Soft computing

K.N. Toosi University of Technology

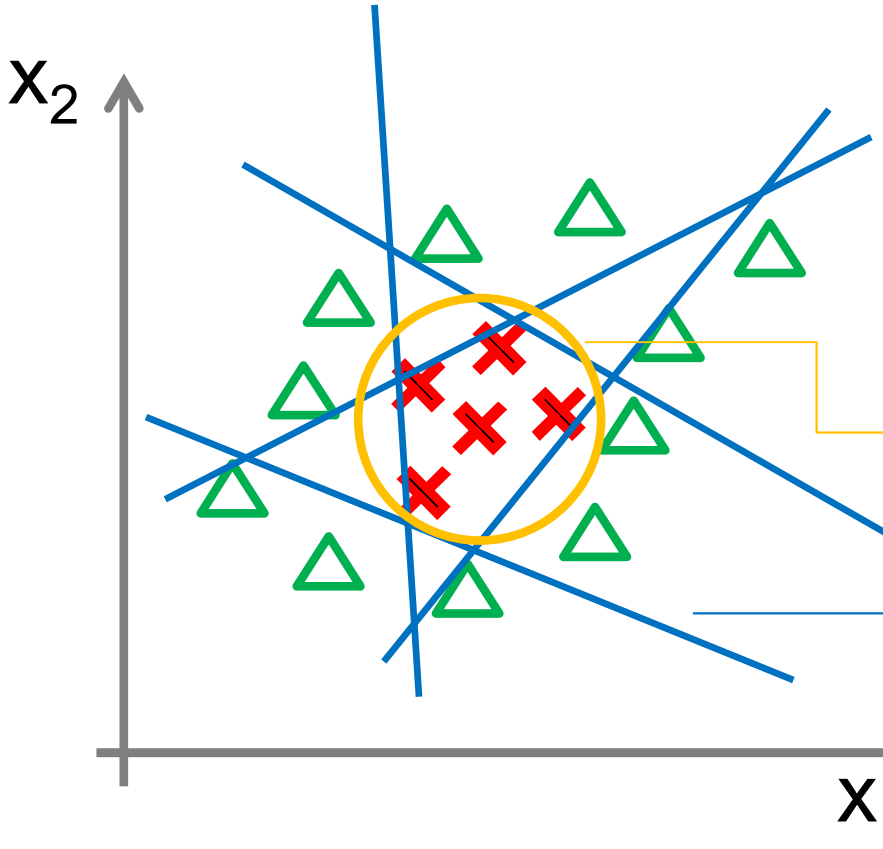
Hasan Ghasemzadeh
http://wp.kntu.ac.ir/ghasemzadeh

یادگیری جمعی(Ensemble Learning)

Decision Tree- Random Forest درخت تصمیم و جنگل تصادفی

یادگیری جمعی (Ensemble Learning)

آیا با یک رگرسیون ساده می‌توان داده های زیر را دسته بندی کرد؟



استفاده از مدلهای ساده و رای گیری بین مدلها جهت طبقه بندی

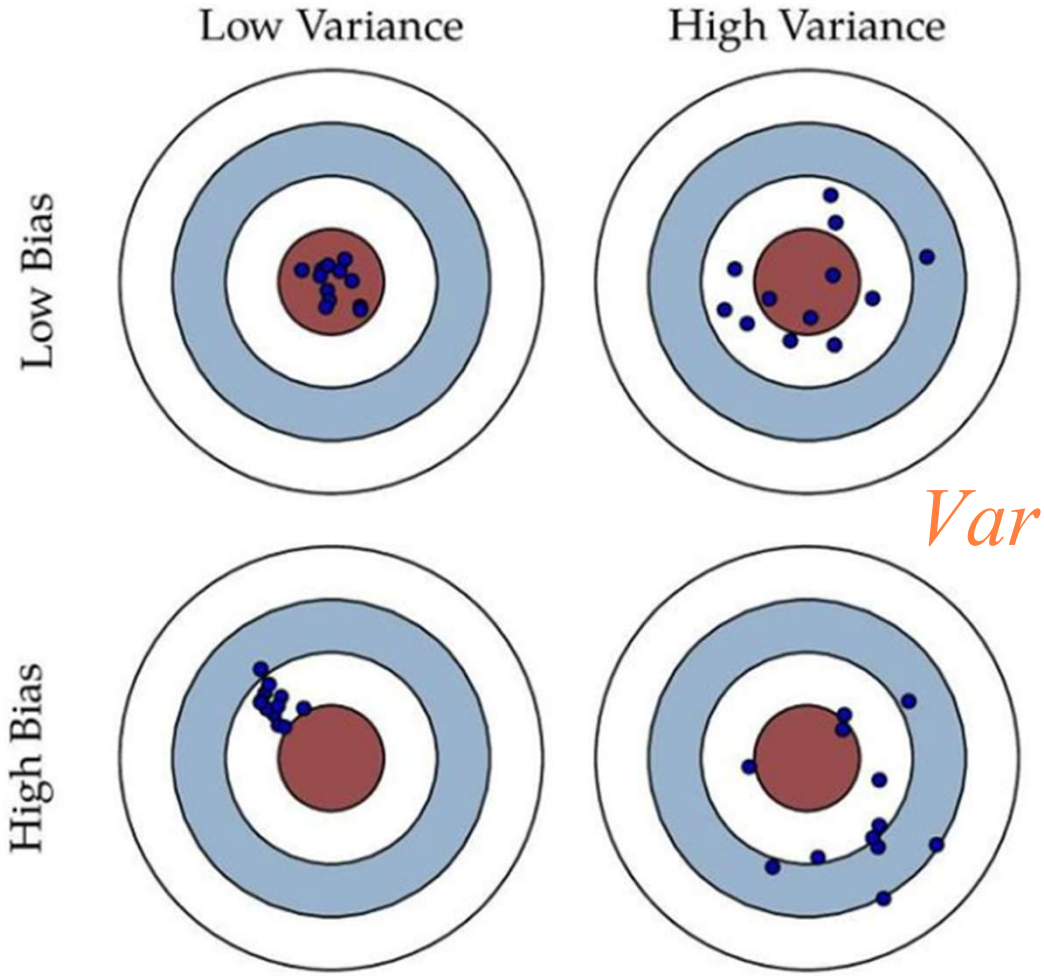
مدل پیچیده
(Strong learner)

مدل ساده
(Weak learner)

این روش برای داده‌های جدولی مناسب است

واریانس و بایاس

یک مدل در پیش‌بینی‌ها دارای واریانس(پراکندگی) و بایاس (اریبی) است.
در مدلهای مختلف سعی بر حداقل کردن این مقادیر است.



$$Bias(h(x)) = E[h(x)] - y$$

$$Var(h(x)) = E\left[\left(h(x) - E[h(x)]\right)^2\right]$$

y مقدار واقعی

$h(x)$ مقدار پیش‌بینی مدل – تابع فرض

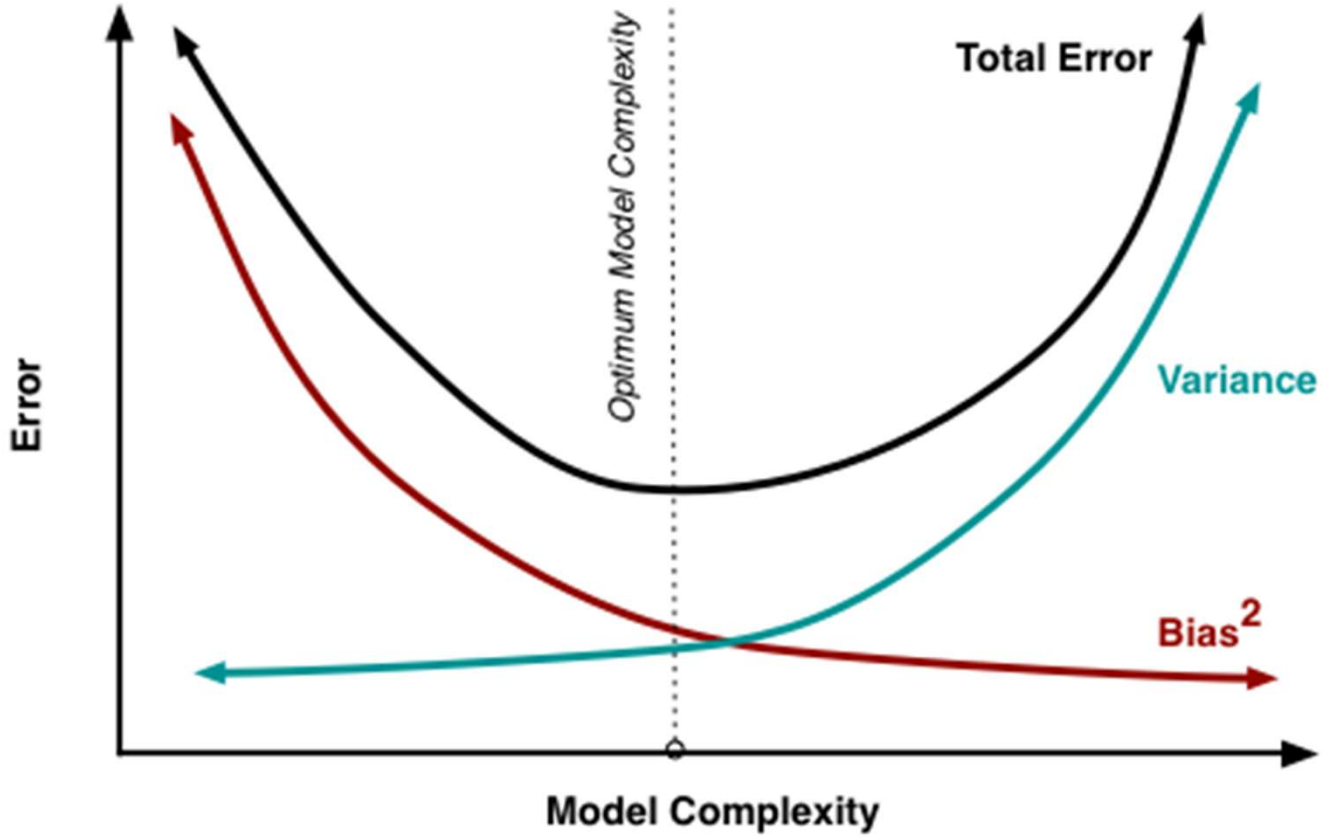
$E()$ امید(میانگین) مقادیر

واریانس و بایاس

در مدلهای ساده بایاس زیاد و واریانس کم است.

