Machine Learning 1 project

Jakub Bandurski & Anirban Das

June 9, 2023

Github repository: *github.com/jtbandurski/ML-1-Project*

Overview

1. Classification task
   - Data preprocessing
   - Validation approach
   - Chosen models
   - Results
2. Regression task
   - Data preprocessing
   - Validation approach
   - Chosen models
   - Results

Classification

Data preprocessing

Preprocessing pipeline outline:

- Locate columns with NaNs (age, sex, salary, amount; 2730, 1283)
- One Hot Encode all factor columns apart from two with NaNs
- Notice that there is no *Platinium* value in test set (add column od 0s)
- Standardise data with z-score transformation
- Use Bayesian Ridge quick imputation method (round factor variables)
- One Hot Encode imputed factors
- Standardise with z-score again to correct for imputation

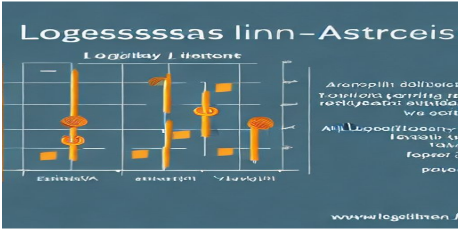
Validation approach

- Since the distribution of the target variable is imbalanced with a ratio of negatives to positives of around 1 : 5 we decided to use Stratified Cross Validation.
- As the size of the data set is manageable twice Repeated 10–fold Stratified Cross Validation has been implemented for all models.
- The evaluation metric used for all models was balanced accuracy.

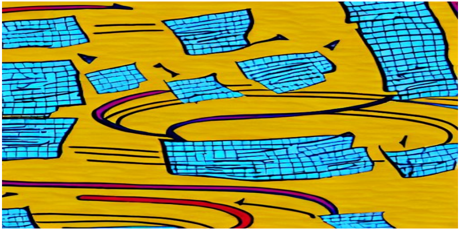
$$BA = \frac{TPR + TNR}{2}$$

Chosen models

logit

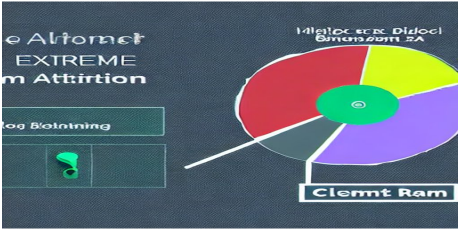


svm



knn

xgb



Source: stablediffusionweb.com

Tuned Hyperparameters

- Logistic regression
  - *penalty*: L1, L2, elastic net, None
  - *solvers*
- k nearest neighbours
  - k
  - Minkowski metric p
- Support Vector Machine
  - Regularisation C
  - *kernel*: polynomial, radial basis function
  - γ scaling factor
  - *degree* of the polynomial

Tuned Hyperparameters

eXtreme Gradient Boosting

- η learning rate
- *max depth* maximum height of trees
- λ regularisation parameter
- *min child weight* minimal number of observations in a leaf
- γ minimal increase in performance to partition a leaf
- *colsample bytree* fraction of features used for each tree

More details in original paper *here* or on *XGBoost website*

Results

Best performing configurations of models covered in class

	mean train	std train	mean validation	std validation
logit	0.740004	0.002419	0.735738	0.016798
knn	0.734287	0.002418	0.652311	0.014095
svm	0.960220	0.001817	0.795377	0.15130

logit: penalty: None, solver: lbfgs
knn: $k = 5$, $p = 1$
svm: $C = 20$, kernel: rbf, $\gamma = 1/num\_features$

Results

	mean train	std train	mean validation	std validation
xgb0	1	0	0.924831	0.013445
xgb1	0.960292	0.002436	0.923720	0.012283
xgb2	0.972100	0.001836	0.922489	0.012369
xgb3	0.967937	0.001835	0.921301	0.014191

	gamma	min_child_weight	eta	max_depth	lambda
xgb0	0.01	1	1	6	10
xgb1	0.1	1	1	2	10
xgb2	1	5	0.1	6	0.1
xgb3	0.01	5	1	2	1

In the case of all xgboost models above colsample_bytree was equal to 1 and number of trees equal to 100.

Final choice classification task

Considering the results of the hyperparameter tuning process with twice Repeated 10–Fold Cross Validation we choose model *xgb3* as the one with the smallest potential of overfitting in the test environment. The parameters that prevent overfitting are min_child_weight with higher value and max_depth with lower value.

The decision was made based on:

- mean validation error
- difference between train and validation error
- parameters min_child_weight 5 and max_depth 2
- similar intervals of expected performance

Final choice classification task

The expected value of prediction score for the chosen model is the validation score which lies in the interval

$$[0.907110, 0.935492]$$

Hyperparameters values:

- $\eta = 1$
- $max\ depth = 2$
- $\lambda = 1$
- $min\ child\ weight = 5$
- $\gamma = 0.01$
- $colsample\ bytree = 1$
- $number\ of\ trees = 100$

Regression

Data preprocessing

Outline of the data:

- Dataset containing 2398116 different observation with 14 different variables.
- Massive NA counts but less than 50
- **Mode imputation** for categorical variables with frequency.
- **Median imputation** for numerical variables with skewed distribution.

Data Validation

- The target variable being continuous, Standard K-fold cross validation is used.
- The dataset is massive, thus 5 fold is used with twice repetition.
- Evaluation Metrics is MAPE (Mean Absolute Percentage Error)

$$\text{MAPE}(y, \hat{y}) = \frac{100\%}{N} \sum_{i=0}^{N-1} \frac{y_i - \hat{y}_i}{y_i}.$$

Chosen Models and Tuned Hyper parameters

- Linear Regression.
- Elastic Net.
  - alpha: The mixing parameter between L1 and L2 regularization. It controls the balance between the two penalties.
  - l1 ratio: The ratio of L1 penalty in the total penalty determining the type regularization.
- Support Vector Regression.
  - C: Regularization parameter
  - epsilon:The margin of tolerance for error.
  - loss: It determines how errors are penalized during training.
- Random Forest.
  - n estimators: The number of decision trees in the random forest.
  - max depth: The maximum depth of each decision tree in the random forest.

Best performing configurations:

	mean train	std train	mean validation	std validation
LR	-0.16312	0.000148	-0.163121	0.000600
Elastic Net	-0.163120	0.000148	-0.163122	0.000599
SVR	-0.189858	0.035672	-0.189888	0.035643
RF	-0.156120	0.000135	-0.156878	0.000573

Hyperparameters: Linear Regression: None
Elastic Net: alpha: 0.01, l1 ratio = 0.9
Support Vector Regression: C = 1, epsilon = 0.01, loss = epsilon insensitive.
Random Forest: n estimators = 200, max depth = 10.

Final Choice of Regression Task

Due to massive size of the data proper cross validation was not possible but we managed to run twice repeated 5 fold cross validation across a varied set of hyperparameters. After careful consideration, the model of choice is **Random Forest**. After running for over 15 hours, for tuned hyperparameters below. The chosen model is number **5** with the lowest mean validation score and std similar to competitors.

	n_est	depth	mean_train	std_train	mean_valid	std_valid
5	200	10	-0.156120	0.000135	-0.156878	0.000573
4	100	10	-0.156127	0.000133	-0.156885	0.000573
3	50	10	-0.156136	0.000140	-0.156892	0.000563
2	200	5	-0.161008	0.000159	-0.161043	0.000566
1	100	5	-0.161010	0.000155	-0.161045	0.000571
0	50	5	-0.161011	0.000161	-0.161047	0.000565

Limitation: Computer Power/Performance

Thank you