Course name: Introduction to Machine Learning

Course code: 822047-B-6

Take-home Assignment 2
Academic Year 2022-2023

Sharon Ong and Görkem Saygili
Department of Cognitive Science and Artificial Intelligence
Tilburg University

The Individual Take-home Assignment 2 is worth 100 points and is 20% of your grade. A pass is not required for this component to pass the course as long as the total grade for all components is 5.5 or higher. Because a pass is not required, no resit will be provided for the whole class. Students who fail the assignment may resubmit for a maximum grade of 6.0. Re-submissions must be received before the final exam.

Late submission policy: There is a 5% penalty per day for late submissions. Zero credit is earned if submitted after 7 days past the due date. Start early, most assignments will take longer than you expect.

You will submit your code in a Jupyter notebook (*.ipynb) and a summary table of your results in a csv file. Additional functions may be written in a Python (*.py) file. At the top of your Jupyter notebook file, please reference any code, methods, ideas that are not your own or not provided in this course. Remember that these codes must be publically available and free to use. You do not have to reference the course worksheets, the textbook nor the lecture slides. Please also include any special instructions that are required to run your code. This assignment is due on **24, May 2023**.

A maximum of 5 points will be awarded for a summary table and good coding standards. In this assignment, you will have the opportunity to work with structured (numerical) and unstructured data (i.e. text data).

This assignment accesses the following learning goals:

- implement solutions to real-world machine learning problems.
- use Python libraries for the purposes of model building, evaluation, and parameter learning.

In this assignment, you can apply what you have learnt in this course to answer the following research question:

To what extent can you **classify the genre of a song** from its lyrics and audio features?

The dataset is a modified dataset obtained from [1]. There are two csv files, spotify_songs_train.csv and spotify_songs_test.csv. The first one is the training set and the second is the test data. Both csv files contain the following numerical columns

label : A target value for the genre of a song, where 0 is for pop music, 1 corresponds to R&B and 2 is for rock music.

track_popularity : The song popularity (0-100) where a higher score is better.

danceability : Danceability describes how suitable a track is for dancing based on a combination of musical elements including tempo, rhythm stability, beat strength, and overall regularity. A value of 0.0 is least danceable and 1.0 is most danceable.

energy : Energy is a measure from 0.0 to 1.0 and represents a perceptual measure of intensity and activity. Typically, energetic tracks feel fast, loud, and noisy. For example, death metal has high energy, while a Bach prelude scores low on the scale. Perceptual features contributing to this attribute include dynamic range, perceived loudness, timbre, onset rate, and general entropy.

key : The estimated overall key of the track. Integers map to pitches using standard Pitch Class notation If no key was detected, the value is -1.

loudness : The average loudness of a track in decibels (dB). Values typical range between -60 and 0 db.

mode : The modality (major or minor) of a track, the type of scale from which its melodic content is derived. Major is represented by 1 and minor is 0.

speechiness : The presence of spoken words in a track. The more exclusively speech-like the recording (e.g. talk show, audio book, poetry), the closer to 1.0 the attribute value. Values above 0.66 describe tracks that are probably made entirely of spoken words. Values between 0.33 and 0.66 describe tracks that may contain both music and speech, either in sections or layered, including such cases as rap music. Values below 0.33 most likely represent music and other non-speech-like tracks.

acousticness : A confidence measure from 0.0 to 1.0 of whether the track is acoustic. 1.0 represents high confidence the track is acoustic.

instrumentalness : Predicts whether a track contains no vocals. "Ooh" and "aah" sounds are treated as instrumental in this context. Rap or spoken word tracks are clearly "vocal". The closer the instrumentalness value is to 1.0, the greater likelihood the track contains no vocal content. Values above 0.5 are intended to represent instrumental tracks, but confidence is higher as the value approaches 1.0.

liveness Higher liveness values represent an increased probability that the track was performed live. A value above 0.8 provides strong likelihood that the track is live.

valence A measure from 0.0 to 1.0 describing the musical positiveness conveyed by a track. Tracks with high valence sound more positive (e.g. happy, cheerful, euphoric), while tracks with low valence sound more negative (e.g. sad, depressed, angry).

tempo The overall estimated tempo of a track in beats per minute (BPM). In musical terminology, tempo is the speed or pace of a given piece and derives directly from the average beat duration.

duration_ms Duration of song in milliseconds

A Classification with Numerical Data

This section is worth 70 points.

1. (5 points) Load the train and test dataset, spotify_songs_train.csv and spotify_songs_test.csv. Create a feature set without 'label'. (For train data, assign all numerical data except the column 'label' to "X_train". For test data, assign all the numerical data except the columns "label" to "X_test") Assign the 'label" as a target variable. (For train data, assign the column "label" to "y_train". For test data, assign the column "label" to "y_test").

   Points breakdown: 2 points for loading the dataset, 2 points for creating the feature sets ("X_train" and "X_test") and 1 point for creating your target variable ("y_train" and "y_test")

2. (5 points) Display a histogram plot of the target variable. Visualize each numerical feature in "X_train" in a separate histogram plot.

   Points breakdown: 2 points for displaying a histogram plot of "y_train", the target variable. 3 points for displaying the histogram plot of each numerical feature in "X_train".

3. (5 points) Train a Dummy Classifier on the training data. Evaluate the accuracy, weighted F1-score and confusion matrix on the train and test set.

   Points breakdown: 2 points for training a Dummy Classifier. 2 points for evaluation on the test set, 1 point for evaluation on the training set.

