

# Support Vectors

LINEAR CLASSIFIERS IN PYTHON

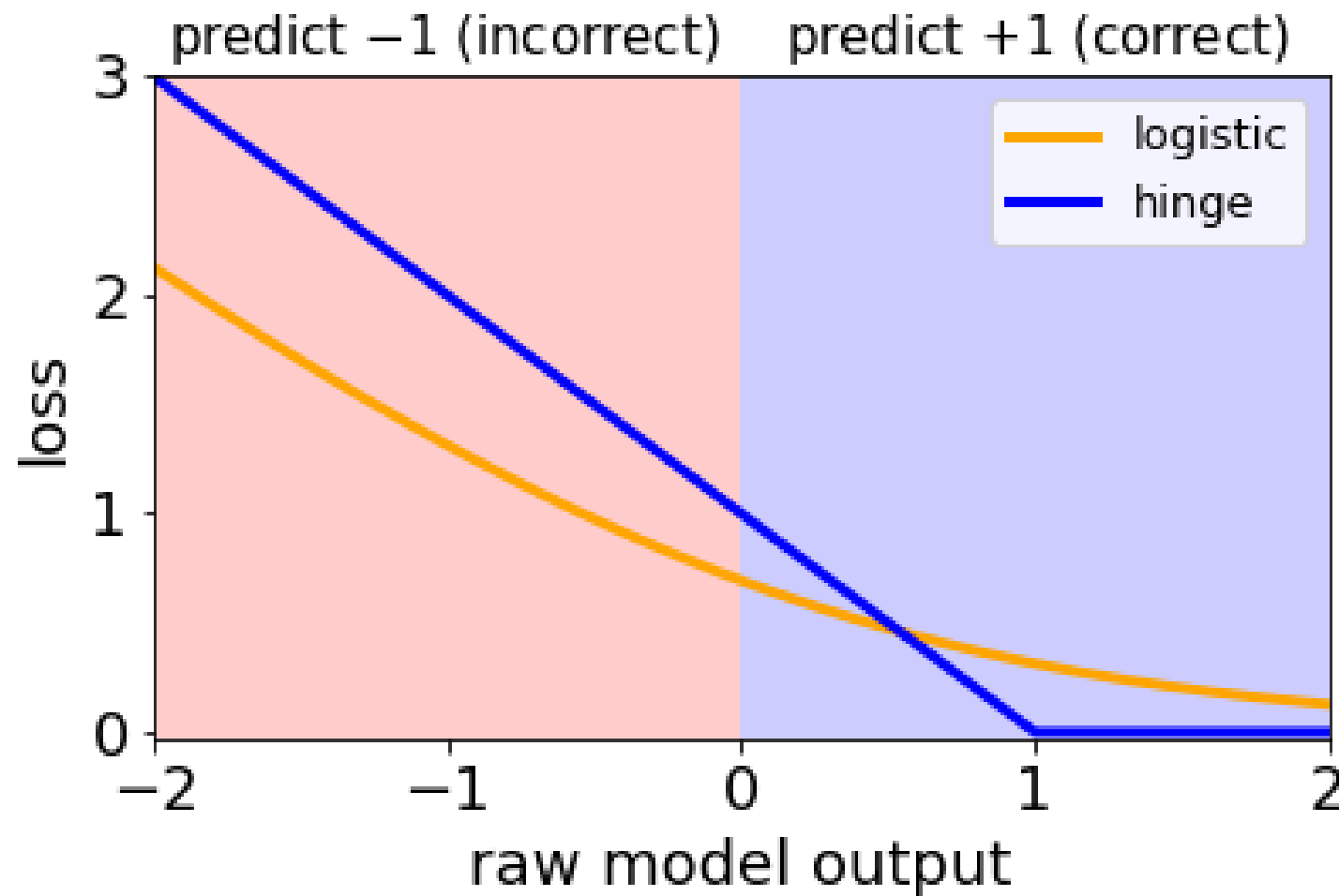


**Michael (Mike) Gelbart**

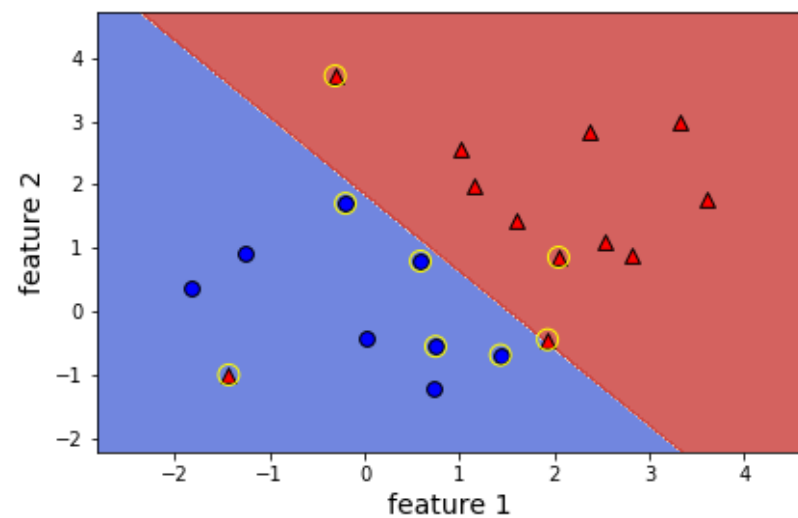
Instructor, The University of British  
Columbia

