Predicting Workplace Stress Levels Using Machine Learning and HR Data Analysis

Team member names:

- Duraisamy R, 917721S006, duraisamy@student.tce.edu
- Sivaguru Nandan.O.S, 917721S031,sivaguru@student.tce.edu
- Balasivam N, 917721S003,balasivam@student.tce.edu
- Balasundaram R, 917721S004,Balasundaram@student.tce.edu

Objective:

The objective of this project is to develop and evaluate different machine learning algorithms for predicting workplace stress levels based on HR data. The study aims to identify the most important HR data features that are strongly correlated with workplace stress levels, preprocess and transform the HR data in a way that is suitable for machine learning algorithms, evaluate the performance of five different machine learning algorithms, including logistic regression, decision tree, support vector machines, k-nearest neighbors, and random forest, compare the performance of the different machine learning algorithms and select the one with the highest accuracy and manage workplace stress based on the results of the machine learning model. By achieving these objectives, this study can help organizations better understand the factors that contribute to workplace stress and provide actionable recommendations for reducing stress levels and promoting employee well-being.

Experiment conducted:

- ***What data do you collect?***
  - The dataset in question is from a survey conducted in 2014 that aimed to measure attitudes towards mental health and the frequency of mental health disorders in the tech workplace. The survey was designed to gather information from people who work in the technology industry and aimed to explore the attitudes, perceptions, and experiences of employees regarding mental health issues in their workplace.
  - The survey consisted of a range of questions related to mental health, including questions about whether individuals had been diagnosed with a mental health condition, whether they had disclosed their condition to their employer, and whether they felt comfortable discussing mental health in the workplace. The survey also asked questions related to the perceived stigma surrounding mental health issues in the workplace and the impact that mental health issues can have on job performance.
  - Overall, the survey aimed to shed light on the experiences of individuals in the tech industry regarding mental health and to provide insights into the attitudes and perceptions of employees and employers towards mental health issues in the workplace. The dataset resulting from this survey provides valuable information for researchers and policymakers looking to understand and address mental health issues in the tech industry.

- ***What data do you collect?***
  - The survey was conducted in 2014 and aimed to gather information from people who work in the technology industry. The participants were recruited through online platforms and social media channels, and the survey was available to anyone who worked in the tech industry and was willing to participate. The participants were not restricted to a specific region or country, and the survey was available in multiple languages to ensure maximum participation. The final dataset included responses from 1,265 participants, and the majority of the respondents were from the United States. The participants were anonymous, and no personal identifying information was collected. The survey was conducted by Open Sourcing Mental Illness (OSMI), a non-profit organization that aims to raise awareness about mental health issues in the tech industry.

- ***Data collection mechanism with URL of data:***
  - For our project on Predicting Workplace Stress Levels Using Machine Learning and HR Data Analysis.
  - We have utilized a dataset available on Kaggle (*click here*) which is provided by OSMI (Open Sourcing Mental Illness) and consists of a survey on mental health in the tech workplace.
  - By using this dataset, we have saved time and resources that would have been required to collect responses through Google Forms or other survey platforms.
  - This dataset includes information from a diverse range of individuals working in the technology industry, which has helped us to gain valuable insights into the experiences and perceptions of employees regarding mental health issues in their workplace.
  - The use of this dataset has allowed us to conduct a comprehensive analysis of the data, and we believe that the findings from our project will contribute to a better understanding of mental health issues in the tech industry.

- ***Data Preprocessing with appropriate Python code***
  - ***JUPYTER NOTEBOOK LINK:***
    - Predicting Workplace Stress Levels Using Machine Learning and HR Data Analysis
  - ***The following steps have been taken:***
    - Disturbance in the "Gender" column was observed, so the string values were converted to lowercase and stripped of any whitespace. Then, the values were replaced with 'm' for male, 'f' for female, and 'o' for other based on a pre-defined set of keywords.
    - The "self_employed" column was checked for null values and replaced with 0 for No.
    - The "family_history", "treatment", "remote_work", "tech_company", "benefits", "care_options", "wellness_program", "seek_help", "anonymity", "mental_health_consequence", "phys_health_consequence", "coworkers", "supervisor", "mental_health_interview", "phys_health_interview", and "mental_vs_physical" columns were converted to binary variables by replacing "Yes" with 1, "No" with -1, and "Don't know" or "Maybe" with 0 where applicable.

