Session: Regressions

Regression Algorithms

Linear Regression

Logistic Regression

Licensed for personal use only for Vincent Chang <vincent.chang@macys.com> from Python ML @ Macy's 04/25/2108 @ 2018-04-25

Regressions

➔ **Regression Algorithms**

Linear Regression

Logistic Regression

Licensed for personal use only for Vincent Chang <vincent.chang@macys.com> from Python ML @ Macy's 04/25/2108 @
2018-04-25

What is Regression Analysis

- Regression models relationship between **independent variable(s) (predictor)** and **dependent variable (target)**

- Regressions are used to predict 'numeric' data
  - House prices
  - Stock price

Regression Algorithms

Algorithm	Description	Use Case
Linear Regression	Establishes a best fit 'straight line' Advantages: - Simple, well understood - Scales to large datasets Disadvantages - Prone to outliers	- House prices - Stock market
Logistic Regression	- Calculates the probability of outcome (success or failure) - Used for 'classification' ☺ - Needs large sample sizes for accurate prediction	- Mortgage application approval

Regression Algorithms

Algorithm	Description	Use Case
Polynomial Regression	If power of independent variable is more than 1. $Y = a + b * X^2$ - Can be prone to overfitting - Results can be hard to explain	
Stepwise Regression	- When we have multiple independent variables, automatically selects significant variables - No human intervention - AIC	- House price predictor

Licensed for personal use only for Vincent Chang <vincent.chang@macys.com> from Python ML @ Macy's 04/25/2108 @ 2018-04-25

Regression Algorithms

Algorithm	Description	Use Case
Ridge Regression	- used when independent variables are highly correlated - Uses L2 regularization	
Lasso Regression	- Uses L1 regularization	
ElasticNet Regression	- Hybrid of Lasso and Ridge regressions	

Licensed for personal use only for Vincent Chang <vincent.chang@macys.com> from Python ML @ Macy's 04/25/2108 @ 2018-04-25

Linear Regression

Regression Algorithms

➔ **Linear Regression**

Logistic Regression

Licensed for personal use only for Vincent Chang <vincent.chang@macys.com> from Python ML @ Macy's 04/25/2108 @ 2018-04-25

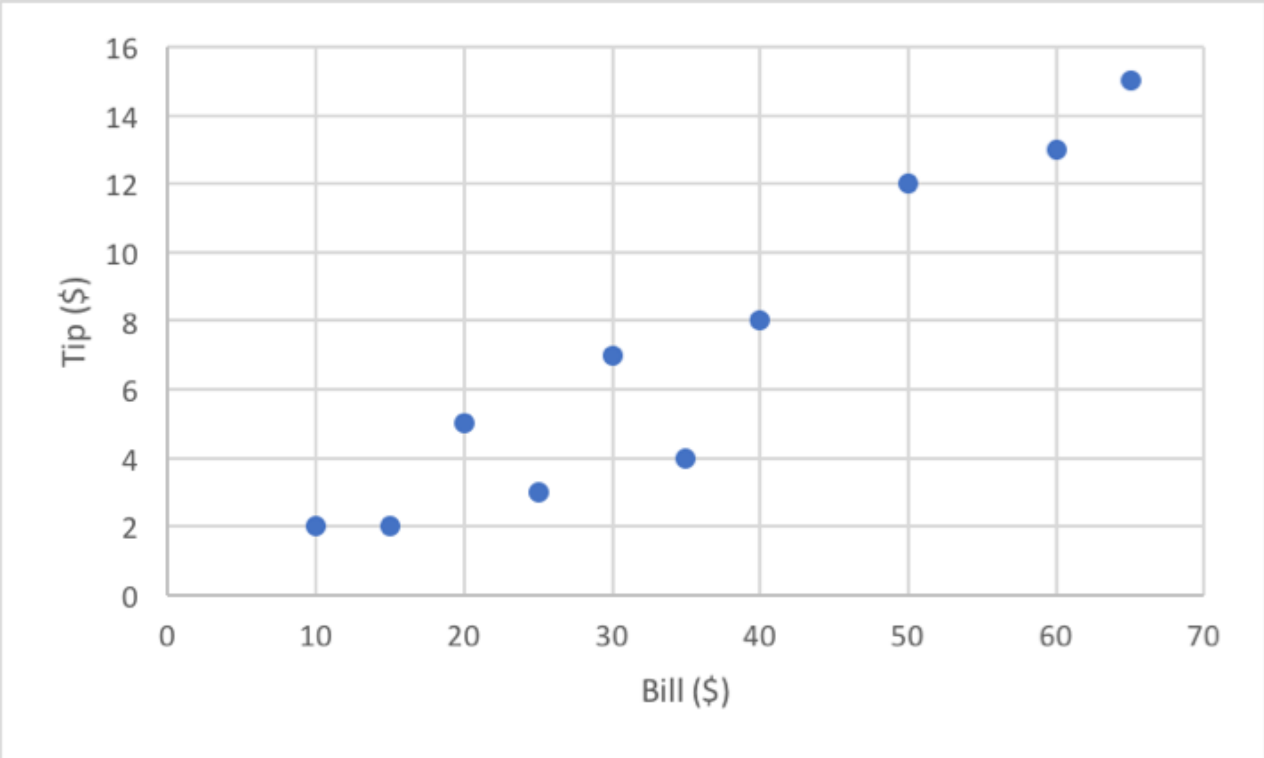
Where Are We?

Category	Sub Category	Example	Algorithms
supervised	Regressions	- Predict house prices	- **➔ Linear Regression 🡐** - Logistic
	Classifications	- Cancer or not - Spam or not	- Trees (random forest ..etc) - SVM
Unsupervised	Clustering	- Group customers (soccer mom, nascar dad)	- Kmeans - Hierarchical clustering
	Dimensionality reduction	- Reduce the number of attributes to consider	- PCA
Semi-supervised		(large amount of data, but only a very small subset is labelled)	

Problem: Tip Calculation

- Now our tip data include total bill amount too !

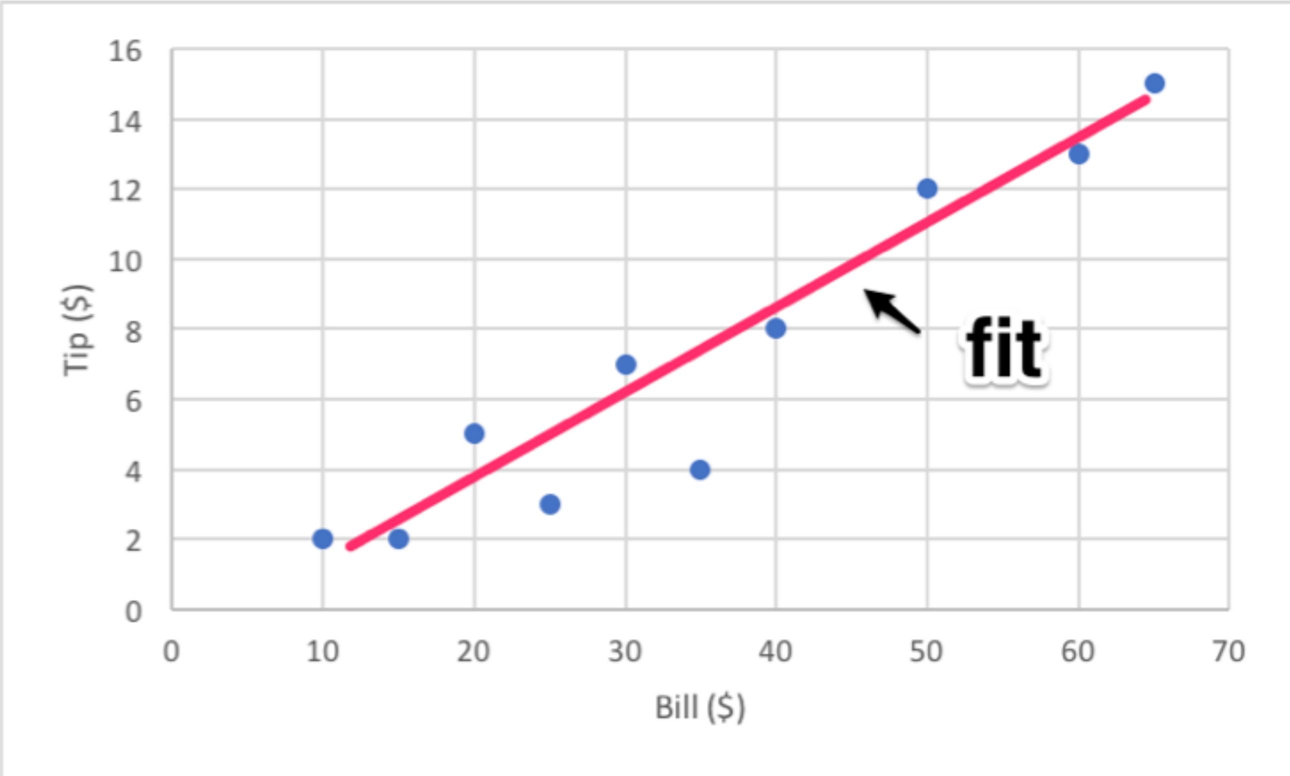
- Do you see any correlation?



