Main objective of the analysis that specifies whether your model will be focused on prediction or interpretation and the benefits that your analysis provides to the business or stakeholders of this data.

Main objective is to explore different types of Classification models to forecast the Overall Quality assessment as given by Ames Housing, with an emphasis on interpretation to identify and understand how the key attributes are contributing to this critical factor that largely drives the final Sale Price of a typical property sold by Ames.

Brief description of the data set you chose, a summary of its attributes, and an outline of what you are trying to accomplish with this analysis.

Given Ames Housing data on 1379 properties sold during 2006-2010, with 80 feature measurements covering the following:

1) construction history (e.g. year built, year remodeled, lot area, lot shape);
2) infrastructure (e.g. #total rooms, garage area, utility, pool);
3) assessments & styles (e.g. overall quality, overall condition, neighborhood, house style);
4) sale information (e.g sale type, sale condition, year/month sold, sale price)

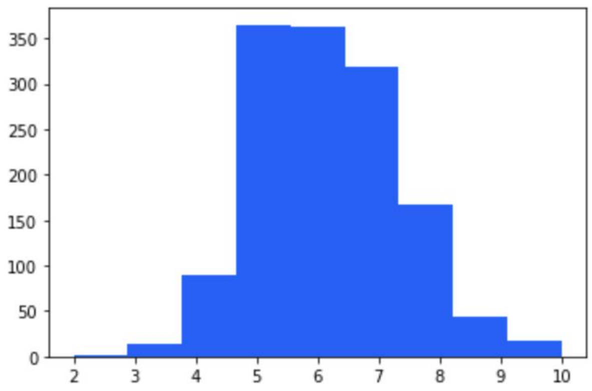
Plan is to focus on the assessment & style attributes and try to identify any relationship between the individual assessment/style attributes and the Overall Quality level.

Brief summary of data exploration and actions taken for data cleaning and feature engineering.

1) Identify target variable: OverallQual
   - take values: integers 1~10
   - median: 6
   - mean: 6.19
   - standard deviation: 1.35

A quick histogram suggests that OverallQual <5 or >7 are highly underrepresented. Given the fact that the business may not have the need to distinguish those extreme cases at that granular a level, it is decided to group OverallQual into 5 classes: <5, 5, 6, 7, >7.

For convenience, those 5 classes then get renamed as 0, 1, 2, 3, 4 accordingly.



2) Focus on 18 assessment & style attributes as contributing features:

- 13 categorical: Neighborhood, HouseStyle, SaleType, BsmtCond, BsmtQual, ExterCond, ExterQual, GarageCond, GarageQual, HeatingQC, KitchenQual, PoolQC, OverallCond
- 5 numerical: SaleCondition, YearBuilt, YearRemodAdd, YrSold, GarageYrBlt

However, it is noticed that the two attributes YearBuilt and GarageYrBlt are highly correlated (with correlation = 0.83). GarageYrBlt most likely will not provide too much additional value to our analysis, as a garage was in most cases built in the same year as the property itself.

The attribute GarageYrBlt hence gets removed from our feature list.

3) Data Cleaning

- missing values: none, no imputation needed.
- outliers: PoolQC is not applicable in most cases and its predictive value could be understated. However, our focus is more on interpretability instead of getting high forecast accuracy at the individual level, and hence no special treatment is needed at this point.

4) Feature Engineering

- skew transformation: 1 numerical feature (OverallCond) is identified to be positively skewed (>0.75). Log transformation is conducted to convert it to be more Normally distributed and will potentially improve the performance of our Classification models.
- scaling: numerical features are on different scales. Standard Scaling is conducted to standardize numerical variables to comparable scales.
- encoding: 13 categorical features are One-Hot Encoded into 75 columns, with the first dummy column for each feature dropped to prevent multi-colinearity.

Summary of training at least three different classifier models, preferably of different nature in explainability and predictability. For example, you can start with a simple logistic regression as a baseline, adding other models or ensemble models. Preferably, all your models use the same training and test splits, or the same cross-validation method.

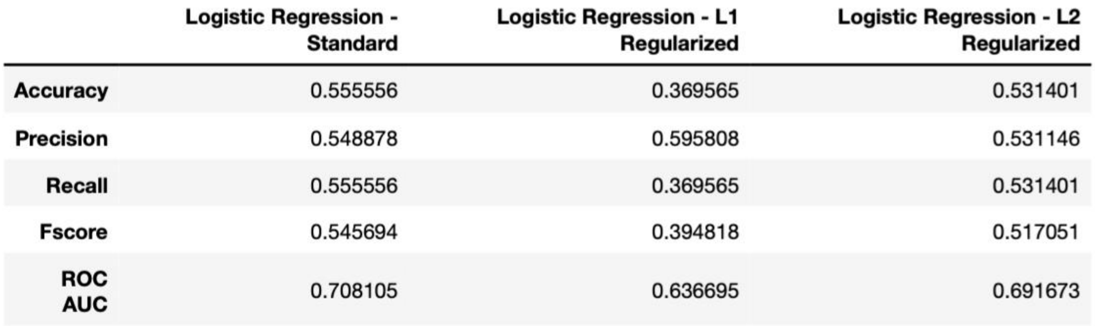
All types of Classification models covered in the course have been trained and tested using the same stratified train-test split that preserves the same class mix as in the provided sample as a whole.

