Group 24 – House Prices Prediction

Botao Li, Qili Zeng, Wenbin Teng, Ziran Min

1. Project Task

The goal of this project is to build Machine Learning models to predict the sale prices of houses in Ames, Iowa given the data sets (year 2006-2010) from Kaggle.

The difficulties of this project include cleaning 80 original variables, building effective algorithms by ourselves, finding best parameters for models, and ensembling models to reach final high score standing in Kaggle Leaderboard.

2. Dataset, Metric & Feature Engineering

Training set 1460 rows & Testing set 1459 rows.

Find potential important features,
Delete highly correlated X's; Delete outliers
Fill missing value: add 0, "None", or mode case by case
Continuous: Log transformation
Ordinal: Label encoding; Categorical: One hot encoding (dummies)

Training set 1456 rows * (202 features + 1) columns

Training	Validation	Testing	Total
1092	364	1459	2915

Metric: 4-fold cross validation - root mean square error (RMSE) between log(predicted sale price) and log(observed sale price)

3. Approach

1. Linear Regression:

Minimize cost function:

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$$

- Method 1: Gradient Descent
- Method 2: Normal Equation

$$\theta = (X^T X)^{-1} X^T y$$

- Method 3: scikit-learn package

2. Ridge Regression:

Minimize cost function:

$$J(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^{n} {\theta_j}^2\right]$$

- Method 1: Gradient Descent
- Method 2: Normal Equation:

$$\theta = (X^T X + \lambda I')^{-1} X^T y, \lambda > 0, I' = diag(0,1,1,\dots,1)$$

- Method 3: scikit-learn package

3. Lasso Regression:

Minimize cost function:

$$J(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^{n} |\theta_j|\right]$$

- Method 1: Gradient Descent
- Method 2: scikit-learn package

4. Elastic Net Regression:

Minimize cost function:

$$J(\theta) = \frac{1}{2m}\left[\begin{matrix}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 \\ +\lambda_1 \sum_{j=1}^{n} {\theta_j}^2 + \lambda_2 \sum_{j=1}^{n} |\theta_j|\end{matrix}\right]$$

- Method 1: Gradient Descent
- Method 2: scikit-learn package

Figure 1: Difference between Ridge, Lasso and Elastic Net Regression

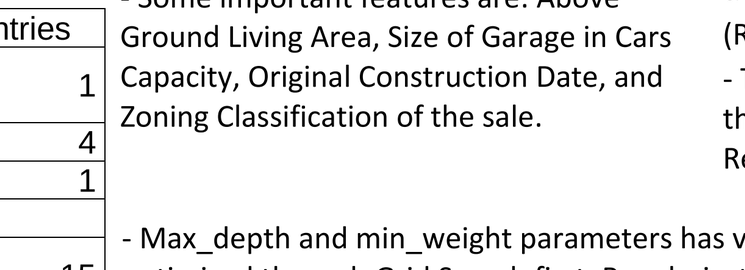

Figure2: Comparison between SVM (left) and SVR (right)

5. Support Vector Regression:

$$\min \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{\ell}(\xi_i + \xi_i^*)$$

$$s.t.\begin{cases} y_i - < w, x_i > -b \le \varepsilon + \xi_i \\ < w, x_i > +b \le \varepsilon + \xi_i^* \\ \xi_i + \xi_i^* \ge 0 \end{cases}$$

Dual Form:

$$L(a,\hat{a}) = -\frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N}(a_n - \hat{a}_n)(a_m - \hat{a}_m)k(x_n, x_m) - \varepsilon\sum_{n=1}^{N}(a_n + \hat{a}_n) + \sum_{n=1}^{N}(a_n - \hat{a}_n)t_n$$

$$s.t.\ \sum_{n=1}^{N}(\alpha_n - \hat{\alpha}_n) = 0,$$

$$0 \le \alpha_n, \hat{\alpha}_n \le C$$

Prediction:

$$y(\mathbf{x}) = \sum_{n=1}^{N}(a_n - \hat{a}_n)k(\mathbf{x}, \mathbf{x}_n) + b$$

6. XGBoost Regression

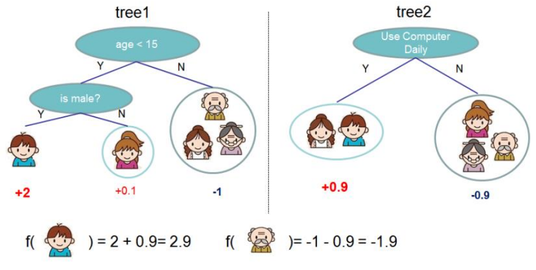

Figure 3: Tree Ensemble Model.

