

Applied Machine Learning HW3 (15%)

Matthew Frantz

1 Understanding the Evaluation Metric

This kaggle contest uses “Root Mean Squared Log Error” (RMSLE) rather than the standard “Root Mean Squared Error” (RMSE).

1. What exactly is this RMSLE error? (write the mathematical definition).
 - RMSLE takes the logarithm of both the value to be predicted and the value being measured against. Equation (1) is the mathematical definition of RMSLE with p_i being the predicted value and a_i being the actual value. RMSLE is useful because it avoids penalizing huge differences in predictions when you have large numbers such as housing prices.

$$RMSLE = \sqrt{\frac{1}{n} \sum_{i=1}^n (\log(p_i + 1) - \log(a_i + 1))^2} = \sqrt{\frac{1}{n} \sum_{i=1}^n \left(\frac{\log(p_i + 1)}{\log(a_i + 1)} \right)^2} \quad (1)$$

2. What’s the difference between RMSLE and RMSE?
 - The difference primarily comes from having large values in the data set. Once predicted and actual are large numbers, RMSE > RMSLE and the larger those numbers become the more they diverge. For example if two data points were predicted = 1000 and actual = 700, the error would be the same for 100000 and 70000 using RMSLE but would be huge for RMSE.
3. Why does this contest adopt RMSLE rather than RMSE?
 - According to their website they use RMSLE because the errors from cheap or expensive houses would be the same. This makes sense based on the properties of RMSLE described above.
4. Our TA Liang Zhang got an RMSLE score of 0.11 and was ranked 28 last Spring. What does this 0.11 mean intuitively, in terms of housing price prediction error?
 - Intuitively this means that log of the mean squared error was low meaning the actual error was quite small.
5. What are your RMSLE error and ranking if you just submit `sample_submission.csv`?
 - Score = 0.40890
6. What is your “Team name” on kaggle (note this HW should be done individually)?
 - mlfrantz

Table 1: Number of Features

Feature	Number	Feature	Number
MSSubClass	16	CentralAir	3
MSZoning	6	Electrical	7
LotFrontage	109	1stFlrSF	722
LotArea	990	2ndFlrSF	391
Street	3	LowQualFinSF	22
Alley	4	GrLivArea	811
LotShape	5	BsmtFullBath	5
LandContour	5	BsmtHalfBath	4
Utilities	3	FullBath	5
LotConfig	6	HalfBath	4
LandSlope	4	BedroomAbvGr	9
Neighborhood	26	KitchenAbvGr	5
Condition1	10	KitchenQual	5
Condition2	9	TotRmsAbvGrd	13
BldgType	6	Functional	8
HouseStyle	9	Fireplaces	5
OverallQual	11	FireplaceQu	7
OverallCond	10	GarageType	8
YearBuilt	111	GarageYrBlt	98
YearRemodAdd	62	GarageFinish	5
RoofStyle	7	GarageCars	6
RoofMatl	9	GarageArea	423
Exterior1st	16	GarageQual	7
Exterior2nd	17	GarageCond	7
MasVnrType	6	PavedDrive	4
MasVnrArea	306	WoodDeckSF	254
ExterQual	5	OpenPorchSF	194
ExterCond	6	EnclosedPorch	117
Foundation	7	3SsnPorch	18
BsmtQual	6	ScreenPorch	73
BsmtCond	6	PoolArea	9
BsmtExposure	6	PoolQC	5
BsmtFinType1	8	Fence	6
BsmtFinSF1	602	MiscFeature	6
BsmtFinType2	8	MiscVal	22
BsmtFinSF2	132	MoSold	13
BsmtUnfSF	731	YrSold	6
TotalBsmtSF	687	SaleType	10
Heating	7	SaleCondition	7
HeatingQC	5	Total	7227

2 Naive data processing: binarizing all fields

1. How many features do you get? (Hint: around 7230).

I found that there are 7227 binary features in the dataset.

2. How many features are there for each field?

Table 2 shows the number of features for each field.

3. Do you need to augment the space (add bias dimension) like in HW1, or does your regression tool automatically does it for you?