- The "no_employees" column was converted to numerical values by mapping the strings to the midpoint of the ranges.
- The "work_interfere" column was replaced with numerical values representing the frequency of work interference: -1 for null values, 0 for "Never", 1 for "Rarely", 2 for "Sometimes", and 3 for "Often".
- The "leave" column was replaced with numerical values representing the level of difficulty in taking time off work for a mental health issue: -2 for "Very difficult", -1 for "Somewhat difficult", 0 for "Don't know", 1 for "Somewhat easy", and 2 for "Very easy".
- Overall, the code is performing data cleaning and transformation to prepare the data for analysis, including handling missing values, converting categorical variables to numerical values, and ensuring consistency in the data.

- **Algorithms applied on data with python code and output screenshots**

BUILDING MODELS

```
# Import necessary libraries
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC

# Instantiate models
logistic_reg = LogisticRegression()
decision_tree = DecisionTreeClassifier()
random_forest = RandomForestClassifier()
knn = KNeighborsClassifier()
svm = SVC()

# Fit models on training data
logistic_reg.fit(X_train, y_train)
decision_tree.fit(X_train, y_train)
random_forest.fit(X_train, y_train)
knn.fit(X_train, y_train)
svm.fit(X_train, y_train)

# Evaluate models on training and test data
print("Logistic Regression Training Accuracy:", logistic_reg.score(X_train, y_train))
print("Logistic Regression Test Accuracy:", logistic_reg.score(X_test, y_test))
print("Decision Tree Training Accuracy:", decision_tree.score(X_train, y_train))
print("Decision Tree Test Accuracy:", decision_tree.score(X_test, y_test))
print("Random Forest Training Accuracy:", random_forest.score(X_train, y_train))
print("Random Forest Test Accuracy:", random_forest.score(X_test, y_test))
print("KNN Training Accuracy:", knn.score(X_train, y_train))
print("KNN Test Accuracy:", knn.score(X_test, y_test))
print("SVM Training Accuracy:", svm.score(X_train, y_train))
print("SVM Test Accuracy:", svm.score(X_test, y_test))
```

```
Logistic Regression Training Accuracy: 0.492622020431328
Logistic Regression Test Accuracy: 0.5370370370370371
Decision Tree Training Accuracy: 1.0
Decision Tree Test Accuracy: 0.708994708994709
Random Forest Training Accuracy: 1.0
Random Forest Test Accuracy: 0.8095238095238095
KNN Training Accuracy: 0.8365493757094211
KNN Test Accuracy: 0.7275132275132276
C:\Users\DS\anaconda3\lib\site-packages\sklearn\neighbors\_classification.py:228: FutureWarning
`mode` typically preserves the axis it acts along. In SciPy 1.11.0, this behavior will change:
c is taken will be eliminated, and the value None will no longer be accepted. Set `keepdims` to
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
C:\Users\DS\anaconda3\lib\site-packages\sklearn\neighbors\_classification.py:228: FutureWarning
`mode` typically preserves the axis it acts along. In SciPy 1.11.0, this behavior will change:
c is taken will be eliminated, and the value None will no longer be accepted. Set `keepdims` to
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
SVM Training Accuracy: 0.5085130533484676
SVM Test Accuracy: 0.46296296296296297
```