# What is an SVM?

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

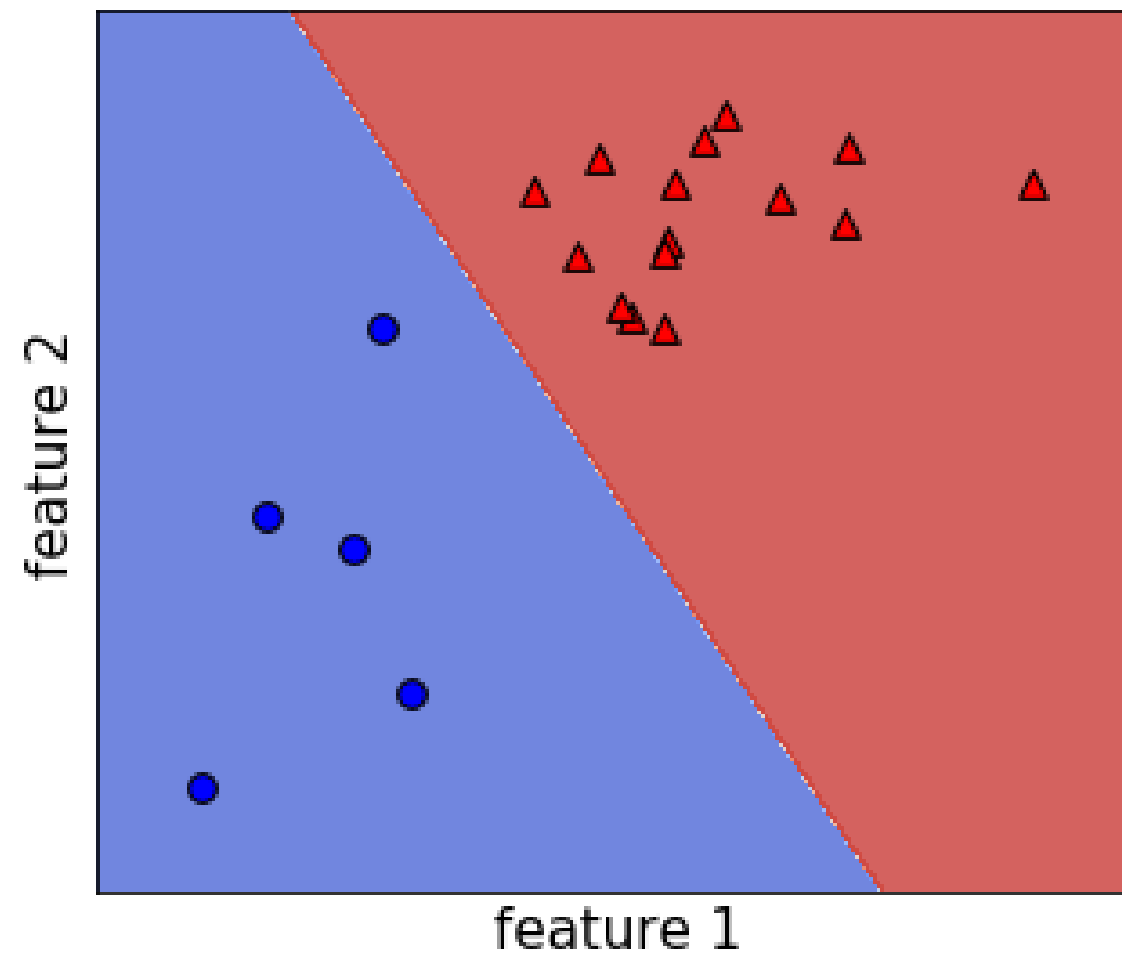


- Support vector: a training example **not** in the flat part of the loss diagram
- Support vector: an example that is incorrectly classified **or** close to the boundary
- If an example is not a support vector, removing it has no effect on the model
- Having a small number of support vectors makes kernel SVMs really fast