Meal #	Bill ($)	Tip ($)
1	50	12
2	30	7
3	60	13
4	40	8
5	65	15
6	20	5
7	10	2
8	15	2
9	25	3
10	35	4

Tips vs Bill

- There is clearly a correlation between bill amount and tip
- We can fit a line to predict tip
- This is **linear regression**!



∑ Math ∫

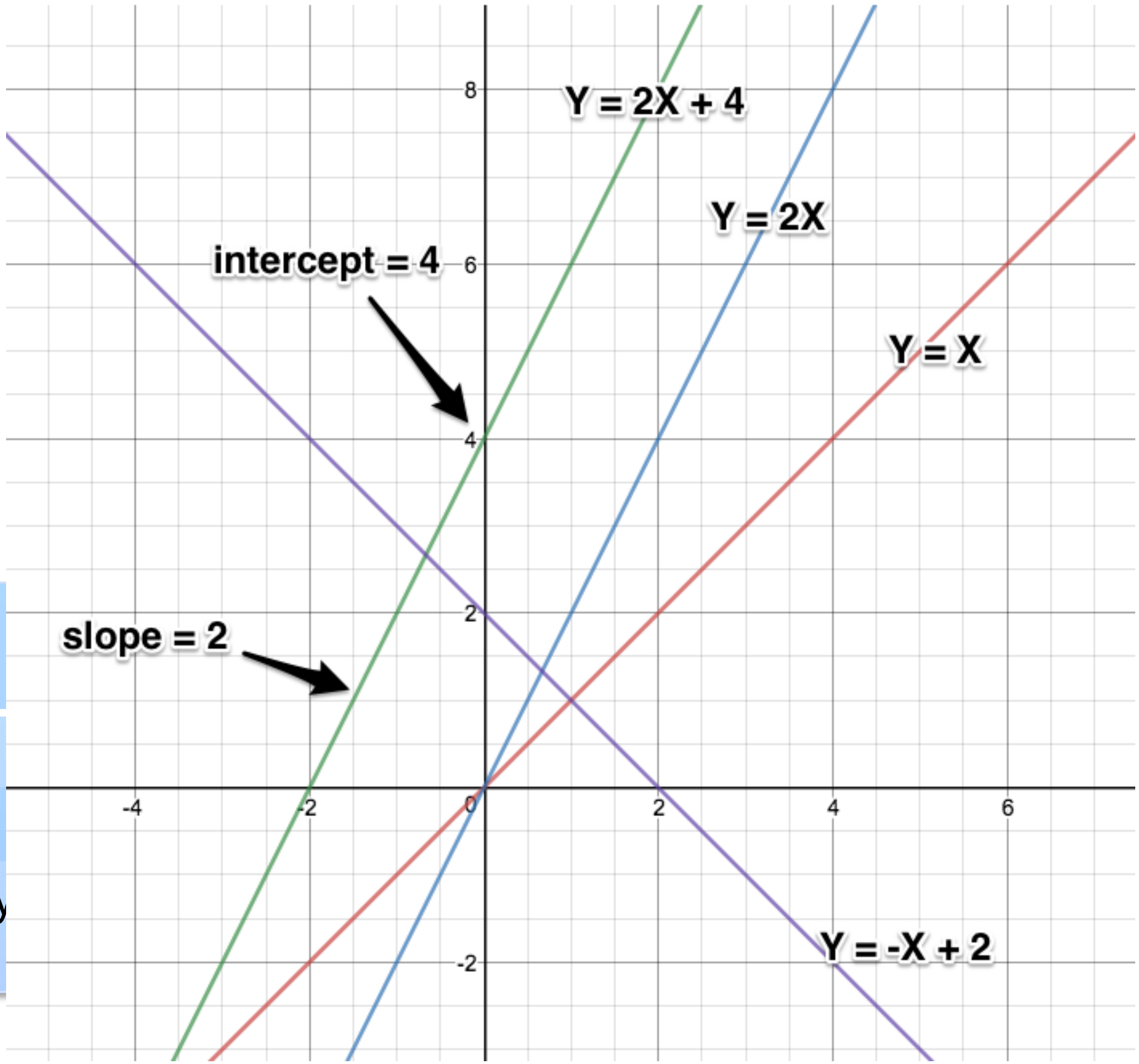
$Y = aX + b$

$Y = 2X + 4$

2 – slope of line

4 – intercept

Y	Dependent variable (depends on X)
X	Independent variable
a	Slope of line
b	Intercept (line meets y axis)



Licensed for personal use only for Vincent Chang <vincent.chang@macys.com> from Python ML @ Macy's 04/25/2108 @ 2018-04-25

Linear Regression Vocabulary

$$Y = aX + b$$

$$Y = b_0 + b_1 X$$

Term	Description	Synonyms
Independent Variable	The variable used to predict the response.	- X-variable - Feature - attribute
Response	The variable we are trying to predict.	- Y-variable - Dependent variable - Target - Outcome
Intercept	The intercept of the regression line - that is, the predicted value when X= 0	- b , b_0 , β_0
Regression coefficient	The slope of the regression line.	- Slope - parameter estimates - Weights - a , b1

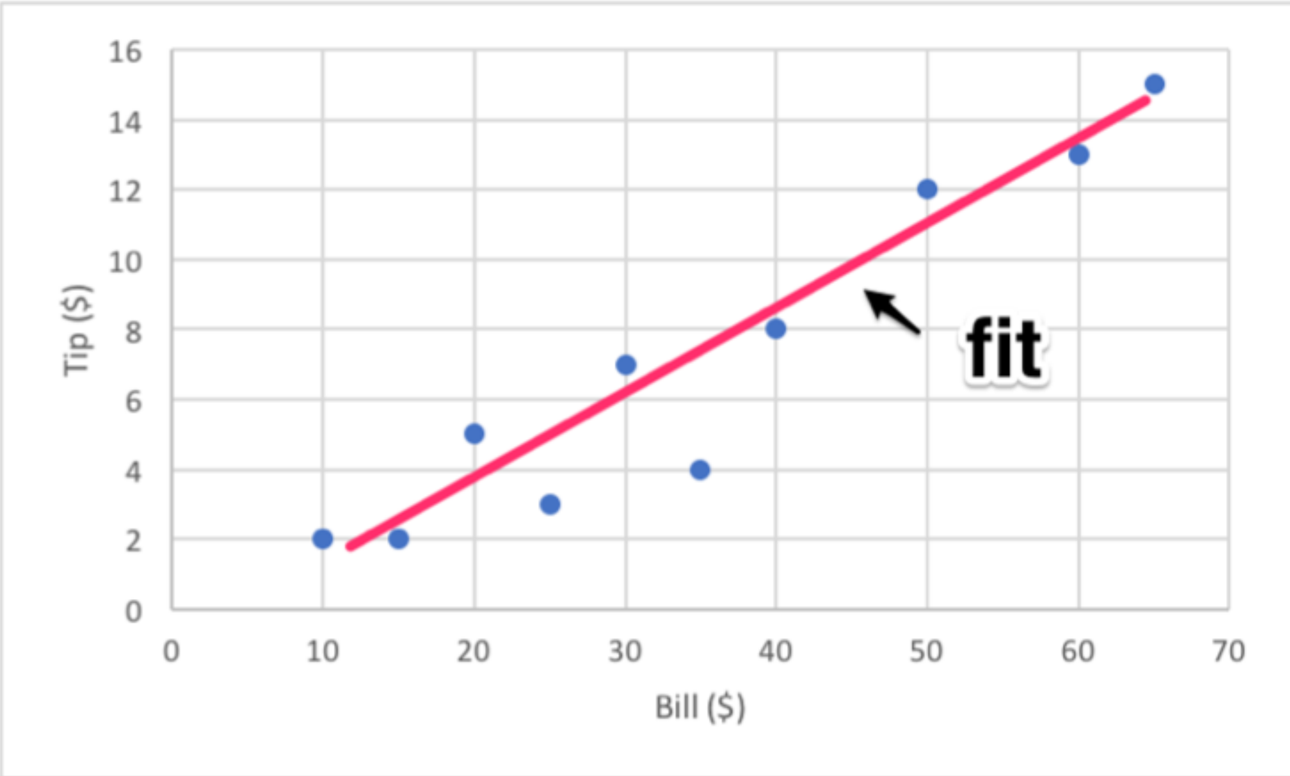
Using Linear Regression for Tips

- Linear regression model closely resembles algebra model

  Y = a X + b

  Tip = a * bill + b

- If we figure out 'a' and 'b', then we can estimate tip for any amount



Licensed for personal use only for Vincent Chang <vincent.chang@macys.com> from Python ML @ Macy's 04/25/2108 @
2018-04-25

