# COMP5310 Project Stage 2 Report

# 

# 

# 

# 

# 

# 

# 

# Group Name: Activity 4 (Tommy) - Group 5

# 

# Group Members:

# 

# 

# 1. hcao0570, 530541978

# 2. lwan0620, 520502178

# 3. siwu9858, 530236638

# 

# 1. Setup

## 1.1 Topic and research question

Our research is focusing on deeply analyzing the furniture retailer sales data to identify the key factors which can bring influence to the store sales performance. The research question can be specified as: what are the factors that can influence the store sales amount, and how can we predict future sales level based on the given dataset? The question itself has significant meaning in the field of retailing management and operation optimization. By accurately analyzing key factors in the sales data including product categories, discounts and regions, we can build up predictive models to forecast the sales performance in different cases. So that we can solve these problems properly. And for various stakeholders like store managers and market analysts, solving mentioned problems can not only help them get more profit, but also strengthen their market competitiveness.

## 1.2 Dataset

The dataset we select is designed for the furniture retailing store. It’s mainly used for forecasting furniture sales in the future year to prevent inventory problems like excess stock or insufficient stock. The prediction can ensure the best optimization of customer experience, prevent loss and maintain the sustainability of the store. It’s provided by Kaggle user Zahra’ Alaa Tag el-Dein, and can be accessed via following link to Kaggle:https://www.kaggle.com/datasets/zahraaalaatageldein/sales-for-furniture-store/data. The geographical coverage of this dataset is within the United States. Also, it belongs to Mentorness Mentorness Machine Learning Internship program from the dataset metadata contents. The dataset after pre-processing includes 2121 records which contain 18 attributes like: Order ID, Order Date, Ship Date, Ship Mode, Customer ID, Customer Name, Segment, City, State, Region, Product ID, Sub-category, Product Name, Sales, Quantity, Discount, Profit. The time span and geographical coverage of these data as well as detailed sales description, show the richness and diversity of this dataset. Based on the information above, we can know the challenges and biases as follows: **Product category bias:** The dataset might have more sales data records in certain categories. This imbalance probably causes the model to have a more accurate sales forecast in those categories while more imprecise in other categories’ analysis. **The impact of discount and profit:** The dataset contains attributes of discount and profit, which may impact explanations towards sales data. For example, a high discount is likely to boost sales but decrease profit. Thus, our models need to properly handle the interaction of these attributes. Therefore, the product category bias is required to decrease its impact by techniques like stratified sampling, so that we can ensure the model generalization. Moreover, when processing attributes like discount and profit, it’s essential to take interaction effects into consideration to fully understand how sales promotion affects the store sale.

## 1.3 Modeling agreements

For our dataset, we select “Sales” as the prediction attribute because it’s required for the research question. And we select F1-score and precision-recall curves as evaluation indexes. F1-score is the harmonic mean of precision and recall. The calculation function is: [1]. In the case of predicting sales, it can be categorized as “high sales” and “low sales”. Thus, it’s a vital measure because it balances the ability of identifying high sales and accuracy of correctly predicting high sales. Especially when sales data is probably apt more to a certain category, only using precision or recall can cause misleading. Precision-recall curves is a chart that shows the model precision and recall relation in different threshold sets. In our sales prediction, the balance of precision and recall is essential particularly when data categories are unbalanced [2]. By analyzing this measure, we can better understand the precision change in different recall levels. This can help us select a suitable threshold to optimize the actual impact of the model to business strategy.

Decision tree(Lei Wang):

# 1. Predictive model:

## 1.1. Model description:

## Decision tree is the technique I used. Decision tree is a widely used model which can both work with regression and classification tasks. It can divide attributes in a dataset step by step, and create a structure of tree on which each spot represents a test of characteristic attributes, and each branch represents a result of the test, each leaf represents a class or numeric data. During the test process, starting from the root spot, the test goes through the tree based on the characteristic attributes, and finally reaches the leaf spot, getting the classification or regression results.[5]

**The main assumptions of the decision tree** are:

1. all attributes are independent with each other, each of them can’t be affected by the other attribute, in our dataset, I think the sales attribute is completely independent from the attribute region and furniture category;
2. dataset includes enough information for the test, so that the tree can grow high enough, the dataset has over 2000 samples, which is not very large but enough for decision tree;
3. samples are evenly distributed, making the decision tree able to classify the data, the data was collected in different time periods, which makes it even.

**Strength of decision tree**:

1. decision tree is easily comprehensive, which makes it able to give clear path of solution, and makes the model training easier for me;
2. decision tree can deal with multiple attributes, including continues and categorical data, our research is categorical;
3. the decision tree doesn’t care about absent data, which makes it more useful, luckily, our dataset has been cleaned of absent data.