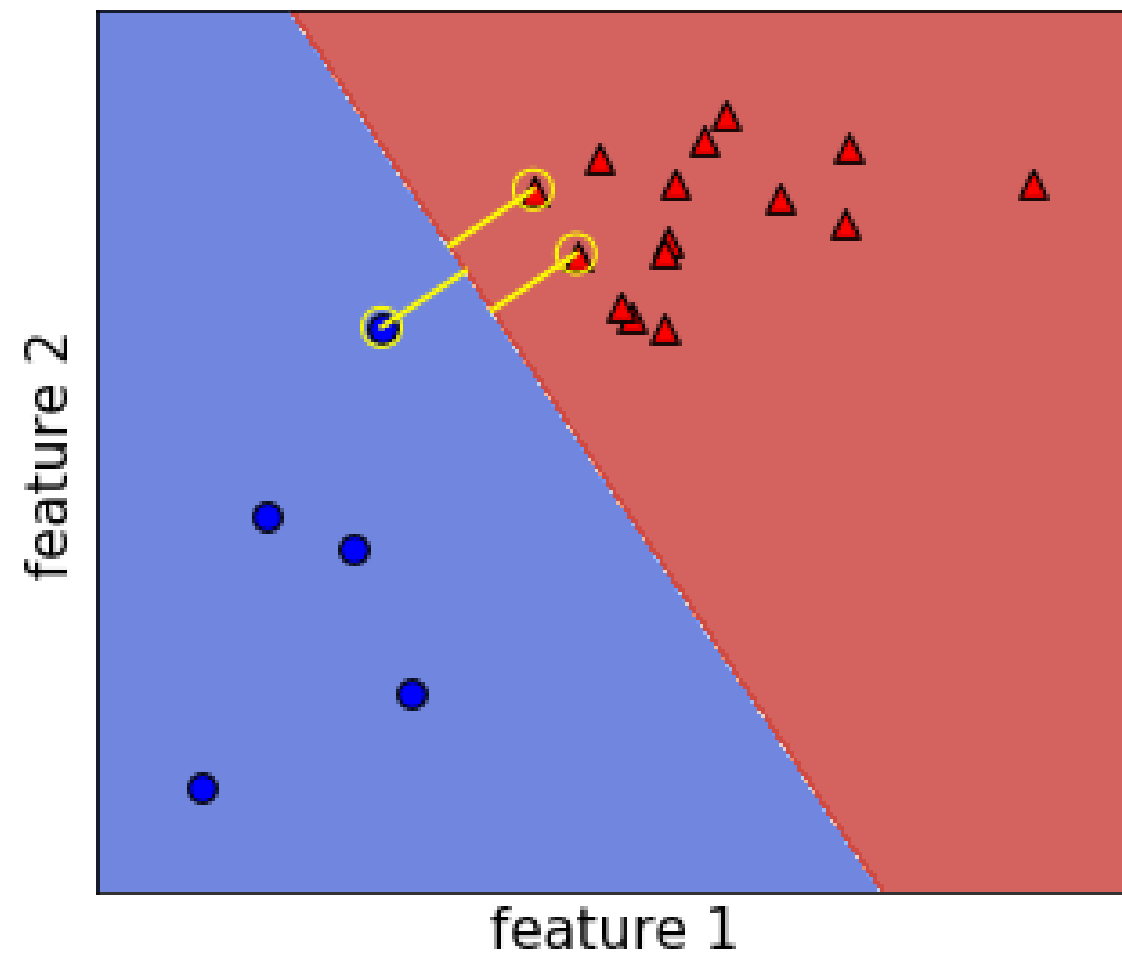
# Max-margin viewpoint

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



# Max-margin viewpoint

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