Calculating Linear Regression Model

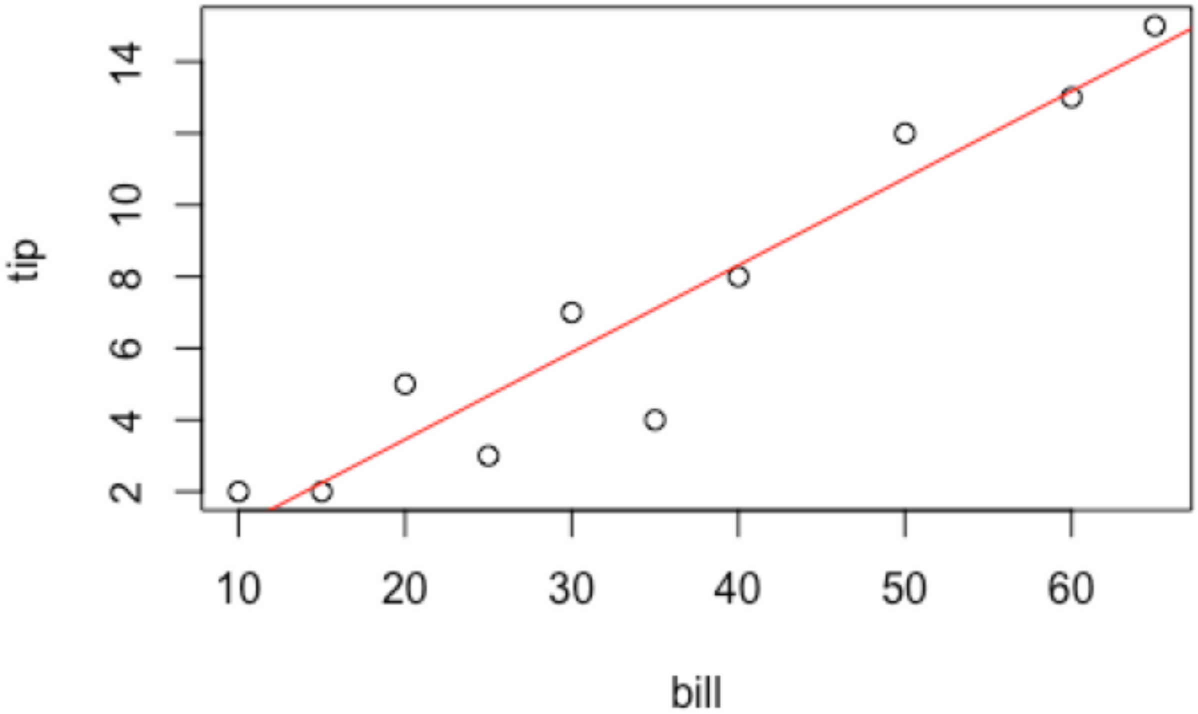
- Here is my Linear Regression Model coefficients

Tip = a * bill + b

a = 0.242

b = −1.40000

Seems like a reasonably good fit



Using Linear Regression Model

Tip =0.2428571 * amount – 1.40

(Tip = a * bill + b)

We can use this formula to predict tips.

Tip for $100 bill

= **0.2428571 * 100 – 1.40**

= $ 22.88

	Bill ($)	Actual tip ($)	estimated tip
observed / known data	50	12	10.742855
	30	7	5.885713
	60	13	13.171426
	40	8	8.314284
	65	15	14.3857115
	20	5	3.457142
	10	2	1.028571
	15	2	2.2428565
	25	3	4.6714275
	35	4	7.0999985
New Data	70	?	15.599997
	80	?	18.028568
	90	?	20.457139
	100	?	22.88571

Licensed for personal use only for Vincent Chang <vincent.chang@macys.com> from Python ML @ Macy's 04/25/2108 @ 2018-04-25

Formula for RSS

∑ Math ∫

$$SS_{(residuals)} = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

- Y_i = actual value
- Y-hat-$_i$ = predicted value

Evaluating Linear Regression Model : RSS

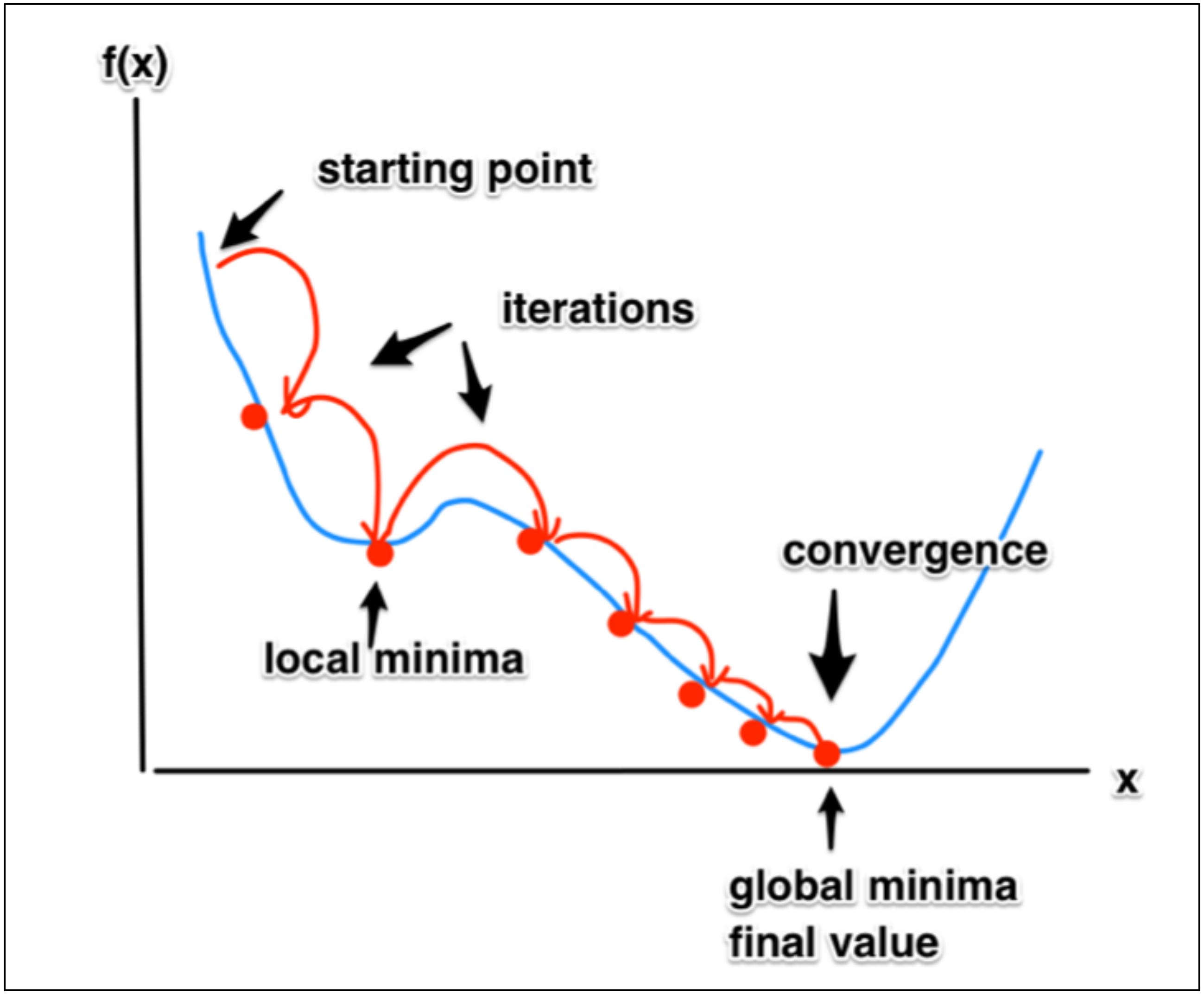
- Quiz:
- Explain the 'observed tip' vs. 'predicted tip'
- Why is sum of residuals zero?
- Why is SSE not zero?
- Why is there no residual on $100 bill ?

	Bill ($)	tip ($)	estimated tip	residual (actual tip - estimated tip)	residual squared	
observed / known data	50	12	10.742855	1.257145	1.580413551	
	30	7	5.885713	1.114287	1.241635518	
	60	13	13.171426	-0.171426	0.029386873	
	40	8	8.314284	-0.314284	0.098774433	
	65	15	14.3857115	0.6142885	0.377350361	
	20	5	3.457142	1.542858	2.380410808	max
	10	2	1.028571	0.971429	0.943674302	
	15	2	2.2428565	-0.2428565	0.05897928	
	25	3	4.6714275	-1.6714275	2.793669888	
	35	4	7.0999985	-3.0999985	9.6099907	min
				0	19.11428571	SSE
New Data	70	?	15.599997			
	80	?	18.028568			
	90	?	20.457139			
	100	?	22.88571			

Licensed for personal use only for Vincent Chang <vincent.chang@macys.com> from Python ML @ Macy's 04/25/2108 @ 2018-04-25

