Analysis of Global Covid-19 Data

Dataset

- Import Data Table from URL

```
owidCovid = Import[
   "https://raw.githubusercontent.com/owid/covid-19-data/master/public/data/owid-
     covid-data.csv"];
{nrow, ncol} = Dimensions[owidCovid]
Now
```

In[ ]:=

```
owidAsOf = Block[
  {$DateStringFormat = {"(", "Year", ".", "Month", ".", "Day", ")"}}, DateString[]]
```

In[ ]:=

```
owidCovid⟦1⟧
```

In[ ]:=

```
owidCovidReference = owidCovid⟦1, ;; 4⟧
```

- Create Dataset Object

In[ ]:=

```
datasetCovid = Dataset[
   Table[<|Table[owidCovid⟦1, j⟧ → owidCovid⟦i, j⟧, {j, ncol}]|>, {i, 2, nrow}]];
```

```
Length[
 headerCovidTimeSeries =
  {"total_cases", "new_cases", "new_cases_smoothed",
   "total_deaths", "new_deaths", "new_deaths_smoothed",
   "total_cases_per_million",
   "new_cases_per_million", "new_cases_smoothed_per_million",
   "total_deaths_per_million", "new_deaths_per_million",
   "new_deaths_smoothed_per_million",
   "new_vaccinations", "new_vaccinations_smoothed"}]
```

In[ ]:=

```
Length[
 headerCovidLatestStats =
  {"total_cases", "total_deaths",
   "total_cases_per_million", "total_deaths_per_million",
   "people_vaccinated_per_hundred", "population", "population_density",
   "median_age", "aged_70_older", "gdp_per_capita",
   "cardiovasc_death_rate", "diabetes_prevalence",
   "hospital_beds_per_thousand", "life_expectancy", "human_development_index"}]
```

In[∘]:=

```
datasetCovid = datasetCovid[All, {"continent" → Replace["" → "Planet"]}];
datasetCovid = datasetCovid[Select[#location ≠ "International" &]];
datasetCovid = datasetCovid[ReplaceAll["" → 0]];
Dimensions[datasetCovid]
```

In[∘]:=

```
datasetCovidTimeSeries = datasetCovid[GroupBy["location"],
  All, Join[owidCovidReference, headerCovidTimeSeries]]
```

In[∘]:=

```
datasetCovidLatestStats = datasetCovid[GroupBy["location"],
  Last, Join[owidCovidReference, headerCovidLatestStats]]
```

In[∘]:=

```
datasetCovidLatestStatsContinentAve = datasetCovidLatestStats[
   GroupBy["continent"], Mean, Prepend[headerCovidLatestStats, "continent"]];
datasetCovidLatestStatsContinentAve = datasetCovidLatestStatsContinentAve[
  Select[#continent ≠ "Planet" &]]
```

In[∘]:=

```
{Length[listLocations = Normal[DeleteDuplicates[datasetCovid[All, "location"]]]],
 Length[listContinents = Normal[Keys[datasetCovidLatestStatsContinentAve]]],
 Length[listCountries = Complement[listLocations, Append[listContinents, "World"]]]}
```

■ Visualise Time Series

In[∘]:=

```
viewCovidTimeSeries[location0_ : "World", series0_ : "new_cases"] :=
 Manipulate[
  If[asTimeSeries,
   DateListPlot[TimeSeries[
     Normal@Values@datasetCovidTimeSeries[location, All, {"date", series}]],
    PlotRange → All, Filling → Axis, PlotLabel →
     Style[location <> ":  " <> series <> "  " <> owidAsOf, Bold]],
   ListPlot[datasetCovidTimeSeries[location, All, series],
    PlotRange → All, Filling → Axis,
    PlotLabel → Style[location <> ":  " <> series <> "  " <> owidAsOf, Bold]]],
  {asTimeSeries, {False, True}},
  {{location, location0}, listLocations},
  {{series, series0}, headerCovidTimeSeries},
  ControlPlacement → Bottom]
```

In[∘]:=

```
viewCovidTimeSeries[]
```

■ Visualise Scatter Plot

In[◦]:=

```
viewCovidLatestStats[] :=
 Manipulate[
  ListPlot[datasetCovidLatestStats[All, {stat1, stat2}],
   AxesLabel → {stat1, stat2},
   PlotLabel → Style[stat1 <> " vs. " <> stat2 <> " " <> owidAsOf <> "\n", Bold],
   ImageSize → Large],
  {stat1, headerCovidLatestStats},
  {stat2, headerCovidLatestStats⟦2 ;;⟧},
  ControlPlacement → Bottom]
```