1) Logistic Regression Models

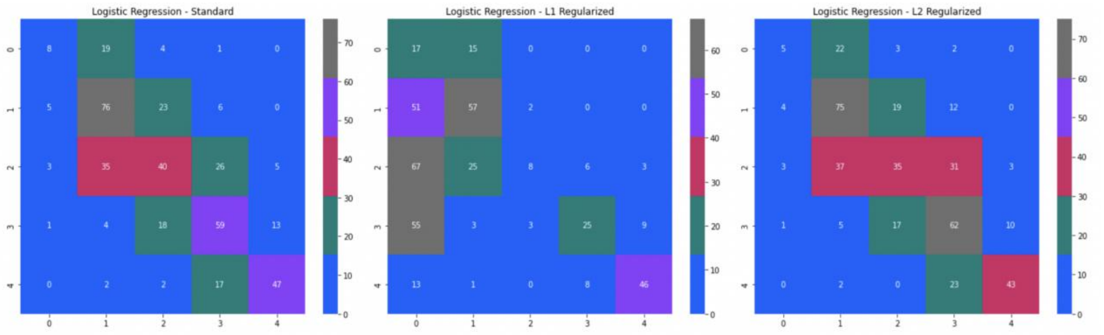
Among the three variations of Logistic Regression models below, the standard model generalizes the best especially in terms of Accuracy (0.56), as well as the more balanced metrics F1 Score (0.55), and ROC AUC (0.71).

The comparison of confusion matrixes suggest the same, where the standard model has data more concentrated along the diagonal.

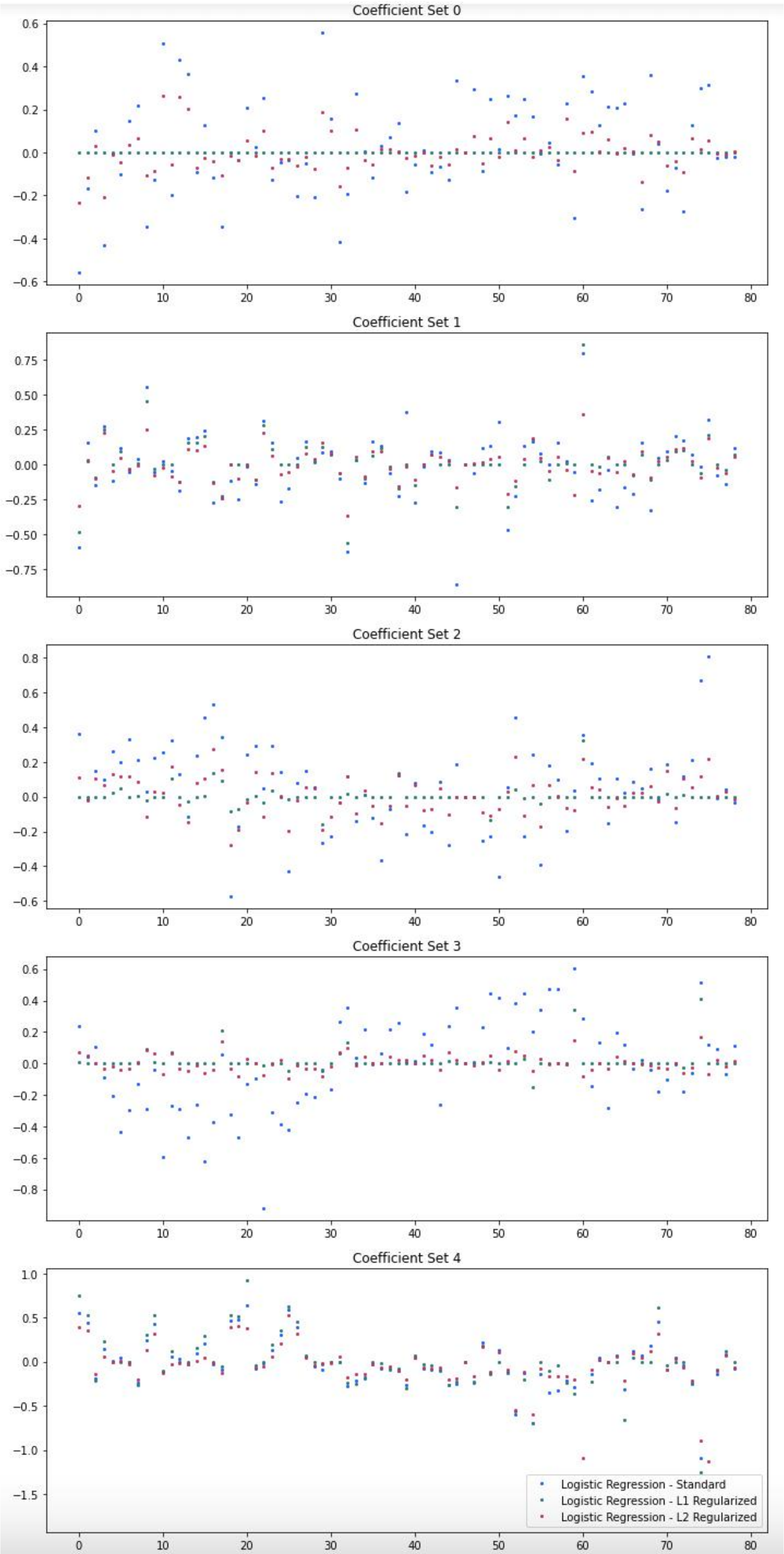
That said, it is noticed that even the best performing Logistic Regression model has a quite low Accuracy score of 0.56. Its confusion matrix shows also its ability to narrow down the OverallQual range for most of the sample that is off by no more than 1 level, but it is rather difficult to narrow it further down and specify the correct level especially for mid-low classes.



