Project

Spring 2017

In this project we are trying to figure out, what base attributes of Pokemon makes it most used by assesing overused(Standard class) Pokemons battle data and random battle data where any Pokemon can fight with any Pokemon randomly.

First we attempted to understand the attributes in the data set and its distribution.

```r
library(ggplot2)
library(GGally)
library(tidyr)
library(gridExtra)
library(arm)
```

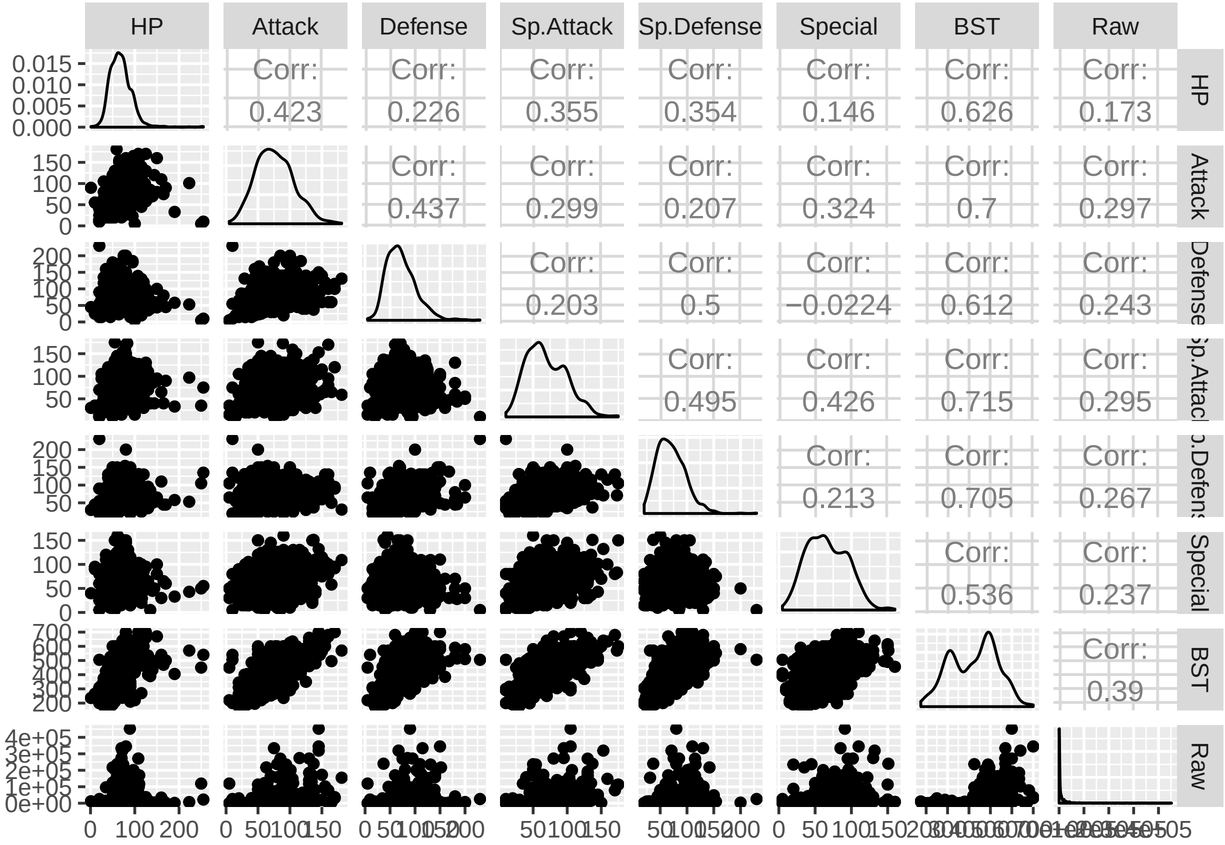
```
## Loading required package: MASS

## Loading required package: Matrix

##
## Attaching package: 'Matrix'

## The following object is masked from 'package:tidyr':
##
##     expand

## Loading required package: lme4

##
## arm (Version 1.9-3, built: 2016-11-21)

## Working directory is /Users/naveenkumar2703/Documents/Assignments/EDA/Project
```

```r
getwd()
```

```
## [1] "/Users/naveenkumar2703/Documents/Assignments/EDA/Project"
```

```r
Pokemon.Attributes = read.csv('data/Pokeys.csv')
OverUsed = read.csv('data/gen7ou-0.txt',skip = 2)
OverUsed1500 = read.csv('data/gen7ou-1500.txt',skip = 2)
OverUsed1695 = read.csv('data/gen7ou-1695.txt',skip = 2)
OverUsed1825 = read.csv('data/gen7ou-1825.txt',skip = 2)
UnderUsed = read.csv('data/gen7uu-0.txt',skip = 2)
RandomBattle = read.csv('data/gen7randombattle-0.txt',skip = 2)
Uber = read.csv('data/gen7ubers-0.txt',skip = 2)
LC = read.csv('data/gen7lc-0.txt',skip = 2)
OUPokemon <- merge(Pokemon.Attributes,OverUsed,by="Pokemon")
RUPokemon <- merge(Pokemon.Attributes,RandomBattle,by="Pokemon")
```

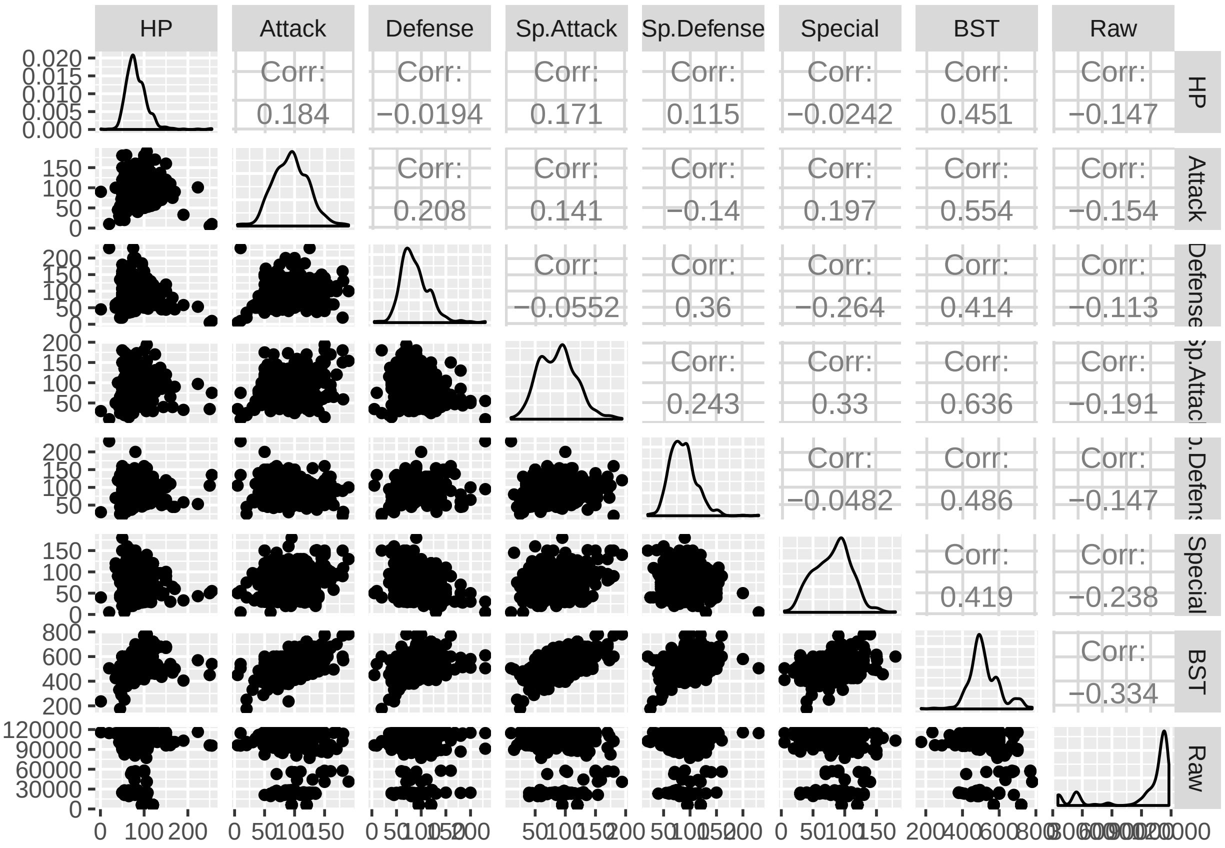
```r
attach(OUPokemon)
ggpairs(data.frame(HP, Attack, Defense, Sp.Attack,Sp.Defense, Special, BST, Raw))
```



```r
attach(RUPokemon)
```

```
## The following objects are masked from OUPokemon:
##
##     Abilities, Ability.1, Ability.2, Ability.3, Adaptability,
##     Aerilate, Aftermath, Air.Lock, Analytic, Anger.Point,
##     Anticipation, Arena.Trap, Aroma.Veil, Attack, Aura.Break,
##     Bad.Dreams, Battery, Battle.Armor, Battle.Bond, Beast.Boost,
##     Berserk, Big.Pecks, Blaze, BST, Bug, Bulletproof, Cheek.Pouch,
##     Chlorophyll, Clear.Body, Cloud.Nine, Color.Change, Comatose,
##     Competitive, Compound.Eyes, Contrary, Corrosion, Cursed.Body,
##     Cute.Charm, Damp, Dancer, Dark, Dark.Aura, Dazzling,
##     Defeatist, Defense, Defiant, Delta.Stream, Desolate.Land,
##     Disguise, Download, Dragon, Drizzle, Drought, Dry.Skin,
##     Early.Bird, Effect.Spore, Electric, Electric.Surge,
##     Emergency.Exit, Fairy, Fairy.Aura, Fighting, Filter, Fire,
##     Flame.Body, Flare.Boost, Flash.Fire, Flower.Gift, Flower.Veil,
##     Fluffy, Flying, Forecast, Forewarn, Friend.Guard, Frisk,
##     Full.Metal.Body, Fur.Coat, Gale.Wings, Galvanize, Generation,
##     Ghost, Gluttony, Gooey, Grass, Grass.Pelt, Grassy.Surge,
##     Ground, Guts, Harvest, Healer, Heatproof, Heavy.Metal,
##     Honey.Gather, HP, Huge.Power, Hustle, Hydration, Hyper.Cutter,
##     Ice, Ice.Body, Illuminate, Illusion, Immunity, Imposter,
##     Infiltrator, Innards.Out, Inner.Focus, Insomnia, Intimidate,
##     Iron.Barbs, Iron.Fist, Justified, Keen.Eye, Klutz, Leaf.Guard,
##     Levitate, Light.Metal, Lightning.Rod, Limber, Liquid.Ooze,
##     Liquid.Voice, Long.Reach, Magic.Bounce, Magic.Guard, Magician,
##     Magma.Armor, Magnet.Pull, Marvel.Scale, Mega.Launcher,
##     Merciless, Minus, Misty.Surge, Mold.Breaker, Moody,
##     Motor.Drive, Mountaineer, Moxie, Multiscale, Multitype, Mummy,
##     Natural.Cure, No.Guard, Normal, Normalize, Oblivious,
##     Overcoat, Overgrow, Own.Tempo, Parental.Bond, Persistent,
##     Pickpocket, Pickup, Pixilate, Plus, Poison, Poison.Heal,
##     Poison.Point, Poison.Touch, Pokemon, Power.Construct,
##     Power.of.Alchemy, Prankster, Pressure, Primordial.Sea,
##     Prism.Armor, Protean, Psychic, Psychic.Surge, Pure.Power,
##     Queenly.Majesty, Quick.Feet, Rain.Dish, Rank, Rattled, Raw,
##     Raw.percentage, Real, Real.percentage, Rebound, Receiver,
##     Reckless, Refrigerate, Regenerator, Rivalry, RKS.System, Rock,
##     Rock.Head, Rough.Skin, Run.Away, Sand.Force, Sand.Rush,
##     Sand.Stream, Sand.Veil, Sap.Sipper, Schooling, Scrappy,
##     Serene.Grace, Shadow.Shield, Shadow.Tag, Shed.Skin,
##     Sheer.Force, Shell.Armor, Shield.Dust, Shields.Down, Simple,
##     Skill.Link, Slow.Start, Slush.Rush, Sniper, Snow.Cloak,
##     Snow.Warning, Solar.Power, Solid.Rock, Soul.Heart, Soundproof,
##     Sp.Attack, Sp.Defense, Special, Speed.Boost, Stakeout, Stall,
##     Stamina, Stance.Change, Static, Steadfast, Steel, Steelworker,
##     Stench, Sticky.Hold, Storm.Drain, Strong.Jaw, Sturdy,
##     Suction.Cups, Super.Luck, Surge.Surfer, Swarm, Sweet.Veil,
##     Swift.Swim, Symbiosis, Synchronize, Tangled.Feet,
##     Tangling.Hair, Technician, Telepathy, Teravolt, Thick.Fat,
##     Tinted.Lens, Torrent, Tough.Claws, Toxic.Boost, Trace, Triage,
##     Truant, Turboblaze, Type.1, Type.2, Type.3, Type.4, Types,
##     Unaware, Unburden, Unnerve, Usage.Category, Usage.percentage,
##     Victory.Star, Vital.Spirit, Volt.Absorb, Water, Water.Absorb,
##     Water.Bubble, Water.Compaction, Water.Veil, Weak.Armor,
##     White.Smoke, Wimp.Out, Wonder.Guard, Wonder.Skin, Zen.Mode
```

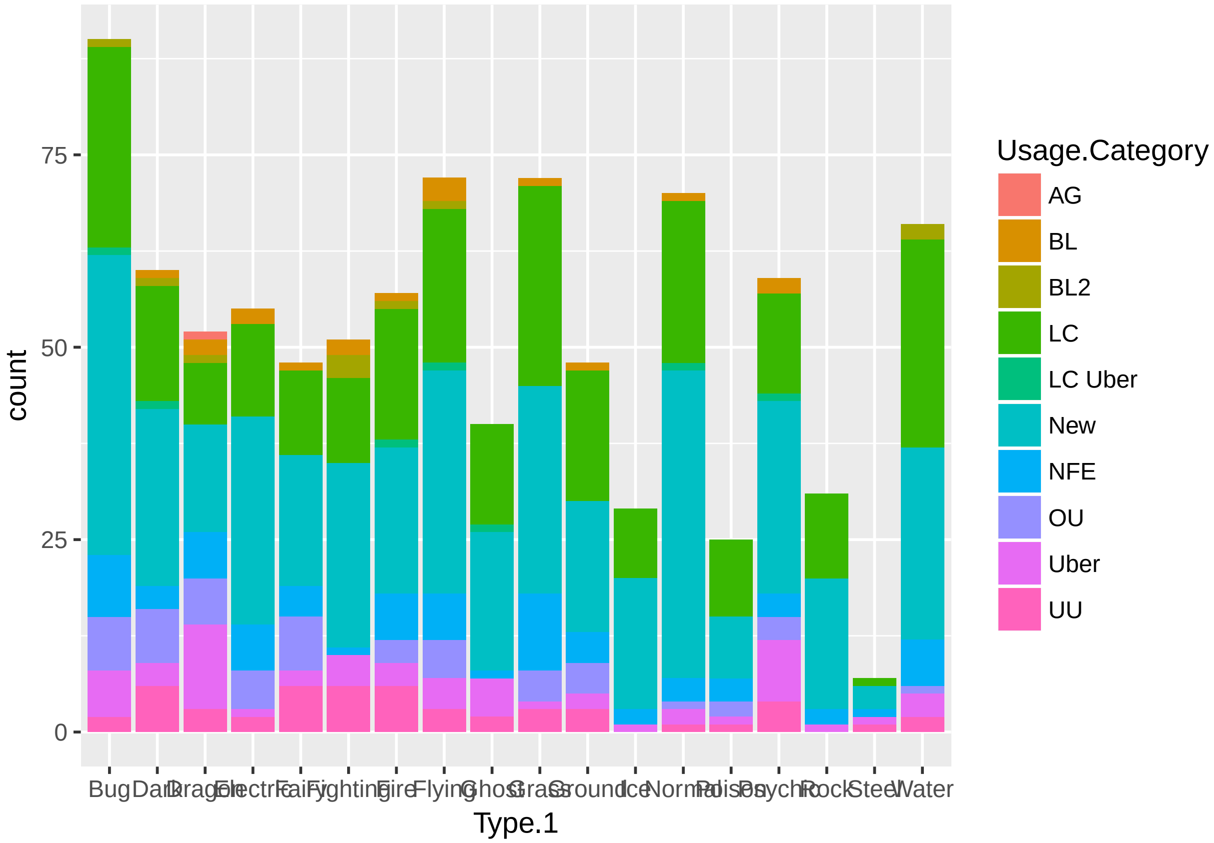
```
ggpairs(data.frame(HP, Attack, Defense, Sp.Attack,Sp.Defense, Special, BST, Raw))
```



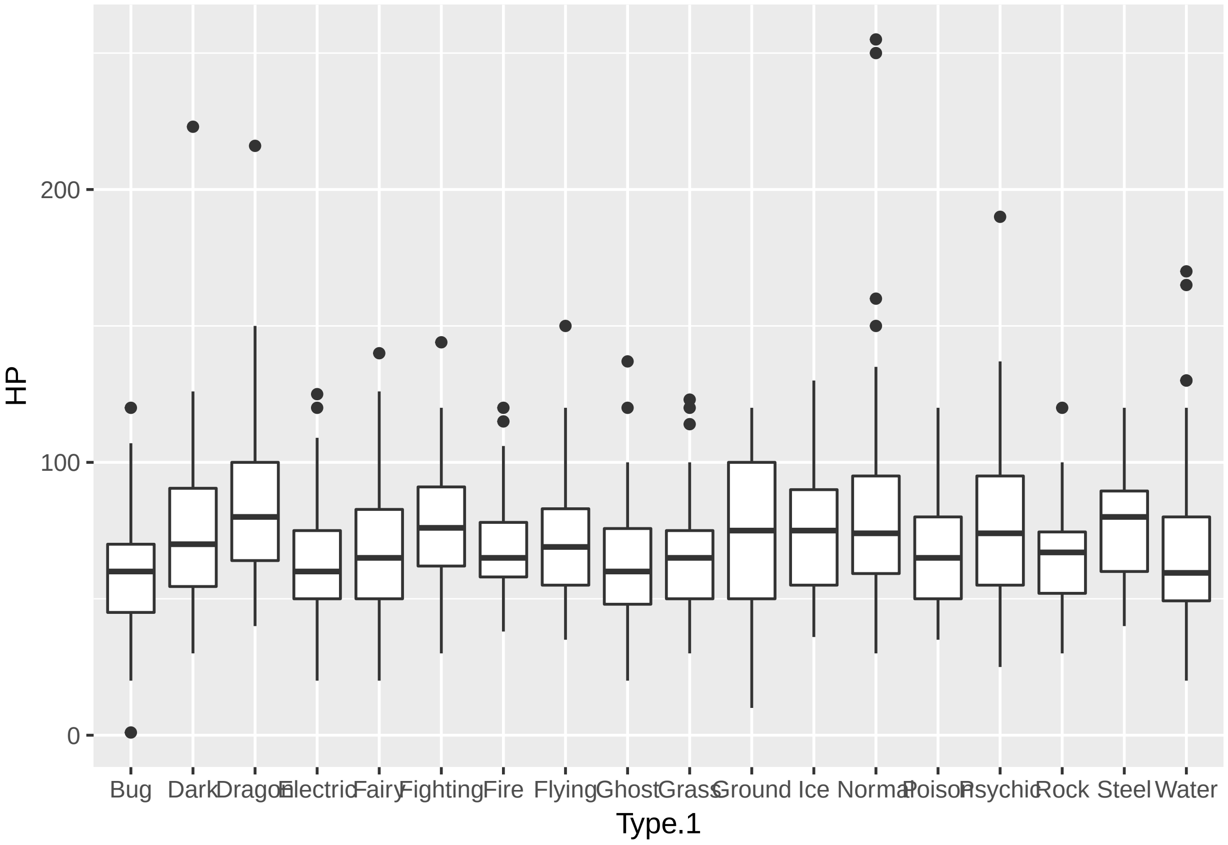
Now let's try a *histogram*:

```
ggplot(Pokemon.Attributes,aes(x=Type.1,fill=Usage.Category)) + geom_histogram(stat="count")
```

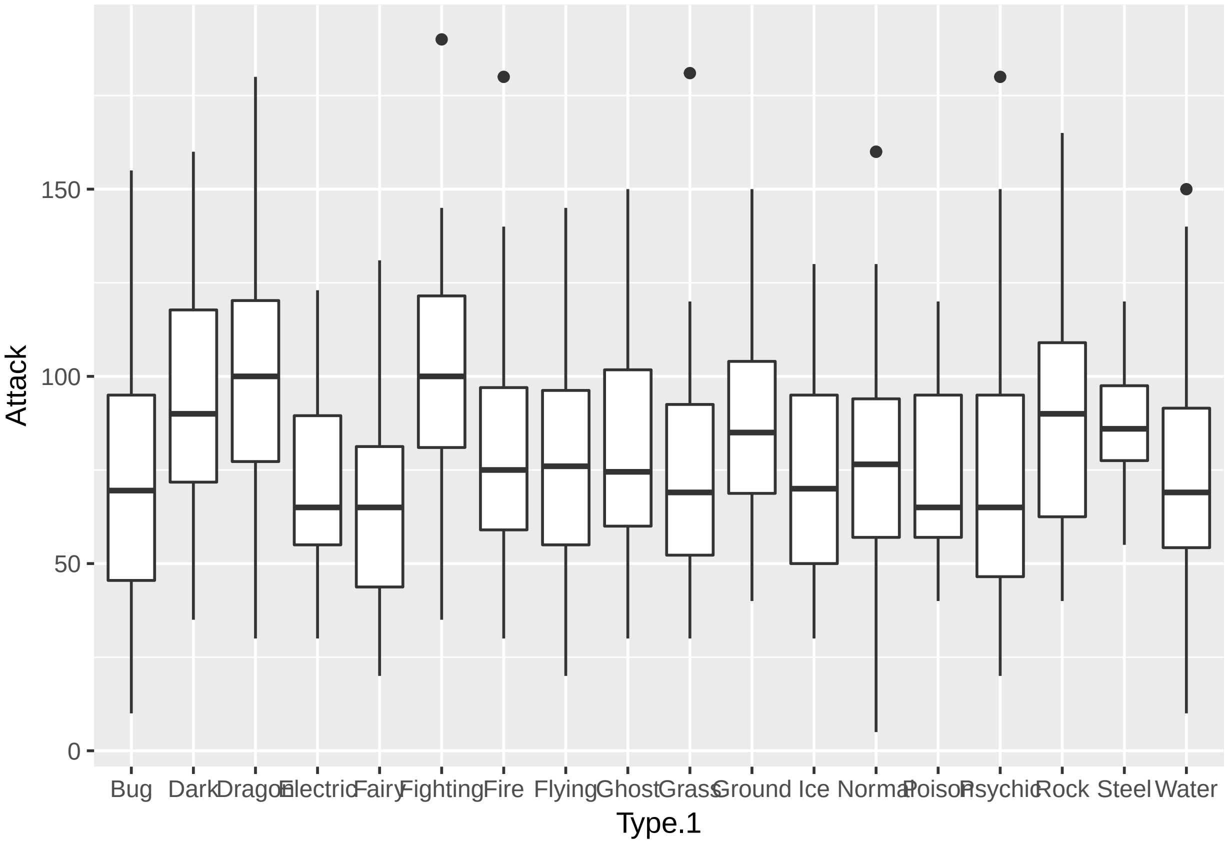
```
## Warning: Ignoring unknown parameters: binwidth, bins, pad
```



