Mathematicaでデータサイエンス

リモートカーネルの起動

In[ ]:=
```
(* リモート・カーネルを起動するホスト名を列挙する。 *)
hosts = {"localhost", "node11", "node12", "node13", "node14", "node15",
   "node16", "node17", "node18", "node19", "node20", "node21"};
(* ホスト一覧に対してカーネル数4、
 起動タイムアウト10秒のリモート・カーネル・オブジェクトを生成する。 *)
$DefaultKernels =
  RemoteKernelObject[#, "KernelCount" → 4, "TimeConstraint" → 30] & /@ hosts;
(* すべてのカーネルを起動する。 *)
LaunchKernels[]
```

クロス・バリデーション用のデータセットの作成

In[ ]:=
```
(*trani.csvから各属性の型を指定してリストとして取り込む。*)
fields = <|"PassengerId" → "Integer", "Survived" → "Integer",
   "Pclass" → "Integer", "Name" → "String", "Sex" → "String", "Age" → "Real",
   "SibSp" → "Integer", "Parch" → "Integer", "Ticket" → "String",
   "Fare" → "Real", "Cabin" → "Integer", "Embarked" → "String"|>;
original = SemanticImport["titanic/train.csv",
   fields, "Rows", MissingDataRules → {"" → Missing[]}];

(*チケットのクラス、年齢、性別、生存・死亡の別の属性のみ取り出す。*)
dataset = Table[List[Replace[original[[n, 3]], {1 → "1st", 2 → "2nd", 3 → "3rd"}],
     original[[n, 6]], original[[n, 5]]] → Replace[original[[n, 2]],
     {0 → "died", 1 → "survived"}], {n, Length[original]}];
(*クロス・バリデーション用に4つのfoldを作成する。*)
(*各foldは学習用データと検証用データのペア*)
fold = Table[TakeList[dataset, {s ;;  ;; 4, All}], {s, 4}]
```

勾配ブースティング決定木(GBDT)による分類関数の作成

In[ ]:=
```
(*4つのfoldに対して4つの分類関数を生成する。*)
classifiers = Table[Classify[fold[[i, 2]], Method → {"GradientBoostedTrees"},
    ValidationSet → fold[[i, 1]]], {i, Length[fold]}];
(*4つのfoldの推測結果の平均値を使って全体の推測値とする。*)
fmean[v_] := Mean[
   Table[classifiers[[i]][v, "Probability" → "survived"], {i, Length[fold]}]];
Information[classifiers]
```

分類結果をグラフ化し、正答率を計算する。

In[ ]:=
```
(*プロットする点を事前に並列計算する。*)
points = Partition[
   Parallelize@(fmean /@ Flatten[{{"1st", #, "female"}, {"3rd", #, "female"},
           {"1st", #, "male"}, {"3rd", #, "male"}} & /@ Range[0, 100], 1]), 4];
fx[x_, i_] := points[[Round[x] + 1, i]];
Plot[{fx[x, 1], fx[x, 2], fx[x, 3], fx[x, 4]}, {x, 0, 100},
 PlotLegends → {"1st-female", "3rd-female", "1st-male", "3rd-male"},
 Frame → True, FrameLabel → {"Age", "Survival Rate"},
 GridLines → Automatic, ImageSize → Large, PlotStyle → {Directive[Thick],
   Directive[Dashed, Thick], Directive[Thick], Directive[Dashed, Thick]}]
N[Count[(If[Last[#] == "survived", 1, 0] & /@ dataset) -
    (Round[#] & /@ fmean[First /@ dataset]), 0] / Length[dataset]]
```

In[ ]:=
```
(* リモート・カーネルを起動するホスト名を列挙する。 *)
hosts = {"localhost"};
(* ホスト一覧に対してカーネル数4、
 起動タイムアウト10秒のリモート・カーネル・オブジェクトを生成する。 *)
$DefaultKernels =
  RemoteKernelObject[#, "KernelCount" → 1, "TimeConstraint" → 30] & /@ hosts;
(* すべてのカーネルを起動する。 *)
LaunchKernels[]
points = Partition[Parallelize@(fmean /@
       Flatten[{{"1st", #, "female"}, {"3rd", #, "female"}, {"1st", #, "male"},
           {"3rd", #, "male"}} & /@ Range[0, 100], 1]), 4]; // AbsoluteTiming
```

In[ ]:=
```
CloseKernels[]
```

複数の分類アルゴリズムを並列実行しグラ

フ化する。

```
In[ ]:= algorithms = {"GradientBoostedTrees", "LogisticRegression",
          "NearestNeighbors", "NeuralNetwork", "SupportVectorMachine"};
        p = Length[fold];
        q = Length[algorithms];
        classifiers2 = Partition[Parallelize@
          Table[Classify[fold[[Mod[i, p] + 1, 2]], Method → algorithms[[Quotient[i, p] + 1]],
            ValidationSet → fold[[Mod[i, p] + 1, 1]]], {i, 0, p * q - 1}], p]
        fmean2[a_, v_] :=
          Mean[Table[classifiers2[[a, i]][v, "Probability" → "survived"], {i, 3}]];
        samples = Flatten[Table[
            {{a, {"1st", #, "female"}}, {a, {"3rd", #, "female"}}, {a, {"1st", #, "male"}},
              {a, {"3rd", #, "male"}}} & /@ Range[0, 100], {a, Length[algorithms]}], 2];
        points = ArrayReshape[
          Parallelize@(fmean2 @@ # & /@ samples), {Length[algorithms], 101, 4}];
        fx2[a_, x_, i_] := points[[a, Round[x] + 1, i]];
        Parallelize@
         Table[Plot[{fx2[a, x, 1], fx2[a, x, 2], fx2[a, x, 3], fx2[a, x, 4]}, {x, 0, 100},
          PlotLegends → {"1st-female", "3rd-female", "1st-male", "3rd-male"},
          Frame → True, FrameLabel → {"Age", "Survival Rate"}, GridLines → Automatic,
          ImageSize → Large, PlotStyle → {Directive[Thick], Directive[Dashed, Thick],
            Directive[Thick], Directive[Dashed, Thick]}], {a, Length[algorithms]}]
        Table[N[Count[(If[Last[#] == "survived", 1, 0] & /@ dataset) -
            (Round[#] & /@ fmean2[a, First /@ dataset]), 0] /
          Length[dataset]], {a, Length[algorithms]}]
```

Kaggle提出用のファイルを作成する。

```
In[ ]:= testdata = SemanticImport["titanic/test.csv", <|"PassengerId" → "Integer",
            "Pclass" → "Integer", "Sex" → "String", "Age" → "Real"|>,
          "Rows", MissingDataRules → {"" → Missing[]}];
        formatted = Table[
          List[testdata[[n, 1]], Replace[testdata[[n, 2]], {1 → "1st", 2 → "2nd", 3 → "3rd"}],
            testdata[[n, 4]], testdata[[n, 3]]], {n, Length[testdata]}];
        submission = Table[List[formatted[[n, 1]],
            If[fmean2[1, formatted[[n, 2 ;;]]] ≥ 0.5, 1, 0]], {n, Length[formatted]}];
        Export["titanic/submission.csv", submission,
         "TableHeadings" → {PassengerId, Survived}, "TextDelimiters" → ""]
```