



MUSIC & MENTAL HEALTH SURVEY RESULTS

STAT 574 Data Mining



Background Information

Review Article | [Open access](#) | Published: 22 June 2021

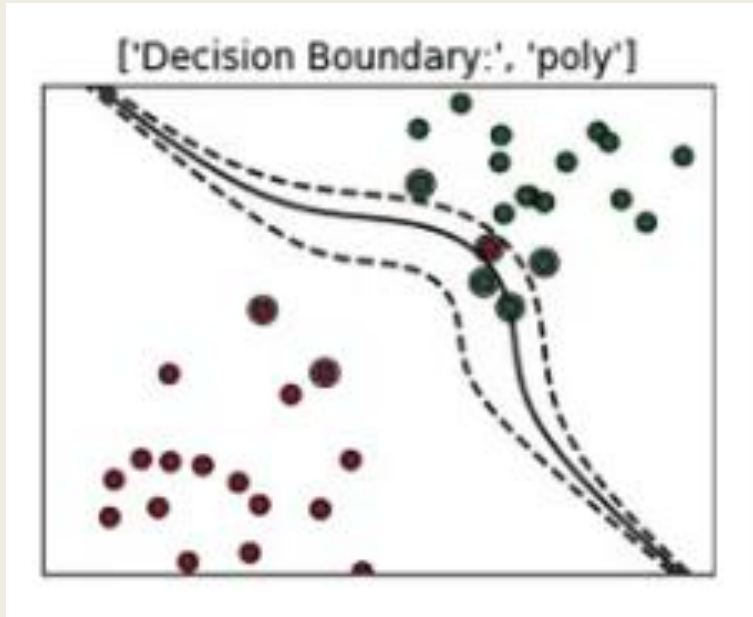
Mental health and music engagement: review, framework, and guidelines for future studies

[Daniel E. Gustavson](#) , [Peyton L. Coleman](#), [John R. Iversen](#), [Hermine H. Maes](#), [Reyna L. Gordon](#) & [Miriam D. Lense](#)

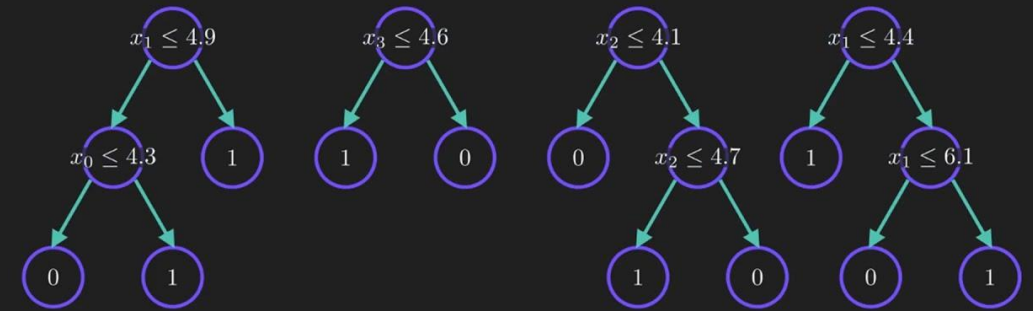
[Translational Psychiatry](#) 11, Article number: 370 (2021) | [Cite this article](#)

Question: Can the effects of music on an individual's mental health be predicted accurately based on their personal characteristics, music listening habits, and mental health indicators?





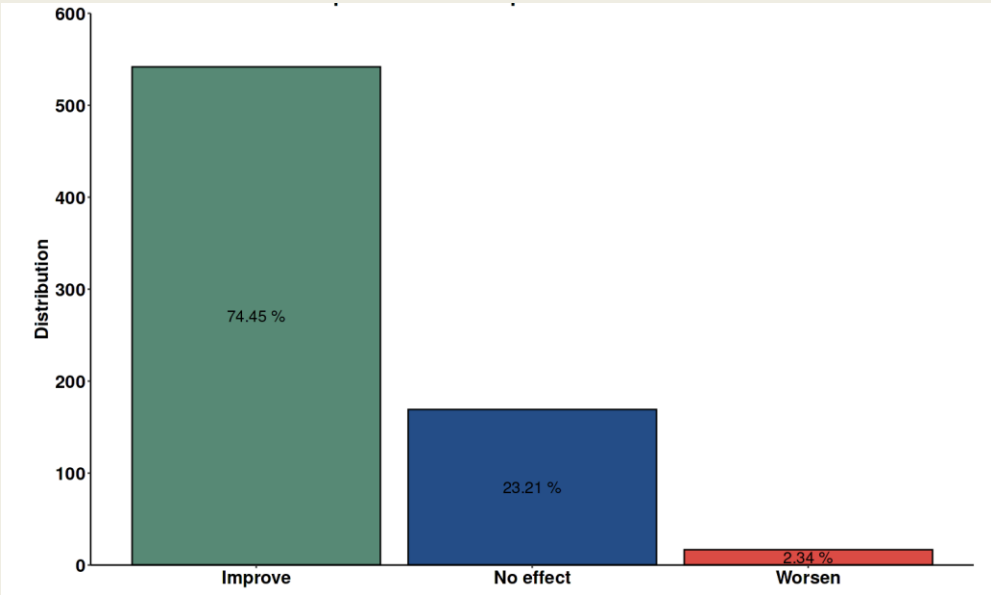
Random Forest



MACHINE LEARNING MODELING: RF & SVM (POLY)

DATASET CHARACTERISTICS

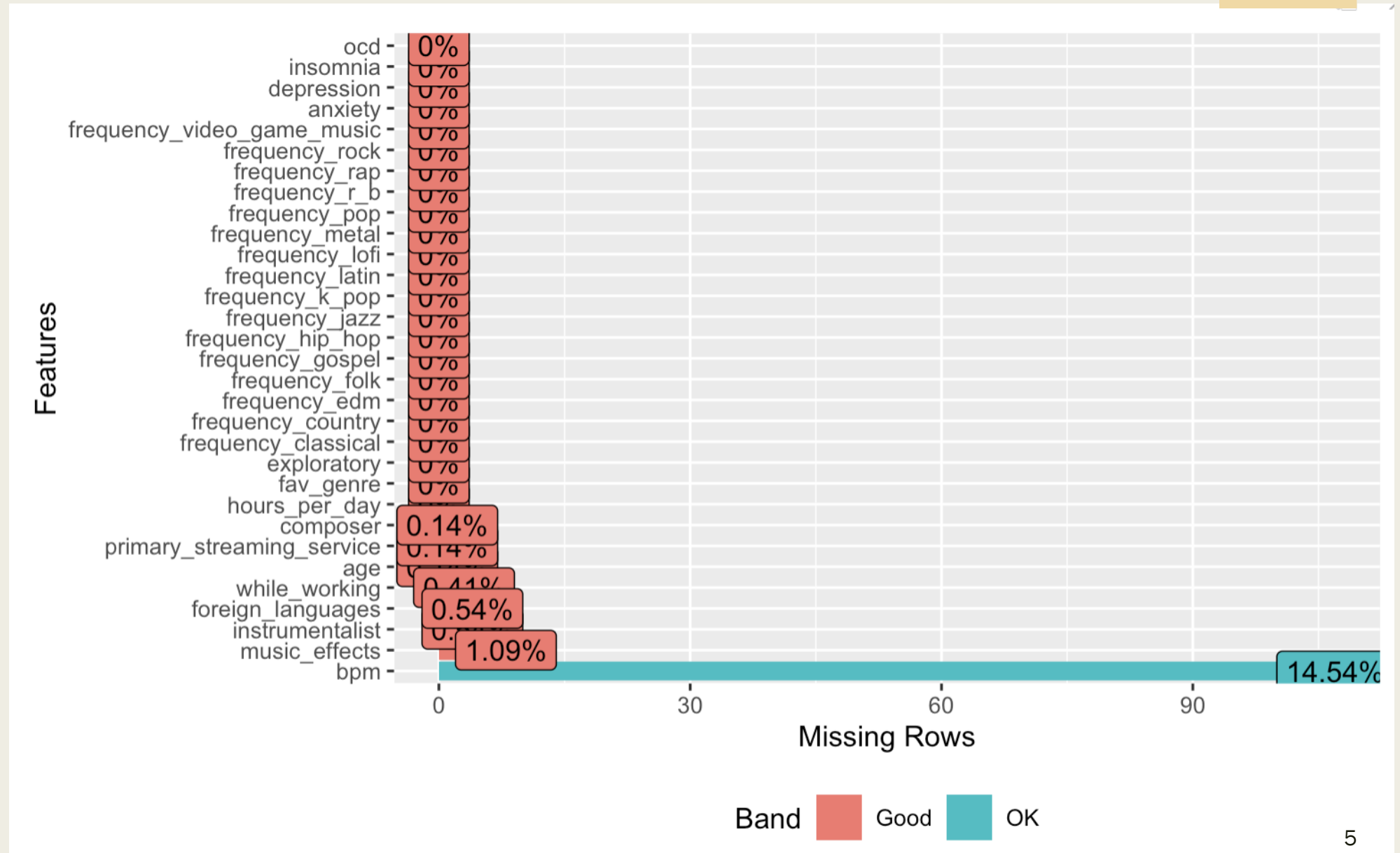
Data Summary			
Number of rows	736		
Number of columns	31	character	24
		numeric	7



