Assignment 2: Data Analytics

You have to use the Machine Learning platform (1) WEKA or (2) Scikit-Learn or (3) Spark / MLlib 8locally or on cluster)according to your preferences, to conduct the **Data Mining and Machine Learning tasks** below.

Preliminaries

- Information on how to obtain and use **WEKA** is available in the WEKA tutorial slides available on TUWEL.
- For Scikit-Learn you may consult the tutorial available at https://scikit-learn.org/stable/tutorial/index.html
- For **Spark/MLlib** you can use a local installation. If you want to operate on larger volumes of data or use the more powerful resources provided by the cluster rather than your local machines, we can also provide you with a group account on the cluster via which you can log in to Jupyter Hub at https://lbd.zserv.tuwien.ac.at:8000. Create a notebook using the PySpark3 kernel. When using the cluster, make sure to use the functions provided by Spark and MLlib to not block resources on the login node. Alternatively, you can setup Spark at your local machine and perform experiments offline. (Note: we can only provide limited support for the Spark/MLib platform. Please see the slides provided on TUWEL for an introduction to Spark/Mlib and consult the manual available at https://spark.apache.org/docs/latest/ml-guide.html)
- Structure your **report** for this assignment based on the structure in this assignment paper**.** For the experiments, provide detailed documentation of all steps to **ensure reproducibility** of all results based on the information provided.

A) Dataset

(1) **Select a data set** from the OpenML Machine Learning Repository (http://www.openml.org), Kaggle (https://www.kaggle.com/datasets), or a similar benchmark data repository with the *following requirements:*

- minimum 2.000 instances,
- minimum 20 attributes,
- minimum 4 class labels,
- not an “artificial” dataset, i.e., a dataset consisting of synthesized, sampled or interpolated values (e.g. the BNG* datasets on OpenML).

(2) ***Register* the dataset** you picked with your group number in the TUWEL Wiki. You must make sure that your dataset is unique, i.e. no two groups may take the same data set! (first come, first serve - do it early to get a data set that you also find interesting to work with.) ***Analyze* the characteristics** of the dataset (size, attribute types and semantics as discussed in class, value ranges, sparsity, min/max values, outliers, missing values, ...), and describe this in the report.

B) Classification: Analysis of Train/Test Set Splits, Performance and Parameters

(1) **Select two algorithms or two platforms:** *Pick two significantly different classification algorithms,* i.e. NO two variations of the same algorithm. Describe why

you chose the respective algorithms and briefly summarize their characteristics and the semantics underlying its parameters.

Alternatively, you can choose to *use only one algorithm*, but perform identical experiments *on two different platforms, i.e. any pairing of Scikit-learn, Spark/MLlib, WEKA*, comparing the results!).

(2) **Preprocessing**: Get the data into the form needed for training your two algorithms. Describe your preprocessing steps (e.g. transcoding, scaling), *why* you did it and *how* you did it.

(3) **Subsampling**: If the entire dataset is too large to be processed in its entirety, choose a subsampling strategy to get the dataset to a manageable size. Describe in your report why and how you did it. Make sure your experiment is repeatable. (No manual selection of instances, everything must be in code.)

(4) **Training & Testing:** Train your two algorithms in 3 separate experiment tracks as detailed below and evaluate your results with a reasonable quality measure for your algorithms (e.g.: (micro/macro) Precision/Recall, Mean Absolute Error,…). Interpret your results using both graphs and summaries (e.g. confusion matrices). For each of the 3 experiment tracks you should separately vary and document:

a. *Parameters:* If the classifier has specific parameters, explore their effect with different settings using 10-fold cross-validation and document the parameters and the results and analyze the sensitivity of classification outcomes against these parameters. Specifically, test extreme/obviously wrong settings and analyze the results.

b. *Scaling:* where possible, try different scaling approaches (min/max, zero mean/unit variance, length) using the best parameters identified above and observe the difference in classification performance using 10-fold cross-validation. Analyze the reasons for the effects observed, test useful and also non-useful (!) scalings and summarize your findings as well as analyze reasons why specific scalings make sense in a given setting

c. *Training / test set splits:* Use the best parameter setting and scaling identified above and evaluate the effect of different training and test set splits. Start with a small training set and increase it in small increments (e.g. 10 sets from 5% / 95% (train/test) in 10%-increments to 95%/5% (train/test) and observe performance changes. Perform multiple runs with each training set size to observe the sensitivity to the actual subset used for training a specific run. Analyze the variance in performance obtained

