Lecture #3: Modern Regression Analysis

Regression Analysis (Linear, Non-linear, and Logistic)

Correlation vs Causation

Model Analysis and Evaluation

Model Explainability

Michael Fu

https://michaelfu1998-create.github.io/

Dr. Kla Tantithamthavorn

Senior Lecturer in Software Engineering

http://chakkrit.com @klainfo

Schedule *http://chakkrit.com/teaching/quantitative-research-methods*

Date	Location	Topics
~~11 November 2024~~	~~G.13 Woodside Building (20 Exhibition Walk)~~	~~Design Science Paradigm~~
~~15 November 2024~~	~~G.13 Woodside Building (20 Exhibition Walk)~~	~~Statistical Analysis~~
25 November 2024	G.13 Woodside Building (20 Exhibition Walk)	Modern Regression Analysis
29 November 2024	G.13 Woodside Building (20 Exhibition Walk)	ML Quality Assurance

Regression Analysis

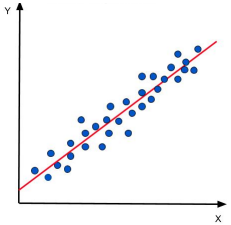
Regression analysis is a statistical process for estimating the relationships between a dependent variables (or response variables) and one or more independent variables (or explanatory variables).

Use cases

- Predict the value of the response variable based on the given independent variables
- Find a relationship between a dependent variable (Y) and independent variables (X)

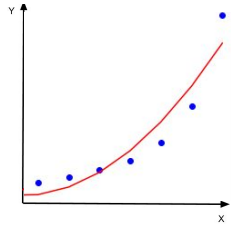
There are three types of regression analysis (or regression model):

Linear Regression



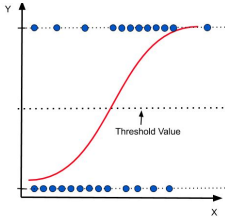
Predicting a numerical variable, assuming a linear relationship between X and Y

Non-Linear Regression



Predicting a numerical variable, assuming a non-linear relationship between X and Y

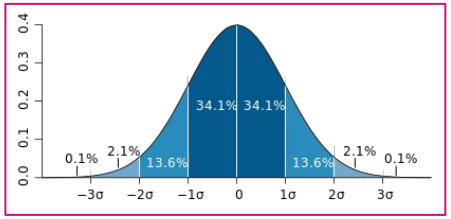
Logistic Regression



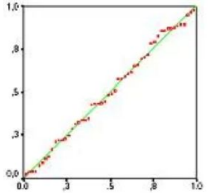
Predicting a binary variable

Assumptions in Regression Analysis

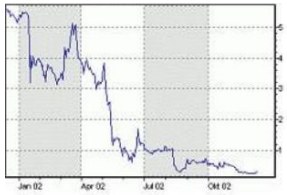
1. All variables are normally distributed (multivariate normality). Suggest to check the goodness of fit, e.g., Kolmogoroc-Smirnov test
2. Independent variables are not highly correlated with each other (no multicollinearity).
   - check correlations among the independent variables and remove one from the highly-correlated pair
3. Residuals (observations) are independent from each other (no autocorrelation), e.g., stock prices have autocorrelation
4. Residuals are equal across the regression line (homoscedasticity)
   - check scatter plot or use Goldfeld-Quandt test



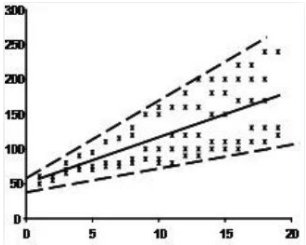
A variable is normally distributed



Two variables are highly-correlated



Stock price at time T is not independent from Stock price at time T+1

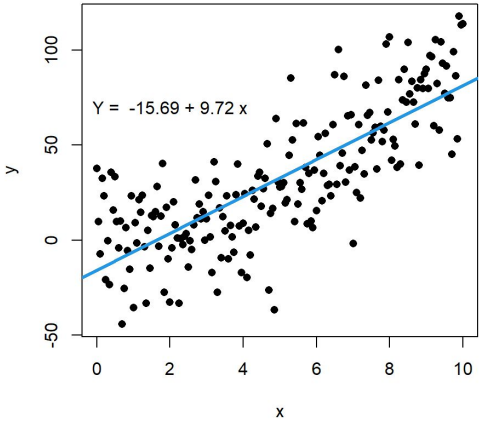


The data has homoscedasticity as the residuals are scattered equally on the regression line

Theory: Linear Regression

A mathematical model to predict a numerical outcome using a linear equation (Y value is interval/Ratio).

A formula for a linear regression is $Y = a + b_1X_1 + b_2X_2 + \ldots + B_nX_n + \varepsilon$
where ***X*** is the explanatory variable,
Y is the dependent variable, $\mathbf{b_1}$-$\mathbf{b_n}$ is regression coefficient (i.e., the the slope of the line), ***a*** is the intercept (the value of ***y*** when ***x*** = 0), and **ε** is the error term.



Linear Regression finds the line of best fit line through the dataset by searching for the regression coefficient (B_1) that minimizes the total error (e) of the model (i.e., cost function = Least Sum of Square of Errors, optimized using Gradient Descent)

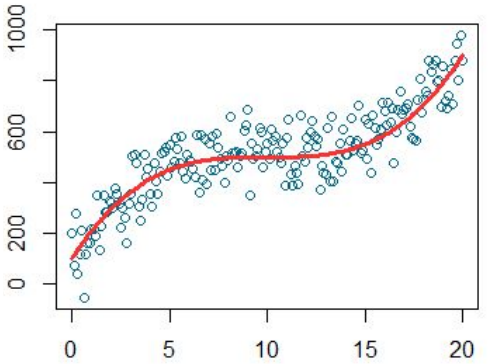
Additional Assumptions

1. The relationship between independent and dependent variable is linear (linear relationship) (check the scatter plot)
2. The dependent variable is interval/ratio

Theory: Non-Linear Regression

A mathematical model to predict a numerical outcome using a non-linear (curved) equation (Y value is interval/Ratio)

A formula for a linear regression is $Y = f(x,\beta) + \epsilon$ where **x** is a vector of P predictors, **β** is a vector of k parameters, **f**() is the known regression function, and **ϵ** is the error term



The regression function **f(x,β)** can have different form.
For example, if we want to predict a house price, we can use the polynomial function $f(x) = w_1x + w_2x^2 + w_3x^3 + b$, where **x** is the size of the houses.

Non-Linear Regression finds the **curved** line of best fit line through the dataset by searching for the regression coefficient (B1) that minimizes the total error (e) of the model.

Additional Assumptions

1. The dependent variable is interval/ratio

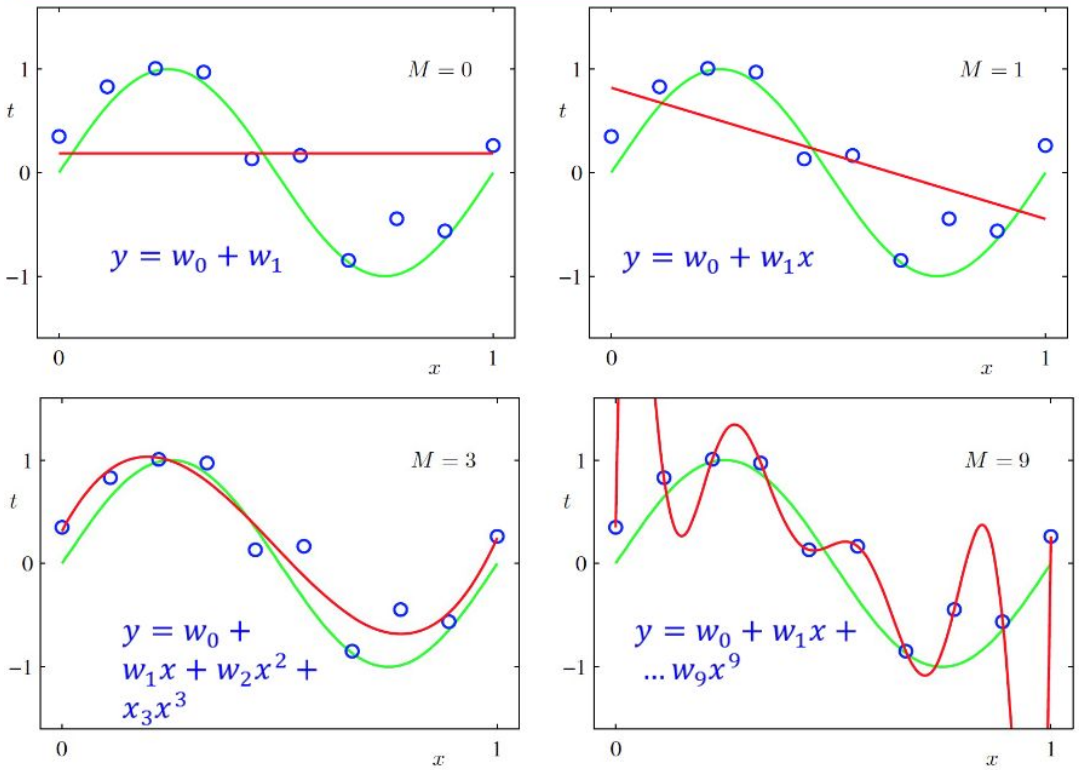
$$y(x, \mathbf{w}) = w_0 + w_1x + w_2x^2 + \cdots + w_M x^M = \sum_{j=1}^{M} w_j x^j$$

M

The higher the order, the more complex the model.