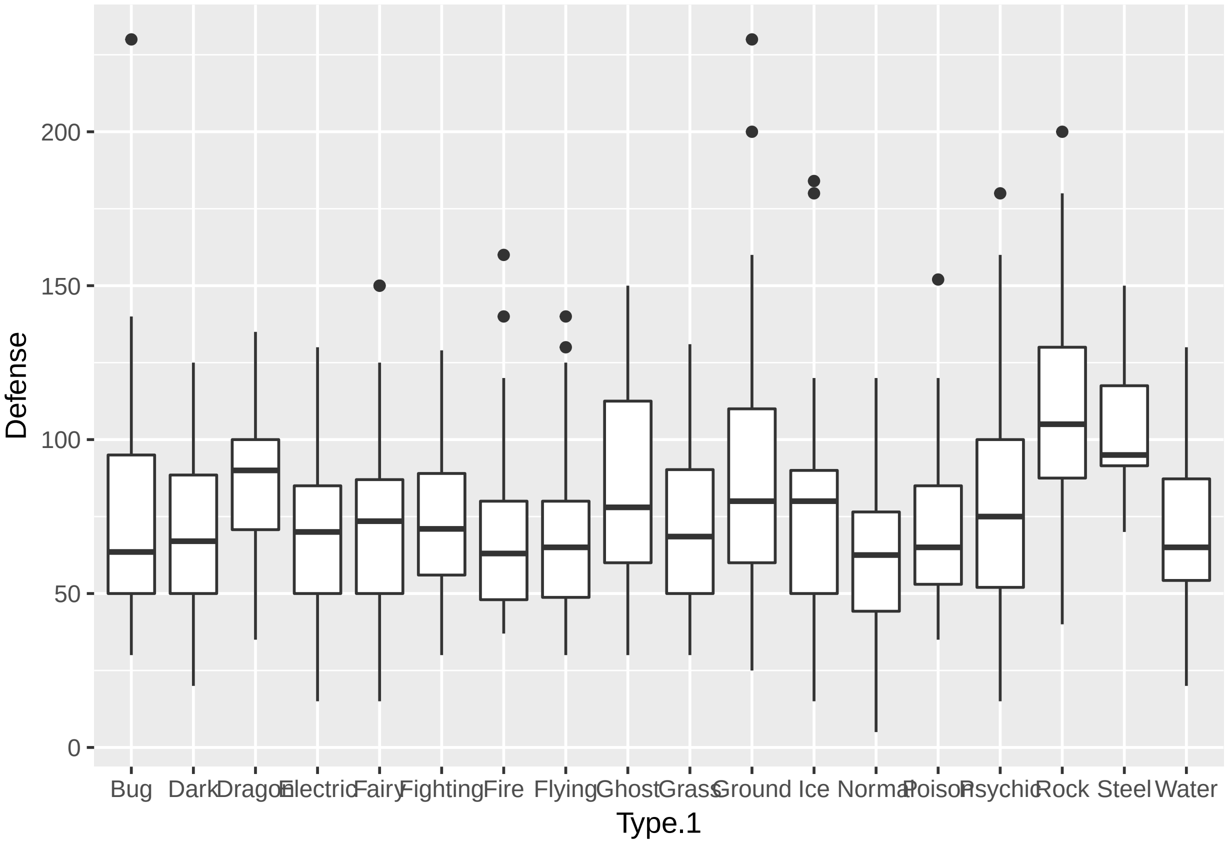
```
ggplot(Pokemon.Attributes,aes(x=Type.1,y=HP)) + geom_boxplot()
```



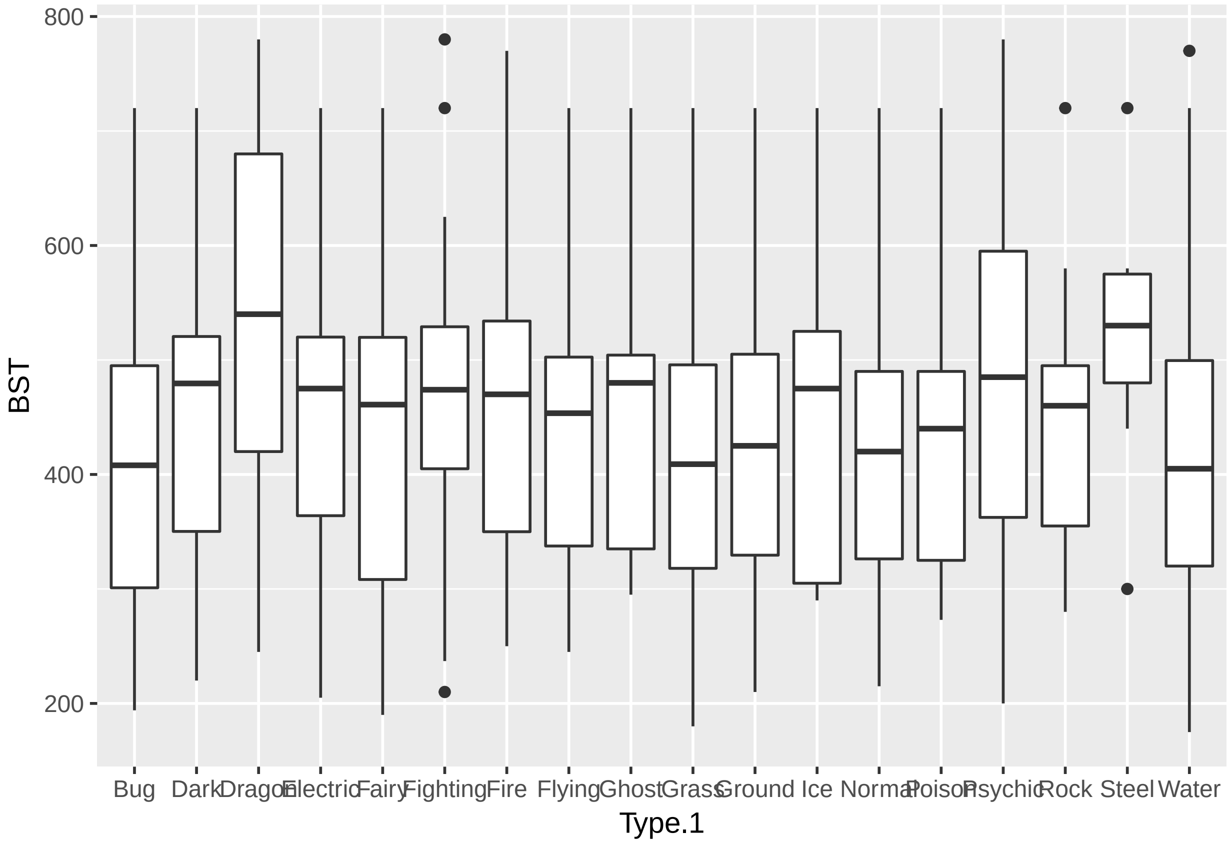
```
ggplot(Pokemon.Attributes,aes(x=Type.1,y=Attack)) + geom_boxplot()+guides(fill=FALSE)
```



```
ggplot(Pokemon.Attributes,aes(x=Type.1,y=Defense)) + geom_boxplot()+guides(fill=FALSE)
```



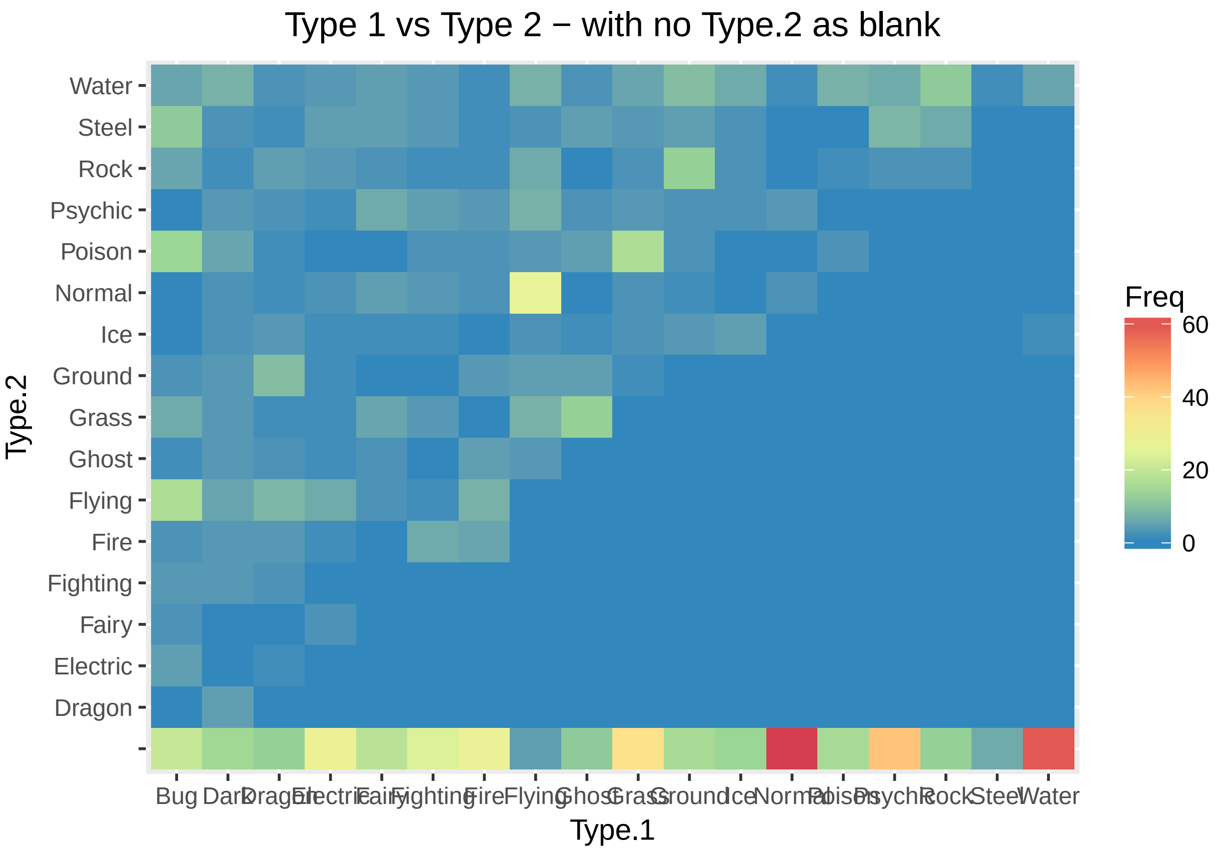
```
ggplot(Pokemon.Attributes,aes(x=Type.1,y=BST)) + geom_boxplot()+guides(fill=FALSE)
```



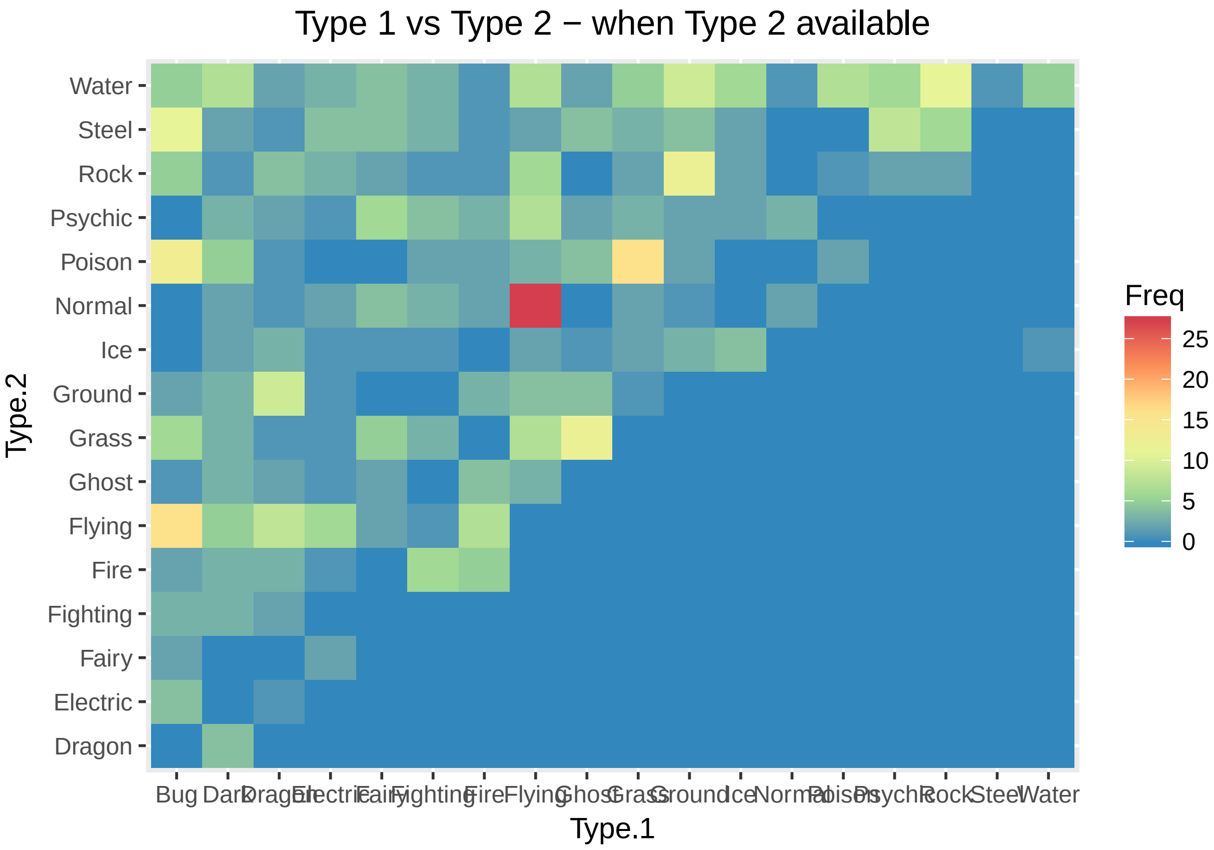
```
attach(Pokemon.Attributes)
```

```
## The following objects are masked from RUPokemon:
##
##     Abilities, Ability.1, Ability.2, Ability.3, Adaptability,
##     Aerilate, Aftermath, Air.Lock, Analytic, Anger.Point,
##     Anticipation, Arena.Trap, Aroma.Veil, Attack, Aura.Break,
##     Bad.Dreams, Battery, Battle.Armor, Battle.Bond, Beast.Boost,
##     Berserk, Big.Pecks, Blaze, BST, Bug, Bulletproof, Cheek.Pouch,
##     Chlorophyll, Clear.Body, Cloud.Nine, Color.Change, Comatose,
##     Competitive, Compound.Eyes, Contrary, Corrosion, Cursed.Body,
##     Cute.Charm, Damp, Dancer, Dark, Dark.Aura, Dazzling,
##     Defeatist, Defense, Defiant, Delta.Stream, Desolate.Land,
##     Disguise, Download, Dragon, Drizzle, Drought, Dry.Skin,
##     Early.Bird, Effect.Spore, Electric, Electric.Surge,
##     Emergency.Exit, Fairy, Fairy.Aura, Fighting, Filter, Fire,
##     Flame.Body, Flare.Boost, Flash.Fire, Flower.Gift, Flower.Veil,
##     Fluffy, Flying, Forecast, Forewarn, Friend.Guard, Frisk,
##     Full.Metal.Body, Fur.Coat, Gale.Wings, Galvanize, Generation,
##     Ghost, Gluttony, Gooey, Grass, Grass.Pelt, Grassy.Surge,
##     Ground, Guts, Harvest, Healer, Heatproof, Heavy.Metal,
##     Honey.Gather, HP, Huge.Power, Hustle, Hydration, Hyper.Cutter,
##     Ice, Ice.Body, Illuminate, Illusion, Immunity, Imposter,
##     Infiltrator, Innards.Out, Inner.Focus, Insomnia, Intimidate,
##     Iron.Barbs, Iron.Fist, Justified, Keen.Eye, Klutz, Leaf.Guard,
##     Levitate, Light.Metal, Lightning.Rod, Limber, Liquid.Ooze,
##     Liquid.Voice, Long.Reach, Magic.Bounce, Magic.Guard, Magician,
##     Magma.Armor, Magnet.Pull, Marvel.Scale, Mega.Launcher,
##     Merciless, Minus, Misty.Surge, Mold.Breaker, Moody,
##     Motor.Drive, Mountaineer, Moxie, Multiscale, Multitype, Mummy,
##     Natural.Cure, No.Guard, Normal, Normalize, Oblivious,
##     Overcoat, Overgrow, Own.Tempo, Parental.Bond, Persistent,
##     Pickpocket, Pickup, Pixilate, Plus, Poison, Poison.Heal,
##     Poison.Point, Poison.Touch, Pokemon, Power.Construct,
##     Power.of.Alchemy, Prankster, Pressure, Primordial.Sea,
##     Prism.Armor, Protean, Psychic, Psychic.Surge, Pure.Power,
##     Queenly.Majesty, Quick.Feet, Rain.Dish, Rattled, Rebound,
##     Receiver, Reckless, Refrigerate, Regenerator, Rivalry,
##     RKS.System, Rock, Rock.Head, Rough.Skin, Run.Away, Sand.Force,
##     Sand.Rush, Sand.Stream, Sand.Veil, Sap.Sipper, Schooling,
##     Scrappy, Serene.Grace, Shadow.Shield, Shadow.Tag, Shed.Skin,
##     Sheer.Force, Shell.Armor, Shield.Dust, Shields.Down, Simple,
##     Skill.Link, Slow.Start, Slush.Rush, Sniper, Snow.Cloak,
##     Snow.Warning, Solar.Power, Solid.Rock, Soul.Heart, Soundproof,
##     Sp.Attack, Sp.Defense, Special, Speed.Boost, Stakeout, Stall,
##     Stamina, Stance.Change, Static, Steadfast, Steel, Steelworker,
##     Stench, Sticky.Hold, Storm.Drain, Strong.Jaw, Sturdy,
##     Suction.Cups, Super.Luck, Surge.Surfer, Swarm, Sweet.Veil,
##     Swift.Swim, Symbiosis, Synchronize, Tangled.Feet,
##     Tangling.Hair, Technician, Telepathy, Teravolt, Thick.Fat,
##     Tinted.Lens, Torrent, Tough.Claws, Toxic.Boost, Trace, Triage,
##     Truant, Turboblaze, Type.1, Type.2, Type.3, Type.4, Types,
##     Unaware, Unburden, Unnerve, Usage.Category, Victory.Star,
##     Vital.Spirit, Volt.Absorb, Water, Water.Absorb, Water.Bubble,
##     Water.Compaction, Water.Veil, Weak.Armor, White.Smoke,
##     Wimp.Out, Wonder.Guard, Wonder.Skin, Zen.Mode

## The following objects are masked from OUPokemon:
##
##     Abilities, Ability.1, Ability.2, Ability.3, Adaptability,
##     Aerilate, Aftermath, Air.Lock, Analytic, Anger.Point,
##     Anticipation, Arena.Trap, Aroma.Veil, Attack, Aura.Break,
##     Bad.Dreams, Battery, Battle.Armor, Battle.Bond, Beast.Boost,
##     Berserk, Big.Pecks, Blaze, BST, Bug, Bulletproof, Cheek.Pouch,
##     Chlorophyll, Clear.Body, Cloud.Nine, Color.Change, Comatose,
##     Competitive, Compound.Eyes, Contrary, Corrosion, Cursed.Body,
##     Cute.Charm, Damp, Dancer, Dark, Dark.Aura, Dazzling,
##     Defeatist, Defense, Defiant, Delta.Stream, Desolate.Land,
##     Disguise, Download, Dragon, Drizzle, Drought, Dry.Skin,
##     Early.Bird, Effect.Spore, Electric, Electric.Surge,
##     Emergency.Exit, Fairy, Fairy.Aura, Fighting, Filter, Fire,
##     Flame.Body, Flare.Boost, Flash.Fire, Flower.Gift, Flower.Veil,
##     Fluffy, Flying, Forecast, Forewarn, Friend.Guard, Frisk,
##     Full.Metal.Body, Fur.Coat, Gale.Wings, Galvanize, Generation,
##     Ghost, Gluttony, Gooey, Grass, Grass.Pelt, Grassy.Surge,
##     Ground, Guts, Harvest, Healer, Heatproof, Heavy.Metal,
##     Honey.Gather, HP, Huge.Power, Hustle, Hydration, Hyper.Cutter,
##     Ice, Ice.Body, Illuminate, Illusion, Immunity, Imposter,
##     Infiltrator, Innards.Out, Inner.Focus, Insomnia, Intimidate,
##     Iron.Barbs, Iron.Fist, Justified, Keen.Eye, Klutz, Leaf.Guard,
##     Levitate, Light.Metal, Lightning.Rod, Limber, Liquid.Ooze,
##     Liquid.Voice, Long.Reach, Magic.Bounce, Magic.Guard, Magician,
##     Magma.Armor, Magnet.Pull, Marvel.Scale, Mega.Launcher,
##     Merciless, Minus, Misty.Surge, Mold.Breaker, Moody,
##     Motor.Drive, Mountaineer, Moxie, Multiscale, Multitype, Mummy,
##     Natural.Cure, No.Guard, Normal, Normalize, Oblivious,
##     Overcoat, Overgrow, Own.Tempo, Parental.Bond, Persistent,
##     Pickpocket, Pickup, Pixilate, Plus, Poison, Poison.Heal,
##     Poison.Point, Poison.Touch, Pokemon, Power.Construct,
##     Power.of.Alchemy, Prankster, Pressure, Primordial.Sea,
##     Prism.Armor, Protean, Psychic, Psychic.Surge, Pure.Power,
##     Queenly.Majesty, Quick.Feet, Rain.Dish, Rattled, Rebound,
##     Receiver, Reckless, Refrigerate, Regenerator, Rivalry,
##     RKS.System, Rock, Rock.Head, Rough.Skin, Run.Away, Sand.Force,
##     Sand.Rush, Sand.Stream, Sand.Veil, Sap.Sipper, Schooling,
##     Scrappy, Serene.Grace, Shadow.Shield, Shadow.Tag, Shed.Skin,
##     Sheer.Force, Shell.Armor, Shield.Dust, Shields.Down, Simple,
##     Skill.Link, Slow.Start, Slush.Rush, Sniper, Snow.Cloak,
##     Snow.Warning, Solar.Power, Solid.Rock, Soul.Heart, Soundproof,
##     Sp.Attack, Sp.Defense, Special, Speed.Boost, Stakeout, Stall,
##     Stamina, Stance.Change, Static, Steadfast, Steel, Steelworker,
##     Stench, Sticky.Hold, Storm.Drain, Strong.Jaw, Sturdy,
##     Suction.Cups, Super.Luck, Surge.Surfer, Swarm, Sweet.Veil,
##     Swift.Swim, Symbiosis, Synchronize, Tangled.Feet,
##     Tangling.Hair, Technician, Telepathy, Teravolt, Thick.Fat,
##     Tinted.Lens, Torrent, Tough.Claws, Toxic.Boost, Trace, Triage,
##     Truant, Turboblaze, Type.1, Type.2, Type.3, Type.4, Types,
##     Unaware, Unburden, Unnerve, Usage.Category, Victory.Star,
##     Vital.Spirit, Volt.Absorb, Water, Water.Absorb, Water.Bubble,
##     Water.Compaction, Water.Veil, Weak.Armor, White.Smoke,
##     Wimp.Out, Wonder.Guard, Wonder.Skin, Zen.Mode
```

```
ggplot(data = as.data.frame(table(Type.1,Type.2)), aes(x = Type.1, y = Type.2)) +geom_tile(aes(fill=Fre
```

