Задача непрерывной классификации для Iris flower data set.

Баталов Семен

18.02.2021

1. Iris flower data set

Это стандартный набор данных, интегрированнный в модуль «**sklearn**» языка «**Python**» (Рис. 1). Набор данных состоит из 150 образцов каждого из трех видов ириса (Iris setosa, Iris virginica и Iris versicolor). Для каждого образца были измерены четыре характеристики: длина и ширина чашелистиков и лепестков в сантиметрах. Основываясь на комбинации этих четырех характеристик, можно разработать несложный классификатор, чтобы отличать виды друг от друга.

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	target	name
0	5.1	3.5	1.4	0.2	0	setosa
1	4.9	3.0	1.4	0.2	0	setosa
2	4.7	3.2	1.3	0.2	0	setosa
3	4.6	3.1	1.5	0.2	0	setosa
4	5.0	3.6	1.4	0.2	0	setosa
...	...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	2	virginica
146	6.3	2.5	5.0	1.9	2	virginica
147	6.5	3.0	5.2	2.0	2	virginica
148	6.2	3.4	5.4	2.3	2	virginica
149	5.9	3.0	5.1	1.8	2	virginica

150 rows × 6 columns

Рис. 1. Набор данных трех видов ириса.

2. Классификаторы

Классификаторы были написан на языке «**Python**». Подробнее о программе можно узнать в папке «**source**» проекта. Было использовано два алгоритма классификации: SVM (support vector machine), KNN (k-nearest neighbors algorithm). Сначала рассмотрим алгоритм SVM.

2.1. Метод опорных векторов (SVM)

При написании программы использовался модуль «**sklearn**», из него был взят алгоритм «**SupportVectorClassification**». Обучающая выборка составила 50% от всех полей (переставленных в случайном порядке) в наборе. После была произведена проверка работоспособности классификатора на оставшихся в датасете примерах. Результат проверки изображен на рисунке (Рис. 2).

	precision	recall	f1-score	support
setosa	1.00	1.00	1.00	27
versicolor	0.92	1.00	0.96	23
virginica	1.00	0.92	0.96	25
accuracy			0.97	75
macro avg	0.97	0.97	0.97	75
weighted avg	0.98	0.97	0.97	75

Рис. 2. Оценка работы классификатора SVM.

2.2. Метод «k» ближайших соседей (KNN)

При написании программы использовался модуль «**sklearn**», из него был взят алгоритм «**KNeighborsClassifier**». Обучающая выборка составила 50% от всех полей (переставленных в случайном порядке) в наборе. После была произведена проверка работоспособности классификатора на оставшихся в датасете примерах. При этом количество соседей составило 10, а мера расстояния между данными была стандартной евклидовой. Результат проверки изображен на рисунке (Рис. 3).

	precision	recall	f1-score	support
setosa	1.00	1.00	1.00	27
versicolor	0.88	1.00	0.94	23
virginica	1.00	0.88	0.94	25
accuracy			0.96	75
macro avg	0.96	0.96	0.96	75
weighted avg	0.96	0.96	0.96	75

Рис. 3. Оценка работы классификатора KNN.

Так же был произведен анализ того, какое количество соседей в данной задаче является оптимальным. Для этого был использован алгоритм «**k-fold кросс-валидации**, идея которого заключается в разбиении всего набора данных на «k» блоков схожих размеров. Далее поочередно один из этих блоков используется как тестировочная выборка, а оставшиеся, как обучающая выборка, производится обучение и анализ точности классификатора (под точность подразумеваем отношение

числа правильно классифицированных наборов данных к общему количеству этих наборов). После этого из «k» независимых итераций обучения находится среднее значение точности работы классификатора. В нашем случае эти значения изображены на графике (Рис. 4).

