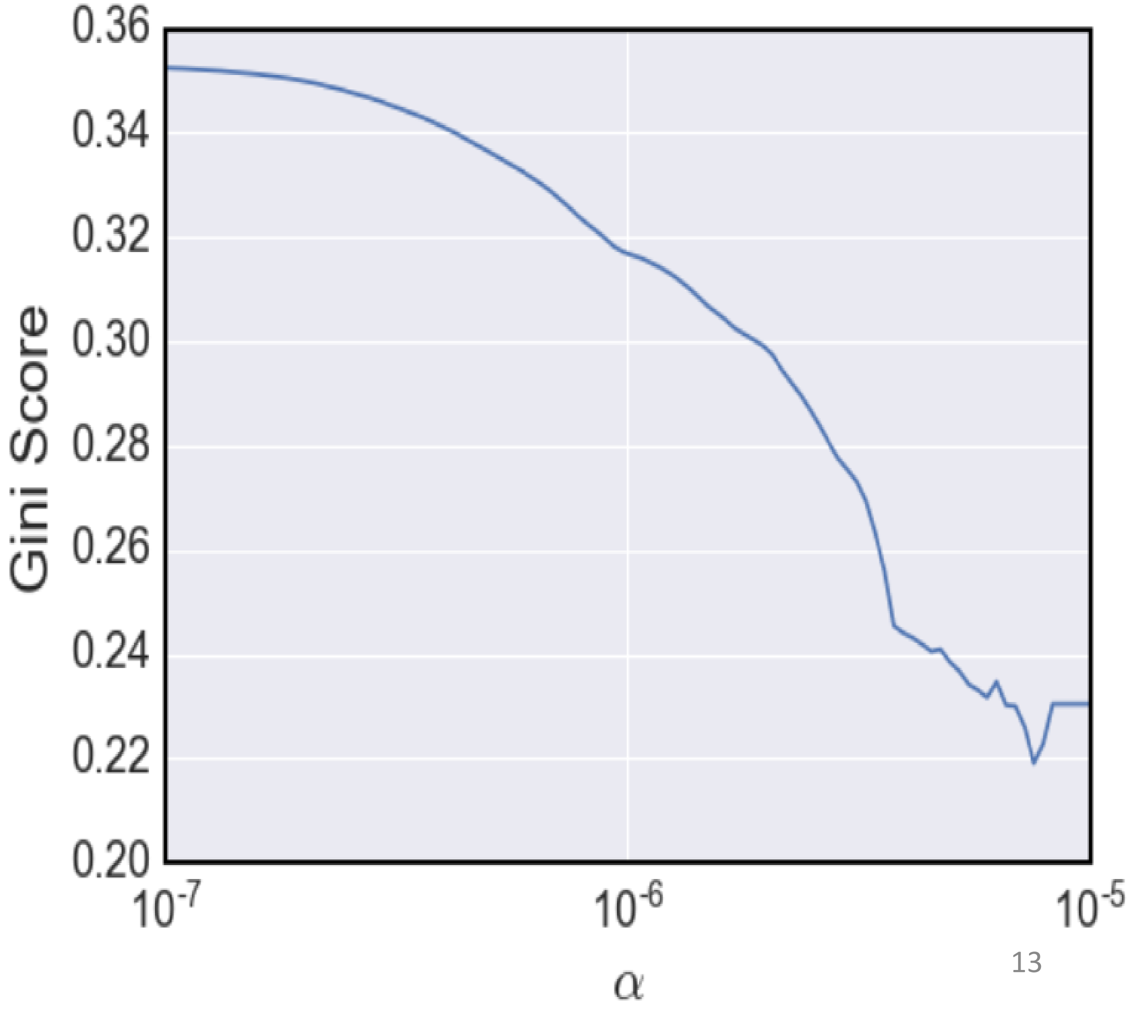




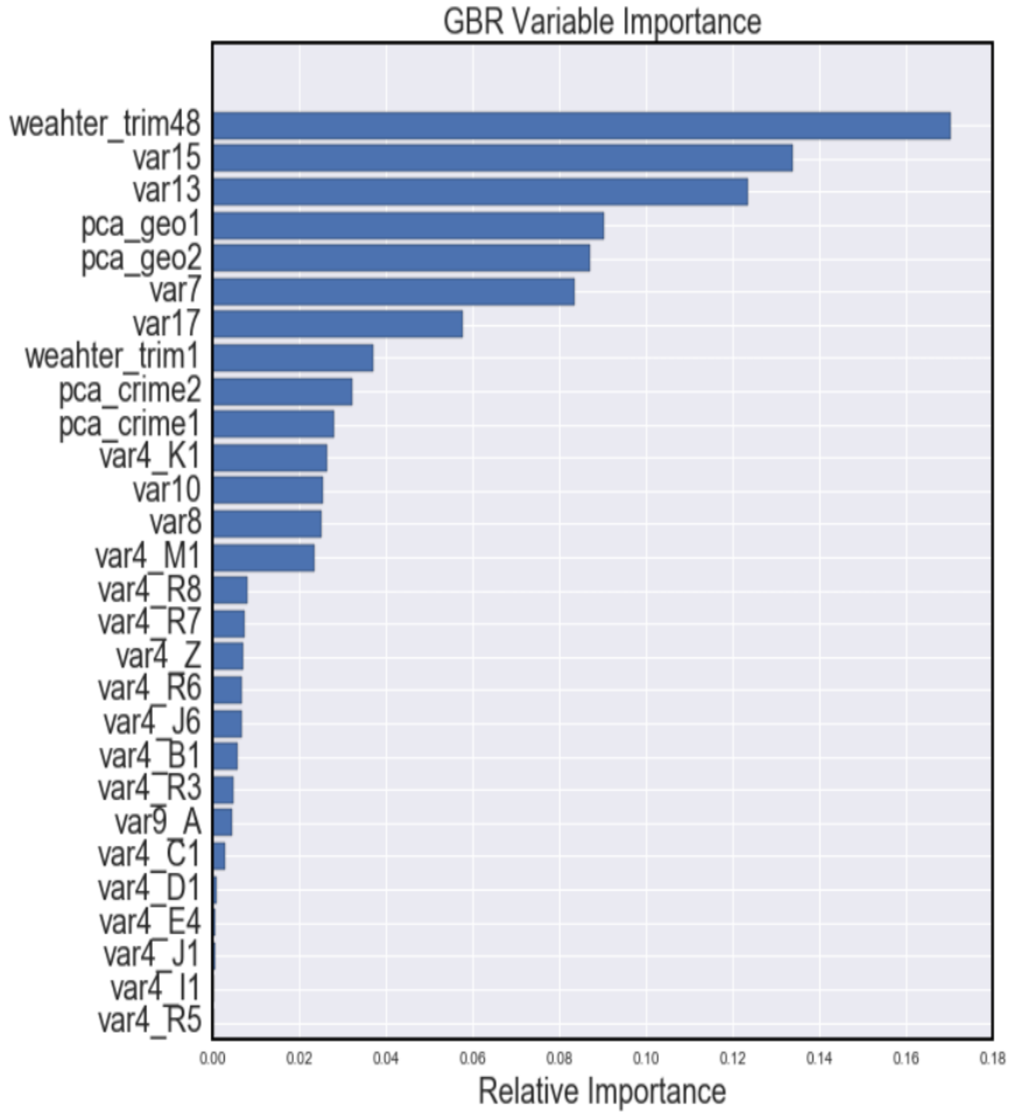
Model 2 - Linear Regression ElasticNet

