House Prices – Advanced Regression Techniques

Yoel Bokobza & Oded George

Background

- In this mini-project, we aim to predict the sale price of houses based on the related features, using various machine learning (ML) techniques.
- To this aim, we are using the data set supplied by the Kaggle competition – “**House Prices - Advanced Regression Techniques**”
- Housing sales price are determined by numerous factors such as material quality, living area square feet ,Size of garage, location of the house and so on.

Data

- The data set is divided into training set and test set
- The training set contain 1460 samples, each consists 79 features and the corresponded house sale price.
- The test set contain 1459 samples, each consists 79 features.
- Among the features, 23 are nominal, 23 are ordinal, 14 are discrete, and 19 are continuous.
- The sale price is a continuous value.

Formulating the Research Question

- Let X be a vector of **Explanatory variables** (features vector).
- X contains 79 different features
- Our mission is to find a mapping $f: X \rightarrow \mathbb{R}$, such that given a new explanatory variables X, $f(X)$ will be an estimation of the house sale price.

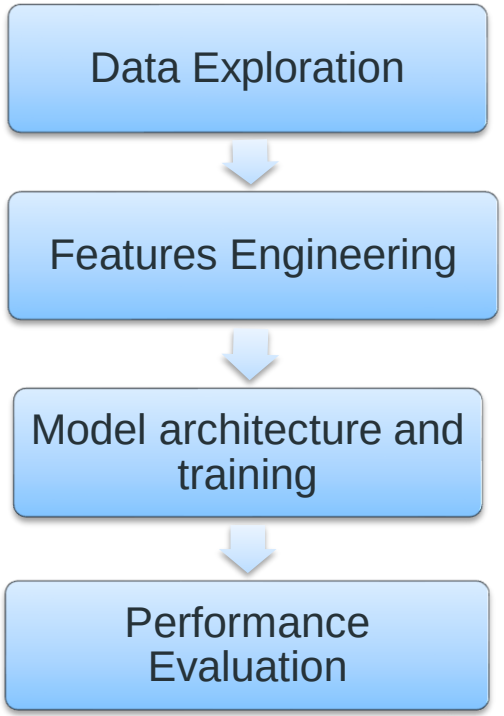
Methods

- A data driven approach has used.
- We use the train explanatory variables, X_{train} , and the corresponded Response variable (target variable), $Y_{train} \in \mathbb{R}$, to train a model for finding the mapping f.
- As the response variable is continuous, our problem is a regression problem.
- We plan to use ML regression algorithms to find the mapping f that is well generalized to unseen data.

Performance Evaluation

- Performance are tested by applying the function f over some feature vectors X_{test}, which are not used for training.
- The corresponded target variables are submitted to Kaggle to compare the predicted sale prices with the actual sale prices.
- Define $\hat{Y}_{test}^{(i)}$ and $Y_{test}^{(i)}$ to be the predicted house sale price and the actual house sale price of the i'th test sample , respectively.
- The score is $\sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\log\left(\hat{Y}_{test}^{(i)}\right)-\log\left(Y_{test}^{(i)}\right)\right)^2}$, where n is the number of test samples.
- Smaller score indicates a better model.

Workflow


Data Exploration
Features Engineering
Model architecture and training
Performance Evaluation

