对copyInfo信息进行校验

```go
if err := utils.Verify(copyInfo, utils.OldAuthorityVerify); err != nil {
        response.FailWithMessage(err.Error(), c)
        return
}
```

validtor.go

```go
func Verify(st interface{}, roleMap Rules) (err error) {
    compareMap := map[string]bool{
        "lt": true,
        "le": true,
        "eq": true,
        "ne": true,
        "ge": true,
        "gt": true,
    }

    typ := reflect.TypeOf(st)
    val := reflect.ValueOf(st) // 获取reflect.Type类型

    kd := val.Kind() // 获取到st对应的类别
    if kd != reflect.Struct {
        return errors.New("expect struct")
    }
    num := val.NumField()
    // 遍历结构体的所有字段
    for i := 0; i < num; i++ {
        tagVal := typ.Field(i)
        val := val.Field(i)
        if len(roleMap[tagVal.Name]) > 0 {
            for _, v := range roleMap[tagVal.Name] {
                switch {
                case v == "notEmpty":
                    if isBlank(val) {
                        return errors.New(tagVal.Name + "值不能为空")
                    }
                case strings.Split(v, "=")[0] == "regexp":
                    if !regexpMatch(strings.Split(v, "=")[1], val.String()) {
                        return errors.New(tagVal.Name + "格式校验不通过")
                    }
                case compareMap[strings.Split(v, "=")[0]]:
                    if !compareVerify(val, v) {
                        return errors.New(tagVal.Name + "长度或值不在合法范围," + v)
                    }
                }
            }
        }
    }
    return nil
}
```

utils.go

```go
package utils

var (
    IdVerify               = Rules{"ID": {NotEmpty()}}
    ApiVerify              = Rules{"Path": {NotEmpty()}, "Description": {NotEmpty()}, "ApiGroup": {NotEmpty()}, "Method": {NotEmpty()}}
    MenuVerify             = Rules{"Path": {NotEmpty()}, "ParentId": {NotEmpty()}, "Name": {NotEmpty()}, "Component": {NotEmpty()}, "Sort": {Ge("0")}}
    MenuMetaVerify         = Rules{"Title": {NotEmpty()}}
    LoginVerify            = Rules{"CaptchaId": {NotEmpty()}, "Captcha": {NotEmpty()}, "Username": {NotEmpty()}, "Password": {NotEmpty()}}
    RegisterVerify         = Rules{"Username": {NotEmpty()}, "NickName": {NotEmpty()}, "Password": {NotEmpty()}, "AuthorityId": {NotEmpty()}}
    PageInfoVerify         = Rules{"Page": {NotEmpty()}, "PageSize": {NotEmpty()}}
    CustomerVerify         = Rules{"CustomerName": {NotEmpty()}, "CustomerPhoneData": {NotEmpty()}}
    AutoCodeVerify         = Rules{"Abbreviation": {NotEmpty()}, "StructName": {NotEmpty()}, "PackageName": {NotEmpty()}, "Fields": {NotEmpty()}}
    AuthorityVerify        = Rules{"AuthorityId": {NotEmpty()}, "AuthorityName": {NotEmpty()}, "ParentId": {NotEmpty()}}
    AuthorityIdVerify      = Rules{"AuthorityId": {NotEmpty()}}
    OldAuthorityVerify     = Rules{"OldAuthorityId": {NotEmpty()}}
    ChangePasswordVerify   = Rules{"Username": {NotEmpty()}, "Password": {NotEmpty()}, "NewPassword": {NotEmpty()}}
    SetUserAuthorityVerify = Rules{"AuthorityId": {NotEmpty()}}
)

```