Probability and Statistics for Data Science

Y-DATA

13.11.2020

This course

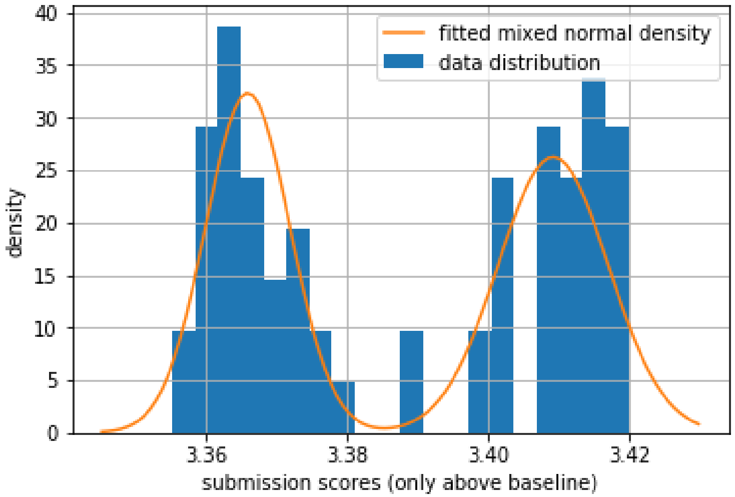
Probability & decision making	How probability is applied in data science Decision making with probabilistic models Descriptive statistics and visualization
Distributions and parameters	Important distributions and their characteristics Methods of parameter estimation
Hypotheses testing	Evaluating uncertainty of sample estimates Tests for comparing means Tests for goodness of fit
Predictive models	Mathematics of joint distributions Linear regression as a statistical tool Inference with nonlinear models

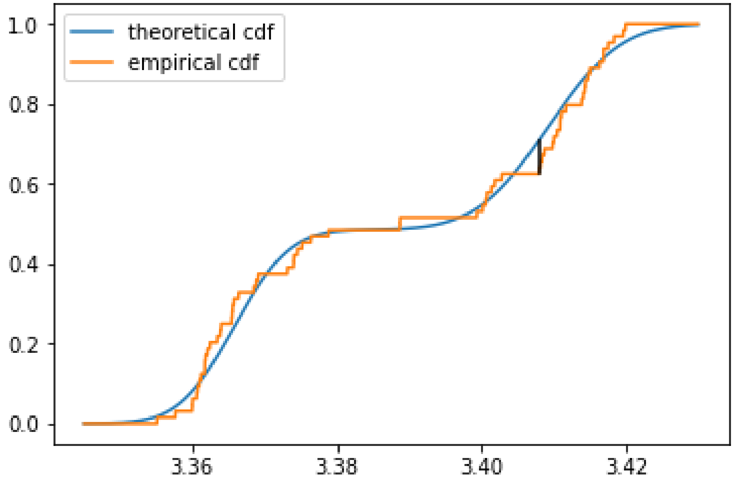
Testing for goodness of fit

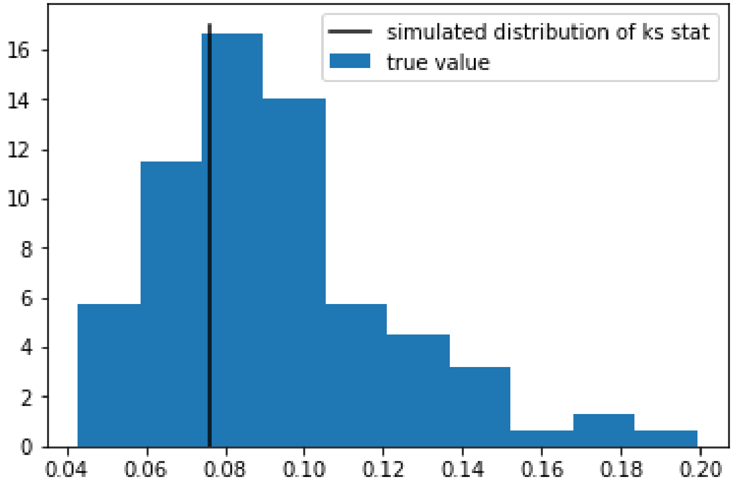
If one normal distribution cannot fit the data, maybe a mixture of two normal distributions is ok?

