Project 3: Dogs, Fried Chicken or Blueberry Muffins?

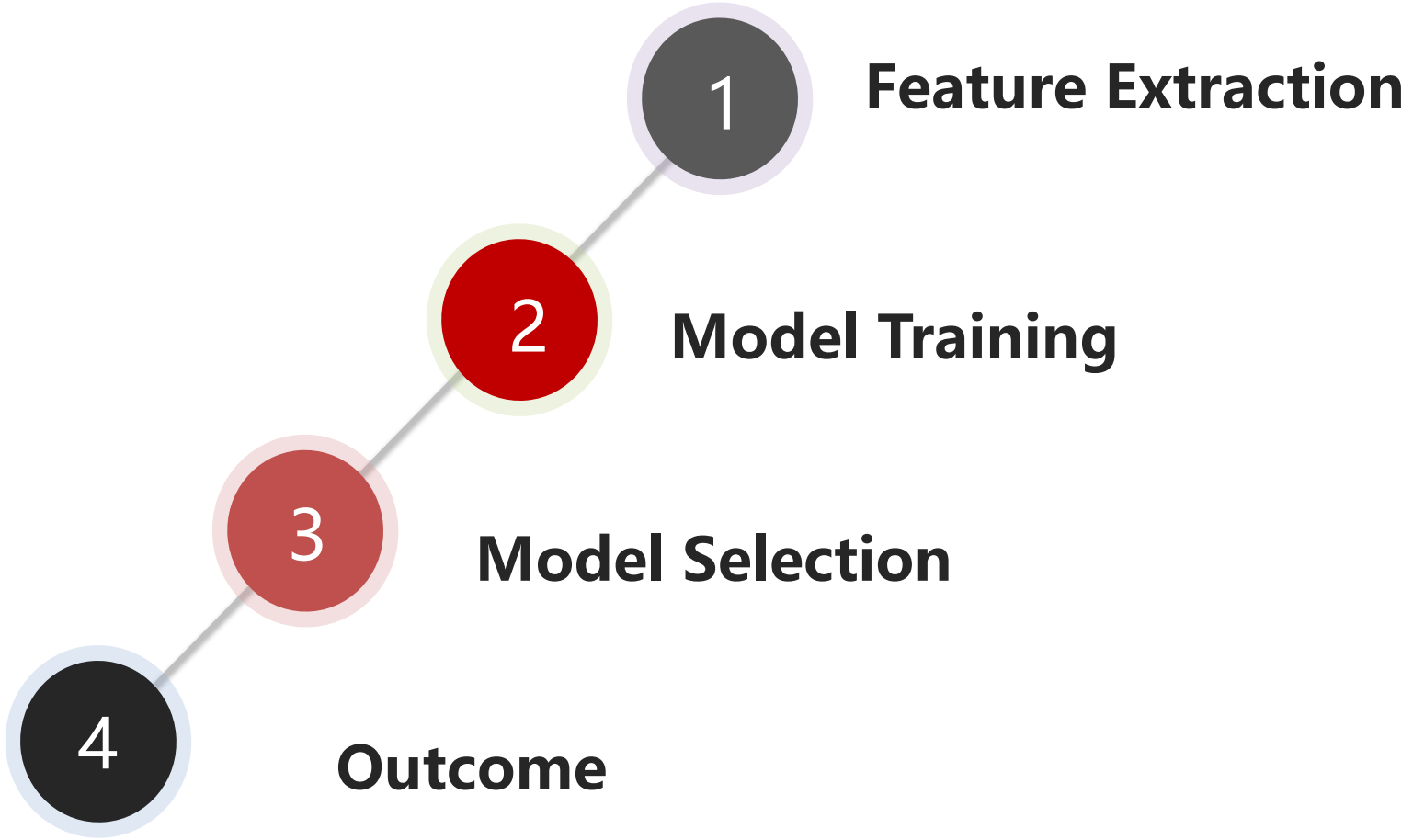
Group 10 Members:

Wenyuan Gu
Wenshan Wang
Yiyi Zhang
Xinlei Cao

Introduction

In this project, we have created a classification engine for images of poodles, fried chicken and blueberry muffins. Our goal is to improve the prediction accuracy from the baseline model (GBM with decision stumps on 2000 SIFT features) and to enhance computational efficiency in terms of running time, storage and memory cost.