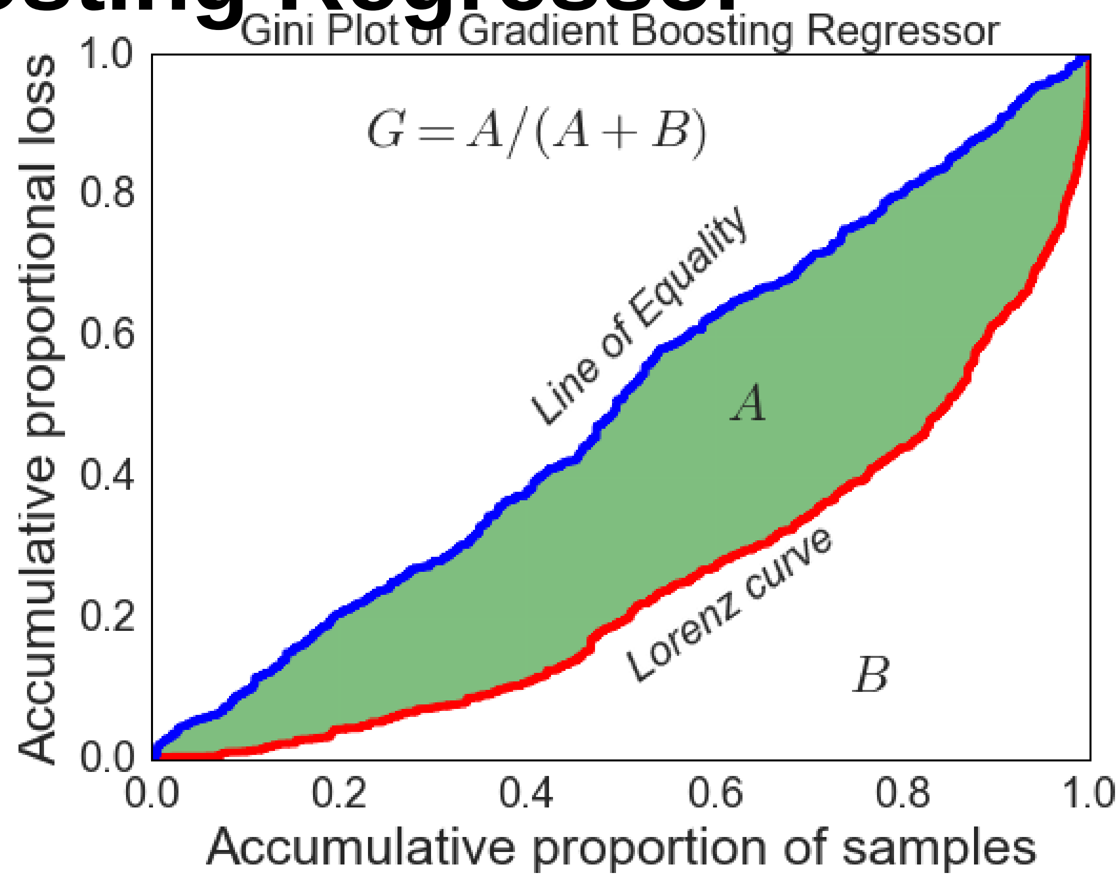
⟨ =1⊕10 ^{7} ⟩ =0.5; target: Gini=0.253; log-target: Gini=0.2





