In [ ]:
```
This document shows the Julia programming language code
for the following medical data analysis research project.
```

In [ ]:
```
Tinnitus data classification prediction accuracy results comparison
on the testing dataset.
```

In [1]:
```
import Pkg; Pkg.add("DataFrames")
import Pkg; Pkg.add("CSV")
using CSV, DataFrames, Plots
```

```
    Updating registry at `C:\Users\zizhe\.julia\registries\General.toml`
   Resolving package versions...
   Installed OpenJpeg_jll ──── v2.4.0+0
   Installed ImageMagick_jll ─ v6.9.12+4
   Installed LittleCMS_jll ─── v2.12.0+0
  No Changes to `C:\Users\zizhe\.julia\environments\v1.7\Project.toml`
    Updating `C:\Users\zizhe\.julia\environments\v1.7\Manifest.toml`
  [c73af94c] ↑ ImageMagick_jll v6.9.12+3 ⇒ v6.9.12+4
  [d3a379c0] + LittleCMS_jll v2.12.0+0
  [643b3616] + OpenJpeg_jll v2.4.0+0
Precompiling project...
  ✓ LittleCMS_jll
  ✓ OpenJpeg_jll
  ✓ ImageMagick_jll
  ✓ ImageMagick
  ✓ Images
  5 dependencies successfully precompiled in 27 seconds (370 already precompiled)
   Resolving package versions...
  No Changes to `C:\Users\zizhe\.julia\environments\v1.7\Project.toml`
  No Changes to `C:\Users\zizhe\.julia\environments\v1.7\Manifest.toml`
```

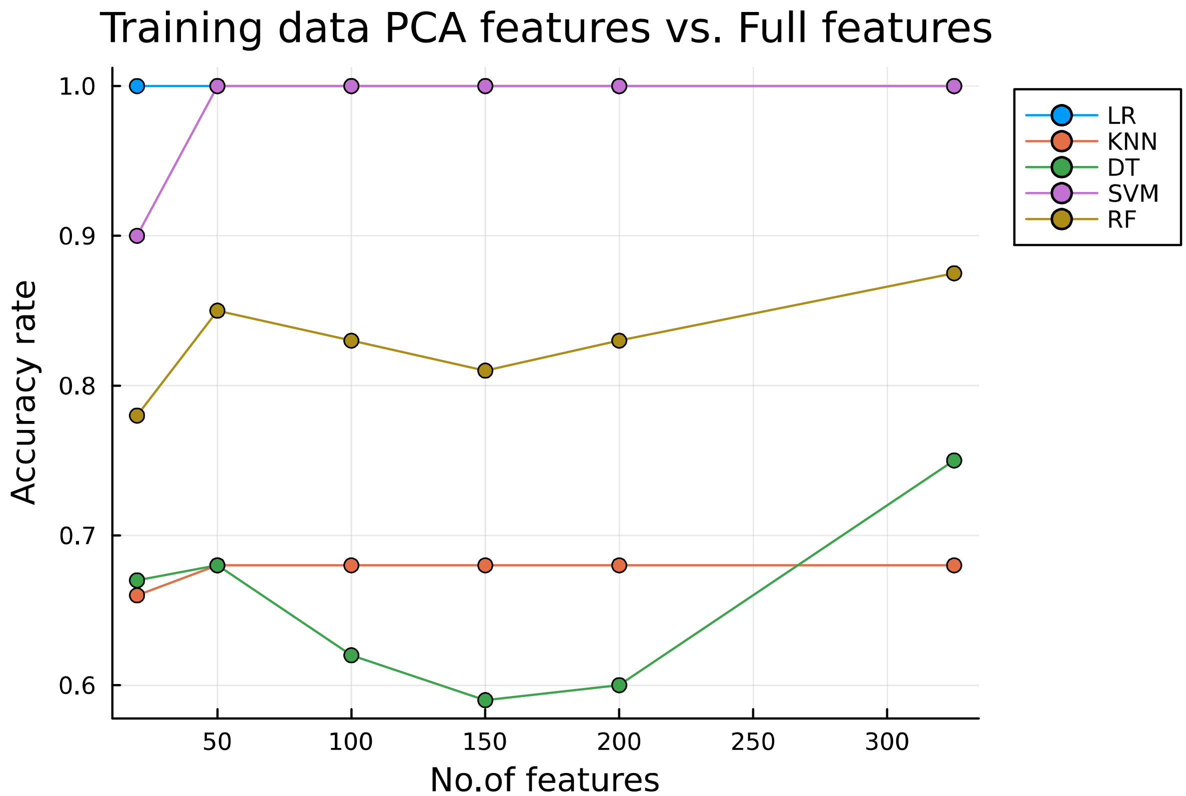
In [2]:
```
# Load the PCA training data result
trPCA = CSV.read("C:/Users/Leo/Downloads/Training data PCA result in csv.csv",
header=true,DataFrame)
```

