

06.08.25.

STUDY OF CLASSIFIERS WRT STATISTICAL PARAMETERS.

Aim:

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

pseudocode:

For KNN:-

1. compute the distance x -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 freq. the predicted class.

For LOGISTIC REGRESSION:-

1. compute linear combination $(z) = \tilde{z} = x_0 + b$
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
 $P(c) = \text{count}(c) / \text{total-sample}$
 - for each feature j :
2. for test point x -test: $P(A|B) = \frac{P(B|A)P(A)}{P(B)}$

Observation:

① → KNN

Accuracy ~~was~~ (before noise): 0.9607

Accuracy (after noise): 0.840

② → Logistic Regression.

Accuracy ~~was~~ (before noise) = 0.973

Accuracy (after noise): 0.827

③ → Naive Bayes

Accuracy ~~was~~ (before noise): 0.960

Accuracy (after noise): 0.833

Justification:

- clean data
- small samples
- well separated feature
- Balanced classes

Table:

	KNN	logistic regression.	Naive Bayes.
Accuracy	0.840 0.840	0.827 0.827	0.833 0.833
Precision	0.8486	0.838 0.838	0.833 0.833
Recall	0.8400	0.827 0.827	0.833 0.833
F1 Score	0.8410	0.827 0.827	0.833 0.833

Result: Implemented diff. classifier same data set & analysed accuracy rate.

~~14/8/25~~



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 x

Week3.py x +



Code



Notebook



Python 3 (ipykernel)



```
[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 classifica
    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

CALLSTA...

BREAKPOINTS

SOURCE

KERNEL SOURCES



Filter files by name 🔍

/ Foundation of AI / SEM 5 DLT LAB /

Name	Last Modified
Week2.py	next year
Week3.py	next year

```
Week2.py x Week3.py x +
+ ✂ 📄 ▶ ⏮ ⏭ 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

The screenshot displays a Jupyter Notebook environment. On the left, a file explorer shows the directory structure: / Foundation of AI / SEM 5 DLT LAB /, with files Week2.py and Week3.py listed. The main notebook editor shows the output of a classification task, comparing four models: Logistic Regression, Random Forest, SVM, and KNN. Each model's performance is evaluated using Precision, Recall, F1 Score, and Accuracy. The results are as follows:

Classifier	Precision	Recall	F1 Score	Accuracy
Logistic Regression	0.882	0.862	0.864	0.867
Random Forest	0.868	0.887	0.870	0.867
SVM	0.871	0.854	0.860	0.867
KNN	0.862	0.870	0.862	0.867

The right sidebar contains sections for VARIABLES, CALLSTACK, BREAKPOINTS, SOURCE, and KERNEL SOURCES, which are currently empty.