

Exp 3.

06.0.8.25.

STUDY OF CLASSIFIERS UNT STATISTICAL PARAMETERS.

Aim:

To implement various classifier IRIS dataset and analysis the statistical parameter.

Pseudo code:

for KNN:-

1. complete the distance $x - \text{test}, x_i$
2. Sort all distance in ascending order.
3. Select first K training points.
4. Count frequency of each label
5. Return the label of highest frequency predicted class.

for LOGISTIC REGRESSION:-

1. Compute linear combination (Z) = $Z = x_0 + b$
2. Apply sigmoid function: $\hat{y} = \text{sigmoid}(Z) = 1/(1+e^{-Z})$
3. Compute loss.
4. Compute gradients.
5. Update parameters.

$$w = w - a^*dw$$

$$b = b - a^*db$$

for NAIVE BAYES:-

Training phase:-

1. for each class c in all class:
→ cal. prior probability
 $P(c) = \text{count}(c) / \text{total sample}$
2. for each feature j :
3. for test point $x - \text{test}$: $P(A|B) = \frac{P(B|A)P(A)}{P(B)}$

Observation:

① → KNN

Accuracy ~~before noise~~: 0.9607

Accuracy (after noise): 0.840.

② → Logistic Regression.

Accuracy ~~before noise~~ = 0.973

Accuracy (after noise): 0.827

③ → Naive Bayes.

Accuracy ~~before noise~~: 0.960

Accuracy (after noise): 0.833

Justification:

- clean data provides + high accuracy
- small samples, generalization
- well separated feature
- Balanced classes

Table:

	KNN	logistic regression.	Naive Bayes
Accuracy.	0.840	before noise 0.827	before noise 0.833
Precision	0.8486	before noise 0.838	before noise 0.833
recall	0.8400	before noise 0.827	before noise 0.833
f1 score.	0.84100	before noise 0.827	before noise 0.833

Result:

Implemented diff. classifier on same data set & analyzed accuracy rate.

~~14/10/28~~

MSN | Personalized News, Top | X Week3.py (7) - JupyterLab X +

Not Secure http://10.1.38.19/user/ra2311047010012/lab/tree/Foundation of AI/SEM 5 DLT LAB/Week3.py

File Edit View Run Kernel Tabs Settings Help

Week2.py Week3.py

[3]:

```
import numpy as np
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import precision_score, recall_score, f1_score, accuracy_score
data = load_iris()
X, y = data.data, data.target
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
classifiers = {
    'Logistic Regression': LogisticRegression(max_iter=200),
    'Random Forest': RandomForestClassifier(n_estimators=50),
    'SVM': SVC(gamma='scale'),
    'KNN': KNeighborsClassifier(n_neighbors=5)
}

def perturb_predictions(y_pred, noise_level=0.1):
    """Randomly flip 'noise_level' fraction of predictions to simulate imperfect classification"""
    y_pred_noisy = y_pred.copy()
    n_flip = int(len(y_pred) * noise_level)
    flip_indices = np.random.choice(len(y_pred), size=n_flip, replace=False)

    for idx in flip_indices:
        current_class = y_pred_noisy[idx]
        possible_classes = list(set(y_pred_noisy) - {current_class})
        y_pred_noisy[idx] = np.random.choice(possible_classes)
```

VARIABLES
CALLSTAS...
BREAKPOINTS
SOURCE
KERNEL SOURCES

Simple 1 6 Python 3 (ipykernel) | idle Mem: 523.22 MB Mode: Command Ln 1, Col 1 Week3.py 1 09:37 ENG 07-08-2025

Type here to search

File Edit View Run Kernel Tabs Settings Help



Filter files by name



/ Foundation of AI / SEM 5 DLT

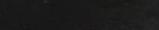
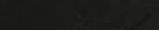
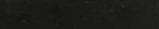
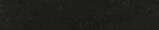
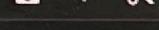
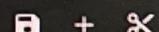
LAB /

Name

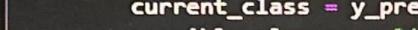
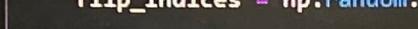
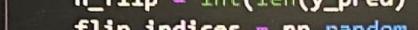
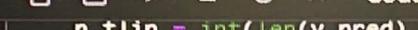
Last Modified

- Week2.py next year
- Week3.py next year

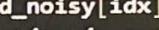
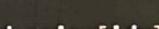
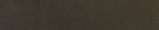
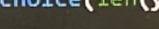
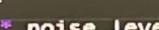
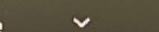
Week2.py



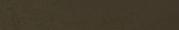
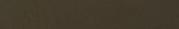
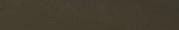
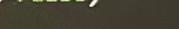
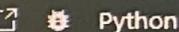
Week3.py



Code



Notebook



Python 3 (ipykernel)



```
n_flip = int(len(y_pred) * noise_level)
flip_indices = np.random.choice(len(y_pred), size=n_flip, replace=False)

for idx in flip_indices:
    current_class = y_pred_noisy[idx]
    possible_classes = list(set(y_pred_noisy) - {current_class})
    y_pred_noisy[idx] = np.random.choice(possible_classes)

return y_pred_noisy

for name, clf in classifiers.items():
    clf.fit(X_train, y_train)
    y_pred = clf.predict(X_test)
    y_pred_noisy = perturb_predictions(y_pred, noise_level=0.15)
    precision = precision_score(y_test, y_pred_noisy, average='macro', zero_division=0)
    recall = recall_score(y_test, y_pred_noisy, average='macro', zero_division=0)
    f1 = f1_score(y_test, y_pred_noisy, average='macro', zero_division=0)
    accuracy = accuracy_score(y_test, y_pred_noisy)
    print(f'Classifier: {name}')
    print(f'Precision: {precision:.3f}')
    print(f'Recall: {recall:.3f}')
    print(f'F1 Score: {f1:.3f}')
    print(f'Accuracy: {accuracy:.3f}')
    print('-----')
```

Classifier: Logistic Regression

Precision: 0.882

Recall: 0.862

F1 Score: 0.864

Accuracy: 0.867

Simple



1



S



6



G

Python 3 (ipykernel) | Idle

Mem: 523.22 MB

Mode: Com



Type here to search