Music effects

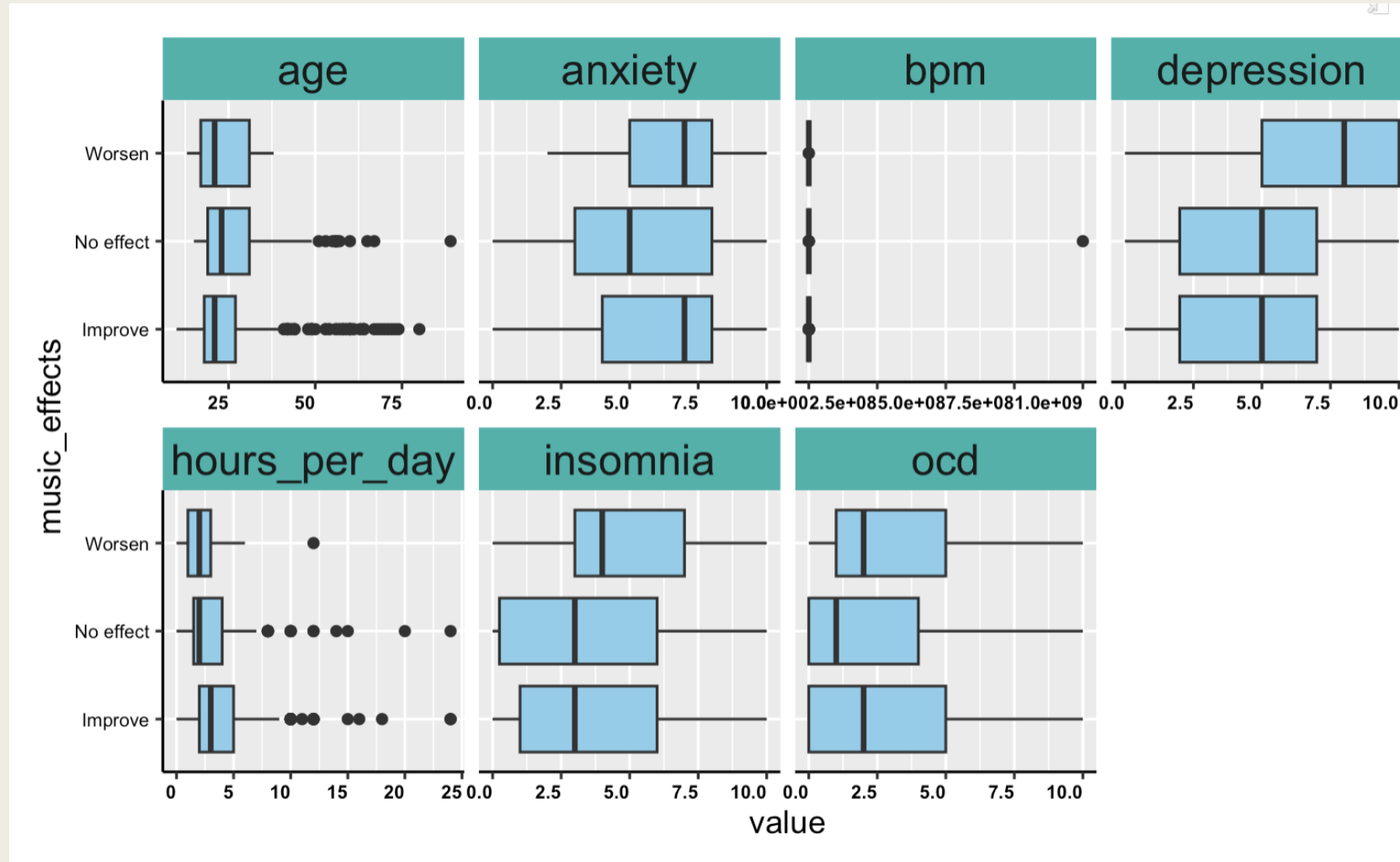
variable	type	Data
Age	numeric	18, 63, 18, 61
Primary streaming service	categorical	Spotify, "Pandora", "YouTube Music"
Hours per day	numeric	3.0, 1.5, 4.0, 2.5,
While working	binary	Yes, "Yes", "No", "Yes"
Instrumentalist	binary	Yes, "No", "No", "No"
Composer	binary	Yes, "No", "No", "Yes"
Exploratory	binary	Yes, "Yes", "No", "Yes"
Foreign languages	binary	Yes, "No", "Yes", "Yes"
BPM	numeric	156, 119, 132, 84, 107, 86
Frequency [Classical]	nominal	Rarely, "Sometimes", "Never"
Frequency [Country]	nominal	Rarely, "Sometimes", "Never"
Frequency [EDM]	nominal	Rarely, "Sometimes", "Never"
Frequency [EDM]	nominal	Rarely, "Sometimes", "Never"
Frequency [Folk]	nominal	Rarely, "Sometimes", "Never"
Frequency [Gospel]	nominal	Rarely, "Sometimes", "Never"
Frequency [Hip hop]	nominal	Rarely, "Sometimes", "Never"
Frequency [Jazz]	nominal	Rarely, "Sometimes", "Never"
Frequency [K pop]	nominal	Rarely, "Sometimes", "Never"
Frequency [Latin]	nominal	Rarely, "Sometimes", "Never"
Frequency [Lofi]	nominal	Rarely, "Sometimes", "Never"
Frequency [Metal]	nominal	Rarely, "Sometimes", "Never"
Frequency [Pop]	nominal	Rarely, "Sometimes", "Never"
Frequency [R&B]	nominal	Rarely, "Sometimes", "Never"
Frequency [Rap]	nominal	Rarely, "Sometimes", "Never"
Frequency [Rock]	nominal	Rarely, "Sometimes", "Never"
Frequency [Video game music]	nominal	Rarely, "Sometimes", "Never"
Anxiety	ordinal	3, 7, 7, 9, 7, 8, 4
Depression	ordinal	0, 2, 7, 7, 2, 8, 8
Insomnia	ordinal	1, 2, 10, 3, 5, 7, 6
OCD	ordinal	0, 1, 2, 3, 9, 7, 0
Music effects	categorical	No effect, "Improve", "worsen"