Gradient Descent Algorithm

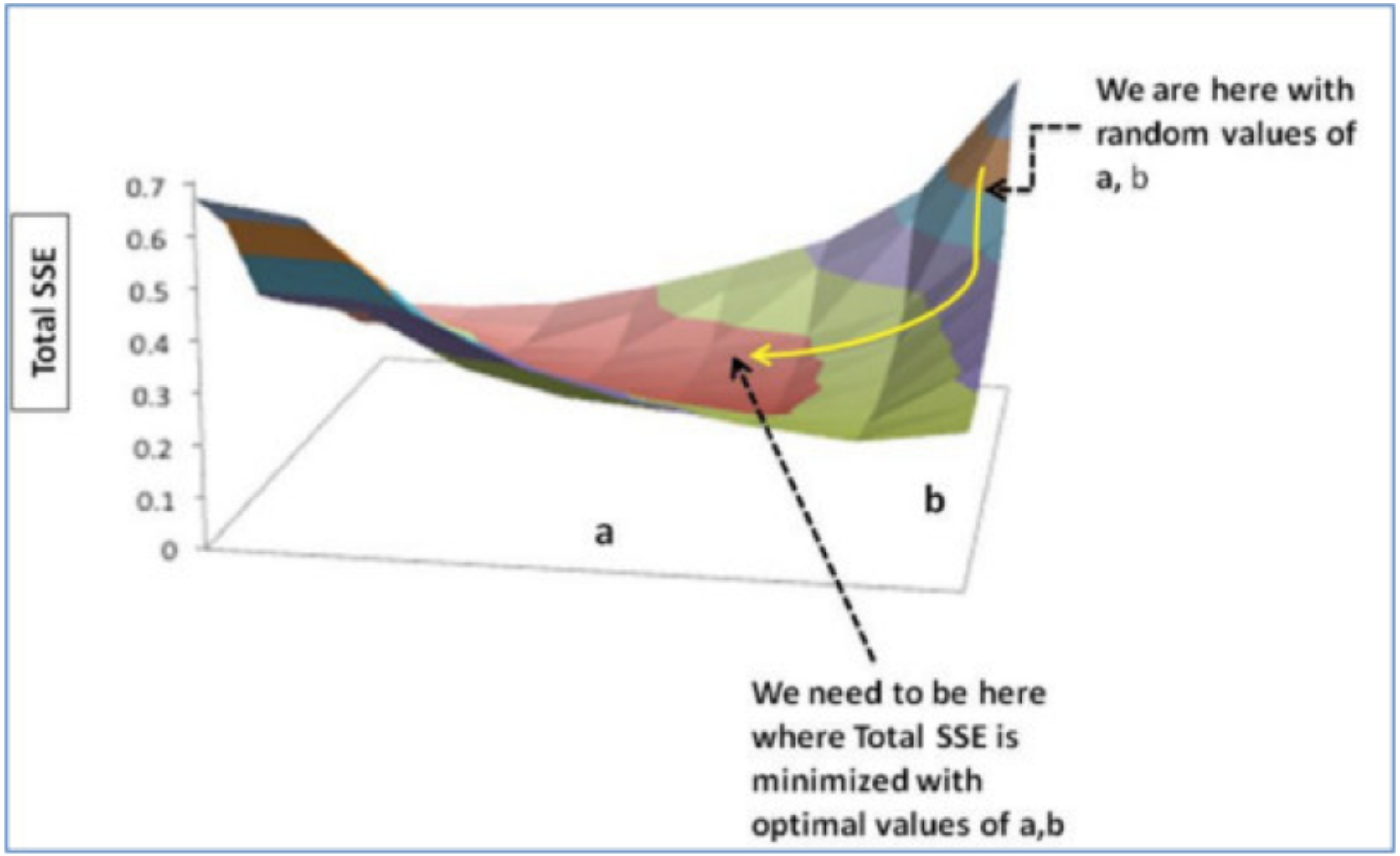
- Iterative algorithm to find **'minimum'** of a function



Licensed for personal use only for Vincent Chang <vincent.chang@macys.com> from Python ML @ Macy's 04/25/2108 @ 2018-04-25

Gradient Descent Algorithm

- Another example in 3D data



Licensed for personal use only for Vincent Chang <vincent.chang@macys.com> from Python ML @ Macy's 04/25/2108 @ 2018-04-25

Evaluating Linear Regression Models

- What is the accuracy of the model
  - Residual Sum of Squares(**RSS**) / Sum of the Squared Errors (**SSE**) / Sum of Squared Residuals (**SSR**):

- How well does our regression equation represent data?
  - Two measures
  - correlation coefficient (r)
  - coefficient of determination (r^2)

- t-statistic

- p-value

Fitted Values & Residuals

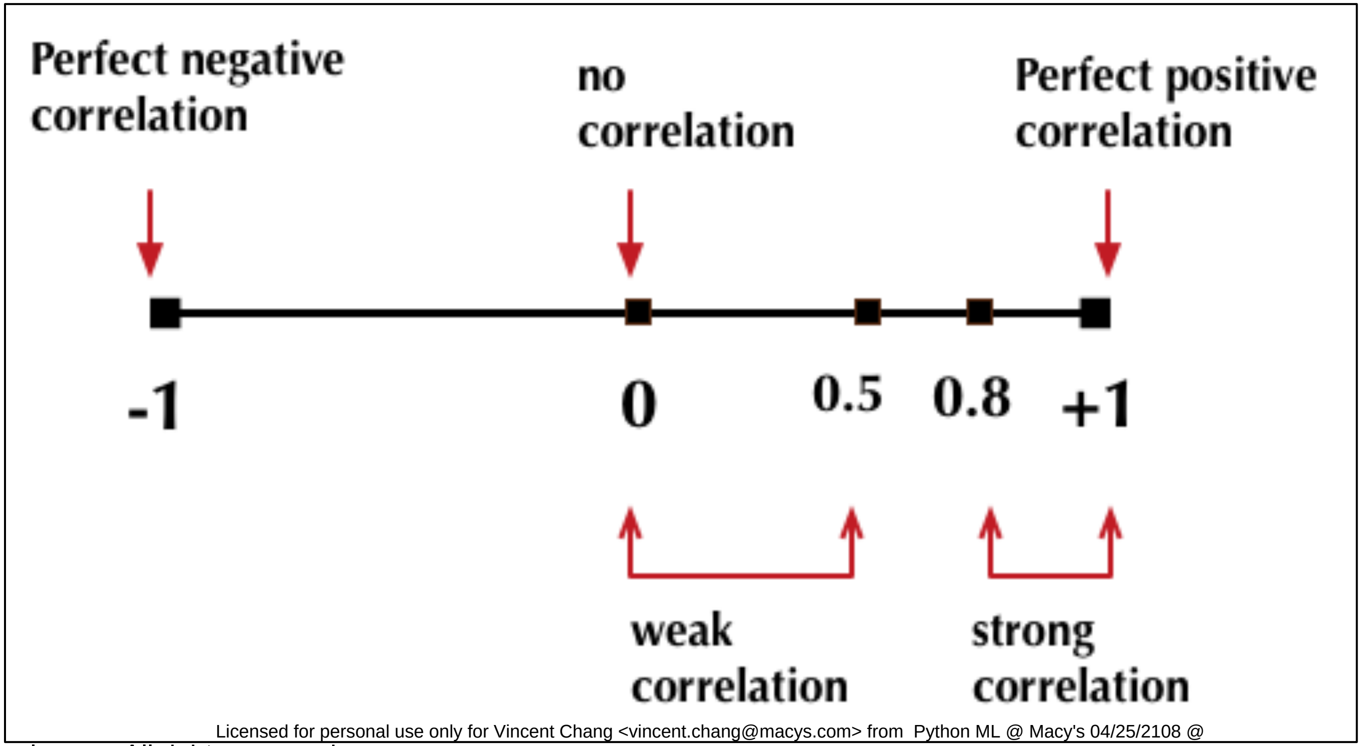
- Data doesn't fall exactly on line
- There is usually a 'delta' or 'error' between actual value and predicted value
- Residual Sum of Squares(**RSS**) / Sum of the Squared Errors (**SSE**) / Sum of Squared Residuals (**SSR**): measures this
- Fitting algorithms try to minimize RSS

Term	Description	Synonyms
Fitted values	The estimates obtained from the regression line.	- predicted values
Residuals	The difference between the observed values and the fitted values.	- errors
Least squares	The method of fitting a regression by minimizing the sum of squared residuals.	- ordinary least squares

Licensed for personal use only for Vincent Chang <vincent.chang@macys.com> from Python ML @ Macy's 04/25/2108 @ 2018-04-25

Evaluating Linear Regression: Correlation Coefficient

- **Perfect correlation** occurs when
  - r = -1 (negative)
  - Or r = +1 (positive)
  - This is when the data points all lie in straight line (regression line!)
- A correlation |r| >= 0.8 is considered **strong**
- A correlation |r| < 0.5 is considered **weak**.