Underfitting and Overfitting

M=0, 1: poor fits to the data, thus poor representation of $\sin(2\pi x)$



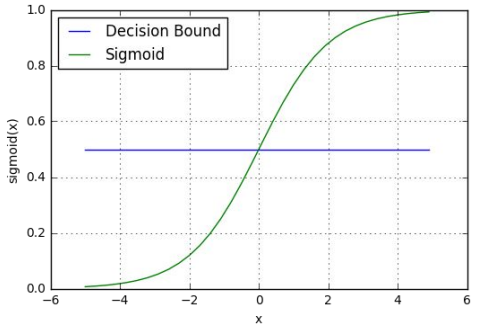
M=3: well fit to $\sin(2\pi x)$

M=9: an excellent fit to the training

Theory: Logistic Regression

A mathematical model for predicting the probability of a binary outcome (Y value is Binomial).

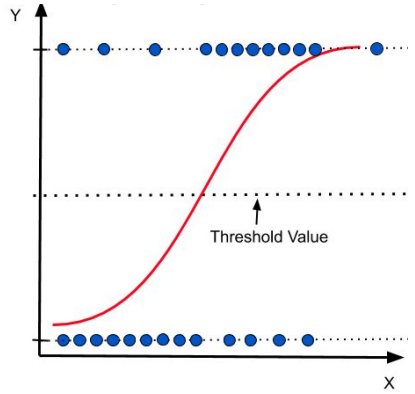
Instead of fitting a straight line, the logistic regression uses the logistic function (logit) to transform the output of a linear equation to the value between 0 and 1.



The logistic function is: $\text{logistic}(\eta) = \frac{1}{1 + exp(-\eta)}$

Hence, the equation for logistic regression is:

$$P(y^{(i)} = 1) = \frac{1}{1 + exp(-(\beta_0 + \beta_1 x_1^{(i)} + \ldots + \beta_p x_p^{(i)}))}$$



Assumption

1. Dependent variable is binary

Example – Predicting the House Prices Using Non-Linear

House price dataset https://www.kaggle.com/competitions/house-prices-advanced-regression-techniques/data

	SalePrice	GrLivArea	LotArea
1	208500	1710	8450
2	181500	1262	9600
3	223500	1786	11250
4	140000	1717	9550
5	250000	2198	14260
6	143000	1362	14115
7	307000	1694	10084
8	200000	2090	10382
9	129900	1774	6120
10	118000	1077	7420

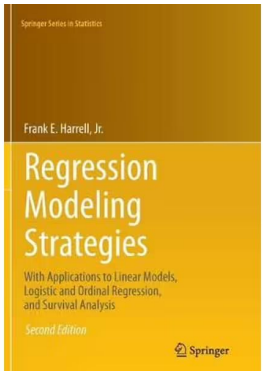
Q1: What factors has the highest association with the house price? or What are the most important factors that impact the house price?

Q2: What is the relationship of the most important factors (positive vs negative)?

```
View(dplyr::select(trainDf, SalePrice, GrLivArea, YearBuilt))
```

Steps to Build the Non-Linear Regression Model

1. Perform correlation analysis to remove the highly correlated variables
2. Perform redundancy analysis to remove the redundant variables
3. Allocate degree of freedom for each independent variables
4. Building the Non-Linear Regression model
5. Measure the prediction performance
6. Examine the relationship between each dependent variables and the response variables



Harrell, Frank E. *Regression modeling strategies: with applications to linear models, logistic regression, and survival analysis*. Vol. 608. New York: springer, 2001.

Steps to Build the Non-Linear Regression Model

Let's get started with loading the dataset and removing the outliers

```
library(rms)
df <- read.csv("train.csv", header=TRUE) # 1460 rows
# remove outlier, 1325 rows left
df = dplyr::select_if(df, is.numeric) # used only numerical column
df = df[, !(colnames(df) %in% c("Id"))] # remove ID
df[is.na(df)] <- 0 # fill in NA value with 0
ind_vars = colnames(df[ , !(names(df) %in% 'SalePrice')]) # list the independent
variables

# RMS pckage that we will used requires a data distribution when building a model
dd <- datadist(df)
options(datadist='dd')
```

How many features do we have?

```
colnames(df)
> colnames(df)
 [1] "Id"            "MSSubClass"    "MSZoning"      "LotFrontage"
 [5] "LotArea"       "Street"        "Alley"         "LotShape"
 [9] "LandContour"   "Utilities"     "LotConfig"     "LandSlope"
[13] "Neighborhood"  "Condition1"    "Condition2"    "BldgType"
[17] "HouseStyle"    "OverallQual"   "OverallCond"   "YearBuilt"
[21] "YearRemodAdd"  "RoofStyle"     "RoofMatl"      "Exterior1st"
[25] "Exterior2nd"   "MasVnrType"    "MasVnrArea"    "ExterQual"
[29] "ExterCond"     "Foundation"    "BsmtQual"      "BsmtCond"
[33] "BsmtExposure"  "BsmtFinType1"  "BsmtFinSF1"    "BsmtFinType2"
[37] "BsmtFinSF2"    "BsmtUnfSF"     "TotalBsmtSF"   "Heating"
[41] "HeatingQC"     "CentralAir"    "Electrical"    "X1stFlrSF"
[45] "X2ndFlrSF"     "LowQualFinSF"  "GrLivArea"     "BsmtFullBath"
[49] "BsmtHalfBath"  "FullBath"      "HalfBath"      "BedroomAbvGr"
[53] "KitchenAbvGr"  "KitchenQual"   "TotRmsAbvGrd"  "Functional"
[57] "Fireplaces"    "FireplaceQu"   "GarageType"    "GarageYrBlt"
[61] "GarageFinish"  "GarageCars"    "GarageArea"    "GarageQual"
[65] "GarageCond"    "PavedDrive"    "WoodDeckSF"    "OpenPorchSF"
[69] "EnclosedPorch" "X3SsnPorch"    "ScreenPorch"   "PoolArea"
[73] "PoolQC"        "Fence"         "MiscFeature"   "MiscVal"
[77] "MoSold"        "YrSold"        "SaleType"      "SaleCondition"
[81] "SalePrice"
```

1) Perform correlation analysis

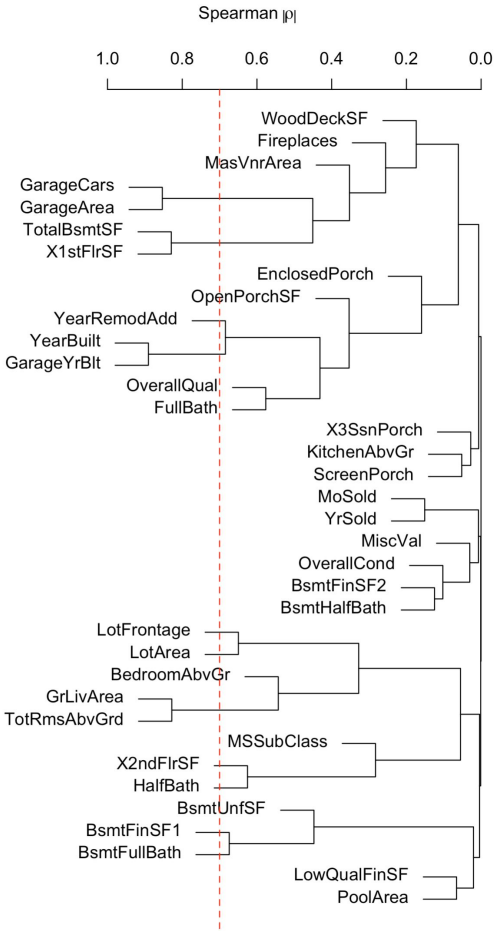
to remove highly-correlated variables

Calculate Spearman's correlation between independent vars

```
vc <- varclus(~ ., data=df[,ind_vars], trans="abs")

# Plot hierarchical clusters
# and the spearman's correlation threshold of 0.7
plot(vc)
threshold <- 0.7
abline(h=1-threshold, col = "red", lty = 2)
```

Remove the highly-correlated variables

```
#Remove the highly correlated variable from the hierarchical
clusters
reject_vars <- c('GarageCars', 'X1stFlrSF', 'YearRemodAdd',
'GarageYrBlt', 'LotFrontage', 'TotRmsAbvGrd', 'X2ndFlrSF',
'BsmtFinSF1')

ind_vars <- ind_vars[!(ind_vars %in% reject_vars)]
```

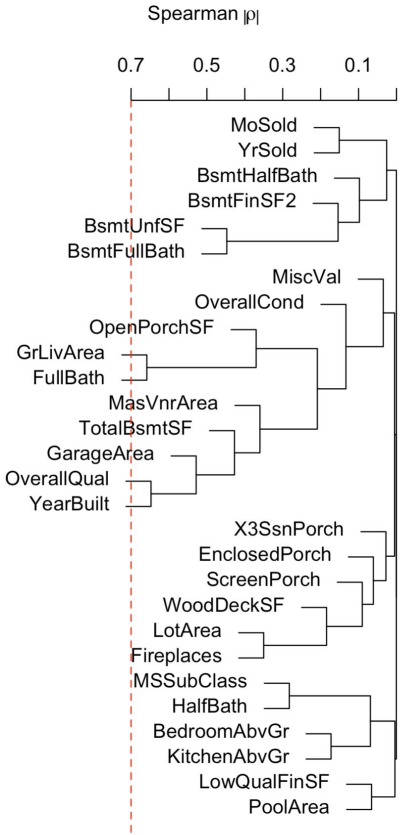


1) Perform correlation analysis (cont.)

to remove highly-correlated variables

Re-calculate the Spearman's correlation

```
vc <- varclus(~ ., data=df[,ind_vars], trans="abs")

#Re-plot hierarchical clusters and the spearman's correlation
threshold of 0.7
plot(vc)
threshold <- 0.7
abline(h=1-threshold, col = "red", lty = 2)
```



Remove the highly-correlated variables again

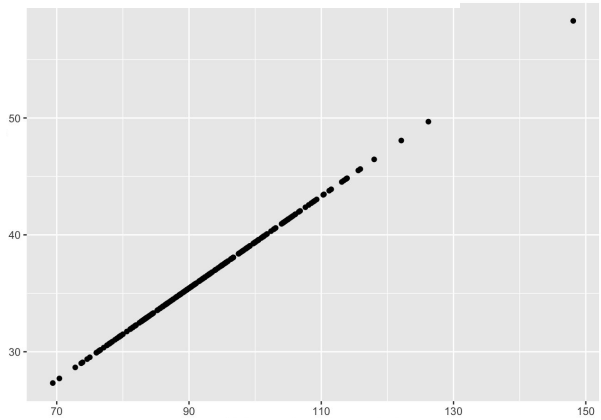
```
reject_vars <- c('FullBath','OverallQual')
ind_vars <- ind_vars[!(ind_vars %in% reject_vars)]
```

2) Perform redundancy analysis

to remove the redundant variables

A variable is considered redundant when they contain exactly the same information and predict each other in a perfect line.