	Logistic Regression - Standard	Logistic Regression - L1 Regularized	Logistic Regression - L2 Regularized
Accuracy	0.555556	0.369565	0.531401
Precision	0.548878	0.595808	0.531146
Recall	0.555556	0.369565	0.531401
Fscore	0.545694	0.394818	0.517051
ROC AUC	0.708105	0.636695	0.691673



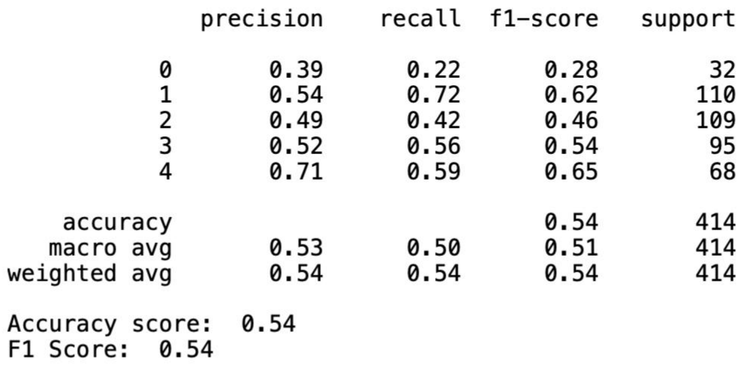
Furthermore, the coefficient sets below for each of the 5 classes imply that it is likely OverallQual is not a direct function of the individual attributes, as the three models don’t seem to align that well and the magnitudes of the coefficients tend to be close to 0 under regularization. There could be other key factors driving the rating of OverallQual that are not captured by our selected contributing attributes. This means the achievable predictive power may be low, and suppressing the weight of our features will only have a counter effect. And this may have resulted in our regularized models performing worse.


Coefficient Set 0
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
0
10
20
30
40
50
60
70
80
Coefficient Set 1
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
0
10
20
30
40
50
60
70
80
Coefficient Set 2
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
0
10
20
30
40
50
60
70
80
Coefficient Set 3
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.8
0
10
20
30
40
50
60
70
80
Coefficient Set 4
1.0
0.5
0.0
−0.5
−1.0
−1.5
0
10
20
30
40
50
60
70
80
Logistic Regression - Standard
Logistic Regression - L1 Regularized
Logistic Regression - L2 Regularized

2) K Nearest Neighbors Model

Based on elbow method, the hyperparameter K = 12 gives the optimal performance in terms of Error Rate (1 - Accuracy Score). Evaluation of a more balanced metric F1-Score suggests the same optimal value of K.

Validation on test set gives an Accuracy Score of 0.54, and an F1 Score of 0.54. Note that a weighted average is used to aggregate error metrics for our multi-class scenario, assuming we're not biased towards/against any particular class in assessment of model performance.

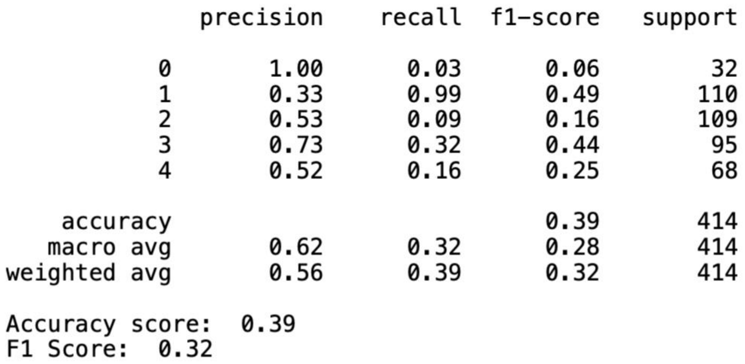


```
              precision    recall  f1-score   support

           0       0.39      0.22      0.28        32
           1       0.54      0.72      0.62       110
           2       0.49      0.42      0.46       109
           3       0.52      0.56      0.54        95
           4       0.71      0.59      0.65        68

    accuracy                           0.54       414
   macro avg       0.53      0.50      0.51       414
weighted avg       0.54      0.54      0.54       414

Accuracy score:  0.54
F1 Score:  0.54
```

3) Support Vector Machine with Gaussian Kernel

Gaussian Kernel is needed is we want to incorporate all of the selected features. A Grid Search with Cross-Validation identifies the optimal values for Kernel parameter: gamma = 10, and for Regularization parameter: c = 2.

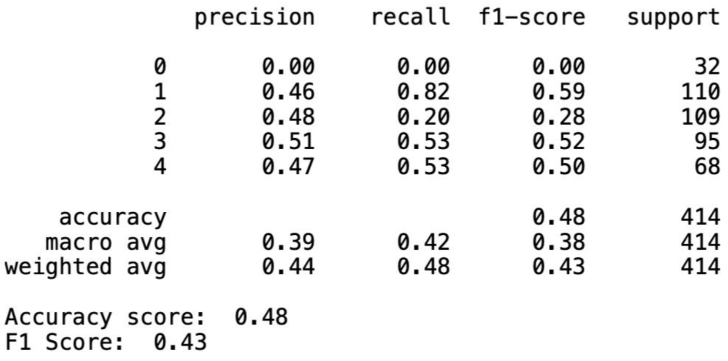
The Gaussian SVM model gives an Accuracy Score of 0.39 and an F1 Score of 0.32 on the test set.



```
              precision    recall  f1-score   support

           0       1.00      0.03      0.06        32
           1       0.33      0.99      0.49       110
           2       0.53      0.09      0.16       109
           3       0.73      0.32      0.44        95
           4       0.52      0.16      0.25        68

    accuracy                           0.39       414
   macro avg       0.62      0.32      0.28       414
weighted avg       0.56      0.39      0.32       414

Accuracy score:  0.39
F1 Score:  0.32
```