Out[2]: 6 rows × 7 columns

	Connectivity Features	Number of Features	LR	KNN	DT	SVM	RF
	String	Int64	Int64	Float64	Float64	Float64	Float64
1	Full features	325	1	0.68	0.75	1.0	0.875
2	PCA feature extraction	200	1	0.68	0.6	1.0	0.83
3	PCA feature extraction	150	1	0.68	0.59	1.0	0.81
4	PCA feature extraction	100	1	0.68	0.62	1.0	0.83
5	PCA feature extraction	50	1	0.68	0.68	1.0	0.85
6	PCA feature extraction	20	1	0.66	0.67	0.9	0.78

In [3]:
```
plot(trPCA[:,2],Matrix(trPCA[:,3:7]),xlabel="No.of features",
ylabel="Accuracy rate",marker=:c,title="Training data PCA features vs. Full features",
legend=:outertopright,label=["LR" "KNN" "DT" "SVM" "RF"])
```

Out[3]:



In [4]:

```
# load the extra tree training data result
trET = CSV.read("C:/Users/Leo/Downloads/Training data extra tree result in csv.csv",
header=true,DataFrame)
```

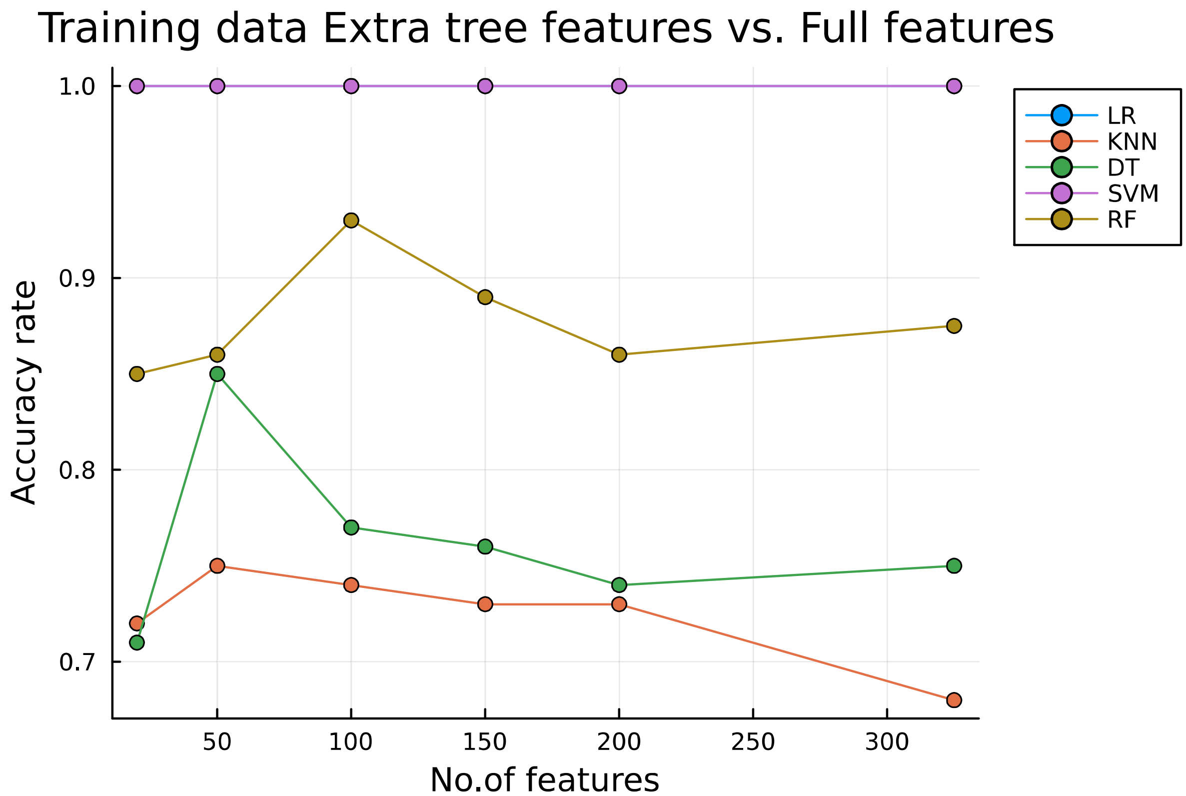
Out[4]: 6 rows × 7 columns

	Connectivity Features	Number of Features	LR	KNN	DT	SVM	RF
	String	Int64	Int64	Float64	Float64	Int64	Float64
1	Full features	325	1	0.68	0.75	1	0.875
2	Extra tree top features	200	1	0.73	0.74	1	0.86
3	Extra tree top features	150	1	0.73	0.76	1	0.89
4	Extra tree top features	100	1	0.74	0.77	1	0.93
5	Extra tree top features	50	1	0.75	0.85	1	0.86
6	Extra tree top features	20	1	0.72	0.71	1	0.85

In [5]:

```
plot(trET[:,2],Matrix(trET[:,3:7]),xlabel="No.of features",ylabel="Accuracy rate",
marker=:c,title="Training data Extra tree features vs. Full features",
legend=:outertopright,label=["LR" "KNN" "DT" "SVM" "RF"])
```

Out[5]:



In [6]:

```
# load the PCA testing data result
tePCA  = CSV.read("C:/Users/Leo/Downloads/Testing data PCA result in csv.csv",
header=true,DataFrame)
```

