Support Vectors

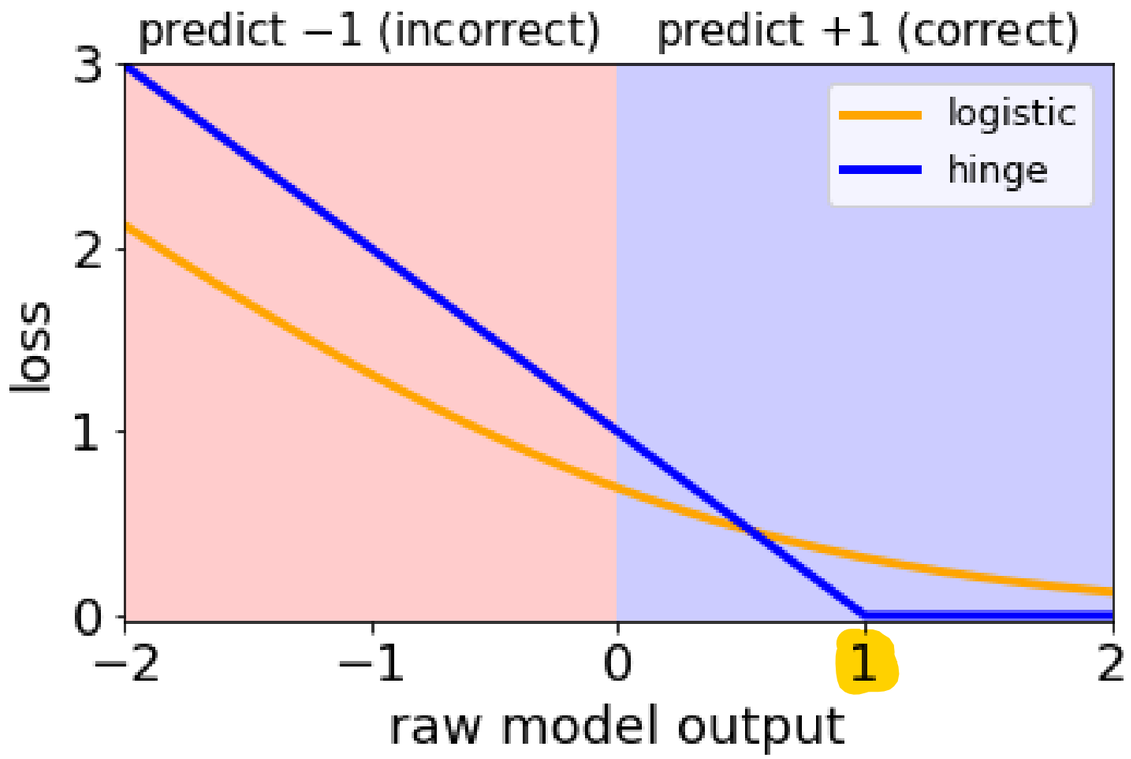
LINEAR CLASSIFIERS IN PYTHON

Michael (Mike) Gelbart
Instructor, The University of British Columbia

datacamp

What is an SVM?

- Linear classifiers (so far)
- Trained using the hinge loss and L2 regularization



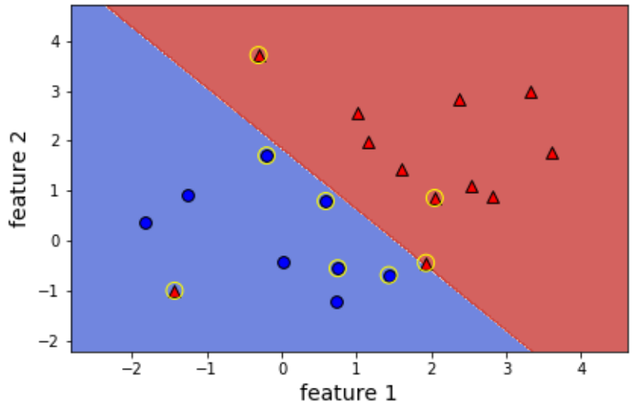
Logistic regression :
logistic loss fuction 으로 학습한 선형 분류기

SVM도 선형 분류기이지만 hinge loss를 대신 사용
SVM에 L2 정규화도 포함됨

logistic, hinge loss는 비슷해보이지만 주요 차이점은
힌지 loss에는 "평평한" 부분이 이쓰며,
이는 raw model output이 1보다 클때 발생하며,
이는 일부 오차 한계를 넘어 예제를 정확하게 예측했음을 의미

