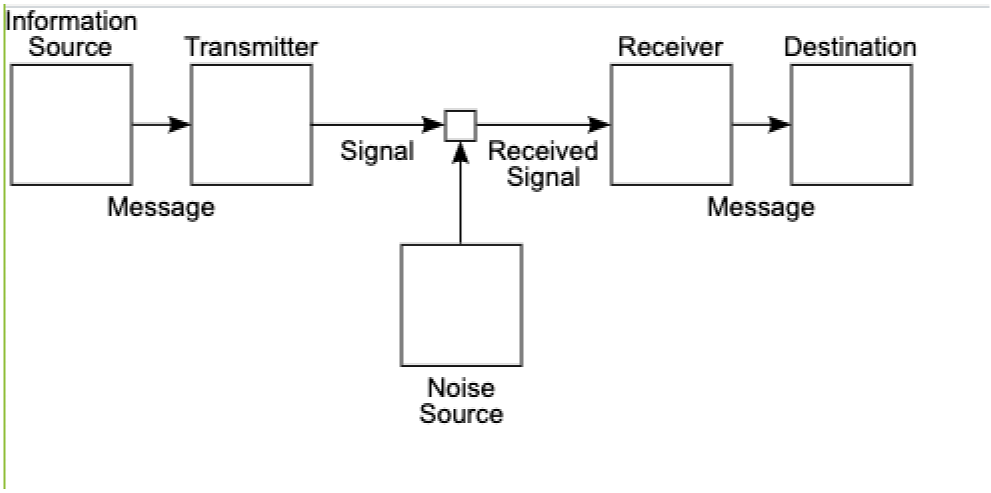
Homework 3, Business Data Science
Preety Pinghal, Preeti Gupta, Jonathan Cope

Problem 1: A Bit of Information Theory

In 1948, Claude Shannon revolutionized the world with his Information theory. This was the first time the word 'bit' was used. He explained the mathematics behind quantifying the information. He simplified the entire communication system with the following:



Source: WikiPedia (Public Domain)[1]

Entropy Encoding Theory:

The entropy was originally created by Shannon as part of his theory of communication, in which a data communication system as shown above comprises of three major components a source of data (Transmitter), a channel via which information can travel and a receiver. The major problem is for the receiver to be able to identify the signal which is sent by the transmitter, based on the signal it receives from the channel. Shannon considered various ways to encode, compress, and transmit messages from a data source, and proved that the entropy represents an absolute mathematical limit on how well data from the source can be losslessly compressed onto a perfectly noiseless channel. This entropy was given by:

$$H(X) = -\sum_{i=1}^{n} P(x_i)\, log\, P(x_i)$$

Shannon's Noise-channel coding theory:

[1] "A Mathematical Theory of Communication - Wikipedia." https://en.wikipedia.org/wiki/A_Mathematical_Theory_of_Communication. Accessed 26 Sep. 2021.

The Shannon limit or Shannon capacity of a communication channel refers to the maximum rate (or the maximum number of bits /sec) of error-free data that can theoretically be transferred over the channel if the link is subject to random data transmission errors, for a particular noise level.

He proved that it is impossible to send data at a rate higher than this specified limit mathematically.

The Shannon theorem states that given a noisy channel with channel capacity C and information transmitted at a rate R,

$$R < C$$

there exist codes that allow the probability of error at the receiver to be made arbitrarily small. This means that, theoretically, it is possible to transmit information nearly without error at any rate below a limiting rate, C.

The converse is also important. If,

$$R > C$$

an arbitrarily small probability of error is not achievable. All codes will have a probability of error greater than a certain positive minimal level, and this level increases as the rate increases. So, information cannot be guaranteed to be transmitted reliably across a channel at rates beyond the channel capacity. However, he doesn't speak about these codes in his paper.

Problem 3: More on Kaggle Advanced Regression

Part 1.