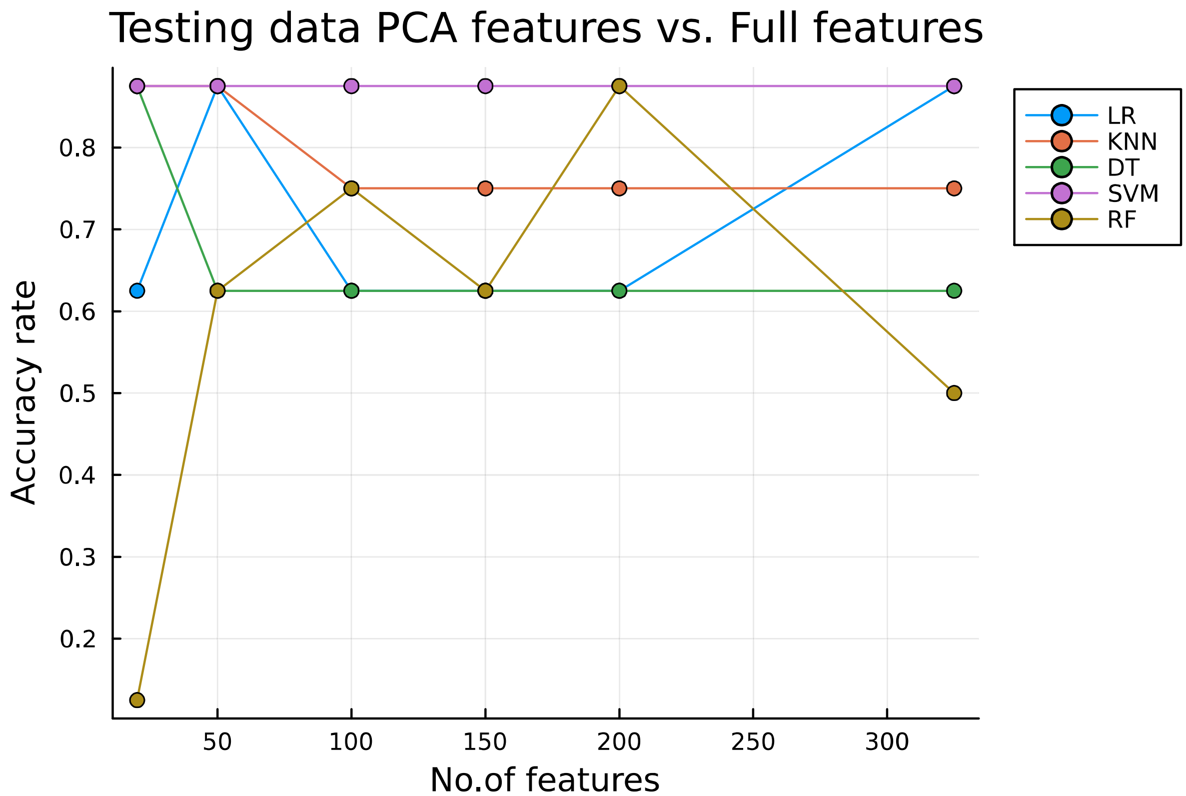
Out[6]: 6 rows × 7 columns (omitted printing of 1 columns)

	Connectivity Features	Number of Features	LR	KNN	DT	SVM
	String	Int64	Float64	Float64	Float64	Float64
1	Full features	325	0.875	0.75	0.625	0.875
2	PCA feature extraction	200	0.625	0.75	0.625	0.875
3	PCA feature extraction	150	0.625	0.75	0.625	0.875
4	PCA feature extraction	100	0.625	0.75	0.625	0.875
5	PCA feature extraction	50	0.875	0.875	0.625	0.875
6	PCA feature extraction	20	0.625	0.875	0.875	0.875

In [7]:

```
plot(tePCA[:,2],Matrix(tePCA[:,3:7]),xlabel="No.of features",ylabel="Accuracy rate",
marker=:c,title="Testing data PCA features vs. Full features",
legend=:outertopright,label=["LR" "KNN" "DT" "SVM" "RF"])
```

Out[7]:



In [8]:

```
# load the extra tree training data result
teET = CSV.read("C:/Users/Leo/Downloads/Testing data extra tree result in csv.csv",
header=true,DataFrame)
```

