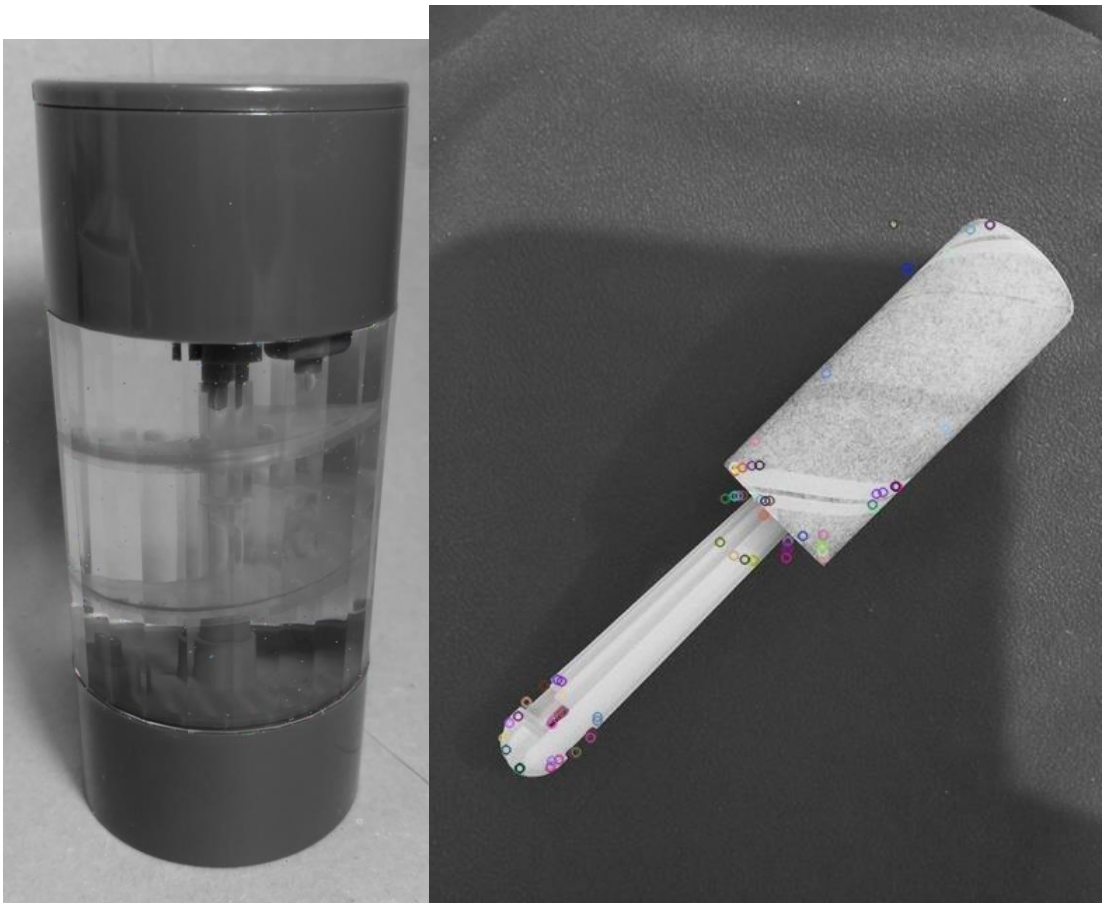
Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут ім. І. Сікорського»
Інститут Прикладного Системного Аналізу

Лабораторна робота № 3 з дисципліни “Розпізнавання образів”

Виконали:
студенти групи КА-71, КА-77
Батейко Едуард
Шепель Ірина

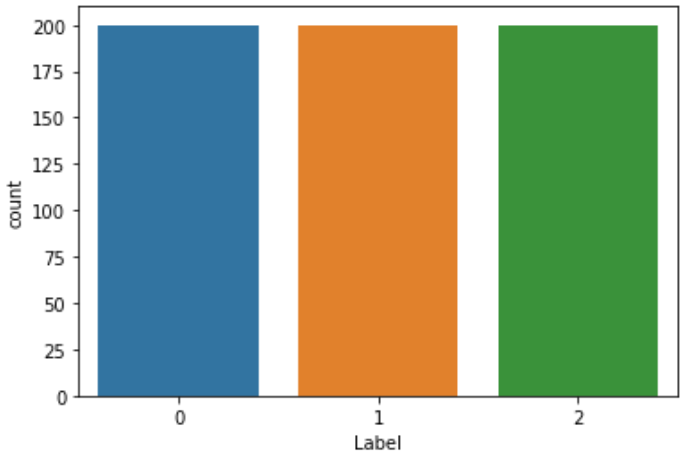
Київ 2020

Пісочний годинник та роллер для чистки одежі. Еталоні фото предметів зроблені в розширені 3006×5344.