We determine the redundant variable(s) based on R^2 value ($R^2 > 0.9$)



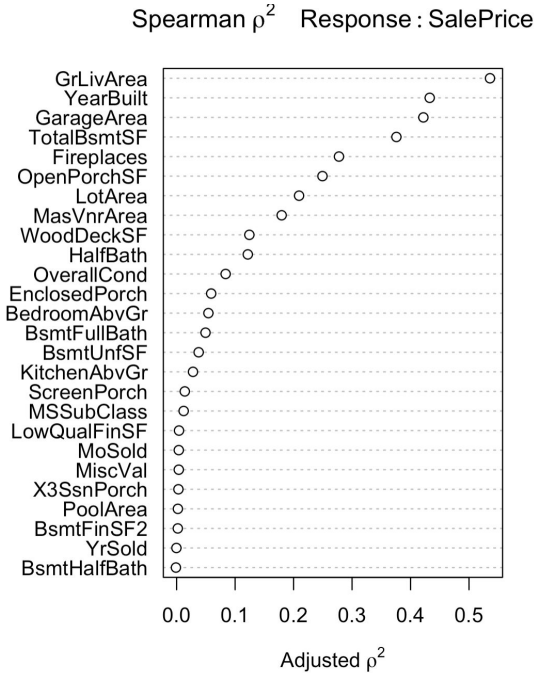
Lists the explanatory variables where models are with R^2 value > 0.9

```
red <- redun(~., data=df[,ind_vars], nk=0)
reject_vars <- red$Out
ind_vars <- ind_vars[!(ind_vars %in% reject_vars)]
```

3) Allocate degree of freedom for each independent variables

Plot Spearman rho^2 (explanatory power) between each independent variable and the response

```
sp <- spearman2(formula(paste("SalePrice" ,"~",
paste0(ind_vars, collapse=" + "))), data=df, p=2)
plot(sp)
```



Assign degrees of freedom for independent variables

(5 for rho^2 > 0.3 and 3 for 0.15 < rho^2 <= 0.3)

```
formula = paste0("SalePrice", " ~ ")
for (i in 1:length(ind_vars)) {
  var = rownames(sp)[i]
  rho2 = sp[i]
  if (rho2 > 0.3) {
    formula = paste0(formula, "rcs(", var, ",", 5, ") + ")
  } else if (rho2 > 0.15) {
    formula = paste0(formula, "rcs(", var, ",", 3, ") + ")
  } else {
    formula = paste0(formula, var, " + ")
  }
}
formula = substr(formula,1,nchar(formula)-3)
print(formula)
```

SalePrice ~ MSSTotalBsmtSF,5) + LowQualFinSF + rcs(GrLivArea,5) + BsmtFullBath + BsmtHalfBath + HalfBath + BedroomAbvGr + KitchenAbvGr + rcs(Fireplaces,3) + rcsubClass + rcs(LotArea,3) + OverallCond + rcs(YearBuilt,5) + rcs(MasVnrArea,3) + BsmtFinSF2 + BsmtUnfSF + rcs((GarageArea,5) + WoodDeckSF + rcs(OpenPorchSF,3) + EnclosedPorch + X3SsnPorch + ScreenPorch + PoolArea + MiscVal + MoSold + YrSold

4) Building the Non-Linear Regression model

```
fit <- Glm(SalePrice ~ MSSubClass + rcs(LotArea, 3) + OverallCond + rcs(YearBuilt, 5) +
rcs(MasVnrArea, 3) + BsmtFinSF2 + BsmtUnfSF + rcs(TotalBsmtSF, 5) + LowQualFinSF +
rcs(GrLivArea, 5) + BsmtFullBath + BsmtHalfBath + HalfBath + BedroomAbvGr + KitchenAbvGr
+ Fireplaces + rcs(GarageArea, 5) + WoodDeckSF + rcs(OpenPorchSF, 3) + EnclosedPorch +
X3SsnPorch + ScreenPorch + PoolArea + MiscVal + MoSold + YrSold, data = df)

# We do not assign degree of freedom to Fireplace since there are only 4 unique value,
which are too few for 3 degree of freedoms, i.e., 3 knots.
```

5) Measure the prediction performance

The lower value of MAE, MSE, and RMSE implies higher accuracy of a regression mode

```
df$predicted = predict(fit, df)
```

R-Squared (or the coefficient of determination is the proportion of the variance in the dependent variable which is explained by the linear regression model. (ranged from 0-1)

$$R^2 = 1 - \frac{\sum(y_i - \hat{y})^2}{\sum(y_i - \bar{y})^2}$$

```
library(Metrics)
mae(df$SalePrice, df$predicted)
# 20597.66
```

Where,

$\hat{y}$ – *predicted value of y*

$\bar{y}$ – *mean value of y*

Mean Squared Error is the average of the absolute difference between the actual and predicted values in the dataset. It measures the average of the residuals in the dataset.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y})^2$$

```
mse(df$SalePrice, df$predicted)
# 1144668440
```

Where,

$\hat{y}$ – *predicted value of y*

$\bar{y}$ – *mean value of y*

Root Mean Squared Error is the average of the absolute difference between the actual and predicted values in the dataset. It measures the average of the residuals in the dataset.

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y})^2}$$

```
rmse(df$SalePrice, df$predicted)
# 33832.95
```

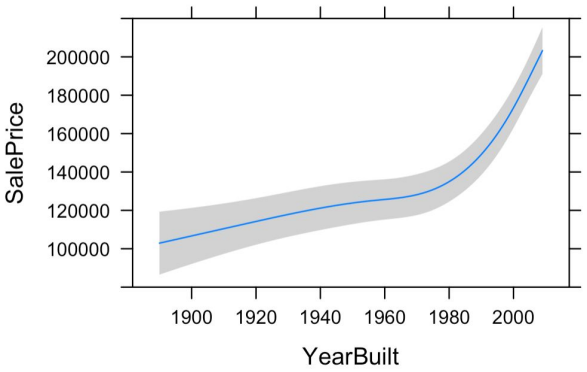
Where,

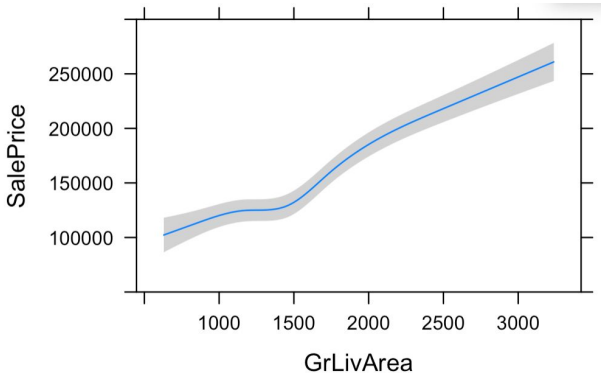
$\hat{y}$ – *predicted value of y*

$\bar{y}$ – *mean value of y*

6) Examine the relationship between each dependent variable and the response variables (i.e., explanatory power)

```
explantory_power = data.frame(anova(fit,test='Chisq'))
explantory_power = explantory_power[order(explantory_power$Chi.Square, decreasing =
TRUE),]
print(explantory_power)

predict <- Predict(fit,YearBuilt,fun=function(x)x)
plot(predict, ylab='Odds')
```





	Chi.Square	d.f.	P
TOTAL	6394.7655999	41	0.000000e+00
YearBuilt	425.4487027	4	0.000000e+00
GrLivArea	413.2798898	4	0.000000e+00
TOTAL.NONLINEAR	374.9057004	15	0.000000e+00
TotalBsmtSF	144.5225727	4	0.000000e+00
X.Nonlinear.1	137.8684100	3	0.000000e+00
OverallCond	116.7582167	1	0.000000e+00
X.Nonlinear.3	91.6064479	3	0.000000e+00
X.Nonlinear.4	63.2302118	3	1.199041e-13
GarageArea	48.5886964	4	7.114135e-10
MasVnrArea	47.2500904	2	5.492540e-11
Fireplaces	33.3740548	1	7.603155e-09

There are positive relationship between SalePrice and YearBuilt/GrLivArea

7) Examine the partial effect of each variable

```
partialEffectDf = data.frame(summary(fit))
partialEffectDf = partialEffectDf[order(partialEffectDf$Effect, decreasing =
TRUE),]
```