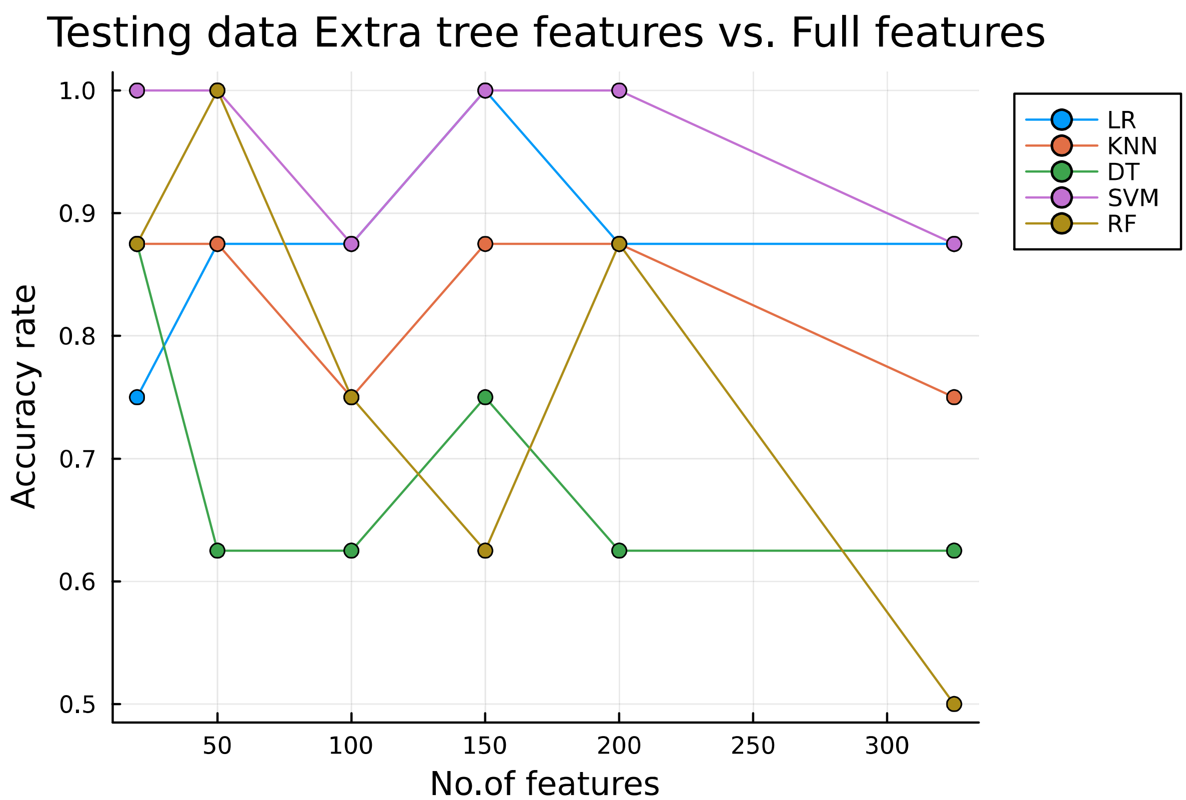
Out[8]: 6 rows × 7 columns (omitted printing of 1 columns)

	Connectivity Features	Number of Features	LR	KNN	DT	SVM
	String	Int64	Float64	Float64	Float64	Float64
1	Full features	325	0.875	0.75	0.625	0.875
2	Extra tree top features	200	0.875	0.875	0.625	1.0
3	Extra tree top features	150	1.0	0.875	0.75	1.0
4	Extra tree top features	100	0.875	0.75	0.625	0.875
5	Extra tree top features	50	0.875	0.875	0.625	1.0
6	Extra tree top features	20	0.75	0.875	0.875	1.0

In [9]:

```
plot(teET[:,2],Matrix(teET[:,3:7]),xlabel="No.of features",ylabel="Accuracy rate",
marker=:c,title="Testing data Extra tree features vs. Full features",
legend=:outertopright,label=["LR" "KNN" "DT" "SVM" "RF"])
```

Out[9]:



In [ ]:

In [ ]:

In [10]:

```
# load the full testing data result
te = CSV.read("C:/Users/Leo/Downloads/Testing data result in csv.csv",
header=true,DataFrame)
```

