Soft computing

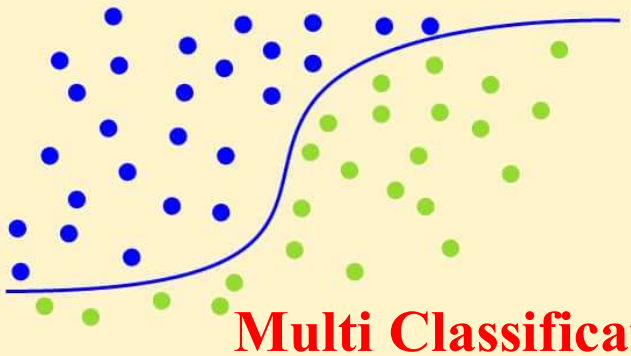
K.N. Toosi University of Technology

Soft Computing 1

Artificiel Intelligence and Soft computing

K.N.TOOSI University of Technology

طبقه بندی چند کلاسی



Multi Classification

Soft Computing 2

نمونه رگرسیون لجستیک با بیش از یک ویژگی

Age	Heart Rate	Heart Attack
45	50	N
50	50	N
55	50	Y
60	50	N
65	70	N
70	70	Y
75	90	Y
80	90	Y
85	90	N
90	90	Y
95	90	Y

مثال : احتمال حمله قلبی با دو ویژگی ضربان قلب و سن فرد
احتمال حمله قلبی شخصی ۲۰ ساله با ضربان قلب ۸۰ چقدر است؟

بسیاری از عوامل دیگر مانند فشار خون، سطح کلسترول، سابقه خانوادگی، و عادات زندگی (مثل سیگار کشیدن و ورزش نکردن) نیز در تعیین خطر حمله قلبی مؤثرند، که در این جدول لحاظ نشده است.

$$P(Y=1|X=x)=p(x)=\frac{1}{1+e^{-(\theta_0+\theta_1 x_1+\theta_2 x_2+\ldots)}}$$

$$P(Heart\ Attack=1|X=Age, Heart\ rate)=\frac{1}{1+e^{-(\theta_0+\theta_1.Age+\theta_2.Heart\ rate)}}$$

Soft Computing 3

نمونه رگرسیون لجستیک با بیش از یک ویژگی

Age	Heart Rate	Heart Attack
45	50	N
50	50	N
55	50	Y
60	50	N
65	70	N
70	70	Y
75	90	Y
80	90	Y
85	90	N
90	90	Y
95	90	Y

مثال : احتمال حمله قلبی با دو ویژگی ضربان قلب و سن فرد

بسیاری از عوامل دیگر مانند فشار خون، سطح کلسترول، سابقه خانوادگی، و عادات زندگی (مثل سیگار کشیدن و ورزش نکردن) نیز در تعیین خطر حمله قلبی مؤثرند، که در این جدول لحاظ نشده است.

$$P(Y=1|X=x)=p(x)=\frac{1}{1+e^{-(\theta_0+\theta_1 x_1+\theta_2 x_2+\ldots)}}$$

$$P(Heart\ Attack=1|X=Age, Heart\ rate)=\frac{1}{1+e^{-(\theta_0+\theta_1.Age+\theta_2.Heart\ rate)}}$$

Soft Computing 4

نمونه رگرسیون لجستیک با بیش از یک ویژگی

فایل: 8 regression logistic two feature.py

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split

# Define the dataset
data = {
    'Age': [45, 50, 55, 60, 65, 70, 75, 80, 85, 90, 95],
    'Heart Rate': [50, 50, 50, 50, 70, 70, 90, 90, 90, 90, 90],
    'Heart Attack': [0, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1]
}
df = pd.DataFrame(data)

# Prepare the data
X = df[['Age', 'Heart Rate']]
y = df['Heart Attack']
```


نمونه رگرسیون لجستیک با بیش از یک ویژگی

```
# Train the logistic regression model
model = LogisticRegression()
model.fit(X, y)

# model's parameters
print("Coefficients:", model.coef_)
print("Intercept:", model.intercept_)
theta_0 = model.intercept_[0]
theta_1 = model.coef_[0][0]
theta_2 = model.coef_[0][1]
print("Intercept (θ0):", theta_0)
print("Coefficient for Age (θ1):", theta_1)
print("Coefficient for Heart Rate (θ2):", theta_2)
```

