

Student Name: Abdulaziz Dafer Ali Alobud

Student Id : 253001366

Dataset link : <https://www.kaggle.com/datasets/imdevskp/corona-virus-report>

```
import os
import numpy as np
import pandas as pd

from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.impute import SimpleImputer

from sklearn.metrics import (
    accuracy_score, f1_score, precision_score, recall_score,
    confusion_matrix, classification_report
)

from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC

from sklearn.feature_selection import SelectKBest,
mutual_info_classif, RFE

DATASET_LINK = "https://www.kaggle.com/datasets/imdevskp/corona-virus-report"
DATA_PATH = "/content/drive/MyDrive/country_wise_latest.csv"
TARGET_COL = "WHO Region"

RANDOM_STATE = 42
TEST_SIZE = 0.2
K_BEST = 10
RFE_N_FEATURES = 10

def evaluate(y_true, y_pred, title):
    print(f"\n=== {title} ===")
    print("Accuracy :", round(accuracy_score(y_true, y_pred), 4))
    print("F1-score :", round(f1_score(y_true, y_pred,
average="weighted"), 4))
    print("Precision:", round(precision_score(y_true, y_pred,
average="weighted", zero_division=0), 4))
```

```

    print("Recall    :", round(recall_score(y_true, y_pred,
average="weighted"), 4))
    print("\nConfusion Matrix:\n", confusion_matrix(y_true, y_pred))
    print("\nClassification Report:\n", classification_report(y_true,
y_pred, zero_division=0))

def main():
    print("Dataset link:", DATASET_LINK)

    df = pd.read_csv(DATA_PATH)
    print("Dataset shape:", df.shape)

    if "Country/Region" in df.columns:
        df = df.drop(columns=["Country/Region"])

    numeric_cols = df.select_dtypes(include=[np.number]).columns
    df[numeric_cols] = df[numeric_cols].apply(pd.to_numeric,
errors="coerce")
    df.replace([np.inf, -np.inf], np.nan, inplace=True)

    df = df.dropna(subset=[TARGET_COL]).copy()

    X = df.drop(columns=[TARGET_COL])
    y = df[TARGET_COL]

    X_train, X_test, y_train, y_test = train_test_split(
        X, y,
        test_size=TEST_SIZE,
        random_state=RANDOM_STATE,
        stratify=y
    )

    num_cols =
X_train.select_dtypes(include=[np.number]).columns.tolist()
    cat_cols = [c for c in X_train.columns if c not in num_cols]

    preprocessor = ColumnTransformer(
        transformers=[
            ("num", Pipeline([
                ("imputer", SimpleImputer(strategy="median")),
                ("scaler", StandardScaler())
            ]), num_cols),
            ("cat", Pipeline([
                ("imputer", SimpleImputer(strategy="most_frequent")),
                ("onehot", OneHotEncoder(handle_unknown="ignore"))
            ]), cat_cols)
        ]
    )

```

```

    ]
)

models = {
    "LogisticRegression": (
        LogisticRegression(max_iter=3000),
        {"model__C": [0.1, 1, 10]}
    ),
    "RandomForest": (
        RandomForestClassifier(random_state=RANDOM_STATE),
        {"model__n_estimators": [200, 400],
         "model__max_depth": [None, 10, 20]}
    ),
    "SVC": (
        SVC(),
        {"model__C": [0.5, 1, 2],
         "model__kernel": ["rbf", "linear"]}
    )
}

print("\n##### BEFORE FEATURE SELECTION
#####")
before_results = []

for name, (model, params) in models.items():
    pipe = Pipeline([("prep", preprocessor), ("model", model)])
    grid = GridSearchCV(pipe, params, cv=5, scoring="f1_weighted",
n_jobs=-1)
    grid.fit(X_train, y_train)

    preds = grid.best_estimator_.predict(X_test)
    evaluate(y_test, preds, f"{name} (Best)")

    before_results.append({
        "Model": name,
        "BestParams": str(grid.best_params_),
        "Accuracy": accuracy_score(y_test, preds),
        "F1": f1_score(y_test, preds, average="weighted")
    })

before_df = pd.DataFrame(before_results)
print("\nSummary Before Feature Selection:\n", before_df)

X_train_p = preprocessor.fit_transform(X_train)
X_test_p = preprocessor.transform(X_test)

kbest = SelectKBest(mutual_info_classif, k=min(K_BEST,