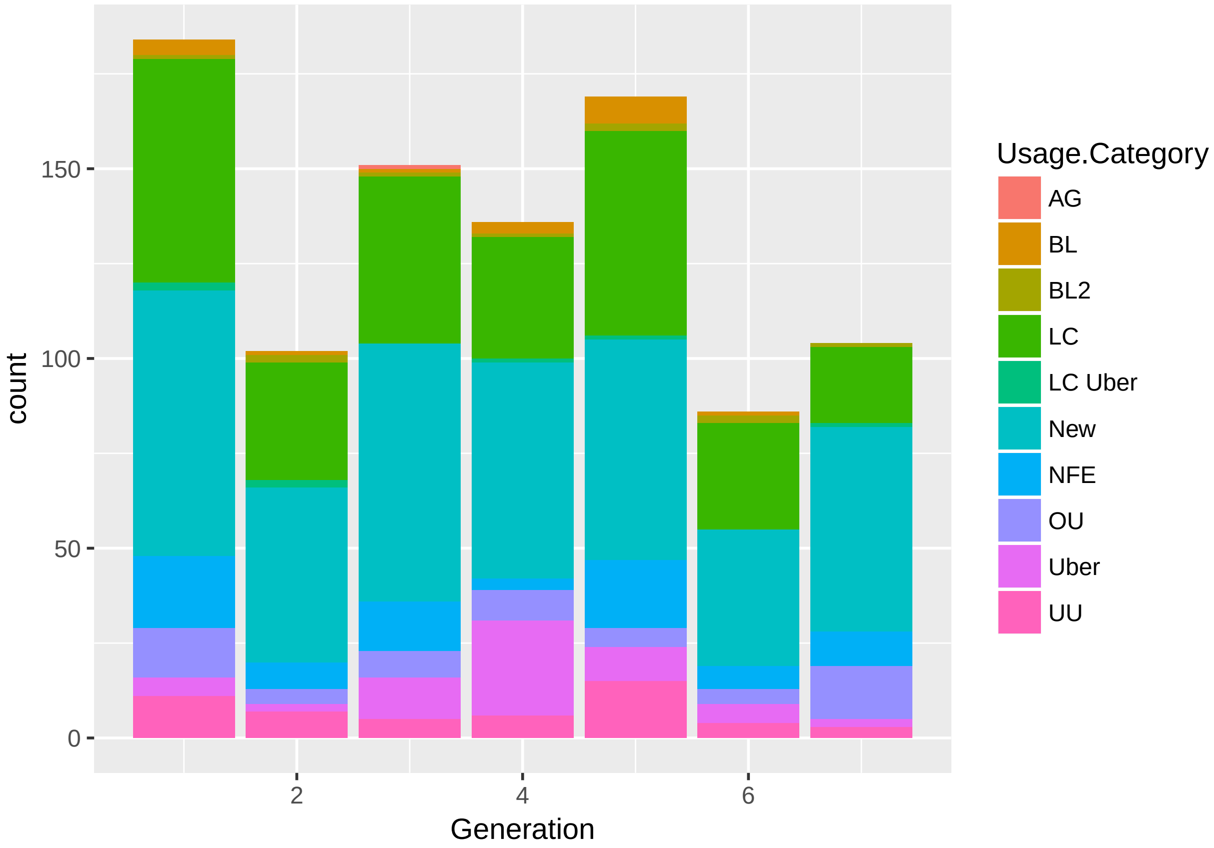


```
type1_type_2 = as.data.frame(table(Type.1,Type.2))
type1_type_2 <- type1_type_2[(type1_type_2$Type.2 != ''),]
ggplot(data = type1_type_2, aes(x = Type.1, y = Type.2,fill=Freq)) + geom_tile() + scale_fill_distiller
```

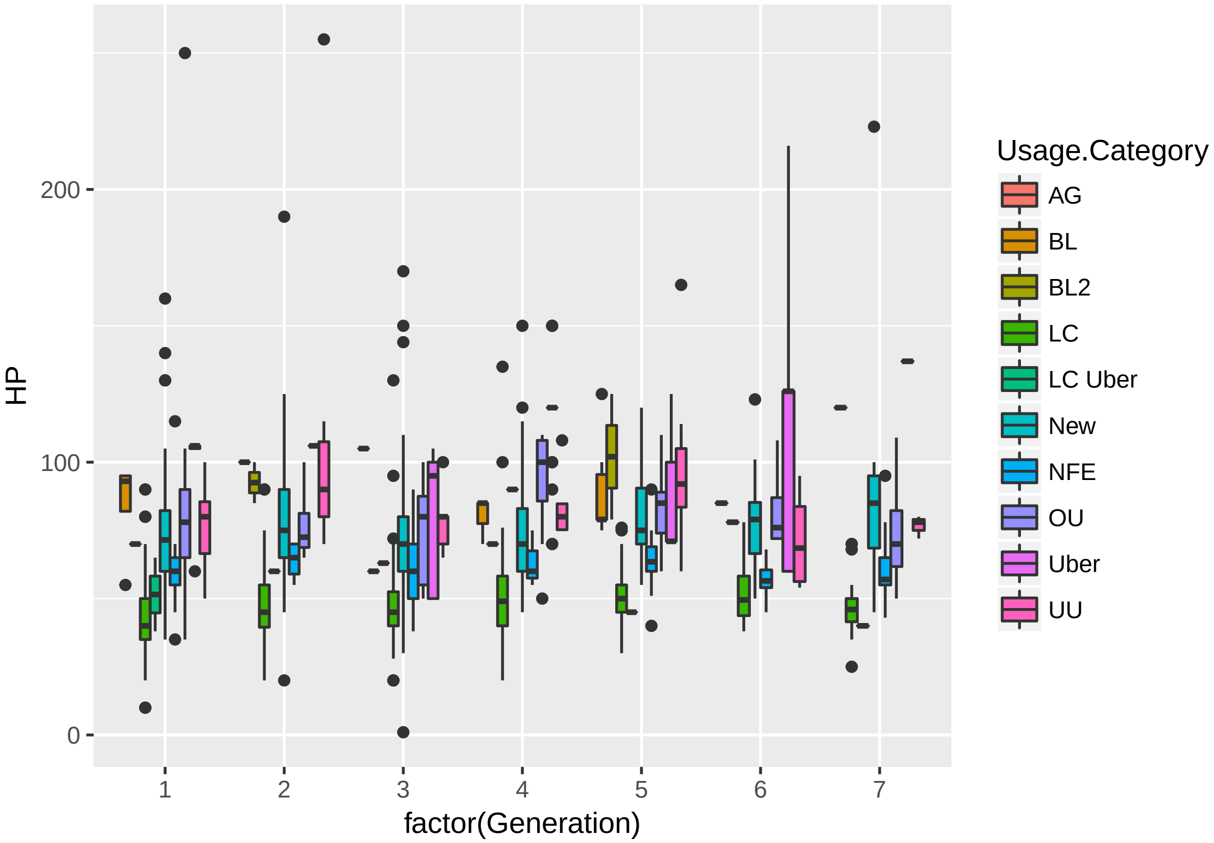


```
ggplot(Pokemon.Attributes,aes(x=Generation,fill=Usage.Category)) + geom_histogram(stat="count")
```

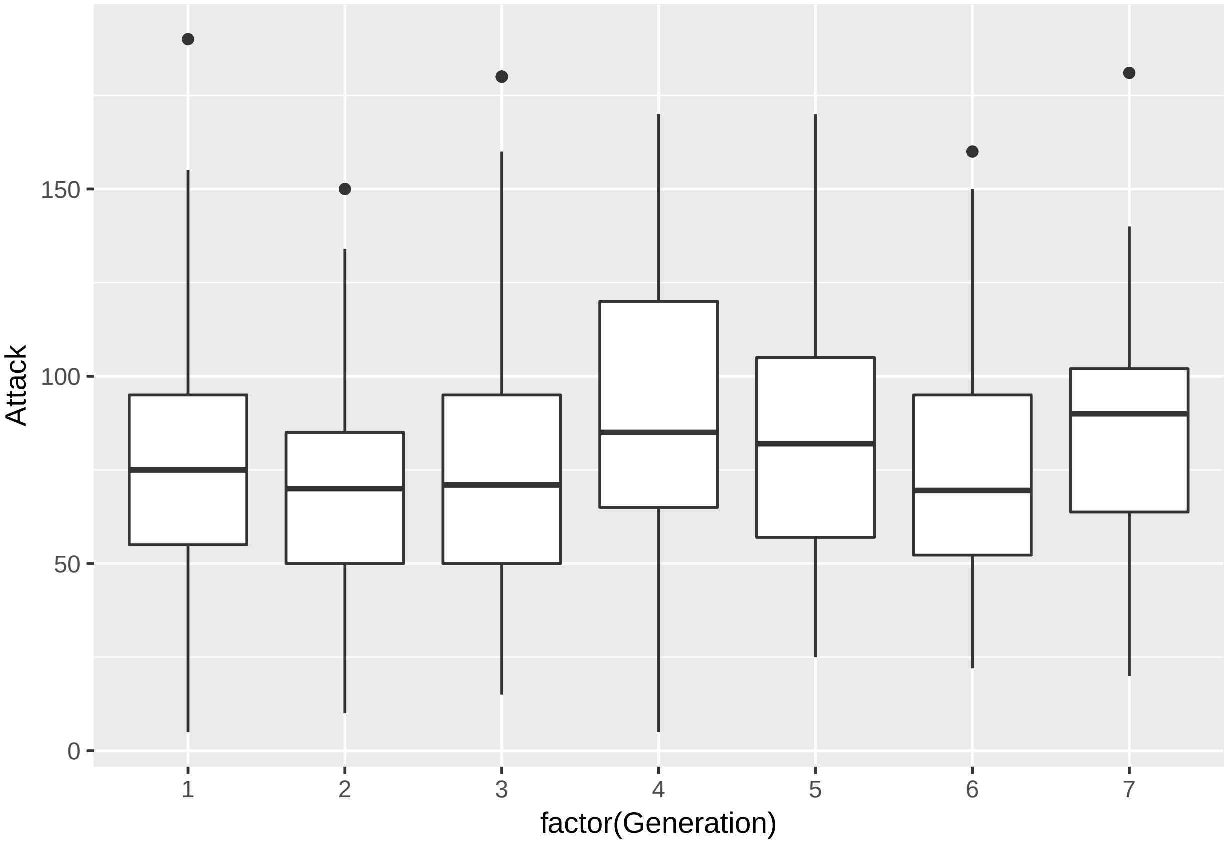
```
## Warning: Ignoring unknown parameters: binwidth, bins, pad
```



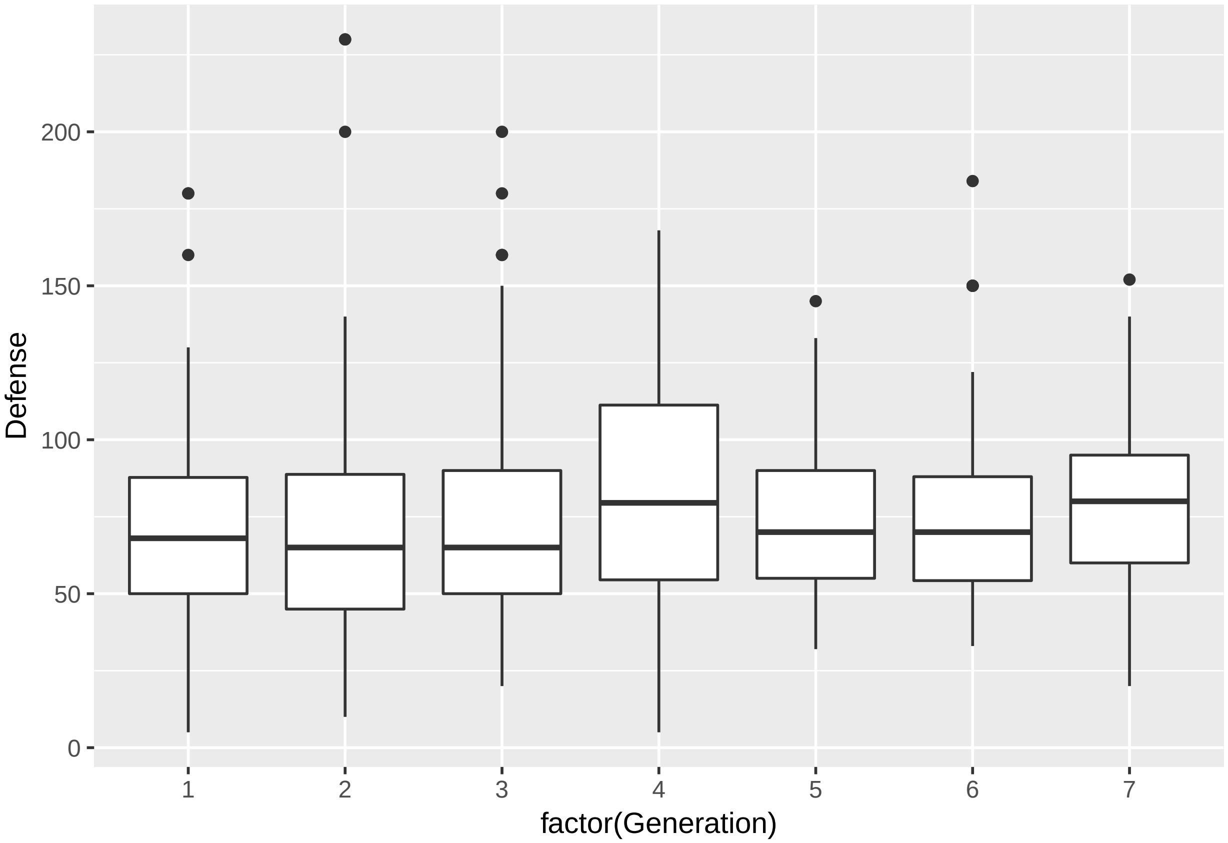
```
ggplot(Pokemon.Attributes,aes(x=factor(Generation),y=HP,fill=Usage.Category)) + geom_boxplot()
```



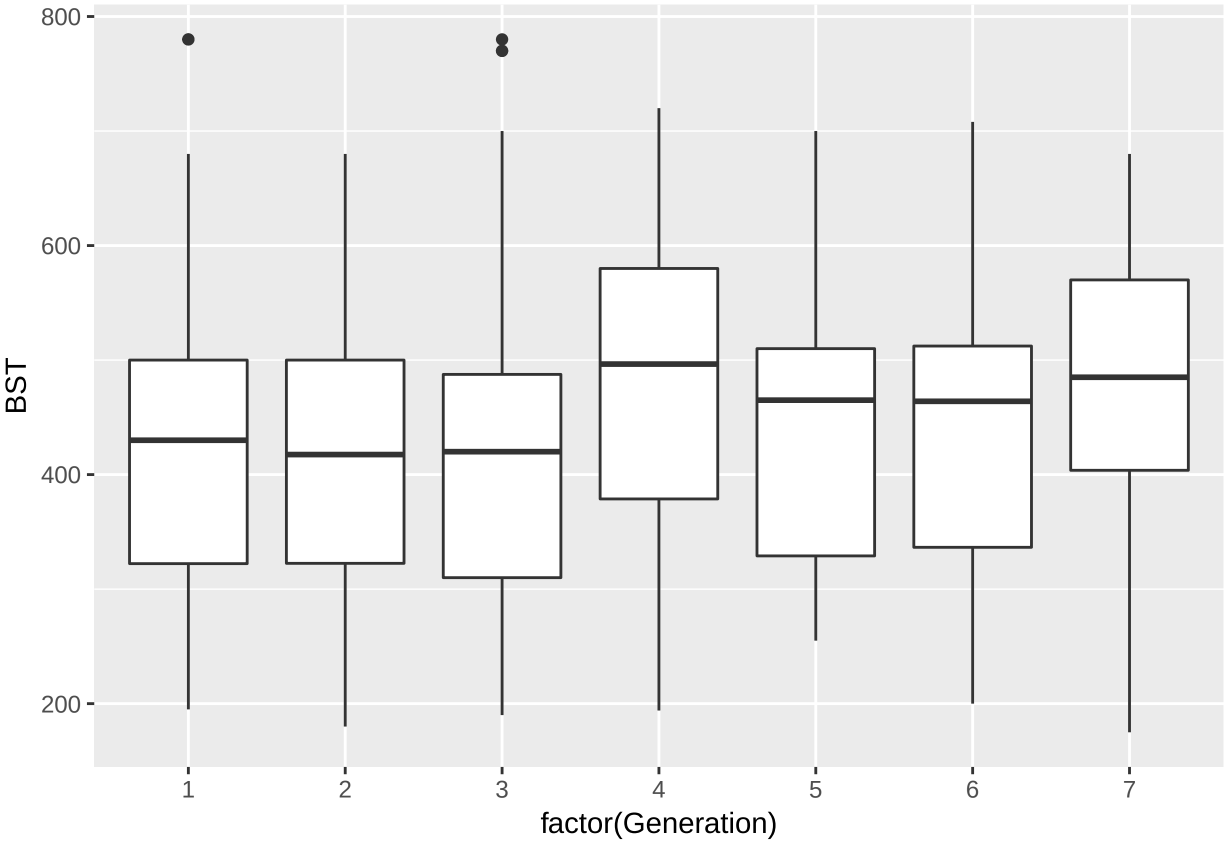
```
ggplot(Pokemon.Attributes,aes(x=factor(Generation),y=Attack)) + geom_boxplot()+guides(fill=FALSE)
```



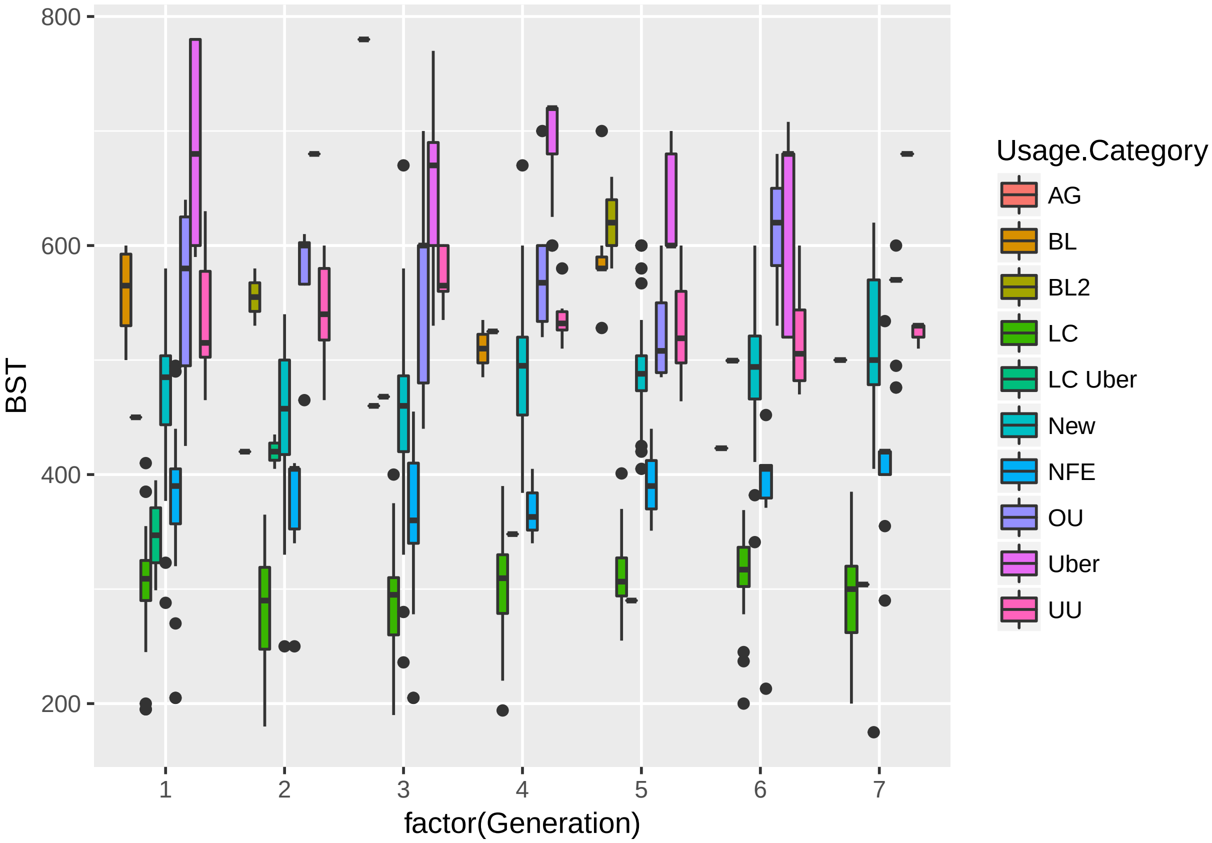
```
ggplot(Pokemon.Attributes,aes(x=factor(Generation),y=Defense)) + geom_boxplot()+guides(fill=FALSE)
```