Preprocessing 1. Remove missing data : 126 v 5 s



Preprocessing 2. Corrected outliers

6



Preprocessing 3. One-hot encoding

7

```
data = pd.get_dummies(data, drop_first=True)
```

```
Music effects    Primary streaming service_I do not use a streaming service. \
2                0                                                            0
3                1                                                            0
4                1                                                            0
5                1                                                            0
6                1                                                            0
```

```
Primary streaming service_Other streaming service ... \
2                0 ...
3                0 ...
4                0 ...
5                0 ...
6                0 ...
```

```
Frequency [R&B]_Very frequently    Frequency [Rap]_Rarely \
2                0                1
3                0                0
4                1                0
5                1                0
6                0                0
```

```
Frequency [Rap]_Sometimes    Frequency [Rap]_Very frequently \
2                0                0
3                0                0
4                0                1
5                0                1
6                0                0
```

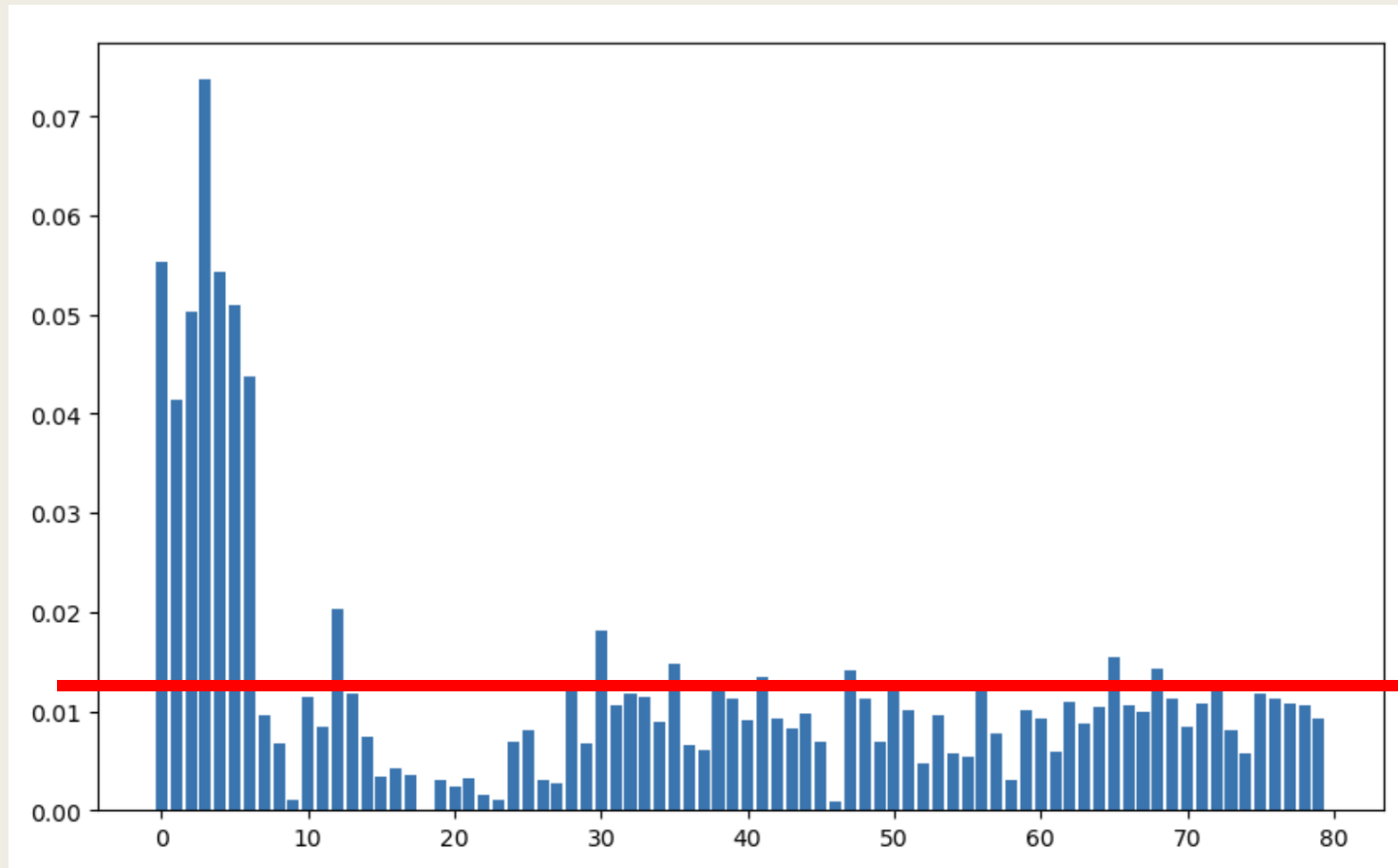
Preprocessing 4. Standardization

```
scaler = StandardScaler()  
X_train = scaler.fit_transform(X_train)  
X_test = scaler.transform(X_test)
```

```
[[-0.48263085 -0.56971086 -0.04569196 ... -0.65140797 -0.59981111  
 -0.42008403]  
 [-0.13634863  0.13309359 -0.04569231 ...  1.53513627 -0.59981111  
 -0.42008403]  
 [ 1.50849193 -1.18466475 -0.0456906  ... -0.65140797 -0.59981111  
 -0.42008403]  
 ...  
 [-0.22291918 -0.21830863 -0.04569082 ... -0.65140797 -0.59981111  
  2.38047614]  
 [-0.22291918  1.18730027 -0.04569143 ...  1.53513627 -0.59981111  
 -0.42008403]  
 [-0.22291918 -0.92111308 -0.04569108 ... -0.65140797 -0.59981111  
 -0.42008403]]
```


Feature Selection by Random Forest

9



Threshold of selection
by mean: 0.0125

Number of selected
features: 17

Threshold of selection
by 75%: 0.01174

Number of selected
features: 20

Python Code for Feature Selection

10

```
forest = RandomForestClassifier(random_state=42)
forest.fit(X_train, y_train)
importances = forest.feature_importances_
print("Threshold of selection by Mean:", np.mean(importances))

# Select features based on importance
selector = SelectFromModel(forest, threshold=np.mean(importances), prefit=True)
X_train_selected = selector.transform(X_train)
X_test_selected = selector.transform(X_test)

# Get selected feature names
selected_features = X.columns[selector.get_support()]
selected_features_list = selected_features.tolist()

# Output the selected features and their number
print("Number of selected features:", len(selected_features_list))
print("Selected features:", selected_features_list)
```

Threshold of selection by Mean: 0.0125

Number of selected features: 17

Selected features: ['Age', 'Hours per day', 'BPM', 'Anxiety', 'Depression', 'Insomnia', 'OCD', 'While working_Yes', 'Exploratory_Yes', 'Frequency [Country]_Rarely', 'Frequency [Folk]_Rarely', 'Frequency [Hip hop]_Rarely', 'Frequency [Jazz]_Rarely', 'Frequency [Latin]_Rarely', 'Frequency [Pop]_Rarely', 'Frequency [R&B]_Rarely', 'Frequency [Rap]_Sometimes']

