ภาคผนวก

ตัวอย่างโค้ดในส่วนการประมวลผลของการพัฒนาระบบการสร้างต้นไม้ตัดสินใจที่มีประสิทธิภาพ

1. file func.inc.php

ไฟล์ func.inc.php เป็นไฟล์ที่เก็บรวมรวมฟังก์ชันต่างๆที่ใช้ในการประมวลผลในขั้นตอน Pre-processing ประกอบไปด้วย 3 คลาสดังนี้

1.1. Class getElement

เป็นคลาสที่เก็บฟังก์ชันสำหรับรับข้อมูล Array และแปลงข้อมูลเพื่อส่งต่อไปยังส่วนการทำงานต่อไป

1.1.1. Function getClass($arr,$col)

เป็นฟังก์ชันที่ใช้รับข้อมูลคลาสในรูปแบบ Array และ หมายเลขของ คอลัมน์ที่รับข้อมูลมา

```
$data = array();
for($i=1;$i<count($arr);$i++){
array_push($data,$arr[$i][$col]);}
return  $data;
```

1.1.2. Function getEle($arr,$col)

เป็นฟังก์ชันที่ใช้รับค่าข้อมูลโหนดในรูปแบบ Array และหมายเลขของ คอลัมน์ที่รับข้อมูลมา

```
$data = array();
for($i=1;$i<count($arr);$i++){
array_push($data,$arr[$i][$col]);}
return  $data;
```

1.1.3. Function printArr($arr,$dy)

เป็นฟังก์ชันในการเขียนข้อมูล Array ในรูปแบบ Text เพื่อใช้ในการแสดงผลบนตาราง ในหน้าที่ต่อผสานกับผู้ใช้งาน

```
if($dy==1){
for($i=0;$i<count($arr);$i++){
echo $arr[$i];
if($i!=count($arr)-1){
echo " , ";
}else{  }}}
elseif($dy==2){
```

```
for($i=1;$i<count($arr);$i++){
for($j=0;$j<count($arr[$i]);$j++){
echo $arr[$i][$j];
if($j!=count($arr[$i])-1){
echo " , ";
}else{}}
echo "<br>";}
}else{ die(" Error !"); }
```

1.1.4. Function mean_col($arr)

เป็นฟังก์ชันที่ใช้ในการคำควนค่า Mean

```
for($i=0;$i<count($arr);$i++){
$num = $num + $arr[$i];}
$meant = $num/count($arr);
return $meant;
```

1.1.5. Function sd($arr)

เป็นฟังก์ชันที่ใช้ในการหาค่า SD

```
$app = new self();
$mean = $app->mean_col($arr);
$num=0;
for($i=0;$i<count($arr);$i++){
$num = pow(($arr[$i]-$mean),2)+$num;}
$sd = sqrt($num/count($arr));
return $sd;
```

1.1.6. Function Cpare($arr,$key)

เป็นฟังค์ชันที่ใช้เปรียบเทียบค่าของข้อมูล

```
$newVal=array();
$app = new prep();
if($key==1){
foreach($arr as $val){ array_push($newVal,$val); }
return $newVal; }else
if($key==2){
$sc = $app->zScore($arr);
```

```
foreach($sc as $val){ $val=$val*1;
if($val<0){ array_push($newVal,"under");
}else{ array_push($newVal,"over"); }}
return $newVal; } else
if($key==3){
foreach($arr as $val){ $sc = $app->bindDepth_cal($arr,1);
if($val<$sc[0]){ array_push($newVal,"under"); }else{
array_push($newVal,"over");}}
return $newVal;} else
if($key==4){
foreach($arr as $val){ $sg = $app->bindDepth_cal($arr,2);
 if($val<$sg[0]){ array_push($newVal,"low"); }else{
if($val<$sg[1]){ array_push($newVal,"midium"); }else{
array_push($newVal,"hight"); }}}
return $newVal; } else
 if($key==5){
 foreach($arr as $val){ $sc = $app->bindMean_cal($arr,1);
if($val<$sc[0]){ array_push($newVal,"under"); }else{ array_push($newVal,"over");
}}
return $newVal;   }else
if($key==6){ foreach($arr as $val){ $sg = $app->bindMean_cal($arr,2);
if($val<$sg[0]){ array_push($newVal,"low"); }else{
if($val<$sg[1]){ array_push($newVal,"midium"); }else{
array_push($newVal,"hight");}}}
 return $newVal; } else
 if($key==7){  $sc = $app->mNor($arr);
foreach($sc as $val){ $val=$val*1;
if($val<5){ array_push($newVal,"under");}else{ array_push($newVal,"over");}}
return $newVal; }
```

