

Name: Nirmil Patel

Email: patenirm@oregonstate.edu

1 Understanding the Evaluation Metric (1.5 pts)

1. What exactly is this RMSLE error? (write the mathematical definition). (0.5 pts)

The Root Mean Squared Logarithmic Error (RMSLE) is a metric used to evaluate the accuracy of predictions on datasets. The mathematical definition of RMSLE is given by:

$$RMSLE = \sqrt{\frac{1}{n} \sum_{i=1}^n (\log(\hat{y}_i + 1) - \log(y_i + 1))^2}$$

(from ProgrammerAH)

where:

- ☐ n is the number of observations in the dataset,
- ☐ \hat{y}_i is the predicted value,
- ☐ y_i is the actual value.
- ☐ $+1$ is to avoid taking log of 0

2. What's the difference between RMSLE and RMSE? (0.25 pts)

The main difference between RMSLE (Root Mean Squared Logarithmic Error) and RMSE (Root Mean Squared Error) lies in their treatment of errors. RMSE calculates the square root of the average squared differences between predicted and actual values, emphasizing large errors. On the other hand, RMSLE calculates the square root of the average squared differences of the logarithms of predicted and actual values.

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2}$$

(from ProgrammerAH)

3. Why does this contest adopt RMSLE rather than RMSE? (0.25 pts)

The contest adopts RMSLE rather than RMSE because RMSLE is better suited for scenarios where the target variable has a wide range of values. Taking the logarithm ensures that errors in predicting expensive houses and cheap houses equally contribute to the evaluation metric, providing a fair assessment of the model's performance across the entire range of house prices.

4. One of our TAs got an RMSLE score of 0.11 and was ranked 28 in Spring 2018. What does this 0.11 mean intuitively, in terms of housing price prediction error? (0.25 pts)

An RMSLE score of 0.11 implies that, on average, the model's predicted logarithm of housing prices deviates by approximately 11% from the logarithm of the actual prices.

5. What are your RMSLE error and ranking if you just submit sample_submission.csv? (0.25 pts)

Score: 0.406

Ranking: 4458

6. What is your “Team name” on kaggle (note this HW should be done individually)? (0 pts)

Nirmit B Patel

It is highly recommended that you take the logarithm for the y field (SalePrice) before doing any experiment, so that you convert the objective back to RMSE (i.e., optimizing RMSE of $\log y$ is equivalent to optimizing RMSLE of y). However, do not forget to exponentiate it back before submitting to kaggle. In other words, you train your regression systems to predict the logarithm of housing prices, $\log(y)$, but you submit your predictions in the original prices, not the ones after logarithm.

[Extra credit question \(+1 pt\): Why do you need to do this?](#)

Taking the logarithm of housing prices makes the predictions more accurate and easier to work with. It helps the model better understand and represent the patterns in housing prices. Additionally, it ensures that the model is equally sensitive to both small and large price differences, making the predictions more balanced and reliable.

2 Naive data processing: binarizing all fields (4.75 pts)

1. How many features do you get? (Hint: around 7,230). (0.75 pts)

7227

2. How many features are there for each field? (0.5 pts)

MSSubClass	16
MSZoning	6
LotFrontage	109
LotArea	990
Street	3
Alley	4
LotShape	5
LandContour	5
Utilities	3
LotConfig	6
LandSlope	4
Neighborhood	26
Condition1	10
Condition2	9
BldgType	6
HouseStyle	9
OverallQual	11
OverallCond	10
YearBuilt	111
YearRemodAdd	62
RoofStyle	7
RoofMatl	9
Exterior1st	16
Exterior2nd	17
MasVnrType	6
MasVnrArea	306
ExterQual	5
ExterCond	6
Foundation	7
BsmtQual	6
BsmtCond	6
BsmtExposure	6
BsmtFinType1	8
BsmtFinSF1	602
BsmtFinType2	8
BsmtFinSF2	132
BsmtUnfSF	731
TotalBsmtSF	687
Heating	7
HeatingQC	5

CentralAir	3
Electrical	7
1stFlrSF	722
2ndFlrSF	391
LowQualFinSF	22
GrLivArea	811
BsmtFullBath	5
BsmtHalfBath	4
FullBath	5
HalfBath	4
BedroomAbvGr	9
KitchenAbvGr	5
KitchenQual	5
TotRmsAbvGrd	13
Functional	8
Fireplaces	5
FireplaceQu	7
GarageType	8
GarageYrBlt	98
GarageFinish	5
GarageCars	6
GarageArea	423
GarageQual	7
GarageCond	7
PavedDrive	4
WoodDeckSF	254
OpenPorchSF	194
EnclosedPorch	117
3SsnPorch	18
ScreenPorch	73
PoolArea	9
PoolQC	5
Fence	6
MiscFeature	6
MiscVal	22
MoSold	13
YrSold	6
SaleType	10
SaleCondition	7

3. 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) on dev? (Hint: should be ~0.152). (1 pt)

0.152

4. What are your top 10 most positive and top 10 most negative features? Do they make sense? (1 pt)

Top 10 most positive features:

	Feature	Coefficient
5900	FullBath_3	0.139233
1203	OverallQual_9	0.137390
1162	Neighborhood_StoneBr	0.125355
4712	2ndFlrSF_472	0.113156
1202	OverallQual_8	0.106514
1398	RoofMatl_WdShngl	0.092514
5283	GrLivArea_1192	0.089529
1155	Neighborhood_NoRidge	0.087764
6060	GarageCars_3	0.086167
391	LotArea_8029	0.085358

Top 10 most negative features:

	Feature	Coefficient
15	MSZoning_C(all)	-0.204502
5165	GrLivArea_968	-0.126627
7040	EnclosedPorch_236	-0.123473
1197	OverallQual_3	-0.112828
2443	BsmtFinSF2_311	-0.108804
420	LotArea_8281	-0.107356
1207	OverallCond_3	-0.101119
6058	GarageCars_1	-0.094178
218	LotArea_5000	-0.091047
1195	OverallQual_1	-0.089260

Yes, the features do make sense.

5. Do you need to add the bias dimension (i.e., augmented space) explicitly like in HW2, or does your regression tool automatically handle it for you? Hint: `coef_` and `intercept_`. What's your feature weight for the bias dimension? Does it make sense? (0.5 pts)

No, I don't need to add a bias dimension explicitly. The `LinearRegression` from `sklearn` takes care of it. We can get the bias dimension feature weight by using `intercept_`.
I am getting bias weight as: 12.173

Extra credit question (+1 pt): What's the intuitive meaning (in terms of housing price) of this bias feature weight?

The bias term, obtained from the intercept in linear regression (12.173), represents the baseline or expected housing price when all other features are zero. It reflects the inherent value of a house, accounting for factors not explicitly captured by the dataset features. This baseline price serves as a starting point in housing price prediction, encompassing basic construction costs, land value, and other fixed expenses.


6. Now predict on `test.csv`, and submit your predictions to the kaggle server. What's your score (RMSLE, should be around 0.16) and ranking? (1 pt)

Score: 0.157

Ranking: 3113

3113


Nirmit B Patel



0.15742

5

20s



Your Best Entry!

Your most recent submission scored 0.15742, which is an improvement of your previous score of 0.19419. Great job!

[Tweet this](#)

3 Smarter binarization: Only binarizing categorical features (3.5 pts)

1. What are the drawbacks of naive binarization? (Hint: data sparseness, etc.) (0.5 pts)

Naive binarization leads to a loss of continuous information, especially in numerical features eroding their predictive capacity. The process results in data sparsity and increased dimensionality. This approach can obscure meaningful patterns in numerical features and make models more susceptible to overfitting.

2. Now binarize only the categorical features, and keep the numerical features as is. What about the mixed features such as LotFrontage and GarageYrBlt? (1 pt)

For mixed features like LotFrontage and GarageYrBlt, which have both numerical values and some missing (NA) values, the approach is to treat them as numerical features. I filled the missing values with zeros to ensure numerical processing. This allows the model to handle them consistently with other numerical features during binarization.

3. Redo the following questions from the naive binarization section. (Hint: the new dev error should be around 0.14, which is much better than naive binarization). (2 pts)

(a) How many features are there in total? (0.25 pts)

316

(b) What's the new dev error rate (RMSLE)? (0.5 pts) (should be ~0.12)

0.129

(c) What are the top 10 most positive and top 10 most negative features? Are they different from the previous section? (0.5 pts)

Top 10 most positive features:

	Feature	Coefficient
106	cat_RoofMatl_Membran	0.629644
255	cat_PoolQC_nan	0.519604
80	cat_Condition2_PosA	0.475728
107	cat_RoofMatl_Metal	0.472267
103	cat_RoofStyle_Shed	0.407139
91	cat_HouseStyle_1.5Unf	0.352872
261	cat_MiscFeature_Gar2	0.284472
237	cat_GarageQual_Ex	0.281798
111	cat_RoofMatl_WdShngl	0.268038
108	cat_RoofMatl_Roll	0.256014

Top 10 most negative features:

	Feature	Coefficient
104	cat_RoofMatl_ClyTile	-2.247478
81	cat_Condition2_PosN	-0.700464
82	cat_Condition2_RRAe	-0.627747
7	cat_MSSubClass_45	-0.366650
15	cat_MSZoning_C (all)	-0.346170
253	cat_PoolQC_Fa	-0.282552
114	cat_Exterior1st_BrkComm	-0.232786
218	cat_Functional_Sev	-0.217972
206	cat_Electrical_Mix	-0.200493
214	cat_Functional_Maj2	-0.196253

Yes, they are different from previous section.


- (d) Now predict on test.csv, and submit your predictions to the kaggle server. What's your score (RMSLE, should be ~0.13) and ranking? (0.75 pts)

Score: 0.151

Ranking: 2779

2779


Nirmit B Patel



0.15131

6

19s



Your Best Entry!
Your most recent submission scored 0.15131, which is an improvement of your previous score of 0.15742. Great job!