Higher Effect = influencing model

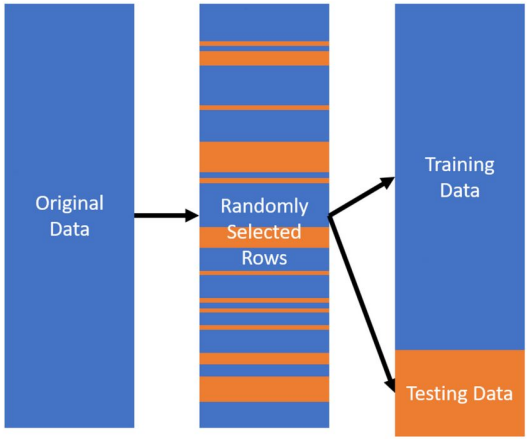
	Low	High	Diff.	Effect	S.E.	Lower.0.95	Upper.0.95	Type
YearBuilt	1954.00	2000.00	46.00	48798.4543	3486.0221	41960.1396	55636.769	1
GrLivArea	1129.50	1776.75	647.25	39576.8968	3514.3859	32682.9425	46470.851	1
ScreenPorch	0.00	480.00	480.00	34508.2023	8233.8527	18356.3609	50660.044	1
TotalBsmtSF	795.75	1298.25	502.50	32418.1828	3336.5426	25873.0929	38963.273	1
X3SsnPorch	0.00	508.00	508.00	19105.1773	15853.5048	-11993.6659	50204.020	1
EnclosedPorch	0.00	552.00	552.00	13455.2053	9229.9587	-4650.6357	31561.046	1
Fireplaces	0.00	1.00	1.00	10216.1105	1768.4024	6747.1445	13685.077	1
OverallCond	5.00	6.00	1.00	9986.7209	924.2281	8173.7196	11799.722	1
LotArea	7553.50	11601.50	4048.00	6998.0612	1384.1769	4282.8067	9713.316	1
OpenPorchSF	0.00	68.00	68.00	6535.7779	2378.3630	1870.2897	11201.266	1
HalfBath	0.00	2.00	2.00	6415.5506	4769.8538	-2941.1776	15772.279	1
GarageArea	334.50	576.00	241.50	5936.9609	2734.8429	572.1880	11301.734	1
BsmtFullBath	0.00	1.00	1.00	4452.4286	2607.8470	-663.2241	9568.081	1
BsmtHalfBath	0.00	2.00	2.00	4155.0054	8107.2833	-11748.5525	20058.563	1
WoodDeckSF	0.00	168.00	168.00	3255.5307	1346.2278	614.7186	5896.343	1
MoSold	5.00	8.00	3.00	470.2271	1022.3021	-1535.1599	2475.614	1
YrSold	2007.00	2009.00	2.00	-612.7320	1388.0770	-3335.6371	2110.173	1
MSSubClass	20.00	70.00	50.00	-717.5503	1582.7130	-3822.2609	2387.160	1
MasVnrArea	0.00	164.25	164.25	-2007.4984	2065.4917	-6059.2461	2044.249	1
BedroomAbvGr	2.00	3.00	1.00	-5179.3292	1576.2662	-8271.3935	-2087.265	1
PoolArea	0.00	738.00	738.00	-6689.3169	17778.4778	-41564.2610	28185.627	1

Validating the Performance (Single Split)

Splitting the training and testing dataset, once

To report the prediction performance, we predict the y value (i.e., SalePrice) of unseen dataset. Hence, we separate the dataset into training and testing. The proportion can be varied, e.g., 75:25, 80:20.

```
# splitting 75% train, 25% test
sample <- sample.int(n = nrow(df), size = floor(.75*nrow(df)), replace =
train <- df[sample, ]
test  <- df[-sample, ]
```



However, splitting the data only once **may not well reflect the robustness of the model** on different datasets

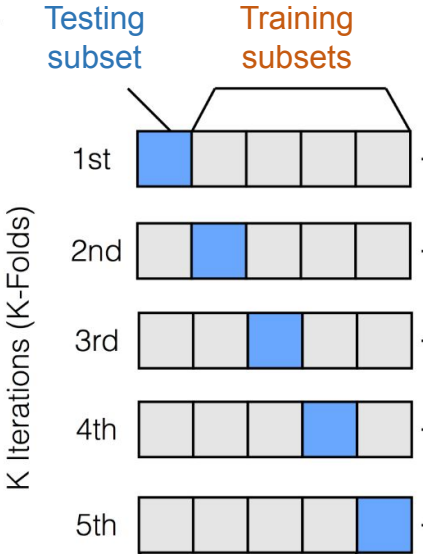
Validating the Performance (K-Fold Cross-Validation)

Splitting the training and testing dataset, K-time

The k-fold cross-validation method evaluates the model performance on different subset of the training data and then calculate the average prediction error rate.

In K-fold cross-validation,

1. Randomly split the data set into k-subsets (or k-fold)
2. Reserve one testing subset and train the model on all other training subsets
3. Test the model on the testing subset and record the prediction error
4. Repeat this process until each of the k subsets has served as the testing subset.
5. Compute the average of the k recorded errors, called cross-validation error.




```
# Define training control
library(caret)
# Define training control
train.control <- trainControl(method = "cv", number = 10)
# Train the model
model <- train(SalePrice ~., data = df, method = "glm", trControl = train.control)
# Summarize the results
print(model)
```

However, splitting the data in K-folds may cause the testing set to be too small.

Validating the Performance (Bootstrap)

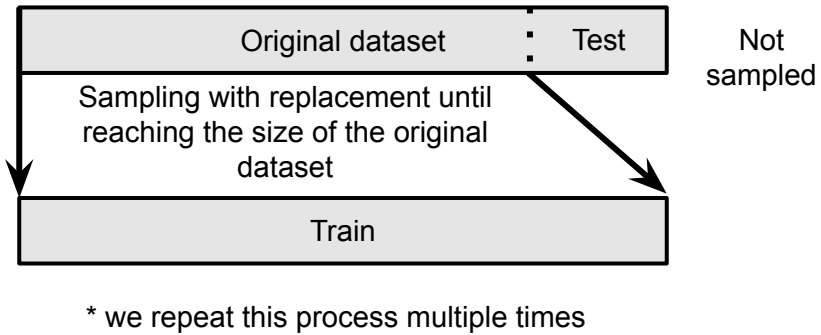
Applying the sampling with replacement technique to augment the training set

- A bootstrap sample is a random sample conducted with replacement
  1. Randomly select an observation from the original data
  2. Write it down
  3. Put it back (i.e. Any observation can be selected more than once)

Repeat these steps 1-3 N times; N is the number of observations in the original sample

Final Result: One "bootstrap sample" with N observations

```
# load the data again as we need the Id
df <- read.csv("train.csv", header=TRUE)

# sampling with replacement
train <- df[sample(nrow(df), nrow(df),
replace=TRUE),]
# nrow(train) = 1460, same size as original df

# The rows that are not sampled are used for testing
test <- subset(df, !(df$Id %in% train$Id))
# nrow(test) = 527
```



* we repeat this process multiple times

More About Correlation

Correlation is an **average across subsamples** and **may not reflect the relationship between any two individuals**.

Imagine we find a **strong correlation** between height and shoe size in a group of men and women combined. This means that **on average**, taller people tend to have bigger shoe sizes.

But, if we compare a tall woman and a short man, the relationship might not follow this trend. For example:

- A woman who is 5'10" might have a smaller shoe size than a man who is 5'6".
- This happens because gender differences in shoe size can override the height-shoe size trend.

Similarly, even if two people have the **same height**, their shoe sizes can differ due to factors like gender or individual variation (e.g., foot shape).

More About Correlation

The uncertainty around measuring correlation means that small correlations are often meaningless noise, and moderate correlations are uncertain, while high correlations could be solid evidence.

Small correlations are often too weak to draw conclusions because they can arise from random variation.

Moderate correlations hint at a possible relationship but need further investigation to confirm.

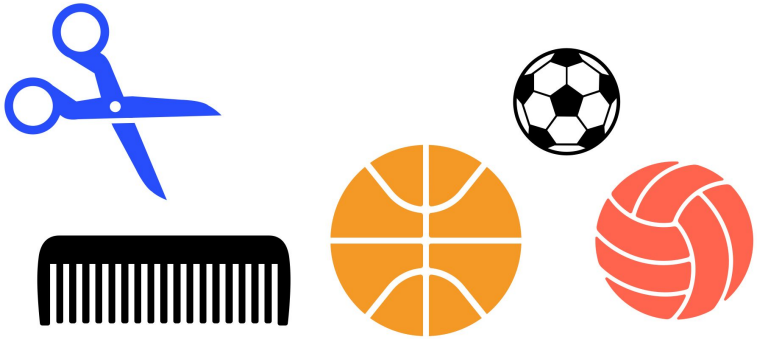
High correlations provide stronger evidence of a meaningful relationship, though they still require proper context and causation checks.

Correlation does not imply causation

Correlation

A connection between two or more things that don't cause each other

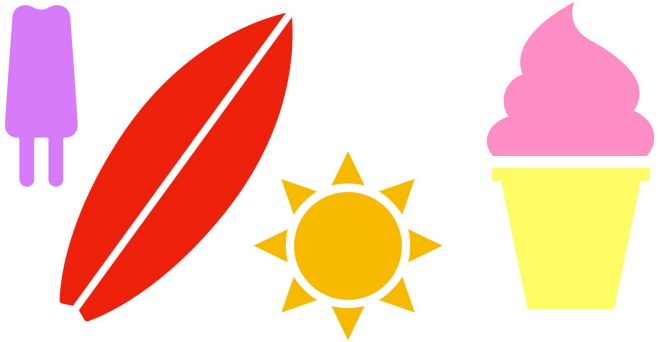
For example, the amount of sports you play might have a connection or be similar to how often you get your hair cut



Causation

When one thing is the cause of something else

For example, summer causes people to eat more frozen treats!