```
Coefficients: [[0.07723627 0.00274301]]
Intercept: [-5.3543177]
Intercept (θ0): -5.354317698549416
Coefficient for Age (θ1): 0.07723627391026841
Coefficient for Heart Rate (θ2): 0.002743013264610142
```

نمونه رگرسیون لجستیک با بیش از یک ویژگی

```
# Example prediction for a person with Age = 20 and Heart Rate = 80
example_data = np.array([[20, 80]])
probability_heart_attack = model.predict_proba(example_data)[0][1]
print("Probability of Heart Attack for Age=20 and Heart Rate=80:",
probability_heart_attack)
```

```
Probability of Heart Attack for Age=20 and Heart Rate=80: 0.02685
```

```
Intercept (00): -5.354317698549416
Coefficient for Age (01): 0.07723627391026841
Coefficient for Heart Rate (02): 0.002743013264610142
```

$$P(Heart\ Attack = 1 \mid X = Age, Heart\ rate) = \frac{1}{1 + e^{-(-5.354 + 0.077\,Age + 0.003\,Heart\ rate)}}$$

Soft Computing 7

ارزیابی الگوریتم

پس از مدلسازی و تحلیل داده ها نیاز به ارزیابی الگوریتم است

اگر الگوریتم ارزیابی نشود نمی‌توان آنرا بهبود بخشید.

Soft Computing 8

ارزیابی الگوریتم

پس از مدلسازی و تحلیل داده های دودویی نتایج را به صورت زیر می توان دسته بندی نمود:

مثبت صحیح (TP: True Positive):
تعداد حالت‌هایی که مقدار پیش‌بینی شده با مقدار واقعی برابر بوده و متغیر پاسخ برابر با یک است .
(شناسایی درست - **معیار مثبت**).

مثبت کاذب (FP: False Positive):
تعداد حالت‌هایی که مقدار پیش‌بینی شده برای متغیر پاسخ یک و مقدار واقعی برابر صفر است.
(شناسایی غلط - **معیار منفی**).

منفی صحیح (TN: True Negative) :
تعداد حالت‌هایی که مقدار پیش‌بینی شده با مقدار واقعی برابر بوده و متغیر پاسخ برابر با صفر است.
(مردودی درست - **معیار مثبت**).

منفی کاذب (FN: False Negative) :
تعداد حالت‌هایی که مقدار پیش‌بینی شده برای متغیر پاسخ صفر و مقدار واقعی برابر یک است.
(مردودی غلط - **معیار منفی**).

9

Soft Computing

ارزیابی الگوریتم

مثال: الگوریتمی بیمارهای دارای سرطان را شناسایی می کند و نتایج زیر را داده است.

Individual Number	1	2	3	4	5	6	7	8	9	10	11	12
Actual Classification	1	1	1	1	1	1	1	1	0	0	0	0
Predicted Classification	0	0	1	1	1	1	1	1	1	0	0	0
Result	FN	FN	TP	TP	TP	TP	TP	TP	FP	TN	TN	TN

مثبت صحیح (TP: True Positive) مثبت کاذب (FP: False Positive)
منفی صحیح (TN: True Negative) منفی کاذب (FN: False Negative)

چگونه از این شاخص ها استفاده کنیم؟

10

Soft Computing

ارزیابی الگوریتم

ماتریس اختلاط یا ماتریس درهم ریختگی (Confusion Matrix or Error Matrix)

به منظور سنجش مدل از یک ماتریس دو در دو به نام ماتریس درهم ریختگی استفاده می شود

مثبت صحیح (TP) مثبت کاذب (FP)
منفی صحیح (TN) منفی کاذب (FN)

		Predicted condition	
	Total population = P + N	Positive (PP) $PP = TP + FP$	Negative (PN) $PN = FN + TN$
Actual condition	Positive (P) $P = TP + FN$	True positive (TP)	False negative (FN)
	Negative (N) $N = FP + TN$	False positive (FP)	True negative (TN)