훈련의 예가 "zero loss" 영역에 해당되면 예를 제거해도 아무것도 변하지 않음
(SVM의 핵심 속성)

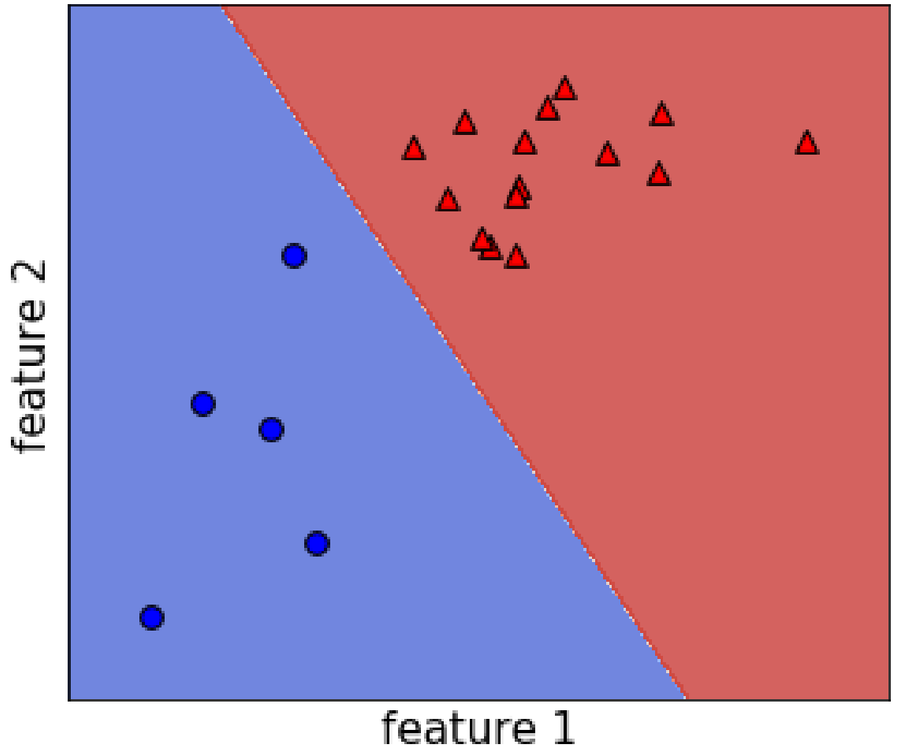
- Support vector: a training example **not** in the flat part of the loss diagram
  support vetor는 loss diagram의 flat 부분에 없는 예로 정의.
- Support vector: an example that is incorrectly classified **or** close to the boundary
  support vector를 정의하는 또 다른 방법은 잘못 분류된 예제뿐만 아니라 경계에 가까운 정확하게 분류된 예제도 포함하는 것.
- If an example is not a support vector, removing it has no effect on the model
  support vector가 아니라면, lost는 이미 0이기 때문에 이를 제거하는 것은 model에 영향을 끼치지 않음
- Having a small number of support vectors makes kernel SVMs really fast
  support vector의 수가 적으면 kernal SVM이 매우 빨라짐



그림에서 서포터 벡터는 둘레에 노란색 원이 표시되어 있음.

