# Data access

When we are versioning data and models with DVC and storing them on a remote, sooner or later we will want to access those artifacts again. Common use cases include:

- Downloading the latest version of a model
- Downloading a specific version of a model
- Reusing datasets across different projects
- Getting the features used to train a specific version of a model

When connecting with a remote, we may not immediately know what is stored there. Using `dvc list` in combination with the URL to our remote and the name of the dataset we want to access, we can get an overview of all tracked artifacts:

```
# run command
dvc list https://github.com/iterative/dataset-registry use-cases

# output
.gitignore
cats-dogs
cats-dogs.dvc
```

We can then use one of two commands to download said data: `dvc get` and `dvc import`. The first command simply downloads the data from your remote to your current directory. The latter one does so as well, but also adds the data under DVC control. This means that any changes you make to the data will be tracked.

Say we have a remote with `dataset.csv`, for example. If we used `dvc get`, DVC will download `dataset.csv` to our local machine. In this case, DVC will not keep track of whether we change the contents of `dataset.csv`. If we want to change those contents and upload the updated file back to the remote, we need to use `dvc import` instead.

These functions serve slightly different purposes. Say we are updating an existing model. In that case, we would probably use `dvc import` so that we can upload the updated model to the remote. If we are accessing the remote to serve the model instead, we would rather use `dvc get`, because the server won't be making any changes to the model itself.