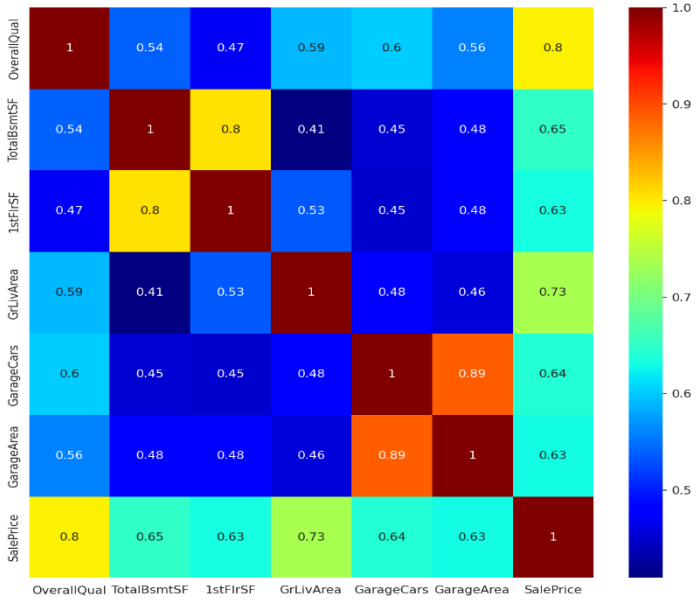
1.
Data Exploration

“God is in the details” – Ludwig Mies van der Rohe

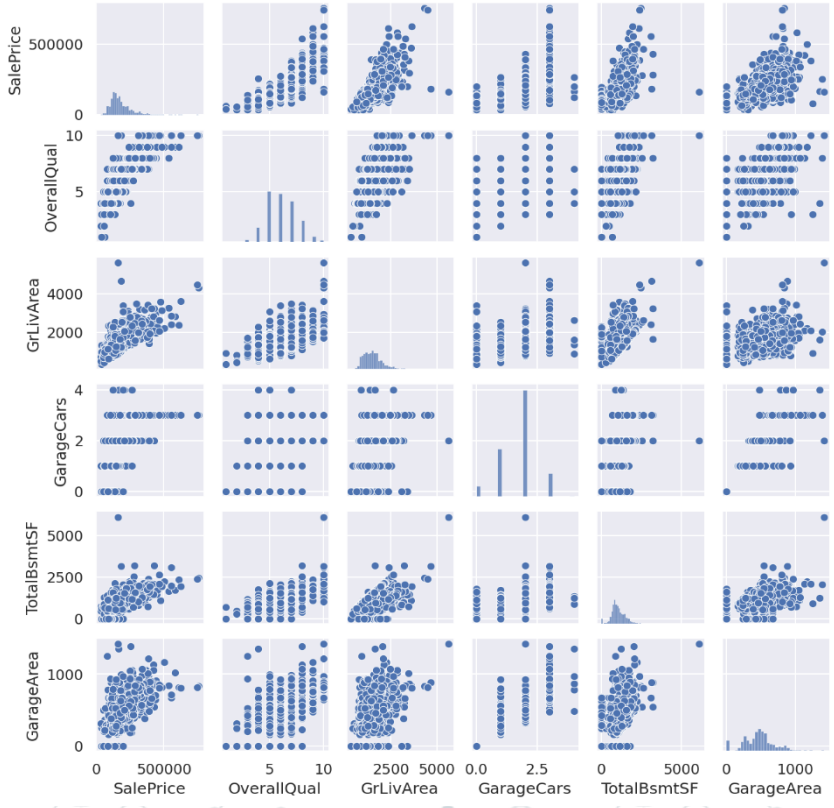
Correlation Matrix

- GarageCars and GarageArea are highly correlated (0.89)
- TotalBsmtSF and 1stFlrSF are also high correlated (0.8)
- OverallQual and GrLivArea are the most correlated features with SalePrice

