ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
**«САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО»**

Институт компьютерных наук и технологий

**Высшая школа программной инженерии**

# ЛАБОРАТОРНАЯ РАБОТА №4

по дисциплине «Машинное обучение»

Студент А. М. Потапова
гр. 3530202/90202

Руководитель И. А. Селин

Санкт-Петербург
2022 г

# Содержание

## Задание 1

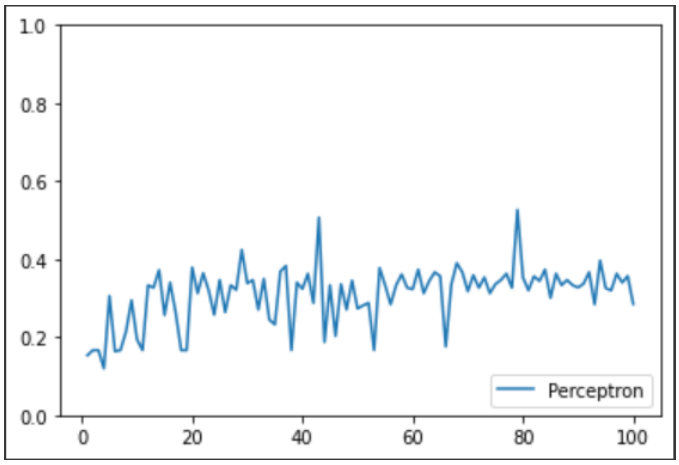
Исследуйте зависимость качества классификации от количества классификаторов в ансамбле для алгоритмов бэггинга на наборе данных glass.csv с различными базовыми классификаторами. Постройте графики зависимости качества классификации при различном числе классификаторов, объясните полученные результаты.

*Исходные данные:*

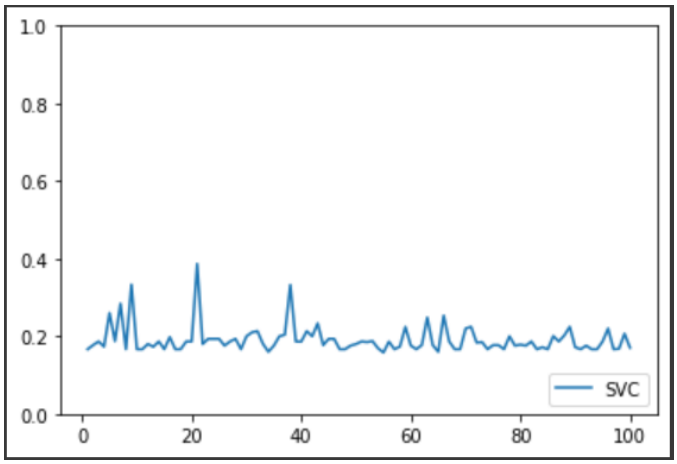
| | RI | Na | Mg | Al | Si | K | Ca | Ba | Fe | Type |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1.52101 | 13.64 | 4.49 | 1.10 | 71.78 | 0.06 | 8.75 | 0.0 | 0.0 | 1 |
| 1 | 1.51761 | 13.89 | 3.60 | 1.36 | 72.73 | 0.48 | 7.83 | 0.0 | 0.0 | 1 |
| 2 | 1.51618 | 13.53 | 3.55 | 1.54 | 72.99 | 0.39 | 7.78 | 0.0 | 0.0 | 1 |
| 3 | 1.51766 | 13.21 | 3.69 | 1.29 | 72.61 | 0.57 | 8.22 | 0.0 | 0.0 | 1 |
| 4 | 1.51742 | 13.27 | 3.62 | 1.24 | 73.08 | 0.55 | 8.07 | 0.0 | 0.0 | 1 |

Алгоритм бэггинга – BaggingClassifier
Базовые классификаторы – Perceptron, SVC и DecisionTreeClassifier
Метрика классификатора – balanced_accuracy_score

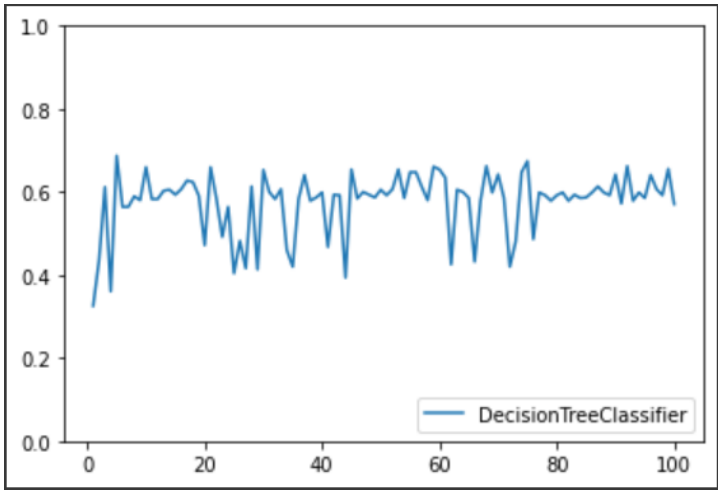
- Используем классификатор Perceptron. Зависимость значения метрики от количества классификаторов:



