In this project, we investigated several models using gradient boosting, a type of machine learning algorithm, and compared them to each other.

In this presentation, we will first explain what gradient boosting is and how it works, then introduce three variations of gradient boosting: LightGBM, XGBoost, and Catboost. We will then compare these three models with each other and share the results of implementing them in python code using real datasets.

Gradient boosting is an ensemble learning method that iteratively combines several weak models, typically decision trees, to create a strong predictive model. And it shows effective predictive performance in various problems, especially in classification and regression tasks. Also. It is less sensitive to feature scaling or outlier handling, so it is used to evaluate the importance of variables.

Gradient Boosting uses a tree-based model called the Gradient Boosting Decision Tree. It is an ensemble learning technique where trees are sequentially trained to minimize the residuals of the previous trees. And each tree is constructed by focusing on improving predictions based on the residuals of the previous tree. At this point, the residual is a negative gradient if the loss function is set as a squared error. Finally, GBDT uses the gradient decent to reduce prediction errors, then it adjusts the splitting points and predicted values of the trees using gradients.

In the next chapter, we'll tell you about the three models based on the Gradient Boosting algorithm: XGBoost, LightGBM and CatBoost.

The first variant of Gradient Boosting is XGBoost. XGBoost uses level-wise tree growth for tree growth method and pre-sorted and histogram-based algorithms for splitting methods. And also XGBoost’s parallel computation and distributed training help to faster training, handling big data and low memory usage

XGBoost's tree growth method is level-by-level tree growth. In this method, the tree is expanded level by level, and all nodes at the same level are processed together before the next level. It assigns the current information gain to each node at the same depth, then selects the split with the largest gain and evolves the tree. It also performs well on small data sets. Next,

XGBoost's splitting methods are Pre-sorted algorithm & Histogram based algorithm. Pre-sorted algorithm considers all features and sorts them by feature value. But histogram-based algorithm splits all data points for a feature into discrete bins and uses these bins to find the split value of the histogram. In general, the histogram-based algorithm is more efficient than the pre-sorted algorithm.

The next variant of gradient boosting is LightGBM. Unlike XGBoost, LightGBM has the ability to automatically handle categorical features without preprocessing steps. And it uses leaf-wise tree growth method and gradient-based one-side sampling for splitting method

LightGBM's tree growth method is leaf-wise tree growth. It expands the leaf nodes in a depth-first manner. It selects the splits with the greatest information gain first. Unlike level-wise tree growth, it evolves the tree by splitting it vertically. It provides a faster learning rate and lower memory consumption than the level-wise method.

LightGBM's splitting method is gradient-based one-side sampling. It retains all data instances with large gradients and performs random sampling for data instances with small gradients. It is more efficient than the presorted and histogram-based algorithms.

The final variant is CatBoost. Unlike the previous models. Its tree growth method is the form of symmetric trees whose splitting conditions are consistent across all nodes of the same depth. CatBoost works by the ordered boosting method and takes the ordered target statistic as a regularization technique. Ordered Boosting is the core method of CatBoost. It generates a random permutation and sequentially calculates residuals to train the trees, which helps to prevent goal leakage.

Based on these models, we had the comparison tasks between these models. And we compared these models in pairs. First, we compare XGBoost and LightGBM, XGBoost is functioned based on level-wise tree growth, so it shows fast training speed and excellent performances on small datasets, but it has the limitation on the large datasets because it shows the tendency to use a lot of memory. However, LightGBM is based on leaf-wise tree growth, so it shows excellent performance on large datasets. Also, LightGBM's efficient memory usage makes it advantageous for processing large datasets.

The next comparison is between LightGBM and CatBoost. Both show excellent performance even on large datasets. However, when the dataset has a large number of categorical variables, CatBoost's performance is better. Also, in terms of handling categorical variables, LightGBM's performance drops slightly compared to CatBoost.

The last comparison is between CatBoost and XGBoost. CatBoost specializes in handling categorical variables, but XGBoost has strengths in continuously reducing training errors. And in general, CatBoost's training time is longer than XGBoost's

.In last chapter, we implemented the previous comparison tasks in Python. We prepared four datasets are 'credit-g', 'adult', 'higgs', 'covertype'. 'credit-g', 'adult', 'higgs' are all binary classification problem, and ‘covertype’ has 7 categories. And 'adult' and 'higgs' were chosen to prove that the more a dataset has categorical variables, the better CatBoost's performance is than LightGBM's performance. The datasets were preprocessed uniformly and to ensure an accurate performance comparison, number of trees, learning rate and maximum tree depth were unified

When we compared time and memory, as the size of the datasat increases, XGBoost slowed down significantly, and CatBoost also slowed down. But LightGBM showed stable speeds when the dataset gets very large. Concerning memory usage, all three models show similar behavior.

Next, when we compared model performance, we concluded that a significant difference in performance wasn’t existed.

Lastly, when we compared with logistic regression, it was clearly showed that the gradient boosting based models outperform logistic regression

From our research results, we found that CatBoost has an advantage in datasets with high categorical values, but this was not the case when we applied the model. Also, the performance difference between the three models did not increase significantly as the size of data increased or as the number of features increased. So we concluded that there seems to be a slight difference between theory and practice.

In conclusion, we learned that it’s not important to use one model, but to test different models with different hyperparameter settings and cross-validation to find the best model