Outline


1
Feature Extraction
2
Model Training
3
Model Selection
4
Outcome

1

PART

Feature Extraction

Feature extraction

SIFT

- Baseline
- A classical approach, local descriptor
- Rotation and scale invariant

HOG

- Histogram of Oriented Gradients
- Global descriptor

LBP

- Local Binary Patterns

2

PART

Model Training

Models

1. Gradient Boosting Model (baseline)
2. Support Vector Machine (SVM)
3. Random Forest (RF)
4. Logistic Regression
5. Xgboost
6. Convolutional Neural Networks(CNN)

3

PART

Model selection

Tune parameter

```
run.sift = F
run.hog = F
run.lbp = F
```

```
run.gbm = F
run.svm = F
run.xgboost = F
run.rf = F
run.logistic = F
```

```
model_values <- seq(3, 11, 2) # depth for GBM
# model_labels = paste("GBM with depth =", model_values)

svm_values <- seq(0.01, 0.1, by = 0.02) # gamma for svm
# svm_labels = paste("SVM with gamma =", svm_values)

xgboost_values <- seq(0.1, 0.5, by = 0.1) # eta for xgboost
# xgboost_labels = paste("XGBoost with eta =", xgboost_values)
```

Tune parameter

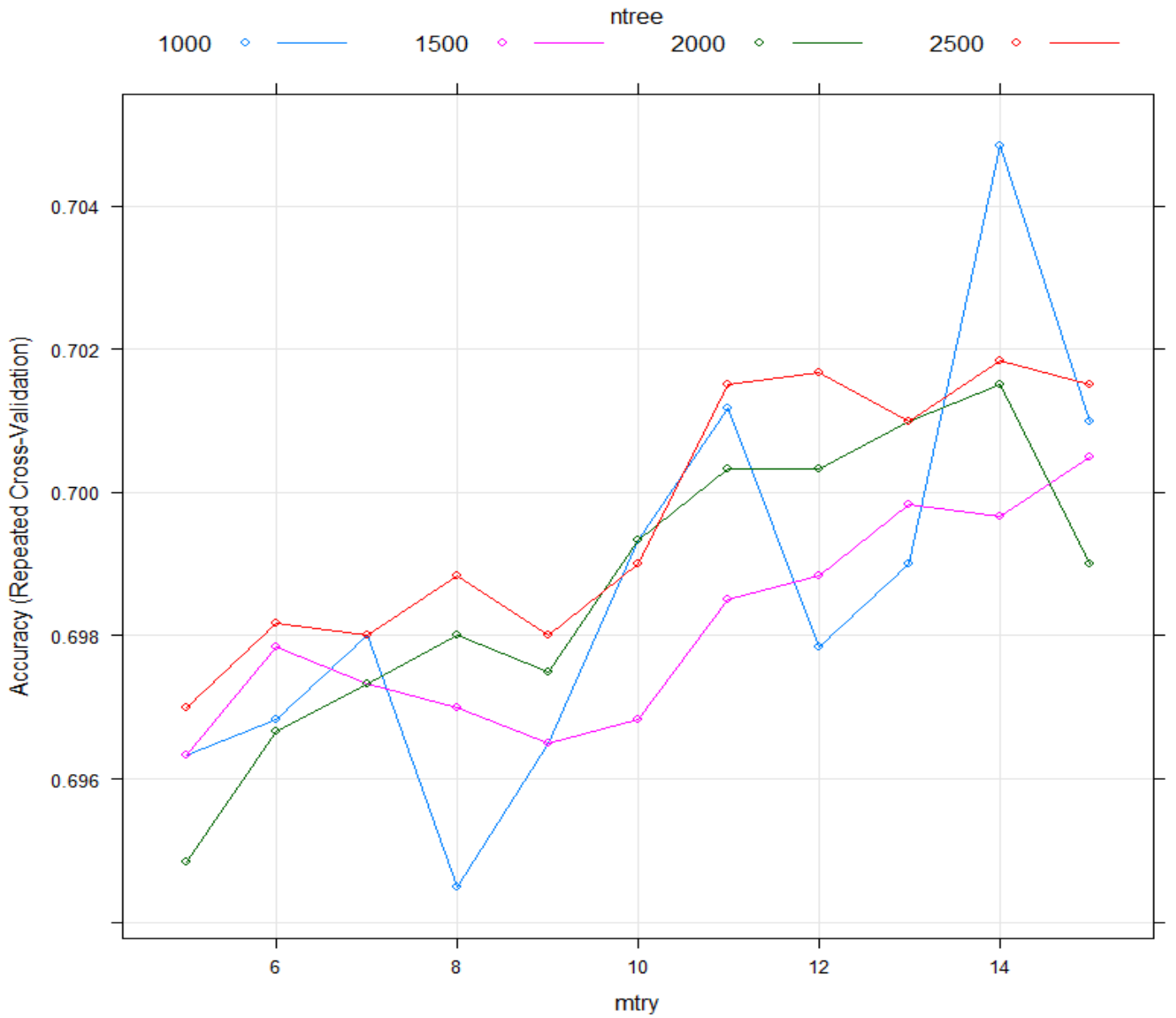
```
> cv.result[[1]]
          depth = 3 depth = 5 depth = 7 depth = 9 depth = 11
hog_gbm   0.4213333 0.4126667 0.4090000 0.4013333  0.4193333
lbp_gbm   0.3670000 0.3520000 0.3350000 0.3353333  0.3390000
sift_gbm  0.3096667 0.3103333 0.3033333 0.3066667  0.3040000
```

```
> cv.result[[2]]
             eta = 0.1 eta = 0.2 eta = 0.3 eta = 0.4 eta = 0.5
hog_xgboost  0.3693333 0.3573333 0.3650000 0.3736667 0.3796667
lbp_xgboost  0.2513333 0.2566667 0.2633333 0.2590000 0.2690000
sift_xgboost 0.2490000 0.2536667 0.2576667 0.2693333 0.2643333
```