**Limitation of decision tree**:

1. decision tree may get overfit easily, which did happen in my training;
2. decision tree is very sensitive to outliers, but our data has cleaned of outliers;
3. The decision tree has difficulties with high-dimensional data, but our dataset is not high-dimensional.

## 1.2. Model algorithm:

**The principle of decision tree:**

1. Feature selection: decision tree algorithm will select the top feature in the process, and conduct the prediction based on it, after many generations of nodes, it will have many leaves that can give a good prediction.
2. Splitting node: all nodes in the decision tree are split from each other, thus the process of prediction continues until one condition is met, in this way, the classification can be very accurate.
3. Construction of tree: the structure can target the input and output, with the relationship between input and output, the prediction can be done.

**Hyperparameters:**

1. max depth: the highest depth of tree, limits the possibility of overfitting;
2. minimum sample splits: minimum number of samples to split a node;
3. minimum sample leaf: minimum number of leaves on tree, limits the number of categories

**Potential variations:**

random forest, gradient boosting trees, tree boosting, CART

**Key steps:**

1.load the csv file to get data;2.preprocess the data; 3.split the training, testing and validation dataset;4.create the decision tree;5.fit the tree with dataset; 6.get prediction; 7.evaluate the prediction; 8.optimize the model9.get the result of the model

**Advancements:**1**.**the combination of decision tree with deep learning can deal with larger dataset and has better performance;2.Pruning technology can reduce some branches or nodes that are not compulsory, and boost the performance of decision trees.

## 1.3. Model development:

**Process:**

1. **preprocessing:**

After loading the dataset with the pandas package, I have to process the data because since our project set classification as goal, I can’t use the continuous data in column ‘Sales’ to make a classification prediction.

So I labeled the sales data into three kinds: low sales, medium sales and high sales. The sales data was cut and divided evenly, so that the prediction can have the target outcome: low sales, medium sales, high sales.

Then, I processed the input columns: region and sub-category(which is the category of furniture) using the labelencoder function imported sklearn.

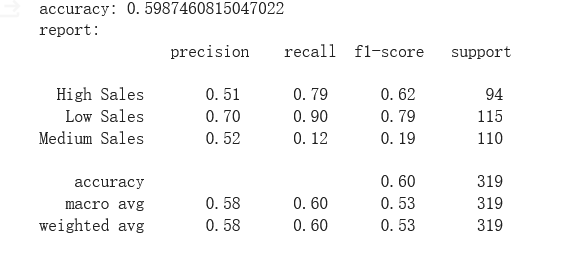
Actually I used PCA to process but found it not useful with these attributes. I also used feature engineering, created one more feature: sales per furniture(Salesbycategory), but the result was overfitting, so I quit it.

1. **splitting**

Then I split the dataset by 7:3 into a training set and a test set. During the process, I used the train test split function from sklearn. Then, I used it again to split the test set into a test set and a validation set by 5:5.

1. **modeling**

I used the decision tree classifier from sklearn.tree, fit it with the training set, and get the desired results: accuracy, F1 score.

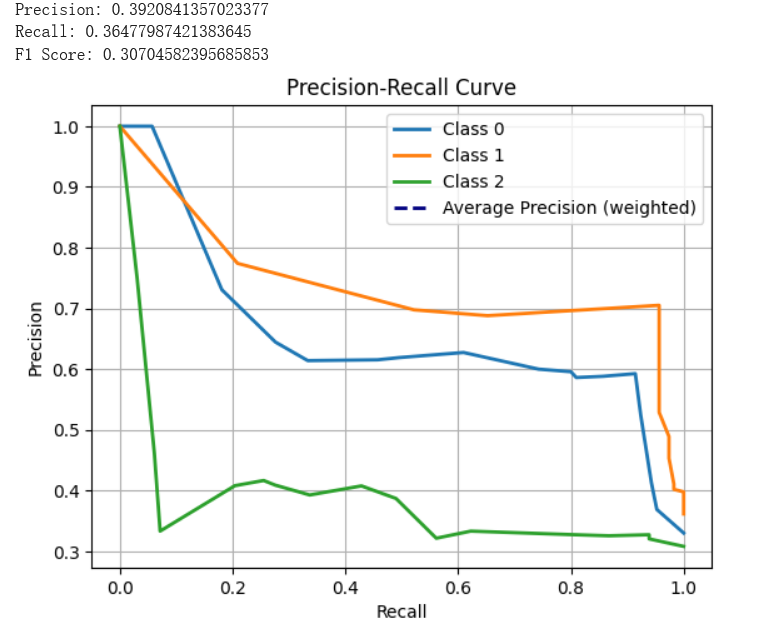
The result is as follows:

# 2. Model evaluation and optimization

## 2.1. Model evaluation:

1. **evaluation result:**

One important thing of evaluation is that the test set must be converted into 2-dimensional data so that it can map the prediction data and create the precision-recall curve.