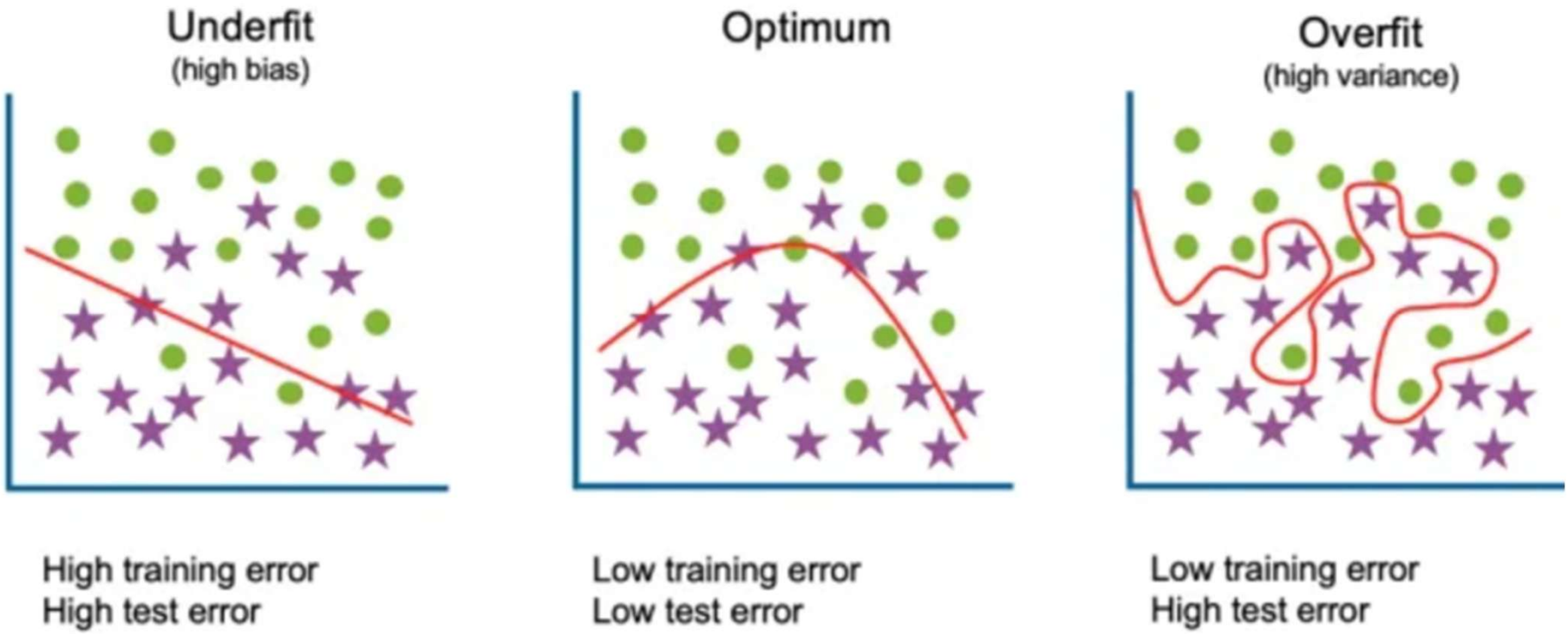
با پیچیده شدن مدل بایاس کاهش و واریانس افزایش پیدا می کند

در مدل بدنبال شرایط بهینه این مقادیر هستیم.



واریانس و بایاس

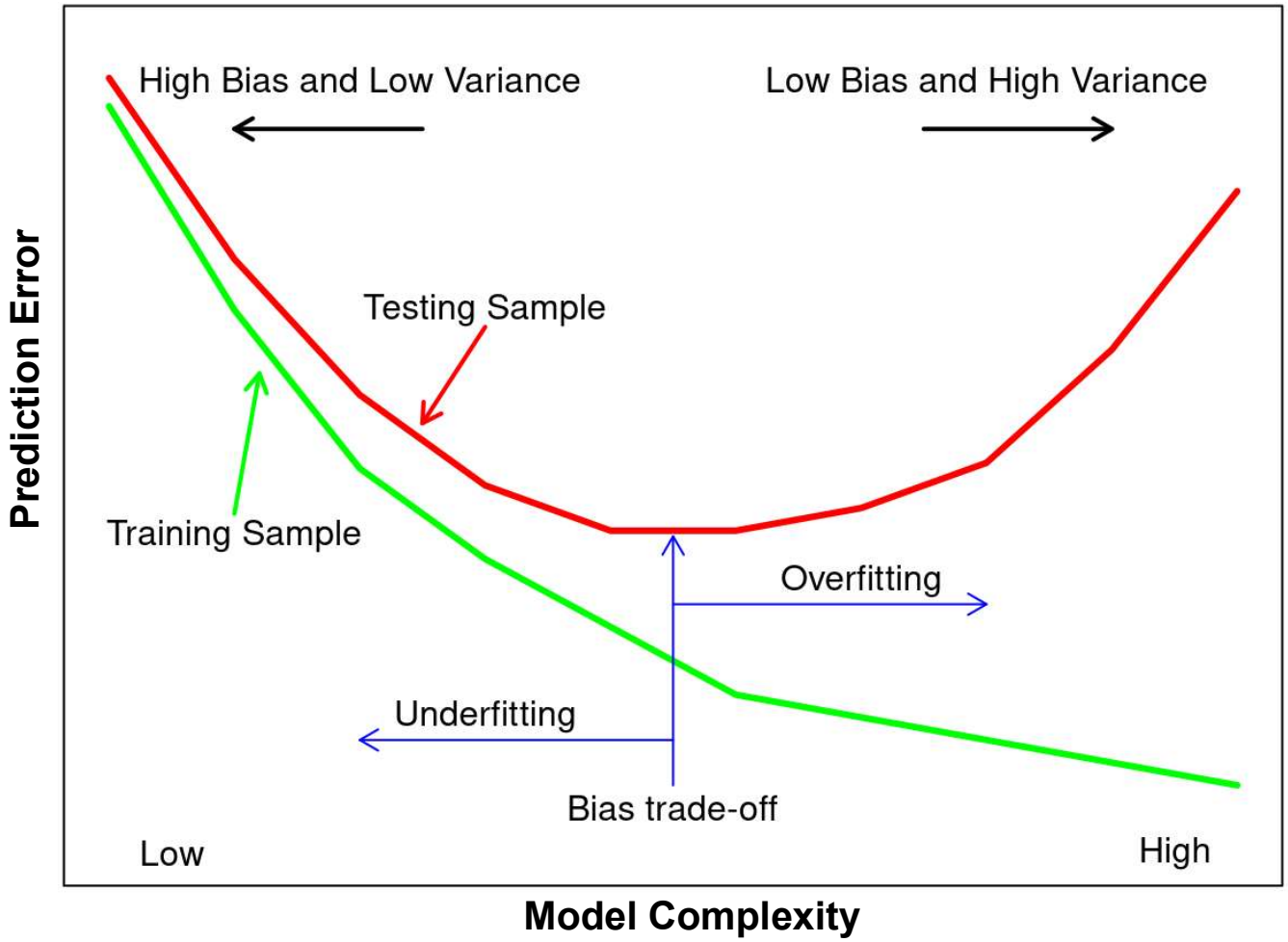
با ایجادتوازن بین بایاس و واریانس می‌توان مدل بهینه‌ای یافت که از کم‌برازش و بیش‌برازش جلوگیری کند.



واریانس و بایاس

بایاس یا اریبی: خطای کلی مرتبط با داده های آموزشی

واریانس: خطای کلی مرتبط با داده های آزمایشی



واریانس و بایاس چند مدل نسبت به هم

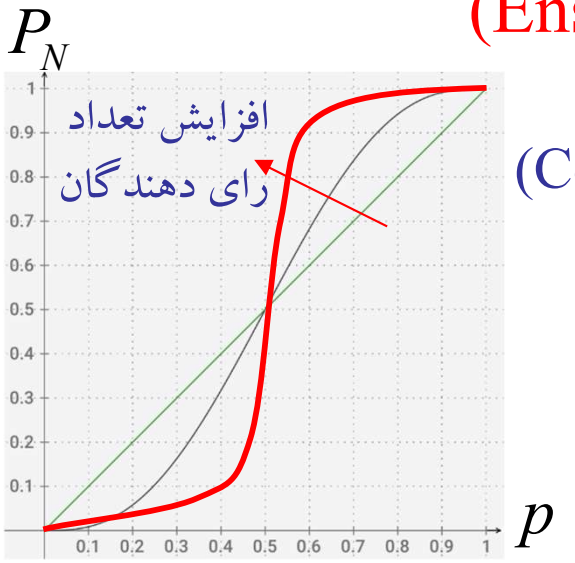
اگر چندین مدل پیش‌بینی داشته باشیم مدلها نیز نسبت به یکدیگر واریانس و بایاس خواهند داشت.

در یادگیری جمعی یک توازن بین بایاس و واریانس بایستی ایجاد کرد

$$MSE = \left(E[h(x)] - y\right)^2 + E\left[\left(h(x) - E[h(x)]\right)^2\right] + \sigma_e^2$$

خطای مربع میانگین — بایاس — واریانس — خطای غیرقابل کاهش به دلیل نوفه داده‌ها

یادگیری جمعی (Ensemble Learning)



قضیه هیئت منصفه کندورسه (Condorcet's jury theorem, 1785)

N رأی‌دهنده با احتمال مستقل p برای دو حالت
درست و غلط می‌دهند.

- اگر شرکت کنندگان با احتمال بیشتری درست رای دهند($p>0.5$)
وقتی تعداد رای دهندگان به بی نهایت میل کند احتمال صحت تصمیم گیری به یک میل می کند

$$p > 0.5 \Rightarrow \lim_{N\to\infty} P_N = 1$$

- اگر شرکت کنندگان با احتمال بیشتری نادرست رای دهند($p<0.5$)
وقتی تعداد رای دهندگان به کمترین حد یعنی یک نفر برسد حالت بهینه داریم.