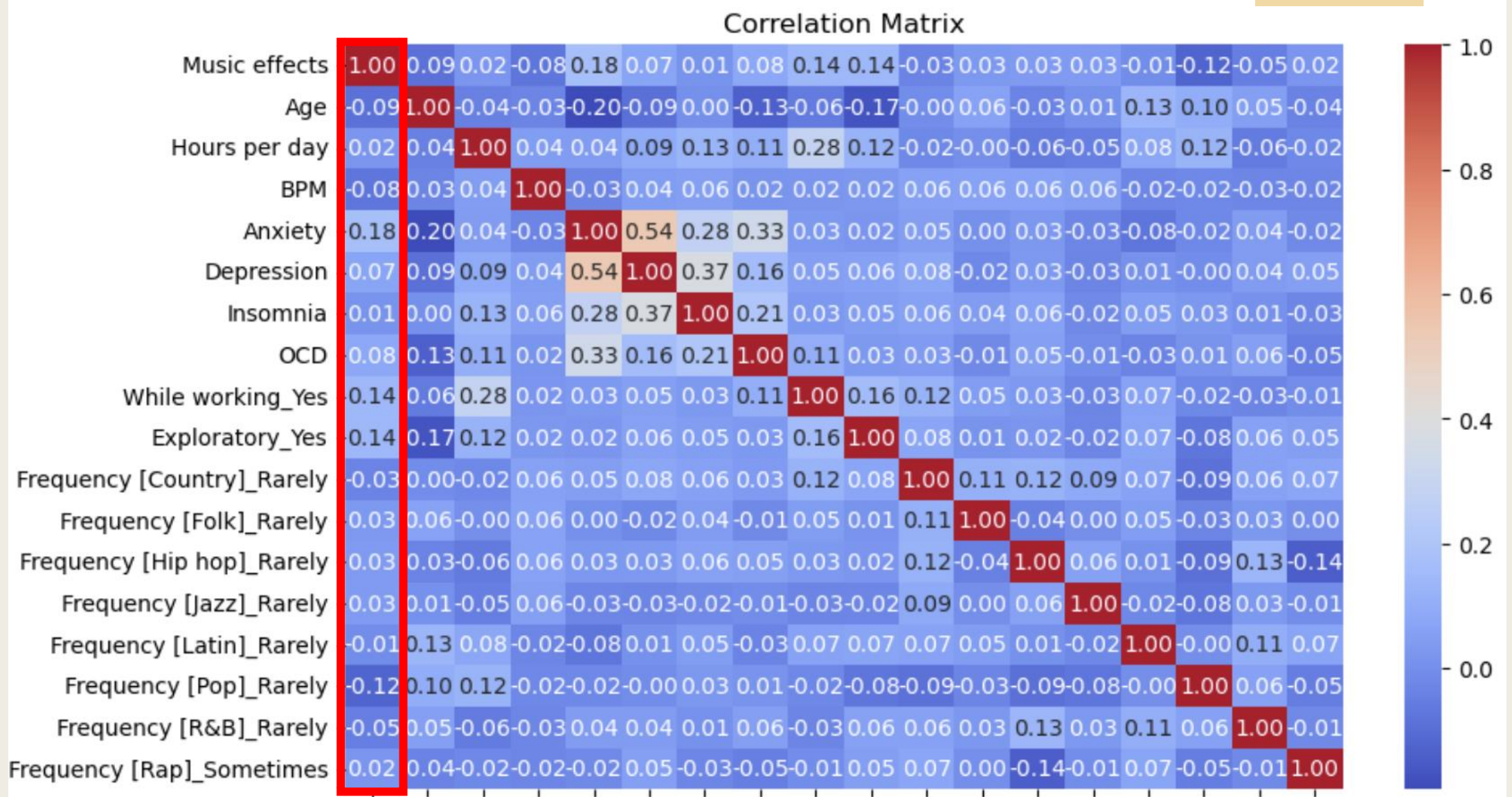
```
selector = SelectFromModel(forest, threshold=np.percentile(importances, 75), prefit=True)
```

Threshold of selection by 75%: 0.011739876330295903

Number of selected features: 20

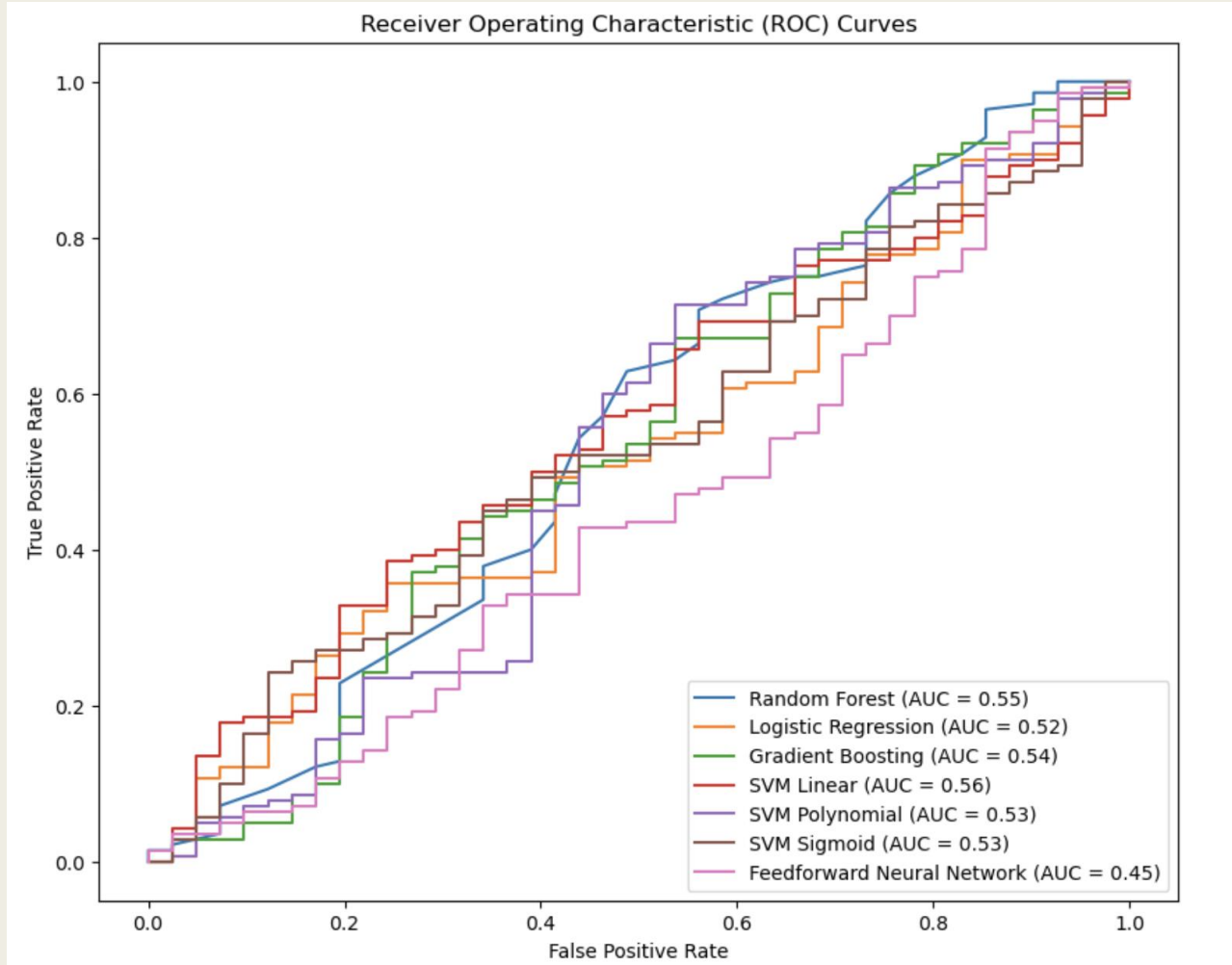
Correlation: Heatmap

11



Python: Model Selection

12



Model	Accuracy
RF Full model	0.76
RF Mean threshold model	0.78
RF 75% threshold model	0.78
Logistic Regression	0.72
Gradient Boosting	0.75
SVM Linear	0.77
SVM Polynomial	0.77
SVM Sigmoid	0.77
FNN	0.65