در واقعیت صحیح (P)

در واقعیت منفی (N)

ماتریس درهم ریختگی

ارزیابی الگوریتم

ماتریس درهم ریختگی برای مثال بیماران سرطانی

Individual Number	1	2	3	4	5	6	7	8	9	10	11	12
Actual Classification	1	1	1	1	1	1	1	1	0	0	0	0
Predicted Classification	0	0	1	1	1	1	1	1	1	0	0	0
Result	FN	FN	TP	TP	TP	TP	TP	TP	FP	TN	TN	TN

		Predicted condition	
	Total population = P + N	Positive (PP)	Negative (PN)
Actual condition	Positive (P)	True positive (TP)	False negative (FN)
	Negative (N)	False positive (FP)	True negative (TN)

		Predicted condition	
	Total 8 + 4 = 12	Cancer 7	Non-cancer 5
Actual condition	Cancer 8	6	2
	Non-cancer 4	1	3

ارزیابی الگوریتم

معیارها در ماتریس اختلاط یا ماتریس درهم ریختگی

به منظور اندازه‌گیری کارایی مدل از معیارهای زیر برای حالت مثبت (مقادیر یک) استفاده می شود

صحت یا حساسیت (Sensitivity, Recall)

Hit rate, True Positive Rate (TPR)

$$\text{TPR (Recall)} = \frac{TP}{P} = \frac{TP}{TP+FN}$$

صحت برابرست با تعداد مثبت صحیح به تعداد صحیح در واقعیت

این معیار نسبت پیش بینی مثبت درست مدل به کل مقادیر مثبت درست واقعی را نشان می‌دهد

دقت (Precision)

Positive Predictive Value (PPV)

$$\text{PPV (Precision)} = \frac{TP}{PP} = \frac{TP}{TP+FP}$$

دقت برابرست با تعداد مثبت صحیح به تعداد صحیح در مدل(پیش بینی)

این معیار نسبت پیش بینی مثبت درست مدل به کل مقادیر مثبت درست مدل را نشان می‌دهد

ارزیابی الگوریتم

معیارها در ماتریس اختلاط یا ماتریس درهم ریختگی

مشابه حالت قبل از معیارهای زیر برای حالت منفی (مقدار صفر) استفاده می شود

Specificity

True Negative Rate(TNR) ,Selectivity

$$\text{TNR} = \frac{TN}{N} = \frac{TN}{TN+FP}$$

برابرست با تعداد منفی صحیح به تعداد منفی در واقعیت
این معیار در برابر حساسیت است

Negative Predictive Value (NPV)

$$\text{NPV} = \frac{TN}{TN+FN}$$

ارزیابی الگوریتم

معیارها در ماتریس اختلاط یا ماتریس درهم ریختگی

به منظور اندازه‌گیری کارایی مدل از معیارهای زیر نیز استفاده می شود

تکیه‌گاه (Support)

$$\text{Support} = P$$

تکیه‌گاه تعداد مشاهده مقدار ۱ (هر کلاسی از داده ها) در مشاهدات است.

اندازه F_1

میانگین همساز دو معیار دقت و صحت است.
مقدار صفر بدترین حالت و مقدار یک بهترین حالت است و تعادل دقت و صحت را دربردارد

$$F_1 = \frac{1}{\frac{1}{2\text{Precision}} + \frac{1}{2\text{Recall}}} = 2\frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

صحت و دقت (Accuracy)

$$ACC = \frac{TP+TN}{P+N} = \frac{TP+TN}{TP+TN+FP+FN}$$

صحت سنجی دو کلاسی

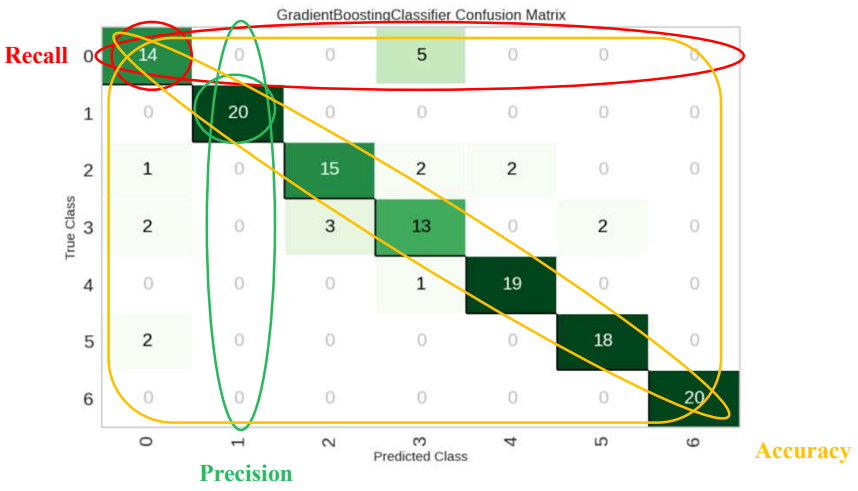
ماتریس درهم ریختگی (Confusion Matrix)

		Predicted Class: Positive	Predicted Class: Negative	
Actual Class	Positive	True Positive (TP)	False Negative (FN) Type II Error	**Sensitivity** $\frac{TP}{(TP+FN)}$
	Negative	False Positive (FP) Type I Error	True Negative (TN)	**Specificity** $\frac{TN}{(TN+FP)}$