# Let's practice!

## LINEAR CLASSIFIERS IN PYTHON

# Kernel SVMs

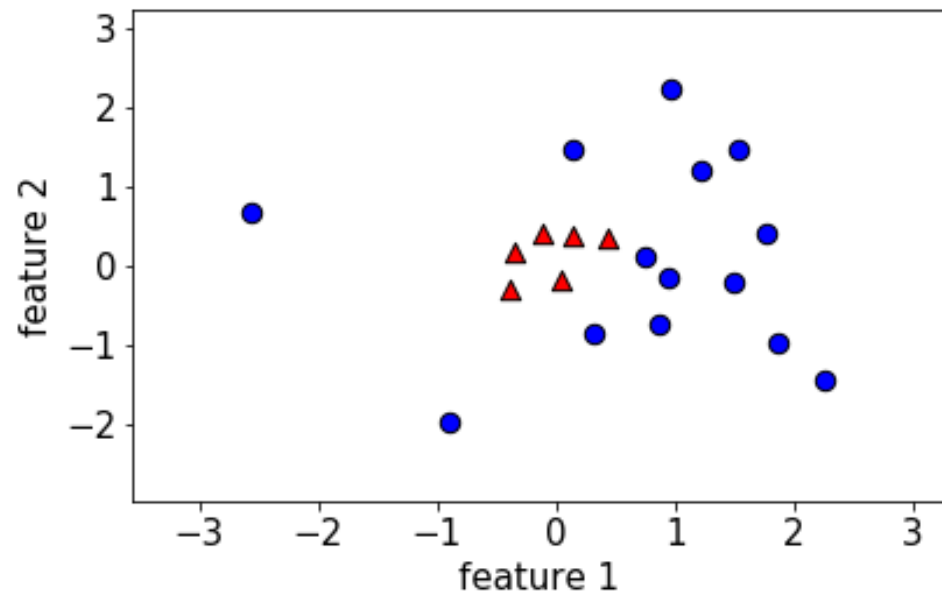
LINEAR CLASSIFIERS IN PYTHON



**Michael (Mike) Gelbart**

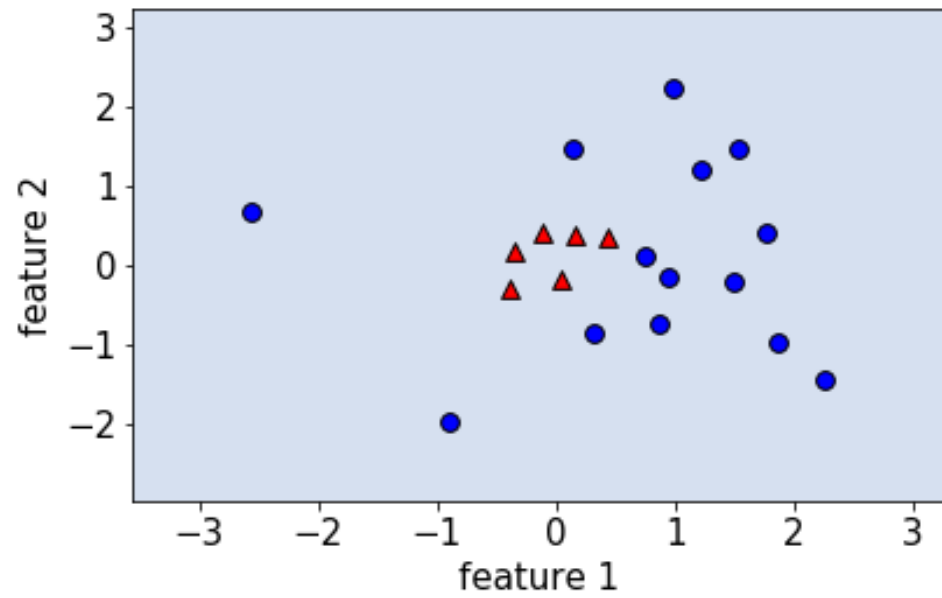
Instructor, The University of British  
Columbia

