NAMA : Muhamad Rasyid Habibbarkah

KELAS : A1 – INF

NPM : 41155050190016

Semester 7

UAS MACHINE LEARNING

JAWABAN

BAGIAN I

1. Linear Regression adalah suatu teknik permodelan masalah keterhubungan antara suatu variabel independen terhadap variabel dependen. Fungsinya untuk membuat prediksi kuantitatif (bernilai nyata) berdasarkan suatu (vektor) fitur atau atribut. Logistic Regression adalah teknik analisis data yang menggunakan matematika untuk menemukan hubungan antara dua faktor data. Kemudian menggunakan hubungan ini untuk memprediksi nilai dari salah satu faktor tersebut berdasarkan faktor yang lain. Fungsinya untuk melakukan prediksi apakah suatu bernilai benar atau salah (0 atau 1).
2. Support Vector Machine atau SVM adalah algoritme pembelajaran mesin yang diawasi yang dapat digunakan untuk klasifikasi dan regresi. Fungsinya untuk mengolah data menjadi Hyperplane yang mengklasifikasikan ruang input menjadi dua kelas.
3. K-Nearest Neighbor adalah algoritma supervised learning dimana hasil dari instance yang baru diklasifikasikan berdasarkan mayoritas dari kategori k-tetangga terdekat. Fungsinya untuk mengklasifikasikan obyek baru berdasarkan atribut dan sample-sample dari training data.
4. Naive Bayes adalah metode pengklasifikasian berdasarkan probabilitas sederhana dan dirancang agar dapat dipergunakan dengan asumsi antar variabel penjelas saling bebas (independen). Fungsinya untuk untuk melakukan evaluasi di banyak hal, contohnya untuk memprediksi apakah besok cuaca akan cerah, hujan atau berangin berdasarkan parameter suhu, kelembapan, tekanan dan lain-lain.
5. Decision Tree adalah salah satu cara data processing dalam memprediksi masa depan dengan cara membangun klasifikasi atau regresi model dalam bentuk struktur pohon. Hal tersebut dilakukan dengan cara memecah terus ke dalam himpunan bagian yang lebih kecil

lalu pada saat itu juga sebuah pohon keputusan secara bertahap dikembangkan. Hasil akhir dari proses tersebut adalah pohon dengan node keputusan dan node daun. Fungsinya untuk untuk mem-break down proses pengambilan keputusan yang kompleks menjadi lebih simple, sehingga pengambil keputusan akan lebih menginterpretasikan solusi dari permasalahan.

6. Random Forest adalah kumpulan dari decision tree atau pohon keputusan. Algoritma ini merupakan kombinasi masing-masing tree dari decision tree yang kemudian digabungkan menjadi satu model. Fungsinya dapat digunakan untuk masalah klasifikasi dan regresi.
7. K-Means adalah suatu metode penganalisaan data yang melakukan proses pemodelan unsupervised learning dan menggunakan metode yang mengelompokan dataset yang belum dilabel ke dalam kluster yang berbeda. Fungsinya untuk mengelompokkan data kedalam data kluster.
8. Agglomerative Clustering adalah strategi pengelompokan hirarki yang dimulai dengan setiap objek dalam satu cluster yang terpisah kemudian membentuk cluster yang semakin membesar. Jadi, banyaknya cluster awal adalah sama dengan banyaknya objek. Fungsinya untuk mengelompokkan objek-objek berdasarkan karakteristik yang dimilikinya, yang dimulai dengan objek-objek individual sampai objek-objek tersebut bergabung menjadi satu cluster tunggal.
9. Apriori Algorithm adalah algoritma yang digunakan untuk mencari frequent item/itemset pada transaksional database. Fungsinya untuk mengatasi munculnya frequent item/itemset dalam pencarian nilai support dan nilai confidence pada database yang cukup besar, sehingga dapat menghasilkan asosiasi rule mining tanpa melakukan candidate generation.
10. Self-Organizing Maps (SOM) adalah teknik pembelajaran mesin tanpa pengawasan yang digunakan untuk menghasilkan representasi dimensi rendah (biasanya dua dimensi) dari kumpulan data berdimensi lebih tinggi sambil mempertahankan struktur topologi dari data. Fungsinya adalah untuk melakukan visualisasi data dengan cara mengurangi dimensi data melalui penggunaan self-organizing neural networks sehingga manusia dapat mengerti high-dimensional data yang dipetakan dalam bentuk low-dimensional data.