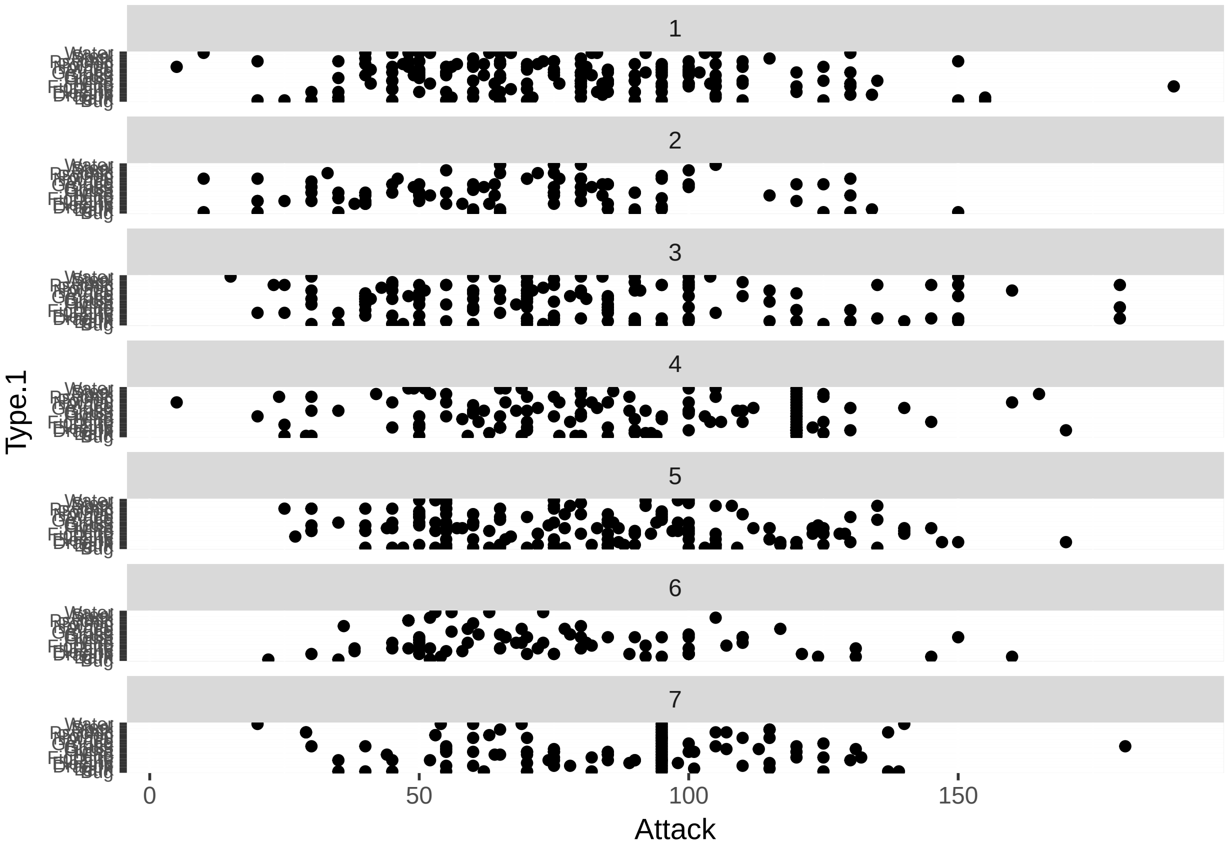
```
ggplot(Pokemon.Attributes,aes(x=factor(Generation),y=BST)) + geom_boxplot()+guides(fill=FALSE)
```



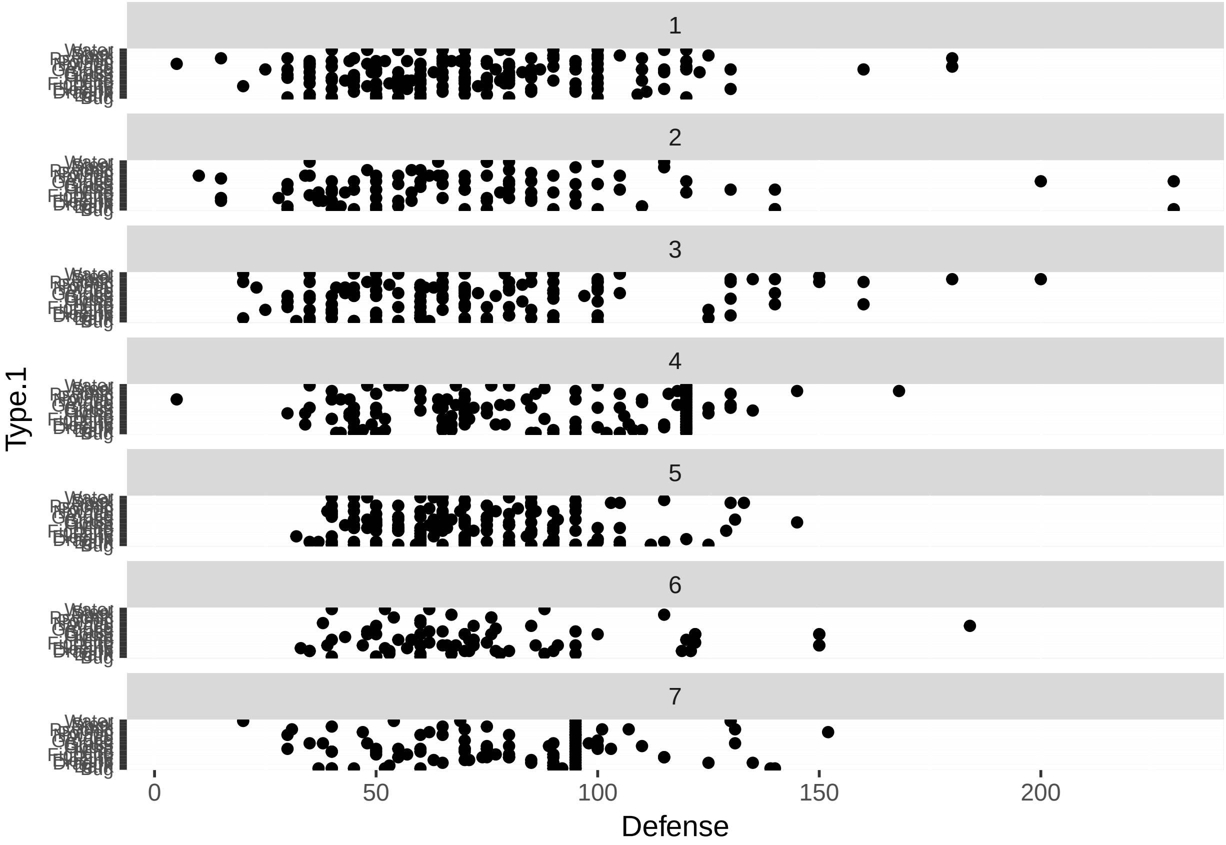
```r
ggplot(Pokemon.Attributes,aes(x=factor(Generation),y=BST,fill=Usage.Category)) + geom_boxplot()
```



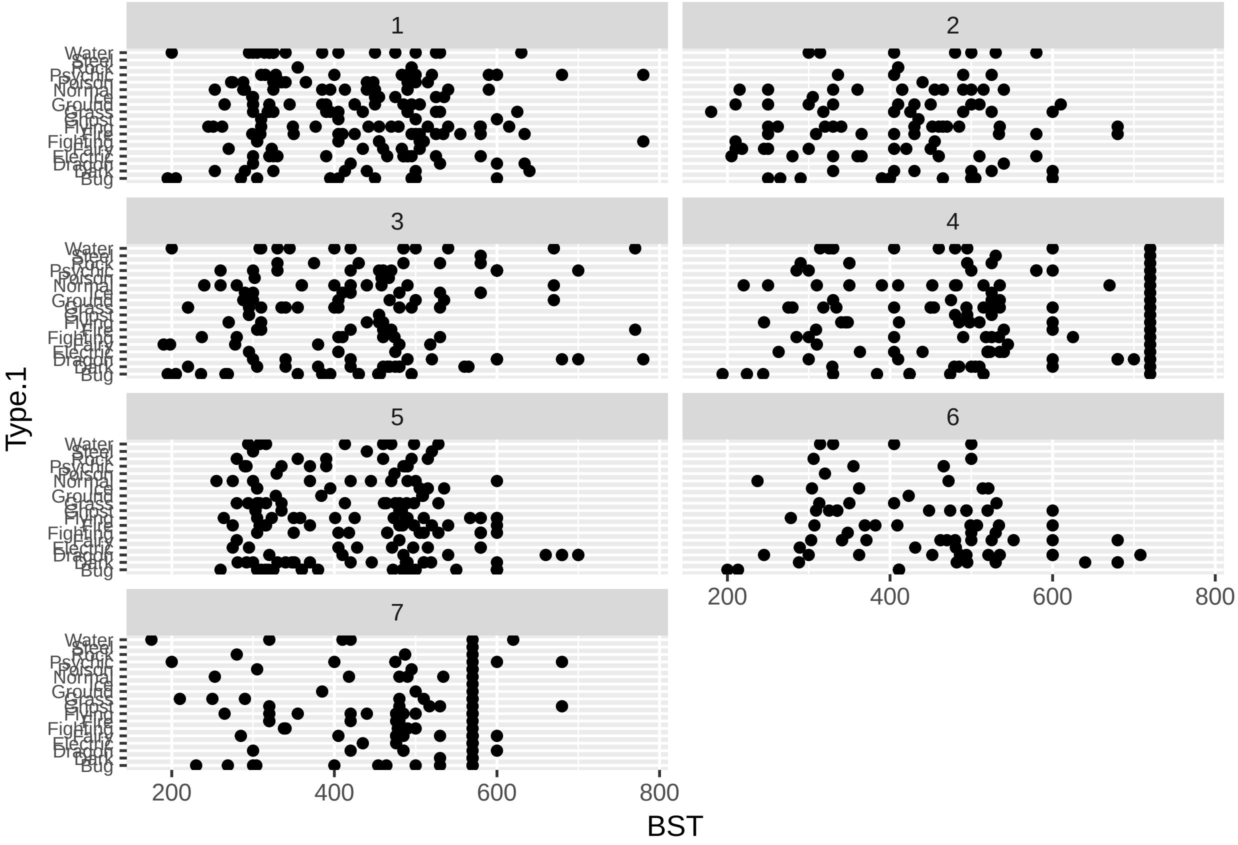
```r
ggplot(Pokemon.Attributes, aes(x = Attack, y = Type.1)) + geom_point() + theme(axis.text.y = element_te
```



```
ggplot(Pokemon.Attributes, aes(x = Defense, y = Type.1)) + geom_point() + theme(axis.text.y = element_te
```



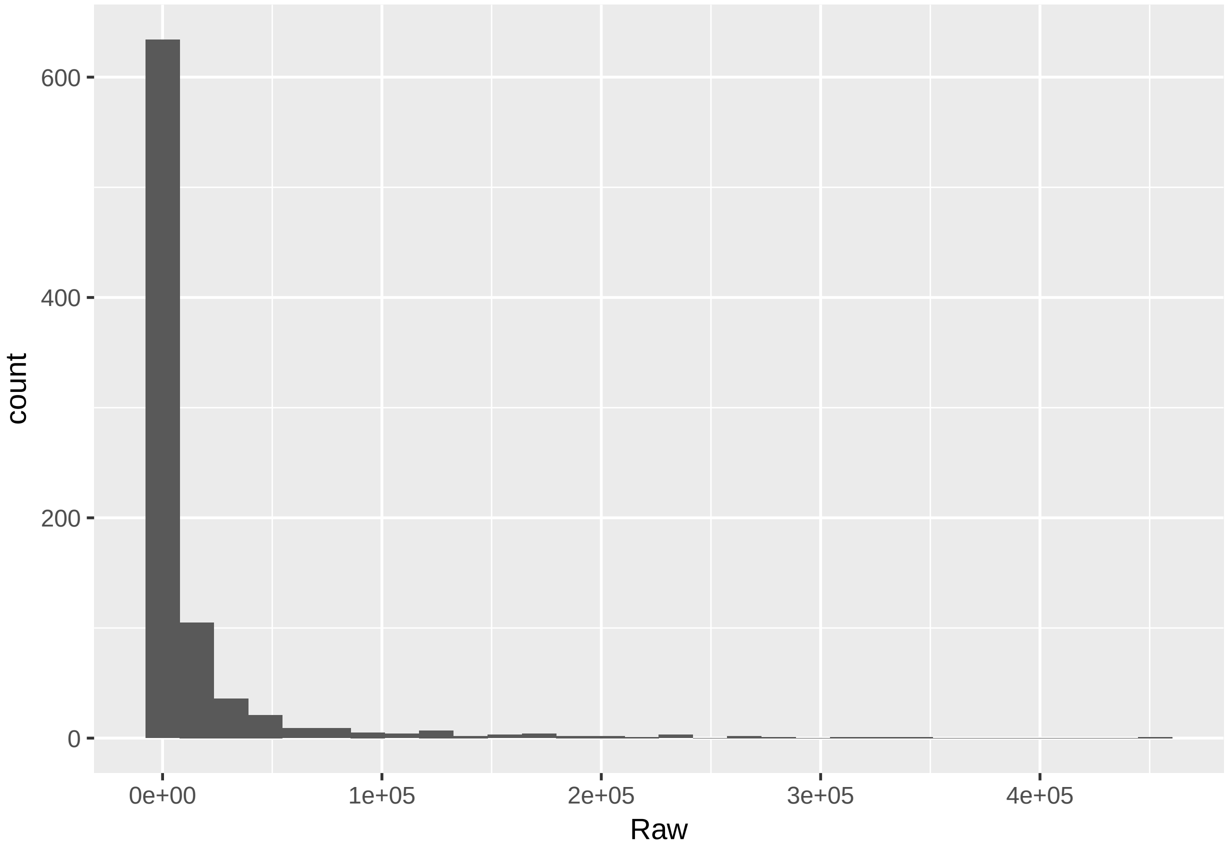
```
ggplot(Pokemon.Attributes, aes(x = BST, y = Type.1)) + geom_point() + theme(axis.text.y = element_text(
```



```
detach(Pokemon.Attributes)
detach(OUPokemon)
detach(RUPokemon)
```

```
OUPokemon <- merge(Pokemon.Attributes,OverUsed,by="Pokemon")
#OUPokemon <- OUPokemon[order(OUPokemon$Raw,decreasing = TRUE),]
RUPokemon <- merge(Pokemon.Attributes,RandomBattle,by="Pokemon")
UUPokemon <- merge(Pokemon.Attributes,UnderUsed,by="Pokemon")
UberPokemon <- merge(Pokemon.Attributes,Uber,by="Pokemon")
LCPokemon <- merge(Pokemon.Attributes,LC,by="Pokemon")

ggplot(OUPokemon, aes(x = Raw)) + geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

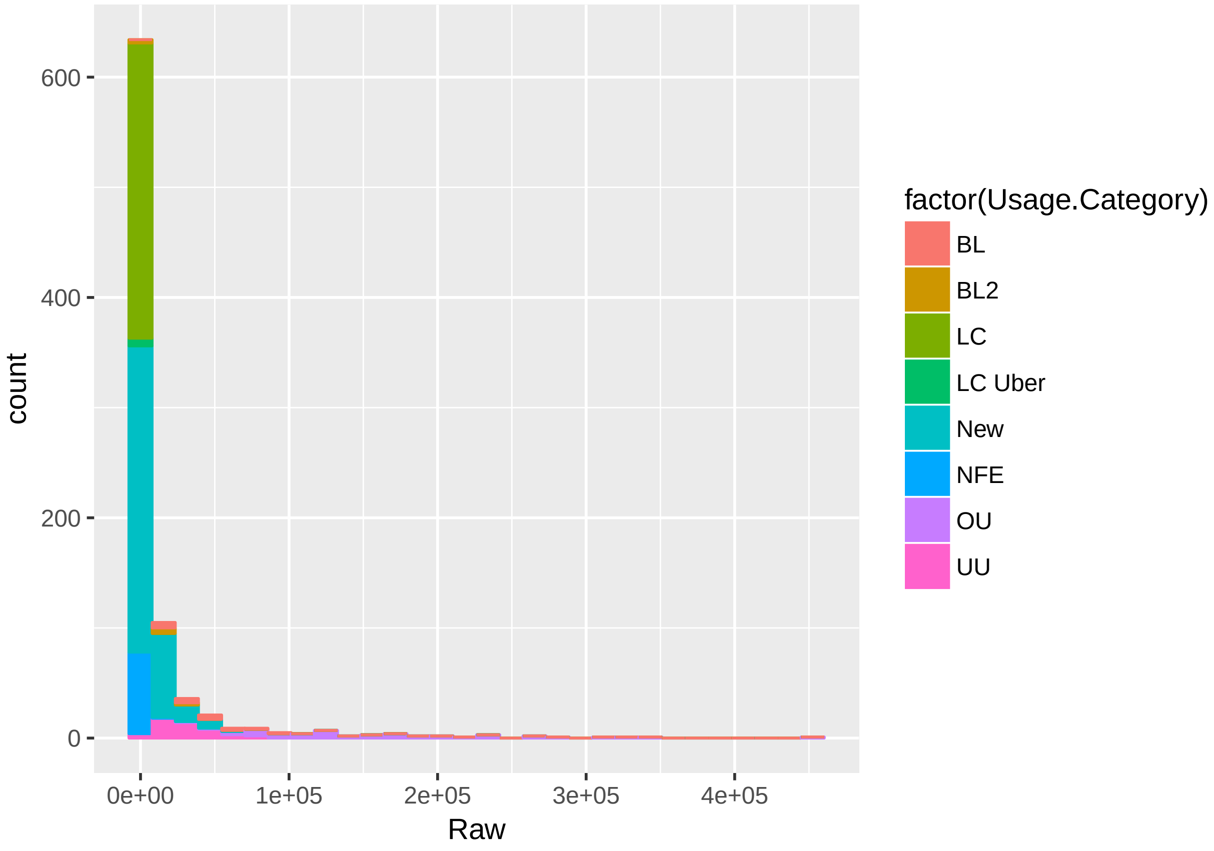


```
summary(OUPokemon$Raw)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##      1.0     78.8   1029.0  14890.0   8341.0 452500.0
```

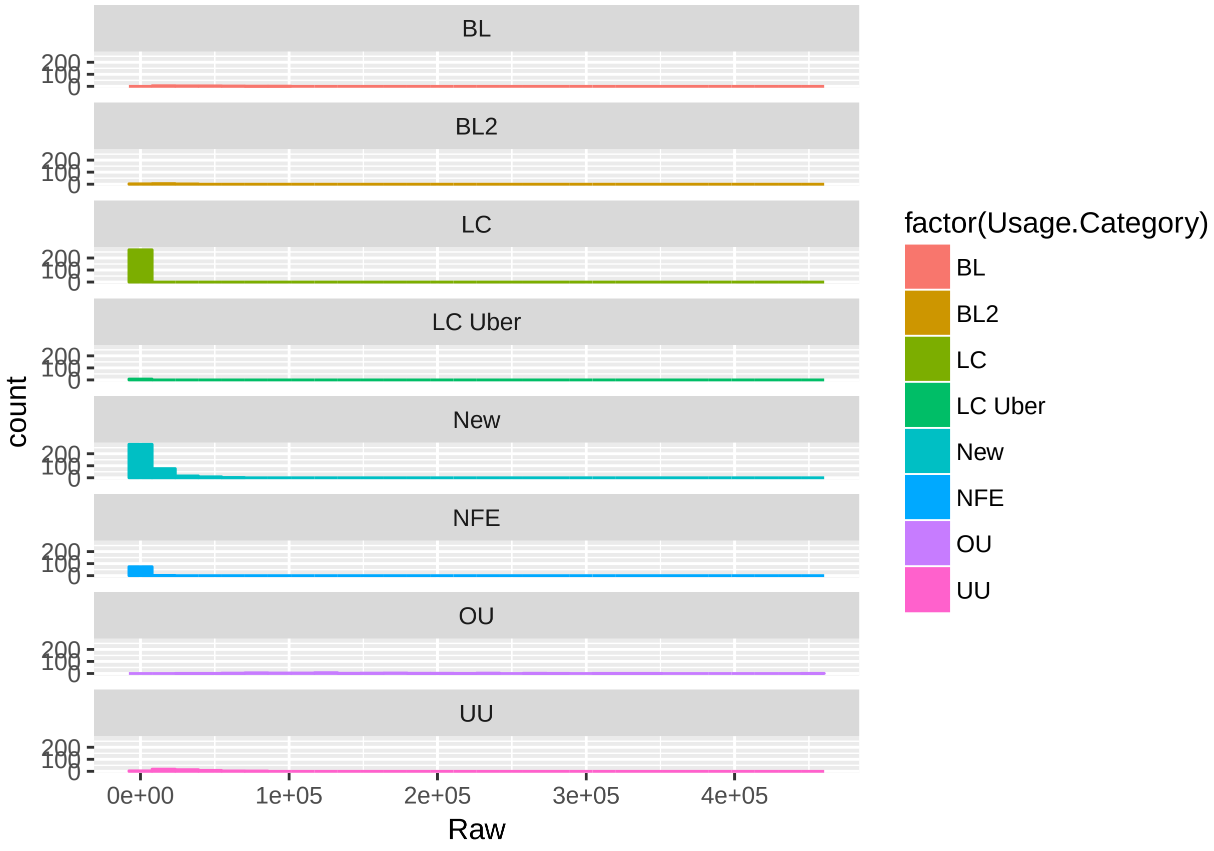
```
ggplot(OUPokemon, aes(x = Raw, color = factor(Usage.Category), fill = factor(Usage.Category))) + geom_hi
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



