Example Examination:
Programming for Data Science (ID2214)

Course code: ID2214
Course name: Programming for data science
Literature and tools:
It is allowed to consult literature, other documents, including lecture slides, notes, etc. It is not allowed to search for solutions online or to communicate in any way with any other individual or forum to either get or provide help to answer the questions both during the exam and before the follow-up Zoom session after the exam. It is allowed to use any text editor/word processor (on a computer, tablet or phone) for writing answers to the questions, and any tool, such as an editor, Jupyter notebook or IDE may be used for developing the programs on part II.
Date and time: Oct. 2, 2020, 08:00-12:00
Examiner: Henrik Boström

Requirements to pass: 5 points on part I and 10 points on part II

On part I, keep the text short and to the point.
On part II, only the Python standard library, the NumPy and pandas libraries may be assumed, in addition to functions explicitly stated in the tasks.
All answers (including blank ones) should be numbered and ordered, and compiled into one single pdf-document that should be uploaded according to the instructions in Canvas.
It is recommended that the provided Jupyter notebook template is used for producing the pdf-document.
Unreadable answers will be ignored.
Good luck!

Part I (Theory, 10 points)

1a. Methodology, 2 points

Assume that we have trained ten different predictive models using various algorithms and parameter settings and then selected the best model based on its accuracy as estimated through ten-fold cross-validation. Is the estimated accuracy for the best model an unbiased estimate of its accuracy on independent test instances? Explain your reasoning.

1b. Data preparation, 2 points

Will mean-value imputation have the same effect, if performed before normalization, on the distribution of the normalized values for those values that were originally not missing, for min-max normalization and z-normalization? Explain your reasoning.

1c. Naive Bayes/k-nearest neighbor (kNN), 2 points

Assume that a large number of binary features are added to a dataset with two class labels c1 and c2, such that for each added feature f, the class conditional probability $P(f = 0|c1) = P(f = 0|c2)$. What potential effect will the addition of such features have on the accuracy of naive Bayes and kNN respectively?

1d. Performance metrics, 2 points

Assume that we have two classification models M1 and M2 that are evaluated on five test instances. Show with an example that M1 can have a higher accuracy than M2, while at the same time M2 has a higher area under the ROC curve (AUC) than M1.

1e. Decision trees/combining models, 2 points

Explain how the generation of a random forest with 100 trees from some dataset may be faster than the generation of a single decision tree (using the standard divide-and-conquer algorithm), even when parallelization is not employed.

Part II (Programming, 20 points)

2a. Data preparation, 10 points

Your task is to define the following Python function that converts each categorical feature in a pandas dataframe to a set of binary features, using one-hot-encoding, where categorical values that occur less than a specified number of times (`min_freq`) should be ignored:

```
create_one_hot(df,min_freq)
```

which given a dataframe `df`, where the columns correspond to features (except for a column named `CLASS`) and the rows to instances, and the integer `min_freq` (which defaults to 5) specifies the minimum number of instances with a specific feature value that is required for a binary feature to be generated for that value. The function should return a new data frame, which is a copy of the original but where each categoric feature has been replaced by a set of binary features (with at most as many new features as there are possible values, and possibly fewer), and a mapping (dictionary) from column name to a set of categorical values (the values for the feature that should not been ignored).
Hint: you may use `counts = df[col].value_counts()` to obtain a pandas data series with the counts of the values in the column `col`. E.g., `counts[val]` will return the number of occurrences of val in the series.

2b. Decision trees, 10 points

Assume that a binary regression tree is represented by an array `T`, where each element in the array corresponds to a node in the tree, on the following form:
A leaf node `N` is represented by:

```
N = [V]
```

where `V` is a predicted value (float).

An internal node `N` is represented by:

```
N = [feature,threshold,left_child_node,right_child_node,left_weight]
```

where `feature` is the name of a numerical feature (name of a column in a pandas dataframe of type float), `left_child_node` and `right_child_node` are indexes referring to the positions in `T` which contain the corresponding child nodes of `N`, where instances with a value on the feature which is less than or equal to `threshold` should be directed to the left child node while instances with a value on the feature greater than the threshold should be directed to the right child node. `left_weight` is a float in the interval [0,1] which corresponds to the fraction of training instances that have reached `N` and then been directed to the left child node.
Your task is to define a Python function

```
predict(T,i)
```

which taken a tree `T` (using the above representation, and assuming that the root node of the tree has index 0) and an instance `i` (a row of a pandas data frame or equivalently, a pandas data series) returns the tree prediction for that instance, i.e., as guided by the conditions (formed by features and thresholds in the internal nodes). If the instance has reached an internal node for which the value for the feature in the condition is missing, then the resulting prediction should be a weighted sum of the predictions obtained from the child nodes, using `left_weight` for the left child node and `1-left_weight` for the right child node. Note that the prediction for an instance that contains no missing values will simply be the prediction of the leaf node which is reached from the root as directed by the encountered conditions.