Best resource found : https://github.com/chouhbik/Kaggle-House-Prices
Exploratory Data Analysis

1. Used correlation between the target variable and the given input columns to decide the top 10 features which are highly correlated to the target variable.
2. Imputing Null Values :
    a. Replaced meaningful Nan values with “None”, in some cases Nan value represents something meaningful , such as BsmtQual: Nan – this might mean that there is now basement in the house.
    b. Treating columns with many missing values and few missing values differently for the purpose of imputing Nan.
        i. Dropping the columns which have too high a number of Nan values as they might not be able to contribute much meaningful information in the model.
        ii. Columns with few nan values, imputing nan with mean if numerical data or with the most common occurring category of categorical data.
3. Feature Engineering :
    a. Since the target variable is left skewed, it is more useful to take a logarithm of it in order to get a more normal distribution. Generally machine learning models work better with normal distribution targets.
    b. Adding a few new features which are a combination of the already existing features:
        i. Total SF equals basement SF + 1st floor SF and 2nd floor SF.

df_train_add['TotalSF']=df_train_add['TotalBsmtSF'] + df_train_add['1stFlrSF'] + df_train_add['2ndFlrSF']

ii. Total_Bathrooms equals the weighted average of full bathrooms and half bathrooms and full bathroom basement and half bathroom basement.

df_train_add['Total_Bathrooms'] = (df_train_add['FullBath'] + (0.5 * df_train_add['HalfBath']) + df_train_add['BsmtFullBath'] + (0.5 * df_train_add['BsmtHalfBath']))

iii. df_train_add['Total_porch_sf'] = (df_train_add['OpenPorchSF'] + df_train_add['3SsnPorch'] + df_train_add['EnclosedPorch'] + df_train_add['ScreenPorch'] + df_train_add['WoodDeckSF'])

c. Adding a few features which are based on whether the house has certain features for example “haspool” if the poolArea for the house is greater than 0

4. Finding Outliers:
5. Replacing categorical data with numerical categories.

ML Models :

- Cost functions: R2 score and RMSE. Using Cross Validation to optimize hyperparameters.

1. Linear Regression:

Using the log of the target variable as the target variable. Split the data in train and cross validation data. Calculated the mean absolute error , Mean squared error and root mean squared error. Then adding the gridSearchCV function for hyperparameter tunningnand getting the best model for linear regression. R2 score is being used for scoring here.

Highlight : hyperparameter tuning with GridSearch Cv and then retraining again with the best hyperparameters using 10 - fold cross validation.

The model gives a Cross Validation Score: 0.8829144856815347 with 10-fold cross validation.

2. Ridge Regression:

Similarly for ridge regression, hyperparameter tuning and then training again with 10 fold cross validation.

Cross Validation Score: 0.8829179714067902

Part 2

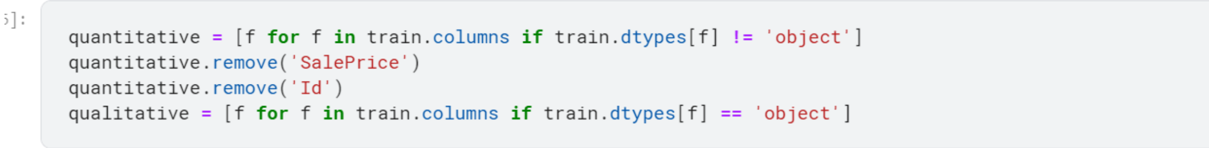
The best public leaderboard score which we managed to achieve is shown below:

The approach we took for this is the following:

Data Pre Processing:

We first took a deep dive into the data and classified it into ‘qualitative ‘ and ‘quantitative’ attributes to study the features and their correlations.