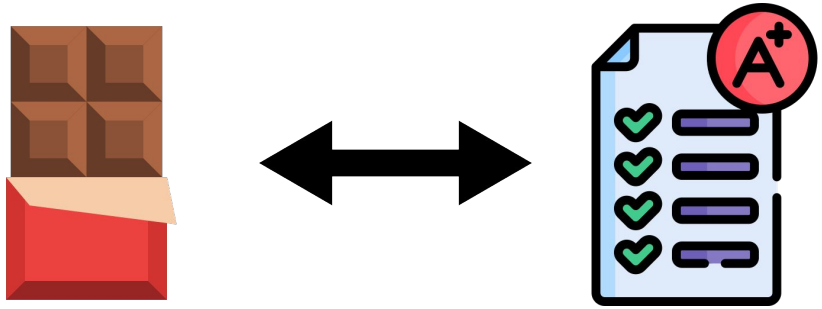


Correlation vs. Causation: Beware of Selection Bias

Correlation doesn't always mean causation, especially when researchers **select variables based on a desired pattern.**

A researcher wants to show a strong connection between eating chocolate and better math scores.

They look at data from 100 schools and **only report the 5 schools** where chocolate consumption happens to be high and math scores are good.



Correlation vs. Causation: Beware of Selection Bias

Correlation doesn't always mean causation, especially when researchers **select variables based on a desired pattern.**

A researcher wants to show a strong connection between eating chocolate and better math scores.

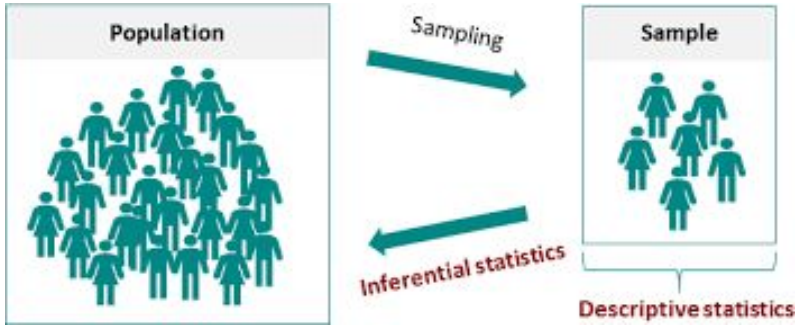
They look at data from 100 schools and **only report the 5 schools** where chocolate consumption happens to be high and math scores are good.

- The correlation they report (chocolate ↔ math) might just be **random coincidence**, not a real relationship.
- This happens because they **ignored the other schools where chocolate had no effect**.
- If you actively search for patterns and only report the ones that fit, the results may look convincing but might not tell the true story.

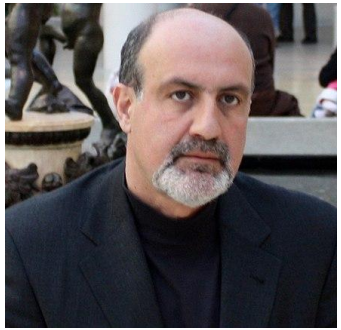
What's Even Worse…

Correlation Does Not Even Imply Correlation

That is, correlation in *the data you happen to have* (even if it happens to be "statistically significant") does not necessarily imply correlation **in the population of interest.**

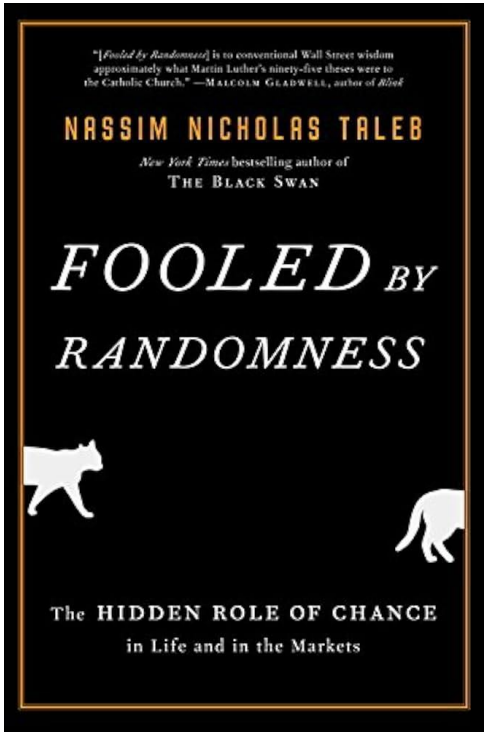


Learn More About Correlation & Causation



Nassim Nicholas Taleb is a Lebanese-American essayist, mathematical statistician, former option trader, risk analyst, and aphorist.

His work concerns problems of **randomness, probability, complexity, and uncertainty.**



Lecture #3: Model Explainability

Explainable AI

Michael Fu

https://michaelfu1998-create.github.io/

Dr. Kla Tantithamthavorn

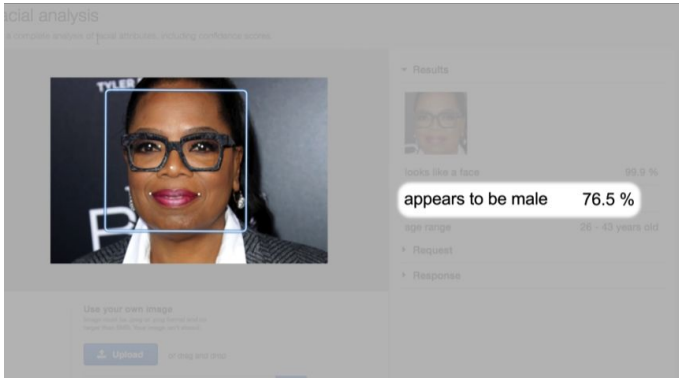
Senior Lecturer in Software Engineering

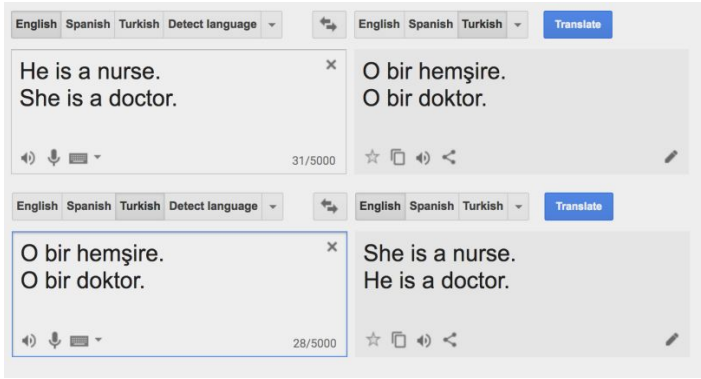
http://chakkrit.com @klainfo

AI Bias and Unfair Judgement | Should human be judged by AI?

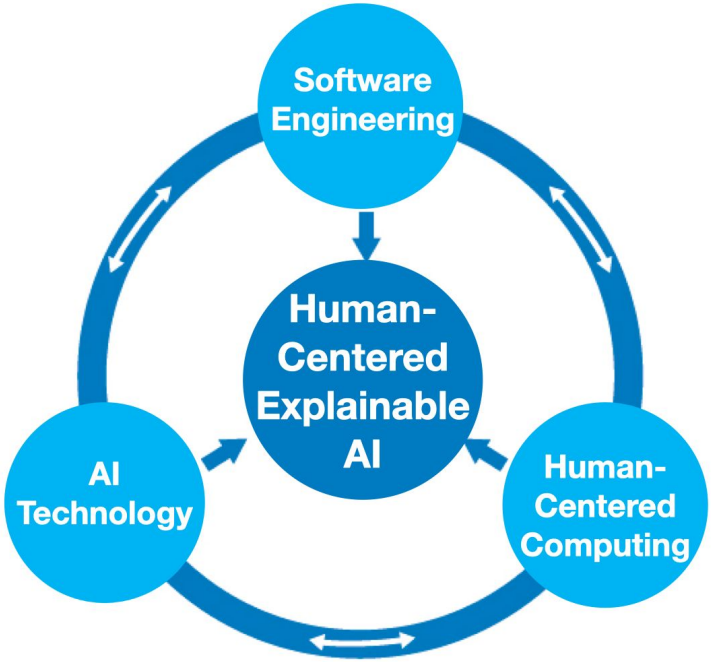
Teachers had been terminated based on AI's suggestion
- Coded Bias

Artificial Intelligence has a racial and gender bias problem





Human-Centered Explainable AI



Explainable AI is a set of techniques and tools that make a model's decision understandable by humans.

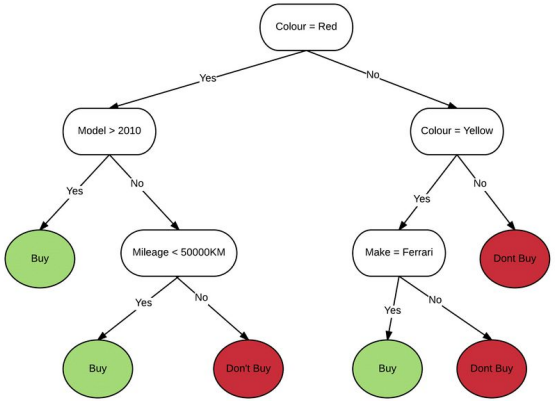
However, most of the time, AI experts just build their explainable AI techniques for themself without considering the end users' needs.

Human-Centered Explainable AI is a multidisciplinary approach that aims to design Explainable AI techniques that most suit end-users' needs.

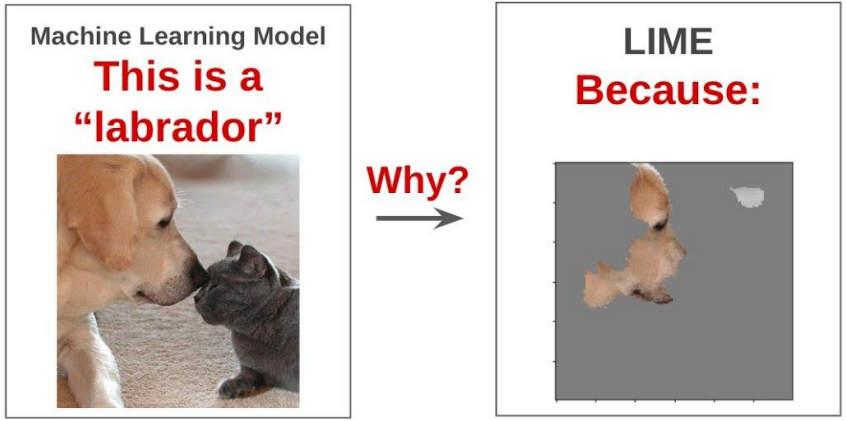
Explainable AI | What is the explainability for AI?