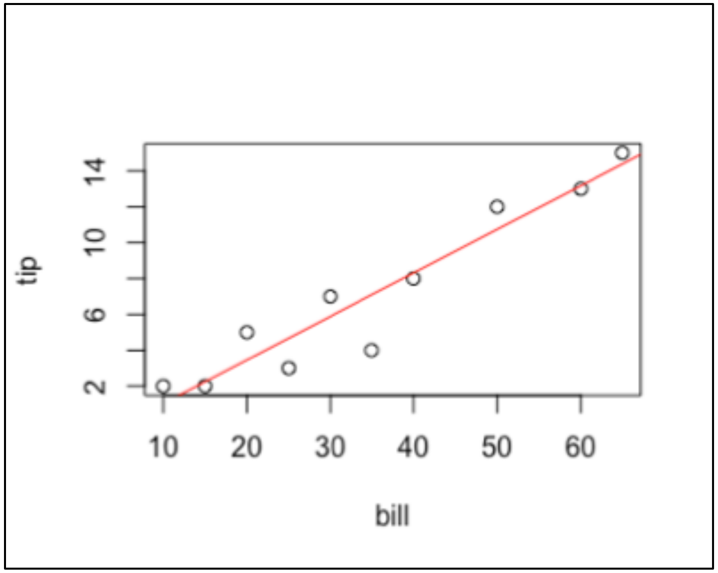


Evaluating Linear Regression Model : Coefficient of Determination

- **'Coefficient of Determination'** tells us how well our model 'fits' the data
- Also referred as **'R squared'** / R^2 / r^2
- Coefficient of Determination = (Correlation Coefficient)^2
- $0 <= r^2 <= 1$
  - $r^2 = 1$: regression line passes through all data points
- In our model
  r = 0.9522154
  r^2 = 0.9067141 = 90.67 %
  That is a pretty good fit !
- Represents the percent of the data that is the closest to the line of best fit
  - So in our case : 90.67% of total variation in Y (tips) can be explained by linear relation between Y (tip) and X (bill)
  - The rest is 'unexplained' by the model

Linear Regression Code (R)

```
tip_data = data.frame(bill = c(50,30,60,40,65,20,10,15,25,35),
                      tip =  c(12, 7,13, 8,15, 5,2, 2, 3, 4))

View(tip_data) # R Studio
plot(tip_data)

tip.lm = lm(tip ~ bill, data=tip_data)

# plot regression against data
abline(tip.lm, col='red')

summary(tip.lm)

Call:lm(formula = tip ~ bill, data = tip_data)

Residuals:
    Min      1Q  Median      3Q     Max
-3.1000 -0.2964  0.2214  1.0786  1.5429

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -1.40000    1.08078  -1.295    0.231
bill         0.24286    0.02754   8.818 2.15e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.546 on 8 degrees of freedom
Multiple R-squared:  0.9067,Adjusted R-squared:  0.8951
F-statistic: 77.76 on 1 and 8 DF,  p-value: 2.153e-05
```



Licensed for personal use only for Vincent Chang <vincent.chang@macys.com> from Python ML @ Macy's 04/25/2108 @ 2018-04-25

Evaluating Linear Model

	Coefficient	Std. Error	t-statistic	p-value
Intercept	-1.4	1.080	-1.295	0.23
Slope (bill)	0.242	0.027	8.818	**0.000215** (2.15e-5)

- **Slope** (bill coefficient) indicates, every $1 increase in bill, will result in $0.242 (almost 25c) in tips.
  $100 increase in bill → $24.2 in tips

- Small **p-values** indicates a strong association between predictor (X) and response (Y).
  - X is statistically significant in deciding Y

Linear Regression Code (R)


```
# estimating fitted values and residuals
fitted = predict(tip.lm)
resid = residuals(tip.lm)

# get them all in one dataframe
a = cbind(tip_data, fitted)
b = cbind(a, resid)
resid.squared = b['resid'] * b['resid']
colnames(resid.squared) = c("resid.squared")
c = cbind(b, resid.squared)
View(c)  # next slide

# calculate some
sum(c['resid'])
-5.551115e-17  #  almost zero!

sum(c['resid.squared'])
19.11429
```

Licensed for personal use only for Vincent Chang <vincent.chang@macys.com> from Python ML @ Macy's 04/25/2108 @
2018-04-25

Visualizing Fitted / Residuals / Residual Squared

actual tip

predicted tip

residuals, delta

squared

	bill	tip	fitted	resid	resid.squared
1	50	12	10.742857	1.2571429	1.58040816
2	30	7	5.885714	1.1142857	1.24163265
3	60	13	13.171429	-0.1714286	0.02938776
4	40	8	8.314286	-0.3142857	0.09877551
5	65	15	14.385714	0.6142857	0.37734694
6	20	5	3.457143	1.5428571	2.38040816
7	10	2	1.028571	0.9714286	0.94367347
8	15	2	2.242857	-0.2428571	0.05897959
9	25	3	4.671429	-1.6714286	2.79367347
10	35	4	7.100000	-3.1000000	9.61000000

Linear Regression, Good & Bad

The Good	The Bad
- Relatively simple to understand - Computationally simple, very scalable to large data sets	- Linear algorithm : will perform poorly if the inputs are not aligned along boundary - Can *underfit* data

Lab: Linear Regressions

Lab

- **Overview**:
  Practice Linear Regressions

- **Approximate Time**:
  30 mins

- **Instructions**:
  - **Linear-regression / 1-lr**
  - Follow appropriate Python / R / Spark instructions

Licensed for personal use only for Vincent Chang <vincent.chang@macys.com> from Python ML @ Macy's 04/25/2108 @ 2018-04-25

Multiple Linear Regression

Regression Algorithms

➔ **Linear Regression**

Logistic Regression

Problem: House Prices

Sale Price $	Bedrooms	Bathrooms	Sqft_Living	Sqft_Lot
280,000	6	3	2,400	9,373
1,000,000	4	3.75	3,764	20,156
745,000	4	1.75	2.060	26,036
425,000	5	3.75	3,200	8,618
240,000	4	1.75	1,720	8,620
327,000	3	1.5	1,750	34,465
347,000	4	1.75	1,860	14,650

- Multiple factors decide house prices
- It is not a simple Y ~ X any more
- We will use **multiple linear regression**

Multiple Linear Regression

$$Y = b_0 + b_1 X_1 + b_2 X_2 + \ \dots + b_p X_p + e$$

- Outcome depends on multiple variables

Licensed for personal use only for Vincent Chang <vincent.chang@macys.com> from Python ML @ Macy's 04/25/2108 @ 2018-04-25

Multitple Linear Regression Code (R)

```
house.sales = read.csv("house-sales.csv")
# 27,000 entries

# run mlr
house.lm = lm(SalePrice ~ Bedrooms + Bathrooms + SqFtTotLiving + SqFtLot,
              data = house.prices, na.action = na.omit)

summary(house.lm)

Call:lm(formula = SalePrice ~ Bedrooms + Bathrooms + SqFtTotLiving +     SqFtLot, data =
house.prices, na.action = na.omit)

Residuals:
     Min        1Q    Median       3Q       Max
 -1955089   -114575   -13670    81734   9081935

Coefficients:
                  Estimate   Std. Error t value              Pr(>|t|)
(Intercept)   106303.30612   6254.77558  16.996 < 0.0000000000000002 ***
Bedrooms      -65211.73613   2151.67471 -30.307 < 0.0000000000000002 ***
Bathrooms      16274.19139   2970.77108   5.478         0.0000000434 ***
SqFtTotLiving    277.84805      2.66890 104.106 < 0.0000000000000002 ***
SqFtLot           -0.07457      0.05472  -1.363                0.173

---Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 246400 on 27058 degrees of freedom
Multiple R-squared:  0.4835,Adjusted R-squared:  0.4834
F-statistic:  6332 on 4 and 27058 DF,  p-value: < 0.00000000000000022
```

Interpreting Results


```
summary(house.lm)

Call:lm(formula = SalePrice ~ Bedrooms + Bathrooms + SqFtTotLiving +     SqFtLot, data =
house.prices, na.action = na.omit)

Residuals:
     Min        1Q    Median       3Q      Max
 -1955089  -114575   -13670    81734  9081935

Coefficients:
                  Estimate   Std. Error t value             Pr(>|t|)
(Intercept)   106303.30612   6254.77558  16.996 < 0.0000000000000002 ***
Bedrooms      -65211.73613   2151.67471 -30.307 < 0.0000000000000002 ***
Bathrooms      16274.19139   2970.77108   5.478         0.0000000434 ***
SqFtTotLiving    277.84805      2.66890 104.106 < 0.0000000000000002 ***
SqFtLot           -0.07457      0.05472  -1.363                0.173

---Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 246400 on 27058 degrees of freedom
Multiple R-squared:  0.4835,Adjusted R-squared:  0.4834
F-statistic:  6332 on 4 and 27058 DF,  p-value: < 0.00000000000000022
```

- Adding one extra 'sqftTotLiving' space increases the house price by $277.85
  - While holding all other variables the same

Predicting Prices – Sample Code (R)

- Let's predict some home prices based on our model

```
new.data = data.frame ('Bedrooms' = c(5,3,2),
                       'Bathrooms' = c(3,2,1.5),
                       'SqFtTotLiving' = c(4400, 1800, 1500),
                       'SqFtLot' = c(10000, 5000, 4000))

predicted.prices = predict(house.lm, new.data)

a = cbind(new.data, predicted.prices)

View(a)
```

	Bedrooms	Bathrooms	SqFtTotLiving	SqFtLot	predicted.prices
1	5	3.0	4400	10000	1050852.9
2	3	2.0	1800	5000	442970.1
3	2	1.5	1500	4000	416764.9

new data

Licensed for personal use only for Vincent Chang <vincent.chang@macys.com> from Python ML @ Macy's 04/25/2108 @
2018-04-25

Evaluating The Model

- **Root Mean Squared Error (RMSE):**
  Square root of the average squared error in predicted values

