Comp1204 Unix Coursework

Callum Anderson
crea1g15

February 24, 2016

Contents

1 Scripts

1.1 Review Counting

This script searches a folder (given as an argument) for .dat files and counts the number of reviews in each file.

```
1  #!/bin/bash
2
3  for filename in $1/*.dat; do
4    name=$(echo $filename | cut -d/ -f2 | cut -d. -f1)
5    echo "$name $(grep Author $filename | wc -l)"
6  done | sort -t" " -k2nr
```

countreviews.sh

Line 4 removes the extension and directory path from the filename, so the hotel name itself can be printed next to the review count.

The for loop goes through each file, and then pipes a string of format "$hotelname $reviewcount" into sort, which prints out the sorted list of hotels.

1.2 Average Review Score

This script runs through all .dat files in the given folder and finds the average review score for each hotel, then prints the sorted list to the terminal.

```
1  #!/bin/bash
2
3  for filename in $1/*.dat; do
4    echo $filename
5    name=$(echo $filename | cut -d/ -f2 | cut -d. -f1)
6    sum=0
7    count=0
8    for i in $(grep "<Overall>" $filename | sed "s/<Overall>//g"); do
9      count=$(($count+1))
10     i=`echo $i | tr -d $'\r'`
11     sum=$((sum+i))
12   done
13   avg=`echo "scale=2; $sum/$count" | bc`
14   echo "$name $avg"
15 done | sort -t" " -k2nr
```

averagereviews.sh

Line 8 loops through each line in the file containing "¡Overall¿", removing the 'overall' tag from it so it's just a number. Line 10 removes any linebreak characters that still remain.

1.3 t-Testing

```
#!/bin/bash

argnum=$((0))
for arg in "$@"; do

  argnum=$(($argnum+1))

  #get scores for that hotel and mean
```

```bash
  count[$argnum]=$((0))
  mean[$argnum]=$((0))
  for i in $(grep "<Overall>" $arg | sed "s/[^0-9]*//g"); do
    count[$argnum]=$((${count[$argnum]}+1))
    scores[${count[argnum]}]=$(echo $i)
    mean[$argnum]=$((${mean[$argnum]} + ${scores[${count[$argnum]}]}))
  done
  mean[$argnum]=`echo "scale=20; ${mean[$argnum]}/${count[$argnum]}" | bc`


  #get variance
  sigma=$((0))
  for i in "${scores[@]}"; do
    sigma=`echo "scale=20; $sigma+($i-${mean[$argnum]})^2" | bc`
  done

  var[$argnum]=`echo "scale=20; (1/(${count[$argnum]}-1))*$sigma" | bc`
done

#Calculate t-stat
sx1x2=`echo "scale=20; sqrt(((${count[1]}-1)*${var[1]} + (${count[2]}-1)*${var[2]}) / (${count[1]}+${count[2]}-2))" | bc`
t=`echo "scale=20; (${mean[1]}-${mean[2]}) / ($sx1x2*sqrt((1/${count[1]})+(1/${count[2]})))" | bc`

#print all the stuff
echo "t: $(printf %.2f $t)"
sd1=$(echo "scale=2; sqrt(${var[1]})" | bc)
sd2=$(echo "scale=2; sqrt(${var[2]})" | bc)
echo "Mean $(echo $1 | cut -d/ -f2 | sed 's/\..*//'): $(printf %.2f ${mean[1]}), SD: $(printf %.2f $sd1)"
echo "Mean $(echo $2 | cut -d/ -f2 | sed 's/\..*//'): $(printf %.2f ${mean[2]}), SD: $(printf %.2f $sd2)"

critvalue=1.972731033408872

#bash only does int arithmetic, so multiply the t-value and
#crit value by big number to allow meaningful comparison
t=`echo "scale=0; $t*1000000000" | bc`
t=${t%.*}
critvalue=`echo "scale=0; $critvalue*1000000000" | bc`
critvalue=${critvalue%.*}
negcritvalue=$(($critvalue*-1))

#if inside critical range, return 1
if [ $t -gt $critvalue ] || [ $t -lt $negcritvalue ]
then
  sig=$((1))
else
  sig=$((0))
fi

echo $sig
```

statistical_sig.sh

2 Hypothesis Testing

3 Discussion