Always describe and summarize the findings intellectually (DO NOT simply copy the output into the report). Summarize the results while **focusing on the following questions**:

a. What trends do you observe in each set of experiments?

b. How easy was it to interpret the algorithm and its performance?

c. Which classes are most frequently mixed-up? (and why?)

d. What parameter settings cause performance changes?

e. Do both algorithms (or two system environments, i.e. any pairing of Spark/MLlib vs WEKA vs Scikit-Learn, if you chose to work on one algorithm

only) show the same behavior in performance, performance degradation / robustness against

i. smaller and larger training set sizes?

ii. variations in parameter settings?

f. Did you observe or can you force and document characteristics such as over-learning?

g. How does the performance change with different amounts of training data being available? What are the best scalings (per attribute / per vector) and why?

C) Missing Values:

(1) Write a little program/script to create new versions of the dataset by replacing x% of selected attributes by missing values (e.g. null or NaN in Spark or a "?" in WEKA). Missing values should be distributable

a. randomly across attributes

b. with specific percentages of missing values per attribute.
(Note: keep the initialization parameter of the random number generator as a parameter to the tool to ensure reproducibility). Describe the script, configuration parameters and settings in the report.

(2) Generate 6 different datasets and describe them in your report, varying

a. a small and large fraction of missing values *in an* attribute that has high / low information gain given the classification task) (4 data sets)

b. a small / large fraction of missing values randomly distributed *across all attributes* (2 data sets)

(3) Implement different strategies to deal with these missing values and describe their implementation, by

a. *ignoring* the respective attributes completely in the dataset

b. *replacing* the missing attribute values by the *mean / median* value of that *attribute* in the entire dataset

c. *replacing* the missing attribute by the *mean / median* value of that *attribute* in the respective class

(4) Pick one of the two classifiers from part B above and train it with training sets created by the different strategies to deal with missing values (using 10-fold cross-validation, best parameter setting from the experiments above). Document experiment settings and summarize the results in the report. Analyze the effect of increasing percentages of missing values in single attributes and across several attributes. For the same amount of missing values, does the classifier performance degrade identically, irrespective of which attribute these missing values occur in? Analyze the behavior according to questions such as

a. Do missing values in some attributes cause more damage than in others? If so, why? Why not?

   b. How do the different replacement strategies work? Which ones have the most positive effect on classifier performance?
   c. How do the strategies degrade with increasing fractions of values of a specific attribute missing?

D) Summarize your findings

1. Summarize your overall findings and lessons learned
2. Comparison with State of the Art**:** Find other works (scientific papers, scores on challenge leaderboards) that have evaluated their approach on the same data set you have chosen and see how your results compare to their numbers. Compare to the best performing algorithms (”state of the art”) and to a simple algorithm, such as an algorithm that always predicts the most frequent class (“baseline”) and estimate your performance in relation. Give short explanations of these algorithms if necessary. Analyze—or give your intuition—why the state of the art is performing better than your algorithm (or even more interesting: why your algorithm outperforms the state of the art!).
3. (**optional**) Provide feedback on the exercise in general: which parts were useful / less useful; which other kind of experiment would have been interesting, … (this section is, obviously, optional and will not be considered for grading. You may also decide to provide that kind of feedback anonymously via the feedback mechanism in TISS – in any case we would appreciate learning about it to adjust the exercises for next year.)

Submission guidelines:

- **Upload ONE [zip/tgz/rar] file** to TUWEL that **contains all your files** (all notebooks/scripts/programs you wrote and subsidiary information for repeating experiments) with the **report as a PDF file inside that zip file (no Word files, no TEX sources).** You must follow this naming convention:
  - `BI2019_A2_group_<groupno>_<Matnr.1>_<Matnr.2>_<Matnr.3>.zip`
  - Example: A submission of group 99 with 2 students (ids: 00059999, 00039999) looks like this: `BI2019_A2_group_99_00059999_00039999.[zip/tgz/rar]`
  - Example: A submission of a single student (with group no. 1) (id: 00968787) looks like this: `BI2019_A2_group_01_00968787.[zip/tgz/rar]`
  - Apply the same naming convention to the report (but obviously with pdf extension)
- **Follow the ACM formatting guidelines, using the templates provided at** https://www.acm.org/publications/proceedings-template. (Proceedings Style File: LaTeX2e - Strict Adherence to SIGS style) LaTeX recommended, but Word/OpenOffice is also ok.
- **Put your group number, names and your student IDs in the report!** (as author info)
- **Report page limit:** Maximum **12 pages. Focus on the key aspects!**