```

```

X_train_p.shape[1]))
X_train_k = kbest.fit_transform(X_train_p, y_train)
X_test_k = kbest.transform(X_test_p)

rfe = RFE(LogisticRegression(max_iter=3000),
          n_features_to_select=min(RFE_N_FEATURES,
X_train_p.shape[1]))
X_train_r = rfe.fit_transform(X_train_p, y_train)
X_test_r = rfe.transform(X_test_p)

print("\n##### AFTER FEATURE SELECTION
#####")

lr = LogisticRegression(max_iter=3000)
rf = RandomForestClassifier(random_state=RANDOM_STATE)

lr.fit(X_train_k, y_train)
preds = lr.predict(X_test_k)
evaluate(y_test, preds, "LogReg + SelectKBest")

lr.fit(X_train_r, y_train)
preds = lr.predict(X_test_r)
evaluate(y_test, preds, "LogReg + RFE")

rf.fit(X_train_k, y_train)
preds = rf.predict(X_test_k)
evaluate(y_test, preds, "RF + SelectKBest")

rf.fit(X_train_r, y_train)
preds = rf.predict(X_test_r)
evaluate(y_test, preds, "RF + RFE")

before_df.to_csv("results_before_feature_selection.csv",
index=False)
print("\nSaved: results_before_feature_selection.csv")

main()

Dataset link: https://www.kaggle.com/datasets/imdevskp/corona-virus-report
Dataset shape: (187, 15)

##### BEFORE FEATURE SELECTION #####

=== LogisticRegression (Best) ===
Accuracy : 0.3684
F1-score : 0.3194
Precision: 0.3374

```

Recall : 0.3684

Confusion Matrix:

```
[[6 0 0 4 0 0]
 [4 1 1 1 0 0]
 [1 1 1 2 0 0]
 [4 0 1 6 0 0]
 [1 0 0 1 0 0]
 [1 0 0 2 0 0]]
```

Classification Report:

	precision	recall	f1-score	support
Africa	0.35	0.60	0.44	10
Americas	0.50	0.14	0.22	7
Eastern Mediterranean	0.33	0.20	0.25	5
Europe	0.38	0.55	0.44	11
South-East Asia	0.00	0.00	0.00	2
Western Pacific	0.00	0.00	0.00	3
accuracy			0.37	38
macro avg	0.26	0.25	0.23	38
weighted avg	0.34	0.37	0.32	38

=== RandomForest (Best) ===

Accuracy : 0.4737

F1-score : 0.4557

Precision: 0.4891

Recall : 0.4737

Confusion Matrix:

```
[[4 1 2 3 0 0]
 [2 2 1 2 0 0]
 [1 1 3 0 0 0]
 [1 1 1 8 0 0]
 [0 0 2 0 0 0]
 [0 0 1 1 0 1]]
```

Classification Report:

	precision	recall	f1-score	support
Africa	0.50	0.40	0.44	10
Americas	0.40	0.29	0.33	7
Eastern Mediterranean	0.30	0.60	0.40	5
Europe	0.57	0.73	0.64	11
South-East Asia	0.00	0.00	0.00	2
Western Pacific	1.00	0.33	0.50	3
accuracy			0.47	38

macro avg	0.46	0.39	0.39	38
weighted avg	0.49	0.47	0.46	38

=== SVC (Best) ===

Accuracy : 0.2895

F1-score : 0.2123

Precision: 0.1682

Recall : 0.2895

Confusion Matrix:

```
[[6 0 0 4 0 0]
 [6 0 1 0 0 0]
 [1 0 0 4 0 0]
 [5 0 1 5 0 0]
 [0 0 0 2 0 0]
 [1 0 0 2 0 0]]
```

Classification Report:

	precision	recall	f1-score	support
Africa	0.32	0.60	0.41	10
Americas	0.00	0.00	0.00	7
Eastern Mediterranean	0.00	0.00	0.00	5
Europe	0.29	0.45	0.36	11
South-East Asia	0.00	0.00	0.00	2
Western Pacific	0.00	0.00	0.00	3
accuracy			0.29	38
macro avg	0.10	0.18	0.13	38
weighted avg	0.17	0.29	0.21	38