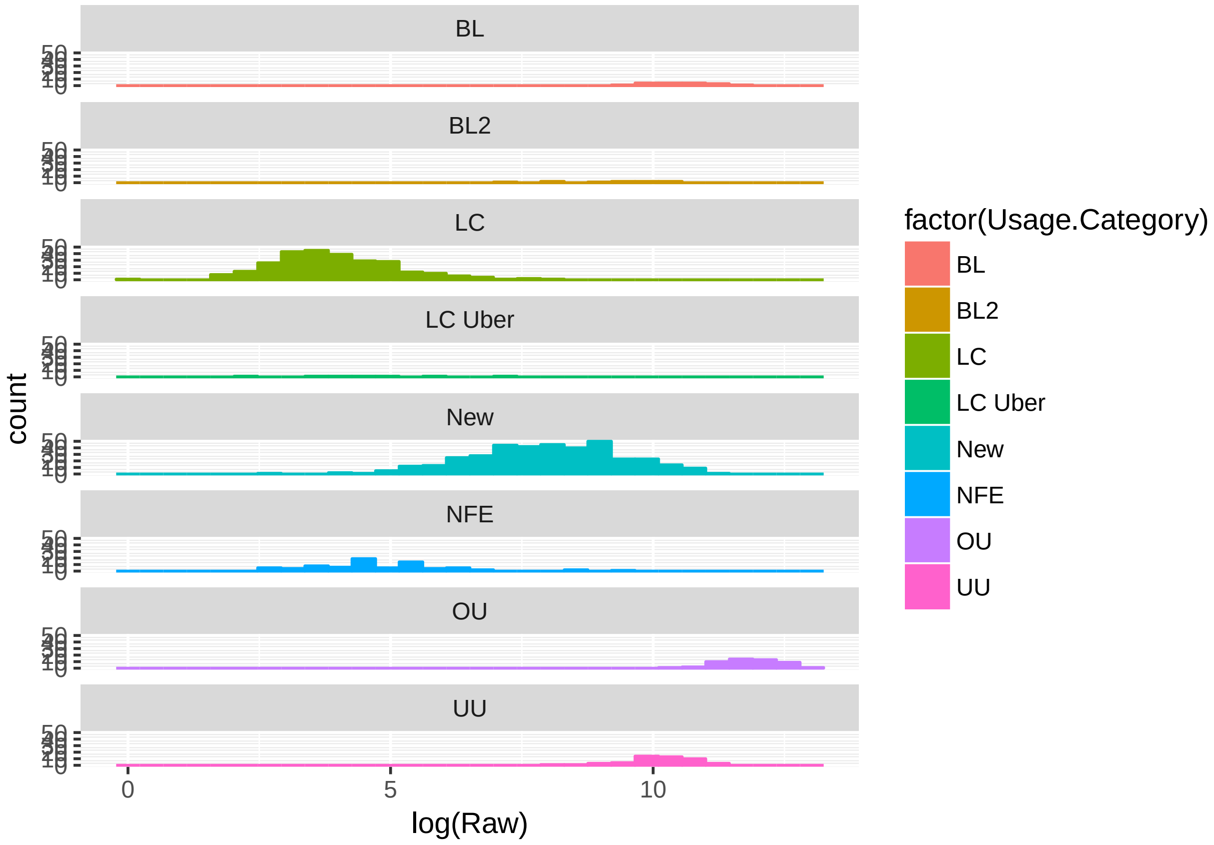
```
ggplot(OUPokemon, aes(x = Raw, color = factor(Usage.Category), fill = factor(Usage.Category))) + geom_hi
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



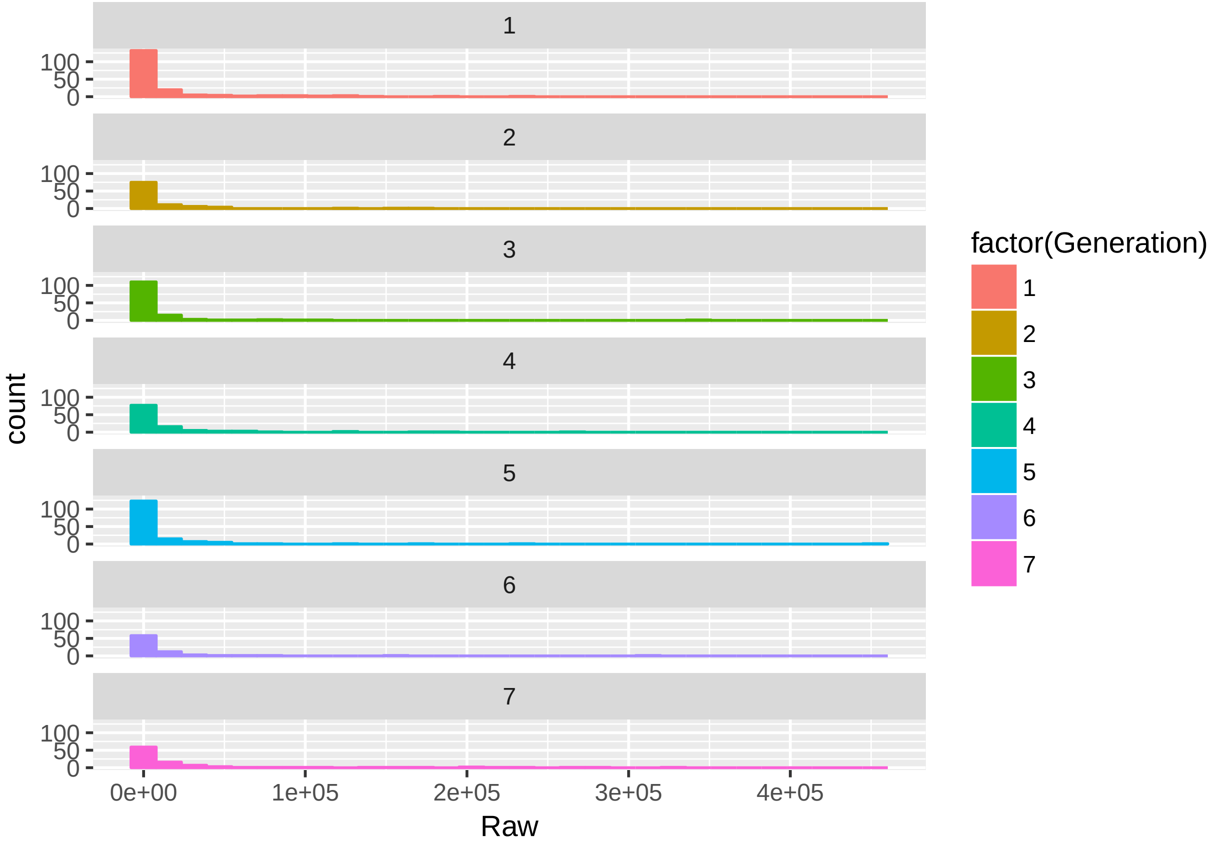
```
ggplot(OUPokemon, aes(x = log(Raw), color = factor(Usage.Category), fill = factor(Usage.Category))) + g
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



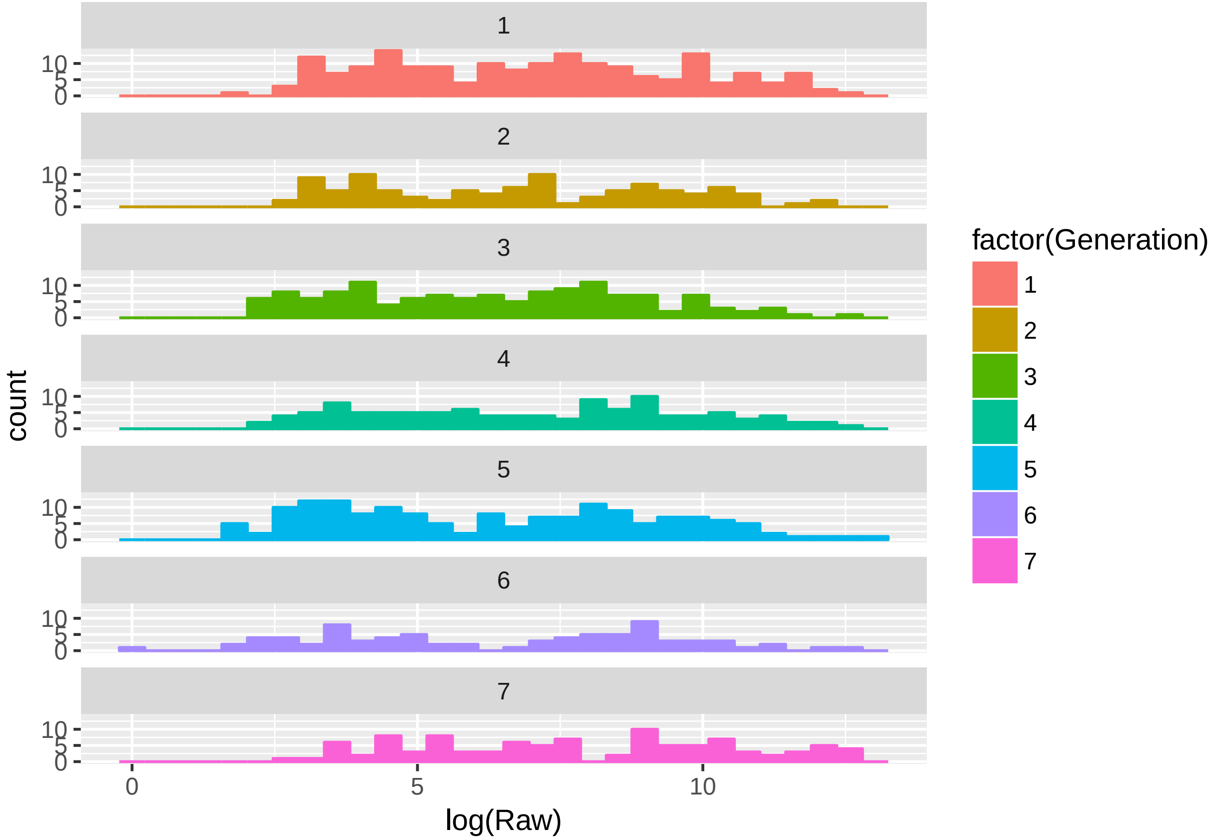
```
ggplot(OUPokemon, aes(x = Raw, color = factor(Generation), fill = factor(Generation))) + geom_histogram
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
ggplot(OUPokemon, aes(x = log(Raw), color = factor(Generation), fill = factor(Generation))) + geom_histo
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
ggplot(OUPokemon, aes(x = Raw, color = factor(Type.1), fill = factor(Type.1))) + geom_histogram() + face
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

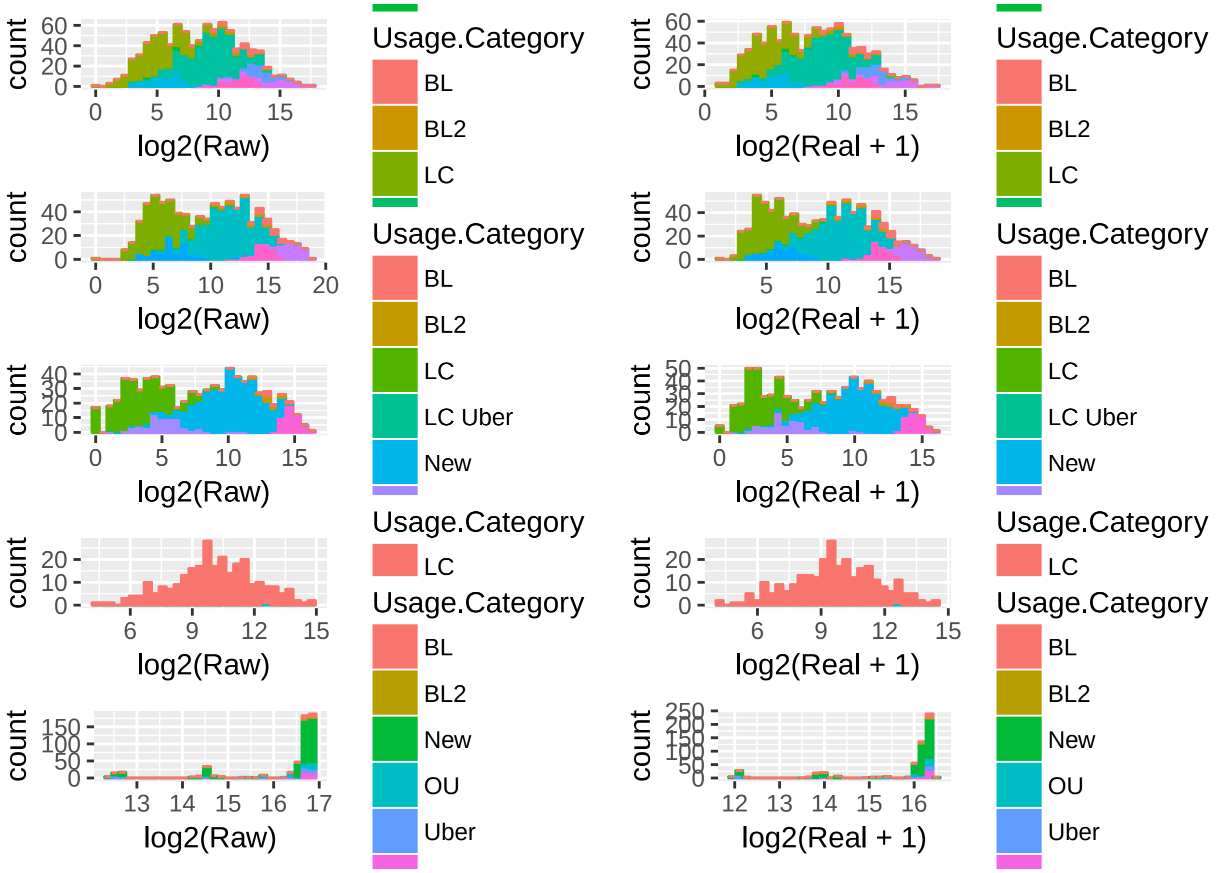
Dragon
Electric
Fairy
Fighting
Fire
Flying
Ghost
Grass
Ground
Ice
Normal
Poison
Psychic
Rock
Steel
count
40
factor(Type.1)
Bug
Dark
Dragon
Electric
Fairy
Fighting
Fire
Flying
Ghost
Grass
Ground
Ice
Normal
Poison
Psychic
Rock
Steel
Water

```r
ggplot(OUPokemon, aes(x = log(Raw), color = factor(Type.1), fill = factor(Type.1))) + geom_histogram() +
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

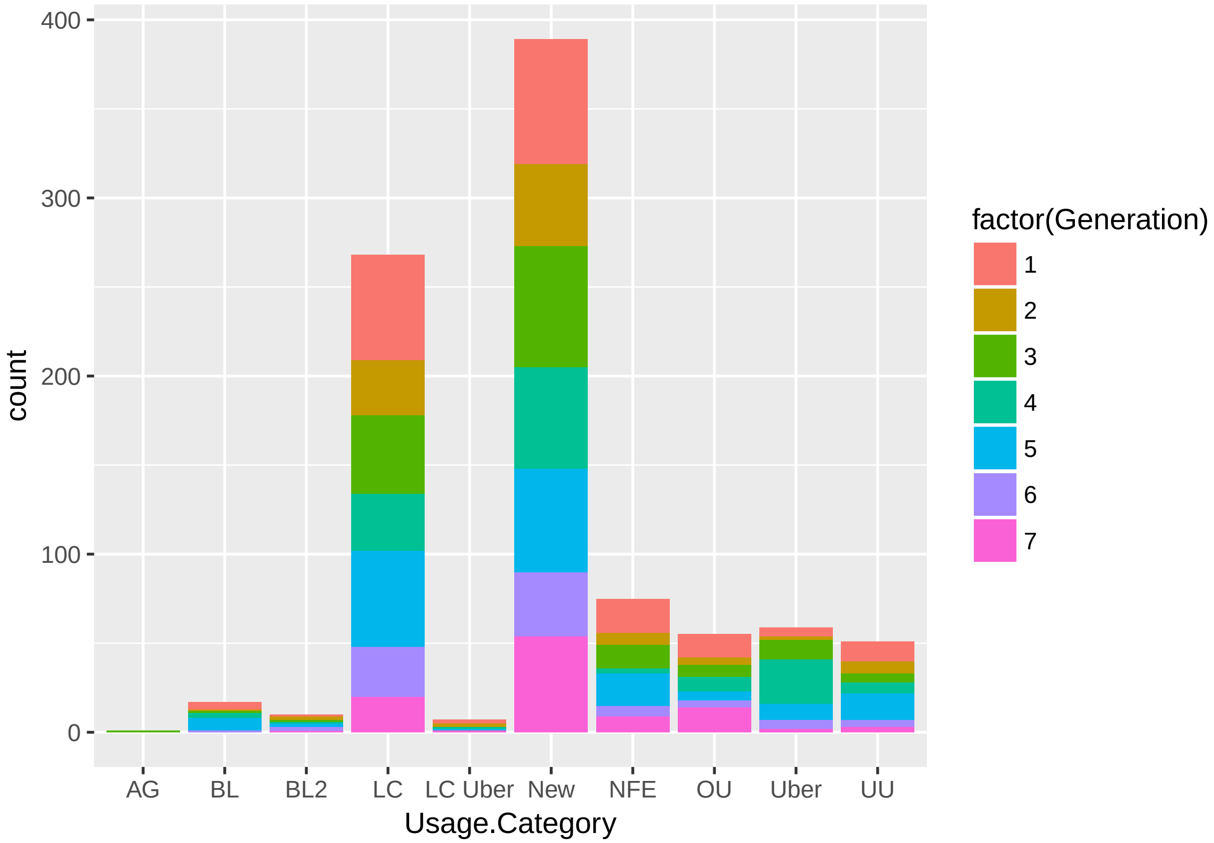
```r
pOU <- ggplot(OUPokemon, aes(x=log2(Raw), fill=Usage.Category, color=Usage.Category)) + geom_histogram(
pUU <- ggplot(UUPokemon, aes(x=log2(Raw), fill=Usage.Category, color=Usage.Category)) + geom_histogram(
pUber <- ggplot(UberPokemon, aes(x=log2(Raw), fill=Usage.Category, color=Usage.Category)) + geom_histog
pLC <- ggplot(LCPokemon, aes(x=log2(Raw), fill=Usage.Category, color=Usage.Category)) + geom_histogram(
pRandom <- ggplot(RUPokemon, aes(x=log2(Raw), fill=Usage.Category, color=Usage.Category)) + geom_histog

pOURe <- ggplot(OUPokemon, aes(x=log2(Real+1), fill=Usage.Category, color=Usage.Category)) + geom_histo
pUURe <- ggplot(UUPokemon, aes(x=log2(Real+1), fill=Usage.Category, color=Usage.Category)) + geom_histo
pUberRe <- ggplot(UberPokemon, aes(x=log2(Real+1), fill=Usage.Category, color=Usage.Category)) + geom_hi
pLCRe <- ggplot(LCPokemon, aes(x=log2(Real+1), fill=Usage.Category, color=Usage.Category)) + geom_histo
pRandomRe <- ggplot(RUPokemon, aes(x=log2(Real+1), fill=Usage.Category, color=Usage.Category)) + geom_hi

#grid.arrange(pUber, pOU, pUU, pLC, pRandom)
grid.arrange(pUber,pUberRe, pOU,pOURe, pUU,pUURe, pLC,pLCRe, pRandom,pRandomRe, ncol=2)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



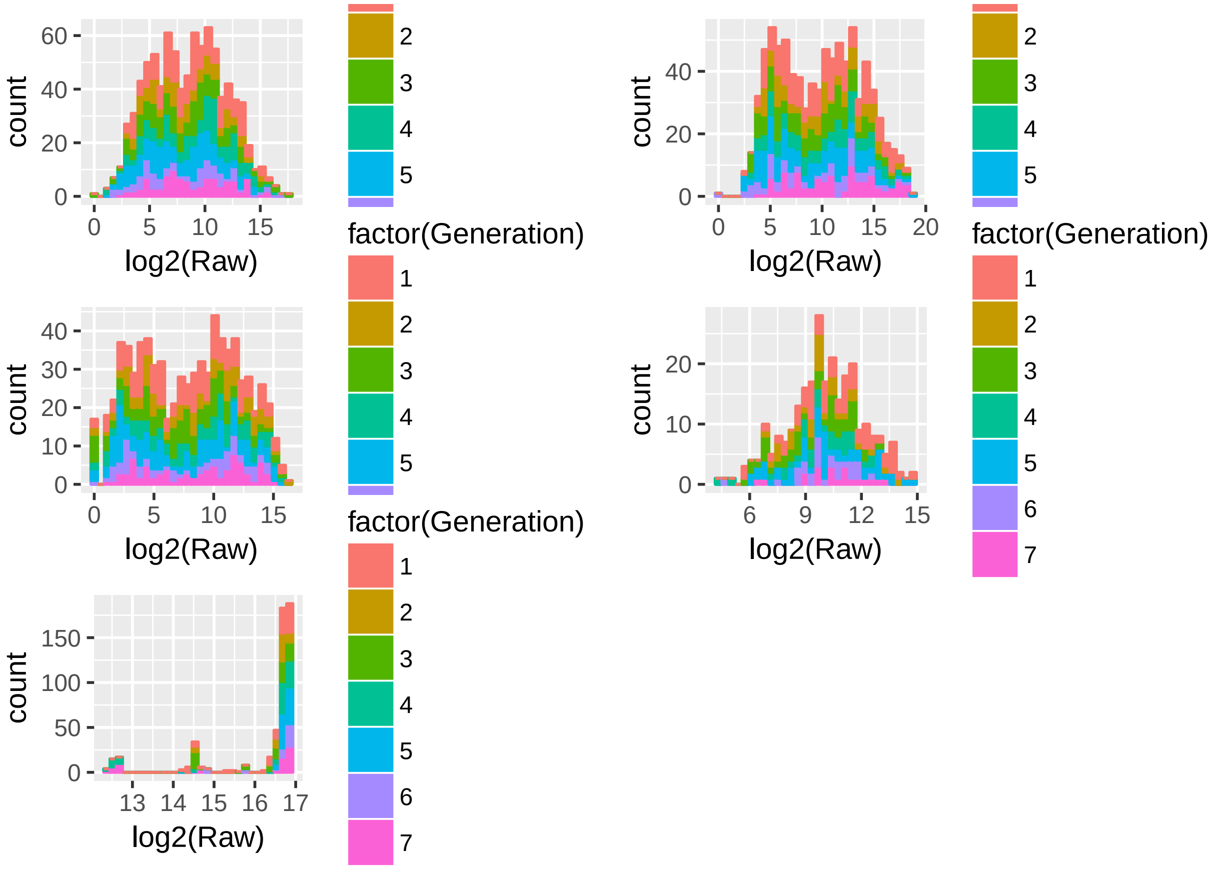
```
ggplot(Pokemon.Attributes,aes(x=Usage.Category,fill=factor(Generation))) + geom_histogram(stat="count")
```

