Assignment lab 3 - Dataset preprocessing, classification models trials and results analysis

The purpose of this assignment

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

- Dataset preprocessing.
- Logistic regression.
- K-nearest neighbours.
- Support vector machine classification.
- Naive Bayes.
- Decision tree classification.
- Random forest classification.

Business problem description

There is need to develop machine learning model to predict is student likely to not continue studies after 4th semester based on collected data.

Goal is to predict students not likely to continue studies. This introduces constraints with model "accuracy" because most students will continue studies and there can be additional not known factors contributing to decision not to continue. Goal can be achieved to find out model with minimum amount of false positives. This can make model too pessimistic and most likely to predict many false negatives.

Dataset description

BIG FAT WARNING. Dataset is not public and shall be not used outside of ICS0030 Machine learning course context.

Dataset contains anonymised data is available for model training and test. Dataset is extract from real data, but does not correspond to full real data because several records are removed.

Original dataset is in Microsoft Excel format (.xlsx). Dataset contains both nominal categorical and continuous numeric features. Missing continuous numeric values shall be replaced with 0. Missing categorical values shall be replaced with `Missing` category.

There is real world constraint how different exam points are considered. Current approach is simplified and relies to assumption that all exams have equal weight and missing exam is 0 points. Alternative would be encode exam types and have different weights for different exam points. However practical trials showed that there is no real difference in models accuracy.

Features:

- Faculty
- Paid tuition
- Study load
- Previous school level
- Previous school study language
- Recognition
- Estonian language exam points
- Estonian as second language exam points
- Mother tongue exam points
- Narrow mathematics exam points
- Wide mathematics exam points
- Mathematics exam points
- Study language
- Foreign student

Dependent variable:

- In university after 4 semesters

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. Read dataset file to pandas data frame.
5. Save dataset description and categorical features to file in results directory. **See guidelines below.**
6. Filter out students not continuing studies (In university after 4 semesters is No) and save filtered description and categorical features to file. **See guidelines below.**
7. By comparing and studying results saved in steps 4 and 5 **try to get familiar with data**.
8. Preprocess data by encoding categorical features and by replacing missing numeric exam points with 0. **See guidelines below.**
9. Create logistic regressor. Split test and training data (0.25 is default). Scale features. Train regressor. Predict test set results and print out metrics. **See recommended classifier parameters in guidelines below.**
10. Create K-nn classifier. Split test and training data (0.25 is default). Scale features. Train classifier. Predict test set results and print out metrics. **See recommended classifier parameters in guidelines below.**
11. Create SVM classifier - SVC. Split test and training data (0.25 is default). Scale features. Train classifier. Predict test set results and print out metrics. **See recommended classifier parameters in guidelines below.**
12. Create Naive Bayes classifier. Split test and training data (0.25 is default). Scale features. Train classifier. Predict test set results and print out metrics. **See recommended classifier parameters in guidelines below.**
13. Create decision tree classifier. Split test and training data (0.25 is default). Scale features. Train classifier. Predict test set results and print out metrics. **See recommended classifier parameters in guidelines below.**
14. Create Random forest classifier. Split test and training data (0.25 is default). Scale features. Train classifier. Predict test set results and print out metrics. **See recommended classifier parameters in guidelines below.**
15. Select most suitable classifier to solve business problem and submit your selection with rationale as answer in Moodle in addition to pipeline link.

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
    ├── classification_metrics.py
|   ├── describe_data.py
|   └── test_env.py
├── data
|   └── students.xlsx
├── lab3.py
└── results
    └── .placeholder
```

`lab3.py` shall be created by student.

For `.gitignore`, `.gitlab-ci.yml`, `pylintrc`, `data` and `common` files from lab3 template shall be used.

`lab3.py` template is available via snippet Lab 3 Dataset preprocessing and classification models trials and results analysis 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 read to pandas data frame

Project template contains dataset `students.xlsx`. Dataset description is above. Data can be read by calling function `read_data()` and providing file name as an argument:

```
df = read_data('data/students.xlsx')
```

Be aware if you are using your own dataset you need to most likely change, separator, delimiter and encoding.

Dataset overview and categorical data save and print

You can print out dataset overview by calling function `print_overview()` in module `common.describe_data` by providing data frame and optional file name with path as arguments. Same for categorical data with function `print_categorical()`. Module shall be "imported" first. Add imports to beginning of the file!