Correlation matrix of high correlated features with SalePrice

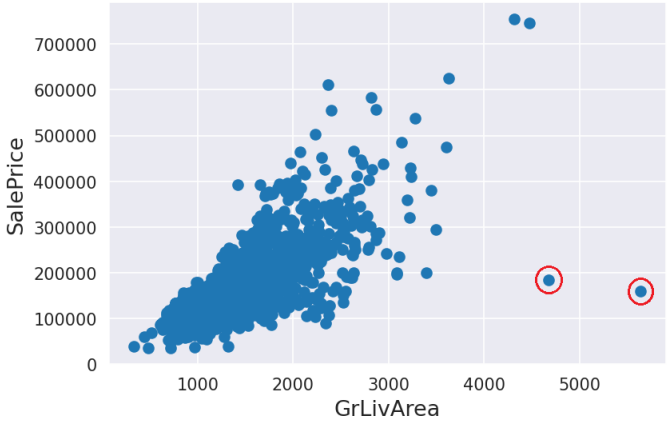


Scatter plots



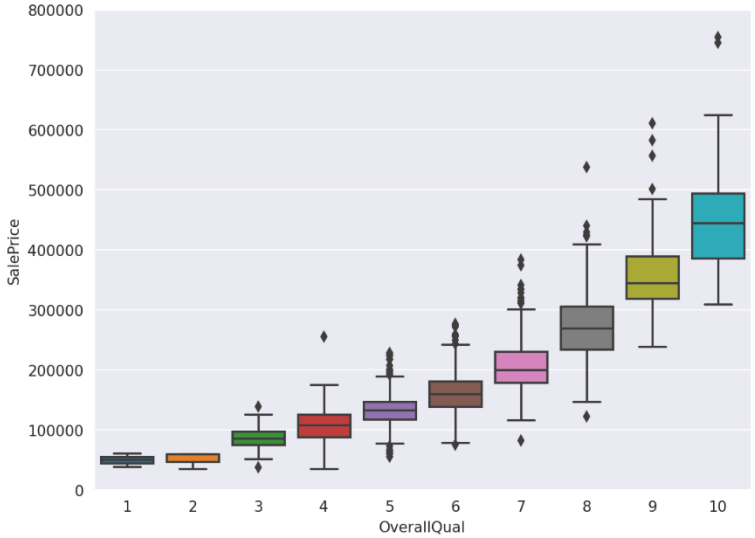
SalePrice VS GrLivArea

- In the scatter plot we see that there is 2 samples with more than 4000 square feet in which the sale prices are dramatically small.
- These samples are removed.



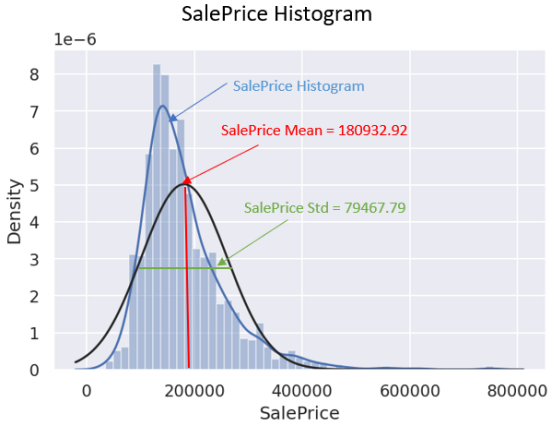
SalePrice VS OverallQual

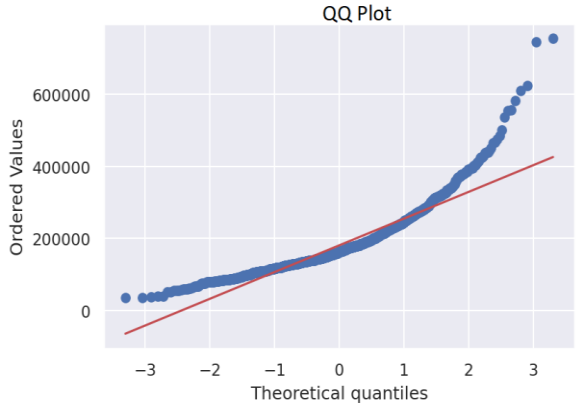
- OverallQual is a categorical feature, range from 1 up to 10.
- From this plot we deduce that the overall quality is indeed highly correlated with the sale price.