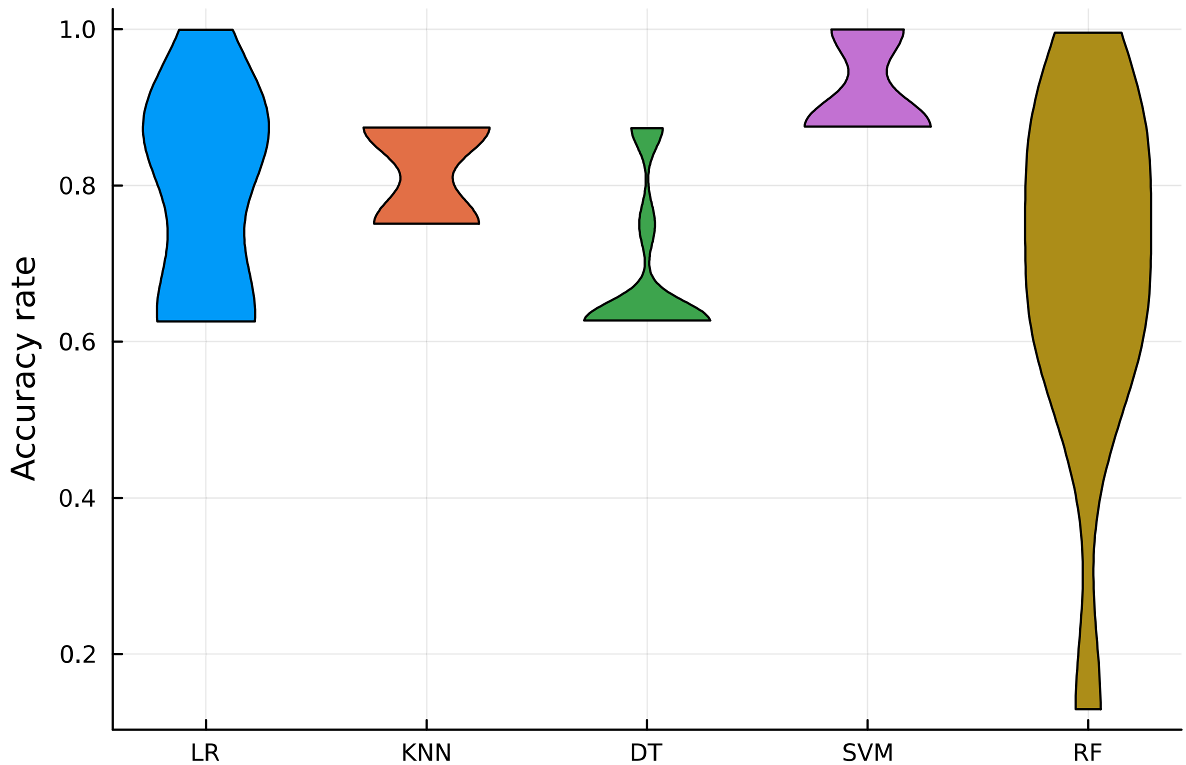
Out[10]: 11 rows × 7 columns (omitted printing of 1 columns)

	Connectivity Features	Number of Features	LR	KNN	DT	SVM
	String	Int64	Float64	Float64	Float64	Float64
1	Full features	325	0.875	0.75	0.625	0.875
2	PCA feature extraction	200	0.625	0.75	0.625	0.875
3	PCA feature extraction	150	0.625	0.75	0.625	0.875
4	PCA feature extraction	100	0.625	0.75	0.625	0.875
5	PCA feature extraction	50	0.875	0.875	0.625	0.875
6	PCA feature extraction	20	0.625	0.875	0.875	0.875
7	Extra tree top features	200	0.875	0.875	0.625	1.0
8	Extra tree top features	150	1.0	0.875	0.75	1.0
9	Extra tree top features	100	0.875	0.75	0.625	0.875
10	Extra tree top features	50	0.875	0.875	0.625	1.0
11	Extra tree top features	20	0.75	0.875	0.875	1.0

In [11]:
```
Pkg.add("StatsPlots"); using StatsPlots
violin(["LR"], te[:,3], label=nothing, ylabel="Accuracy rate")
violin!(["KNN"], te[:,4], label=nothing)
violin!(["DT"], te[:,5], label=nothing)
violin!(["SVM"], te[:,6], label=nothing)
violin!(["RF"], te[:,7], label=nothing)
```

```
   Resolving package versions...
  No Changes to `C:\Users\zizhe\.julia\environments\v1.7\Project.toml`
  No Changes to `C:\Users\zizhe\.julia\environments\v1.7\Manifest.toml`
```

