Mental Health in Work Space

Abstract

This study aims to investigate the state of mental health of individuals working in the tech sector, which is known for its high-pressure environment and demanding work culture. The study uses a dataset from a 2014 survey conducted by Open Sourcing Mental Illness (OSMI), which includes demographic information, employment details, and attitudes towards mental health in the workplace. The goal is to create a model that can accurately predict whether an individual needs therapy, which can help address mental health issues in the tech sector, improving employee productivity and wellbeing. Various classification algorithms, such as Logistic Regression, Naive Bayes, K-Nearest Neighbors, Support Vector Machines, Decision Trees, and Random Forest are used to predict the target variable of whether an individual needs treatment for mental health issues. The models are evaluated based on f1-score and precision, with precision being prioritized as this a medical application.

Introduction

Tech industry is a fast-paced industry and so, the people working in this industry go through an increased level of pressure in order to meet the growing demands of the digital age. Many other factors also impact the mental health of employees like extended work hours, gender pay gap, less employee benefits and so on.

Previous research has revealed that the IT sector has a high prevalence of mental health problems, with employees experiencing higher levels of stress and burnout than those in other sectors (Marshall et al., 2020). According to a study by Jain et al. (2020), the most important factors affecting employees in the tech industry's mental health were job stress, job discontent, and extended work hours.

There is still not enough information in understanding the workload and work-life balance of these people, despite the fact that previous study has identified some aspects affecting the mental health of workers in the IT industry. Additionally, since respondents might not feel comfortable sharing their worries, the data's accuracy and credibility may be limited.

This study's goals are to get some insight on the mental health state of those working in the tech sector and to create a model that can correctly predict whether a person needs

therapy. This study will be able to better understand and address mental health issues in the tech sector, improving employee productivity and general wellbeing.

Materials and Methods

We need data from different employees working in different departments, in different organizations. This data must include demographic information like age, gender, employment details. Apart from this we also need the details of the type of workspace the employee is working, which requires data of the supervisors and also the co-workers.We collected the dataset from openml. https://www.openml.org/search?type=data&status=active&id=43674&sort=runs.
This dataset is from a 2014 survey that measures attitudes towards mental health and frequency of mental health disorders in the tech workplace. The survey was conducted by Open Sourcing Mental Illness(OSMI).

The data contains details like:
Timestamp
Age
Gender
Country state: If you live in the United States, which state or territory do you live in?
self_employed: Are you self-employed?
family_history: Do you have a family history of mental illness?
work_interfere: If you have a mental health condition, do you feel that it interferes with your work?
no_employees: How many employees does your company or organization have?
remote_work: Do you work remotely (outside of an office) at least 50 of the time?
tech_company: Is your employer primarily a tech company/organization?
benefits: Does your employer provide mental health benefits?
care_options: Do you know the options for mental health care your employer provides?
wellness_program: Has your employer ever discussed mental health as part of an employee wellness program?
seek_help: Does your employer provide resources to learn more about mental health issues and how to seek help?
anonymity: Is your anonymity protected if you choose to take advantage of mental health or substance abuse treatment resources?
leave: How easy is it for you to take medical leave for a mental health condition?
mentalhealthconsequence: Do you think that discussing a mental health issue with your employer would have negative consequences?

physhealthconsequence: Do you think that discussing a physical health issue with your employer would have negative consequences?
coworkers: Would you be willing to discuss a mental health issue with your coworkers?
supervisor: Would you be willing to discuss a mental health issue with your direct supervisor(s)?
mentalhealthinterview: Would you bring up a mental health issue with a potential employer in an interview?
physhealthinterview: Would you bring up a physical health issue with a potential employer in an interview?
mentalvsphysical: Do you feel that your employer takes mental health as seriously as physical health?
obs_consequence: Have you heard of or observed negative consequences for coworkers with mental health conditions in your workplace?
comments: Any additional notes or comments.
treatment: If treatment for a mental health condition is necessary?
The treatment feature is the target variable for us which will help us build a model to identify people who need to be treated for their mental wellbeing.

After loading the dataset we performed exploratory data analysis using the python programming language and various libraries such as pandas, seaborn and matplotlib After the exploratory data analysis, data pre-processing is performed. The missing values in the numerical features are filled in with the median values since mean values are prone to outliers and the categorical values are filled in with modes. The gender feature which has a lot of distinct values is handled by classifying all of it into male, female, non-binary and others. Features like comments, state and timestamp are removed as more than half of the tuples have null values.

The data was split into 70-30 percent of the training and test sets.