```
## Warning: Ignoring unknown parameters: binwidth, bins, pad
```

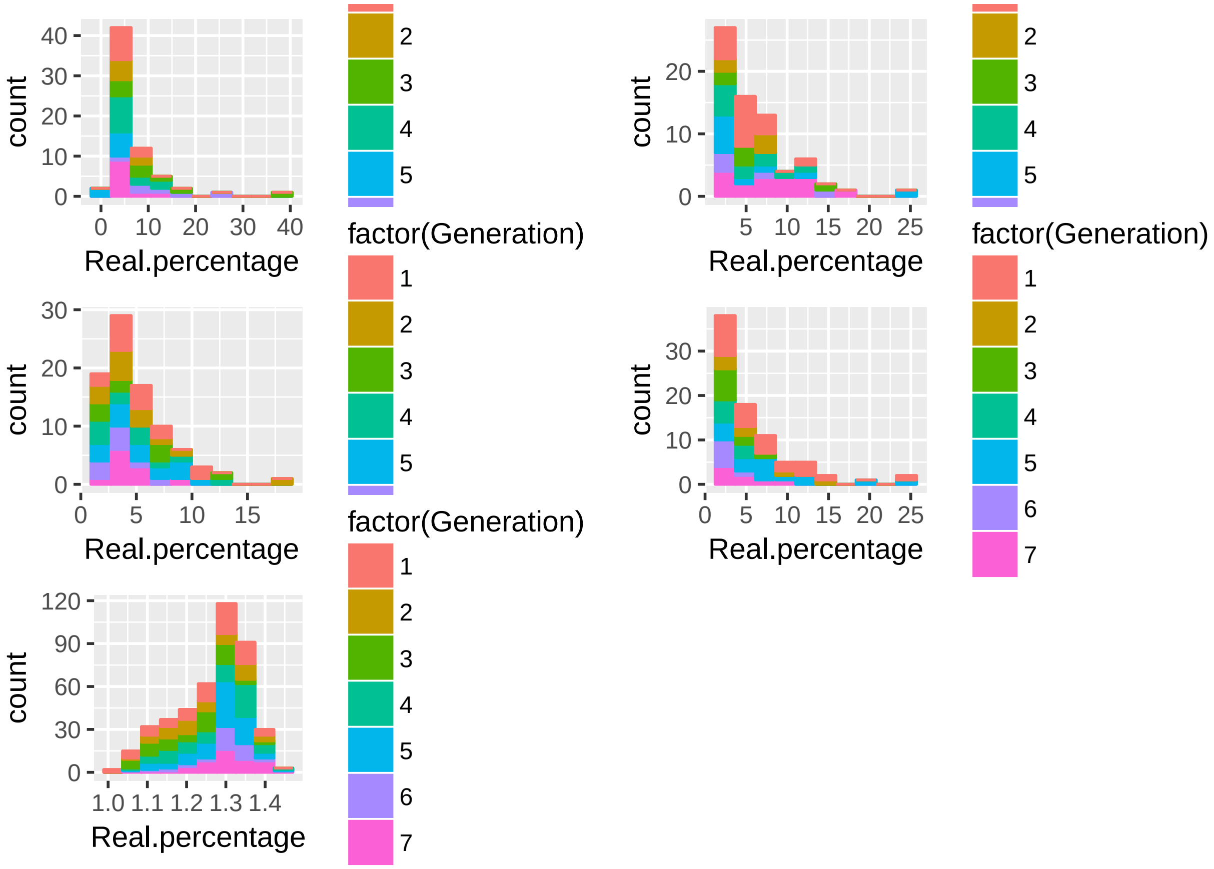


```
pOU <- ggplot(OUPokemon, aes(x=log2(Raw), fill=factor(Generation), color=factor(Generation))) + geom_hi
pUU <- ggplot(UUPokemon, aes(x=log2(Raw), fill=factor(Generation), color=factor(Generation))) + geom_hi
pUber <- ggplot(UberPokemon, aes(x=log2(Raw), fill=factor(Generation), color=factor(Generation))) + geom
pLC <- ggplot(LCPokemon, aes(x=log2(Raw), fill=factor(Generation), color=factor(Generation))) + geom_hi
pRandom <- ggplot(RUPokemon, aes(x=log2(Raw), fill=factor(Generation), color=factor(Generation))) + geom

grid.arrange(pUber, pOU, pUU, pLC, pRandom)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

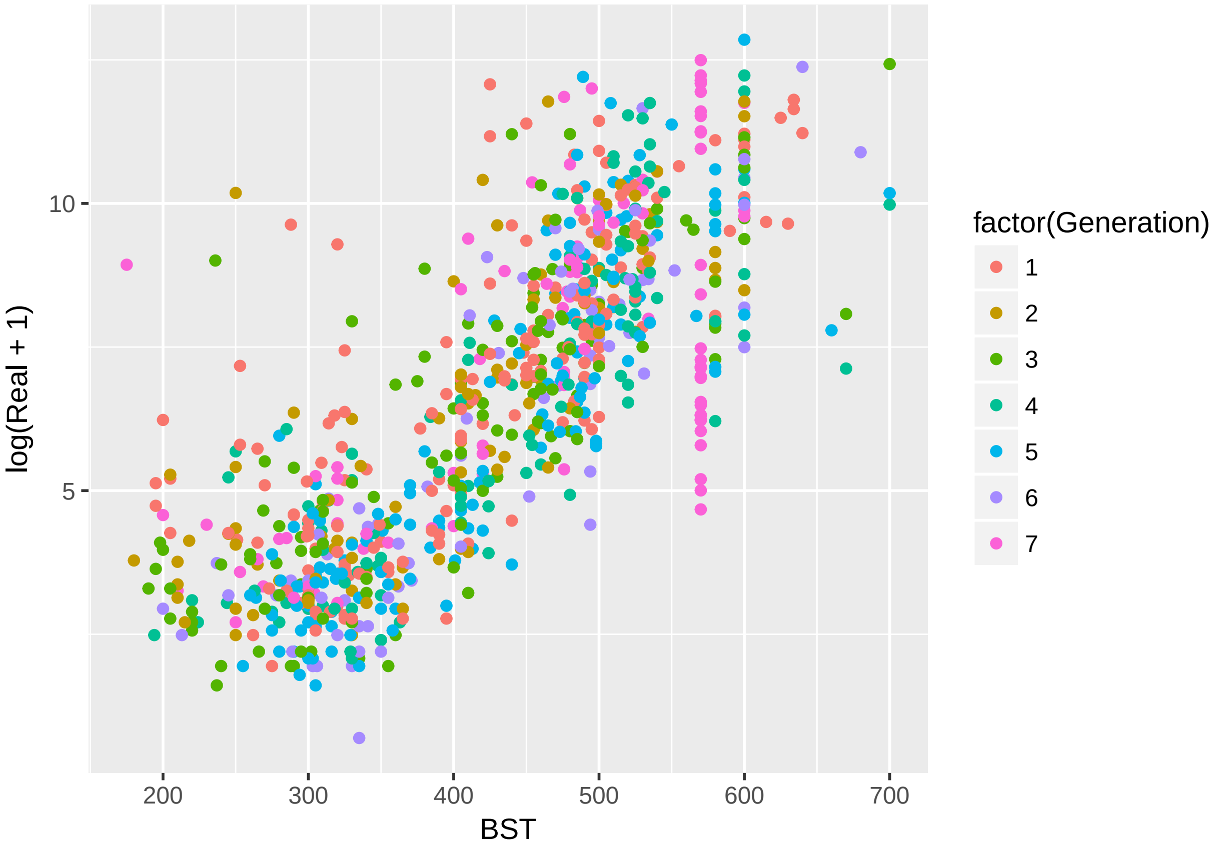


```
pOU <- ggplot(OUPokemon[(OUPokemon$Real.percentage > 2),], aes(x=Real.percentage, fill=factor(Generatio
pUU <- ggplot(UUPokemon[(UUPokemon$Real.percentage > 2),], aes(x=Real.percentage, fill=factor(Generatio
pUber <- ggplot(UberPokemon[(UberPokemon$Real.percentage > 2),], aes(x=Real.percentage, fill=factor(Gene
pLC <- ggplot(LCPokemon[(LCPokemon$Real.percentage > 2),], aes(x=Real.percentage, fill=factor(Generatio
pRandom <- ggplot(RUPokemon[(RUPokemon$Real.percentage > 1),], aes(x=Real.percentage, fill=factor(Genera

grid.arrange(pUber, pOU, pUU, pLC, pRandom)
```

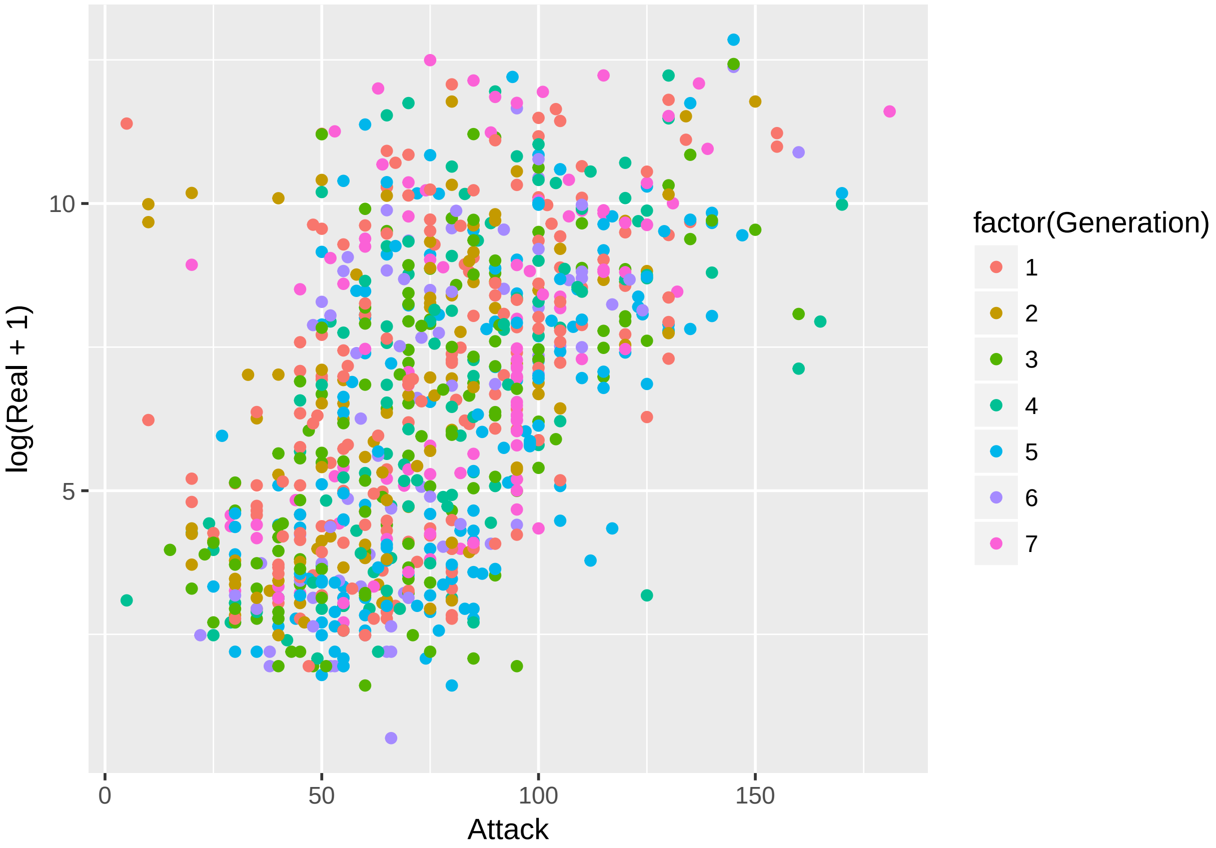


```
#pOU <- ggplot(OUPokemon[(OUPokemon$Real.percentage > 10),], aes(x=Real.percentage, fill=factor(Generat
#pUU <- ggplot(UUPokemon[(UUPokemon$Real.percentage > 10),], aes(x=Real.percentage, fill=factor(Generat
#pUber <- ggplot(UberPokemon[(UberPokemon$Real.percentage > 10),], aes(x=Real.percentage, fill=factor(G
#pLC <- ggplot(LCPokemon[(LCPokemon$Real.percentage > 10),], aes(x=Real.percentage, fill=factor(Generat
#pRandom <- ggplot(RUPokemon[(RUPokemon$Real.percentage > 1.3),], aes(x=Real.percentage, fill=factor(Ge

#grid.arrange(pUber, pOU, pUU, pLC, pRandom)
```

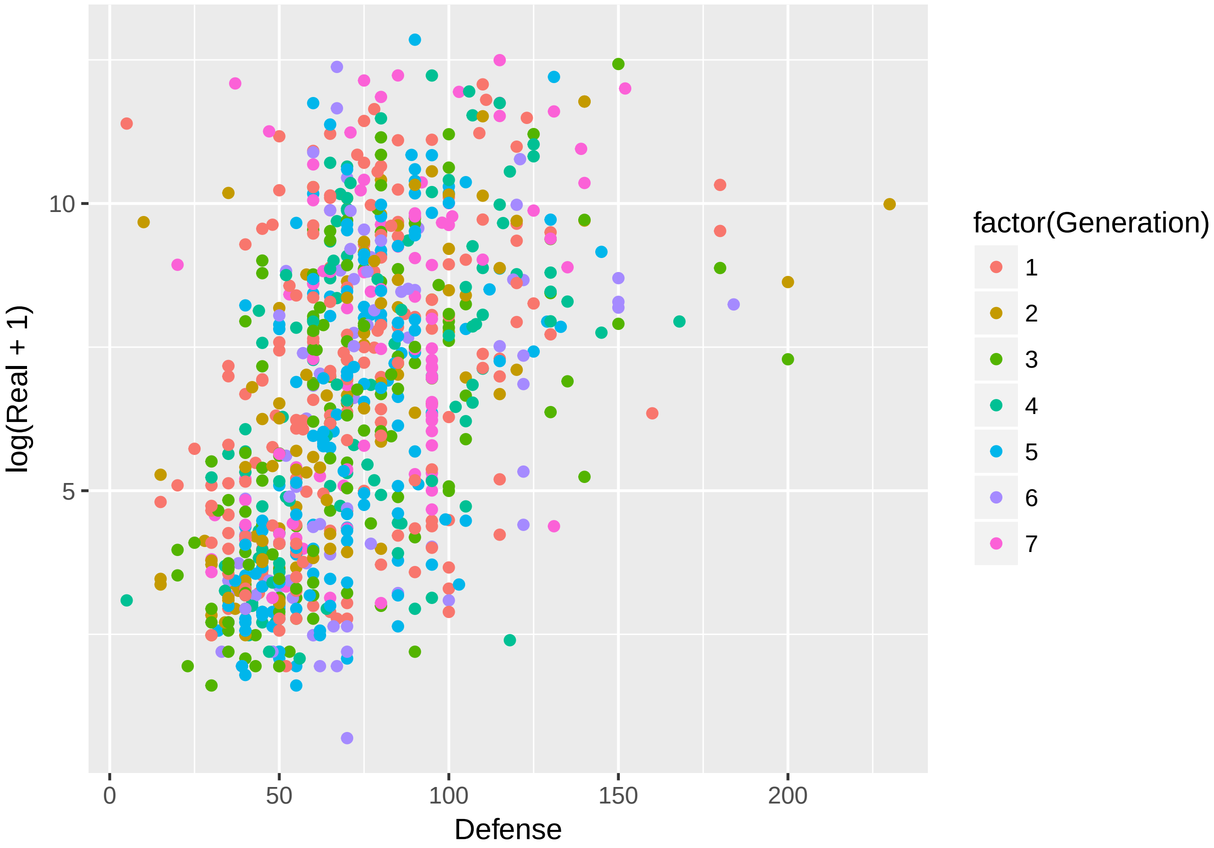
```
ggplot(OUPokemon, aes(x = BST, y = log(Real+1), color = factor(Generation))) + geom_point()
```



```
ggplot(OUPokemon, aes(x = Attack, y = log(Real+1), color = factor(Generation))) + geom_point()
```



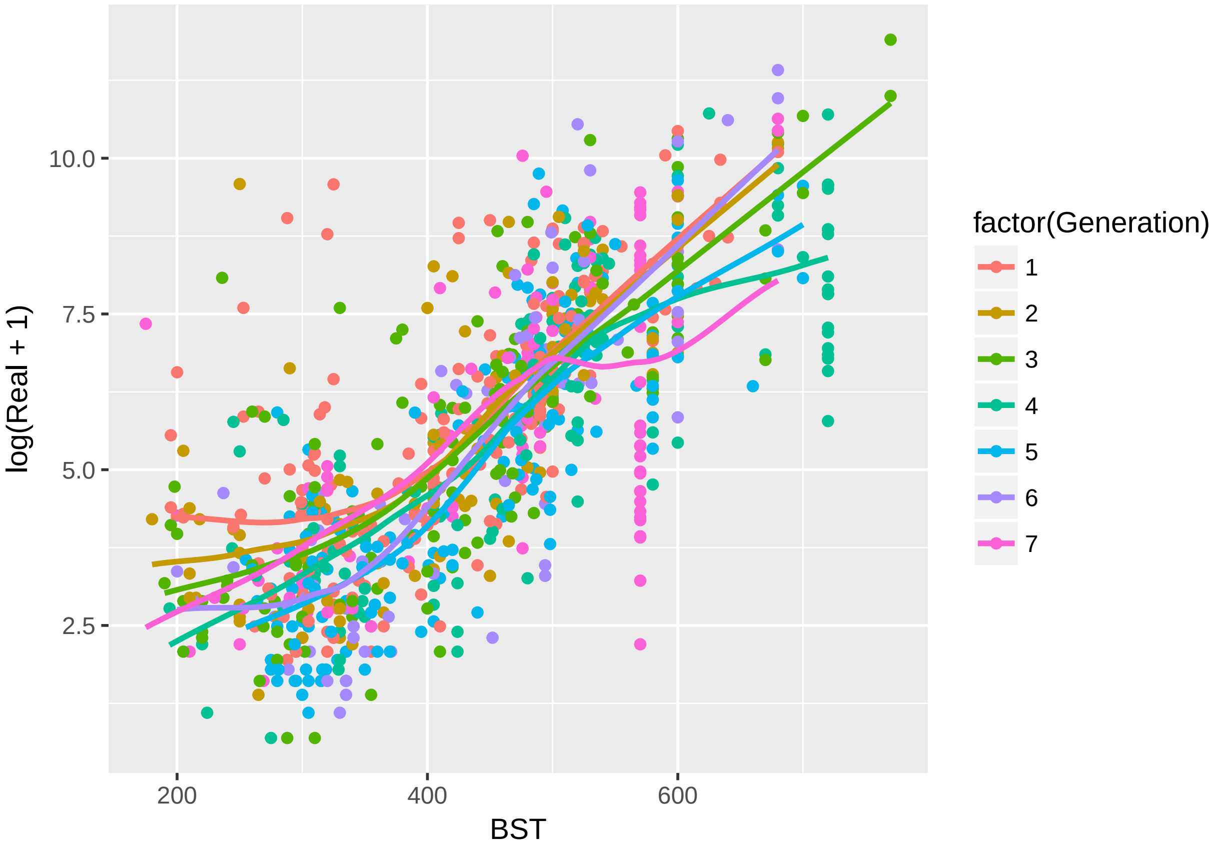
```r
ggplot(OUPokemon, aes(x = Defense, y = log(Real+1), color = factor(Generation))) + geom_point()
```



```
#ggplot(OUPokemon, aes(x = BST, y = log(Real+1), color = Ability.1)) + geom_point()
#ggplot(OUPokemon, aes(x = BST, y = log(Real+1), group = Ability.1, color = Ability.1)) + geom_point()
```

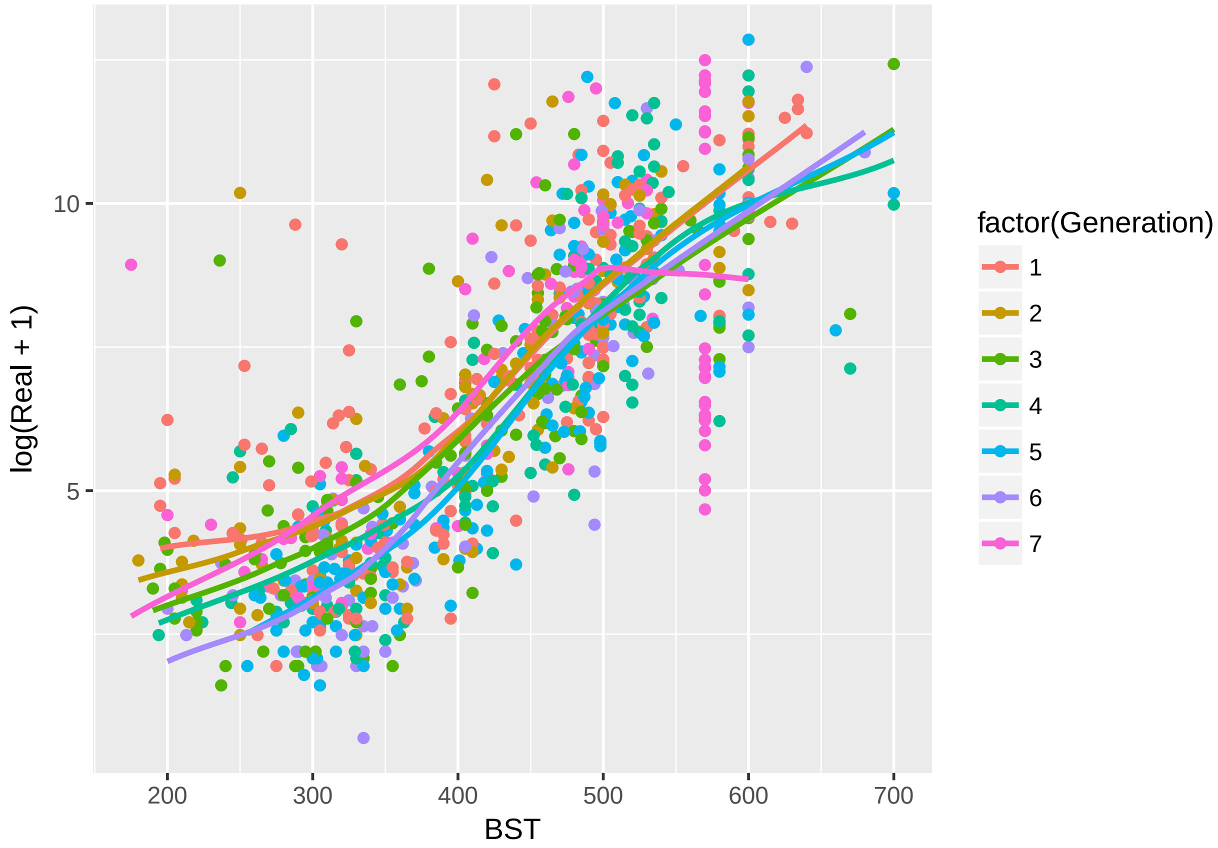
```
ggplot(UberPokemon, aes(x = BST, y = log(Real+1), group = Generation, color = factor(Generation))) + ge
```

```
## `geom_smooth()` using method = 'loess'
```



