

# Adding an ML adapter to Mascot Server

Author: Ville Koskinen, Matrix Science Ltd

Date: 28 April 2025

## Example adapter

example\_ML\_adapter.py is an example script that performs the following steps:

1. Given a Mascot results file (--resfile), open it using Mascot Parser.
2. Read and parse the query list file (--query\_list), or exit if failure.
3. Iterate over the query list. Compute peptide length for each PSM.
4. Iterate over the query list again. Print the calculated values in the output file (--output).

The script implements the ML adapter protocol described in chapter 13 of the Installation & Setup manual (manual.pdf).

## Running the script

To run the script from the command line:

1. Install Python if not already installed. This document assumes Python is installed in C:\Python\Python311.
2. Copy example\_ML\_adapter.py to C:\inetpub\mascot\bin\ML\_adapters\custom, assuming Mascot Server is installed in C:\inetpub\mascot.
3. Download Mascot Parser from: [https://www.matrixscience.com/msparser\\_download.html](https://www.matrixscience.com/msparser_download.html)
4. Extract it in a suitable directory. The Python module is in python36\_or\_later.
5. Copy msparser.py and \_msparser.pyd from python36\_or\_later to C:\inetpub\mascot\bin\ML\_adapters\custom, adjacent to example\_ML\_adapte.rpy.
6. Run the script:

```
C:\n> cd \inetpub\mascot\bin\n> C:\python\python311\python.exe ML_adapters\custom\example_ML_adapter.py &\n--help
```

The script must be run from the mascot/bin directory, because it needs access to mascot/config/mascot.dat. However, the script itself does not need to be saved in mascot/bin. A convenient location is the bin\ML\_adapters\custom directory.

## Testing the script

To test the script, use one of the example searches shipped with Mascot.

1. Go to your local Mascot home page, Help, Example data sets shipped with Mascot. Open the example search “Yeast example (CPTAC study 6)”, or just enter the URL [http://localhost/mascot/cgi/master\\_results\\_2.pl?file=F981142.msr](http://localhost/mascot/cgi/master_results_2.pl?file=F981142.msr) if Mascot is running at localhost.
2. Expand a protein family and make a note of a PSM with a match. For example, query 244 rank 1 has a suitable match for testing.
3. Create a small text file query\_list.tsv with the following contents (note, this is a tab-separated value file):

```
query      rank  label
244        1    1
```

#### 4. Run the script

```
C:
cd \inetpub\mascot\bin
C:\python\python311\python.exe ML_adapters\custom\example_ML_adapter.py &
  --query_list query_list.tsv --resfile ..\data\F981142.msr &
  --output output.tsv
```

#### 5. Check that output.tsv contains the computed peptide length:

```
query      rank  label      peptideLength
244        1    1          7
```

## Adding the new ML adapter to the user interface

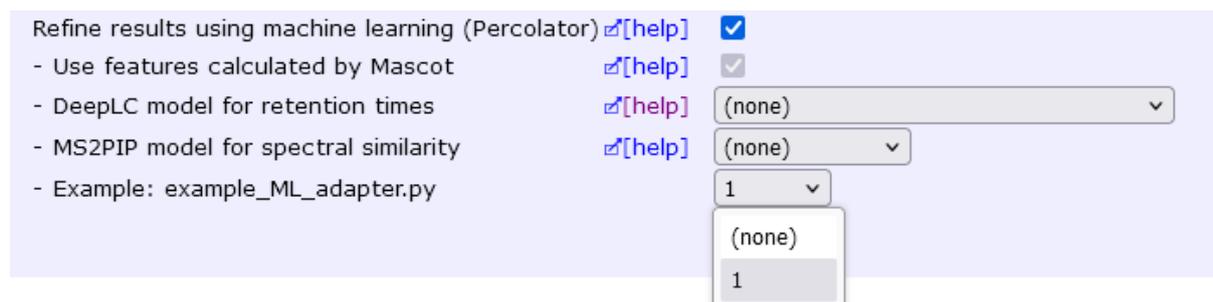
Edit mascot/config/ML\_adapters.toml. Follow the instructions in chapter 13 of the Installation & Setup manual to add the new adapter. Briefly, the following minimal configuration will work:

```
[example_ML_adapter]
interpreter = "C:/python/python311/python.exe"
program = "C:/inetpub/mascot/ML_adapters/custom/example_ML_adapter.py"
visible_in_user_interface = true

[[example_ML_adapter.parameters]]
name = "enabled"
title = "Example: example_ML_adapter.py"
values = [ 1 ]
```

This assumes you have saved the script in C:\inetpub\mascot\ML\_adapters\custom. Make sure you also copy `_msparger.pyd` and `msparger.py` to the same directory.

Open F981142.msr in Protein Family Summary on your local Mascot server. You should now have a new ML adapter option:



The parameter value in this case is not used by the script, but at least one parameter value must be defined to be able to run the script.