BAGIAN II

Metode Logistic Regression

In [1]:
```
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
from sklearn.datasets import make_classification
from sklearn.preprocessing import StandardScaler
```

In [2]:
```
dataLiga = pd.read_csv("../Machine Learning/Liga120192021.csv")
```

In [3]:
```
dataLiga.head()
```

Out[3]:

	Pass1	Pass2	Pass3	Pass4	Pass5	Pass6	Pass7	Pass8	Pass9	Pass10
0	11	24	2	20	10	11	13	11	16	71
1	10	11	13	11	20	12	13	20	77	71
2	16	8	16	17	21	22	3	20	10	13
3	22	16	8	16	2	17	23	8	82	4
4	20	12	16	8	16	17	21	23	22	13

In [4]:
```
dataLiga.isna().values.any()
```

Out[4]:
```
False
```

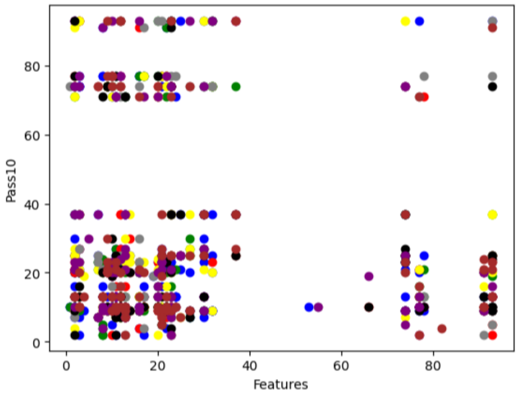
In [5]:
```
print(dataLiga.dtypes)
```

```
Pass1     int64
Pass2     int64
Pass3     int64
Pass4     int64
Pass5     int64
Pass6     int64
Pass7     int64
Pass8     int64
Pass9     int64
Pass10    int64
dtype: object
```

In [6]:
```
X = dataLiga.iloc[:, :-1]
y = dataLiga.iloc[:, -1]
```

In [7]:
```
plt.xlabel("Features")
plt.ylabel("Pass10")

pltX = dataLiga.loc[:, "Pass1"]
pltY = dataLiga.loc[:, "Pass10"]
plt.scatter(pltX, pltY, color="blue", label="Pass1")

pltX = dataLiga.loc[:, "Pass2"]
pltY = dataLiga.loc[:, "Pass10"]
plt.scatter(pltX, pltY, color="blue", label="Pass2")

pltX = dataLiga.loc[:, "Pass3"]
pltY = dataLiga.loc[:, "Pass10"]
plt.scatter(pltX, pltY, color="green", label="Pass3")

pltX = dataLiga.loc[:, "Pass4"]
pltY = dataLiga.loc[:, "Pass10"]
plt.scatter(pltX, pltY, color="red", label="Pass4")

pltX = dataLiga.loc[:, "Pass5"]
pltY = dataLiga.loc[:, "Pass10"]
plt.scatter(pltX, pltY, color="yellow", label="Pass5")

pltX = dataLiga.loc[:, "Pass6"]
pltY = dataLiga.loc[:, "Pass10"]
plt.scatter(pltX, pltY, color="gray", label="Pass6")

pltX = dataLiga.loc[:, "Pass7"]
pltY = dataLiga.loc[:, "Pass10"]
plt.scatter(pltX, pltY, color="black", label="Pass7")

pltX = dataLiga.loc[:, "Pass8"]
pltY = dataLiga.loc[:, "Pass10"]
plt.scatter(pltX, pltY, color="purple", label="Pass8")

pltX = dataLiga.loc[:, "Pass9"]
pltY = dataLiga.loc[:, "Pass10"]
plt.scatter(pltX, pltY, color="brown", label="Pass9")
```

