This plugin it is based on a freeware utility (you can run it in the 'prompt' command line window - Windows users - or in the Linux 'Terminal' with 'wine' installed, or by compiling Linux executables) that you can find on the net: 'midicsv' which can turn a midi file into a text file (.csv) and vice versa.
Even this plugin, which uses the Musescore API only for its display, could be transformed (with some modifications) into an independent executable. Personally, i created a version with Delphi; everyone can rewrite it and compile it in their favorite language.

It is possible to 'drag' a midi file directly into the text box to have the plugin produce a '.bat' or '.sh' type file to transform it into a CSV file.
To achieve this, however, some parameters must be correctly set in the '.qml' file, in the lines where we find '// HERE THE CHANGE!' written.
Firstly the 'createBatch' property will need to be set to 'true'.
Secondly the 'usaWine' property will need to be set to 'true' if the Linux user plans to use executables through Wine.
Finally, depending on the OS used, it will be necessary to indicate the exact path where the executables are located and where the 'batch' files will be saved.

Once the midi file has been converted to csv, through the plugin it is possible:

First thing: need to copy (or, better, drag & drop the file CSV in the box text) the text of the csv file before carrying out the operations.

1) Button 'Apply': starts (via a context menu accessible with the right mouse click) 7 different actions.

___a) (default) CC (control change) to velocity: assign the velocity of the notes the value of a Control Change (typically the number 7, but it can be changed). Options: you can delete the CC values; it is possible to set the velocity to a fixed value (the CC value) or keep the previous value in proportion to the CC value (in this case the CC value is then set to 127). The plugin also offers the ability to assign the value to the 'note off' event, more for demonstration intent than for practical purposes: for those who are able to do so, the plugin could be customized for other types of replacement.
After pressing 'Apply', controls appear before activating the function to choose whether to apply the action to all tracks or to a single track, and whether to apply the action to all bars or to a chosen interval: it is also possible to delete all notes outside the chosen range.

___b) Add value to velocity: remove or add a fixed value to the velocity. Simply set the positive or negative value to add.
After pressing 'Apply', controls appear before activating the function to choose whether to apply the action to all tracks or to a single track, and whether to apply the action to all bars or to a chosen interval: it is also possible to delete all notes outside the chosen range.

___c) extract the lyrics. No option for this feature.

___d) extract the text. No option for this feature.

___e) change ticks for quarter. The midi time division of the uploaded file is displayed, and you can set it to a different value by clicking 'Apply' for the changes to take effect.

___f) add/subtract tick to pos(ition). Experimental, it allows you to add or remove 'ticks' to the position, to recover certain midi files with the notes slightly shifted with respect to the beat. Only use this feature if you know what you're doing!
After pressing 'Apply', controls appear before activating the function to choose whether to apply the action to all bars or to a chosen interval: it is also possible to delete all notes outside the chosen range. Apply on all tracks.

___g) consistency check. Check that the processing has not produced any unsuitable results: the events of each track must be strictly in chronological order. In the event of an error, a file in 'txt' format is saved with the list of anomalies.

2) Button 'Save File CSV': the changes appear in the text box; you can save your changes to a file with a default name and destination (if the file was loaded with drag & drop the name is obtained from the latter with different suffixes which signal the applied transformation), or by copying the contents of the text box, and then proceed to its retransformation into a MIDI file.

3) Button 'Stretch Midi': starts (via a context menu accessible with the right mouse click) 6 different actions.

___a) (default) 'Stretch Midi': lengthen or shorten the durations of the notes (and events) according to the percentages indicated, adapting the time signature to the desired conversion if necessary. It is possible, for example, by choosing 'Triplet' (66.6% periodic) to transform a 12/8 piece into a piece with C time with triplets; by choosing 150% it is possible to achieve the opposite. 25%, 50%, 200% are the most logical conversions, for the others (75% 125% 175%)... someone will be able to imagine a possible use. I left 100% as the function to delete all events other than notes (after activating the check box). You can also enter a percentage value that you decide: the format must have a number at the beginning, or simply write a number without a percentage. The function can be applied to notes only, or to all events (Default).
IMPORTANT: if the file was inserted without drag & drop, click the button 'Get Time Sign.' before to start the operations: it is used to set the time signature present in the midi file (only the first one is read); then you can edit it as you like.

