СЕКЦІЯ «ІНФОРМАТИКА»

УДК 004.85:519.6

DOI 10.31651/2076-5886-2019-1-42-53

PACS 02.70.Wz, 07.05.Kf, 07.05.Mh, 07.05.Tp

ПІСКУН Олександр Варфоломійович,
к. т. н., доцент, доцент кафедри автоматизації та комп'ютерно-інтегрованих технологій, Черкаський національний університет імені Богдана Хмельницького
e-mail: piskun@ukr.net
ORCID 0000-0001-5334-6337

ЗАСТОСУВАННЯ МЕТОДІВ МАШИННОГО НАВЧАННЯ ДЛЯ ПОБУДОВИ МОДЕЛІ РІШЕННЯ ЗАДАЧІ КЛАСИФІКАЦІЇ

У роботі наведено алгоритм побудови моделі рішення задачі класифікації щодо визначення наявності або відсутності захворювань серця у людини на основі методів машинного навчання. Попередня обробка та аналіз даних дали можливість виявити нелінійну залежність цільової змінної, а також підготувати дані для ефективного моделювання. Для побудови моделі класифікатора застосований метод SVM з rbf ядром. Оптимізація параметрів моделі проведена, використовуючи пошук по сітці з крос-валідацією кожної комбінації параметрів.

Ключові слова: *машинне навчання, класифікація, метод опорних векторів, інтелектуальна обробка даних, t-SNE візуалізація.*

Постановка проблеми

Задача класифікації - одна з найбільш поширених задач в аналізі даних і розпізнаванні образів. Вона має широку область практичних застосувань в автоматиці, управлінні, економіці, соціології, медицині, геології, астрономії, ядерній фізиці і т.д.

Класифікація – це один із розділів машинного навчання, що вирішує наступну задачу: нехай маємо множину об'єктів, що розділені на класи за деякою або рядом ознак. Окрім цього, задана скінчена множина об'єктів, щодо яких відомо, яким саме класам вони належать. Цю множину називають навчальною вибіркою. До якого класу відносяться решта об'єктів – невідомо. Необхідно побудувати алгоритм, що буде здатний класифікувати будь-який з об'єктів вихідної множини. Під класифікацією об'єкта розуміють відповідь алгоритму, що вказує номер (або назву) класу, до якого він відноситься [1].

Виклад основного матеріалу

Розглянемо найбільш поширені алгоритми класифікації даних [2].

Дерево рішень. Принцип даного методу полягає в рекурсивному розбитті множини елементів на підмножини, що містять елементи, які належать одному класу. Різні різновиди дерев рішень відрізняються знаходженням цільової змінної та її атрибутів для побудови нового вузла.

Найбільш поширеним і успішним алгоритмом є алгоритм, в якому вибір наступної змінної та її атрибутів ґрунтується на визначенні взаємозв'язку між ентропією та класифікованої інформацією. Вибір зупиняється на елементах, з мінімальним значенням ентропії і максимальним значенням інформативності відповідно.

Байєсова класифікація. Алгоритм базується на теоремі Байеса та передбачає незалежність розглянутих ознак. Метод використовує ймовірності, отримані в ході навчання за тестовою вибіркою, а саме апріорні ймовірності належності будь-якого з розглянутих об'єктів даного класу та умовні ймовірності того, що об'єкт, який належить до даного класу, має даний набір ознак. Потім визначається ймовірність належності даного об'єкту з тестового набору до кожного з класів. Вибирається змінна з найвищою ймовірністю.

Метод k найближчих сусідів. Метод передбачає, що черговий об'єкт з тестового набору належить тому класу, до якого належать більшість його найближчих «сусідів» з навчальної вибірки. Сусідами, в даному випадку, називають найбільш близькі до тестового об'єкту спостереження. Таким чином, новий об'єкт поміщається в клас, що містить найбільшу кількість найближчої схожості. Параметр k, що задається, визначає кількість розглянутих найближчих сусідів, тобто визначає скільки сусідів буде впливати на класифікацію. Близькість визначається «відстанню» між об'єктами.

Метод опорних векторів. Даний метод передбачає використання гиперплощини для поділу простору на підпростори, що характеризують окремі класи. Об'єкти, які лежать при цьому на кордоні освічених областей, називають опорними векторами. Метод намагається провести дане розбиття таким чином, щоб відстань між елементами різних класів була максимальною. Щодо вибору розділяючої гиперплощини, це означає її максимальне віддалення щодо найближчих точок обох класів. У разі виникнення проблеми лінійної нероздільності, використовується перетворення ядра.

Метою даної роботи є покроковий виклад та застосування на практиці алгоритму побудови моделі для вирішення задачі класифікації.