Python Code for Model Selection

13

```
full_forest = RandomForestClassifier(random_state=42)

log_reg = LogisticRegression(random_state=42, max_iter=1000)
gb = GradientBoostingClassifier(random_state=42)
svm_linear = SVC(kernel='linear', probability=True, random_state=42)
svm_poly = SVC(kernel='poly', probability=True, random_state=42)
svm_sigmoid = SVC(kernel='sigmoid', probability=True, random_state=42)
fnn = MLPClassifier(random_state=42, max_iter=1000)

classifiers = {

    "Random Forest": mean_forest,

    "Logistic Regression": log_reg,
    "Gradient Boosting": gb,
    "SVM Linear": svm_linear,
    "SVM Polynomial": svm_poly,
    "SVM Sigmoid": svm_sigmoid,
    "Feedforward Neural Network": fnn
}

fig, ax = plt.subplots(figsize=(10, 8))

for name, clf in classifiers.items():
    clf.fit(X_train, y_train)
    y_pred = clf.predict(X_test)
    y_proba = clf.predict_proba(X_test)[:, 1] if hasattr(clf, "predict_proba") else clf.decision_function(X_test)
    fpr, tpr, _ = roc_curve(y_test, y_proba)
    roc_auc = auc(fpr, tpr)
    RocCurveDisplay(fpr=fpr, tpr=tpr, roc_auc=roc_auc, estimator_name=name).plot(ax=ax)
    print(f"{name} Classification Report:")
    print(classification_report(y_test, y_pred))
    #print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred), "\n")

plt.title('Receiver Operating Characteristic (ROC) Curves')
plt.show()
```

	precision	recall	f1-score	support
0	0.19	0.17	0.18	41
1	0.76	0.79	0.77	140
accuracy			0.65	181
macro avg	0.48	0.48	0.48	181
weighted avg	0.63	0.65	0.64	181

Python Output for Model Selection

14

Random Forest Classification Report:

	precision	recall	f1-score	support
0	0.60	0.07	0.13	41
1	0.78	0.99	0.87	140
accuracy			0.78	181
macro avg	0.69	0.53	0.50	181
weighted avg	0.74	0.78	0.71	181

Logistic Regression Classification Report:

	precision	recall	f1-score	support
0	0.30	0.17	0.22	41
1	0.78	0.89	0.83	140
accuracy			0.72	181
macro avg	0.54	0.53	0.53	181
weighted avg	0.68	0.72	0.69	181

Gradient Boosting Classification Report:

	precision	recall	f1-score	support
0	0.39	0.17	0.24	41
1	0.79	0.92	0.85	140
accuracy			0.75	181
macro avg	0.59	0.55	0.54	181
weighted avg	0.70	0.75	0.71	181

SVM Linear Classification Report:

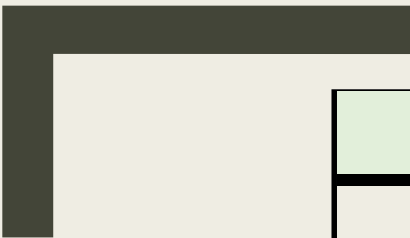
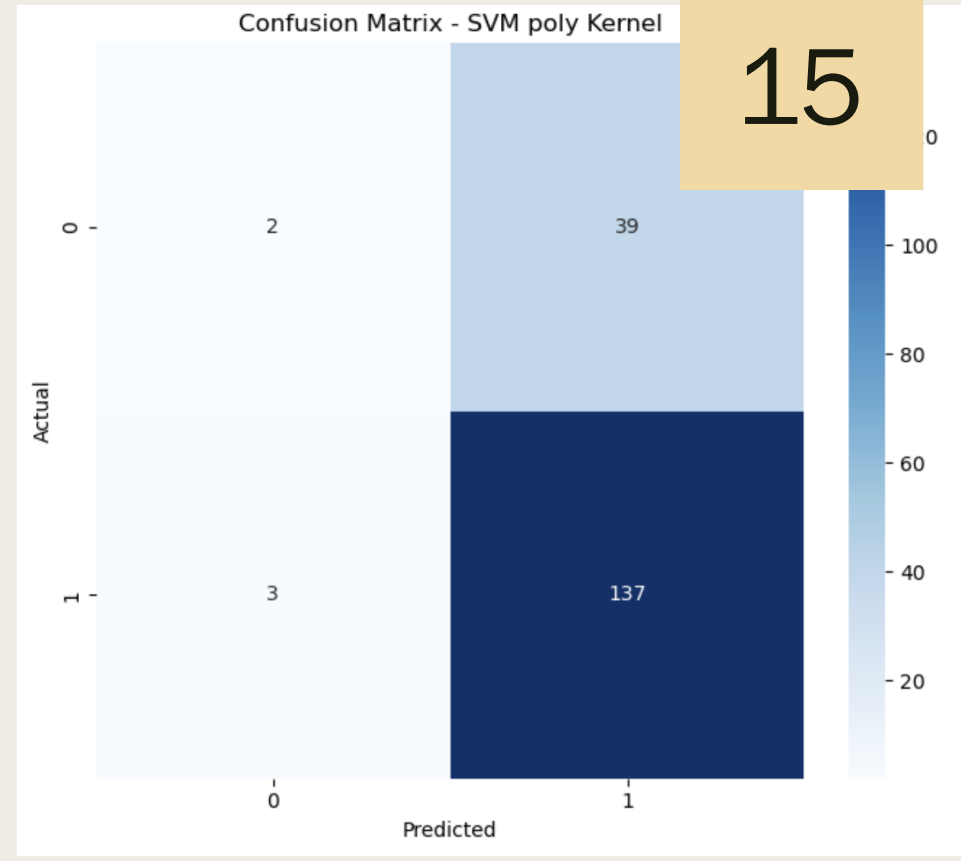
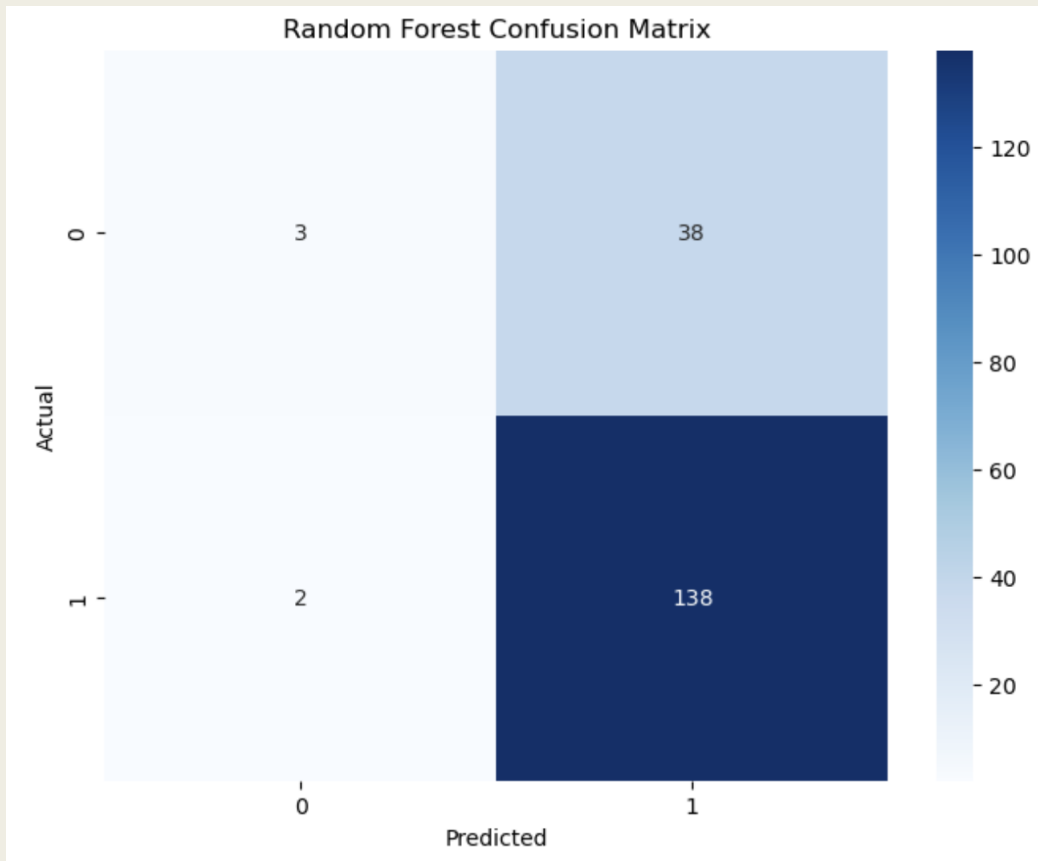
	precision	recall	f1-score	support
0	0.00	0.00	0.00	41
1	0.77	1.00	0.87	140
accuracy			0.77	181
macro avg	0.39	0.50	0.44	181
weighted avg	0.60	0.77	0.67	181

