Marketing Targets

--- Customer Subscription Prediction

-- By Xufei Li

Problem:

Client: a Portuguese banking institution

- “Term deposits” - a major source of income
- Direct marketing campaigns (phone calls)
- Goal - predict if the customer will convert.

Data: Kaggle - Banking Dataset - Marketing Targets

(Train :45,211 rows; Test: 4,521 rows)

1

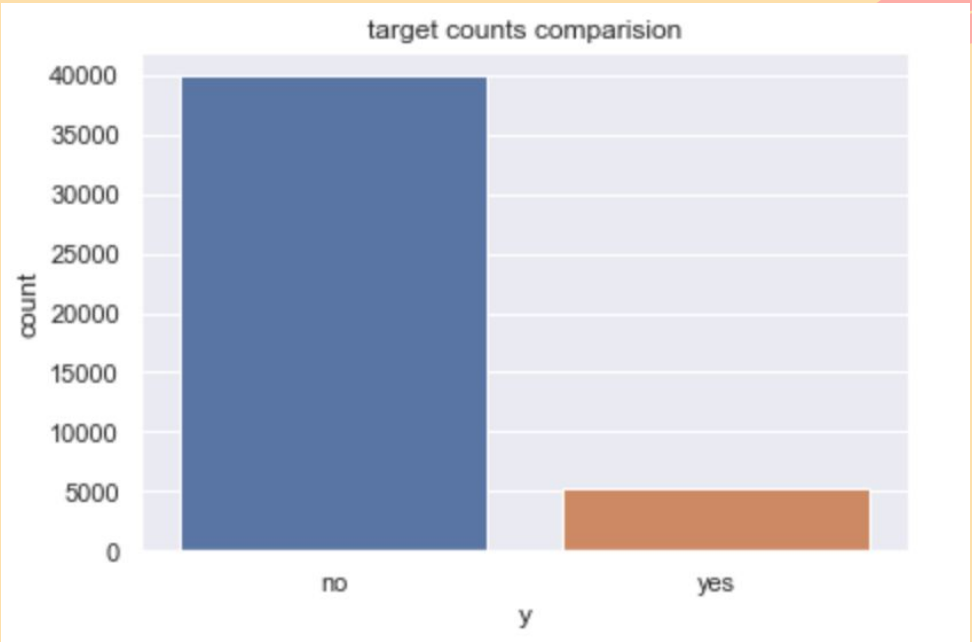
Data Cleaning & EDA

Data Cleaning & EDA

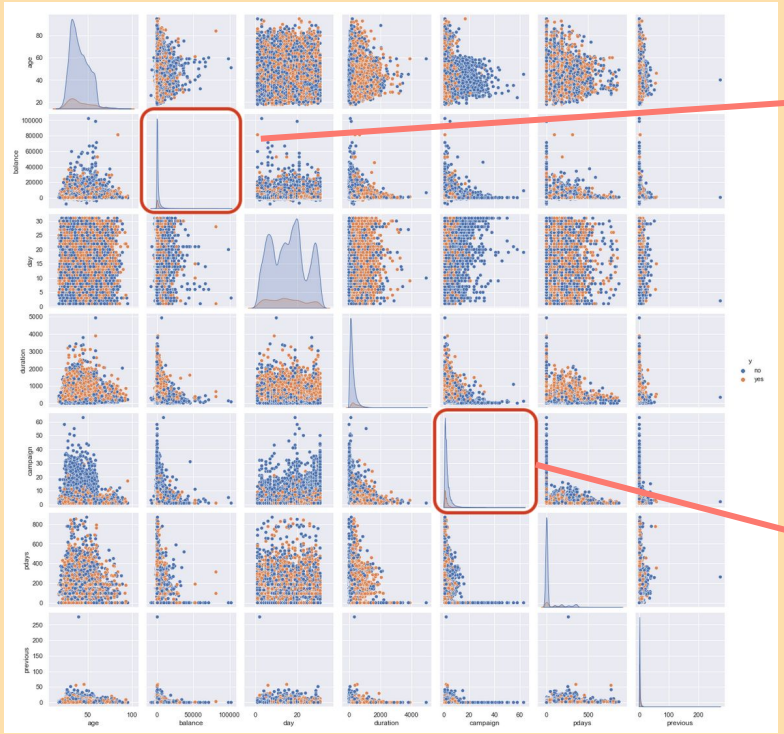
Drop “duration” feature

Link for the reason

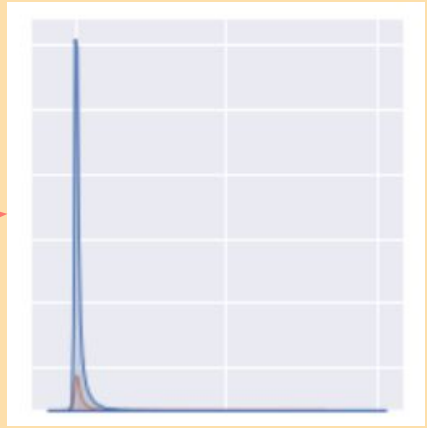
Class imbalance: Oversampling



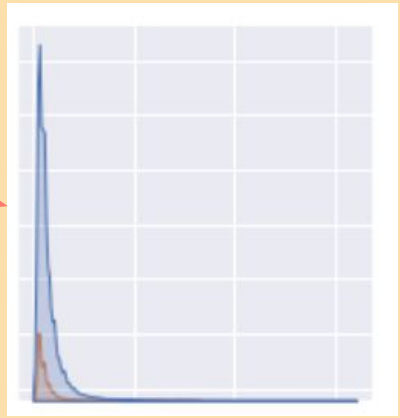
EDA



“balance”