Види статистичних даних [3]

Статистичні дані можуть бути представлені як кількісними (числовими неперервними або дискретними), так і якісними (категоріальними порядковими або номінальними) змінними.

Кількісні (числові) дані – це дані, які приймають деякі числові значення. Вони, в свою чергу, поділяються на: дискретні дані, які можуть приймати строго визначені значення, та неперервні, які можуть бути представлені будь-якими значеннями.

Категоріальні дані застосовуються для опису стану об'єкта шляхом присвоєння йому номера, відповідного до категорії, куди цей об'єкт належить. Важливою умовою для застосування категоріальних даних є приналежність одного об'єкта дослідження тільки до однієї можливої категорії для одного критерію.

Якісні номінальні дані використовуються в тому випадку, якщо категорії не впорядковані. Числа в даному випадку є лише позначенням для стану об'єкта і не впорядковують цей стан. Наприклад, по статі: 1 – чоловіча, 2 – жіноча.

Якісні порядкові (рангові, ординарні) дані – це дані, для яких категорії можуть бути впорядковані. Наприклад, від поганого самопочуття до гарного: 1 – гарне, 2 – задовільне, 3 – погане.

Опис набору даних

В якості експериментальних даних для дослідження використовується набір даних (датасет) з серцевих захворювань Statlog (Heart) з репозиторію UCI ML [4]. Дані містять 13 атрибутів (табл. 1) і 180 спостережень.

Змінна для прогнозування **y** може приймати два значення: 0 – відсутність, 1 – наявність серцевих захворювань.

Дослідження проводимо з використанням бібліотеки машинного навчання Scikit-Learn, в середовищі Python 3.

Таблиця 1

Атрибути експериментальних даних з серцевих захворювань

№	Назва	Опис	
1	age	вік в роках	
2	sex	стать	1 = чоловіча
			0 = жіноча
3	chest pain type (4 values)	тип болю в грудях (4 значення: 1, 2, 3, 4)	
4	resting blood pressure	артеріальний тиск у спокої (мм рт. ст.)	
5	serum cholestoral in mg/dl	рівень холестерину в крові в мг/дл	
6	fasting blood sugar > 120 mg/dl	рівень цукру в крові > 120 мг/дл	1 = так
			0 = ні
7	resting electrocardiographic results (values 0, 1, 2)	електрокардіографічні результати в стані спокою (значення 0, 1, 2)	
8	maximum heart rate achieved	максимальна частота серцевих скорочень	
9	exercise induced angina	стенокардія, викликана фізичними вправами	1 = так
			0 = ні
10	oldpeak = ST depression induced by exercise relative to rest	пониження сегменту ST на ЕКГ, викликане фізичними вправами відносно спокою	
11	slope of the peak exercise ST segment	нахил пікового сегмента ST	1 = зростає
			2 = фіксований
			3 = убуває
12	number of major vessels (0-3) colored by fluoroscopy	кількість основних судин (0-3), забарвлених при флюороскопії	
13	thal: 3 = normal; 6 = fixed defect; 7 = reversable defect	тип дефекту	3 = в межах норми
			6 = фіксований дефект
			7 = оборотний дефект

Підготовка та аналіз даних

Завантажуємо дані про захворювання серця. Дані мають наступний вигляд (перші 5 записів):

```
In [123]: X.head()
Out[123]:
           slope_of_peak_exercise_st_segment              thal  \
patient_id
0z64un                                     1            normal
ryoo3j                                     2            normal
yt1s1x                                     1            normal
l2xjde                                     1  reversible_defect
oyt4ek                                     3  reversible_defect
           resting_blood_pressure  chest_pain_type  num_major_vessels  \
patient_id
0z64un                        128                2                  0
ryoo3j                        110                3                  0
yt1s1x                        125                4                  3
l2xjde                        152                4                  0
oyt4ek                        178                1                  0
           fasting_blood_sugar_gt_120_mg_per_dl  resting_ekg_results  \
patient_id
0z64un                                        0                    2
ryoo3j                                        0                    0
yt1s1x                                        0                    2
l2xjde                                        0                    0
oyt4ek                                        0                    2
```

```
            serum_cholesterol_mg_per_dl  oldpeak_eq_st_depression  sex  age  \
patient_id
0z64un                              308                       0.0    1   45
ryoo3j                              214                       1.6    0   54
yt1s1x                              304                       0.0    1   77
l2xjde                              223                       0.0    1   40
oyt4ek                              270                       4.2    1   59
            max_heart_rate_achieved  exercise_induced_angina
patient_id
0z64un                          170                        0
ryoo3j                          158                        0
yt1s1x                          162                        1
l2xjde                          181                        0
oyt4ek                          145                        0
```

Перевіряємо наші дані на наявність пропущених значень:

```
In [125]: X.isnull().sum()
Out[125]:
slope_of_peak_exercise_st_segment     0
thal                                  0
resting_blood_pressure                0
chest_pain_type                       0
num_major_vessels                     0
fasting_blood_sugar_gt_120_mg_per_dl  0
resting_ekg_results                   0
serum_cholesterol_mg_per_dl           0
oldpeak_eq_st_depression              0
sex                                   0
age                                   0
max_heart_rate_achieved               0
exercise_induced_angina               0
dtype: int64
```

Пропусків немає.