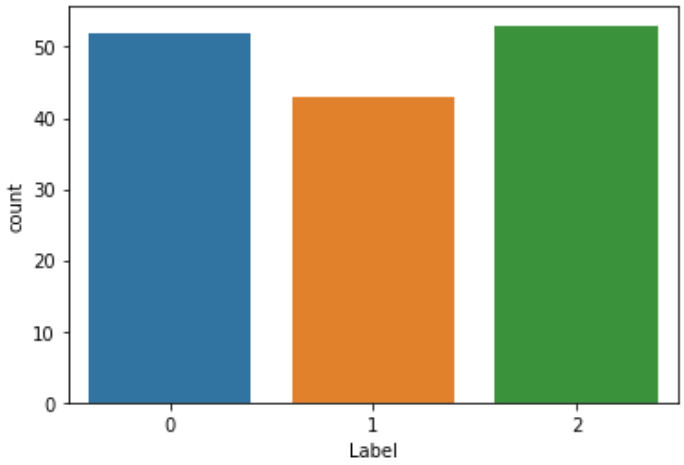


Дестриптори: AKAZE та SIFT.

Класифікатор на розгляді XGBoost, RandomForest, NaiveBayes, KNN, LR.

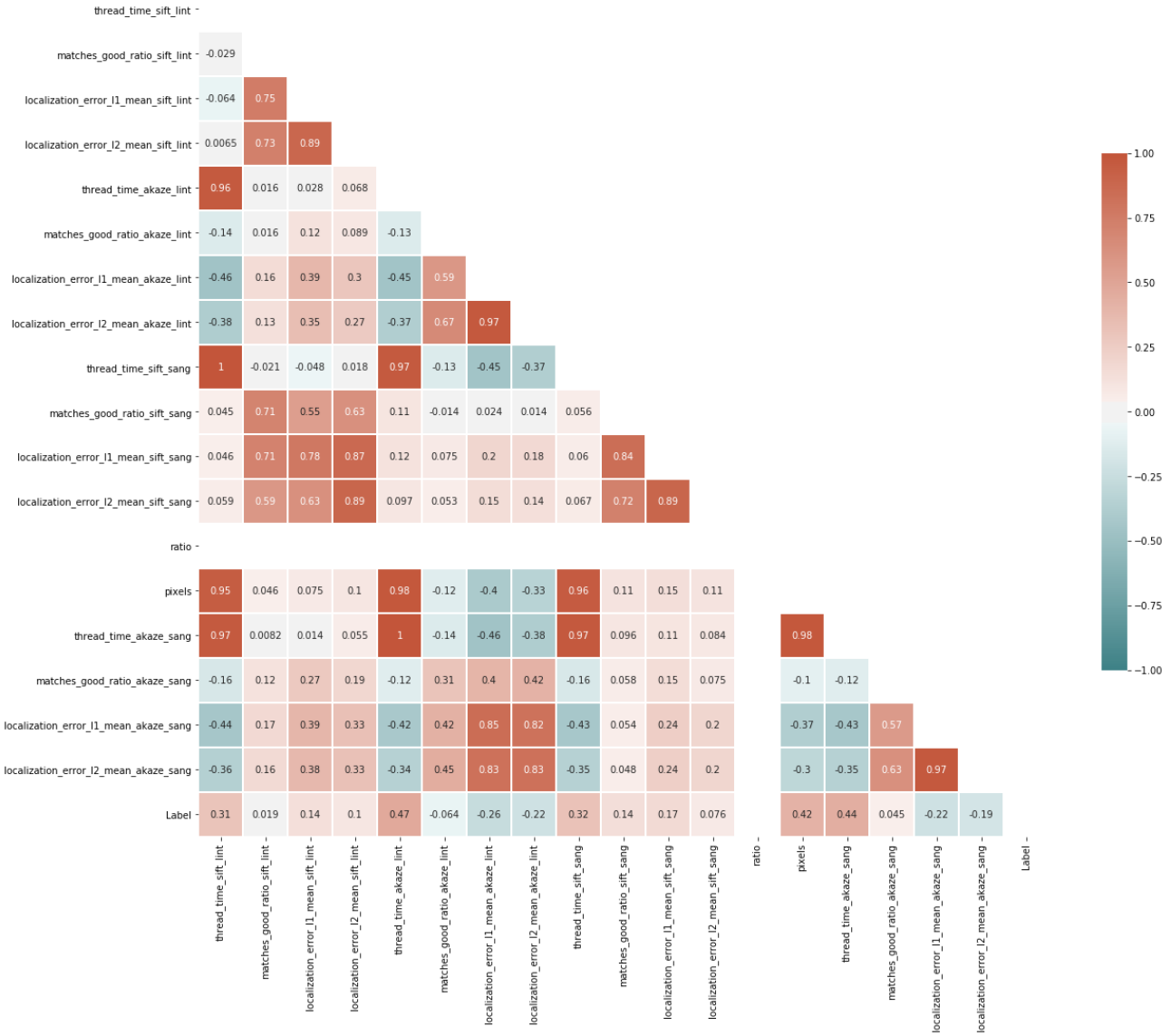


Діаграми показують, що тренувальні дані збалансовані і по кожному з класів є по 200 фото.



Тестові дані частково збалансовані, як ми бачимо.

В наступному етапі ми маємо 19 features



Features, які мали кореляцію більше 0.1 були віднесені до основних feathers в одночас між собою слабо корельовані були вибрані для подальшої роботи.

