Project 2

Kaggle Competition

House Prices: Advanced Regression Techniques

Team Members

Karthik Karunanithi (802827527)

Bharath Krishnan (893429449)

Shankar Tiwari (803012350)

Gargi Mrunal Kulkarni (893210922)

CPSC 483

Data Mining and Pattern Recognition

Fall, 2016

Prof: Kenytt Avery

Department of Computer Science

California State University, Fullerton

October 27, 2016

TABLE OF CONTENTS

1 INTRODUCTION

In this project we signed up for the Kaggle competition. The aim was to complete the competition and submit the code for rank. Then improve the code for better analysis to improve the rank. The competition we entered was "House Prices: Advanced Regression Techniques". The competition details can be found on site: https://www.kaggle.com/c/house-prices-advanced-regression-techniques. We implemented different advanced regression algorithms and compared the results for best prediction.

Following are the details for the project implementations:

Dataset: Provided by Kaggle and in known as Ames Housing Dataset

Data Mining Tool: Python scikit library.

Analysis & Prediction:

Prediction of the sale price of the houses

Algorithms: The following algorithms were implemented in the project:

Advanced Regression Techniques like LASSO, XgBoost, PCA etc.

2 ABOUT THE DATASET AND ALGORITHMS

2.1 DATASET

The dataset was provided by Kaggle. It is Ames Housing Dataset. It's an alternative, modernized and expanded version, of the often cited Boston Housing dataset. The details are available on site https://www.kaggle.com/c/house-prices-advanced-regression-techniques/data.

2.2 ALGORITHMS

The algorithms implemented to predict sale price using Ames Housing Dataset are as given below :

1. Lasso Regression: In statistics and machine learning, lasso (least absolute shrinkage and selection operator) (also Lasso or LASSO) is a regression analysis method that performs both variable selection and regularization in order to enhance the prediction accuracy and interpretability of the statistical model it produces. Lasso was originally formulated for least squares models and this simple case reveals a substantial amount about the behavior of the estimator, including its relationship to ridge regression and best subset selection and the connections between lasso coefficient estimates and so-called soft thresholding. It also reveals that (like standard linear regression) the coefficient estimates need not be unique if covariates are collinear. Though originally defined for least squares, lasso regularization is easily extended to a wide variety of statistical models including generalized linear models, generalized estimating equations, proportional hazards models, and M-estimators, in a straightforward fashion. Lasso's

ability to perform subset selection relies on the form of the constraint and has a variety of interpretations including in terms of geometry, Bayesian statistics, and convex analysis.

2. Xgboost: Xgboost is an open-source software library which provides the Gradient boosting framework for C++, Java, Python, R, and Julia. It works on Linux, Windows, and macOS. From the project description, it aims to provide a "Scalable, Portable and Distributed Gradient Boosting (GBM, GBRT, GBDT) Library". Other than running on a single machine, it also supports the distributed processing frameworks Apache Hadoop, Apache Spark, and Apache Flink. It has gained much popularity and attention recently as it was the algorithm of choice for many winning teams of a number of machine learning competitions like Kaggle.
3. Ridge Regression: Similar to LASSO, ridge regression is a regression analysis method that performs both variable selection and regularization in order to enhance the prediction accuracy and interpretability of the statistical model it produces. Linear regression models are often fitted using the least squares approach, but they may also be fitted in other ways, such as by minimizing the "lack of fit" in some other norm (as with least absolute deviations regression), or by minimizing a penalized version of the least squares loss function as in ridge regression (L2-norm penalty) .
4. Random Forest regression: Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks, that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees' habit of overfitting to their training set.
5. Principal Component Analysis(PCA): Principal component analysis (PCA) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components. The number of principal components is less than or equal to the number of original variables. This transformation is defined in such a way that the first principal component has the largest possible variance (that is, accounts for as much of the variability in the data as possible), and each succeeding component in turn has the highest variance possible under the constraint that it is orthogonal to the preceding components. The resulting vectors are an uncorrelated orthogonal basis set. PCA is sensitive to the relative scaling of the original variables. PCA is mostly used as a tool in exploratory data analysis and for making predictive models. PCA can be done by eigenvalue decomposition of a data covariance (or correlation) matrix or singular value decomposition of a data matrix, usually after mean centering (and normalizing or using Z-scores) the data matrix for each attribute.[4]The results of a PCA are usually discussed in terms of component scores, sometimes called factor scores (the transformed variable values corresponding to a particular data point), and loadings (the weight by which each standardized original variable should be multiplied to get the component score). In regression analysis, the larger the number of explanatory variables allowed, the greater is the chance of overfitting the model, producing conclusions that fail to generalise to other datasets. One approach, especially when there are strong correlations between different possible explanatory variables, is to reduce them to a few principal components and then run the regression against them, a method called principal component regression.