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

- **Residual Standard Error (RSE):**
  n - observations, p - predictors

$$RSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{(n-p-1)}}$$

Interpreting Results


```
summary(house.lm)

Call:lm(formula = SalePrice ~ Bedrooms + Bathrooms + SqFtTotLiving +     SqFtLot, data =
house.prices, na.action = na.omit)

Residuals:
     Min        1Q    Median       3Q      Max
 -1955089  -114575   -13670    81734  9081935

Coefficients:
                  Estimate   Std. Error t value              Pr(>|t|)
(Intercept)   106303.30612   6254.77558  16.996 < 0.0000000000000002 ***
Bedrooms      -65211.73613   2151.67471 -30.307 < 0.0000000000000002 ***
Bathrooms      16274.19139   2970.77108   5.478         0.0000000434 ***
SqFtTotLiving    277.84805      2.66890 104.106 < 0.0000000000000002 ***
SqFtLot           -0.07457      0.05472  -1.363                0.173

---Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 246400 on 27058 degrees of freedom
Multiple R-squared:  0.4835,Adjusted R-squared:  0.4834
F-statistic:  6332 on 4 and 27058 DF,  p-value: < 0.00000000000000022
```

- RSE is **246400**

Coefficient of Determination (R^2)

- R^2 ranges from 0 to 1
- Measures how well the model fits the data

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2}$$

Interpreting Results – R^2

```
summary(house.lm)

Call:lm(formula = SalePrice ~ Bedrooms + Bathrooms + SqFtTotLiving +     SqFtLot, data =
house.prices, na.action = na.omit)

Residuals:
     Min        1Q    Median        3Q       Max
 -1955089  -114575   -13670     81734   9081935

Coefficients:
                  Estimate   Std. Error t value             Pr(>|t|)
(Intercept)   106303.30612   6254.77558  16.996 < 0.0000000000000002 ***
Bedrooms      -65211.73613   2151.67471 -30.307 < 0.0000000000000002 ***
Bathrooms      16274.19139   2970.77108   5.478         0.0000000434 ***
SqFtTotLiving    277.84805      2.66890 104.106 < 0.0000000000000002 ***
SqFtLot           -0.07457      0.05472  -1.363                0.173

---Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 246400 on 27058 degrees of freedom
Multiple R-squared:  0.4835,Adjusted R-squared:  0.4834
F-statistic:  6332 on 4 and 27058 DF,  p-value: < 0.00000000000000022
```

- R^2 is 0.4835 - not a great fit
- Adjusted R^2 - which adjusts for degrees of freedom. Pretty much the same as R^2 here
- **Question for class** :
  Why is R^2 not close to 1? (as in why is it not a great fit?)

Adding More Variables


```
house.lm = lm(SalePrice ~ Bedrooms + Bathrooms + SqFtTotLiving
+ SqFtLot,    data = house.prices, na.action = na.omit)
```

- Our current formula included only a few attributes : Bedrooms + Bathrooms + SqFtTotLiving + SqFtLot
- Can we add more attributes?

```
house_full <- lm(SalePrice ~ SqFtTotLiving + SqFtLot +
Bathrooms + Bedrooms + BldgGrade + PropertyType +
NbrLivingUnits + SqFtFinBasement + YrBuilt + YrRenovated +
NewConstruction,
         data=house, na.action=na.omit)
```


Licensed for personal use only for Vincent Chang <vincent.chang@macys.com> from Python ML @ Macy's 04/25/2108 @
2018-04-25

Deciding Important Variables

$$Y = b_0 + b_1 X_1 + b_2 X_2 + \ \ldots + b_p X_p + e$$

- In Multiple Linear Regressions many predictors determine the value of response
- How can we know which ones are important?
- Imagine an equation
  $Y = b0 + b1\ X_1 + b2\ X_2$
- We have two predictors X_1 & X_2 ($p = 2$)
- Possible combinations $2^p = 2^2 = 4$
  - No variables
  - X_1 only
  - X_2 only
  - Both X_1 and X_2

Deciding Important Variables

- Possible combinations 2^p can get large for sizeable p values.
  - P = 10 ➔ 2^{10} ➔1024 combinations
  - P = 20 ➔ 2^{20} ➔ 1,048,576 (1 million+) combinations

- Some algorithms to decide important variables quickly
  - Mallow's Cp
  - Akaike Information Criterion (AIC)
  - Bayesian Information Criterion (BIC)

Deciding Important Variables

- There are 3 classical approaches
- **Forward Selection**
  - Begin with null model (has only intercept, and no variables)
  - Run p simple linear regressions and add to null model that results in lowest RSS
- **Backward Selection**
  - Start with all variables
  - Remove variables with largest p-value (least statistically significant)
  - Keep going until desired p-value threshold is reached
- **Mixed Selection**
  - Combination of forward / backward selection

Akaike's Information Criteria (AIC)

- Adding more variables will reduce RMSE and increase R^2 (towards 1)
- How ever that doesn't mean we have a better model
- So we need other measures to evaluate the model
- **Akaike's Information Criteria (AIC)** can be helpful
  - Developed by Hirotugu Akaike, a prominent Japanese statistician
- If I add 'k' more variables the AIC is penalized by atleast 2k
- Goal is to find minimal 'AIC'

```
AIC = 2p  + n log (RSS / n)

p - number of variables
n – number of records
```

Calculating AIC – Sample Code (R)

```
options(scipen=999)
library(MASS)

house.prices = read.csv("house-sales-full.csv")

# using all attributes for LM
house.lm.full <- lm(SalePrice ~ SqFtTotLiving + SqFtLot +
Bathrooms + Bedrooms + BldgGrade + PropertyType +
NbrLivingUnits + SqFtFinBasement + YrBuilt + YrRenovated +
NewConstruction,
       data=house.prices, na.action=na.omit)

step <- stepAIC(house.lm.full, direction="both")

step
```

Calculating AIC – Sample Code (R)


```
# original LM formula
house.lm.full <- lm(SalePrice ~ SqFtTotLiving + SqFtLot + Bathrooms + Bedrooms +
BldgGrade + PropertyType + NbrLivingUnits + SqFtFinBasement + YrBuilt +
YrRenovated + NewConstruction,
       data=house.prices, na.action=na.omit)
```

```
step

Call:
lm(formula = SalePrice ~ SqFtTotLiving + Bathrooms + Bedrooms +     BldgGrade + PropertyType
+ SqFtFinBasement + YrBuilt + NewConstruction,
     data = house.prices, na.action = na.omit)

Coefficients:
             (Intercept)             SqFtTotLiving                  Bathrooms
             5730856.779                   170.255                  37950.708
                Bedrooms                 BldgGrade  PropertyTypeSingle Family
              -44124.897                122498.089                  14862.934
    PropertyTypeTownhouse           SqFtFinBasement                    YrBuilt
               77562.844                     8.153                  -3286.098
      NewConstructionTRUE                  7886.546
```

- **stepAIC** has come up with a new formula
- Dropped attributes :
  SqFtLot, NbrLivingUnits, YrRenovated, and NewConstruction.

Licensed for personal use only for Vincent Chang <vincent.chang@macys.com> from Python ML @ Macy's 04/25/2108 @ 2018-04-25

Lab: Multiple Linear Regression

Lab

- **Overview**:
  Practice Multiple Linear Regressions

- **Approximate Time**:
  30 mins

- **Instructions**:
  Follow appropriate Python / R / Spark instructions
  - LIR-2 : House prices
  - LIR-3 : AIC

Licensed for personal use only for Vincent Chang <vincent.chang@macys.com> from Python ML @ Macy's 04/25/2108 @ 2018-04-25

Linear Regression: Further Reading

Logistic Regression

Regression Algorithms

Linear Regression

➔ **Logistic Regression**

Licensed for personal use only for Vincent Chang <vincent.chang@macys.com> from Python ML @ Macy's 04/25/2108 @ 2018-04-25

Where Are We?

Category	Sub Category	Example	Algorithms
supervised	Regressions	- Predict house prices	- Linear Regression - ➔ **Logistic** 🡐
	Classifications	- Cancer or not - Spam or not	- Trees (random forest ..etc) - SVM
Unsupervised	Clustering	- Group customers (soccer mom, nascar dad)	- Kmeans - Hierarchical clustering
	Dimensionality reduction	- Reduce the number of attributes to consider	- PCA
Semi-supervised		(large amount of data, but only a very small subset is labelled)	

Problem : Applying for Credit Card

- In US most adults have a 'credit score' (a.k.a. FICO score)
- Ranges from 300 (very poor) to 850 (excellent)
- Credit score is a big determining factor when applying for loans / mortgages / credit cards