The explainable AI (XAI) aims to create a suite of AI/ML techniques that:
- produce more explainable model, while maintaining a high level of prediction accuracy; and
- Enable human users to understand and build an appropriate trust to the predictions

Interpretable AI
Using models that are inherently interpretable, e.g., small decision trees or linear models (globally explainable)



Explainable AI
Applying a method that models the output of a more complex model after training the model. (locally explainable)



Black-Box AI Creates Confusion and Doubt

Black box AI

Why I am getting this recommendation?

Business owner:
Can I trust our AI decisions?

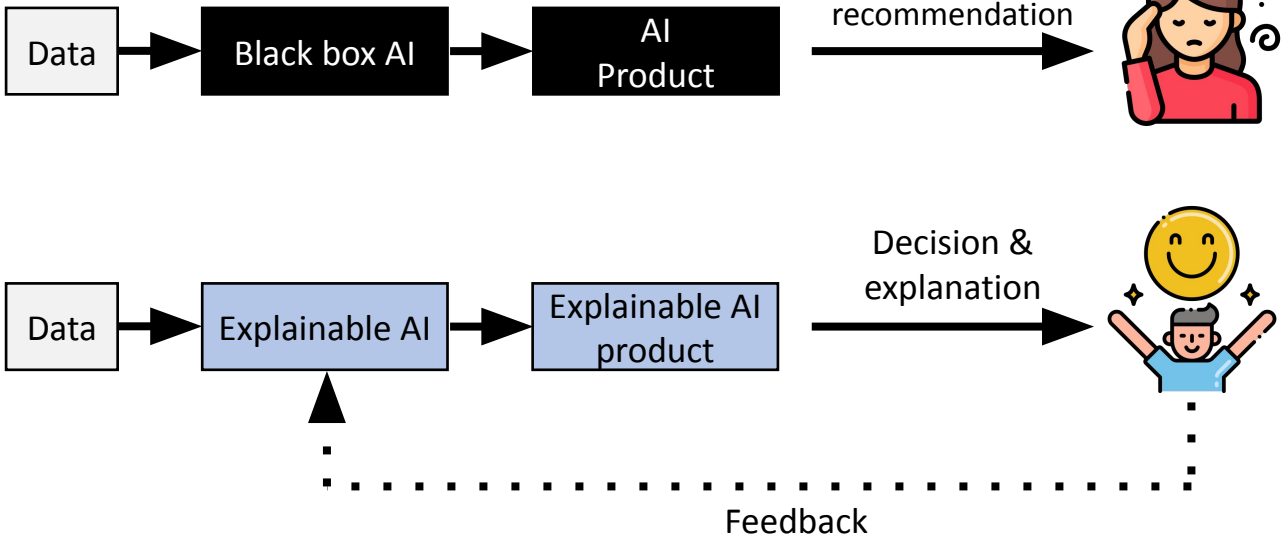
Customer support:
How do I answer this customer complaint?

IT & Operations team:
How do I monitor and debug this model?

Data Scientist:
Is this the best model that can be built?

Audit & Regulators:
Are these AI system decisions fair?

Explainable AI



Confusion with AI Black box
Why did you do that?
Why did you not do that?
Why do you succeed or fail?
How do I correct an error?

Clear & transparent predictions
I understand why
I understand why not
I know why you succeed or fail
I understand, so I trust you

Why Do We Need Explainability in AI

Explainability helps human build trust with AI's applicability for real world usages

Reasonable
- The ability to understand the reasoning behind each individual prediction.

Traceable
- The ability to trace the prediction process from logic of math algorithm to nature of data.

Understandable
- The ability to understand the model upon which the AI decision making is based.

Patient

Age	Gender	Condition
25	Female	Cold
32	Male	N/A
31	Male	Cough

Model (Decision trees)

Model understanding

```
If gender = female,
   if ID_num > 200
      then sick

If gender = male,
   if cold = true and cough = true,
      then sick
```

This model is using irrelevant features when predicting on female subpopulation.

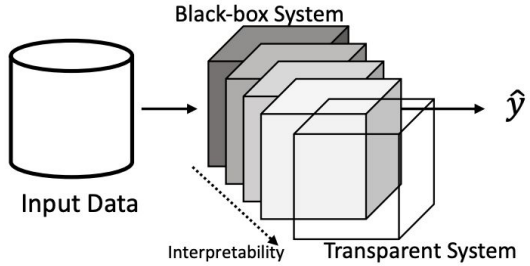
I should not trust its predictions for that group.

Two Types of Explainable AI Systems

Explainability helps human build trust with AI's applicability for real world usages

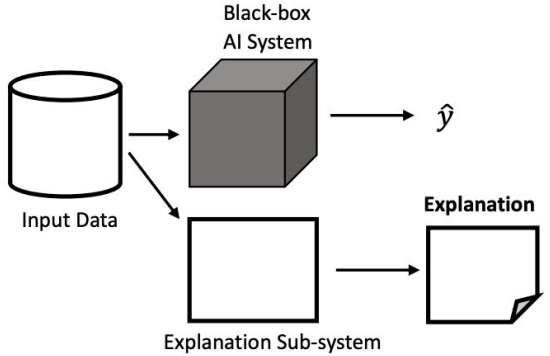
Interpretable AI (Transparent by design)
Using white-box algorithm, e.g., Linear regression, Logistic regression, Decision Tree, Naive Bayes, KNNs

- Not accurate, but interpretable
- Interpretability at global (model) level



Explainable AI (Post-hoc explanation)
Using an explainer algorithm to explain the prediction of a black-box model, e.g., neural networks

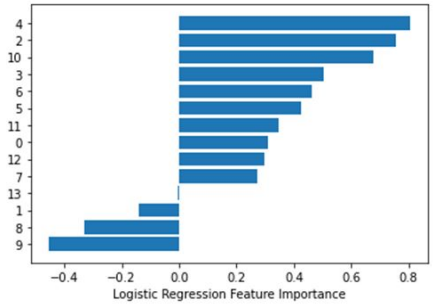
- Generally more accurate, but cannot be interpret directly
- interpretability at prediction level



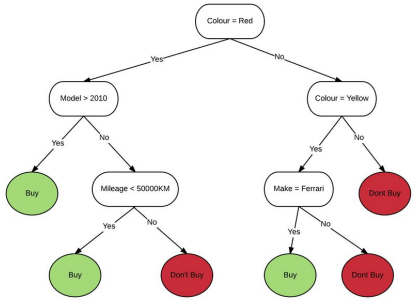
Explainable AI (Global Explanations)

Multiple methods are being used to explain the model

- Feature Importance
- Rule Based



Feature importance of Logistic Regression

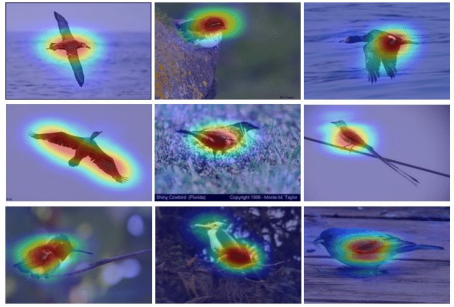


Rules of decision trees

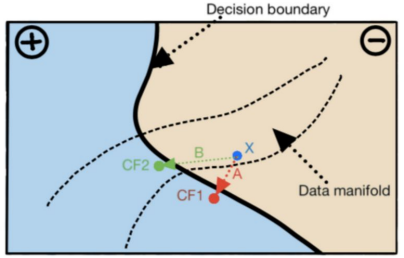
Explainable AI (Local Explanations)

Multiple methods are being used to explain the model

- Saliency Map for Images
- Counterfactuals – (what if we do this, would it change the prediction?)



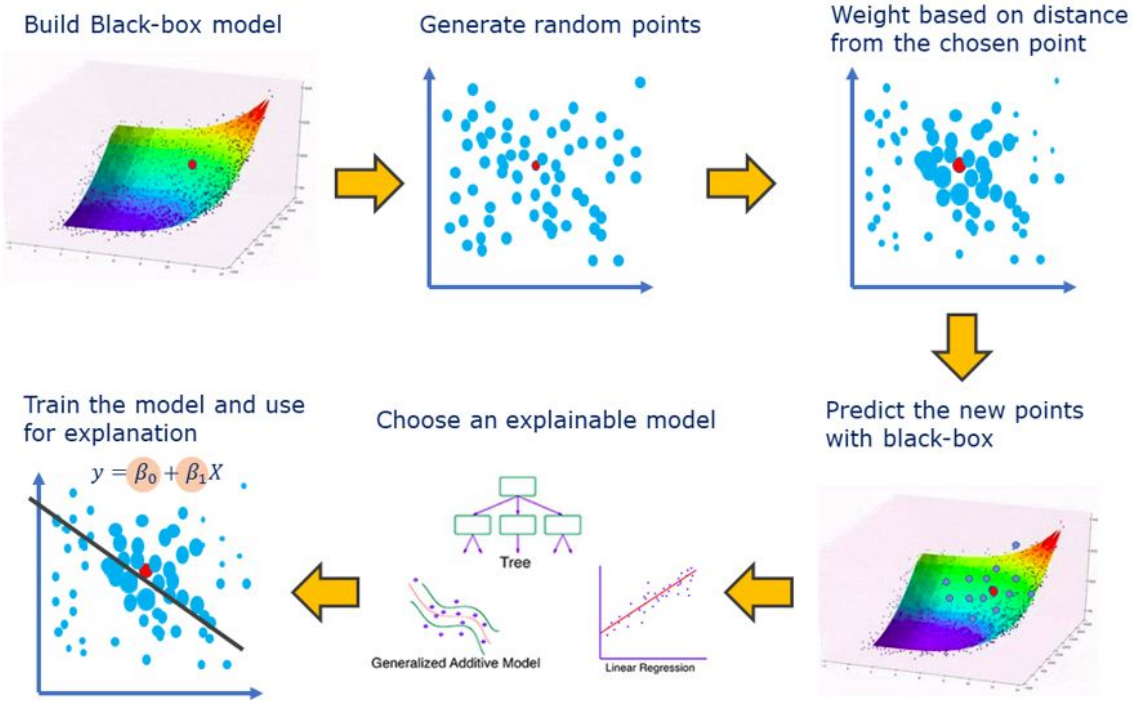
Saliency map for image classification



Counterfactual Explainability

Explainable AI (Local Explanations)

Approximates any black box machine learning model with a local interpretable model to explain each individual prediction.



