Assignment lab 2 - Regression models trials and metrics

The purpose of this assignment

The goal of this lab is to check that the student has knowledge in the following topics:

- Multiple linear regression.
- Polynomial regression.
- Support vector regression.
- Decision tree regression.
- Random forest regression.

Example dataset Boston house prices toy dataset is used in this assignment. Dataset does not require preprocessing and it has no categorical features. Downside of using example dataset is that different regression algorithms metrics do not differ remarkably.

Task

1. Install Python Anaconda distribution (or Python with required modules) if it was not installed before.
2. Create software project in GitLab. Use one of to https://gitlab.cs.ttu.ee or https://gitlab.com. See class 1 material for details.
3. Print out Python and available modules versions.
4. Create matrix of features(X) and independent variable(y).
5. Split dataset into training and tests sets with size 0.25. Create linear regressor. Train regressor with training data. Print out R squared with function print_metrics.
6. Use backward elimination for feature selection. Create training and test sets with selected features. Call linear regression function created in step 5 with selected features. **See guidelines for details**.
7. Create polynomial regressor with degree 2. Transform features with polynomial regressor. Call linear regression function created in step 5 with transferred features. **See guidelines for details**.
8. Scale features and dependent variable with Standard scaler. Split dataset into training and tests sets with size 0.25. Create SVR regressor with rbf kernel and auto gamma. Train regressor with training set. Print out R squared with function print_metrics. **See guidelines for details**.
9. Split dataset into training and tests sets with size 0.25. Create decision tree regressor. Train regressor with training set. Print out R squared with function print_metrics.
10. Split dataset into training and tests sets with size 0.25. Create random forest regressor with 10 estimators. Train regressor with training set. Print out R squared with function print_metrics.

Guidelines

Project repository structure and files

Project shall consist of following files (excluding directories `.git` and also `builds` if local gitlab-runner is used).

```
.
├── .gitignore
├── .gitlab-ci.yml
├── .pylintrc
├── common
|   ├── feature_selection.py
|   └── test_env.py
├── lab2.py
```

`lab2.py` shall be created by student.

For `.gitignore`, `.gitlab-ci.yml`, `pylintrc`, and `common` files from lab2 template shall be used.

`lab2.py` template is available via snippet Lab 2 Regression models trials and metrics template to avoid merge conflicts when working with multiple remotes.

NB! Be aware that if you want to use different file names you need to modify CI configuration and tests accordingly.

Dataset

There is no need to read dataset from file. Following can be used to create matrix of features and dependent variable:

```
from sklearn.datasets import load_boston
# https://scikit-learn.org/stable/datasets/index.html#boston-house-prices-dataset
X, y = load_boston(return_X_y=True)
```

Program composition

It can be feasible to divide your program into functions so that every regression model has its own function. Matrix of features and independent variable shall be passed to those functions as arguments. If you want reuse particular regressor (for example with new data), function should return regressor object. As linear regressor is used in several models, linear "all in" function can be reused by linear regression with selected features and polynomial regression. With such approach you will also avoid confusion with modifying global variables.

lab2 template contains prepared functions for linear regression all in and linear regression with selected features as and example. **Student shall extend linear regression with selected features with feature selection.**

Feature selection

In `common/feature_selection.py` there is function `backward_elimination(X, y, p_threshold=0.05, verbose=False)` what can be used for feature selection. Usage example:

```
import common.feature_selection as feat_sel
X_sel = feat_sel.backward_elimination(X, y)
```

Polynomial regression

As polynomial regression uses linear regressor function `linear_regression()` from lab2 template can be used in same way as it is for linear regression with feature selection.

Scaling for SVR

Both features and dependent variable shall be scaled. It can be done before test-train split. As in assignment there is no need to inverse results, same scaler object can be reused. **NB! pay attention how dependent variable dimensions shall be handled with np.expand_dims() and np.squeeze().**

```
sc = StandardScaler()
X = sc.fit_transform(X)
y = sc.fit_transform(np.expand_dims(y, axis=1))

# STUDENT SHALL ADD TEST-TRAIN SPLIT AND REGRESSOR CREATION AND TRAINING HERE

print_metrics(np.squeeze(y_test), np.squeeze(reg.predict(X_test)), 'SVR')
```

Printouts example

Printouts example is available in snippet Lab 2 Regression models trials and metrics printout example.

Feel free to customise and change your printouts, but print out each model R squared.

Automation and GitLab CI stages

- Check-files
  - Tests existence of required files and fail if all files are not present.
  - List repository files excluding `.git` and `build` directories.
- Lint
  - Test `lab2.py` formatting with `pep8`.
  - Lint `lab2.py` with `pylint` by using configuration from file `.pylintrc`.
- Run-lab
  - Run `lab2.rb`

Content of results directory is archived as build artefacts and can be downloaded.

Formatting and lint

autopep8 is used to test code formatting. autopep8 is supported by VS Code. For other editors it can be installed with conda:

```
$ conda install -c conda-forge autopep8
```

To run formatter from command line:

```
$ autopep8 --in-place lab2.py
```

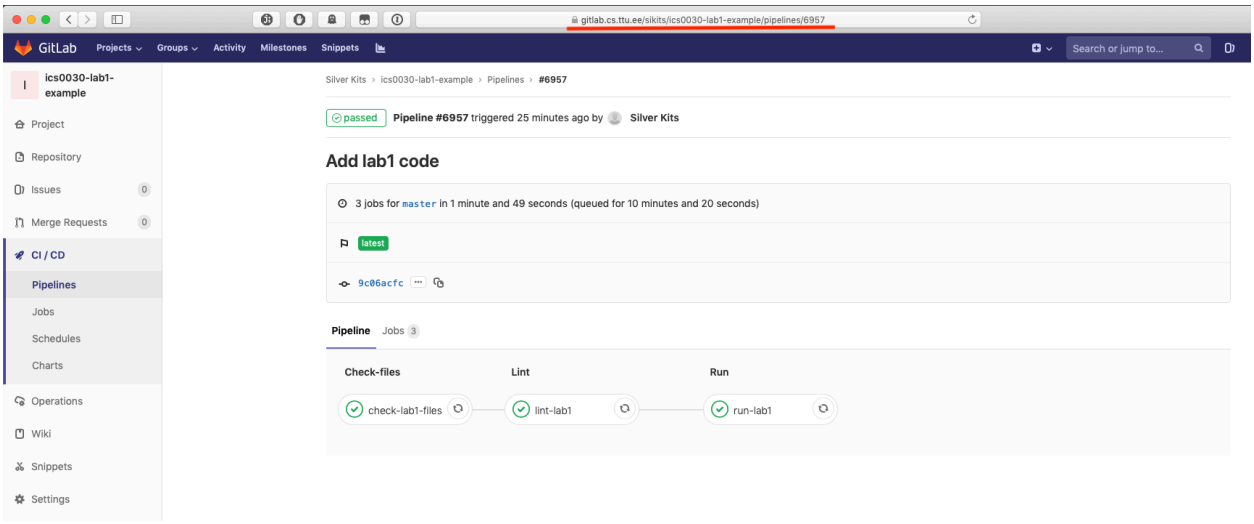
pylint is used for lint. Project template contains `.pylintrc`. Settings in this file are inline with VSCode default settings.

To run pylint from command line:

```
$ pylint lab2.py
```

Submission instructions

1. Be sure that your pipeline succeeds before submitting assignment in Moodle.



2. Submit only link to the pipeline as an answer in Moodle. **Please make link HTML URL!**