Max-margin viewpoint

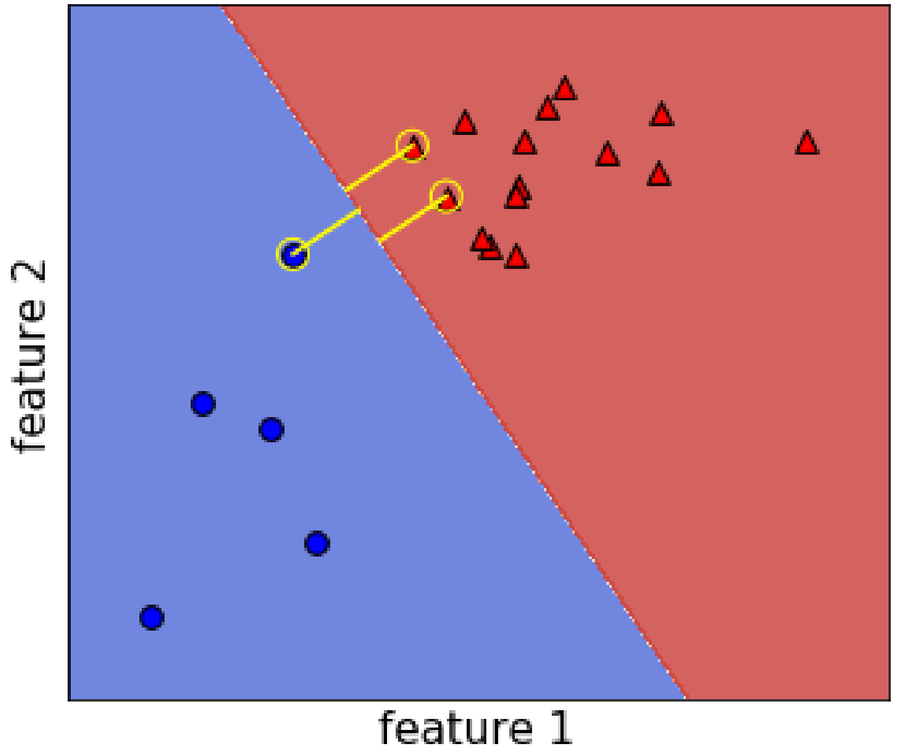
- The SVM maximizes the "margin" for linearly separable datasets margin을 극대화
- Margin: distance from the boundary to the closest points



이 다이어그램은 선형으로 분리 가능한 데이터셋에 적합한 SVM을 보여줌.
학습된 경계는 두 class사이에 있음
= > 다른 정보가 없을때, 이 경계는 다른 class보다 한 class에 훨씬 가까운 경계보다 더 의미가 있음.

Max-margin viewpoint

- The SVM maximizes the "margin" for linearly separable datasets
- Margin: distance from the boundary to the closest points



yellow line은 vector에서 경계까지의 거리를 나타냄
3가지 경우 모두에 대해 동일한 노란색 선의 길이를 margin이라고 함

정규화 강도가 너무 크지 않으면 SVM은 선형으로 분리 가능한 dataset의 한계를 최대화 = > but, 대부분의 데이터 세트는 선형으로 분리할 수 없음.
= > 즉, 일반적으로 100%의 교육 정확도를 기대하지 않음.

SVM은 L2정규화로 인한 힌지 손실

Let's practice!