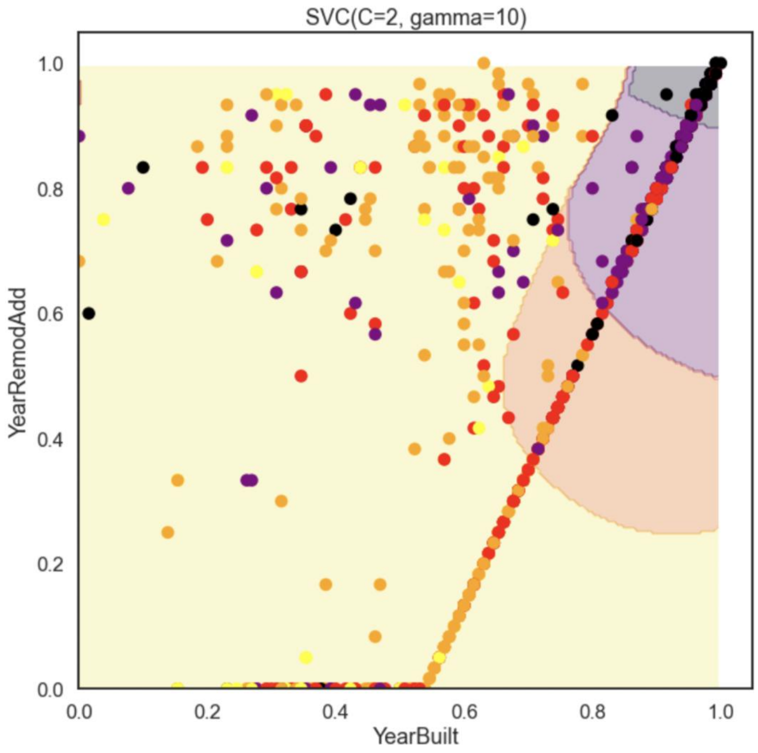
However, the model seems to be overly complicating the decision boundary given the overall weak correlation between our contributing features and the target class OverallQual.

To support this, a simplified Gaussian SVM model using only the top 2 features (YearBuilt, YearRemodAdd - both Min-Max scaled to be in the [0,1] range) most highly correlated with OverallQual has been trained. And the Classification Report on the test set shows better results in terms of Accuracy (0.48) and F1 Score (0.43), as expected.



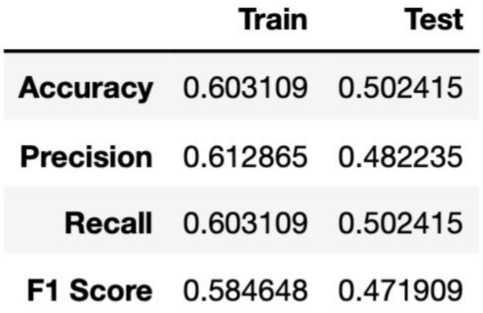
```
              precision    recall  f1-score   support

           0       0.00      0.00      0.00        32
           1       0.46      0.82      0.59       110
           2       0.48      0.20      0.28       109
           3       0.51      0.53      0.52        95
           4       0.47      0.53      0.50        68

    accuracy                           0.48       414
   macro avg       0.39      0.42      0.38       414
weighted avg       0.44      0.48      0.43       414

Accuracy score:  0.48
F1 Score:  0.43
```

With only 2 features, we're able to plot out the decision boundary as shown below. While the limited applicability of YearRemodAdd and the correlation between the 2 features may have added difficulty to the forecast, it can be seen that the decision boundaries do not seem to be able to separate our classes with confidence and there is a rather large "impure" area that is not separable without further over-complication.



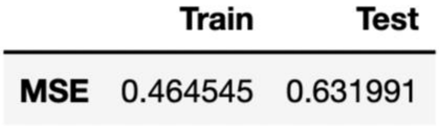
4) <u>Decision Tree Models</u>

Using Grid Search with Cross-Validation, a Classification Tree with 59 nodes and maximum depth of 5 is considered optimal, which renders an Accuracy Score of 0.50 and an F1 Score of 0.47 on the test set.



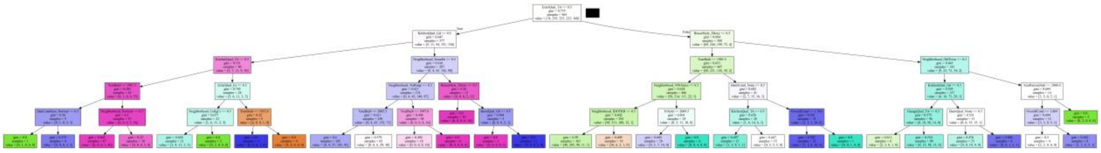
	Train	Test
Accuracy	0.603109	0.502415
Precision	0.612865	0.482235
Recall	0.603109	0.502415
F1 Score	0.584648	0.471909

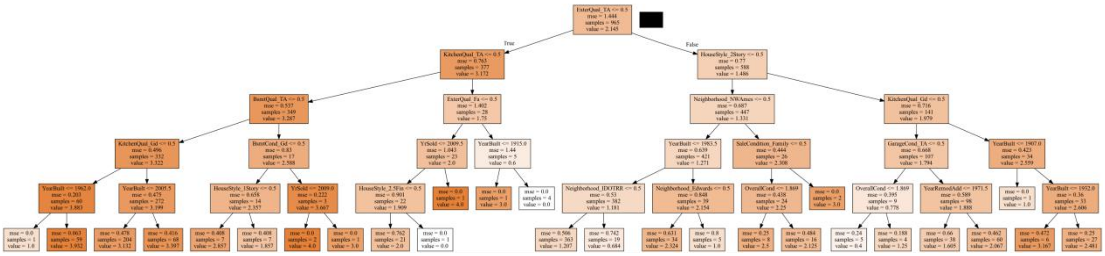
Similarly, an optimal Regression Tree with 53 nodes and maximum depth of 5 has been trained, which renders an MSE (Mean Squared Error) of 0.63 on the test set.



