Classification workflows conversational agent example runs

Anton Antonov
MathematicaForPrediction at GitHub
MathematicaForPrediction at WordPress
ConversationalAgents at GitHub
May 2018

Introduction

This notebook gives basic demonstrations of the functionalities developed in the project “Classification workflows conversational agent”.

Load preliminary code

Load the packages for the `ClCon` monad, monad tracing, functional parsers, and data obtaining:

```
In[1]:= Import["https://raw.githubusercontent.com/antononcube/MathematicaForPrediction/master/MonadicProgramming/MonadicContextualClassification.m"]
Import["https://raw.githubusercontent.com/antononcube/MathematicaForPrediction/master/MonadicProgramming/MonadicTracing.m"]
Import["https://raw.githubusercontent.com/antononcube/MathematicaForPrediction/master/FunctionalParsers.m"]
Import["https://raw.githubusercontent.com/antononcube/MathematicaVsR/master/Projects/ProgressiveMachineLearning/Mathematica/GetMachineLearningDataset.m"]
```

```
» Importing from GitHub: MathematicaForPredictionUtilities.m
» Importing from GitHub: MosaicPlot.m
» Importing from GitHub: CrossTabulate.m
» Importing from GitHub: StateMonadCodeGenerator.m
» Importing from GitHub: ClassifierEnsembles.m
» Importing from GitHub: ROCFunctions.m
» Importing from GitHub: VariableImportanceByClassifiers.m
» Importing from GitHub: SSparseMatrix.m
» Importing from GitHub: OutlierIdentifiers.m
```

Load project code

Get and run the parsers specification and generation code:

```
In[14]:= Clear["ebnf*"]
         Get["https://raw.githubusercontent.com/antononcube/ConversationalAgents/master/EBNF/ClassifierWorkflowsGrammar.ebnf"]
         Names["ebnf*"]
         LeafCount /@ res

Out[16]= {ebnfClassifierEnsembleMaking, ebnfClassifierMaking, ebnfClassifierQuery, ebnfClassifierTesting, ebnfCommand,
          ebnfDataLoad, ebnfDataOutliers, ebnfDataStatistics, ebnfDataTransform, ebnfPipelineCommands, ebnfSplitting, ebnfVerification}

Out[17]= {628, 763, 897, 466, 424, 960, 531, 905, 4465, 491, 843, 24}
```

Load the translator package:

```
In[18]:= Import["https://raw.githubusercontent.com/antononcube/ConversationalAgents/master/Projects/ClassficationWorkflowsAgent/Mathematica/ClConTranslator.m"]
```

Get data

Get Titanic data (using WL’s repository):

```
In[19]:= dsTitanic = GetMachineLearningDataset["Titanic"];
```

Generate pipeline

Generate a `ClCon` pipeline from a sequence of natural language commands:

```
In[20]:= clCommands = {
            "load the titanic data",
            "split the data with 65 percent for training", "summarize data", "train a random forest classifier",
            "show classifier information",
            "display classifier training time", "show accuracy, precision, recall, and area under roc curve", "display confusion matrix plot",
            "compute the variable importance estimates"};
         pl = ToClConPipelineFunction[clCommands]

Out[21]= Function[{x, c}, (((((((ClConUnit[x, c] ⟹ ClConSplitData[0.65]) ⟹ ClConEchoFunctionValue[summaries:, (Multicolumn[#1, 5] &) /@ RecordsSummary /@ #1 &]) ⟹ ClConMakeClassifier[RandomForest]) ⟹
                     ClConEchoFunctionContext[classifier info:, If[AssociationQ[#1[classifier]], ClassifierInformation /@ #1[classifier], ClassifierInformation[#1[classifier]]] &]) ⟹
                   ClConEchoFunctionContext[classifier property "TrainingTime" :, If[AssociationQ[#1[classifier]], (ClassifierInformation[#1, TrainingTime] &) /@ #1[classifier],
                      ClassifierInformation[#1[classifier], TrainingTime]] &]) ⟹
                 Function[{x$, c$}, ClConUnit[x$, c$] ⟹ ClConClassifierMeasurements[{Accuracy, Precision, Recall, AreaUnderROCCurve}] ⟹ ClConEchoValue]) ⟹
               Function[{x$, c$}, ClConUnit[x$, c$] ⟹ ClConClassifierMeasurements[{ConfusionMatrixPlot}] ⟹ ClConEchoValue]) ⟹
             Function[{x, c}, ClConUnit[x, c] ⟹ ClConAccuracyByVariableShuffling[] ⟹ ClConEchoValue]]
```