No. In `sklearn.linear_model.LinearRegression` one of the parameters is `fit_intercept` which automatically calculates the bias terms if set to true.

4. Train linear regression using `sklearn.linear_model.LinearRegression` or `np.polyfit` on `my_train.csv` and test on `my_dev.csv`. What's your root mean squared log error (RMSLE)?

The best value I was able to get was 0.1512. I made sure that I had the correct number of features and that they were all binarized. I noticed at first that I was adding additional unseen features from the dev set to my mapping, but I then removed those as well.

5. What are your top 10 most positive and top 10 most negative features? Do they make sense?

The top 10 most positive feature are:

- (a) FullBath = 3
- (b) Neighborhood = StoneBr
- (c) Neighborhood = NoRidge
- (d) OverallQual = 10
- (e) RoofMatl = WdShngl
- (f) OverallQual = 9
- (g) LotArea = 13891
- (h) GarageCars = 3
- (i) TotRmsAbvGrd = 10
- (j) PoolArea = 555

The above list is mostly not surprising as we expect houses with more bathrooms, stone veneers, wood roof shingles, higher quality, bigger garages, and a pool to be more expensive.

The bottom 10 negative features are:

- (a) MSZoning = C (all)
- (b) FullBath = 1
- (c) GarageCars = 1
- (d) TotRmsAboveGrd = 4
- (e) OverallQual = 4
- (f) OverallQual = 3
- (g) LotArea = 40094
- (h) 2ndFlrSF = 1538
- (i) TotalBsmtSF = 3138
- (j) BsmtFinSF1 = 2260

This again is not surprising as the list consists smaller houses with lower quality construction. The ones that a little strange are the lot size and square footage of the basement. My guess is that some of these are farm properties or have poorly finished basements which would be undesirable.

6. What's your feature weight for the bias dimension? Does it make sense?

The feature weight for the bias dimension was 251548.94. This value makes sense as the mean value of homes in the data set is 182159.05 and we know we have some outliers with home prices so we expect the intercept to be a little higher.

7. Now predict on `test.csv`, and submit your predictions to the kaggle server. What's your score (RMSLE) and ranking?

I was able to get a score of 0.17452 which put me in place 3557 on the leader board at the time of this typing.

3 Smarter binarization: Only binarizing categorical features

You might have observed that most numerical features shouldn't have been binarized: for example, features like `LotArea` are always positively correlated with the sale price.

1. What are the drawbacks of naive binarization? (Hint: data sparseness)

Some of the drawbacks include the aforementioned data sparseness since you have so many variables that are 0 valued for thousands of features in each data example. Also, as we saw in the KNN HW, binary values for numerical categories perform poorly because values such as 0SF for a Garage and 1000SF would be the same distance as 500SF and 600SF even though that is not the physical case.

2. Now binarize only the categorical features, and keep the numerical features as is. What about the mixed features such as `LotFrontage` and `GarageYrBlt`?

For mixed features I handled one each individually. For `LotFrontage` I sen NA values to 0, and for `GarageYrBlt` I replaced the NA value with the median value of all `GarageYrBlt` which was 1980.

3. Redo all the questions asked in the naive binarization section. (Hint: the new dev error should be around 0.14, which is much better than naive binarization).

- (a) How many features do you get?

I found that there are 304 binary features in the dataset.

- (b) How many features are there for each field?

Table 2 shows the number of features for each field.

- (c) Do you need to augment the space (add bias dimension) like in HW1, or does your regression tool automatically does it for you?

No. In `sklearn.linear_model.LinearRegression` one of the parameters is `fit_intercept` which automatically calculates the bias terms if set to true.

- (d) Train linear regression using `sklearn.linear_model.LinearRegression` or `np.polyfit` on `my_train.csv` and test on `my_dev.csv`. What's your root mean squared log error (RMSLE)?

The best value I was able to get was 0.1245.

- (e) What are your top 10 most positive and top 10 most negative features? Do they make sense?