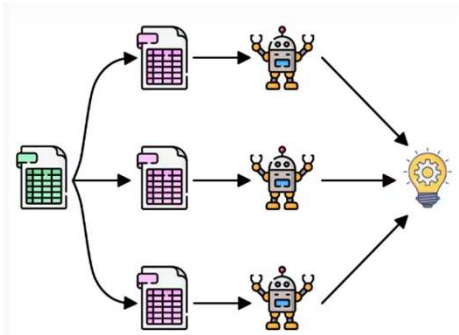
$$p < 0.5 \Rightarrow \lim_{N\to\infty} P_N = 0$$

احتمال درستی رای اکثریت

$$P_N = \sum_{i=(N+1)/2}^{N} \frac{N!}{(N-i)!i!} p^i (1-p)^{N-i}$$

یادگیری جمعی (Ensemble Learning)

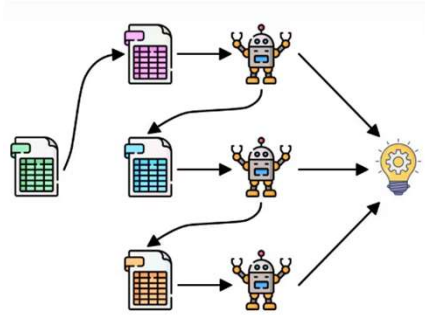
در یادگیری جمعی از الگوریتم‌های مختلف برای پیش‌بینی نتایج استفاده می‌شود.



PARALLEL

BAGGING

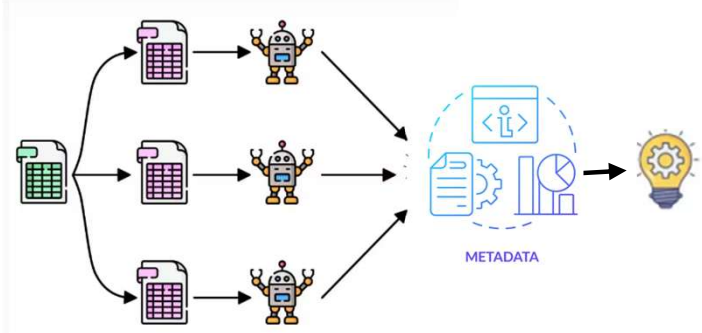
Bootstrap aggregation



SEQUENTIAL

BOOSTING

Iterative Learning



STACKING

STACKING

and Blending

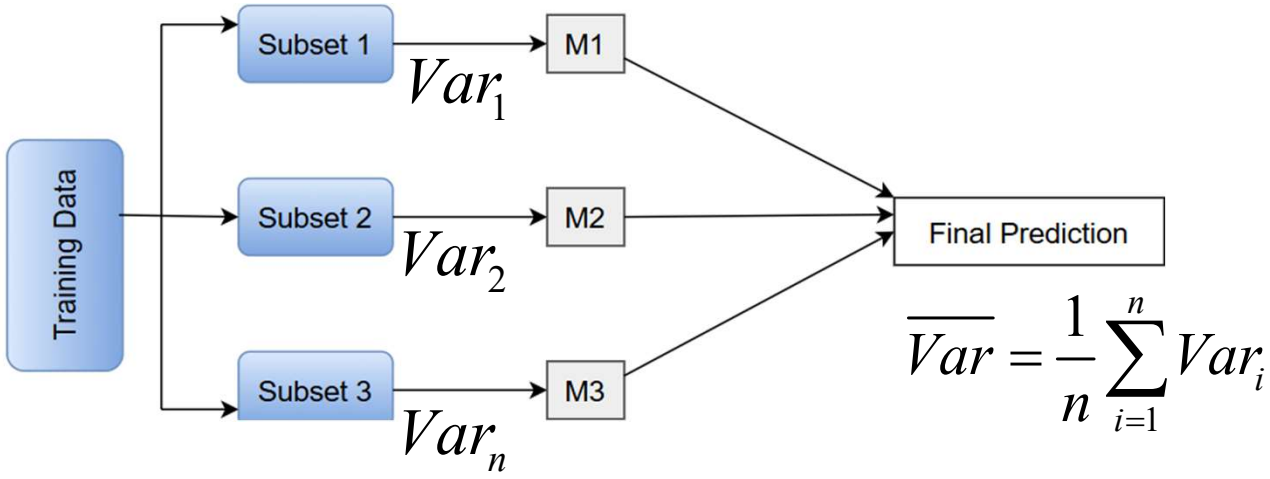
HYBRID

یادگیری جمعی (Ensemble Learning)

روش موازی (کیسه‌ای)

BAGGING

Bootstrap aggregation



در این روش از داده‌ها به صورت تصادفی چندین گروه تشکیل می‌دهیم(bootstrap) و مدل پایه ساده یکسانی برای آنها بکار گرفته می‌شود و در فرایند تصمیم گیری، نظر مدل‌ها باهم ترکیب می شود(aggregating).
ترکیب معمولا با استفاده از میانه یا میانگین داده‌ها انجام می‌شود.

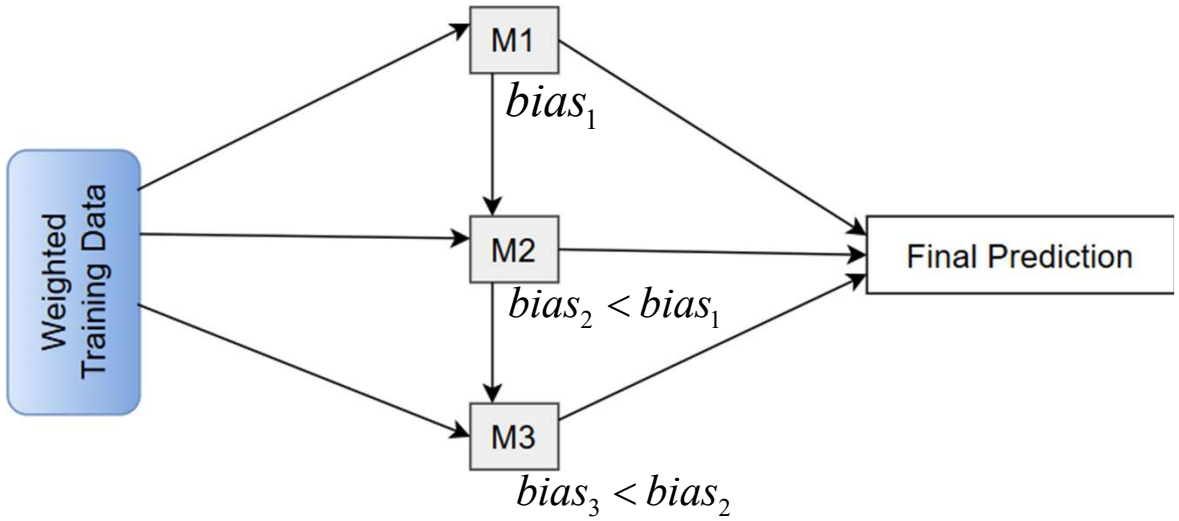
این روش برای **حداقل کردن واریانس** مدل استفاده می شود.
تعداد زیاد مدلهای ساده استفاده شده تفسیرپذیری مدل را کم می کند.
جنگل تصادفی یادگیری جمعی به روش کیسه‌ای است.

یادگیری جمعی (Ensemble Learning)

روش تقویت

BOOSTING

Iterative Learning



در این روش در هر مرحله مدل جدید براساس اشتباهات مدل قبلی ساخته می‌شود.

وزن‌هایی به هر نمونه براساس اشتباه‌های مدل قبلی اختصاص داده می شود (نمونه‌هایی که در مدل قبلی اشتباه پیش‌بینی شده اند وزن بیشتری دریافت می کنند تا مدل جدید اشتباه‌های مدل قبلی را اصلاح کند.) و این فرایند چندین بار تکرار می‌شود تا به دقت قابل قبول برسیم.

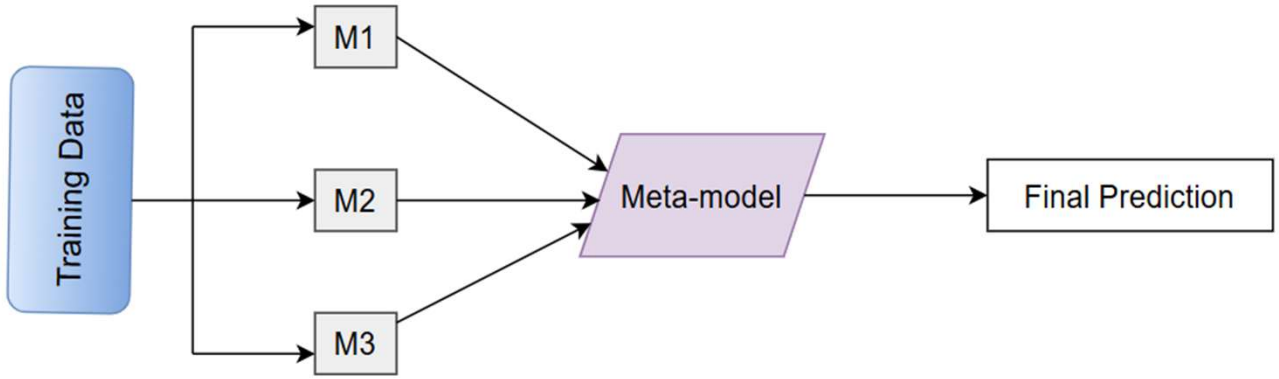
این روش برای **حداقل کردن بایاس** استفاده می شود.

XGBoost, Gradient boosting , AdaBoost مثال‌هایی از روش تقویت هستند.