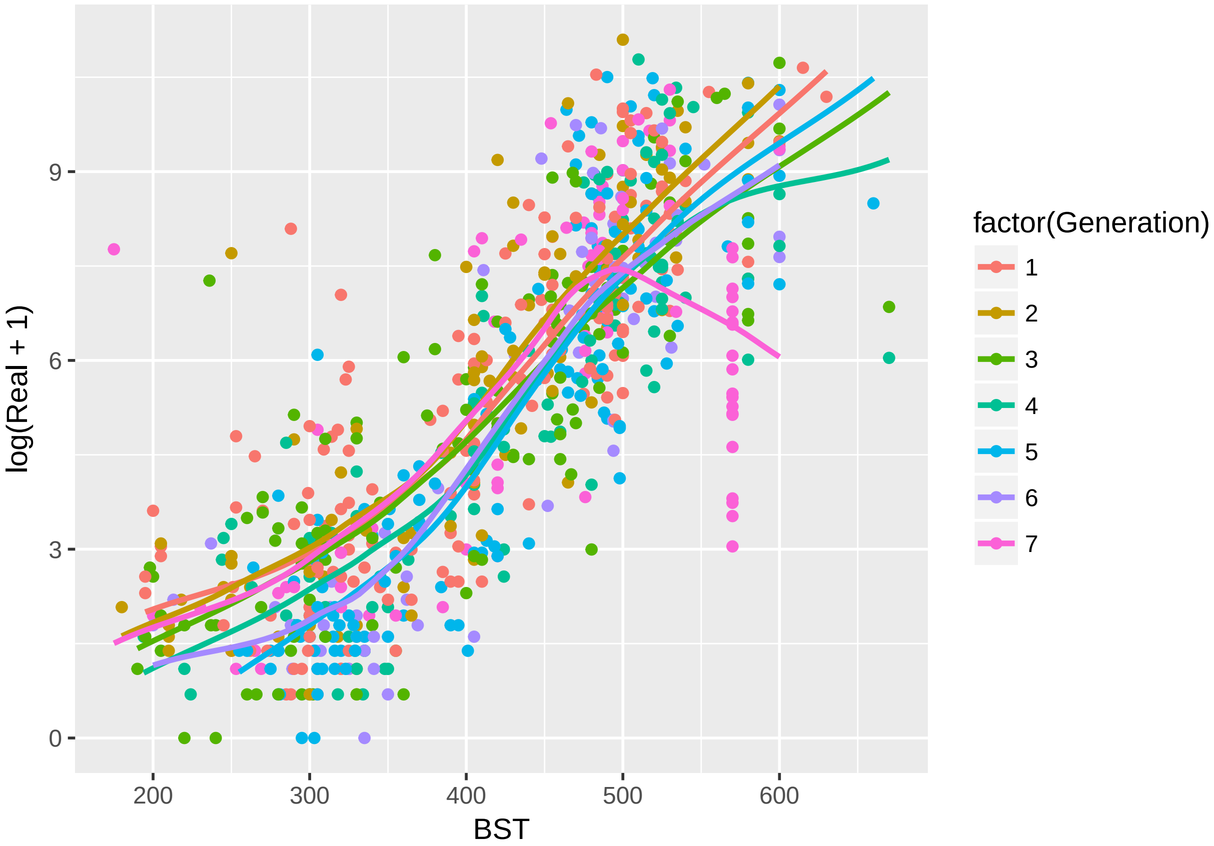
```
ggplot(OUPokemon, aes(x = BST, y = log(Real+1), group = Generation, color = factor(Generation))) + geom
```

```
## `geom_smooth()` using method = 'loess'
```



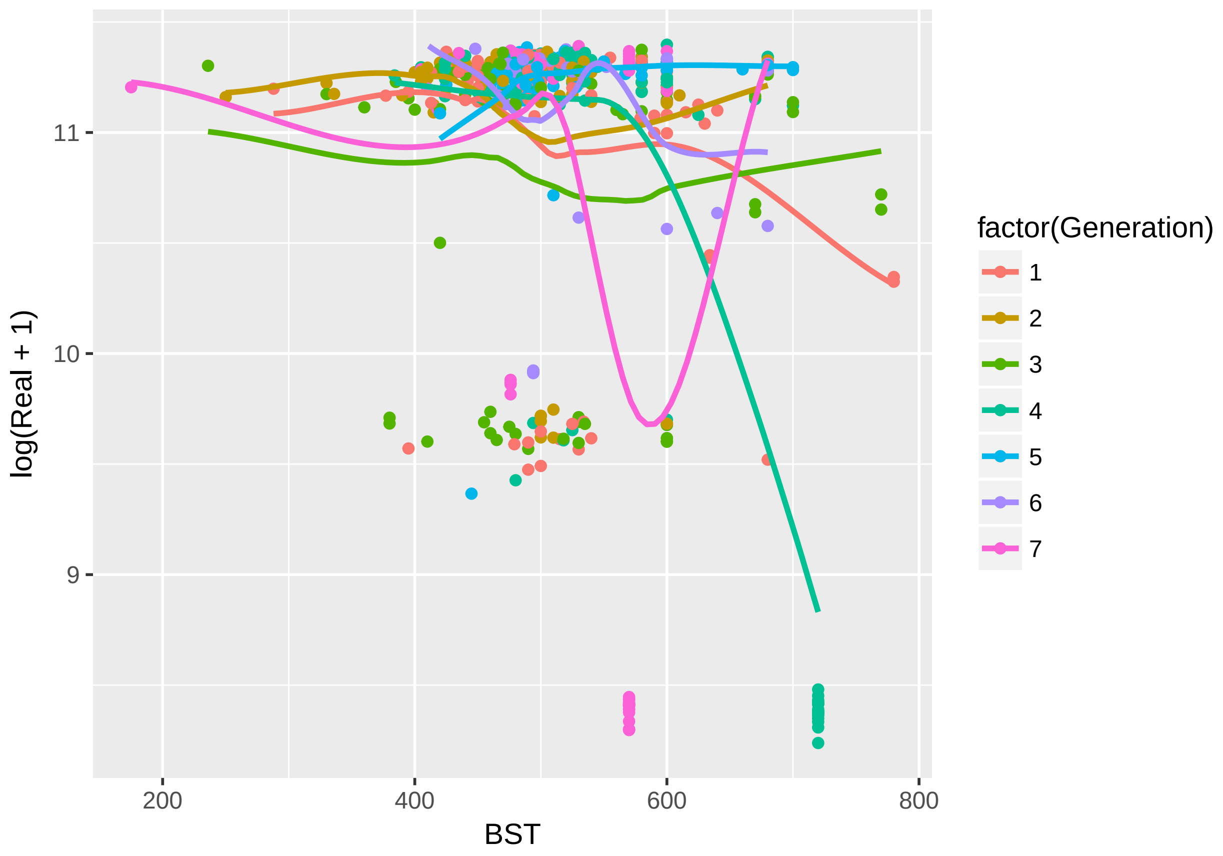
```
ggplot(UUPokemon, aes(x = BST, y = log(Real+1), group = Generation, color = factor(Generation))) + geom_
```

```
## `geom_smooth()` using method = 'loess'
```

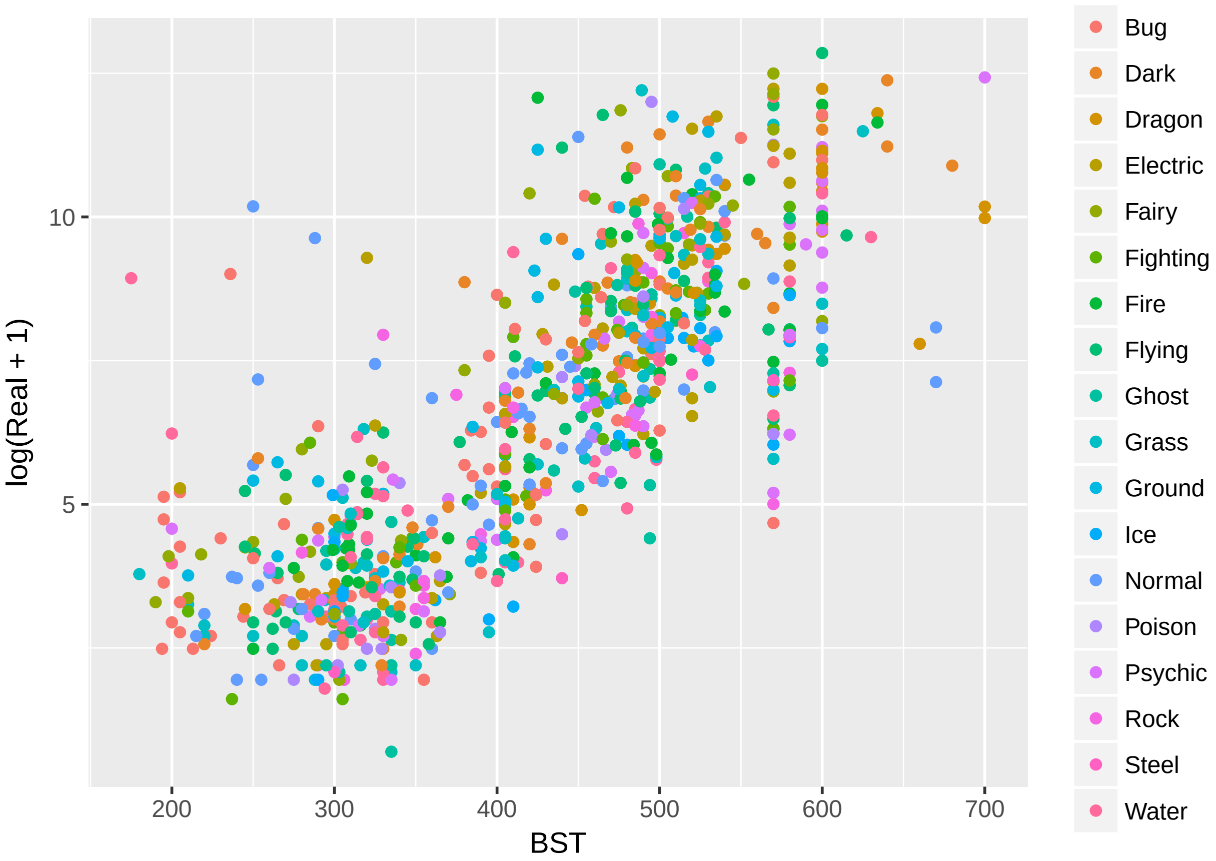


```
ggplot(RUPokemon, aes(x = BST, y = log(Real+1), group = Generation, color = factor(Generation))) + geom_
```

```
## `geom_smooth()` using method = 'loess'
```

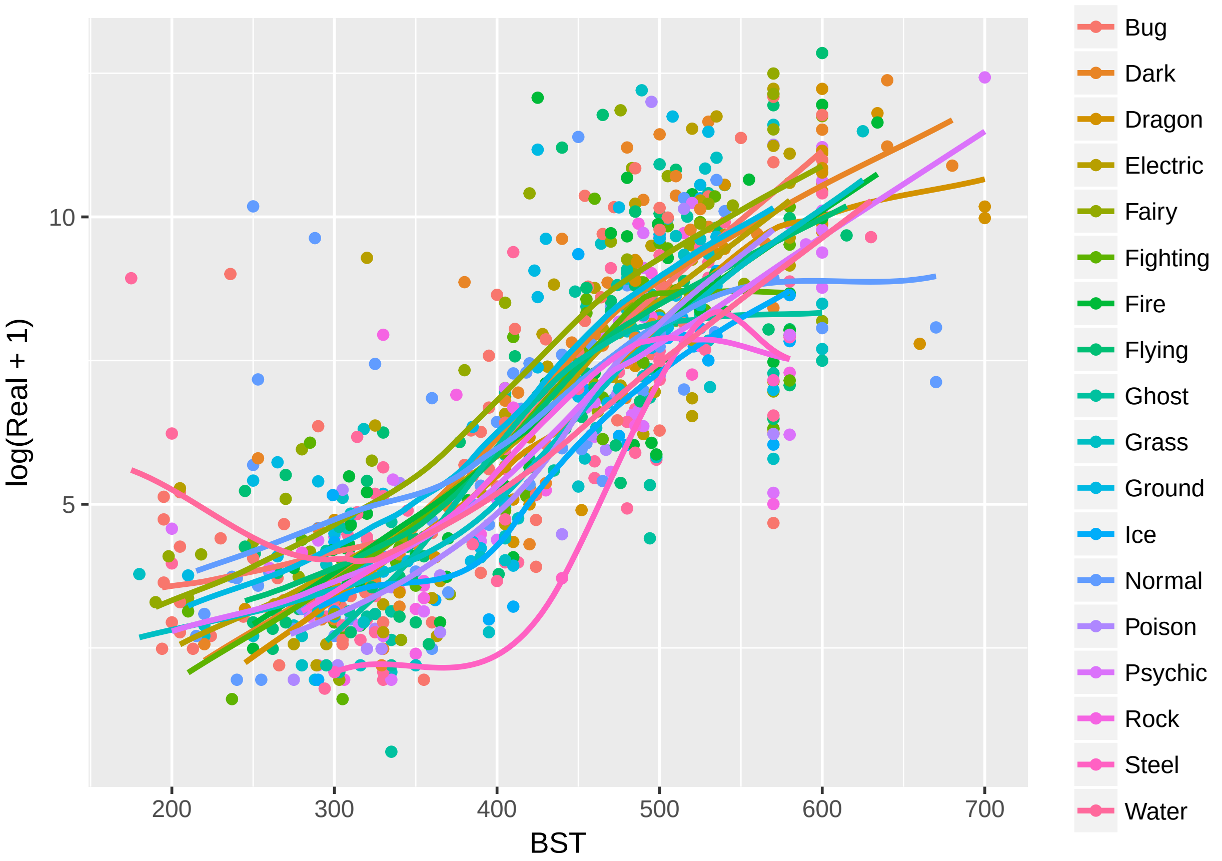


```
ggplot(OUPokemon, aes(x = BST, y = log(Real+1), color = Type.1)) + geom_point()
```



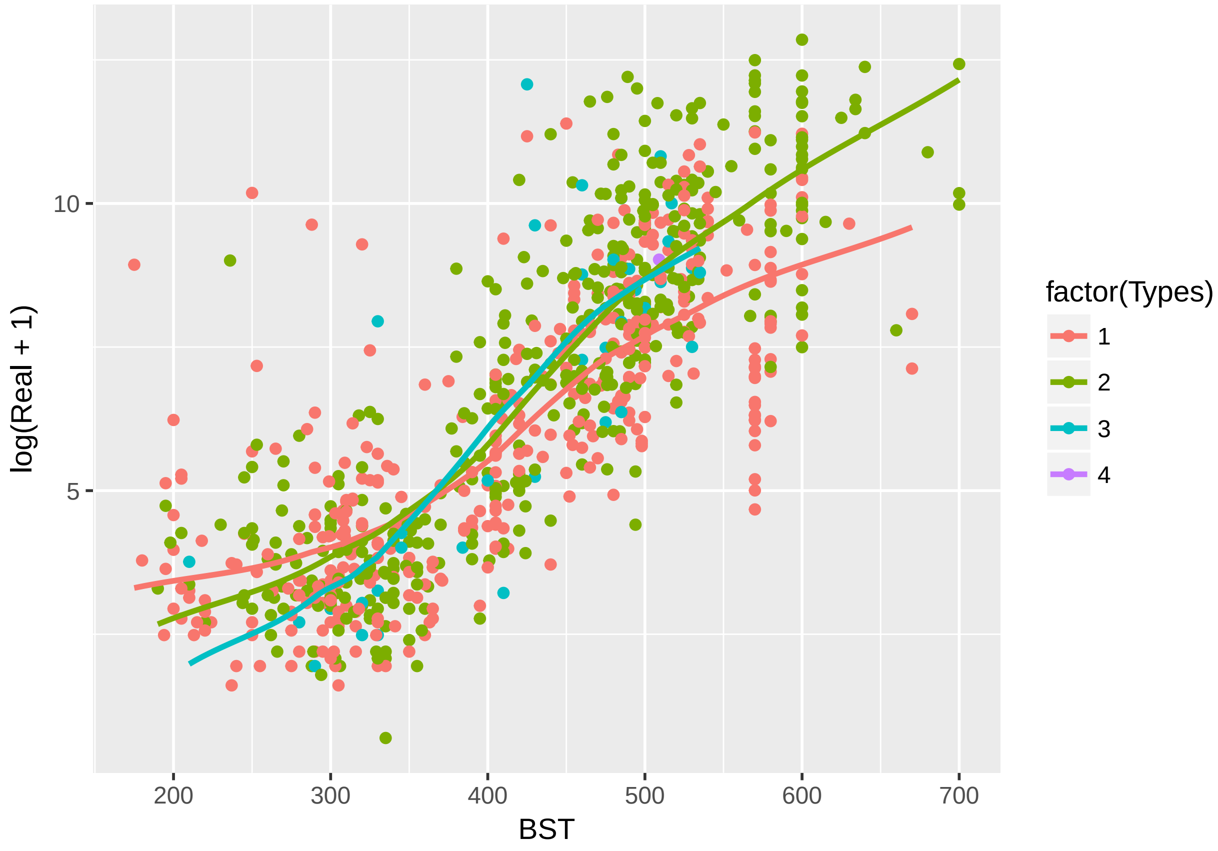
```
ggplot(OUPokemon, aes(x = BST, y = log(Real+1), group = Type.1, color = Type.1)) + geom_point() + geom_
```

```
## `geom_smooth()` using method = 'loess'
```



```
ggplot(OUPokemon, aes(x = BST, y = log(Real+1), group = Types, color = factor(Types))) + geom_point() +
```

```
## `geom_smooth()` using method = 'loess'
```



```
constant.glm = glm(Raw ~ 1, family = poisson, data = OUPokemon)
display(constant.glm)
```

```
## glm(formula = Raw ~ 1, family = poisson, data = OUPokemon)
##             coef.est coef.se
## (Intercept) 9.61     0.00
## ---
##   n = 854, k = 1
##   residual deviance = 42179888.3, null deviance = 42179888.3 (difference = 0.0)
```

```
offset.glm = glm(Raw ~ 1, family = poisson, offset = log(BST), data = OUPokemon)
display(offset.glm)
```

```
## glm(formula = Raw ~ 1, family = poisson, data = OUPokemon, offset = log(BST))
##             coef.est coef.se
## (Intercept) 3.56     0.00
## ---
##   n = 854, k = 1
##   residual deviance = 35972698.8, null deviance = 35972698.8 (difference = 0.0)
```

```
bst.glm = glm(Raw ~ BST, family = poisson, data = OUPokemon)
display(bst.glm)
```

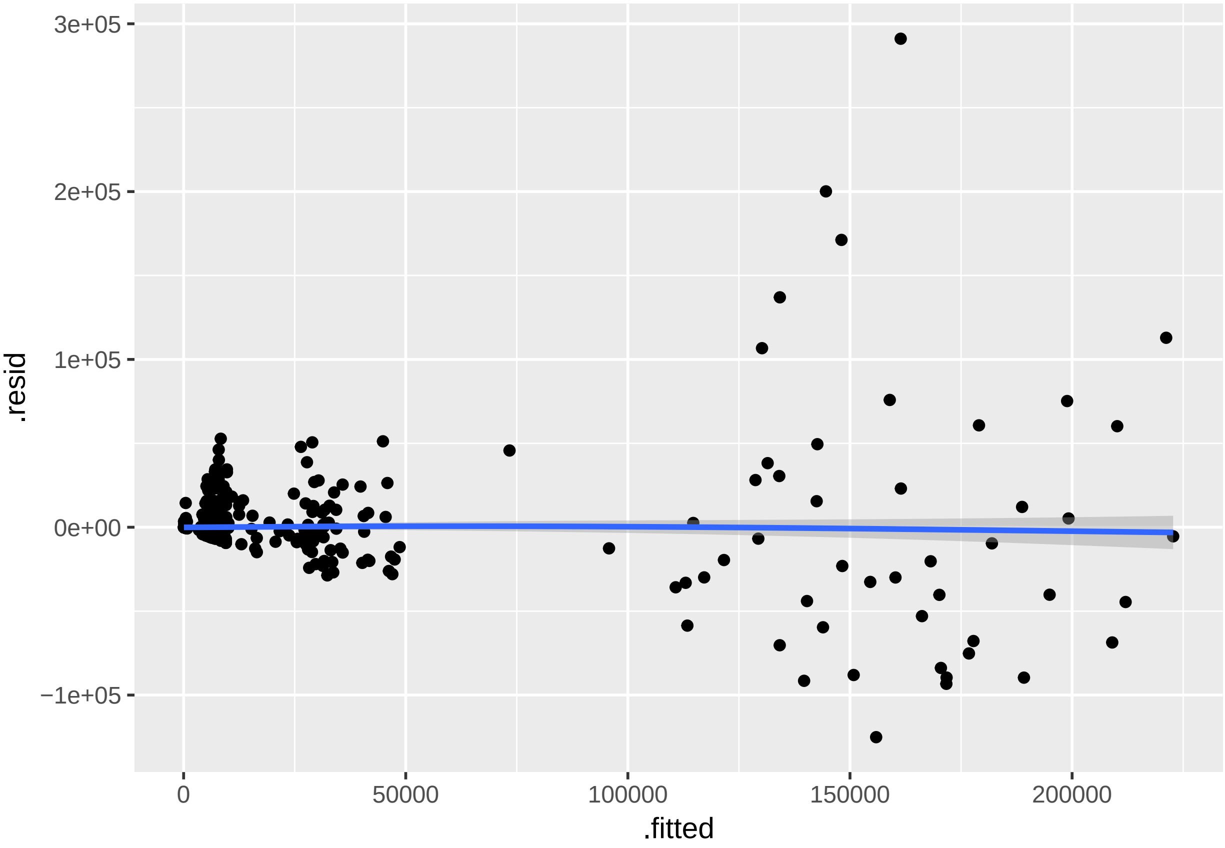
```
## glm(formula = Raw ~ BST, family = poisson, data = OUPokemon)
##             coef.est coef.se
## (Intercept) 3.20     0.00
## BST         0.01     0.00
## ---
##   n = 854, k = 2
##   residual deviance = 23927536.5, null deviance = 42179888.3 (difference = 18252351.8)
```

```
bst.UC.glm = glm(Raw ~ Attack + Defense + Special + Sp.Attack + Sp.Defense + HP +factor(Usage.Category)
deviance(bst.UC.glm)
```

```
## [1] 6117120
```