Summary Before Feature Selection:

Model

BestParams \

0 LogisticRegression {'model__C': 10}

1 RandomForest {'model__max_depth': 10, 'model__n_estimators': ...}

2 SVC {'model__C': 2, 'model__kernel': 'rbf'}

	Accuracy	F1
0	0.368421	0.319444
1	0.473684	0.455731
2	0.289474	0.212276

AFTER FEATURE SELECTION

=== LogReg + SelectKBest ===

Accuracy : 0.2895

F1-score : 0.2132

Precision: 0.1717

Recall : 0.2895

Confusion Matrix:

```
[[6 0 0 4 0 0]
 [6 0 0 1 0 0]
 [2 0 0 3 0 0]
 [5 0 1 5 0 0]
 [1 1 0 0 0 0]
 [1 0 0 2 0 0]]
```

Classification Report:

	precision	recall	f1-score	support
Africa	0.29	0.60	0.39	10
Americas	0.00	0.00	0.00	7
Eastern Mediterranean	0.00	0.00	0.00	5
Europe	0.33	0.45	0.38	11
South-East Asia	0.00	0.00	0.00	2
Western Pacific	0.00	0.00	0.00	3
accuracy			0.29	38
macro avg	0.10	0.18	0.13	38
weighted avg	0.17	0.29	0.21	38

=== LogReg + RFE ===

Accuracy : 0.3158

F1-score : 0.2566

Precision: 0.3509

Recall : 0.3158

Confusion Matrix:

```
[[5 0 0 5 0 0]
 [4 1 0 2 0 0]
 [0 0 0 5 0 0]
 [5 0 0 6 0 0]
 [0 0 0 2 0 0]
 [1 0 0 2 0 0]]
```

Classification Report:

	precision	recall	f1-score	support
Africa	0.33	0.50	0.40	10
Americas	1.00	0.14	0.25	7
Eastern Mediterranean	0.00	0.00	0.00	5
Europe	0.27	0.55	0.36	11

South-East Asia	0.00	0.00	0.00	2
Western Pacific	0.00	0.00	0.00	3
accuracy			0.32	38
macro avg	0.27	0.20	0.17	38
weighted avg	0.35	0.32	0.26	38

=== RF + SelectKBest ===

Accuracy : 0.4211
F1-score : 0.3632
Precision: 0.3245
Recall : 0.4211

Confusion Matrix:

```
[[5 1 2 2 0 0]
 [2 0 2 3 0 0]
 [1 1 3 0 0 0]
 [2 0 1 8 0 0]
 [1 0 1 0 0 0]
 [0 0 1 1 1 0]]
```

Classification Report:

	precision	recall	f1-score	support
Africa	0.45	0.50	0.48	10
Americas	0.00	0.00	0.00	7
Eastern Mediterranean	0.30	0.60	0.40	5
Europe	0.57	0.73	0.64	11
South-East Asia	0.00	0.00	0.00	2
Western Pacific	0.00	0.00	0.00	3
accuracy			0.42	38
macro avg	0.22	0.30	0.25	38
weighted avg	0.32	0.42	0.36	38

=== RF + RFE ===

Accuracy : 0.3947
F1-score : 0.368
Precision: 0.3459
Recall : 0.3947

Confusion Matrix:

```
[[4 1 2 3 0 0]
 [3 2 1 1 0 0]
 [1 1 2 1 0 0]
 [1 3 0 7 0 0]
 [1 1 0 0 0 0]
 [0 1 1 1 0 0]]
```

Classification Report:

	precision	recall	f1-score	support
Africa	0.40	0.40	0.40	10
Americas	0.22	0.29	0.25	7
Eastern Mediterranean	0.33	0.40	0.36	5
Europe	0.54	0.64	0.58	11
South-East Asia	0.00	0.00	0.00	2
Western Pacific	0.00	0.00	0.00	3
accuracy			0.39	38
macro avg	0.25	0.29	0.27	38
weighted avg	0.35	0.39	0.37	38

Saved: results_before_feature_selection.csv