Bank Marketing Campaign

Uday Singh

26-March-2023

Agenda

- Problem Statement
- Models
- Decision Trees
- Summary
- Recommendations

Background

- This is a binary classification problem where the response variable is ‘y’ (yes or no). The goal is to predict whether or not a customer will respond positively based on the data provided.
- The dataset has 20 features, excluding response, half of which are categorical variables (job, education, marital etc.) and the rest numerical (age, duration etc.).
- The dataset is also heavily imbalanced, in that 89% of the responses are ‘no’. This makes it hard to achieve accuracy in prediction.

Approach:

- Choose different classes of models (Logistic Regression, SVM, Decision Trees, KNN etc.).
- Build a pipeline for preprocessing data then use re-sampling techniques to adjust for imbalance in data.
- Use GridSearch to tune hyper parameters for each model.
- Understand which metric to use when comparing model performance (Accuracy, F1-Score, AUC etc.)
- Choose best model(s) based on performance, computational cost and training time.

Models

Models to test:

- Logistic Regression: a statistical model that analyses the relationship between multiple input variables and a binary output variable, and outputs the probability of the binary outcome.
- Support Vector Classifier (Linear and Non-Linear kernels): a machine learning algorithm that separates data points using a hyperplane in a high-dimensional space, and can be used for both linearly and non-linearly separable data.
- Stochastic Gradient Descent: an iterative optimisation algorithm used to minimise a loss function that is commonly used in deep learning for updating the weights of neural networks.
- K-Nearest-Neighbours: a machine learning algorithm that identifies the k-nearest neighbours of a data point and uses their classification to determine the classification of the point.
- Bayes Classifier (Naive Bayes and Complement Bayes): a probabilistic model that utilises Bayes' theorem to classify data based on the probability of an event given prior knowledge.
- Decision Trees: a tree-like model that uses a set of rules to classify data based on features, by splitting the data into smaller groups until a decision is made.

Metrics

Common metrics used test model performance:

- `Training Time`: the amount of time (in minutes) taken to perform grid search for hyper parameters and train the model.
- `Accuracy`: the proportion of correctly classified instances out of all instances.
- `Balanced Accuracy`: similar to accuracy, but it adjusts for imbalanced classes.
- `F1-score`: the harmonic mean of precision and recall, a metric that combines both measures.
- `AUC-ROC`: the area under the Receiver Operating Characteristic (ROC) curve, which measures the model's ability to distinguish between positive and negative instances.
- `Precision`: the proportion of true positive predictions out of all positive predictions.
- `Recall`: the proportion of true positive predictions out of all actual positive instances.
- `MCC`: Matthews Correlation Coefficient, a correlation coefficient between the predicted and actual binary classifications.
- `Log-Loss`: a measure of the difference between the predicted and actual probabilities of the model's predictions.

When dealing with imbalanced data, using metrics such as accuracy can be misleading as it can be high even when the model is not performing well on minority classes. In such cases, AUC-ROC, F1-Score and Balanced Accuracy are more reliable metrics.

Linear Classifiers

Logistic Regression

Training Time	Accuracy	Balanced Accuracy	F1-score	AUC-ROC	Precision	Recall	MCC	Log-Loss
1.822	0.86559	0.72619	0.48821	0.7914	0.44295	0.54377	0.41464	4.64242

SGD Classifier

Training Time	Accuracy	Balanced Accuracy	F1-score	AUC-ROC	Precision	Recall	MCC	Log-Loss
1.047	0.86887	0.71154	0.47624	0.79163	0.45005	0.50566	0.40247	4.52918

- Logistic Regression and SGD are both linear models that are popular for classification problems.
- Grid Search takes slightly longer for LR as there are many parameters to explore, such as ‘solver’ and ‘penalty’.
- LR grid search results suggest the ‘l2’ penalty using the ‘lbgfs’ solver.
- However, for the SGD clarifier, grid search results suggest it performs best when there is no penalty.
- LR performs better in terms of Balanced Accuracy and F1-Score, but SGD has a slightly higher AUC.