3 About the tool – Python SKLearn

Scikit-learn (formerly scikits.learn) is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

scikit-learn is known as Machine Learning in Python with following features:
Simple and efficient tools for data mining and data
analysis Accessible to everybody, and reusable in
various contexts Built on NumPy, SciPy, and Matplotlib
Open source, commercially usable - BSD license

3.1 System Requirements and Installation steps

The system requirement for Python SKLearn have no any minimal specification. Since data analysis is a computationally intensive task—the better your hardware, the better your experience. Also, the memory should be enough to handle big data sets.

The installation steps of SKLearn is given on the site: http://scikit-learn.org/stable/install.html Other installation required to support the scikit learn library are:
Python (>= 2.6 or >= 3.3) - Python version of project is
version 2.6 NumPy (>= 1.6.1)
SciPy (>= 0.9)
Note: The Python version required is 2.6. If version is different and libraries installed are for different version, code will fail to execute. For correct installation please check online instructions.

The steps to run the code are as follows:
Install Python 2.6 and sklearn library, sciPy, numPy, matplotlib and
pandas Download the dataset form here and save it on local drive.
Copy the code file on the local drive.
Open the file and change the dataset path (Test and Train) to the data files saved on the local machine.

```
# Reading the datasets

train_df =
pd.read_csv('/home/bharathkrishnan/bharath/Pro
2/train.csv', index_col=0)
test_df =
pd.read_csv('/home/bharathkrishnan/bharath/Pro
2/test.csv', index_col=0)
```

Open command window and change the directory to one where code file is located. Run the code to produce output CSV file as required by the competition.
The algorithms in the code have been commented out. In order to run a particular algorithm, remove the comments from the respective code and assign the respective output variable to the variable y_final.

4 PROJECT IMPLEMENTATION

In this project, we implemented different regression techniques on Ames Housing Dataset and picked the one giving best predictions.

4.1 DATA PRE-PROCESSING

Most of the real world data are generally Incomplete, noisy or inconsistent. As a result, a certain set of procedures are followed to make the data fit for the analysis. The following measures are considered in the project:

1. Categorical Variables are coded using the dummy variables. The number of dummy variables for a categorical feature is equal to (Number of categories-1). Pandas, a Python Library, allows you to code the categorical variables into the dummy variables using the function,

Dataframe.*get_dummies()*

2. Handling the missing Variables: Missing variables can be handled in different ways: Substituting it with the attribute mean, ignore the observation corresponding to the the missing value, Use the Label of the missing value as the target variable. Since the number of rows are significantly low, removing any number of observation will lead to a significant amount of information in the data. As a result, the missing values are substituted with the attribute mean using the following function.

Dataframe.*isnull*(Dataframe.*mean*())