Out[11]:



In [12]:
```
using Statistics
mean(te[:,3]),std(te[:,3])
```

Out[12]:
```
(0.7840909090909091, 0.13796409282523808)
```

In [13]:
```
mean(te[:,4]),std(te[:,4])
```

Out[13]:
```
(0.8181818181818182, 0.06527912098338667)
```

In [14]:
```
mean(te[:,5]),std(te[:,5])
```

Out[14]:
```
(0.6818181818181818, 0.10252494153309054)
```

In [15]:
```
mean(te[:,6]),std(te[:,6])
```

Out[15]:
```
(0.9204545454545454, 0.06306562238868912)
```

In [16]:
```
mean(te[:,7]),std(te[:,7])
```

Out[16]:
```
(0.6931818181818182, 0.239554510782752)
```

In [17]:

```julia
# as their variance are different, so use UnequalVarianceTTest
import Pkg; Pkg.add("HypothesisTests")
using HypothesisTests

UnequalVarianceTTest(te[:,3], te[:,4])
```

```
   Resolving package versions...
  No Changes to `C:\Users\zizhe\.julia\environments\v1.7\Project.toml`
  No Changes to `C:\Users\zizhe\.julia\environments\v1.7\Manifest.toml`
```

Out[17]:

```
Two sample t-test (unequal variance)
------------------------------------
Population details:
    parameter of interest:   Mean difference
    value under h_0:         0
    point estimate:          -0.0340909
    95% confidence interval: (-0.1326, 0.06444)

Test summary:
    outcome with 95% confidence: fail to reject h_0
    two-sided p-value:           0.4708

Details:
    number of observations:   [11,11]
    t-statistic:              -0.740797197487194
    degrees of freedom:       14.263894781501907
    empirical standard error: 0.04601921984390133
```

In [18]:

```julia
# statistical test between LR and KNN
MannWhitneyUTest(te[:,3], te[:,4])
```

Out[18]:

```
Approximate Mann-Whitney U test
-------------------------------
Population details:
    parameter of interest:   Location parameter (pseudomedian)
    value under h_0:         0
    point estimate:          0.0

Test summary:
    outcome with 95% confidence: fail to reject h_0
    two-sided p-value:           0.6435

Details:
    number of observations in each group: [11, 11]
    Mann-Whitney-U statistic:             53.5
    rank sums:                            [119.5, 133.5]
    adjustment for ties:                  1590.0
    normal approximation (μ, σ):          (-7.0, 14.0433)
```

In [19]:

```
Above test all got "fail to reject h_0", since the null hyposis
is the means of the two groups are statistically identical,
the result fail to reject h_0" means there is at least 95%
confidence that accuracy rates of
logistic regression classifier and KNN are statistically identical!
Both classifier are equally good/bad
```

In [20]:

```julia
# statistical test between decision tree and KNN
UnequalVarianceTTest(te[:,4], te[:,5])
```

Out[20]:
```
Two sample t-test (unequal variance)
------------------------------------
Population details:
    parameter of interest:   Mean difference
    value under h_0:         0
    point estimate:          0.136364
    95% confidence interval: (0.05903, 0.2137)

Test summary:
    outcome with 95% confidence: reject h_0
    two-sided p-value:           0.0017

Details:
    number of observations:   [11,11]
    t-statistic:              3.721042037676257
    degrees of freedom:       16.963613550815555
    empirical standard error: 0.03664662612862977
```

In [21]:
```
MannWhitneyUTest(te[:,4], te[:,5])
```

Out[21]:
```
Approximate Mann-Whitney U test
-------------------------------
Population details:
    parameter of interest:   Location parameter (pseudomedian)
    value under h_0:         0
    point estimate:          0.25

Test summary:
    outcome with 95% confidence: reject h_0
    two-sided p-value:           0.0038

Details:
    number of observations in each group: [11, 11]
    Mann-Whitney-U statistic:             102.5
    rank sums:                            [168.5, 84.5]
    adjustment for ties:                  1218.0
    normal approximation (μ, σ):          (42.0, 14.3295)
```