Target Variable

- SalePrice Histogram and QQ plot



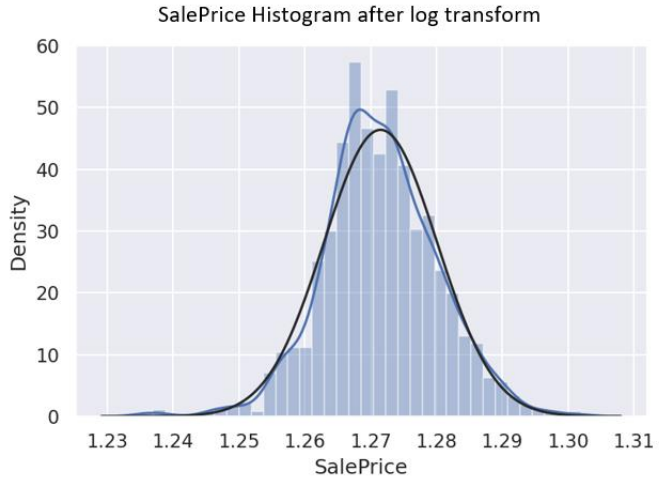


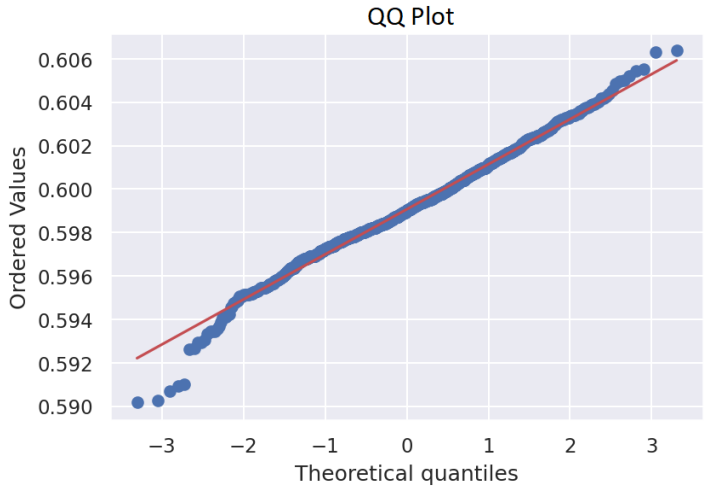
- The SalePrice distribution is positive skewed

2.
Features Engineering

Target Variable Engineering

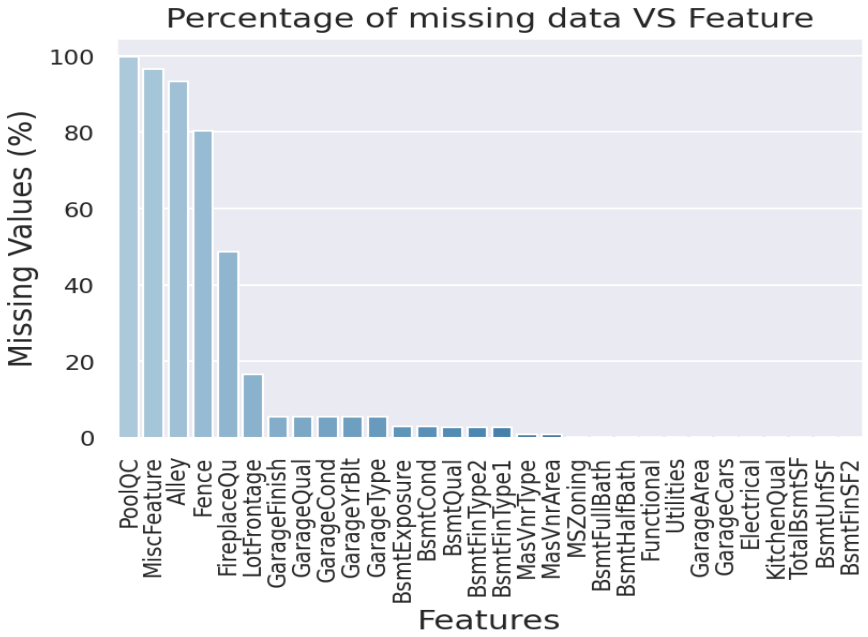
- Applying the transform $x \rightarrow \log(1 + x)$ to fix skewness





Missing Data

- Some of the samples contains missing features values



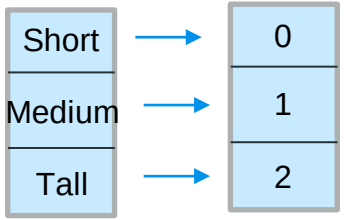
Dealing with Missing Data

- We used some strategies to deal with missing data:
  - Fill missing categorical features with the most common value from all samples.
  - Fill missing categorical features with “None”.
  - Fill missing numerical features with 0.
  - Fill some missing numerical features with the median value of its neighborhood.
  - Fill missing "LotFrontage“ values by take the median value among all houses at the same “neighborhood”.

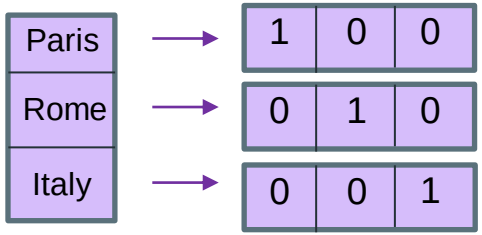
Encoding Categorical Features

- Categorical features can not be processed in regression algorithms.
- Categorical features needs to be converted into a numeric form.
- Solution:
  - Applying label encoding over ordinal features.
  - Applying one hot encoding over nominal features.