Проводимо перевірку наявності дисбалансу цільових класів (функція countplot, рис. 1).

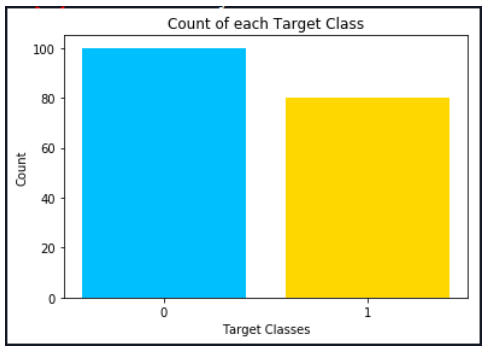

Рис. 1

Класи можна вважати збалансованими (об’єктів класу «0» - 100, «1» - 80).

Розглянемо, наскільки відрізняються значення атрибутів для цільових класів. Для цього побудуємо діаграму розсіювання для кожного з атрибутів за допомогою функції swarmplot, в якій точки коригуються уздовж категорійної осі так, щоб вони не перекривалися. Це дає кращу картину розподілу значень:

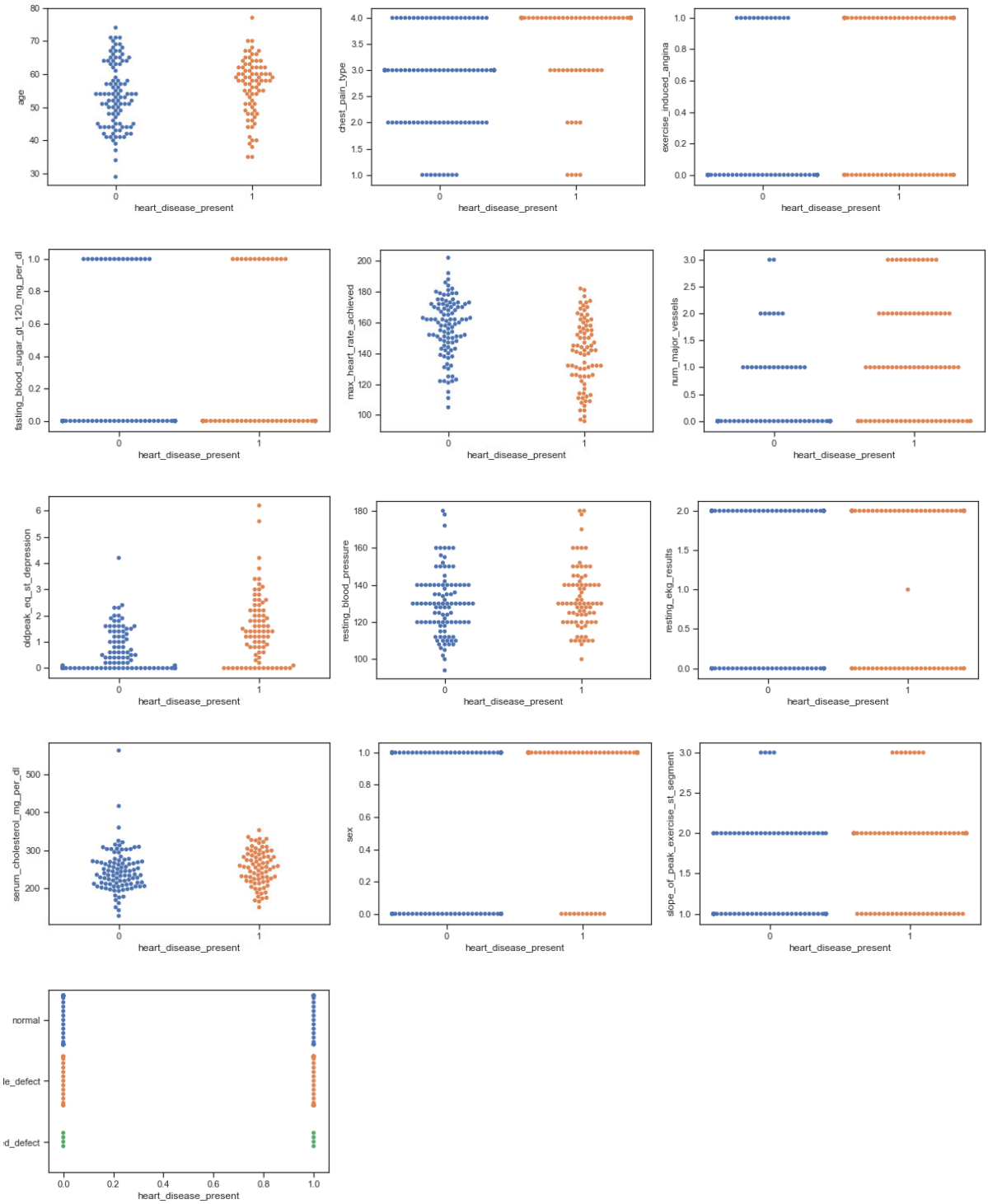

Рис. 2

Як видно з діаграм, незважаючи на присутність деяких відмінностей в значеннях атрибутів у різних цільових груп, ми не можемо чітко їх розмежувати. Це вказує на необхідність застосування класифікатора для побудови моделі.

Виконаємо обробку даних: перетворимо категоріальні дані у бінарні, нормалізуємо усі числові до одного діапазону [0..1].

```
# Data Preprocessing

## Handle Categorical features
def Preprocessing_categorical(X):

    import pandas as pd

    trainThal = pd.get_dummies(X['thal'])
    trainSlope = pd.get_dummies(X['slope_of_peak_exercise_st_segment'], prefix='slope')
    trainChestPain = pd.get_dummies(X['chest_pain_type'], prefix='chestPain')
    trainResting = pd.get_dummies(X['resting_ekg_results'], prefix='restingEkg')
    trainnum = pd.get_dummies(X['num_major_vessels'], prefix='num_major_vessels')

    X.drop(['thal', 'slope_of_peak_exercise_st_segment','chest_pain_type','resting_ekg_results', 'num_major_vessels'], axis=1, inplace=True)