Note: 0: High Sales1: Low Sales2: Medium Sales

You can see the accuracy, f1 score and the precision-recall curve here, which shows the prediction of sales with customers’ region and furniture choice has about 60% accuracy, which can actually help the worker of the store to sell furniture.

In the ideal circumstance, the worker in store can just ask customers about their hometown and types of furniture they wanted, then decide whether sell the expensive furniture to him/her.

1. **potential improvement:**

It is obvious that the dataset doesn’t have problems of imbalance or noise, because it has been cleaned. I have interrupted the sales into 3 categories, which makes it more comprehensive.

The model hasn’t gone through fine-tune yet, so maybe some hyperparameters can greatly affect the result.

## 2.2. Model optimization:

1. **Grid search:**

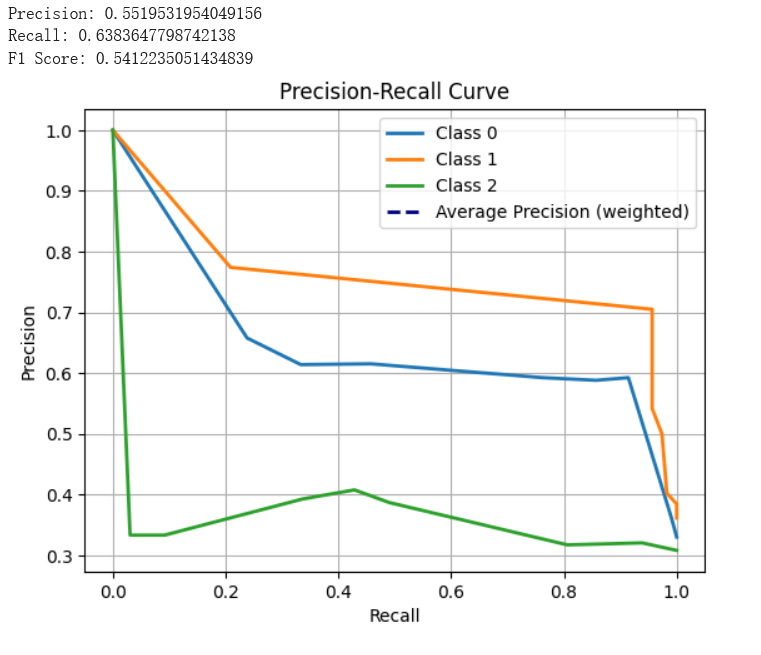
I used grid search from sklearn, and tested all hyperparameters for decision tree, because it only used 2000 pieces of data, which will not cost a lot of time for fine tune.

The final result of hyperparameter is here:

{'criterion': 'entropy', 'max\_depth': 3, 'max\_features': 'sqrt', 'min\_samples\_leaf': 2, 'min\_samples\_split': 10}

Accuracy: 0.60062893081761

1. **Evaluation after grid search:**

Here is the result of evaluation after grid search, which shows that the change of hyperparameter can improve the performance of the model. making the prediction of sales more precise.

Random Forest(siwu9858)

# 1. Predictive model

## 1.1. Model description:

Random forest is a machine learning algorithm based on ensemble learning, which consists of multiple decision trees and obtains the final prediction result by averaging or voting the prediction results of each decision tree.

**Here is a assumptions of the random forest model for the furniture sales dataset:**

• Feature independence Assumption: Random forests assume that features are independent of each other. In the furniture sales dataset, it is assumed that the order date, sales volume, discount and other features are independent of each other.

• Decision Tree Assumption: Random forests assume that each decision tree is a weak learner, that is, their predictive performance is slightly better than random guess. In the furniture sales dataset, each decision tree is assumed to be able to make reasonable predictions about sales trends.

**Strength:**

• Adapt to complex relationships: Random forests are able to effectively capture complex nonlinear relationships and interactions in furniture sales datasets, including complex relationships between order dates, sales, discounts, and other factors.

Reduce the risk of overfitting: Random forests reduce the risk of overfitting and improve the generalization ability of the model by randomly selecting a subset of features and samples to train on.

• Provide feature importance estimates: Random forests are able to provide importance estimates for each feature, helping to identify key factors that affect sales.

**Limitations:**

• Black box model: Random forest is a black box model and it is difficult to explain the specific decision process of each decision tree. This can limit a deep understanding of the mechanisms behind the sales data.