Run the generated pipeline

Run the generated pipeline over the Titanic data:

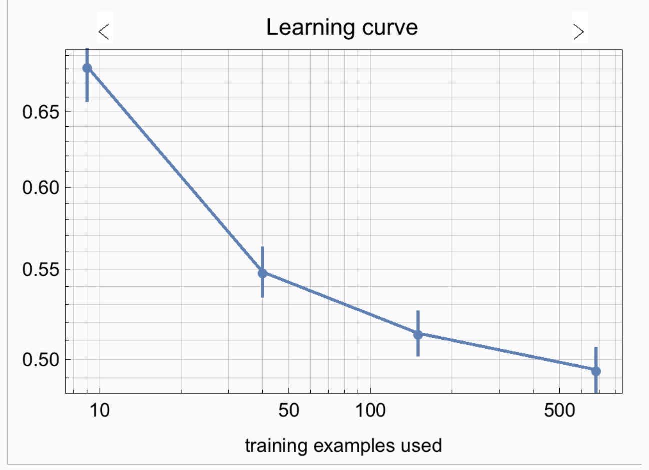
```
In[22]:= ClConUnit[dsTitanic] ⟹ pl;
```

```
                           1 id                                3 passengerAge                                                                  1 id                                  3 passengerAge
                           Min     1       2 passengerClass    Min     -1                                                                      Min     2       2 passengerClass      Min     -1
                           1st Qu  345                         1st Qu  10      4 passengerSex  5 passengerSurvival                             1st Qu  308                           1st Qu  0       4 passengerSex  5 passengerSurvival
» summaries: ⟨|trainingData → Mean    667.188  3rd  470    Median  20      male    545     died      525  , testData → Median  622     3rd  239          Median  20      male    298     died      284  |⟩
                           Median  675.5    1st  203       Mean    23.9965 female  305     survived  325               Mean    632.429  1st  120         Mean    22.7233 female  161     survived  175
                           3rd Qu  1000     2nd  177       3rd Qu  40                                                  3rd Qu  956.75   2nd  100         3rd Qu  37.5
                           Max     1307                    Max     80                                                  Max     1309                      Max     80
```

» classifier info:

Classifier information

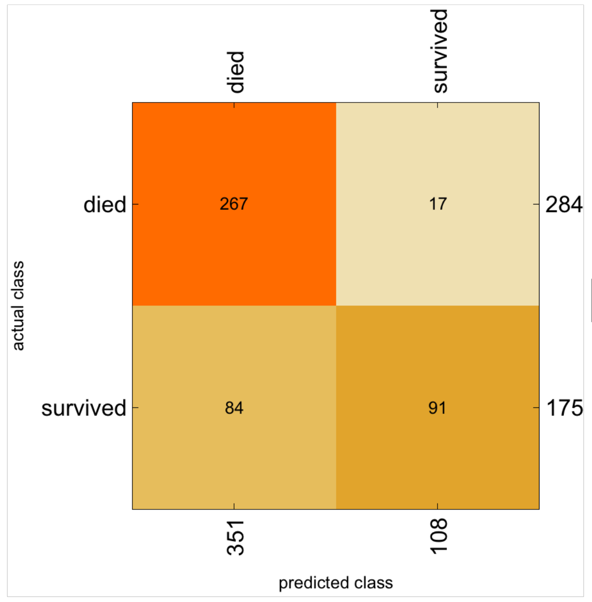
Input type	Mixed (number: 4)
Classes	died, survived
Method	RandomForest
Accuracy	78.2% ± 1.2%
Loss	0.494 ± 0.011
Single evaluation time	4.1 ms/example
Batch evaluation speed	37.1 examples/ms
Classifier memory	257. kB
Training examples used	850 examples
Training time	2.13 s