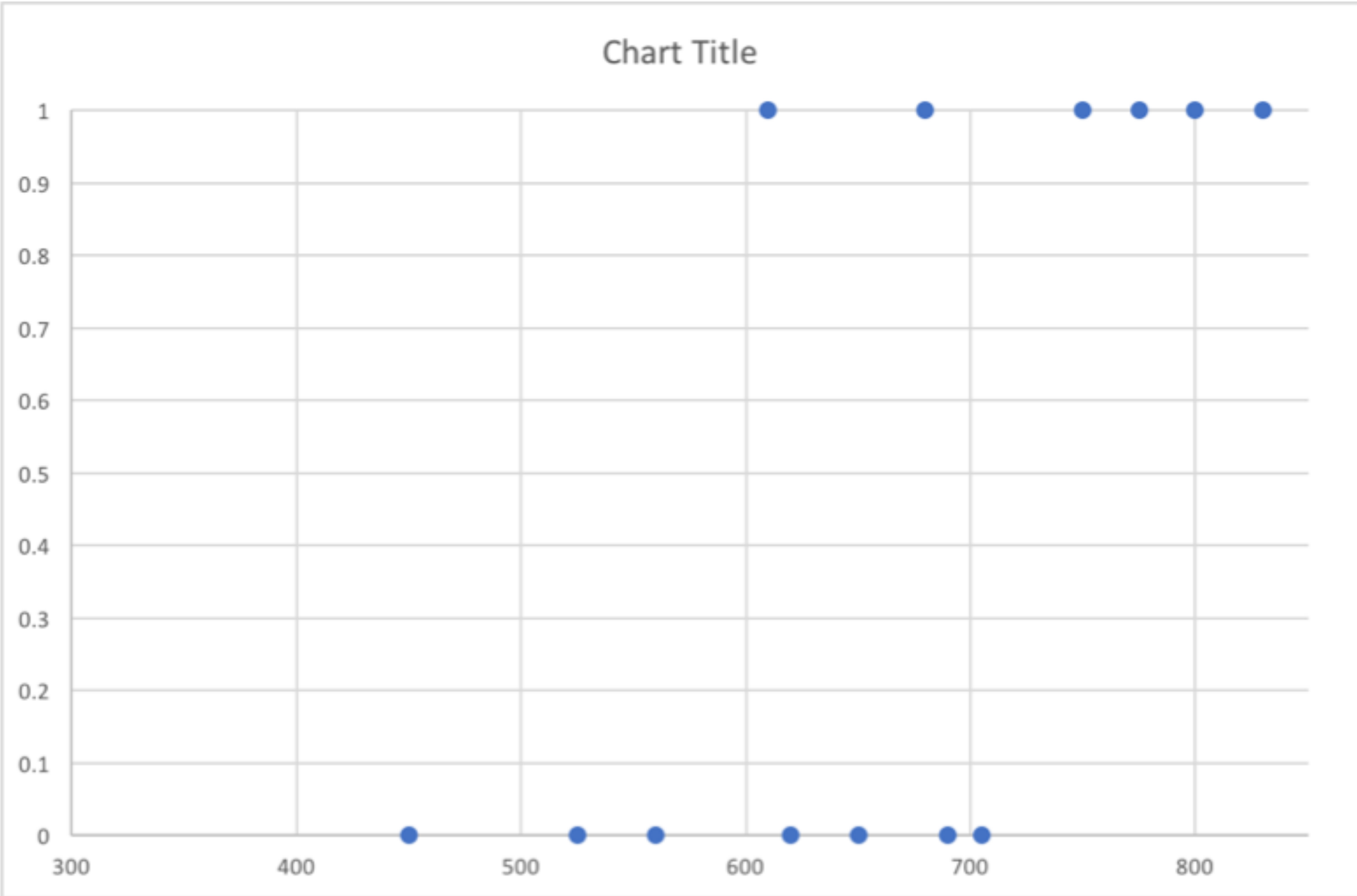
Licensed for personal use only for Vincent Chang <vincent.chang@macys.com> from Python ML @ Macy's 04/25/2108 @ 2018-04-25

Problem : Applying for Credit Card

- Here is historical data on credit score and if the credit application is approved
- What is the chance some one with score of **700** getting a credit card approved?

Credit Score	Approved?
560	No
750	Yes
680	Yes
650	No
450	No
800	Yes
775	Yes
525	No
620	No
705	No
830	Yes
610	Yes
690	No

Plotting Credit Approval Data



Licensed for personal use only for Vincent Chang <vincent.chang@macys.com> from Python ML @ Macy's 04/25/2108 @ 2018-04-25

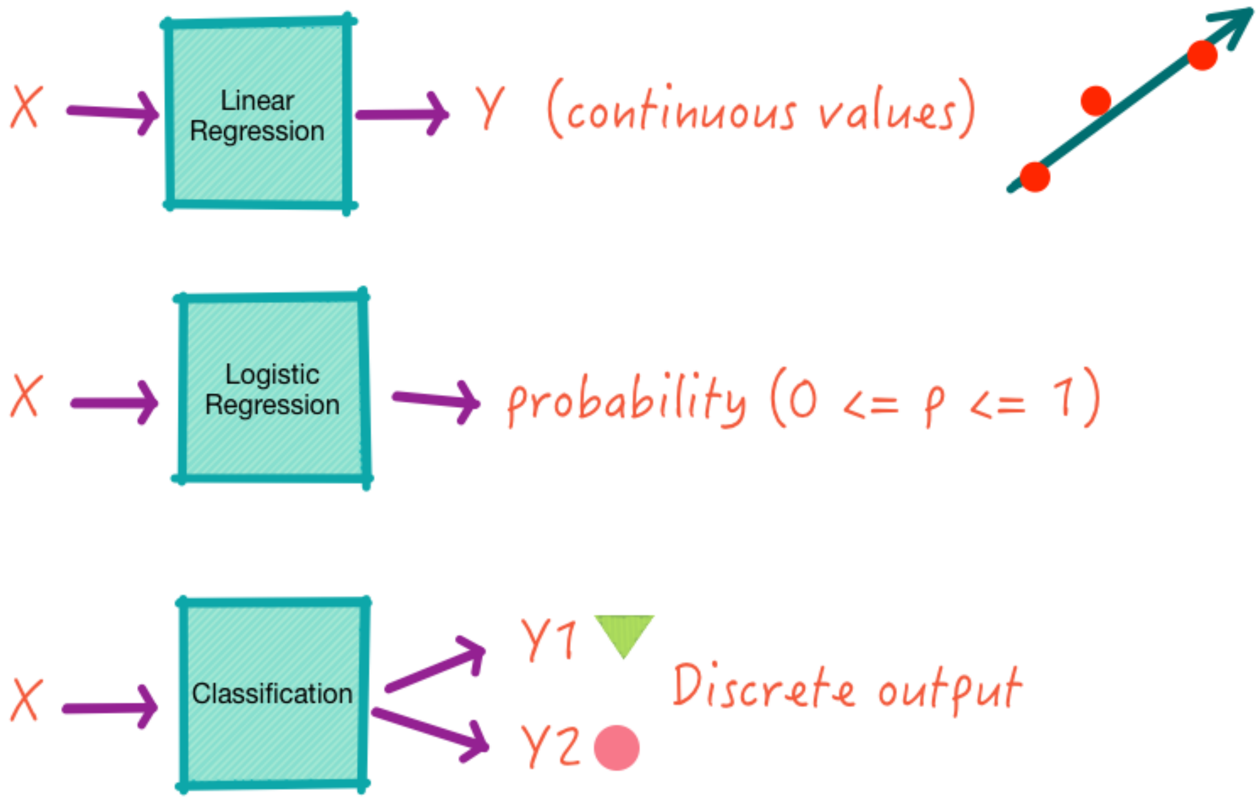
Plotting Credit Approval Data

- X axis = credit score
- Y axis = 0 (declined) , 1 (approved) , nothing in between
- There is no linear fit line !

Licensed for personal use only for Vincent Chang <vincent.chang@macys.com> from Python ML @ Macy's 04/25/2108 @ 2018-04-25

Linear vs. Logistic

- Linear Regression provides continuous Y values
- Classification gives out discrete output (Spam / Not-Spam)
- Logistic Regression is in between
  - Predicts binary outcomes (approved / not–approved)
  - But gives out probability (78% chance this is SPAM, 45% of loan being approved)
  - That is why it is a 'regression' not 'classification'

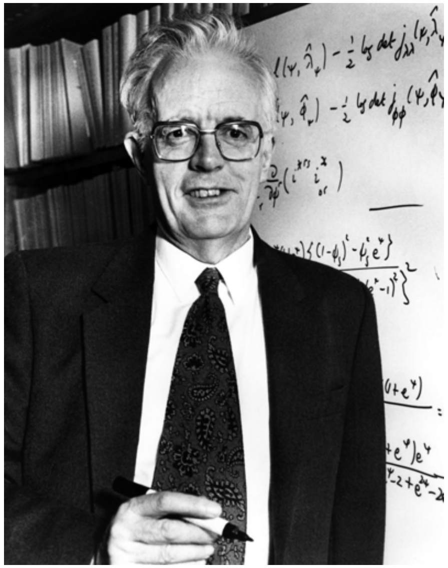


Licensed for personal use only for Vincent Chang <vincent.chang@macys.com> from Python ML @ Macy's 04/25/2108 @ 2018-04-25

Logistic Regression

- Logistic Regression gives out probability
  - 70% chance this email is Spam
- When predicting two outcomes (approved / denied)
  - Binary logistic regression (two outcomes)