    X = X.join([trainThal, trainSlope, trainChestPain, trainResting, trainnum])

    del trainThal, trainSlope, trainChestPain, trainResting, trainnum

    return (X)


## Normalization of numerical features
def Preprocessing_numerical(X):

    from sklearn.preprocessing import StandardScaler

    col_names = ['resting_blood_pressure', 'serum_cholesterol_mg_per_dl', 'oldpeak_eq_st_depression', 'age', 'max_heart_rate_achieved']
    features = X[col_names]
    scaler = StandardScaler().fit(features.values)
    X[col_names] = scaler.transform(features.values)

    return (X)
```

Проведемо t-SNE візуалізацію змінної **y**. T-Distributed Stochastic Neighbor Embedding (t-SNE) - це метод нелінійного зменшення розмірності, який добре підходить для візуалізації багатовимірних наборів даних [5]. Метод моделює кожен об'єкт простору високої розмірності дво- або тривимірною точкою таким чином, що близькі за характеристиками елементи даних в багатовимірному просторі (наприклад, датасета з великим числом стовпців) проектуються в сусідні точки, а різнорідні об'єкти з великою ймовірністю моделюються точками, які стоять далеко одна від одної.

```
def t_SNE_visualization(X,y):

    from sklearn.manifold import TSNE
    import matplotlib.pyplot as plt

    model = TSNE(learning_rate=50)  #50-200
    transformed = model.fit_transform(X)
    xs = transformed[:,0]
    ys = transformed[:,1]
    plt.scatter(xs, ys, c=y, alpha=0.5,cmap='viridis')

    return plt.show()
```

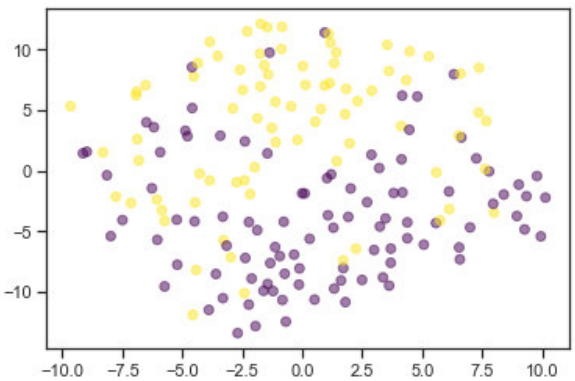

Рис. 3

Отримані результати показують, що ми маємо лінійно нероздільну вибірку.