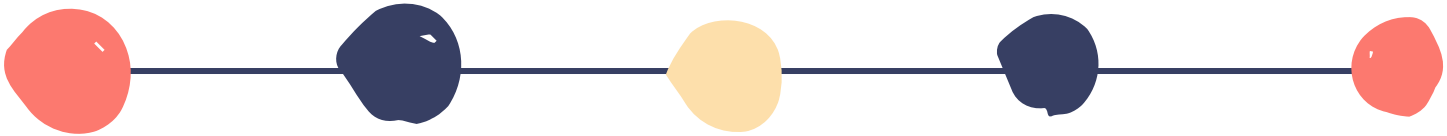
“campaign”



2

Modeling

Workflow



Metric Choosing

F-beta(beta = 2)
ROC_AUC

Baseline model

Logistic regression

Model Selection

6 model's comparison

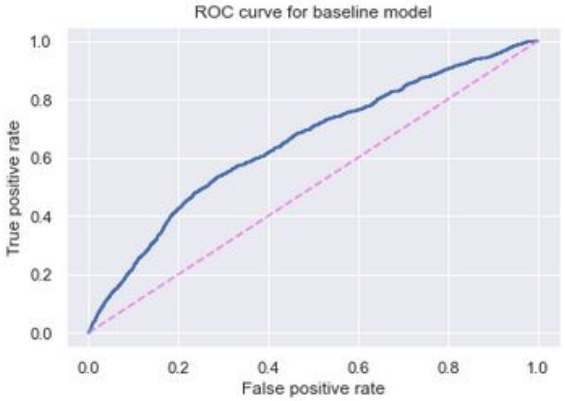
Retrain Model

Retrain on (train+val)

Model evaluation & Interpretation

Baseline Model

Features: numerical features only - age, balance, day, campaign, pdays, previous



F-beta: 0.7582
ROC_AUC: 0.6502

Improving Baseline Model

Baseline Model
+ Scaling
+ Feature Engineering

F-beta: 0.7933
ROC_AUC: 0.7687

Candidate Models

Baseline model:
F-beta: 0.7582
ROC_AUC: 0.6502

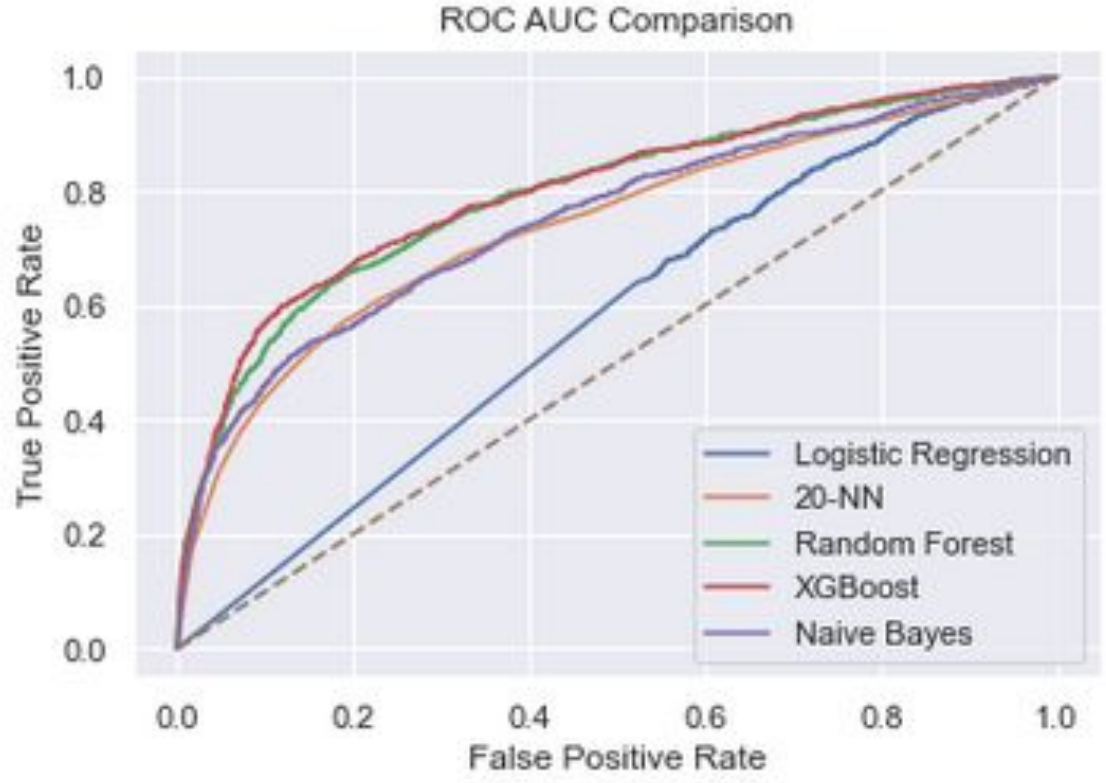
F-beta(beta = 2)