The top 10 most positive feature are:

- i. `MSSubClass`
- ii. `LotArea`
- iii. `Utilities = AllPub`
- iv. `Utilities = AllPub`

Table 2: Number of Features

Feature	Number	Feature	Number
MSSubClass	1	CentralAir	3
MSZoning	6	Electrical	7
LotFrontage	1	1stFlrSF	1
LotArea	1	2ndFlrSF	1
Street	3	LowQualFinSF	1
Alley	4	GrLivArea	1
LotShape	5	BsmtFullBath	1
LandContour	5	BsmtHalfBath	1
Utilities	3	FullBath	1
LotConfig	6	HalfBath	1
LandSlope	4	BedroomAbvGr	1
Neighborhood	26	KitchenAbvGr	1
Condition1	10	KitchenQual	5
Condition2	9	TotRmsAbvGrd	1
BldgType	6	Functional	8
HouseStyle	9	Fireplaces	1
OverallQual	1	FireplaceQu	7
OverallCond	1	GarageType	8
YearBuilt	1	GarageYrBlt	1
YearRemodAdd	1	GarageFinish	5
RoofStyle	7	GarageCars	1
RoofMatl	9	GarageArea	1
Exterior1st	16	GarageQual	7
Exterior2nd	17	GarageCond	7
MasVnrType	6	PavedDrive	4
MasVnrArea	1	WoodDeckSF	1
ExterQual	5	OpenPorchSF	1
ExterCond	6	EnclosedPorch	1
Foundation	7	3SsnPorch	1
BsmtQual	6	ScreenPorch	1
BsmtCond	6	PoolArea	1
BsmtExposure	6	PoolQC	5
BsmtFinType1	8	Fence	6
BsmtFinSF1	1	MiscFeature	6
BsmtFinType2	8	MiscVal	1
BsmtFinSF2	1	MoSold	1
BsmtUnfSF	1	YrSold	1
TotalBsmtSF	1	SaleType	10
Heating	7	SaleCondition	7
HeatingQC	5	Total	304

- v. LotConfig = CulDSac
- vi. LotConfig = Corner
- vii. LotConfig = Inside
- viii. LotConfig = FR3
- ix. LotConfig = FR2
- x. MiscFeature = Shed

The above list is mostly not surprising as houses that on Cul-de-sacs, have all the utilities, or are on corner lots which tend to be bigger would be positively correlated with values.

The bottom 10 negative features are:

- i. Street = Grvl
- ii. Street = Pave
- iii. Alley = NA
- iv. Alley = Grvl
- v. Alley = Pave
- vi. LotShape = IR3
- vii. LotShape = IR1
- viii. LotShape = IR2
- ix. LotShape = Reg
- x. Condition1 = RRAe

It seems that the features that distract from the house price all have to deal with the lot and surrounding street. Although it seems from this list that it does not matter what your lot looks like or if you are near an alley, the price is going down.

- (f) What's your feature weight for the bias dimension? Does it make sense?

The feature weight for the bias dimension was 9149802.4. This value does not makes sense as the mean value of homes in the data set is 182159.05 and we know we have some outliers with home prices so we expect the intercept to be a little higher. I was unable to come up a logical reason for this value.

- (g) Now predict on `test.csv`, and submit your predictions to the kaggle server. What's your score (RMSLE) and ranking?

I was able to get a score of 0.14452 which put me in place 2714 on the leader board at the time of this typing.

4 Experimentation

Try the following to improve your score.

1. Try regularized linear regression (`sklearn.linear_model.Ridge`). Tune α on dev. Should improve both naive and smart binarization by a little bit.

For the all binary set I found hte best alpha was around 5.5 with a score of 0.1390 which was a large improvement over the previous value of 0.1512.

For the smart binary I did not see as much of an improvement as I went from 0.1245 to 0.1203 with an alpha value of 8.1 as the best one.

2. Try non-linear features: what if the sale price is quadratically correlated with some of the most important numerical features such as **OverallArea** (also known as square footage) and **LotArea**?

I found the following combination of nonlinear features by taking the logarithm of the following numerical features: 'LotArea', 'MasVnrArea', '1stFlrSF', '2ndFlrSF', 'GarageArea', 'YearBuilt', 'YearRemodAdd', 'YrSold'. By doing this I achieved a RMSLE of 0.1179 on the dev set. This seemed to work because all of those values were quite large numbers.