	Train	Test
MSE	0.464545	0.631991

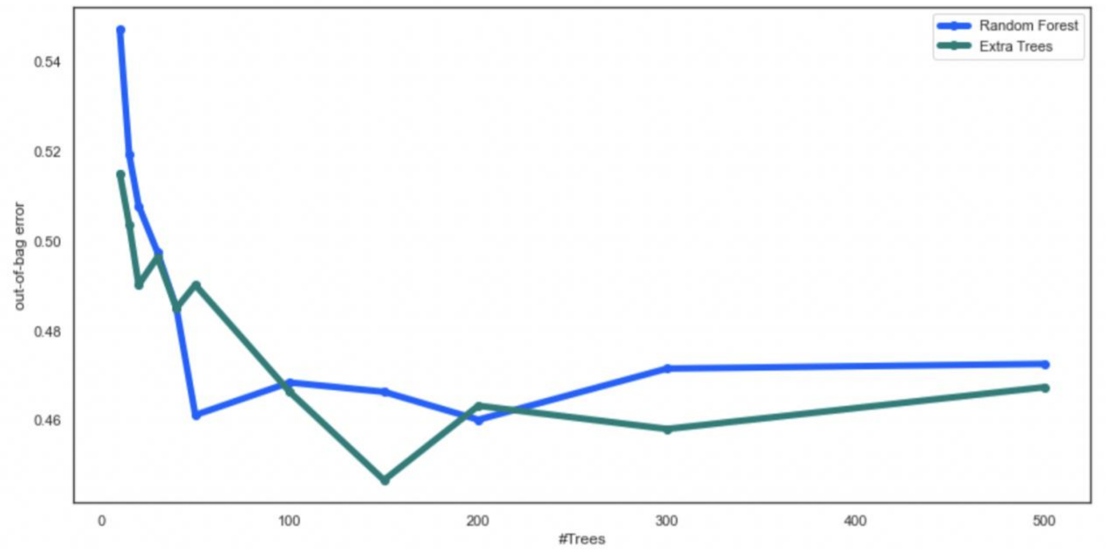
The two trained trees are also plotted out for future deep-dive interpretation:



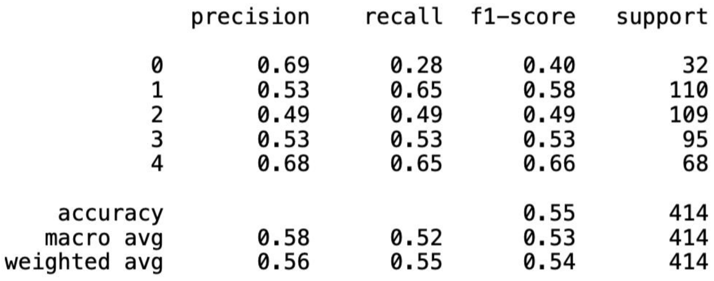

ExterQual_TA <= 0.5 mse = 1.444 samples = 965 value = 2.145
[redacted]
True
False
KitchenQual_TA <= 0.5 mse = 0.763 samples = 377 value = 3.172
HouseStyle_2Story <= 0.5 mse = 0.77 samples = 588 value = 1.486
BsmtQual_TA <= 0.5 mse = 0.537 samples = 349 value = 3.287
ExterQual_Fa <= 0.5 mse = 1.402 samples = 28 value = 1.75
Neighborhood_NWAmes <= 0.5 mse = 0.687 samples = 447 value = 1.331
KitchenQual_Gd <= 0.5 mse = 0.716 samples = 141 value = 1.979
KitchenQual_Gd <= 0.5 mse = 0.496 samples = 332 value = 3.322
BsmtCond_Gd <= 0.5 mse = 0.83 samples = 17 value = 2.588
YrSold <= 2009.5 mse = 1.043 samples = 23 value = 2.0
YearBuilt <= 1915.0 mse = 1.44 samples = 5 value = 0.6
YearBuilt <= 1983.5 mse = 0.639 samples = 421 value = 1.271
SaleCondition_Family <= 0.5 mse = 0.444 samples = 26 value = 2.308
GarageCond_TA <= 0.5 mse = 0.668 samples = 107 value = 1.794
YearBuilt <= 1907.0 mse = 0.423 samples = 34 value = 2.559
YearBuilt <= 1962.0 mse = 0.203 samples = 60 value = 3.883
YearBuilt <= 2005.5 mse = 0.475 samples = 272 value = 3.199
HouseStyle_1Story <= 0.5 mse = 0.658 samples = 14 value = 2.357
YrSold <= 2009.0 mse = 0.222 samples = 3 value = 3.667
HouseStyle_2.5Fin <= 0.5 mse = 0.901 samples = 22 value = 1.909
mse = 0.0 samples = 1 value = 4.0
mse = 0.0 samples = 1 value = 3.0
mse = 0.0 samples = 4 value = 0.0
Neighborhood_IDOTRR <= 0.5 mse = 0.53 samples = 382 value = 1.181
Neighborhood_Edwards <= 0.5 mse = 0.848 samples = 39 value = 2.154
OverallCond <= 1.869 mse = 0.438 samples = 24 value = 2.25
mse = 0.0 samples = 2 value = 3.0
OverallCond <= 1.869 mse = 0.395 samples = 9 value = 0.778
YearRemodAdd <= 1971.5 mse = 0.589 samples = 98 value = 1.888
mse = 0.0 samples = 1 value = 1.0
YearBuilt <= 1932.0 mse = 0.36 samples = 33 value = 2.606
mse = 0.0 samples = 1 value = 1.0
mse = 0.063 samples = 59 value = 3.932
mse = 0.478 samples = 204 value = 3.132
mse = 0.416 samples = 68 value = 3.397
mse = 0.408 samples = 7 value = 2.857
mse = 0.408 samples = 7 value = 1.857
mse = 0.0 samples = 2 value = 4.0
mse = 0.0 samples = 1 value = 3.0
mse = 0.762 samples = 21 value = 2.0
mse = 0.0 samples = 1 value = 0.0
mse = 0.506 samples = 363 value = 1.207
mse = 0.742 samples = 19 value = 0.684
mse = 0.631 samples = 34 value = 2.324
mse = 0.8 samples = 5 value = 1.0
mse = 0.25 samples = 8 value = 2.5
mse = 0.484 samples = 16 value = 2.125
mse = 0.24 samples = 5 value = 0.4
mse = 0.188 samples = 4 value = 1.25
mse = 0.66 samples = 38 value = 1.605
mse = 0.462 samples = 60 value = 2.067
mse = 0.472 samples = 6 value = 3.167
mse = 0.25 samples = 27 value = 2.481