features = ['thread_time_sift_lint',

'localization_error_l1_mean_sift_lint',

'localization_error_l2_mean_sift_lint',

'thread_time_akaze_lint',

'matches_good_ratio_sift_sang',

'localization_error_l1_mean_sift_sang']

Далі провели нормалізацію виборок ОКРЕМО. Для подальшої класифікації об’єктів ми обрали такі моделі:

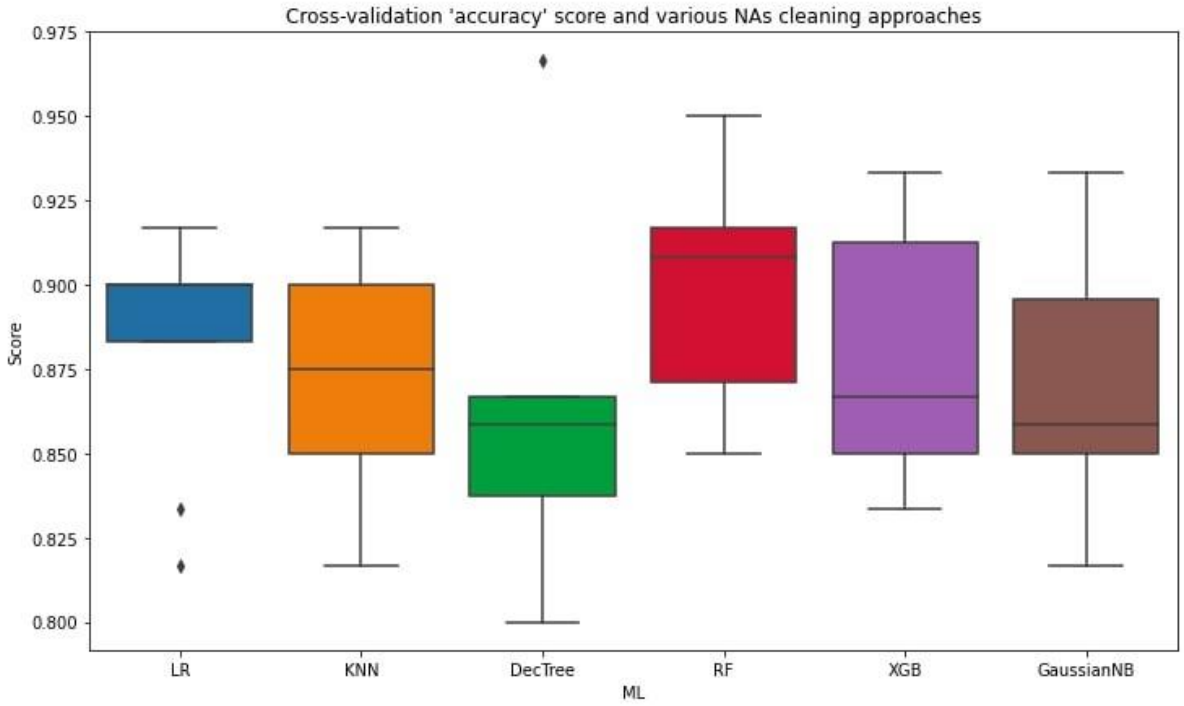
- ✓ Логістична
- ✓ Метод найближчих сусідів
- ✓ Дерево рішень
- ✓ RandomForest
- ✓ NaiveBayes
- ✓ XGBoost