		Precision $\frac{TP}{(TP+FP)}$	**Negative Predictive Value** $\frac{TN}{(TN+FN)}$	**Accuracy** $\frac{TP+TN}{(TP+TN+FP+FN)}$

ماتریس درهم ریختگی می تواند برای کلاسهای بیشتر نیز توسعه یابد

صحت سنجی چند کلاسی

ماتریس درهم ریختگی چندطبقه



صحت سنجی چند کلاسی

ماتریس درهم ریختگی چندطبقه

Recall is the fraction of samples in class i that the model correctly classified as class i

$$\text{Recall} = \frac{C_{ii}}{\sum_j C_{ij}}$$

C_{ii}: The count of samples correctly classified as class i (True Positives for class i).

$\sum_j C_{ij}$: The total number of samples that actually belong to class i, regardless of how they were classified (True Positives + False Negatives for class i).

Precision is the fraction of samples that the model assigned to class i that actually belong to class i.

$$\text{Precision} = \frac{C_{ii}}{\sum_j C_{ji}}$$

C_{ii}: The count of samples correctly classified as class i (True Positives for class i).

$\sum_j C_{ji}$: The total number of samples predicted as class i, regardless of their actual class (True Positives + False Positives for class i).

Accuracy is the overall fraction of correctly classified samples across all classes

$$\text{Accuracy} = \frac{\sum_i C_{ii}}{\sum_i \sum_j C_{ij}}$$

$\sum_i C_{ii}$: The sum of True Positives across all classes (correctly classified samples for each class).

$\sum_i \sum_j C_{ij}$: The total number of samples across all classes (including both correct and incorrect classifications).

18

صحت سنجی چند کلاسی

میانگین گیری بدون وزن بین چند کلاس

- **Macroaveraging:** treats each class equally by computing the performance measure (like F1 score) for each class independently and then averaging them.
- **Implication**: Macroaveraging gives equal weight to all classes, regardless of the number of samples in each class. This is useful when you want each class to have an equal impact on the overall metric, even if some classes have fewer samples.

میانگین گیری وزن دار بین چند کلاس

- **Microaveraging:** Microaveraging aggregates all true positives (TP), false positives (FP), and false negatives (FN) across all classes and then computes a single performance measure.
- **Implication:** Microaveraging takes into account the imbalance of class distributions by weighing each sample equally. It is especially useful in cases where class imbalance is significant, as it doesn't prioritize one class over another.

صحت سنجی رگرسیون لجستیک

ماتریس درهم ریختگی (Confusion Matrix)

به منظور اندازه‌گیری کارایی مدل ابتدا داده های مساله را به دو قسمت آموزش و تست تقسیم می کنیم.
در دستور زیر داده های تست ۳۰ درصد داده های مساله در نظر گرفته شده اند

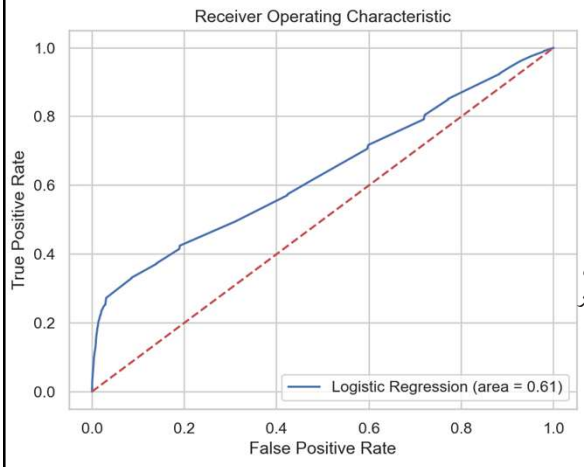
```
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