LINEAR CLASSIFIERS IN PYTHON

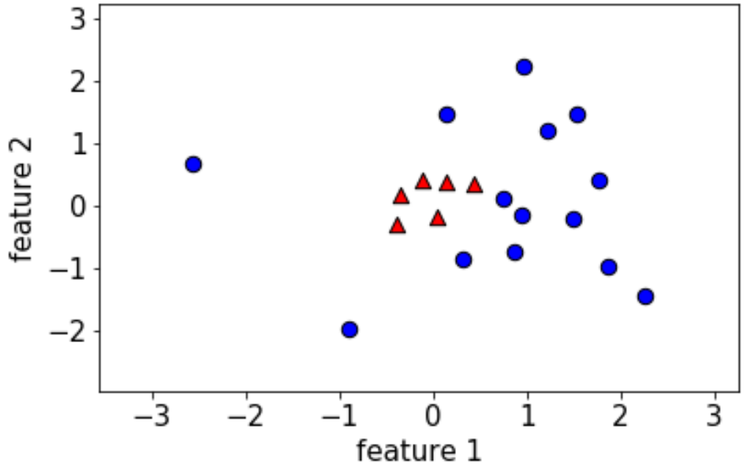
Kernel SVMs

LINEAR CLASSIFIERS IN PYTHON

Michael (Mike) Gelbart
Instructor, The University of British Columbia

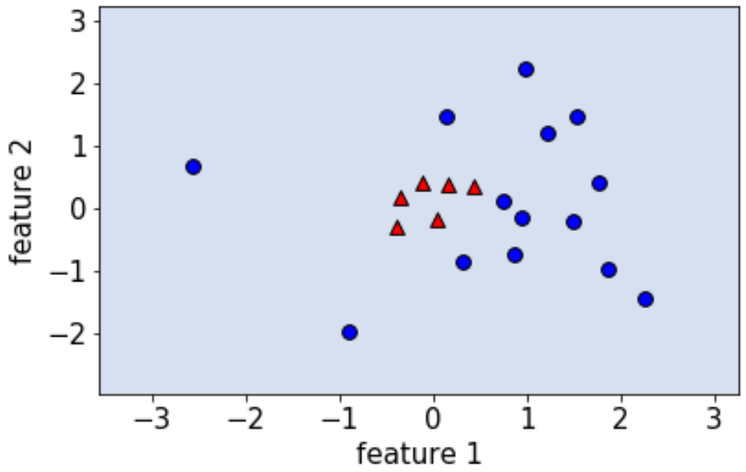
datacamp

Transforming your features



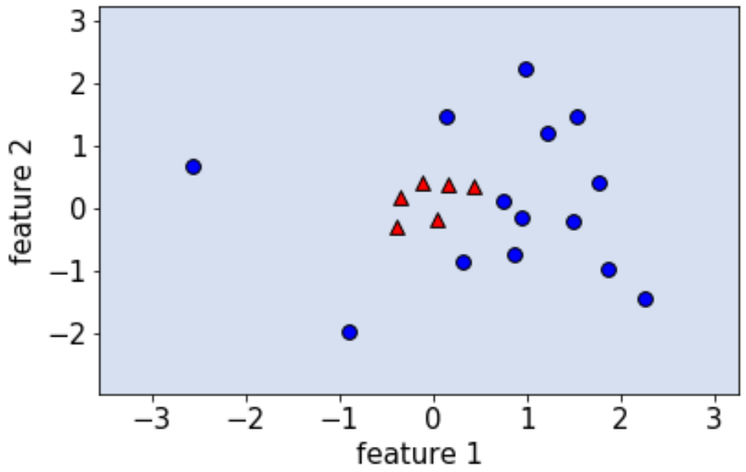
두 class는 선형으로 분리 할 수 없음
= > 모든 점을 완벽하게 분류하는 linear boundary는 없음

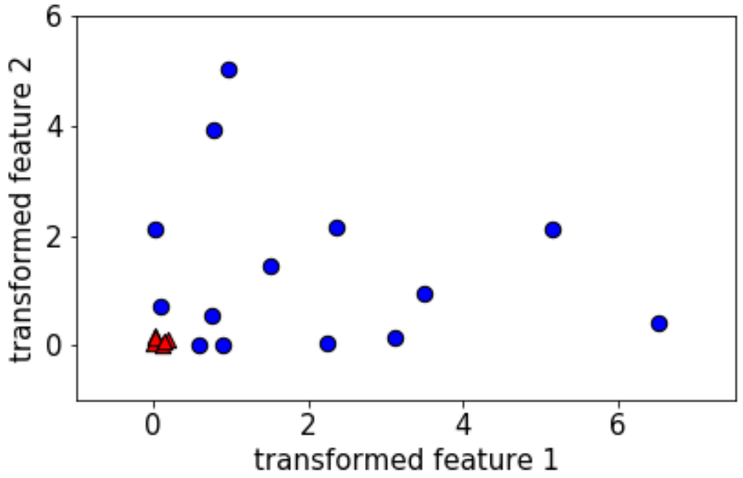
Transforming your features



빨간색 점은 모두 (0,0)에 가까움
하나는 feature 1^(제곱)이고 다른 하나는 feature 2^(제곱)인 두개의 새로운 feature을 만들어 보자
즉, 0에 가까운 값은 작은 값이 되고, 양수와 음수 모두 0에서 멀어질 수록 커짐