Model 3 - Gradient Boosting Regressor





Using target: Gini=0.285

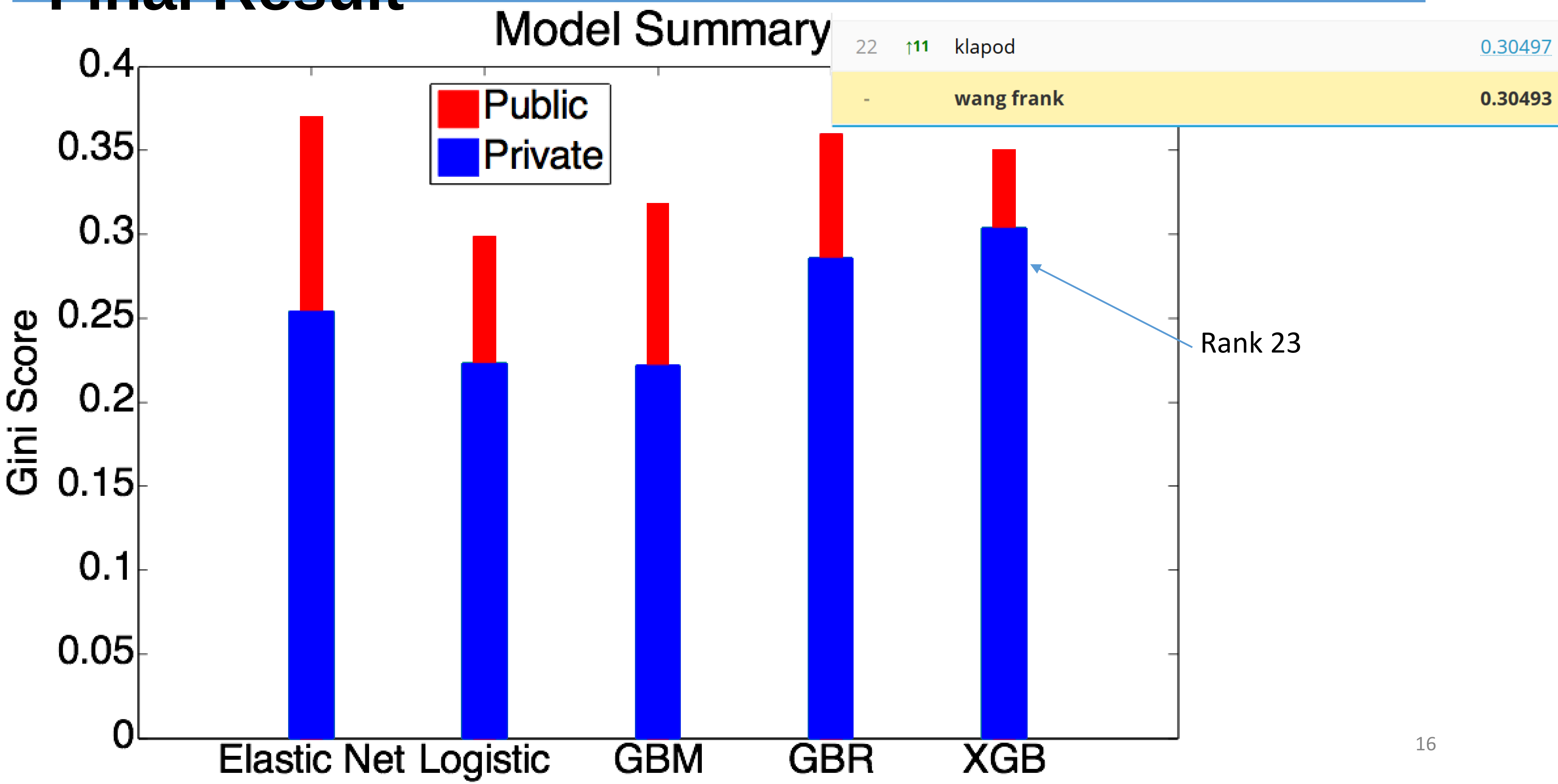
Using log-target. Gini=0.274

Model: n_estimators=100, learning_rate=0.05

Outline

- Dataset Overview
- Feature Engineering
- Modeling:
    - Overall Approach
    - Measure Metrics
    - Model Fitting
- **Final Result**
- **Takeaways**

Final Result



Takeaways

- **Feature Engineering is KEY:**
  - Extracting value from blocks of features;
  - Reduce correlation between variables - PCA