```
import common.describe_data as describe_data

...
describe_data.print_overview(df, file='results/students_overview.txt')
describe_data.print_categorical(df,
                                file='results/students_categorical_data.txt')
```

If you want to print results only to standard output(stdout) leave out optional argument.

```
describe_data.print_overview(df)
describe_data.print_categorical(df)
```

Dataset preprocessing

Extend function `preprocess_data()` with following functionality.

1. Encode all categorical features with pandas get_dummies.

   Variable `categorical_columns` holds list with all categorical columns as pandas data frame column labels. See class 3 example `preprocessing_dataframe.py` for details.

   You need to replace missing categorical values with common label. For example with "Missing". Otherwise you will not correct amount of feature columns. For example within for loop:

   ```
   df[column] = df[column].fillna(value='Missing')
   ```

   NB! Do not forget to drop first column! You can extract categorical features encoding to function. In that case function shall have two arguments data frame and list with categorical columns and return encoded data frame.

2. Replace rest of missing values (missing exam points) with 0.

Feature selection

You can remove features by adding features as pandas data frame column labels to `drop_columns` list. Currently there is no easy automated way to select features as it was for regression models. You can study dataset with help of previously created statistics and leave features out if needed.

NB! Printout example is with all features

Classifier tuning

You could try with following non-default classifiers arguments to improve accuracy. **Be aware that in some cases changing arguments can make accuracy worse.**

1. Logistic regression.

   You should use other solvers than `liblinear` and multinomial multi_class.

   With some solvers you can get warning `ConvergenceWarning: The max_iter was reached which means the coef_ did not converge`.

2. K-nearest neighbours.

   Try to increase number of neighbours, but not too much.

3. SVC

   Try to use sigmoid kernel with small gamma, bigger tolerances and with probability estimates.

4. Naive Bayes

   Instead GaussianNB use MultinomialNB or ComplementNB.

5. Decision tree classifier

   Default parameters can be used.

6. Random forest classifier

   Try with more than 10 estimators.

Classifier metrics printouts

For each model you need to print out confusion matrix with function `print_metrics()`. You need to provide `y_true`, `y_pred` and `label` (classifier name) as arguments.

There is optional argument `verbose`. When it is set to `True` classification report and accuracy score is printed in addition.

Printouts example

Printouts example is available in snippet Lab 3 Dataset preprocessing and classification models trials and results analysis printout example.

Feel free to customise and change your printouts.

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 `lab3.py` formatting with `pep8`.
  - Lint `lab3.py` with `pylint` by using configuration from file `.pylintrc`.
- Run-lab
  - Run `lab3.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 lab3.py
```