___b) 'Quantize': quantization is applied to the positions of the note, not to its duration. For values, 'D' = 'Dotted', 'T' = 'Triplet'.
All of these transformations can be applied to notes only, or to all events (Default), and are adapted to the 'division' (ticks for quarter) of the MIDI file.
Through a context menu linked to the 'combobox' for setting the durations, it is possible to choose which type of quantization to apply, 'Mixed' (default) or 'Fixed' (for dotted values, only the 'Fixed' option will still be applied).
The 'Mixed' option considers the chosen value plus the immediately smallest value in the 'regular note-triplet note' hierarchy. For example, if you choose the regular 1/8 value, you will also look for the presence of 1/8 triplet notes; and if you choose the value 1/8 triplet you will also look for the notes at 1/16 regular.
The 'Fixed' option only considers the chosen value; The precision is greater but clearly this choice will be optimal only for those songs that contain only regular values or only triplet values. The choice must therefore be made considering the nature of the piece, and remembering that it is possible to apply quantization even for fragments.
A particular version of 'quantization' is available: straight to swing and vice versa. Through a context menu (located on the check box) you can decide which method to apply (simple or complex). IMPORTANT: before applying this function you must apply the 'Legato' with the 'fixed note channel' option activated, and the midi must be perfectly quantized.
The simple method requires you to have notes all of the same value, e.g. all eighths (1/8T quantization value). From straight to swing there can also be triplets of eighth notes, in the reverse process only triplets formed by 1/4 + 1/8 must be present (1/8 quantization value).
The complex method (available only from straight to swing) allows you to have even the notes with the immediately smallest value (chosen 1/8T as the quantization value, there can also be notes from 1/16), which the algorithm will try to manage. This is a rather rudimentary function, which will only be useful in the simplest cases.
After pressing 'Quantize', controls appear before activating the function to choose whether to apply the action to all tracks or to a single track, and whether to apply the action to all bars or to a chosen interval: it is also possible to delete all notes outside the chosen range.

___c) 'Legato': notes (or groups of notes on the same 'tick') end when the next note (or group of notes) begins. I have limited the maximum length to 4/4, but it can be changed by changing the value of 'numeroDiQuarti' in the qml file. This function can only be applied to one track at a time, selecting it appropriately. In general, before applying this function, it is better to clean the midi from events that aren’t notes.
To have the notes assigned to a single voice when importing the file into Musescore, activate the 'fixed note channel' option.

___d) 'Middle C': useful for pianists, or to separate a single staff into two voices (even if it is Musescore that decides the assignment of the voices); established a reference pitch (which can be changed in the control that appears) the notes will be assigned to two different voices (or staves) when importing the file into Musescore. This function can only be applied to one track at a time.
The tracks are automatically read if the file is already loaded in the text box, otherwise you have to reactivate with the context menu to make sure that they are read. If the midi file has only one track, the control remains off.
After pressing 'Middle C', controls appear before activating the function to choose whether to apply the action to all bars or to a chosen interval: it is also possible to delete all notes outside the chosen range.

___e) 'Export single channel': useful for MIDI type 0 files; extracts a channel as a separate track. The file can only be uploaded by dragging and dropping the button itself and is not displayed in the text box. Once loaded you can choose which channel to extract. The order is the same as you can see in Musescore.

___f) 'Export single track': useful for the other MIDI type files; extracts a track as a separate track. The file can only be uploaded by dragging and dropping the button itself and is not displayed in the text box. Once loaded you can choose which track to extract. The order is the same as you can see in Musescore.

For advanced users, it is also possible to indicate (via the tick position) the start and end point of the bars range very precisely. The indication could be StartBar=1 at tick 0, EndBar=4 at tick 1920: the tick value relative to StartBar can only be increased, that of EndBar decreased, to narrow the selected range.

