Combinando Modelos (Ensemble Learning)

Classificadores por Votação

Slides extraídos do livro "*Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*" de Aurélien Géron

Sumário

Aprendizagem por Conjuntos (Ensemble Learning)

- Suponha que você faça uma pergunta complexa a milhares de pessoas aleatórias e depois agregue suas respostas. Em muitos casos, você descobrirá que essa resposta agregada é melhor que a resposta de um especialista. Isso é chamado de **sabedoria da multidão**.

- Da mesma forma, se você agregar as previsões de um grupo de preditores (como classificadores ou regressores), muitas vezes obterá melhores previsões do que com o melhor preditor individual.

- Um grupo de preditores é chamado de um **ensemble**; portanto, essa técnica é chamada de **Aprendizagem por Conjuntos (Ensemble Learning)**, e um algoritmo de Aprendizagem por Conjuntos é chamado de **Método de Ensemble**.

- Por exemplo, um conjunto de Árvores de Decisão é chamado de **Floresta Aleatória (Random Forest)**.

Sumário

Classificadores por Votação (Voting Classifiers)

Suponha que você tenha treinado alguns classificadores diversos:

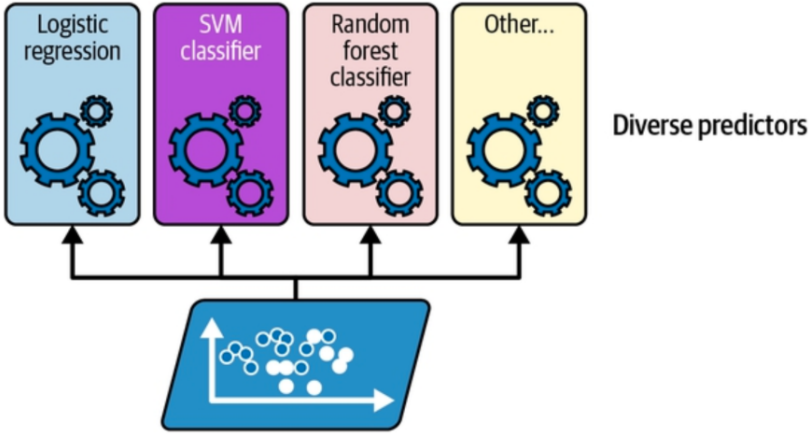

Figure: Treinando classificadores diversos.

Classificador por Votação Majoritária (Hard Voting)

Uma maneira muito simples de criar um classificador ainda melhor é agregar as previsões de cada classificador e prever a classe que obtém mais votos.

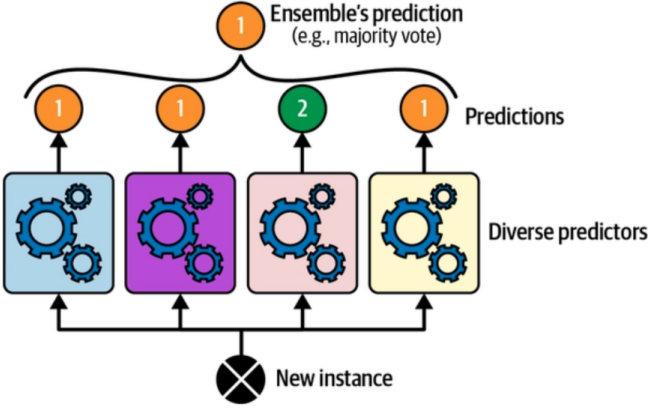

Figure: Previsões de um classificador por votação majoritária.

A Lógica por Trás da Votação

- Classificadores por votação frequentemente alcançam acurácia maior que o melhor classificador individual.
- Mesmo que cada classificador seja um *weak learner*, o conjunto ainda pode ser um *strong learner*.
  - *Weak learner*: classif. apenas ligeiramente melhor que palpite aleatório (ex.: 51% acurácia p/ tarefa binária).
- Isso se deve à **lei dos grandes números**. Se você tem uma moeda ligeiramente viciada (51% de chance de cara), após 1.000 lançamentos, a probabilidade de obter a maioria de caras é de cerca de 75%.
  - Ou seja, é mais difícil prever o lado de maior probabilidade usando poucos lançamentos.
  - Da mesma forma, é mais difícil acertar a classe correta com poucos modelos.

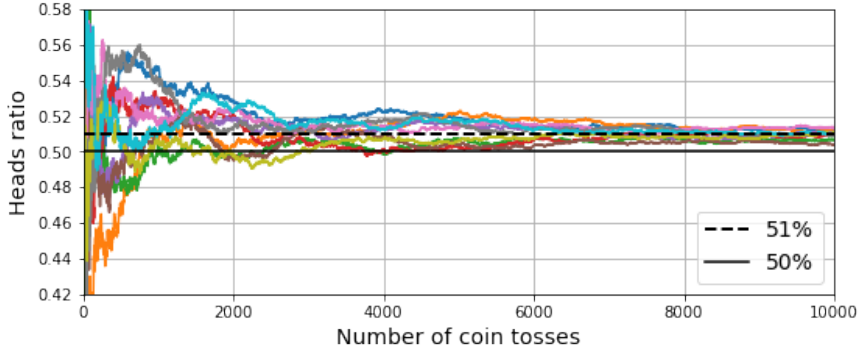

Figure: Aumentando o número de lançamentos, a proporção de caras se aproxima de 51%.

A Lógica por Trás da Votação

Condição

Isso só é verdade se todos os classificadores forem perfeitamente independentes, cometendo erros não correlacionados. Uma maneira de obter classificadores diversos é treiná-los usando algoritmos muito diferentes.

Implementação com Scikit-Learn

O Scikit-Learn fornece uma classe `VotingClassifier` que é bastante fácil de usar.

```
from sklearn.ensemble import RandomForestClassifier, VotingClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.datasets import make_moons
from sklearn.model_selection import train_test_split

X, y = make_moons(n_samples=500, noise=0.30, random_state=42)
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=42)

voting_clf = VotingClassifier(
    estimators=[
        ('lr', LogisticRegression(random_state=42)),
        ('rf', RandomForestClassifier(random_state=42)),
        ('svc', SVC(random_state=42))
    ]
)
voting_clf.fit(X_train, y_train)
```

Performance da Votação Majoritária (Hard Voting)

Vamos olhar a acurácia de cada classificador no conjunto de teste:

```
>>> for name, clf in voting_clf.named_estimators_.items():
...     print(name, "=", clf.score(X_test, y_test))
...
lr = 0.864
rf = 0.896
svc = 0.896
```

Agora, a performance do classificador por votação (hard voting):

```
>>> voting_clf.score(X_test, y_test)
0.912
```

O classificador por votação supera todos os classificadores individuais.

Votação Suave (Soft Voting)

Soft Voting

Se todos os classificadores puderem estimar probabilidades de classe (tiverem um método `predict_proba()`), você pode prever a classe com a maior probabilidade de classe, calculando a média sobre todos os classificadores individuais. Isso é chamado de votação suave.

```
>>> voting_clf.voting = "soft"
>>> voting_clf.named_estimators["svc"].probability = True # Precisa habilitar no SVC
>>> voting_clf.fit(X_train, y_train)
>>> voting_clf.score(X_test, y_test)
0.92
```

Fim