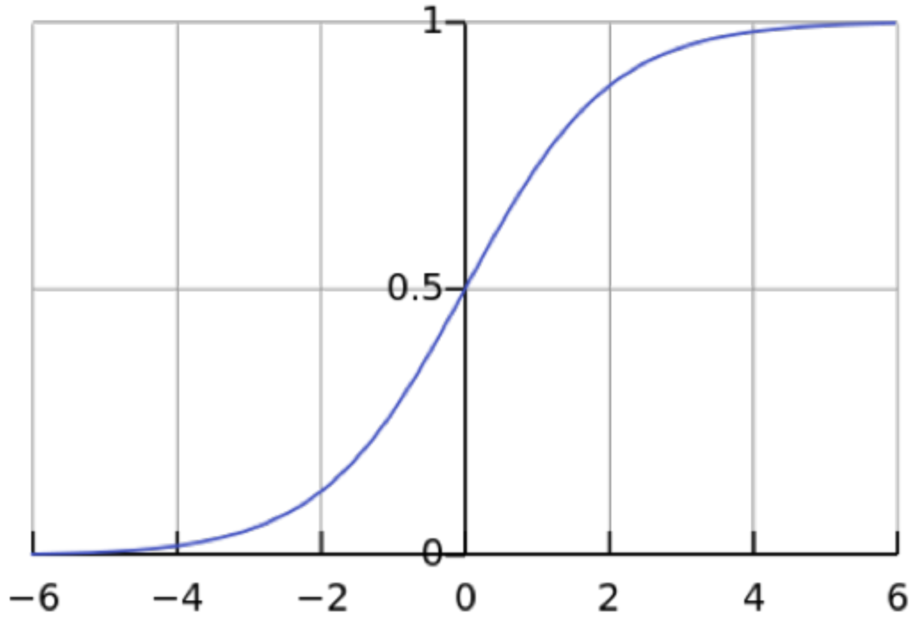
- Invented by Sir David Cox
  (author of 364 books and papers!)



Math Behind Logistic Regression

∑ *Math* ∫

- 'Logit' function
  - Calculates 'odds'



$$f(x) = \frac{L}{1 + e^{-k(x-x_0)}}$$

where

- e = the natural logarithm base (also known as Euler's number),
- x_0 = the x-value of the sigmoid's midpoint,
- L = the curve's maximum value, and
- k = the steepness of the curve.[1]

Math Behind Simple Logistic Regression

$\sum$ *Math* $\int$

Let's say

- β represents parameters
- X is independent variable

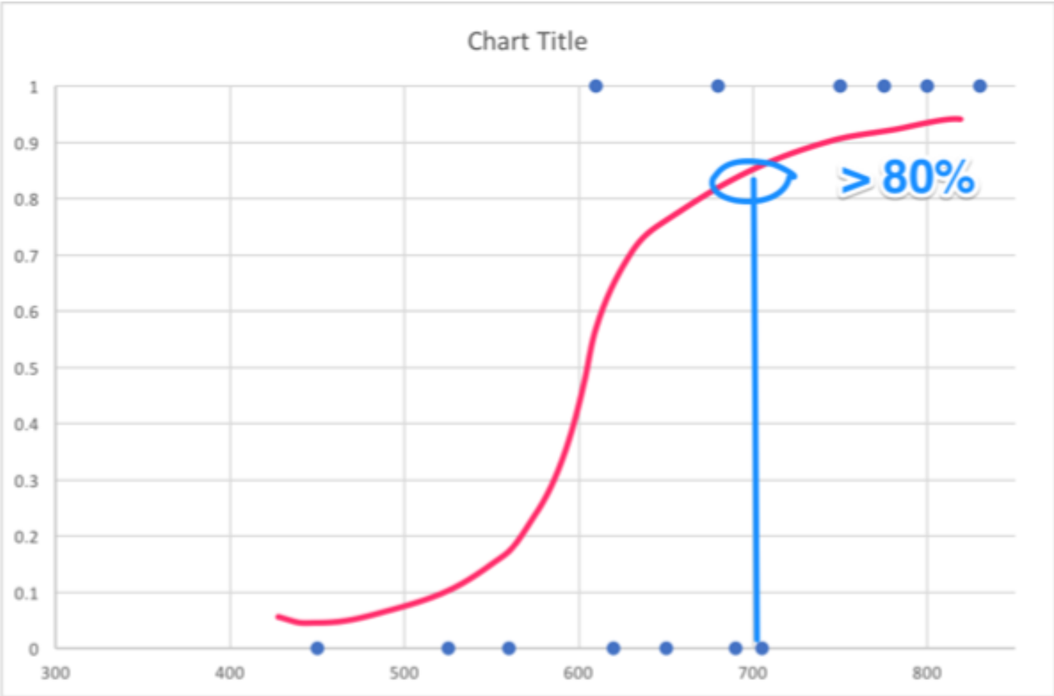
Log(odds) = ln (y / (1-y)) = $\beta_0 + \beta_1 * x_1$

Log (odds) or log-odds ration = $\ln\left(\frac{p}{1-p}\right)$

Where p is the probably the event will occur

Applying Logistic Regression To Credit Card Application

- LR predicts if I have a credit score of 700 (X)
  - I have **~85%** probability of getting a card approved



Licensed for personal use only for Vincent Chang <vincent.chang@macys.com> from Python ML @ Macy's 04/25/2108 @ 2018-04-25

Advantages of logistic regression

- Advantages
  - Provides probability as output
  - Includes multiple explanatory variable (dependent variable)
  - Provides a quantified value for the strength of the association adjusting for other variables

- Preconditions
  - Needs enough data with each possible set of explanatory variable
  - Need to test the assumption of linearity before including it in the model
  - Combines both binomial and normal distribution. This can sometimes cause problems
  - Defining variables can be complicated and must be carefully planned

Licensed for personal use only for Vincent Chang <vincent.chang@macys.com> from Python ML @ Macy's 04/25/2108 @ 2018-04-25

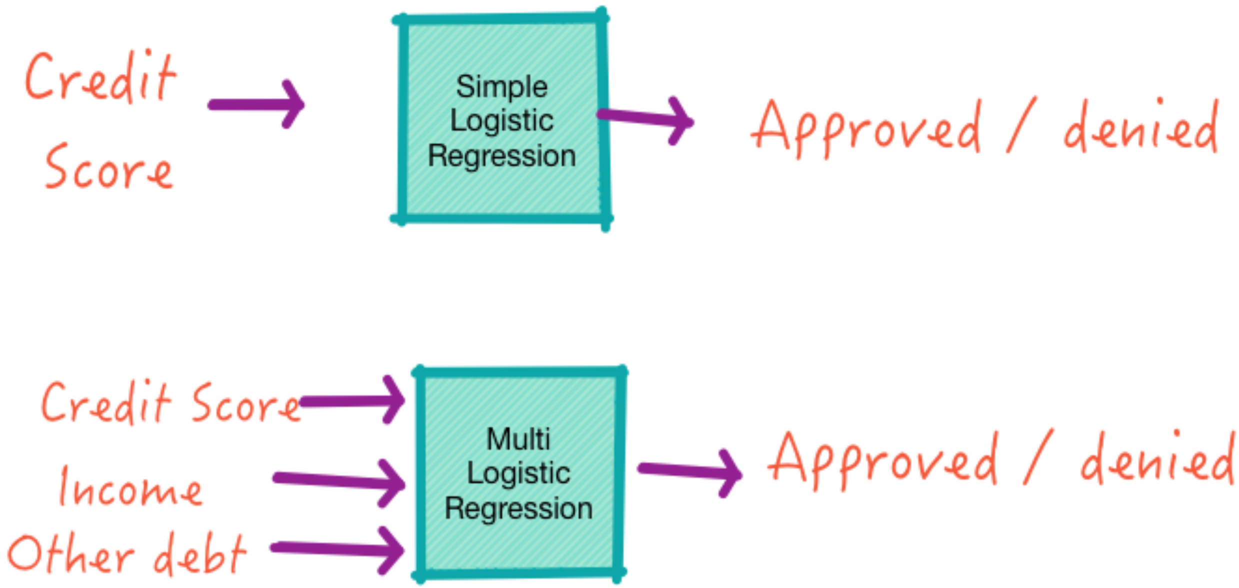
Multinomial logistic regression

- We have seen Logistic Regression predicting binary outcomes
  - Approved / Denied
- We can use it to calculate 'more than two' states as well
  - multinomial logistic regression

- For K possible outcomes
  - Chose one outcome as a "pivot"
  - The other K−1 outcomes can be separately regressed
    - against the pivot outcome

Multiple Logistic Regression

- So far we have seen ONE predictor determining the outcome
  - Credit score determining approval / denial
- We can have multiple factors (independent variables) determining an outcome as well
  - This is called 'multiple logistic regression'



Math Behind Multiple Logistic Regression

$\sum$ *Math* $\int$

Let's say

- β represents parameters
- X is independent variable (we have more than one)

Log(odds) = ln (y / (1-y)) = $\beta_0 + \beta_1{*}x_1 + \beta_2{*}x_2 + \ldots. + \beta_n{*}x_n$

Log (odds) or log-odds ration = $\ln(\frac{p}{1-p})$

Where p is the probably the event will occur

Lab: Logistic Regression

Lab

- **Overview**:
  Practice Logistic Regression
- **Approximate Time**:
  30 mins
- **Instructions**:
  Follow appropriate Python / R / Spark instructions
  - LOGR-1 : Credit card approval
  - LOGR-2 :

Review Questions

Logistic Regression: Further Readings

Licensed for personal use only for Vincent Chang <vincent.chang@macys.com> from Python ML @ Macy's 04/25/2108 @
2018-04-25