Label Encoding

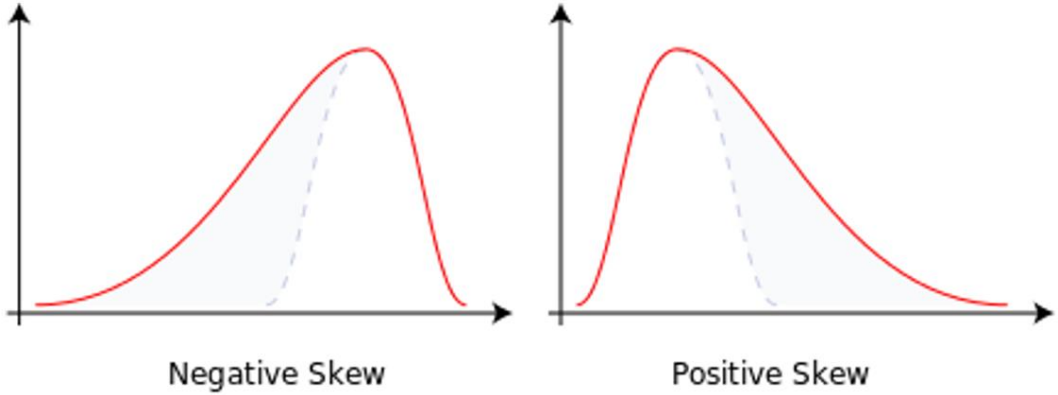


One Hot Encoding



Skewed Features

- Skewness is the degree of distortion from the symmetrical bell curve or the normal curve. (So, a symmetrical distribution will have a skewness of "0").
- There are two types of Skewness: Positive and Negative.



Skewed Features

- Some of the numerical features are highly skewed (positively/negatively)
- Reducing skewness, makes the distribution closer to normal distribution.
- To reduce skewness, we applied Box-Cox transformation over features with $|skewness| > 0.75$.
- $x \to BoxCox(x+1) = \frac{(1+x)^{\lambda}-1}{\lambda}, \lambda \in [-5,5]$

3. Model Architecture and Training

Notations

- Let $D \stackrel{\text{def}}{=} \{(x_i, y_i)\}_{i=1}^{n}$ be the data set.
- Let $x_i \in \mathbb{R}^p$ denote the i'th feature vector.
- Let $y_i \in \mathbb{R}$ denote the i'th target value.

Kernel Ridge Regression (KRR)

- Find $w \in \mathbb{R}^d$ which minimizes the cost function - $\sum_{i=1}^{n}\left(y^{(i)} - w^T\phi(x^{(i)})\right)^2 + \alpha \cdot w^T w$
- ϕ: $\mathbb{R}^p \to \mathbb{R}^d$ transforms the data into some high dimensional space.
- $\alpha \in \mathbb{R}_+$ – regularization strength – determine regularization strength to reduce overfitting
- Using the kernel trick for prediction, for given new $\tilde{x} \in \mathbb{R}^p$, $y_{pred} = k^T \cdot (K + \alpha I)^{-1} \cdot y$
- $\kappa \in \mathbb{R}^{p \times p}$, $\mathrm{k} \in \mathbb{R}^p$, where $[\kappa]_{i,j} = K(x_i, x_j)$, $[k]_i = K(\tilde{x}, x_i)$
- $K(\cdot,\cdot)$ is the kernel function.

Lasso Regression

- Lasso regression is a standard regularization problem with l_1 regularization term.
- The cost function: $\frac{1}{\mathrm{n}}\sum_{i=1}^{n}\left(y^{(i)} - w^T x^{(i)}\right)^2 + \alpha \cdot \left|\left|w\right|\right|_1$
- $\alpha \in \mathbb{R}_+$ – regularization strength – determine regularization strength to reduce overfitting
- Lasso known as effectively reducing the number of features upon which the given solution is dependent

Elastic-Net

- Elastic-Net is a linear regression model trained with both l_1 and l_2 norm regularization.
- The cost function: $\frac{1}{n}\sum_{i=1}^{n}\left(y^{(i)} - w^T x^{(i)}\right)^2 + \alpha \cdot \rho\left|\left|w\right|\right|_1 + \alpha \cdot \frac{1-\rho}{2}\left|\left|w\right|\right|_2$
- $\alpha \in \mathbb{R}_+$ – regularization strength – determine regularization strength.
- $\rho \in [0,1]$ – determine the trade-off between Ridge regularization term and Lasso regularization term.
- Elastic-Net allows to enjoy the advantages of both Ridge and Lasso regularization.