» classifier property "TrainingTime" : 2.13723 s

» value: ⟨|Accuracy → 0.779956, Precision → ⟨|died → 0.760684, survived → 0.842593|⟩, Recall → ⟨|died → 0.940141, survived → 0.52|⟩, AreaUnderROCCurve → ⟨|died → 0.86994, survived → 0.824004|⟩|⟩

```
» value: ⟨|ConfusionMatrixPlot →
```



```
|⟩
```

```
» value: ⟨|None → 0.779956, id → 0.786492, passengerClass → 0.782135, passengerAge → 0.764706, passengerSex → 0.605664|⟩
```

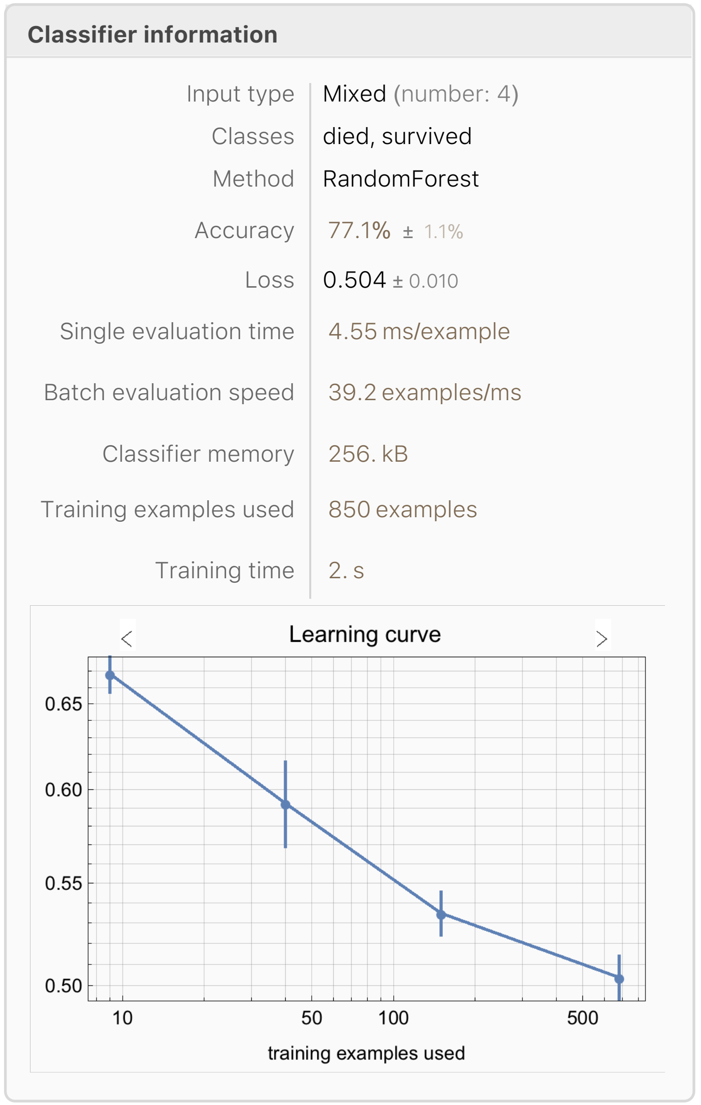
Trace run and command-code table

Run the generated pipeline through TraceMonad in order to obtained tabulated correspondence between (1) the generated ClCon pipeline components and (2) the natural language commands used to generate them:

```
In[23]:= (p =
    ClConUnit[dsTitanic] ⟹
     ToClConPipelineFunction[clCommands, "Trace" → True]) ⟹
  TraceMonadTakeGrid[]
```

```
                              1 id                          3 passengerAge                                             1 id                          3 passengerAge
                              Min      3    2 passengerClass Min      -1                                               Min      1     2 passengerClass Min      -1
                              1st Qu  348                    1st Qu  10       4 passengerSex  5 passengerSurvival      1st Qu  276.5                    1st Qu  10       4 passengerSex  5 passengerSurvival
» summaries: ⟨|trainingData → Median  656.5  3rd  468        Median  20       male    556     died      525  , testData → Mean    641.438 3rd  241        Median  20       male    287     died      284  |⟩
                              Mean    662.324 1st  193       Mean    23.1576  female  294     survived  325            Median  643    1st  130         Mean    24.2767  female  172     survived  175
                              3rd Qu  975    2nd  189        3rd Qu  40                                                3rd Qu  996.5  2nd  88          3rd Qu  40
                              Max     1309                   Max     70                                                Max     1307                    Max     80
```