Для побудови моделі класифікатора використаємо метод опорних векторів (SVM) с rbf (радіальна базова функція) ядром [6, 7].

Побудова моделі класифікатора

Розбиваємо вибірку на тренувальну та тестову (75% / 25%):

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=109)

Навчання моделі класифікатора проведемо на основі методу опорних векторів. Створимо об'єкт класифікатора, використовуючи параметри за замовченням. Проведемо навчання моделі на даних тренувальної вибірки. Використовуючи навчену модель, виконаємо передбачення класу даних тестової вибірки та визначимо базову оцінку якості моделі класифікатора:

```
from sklearn.svm import SVC
clf = SVC(kernel="rbf", probability=True)
clf.fit(X_train, y_train)

print('Train Accuracy:', clf.score(X_train, y_train))
print('Test Accuracy:', clf.score(X_test, y_test))
```

Ми отримали точність на тренувальному наборі 0,89 та 0,8 на тестовому. Для підвищення якості моделі необхідна оптимізація параметрів.

Побудуємо криві навчання.

```
## Learning curves
Model_Development_Functions.Learning_curves(clf,X_train,y_train)
```

```
def Learning_curves(estimator,X_train,y_train):

    import numpy as np
    from sklearn.svm import SVC
    from sklearn.model_selection import learning_curve
    import matplotlib.pyplot as plt

    train_sizes, train_scores, test_scores = \
        learning_curve(estimator=estimator,
                       X=X_train,
                       y=y_train,
                       cv=10,
                       n_jobs=-1,
                       train_sizes=np.linspace(0.1, 1.0, 10))

    train_scores_mean = np.mean(train_scores, axis=1)
    train_scores_std = np.std(train_scores, axis=1)
    test_scores_mean = np.mean(test_scores, axis=1)
    test_scores_std = np.std(test_scores, axis=1)

    plt.plot(train_sizes, train_scores_mean, 'o-', color="r", label="Training score")
    plt.fill_between(train_sizes, train_scores_mean - train_scores_std,
                         train_scores_mean + train_scores_std, alpha=0.1,
                         color="r")

    plt.plot(train_sizes, test_scores_mean, 'o-', color="g", label="Cross-validation score")
    plt.fill_between(train_sizes, test_scores_mean - test_scores_std,
                         test_scores_mean + test_scores_std, alpha=0.1, color="g")

    plt.xlabel('Training examples')
    plt.ylabel('Score')
    plt.legend(loc="best")
    plt.ylim([0.4, 1.0])
    plt.grid()

    return plt.show()
```

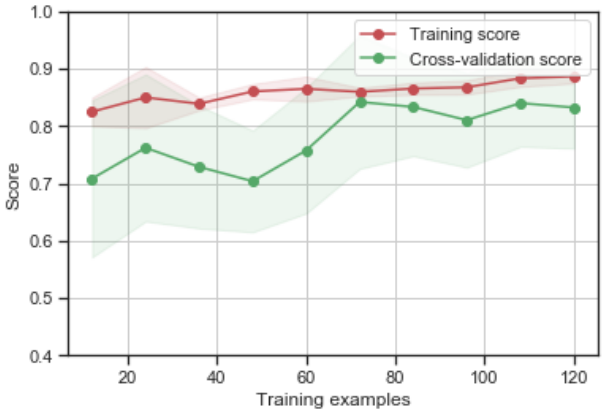

Рис. 4

Отримані криві навчання показують, що збільшення об’єму тренувальної вибірки може покращити якість моделі.

Побудуємо криві валідації для параметру C:

```
## Validation curves
param_name= 'C'
param_range = [0.001, 0.01, 0.1, 0.25, 0.5, 0.75, 1.0, 10.0, 100.0]
Model_Development_Functions.Validation_curves(clf, param_range, param_name, X_train, y_train)
```

```
def Validation_curves(estimator, param_range, param_name, X_train, y_train):

    import numpy as np
    from sklearn.svm import SVC
    import matplotlib.pyplot as plt
    from sklearn.model_selection import validation_curve

    train_scores, test_scores = validation_curve(
            estimator=estimator,
            X=X_train,
            y=y_train,
            param_name=param_name,
            param_range=param_range,
            cv=10,
            n_jobs=-1)

    train_mean = np.mean(train_scores, axis=1)
    train_std = np.std(train_scores, axis=1)
    test_mean = np.mean(test_scores, axis=1)
    test_std = np.std(test_scores, axis=1)

    plt.plot(param_range, train_mean, 'o-', color="r", label="Training score")
    plt.fill_between(param_range, train_mean - train_std,
                         train_mean + train_std, alpha=0.1,
                         color="r")

    plt.plot(param_range, test_mean, 'o-', color="g", label="Cross-validation score")
    plt.fill_between(param_range, test_mean - test_std,
                         test_mean + test_std, alpha=0.1, color="g")

    plt.xscale('log')
    plt.xlabel('Parameter ' + param_name)
    plt.ylabel('Score')
    plt.legend(loc="best")
    plt.ylim([0.4, 1.0])
    plt.grid()

    return plt.show()
```