Transforming your features



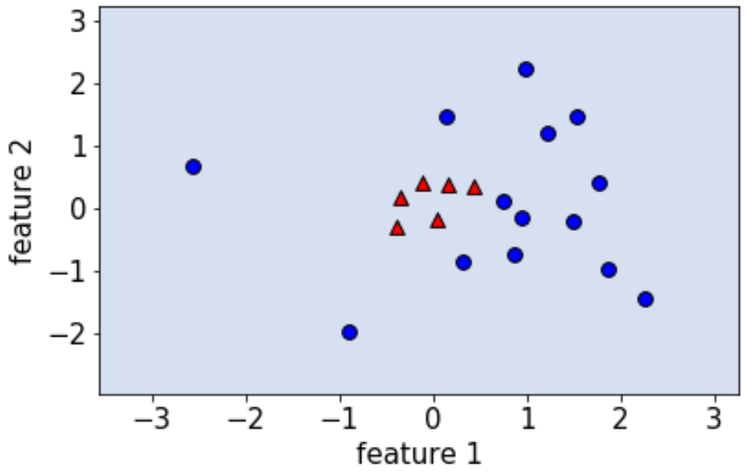


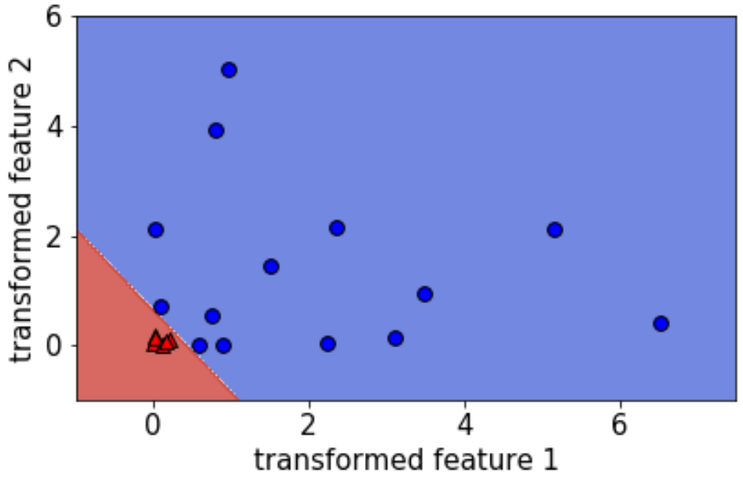
변환 시켜서 그래프로 그리면 선형으로 분리할 수 있음

이러한 새로운 feature을 사용하여 선형 SVM을 맞출 수 있고 결과는 완벽하게 분류할 수 있음

$$\text{transformed feature} = (\text{original feature})^2$$

Transforming your features

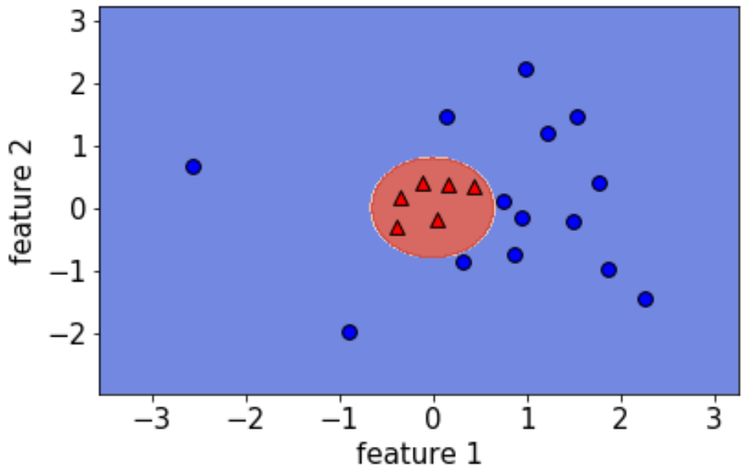


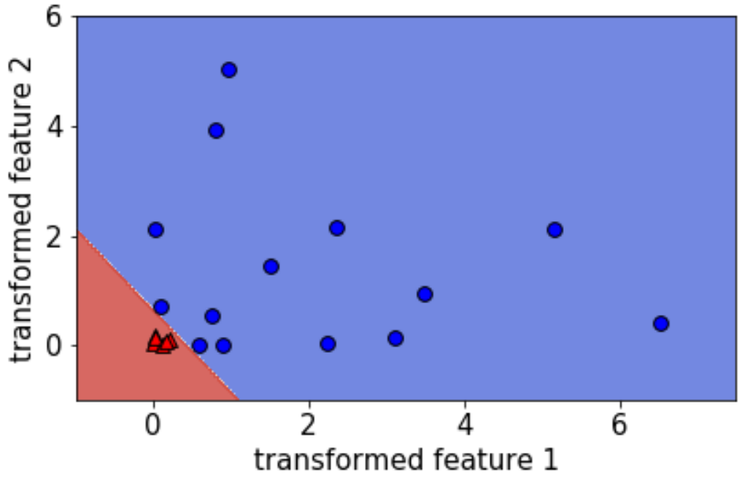


$$\text{transformed feature} =$$

$$(\text{original feature})^2$$

Transforming your features





원래 공간에서 이 선형 경계를 나타냈을 때

$$\text{transformed feature} = (\text{original feature})^2$$

즉, 우리가 이 축들을 가지고 제곱을 푼다면, 경계의 모양은 어떻게 될까
= > 원을 얻음

여기서 얻는 메세지 : 변환된 공간에서 선형 모델을 맞추는 것은
원래 공간에 비선형 모델을 맞추는 것에 해당 (항상 그런것은 아님)
제곱과 경계가 항상 타원이 되는 것은 아님

