Introduction to ensemble methods

ENSEMBLE METHODS IN PYTHON

Román de las Heras
Senior Data Scientist, Chartboost

datacamp

Choosing the best model

Classifier	Accuracy
Decision Tree	75%
Logistic Regression	72%
K-Nearest Neighbors	74%

Surveys

Classifier	Accuracy
Decision Tree	75%
Logistic Regression	72%
K-Nearest Neighbors	74%

Ensemble Methods

Classifier	Accuracy
Combined Model	79%

Prerequisite knowledge

- Supervised Learning with scikit-learn
- Machine Learning with Tree-Based Models in Python
- Linear Classifiers in Python

Technologies

- `scikit-learn`
- `numpy`
- `pandas`
- `seaborn`


```
from sklearn.ensemble import MetaEstimator
# Base estimators
est1 = Model1()
est2 = Model2()
estN = ModelN()
# Meta estimator
est_combined = MetaEstimator(
   estimators=[est1, est2, ..., estN],
   # Additional parameters
)
# Train and test
est_combined.fit(X_train, y_train)

pred = est_combined.predict(X_test)
```

Learners, ensemble!

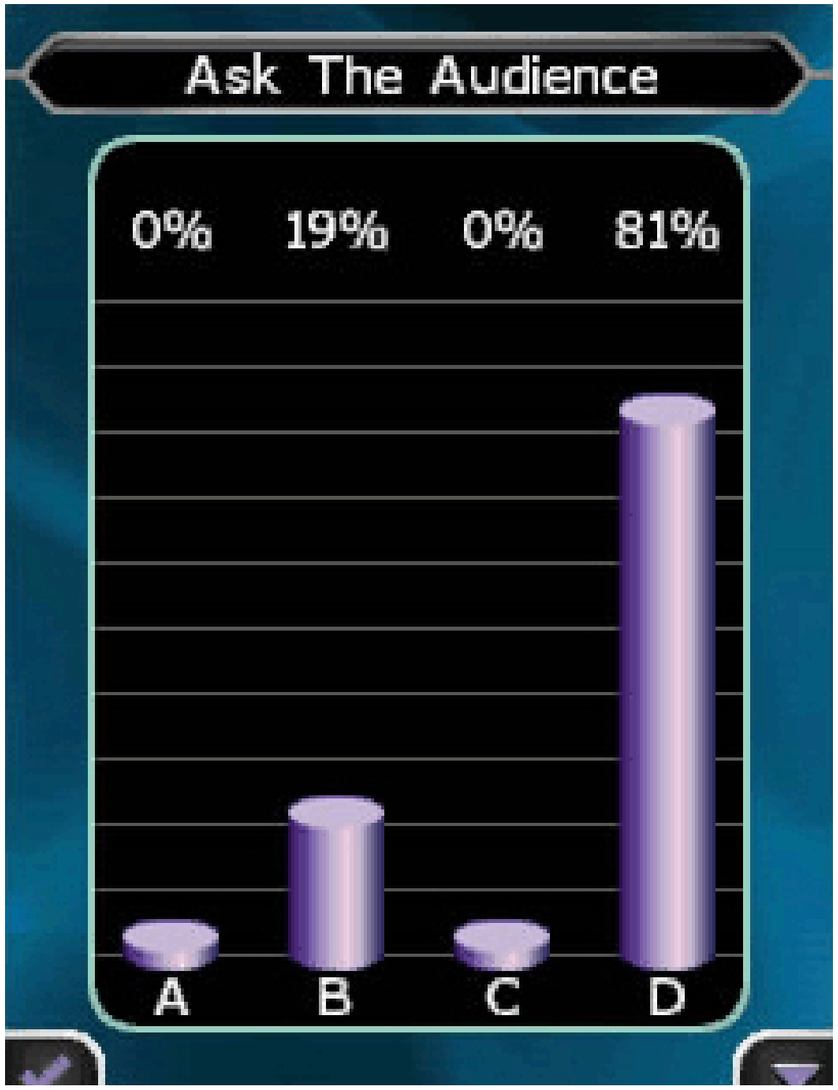
ENSEMBLE METHODS IN PYTHON

Voting

ENSEMBLE METHODS IN PYTHON

Román de las Heras
Senior Data Scientist, Chartboost

Ask the audience



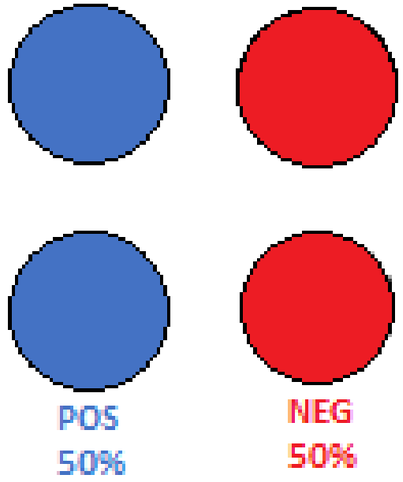
Wisdom of the crowd

- Collective intelligence
- Large group of individuals >= Single expert
- Problem solving
- Decision making
- Innovation
- Prediction

Majority voting

Properties

- Classification problems
- Majority Voting: Mode
- Odd number of classifiers (3+)



Wise Crowd Characteristics:

- Diverse: different algorithms or datasets
- Independent and uncorrelated
- Use individual knowledge
- Aggregate individual predictions

Voting ensemble using scikit-learn

```
from sklearn.ensemble import VotingClassifier
clf_voting = VotingClassifier(
    estimators=[
       ('label1', clf_1),
       ('label2', clf_2),
       ('labelN', clf_N)])
```

Evaluate the performance

```
# Get the accuracy score
acc = accuracy_score(y_test, y_pred)
print("Accuracy: {:0.3f}".format(acc))
```

```
Accuracy: 0.938
```

```
# Create the individual models
clf_knn = KNeighborsClassifier(5)
clf_dt = DecisionTreeClassifier()
clf_lr = LogisticRegression()
# Create voting classifier
clf_voting = VotingClassifier(
    estimators=[
       ('knn', clf_knn),
       ('dt', clf_dt),
       ('lr', clf_lr)])
# Fit it to the training set and predict
clf_voting.fit(X_train, y_train)
y_pred = clf_voting.predict(X_test)
```

Let's give it a try!