SVM Polynomial Classification Report:

	precision	recall	f1-score	support
0	0.40	0.05	0.09	41
1	0.78	0.98	0.87	140
accuracy			0.77	181
macro avg	0.59	0.51	0.48	181
weighted avg	0.69	0.77	0.69	181

SVM Sigmoid Classification Report:

	precision	recall	f1-score	support
0	0.00	0.00	0.00	41
1	0.77	1.00	0.87	140
accuracy			0.77	181
macro avg	0.39	0.50	0.44	181
weighted avg	0.60	0.77	0.67 ₁₄	181



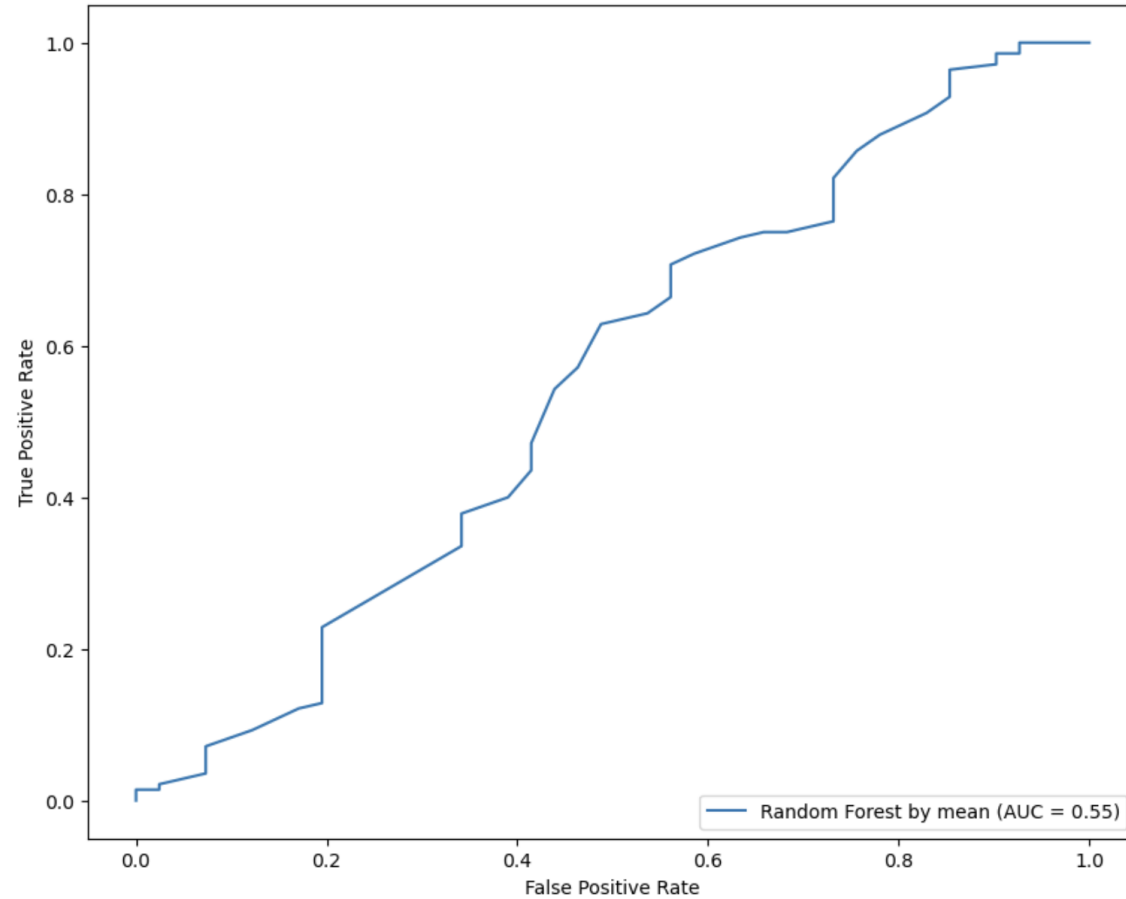
Random Forest
ACCURACY
0.77901

SVM Polynomial
ACCURACY
0.768

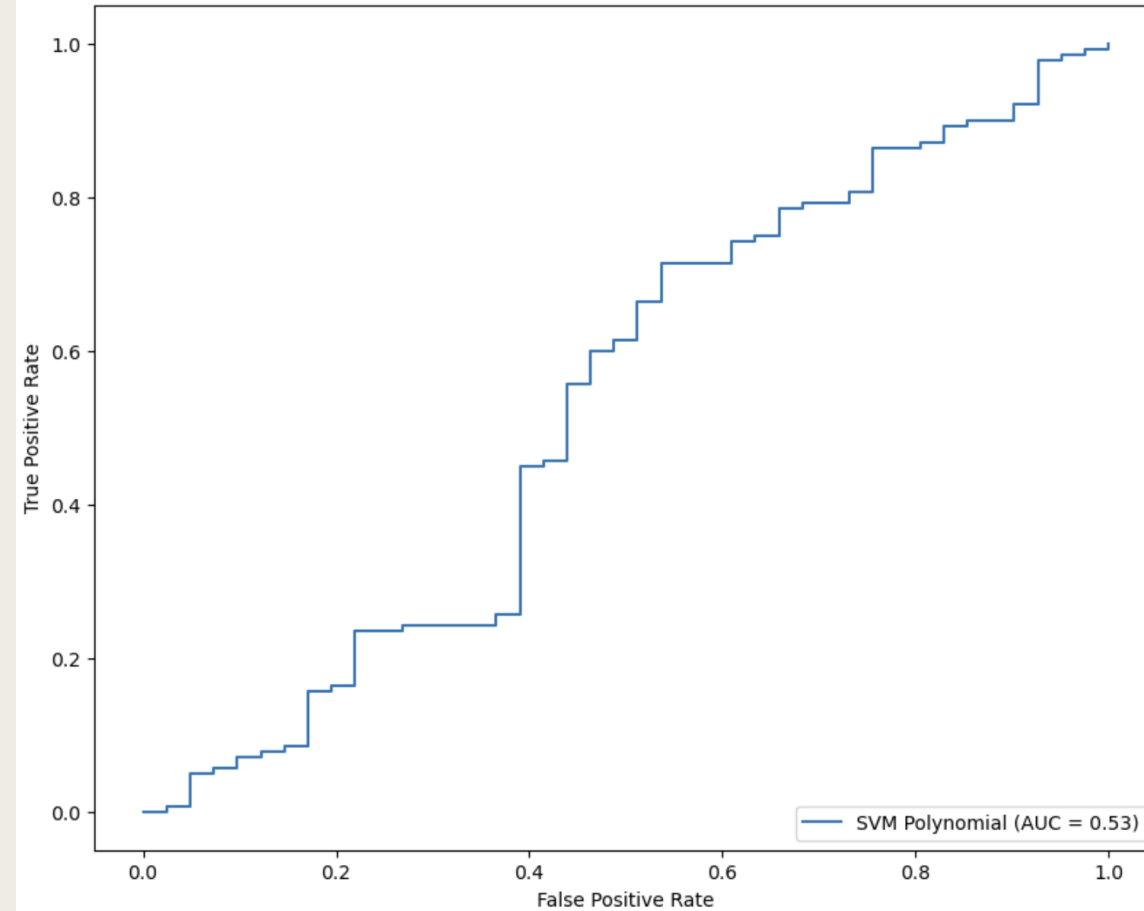


ROC Curves

Random Forest model (ROC) Curves



SVM polynomial model (ROC) Curves



Python code for ROC curve and Outco 17

```
classifiers = {
    "SVM Polynomial": svm_poly,
}

fig, ax = plt.subplots(figsize=(10, 8))

for name, clf in classifiers.items():
    clf.fit(X_train, y_train)
    y_pred = clf.predict(X_test)
    y_proba = clf.predict_proba(X_test)[:, 1] if hasattr(clf, "predict_proba") else clf.decision_function(X_test)
    fpr, tpr, _ = roc_curve(y_test, y_proba)
    roc_auc = auc(fpr, tpr)
    RocCurveDisplay(fpr=fpr, tpr=tpr, roc_auc=roc_auc, estimator_name=name).plot(ax=ax)
    print(f"{name} Classification Report:")