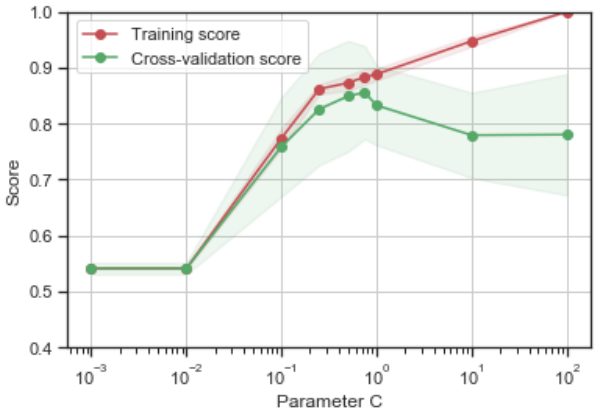

Рис. 5

Ми бачимо, що точність на тренувальній вибірці зростає зі збільшення параметра C, але на тестовому наборі вона починає падати з C = 0,75.

Побудуємо криві валідації для параметру gamma (γ) при C = 0,75:

```
clf = SVC(kernel="rbf", probability=True, C=0.75)
param_name= 'gamma'
param_range = np.linspace(0.0001, 10, 100)
Model_Development_Functions.Validation_curves(clf, param_range, param_name, X_train, y_train)
```

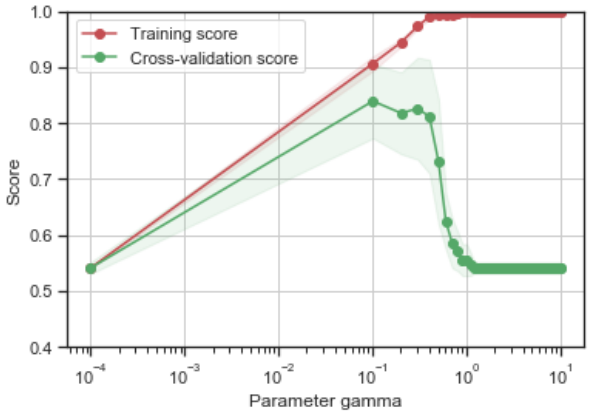

Рис. 6

Виходячи з отриманих кривих, оптимальним значенням gamma є 0,1.

За допомогою кривих валідації ми отримали, в першому наближенні, область оптимальних значень параметрів. Далі, для визначення найкращої комбінації C і γ використаємо пошук по сітці. Для цього згенеруємо по 100 значень параметрів C і γ. Таким чином, ми отримаємо сітку 100x100 комбінацій C і γ для нашої моделі. Кожна комбінація перевіряється з використанням крос-валідації, і вибирається та, яка проявила себе краще за всіх інших. Фінальна модель, яка використовується для тестування та класифікації нових даних, навчається потім на всій множині з використанням обраних параметрів.

```
## Grid_Search_CV Hyperparameter Tuning
param_grid = {"C": np.linspace(0.1, 1, 100), "gamma": np.linspace(0.005, 0.4, 100)}
best_clf=Model_Development_Functions.Grid_Search_CV(clf, param_grid, X_train, y_train, X_test, y_test)
```

```
def Grid_Search_CV(estimator, param_grid, X_train, y_train, X_test, y_test):

    import time
    from sklearn.svm import SVC
    from sklearn.model_selection import GridSearchCV

    searchCV = GridSearchCV(estimator, param_grid, cv=10, scoring='accuracy', n_jobs = 3, verbose = 0)
    start = time.time()
    searchCV.fit(X_train, y_train)
    end = time.time()

    print("Elapsed time: " + str(round((end-start)/60)) + " min.")

# Examine the best model
    print('Examine the best model:')
    print('Best Score: ' + str(searchCV.best_score_))
    print('Best Parameters: ' + str(searchCV.best_params_))
    print('Best Estimator: ' + str(searchCV.best_estimator_))

    best_svm=searchCV.best_estimator_
    best_svm.fit(X_train, y_train)

    print('Training Accuracy:', best_svm.score(X_train, y_train))
    print('Testing Accuracy:', best_svm.score(X_test, y_test))

    return best_svm
```