Non-Linear Classifiers

SVC

Training Time	Accuracy	Balanced Accuracy	F1-score	AUC-ROC	Precision	Recall	MCC	Log-Loss
148.728	0.86875	0.7253	0.49129	0.77457	0.45234	0.53759	0.41863	4.53338

KNN

Training Time	Accuracy	Balanced Accuracy	F1-score	AUC-ROC	Precision	Recall	MCC	Log-Loss
2.644	0.80707	0.70863	0.41473	0.74429	0.32282	0.57981	0.32936	6.6638

Decision Tree

Training Time	Accuracy	Balanced Accuracy	F1-score	AUC-ROC	Precision	Recall	MCC	Log-Loss
2.747	0.86899	0.71562	0.481	0.7824	0.45126	0.51493	0.40757	4.52499

- SVC, KNN and DT are some of the non-linear models commonly used for classification.
- Although SVC can also be run with a linear kernel, GridSearch suggests to use ‘ref’ kernel which makes it non-linear.
- SVC takes considerable time to train and its performance is not significantly better than the other models.
- Decision Tree has great performance in terms of AUC and training time.
- DT is specially suited to dealing with imbalanced data and also data where there are a lot of categorical features.
- This is only the base Decision Trees Classifier, we will later look at some ensemble methods which perform even better.

Probabilistic Classifiers

Gaussian NB

Training Time	Accuracy	Balanced Accuracy	F1-score	AUC-ROC	Precision	Recall	MCC	Log-Loss
0.18	0.81933	0.71602	0.43119	0.7764	0.34286	0.58084	0.3485	6.24023

Complement NB

Training Time	Accuracy	Balanced Accuracy	F1-score	AUC-ROC	Precision	Recall	MCC	Log-Loss
0.188	0.76506	0.7205	0.39925	0.7779	0.28578	0.6622	0.31916	8.11484

- Gaussian Naive Bayes and Complement Bayes are probabilistic classifiers that can be easily adapted to different types of data by modifying the prior and likelihood distributions used in the model.
- They are also transparent and interpretable, making it easier to understand how the model arrived at its predictions.
- GNB assumes a Gaussian distribution for the features, thus it is a non-linear model, whereas CNB is a linear model because it uses a linear decision boundary to classify instances.
- As we can see, there is no clear winner. GNB has higher F1-Score whereas CNB has a higher balanced accuracy.
- Since we do not have many numerical features which are (approximately) Gaussian, it is not surprising that GNB does not perform too well.
- NB classifiers do not seem to have any advantage over the other models we have looked at so far.

DT - Ensemble Methods

Models to test:

- `DecisionTree`: This is a basic decision tree model that recursively splits the data based on the value of a single feature at each node.
- `RandomForest`: This is an ensemble model that combines multiple decision trees by randomly selecting subsets of features and instances to build each tree. The final prediction is made by aggregating the predictions of all trees.
- `AdaBoost`: This is another ensemble model that combines multiple decision trees, but it does so by sequentially adding trees that focus on misclassified instances from the previous trees.
- `XGBoost`: This is a gradient boosting model that uses a similar approach as AdaBoost, but it uses gradient descent to optimise the model's performance and prevent overfitting.
- `BaggingClassifier`: This is another ensemble model that combines multiple decision trees by bootstrapping the data and aggregating the predictions of all trees.
- `LGBMClassifier`: This is a gradient boosting model that uses a similar approach as XGBoost, but it uses a different algorithm to optimise the model's performance and can handle large datasets with high-dimensional features.
- `CatBoostClassifier`: This is another gradient boosting model that uses a different approach to handle categorical features, reducing the need for pre-processing and feature engineering.
- `RGFClassifier`: This is a decision tree-based model that uses a different algorithm to build the trees and can handle high-dimensional features and noisy data.
- `ExtraTreesClassifier`: This is another ensemble model that is similar to RandomForest, but it uses random splits at each node rather than searching for the best split, making it faster but potentially less accurate than RandomForest.

DT, RF and XT

Decision Tree

Training Time	Accuracy	Balanced Accuracy	F1-score	AUC-ROC	Precision	Recall	MCC	Log-Loss
2.747	0.87433	0.7182	0.4909	0.78516	0.46987	0.5139	0.41993	4.34047

Random Forest

Training Time	Accuracy	Balanced Accuracy	F1-score	AUC-ROC	Precision	Recall	MCC	Log-Loss
1.191	0.87118	0.73069	0.50024	0.79608	0.46094	0.54686	0.42896	4.44951

Extra-Trees Classifier

Training Time	Accuracy	Balanced Accuracy	F1-score	AUC-ROC	Precision	Recall	MCC	Log-Loss
45.154	0.8696	0.73292	0.50047	0.79418	0.45632	0.55407	0.42894	4.50403

- Both Random Forest and Extra-Trees are ensemble methods that use multiple decision trees to predict classes.
- Extra Trees takes long (45 min) to train, on top of which, the model is highly complex and the trained model file (.pkl) turns out to be extremely large (~100 mb).
- Although there is a slight improvement in performance (over DT base model), it is not significant enough.
- Random Forest, on the other hand, not only performs well but it is also very quick to train.
- RF is definitely an improvement on the base model DT.