  - Reduce noise by significance - L1 penalty
  - Our scored improved on average 15% just by feature selections.
- **Sampling technique** is important with very rare event:
  - 100K including all zero losses and the 1188 response
  - cross validation
- **Poisson distribution** as the link function is suitable for RARE count event.
- **Log transformation** on the response variable is not necessary for tree-based regressors

Backup slides

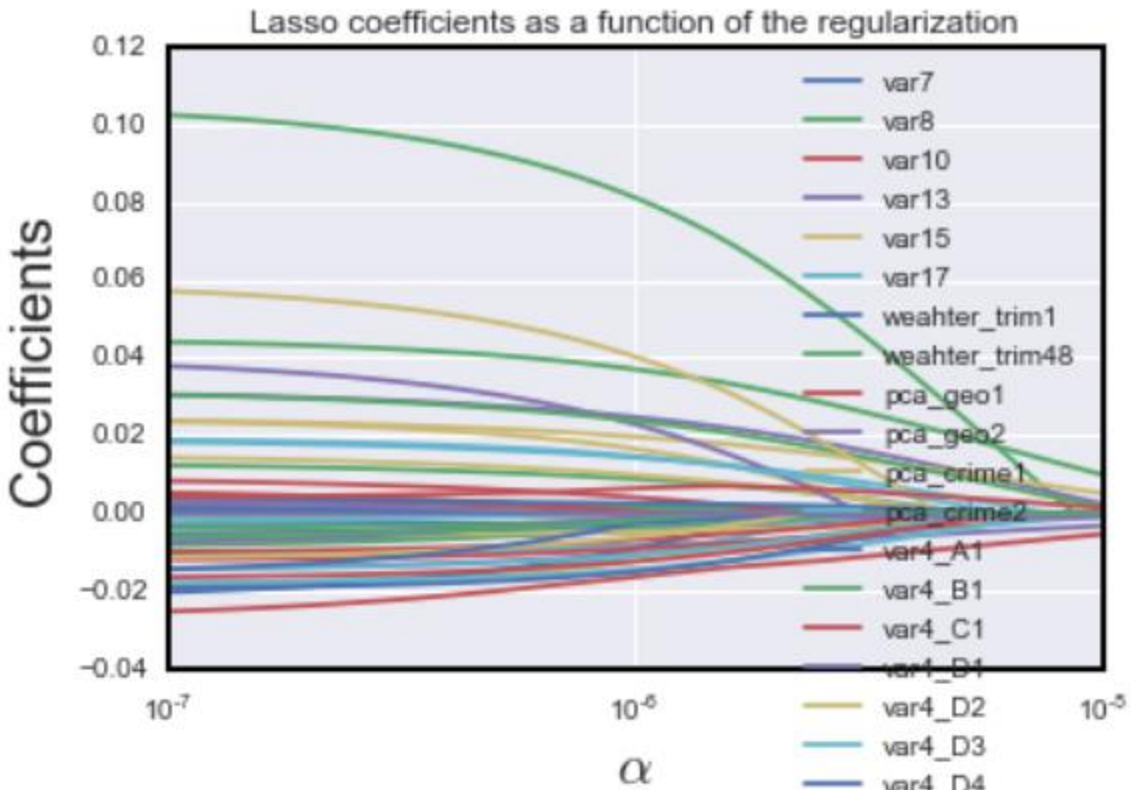
Linear regression, ElasticNet, fine steps,logloss