Результати оптимізації:

```
In [136]: best_clf=Model_Development_Functions.Grid_Search_CV(clf, param_grid, X_train, y_train, X_test, y_test)
Elapsed time: 4 min.
Examine the best model:
Best Score: 0.8592592592592593
Best Parameters: {'C': 0.5272727272727272, 'gamma': 0.04090909090909091}
Best Estimator: SVC(C=0.5272727272727272, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma=0.04090909090909091,
  kernel='rbf', max_iter=-1, probability=True, random_state=None,
  shrinking=True, tol=0.001, verbose=False)
Training Accuracy: 0.8740740740740741
Testing Accuracy: 0.8222222222222222
```

Таким чином, оптимальними значеннями параметрів C і γ є:
best_clf =SVC(C=0.527, gamma=0.041, kernel='rbf')
При цьому точність моделі на тестовому наборі збільшилась до 82,2%.

Висновки

Для побудови моделі, насамперед, необхідно провести аналіз наявних даних, що дасть можливість визначити складність поставленого завдання та звузить коло відповідних методів. Далі необхідно провести чистку і обробку даних, тобто підготовку до моделювання. Потім застосовується обрана модель, в нашому випадку класифікації, з її подальшою оптимізацією. При підборі параметрів необхідно використовувати крос-валідацію як на тренувальній та тестовій вибірках, так і всередині тренувального набору даних. Таким чином можна отримати високу точність моделі при її робастності.

Список використаної літератури:

1. Классификация [Електронний ресурс]. – Режим доступу: http://www.machinelearning.ru/wiki/index.php?title=Классификация.
2. Машинное обучение (курс лекций, К. В. Воронцов) [Електронний ресурс]. – Режим доступу: http://www.machinelearning.ru/wiki/index.php?title=Машинное_обучение_%28курс_лекций%2C_К.В.Воронцов%29.
3. Малик І.В. Методи машинного навчання для статистичної обробки медичних даних / І.В. Малик, Т.В. Кнігніцька // Науковий вісник Чернівецького національного університету. Серія: Комп'ютерні системи та компоненти. – 2017. – Том 8, випуск 2. – С. 77 – 85.

4. Statlog (Heart) Data Set [Електронний ресурс]. – Режим доступу: http://archive.ics.uci.edu/ml/datasets/statlog+(heart).
5. L.J.P. van der Maaten. Visualizing High-Dimensional Data Using t-SNE / L.J.P. van der Maaten, G.E. Hinton // Journal of Machine Learning Research. – 2008. – № 9. – P. 2579 – 2605.
6. Cortes C. Support-vector networks / Cortes C., Vapnik V. // Machine Learning. – 1995. – № 20 (3). – P. 273–297.
7. Cristianini N. An Introduction to Support Vector Machines and Other Kernel-based Learning Methods / N. Cristianini, J. Shawe-Taylor. – Cambridge, U.K.: Cambridge University Press, 2000. – 189 p.

Bibliography:

1. Klassifikatsiya [Classification]. (2011). Retrieved from http://www.machinelearning.ru/wiki/index.php?title= Classification [in Russian]
2. Mashinnoe obuchenie (kurs lektsiy, K. V. Vorontsov) [Machine learning (lecture course, K.V. Vorontsov)]. (2019). Retrieved from http://www.machinelearning.ru/wiki/index.php?title=Machine_learning_%28course_lecture%2C_K.V.Vorontsov%29 [in Russian]
3. Malyk I.V., Knizhnitskaya T.V. (2017). Metody mashynnoho navchannia dlia statystychnoi obrobky medychnykh danykh [Methods of machine learning for statistical processing of medical data]. Naukovy Visnyk Chernivetskogo Natsionalnogo Universitetu. Seriia: Kompiuterni systemy ta komponenty, – Scientific Bulletin of Chernivtsi National University. Series: Computer Systems and Components , v. 8, is. 2, 77–85 [in Ukrainian].
4. Statlog (Heart) Data Set. (n.d.). Retrieved from http://archive.ics.uci.edu/ml/datasets/statlog+(heart).
5. L.J.P. van der Maaten and G.E. Hinton. (2008) Visualizing High-Dimensional Data Using t-SNE. Journal of Machine Learning Research, 9(Nov), 2579-2605.
6. Cortes C., Vapnik V. (1995) Support-vector networks. Machine Learning, 20 (3), 273–297.
7. Nello Cristianini, John Shawe-Taylor. (2000) An Introduction to Support Vector Machines and Other Kernel-based Learning Methods. Cambridge, U.K.: Cambridge University Press.

