17.5.2

SVM in Practice

Although the ideas behind support vector machines are different from those behind logistic regression, actually implementing a SVM model is very similar to what you have done. As before, you will split your dataset, create and train a model, create predictions, then validate the model.

Now that we have looked at how an SVM model works, let's look at using SVM in practice. To get started, download the following files.

Download 17-5-2-svm.zip (https://2u-data-curriculum-team.s3.amazonaws.com/dataviz-online/module_17/17-5-2-svm.zip)

Open the notebook and load the dataset:

```
from path import Path
import numpy as np
import pandas as pd

data = Path('../Resources/loans.csv')
df = pd.read_csv(data)
df.head()
```

Each row in the dataset represents an application for a loan, and information is available on the applicant's assets, liabilities, income, credit score, and mortgage size. We also have information on whether the application was approved or denied. Here, the target variable is `status`, and all other columns are features used to predict the loan application status.

It's worth noting that the data in this dataset have been normalized. In this case, the data in the numerical features, such as assets and liabilities, have been scaled to be between 0 and 1.

We will discuss scaling in greater detail later, but note for now that some models require scaling the data, and that in this dataset, the scaling has been done for you:

	assets	liabilities	income	credit_score	mortgage	status
0	0.210859	0.452865	0.281367	0.628039	0.302682	deny
1	0.395018	0.661153	0.330622	0.638439	0.502831	approve
2	0.291186	0.593432	0.438436	0.434863	0.315574	approve
3	0.458640	0.576156	0.744167	0.291324	0.394891	approve
4	0.463470	0.292414	0.489887	0.811384	0.566605	approve

The next two steps should be familiar. We separate the dataset into features (X) and target (y):

```
y = df["status"]
X = df.drop(columns="status")
```

We then further split the dataset into training and testing sets. Note that the shape of the training is (75, 5), meaning 75 rows and five columns. It is generally good practice to stratify the data when splitting into training and testing sets, especially when the dataset is small, as is the case here:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,
   y,  random_state=1, stratify=y)
X_train.shape
```

Next, we import the SVC module from Scikit-learn, then instantiate it. The kernel specifies the mathematical functions used to separate the classes. The kernel, in this example, identifies the orientation of the hyperplane as linear. However, a number of kernels exist that define nonlinear boundaries:

```
from sklearn.svm import SVC
model = SVC(kernel='linear')
```

We then train the model with `fit()`:

```
model.fit(X_train, y_train)
```

Next, we create predictions with the model:

```
y_pred = model.predict(X_test)
results = pd.DataFrame({
   "Prediction": y_pred,
   "Actual": y_test
}).reset_index(drop=True)
results.head()
```

We assess the `accuracy_score` of the model, which is 0.6:

```
from sklearn.metrics import accuracy_score
accuracy_score(y_test, y_pred)
```

We then generate a `confusion_matrix` and print the classification report:

```
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test, y_pred)

from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

     approve       0.58      0.58      0.58        12
        deny       0.62      0.62      0.62        13

    accuracy                           0.60        25
   macro avg       0.60      0.60      0.60        25
weighted avg       0.60      0.60      0.60        25
```

In summary, much of using a SVM model in practice follows the pattern we saw with logistic regression: split the dataset, create a model, train the model, create predictions, then validate the model.

After looking over the summary of the results in the classification report, answer the following question. What is the recall rate of the approve category?

- ○ 0.58
- ○ 0.71
- ○ 0.89

Check Answer

Finish ▶

SKILL DRILL

Assess the performance of a logistic regression model, namely the precision, recall, and F1 scores for the `approve` category. Compare it with the performance of the SVM model. Which model performs better?