In[◦]:=

```
viewCovidLatestStats[]
```

■ Visualise Radar Plot

In[◦]:=

```
Length[latestStatsForComparison = DeleteCases[headerCovidLatestStats, "population"]]
```

In[◦]:=

```
viewCovidLatestStatsByContinentAve[
  continents_ : listContinents, stats_ : latestStatsForComparison] :=
 Module[
  {datasetToCompare, vector, lineThickness},
  datasetToCompare = datasetCovidLatestStatsContinentAve[
    Select[MemberQ[continents, #continent] &], stats];
  vector = Transpose[Standardize /@ Transpose[
      Normal[Values /@ Values[datasetToCompare]]]];
  lineThickness = Normal@Values@datasetCovidLatestStatsContinentAve[
        continents, "population"] / Total@Normal@
      Values@datasetCovidLatestStatsContinentAve[continents, "population"] / 25;
  RadialAxisPlot[vector,
   PlotStyle → Thickness /@ (lineThickness),
   PlotLegends → continents, AxesLabel → stats,
   PlotLabel →
    Style["Compare Continents by (Ave) Latest Stats " <> owidAsOf <> "\n", Bold],
   ImageSize → Large]]
```

In[◦]:=

```
viewCovidLatestStatsByContinentAve[]
```

In[◦]:=

```
viewCovidLatestStatsByContinentAve[
 {"Asia", "Europe", "Africa", "Oceania"},
 {"total_cases_per_million", "total_deaths_per_million",
  "population_density", "aged_70_older",
  "diabetes_prevalence", "hospital_beds_per_thousand",
  "gdp_per_capita", "human_development_index"}]
```

Analysis

■ Cluster Analysis (Unsupervised Machine Learning)

In[ ]:=

```
Length[attributeToClusterBy = DeleteCases[
   DeleteCases[latestStatsForComparison, "total_cases"], "total_deaths"]]
datasetToCluster = datasetCovidLatestStats[All, attributeToClusterBy];

vectorToCluster = Normal[Values /@ Values[datasetToCluster]];
standardizedVectorToCluster = Transpose[Standardize /@ Transpose[vectorToCluster]];
```

In[ ]:=

```
clusterAllLocations =
  FindClusters[Association[Thread[listLocations → standardizedVectorToCluster]]];
Grid[
 Prepend[
  Transpose[{Range[Length@clusterAllLocations],
    clusterAllLocations, Keys[Counts[Length /@ clusterAllLocations]]}],
  {"Cluster", "Members", "Count"}], Alignment → Left, Frame → All]
```

In[76]:=

```
viewCovidLatestStatsByCluster[location0_ : "World"] :=
 Manipulate[
  ListPlot[
   datasetCovidLatestStats[clusterAllLocations⟦clusterNumber⟧, {stat1, stat2}],
   AxesLabel → {stat1, stat2},
   PlotLabel → Style[stat1 <> " vs. " <> stat2 <> " for Cluster #" <>
      ToString@clusterNumber <> " " <> owidAsOf <> "\n", Bold],
   ImageSize → Large],
  {{clusterNumber, Position[MemberQ[#, location0] & /@ clusterAllLocations, True]⟦
     1, 1⟧}, 1, Length@clusterAllLocations, 1,
   ControlType → RadioButtonBar},
  {stat1, attributeToClusterBy},
  {stat2, attributeToClusterBy⟦2 ;;⟧},
  ControlPlacement → Bottom]
```

In[ ]:=

```
viewCovidLatestStatsByCluster[]
```

In[◦]:=

```
viewClustersByCentroidAttributes[] :=
 Module[
  {datasetLatestStatsClusterAve, clusterAveLatestStats,
   standardizedClusterAveLatestStats, linethickness},
  datasetLatestStatsClusterAve =
   Table[datasetCovidLatestStats[GroupBy[
      MemberQ[clusterAllLocations⟦i⟧, #location] &], Mean, attributeToClusterBy],
    {i, Length@clusterAllLocations}];
  clusterAveLatestStats =
   Normal[Values /@ Values[#]]⟦1⟧ & /@ datasetLatestStatsClusterAve;
  standardizedClusterAveLatestStats =
   Transpose[Standardize /@ Transpose[clusterAveLatestStats]];
                           Length /@ clusterAllLocations
  linethickness = N@ ──────────────────────────────────── / 25;
                      Total[Length /@ clusterAllLocations]
  RadialAxisPlot[standardizedClusterAveLatestStats,
   PlotStyle → Thickness /@ (linethickness),
   PlotLegends → Automatic, AxesLabel → attributeToClusterBy,
   PlotLabel →
    Style["Compare Cluster Centroids by Latest Stats " <> owidAsOf <> "\n", Bold],
   ImageSize → Large]]
```