Support Vector Regression - SVR

- A version of SVM for regression
- The objective: $\min_{w,b,\zeta,\zeta^*} \frac{1}{2}||w||_2^2 + C \sum_{i=1}^{n} (\zeta_i + \zeta_i^*)$
- The constraints:

$$y_i - w^T \phi(x_i) - b \leq \varepsilon + \zeta_i$$

$$w^T \phi(x_i) + b - y_i \leq \varepsilon + \zeta_i^*$$

$$\zeta_i, \zeta_i^* \geq 0, i = 1, \ldots, n$$

- The free parameters in the model are C, γ , ϵ

C - regularization parameter.

γ - inversely proportional to σ , where σ is the std of the RBF kernel.

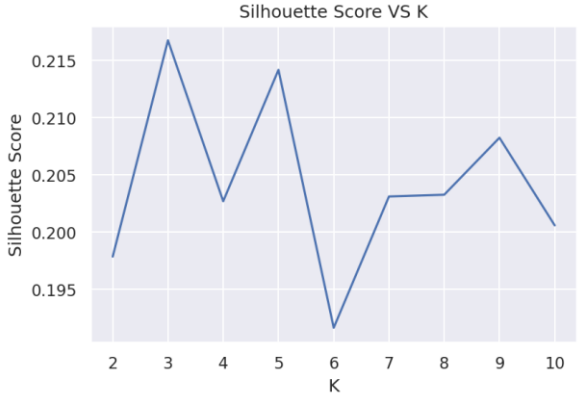
ε – define a margin of tolerance where no penalty is given to errors.

Stacking Models

- Stacking model is stacking the output of some basic regression models and use a regressor (Meta model) to compute the final prediction.
- stacking allows to use the strength of each individual estimator by using their output as input of a final estimator.
- Our base models are:
  - Lasso Regression with $\alpha = 0.0005$
  - Elastic-Net with $\alpha = 0.0005, \rho = 0.9$
  - SVR with $C = 20, \epsilon = 0.008, \gamma = 3 \cdot 10^{-4}$
- The meta model:
  - KRR with $\alpha = 0.6$ and the kernel function $K(x_i, x_j) = (x_i^T x_j + 2.5)^2$
- To generalize and avoid over-fitting, the meta model is trained using 5-fold cross-validated predictions of the base estimators .

K-Means

- K-Means is a clustering method, which used to split a given data set into K clusters
- As the data set is large, it might be a good idea to split it into K clusters, and to train a model for each cluster separately.
- K-Means split the space into smaller regions where interactions may be more manageable.
- To find what is the best K we calculated the silhouette score for $K = 2,3, \dots, 10$.



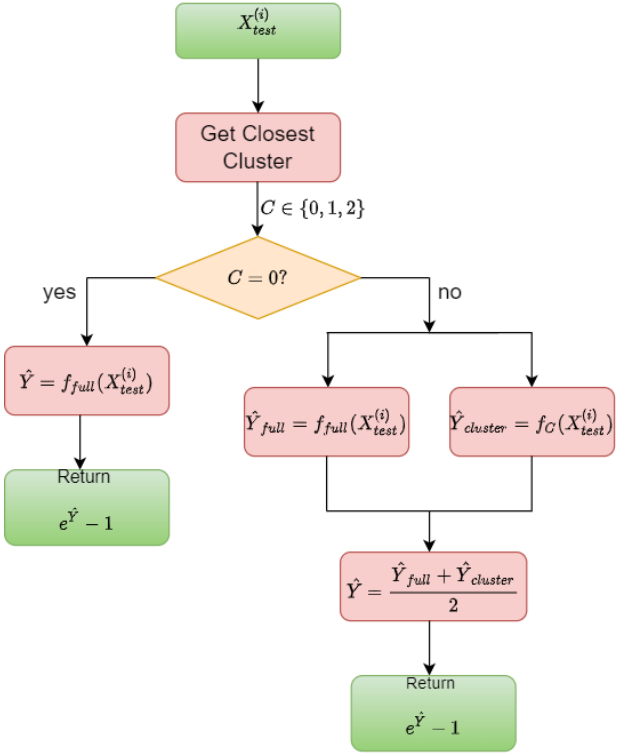
K-Means – cont'

- The best silhouette score has achieved with $K = 3$
- We split the data into 3 sets using K-Means with $K = 3$
- We see that most of the training samples are associated with clusters 1, or 2.
- The idea is to fit a stack model for any cluster
- As less than 100 samples are associated with cluster 0, we will not fit a model for this cluster.