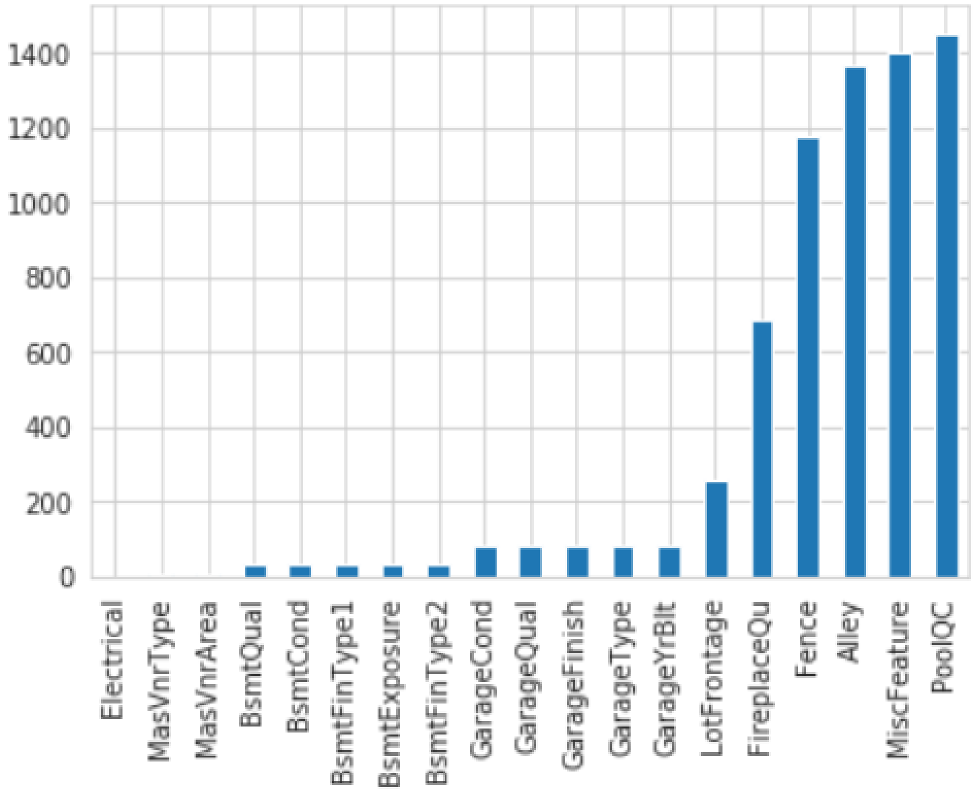
There are 1460 instances of training data and 1460 of test data. Total number of attributes equals 81, of which 36 is quantitative, 43 categorical + Id and Sale Price.



```
quantitative = [f for f in train.columns if train.dtypes[f] != 'object']
quantitative.remove('SalePrice')
quantitative.remove('Id')
qualitative = [f for f in train.columns if train.dtypes[f] == 'object']
```

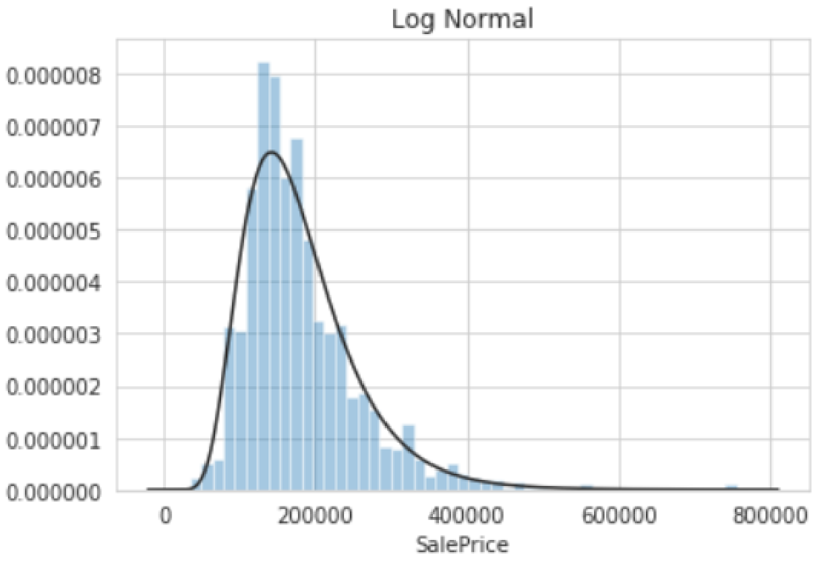
Nan:

We did a sns grid plotting and concluded that-



19 attributes have missing values, 5 over 50% of all data. Most of times NA means lack of subject described by attribute, like missing pool, fence, no garage and basement and they will definitely impact the sales price, so we can't blindly replace them with mean or so.What we did is basically substituting with 0 in case of quantitative variables and substituting with 'None' in case of qualitative.

Next, we found that the data was very skewed.