» classifier info:

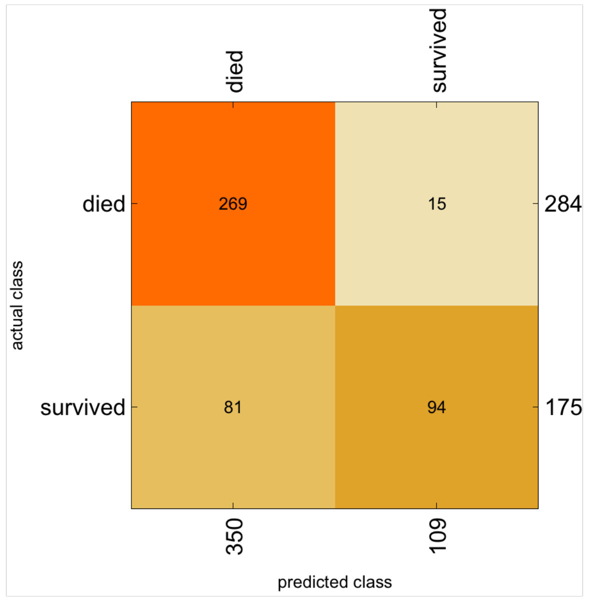


Classifier information

Input type	Mixed (number: 4)
Classes	died, survived
Method	RandomForest
Accuracy	77.1% ± 1.1%
Loss	0.504 ± 0.010
Single evaluation time	4.55 ms/example
Batch evaluation speed	39.2 examples/ms
Classifier memory	256. kB
Training examples used	850 examples
Training time	2. s

» classifier property "TrainingTime" : 2.00355 s

» value: ⟨|Accuracy → 0.79085, Precision → ⟨|died → 0.768571, survived → 0.862385|⟩, Recall → ⟨|died → 0.947183, survived → 0.537143|⟩, AreaUnderROCCurve → ⟨|died → 0.866519, survived → 0.820463|⟩|⟩

» value: ⟨|ConfusionMatrixPlot →  |⟩

```
» value: ⟨|None → 0.79085, id → 0.784314, passengerClass → 0.8061, passengerAge → 0.769063, passengerSex → 0.633987|⟩
```

Out[23]=

```
ClConUnit[x, c] ⟹
  ClConSplitData[0.65`] ⟹                                                                      split the data with 65 percent for training
  ClConEchoFunctionValue["summaries:", (Multicolumn[#1, 5] &) /@ RecordsSummary /@ #1 &] ⟹      summarize data
  ClConMakeClassifier["RandomForest"] ⟹                                                         train a random forest classifier
  ClConEchoFunctionContext["classifier info:",                                                  show classifier information
  If[AssociationQ[#1["classifier"]], ClassifierInformation /@ #1["classifier"], ClassifierInformation[#1["classifier"]]] &] ⟹
  ClConEchoFunctionContext["classifier property "TrainingTime" :",                             display classifier training time
  If[AssociationQ[#1["classifier"]], (ClassifierInformation[#1, "TrainingTime"] &) /@ #1["classifier"],
    ClassifierInformation[#1["classifier"], "TrainingTime"]] &] ⟹
  Function[{x$, c$}, ClConUnit[x$, c$] ⟹                                                        show accuracy, precision, recall, and area under roc curve
   ClConClassifierMeasurements[{"Accuracy", "Precision", "Recall", "AreaUnderROCCurve"}] ⟹ ClConEchoValue] ⟹
  Function[{x$, c$}, ClConUnit[x$, c$] ⟹ ClConClassifierMeasurements[{"ConfusionMatrixPlot"}] ⟹ ClConEchoValue] ⟹      display confusion matrix plot
  Function[{x, c}, ClConUnit[x, c] ⟹ ClConAccuracyByVariableShuffling[] ⟹ ClConEchoValue]      compute the variable importance estimates
```