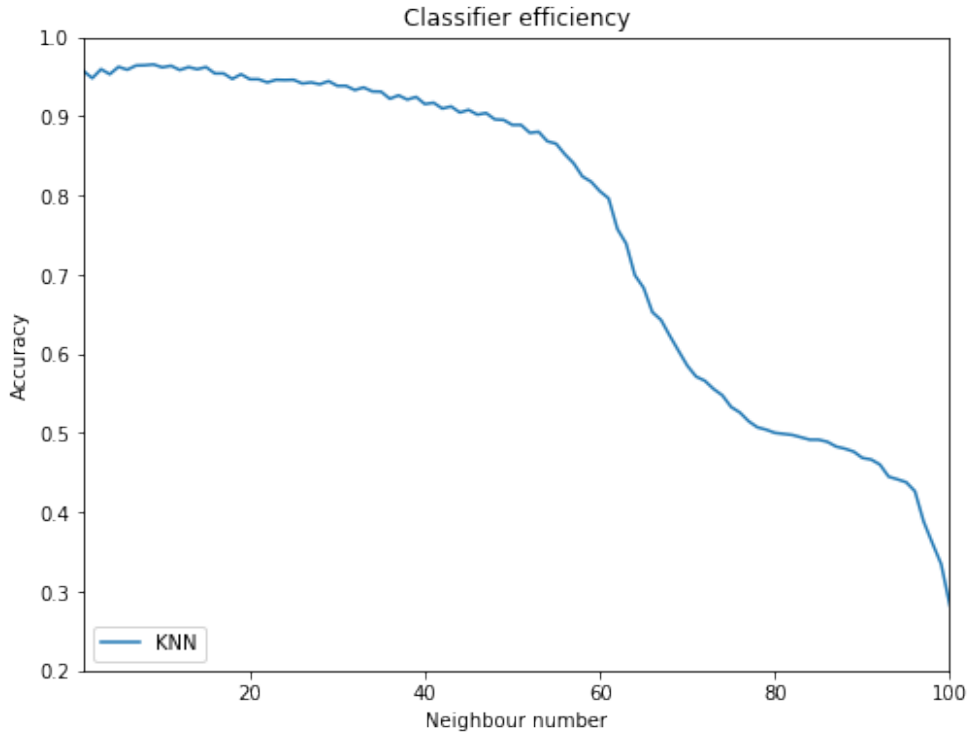

Рис. 4. График зависимости точности классификатора KNN от количества соседей.

Нетрудно видеть, что наилучший результат достигается при $k \in [2, \ldots, 15]$. Более того, чем больше значение «k» тем хуже алгоритм справляется с задачей классификации. В дальнейшем будем использовать (для эффективной работы) значение «k» именно из описанного выше отрезка.

2.3. Сравнение SVM и KNN

Изучим, какой из алгоритмов на данном наборе данных лучше справляется с задачей классификации. Для этого будем искать среднюю эффективность алгоритмов на фиксированной обучающей и тестовой выборке. Стоит отметить, что выборки будут использоваться с разным соотношением числа примеров для тренировки и для тестировки. В алгоритме «KNN» будем брать $k \in [2, \ldots, 15]$.

Результаты обучения алгоритмов представлены на графике (Рис. 5). Здесь число ближайших соседей равно 12.

Видно, что при всех возможных выборках точность работы алгоритма «SVM» выше точности алгоритма «KNN» (причем количество ближайших соседей выбрано оптимальным). Также можно отметить, что точность у обоих подходов убывает при сокращении количества обучающих примеров, но в методе «KNN» она убывает намного быстрее.

Можно сделать вывод, что оптимальное отношение количества примеров в обучающей и тестировочной выборках к общему количеству примеров должно находиться

в диапазоне $[0,1 \ldots 0,4]$ для тестировочной и $[0,6 \ldots 0,9]$ для обучающей. Критическим переходом можно считать соотношение 0.5 для обеих выборок.

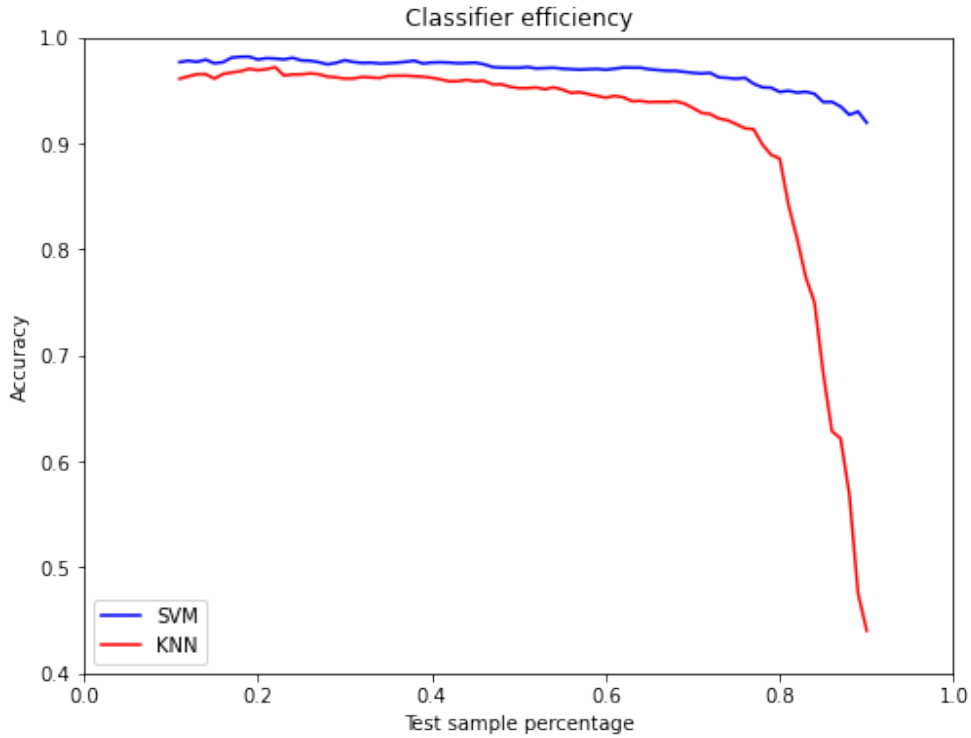

Рис. 5. График зависимости точности классификаторов SVM и KNN от тестировочной выборки (и соответственно обучающей).