- Используем классификатор SVC. График зависимости:




- Используем классификатор DecisionTreeClassifier. График зависимости:




**Вывод**

Исходя из полученных результатов можно отметить, что среди этих 3 классификаторов наилучшие значения метрики balanced_accuracy_score показал классификатор DecisionTreeClassifier, т. к. его качество классификации в среднем равна 0.6. Для него же наблюдаем периодический рост качества классификации в виде часто встречающихся пиков. При использовании алгоритмов бэггинга с ростом числа классификаторов видим несущественный рост качества классификации для базового классификатора Perceptron. У SVC не видим существенного роста качества классификации, а лишь довольно редкие пики.

## Задание 2

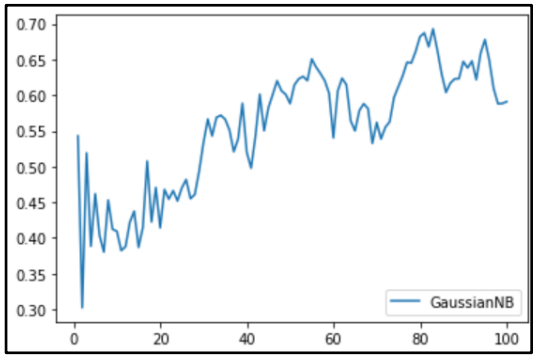
Исследуйте зависимость качества классификации от количества классификаторов в ансамбле для алгоритма бустинга (например, AdaBoost) на наборе данных vehicle.csv с различными базовыми классификаторами. Постройте графики зависимости качества классификации при различном числе классификаторов, объясните полученные результаты.

*Исходные данные:*

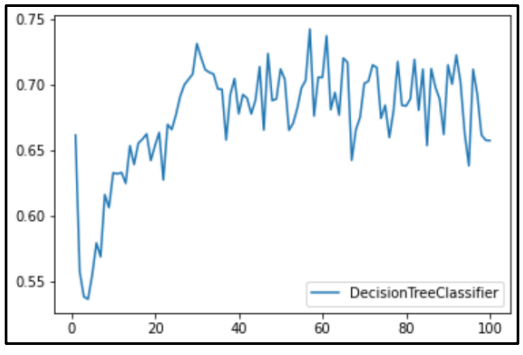
| | Comp | Circ | D.Circ | Rad.Ra | Pr.Axis.Ra | Max.L.Ra | Scat.Ra | Elong | Pr.Axis.Rect | Max.L.Rect | Sc.Var.Maxis | Sc.Var.maxis | Ra.Gyr | Skew.Maxis | Skew.maxis | Kurt.maxis | Kurt.Maxis | Holl.Ra | Class |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 95 | 48 | 83 | 178 | 72 | 10 | 162 | 42 | 20 | 159 | 176 | 379 | 184 | 70 | 6 | 16 | 187 | 197 | van |
| 1 | 91 | 41 | 84 | 141 | 57 | 9 | 149 | 45 | 19 | 143 | 170 | 330 | 158 | 72 | 9 | 14 | 189 | 199 | van |
| 2 | 104 | 50 | 106 | 209 | 66 | 10 | 207 | 32 | 23 | 158 | 223 | 635 | 220 | 73 | 14 | 9 | 188 | 196 | saab |
| 3 | 93 | 41 | 82 | 159 | 63 | 9 | 144 | 46 | 19 | 143 | 160 | 309 | 127 | 63 | 6 | 10 | 199 | 207 | van |
| 4 | 85 | 44 | 70 | 205 | 103 | 52 | 149 | 45 | 19 | 144 | 241 | 325 | 188 | 127 | 9 | 11 | 180 | 183 | bus |

Алгоритм бустинга – AdaBoostClassifier
Базовые классификаторы – GaussianNB и DecisionTreeClassifier
Метрика классификатора – balanced_accuracy_score

- Используем классификатор GaussianNB. Зависимость значения метрики от количества классификаторов:




- Используем классификатор DecisionTreeClassifier. График зависимости:



**Вывод**

В обоих случаях видим сначала резкий спад качества классификации, а затем значительный рост. Базовый классификатор DecisionTreeClassifier оказался немного лучше базового классификатора GaussianNB, т. к. качество классификации у него выше (больше 0.7). Оба случая показали достаточно большие значения метрики balanced_accuracy_score.