• Computational resource consumption: Random forests consist of multiple decision trees and therefore require more computational resources for training and prediction, especially when dealing with large-scale datasets.

**Suitability:**

Diverse features: The furniture sales dataset contains many types of features, such as order date, sales amount, discount, and so on, and the random forest is able to handle all types of features, including numeric and categorical features, allowing for a more comprehensive consideration of the various factors in the sales process.

Feature Importance Evaluation: Random forests are able to provide an importance evaluation for each feature, helping to identify which features are most critical for sales forecasting. This is very helpful for store managers to formulate business strategies and optimize sales processes, which can help them better understand the factors behind sales data.

## 1.2. Model algorithm:

**The principle of random forest:**

Decision Trees: A random forest consists of multiple decision trees, each of which is a classifier. A decision tree is a tree structure based on feature values, where each internal node represents a feature and each leaf node represents a category or a value.

Voting ensemble: For classification tasks, random forest uses a voting mechanism to decide the final classification result. The classification result of each tree is taken as a vote, and the final result is the category with the most votes.

**Hyperparameters:**

n\_estimators: This determines the number of decision trees in the random forest. Increasing this value generally improves the performance of the model, but at the cost of computation.

max\_depth: The maximum depth of each decision tree. Control tree growth to prevent overfitting.

min\_samples\_split: The minimum number of samples required for a node to split.

**The algorithm performs the following steps:**

Data preparation: Read the dataset and split the features and target variables.

Feature processing: the categorical variables were labeled and coded, and the categorical variables were converted into numerical variables. Feature scaling of numeric variables

Data Partitioning: The dataset is divided into training, validation, and test sets.

Feature Dimensionality reduction: PCA is used for feature dimensionality reduction to reduce the dimensionality and computational complexity of the dataset.

Hyperparameter Tuning: Determine the best hyperparameters for the random forest model using grid search and cross-validation.

Model Evaluation: The performance of the model is evaluated on the validation and test sets, including confusion matrix and classification reports.

**Potential variations：**

The performance and generalization ability of random forests are affected by hyperparameters, and different choices of hyperparameters may lead to changes in model performance.Adjusting n\_estimators can balance the accuracy of the model with the computational cost.Tuning max\_depth and min\_samples\_split allows you to control model complexity and prevent overfitting.

## 1.3. Model development:

Feature selection: I selected features from the dataset that are relevant for sales forecasting, including 'Ship Mode', 'Segment', 'Discount', 'Sub-Category', 'Quantity', and 'Region'.

Target variable transformation: Actual sales were converted into three categories: low, medium and high for the classification task.

Advanced data preprocessing Techniques:

Feature scaling: features are normalized using the sklearn.preprocessing.scale function to scale the features to a mean of 0 and a standard deviation of 1.

Principal Component Analysis (PCA) : Using PCA for feature dimensionality reduction reduces the dimensionality and computational complexity of the dataset. I set 95% of the variance explained by the principal components to reduce dimensionality while retaining most of the information.

Dataset split:

Training, validation, and Test set Partitioning:I split the dataset into training, validation, and test sets. The training set is used to train the parameters of the model, the validation set is used to adjust the hyperparameters of the model and evaluate the performance of the model, and the test set is used to finally evaluate the generalization ability of the model.

Stratified sampling: I split the dataset using the sklearn.model\_select.train\_test\_split function and adopt a stratified sampling strategy to ensure that the proportions of samples in the training, validation, and test sets are similar to the proportions in the original dataset.

Model selection and hyperparameter tuning:

Random Forest Model: I chose random forest as the classification model because it performs well in classification problems and is easy to tune.

Hyperparameter Selection: I use grid search to tune the hyperparameters of the random forest model. I chose n\_estimators, max\_depth, and min\_samples\_split as hyperparameters to tune.

Python function and parameter selection:

sklearn library: I used several functions and classes from the sklearn library including LabelEncoder, PCA, SimpleImputer, RandomForestClassifier, GridSearchCV, and others.