1.2. Class prep extends getElement

เป็นคลาสที่เก็บฟังก์ชันที่ใช้ในการประมวลผลในส่วน Pre-processing โดยมีการสืบทอดฟังก์ชันมาจากคลาส getElement

1.2.1. Function classing($arr)

เป็นฟังก์ชันที่แยกจำนวนคลาสจากข้อมูล Array

```
$f = array();
$result =array();
$f = array_unique($arr);
foreach($f as $val){
array_push($result,$val); }
return $result;
```

1.2.2. Function bindDepth_cal($arr,$bin)

เป็นฟังก์ชันที่ใช้ประมวลผล Pre-processing แบบ Binding Depth

```
$old = $arr;
sort($arr);
$length = count($arr);
if($bin==1){ //Binding 2 Bin
if($length%2==0){
$bind = $length/2;
}else{
$bind = ($length+1)/2;
}}else
if($bin==2){ //Binding 3 Bin
if($length%3==0){
$bind = $length/3;}else
if($length%3==1){
$bind = ($length-1)/3;}else{
$bind = ($length+1)/3;}}else{
echo die("data error");}
$nfunc = new self;
$bit = $nfunc->bindDepth_search($arr,$bind,$bin);
$comp = $nfunc->getCompire($bit,$bin);return $comp;
```

1.2.3. Function bindDepth_search($arr,$bind,$bin)
เป็นฟังค์ชันที่ใช้ควบคู่กับ Function bindDepth_cal

```
if($bin==1){
$b=2;}
else
if($bin==2){
$b=3;}
$box=array();
$count=0;
for($i=0;$i<$b;$i++){
for($j=0;$j<$bind;$j++){
$box[$i][$j]= $arr[$count];
$count++;}}
return $box;
```

1.2.4. function getCompire($bitd,$bin)
เป็นฟังค์ชันที่ใช้ควบคู่กับ Function bindDepth_cal

```
$length1 = count($bitd[0]);
$length1 = $length1-1;
$length2 = count($bitd[0]);
$length2 = $length2-1;
$comp =array();
if($bin==1){
array_push($comp, $bitd[1][0]);}
elseif($bin==2){
array_push($comp, $bitd[1][0]);
array_push($comp, $bitd[2][0]);}
else{
echo die("data error");}
return $comp;}
```

1.2.5. function bindMean_cal($arr,$bin)

เป็นฟังค์ชันที่ใช้ประมวลผล Pre-processing แบบ Binding Mean

```
$old = $arr;
@sort($arr);
$length = count($arr);
if($bin==1){ //Binding 2 Bin
if($length%2==0){
$bind = $length/2;
}else{$bind = ($length+1)/2;}}else
if($bin==2){ //Binding 3 Bin
if($length%3==0){
$bind = $length/3;
}else
if($length%3==1){
$bind = ($length-1)/3;
}else{
$bind = ($length+1)/3; }
}else{echo die("data error");}
$nfunc = new self;
$bitd = $nfunc->bindDepth_search($arr,$bind,$bin);
$comp = $nfunc->getMean($bitd,$bin);        return $comp;
```

1.2.6. function getMean($bitd,$bin)

เป็นฟังค์ชันที่ใช้ควบคู่กับ Function bindMean_cal

```
$comp =array();
for($i=0;$i<count($bitd);$i++){
$num=0;
for($j=0;$j<count($bitd[$i]);$j++){
$num = $num+$bitd[$i][$j]; }
$num = $num/count($bitd[$i]);
array_push($comp, intval($num)); }
$com =array();
```

```
if($bin==1){
array_push($com, $comp[0]); }else
if($bin==2){
array_push($com, $comp[0]);
array_push($com, $comp[1]);

}else{echo die("data error");}
return $com;
```

1.2.7. function mNor($arr)