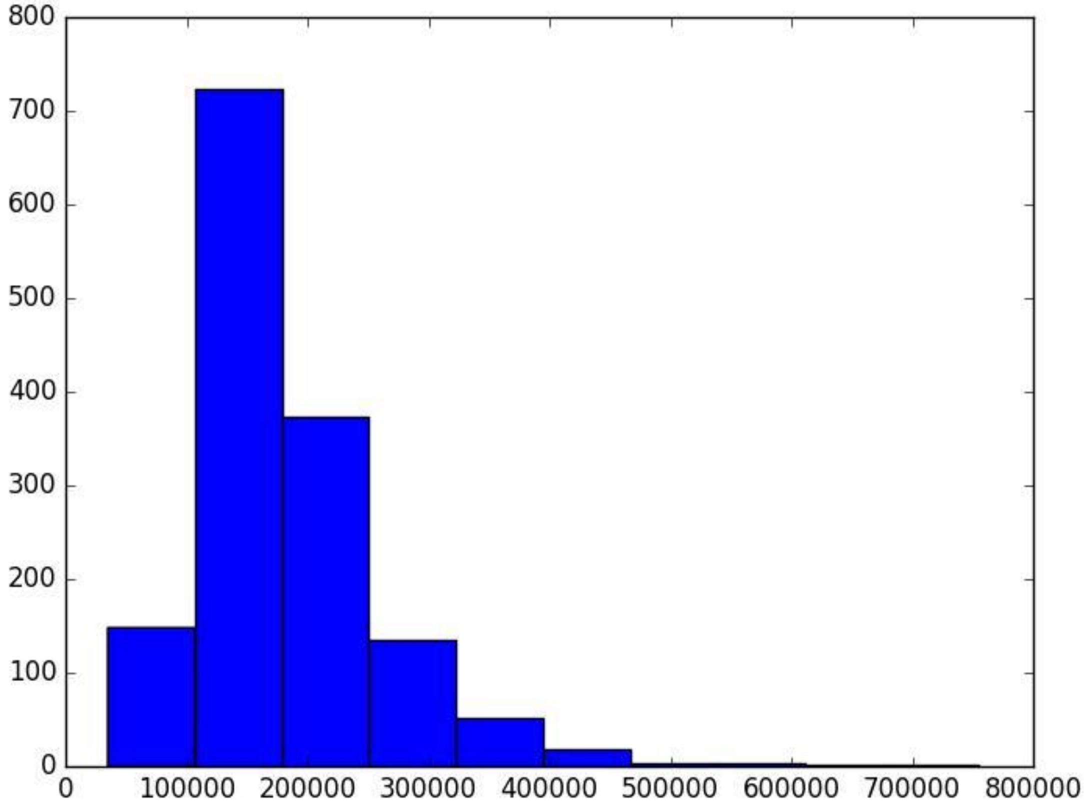
3. Handling the skewness: The data provided was skewed to a large extent. Analysis on a skewed data will lead to incorrect results or inference. The most commonly method used to reduce the skewness of the data is the *log* transformation.

4. Data Normalization: Data is normalized as the features in the data are generally of different unit. the intention is that these **normalized values** allow the comparison of corresponding normalized values for different datasets in a way that eliminates the effects of certain gross influences. Normalization is done using the following formula:

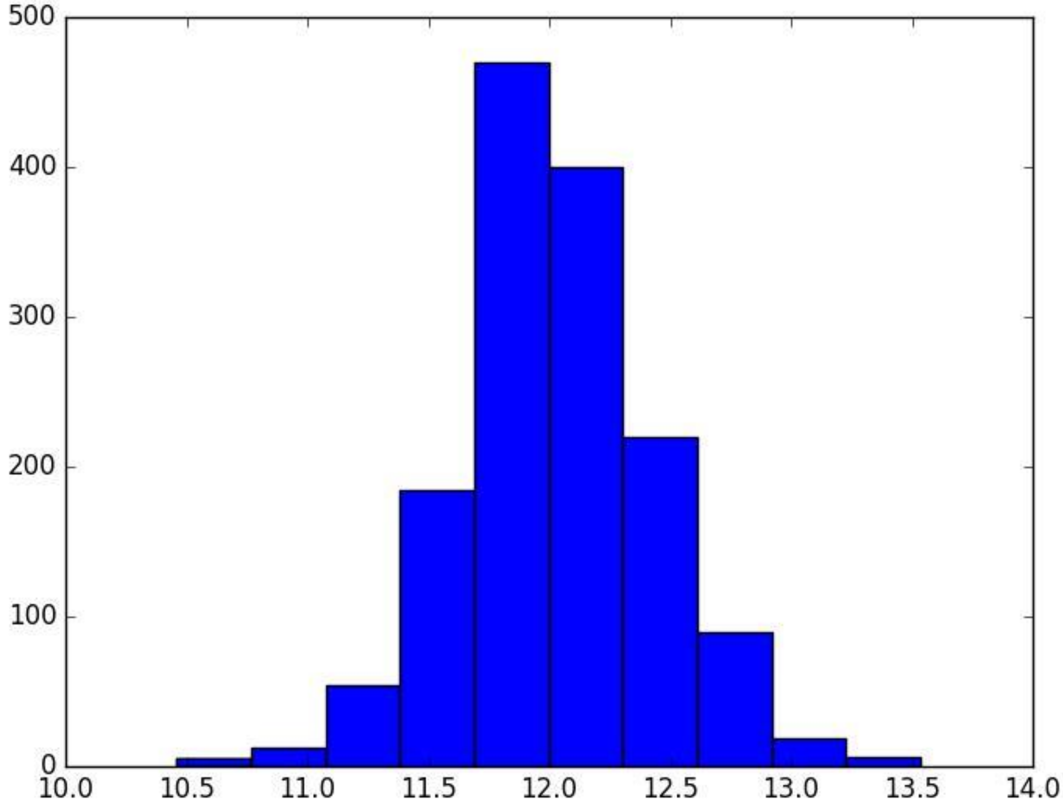
$$\frac{X - \mu}{\sigma}$$

4.2 SCREENSHOTS

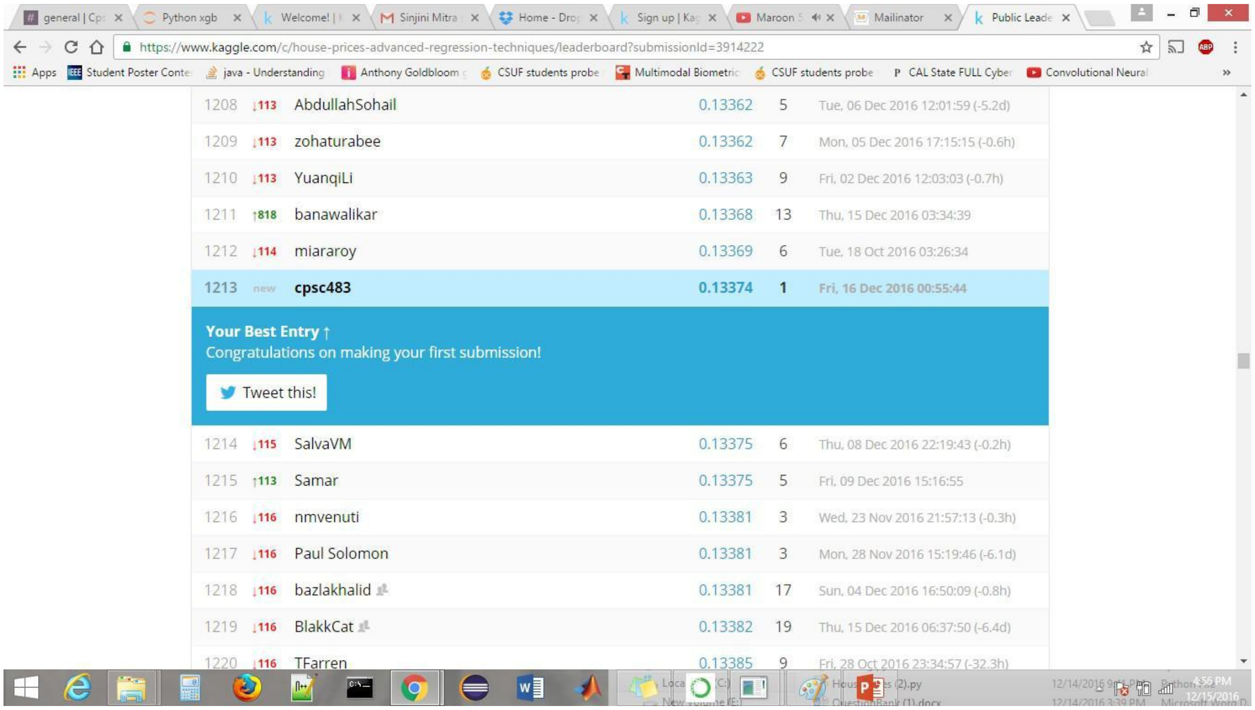
1. Distribution of the data



2. Data after Log Transformation



3. Result of performing Principal Component Analysis and Lasso



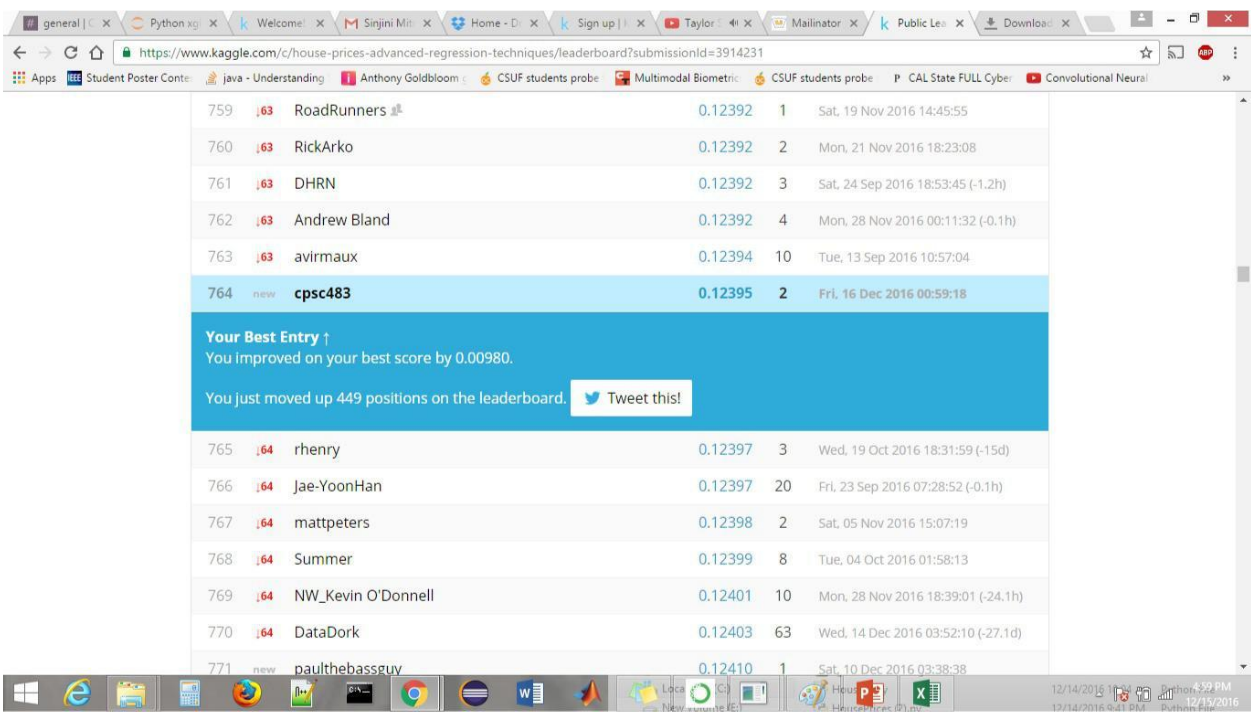
https://www.kaggle.com/c/house-prices-advanced-regression-techniques/leaderboard?submissionId=3914222

1208	↓113	AbdullahSohail	0.13362	5	Tue, 06 Dec 2016 12:01:59 (-5.2d)
1209	↓113	zohaturabee	0.13362	7	Mon, 05 Dec 2016 17:15:15 (-0.6h)
1210	↓113	YuanqiLi	0.13363	9	Fri, 02 Dec 2016 12:03:03 (-0.7h)
1211	↑818	banawalikar	0.13368	13	Thu, 15 Dec 2016 03:34:39
1212	↓114	miararoy	0.13369	6	Tue, 18 Oct 2016 03:26:34
1213	new	**cpsc483**	**0.13374**	**1**	**Fri, 16 Dec 2016 00:55:44**

Your Best Entry ↑
Congratulations on making your first submission!
Tweet this!

1214	↓115	SalvaVM	0.13375	6	Thu, 08 Dec 2016 22:19:43 (-0.2h)
1215	↑113	Samar	0.13375	5	Fri, 09 Dec 2016 15:16:55
1216	↓116	nmvenuti	0.13381	3	Wed, 23 Nov 2016 21:57:13 (-0.3h)
1217	↓116	Paul Solomon	0.13381	3	Mon, 28 Nov 2016 15:19:46 (-6.1d)
1218	↓116	bazlakhalid	0.13381	17	Sun, 04 Dec 2016 16:50:09 (-0.8h)
1219	↓116	BlakkCat	0.13382	19	Thu, 15 Dec 2016 06:37:50 (-6.4d)
1220	↓116	TFarren	0.13385	9	Fri, 28 Oct 2016 23:34:57 (-32.3h)