یادگیری جمعی (Ensemble Learning)

روش پشته‌ای

STACKING

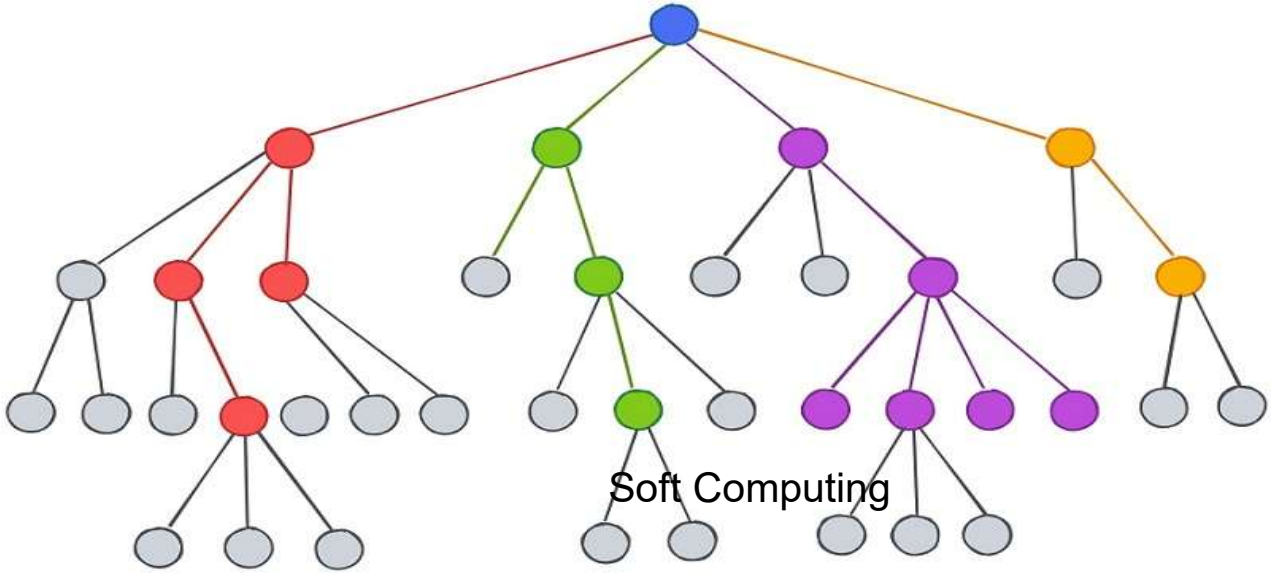


در این روش یک مدل متا روی پیش‌بینی‌های مدل‌های پایه ساخته می‌شود. مدل متا وظیفه یادگیری رابطه میان پیش‌بینی‌های مدل‌های پایه و خروجی واقعی را دارد. مدل متا با مدل پایه یکسان درنظر گرفته نمی‌شود.

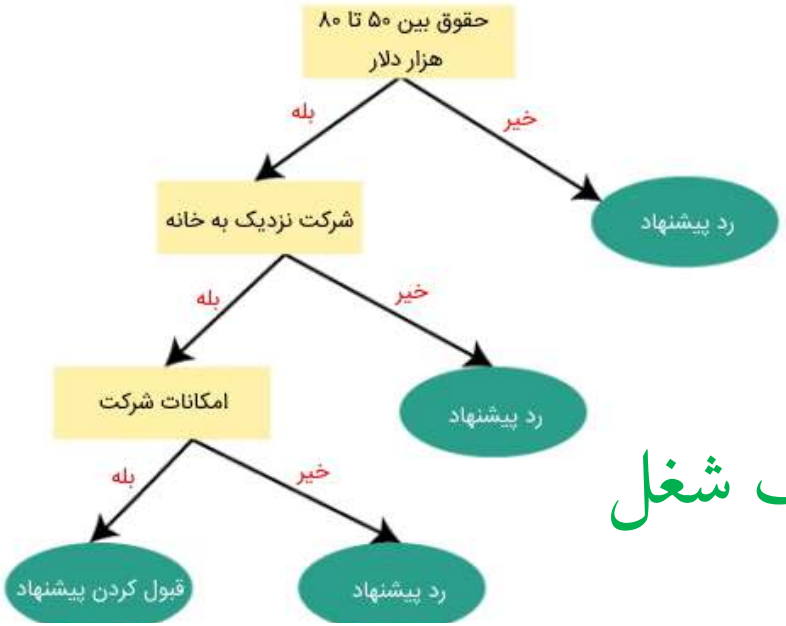
این روش برای **حداقل کردن واریانس و بایاس** استفاده می شود.
این مدل زمان زیادتری جهت آموزش برده و پیچیدگی را افزایش می‌دهد.

ترکیب چند روش یادگیری جمعی را نیز می‌توان بکار گرفت که **مدلهای ترکیبی(هیبریدی)** نام دارند

درخت تصمیم (Decision Tree)



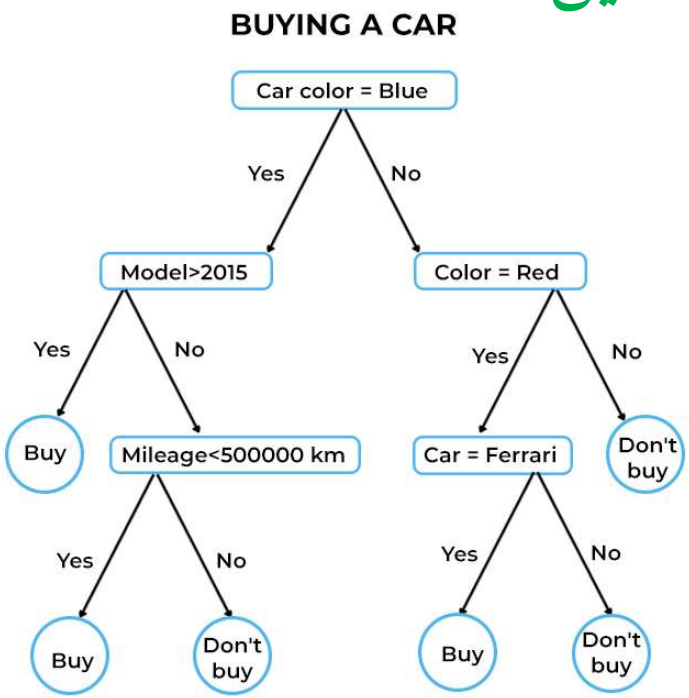
درخت تصمیم



شما هر روزه از درخت تصمیم برای تصمیم‌گیری‌ها و انتخاب‌های خود استفاده می‌کنید.

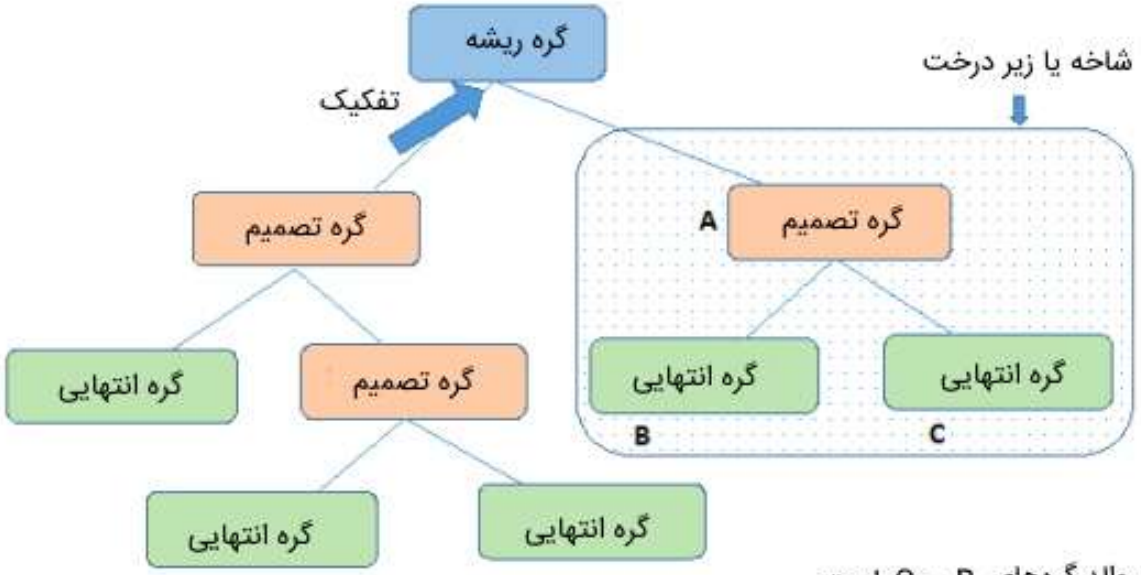
مثال انتخاب شغل

مثال خرید ماشین



درخت تصمیم یک مدل ساده است که در یادگیری جمعی استفاده می‌شود

اجزای درخت تصمیم



اجزای درخت تصمیم

«گره ریشه» (Root Node) گره نشان‌دهنده تمام مجموعه داده مسئله که به دو یا چند مجموعه «همگن» تقسیم شود.

«تفکیک» (Splitting) پردازشی برای تقسیم کردن گره‌ها به دو یا چند «زیر گره» (Sub-Node)دیگر گفته می‌شود.

«گره تصمیم» (Decision Node) زیر گره‌ای که به چند زیر گره دیگر تقسیم می‌شود

«گره برگ یا گره انتهایی» (Leaf | Terminal Node)گره‌های انتهایی هر شاخه و غیرقابل تفکیک

«هرس کردن» (Pruning) مخالف عملیات تفکیک

«شاخه یا زیر درخت» (Branch | Sub-Tree) تمام زیربخش‌های درخت تصمیم

«گره والد و فرزند» (Parent and Child Node)گره‌ای که به زیر گره‌های دیگر تقسیم می‌شود زیر گره‌های ایجاد شده فرزند گره والد نامیده می‌شوند.