# Transforming your features

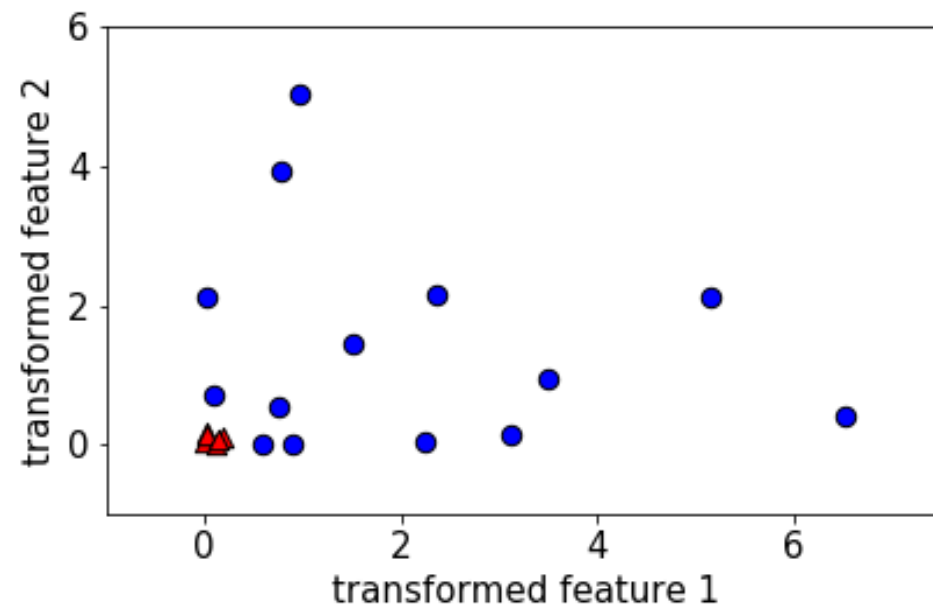
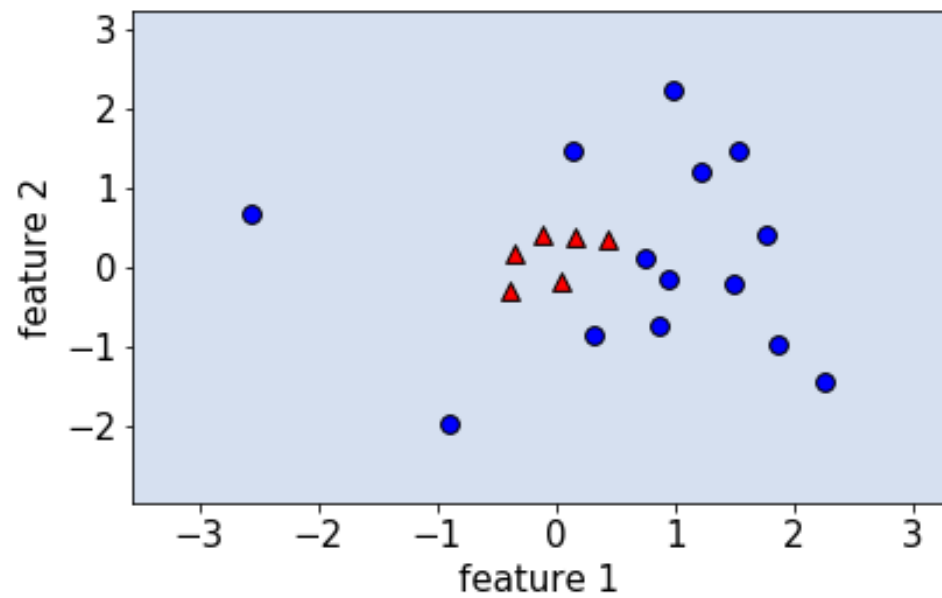