AdaBoost, Bagging and RGF

AdaBoost

Training Time	Accuracy	Balanced Accuracy	F1-score	AUC-ROC	Precision	Recall	MCC	Log-Loss
1.715	0.87263	0.70921	0.47837	0.79204	0.4625	0.49537	0.40624	4.39918

Bagging Classifier

Training Time	Accuracy	Balanced Accuracy	F1-score	AUC-ROC	Precision	Recall	MCC	Log-Loss
1.751	0.86656	0.72317	0.48621	0.79222	0.44521	0.53553	0.41259	4.60887

RGF Classifier

Training Time	Accuracy	Balanced Accuracy	F1-score	AUC-ROC	Precision	Recall	MCC	Log-Loss
6.363	0.8753	0.7076	0.48	0.79482	0.47211	0.48816	0.40925	4.30691

- These are also ensemble methods that use multiple decision trees to predict classes.
- The models are pretty quick to train, with the exception of RGF.
- RGF shows the best performance in terms of AUC but Bagging Classifier is superior in other aspects.
- The training time can be reduced by carefully choosing parameters in the Grid Search.
- AdaBoost and Bagging Classifier seem to be approximately equal in performance, however, AdaBoost is much faster when expanding the Grid Search over multiple parameters.

XGB, LGBM and CatBoost

XGBoost

Training Time	Accuracy	Balanced Accuracy	F1-score	AUC-ROC	Precision	Recall	MCC	Log-Loss
37.692	0.86377	0.72515	0.48485	0.79025	0.43745	0.54377	0.41062	4.70532

LGBM Classifier

Training Time	Accuracy	Balanced Accuracy	F1-score	AUC-ROC	Precision	Recall	MCC	Log-Loss
3.519	0.87664	0.73602	0.51341	0.80302	0.47986	0.55201	0.4446	4.26079

CatBoost

Training Time	Accuracy	Balanced Accuracy	F1-score	AUC-ROC	Precision	Recall	MCC	Log-Loss
6.862	0.87919	0.71471	0.49364	0.80494	0.48793	0.49949	0.4251	4.17272

- These are also ensemble methods that use gradient boosting to arrive at the final estimator.
- The models are not very quick to train, with XGB alone taking about 30 minutes.
- Training time can be reduced by carefully selecting grid search parameters.
- It can be seen that LGBM both other models when looking at balanced accuracy, F1-score and AUC. It is also the quickest to train.
- CatBoost is also a viable option since out data has a lot of categorical variables. Additionally, the CatBoost module has its own internal encoder as well as cross-validator, so that would save a lot of time on data processing and CV.

Summary

We have looked at different type of base models (LR, SVM, KNN etc). Decision tree models offer several advantages in classification tasks:

1. **Interpretability:** Decision trees are easy to understand and interpret, as they mimic human decision-making processes. The tree structure visually represents a series of decisions based on feature values, making it simple to explain the model's logic to non-experts.
2. **Minimal data preprocessing:** Decision trees can handle both numerical and categorical variables and don't require extensive data preprocessing, such as feature scaling or normalisation. They are also less sensitive to outliers compared to some other models.
3. **Handling missing data:** Decision trees can handle missing data more effectively than some other models, as they can make splits based on the available data and create surrogate splits to manage the missing values. This strength of DT will come in handy since our data as ‘unknown’ values in 6 columns.
4. **Nonlinear relationships:** Decision trees can capture nonlinear relationships between features and the target variable, which might not be easily captured by linear models.
5. **Feature selection:** Decision trees perform automatic feature selection as part of the model building process. Features that contribute more to the target variable prediction will appear higher up in the tree, while less important features may not be included at all.
6. **Parallelisable:** Decision tree algorithms, particularly ensemble methods like random forests, can be easily parallelised, leading to faster training times on multi-core machines or distributed systems.

However, decision tree models have some limitations as well, such as their propensity to overfit and their sensitivity to small changes in the training data. Ensemble methods like random forests or gradient boosting machines can help address some of these issues by combining multiple trees to improve generalisation and reduce overfitting.

Recommendations

1. Of the models we have tested, LGBM and CatBoost show the most promise.
2. CatBoost is a bit slow to train, also might not be the best choice if you want to use a standard pre-processing pipeline.
3. However, CatBoost is designed for categorical data so it has some merit for this particular dataset.
4. Random Forest is another good choice. It is fast to train but is slightly worse in performance than CatBoost and LGBM (BA, F1-score and AUC).
5. Ensemble methods using DT consistently perform better than other models (LR, SVC etc.).
6. SMOTE and RandomSampling have been used to correct for the heavy imbalance in the data.
7. The final recommendation is to employ the LGBM model.

Thank You