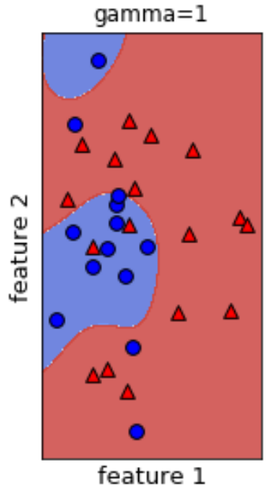
Kernel SVMs

많은 비선형 커널이 존재하지만, 이 과정에서는 RBF에 초점을 맞춤

```
from sklearn.svm import SVC

svm = SVC(gamma=1)     # default is kernel="rbf"
```

default 값은 RBF 또는 Radial Basis Function kernel



Kernal SVM을 사용하면 일반적인 방식으로 fit을 호출하고 예측할 수 있음

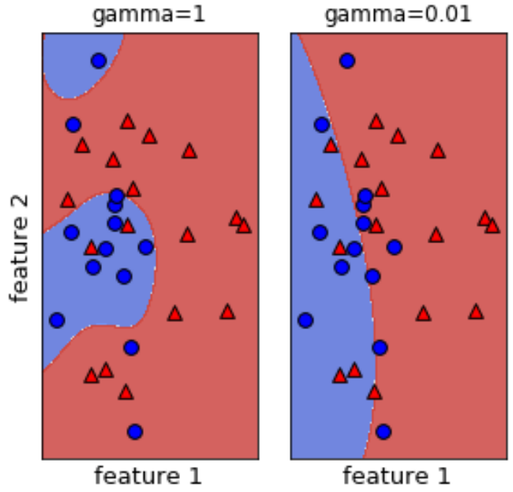
decision boundary가 비선형으로 선형 경계에서 얻을 수 있는 것보다 더 높은 훈련 정확도를 얻음

hyperparameter를 사용하여 경계의 모양을 제어할 수 있는데, 평소처럼 정규화를 제어하는 C hyperparameter가 있고
RBF kernal은 boundary의 유연함을 control 하는 새로운 hyper parmeter인 gamma를 도입함.
gamma를 줄임으로써 경계를 더 부드럽게 만들 수 있음.

Kernel SVMs

```
from sklearn.svm import SVC

svm = SVC(gamma=0.01) # default is kernel="rbf"
```



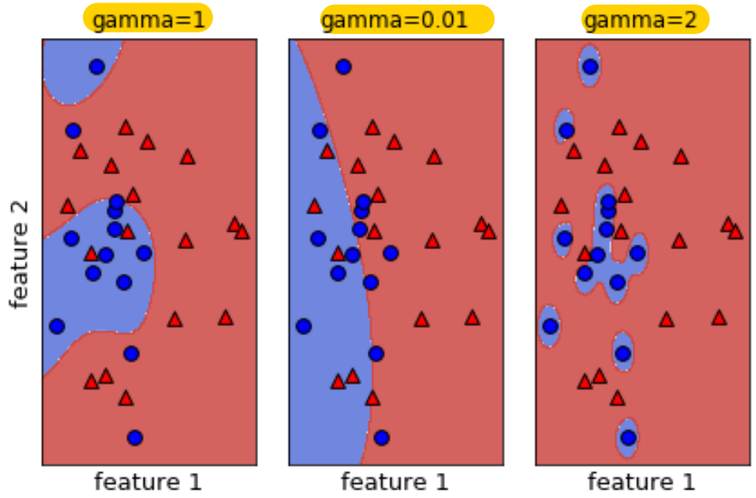
두번째 이미지는 gamma가 0.01로 설정된 동일한 데이터 세트

- smaller `gamma` leads to smoother boundaries

Kernel SVMs

```
from sklearn.svm import SVC

svm = SVC(gamma=2)    # default is kernel="rbf"
```



각 파란색 train example에 파란색 'Island'를 만들어 100% training accuracy를 달성

RBF SVM은 적절한 hyperparameter를 통해 거의 모든 dataset를 완벽하게 분리할 수 있음.
항상 가장 큰 gamma값을 사용하고 가능한 가장 높은 훈련 정확도를 얻는 이유
=> overfitting

- larger `gamma` leads to more complex boundaries

Let's practice!