5) <u>Tree-Ensemble</u> <u>Models</u> <u>(Random</u> <u>Forest,</u> <u>Extra</u> <u>Trees,</u> <u>Gradiant</u> <u>Boosting,</u> <u>Adaptive</u> <u>Boosting)</u>

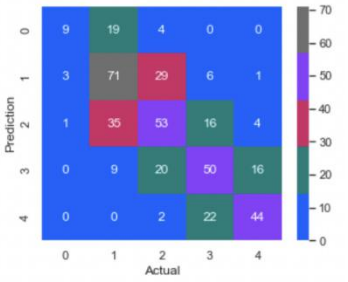
Hyperparameter tuning based on Out-of-Bag Error suggests we use a Random Forest model with 50 tree stumps, and/or an Extra Trees model with 150 trees.

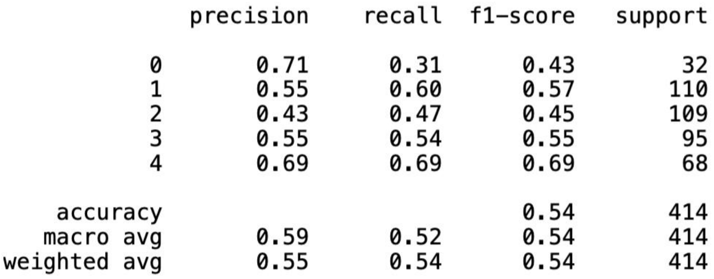


After training on the train set, the Random Forest model gives an Accuracy Score of 0.55 and an F1 Score of 0.54; while the Extra Trees model results in similar score of 0.54 for Accuracy and 0.54 for F1. Details are as below:

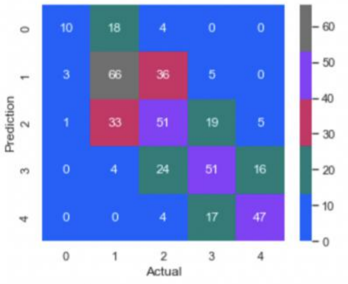


```
              precision    recall  f1-score   support

           0       0.69      0.28      0.40        32
           1       0.53      0.65      0.58       110
           2       0.49      0.49      0.49       109
           3       0.53      0.53      0.53        95
           4       0.68      0.65      0.66        68

    accuracy                           0.55       414
   macro avg       0.58      0.52      0.53       414
weighted avg       0.56      0.55      0.54       414
```

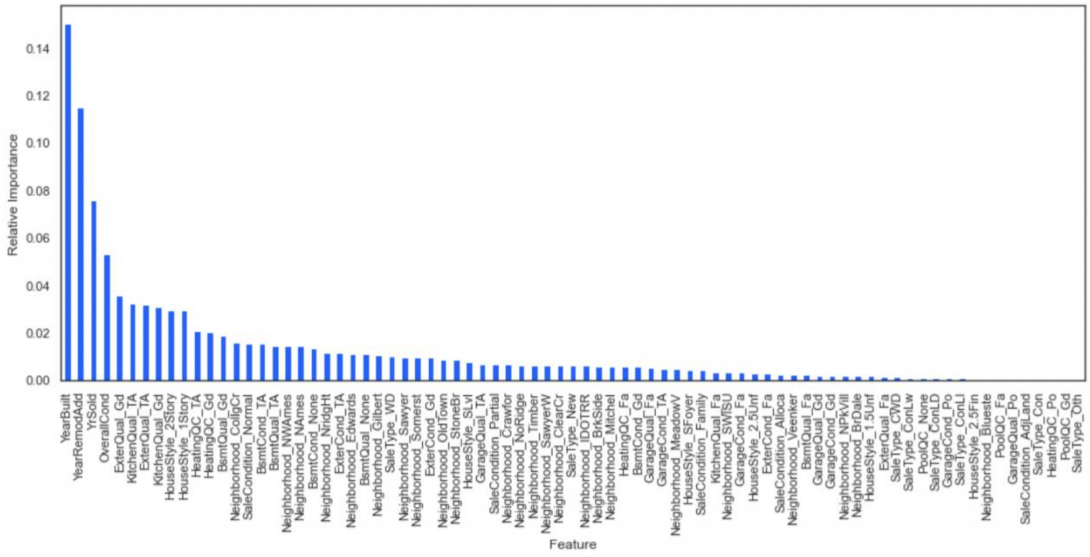




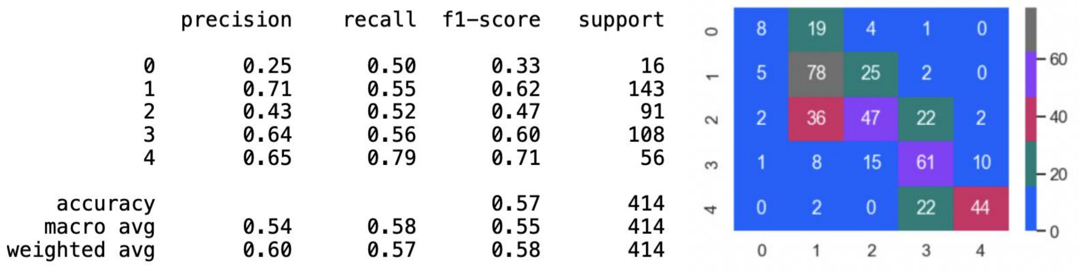
```
              precision    recall  f1-score   support

           0       0.71      0.31      0.43        32
           1       0.55      0.60      0.57       110
           2       0.43      0.47      0.45       109
           3       0.55      0.54      0.55        95
           4       0.69      0.69      0.69        68

    accuracy                           0.54       414
   macro avg       0.59      0.52      0.54       414
weighted avg       0.55      0.54      0.54       414
```