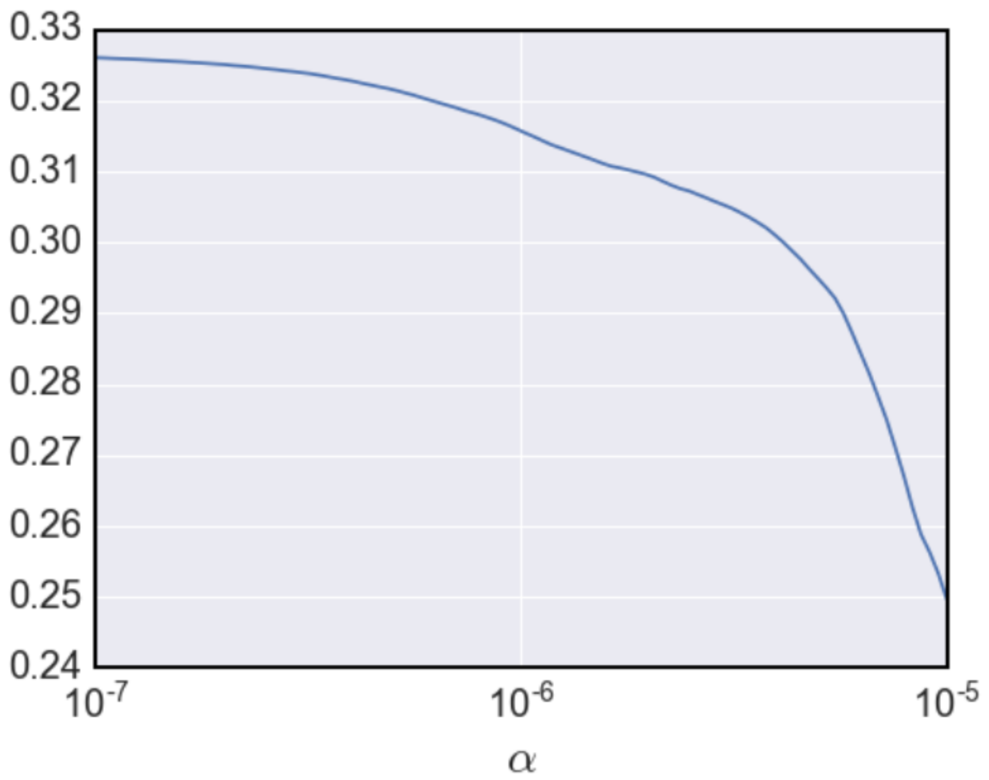
-	wang frank	0.23263	-	Fri, 24 Jun 2016 15:27:15	Post-Deadline

Post-Deadline Entry
If you would have submitted this entry during the competition, you would have been around here on the leaderboard.

$$\min_{\theta} \frac{1}{2n} \parallel X\theta - y \parallel_2^2 + \alpha\rho|\theta|_1 + \frac{\alpha(1-\rho)}{2} \parallel \theta\|_2^2$$

