Mental Health Prediction for Kenyan university students

Business Understanding

What is mental health?

Mental health encompasses emotional, physiological, and social well-being. It affects how we feel, think, interact with others, handles stress, and how we make choices. Good mental health is when a person is aware and utilizes his/her ability to deal with normal day to day life challenges whether at school , work and is productive in what they find fulfilling in life.Bad mental health is when a person finds it difficult to cope with how they are feeling, thinking or reacting to things and this can feel just as bad or even worse than physical illness.

Mental health problems range from common problems such as anxiety and depression to rarer problems such as schizophrenia and bipolar disorder.Many factors contribute to mental health problems, including:

1. Biological factors, such as genes or brain chemistry
2. Life experiences, such as trauma or abuse
3. Family history of mental health problems

Why is mental health important?

Mental health influences a variety of outcomes for an individual all the way to a nation. Healthier lifestyles, better physical health, fewer limitations in daily living, higher education, greater productivity in academics,

employment and earnings, better relationships with others, and their overall quality of life are all influenced by positive mental health. Mental health problems are a common human experience and it's likely that,with a good combination of self-care, treatment and support that works for an individual they will get better.

Problem Statement

Despite resources in Kenyan universities growing immensely over the years, development of support services to students has not.Research has shown high levels of mental health problems among university students specifically depression and anxiety with the most affected group being students from poor backgrounds. The lack of or little provision for support services for such students results in dropouts and their inability to reach their full potential.

The use of technologies including machine learning and AI will potentially transform the delivery of mental health services in the coming years.This challenge aims to develop a machine learning model that classifies statements and questions expressed by university students in Kenya when speaking about the mental health challenges they struggle with and come up with a chatbot that will be used for a prototype of a mental health chatbot designed specifically for university students.

Objectives

General Objective

The general objective is to leverage technology to make mental health services more accessible and more user-friendly for young people in Kenya and around the world.

Specific Objectives

To develop a machine learning model that classifies statements and questions expressed by university students in Kenya when speaking about the mental health challenges they struggle with

To help universities establish mental health support and wellness services to their students.

To help university students in Kenya that are facing mental health problems to find resources and support services that will enable them to get better

Data Understanding

1) Data Loading and Exploration:

The dataset was loaded into the notebook using the pandas read_csv() method. The train and test datasets were loaded separately and their shapes were printed out. The train dataset contains 616 rows and 3 columns while the test dataset contains 309 rows and 2 columns. The columns in the dataset were also printed out.

2) Data Types and Missing Values:

The info() method was used to check the data types and missing values in both datasets. The train dataset contains three columns, all of which are of object data types. There are no missing values in the train dataset. The dataset contains two columns, both of which are of data type. There are also no missing values in the test dataset.

3) Class Proportionality:

The proportionality of the classes in the train dataset was checked by using the value_counts() method. The output shows that there is a slight class imbalance in the dataset, with depression being the most frequent class (57%), followed by alcohol (23%), suicide (11%) and drugs (9%).

4) Data Visualization:

A histogram was plotted to visualize the proportionality of the target classes. The histogram shows that depression is the most frequent class, followed by alcohol, suicide, and drugs.

The distribution of the number of words per text was visualized by plotting a histogram. The histogram shows that the average text length lies between 25 and 40 words.

The effect of the number of words per text on mental health status was visualized by plotting a bar chart. The chart shows that depression has the highest number of words per text on average followed closely by suicide. Alcohol and drugs have almost the same number of words per text which makes sense since alcohol is also a type of drug.

Data Preparation.

The data preparation phase involved several steps aimed at cleaning and pre-processing the raw text data to prepare it for machine learning model training.

The following steps were performed in the data preparation phase:

1. Correcting spelling mistakes: In this step, the TextBlob library was used to correct spelling mistakes in the text data. The ‘correct_sent’ function was applied to the ‘text’ column of the train and validation data frames, and the corrected text was saved in a new column ‘corrected_sent’.
2. Changing text to lowercase: The ‘corrected_sent’ column was then converted to lowercase using the ‘lower()’ function to ensure consistency in the text data.
3. Removing punctuation marks: Punctuation marks were removed from the text data using the ‘string. punctuation’ library. Additionally, special characters such as ‘…’ represented as 'â€¦' were replaced with a space.
4. Removing stop words: Stop words are words that do not add much value to the text data and can be safely removed. The 'nltk.corpus.stopwords.words' function was used to download the English stop words, and the 'WordNetLemmatizer' function was used to perform lemmatization. The 'remove_stopwords' function was applied to the 'corrected_sent' column, and the output was saved in a new column 'no_stopwords'.