# Transforming your features

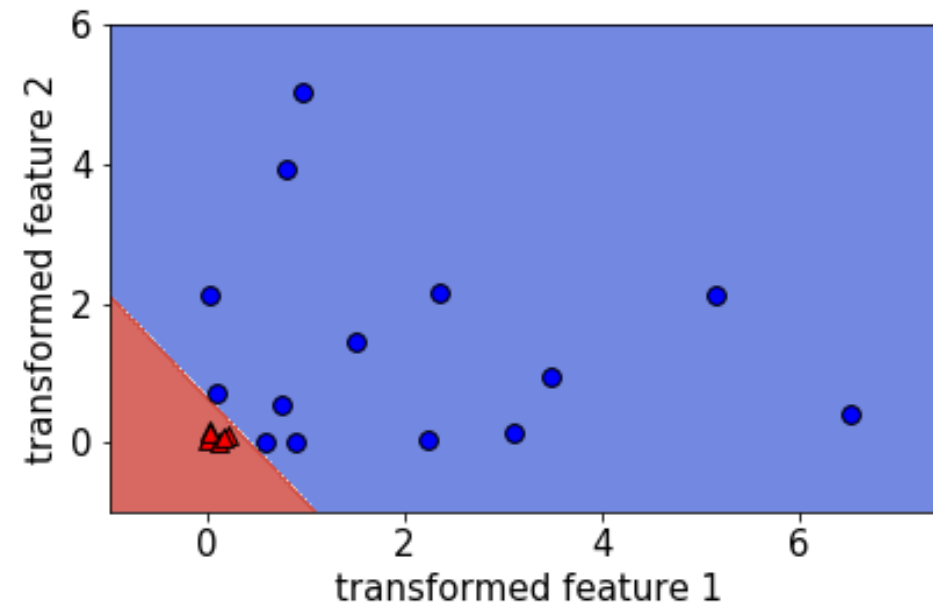
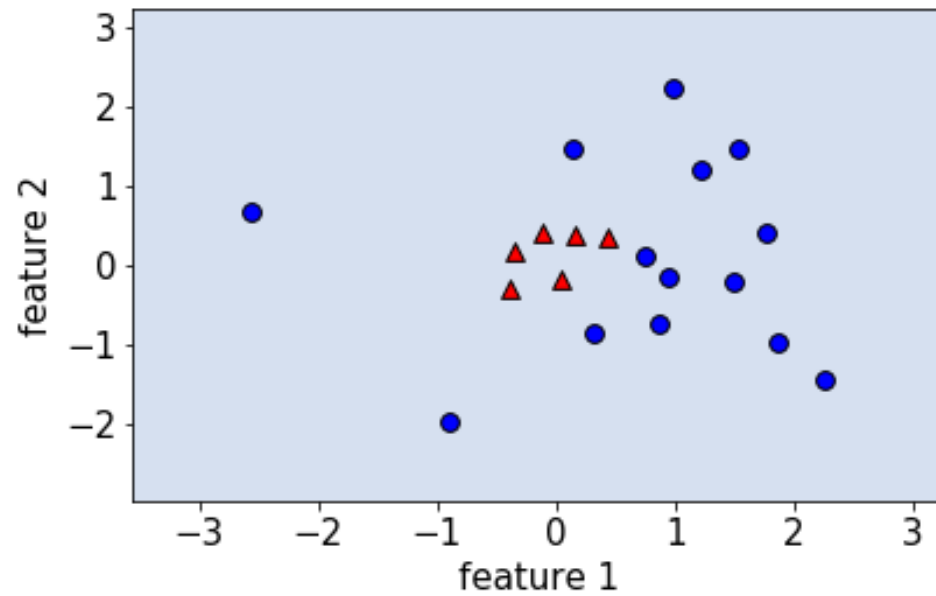