3. Try feature combinations. An obvious candidate is adding a new feature `Years_since_remodeled = YearRemodeled - YearBuilt`. But is a feature like this really useful?

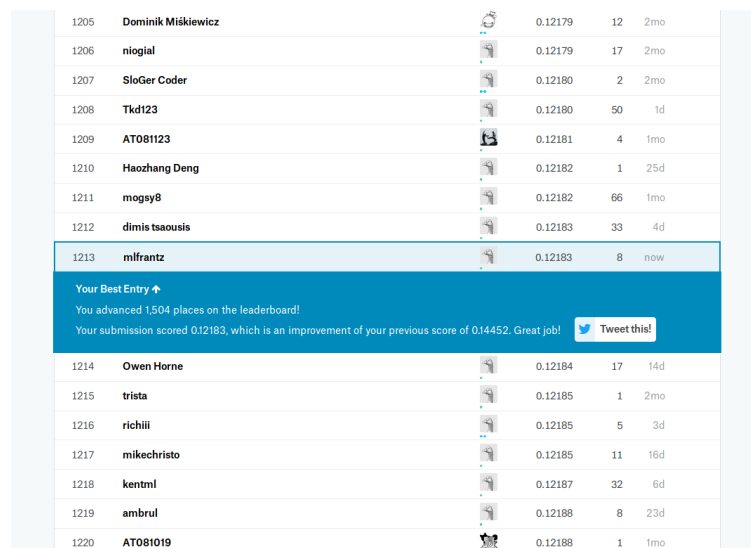
I tried several feature sets. Taking the difference of `OverallQual` and `OverallCond` made no improvement. `YearRemodeled - YearBuilt` also did not improve my score either. The best one I found was `TotalBsmtSF - MasVnrArea` with a dev error of 0.1177. This did not correlate with an improvement on the Kaggle set sadly.

4. BTW, how are these non-linear features (including feature combinations) relate to non-linear features in the perceptron? (think of XOR)

By using non-linear features we could potentially be taking possibly non-separable data when it is a linear feature and making it separable, thus providing better results with greater margins.

What's your best dev error, and what's your best test error and ranking? Take a screen shot of your best test error and ranking, and include your best submission file.

My best dev error was 0.1179 and my best test error was 0.12183 at rank 1213, which I think is pretty good. This was using ridge with non-linear logarithmic features. Figure 1 is a screen shot with my best values.



1205	Dominik Miśkiewicz	0.12179	12	2mo
1206	niogial	0.12179	17	2mo
1207	SloGer Coder	0.12180	2	2mo
1208	Tkd123	0.12180	50	1d
1209	AT081123	0.12181	4	1mo
1210	Haozhang Deng	0.12182	1	25d
1211	mogsy8	0.12182	66	1mo
1212	dimis tsaoasis	0.12183	33	4d
1213	mifrantz	0.12183	8	now
Your Best Entry ↑ You advanced 1,504 places on the leaderboard! Your submission scored 0.12183, which is an improvement of your previous score of 0.14452. Great job! Tweet this!				
1214	Owen Horne	0.12184	17	14d
1215	trista	0.12185	1	2mo
1216	richiii	0.12185	5	3d
1217	mikechristo	0.12185	11	16d
1218	kentnl	0.12187	32	6d
1219	ambrul	0.12188	8	23d
1220	AT081019	0.12188	1	1mo

Figure 1: Best test error and ranking on Kaggle.

Debriefing (required):

1. Approximately how many hours did you spend on this assignment?

Somewhere between 20-25 hours. This one was pretty long.

2. Would you rate it as easy, moderate, or difficult?

Difficult. I faced many issues with the linear regression tools giving me erroneous answers that I spent a lot of time debugging.

3. Did you work on it mostly alone, or mostly with other people?

Alone.

4. How deeply do you feel you understand the material it covers (0%–100%)?

85%. I think making us actually program the linear regression may have been more beneficial instead of just using the tools out there as I'm not sure I fully understand how the multivariate regression was different from the single variable. This seems contrary to the last two HW assignments where you wanted us to learn how these algorithms worked.

5. Any other comments?

None