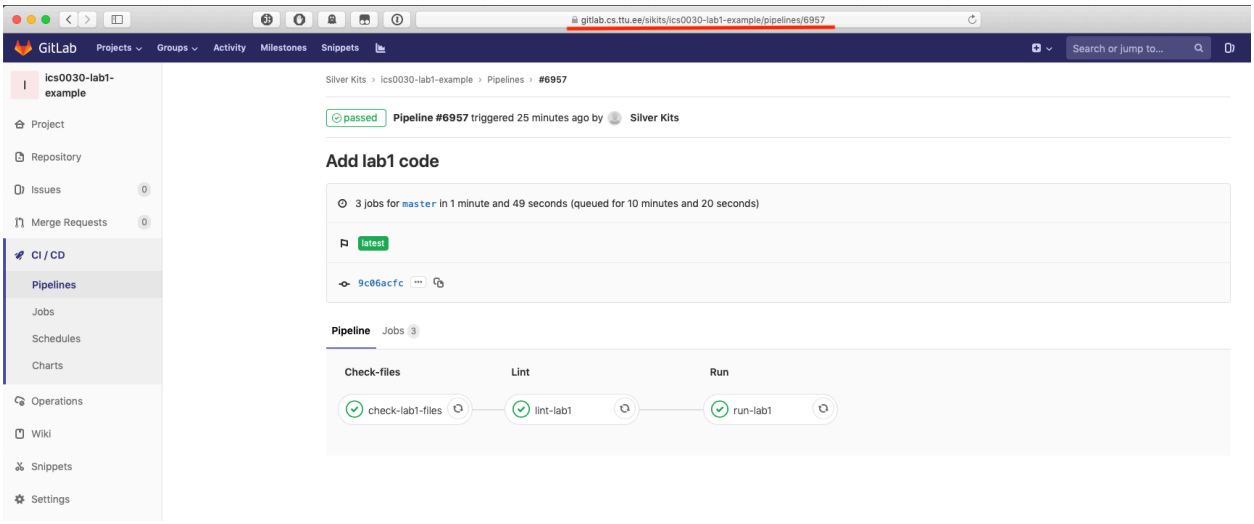
pylint is used for lint. Project template contains `.pylintrc`. Settings in this file are inline with VSCode default settings.

To run pylint from command line:

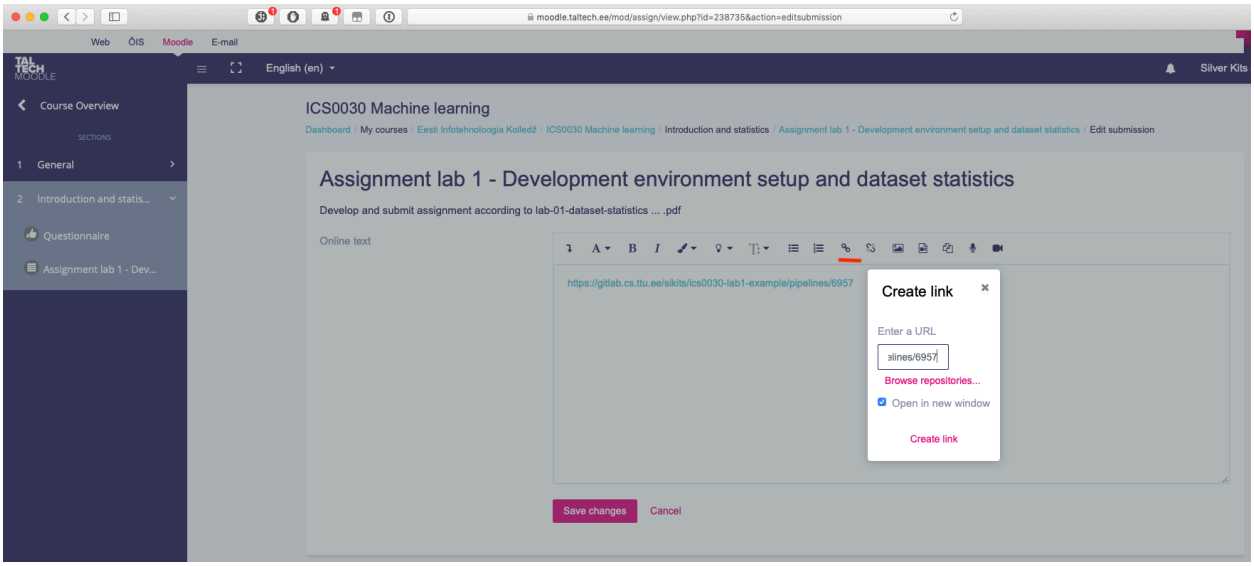
```
$ pylint lab3.py
```

Submission instructions

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



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



3. Submit written answer with rationale for most suitable algorithm to solve business problem as an answer in Moodle.