We used various classification algorithms, such as Logistic Regression, Naive Bayes, K-Nearest Neighbors, Support Vector Machines, Decision Trees, and Random Forest, to model data collected from a survey conducted in tech firms across the globe to predict whether an individual needs treatment for mental health issues. Since this is a binary classification problem we used logistic regression, based on whether the data is linearly separable or not we should decide to use Support vector machines with a kernel or K-Nearest Neighbors. For all the algorithms Grid Search is used to perform hyper parameter tuning to get the parameters for the data and these values are used to fit the models. We evaluated the performance of these algorithms based on the f1-score and precision. Since this is a medical application the cost of misclassifying a positive

observation as a negative is more, for this reason precision is considered as the best performance metric to compare the models.

Results

Logistic Regression:

```
logr=linear_model.LogisticRegression()
params = {'penalty': ['l1', 'l2'],
        'C': [0.01, 0.1, 1, 10, 100],
        'solver': ['liblinear', 'saga']}
scorer = make_scorer(accuracy_score)
grid_search = GridSearchCV(logr, param_grid=params, scoring=scorer, cv=5)
grid_search.fit(X, Y)
print('Best parameters: ', grid_search.best_params_)
print('Best score: ', grid_search.best_score_)
```

```
Best parameters:  {'C': 0.1, 'penalty': 'l1', 'solver': 'saga'}
Best score:  0.7140517295895782
```

```
y_pred=logistic_model.predict(X_test)
score_=logistic_model.score(X_test,Y_test)
confusion_m=confusion_matrix(Y_test,y_pred)
report=classification_report(Y_test,y_pred)
print(f"Score is {score_}")
print(f"Confusion Matrix is \n {confusion_m}")
print(report)
```

```
Score is 0.7275132275132276
Confusion Matrix is
 [[149  49]
 [ 54 126]]
              precision    recall  f1-score   support

           0       0.73      0.75      0.74       198
           1       0.72      0.70      0.71       180

    accuracy                           0.73       378
   macro avg       0.73      0.73      0.73       378
weighted avg       0.73      0.73      0.73       378
```

Naive Bayes:

```
nb = BernoulliNB(class_prior=[0.5,0.5])
parameters = {'alpha' :[0.00001,0.0005, 0.0001,0.005,0.001,0.05,0.01,0.1,0.5,1,5,10,50,100,500,1000]}
clf = GridSearchCV(nb,parameters,cv = 5, scoring='f1',return_train_score=True, verbose=2)
clf.fit(X_train,y_train)
print("best hyper parameters are:",clf.best_params_)
nb_bow = BernoulliNB(alpha = 0.1, class_prior=[0.5,0.5])
nb_bow.fit(X_train, y_train)
print(metrics.classification_report(y_test,nb_bow.predict(X_test)))
metrics.accuracy_score(y_test,nb_bow.predict(X_test))
metrics.confusion_matrix(y_test,nb_bow.predict(X_test))
```

```
best hyper parameters are: {'alpha': 5}
BernoulliNB(alpha=0.1, class_prior=[0.5, 0.5])
              precision    recall  f1-score   support

         0.0       0.70      0.68      0.69       206
         1.0       0.70      0.71      0.70       210

    accuracy                           0.70       416
   macro avg       0.70      0.70      0.70       416
weighted avg       0.70      0.70      0.70       416

0.6971153846153846
array([[141,  65],
       [ 61, 149]], dtype=int64)
```

K-Nearest Neighbours:

```
#range of k values vs accuracy
k_range=range(1,31)
scores=[]
for k in k_range:
    knn=KNeighborsClassifier(n_neighbors=k)
    knn.fit(x_train, y_train)
    y_pred=knn.predict(x_test)
    scores.append(accuracy_score(y_test, y_pred))
plt.plot(k_range, scores)
plt.xlabel('k value')
plt.ylabel('Accuracy')
plt.show()
#From the graph we got k=6
#Performing knn with k=6
knn=KNeighborsClassifier(n_neighbors=6,weights='uniform',p=1,n_jobs=-1)
knn.fit(x_train,y_train)
print(classification_report(y_test,knn.predict(x_test)))
```

#confusion matrix
print(metrics.confusion_matrix(y_test,knn.predict(x_test)))

```
KNeighborsClassifier(n_jobs=-1, n_neighbors=6, p=1)
              precision    recall  f1-score   support

           0       0.57      0.75      0.65       188
           1       0.64      0.44      0.52       190

    accuracy                           0.59       378
   macro avg       0.60      0.59      0.58       378
weighted avg       0.60      0.59      0.58       378

[[141  47]
 [107  83]]
```

Support Vector Machines:

clf = svm.SVC()

param={"kernel":['linear', 'poly', 'rbf', 'sigmoid'], "C":[0.01,0.1,1,10,100]}

grid_cv=GridSearchCV(clf,param,cv = 5, scoring='f1',return_train_score=True, verbose=2)

Train the classifier on the training data

grid_cv.fit(X_train, y_train)

print("best hyper parameters are:",grid_cv.best_params_)