4. Lasso



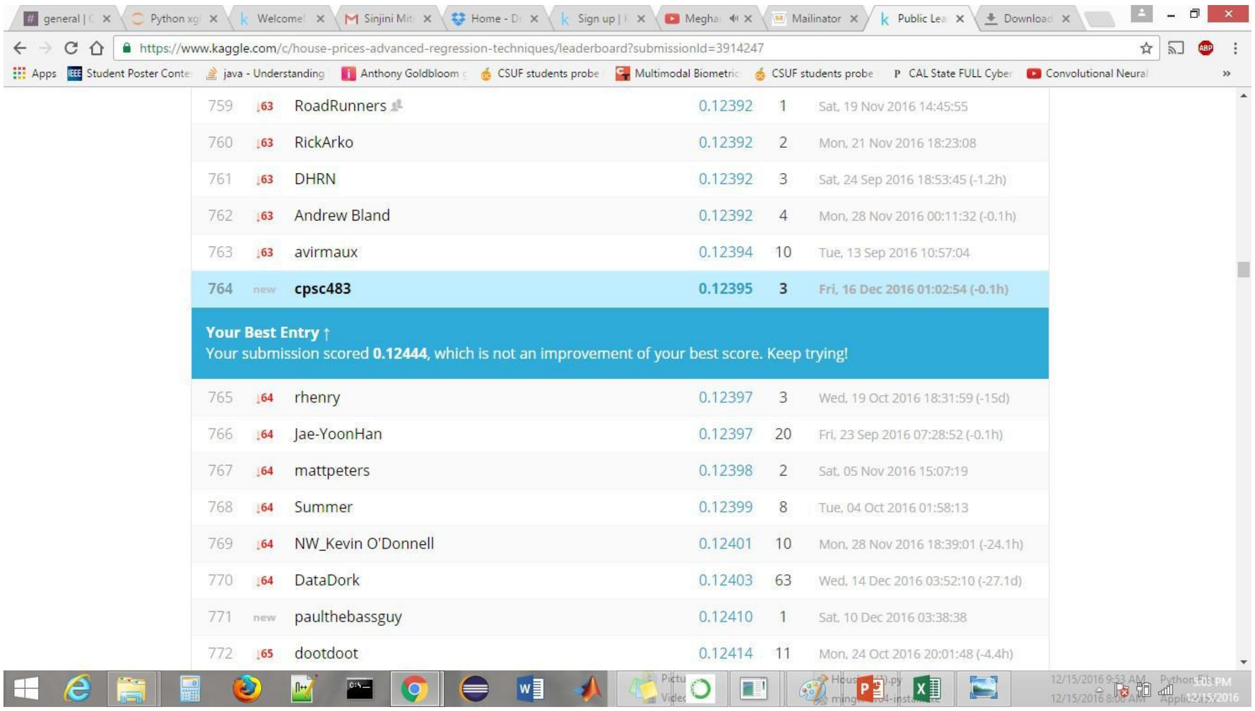
https://www.kaggle.com/c/house-prices-advanced-regression-techniques/leaderboard?submissionId=3914231

759	↓63	RoadRunners	0.12392	1	Sat, 19 Nov 2016 14:45:55
760	↓63	RickArko	0.12392	2	Mon, 21 Nov 2016 18:23:08
761	↓63	DHRN	0.12392	3	Sat, 24 Sep 2016 18:53:45 (-1.2h)
762	↓63	Andrew Bland	0.12392	4	Mon, 28 Nov 2016 00:11:32 (-0.1h)
763	↓63	avirmaux	0.12394	10	Tue, 13 Sep 2016 10:57:04
764	new	**cpsc483**	**0.12395**	**2**	**Fri, 16 Dec 2016 00:59:18**

Your Best Entry ↑
You improved on your best score by 0.00980.
You just moved up 449 positions on the leaderboard. Tweet this!

765	↓64	rhenry	0.12397	3	Wed, 19 Oct 2016 18:31:59 (-15d)
766	↓64	Jae-YoonHan	0.12397	20	Fri, 23 Sep 2016 07:28:52 (-0.1h)
767	↓64	mattpeters	0.12398	2	Sat, 05 Nov 2016 15:07:19
768	↓64	Summer	0.12399	8	Tue, 04 Oct 2016 01:58:13
769	↓64	NW_Kevin O'Donnell	0.12401	10	Mon, 28 Nov 2016 18:39:01 (-24.1h)
770	↓64	DataDork	0.12403	63	Wed, 14 Dec 2016 03:52:10 (-27.1d)
771	new	paulthebassguy	0.12410	1	Sat, 10 Dec 2016 03:38:38