Explaining XGBoost Model's Prediction

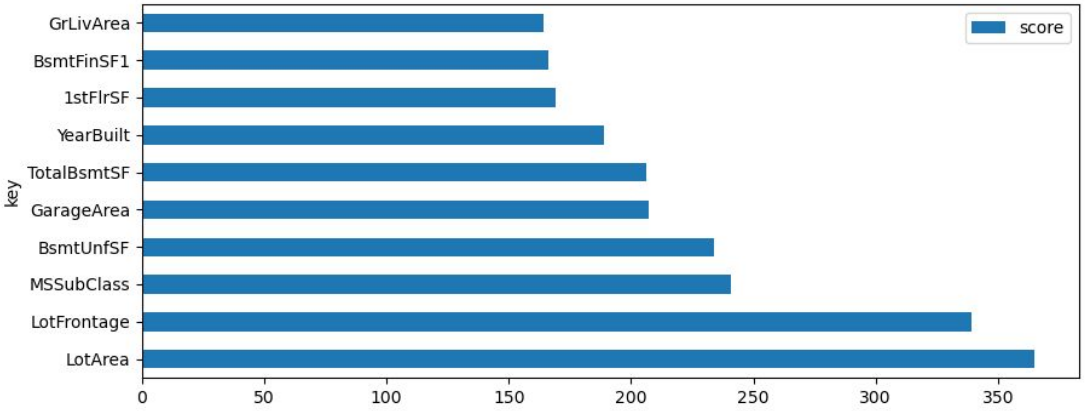
Feature importance and local explanation (LIME)

```
# fit XGBoost regressor model

xgb = XGBRegressor(verbosity=0)
xgb = xgb.fit(X_train, y_train)
y_pred = xgb.predict(X_test)


# extract feature importance

feature_importance = xgb.get_booster().get_score(importance_type='gain')
keys = list(feature_importance.keys())
values = list(feature_importance.values())
data = pandas.DataFrame(data=values, index=keys, columns=["score"]).sort_values(by="score", ascending=False)


# use LIME to explain the price prediction of the house #0

import lime.lime_tabular
explainer = lime.lime_tabular.LimeTabularExplainer(X_test.to_numpy(), feature_names=X_test.columns.tolist(),
class_names=['SalePrice'], verbose=True, mode='regression')
print(y_pred[0])
exp = explainer.explain_instance(X_test.iloc[0].to_numpy(), xgb.predict, num_features=10)
exp.show_in_notebook()
```

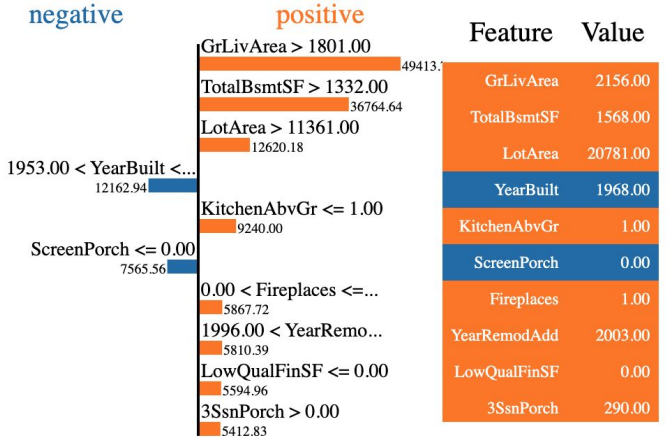
Explaining XGBoost Model's Prediction

Feature importance and local explanation (LIME)

Feature Importance in the model



Local explanation of house #1



Other Explainers

Individual explainers

- **LIME**
  - https://github.com/marcotcr/lime
- **Local surrogates**
  - https://github.com/axa-rev-research/localitv-interpretable-surrogate
- **Anchor**
  - https://github.com/marcotcr/anchor
- **PyCEbox**
  - https://github.com/AustinRochford/PyCEbox

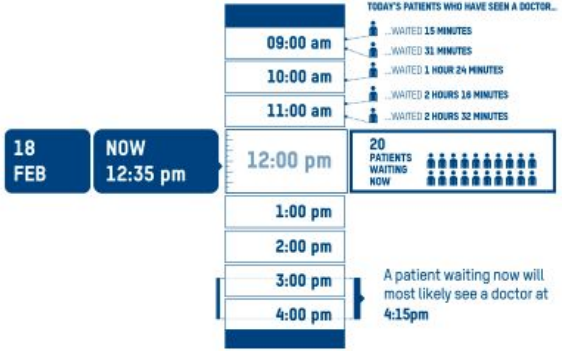
Packages

- **Microsoft's InterpretML**
  - https://github.com/interpretml/interpret
- **IBM's AI Explainability 360**
  - https://github.com/IBM/AIX360
- **Oracle's Skater**
  - https://github.com/oracle/Skater
- **ELI5**
  - https://github.com/TeamHG-Memex/eli5
- **Yellowbrick**
  - https://github.com/DistrictDataLabs/vellowbrick

ED Patient Wait-Time Prediction

Problem: Most patients hope to be seen by a healthcare provider immediately on arrival but usually have to wait for treatment. Thus, deciding where to seek care for acute medical problems is still difficult.

Solution: Emergency department (ED) patient wait time prediction can assist the physical, logistic and psychological needs of patients, which could facilitate optimal patient load-balancing across acute care facilities, reducing the harms of long waits.



Explainable AI for ED Wait-Time Prediction

Explainable hospital wait times for emergency department patients

Ever had to wait in an emergency department with no knowledge of how long you'd have to wait, or why you had to wait a certain amount of time?

This tool aims to solve this problem by generating emergency department wait time estimations, with corresponding explanations for the different factors that contributed to the final prediction.

Predict Wait Time | Learn More

timed.

Your predicted wait time is
13 minutes

Your wait time is estimated to be **13 minutes**, as compared to the average (mean) wait time of **50 minutes.** Other factors that are not from your input (such as the current rolling average waiting time, number of patients in waiting room and in queue) have also been used to produce this prediction.

The **triage category** factor of **Emergency decreased** your waiting time by around **38 minutes** and was the largest contributing factor in the prediction.

Triage Category

- Category 1 Resuscitation
- Category 2 Emergency
- Category 3 Urgent
- Category 4 Semi-Urgent
- Category 5 Non-Urgent

Triage categories 1 and 2 can **decrease** wait time by **~20 to 70 minutes**. Categories 3, 4 and 5 may **increase** your wait time by **~10 minutes.**


Chief Investigator
Dr Chakkrit (Kla) Tantithamthavorn


Andy Zhan

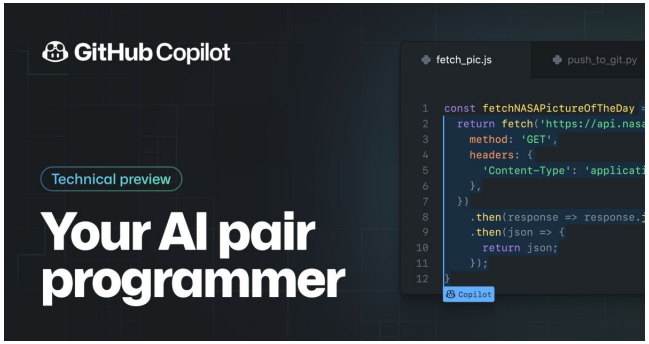

Sara Tran

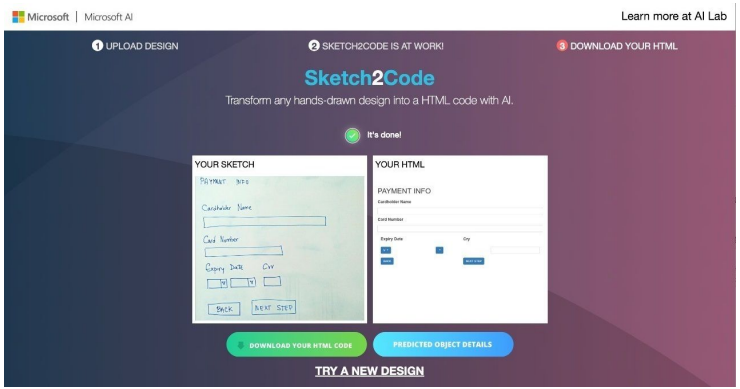

Thev Wickramasinghe


Xi Zhang

Developed by a team of Year-4 B.SE degree students. (http://118.138.245.231/)

AI is eating software

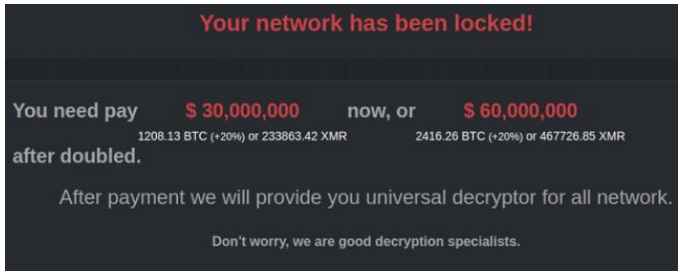




Sorry, developers: Microsoft's new tool fixes the bugs in software code written by AI

Microsoft's Jigsaw can automate the process of checking and editing code outputted by a machine.