The resulting data frames 'train_df' and 'validation_df' now have three columns: 'text', 'label', and 'no_stopwords'. The 'text' column contains the original text data, the 'label' column contains the corresponding label ('Depression', 'Drugs', 'Suicide', or 'Alcohol'), and the 'no_stopwords' column contains the pre-processed text data after removing stop words.

The pre-processing steps performed in the data preparation phase helps to ensure that the text data is clean, consistent, and ready for use in machine learning models. By removing spelling mistakes, converting text to lowercase, and removing punctuation marks and stop words, the resulting

text data is more uniform and easier to work with, which can help to improve the performance of machine learning models.

MODELLING

Machine Learning techniques

When selecting and implementing the different machine learning classifier models, six different classifiers were chosen for comparison. This is with regards to previous studies on document classification. This was also to assist in giving broad comparison results for our base models, and reduce the possibility of a high performing classifier being left out. Although many approaches have been proposed, automated text classification is still a major area of research primarily because the effectiveness of current automated text classifiers is not faultless and still needs improvement. The machine learning models chosen and developed for the study were DecisionTree, KNN , RandomForest , AdaBoost, GradientBoost, XGBoost.

APPLYING MACHINE LEARNING METHODS

The modeling process was done by fitting each of our classifiers to our trainset (X_train,y_train) then predicting the test_set.

For easier tracking of performance of our base models, a DataFrame (df_model_results) was constructed to store values of our evaluation metrics (for each model), and a function defined to fill the DataFrame after each successful run of a base model (model_results).

```
#Models to be tested
models = { 'Model' : ['Baseline Decision Tree', 'Baseline KNN Classifier', 'Baseline Random Forest Classifier',\
                      'Baseline Adaboost Classifier', 'Baseline Gradient Boost', 'baseline XGBoost Classifier',\
                       'XGBoost Classifier-Grid Search'],
          'Train Accuracy Score(%)': [0, 0, 0, 0, 0, 0, 0],
          'Test Accuracy Score(%)': [0, 0, 0, 0, 0, 0, 0],
          'Train Log_loss': [0, 0, 0, 0, 0, 0, 0],
          'Test Log_loss': [0, 0, 0, 0, 0, 0, 0] }


#Dataframe holding the model names and accuracy score
df_model_results = pd.DataFrame(models, columns=['Model','Train Accuracy Score(%)', 'Test Accuracy Score(%)', 'Train Log_loss', 'Test Log_loss'])


#Function to fill the dataframe holding model names and accuracy score

def model_results(model_type, y_train, y_train_pred, y_train_pred_prob, y_test, y_test_pred, y_test_pred_prob):
  index_val = df_model_results[df_model_results['Model']==model_type].index

  df_model_results.loc[index_val, 'Train Accuracy Score(%)'] = round(accuracy_score(y_train, y_train_pred), 2)*100
  df_model_results.loc[index_val, 'Test Accuracy Score(%)'] = round(accuracy_score(y_test, y_test_pred), 2)*100
  df_model_results.loc[index_val, 'Train Log_loss'] = log_loss(y_train, y_train_pred_prob, labels=[0,1,2,3])
  df_model_results.loc[index_val, 'Test Log_loss'] = log_loss(y_test, y_test_pred_prob, labels=[0,1,2,3])

  return df_model_results
```

Decision Tree

A decision tree is a flowchart-like structure in which each internal node represents a test on a feature (e.g. whether a coin flip comes up heads or tails) , each leaf node represents a class label (decision taken after computing all features) and branches represent conjunctions of features that lead to those class labels. The paths from root to leaf represent classification rules. Decision trees are constructed via an algorithmic approach that identifies ways to split a data set based on different conditions (can be considered generally as an IF-ELSE statement). It is one of the most widely used and practical methods for supervised learning.

K-Nearest Neighbor

K-Nearest Neighbor classifies the results of the new instance according to the majority of the k-nearest neighbor categories. To classify an unknown document vector, the k-nearest neighbor algorithm ranks the document’s neighbors among the training document vectors, and use the class labels of the k most similar neighbors to predict the class of the input document. The k-NN is quite simple and effective but its drawbacks is its inefficiency at classification time: it requires the entire training set to be ranked for similarity with the test document, which is expensive.

Random Forest

A random forest is a Meta estimator (takes another estimator), that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. It produces great results most of the time even without hyper-parameter tuning.

AdaBoost

AdaBoost or Adaptive Boosting is an iterative ensemble method. AdaBoost classifier builds a strong classifier by combining multiple poorly performing classifiers so that you will get a high accuracy strong classifier. The basic concept behind Adaboost is to set the weights of classifiers and training the data sample in each iteration such that it ensures the accurate predictions of unusual observations. Any machine-learning algorithm can be used as a base classifier if it accepts weights on the training set. Adaboost is less prone to overfitting as the input parameters are not jointly optimized. The accuracy of weak classifiers can be improved by using Adaboost.