5. Ridge

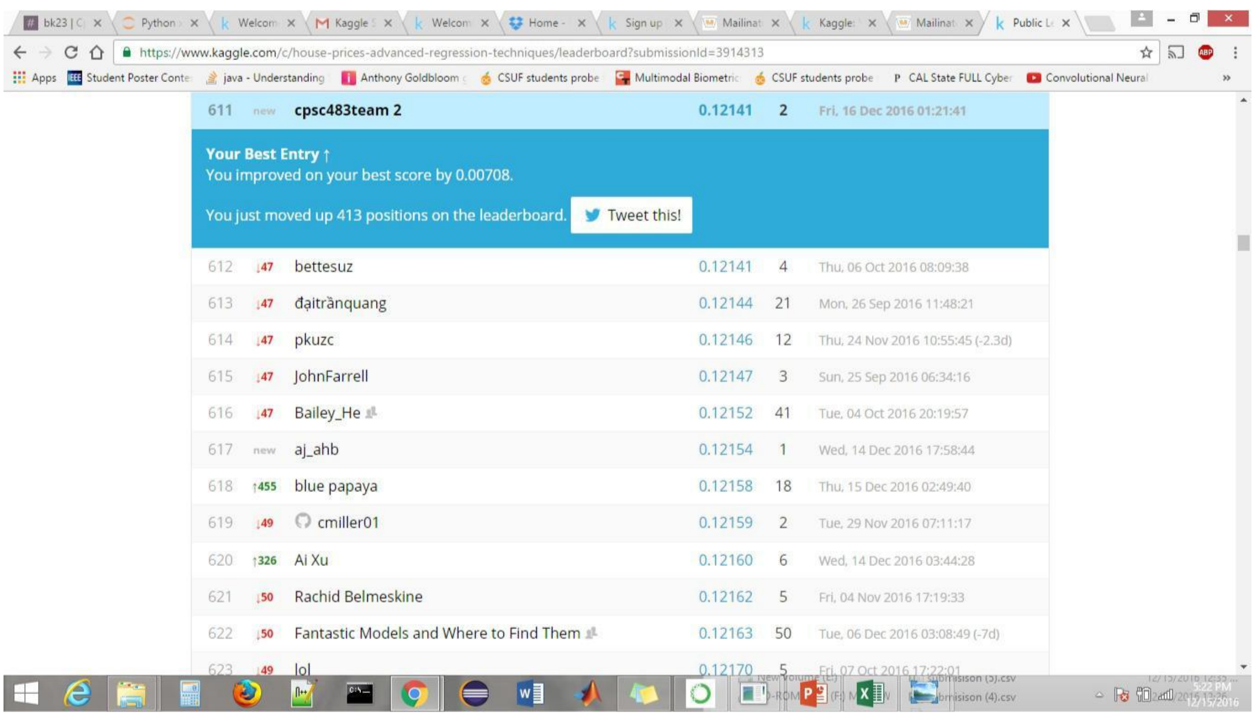


759	↓63	RoadRunners	0.12392	1	Sat, 19 Nov 2016 14:45:55
760	↓63	RickArko	0.12392	2	Mon, 21 Nov 2016 18:23:08
761	↓63	DHRN	0.12392	3	Sat, 24 Sep 2016 18:53:45 (-1.2h)
762	↓63	Andrew Bland	0.12392	4	Mon, 28 Nov 2016 00:11:32 (-0.1h)
763	↓63	avirmaux	0.12394	10	Tue, 13 Sep 2016 10:57:04
764	new	**cpsc483**	**0.12395**	**3**	**Fri, 16 Dec 2016 01:02:54 (-0.1h)**

Your Best Entry ↑
Your submission scored **0.12444**, which is not an improvement of your best score. Keep trying!

765	↓64	rhenry	0.12397	3	Wed, 19 Oct 2016 18:31:59 (-15d)
766	↓64	Jae-YoonHan	0.12397	20	Fri, 23 Sep 2016 07:28:52 (-0.1h)
767	↓64	mattpeters	0.12398	2	Sat, 05 Nov 2016 15:07:19
768	↓64	Summer	0.12399	8	Tue, 04 Oct 2016 01:58:13
769	↓64	NW_Kevin O'Donnell	0.12401	10	Mon, 28 Nov 2016 18:39:01 (-24.1h)
770	↓64	DataDork	0.12403	63	Wed, 14 Dec 2016 03:52:10 (-27.1d)
771	new	paulthebassguy	0.12410	1	Sat, 10 Dec 2016 03:38:38
772	↓65	dootdoot	0.12414	11	Mon, 24 Oct 2016 20:01:48 (-4.4h)

