# Contribution Guidelines

Contributions are welcomed! Here's a few things to know:

\* [Steps to Contributing](#steps-to-contributing)

\* [Coding Guidelines](#recommenders-team-contribution-guidelines)

\* [Microsoft Contributor License Agreement](#microsoft-contributor-license-agreement)

\* [Code of Conduct](#code-of-conduct)

## Steps to Contributing

\*\*TL;DR for contributing: We use the staging branch to land all new features and fixes. To make a contribution, please create a branch from staging, make a modification in the code and create a PR to staging.\*\*

Here are the basic steps to get started with your first contribution. Please reach out with any questions.

1. Use [open issues](https://github.com/Microsoft/Recommenders/issues) to discuss the proposed changes. Create an issue describing changes if necessary to collect feedback. Also, please use provided labels to tag issues so everyone can easily sort issues of interest.

2. [Fork the repo](https://help.github.com/articles/fork-a-repo/) so you can make and test local changes.

3. Create a new branch \*\*from staging branch\*\* for the issue (please do not create a branch from master). We suggest prefixing the branch with your username and then a descriptive title: (e.g. gramhagen/update\_contributing\_docs)

4. Create a test that replicates the issue.

5. Make code changes.

6. Ensure unit tests pass and code style / formatting is consistent (see [wiki](https://github.com/Microsoft/Recommenders/wiki/Coding-Guidelines#python-and-docstrings-style) for more details).

7. Create a pull request against \*\*staging\*\* branch.

Once the features included in a [milestone](https://github.com/microsoft/recommenders/milestones) are completed, we will merge staging into master. See the wiki for more detail about our [merge strategy](https://github.com/Microsoft/Recommenders/wiki/Strategy-to-merge-the-code-to-master-branch).

## Coding Guidelines

We strive to maintain high quality code to make the utilities in the repository easy to understand, use, and extend. We also work hard to maintain a friendly and constructive environment. We've found that having clear expectations on the development process and consistent style helps to ensure everyone can contribute and collaborate effectively.

Please review the [coding guidelines](https://github.com/Microsoft/Recommenders/wiki/Coding-Guidelines) wiki page to see more details about the expectations for development approach and style.

## Microsoft Contributor License Agreement

Most contributions require you to agree to a Contributor License Agreement (CLA) declaring that you have the right to, and actually do, grant us the rights to use your contribution. For details, visit https://cla.microsoft.com.

When you submit a pull request, a CLA-bot will automatically determine whether you need to provide a CLA and decorate the PR appropriately (e.g., label, comment). Simply follow the instructions provided by the bot. You will only need to do this once across all repos using our CLA.

## Code of Conduct

This project has adopted the [Microsoft Open Source Code of Conduct](https://opensource.microsoft.com/codeofconduct/).

For more information see the [Code of Conduct FAQ](https://opensource.microsoft.com/codeofconduct/faq/) or contact [opencode@microsoft.com](mailto:opencode@microsoft.com) with any additional questions or comments.

Apart from the official Code of Conduct developed by Microsoft, in the Recommenders team we adopt the following behaviors, to ensure a great working environment:

#### Do not point fingers

Let’s be constructive.

<details>

<summary><em>Click here to see some examples</em></summary>

"This method is missing docstrings" instead of "YOU forgot to put docstrings".

</details>

#### Provide code feedback based on evidence

When making code reviews, try to support your ideas based on evidence (papers, library documentation, stackoverflow, etc) rather than your personal preferences.

<details>

<summary><em>Click here to see some examples</em></summary>

"When reviewing this code, I saw that the Python implementation the metrics are based on classes, however, [scikit-learn](https://scikit-learn.org/stable/modules/classes.html#sklearn-metrics-metrics) and [tensorflow](https://www.tensorflow.org/api\_docs/python/tf/metrics) use functions. We should follow the standard in the industry."

</details>

#### Ask questions do not give answers

Try to be empathic.

<details>

<summary><em>Click here to see some examples</em></summary>

\* Would it make more sense if ...?

\* Have you considered this ... ?

</details>