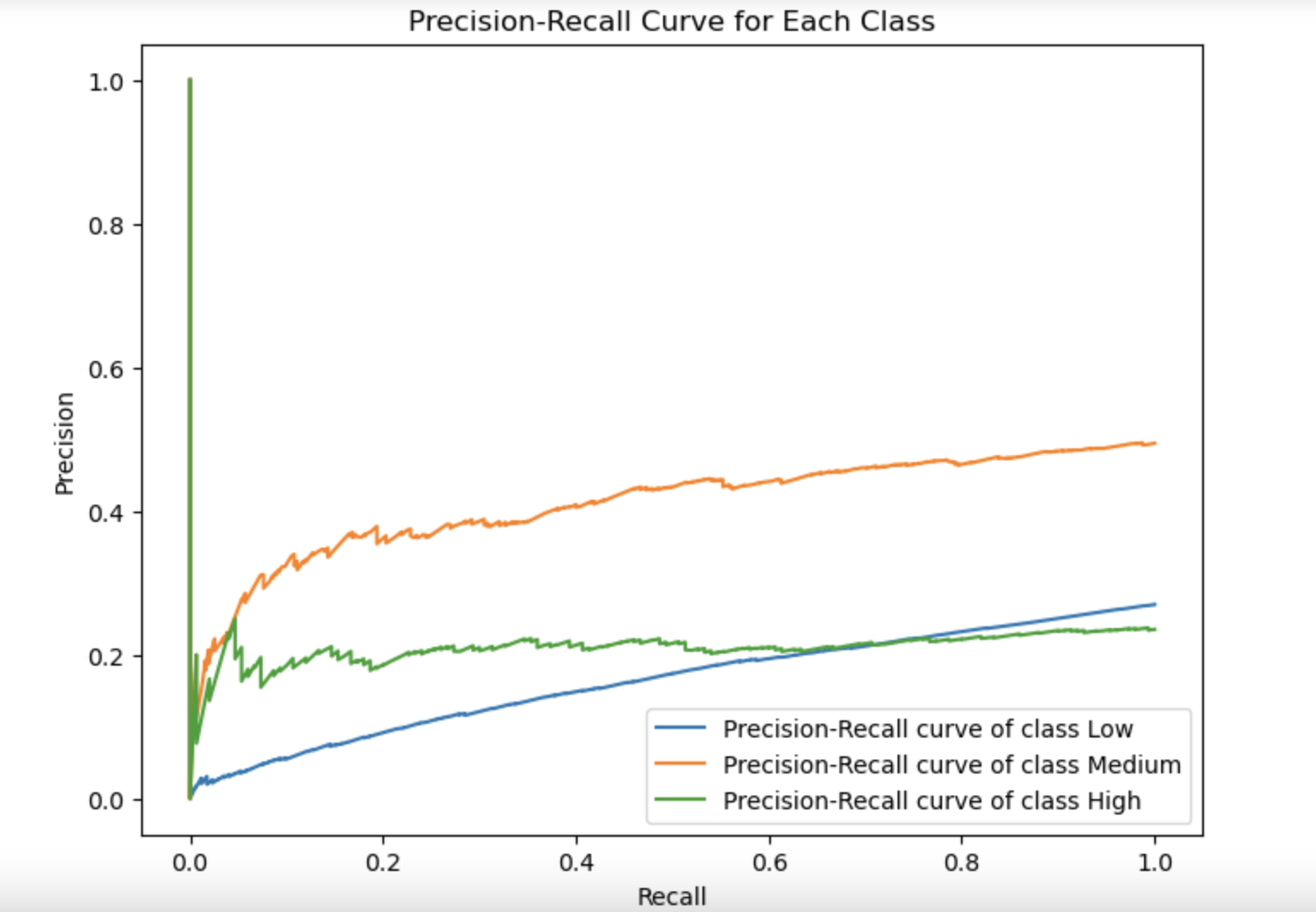
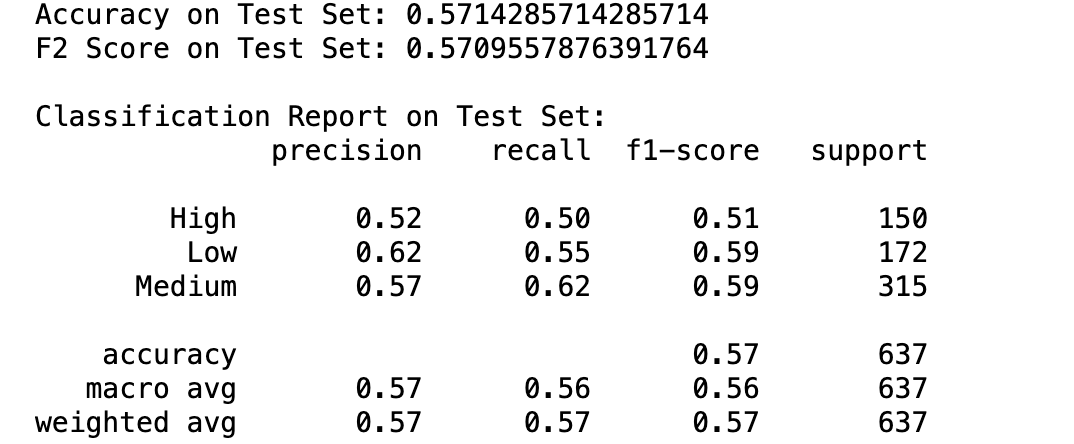
Parameter Meaning: I implement the steps of data preprocessing, model building, hyperparameter tuning, and model evaluation by calling these functions and passing the corresponding parameters.