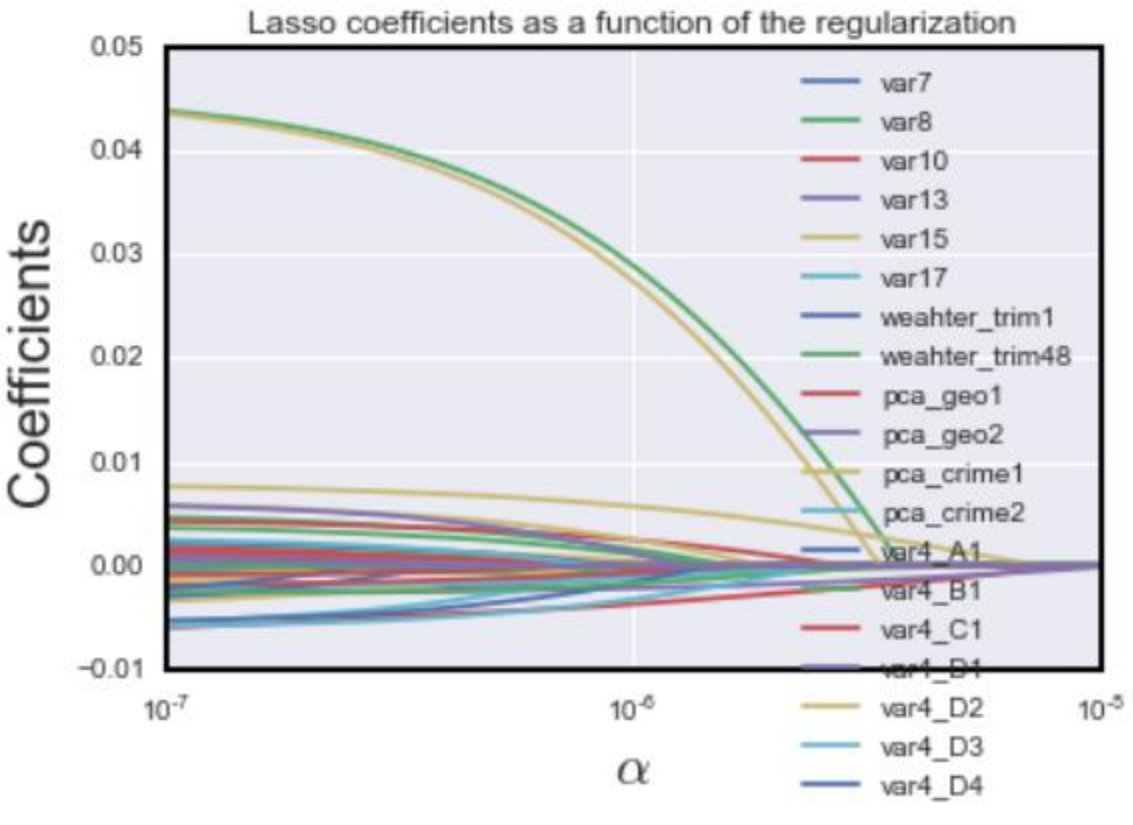
```
elastic = linear_model.ElasticNet(l1_ratio =0.5,normalize=T
elastic.set_params(alpha = 1e-7)
elastic.fit(X_train,y_train)
```

