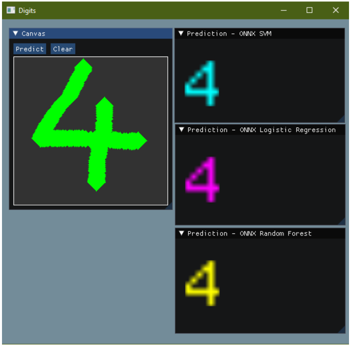
Scikit Model inference in C++

Abhilash Babu

Outline

- Using models trained using scikit-learn in C++
  - Available Options
  - Pros and Cons

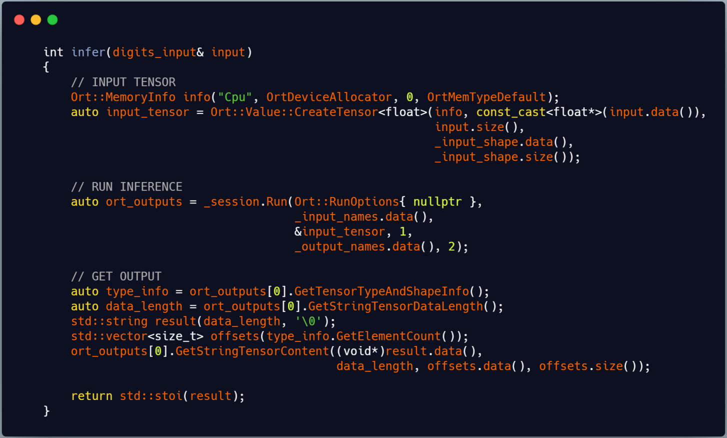

Digits
▼ Canvas
Predict
Clear
▼ Prediction - ONNX SVM
4
▼ Prediction - ONNX Logistic Regression
4
▼ Prediction - ONNX Random Forest
4

- Demo

Available Options

- Use an intermediate format
  - From the scikit-learn documentation
    - ONNX
    - PMML
- Use the same underlying library that scikit learn uses
  - liblinear
  - libsvm
- Other options
  - treelite

ONNX



```cpp
int infer(digits_input& input)
{
    // INPUT TENSOR
    Ort::MemoryInfo info("Cpu", OrtDeviceAllocator, 0, OrtMemTypeDefault);
    auto input_tensor = Ort::Value::CreateTensor<float>(info, const_cast<float*>(input.data()),
                                                        input.size(),
                                                        _input_shape.data(),
                                                        _input_shape.size());

    // RUN INFERENCE
    auto ort_outputs = _session.Run(Ort::RunOptions{ nullptr },
                                    _input_names.data(),
                                    &input_tensor, 1,
                                    _output_names.data(), 2);

    // GET OUTPUT
    auto type_info = ort_outputs[0].GetTensorTypeAndShapeInfo();
    auto data_length = ort_outputs[0].GetStringTensorDataLength();
    std::string result(data_length, '\0');
    std::vector<size_t> offsets(type_info.GetElementCount());
    ort_outputs[0].GetStringTensorContent((void*)result.data(),
                                          data_length, offsets.data(), offsets.size());

    return std::stoi(result);
}
```

PMML

Treelite

Resources

- https://github.com/abhilb/pydata_2021