In[◦]:=

```
viewClustersByCentroidAttributes[]
```

■ Unsupervised Machine Learning - Dimensionality Reduction

In[◦]:=

```
dimReducedVector = DimensionReduce[standardizedVectorToCluster];
datasetDimReducedVector =
  Dataset[Association[Thread[listLocations → dimReducedVector]]];
```

In[◦]:=

```
viewReducedDimensionRepresentation[location0_ : "World"] :=
 Manipulate[
  Show[
   ListPlot[datasetDimReducedVector[clusterAllLocations⟦clusterNumber⟧],
    AxesLabel → {"artificial dimension 1", "artificial dimension 2"},
    PlotLabel →
     Style["Dimensionality Reduced (2D) Representation: Countries in Cluster #" <>
       ToString@clusterNumber <> " vs. All " <> owidAsOf <> "\n", Bold],
    ImageSize → Large,
    PlotStyle → {Blue, PointSize[Large]}],
   ListPlot[datasetDimReducedVector[
     Complement[listLocations, clusterAllLocations⟦clusterNumber⟧]],
    PlotStyle → {Gray, PointSize[Small]}]],
  {{clusterNumber, Position[MemberQ[#, location0] & /@ clusterAllLocations, True]⟦
     1, 1⟧}, 1, Length@clusterAllLocations, 1,
   ControlType → RadioButtonBar},
  ControlPlacement → Bottom]
```

In[∘]:=

```
viewReducedDimensionRepresentation[]
```

■ Supervised Machine Learning - Numerical Prediction

In[94]:=

```
Length[listExplanatoryVariables =
  DeleteCases[DeleteCases[attributeToClusterBy, "total_cases_per_million"],
   "total_deaths_per_million"]]
```

In[ ]:=
```
viewCovidIncidenceActualVsBaselineFit[
  locationsToWatch_ :
   {"World", "Asia", "Thailand", "Europe", "United Kingdom", "North America",
    "United States", "South America", "Brazil", "Africa", "Oceania"}] :=
 Module[
  {x, y1, y2, y1Predictor, y2Predictor,
   y1Fitted, y2Fitted, y1Error, y2Error, rmseY1, rmseY2, fittingResult},
  x = Values /@
    Normal[Values[datasetCovidLatestStats[All, listExplanatoryVariables]]];
  y1 = Values[Normal[datasetCovidLatestStats[All, "total_cases_per_million"]]];
  y2 = Replace[#, "" → 0] & /@
    Values[Normal[datasetCovidLatestStats[All, "total_deaths_per_million"]]];
  y1Predictor = Predict[x → y1];
  y2Predictor = Predict[x → y2];
  y1Fitted = y1Predictor[#] & /@ x;
  y2Fitted = y2Predictor[#] & /@ x;
  y1Error = y1Fitted - y1;
  y2Error = y2Fitted - y2;
  rmseY1 = Round[Norm[y1Error] / Sqrt[Length@x], 1];
  rmseY2 = Round[Norm[y2Error] / Sqrt[Length@x], 1];
  fittingResult = Dataset[Association[Table[listLocations⟦i⟧ →
       <|"cpm" → Round[y1⟦i⟧, 1], "cpmEst" → Round[y1Fitted⟦i⟧, 1],
        "cpm/cpmEst" → Round[y1⟦i⟧ / y1Fitted⟦i⟧, 0.01],
        "dpm" → Round[y2⟦i⟧, 1], "dpmEst" → Round[y2Fitted⟦i⟧, 1],
        "dpm/dpmEst" → Round[y2⟦i⟧ / y2Fitted⟦i⟧, 0.01]|>, {i, Length@listLocations}]]];
  Grid[
   {{ListPlot[fittingResult[All, {"cpm", "cpmEst"}],
      AxesLabel → {"actual", "fitted"},
      PlotLabel → Style["Actual vs Fitted (cases/million) using " <>
         Information[y1Predictor]["Method"] <> " " <> owidAsOf, Bold],
      ImageSize → Large]},
    {ListPlot[fittingResult[All, {"dpm", "dpmEst"}],
      AxesLabel → {"actual", "fitted"},
      PlotLabel → Style["Actual vs Fitted (deaths/million) using " <>
         Information[y1Predictor]["Method"] <> " " <> owidAsOf, Bold],
      ImageSize → Large]},
    {fittingResult[locationsToWatch]}}, Frame → All]]
```

In[ ]:=
```
viewCovidIncidenceActualVsBaselineFit[]
```