ENSEMBLE METHODS IN PYTHON

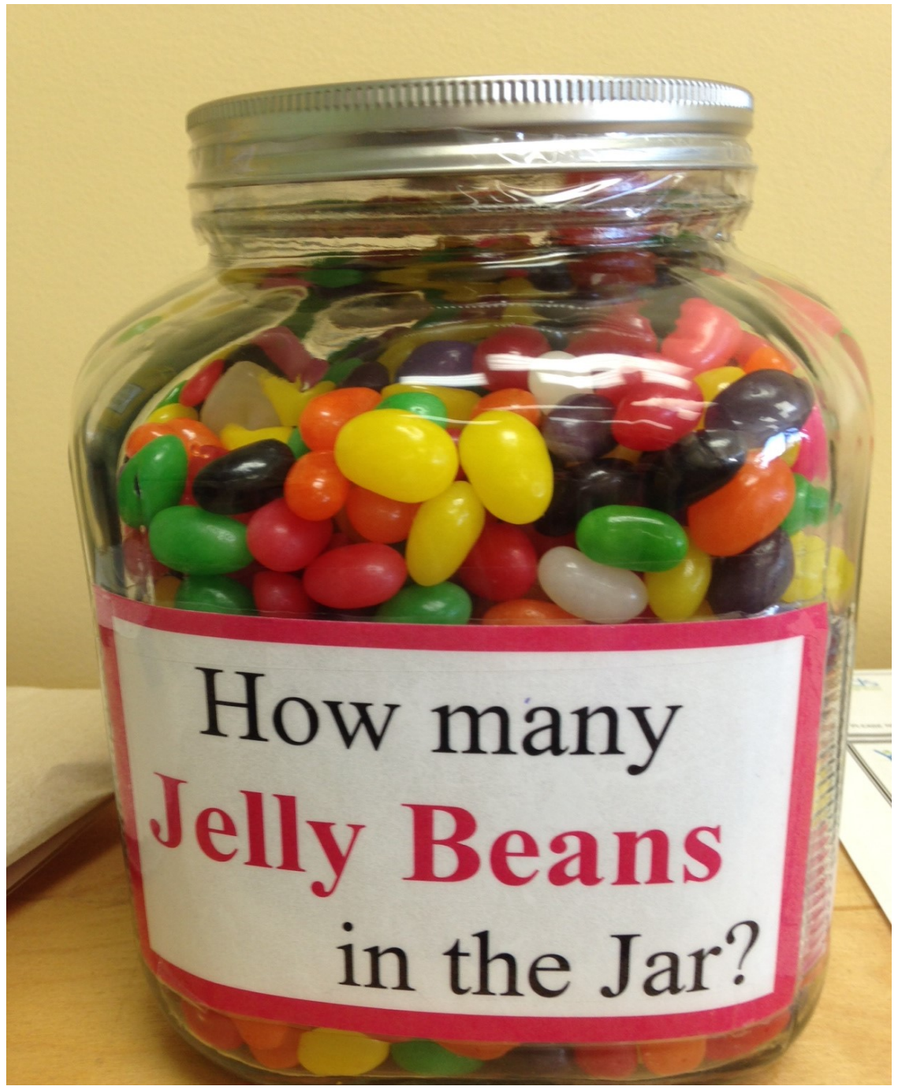
Averaging

ENSEMBLE METHODS IN PYTHON

Román de las Heras
Senior Data Scientist, Chartboost

datacamp

Counting Jelly Beans



How to provide a good estimate?

- Guessing (random number)
- Volume approximation
- Many more approaches

Actual Value ~ mean(estimates)

Averaging (Soft Voting)

Properties

- Classification & Regression problems
- Soft Voting: Mean
  - **Regression:** mean of predicted values
  - **Classification:** mean of predicted probabilities
- Need at least 2 estimators

Averaging ensemble with scikit-learn

Averaging Classifier

```
from sklearn.ensemble import VotingClassifier
clf_voting = VotingClassifier(

    estimators=[
       ('label1', clf_1),
       ('label2', clf_2),
       ...
       ('labelN', clf_N)],

    voting='soft',

    weights=[w_1, w_2, ..., w_N]
)
```

Averaging Regressor

```
from sklearn.ensemble import VotingRegressor
reg_voting = VotingRegressor(

    estimators=[
       ('label1', reg_1),
       ('label2', reg_2),
       ...
       ('labelN', reg_N)],

    weights=[w_1, w_2, ..., w_N]
)
```

scikit-learn example

```
# Instantiate the individual models
clf_knn = KNeighborsClassifier(5)
clf_dt = DecisionTreeClassifier()
clf_lr = LogisticRegression()
```

```
# Create an averaging classifier
clf_voting = VotingClassifier(
    estimators=[
       ('knn', clf_knn),
       ('dt', clf_dt),
       ('lr', clf_lr)],
    voting='soft',
    weights=[1, 2, 1]
)
```

Game of Thrones deaths

Target:

- Predict whether a character is alive or not

Features:

- Age
- Gender
- Books of appearance
- Popularity
- Whether relatives are alive or not

Time to practice!

ENSEMBLE METHODS IN PYTHON