LINEAR CLASSIFIERS IN PYTHON

Comparing logistic regression and SVM

LINEAR CLASSIFIERS IN PYTHON

Michael (Mike) Gelbart
Instructor, The University of British Columbia

datacamp

Logistic regression:

- Is a linear classifier
- Can use with kernels, but slow
- Outputs meaningful probabilities
- Can be extended to multi-class
- All data points affect fit
- L2 or L1 regularization

Support vector machine (SVM):

- Is a linear classifier
- Can use with kernels, and fast
- Does not naturally output probabilities
- Can be extended to multi-class
- Only "support vectors" affect fit
- Conventionally just L2 reaularization

둘다 kernel과 함께 사용할 수 있지만 SVM에서 더 일반적임 (support vector수가 적을 때 예측이 훨씬 빠름)

둘다 output probabilities로 강제 할수 있지만, logistic regression에서 훨씬 자연스러움

둘 다 one - vs - rest 방식으로 손실을 직접 수정하여 multi - class로 확장 할 수 있음

logistic regression에서는 대부분의 방법과 마찬가지로 모든 데이터 요소가 fit에 영향을 주지만 SVM에는 support vecotr 이외에 fit에 영향을 미치지 않고 제거할 수 있는 특수한 속성을 가지고 있음

"logistic regression"라는 용어는 특정한 유형의 정규화를 가리키지 않지만, "SVM"이라는 용어는 일반적으로 hinge loss와 L2 정규화를 사용하는 선형 분류기를 가리킴.

Use in scikit-learn

Logistic regression in sklearn:

- `linear_model.LogisticRegression`

Key hyperparameters in sklearn: 주요 hyperparmeter

- `C` (inverse regularization strength) 정규화의 양을 제어하는 C : C가 작을수록 정규화가 커지고, C가 커질수록 정규화가 작아짐
- `penalty` (type of regularization) L2 및 L1을 지원
- `multi_class` (type of multi-class) 이진 분류기를 다중 클래스로 확장하는 방법 (이 hyperparmeter 은 이전 두개보다 적은 결과에 영향 미침)

SVM in sklearn:

- `svm.LinearSVC` and `svm.SVC` SVM은 LinearSVC를 사용하거나 SVC를 사용
(kernel을 선형으로 설정하여 SVC class를 통해 선형 SVM을 실제로 맞출 수 있지만, LinearSVC가 더 빠를 수 있음)

Use in scikit-learn (cont.)

Key hyperparameters in sklearn: SVM class의 주요 하이퍼 파라미터

- `C` (inverse regularization strength) logistic regression과 같음
- `kernel` (type of kernel) kernal, RBF는 logistic regression과 다름
- `gamma` (inverse RBF smoothness)

Gamma는 RBF 커널에만 적용되고 controls the smoothness
감마 값이 작을수록 decision boundary가 더 부드럽거나 단순해지고
감마 값이 클수록 decision boundary가 더 복잡해짐

LogisticRegression과 같이 더 많은 hyperparmeter 존재)

SGDClassifier

`SGDClassifier` : scales well to large datasets

SGD는 확률적 gradient descent를 나타냄

```
from sklearn.linear_model import SGDClassifier

logreg = SGDClassifier(loss='log')

linsvm = SGDClassifier(loss='hinge')
```

SGDClassifier는 앞에서 논의한 다른 방법보다 매우 큰 dataset를 훨씬 잘 처리할 수 있음.

- `SGDClassifier` hyperparameter `alpha` is like `1/C`

Logistic Regresion과 linear SVM 사이에서 전환하려면 SGDClassifier의 손실 hyperparameter만 설정하면 됨. (동일한 모델에 loss만 변함)
SGDClassifier에서 "gotcah"는 정규화 hyperparameter 변수를 C대신 알파라고 부르며, 큰 알파는 더 많은 정규화를 의미. (기본적으로 알파는 C의 역수)

Let's practice!

LINEAR CLASSIFIERS IN PYTHON

Conclusion

LINEAR CLASSIFIERS IN PYTHON

Michael (Mike) Gelbart
Instructor, The University of British Columbia

datacamp

How does this course fit into data science?

- Data science
- → Machine learning
- →→ Supervised learning
- →→→ Classification
- →→→→ Linear classifiers (this course)

Congratulations & thanks!

LINEAR CLASSIFIERS IN PYTHON