## Задание 3

Постройте мета-классификатор для набора данных titanic_train.csv используя стекинг и оцените качество классификации на titanic_train.csv

*Исходные данные:*

| | PassengerId | Survived | Pclass | Name | Sex | Age | SibSp | Parch | Ticket | Fare | Cabin | Embarked |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 0 | 3 | Braund, Mr. Owen Harris | male | 22.0 | 1 | 0 | A/5 21171 | 7.2500 | NaN | S |
| 1 | 2 | 1 | 1 | Cumings, Mrs. John Bradley (Florence Briggs Th... | female | 38.0 | 1 | 0 | PC 17599 | 71.2833 | C85 | C |
| 2 | 3 | 1 | 3 | Heikkinen, Miss. Laina | female | 26.0 | 0 | 0 | STON/O2. 3101282 | 7.9250 | NaN | S |
| 3 | 4 | 1 | 1 | Futrelle, Mrs. Jacques Heath (Lily May Peel) | female | 35.0 | 1 | 0 | 113803 | 53.1000 | C123 | S |
| 4 | 5 | 0 | 3 | Allen, Mr. William Henry | male | 35.0 | 0 | 0 | 373450 | 8.0500 | NaN | S |

Базовые классификаторы – GaussianNB, SVC, KNeighborsClassifier и DecisionTreeClassifier

- Используем классификатор GaussianNB. Полученные значения метрики balanced_accuracy_score:

```
gaussian = GaussianNB()
gaussian.fit(X_train, y_train)
print('Balanced accuracy score for GaussianNB:', balanced_accuracy_score(y_test, gaussian.predict(X_test)))
```

```
Balanced accuracy score for GaussianNB: 0.7659273545483767
```

*Результат для GaussianNB: 0.7659273545483767*

- Используем классификатор SVC. Полученные значения метрики balanced_accuracy_score:

```
svc = SVC(probability=True)
grid_search_cv_svc = GridSearchCV(svc, {'C': range(1, 500)}, cv=5, n_jobs=-1)
grid_search_cv_svc.fit(X_train, y_train)
print('Best params for SVC:', grid_search_cv_svc.best_params_)
best_svc = grid_search_cv_svc.best_estimator_
print('Balanced accuracy score for SVC:', balanced_accuracy_score(y_test, best_svc.predict(X_test)))
```

```
Best params for SVC: {'C': 311}
Balanced accuracy score for SVC: 0.7524911603985857
```

*Результат для SVC: 0.7524911603985857*

- Используем классификатор KNeighborsClassifier. Полученные значения метрики balanced_accuracy_score:

```
kNN = KNeighborsClassifier()
parameters = {'n_neighbors': range(1, 30),
              'metric': ['euclidean', 'manhattan', 'chebyshev']}
grid_search_cv_kNN = GridSearchCV(kNN, parameters, cv=5, n_jobs=-1)
grid_search_cv_kNN.fit(X_train, y_train)
print('Best params for KNN :', grid_search_cv_kNN.best_params_)
best_kNN = grid_search_cv_kNN.best_estimator_
print('Balanced accuracy score for KNeighborsClassifier:', balanced_accuracy_score(y_test, best_kNN.predict(X_test)))
```

```
Best params for KNN : {'metric': 'manhattan', 'n_neighbors': 7}
Balanced accuracy score for KNeighborsClassifier: 0.7482802957248473
```

*Результат для KNeighborsClassifier: 0.7482802957248473*

- Используем классификатор DecisionTreeClassifier. Полученные значения метрики balanced_accuracy_score:

```
tree = DecisionTreeClassifier()
parameters = {'criterion': ['gini', 'entropy'],
              'max_depth': range(1, 20),
              'min_samples_leaf': range(1, 20),
              'min_samples_split': range(1, 20)}
grid_search_cv_tree = GridSearchCV(tree, parameters, cv=5, n_jobs=-1)
grid_search_cv_tree.fit(X_train, y_train)
print('Best params for DecisionTree:', grid_search_cv_tree.best_params_)
best_tree = grid_search_cv_tree.best_estimator_
print('Balanced accuracy score for DecisionTreeClassifier:', balanced_accuracy_score(y_test, best_tree.predict(X_test)))
```

```
Best params for DecisionTree: {'criterion': 'entropy', 'max_depth': 6, 'min_samples_leaf': 4, 'min_samples_split': 16}
Balanced accuracy score for DecisionTreeClassifier: 0.7151076824172292
```

*Результат для DecisionTreeClassifier: 0.7151076824172292*

- Теперь мы можем узнать значение balanced_accuracy_score для мета-классификатора:

```
best_score = 0

for ratio in np.arange(0.25, 1, 0.01):
    slr = StackedClassifier(ratio=float(ratio), estimators=estimators)
    slr.fit(X_train, y_train)
    slr_predictions = slr.predict(X_test)
    score = balanced_accuracy_score(y_test, slr_predictions)
    if (best_score < score):
        best_score = score

print('Balanced accuracy score for meta-classifier:', best_score)
```

```
Balanced accuracy score for meta-classifier: 0.803953712632594
```

*Результат для мета-классификатора: 0.803953712632594*

**Вывод**

Исходя из полученного результата можно отметить, что качество классификации при построении мета-классификатора, используя стекинг, возросло. При использовании отдельных классификаторов среднее значение balanced_accuracy_score составило 0,7454, а в случае с мета-классификатором – 0,8.