    print(classification_report(y_test, y_pred))
    #print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred), "\n")

plt.title('SVM polynomial model (ROC) Curves')
plt.show()
```

SVM Polynomial Classification Report:

	precision	recall	f1-score	support
0	0.40	0.05	0.09	41
1	0.78	0.98	0.87	140
accuracy			0.77	181
macro avg	0.59	0.51	0.48	181
weighted avg	0.69	0.77	0.69	181

Random Forest by Mean Classification Report:

	precision	recall	f1-score	support
0	0.60	0.07	0.13	41
1	0.78	0.99	0.87	140
accuracy			0.78	181
macro avg	0.69	0.53	0.50	181
weighted avg	0.74	0.78	0.71	181

R: Random Forest

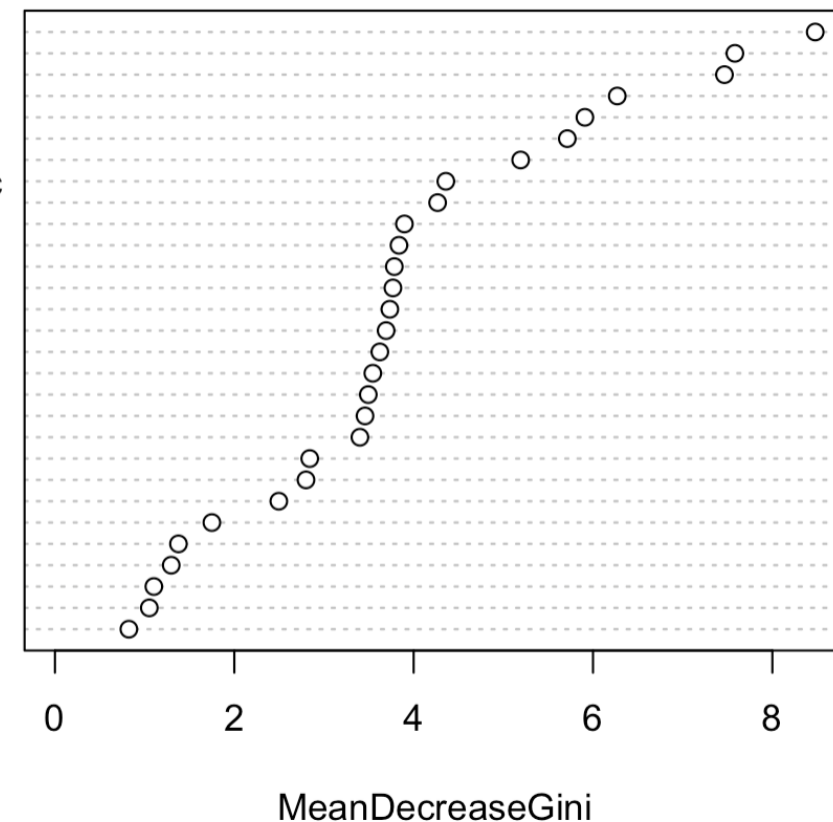
ACCURACY

0.7375415

		TRUE	
		Improve	No effect
PREDICT	Improve	221	74
	No effect	5	1

bpm
 depression
 age
 anxiety
 insomnia
 ocd
 hours_per_day
 frequency-video_game_music
 frequency-classical
 frequency-pop
 frequency-rap
 frequency-metal
 frequency-hip_hop
 frequency-rock
 frequency-lofi
 primary_streaming_service
 frequency-edm
 frequency-folk
 frequency-jazz
 frequency-kpop
 frequency-country
 frequency-latin
 frequency-gospel
 while_working
 exploratory
 foreign_languages
 instrumentalist
 composer

Variable Importance (Random Forest)