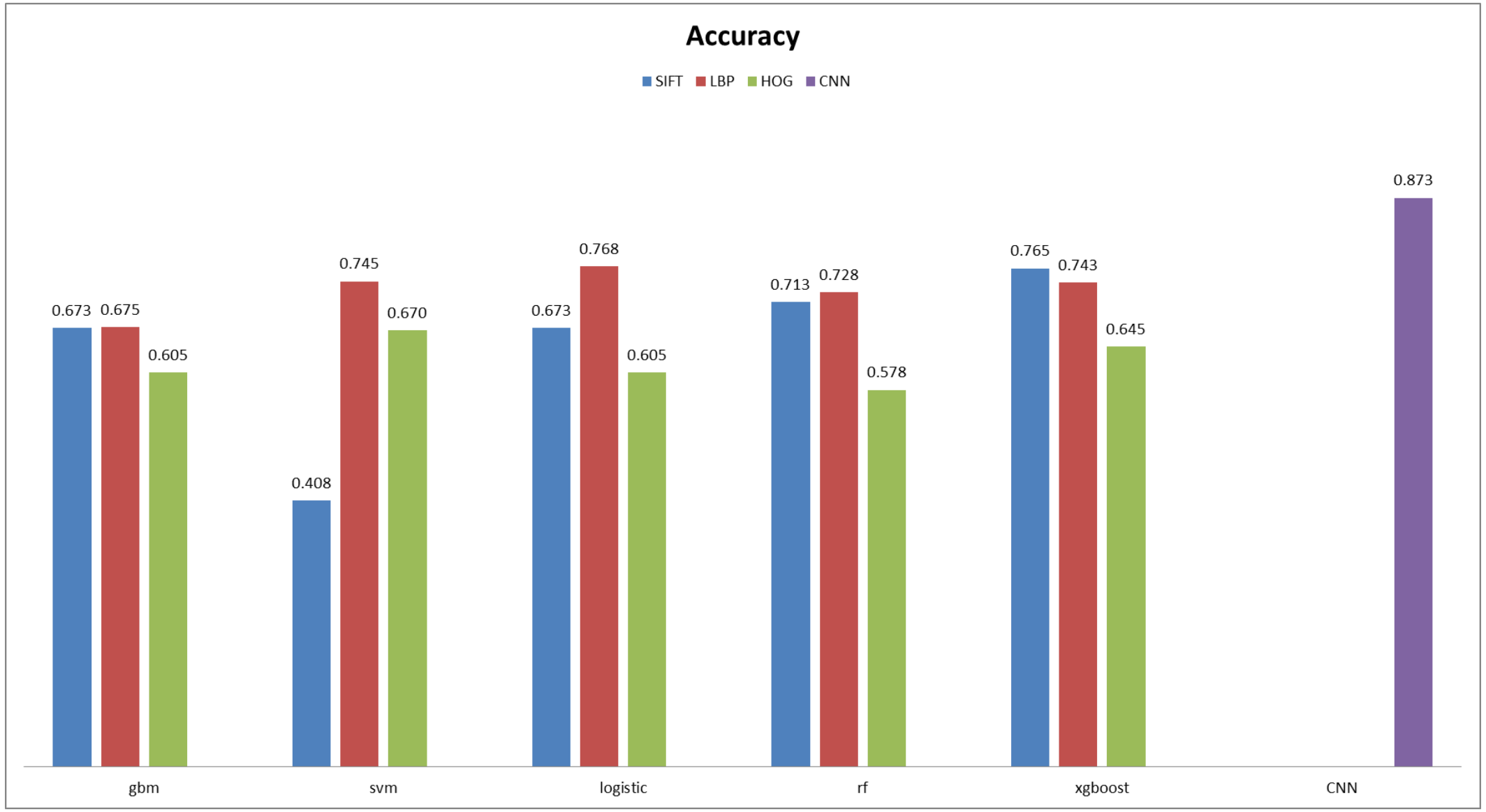
```
> cv.result[[4]]
          ntree = 1000 ntree = 1500 ntree = 2000 ntree = 2500
mtry = 5     0.3720000    0.3773333    0.3663333    0.3646667
mtry = 6     0.3750000    0.3760000    0.3586667    0.3693333
mtry = 7     0.3630000    0.3766667    0.3810000    0.3690000
mtry = 8     0.3716667    0.3823333    0.3636667    0.3630000
mtry = 9     0.3703333    0.3830000    0.3703333    0.3776667
mtry = 10    0.3713333    0.3750000    0.3860000    0.3670000
mtry = 11    0.3720000    0.3833333    0.3786667    0.3716667
mtry = 12    0.3750000    0.3810000    0.3906667    0.3670000
mtry = 13    0.3730000    0.3780000    0.3806667    0.3783333
mtry = 14    0.3730000    0.3816667    0.3780000    0.3993333
mtry = 15    0.3743333    0.3866667    0.3786667    0.3643333
```