เป็นฟังค์ชันที่ใช้ประมวลผล Pre-processing แบบ Max-min Max-Min Normalization ค่าสูงสุดจำนวนใหม่ที่ตั้งไว้ คือ 1 และค่าต่ำจำนวนจำนวนใหม่สุดที่ตั้งไว้คือ 0

```
$nx=10;
$nn=0;
$mNor=array();
for($i=0;$i<count($arr);$i++){
$m=(($arr[$i]-min($arr))/(max($arr)-min($arr)))*($nx-$nn))-$nn;
array_push($mNor,$m); }
return $mNor;
```

1.2.8. function zScore($arr)

เป็นฟังค์ชันที่ใช้ประมวลผล Pre-processing แบบ Z-score

```
$app = new parent();
$mean = $app->mean_col($arr);
$sd = $app->sd($arr);
$zs=array();
for($i=0;$i<count($arr);$i++){
$vp = ($arr[$i]-$mean)/($sd);
array_push($zs,$vp); }
return $zs; }
```

1.2.9. function prepocessing($arr,$selecter)

เป็นฟังค์ชันที่ใช้ประมวลผลในขั้นตอนที่ผู้ใช้งานทำการเลือกวิธีในการ Pre-processing โดยฟังก์ชันนี้จะมีการเรียกใช้งานฟังก์ชันที่กล่าวมาข้างต้นเพื่อประมวลข้อมูล

```
$pre = array();
$app = new self();
foreach($selecter as $key=>$val){
$col = $key+1;
$hold = $app->getEle($arr,$col); // get element from column
if($val=1){ //// cate
$sc = $app->classing($hold);
array_push($pre,$sc); }
elseif($val==2){ //// z - score
$sc = $app->zScore($hold);
array_push($pre,$sc);} else
if($val==3){  ///binding depth 2 bin
$sc = $app->bindDepth_cal($hold,1);
array_push($pre,$sc);}else
if($val==4){ ///binding depth 3 bin
$sc = $app->bindDepth_cal($hold,2);
array_push($pre,$sc);}else
if($val==5){
$sc = $app->bindMean_cal($hold,1);
array_push($pre,$sc); }else
if($val==6){
$sc = $app->bindMean_cal($hold,2);
array_push($pre,$sc); }else
if($val==7){
$sc = $app->mNor($hold);
array_push($pre,$sc);}
```

```
else{ die(" Error! ");
}}return $pre; }
}//End Class
```

1.3. class dataProcess extends prep

เป็นคลาสที่เก็บฟังก์ชันที่ใช้ในการประมวลผลข้อมูล ให้เปลี่ยนไปในรูปแบบต้างๆ โดยมีการสืบทอดคลาส prep

1.3.1. function rubTable($arr)

เป็นฟังก์ชันที่บรรจุข้อมูลที่ผ่านการ Pre-processing ให้อยู่ในรูปแบบที่แสดงผลในฟอร์มของตารางบนส่วนต่อผสานกับผู้ใช้งาน

```
$app = new self();
$runs = array();
$suff =count($arr[0])-1;
for($rou=0;$rou<count($arr[0]);$rou++){
if($rou==$suff){
$getElm = $app->getClass($arr,$rou);
}else{$getElm = $app->getEle($arr,$rou);
array_push($runs,$getElm);
}// end for
 return $runs;
```

1.3.2. function tranForm($data,$head)

เป็นฟังก์ชันที่มีหน้าที่ในการแปลงข้อมูลโดยจะเพิ่มส่วนชื่อของตารางเข้าไปยัง Node แรกของข้อมูลใน Array

```
$tran = array();
$app = new self();
$arr = $app->rubTable($data);
for($gum=0;$gum<count($head);$gum++){
 $sc = $app->Cpare($arr[$gum],$head[$gum]);
array_push($tran,$sc); }
return $tran; }
```

2.Tree.inc.php File

ไฟล์ Tree.inc.php เป็นไฟล์ที่เก็บรวมรวมฟังก์ชันต่างๆที่ใช้ในการประมวลผลในขั้นตอนในการสร้างต้นไม้ตัดสินใจ ประกอบไปด้วย 2 คลาสดังนี้

ประกาศค่าตัวแปรที่รับมาจาก เซสชั่น

```
$data = $_SESSION[‘data’];
$pre = $_SESSION[‘pre’];
$row = count($data);// this variable use for count row of $data array
```

2.1. class setData{

เป็นคลาสที่รวมฟังชก์ชันที่ใช้ในการเตรียมข้อมูลก่อนเข้าสู่การเข้าประมวลผลในฟังก์ชันที่สร้างต้นไม้ตัดสินใจ