R code for Random Forest

19

```
n=nrow(df_m)
prop = 0.5
set.seed(123)
train_id = sample(1:n, size=round(n*prop), replace=FALSE)
test_id = (1:n)[-which(1:n %in% train_id)]

train_set=df_m[train_id,]
test_set=df_m[test_id,]

p=ncol(train_set)-1
set.seed(123)

rf_fit <- randomForest(factor(music_effects) ~ ., data = train_set, mtry = round(sqrt(p))
                        importance = TRUE)

yhat.test_rf=predict(rf_fit, test_set, type = "class")
tb_rf = table(pred=yhat.test_rf, true=test_set$music_effects)
tb_rf
# View the confusion matrix
tb_rf_acc=(tb_rf[1,1]+tb_rf[2,2])/sum(tb_rf)
tb_rf_acc

importance(rf_fit, type=2)

varImpPlot(rf_fit, main = "Variable Importance (Random Forest)", type=2)
```

R: SVM Polynomial

ACCURACY
0.74444

		TRUE	
		Improve	No effect
PREDICT	Improve	123	37
	No effect	9	11

```
svm(formula = music_effects ~ ., data = train_set, kernel = "polynomial", cost = 0.001,
    scale = FALSE)
```

Parameters:

SVM-Type: eps-regression

SVM-Kernel: polynomial

cost: 0.001

degree: 3

gamma: 0.0625

coef.0: 0

epsilon: 0.1

Number of Support Vectors: 230

R code for SVM Polynomial

21

```
library(e1071)
set.seed(1)
tune_svm_poly = tune(svm, music_effects ~., data=train_set, kernel="polynomial",
                     ranges = list(cost=10^seq(-3,2,length.out=6)))
summary(tune_svm_poly)
svm_fit_poly = svm(music_effects ~., data = train_set, kernel="polynomial",
                  cost=0.001, scale=FALSE)
summary(svm_fit_poly)
yhat_test_poly=predict(svm_fit_poly, test_set)
#yhat_test_poly
binary_predictions <- ifelse(yhat_test_poly > 0.899855, "1", "0")
tb_svm_poly=table(pred=binary_predictions, true = test_set$music_effects)
tb_svm_poly
tb_svm_poly_acc=(tb_svm_poly[1,1]+tb_svm_poly[2,2])/sum(tb_svm_poly)
tb_svm_poly_acc
```

SAS:Random Forest

ACCURACY

0.791667

22

Variable	Number of Rules	Gini			OOB Margin
primary_streaming_service	0	0.000000	0	0.000000	0.000000
frequency_rock	51	0.002108	-0.000000	0.000000	0.00017
frequency_gospel	35	0.001696	-0.00237	0.003392	-0.00123
frequency_hip_hop	117	0.006464	-0.00271	0.012928	0.00334
frequency_k_pop	65	0.002841	-0.00309	0.005682	-0.00132
frequency_video_game_music	72	0.004290	-0.00352	0.008580	0.00162
frequency_pop	77	0.004249	-0.00442	0.008497	0.00014
frequency_latin	77	0.003899	-0.00461	0.007799	-0.00055
frequency_r_b	82	0.004809	-0.00494	0.009618	-0.00075
frequency_country	73	0.004064	-0.00530	0.008128	-0.00112
frequency_rap	68	0.003436	-0.00551	0.006872	-0.00142
frequency_edm	81	0.003584	-0.00559	0.007167	-0.00275
frequency_jazz	95	0.003931	-0.00588	0.007863	-0.00305
frequency_folk	109	0.005652	-0.00604	0.011304	0.00014
frequency_metal	89	0.004291	-0.00613	0.008581	-0.00190
frequency_classical	87	0.004025	-0.00685	0.008050	-0.00264
frequency_lofi	111	0.005537	-0.00758	0.011073	-0.00073
anxiety	447	0.029273	-0.00845	0.058546	0.02295
ocd	352	0.019359	-0.01726	0.038718	0.00286
hours_per_day	372	0.020157	-0.02156	0.040315	-0.00056
bpm	411	0.027633	-0.02185	0.055266	0.00313
depression	360	0.020598	-0.02233	0.041196	0.00021
insomnia	370	0.021373	-0.02292	0.042747	-0.00228
age	470	0.029245	-0.02578	0.058491	0.00321

SAS code for

Random Forest

23

```
proc import out=my_data datafile="C:\Users\SowonJung\Desktop\px. replace;
    getnames=yes;
run;
proc surveyselect data=my_data rate=0.8 seed=6132208
    out=my_data_split outall method=srs;
run;
proc hpforest data=my_data_split seed=115607
    maxtrees=60 vars_to_try=4 trainfraction=0.7
    maxdepth=50;
    target music_effects / level=binary;
    input age hours_per_day bpm anxiety depression insomnia ocd / level=interval;
    input primary_streaming_service frequency_classical frequency_country frequency_edm
        frequency_folk frequency_gospel frequency_hip_hop frequency_jazz
        frequency_k_pop frequency_latin frequency_lofi frequency_metal
        frequency_pop frequency_r_b frequency_rap frequency_rock
        frequency_video_game_music / level=nominal;
    partition rolevar=selected(train='1');
    save file='C:\Users\SowonJung\Desktop\random_forest.bin';
run;
data test;
    set my_data_split;
    if selected=0;
run;
proc hp4score data=test;
    id timestamp;
    score file='C:\Users\SowonJung\Desktop\random_forest.bin'
    out=predicted;
run;
data predicted;
    set predicted;
    match=(music_effects=lowercase(I_music_effects));
run;
proc sql;
    select sum(match)/count(*) as accuracy
    from predicted;
quit;
```

SAS: SVM Polynomial

ACCURACY

0.736264



```
data polynomial_kernel;
  set "C:\Users\SowonJung\Desktop\SVM(Poly)\Workspaces\EMWS1\EMSave\em_save_test.sas7bdat";
  match=(music_effects=lowcase(EM_CLASSIFICATION));
run;

proc sql;
  select mean(match) as accuracy
  from polynomial_kernel;
run;
```


Summary of Prediction Accuracy 25

Random Forest

■ Python: 0.77901

■ R: 0.7375415

■ SAS: 0.791667

SVM (Polynomial)

■ Python: 0.768

■ R: 0.744444

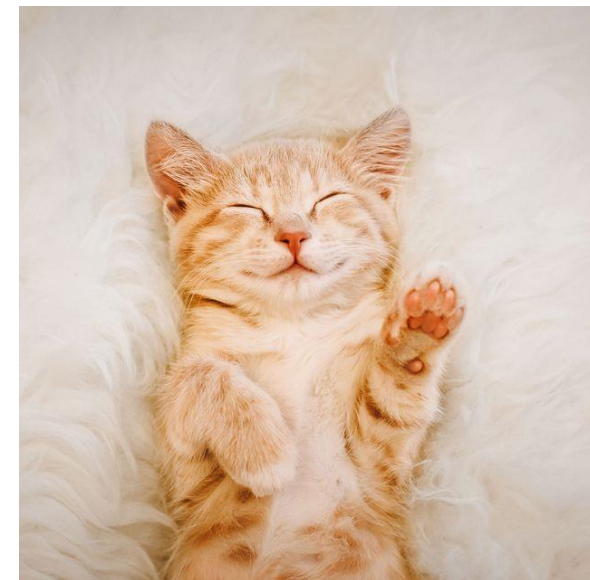
■ SAS: 0.736264

SAS Random Forest has the highest accuracy!



Implication & Future Studies

- High accuracy supports the hypothesis that there is a measurable relationship between music listening habits (like genre preferences) and mental health outcome.
- Music can have a predictable impact on disorders such as anxiety, depression, insomnia, and OCD.
- Feature selection can lead to deeper inquiries into the mechanism through music affects mental health.
- Further research can explore how these features interact with psychological process



THANK YOU!