Tune parameter

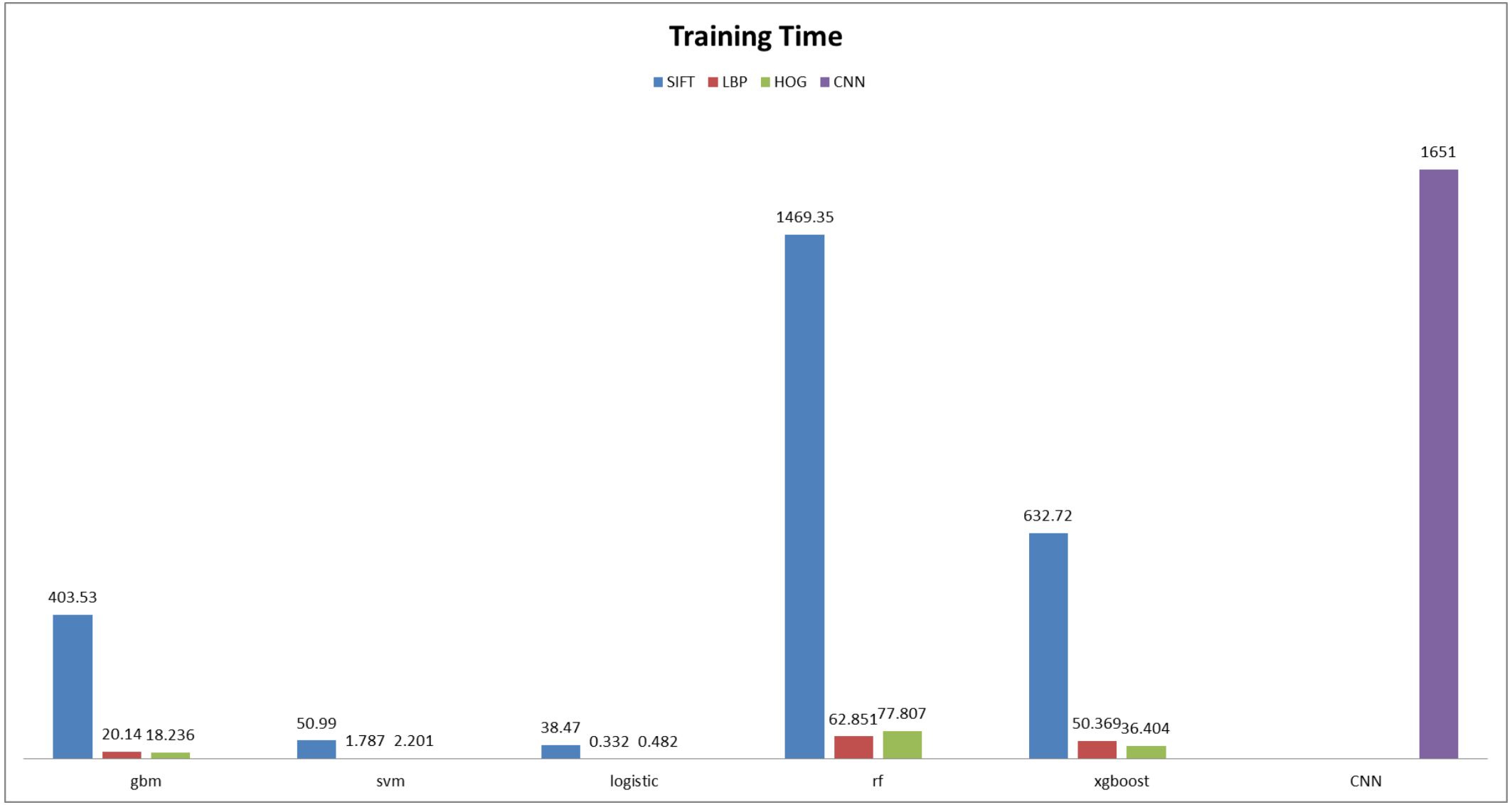
Visualize cross-validation results



Model selection



Model selection



4

PART

Outcome

Our Best Model

Local Binary Patterns
+ Logistic Regression

	Accuracy	Training Time
Baseline(SIFT+GBM)	67.3%	403.53s
Improved(LBP+Logistic)	76.8%	0.33s


THANKS