محاسبه معیارها

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

	precision	recall	f1-score	support
0	0.91	0.97	0.94	10981
1	0.52	0.25	0.34	1376

صحت سنجی رگرسیون لجستیک



منحنی ROC
(Receiver Operating Characteristic)

در نمودار، خطی که به صورت مقطع و قرمز رنگ ظاهر شده دسته بندی تصادفی مشتریان را نشان می‌دهد.
یک مدل رگرسیون لجستیک باید نرخ TP بیشتری نسبت به مدل تصادفی داشته باشد. پس خط آبی رنگ باید دور از خط قرمز رنگ قرار گیرد.

```
from sklearn Import metrics
logistic_roc_auc = metrics.roc_auc_score(y_test, model.predict(X_test))
fpr, tpr, thresholds = metrics.roc_curve(y_test, model.predict_proba(X_test)[:, 1])
```

21
Soft Computing

طبقه بندی چند کلاسی

Multiclass classification

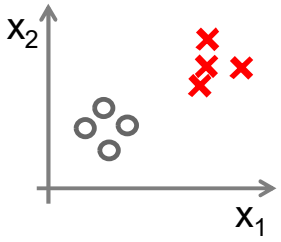
Email foldering/tagging: Work, Friends, Family, Hobby

Medical diagrams: Not ill, Cold, Flu

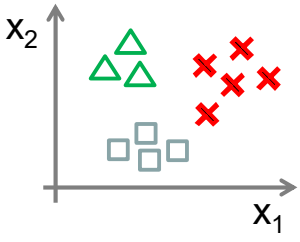
Weather: Sunny, Cloudy, Rain, Snow

0 1 2 3

Binary classification:



Multi-class classification:



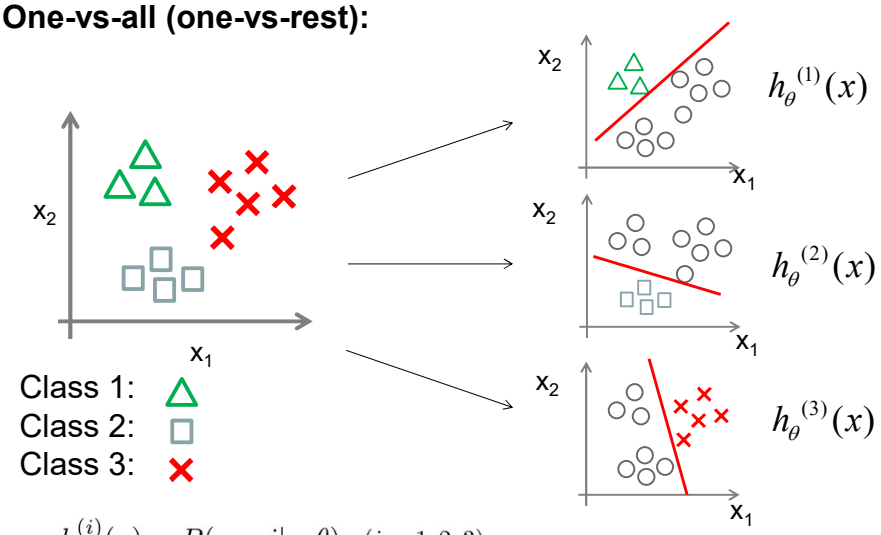
One-vs-all (one-vs-rest)
Softmax Regression

روشهای طبقه بندی چند کلاسی

22
Soft Computing

طبقه بندی چند کلاسی

One-vs-all (one-vs-rest):



$$h_\theta^{(i)}(x) = P(y = i|x;\theta) \quad (i = 1,2,3)$$

Soft Computing 23

طبقه بندی چند کلاسی

Train a logistic regression classifier $h_\theta^{(i)}(x)$ for each class i to predict the probability that $y = i$.