Choose '1' and Apply. If all goes well, you should now see the usual progress bars for machine learning, Percolator, and then caching the results.

Check the refiner log for any warnings or errors: In Protein Family Summary, view page source and scroll to the bottom. At the end of the HTML is a commented out section that lists the cache directory, for example:

Cache files used:

```
../data/cache/2024/08/grh7f4pamjdu727fjx2bgbgpiy/F981142.msr.bum5nzkv5qgycd  
cc5mlynvfsra.sdb  
../data/cache/2024/08/grh7f4pamjdu727fjx2bgbgpiy/F981142.msr.ehhhfkyefw32sj  
pb57xweb24nq.pip
```

The log for refining results is saved in this directory with a file extension .pop.log.

Confirm, for example, that Percolator picked up the new feature, which will appear as something like:

```
0.0991    -0.1680    0.3078    meanAbsFragPPM  
0.5915    0.8372    0.4770    rawScore  
0.2336    0.3526    0.1795    example_ML_adapter:peptideLength  
-1.9084   -1.5569   -1.3579    m0  
Found 2995 test set PSMs with q<0.01.
```

## Evaluating the feature in the ML quality report

In the Protein Family Summary report, open the machine learning quality report from the link provided:

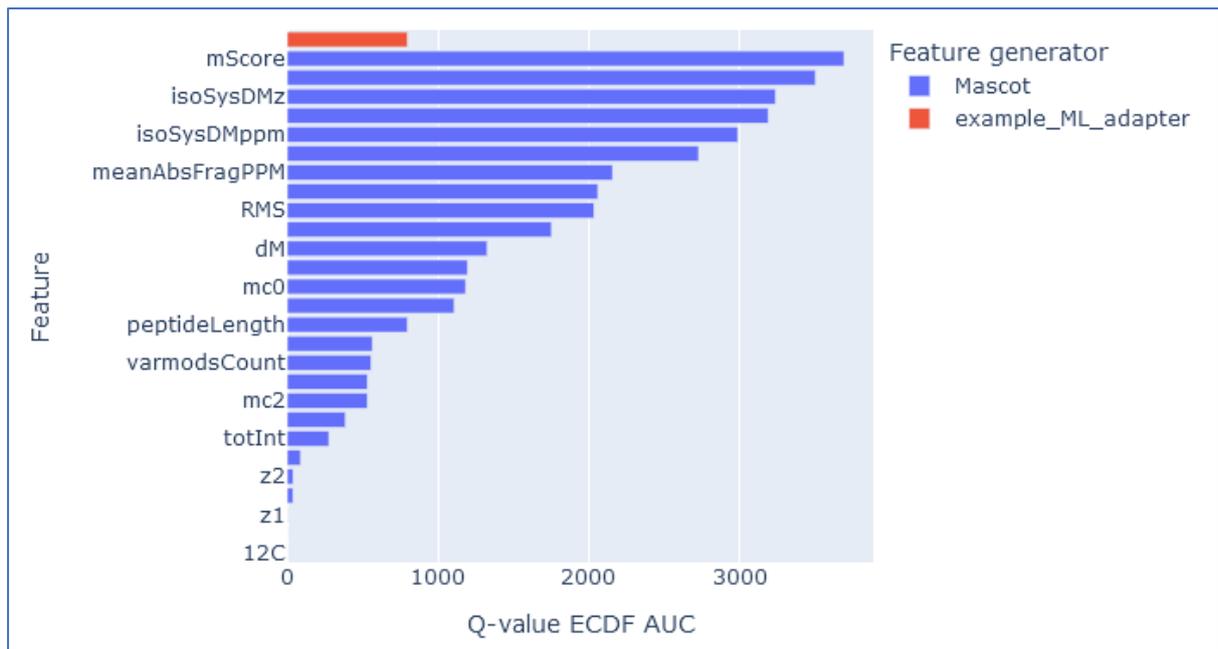
▼ **Sensitivity and FDR (reversed protein sequences)**

	<u>Target</u>	<u>Decoy</u>	<u>FDR</u>
Protein family members	771	25	3.24%
<input type="button" value="Sequences"/> above <input type="button" value="homology"/>	2808	28	1.00%

Details about ML model performance are available in [the machine learning quality report](#).

Open the Rescoring Features tab. This tab displays a graph showing the Q-value ECDF AUC of each active feature. The larger the AUC, the better the feature is at discriminating between target and decoy PSMs.

Mascot groups core features separately from any features computed by ML adapters:



peptideLength is a core feature computed by Mascot. There is no issue with using the same feature name in some other adapter, because Mascot prefixes the feature name with the adapter name before combining with core features. In this case, peptideLength in core features is identical to peptideLength computed by example\_ML\_adapter.py, so they have identical AUC.