عمق درخت (Tree depth) تعداد ترازهای تفکیک است یا فاصله گره ریشه تا دورترین گره برگ

روش تفکیک در درخت تصمیم

در درخت تصمیم بر اساس بهترین ویژگی تفکیک انجام می‌شود

دو رویکرد بهره اطلاعاتی و شاخص جینی در سنجش انتخاب بهترین ویژگی معمولا استفاده می‌شوند

بهره اطلاعاتی (Information Gain):

آنتروپی در درخت تصمیم نشانگر میزان واریانس است

$$E = -\sum_{i}^{C} p_i \log_2 p_i$$

p_i احتمال برداشتن عضو i کلاس C

مثال: در مجموعه ●●●●●● مقدار آنتروپی برابر است با:

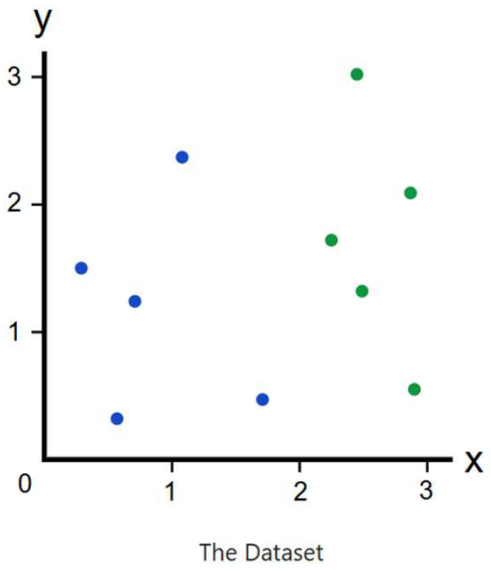
$$E = -(\frac{1}{6}\log_2(\frac{1}{6}) + \frac{2}{6}\log_2(\frac{2}{6}) + \frac{3}{6}\log_2(\frac{3}{6})) = 1.46$$

اگر تمام اجزای مجموعه یک رنگ بودندمقدار آنتروپی برابر بود با:

$$E = -(1\log_2 1) = 0$$

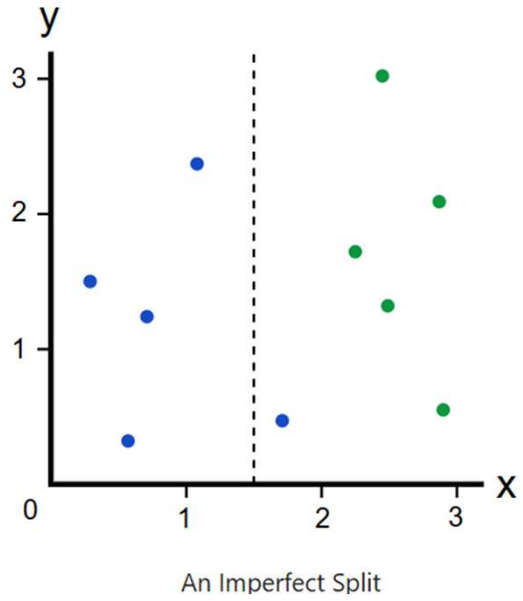
روش تفکیک در درخت تصمیم

بهره اطلاعاتی- آنتروپی



The Dataset

$$E = -(0.5\log_2 0.5 + 0.5\log_2 0.5) = 1$$



An Imperfect Split

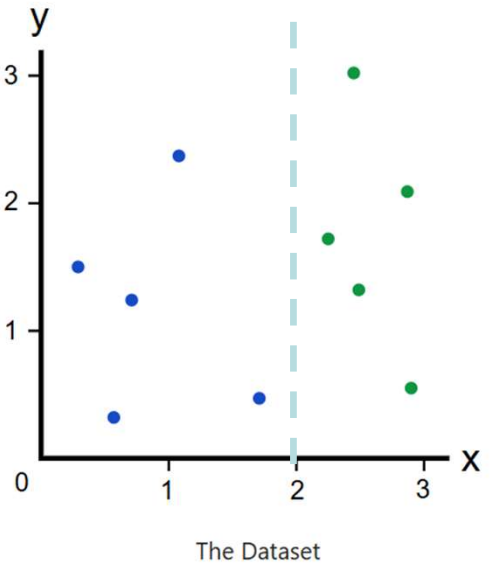
$$E_l = -(1\log_2 1) = 0$$

$$E_r = -(1/6\log_2(1/6) + 5/6\log_2(5/6)) = 0.65$$

$$E_{split} = 0.4*0 + 0.6*0.65 = 0.39$$

نسبت اعضا در کلاس به کل اعضا

$$Gain = 1 - 0.39 = 0.61$$



The Dataset

$$E_l = E_r = -(1\log_2 1) = 0$$

$$E_{split} = 0.4*0 + 0.6*0 = 0$$ **Minimum**

$$Gain = 1 - 0 = 1$$ **Maximum**

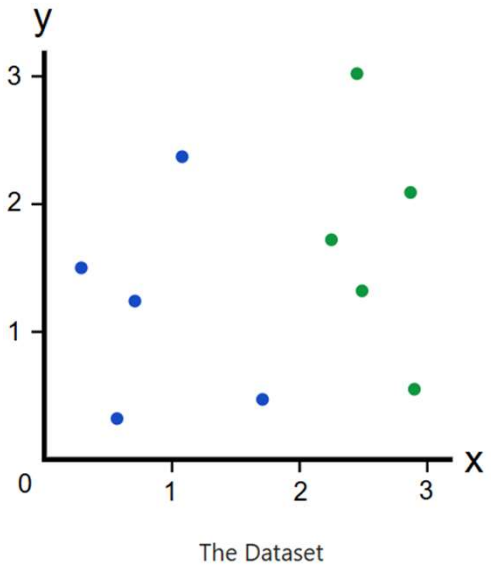
higher Information Gain = more Entropy removed

اجزای درخت تصمیم

$$G = \sum_{i}^{C} p_i * (1 - p_i)$$

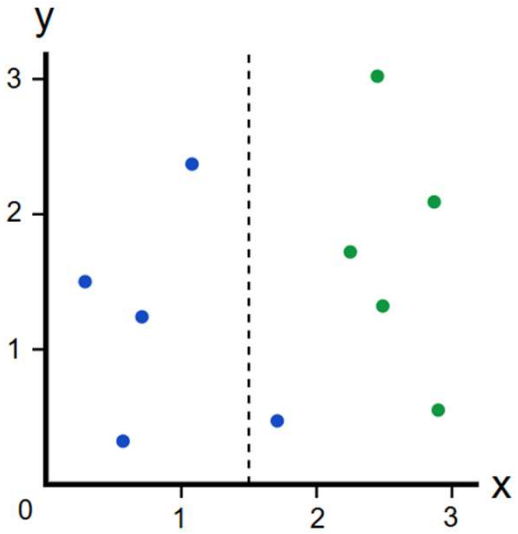
شاخص جینی (Gini Index)

شاخص جینی بین صفر و نیم است



The Dataset

$$G = 0.5 * (1 - 0.5) + 0.5 * (1 - 0.5) = 0.5$$



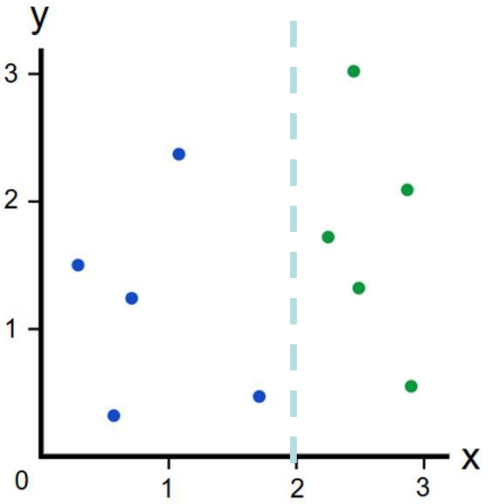
An Imperfect Split

$$G_l = 1 * (1 - 1) = 0$$

$$G_r = 1/6 * (1 - 1/6) + 5/6 * (1 - 5/6) = 0.278$$

$$G_{split} = 0.4 * 0 + 0.6 * 0.278 = 0.167$$

$$Gini\ Gain = 0.5 - 0.167 = 0.333$$



The Dataset

$$G_l = 1 * (1 - 1) + 0 * (1 - 0) = 0$$

$$G_r = 1 * (1 - 1) + 0 * (1 - 0) = 0$$

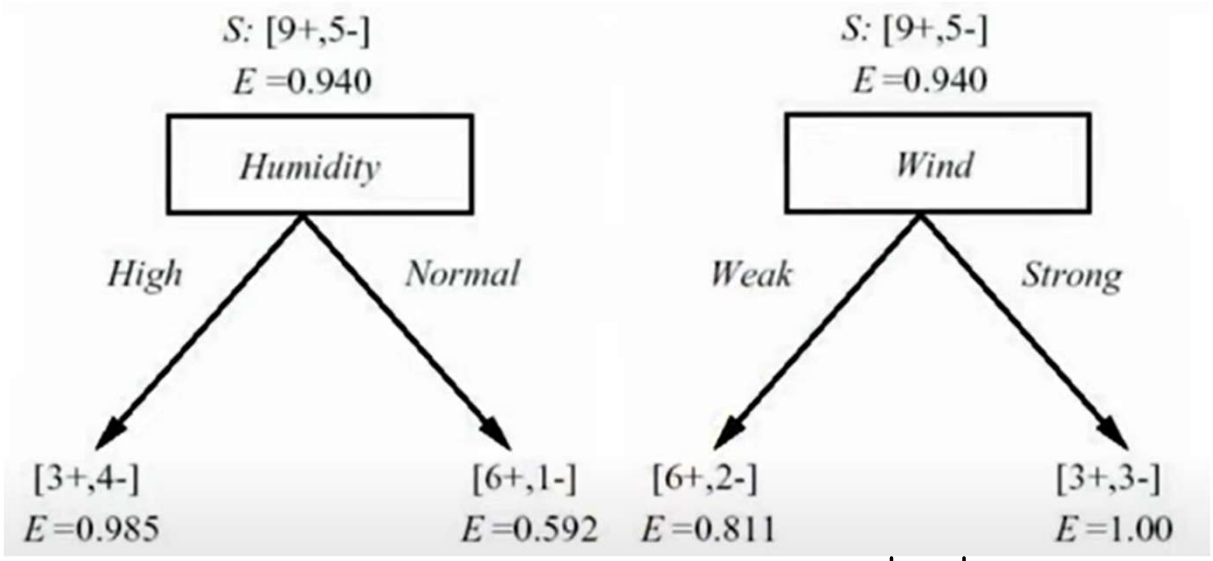
$$G_{split} = 0.4 * 0 + 0.6 * 0 = 0$$

$$Gini\ Gain = 0.5 - 0 = 0.5$$

Higher Gini Gain = Better Split

درخت تصمیم

بهره اطلاعاتی در درخت تصمیم برای متغیر تقسیم کننده A و متغیر هدف Y در مجموعه نمونه S و زیرمجموعه‌های S_i