The '0' and '1' buttons are used to transform the midi file from one type to another. It is possible to choose which tracks (or which channels) to include and exclude. In the transformation from type 1 to type 0 it is also possible to choose whether to keep all the events, or ('fix channel to 1' enabled, checked or not) transfer only the 'note' events. Even in the case of a complete conversion, the focus was only on what Musescore needs. If you want a probably more precise conversion i recommend 'GNMIDI 1 Freeware'.

The larger the file, the slower the operations will be!

In reality, the executable made with Delphi does not have the problems of the plugin (the difference is enormous: a few seconds versus even 2/3 minutes), and support of 'ShellExecute' function allows you to perform all operations from the program window.
Execution times are much faster, and it doesn't crash if you upload files of a certain size. What does a certain size mean? A 100/130 KB midi file becomes a CSV file of approximately 1 mB; the plugin is already in danger of crashing at this size, which is why i implemented channel and track extraction functions without displaying them in the text box. Despite this, unusually long processing times remain.

If it happens that you load a file and the controls do not show the correct values, it is worth reselecting the same function and everything should be fine.

This plugin is better suited for type 1 midi files rather than type 0 ones. Loading a type 0 midi file in the text box produces a warning message in the status bar. Musescore and other freeware editors (like GNMIDI 1 Freeware) can perform the conversion (and Midicsv's package also contains an algorithm - those who have experience programming in 'Perl' could try to create others - to extract individual channels as separate files). However, if the action is applied to the entire file, in some cases there is no problem with type 0 midi.
Functions that move the position of notes (especially 'Legato') can sometimes generate some small errors during the retransformation from csv to midi; they are reported and can be corrected manually.

Usage tips: upload the midi file to musescore and see what can be improved, then make the changes. If the score is already correct, obviously no intervention will be needed, but this rarely happens. In the case of files with many tracks and of a certain size, it is recommended to export and edit one track at a time, otherwise the plugin may crash the application (especially version 4).
Everyone can try to imagine what this plugin can be used for: with the 'Legato' function, for example, it is possible to reduce midi files displayed with absurd polyphonies to a single voice, so as to be able to reassign the voices in a more rational way.
A good procedure could be to load the original midi file into Musescore, think about it, and start modifying it one 'block' at a time, copying the changes into another window.
From personal experience, it is always better to copy the midi into another score (a previously prepared 'template') rather than working on it directly.
Of course, you can also do some manual editing if you're sure of what you're doing.
Without understanding, at least at a basic level, what the structure of a midi file is, this plugin will hardly be useful.

Appendix: The Mysteries of Musescore.
I happened to quantize a fragment of midi that contained triplets of sixteenth notes: i verified that the plugin had done its job perfectly, but Musescore (regardless of the import settings) continued to display absurd durations (Sibelius, on the other hand, imported it perfectly). I then tried to double the duration of the midi (200%), so as to have eighth-note triplets: with the import setting on the minimum duration of a sixteenth, the midi was perfectly imported. Then i reset the durations to their original values with Musescore's functions. Evidently the import of midis has some problems with tuplets: Musescore 4, for example, as the only import option, allows you to change the value of the minimum duration. However, knowing this, in all cases where there are triplets that are too 'small' this expedient can be used. Musescore 3, on the other hand, has many options for editing the uploaded midi file: in this case, it would have been sufficient to allow only triplets among the tuplets to solve the problem.

If we are working on an imported midi file, i remember the recommendation to copy it to another blank score created by MuseScore: otherwise unwanted (and inexplicable, such as the deletion of notes not related to the operations) could occur. Alternatively, save the midi as 'mscz', close the score, and then open it again.

Note on 'drag & drop': tested on Windows OS and LinuxMint. For other OS there may be problems with the interpretation of the overall path: in this case you can try modifying the 'startNameFile' variable in the Qml file. For the tested OS the plugin sets itself automatically.
By dragging and dropping a midi file into the text box, the plugin automatically generates the '.sh' or '.bat' file for transformation into CSV. When you save the processed CSV file, the batch file for MIDI transformation is generated automatically.
Note: To have native Linux executables, you must have the GCC compiler or equivalent installed.