# 2. Model evaluation and optimization

## 2.1. Model evaluation:

**Model's performance:**

Accuracy: The test set accuracy is 0.57, which is slightly lower than the validation set accuracy. This may indicate a slight decrease in model performance on unseen data, but still maintaining some accuracy.F1 score: The F1 score on the test set is 0.56, which is slightly lower than the validation set F1 score. This may indicate that the performance of the model on the test set is slightly degraded compared to the validation set, but it still maintains a certain balance.



ROC curves:It can help us evaluate the classification performance of the model on different sales categories. Because the target variable in the dataset is the category of sales (high, medium, and low), we can plot ROC curves for each category of sales.

Prediction curve:

The predictive accuracy and stability of the model can be assessed by comparing the difference between the predicted values of the model and the true sales. For each sales category, the prediction curve shows the correlation between the model's predictions and the ground truth.

**Interpretation of results and areas for improvement:**

Category imbalance: The lower prediction accuracy of the model on the high and low sales categories may be caused by the small number of samples in these categories in the dataset. Consider a class-balanced strategy, such as oversampling or undersampling.

Feature engineering: Further analysis and extraction of features in the dataset may help to improve the model's ability to identify different sales categories. For example, consider adding new features related to sales or employing more sophisticated feature engineering techniques.

## 2.2. Model optimization:

In the model optimization phase, the Grid Search is used to adjust the hyperparameters. I chose to use grid search because it is a simple and effective hyperparameter tuning method, especially for small datasets and relatively few hyperparameters. We can define a set of candidate parameters in the specified hyperparameter space and perform a grid search to find the best combination of hyperparameters to improve the model performance.

We use grid search to tune three important hyperparameters of the random forest model: n\_estimators (the number of decision trees), max\_depth (the maximum depth of the tree), and min\_samples\_split (the minimum number of samples required to split an internal node). A grid search is then used to find the best combination of parameters. Result: The best combination of parameters was determined by grid search to be {'max\_depth': 10, 'min\_samples\_split': 2, 'n\_estimators': 200}.

Support Vector Machine (hcao0570)

# 1. Predictive model

## 1.1 Model description

I’m going to use Support Vector Machine (SVM) to predict the sale of furniture. SVM is a powerful classification algorithm, which can be used for binary classification and multi-classification. SVM can find the best hyperplane in datasets to distinguish different categories. It tries to maximize the distance between data points and the decision boundary, so as to ensure better generalization ability. For the case of linear inseparability, SVM can make it linearly separable in a new dimension by importing kernel functions to map the data to the high dimension space. The basic hypothesis for SVM is that the data is linearly separable in certain spaces. Even if the original data is linearly inseparable, it can be linearly separable with appropriate kernel functions. Also, SVM assumes that the input data points are independently and identically distributed, as well as that the outliers in the data have minimal impact. SVM has several advantages. It can use maximized edges to enhance the model generalization. Additionally, by selecting the right kernel function, SVM can process the interaction of nonlinear features. However, it still has some limitations. The calculation complexity of SVM is relatively high, especially when large amounts of memory and calculation resources are required. The performance of SVM is highly dependent on kernel functions and regularization function selection.To our dataset, I believe that SVM can handle the potential complex feature relations by its kernel technique. Meanwhile, the preprocessing of data including feature engineering is required [3].

## 1.2 Model algorithm

1. **Basic principles:** The core idea of SVM is to find the best hyperplane in feature space, so that it maximizes the distance between data points and the decision boundary. For data that is linearly separable, the hyperplane found by SVM can ensure the distance to the nearest data points that relative to the margin is maximized. And with linearly inseparable data, SVM can find the best hyperplane in this space via using the kernel function to map data to high dimension space [3].

2. **Core steps**  
(1) **Selecting kernel function:** select proper kernel function based on distribution feature of the data [3].  
(2) **Build up target function:** construct the optimization problem aimed at maximizing the margin [3].  
(3) **Optimization problem solving:** use the Lagrange multiplier method to convert it to a dual problem, then find the best solution by solving the dual problem [3].  
(4) **Construction of decision function:** build up the final decision function using the found support vector and corresponding Lagrange multipliers [3].

3. **Hyper parameters**  
(1) **C (regularization parameter):** it controls the penalty strength of errors; a larger C can decrease the training error, but can also lead to overfitting [3].  
(2) **Kernel function parameters:** parameters like the degree of the polynomial kernel and the gamma parameter of the radial basis function kernel affect the distribution of data in the new feature space [3].