Logistic Regression	20-NN	Random Forest	XGBoost	Naive Bayes	SVM
0.7933	0.7632	0.8333	0.8329	0.8016	0.8388

ROC_AUC

Logistic Regression	20-NN	Random Forest	XGBoost	Naive Bayes	SVM
0.7687	0.7392	0.7928	0.7992	0.7487	N/A

ROC_AUC curve comparison



Winner: XGBoost

Final Model - XGBoost

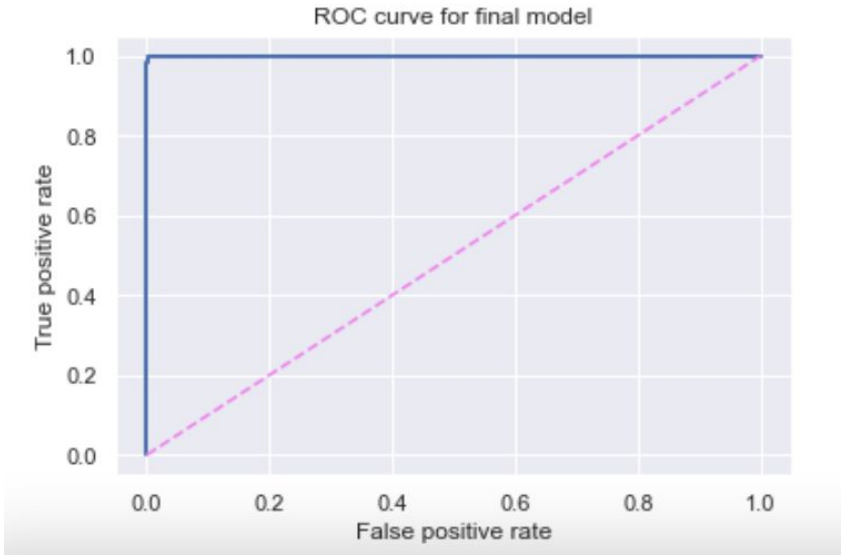
Current performance:	**F-beta**: 0.8329	ROC AUC: 0.7992
Step 1: Tuning hyperparameter for XGBoost Best score on val:	**F-beta: 0.8536**	ROC_AUC: 0.7983
Step 2: Retrain model with (train+val) data **Final Score on Test:**	**F-beta: 0.9946**	ROC_AUC: 0.9998

Model Evaluation & Interpretation

Hard Prediction

Perfect Classifier???

Soft Prediction



Future Work

1. Double check the code settings
2. Try different random_state (validation?)
3. Try Random Forest to see if get similar result
4. Tuning hyperparameters for SVM to see how good it can be.

Appendix 1: Final Model

```
gbm2 = xgb.XGBClassifier(
                         n_estimators=10000,
                         max_depth=11,
                         objective='binary:logistic', #new objective
                         learning_rate=0.05,
                         subsample=.8,
                         min_child_weight=2,
                         colsample_bytree=.8
                        )

eval_set=[(X_tr_rs, y_tr_rs),(X_test_dum,y_test)]  #using test set
fit_model = gbm2.fit(
                     X_tr_rs, y_tr_rs,
                     eval_set=eval_set,
                     eval_metric='auc', #new evaluation metric: classification error (could also use AUC, e.g.)
                                   #can "eval_metric" be F-beta score
                     early_stopping_rounds=20,
                     verbose=False
                    )

y_pred_test = gbm2.predict(X_test_dum, ntree_limit=gbm2.best_ntree_limit)
y_prob_pred_test = gbm2.predict_proba(X_test_dum)[:,1]


print(f"train f-beta score: {fbeta_score(y_tr_rs,gbm2.predict(X_tr_rs),average='weighted',beta = 2)}")
print(f"train ROC_AUC score: {roc_auc_score(y_tr_rs,gbm.predict(X_tr_rs))}")

print(f"test f-beta score: {fbeta_score(y_test,y_pred_test,average='weighted',beta = 2)}")
print(f"test ROC_AUC score: {roc_auc_score(y_test, y_prob_pred_test)}")
```

```
train f-beta score: 0.9967927081981969
train ROC_AUC score: 0.7477706527729072
test f-beta score: 0.9946969122222279
test ROC_AUC score: 0.9998795585412668
```