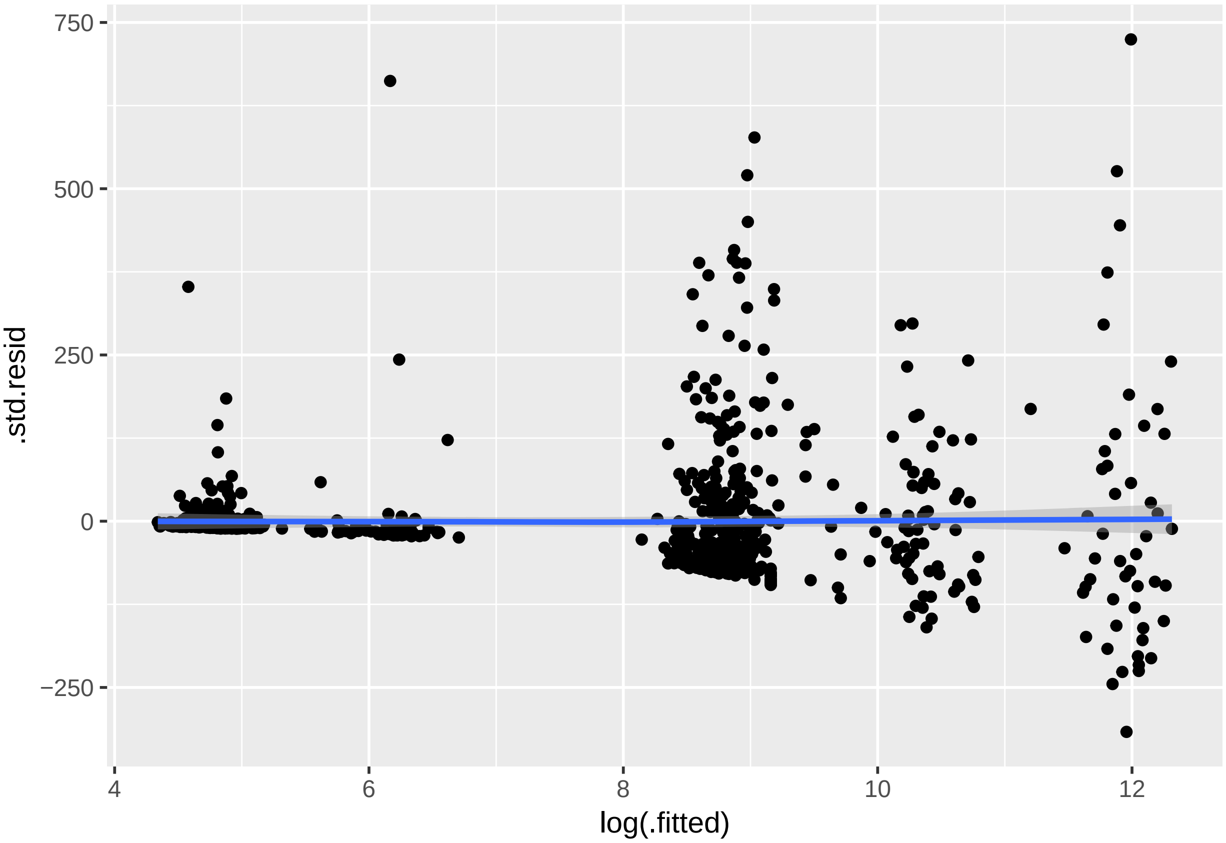
```
fitted = fitted.values(bst.UC.glm)
resid = residuals(bst.UC.glm, type = "response")
df = data.frame(OUPokemon, .fitted = fitted, .resid = resid)
ggplot(df, aes(x = .fitted, y = .resid)) + geom_point() + geom_smooth(span = 1, method.args = list(degre
```

```
## `geom_smooth()` using method = 'loess'
```



```
std.resid = resid/sqrt(fitted)
df$.std.resid = std.resid
ggplot(df, aes(x = log(.fitted), y = .std.resid)) + geom_point() + geom_smooth(span = 1, method.args = l
```

```
## `geom_smooth()` using method = 'loess'
```



```
overdispersion = sum(std.resid^2)/df.residual(bst.UC.glm)
overdispersion
```

```
## [1] 9679.143
```

```
sim1 = rpois(nrow(OUPokemon), lambda = fitted.values(bst.UC.glm))
summary(sim1)
```

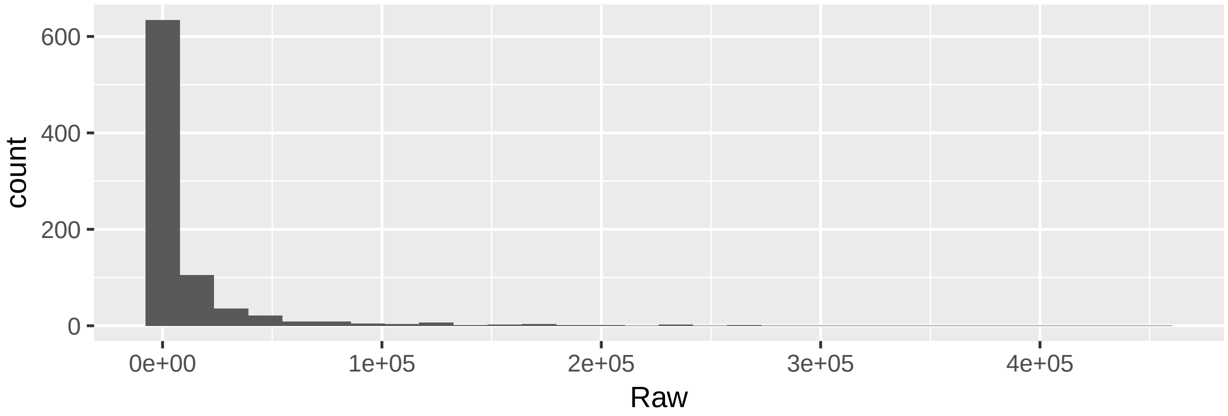
```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##     60.0    142.2   5530.0  14900.0   7473.0 222600.0
```

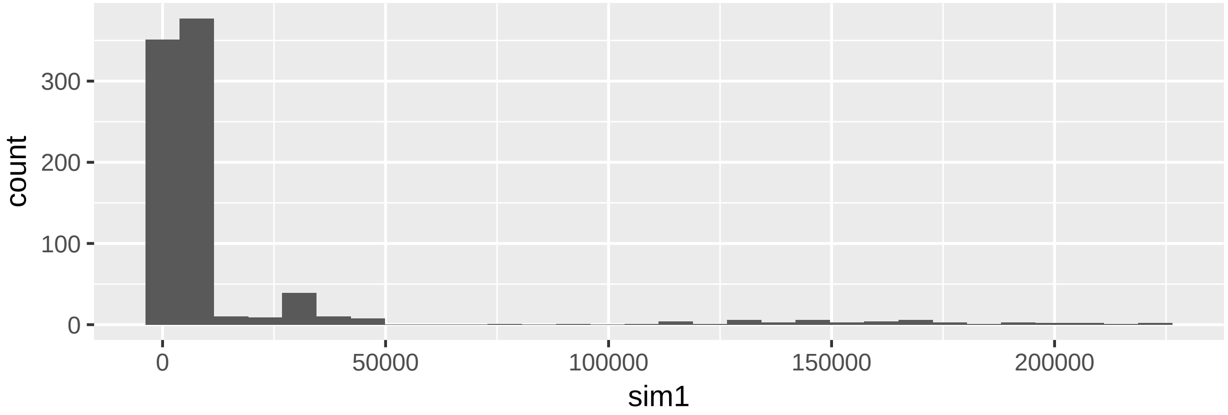
```
summary(OUPokemon$Raw)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##      1.0     78.8   1029.0  14890.0   8341.0 452500.0
```

```
p1  <- ggplot(OUPokemon, aes(x = Raw)) + geom_histogram()
p2 <- ggplot(data.frame(sim1), aes(x = sim1)) + geom_histogram()
grid.arrange(p1, p2)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```





TO DO - Attack, Defense, usage - 3d plot???

Try - linear models or quasi poisson/Negative binomial regression/loess and GAM

```
constant.glm = glm(Raw ~ 1, family = poisson, data = UberPokemon)
display(constant.glm)
```

```
## glm(formula = Raw ~ 1, family = poisson, data = UberPokemon)
##             coef.est coef.se
## (Intercept) 8.19     0.00
## ---
##   n = 905, k = 1
##   residual deviance = 10974244.3, null deviance = 10974244.3 (difference = 0.0)
```

```
offset.glm = glm(Raw ~ 1, family = poisson, offset = log(BST), data = UberPokemon)
display(offset.glm)
```

```
## glm(formula = Raw ~ 1, family = poisson, data = UberPokemon,
##     offset = log(BST))
##             coef.est coef.se
## (Intercept) 2.10     0.00
## ---
##   n = 905, k = 1
##   residual deviance = 9022446.7, null deviance = 9022446.7 (difference = 0.0)
```

```
bst.glm = glm(Raw ~ BST, family = poisson, data = UberPokemon)
display(bst.glm)
```

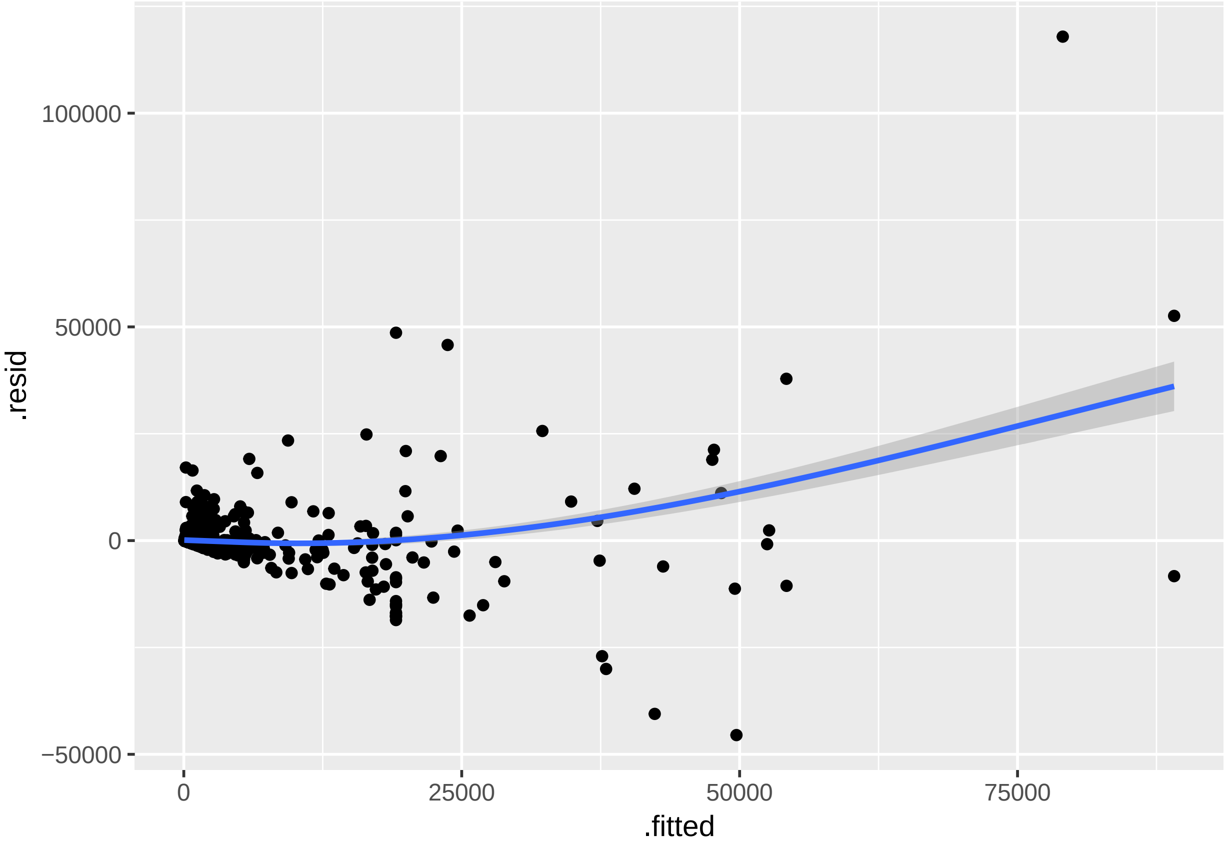
```
## glm(formula = Raw ~ BST, family = poisson, data = UberPokemon)
##             coef.est coef.se
## (Intercept) 2.08     0.00
## BST         0.01     0.00
## ---
##   n = 905, k = 2
##   residual deviance = 4966502.9, null deviance = 10974244.3 (difference = 6007741.4)
```

```
bst.UC.glm = glm(Raw ~ Attack + Defense + Special + Sp.Attack + Sp.Defense + HP +factor(Usage.Category)
deviance(bst.UC.glm)
```

```
## [1] 2453383
```

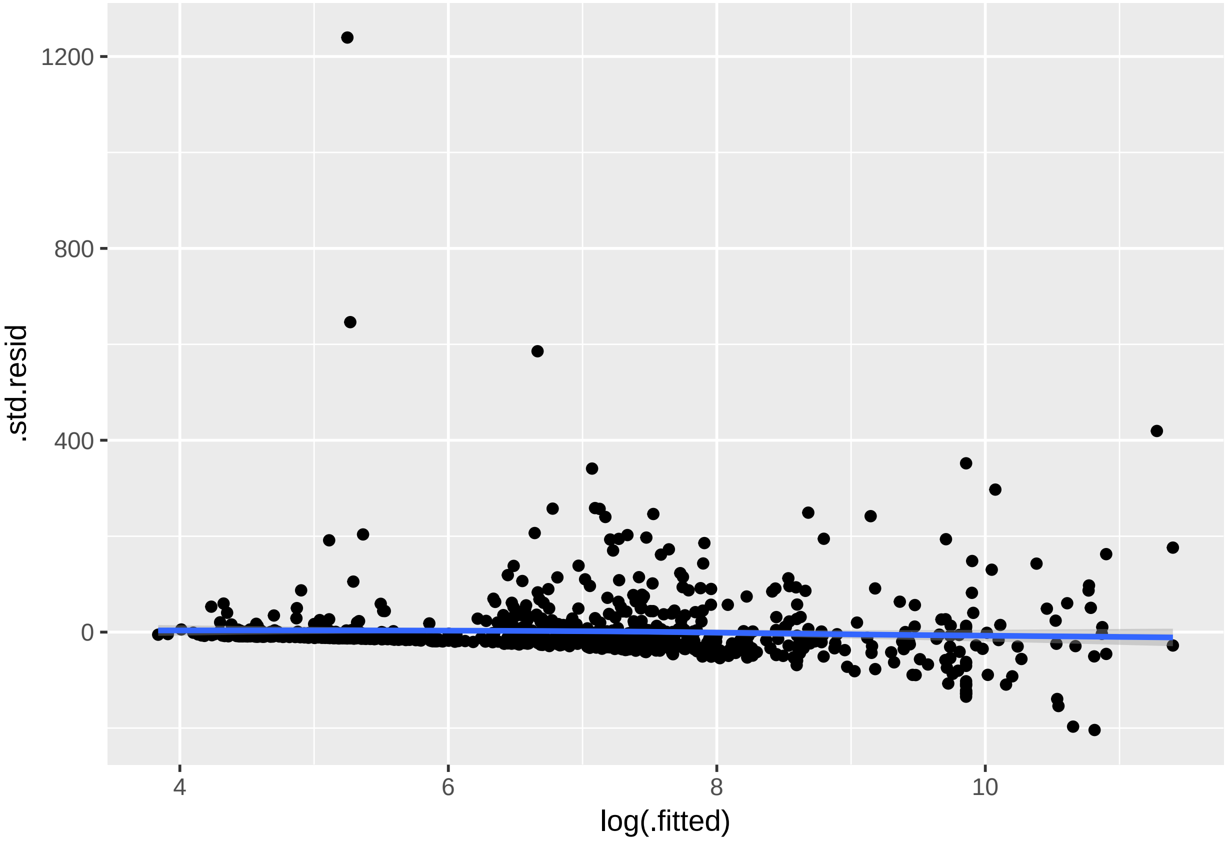
```
fitted = fitted.values(bst.UC.glm)
resid = residuals(bst.UC.glm, type = "response")
df = data.frame(UberPokemon, .fitted = fitted, .resid = resid)
ggplot(df, aes(x = .fitted, y = .resid)) + geom_point() + geom_smooth(span = 1, method.args = list(degre
```

```
## `geom_smooth()` using method = 'loess'
```



```
std.resid = resid/sqrt(fitted)
df$.std.resid = std.resid
ggplot(df, aes(x = log(.fitted), y = .std.resid)) + geom_point() + geom_smooth(span = 1, method.args = 
```

```
## `geom_smooth()` using method = 'loess'
```



```
overdispersion = sum(std.resid^2)/df.residual(bst.UC.glm)
overdispersion
```

```
## [1] 5709.921
```

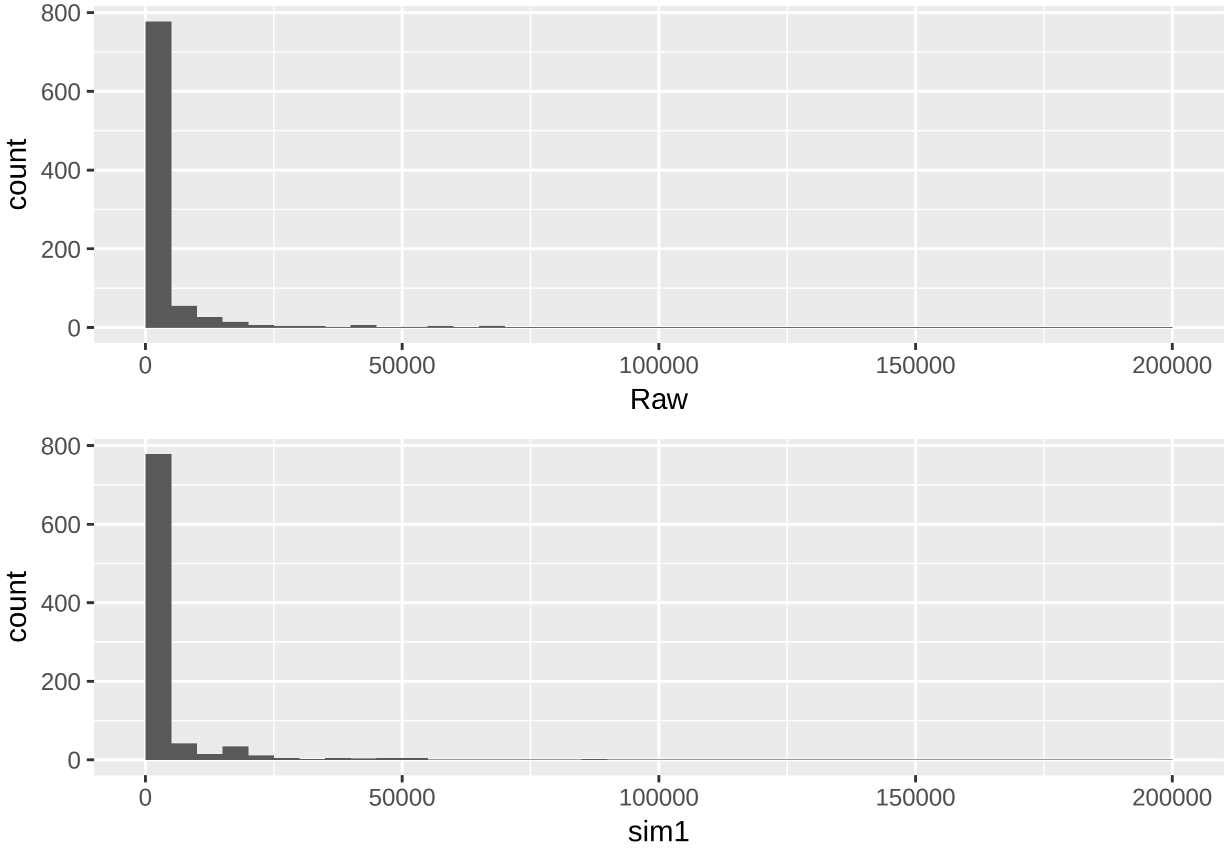
```
sim1 = rpois(nrow(UberPokemon), lambda = fitted.values(bst.UC.glm))
summary(sim1)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      43     205     904    3587    2075   89440
```

```
summary(UberPokemon$Raw)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       1      55     386    3587    1978  197000
```

```r
p1  <- ggplot(UberPokemon, aes(x = Raw)) + geom_histogram(breaks = seq(0, 200000, 5000))
p2 <- ggplot(data.frame(sim1), aes(x = sim1)) + geom_histogram(breaks = seq(0, 200000, 5000))
grid.arrange(p1, p2)
```



Gen - 1

```r
OverUsedG1 = read.csv('data/gen1ou-0.txt',skip = 2)
OverUsedG1 <- merge(Pokemon.Attributes,OverUsedG1,by="Pokemon")
constant.glm = glm(Raw ~ 1, family = poisson, data = OverUsedG1)
display(constant.glm)
```

```
## glm(formula = Raw ~ 1, family = poisson, data = OverUsedG1)
##             coef.est coef.se
## (Intercept) 6.72     0.00
## ---
##   n = 146, k = 1
##   residual deviance = 384588.9, null deviance = 384588.9 (difference = 0.0)
```

```r
offset.glm = glm(Raw ~ 1, family = poisson, offset = log(BST), data = OverUsedG1)
display(offset.glm)
```

```
## glm(formula = Raw ~ 1, family = poisson, data = OverUsedG1, offset = log(BST))
##             coef.est coef.se
## (Intercept) 0.71     0.00
## ---
##   n = 146, k = 1
##   residual deviance = 332178.0, null deviance = 332178.0 (difference = 0.0)
```

```
bst.glm = glm(Raw ~ BST, family = poisson, data = OverUsedG1)
display(bst.glm)
```

```
## glm(formula = Raw ~ BST, family = poisson, data = OverUsedG1)
##             coef.est coef.se
## (Intercept) -0.57     0.02  
## BST          0.02     0.00  
## ---
##   n = 146, k = 2
##   residual deviance = 225251.7, null deviance = 384588.9 (difference = 159337.2)
```

```
#bst.UC.glm = glm(Raw ~ Attack + Defense + Special + Sp.Attack + Sp.Defense + HP +factor(Usage.Category
#deviance(bst.UC.glm)
```

```
#fitted = fitted.values(bst.UC.glm)
#resid = residuals(bst.UC.glm, type = "response")
#df = data.frame(OverUsedG1, .fitted = fitted, .resid = resid)
#ggplot(df, aes(x = .fitted, y = .resid)) + geom_point() + geom_smooth(span = 1, method.args = list(deg
```

```
#std.resid = resid/sqrt(fitted)
#df$.std.resid = std.resid
#ggplot(df, aes(x = log(.fitted), y = .std.resid)) + geom_point() + geom_smooth(span = 1, method.args =
```

```
#overdispersion = sum(std.resid^2)/df.residual(bst.UC.glm)
#overdispersion
#sim1 = rpois(nrow(OverUsedG1), lambda = fitted.values(bst.UC.glm))
#summary(sim1)
#summary(OverUsedG1$Raw)

#p1  <- ggplot(OverUsedG1, aes(x = Raw)) + geom_histogram(breaks = seq(0, 50000, 1000))
#p2 <- ggplot(data.frame(sim1), aes(x = sim1)) + geom_histogram(breaks = seq(0, 50000, 1000))
#grid.arrange(p1, p2)
```