4. **Pseudo-code:**Input: training dataset **T={(x1,y1),(x2,y2),...(xn,yn)}**, **yi∈{-1,+1}**Selecting kernel function **k(x, x1)** and regularization parameter **C**1. Constructing and solving the constrained optimization problem:  
  
  
2. Calculate corresponding to support vector

3. Constructing decision function:  
  
Output: model f(x)

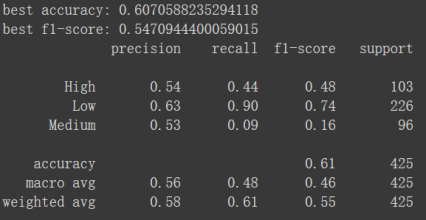
## 1.3 Model development

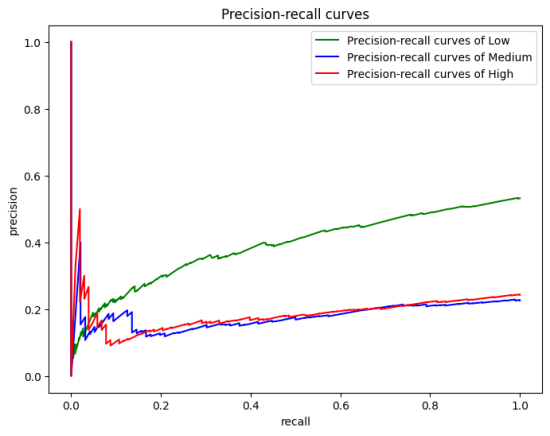
1. **Data preprocessing:**  
(1) **Feature scaling:** using the ‘StandardScaler’ class to have standardized processing to numerical features, which refers to subtracting the mean and dividing by the standard deviation. This makes the feature values approximate a standard normal distribution, which is critical for most machine learning algorithms. Because they’re sensitive to input data.  
(2) **Feature engineering:** using the ‘OneHotEncoder’ class to encode classification features, and turn them into numerical data. So that the model can have a better understanding of data.  
2. **Data segmentation reasons:**  
(1) Using ‘train\_test\_split’ to have data segmentation, with a common split ratio of 80/20, which means 80% of the data is used for training and 20% is used for testing.  
(2) Applying stratified sampling to ensure that the proportion of samples for each category in both the training set and the test set is consistent with the original dataset. This is essential to imbalanced datasets.  
3. **Model-specific functions and hyperparameter selection:**  
(1) **SVM parameters:** ‘kernel=’linear’’: selecting linear kernel function; ‘C=1’: it’s the regularization parameter, which used to control model complexity and avoid overfitting; using RBF kernel (‘kernel=’rbf’’) to optimize the model to fit the interaction of nonlinear features  
(2) **Grid search:** applying ‘GridSearchCV’ class to find the best regularization parameter ‘C’ and ‘gamma’ parameter, combined with cross-validation to boost the model generalization ability.  
4. **Python functions and parameters explanation:**  
(1) **StandardScaler():** it’s used for data standardization  
(2) **OneHotEncoder():** it encodes all categorized features by default  
(3) **SVC(kernel=’linear’, C=1, random\_state=42):** it specifies the type of kernel function used by SVM with the parameter ‘kernel’; the smaller the value of ‘C’ is, the higher the regularization strength; the ‘random\_state’ is used to control the seed of the random number generator  
(4) **Pipeline(steps=[...]):** it’s a collection which simplifies the preprocessing and model training process, where the parameter ‘steps’ defines the processing steps and estimators to be executed.

# 2. Model evaluation and optimization

## 2.1 Model evaluation

To evaluate the SVM model, I used measures including accuracy, precision, recall, F1 score, and precision-recall curves. The F1 score is capable of balancing the effects of precision and recall, and serves as a comprehensive performance metric. And precision-recall curves can provide the relationship between model precision and recall at different thresholds, and be used to evaluate the model's ability to recognize minority classes. Based on the classification report and precision-recall curves shown below, we can know that the best accuracy is 60.71% and weighted f1-score is 54.71%. These indexes show the model ability in correct classification and performance in handling imbalanced datasets. And from the precision-recall curves figure, the PR curve of low sales category performs best, followed by the high sales category, with the worst performance of medium category.





The evaluation results show the problem of the dataset. The model has bad performance in predicting medium sales, which is probably due to the data imbalance between the classes or insufficient differentiation in features among these categories. Also, the retailing data is often influenced by unusual sales activities that can lead to biases in model predictions, especially while inadequate outlier processing is performed.