2.1.1.function partitn($arr)

เป็นฟังก์ชันแปลงข้อมูลที่เป็นตัวเลขให้อยู่ในรูปแบบ int

```
$bin = (count($arr)-1)/3;
$bin = number_format(($bin),0,'.','');
```

2.1.2. function unig($arr)

เป็นฟังก์ชันที่ใช้ในการหาค่าที่ไม่ซ้ำใน Array

```
return     array_unique($arr);
```

2.1.3.function getEle($arr,$col)

เป็นฟังก์ชันที่มีหน้าที่รับค่าข้อมูลโดยระบุเป็นที่อยู่ของคอลัมน์ของข้อมูล

```
$data = array();
for($i=0;$i<count($arr);$i++){
array_push($data,$arr[$i][$col]); }
return  $data;
```

2.1.4. function transpose($array)

เป็นฟังก์ชันที่ใช้ในการกลับค่าใน Array (จาก แถวเป็นคอลัมน์ จากคอลัมน์เป็นแถว)

```
array_unshift($array, null);
return call_user_func_array('array_map', $array);
```

2.1.5.function Log2($argument)

เป็นฟังก์ชันที่ใช้ในการคำนวณค่าของ Log ฐาน 2

Return log($argument, 2);

2.2. class tree extends setData

เป็นคลาสที่รวบรวมฟังก์ชันในการประมวลผลสร้างต้นไม้ตัดสินใจ

2.2.1.function info_di($arr)

เป็นฟังก์ชันในการคำนวณหาค่า info_D

```
$col = count($arr);
$col=$col-1;
$app = new self();
$info=0;
$geo=$arr[$col];
$clu = array_unique($geo);
$clss = array();
foreach($clu as $val){
array_push($clss,$val); }
$arrnum = count($clss);
if($arrnum==2){
$info = $app->info_table2($geo);
}elseif($arrnum==3){
$info =  $app->info_table3($geo);
}return $info;
```

2.2.2. function info_table2($arr)

เป็นฟังก์ชันในการคำนวณหาค่า info_รวม แบบ 2 คลาส

```
$app = new self();
$col=count($arr);
$classCol=$col-1;
$row=count($arr);
```

```
$geo=$arr;
$clu = array_unique($geo);
$clss = array();
foreach($clu as $val){
array_push($clss,$val); }
$cls1=0;
$cls2=0;
for($run = 0 ; $run<$row ; $run++){
if($geo[$run]==$clss[0]){
$cls1=$cls1+1;
}elseif($geo[$run]==$clss[1]){
$cls2=$cls2+1;}
}//////end for
$term1 = $cls1/$row;
$term2 = $cls2/$row;
$logterm1 = $app->Log2($term1);
$logterm2 = $app->Log2($term2);
$info_table = -($term1*$logterm1)-($term2*$logterm2);
return $info_table;
```

2.2.3. function info_table3($arr)

เป็นฟังก์ชันในการคำนวณหาค่า info_รวม แบบ 3 คลาส

```
$app = new self();
$col=count($arr);
$classCol=$col-1;
$row=count($arr);
$geo=$arr;
$clu = array_unique($geo);
$clss = array();
foreach($clu as $val){
array_push($clss,$val); }
```

```
$cls1=0;
$cls2=0;
$cls3=0;
for($run = 0 ; $run<$row ; $run++){
if($geo[$run]==$clss[0]){
$cls1=$cls1+1; }else
if($geo[$run]==$clss[1]){
$cls2=$cls2+1; }else
if($geo[$run]==$clss[2]){
$cls3=$cls3+1; }
}//////end for
$term1 = $cls1/$row;
$term2 = $cls2/$row;
$term3 = $cls3/$row;
$logterm1 = $app->Log2($term1);
$logterm2 = $app->Log2($term2);
$logterm3 = $app->Log2($term3);
$info_table = -($term1*$logterm1)-($term2*$logterm2)-
($term3*$logterm3);
return $info_table;
```

2.2.4. function info_table_D3($arr,$cola)