# Transforming your features



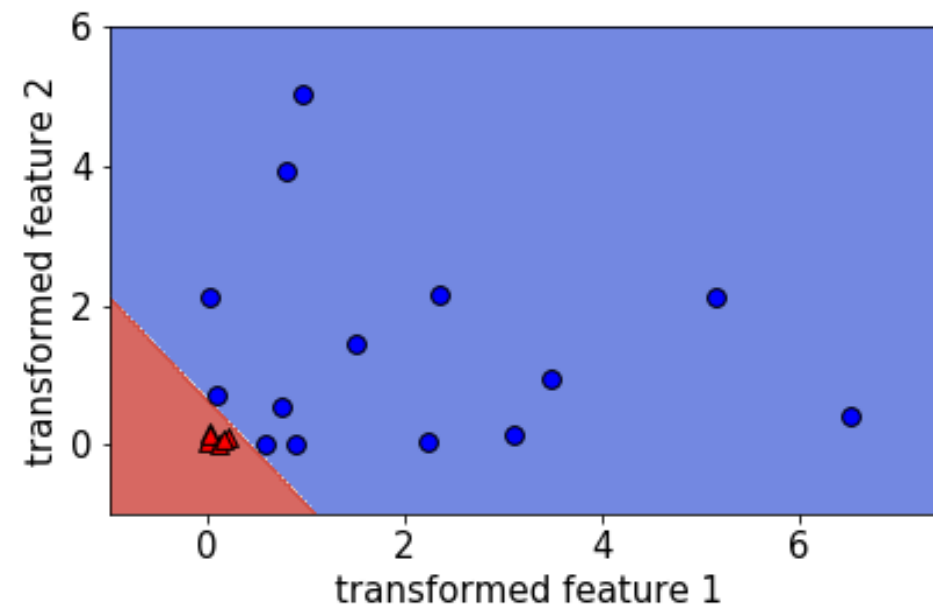
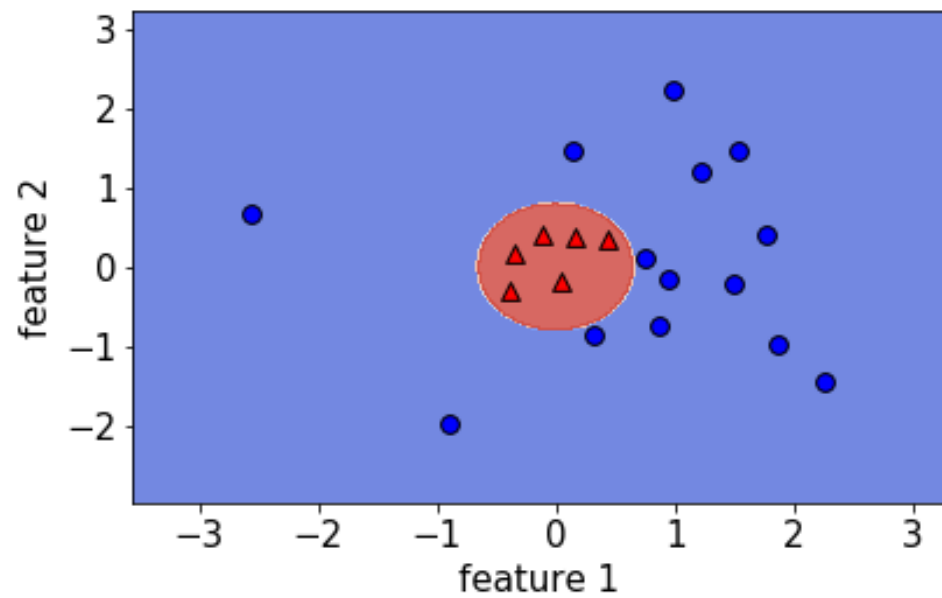
transformed feature =  
 $(\text{original feature})^2$