Далі ми отримуємо метрики accuracy.

```
LR: 0.8833333333333334 (0.03073181485764296)
KNN: 0.8733333333333333 (0.030000000000000006)
DecTree: 0.86 (0.040960685758148346)
RF: 0.8983333333333332 (0.032871804872193364)
XGB: 0.8799999999999999 (0.035590260840104374)
GaussianNB: 0.8683333333333334 (0.03760171390892827)
```

Отримали метрики середнє квадратичне відхилення(дисперсія) accuracy.

З цього бачимо, що RandomForest і XGBoost, через те що є залежність між features.



Для подальшої роботи ми вибрали XGBoost, RandomForest, NaiveBayes.

Підібрали найкращі параметри для 3 моделей і використовуючи Greed Search Cross-Validation.

Отримали найкращі параметри для XGBoost:

```
'colsample_bytree': 0.7,
'learning_rate': 0.02,
'max_depth': 5,
'min_child_weight': 7,
'n_estimators': 20,
'nthread': 12,
'objective': 'multy:logistic',
'seed': 300,
'subsample': 0.8}
```

Найкраща accuracy на тренувальних даних 0.895

Accuracy на тестувальних даних 0.915

Precision 0.98

Recall 0,765

І confusion matrix для тестових даних

```
: array([[199,   1,   0],
         [  0, 153,  47],
         [  0,   3, 197]])
```

Обраховуємо recall для об’єктів 2,3 и precision. Отримуємо величини 0.765, 0.98.

Отримали найкращі параметри для NaiveBayes.

```
: {'criterion': 'entropy',
   'max_depth': 8,
   'max_features': 'sqrt',
   'n_estimators': 6}
```

Найкраща accuracy на тренувальних даних 0.905

Accuracy на тестувальних даних 0.947

Precision 0.97

Recall 0.865

І confusion matrix для тестових даних

```
confusion_matrix(y_test, c

array([[200,   0,   0],
       [  0, 173,  27],
       [  0,   5, 195]])
```

Отримали найкращі параметри для RandomForest.

```
: {'criterion': 'entropy',
   'max_depth': 8,
   'max_features': 'sqrt',
   'n_estimators': 6}
```

Найкраща accuracy на тренувальних даних 0.87

Accuracy на тестувальних даних 0.868

Precision 0.93

Recall 0.685

І confusion matrix для тестових даних

```
confusion_matrix(y_test, clf3.

array([[194,   6,   0],
       [  0, 137,  63],
       [  0,  10, 190]])
```

Для покращення наших моделей ми робимо ансамбль с 3 моделей(RandomForest, NaiveBayes, XGBoost) і з 2 (XGBoost і RandomForest).

Спочатку натренували кожну модель на найкращих параметрах, які нам дав жадібний пошук.

Усереднені результати моделей:

```
[278]: print('Accuracy score:', accuracy_score(y_test, pred_averaged))
       print('Confusion matrix: \n', confusion_matrix(y_test, pred_averaged))
```

```
Accuracy score: 0.9083333333333333
Confusion matrix:
 [[199   1   0]
 [  0 146  54]
 [  0   0 200]]
```

```
[279]: print(classification_report(y_test, pred_averaged))
```

	precision	recall	f1-score	support
0	1.00	0.99	1.00	200
1	0.99	0.73	0.84	200
2	0.79	1.00	0.88	200
accuracy			0.91	600
macro avg	0.93	0.91	0.91	600
weighted avg	0.93	0.91	0.91	600

Далі використовуємо отримуємо результати для ансамбля з 3 моделей.

```
Accuracy score 0.885
Confusion matrix
 [[199   1   0]
 [  0 136  64]
 [  0   4 196]]
```

Метрики для ансамблю 2 моделей

```
Accuracy score 0.9383333333333334
Confusion matrix
 [[200   0   0]
 [  0 168  32]
 [  0   5 195]]
```

```
8]: print(classification_report(y_test, pred_vc))
```

	precision	recall	f1-score	support
0	1.00	1.00	1.00	200
1	0.97	0.84	0.90	200
2	0.86	0.97	0.91	200
accuracy			0.94	600
macro avg	0.94	0.94	0.94	600
weighted avg	0.94	0.94	0.94	600

Отримали найкращі результати на ансамблі з 2 моделей((XGBoost і RandomForest) 0.938.