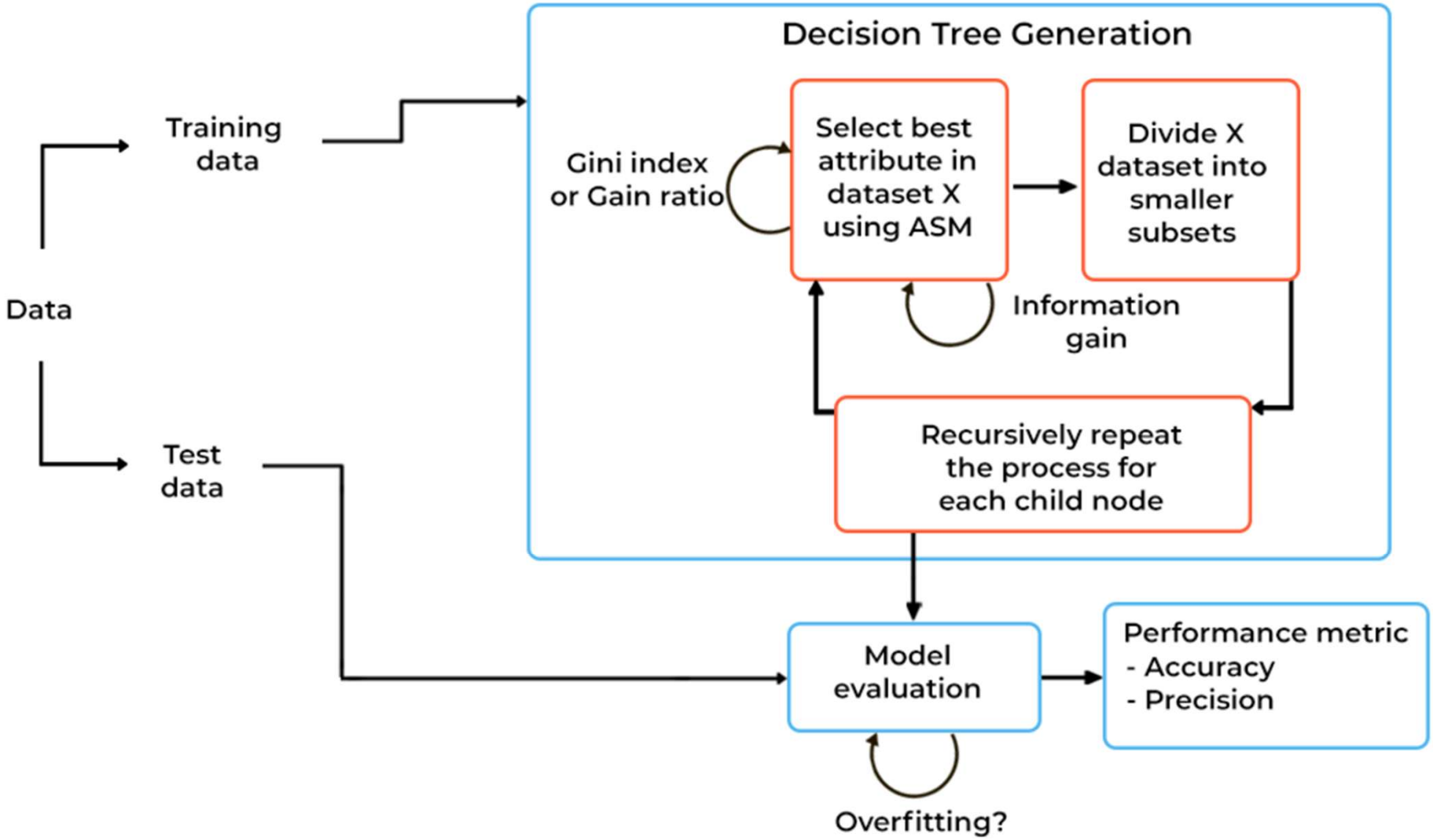
آنتروپی Y روی S $E_S(Y)$

$$Gain(S,A) = E_S(Y) - \sum_{i \in Values(A)} \frac{|S_i|}{|S|} E_{S_i}(Y)$$

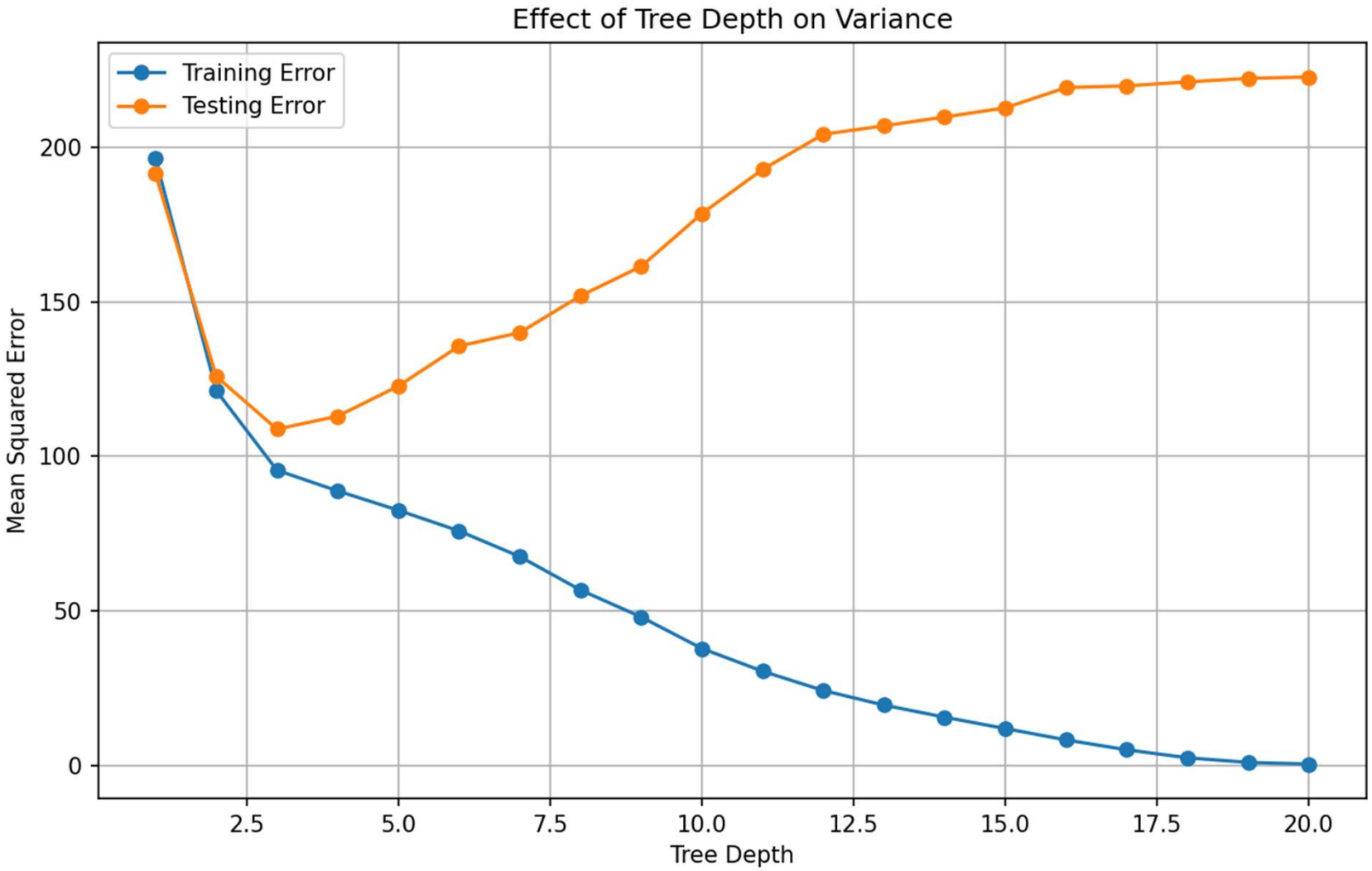
$$Gain(S, Humidity) = 0.94 - 7/14 * 0.985 - 7/14 * 0.592 = 0.151$$

$$Gain(S, Wind) = 0.94 - 8/14 * 0.811 - 6/14 * 1 = 0.48$$

درخت تصمیم


Decision Tree Generation
Training data
Data
Test data
Gini index or Gain ratio
Select best attribute in dataset X using ASM
Divide X dataset into smaller subsets
Information gain
Recursively repeat the process for each child node
Model evaluation
Performance metric
- Accuracy
- Precision
Overfitting?

درخت تصمیم



در درخت تصمیم عمق درخت، میزان واریانس را مشخص می کند. با استفاده از روش «هرس کردن» (Pruning)، مقدار واریانس کنترل می‌شود.

مثال درخت تصمیم

درخت تصمیم فایل گلهای زنبق

نام فایل: 10 DT.py

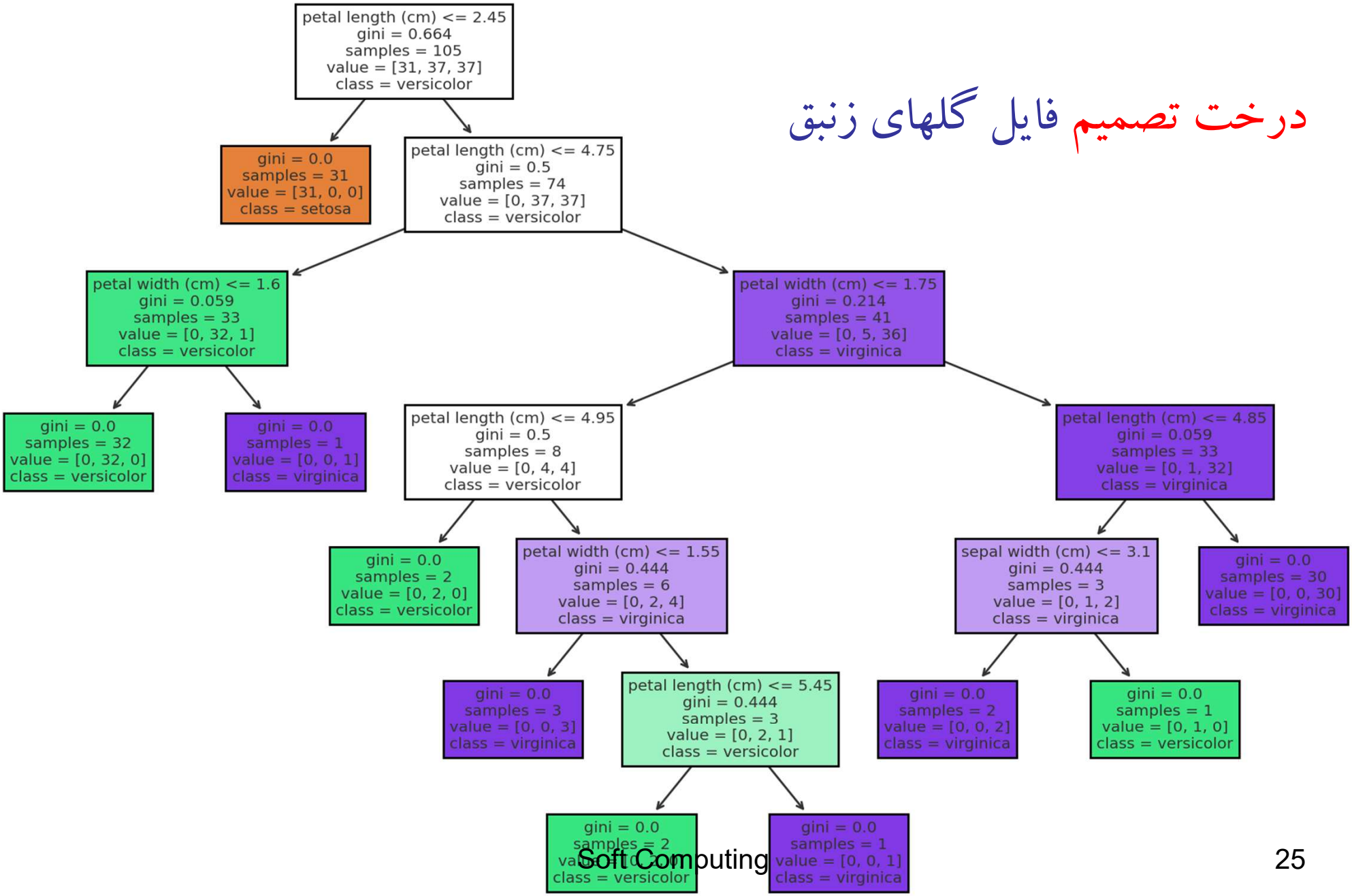
فراخوانی داده‌های فایل گلهای زنبق

```
# Import necessary libraries
from sklearn.datasets import load_iris
from sklearn.tree import DecisionTreeClassifier, plot_tree
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
import matplotlib.pyplot as plt

# Load the Iris dataset
data = load_iris()
X = data.data
y = data.target

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.3, random_state=42)
```