On a new input x, to make a prediction, pick the class i that maximizes

$$\max_i h_\theta^{(i)}(x)$$

```
# Initialize logistic regression model for multiclass
logreg = LogisticRegression(multi_class='multinomial')
logreg.fit(X_train, y_train)
```

Soft Computing 24

گلهای زنبق مختلف

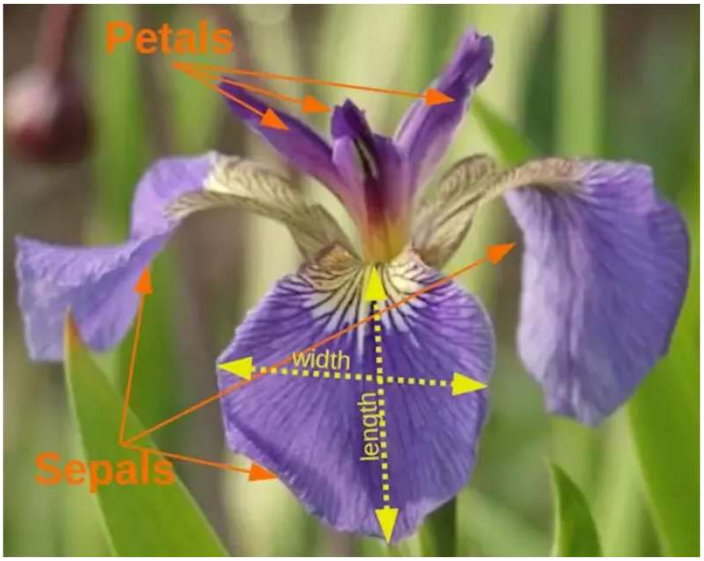
Iris Setosa

Iris Versicolour

Iris Virginica

Soft Computing

ابعاد کاسبرگ و گلبرگ زنبق مختلف



26

ابعاد کاسبرگ و گلبرگ زنبق مختلف

فایل دیتای ایریس دارای طول و عرض کاسبرگ و گلبرگ است برچسب نام گل است که دارای سه نوع زنبق است

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

طول و عرض کاسبرگ و گلبرگ است چهار ویژگی گلها است ترسیم ویژگیها در چهار بعد میسر نیست و از روشهای دیگری باید بهره گرفت

طبقه بندی چند کلاسی

مثال: طبقه بندی چندکلاسی فایل دیتای ایریس

فایل:8 multiclass regression iris petal.py

```
import seaborn as sns
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, classification_report
# Load dataset
data = load_iris()
X = data.data[:, 2:4]  # Select only petal length and petal width
y = data.target
# Map target numbers to iris species names
species_mapping = {0: 'Setosa', 1: 'Versicolor', 2: 'Virginica'}
species_names = [species_mapping[label] for label in y]
# Convert to DataFrame for easier plotting and visualization
df = pd.DataFrame(X, columns=['petal length (cm)','petal width (cm)'])
df['species'] = species_names  # Use species names instead of numbers
# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.3, random_state=0)
```

طبقه بندی چند کلاسی

مثال: طبقه بندی چندکلاسی فایل دیتای ایریس

```
# Initialize and train logistic regression model for multiclass classification
logreg = LogisticRegression(multi_class='multinomial', solver='lbfgs',
max_iter=200)
logreg.fit(X_train, y_train)

# Predict and evaluate
y_pred = logreg.predict(X_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
print(classification_report(y_test, y_pred,
target_names=species_mapping.values()))

# Convert test labels and predictions to species names for plotting
y_test_names = [species_mapping[label] for label in y_test]
y_pred_names = [species_mapping[label] for label in y_pred]
# Plotting classification results for petal length and petal width
plt.figure(figsize=(8, 6))
sns.scatterplot(x=X_test[:, 0], y=X_test[:, 1], hue=y_test_names,
style=y_pred_names, palette='viridis', s=100, edgecolor="k")
plt.xlabel('Petal Length (cm)')
plt.ylabel('Petal Width (cm)')
plt.title('Logistic Regression Classification: Petal Length vs Petal Width')
plt.legend(title="True Species")
plt.show()
```