PISKUN Oleksandr,
PhD, Associate Professor of Department of Automation and Computer-integrated Technologies, The Bohdan Khmelnytsky National University of Cherkasy

CLASSIFICATION MODEL BUILDING USING MACHINE LEARNING METHODS

Summary. Introduction. *Classification is one of the machine learning fields that solves the following problem: let there be a multitude of objects separated into classes according to one or several attributes. A finite set of objects is given for which it is known which classes they belong to. What class the rest of the objects belong to is unknown. It is necessary to build an algorithm capable of classifying an arbitrary object of the original set.*

Let's consider the most common data classification algorithms.

Decision tree. *The principle of this method is to recursively split the set of elements into subsets containing elements that belong to the same class. Different types of decision trees differ in finding the target variable and its attributes to build a new node.*

Bayesian classification. *Bayesian classification belongs to statistical classification methods that predict an object's class with the help of the probability theory. The algorithm is based on the Bayes theorem and assumes the independence of the considered attributes.*

K-nearest neighbors. *The method assumes that the next object from the test set belongs to the class to which the majority of its closest "neighbors" from the training set belong. In this case, the closest observations to the test object defined as neighbors. Thus, the new object is placed in the class containing the largest number of closest similarities.*

Support Vector Machine. *This method uses a hyperplane to divide space into subspaces that characterize individual classes. The objects lying on the border of the formed regions are called support vectors. In the case of a linear inseparability problem kernel transformation is used.*

Purpose. *The purpose of this paper is a step-by-step presentation and practical application of the algorithm for the classification model building.*

Results. *Application of the provided algorithm for the classification model building to determine the presence or absence of heart disease based on the real data set made it possible to achieve high efficiency and robustness of the model.*

Conclusion. *The paper presents an algorithm for building a model of the classification problem solving to determine the presence or absence of heart disease based on machine learning methods.*

Data pre-processing and analysis enabled identification of the non-linear dependencies of the target variable and prepared the data for effective modeling. SVM method with Rbf Kernel was applied for the classification model building. The model parameters were optimized using a grid search with cross-validation of each combination of the parameters.

Building a model first of all it is necessary to analyze the available data in order to reveal the complexity of the given problem and narrow down the range of suitable methods. Next, it is needed to clean and pre-process the data – prepare it for modeling. Then the selected method should be applied and optimization of the parameters should be provided. By the parameter selection, it is crucial to use cross-validation both on the training and test samples, and within the training data set itself. Thus, it is possible to achieve high accuracy of the model with a good degree of its robustness.

Keywords: *machine learning, classification, support vector machine, data mining, t-SNE visualization.*

Одержано редакцією 06.09.2018 р.
Прийнято до публікації 21.11.2018 р.

УДК 004.032.26

DOI 10.31651/2076-5886-2019-1-53-60

PACS 07.05.Mh, 07.05.Kf, 07.05.Pj

КРАСНОШЛИК Наталія Олександрівна
кандидат технічних наук, доцент,
доцент кафедри прикладної математики та інформатики Черкаського національного університету імені Богдана Хмельницького
e-mail: wlik007@ukr.net
ORCID 0000-0003-4661-6997

СЕРДЮК Марина Олександрівна
студентка 4 курсу спеціальності «Прикладна математика», Національний університет «Львівська політехніка», м. Львів
e-mail: mary_serdyuk@ukr.net
ORCID 0000-0003-4661-1234

ЗАСТОСУВАННЯ АНСАМБЛІВ НЕЙРОННИХ МЕРЕЖ ДЛЯ РОЗВ'ЯЗАННЯ ЗАДАЧІ КЛАСИФІКАЦІЇ ЗОБРАЖЕНЬ

У роботі описано існуючі підходи до побудови ансамблів моделей у машинному навчанні. Наведено найбільш популярні системи для навчання нейронних мереж. У якості базової моделі обрано двошарову нейрону мережу прямого розповсюдження, що має один прихований шар. Розглянуто два підходи до побудови ансамблю нейронних мереж, такі як усереднюючий ансамбль та ансамбль з керівником. Вони були реалізовані за допомогою бібліотек Keras і TensorFlow. Проведено дослідження ефективності застосування ансамблів до розв'язання задач класифікації зображень. Для тестування обрано набір даних MNIST для класифікації рукописних цифр. Досліджено ефективність використання ансамблів різної структури з 3-9 нейронних мереж.

Ключові слова: *машинне навчання, задача класифікації, нейронна мережа, ансамбль моделей.*

Постановка проблеми

Останнім часом машинне навчання є однією з передових технологій сучасності. Машинне навчання як область практичної діяльності, і як сфера наукових досліджень алгоритмів, полягає у отримані знань з даних шляхом виявлення прихованих закономірностей в них. Перетворення даних у знання є особливо актуальною і цікавою