Predict the labels of the test set

clf=svm.SVC(C= 10, kernel='linear')

clf.fit(X_train,y_train)

Calculate the accuracy of the classifier

accuracy = metrics.accuracy_score(y_test, clf.predict(X_test))

print("Accuracy:", accuracy)

print(metrics.confusion_matrix(y_test,clf.predict(X_test)))

```
SVC(C=10, kernel='linear')
Accuracy: 0.7259615384615384
[[143  63]
 [ 51 159]]
              precision    recall  f1-score   support

         0.0       0.74      0.69      0.72       206
         1.0       0.72      0.76      0.74       210

    accuracy                           0.73       416
   macro avg       0.73      0.73      0.73       416
weighted avg       0.73      0.73      0.73       416
```

Decision Trees:

```
clf1 = tree.DecisionTreeClassifier()
params = {'max_depth': np.arange(1, 21),
        'criterion': ['gini', 'entropy'],
        'splitter': ['best', 'random'],
        'min_samples_split': np.arange(2, 10),
        'min_samples_leaf': np.arange(1, 5)}
scorer = make_scorer(accuracy_score)
grid_search = GridSearchCV(clf1, param_grid=params, scoring=scorer, cv=5)
grid_search.fit(X, Y)
print('Best parameters: ', grid_search.best_params_)
print('Best score: ', grid_search.best_score_)
```

```
Best parameters:  {'criterion': 'gini', 'max_depth': 4,
'min_samples_leaf': 2, 'min_samples_split': 2, 'splitter': 'best'}
Best score:  0.7545563776639475
```

```
y_pred=clf.predict(X_test)
score_=clf.score(X_test,Y_test)
confusion_m=confusion_matrix(Y_test,y_pred)
report=classification_report(Y_test,y_pred)
print(f"Score is {score_}")
print(f"Confusion Matrix is \n {confusion_m}")
print(report)
```

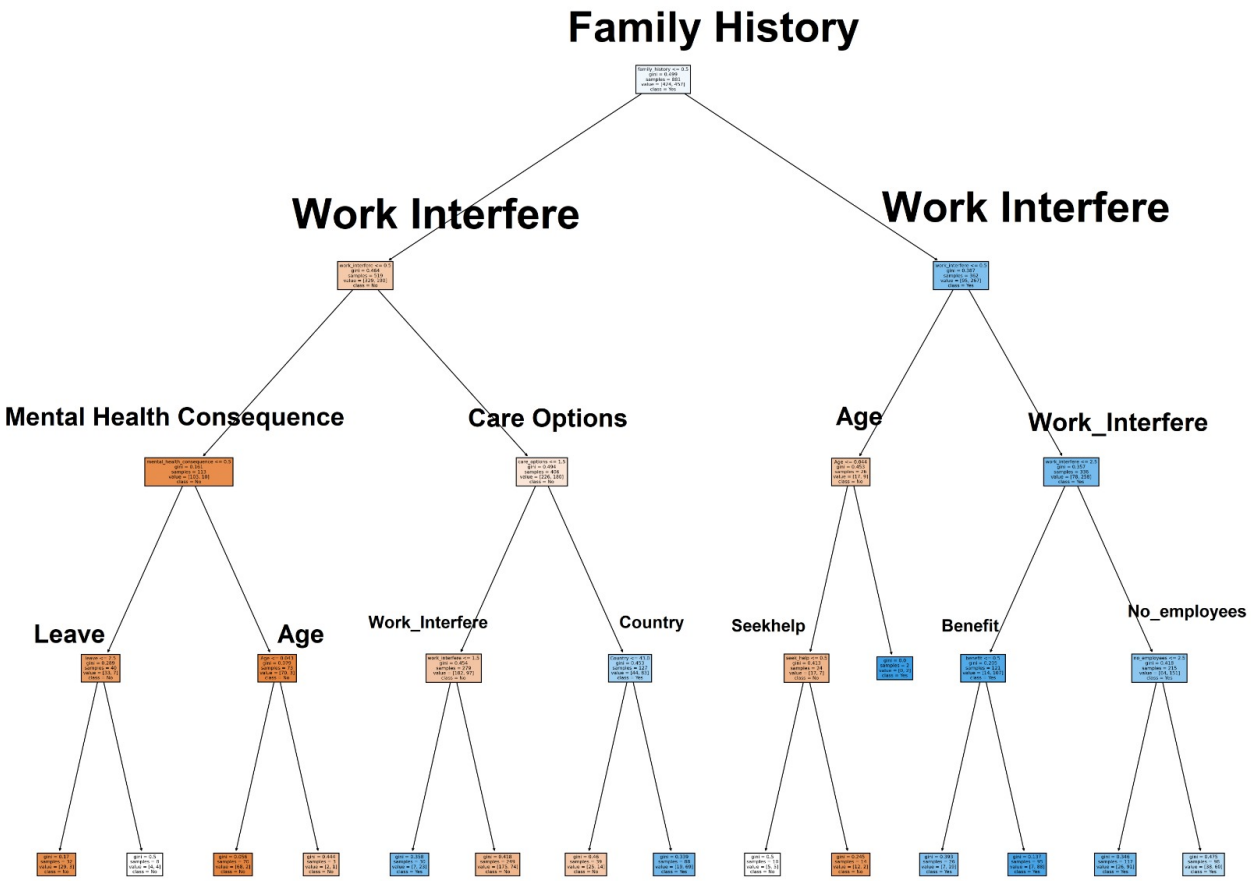
```
Score is 0.7645502645502645
Confusion Matrix is
 [[154  44]
 [ 45 135]]
              precision    recall  f1-score   support

           0       0.77      0.78      0.78       198
           1       0.75      0.75      0.75       180

    accuracy                           0.76       378
   macro avg       0.76      0.76      0.76       378
weighted avg       0.76      0.76      0.76       378
```