## 2.2 Model optimization

I think techniques like ensemble learning and recursive feature elimination can be used for model optimization. Because SVM might not have good performance on certain datasets with difficult-defined boundaries. By using ensemble learning methods like bagging and boosting, the variance of the model on complex dataset can be reduced. And the model generalization ability can be improved [4]. The technique of recursive feature elimination is a kind of greedy optimization algorithm which can be used to find the feature subset. It creates models repeatedly and removes the least weighted feature in each iteration. Accordingly, it can reduce the model complexity, enhance model interpretability and avoid overfitting by removing inessential features [6].

2. Group Component 2

## 2.1 Discussion

Decision tree, random forest and SVM are all commonly used machine learning models, which can deal with many problems of classification, or regression if modified.

But they all have their advantages and disadvantages:

| Model | Strength | Limitation |
| --- | --- | --- |
| Decision tree | 1. easily comprehensive 2. can handle multiple-types of data 3. No need of normalization or standardization | 1. easy overfitting 2. unstable with little effects 3. unable to deal with high-dimensional data |
| Random forest | 1. adapt to complex relationships 2. provide feature importance estimates | 1. Black box model 2. Computational resource consumption |
| SVM | 1. data is linearly separable in certain spaces 2. can handle high-dimensional data 3. good generalization performance with small data | 1. sensitive to hyperparameter 2. high cost of time and gpu |

**Quantitatively**：

1. accuracy: they are all around 60, which can’t tell which is better.
2. F1 score: they are also similar, but SVM is a little bit lower than the others.
3. precision-recall curve, the area under the curve can tell the performance, but they are not in the same scale, which can get some improvement.

**Qualitatively**:

1. comprehensiveness: decision tree is more easy to understand than other two models
2. cost: SVM needs much more cost than the other two

## 2.2 Conclusion

Based on the presented evaluation results:

|  | Accuracy | F1 score |
| --- | --- | --- |
| Decision Tree | 0.60 | 0.56 |
| Random Forest | 0.57 | 0.57 |
| SVM | 0.60 | 0.54 |

The SVM model has the highest accuracy, followed by random forest, and the decision tree has the lowest accuracy. The random forest model is slightly higher than the other two models in F1 score.

Considering both accuracy and F1 score, the random forest model is one of the most effective prediction models. It performs well in accuracy and overall performance, and its performance is more stable compared to other models.

For potential avenues for future research, the following aspects can be considered:

Data collection Strategy: Collect more detailed data on product attributes and Track customer interaction behavior data.

Model refinement techniques: Deep Learning Models and Ensemble learning methods.

Interdisciplinary Cooperation:Collaboration with furniture industry experts and Data Science Team Collaboration.

# Reference

[1] P. Huilgol, “Accuracy vs. F1-Score - Analytics Vidhya - Medium,” *Medium*, Jan. 15, 2024. [Online]. Available: https://medium.com/analytics-vidhya/accuracy-vs-f1-score-6258237beca2

[2] D. Steen, “Precision-Recall Curves - Doug Steen - medium,” *Medium*, Dec. 16, 2021. [Online]. Available: https://medium.com/@douglaspsteen/precision-recall-curves-d32e5b290248

[3] S. Suthaharan, “Support Vector machine,” in *Integrated series on information systems/Integrated series in information systems*, 2016, pp. 207–235. doi: 10.1007/978-1-4899-7641-3\_9.

[4] A. Singh, “A Comprehensive Guide to Ensemble Learning (with Python codes),” *Analytics Vidhya*, Nov. 22, 2023. https://www.analyticsvidhya.com/blog/2018/06/comprehensive-guide-for-ensemble-models/

[5] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees*. Wadsworth, Belmont, CA, 1984.

[6] Avcontentteam, “Recursive Feature elimination: working, Advantages & examples,” *Analytics Vidhya*, May 18, 2023. https://www.analyticsvidhya.com/blog/2023/05/recursive-feature-elimination/

| Model | Strength | Limitation |
| --- | --- | --- |
| Decision tree | 1. easily comprehensive 2. can handle multiple-types of data 3. No need of normalization or standardization | 1. easy overfitting 2. unstable with little effects 3. unable to deal with high-dimensional data |
| Random forest | 1. It can estimate the importance of each feature 2. Reduces the risk of overfitting and is particularly suitable for dealing with high-dimensional data. | 1. Long training time: Building a random forest takes more time than a single decision tree. 2. Take a lot of memory and computational resources due to the inclusion of multiple decision trees. |
| SVM | 1. enhance the model generalization 2. process the interaction of nonlinear features, by selecting the right kernel function | 1. calculation complexity of SVM is relatively high 2. SVM performance is highly dependent on kernel functions and regularization function selection |

# 