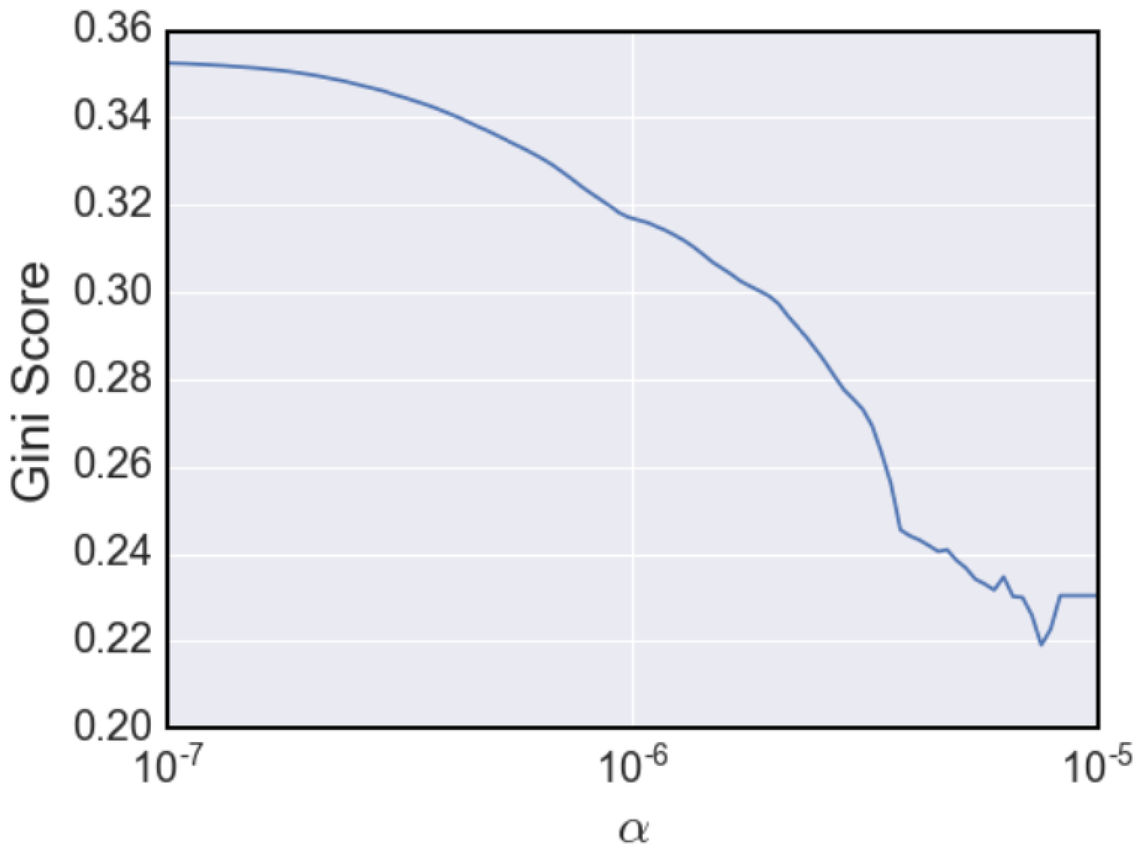




Linear regression, ElasticNet, target

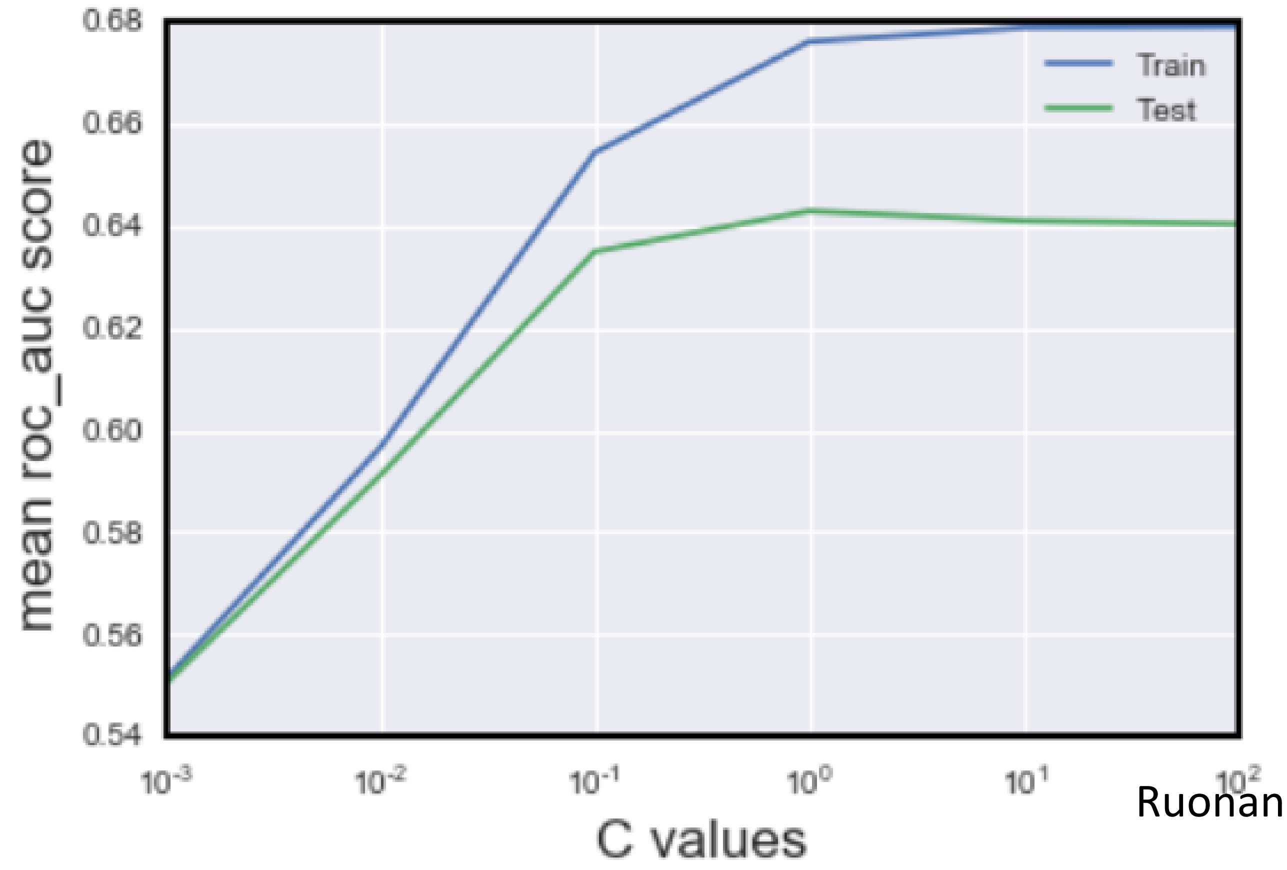
```
elastic = linear_model.ElasticNet(l1_ratio =0.5,normalize=True)
elastic.set_params(alpha = 1e-7)
```