Assignment 2: Data Analytics

- **Use graphs** to visualize findings. Do not just show graphs, also describe what they mean.
- **Use tables** to combine findings and other information for maximum overview whenever possible. Describe what you show and explain the data. Clarify, don't mystify.
- Consider issues of **reproducibility**: ensure you provide sufficient information allowing others to re-produce your experiments.
- **Enumerate and label ALL figures, equations and tables** and refer to them in the report --- describe, explain and integrate them with the text. It must be clear to the reader what information can be learned from them.

General advice:

- Reserve plenty of **time for "playing" with the data** and start early.
- **Collaboration between groups** is welcome, **but** ensure your group uses a **unique data set.**
- **Collaboration inside the group**: Try to perform at least part of the experiments within the group together. Specifically discuss the results amongst each other. Subdividing and **solving tasks alone will cost you more time and not meet the goals of the exercise.** Specifically, we discourage splitting the tasks for part B and C onto the group members. Collaborate, brainstorm and discuss what you find. In the review meeting, **every group member has to demonstrate knowledge of each aspect of the work and the steps taken**.
- Try to **understand your results** and note down any peculiar observations you make, try to provoke "wrong" behavior of the algorithms (over-learning, strange parameters, test wrong encodings, absurd scalings, 99% missing values in an attribute,...) and report these findings as well. Work with the data and with the settings.
- Make sure the **structure of the report** follows the **structure of the tasks** provided here.
- **Explore** Spark/MLlib and the WEKA toolkit beyond the activities required in this assignment – these are powerful tools used in many different settings.

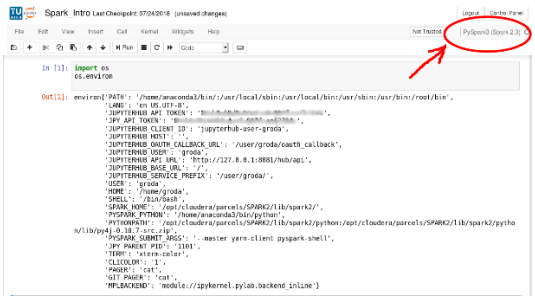
Submission Deadline: January 12, 2020, 23:55

Assignment 2: Data Analytics

Hints on Spark:

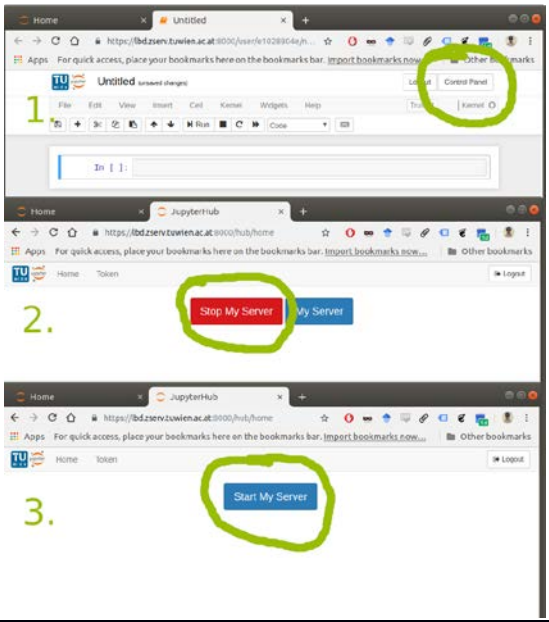
Pyspark kernel in Jupyter notebook

Please make sure that you're using the **PySpark3** kernel when running Spark jobs with pyspark (or PySpark2 if using Python 2). To change kernel use the menu item "Kernel/Change kernel" in the notebook. The Pyspark kernel will run your Spark jobs with an optimized Spark configuration.



Problems connecting to the Spark Server:

After some time of inactivity you may see an error message connecting to the Spark server. In this case go to the Control Panel, klick on 'Stop My Server' and then klick on 'Start My Server' (as described in the picture). After this short procedure you can again open your notebook and start working. Keep in mind that you have to load all the modules etc. again!



188.429 Business Intelligence (VU 4,0) – WS 2019/20

Assignment 2: Data Analytics