مثال درخت تصمیم

درخت تصمیم فایل گلهای زنبق

```
# Initialize and train the Decision Tree Classifier
clf = DecisionTreeClassifier(random_state=42)
clf.fit(X_train, y_train)

# Predict on the test set
y_pred = clf.predict(X_test)

# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred)

# Print the accuracy
print(f"Accuracy of the Decision Tree Classifier: {accuracy:.2f}")
# Plot the decision tree
plt.figure(figsize=(12, 8))
plot_tree(clf, filled=True, feature_names=data.feature_names,
class_names=data.target_names)
plt.title(f"Decision Tree Visualization\nAccuracy: {accuracy:.2f}")
plt.show()
```



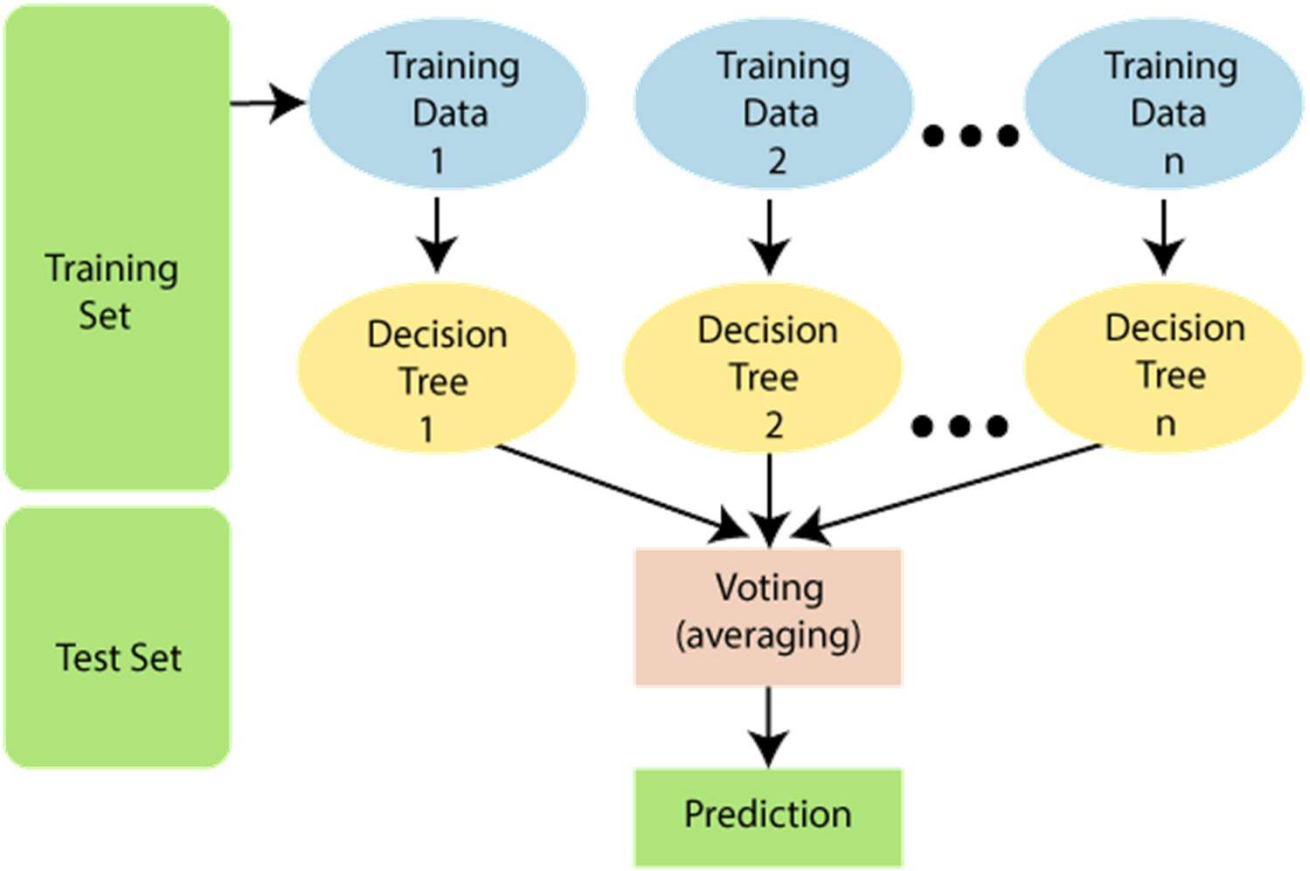
درخت تصمیم فایل گلهای زنبق

جنگل تصادفی(Random Forest)

یادگیری جمعی

جنگل تصادفی

در جنگل تصادفی برای تعدادی از ویژگیهای داده‌ها درخت تصمیم را اجرا می کنیم و سپس با رای گیری جواب بدست می آید.
یعنی مدلهای ساده در جنگل تصادفی همان درخت تصمیم هستند.



جنگل تصادفی

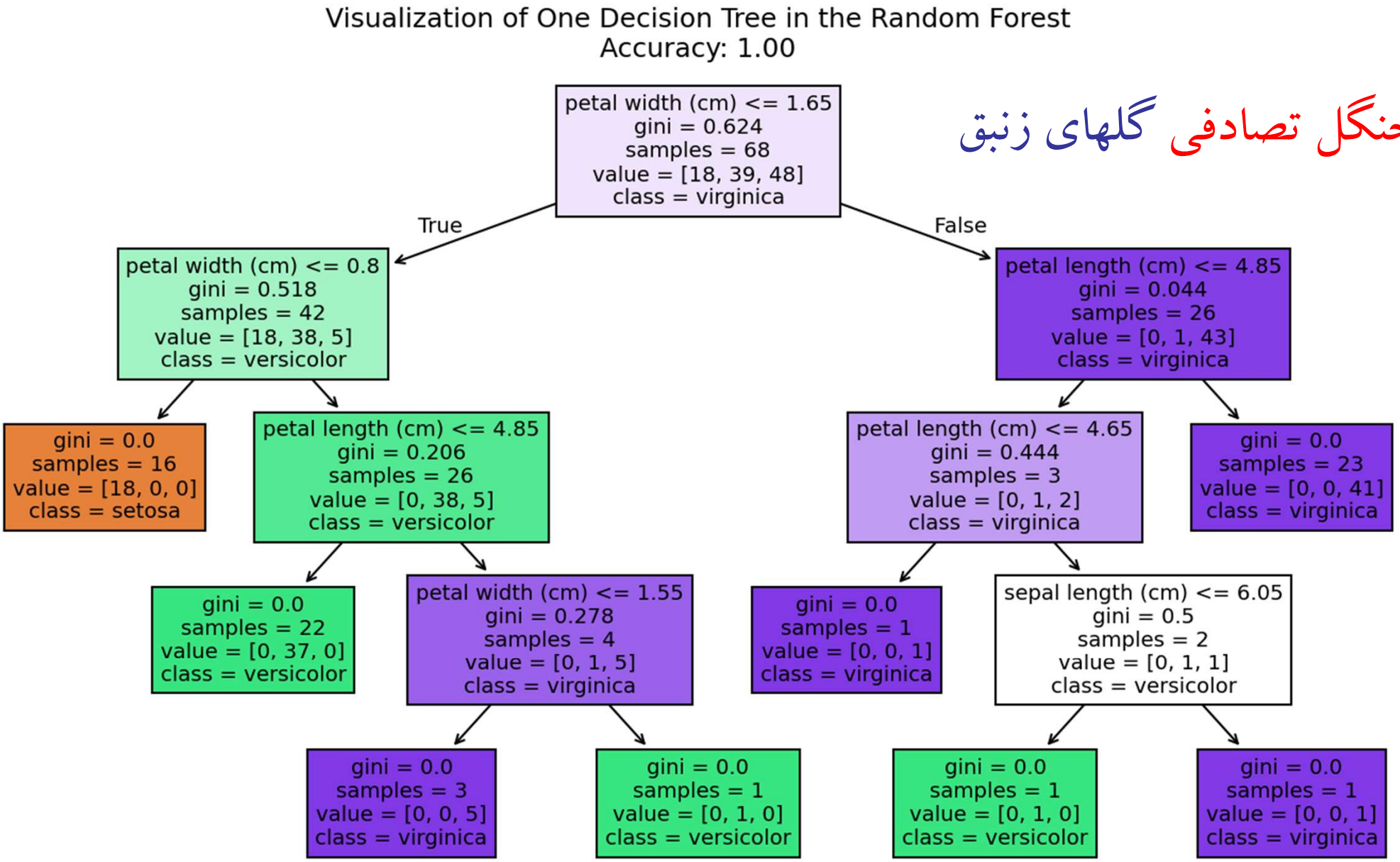
جنگل تصادفی فایل گلهای زنبق

نام فایل: 10 RF.py

```
# Initialize and train the Random Forest Classifier
rf_clf = RandomForestClassifier(n_estimators=10, random_state=42)
rf_clf.fit(X_train, y_train)

# Predict on the test set
y_pred = rf_clf.predict(X_test)

# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred)

# Print the accuracy
print(f"Accuracy of the Random Forest Classifier: {accuracy:.2f}")

# Visualize one of the decision trees from the Random Forest
plt.figure(figsize=(12, 8))
plot_tree(
    rf_clf.estimators_[0],  # Select the first decision tree in the
forest
    filled=True,
    feature_names=data.feature_names,
    class_names=data.target_names,
)
plt.title(f"Visualization of One Decision Tree in the Random
Forest\nAccuracy: {accuracy:.2f}")
plt.show()
```