Algorithm Description

1. Apply K-Means algorithm with $K = 3$ over the training set
2. Fit a stacking model as described earlier for clusters 1 (f_1) , and 2 (f_2) independently.
3. Fit a stack model (f_{full}) using the whole training data set.
4. We must remember that we used log transform over the target values before predictions. Hence, after the prediction we must apply the inverse transform, i.e., $Y \rightarrow expm1(Y) = e^Y - 1$
5. For any sample $X_{test}^{(i)}$:
   - Get the cluster (C) such that its center is the closest to $X_{test}^{(i)}$
   - If $C = 0$, then Predict the SalePrice using f_{full}, i.e., $\hat{Y} = expm1\left(f_{full}\left(X_{test}^{(i)}\right)\right)$
   - Otherwise ,
     1. Calculate $\hat{Y}_{cluster} = f_C\left(X_{test}^{(i)}\right), C \in \{1,2\}$
     2. Calculate $\hat{Y}_{full} = f_{full}\left(X_{test}^{(i)}\right)$
     3. Predict the SalePrice as $\hat{Y} = expm1\left(\frac{\hat{Y}_{cluster} + \hat{Y}_{full}}{2}\right)$

Algorithm Block Diagram



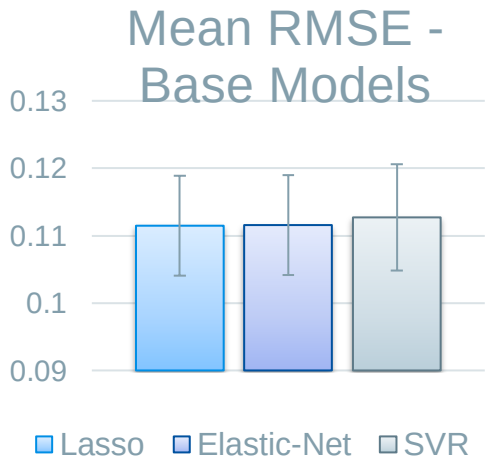
4.
Performance Evaluation

K-fold Cross Validation

- To assess the algorithm performance, we can't simply calculate the loss over the training set.
- In this case, we may perform well over the train examples but poor over the test examples.
- The reason for this is what we call overfitting.
- In this case, the model performance improves over the training because the model begins to memorize the training sample.
- This phenomena can dramatically reduce the results when the model tested over unseen new samples.
- K-fold cross validation is a method used to reduce overfitting.
  The data set is split into K groups. For each group, we take the group as a test set, and train a model with the rest of the training samples. Then, we evaluate the performance of this model over the held-out test set. Finally, we can take the results of all K models and, for example, calculate their mean and the standard deviation.

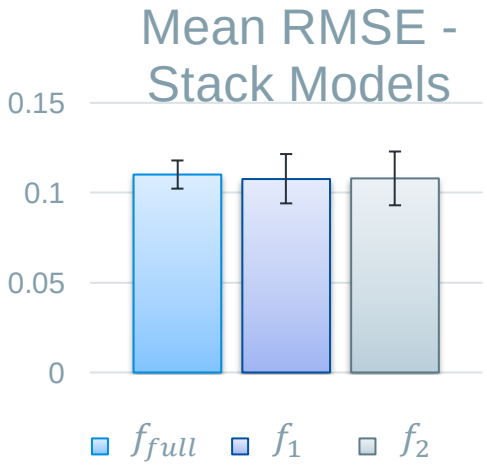
Base Models

- We evaluated the performance of each base model when trained over the whole training set.
- We calculated the Root Mean-Squared Error with 5-fold cross validation over the training set.



Stack Models

- We calculated the Root Mean-Squared Error with 5-fold cross validation over the training set.
- The stack models are f_{full}, f_1, and f_2.



Test Results

Finally, we evaluated the performance of our algorithm over the test set.

The score is 0.12156.

471	Yoel Bokobza & Oded George		0.12156	49	2m

Your Best Entry ↑

Your submission scored 0.12156, which is not an improvement of your best score. Keep trying!