Записали відео з тривалістю на 14 секунд та зберегли його.

Розділили на кадри з проміжком пів-секунди отримали 29 кадрів. Для цих картинок обрахували метрики, щоб подати на готову модель. Виділили лише ті features, які ми вибрали вище. Вибрали 2 види нормалізації: за допомогою математичного сподівання та дисперсії з даних, для яких хочемо зробити класифікацію.

:

	file_name	prediction	class
0	10	1	1
1	11	0	1
2	13	1	2
3	14	0	2
4	15	2	2
5	16	2	2
6	17	2	2
7	18	0	2
8	19	0	2
9	2	1	1
10	20	0	2
11	21	0	2
12	22	2	2
13	23	0	2
14	24	0	2
15	25	0	2
16	26	0	2
17	27	0	2
18	28	2	2
19	29	2	2
20	3	1	1
21	30	2	2
22	31	1	2
23	4	1	1
24	5	0	1
25	6	0	1
26	7	1	1
27	8	1	1
28	9	1	1

Prediction для нормалізації: за допомогою математичного сподівання та дисперсії з тренувальних даних.

	file_name	predictions	class
0	10	2	1
1	11	2	1
2	13	2	2
3	14	2	2
4	15	2	2
5	16	2	2
6	17	2	2
7	18	2	2
8	19	2	2
9	2	2	1
10	20	2	2
11	21	2	2
12	22	2	2
13	23	2	2
14	24	2	2
15	25	2	2
16	26	2	2
17	27	2	2
18	28	2	2
19	29	2	2
20	3	2	1
21	30	2	2
22	31	2	2
23	4	2	1
24	5	2	1
25	6	2	1
26	7	2	1
27	8	2	1
28	9	2	1

Бачимо, що він не відповідає дійсності и не може бути використано.

Натренувавши три моделі які дали найкращий результат вирішили використати ансамбль з них. Оскільки, метод ансамблю об’єднує властивості моделей та зазвичай дає кращий результат. Брали ансамбль з трьох моделей (XGBoost, RandomForest, NaiveBayes), який дав 0.908333 точність. Далі

обрали ансамбль з XGBoost, RandomForest оскільки моделі мали високу точність 0.89 та 0.90, після ансамблю отримали 0.9383, що показує кращі результати ніж кожна модель окремо.

```
#XGBClassifier
%%time
clf4 = clf4.fit(X_train, y_train)
```

```
CPU times: user 695 ms, sys: 23.1 ms, total: 718 ms
Wall time: 77.2 ms
```

```
# RandomForestClassifier
%%time
clf5 = clf5.fit(X_train, y_train)
```

```
CPU times: user 19 ms, sys: 8.71 ms, total: 27.7 ms
Wall time: 107 ms
```

```
# GaussianNB
%%time
clf6 = clf6.fit(X_train, y_train)
```

```
CPU times: user 3.91 ms, sys: 0 ns, total: 3.91 ms
Wall time: 2.8 ms
```

Час використаний для тренування найкращих моделей з їх найкращими параметрами, які обиралися через крос-валідацію.

Найкращий час тренування дала модель NaiveBayes, далі йде XGBoost, RandomForest. Що й дивно, оскільки NaiveBayes -- це проста модель, порівняно з бустинговими моделями. RandomForest повільніший ніж XGBoost, бо XGBoost тренувався на GPU, а RandomForest на CPU. А GPU має набагато кращу обчислювальну потужність.

Висновок:

було розглянуто 6 типів алгоритмів класифікації, таких як LogisticRegression,KneighborsClassifier,DecisionTreeClassifier, RandomForestClassifier,XGBClassifier,GaussianNB. Натренували моделі без параметрів, після цього обрали які в середньому давали кращі значення(XGBoost, RandomForest, NaiveBayes)). Підібрали для них параметри, об’єднали. Записали відео, через яке пропустили наш класифікатор, та прокласифікували кожну секунду в ньому. Тточність отримали не таку ж як на тестувальній вибірці, оскільки був інший фон різке зміщення картинки, проте навіть тоді коли роллер злтвався із фоном класифікатор розпізнав його 4 з 6 разів.