Tweet this

4 Experimentation (5.25 pts)

1. Try regularized linear regression (sklearn.linear_model.Ridge). Tune α on dev. Should improve both naive and smart binarization by a little bit. (0.5 pts)

Using Regularized linear regression on naive binarization:

Alpha: 1e-06, RMSLE on Dev Set: 0.1518761270314034
Alpha: 1e-05, RMSLE on Dev Set: 0.1518741634486964
Alpha: 0.0001, RMSLE on Dev Set: 0.15185457299630173
Alpha: 0.001, RMSLE on Dev Set: 0.15166308380295077
Alpha: 0.01, RMSLE on Dev Set: 0.15009633744357512
Alpha: 0.1, RMSLE on Dev Set: 0.14475500708275763
Alpha: 1.0, RMSLE on Dev Set: 0.14064122620436637
Alpha: 10.0, RMSLE on Dev Set: 0.13946034696364779
Alpha: 100.0, RMSLE on Dev Set: 0.15429300411061558
Alpha: 1000.0, RMSLE on Dev Set: 0.19565803757107875
Alpha: 10000.0, RMSLE on Dev Set: 0.2937950820432581
Alpha: 100000.0, RMSLE on Dev Set: 0.35482143967021834
Alpha: 1000000.0, RMSLE on Dev Set: 0.3639510748753207


Using Regularized linear regression on smart binarization:

Alpha: 1e-05, RMSLE on Dev Set: 0.129341484415729
Alpha: 0.0001, RMSLE on Dev Set: 0.12933869188753663
Alpha: 0.001, RMSLE on Dev Set: 0.12931105327622275
Alpha: 0.01, RMSLE on Dev Set: 0.1290594407478855
Alpha: 0.1, RMSLE on Dev Set: 0.12773790768594187
Alpha: 1.0, RMSLE on Dev Set: 0.12681305739675544
Alpha: 10.0, RMSLE on Dev Set: 0.1258292648816759
Alpha: 100.0, RMSLE on Dev Set: 0.12732200223756526
Alpha: 1000.0, RMSLE on Dev Set: 0.13662714084181343
Alpha: 10000.0, RMSLE on Dev Set: 0.15446973542575876
Alpha: 100000.0, RMSLE on Dev Set: 0.1580726687248719
Alpha: 1000000.0, RMSLE on Dev Set: 0.1593155209836359

Kaggle submission:

1046


Nirmit B Patel



0.13119

7

21s



Your Best Entry!

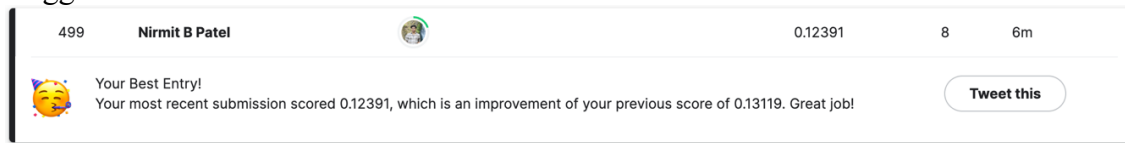
Your most recent submission scored 0.13119, which is an improvement of your previous score of 0.15131. Great job!

[Tweet this](#)

2. Try non-linear regression (sklearn.preprocessing.PolynomialFeatures) (1 pt).

RMSLE on Dev: 0.119

Kaggle submission:



A notification card from Kaggle showing a submission by Nimit B Patel. The card includes a rank of 499, a score of 0.12391, and a '6m' (6 months) time indicator. It features a 'Your Best Entry!' badge with a trophy icon and a congratulatory message: 'Your most recent submission scored 0.12391, which is an improvement of your previous score of 0.13119. Great job!'. There is a 'Tweet this' button on the right.

3. How are these non-linear features (including feature combinations) relate to non-linear features in the perceptron? (think of XOR) (0.25 pts)

The non-linear features, including feature combinations, relate to non-linear features in the perceptron in the context of capturing complex relationships. In the XOR problem, a perceptron with a linear activation function is unable to learn the non-linear decision boundary required to correctly classify the XOR function. Similarly, by introducing non-linear features or combinations in regression, we allow the model to capture non-linear patterns in the data, similar to how non-linear activation functions in neural networks enable them to learn complex mappings. Both scenarios involve introducing non-linearity to handle more intricate relationships between input features and the target variable.

4. Try anything else that you can think of. You can also find inspirations online, but you have to implement everything yourself (you are not allowed to copy other people's code). (0.5 pts)

I tried Random Forest Regression and got RMSLE on dev: 0.137

Kaggle Submission:



A submission status card for 'random_forest.csv'. It shows a green checkmark icon, the filename 'random_forest.csv', and the status 'Complete · 3m ago · Random Forest Regression'. The score '0.14451' is displayed on the right.

Debriefing

1. Approximately how many hours did you spend on this assignment?
30 hours
2. Would you rate it as easy, moderate, or difficult?
Moderate
3. Did you work on it mostly alone, or mostly with other people?
Mostly alone
4. How deeply do you feel you understand the material it covers (0%–100%)?
80%
5. Any other comments?
It was nice that we could test our test predictions on Kaggle so that we can know what progress we have made in our model.