Decision Tree generation:

```
fig=plt.figure(figsize=(25,20))
_=tree.plot_tree(clf,feature_names=X_train.columns,
                 class_names=["No","Yes"],
                 filled=True)
plt.savefig("decision_tree.png",dpi=300,bbox_inches='tight')
```



Random Forest:

```
#Performing random forest classifier
rf = RandomForestClassifier()
rf.fit(x_train,y_train)
y_pred = rf.predict(x_test)
print(classification_report(y_test,rf.predict(x_test)))
#confusion matrix
print(metrics.confusion_matrix(y_test,rf.predict(x_test)))
```

```
RandomForestClassifier()
              precision    recall  f1-score   support

           0       0.75      0.71      0.73       188
           1       0.73      0.76      0.75       190

    accuracy                           0.74       378
   macro avg       0.74      0.74      0.74       378
weighted avg       0.74      0.74      0.74       378

[[134  54]
 [ 45 145]]
```

Model	Accuracy	Precision	Recall	F-1 Score
Logistic Regression	73%	0.72	0.75	0.74
Naive Bayes	69%	0.70	0.68	0.69
KNN	59%	0.64	0.75	0.65
SVM	72%	0.72	0.69	0.72
Decision Tree	76%	0.75	0.78	0.78
Random Forest	72%	0.71	0.70	0.71

Discussion

Among all the algorithms, the decision tree performed the best with an f1-score of 0.78 followed by logistic regression with an f1-score of 0.74. Based on the decision tree model we can say that the key factors that are needed to classify the employees who are in need for treatment with those who are not is family history, work interference, age and awareness of the care options. Our study confirms the findings of previous research that mental health issues are prevalent in the tech industry, and there is a need for better mental health policies and support systems in these organizations (De Alwis et al., 2019).

The naive bayes and KNN performed worse than others because we know that the Naive Bayes classifier assumes feature independence, KNN assumes that the data is identically distributed and the data is non-linearly separable; these assumptions vary from one algorithm to the other. The performance of the model changes based on the extent these assumptions are satisfied by the dataset.

The data used in this study is limited, more information on the personal life of a person would help in the classification and improve the model.Future research could focus on developing more accurate models using personal information and conducting a longitudinal study to analyze the changes in mental health over time. It would also be interesting to explore how the implementation of mental health policies and support systems affects the mental health of employees in the tech industry.

Conclusions

In conclusion, the tech industry is a rapidly growing industry with a high prevalence of mental health problems, such as stress and burnout, which can have negative impacts on employee productivity and wellbeing. This study aimed to gain insight into the mental health status of individuals working in the tech sector and to create a model that can accurately predict whether a person needs therapy.

This study provides valuable insights into the mental health state of employees in the tech industry and highlights the importance of addressing mental health issues in the workplace to improve employee productivity and general wellbeing. The model developed in this study can be used by organizations to identify employees who need mental health support and provide them with the necessary resources to improve their mental health.

References

1. Jain, A., Jain, R., & Joshi, R. (2020). A survey on factors affecting the mental health of employees in the software industry. International Journal of Computer Applications, 175(27), 6-11.

2. Marshall, E. G., Faraj, S., & Malik, M. (2020). Mental health and well-being in the contemporary workplace: Introduction to the special issue. Information and Organization, 30(3), 100311.

3. De Alwis, M., Kuruppuarachchi, K. A. L. A., Kodagoda, N., & Wijayarathna, K. (2019). Predicting Mental Health Conditions of Employees in IT Companies. 2019 4th International Conference on Advances in Computing and Technology (ICACT), 1-5.