تعداد درخت تصمیم

جنگل تصادفی



جنگل تصادفی گلهای زنبق

Adaboost – یادگیری جمعی

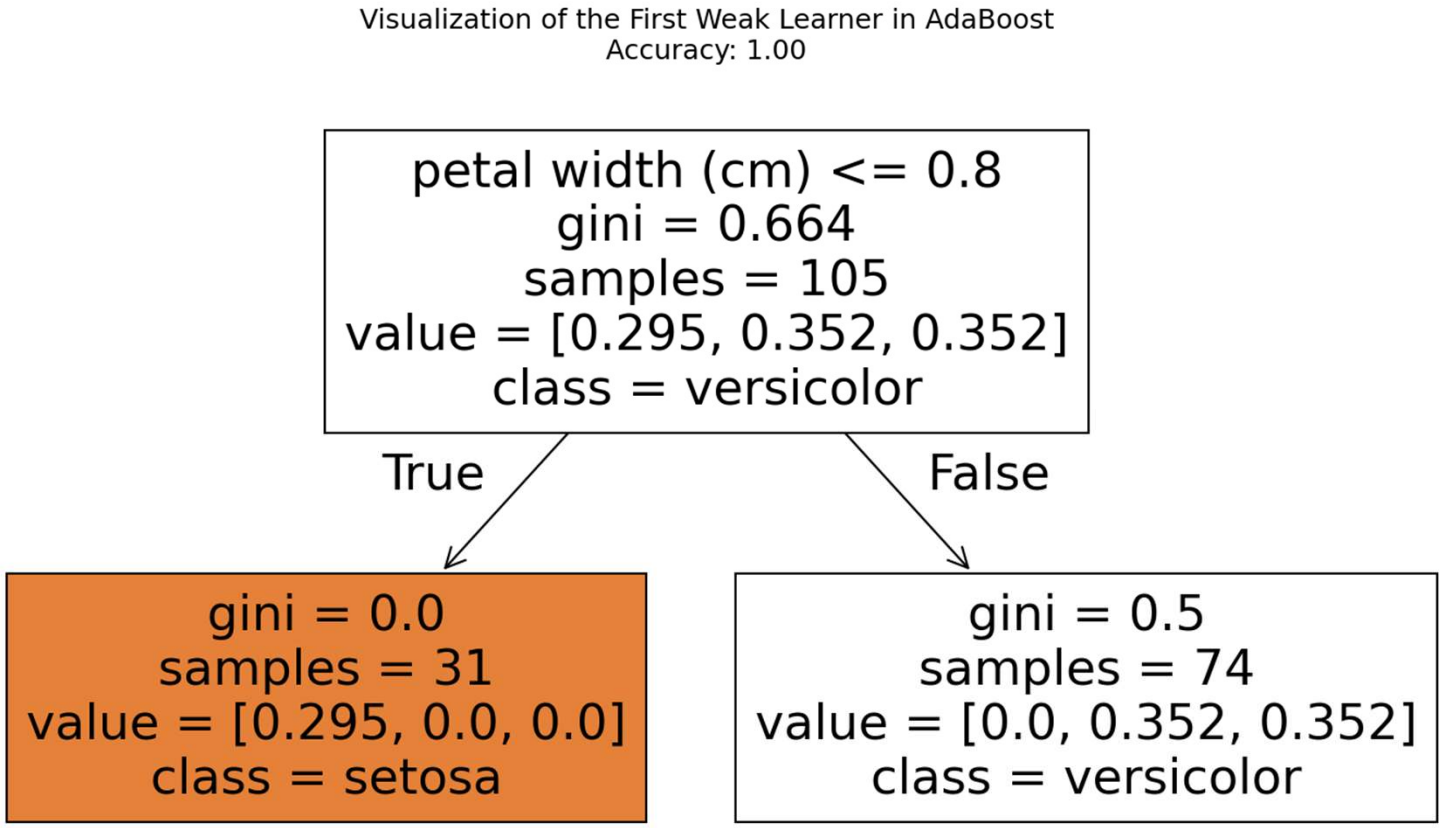
یادگیری جمعی – روش تقویت – گلهای زنبق

نام فایل: 10 adaboost.py

```
# Initialize the base estimator (weak learner)
base_estimator = DecisionTreeClassifier(max_depth=1, random_state=42)

# Initialize and train the AdaBoost Classifier
adaboost_clf = AdaBoostClassifier(estimator=base_estimator,
n_estimators=50, random_state=42)
adaboost_clf.fit(X_train, y_train)

# Predict on the test set
y_pred = adaboost_clf.predict(X_test)

# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred)

# Print the accuracy
print(f"Accuracy of the AdaBoost Classifier: {accuracy:.2f}")

# Visualize the first weak learner (decision tree)
plt.figure(figsize=(12, 8))
plot_tree(
    adaboost_clf.estimators_[0],  # Select the first weak learner
    filled=True,
    feature_names=data.feature_names,
    class_names=data.target_names,
)
plt.title(f"Visualization of the First Weak Learner in
AdaBoost\nAccuracy: {accuracy:.2f}")
plt.show()
```

Adaboost – یادگیری جمعی

یادگیری جمعی – روش تقویت -گلهای زنبق



یادگیری جمعی- مقایسه روشها

یادگیری جمعی – مقایسه روشها-گلهای زنبق

نام فایل: 10 EL comparison.py

```
# Import necessary libraries
from sklearn.datasets import load_iris
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier,
AdaBoostClassifier, GradientBoostingClassifier,
StackingClassifier
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LogisticRegression
from xgboost import XGBClassifier

# Load the Iris dataset
data = load_iris()
X = data.data
y = data.target
# Define base classifiers for stacking
base_classifiers = [
    ('decision_tree', DecisionTreeClassifier(random_state=42)),
    ('random_forest', RandomForestClassifier(n_estimators=100,
random_state=42)),
    ('adaboost', AdaBoostClassifier(n_estimators=50,
random_state=42, algorithm='SAMME')),
    ('gradient_boosting',
GradientBoostingClassifier(n_estimators=100, random_state=42)),
    ('xgboost', XGBClassifier(eval_metric='mlogloss'))
]
```

یادگیری جمعی- مقایسه روشها

یادگیری جمعی – مقایسه روشها-گلهای زنبق

```
# Define the meta-model for stacking
meta_model = LogisticRegression(random_state=42)

# Create the stacking classifier
stacking_clf = StackingClassifier(estimators=base_classifiers,
final_estimator=meta_model)

# Define all classifiers
classifiers = {
    "Decision Tree": DecisionTreeClassifier(random_state=42),
    "Random Forest": RandomForestClassifier(n_estimators=50,
random_state=42, n_jobs=-1),
    "AdaBoost": AdaBoostClassifier(n_estimators=30, random_state=42,
algorithm='SAMME'),
    "Gradient Boosting": GradientBoostingClassifier(n_estimators=50,
random_state=42),
    "XGBoost": XGBClassifier(n_estimators=50, eval_metric='mlogloss',
n_jobs=-1),  # Removed use_label_encoder
    "Stacking": stacking_clf  # Add stacking classifier
}

# Evaluate each classifier using cross-validation and measure time
print("Performance of Ensemble Methods:")
for name, clf in classifiers.items():
    start_time = time.time()
    scores = cross_val_score(clf, X, y, cv=3, scoring='accuracy')  #
Reduced to 3-fold CV
    elapsed_time = time.time() - start_time
    print(f"{name} Accuracy: {scores.mean():.2f} ± {scores.std():.2f},
Time: {elapsed_time:.2f} seconds")
```

یادگیری جمعی- مقایسه روشها

یادگیری جمعی – مقایسه روشها-گلهای زنبق

```
Performance of Ensemble Methods:
Decision Tree Accuracy: 0.96 ± 0.02, Time: 0.00 seconds
Random Forest Accuracy: 0.97 ± 0.02, Time: 0.28 seconds
AdaBoost Accuracy: 0.95 ± 0.02, Time: 0.12 seconds
Gradient Boosting Accuracy: 0.97 ± 0.02, Time: 0.35 seconds
XGBoost Accuracy: 0.95 ± 0.02, Time: 0.10 seconds
Stacking Accuracy: 0.96 ± 0.02, Time: 4.52 seconds
```

تمرین برنامه نویسی

تمرین دهم : یک برنامه به زبان پایتون بنویسید که یک فایل داده را خوانده و بر حسب ویژگیها داده ها را طبقه بندی نماید.

۱- فایل داده را بخوانید

۲- داده ها را به روش درخت تصمیم طبقه بندی نمایید

۳- داده ها را به روش جنگل تصادفی طبقه بندی نمایید

۳- سه داده با ویژگی جدید را تعیین کنید در کدام کلاس داده هستند.

Criteria	Bagging	Boosting	Stacking
Approach	Parallel training of weak models	Sequential training of weak models	Aggregates the predictions of multiple models into a meta-model
Base Models	Homogenous	Homogenous	Can be heterogenous
Subset Selection	Random sampling with replacement	Subsets are not required	Subsets are not required
Goal	Reduce variance	Reduce bias	Reduce variance and bias
Model Combination	Majority voting or averaging	Weighted majority voting or averaging	Using an ML model