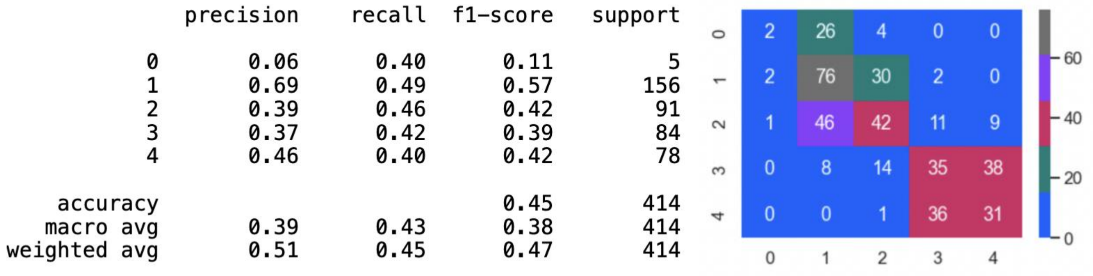
For potential further interpretative analysis, a chart of feature importance is provided below:



Similarly a Grid Search with Cross-Validation suggests we use a Gradient Boost model with 20 trees, maximum features of 4 and 0.5 subsample. After training, it scores 0.57 on Accuracy and 0.58 on F1. Meanwhile an Adaptive Boost model with 100 trees and 0.1 learning rate has also been trained, which scores 0.45 on Accuracy and 0.47 on F1.



```
              precision    recall  f1-score   support

           0       0.25      0.50      0.33        16
           1       0.71      0.55      0.62       143
           2       0.43      0.52      0.47        91
           3       0.64      0.56      0.60       108
           4       0.65      0.79      0.71        56

    accuracy                           0.57       414
   macro avg       0.54      0.58      0.55       414
weighted avg       0.60      0.57      0.58       414
```

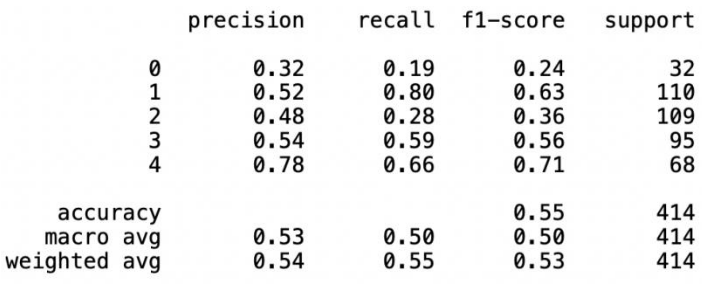


```
              precision    recall  f1-score   support

           0       0.06      0.40      0.11         5
           1       0.69      0.49      0.57       156
           2       0.39      0.46      0.42        91
           3       0.37      0.42      0.39        84
           4       0.46      0.40      0.42        78

    accuracy                           0.45       414
   macro avg       0.39      0.43      0.38       414
weighted avg       0.51      0.45      0.47       414
```

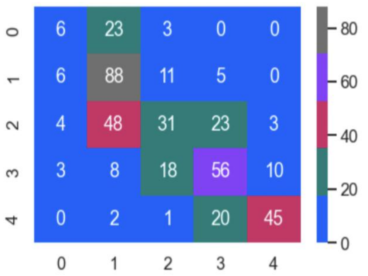
It is as expected that Adaptive Boost model performs much worse than Gradient Boost model in our case, as Gradient Boost is more robust to outliers by using a logarithmic-based Deviance as loss function. Knowing that our selected features can only tell so much about OverallQual, trying to read too much into the patterns in the train data would only be a pursuit of false precision.

6) Stacked Models (Distance-Based, Tree-Based)

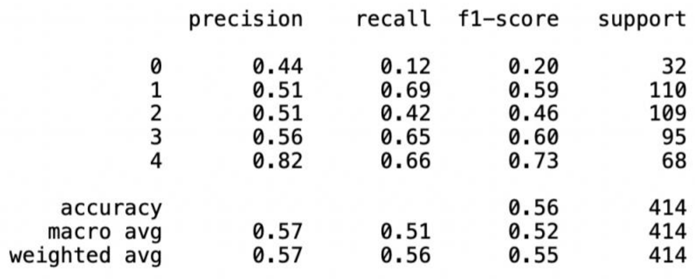
The above trained distance-based classifiers (Standard Logistic Regression model, L1-Regularized Logistic Regression model, L2-Regularized Logistic Regression model, K Nearest Neighbors model, and L2-Regularized Gaussian Support Vector Machine) can be stacked to form a Distance-Based Stacked model. Using hard voting for aggregation, it results in an Accuracy Score of 0.55 and an F1 Score of 0.53 on the test set.



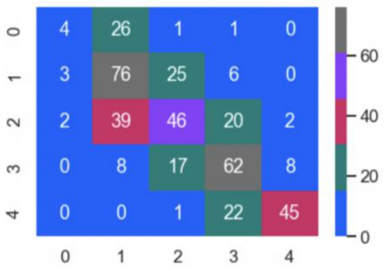
```
              precision    recall  f1-score   support

           0       0.32      0.19      0.24        32
           1       0.52      0.80      0.63       110
           2       0.48      0.28      0.36       109
           3       0.54      0.59      0.56        95
           4       0.78      0.66      0.71        68

    accuracy                           0.55       414
   macro avg       0.53      0.50      0.50       414
weighted avg       0.54      0.55      0.53       414
```



On the other hand, the above trained tree-based classifiers (Decision Tree model, Random Forest model, Extra Trees model, Gradient Boost model, and Adaptive Boost model) can also be stacked to form a Tree-Based Stacked model. Using soft voting for aggregation, it results in an Accuracy Score of 0.56 and an F1 Score of 0.55 on the test set.