It is apparent that SalePrice doesn't follow normal distribution, so before performing regression it has to be transformed. While log transformation does a pretty good job, the best fit is unbounded Johnson distribution.

Feature Transformation:

Feature Transformation was done using Principal Component Analysis that is removing unwanted features, and also by Label encoding for multi classifiers .Also, outliers were handled by Standardization of data.

Models:

We trained various models including –

- Lasso,
- Ridge,
- ElasticNet
- Svr etc

and calculated their Cross-validation score to get the best model. Further we used Model blending to achieve the best score.

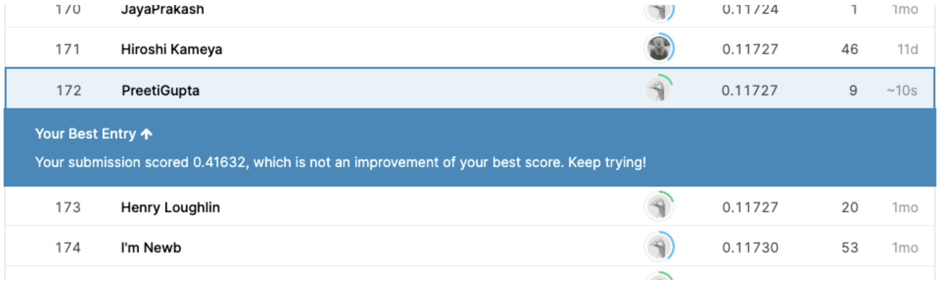
Hyperparameter Tuning :

For hyperparameter tuning we are splitting training data further into train and cross validation

Data. Once the hyperparameter tuning is done then we train on the entire train data as more the

Training data improves the model performance.

Ensembling :Stacking various models together.



170	JayaPrakash	0.11724	1	1mo
171	Hiroshi Kameya	0.11727	46	11d
172	PreetiGupta	0.11727	9	~10s

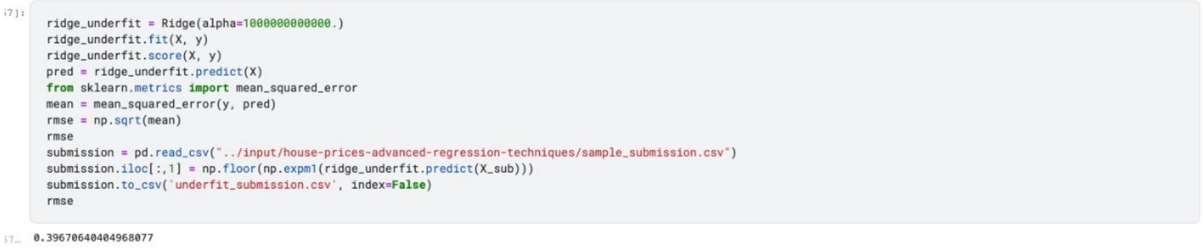
Your Best Entry ↑

Your submission scored 0.41632, which is not an improvement of your best score. Keep trying!

173	Henry Loughlin	0.11727	20	1mo
174	I'm Newb	0.11730	53	1mo

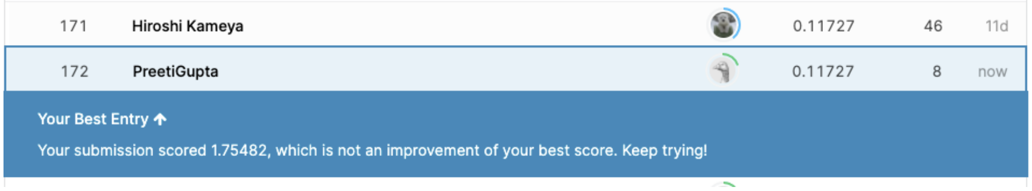
Under Fitting:

RMSE Value



```
ridge_underfit = Ridge(alpha=1000000000000.)
ridge_underfit.fit(X, y)
ridge_underfit.score(X, y)
pred = ridge_underfit.predict(X)
from sklearn.metrics import mean_squared_error
mean = mean_squared_error(y, pred)
rmse = np.sqrt(mean)
rmse
submission = pd.read_csv("../input/house-prices-advanced-regression-techniques/sample_submission.csv")
submission.iloc[:,1] = np.floor(np.expm1(ridge_underfit.predict(X_sub)))
submission.to_csv('underfit_submission.csv', index=False)
rmse
```

```
0.39670640404968077
```