Out[7]: <matplotlib.collections.PathCollection at 0x1bc6c00d190>



In [8]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=42)
```

In [9]:
```
model = LogisticRegression()
model.fit(X_train, y_train)
```

```
C:\Users\LENOVO\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\linear_model\_logistic.py:458: ConvergenceWar
ning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
```

Out[9]:
```
▸ LogisticRegression
LogisticRegression()
```

In [10]:
```
predictions = model.predict(X_test)
print(predictions)
```

```
[93 74  9 30 30 71 21 20 24  9 71 30 74 10 13  9 37  9  7 37  9  5 20  9
 13  9]
```

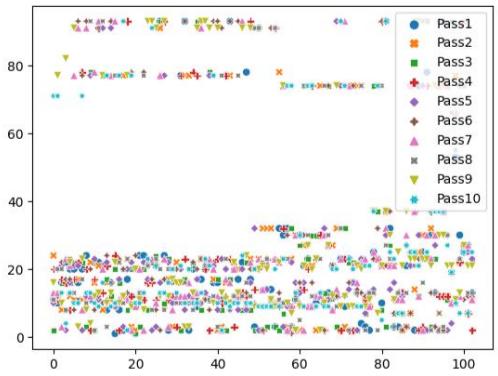
In [11]:
```
print(classification_report(y_test, predictions))
print("accuracy: ", accuracy_score(y_test, predictions))
```

```
              precision    recall  f1-score   support

           2       0.00      0.00      0.00       1.0
           5       0.00      0.00      0.00       0.0
           7       0.00      0.00      0.00       1.0
           9       0.00      0.00      0.00       2.0
          10       0.00      0.00      0.00       7.0
          13       0.00      0.00      0.00       4.0
          16       0.00      0.00      0.00       1.0
          19       0.00      0.00      0.00       1.0
          20       0.00      0.00      0.00       2.0
          21       0.00      0.00      0.00       2.0
          24       0.00      0.00      0.00       0.0
          25       0.00      0.00      0.00       1.0
          30       0.00      0.00      0.00       0.0
          37       0.00      0.00      0.00       0.0
          71       0.00      0.00      0.00       1.0
          74       0.00      0.00      0.00       2.0
          93       0.00      0.00      0.00       1.0

    accuracy                           0.00      26.0
   macro avg       0.00      0.00      0.00      26.0
weighted avg       0.00      0.00      0.00      26.0


accuracy:  0.0
```

```
C:\Users\LENOVO\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\metrics\_classification.py:1344: UndefinedMet
ricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division`
parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\LENOVO\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\metrics\_classification.py:1344: UndefinedMet
ricWarning: Recall and F-score are ill-defined and being set to 0.0 in labels with no true samples. Use `zero_division` paramet
er to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\LENOVO\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\metrics\_classification.py:1344: UndefinedMet
ricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division`
parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\LENOVO\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\metrics\_classification.py:1344: UndefinedMet
ricWarning: Recall and F-score are ill-defined and being set to 0.0 in labels with no true samples. Use `zero_division` paramet
er to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\LENOVO\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\metrics\_classification.py:1344: UndefinedMet
ricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division`
parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\LENOVO\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\metrics\_classification.py:1344: UndefinedMet
ricWarning: Recall and F-score are ill-defined and being set to 0.0 in labels with no true samples. Use `zero_division` paramet
er to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

In [12]:
```
# Scatter Plot
```

In [13]:
```
sns.scatterplot(data = dataLiga)
plt.show()
```



https://github.com/rasyidHabibbarkah/Machine-Learning.git