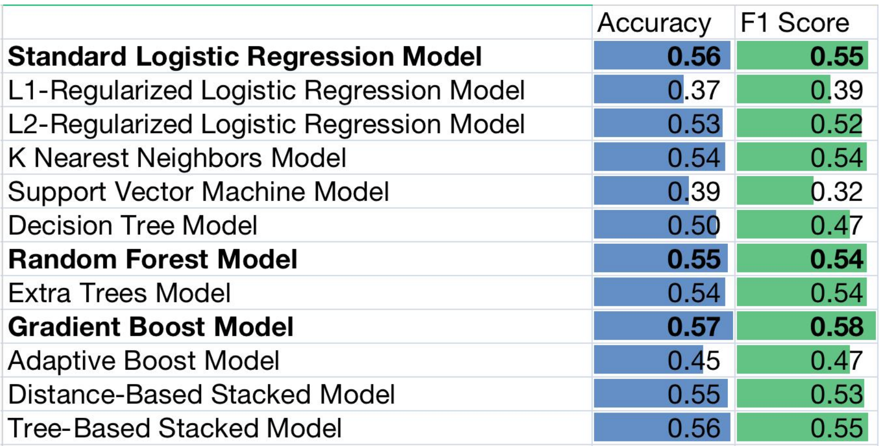


```
              precision    recall  f1-score   support

           0       0.44      0.12      0.20        32
           1       0.51      0.69      0.59       110
           2       0.51      0.42      0.46       109
           3       0.56      0.65      0.60        95
           4       0.82      0.66      0.73        68

    accuracy                           0.56       414
   macro avg       0.57      0.51      0.52       414
weighted avg       0.57      0.56      0.55       414
```



A paragraph explaining which of your classifier models you recommend as a final model that best fits your needs in terms of accuracy and explainability.

In summary, here's how our models perform on the test set:



	Accuracy	F1 Score
Standard Logistic Regression Model	**0.56**	**0.55**
L1-Regularized Logistic Regression Model	0.37	0.39
L2-Regularized Logistic Regression Model	0.53	0.52
K Nearest Neighbors Model	0.54	0.54
Support Vector Machine Model	0.39	0.32
Decision Tree Model	0.50	0.47
Random Forest Model	**0.55**	**0.54**
Extra Trees Model	0.54	0.54
Gradient Boost Model	**0.57**	**0.58**
Adaptive Boost Model	0.45	0.47
Distance-Based Stacked Model	0.55	0.53
Tree-Based Stacked Model	0.56	0.55

Gradient Boost model has the best performance among all trained models, in terms of both Accuracy and F1 Score. While the Boosting method can in general be computationally expensive, in our case we require only 20 trees and a maximum of 4 features and the algorithm does not take long to run. That said, if data magnitude increases largely, Standard Logistic Regression model may serve the purpose just well and much more efficient. These two models will hence be recommended as the final model(s) if we'd like to try and predict OverallQual the best we can.

However, even the best model has a rather poor performance in our case and a couple percentage points may not be worth as much as the interpretative value that we can draw from our Random Forest Model. This is especially true as the main purpose of the exercise is to obtain useful insights on which contributing attributes may be driving the rating of OverallQual and if we can better understand the relationship between them.

For that matter, Random Forest model is recommended as our final model for this exercise, while the Decision Tree model could serve as a secondary supporting model for additional insights.

Summary of Key Findings and Insights, which walks your reader through the main drivers of your model and insights from your data derived from your classifier model.

According to the feature importance chart output from the Random Forest model, recency (YearBlt, YearRemodAdd, YrSold) seems to be an importance factor contributing to a high rating of OverallQual. Next on list are overall condition (OverallCond), quality of Exterior (ExterQual) and Kitchen (KitchenQual), and style of house (HouseStyle). On the contrary, consumers don't relate much of sale info (SaleType, SaleCondition) to the assessment of the overall quality of the property.

The fact that none of our Classification models works well enough to pin point the exact class suggests that the lines between consecutive OverallQual ratings could in fact be blurry and vary from consumer to consumer due to drastically different personal preferences. Our Decision Tree model supports this idea, as unlike Random Forest model, it applies a greedy search to fit the pattern as closely as possible with one tree. And the order of the feature nodes from root to leaves is not so perfectly aligned with the order of feature importance revealed by the Random Forest model. That is, our consumers may be assessing the property as a whole rather than a pure addition of its parts, in that one part extremely to their liking could offset another part being close to "unacceptable", and that different consumers may want to assign different weights to the different part they rate as good and bad..

Suggestions for next steps in analyzing this data, which may include suggesting revisiting this model after adding specific data features that may help you achieve a better explanation or a better prediction.

A key finding is none of our Classification models seem to predict OverallQual well using the individual assessment & style attributes given in the data set, and how well we are able to understand the underlying drivers is hence largely limited.

Below are a few ideas to try and bridge this gap:

1) Conduct focus group research to gather more insights on consumers' assessment process, and try

to improve our selection of contributing attributes and/or adding potential feature interactions.

2) Convert assessment attributes into ordinal numerical values, instead of seeing “Good”, “Excellent” etc each as an independent category to be One-Hot Encoded before modeling.

3) Apply regression models to forecast target variable OverallQual as a numeric value instead of categorical classes.

4) Potentially pick the top performing models to stack together for a Meta Classifier.

5) Obtain more recent property data for 2010 onwards, and could use some due diligence work to look into the “recency” factor for more insights.