# Transforming your features



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

# Transforming your features

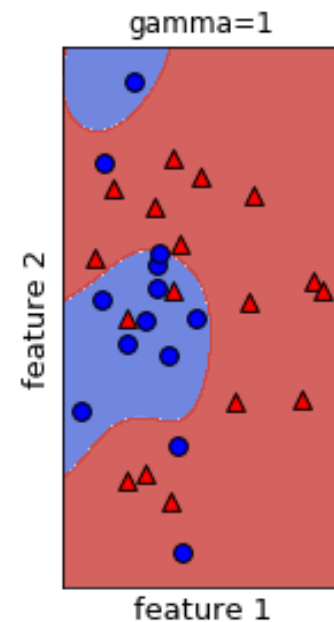


transformed feature =  
 $(\text{original feature})^2$

# Kernel SVMs

```
from sklearn.svm import SVC
```

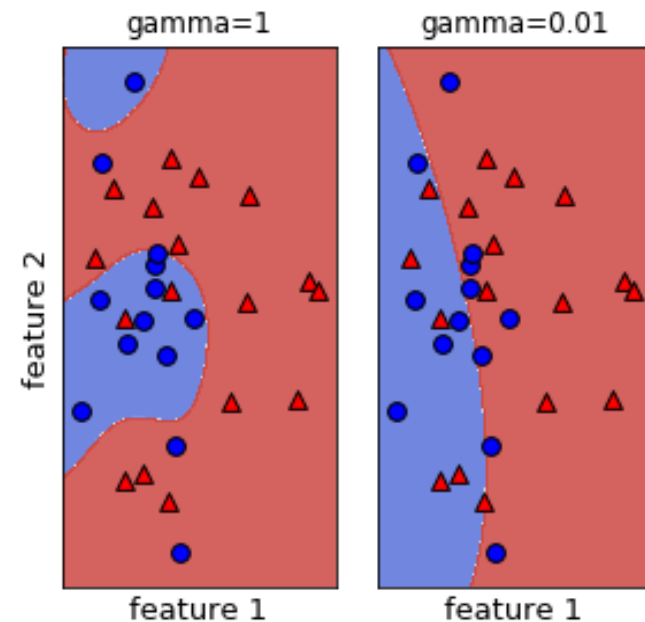
```
svm = SVC(gamma=1)      # default is kernel="rbf"
```



# Kernel SVMs

```
from sklearn.svm import SVC
```

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

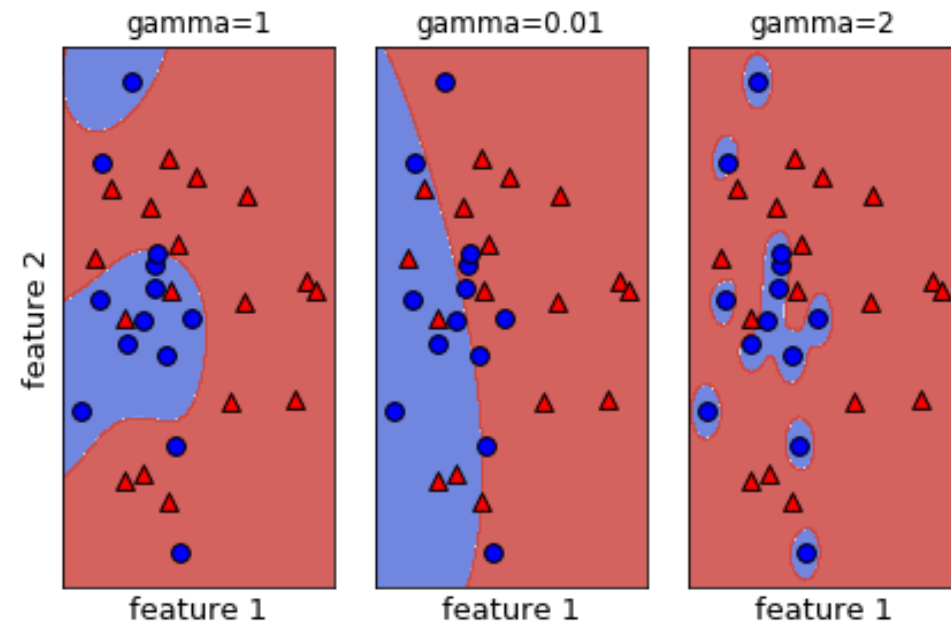


- smaller `gamma` leads to smoother boundaries

# Kernel SVMs

```
from sklearn.svm import SVC
```

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



- 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

## 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 regularization

# Use in scikit-learn

Logistic regression in sklearn:

- `linear_model.LogisticRegression`

Key hyperparameters in sklearn:

- `C` (inverse regularization strength)
- `penalty` (type of regularization)
- `multi_class` (type of multi-class)

SVM in sklearn:

- `svm.LinearSVC` and `svm.SVC`

# Use in scikit-learn (cont.)

Key hyperparameters in sklearn:

- `C` (inverse regularization strength)
- `kernel` (type of kernel)
- `gamma` (inverse RBF smoothness)

# SGDClassifier

`SGDClassifier` : scales well to large datasets

```
from sklearn.linear_model import SGDClassifier
```

```
logreg = SGDClassifier(loss='log_loss')
```

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

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

# Let's practice!

## LINEAR CLASSIFIERS IN PYTHON

# Conclusion

LINEAR CLASSIFIERS IN PYTHON



**Michael (Mike) Gelbart**

Instructor, The University of British  
Columbia

# 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