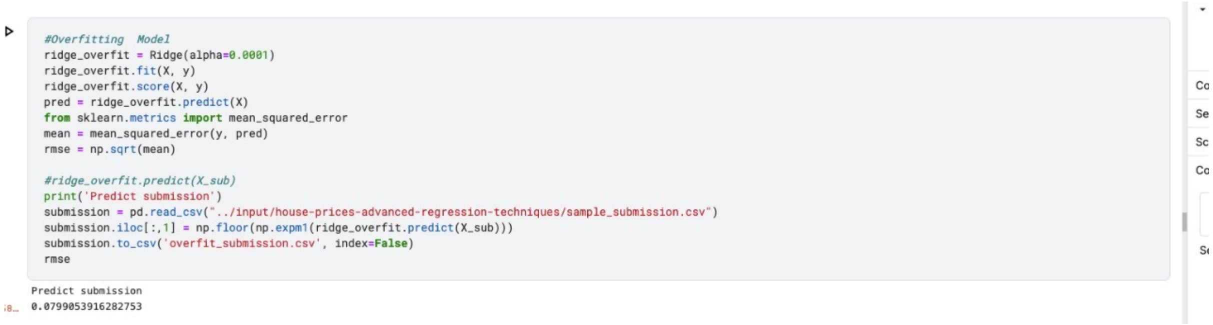


171	Hiroshi Kameya	0.11727	46	11d
172	PreetiGupta	0.11727	8	now

Your Best Entry ↑

Your submission scored 1.75482, which is not an improvement of your best score. Keep trying!

OverFitting Rmse value:

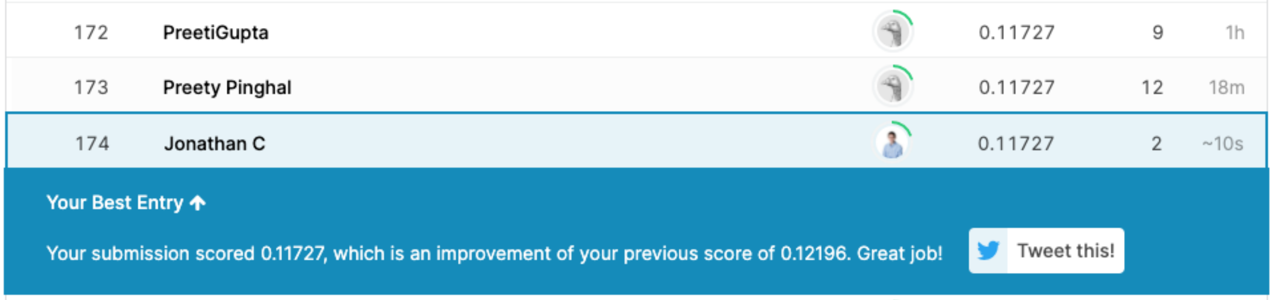


```
#Overfitting  Model
ridge_overfit = Ridge(alpha=0.0001)
ridge_overfit.fit(X, y)
ridge_overfit.score(X, y)
pred = ridge_overfit.predict(X)
from sklearn.metrics import mean_squared_error
mean = mean_squared_error(y, pred)
rmse = np.sqrt(mean)

#ridge_overfit.predict(X_sub)
print('Predict submission')
submission = pd.read_csv("../input/house-prices-advanced-regression-techniques/sample_submission.csv")
submission.iloc[:,1] = np.floor(np.expm1(ridge_overfit.predict(X_sub)))
submission.to_csv('overfit_submission.csv', index=False)
rmse
```

```
Predict submission
0.0799053916282753
```

Overall:



172	PreetiGupta	0.11727	9	1h
173	Preety Pinghal	0.11727	12	18m
174	Jonathan C	0.11727	2	~10s

Your Best Entry ↑

Your submission scored 0.11727, which is an improvement of your previous score of 0.12196. Great job!