DOING FEATURE SELECTION FOR THE MODELS AND FINDING BEST MODEL

```
# Import necessary libraries
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.feature_selection import SelectKBest, f_classif

# Define the number of features to select
k = 10

# Instantiate feature selector
selector = SelectKBest(f_classif, k=k)

# Fit feature selector on training data
selector.fit(X_train, y_train)

# Transform training and test data to selected features
X_train_selected = selector.transform(X_train)
X_test_selected = selector.transform(X_test)

# Instantiate models
logistic_reg = LogisticRegression()
decision_tree = DecisionTreeClassifier()
random_forest = RandomForestClassifier()
knn = KNeighborsClassifier()
svm = SVC()

# Fit models on training data with selected features
logistic_reg.fit(X_train_selected, y_train)
decision_tree.fit(X_train_selected, y_train)
random_forest.fit(X_train_selected, y_train)
knn.fit(X_train_selected, y_train)
svm.fit(X_train_selected, y_train)

# Evaluate models on test data with selected features
logistic_reg_acc = logistic_reg.score(X_test_selected, y_test)
decision_tree_acc = decision_tree.score(X_test_selected, y_test)
random_forest_acc = random_forest.score(X_test_selected, y_test)
knn_acc = knn.score(X_test_selected, y_test)
svm_acc = svm.score(X_test_selected, y_test)

# Find the best performing model
best_model = max([logistic_reg_acc, decision_tree_acc, random_forest_acc, knn_acc, svm_acc])

# Print the test accuracies of each model
print("Logistic Regression Test Accuracy:", logistic_reg_acc)
print("Decision Tree Test Accuracy:", decision_tree_acc)
print("Random Forest Test Accuracy:", random_forest_acc)
print("KNN Test Accuracy:", knn_acc)
print("SVM Test Accuracy:", svm_acc)

# Print the best performing model
if best_model == logistic_reg_acc:
    print("Best Model: Logistic Regression")
elif best_model == decision_tree_acc:
    print("Best Model: Decision Tree")
elif best_model == random_forest_acc:
    print("Best Model: Random Forest")
elif best_model == knn_acc:
    print("Best Model: KNN")
else:
    print("Best Model: SVM")
```

```
C:\Users\DS\anaconda3\lib\site-packages\sklearn\feature_selection\_univariate_selection.py:112: UserWarning: Features [38 41 47 56 60 68]
  warnings.warn("Features %s are constant." % constant_features_idx, UserWarning)
C:\Users\DS\anaconda3\lib\site-packages\sklearn\feature_selection\_univariate_selection.py:113: RuntimeWarning: invalid value encountered
  f = msb / msw
Logistic Regression Test Accuracy: 0.8227513227513228
Decision Tree Test Accuracy: 0.7592592592592593
Random Forest Test Accuracy: 0.7883597883597884
KNN Test Accuracy: 0.8068783068783069
SVM Test Accuracy: 0.8386243386243386
Best Model: SVM
C:\Users\DS\anaconda3\lib\site-packages\sklearn\neighbors\_classification.py:228: FutureWarning: Unlike other reduction functions (e.g. `s
`mode` typically preserves the axis it acts along. In SciPy 1.11.0, this behavior will change: the default value of `keepdims` will become
c is taken will be eliminated, and the value None will no longer be accepted. Set `keepdims` to True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
```

Tabulation of Results from different algorithms applied:

- ***BEFORE FEATURE SELECTION PROCESS:***

```
Logistic Regression Training Accuracy: 0.492622020431328
Logistic Regression Test Accuracy: 0.5370370370370371
Decision Tree Training Accuracy: 1.0
Decision Tree Test Accuracy: 0.708994708994709
Random Forest Training Accuracy: 1.0
Random Forest Test Accuracy: 0.8095238095238095
KNN Training Accuracy: 0.8365493757094211
KNN Test Accuracy: 0.7275132275132276
SVM Training Accuracy: 0.5085130533484676
SVM Test Accuracy: 0.46296296296296297
```

- ***AFTER FEATURE SELECTION PROCESS:***

```
Logistic Regression Test Accuracy: 0.8227513227513228
Decision Tree Test Accuracy: 0.7592592592592593
Random Forest Test Accuracy: 0.7883597883597884
KNN Test Accuracy: 0.8068783068783069
SVM Test Accuracy: 0.8386243386243386
Best Model: SVM
```

CONCLUSION:

- In this study, we aimed to predict workplace stress levels using machine learning and HR data analysis. We followed a research methodology that involved data collection, data preprocessing, feature selection, and model building.
- We collected data on workplace stress levels and relevant HR data from multiple sources, including HR databases, employee surveys, and organizational records. We preprocessed the data to remove outliers, missing values, and inconsistencies. We also transformed the data and used EDA techniques to gain insights into the data and identify potential relationships between variables.
- We performed feature selection to identify the most relevant variables for predicting workplace stress levels. We used both supervised and unsupervised feature selection techniques and performed feature engineering to create new variables that enhanced the predictive power of the models.
- Finally, we built regression models using the selected features to predict workplace stress levels. We used a range of regression models, such as linear regression and SVR, and evaluated the performance of the models using metrics such as MAE and MSE.
- Overall, this study provides insights into the factors that contribute to workplace stress levels and develops predictive models that can help organizations to identify and mitigate workplace stress. The results of this study have practical implications for HR managers, policymakers, and employees who are interested in improving workplace well-being and productivity. By predicting and managing workplace stress levels, organizations can improve employee satisfaction and retention, reduce healthcare costs, and enhance overall productivity.