6. Ridge Lasso and Xgboost

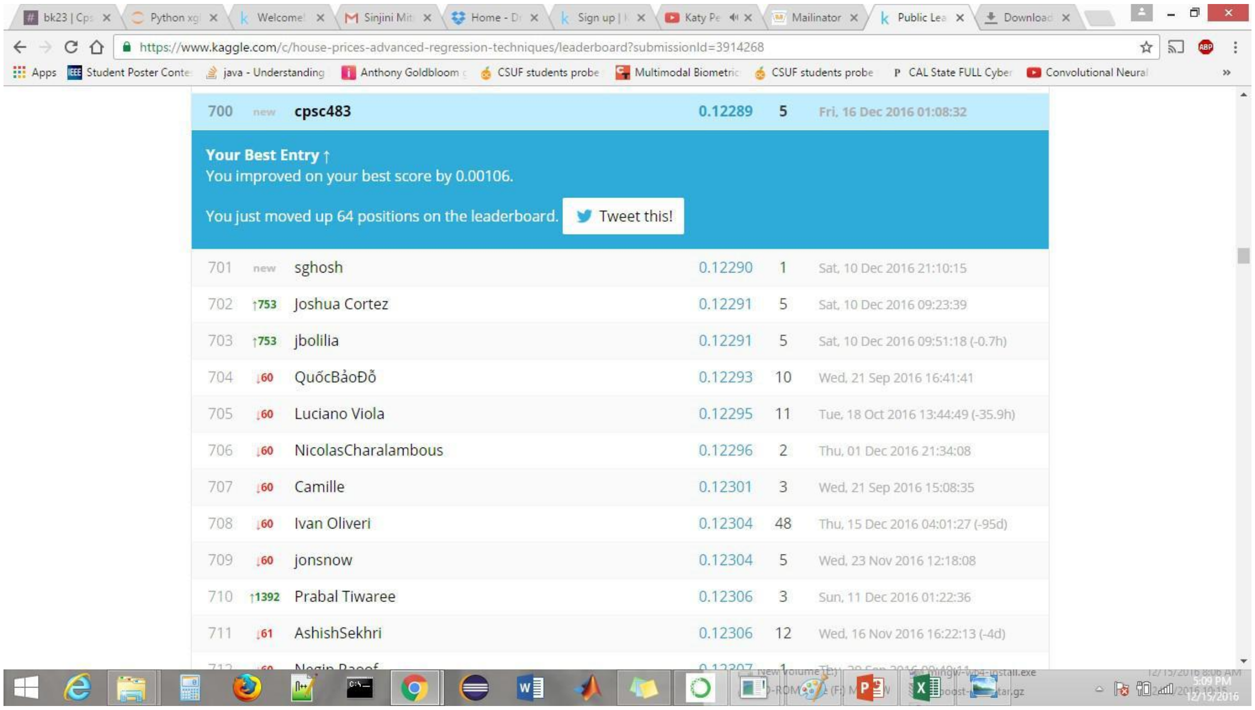


611	new	**cpsc483team 2**	**0.12141**	**2**	**Fri, 16 Dec 2016 01:21:41**

Your Best Entry ↑
You improved on your best score by 0.00708.
You just moved up 413 positions on the leaderboard. Tweet this!

612	↓47	bettesuz	0.12141	4	Thu, 06 Oct 2016 08:09:38
613	↓47	đạitrầnquang	0.12144	21	Mon, 26 Sep 2016 11:48:21
614	↓47	pkuzc	0.12146	12	Thu, 24 Nov 2016 10:55:45 (-2.3d)
615	↓47	JohnFarrell	0.12147	3	Sun, 25 Sep 2016 06:34:16
616	↓47	Bailey_He	0.12152	41	Tue, 04 Oct 2016 20:19:57
617	new	aj_ahb	0.12154	1	Wed, 14 Dec 2016 17:58:44
618	↑455	blue papaya	0.12158	18	Thu, 15 Dec 2016 02:49:40
619	↓49	cmiller01	0.12159	2	Tue, 29 Nov 2016 07:11:17
620	↑326	Ai Xu	0.12160	6	Wed, 14 Dec 2016 03:44:28
621	↓50	Rachid Belmeskine	0.12162	5	Fri, 04 Nov 2016 17:19:33
622	↓50	Fantastic Models and Where to Find Them	0.12163	50	Tue, 06 Dec 2016 03:08:49 (-7d)
623	↓49	lol	0.12170	5	Fri, 07 Oct 2016 17:22:01

7. XGBoost and Lasso



5 References

[1] http://scikit-learn.org/stable/index.html

[2] https://www.wikipedia.org/

[3] https://www.kaggle.com/c/house-prices-advanced-regression-techniques

[4] https://www.kaggle.com/apapiu/house-prices-advanced-regression-techniques/regularized-linear-models

[5] https://www.kaggle.com/humananalog/house-prices-advanced-regression-techniques/xgboost-lasso

[6] https://en.wikipedia.org/wiki/Normalization_(statistics)