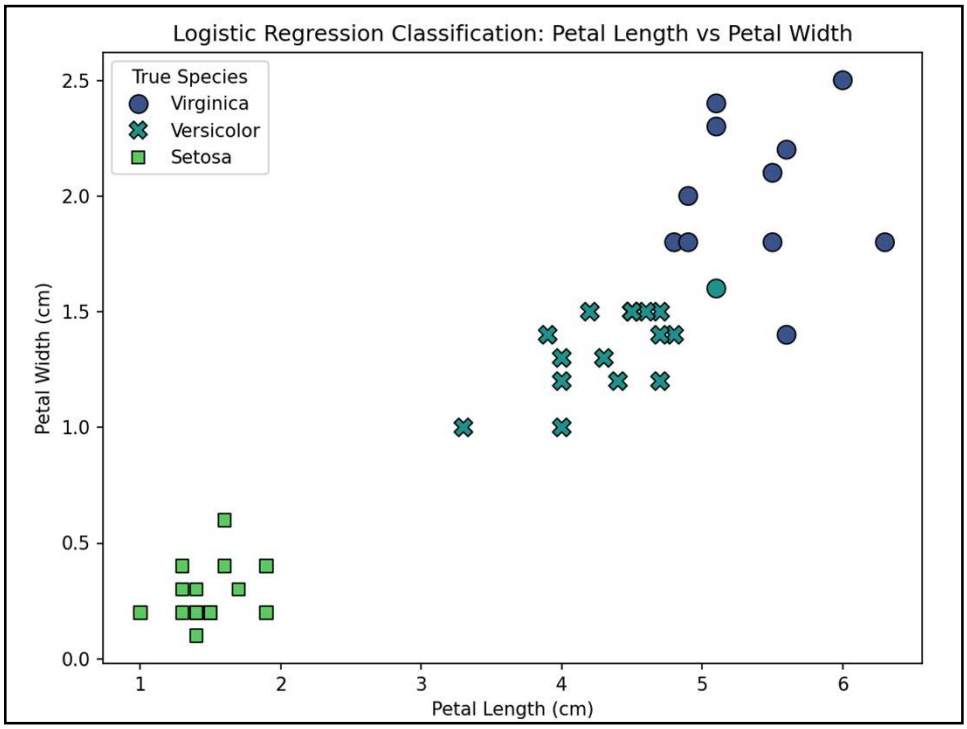

طبقه بندی چند کلاسی

مثال: طبقه بندی چندکلاسی فایل دیتای ایریس

```
Accuracy: 0.9777777777777777
              precision    recall  f1-score   support

      Setosa       1.00      1.00      1.00        16
  Versicolor       1.00      0.94      0.97        18
   Virginica       0.92      1.00      0.96        11

    accuracy                           0.98        45
   macro avg       0.97      0.98      0.98        45
weighted avg       0.98      0.98      0.98        45
```



طبقه بندی چند کلاسی

مثال: طبقه بندی چندکلاسی فایل دیتای ایریس با چهار ویژگی طول وعرض گلبرگ و کاسبرگ

فایل:8 multiclass regression iris pairplot.py

```
import seaborn as sns
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, classification_report

# Load dataset
data = load_iris()
X = data.data
y = data.target
# Map target numbers to iris species names
species_mapping = {0: 'Setosa', 1: 'Versicolor', 2: 'Virginica'}
species_names = [species_mapping[label] for label in y]

# Convert to DataFrame for easier plotting with species names
df = pd.DataFrame(X, columns=data.feature_names)
df['species'] = species_names  # Use species names instead of numbers
```