-	**wang frank**	**0.25396**	-	**Fri, 24 Jun 2016 15:42:00**	**Post-Deadline**

Logistic regression, Gini=0.223



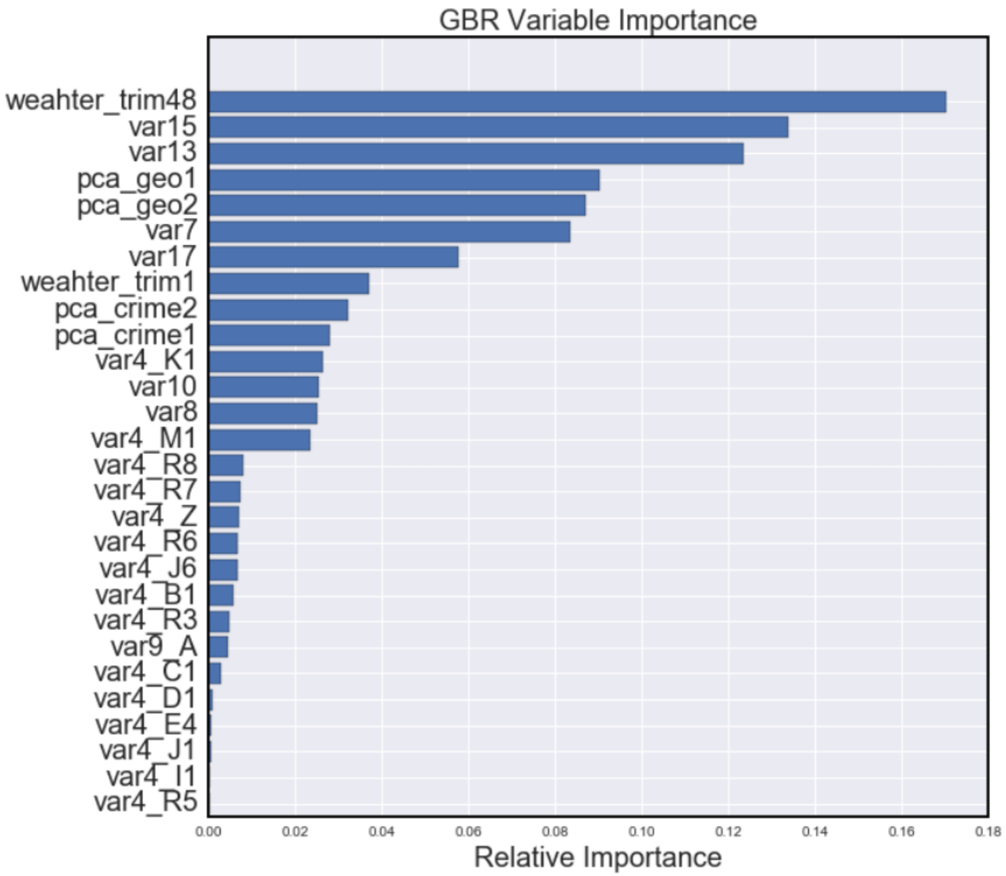

```
Logit_best = Pipeline([('scale', MinMaxScaler()),
('classifier', LogisticRegression())])
Logit_best.set_params(classifier__C=0.5)
```

Ruonan Logistic Regression got 0.21 on the leaderboard too.

-	wang frank	0.22303	-	Fri, 24 Jun 2016 20:27:20	Post-Deadline

GradientBoostingRegressor

- (n_estimators=100, learning_rate=0.05)
- Frank use the target.
- Ruonan use log-target.



-	RuonanDing	0.27362	-	Thu, 23 Jun 2016 15:05:08	Post-Deadline
-	wang frank	0.28522	-	Fri, 24 Jun 2016 17:34:54	Post-Deadline

GradientBoostingClassifier

- gbc_best = GradientBoostingClassifier(n_estimators=120,
-                                     learning_rate=0.05, random_state= 2015)

- Convert target into binary and use the predict_prob as outcome.

-	RuonanDing	0.21125	-	Thu, 23 Jun 2016 15:11:04	Post-Deadline
-	wang frank	0.22498	-	Fri, 24 Jun 2016 19:40:40	Post-Deadline