In [22]:
```
Above test all got "reject h_0", since the null hyposis is
the means of the two groups are statistically identical,
the result reject h_0" means there is at least 95% confidence
that accuracy rates of decision tree classifier and KNN are statistically different!
from the plot, we can see the accuracy rate of KNN is better
```

In [23]:
```
# statistical test between SVM and KNN
UnequalVarianceTTest(te[:,4], te[:,6])
```

Out[23]:
```
Two sample t-test (unequal variance)
------------------------------------
Population details:
    parameter of interest:   Mean difference
    value under h_0:         0
    point estimate:          -0.102273
    95% confidence interval: (-0.1594, -0.04518)

Test summary:
    outcome with 95% confidence: reject h_0
    two-sided p-value:           0.0013

Details:
    number of observations:   [11,11]
    t-statistic:              -3.7370465934182957
    degrees of freedom:       19.976247030878863
    empirical standard error: 0.027367260406346124
```

In [24]:
```
MannWhitneyUTest(te[:,4], te[:,6])
```

Out[24]:
```
Approximate Mann-Whitney U test
-------------------------------
Population details:
    parameter of interest:   Location parameter (pseudomedian)
    value under h_0:         0
    point estimate:          0.0

Test summary:
    outcome with 95% confidence: reject h_0
    two-sided p-value:           0.0037

Details:
    number of observations in each group: [11, 11]
    Mann-Whitney-U statistic:             21.0
    rank sums:                            [87.0, 166.0]
    adjustment for ties:                  2364.0
    normal approximation (μ, σ):          (-39.5, 13.4284)
```

In [25]:
```
Above test all got "reject h_0", since the null hyposis is
the means of the two groups are statistically identical,
the result reject h_0" means there is at least 95% confidence
that accuracy rates of SVM classifier and KNN are statistically different!
from the plot, we can see the accuracy rate of SVM is better
```

In [26]:
```
# statistical test between SVM and random forest
UnequalVarianceTTest(te[:,6], te[:,7])
```

Out[26]:
```
Two sample t-test (unequal variance)
------------------------------------
Population details:
    parameter of interest:   Mean difference
    value under h_0:         0
    point estimate:          0.227273
    95% confidence interval: (0.06355, 0.391)

Test summary:
    outcome with 95% confidence: reject h_0
    two-sided p-value:           0.0108

Details:
    number of observations:   [11,11]
    t-statistic:              3.042903097250921
    degrees of freedom:       11.379512195121952
    empirical standard error: 0.0746894396597954
```

In [27]:
```
MannWhitneyUTest(te[:,6], te[:,7])
```

Out[27]:
```
Approximate Mann-Whitney U test
-------------------------------
Population details:
    parameter of interest:   Location parameter (pseudomedian)
    value under h_0:         0
    point estimate:          0.125

Test summary:
    outcome with 95% confidence: reject h_0
    two-sided p-value:           0.0049

Details:
    number of observations in each group: [11, 11]
    Mann-Whitney-U statistic:             101.5
    rank sums:                            [167.5, 85.5]
    adjustment for ties:                  1140.0
    normal approximation (μ, σ):          (41.0, 14.3887)
```

In [28]:
```
Above test all got "reject h_0", since the null hyposis is
the means of the two groups are statistically identical,
the result reject h_0" means there is at least
95% confidence that accuracy rates of SVM
classifier and random forest are statistically different!
 from the plot, we can see the accuracy rate of SVM is better
```

In [ ]:
```
After a few T test and MannWhitneyUTest,
we can see the accuracy rate of SVM classifier is
statistically different from the other classifiers
from the plot, we can see SVM is the classifier
that has the highest prediction accuracy rate
```