Chapter 5. Support Vector Machines

CSC357 Advanced Topics—Machine Learning

25 January 2020

- *Support Vector Machine (SVM)*—a powerful and versatile Machine Learning model
- capable of linear or nonlinear classification, regression, and even outlier detection
- one of the most popular models in Machine Learning
- well-suited for classification of complex small- or medium-sized datasets
- our goals...
  - explain core concepts of SVMs
  - learn how to use them
  - learn how they work
- Linear SVM Classification
  - explain fundamental idea with pictures (figure 5-1)
  - iris data set
  - 2 classes easily separated w/ straight line
  - classes are *linearly separable*
  - left plot → decision boundaries of 3 possible linear classifiers
  - model w/ decision boundary shown by dashed line → bad / does not properly separate classes
  - other 2 models work on training set—decision boundaries close to instances—models will probably not perform as well on new instances
  - solid line in plot on right is decision boundary of an SVM classifier
    - separates the 2 classes
    - also stays as far from closest training instances as possible
  - SVM like fitting widest possible street between classes
  - (called *large margin classification*)

  - adding training instances "off the street" will not affect decision boundary
  - determined ("supported") by instances located on street's edge
  - these instances are support vectors
  - SVMs sensitive to feature scales
  - prepare data w/ (for example) Scikit-Learns StandardScaler

- Soft Margin Classification
  - *hard margin classification* → all instances "off street"
    - only works if data is linearly separable
    - sensitive to outliers
    - figure 5-3 shows iris dataset w/ just one additional outlier
    - on left, impossible to find a hard margin
    - on right, decision boundary very different from one without the outlier, also probably will not generalize
  - use a more flexible model
  - find balance between keeping street large as possible and limiting the *margin violations*
  - *margin violations* →
  - instances in middle of street, or wrong side
  - this is *soft margin classification*
  - Scikit-Learn's hyperparameter C
  - low value → more margin violations, but maybe a model that generalizes better
  - if SVM model is overfitting, try regularizing it by reducing C
  - here's Scikit-Learn code that...
    - loads the iris dataset
    - scales features
    - trains a linear SVM model
    - (uses LinearSVC class with C=1 and hinge loss function)
    - to detect *Iris virginica* flowers

```
import numpy as np
from sklearn import datasets
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.svm import LinearSVC

iris = datasets.load_iris()
X = iris["data"][:, (2, 3)]  # petal length, petal width
```

```
y = (iris["target"] == 2).astype(np.float64)  # Iris virginica

svm_clf = Pipeline([
        ("scaler", StandardScaler()),
        ("linear_svc", LinearSVC(C=1, loss="hinge")),
    ])

svm_clf.fit(X, y)
```

- use to make predictions...

```
svm_clf.predict([[5.5, 1.7]])
array([1.])
```

- unlike Logistic Regression classifiers, SVM classifiers do not output probabilities for each class
- instead of using LinearSVC class, could use the SVC class with a linear kernel
- create SVC model by writing SVC(kernel="linear", C=1)
- or could use SGDClassifier class, with SGDClassifier(loss="hinge", alpha=1/(m∗C))
  - ∗ applies regular Stochastic Gradient Descent to train a linear SVM classifier
  - ∗ does not converge as fast as LinearSVC class
  - ∗ but could andle online classification tasks or huge datasets that do not fit in memory (out-of-core training)