เป็นฟังก์ชันในการหาค่าและเปรียบเทียบค่าที่ระบุลำดับของโหนดบนต้นไม้ตัดสินใจแบบ 3 คลาส

```
$cola=$cola-1;
$app=new self();
$col=count($arr);//5
$classCol=$col-1;//4
$classes = $arr[$classCol];////class arr[4]
$data = $arr[$cola];////arr[column]
$row=count($data);
$clu = array_unique($classes);
```

```
$dt =array_unique($data);
$clss = array();
$case = array();
foreach($clu as $val){///// class
array_push($clss,$val); }
foreach($dt as $v){ ////col
array_push($case,$v); }
$case1_1=0;
$case1_2=0;
$case1_3=0;
$case2_1=0;
$case2_2=0;
$case2_3=0;
$case3_1=0;
$case3_2=0;
$case3_3=0;
for($run = 0 ; $run<$row ; $run++){
if($data[$run]==$case[0]&&$classes[$run]==$clss[0]){
$case1_1=$case1_1+1;}else
if($data[$run]==$case[0]&&$classes[$run]==$clss[1]){
$case1_2=$case1_2+1;      }else
if($data[$run]==$case[0]&&$classes[$run]==$clss[2]){
$case1_3=$case1_3+1; }else
if($data[$run]==$case[1]&&$classes[$run]==$clss[0]){
$case2_1=$case2_1+1; }else
if($data[$run]==$case[1]&&$classes[$run]==$clss[1]){
$case2_2=$case2_2+1; }else
if($data[$run]==$case[1]&&$classes[$run]==$clss[2]){
$case2_3=$case2_3+1;}else
if($data[$run]==$case[2]&&$classes[$run]==$clss[0]){
$case3_1=$case3_1+1; }else
```

```
if($data[$run]==$case[2]&&$classes[$run]==$clss[1]){
$case3_2=$case3_2+1;
}elseif($data[$run]==$case[2]&&$classes[$run]==$clss[2]){
$case3_3=$case3_3+1; }
}end for
//class 1//
$s1_all=$case1_1+$case2_1+$case3_1;
$s1_term1 = $case1_1/$s1_all;
$s1_term2 = $case2_1/$s1_all;
$s1_term3 = $case3_1/$s1_all;
if($s1_all==0){
$s3_all=1; }
if($s1_term1==0){
$s1_term1=1; }
if($s1_term2==0){
$s1_term2=1;
}if($s1_term3==0){
$s1_term3=1; }
$s1_logterm1 = $app->Log2($s1_term1);
$s1_logterm2 = $app->Log2($s1_term2);
$s1_logterm3 = $app->Log2($s1_term3);
$s1_info_table = -($s1_term1*$s1_logterm1)-
($s1_term2*$s1_logterm2)-($s1_term3*$s1_logterm3);
////class 2 ////
$s2_all=$case1_2+$case2_2+$case3_2;
if($s2_all==0){
$s3_all=1; }
$s2_term1 = $case1_2/$s2_all;
$s2_term2 = $case2_2/$s2_all;
$s2_term3 = $case3_2/$s2_all;
if($s2_term1==0){
```

```
$s2_term1=1; }

if($s2_term2==0){

$s2_term2=1;

}if($s2_term3==0){

$s2_term3=1;

}$s2_logterm1 = $app->Log2($s2_term1);

$s2_logterm2 = $app->Log2($s2_term2);

$s2_logterm3 = $app->Log2($s2_term3);

$s2_info_table = -($s2_term1*$s2_logterm1)-
($s2_term2*$s2_logterm2)-($s2_term3*$s2_logterm3);

///class 3//
$s3_all=$case1_3+$case2_3+$case3_3;

if($s3_all==0){

$s3_all=1; }

$s3_term1 = $case1_3/$s3_all;

$s3_term2 = $case2_3/$s3_all;

$s3_term3 = $case3_3/$s3_all;

if($s3_term1==0){

$s3_term1=1; }

if($s3_term2==0){

$s3_term2=1; }

if($s3_term3==0){

$s3_term3=1; }

$s3_logterm1 = $app->Log2($s3_term1);

$s3_logterm2 = $app->Log2($s3_term2);

$s3_logterm3 = $app->Log2($s3_term3);

$s3_info_table = -($s3_term1*$s3_logterm1)-
($s3_term2*$s3_logterm2)-($s3_term3*$s3_logterm3);

$s1_sum = ($s1_all/$row);

$s2_sum = ($s2_all/$row);

$s3_sum = ($s3_all/$row);
$info_gain=($s1_sum*$s1_info_table)+($s2_sum*$s2_info_table)+($s3_su
m*$s3_info_table);
```

```
return $info_gain;
```