4. (10 points) Using **GridSearchCV**, train a Logistic Regression classifier with a **5-fold** cross validation. Find the best regularization (C) hyperparameter with 5 different C values. Using the best hyperparameter found from grid search on the training set, evaluate the accuracy, weighted F1-score and confusion matrix on the train and test set. (When you call grid_search.fit, the best hyperparameter is automatically found and stored in grid_search)

   Points breakdown: 7 points for evaluating a Logistic Regression classifier with Grid Search and Cross Validation with the correct number of folds. 1 point each for computing the accuracy and 1 point for confusion matrix and 1 point for weighted F1-score. If you use the Logistic Regression classifier without Grid Search and Cross Validation, the maximum possible points is 3. If you run Grid Search without Cross Validation or vice versa, the maximum possible points is 5.

5. (5 points) Scale the train and test features with a Standard Scaler. **Points breakdown**: 3 points for scaling X_train. 2 points for scaling X_test.

6. (5 points) Using the scaled data from Question A.5, train a Logistic Regression classifier with the hyperparameters found in Question A.4.

7. (15 points) Using **GridSearchCV**, Try with two other low-level classifiers (e.g. Decision Trees, SVM, Naive Bayes, MLP, KNN) of your choice with grid search and cross validation. Choose one or two hyperparameters (if any) to search for. Display the best hyperparameters found from grid search. Evaluate the accuracy, weighted F1-score and confusion matrix on the training and test set. It is your choice on whether to use the scaled or unscaled features.

   Points breakdown: For each classifier: 5 points for training each classifier with GridSearchCV. 1.5 points for computing both the accuracy and weighted F1-score for each classifier. 1 point for each confusion matrix. Zero points are awarded if regression algorithms are used.

8. (10 points) Try a Random Forest and Gradient Boosting classifier with **GridSearchCV**. Choose one hyperparameter to search for with a minimum of 5 different values. Evaluate the accuracy, weighted F1-score and confusion matrix on the training and test set. It is your choice on whether to use the scaled or unscaled features.

   Points breakdown: For each classifier: 4 points for each classifier training the classifier with **GridSearchCV**. 1 points for computing the evaluation metrics. A maximum of 2 points will be awarded if **GridSearchCV** is not used.

9. (10 points) Display the feature importance of Random Forests and Gradient Boosting classifiers in bar chart. Plot the coefficients for the Logistic Regression solution. Show only the solutions from the best hyper parameters found via **GridSearchCV**. Comment whether the same features are given similiar importance from the charts and plot. **Points breakdown**: 5 points for display bar charts, 3 points for the coefficient plot, 2 point for your comments.

B Classification with Text Data

This section is worth 15 points. Both the csv files contain the following columns of text

1. (5 points) Load the files spotify_songs_train.csv and spotify_songs_test.csv. Convert the text data into unicode and assign the text to X_train and X_test. You can run the following lines of code to do so:

- `train_data = pd.read_csv('spotify_songs_train.csv')`
  `X_train = train_data['lyrics'].values.astype('str')`
- `test_data = pd.read_csv('spotify_songs_test.csv')`
  `X_test = test_data['lyrics'].values.astype('str')`

Assign the labels in the train dataset to "y_train". Assign the labels in the test dataset to "y_test".

Points breakdown: 2 points for loading the dataset, 2 points for creating the feature sets ("X_train" and "X_test") and 1 point for creating your target variable ("y_train" and "y_test").

2. (5 points) Transform your data (X_train from Question B.1) to a Bag-of-Words representation. Rescale your data using TF-IDF.

   Points breakdown: 3 points for CountVectorizer, 2 points converting your tokens to a TF-IDF format.

3. (5 points) Using the output of Question B.2, train a classifier of your choice. Compute the accuracy, weighted F1-score and confusion matrix on the train and test set.

   Points breakdown: 3 points for training a classifier, 2 points for computing the evaluation metrics.

C Additional solutions to improve your weighted F1-score

This section is worth a maximum of 10 points.

1. (10 points) Implement an additional solutions to improve your weighted F1-score. These solutions can include (but are not limited) to the following:

   - trying different hyper-parameters for text classification (e.g. setting *stop_words* to *english*, different values for *min_df* and different *ngram_range* values) (max 5 points)
   - Using a Voting or Stacking classifier to combine the best classifiers from Part A, B and/or C (max 5 points).
   - Performing univariate transformations on relevant numerical features (max 3 points)
   - Balancing your dataset. (max 3 points)
   - Combine columns of text data (e.g. assign X_train and X_test to the combined contents of the columns 'track_name' and 'lyrics') and retrain your classifiers.
   - Applying dimensionality reduction on either the numerical or text data or both. (max 5 points)

D Summary table and good coding standards

This section is worth 5 points.

1. (2 points) Please include a summary table of all your results in a csv file. The table should include the algorithm used, the data used, the hyperparameter, accuracy (on the test set) and weighted f1 score (on the test set). You can create this table automatically or by manually.
2. (3 points) A maximum of 3 point will be awarded for well organized code that executes without errors.

E Bonus Points and Prizes for best performing solutions

We will to offer bonus points (maximum of 2 points) based on your maximum weighted F1-score. You will only qualify for this bonus if you submit your assignment on time and fill out the form on Canvas indicating your best performing solution and weighted F1 score. Token prizes with be presented to students with the top 10 highest F1-score **on the test set**. Students with the top 3 highest F1-score **on the test set** will receive commemorative certificates and bragging rights.

References

[1] Nakhaee, M. (2020). Audio features and lyrics of Spotify songs. Kaggle. https://www.kaggle.com/datasets/imuhammad/au
features-and-lyrics-of-spotify-songs/