AIBugHunter - Using XAI to Explain Vulnerabilities in Programs



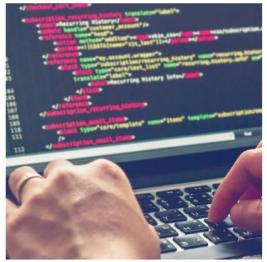
Problem: Cyber criminals continue to find new methods of attack, e.g., stealing your credit card information and password, asking to pay money, damaging business's reputation and customer's trusts.

Solution: AIBugHunter is an AI approach that is learned from millions of software projects to understand the patterns of vulnerabilities so it can automatically detect, locate, explain, and suggest corrections in real-time.

NEWS

Aussie Researchers Reckon They Found The Key To Predicting Software Vulnerabilities

Covered by GIZMODO Australia.



NEWS

Preventing Cyber Attacks Through Code Analysis.

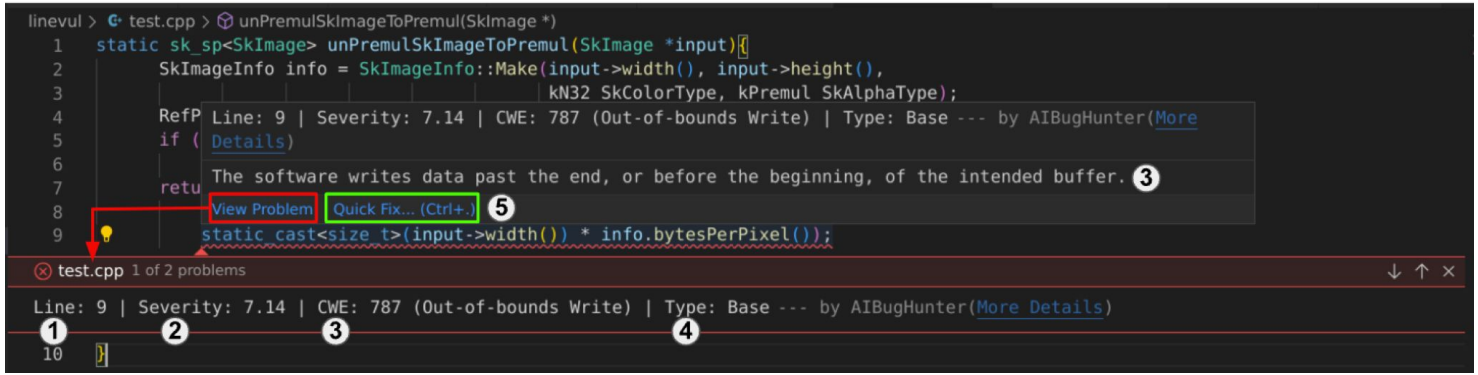
Covered by Australian Computer Society - INFORMATIONAGE.

NEWS

Monash University: Unglitching The System: Advancement In Predicting Software Vulnerabilities.

Covered by India Education Diary.

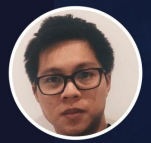
AIBugHunter - Using XAI to Explain Vulnerabilities in Programs



Explainable AI for Software Engineering

A Hands-on Guide on How To Make Software Analytics More Practical, Explainable, and Actionable (https://xai4se.github.io)

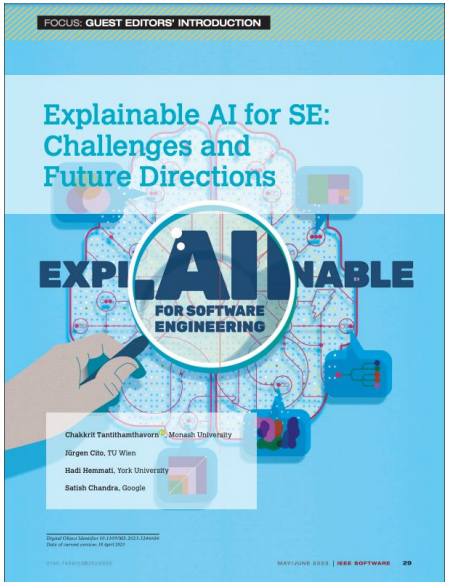
Dr. "Kla" Chakkrit Tantithamthavorn

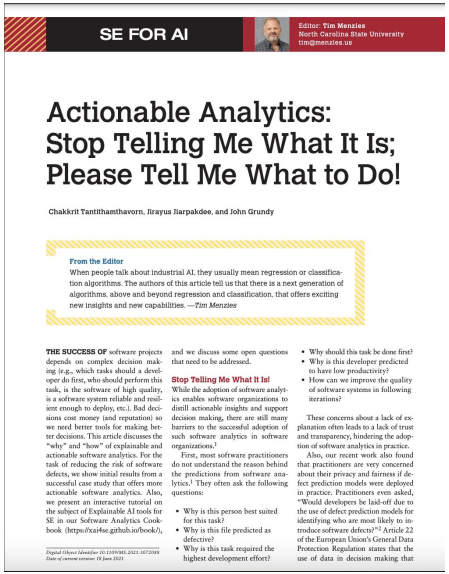


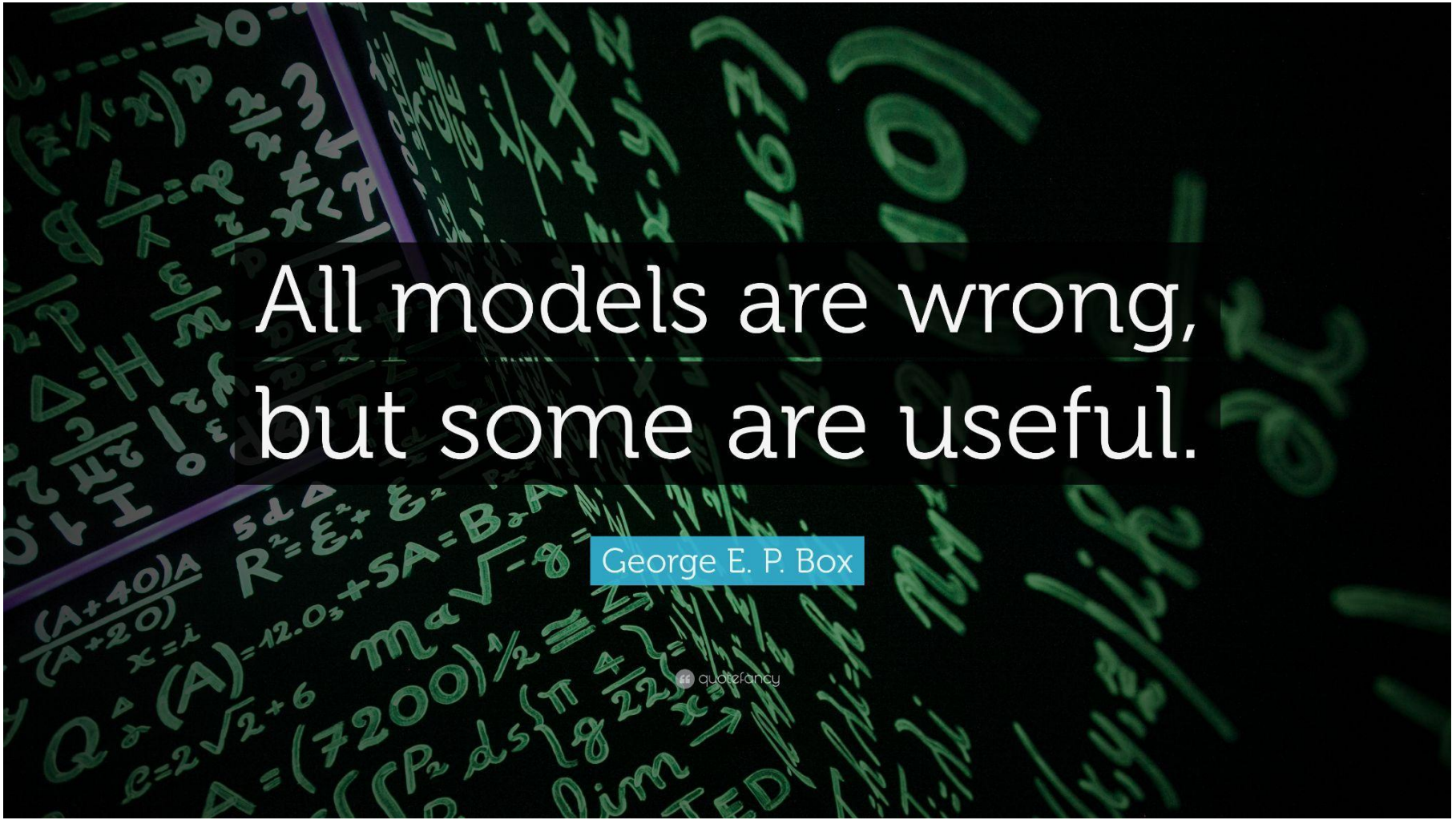
Dr. Jirayus Jiarpakdee

Australian Government Australian Research Council | MONASH University

Email: chakkrit@monash.edu, Twitter: @klainfo






All models are wrong,
but some are useful.
George E. P. Box
quotefancy