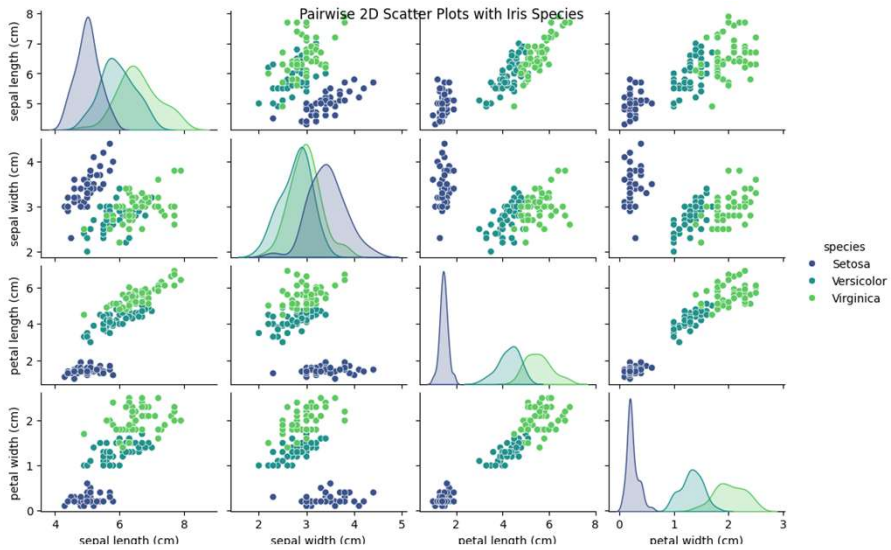
طبقه بندی چند کلاسی

مثال: طبقه بندی چندکلاسی فایل دیتای ایریس با چهار ویژگی طول وعرض گلبرگ و کاسبرگ

```
# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.3, random_state=0)
# Initialize and train logistic regression model for multiclass
classification
logreg = LogisticRegression(multi_class='multinomial',
solver='lbfgs', max_iter=200)
logreg.fit(X_train, y_train)
# Predict and evaluate
y_pred = logreg.predict(X_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
print(classification_report(y_test, y_pred,
target_names=species_mapping.values()))

# Pairplot for 4D visualization using seaborn with species names
plt.figure(figsize=(8, 5))
sns.pairplot(df, hue='species', palette='viridis')
plt.suptitle("Pairwise 2D Scatter Plots with Iris Species", y=0.99)
plt.show()
```

Soft Computing 33



Soft Computing 34

تمرین برنامه نویسی

تمرین هشتم :

یک برنامه به زبان پایتون بنویسید که یک فایل داده را خوانده و رگرسیون لجستیک برای سه ویژگی و دو کلاس را حساب نماید.

۱- داده ها را طبقه بندی کنید

۲- طبقه بندی انجام شده را سه بعدی رسم کنید

۳- دقت طبقه بندی را بدست آورید

۲- برای یک حالت با ویژگیهای مشخص، کلاس را مشخص کنید

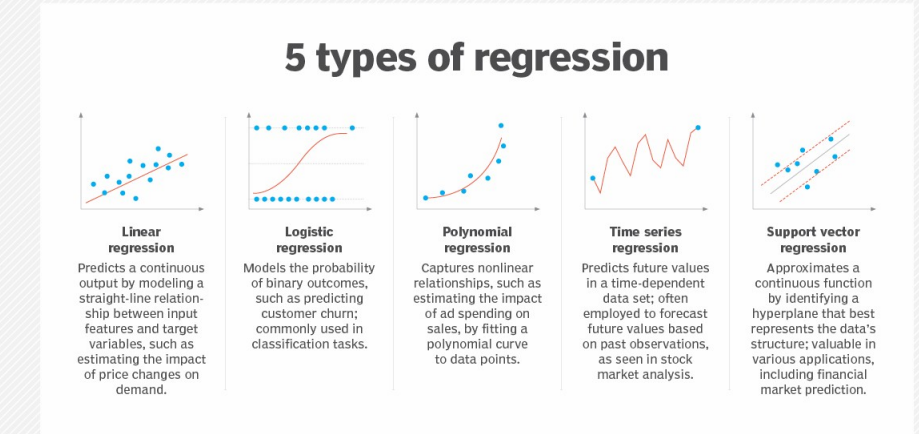
Non-Linear Regression

- Note that linear regression is to regression what the perceptron is to classification
  - Simple, useful models which will often underfit
- The more powerful classification models which we will be discussing going forward in class can usually also be used for non-linear regression
  - MLP with Backpropagation, Decision Trees, Nearest Neighbor, etc.
- They can learn functions with arbitrarily complex high dimensional shapes

Summary

- Linear Regression and Logistic Regression are nice tools for many simple situations
  - But both force us to fit the data with one shape (line or sigmoid) which will often underfit
- Intelligible results
- When problem includes more arbitrary non-linearity then we need more powerful models which we will introduce
  - Yet non-linear data transformations (e.g. Quadric perceptron) can help in these cases while still using a linear model for learning
- These models are commonly used in data mining applications and also as a "first attempt" at understanding data trends, indicators, etc.

رگرسیون



یادگیری نظارت شده – رگرسیون

الگوریتم‌های معروف رگرسیون:

رگرسیون خطی

رگرسیون چندجمله‌ای

رگرسیون ریج

رگرسیون درخت تصمیم

رگرسیون جنگل تصادفی

رگرسیون ماشین بردار پشتیبان