2.2.5. function info_table_D2($arr,$cola)

เป็นฟังก์ชันในการหาค่าและเปรียบเทียบค่าที่ระบุลำดับของโหนดบนต้นไม้ตัดสินใจแบบ 2 คลาส

```
$cola=$cola-1;
$app = new self();
$col=count($arr);//5
$classCol=$col-1;//4
$classes = $arr[$classCol];////class arr[4]
$data = $arr[$cola];////arr[column]
$row=count($data);
$clu = array_unique($classes);
$dt =array_unique($data);
$clss = array();
$case = array();
foreach($clu as $val){///// class
array_push($clss,$val);}
foreach($dt as $v){ ////col
array_push($case,$v);
}$case1_1=0;
$case1_2=0;
$case1_3=0;
$case2_1=0;
$case2_2=0;
$case2_3=0;
for($run = 0 ; $run<$row ; $run++){
if($data[$run]==$case[0]&&$classes[$run]==$clss[0]){
$case1_1=$case1_1+1; }
if($data[$run]==$case[0]&&$classes[$run]==$clss[1]){
$case1_2=$case1_2+1;}
if($data[$run]==$case[0]&&$classes[$run]==$clss[2]){
```

```
$case1_3=$case1_3+1; }
if($data[$run]==$case[1]&&$classes[$run]==$clss[0]){
$case2_1=$case2_1+1; }
if($data[$run]==$case[1]&&$classes[$run]==$clss[1]){
$case2_2=$case2_2+1; }
if($data[$run]==$case[1]&&$classes[$run]==$clss[2]){
$case2_3=$case2_3+1; }
}//////end for
////class 1 /////
$s1_all=$case1_1+$case2_1+$case3_1;
$s1_term1 = $case1_1/$s1_all;
$s1_term2 = $case2_1/$s1_all;
$s1_term3 = $case3_1/$s1_all;
if($s1_term1==0){
$s1_term1=1;      }
if($s1_term2==0){
$s1_term2=1; }
if($s1_term3==0){
$s1_term3=1; }
$s1_logterm1 = $app->Log2($s1_term1);
$s1_logterm2 = $app->Log2($s1_term2);
$s1_logterm3 = $app->Log2($s1_term3);
$s1_info_table = -($s1_term1*$s1_logterm1)-
($s1_term2*$s1_logterm2)-($s1_term3*$s1_logterm3);
//class 2//
$s2_all=$case1_2+$case2_2+$case3_2;
$s2_term1 = $case1_2/$s2_all;
$s2_term2 = $case2_2/$s2_all;
$s2_term3 = $case3_2/$s2_all;
if($s2_term1==0){
$s2_term1=1; }
if($s2_term2==0){
```

$s2_term2=1; }

if($s2_term3==0){

$s2_term3=1; }

$s2_logterm1 = $app->Log2($s2_term1);

$s2_logterm2 = $app->Log2($s2_term2);

$s2_logterm3 = $app->Log2($s2_term3);

$s2_info_table = -($s2_term1*$s2_logterm1)-
($s2_term2*$s2_logterm2)-($s2_term3*$s2_logterm3);

$s1_sum = ($s1_all/$row);

$s2_sum = ($s2_all/$row);

$info_gain=($s1_sum*$s1_info_table)+($s2_sum*$s2_info_table);

return $info_gain;

2.2.6. function info_table($all_di,$t_all,$info_d){

เป็นฟังก์ชันที่ใช้ควบคู่กับการสร้างต้นไม้ตัดสินใจแบบ 3 คลาส

$constant1=info_table($all_di,$t_all,$info_d); $constant_all =
$constant1+$constant at n...

$info_table=($all_di/$t_all)*($info_d);

return $info_tab

ประวัติผู้เขียน

ชื่อ – นามสกุล	นายวาทิต พลฤทธิ์
วัน เดือน ปีเกิด	27 กุมภาพันธ์ 2532
ประวัติการศึกษา	วท.บ. (วิศวกรรมซอฟต์แวร์) มหาวิทยาลัยพายัพ
	ปีการศึกษา 2554
ประวัติการทำงาน	Senior programmer
	บริษัท Danesoe จำกัด