GradientBoost

Gradient boosting is a method standing out for its prediction speed and accuracy, particularly with large and complex datasets. We already know that errors play a major role in any machine-learning algorithm. There are mainly two types of error, bias error and variance error. Gradient boost algorithm helps us minimize bias error of the model. The main difference between AdaBoosting, above, and GradientBoosting algorithm is that the later has a fixed based estimator ie. Decision Trees whereas in the former, this can be changed according to the needs of the writer.

XGBoost

XGBoost or extreme gradient boosting is one of the well-known gradient boosting techniques (ensemble) having enhanced performance and speed in tree-based (sequential decision trees) machine learning algorithms.

Unlike other boosting algorithms where weights of misclassified branches are increased, in Gradient Boosted algorithms the loss function is optimized. XGBoost is an advanced implementation of gradient boosting along with some regularization factors (eg. Gamma, alpha and lambda parameters). XGBoost offers a few technical advantages over other gradient boosting approaches, including a more direct route to the minimum error, converging more quickly with fewer steps, and simplified calculations to improve speed and lower compute costs.

Classification Metrics

To be able to evaluate and measure the performance of the classifiers, the accuracy_score and Log_loss metrics were used for the task. These were executed after each classifier was trained and a prediction for the test data done by using the predict method for the specific machine learning algorithms model.

	Model	Train Accuracy Score(%)	Test Accuracy Score(%)	Train Log_loss	Test Log_loss
)	Baseline Decision Tree	99.0	83.0	0.012334	5.590810
I	Baseline KNN Classifier	82.0	70.0	0.512985	2.842548
2	Baseline Random Forest Classifier	99.0	81.0	0.106574	0.775972
3	Baseline Adaboost Classifier	85.0	77.0	1.031080	1.059045
4	Baseline Gradient Boost	97.0	83.0	0.163581	0.509805
5	baseline XGBoost Classifier	93.0	85.0	0.219129	0.451154
5	XGBoost Classifier-Grid Search	0.0	0.0	0.000000	0.000000

For tuning the best performing model with regards to our evaluation metrics, a hyperparameter optimizer was used. That ensured that the performance of the classifier would be at its most optimized version for the document classification task. The hyperparameter optimization was done with the sklearn.model_selection.GridSearchCV functionality.

```
Tuned_xgboost = XGBClassifier()
param_grid = {'learning_rate': [0.1,0.2],
              'max_depth': [4, 6, 10],
              'n_estimators':[200, 300, 400]}


xgbclassifier_grid_search = GridSearchCV(Tuned_xgboost, param_grid, scoring='accuracy', cv=3, n_jobs=1)

xgbclassifier_grid_search.fit(X_train, y_train)

y_test_pred_prob = xgbclassifier_grid_search.predict_proba(X_test)
y_train_pred_prob = xgbclassifier_grid_search.predict_proba(X_train)

y_test_pred = xgbclassifier_grid_search.predict(X_test)
y_train_pred = xgbclassifier_grid_search.predict(X_train)

print('**********************************************************************')
print(classification_report(y_test, y_test_pred ))

print('**********************************************************************')
model_results('XGBoost Classifier-Grid Search',y_train, y_train_pred, y_train_pred_prob, y_test, y_test_pred, y_test_pred_prob)
```

EVALUATION

Based on the accuracy of our best model, the XGBoost Classifier-Grid Search, had a train accuracy score of 92% and test accuracy of 85%. The same model had a train and test log score of 0.242 and 0.439, respectively. We chose to use log loss as a performance metric because it takes into account the probabilities underlying in the model, not only the final output of the classification. This means, the higher the probabilities the lower the log loss value, which is better because it is basically a measure of uncertainty and the lower it is, the more certain we are of the potential occurrence.

Since XGBoost classifier had the lowest log loss score, and it was chosen to be the best model, even though we had prior models with high training and test data accuracy.In conclusion, the best model for the prediction of future patient mental states is XGBoost Classifier tuned.

Conclusion And Recommendation

Findings:

1. The dataset featured more observations of Depression and Anxiety than other classifications as a result the models used had lower recall and f1 scores in the classification of Drug and Alcohol related problems
2. All models performed fairly well in classifying mental health problems in both the training and testing sets although accuracy was higher on the training data.

Recommendations

From our model's ability to accurately identify and classify mental health problems at least 85% of the time. We would recommend that:

1. The organization @Zindi Africa should collect more data that features drug and alcohol related problems as well as other mental health problems that have not been featured in the observations used here.
2. The model be integrated into the chatbot prototype to carry out tests on actual university students and collect data on how it performed in classifying problems they were facing.
3. The chatbot should also feature a database containing resources and services available to the students based on the problem that the model was able to identify. This will ensure that actual help is availed to the user.