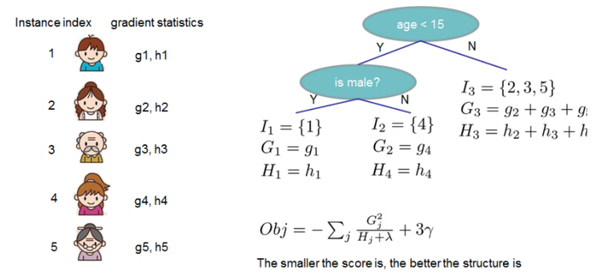

Figure 4: Structure Score Calculation

$$L(\phi) = \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k) \quad \Omega(f) = \gamma T + \frac{1}{2}\lambda \|w\|^2$$

$$\hat{y}_i^{(t)} = \sum_{k=1}^{t} f_k(x_i) = \hat{y}_i^{(t-1)} + f_t(x_i)$$

$$L^{(t)} \approx \sum_{i=1}^{n}\left[l(y_i, \hat{y}^{(t-1)}) + g_i f_t(\mathbf{x}_i) + \frac{1}{2}h_i f_t^2(\mathbf{x}_i)\right] + \Omega(f_t)$$

$$L^{(t)} = \sum_{i=1}^{n}\left[g_i f_t(\mathbf{x}_i) + \frac{1}{2}h_i f_t^2(\mathbf{x}_i)\right] + \gamma T + \frac{1}{2}\lambda\sum_{j=1}^{T} w_j^2$$

$$L^{(t)}(q) = -\frac{1}{2}\sum_{j=1}^{T}\frac{\left(\sum_{i\in I_j} g_i\right)^2}{\sum_{i\in I_j} h_i + \lambda} + \gamma T$$

7. Random Forest Regression

scikit-learn package, try to find best # of tree depth and best # of trees in the forest

8. Stacking:

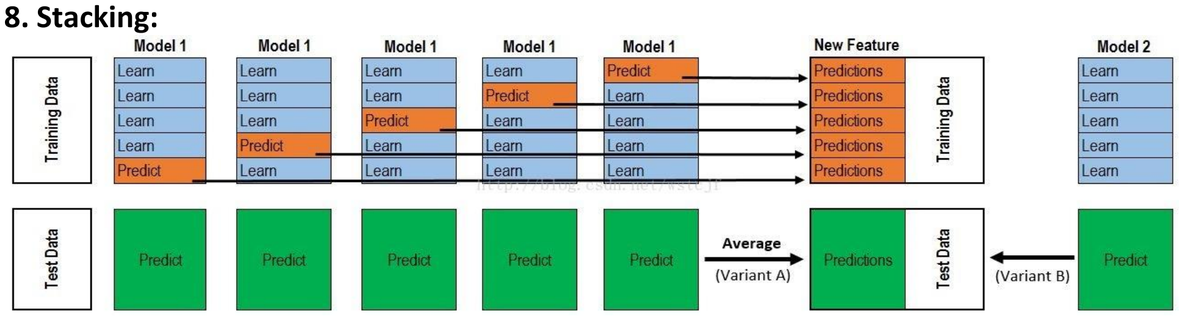

Figure 5: Stacking example flow chart

4. Results

Model	CV RMSE
XGBoost Reg (xgboost)	0.118954
Ridge Reg (sklearn)	0.121036
Linear Reg (sklearn)	0.126783
Support Vector Reg (our)	0.128861
Ridge Reg (our normal eqtn.)	0.131965
Random Forest Reg (sklearn)	0.137224
Lasso Reg (sklearn)	0.142892
Elastic Net Reg (sklearn)	0.223191
Linear Reg (our grad. descent)	0.826657
Lasso Reg (our grad. descent)	0.826657
Ridge Reg (our grad. descent)	0.826686

Kaggle Leader Score

#	Team	Score	Entries
1	Alex plus BG que Lucas	0.00000	1
2	Lucas le bg	0.03176	4
3	Zheng Pan	0.08021	1
...			
803	**Our Best Stacking**	0.11864	15
1389	**Our Best Single Model XGBoost**	0.12543	5

- Compared to other methods, Gradient Descent can't make model to fit Y with low CV RMSE.
- Normal equation for Ridge Regression can work almost as good as Ridge Regression in sklearn, even better in Stacking.
- Normal equations (Lasso & Elastic Net) are hard to compute and invertible singular matrix issue often occurs.
- Lasso and Elastic Net can't make CV RMSE better, but useful for feature selection.
- Some important features are: Above Ground Living Area, Size of Garage in Cars Capacity, Original Construction Date, and Zoning Classification of the sale.

- The more models combined in the first level of the stacking, the better the performance of ensembled model is. (Ridge + Lasso + ElaNet + SVR + XGB + Random)
- The "simpler" the model in the second level of the stacking, the better the performance of ensembled model is. (Linear Reg works best)

- Max_depth and min_weight parameters has very great influence on the final result and should be optimized through Grid Search first. Regularization parameter gamma determines the complexity of the tree and should be optimized then. After fixing these parameters, subsample and colsample_bytree needs furthur optimization. XGBoost model has some insensitive parameters which can be determined manually instead of time-consuming fine-grained Grid Search.

5. Future Works

- Compute normal equations for Lasso and Ela Net.
- Program for decision trees and random forests
- Learn better algorithms from open source code in library sklearn and XGB.
- Try other model ensembling methods.