Yes, we have to estimate parameters numerically (by maximum likelihood)

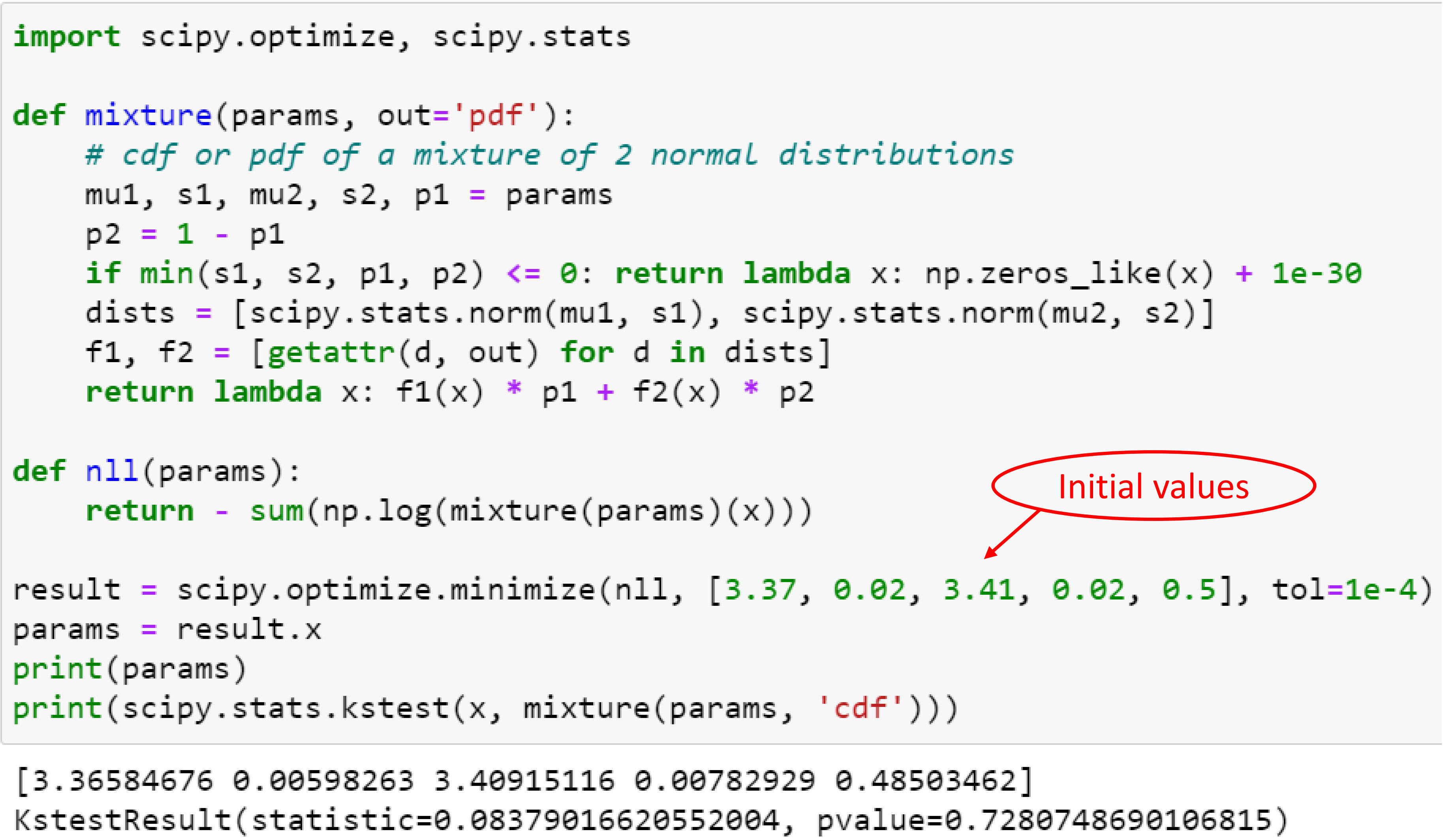
After this, we can use e.g KS to test for the goodness of fit.







Fitting and testing a mixture: code



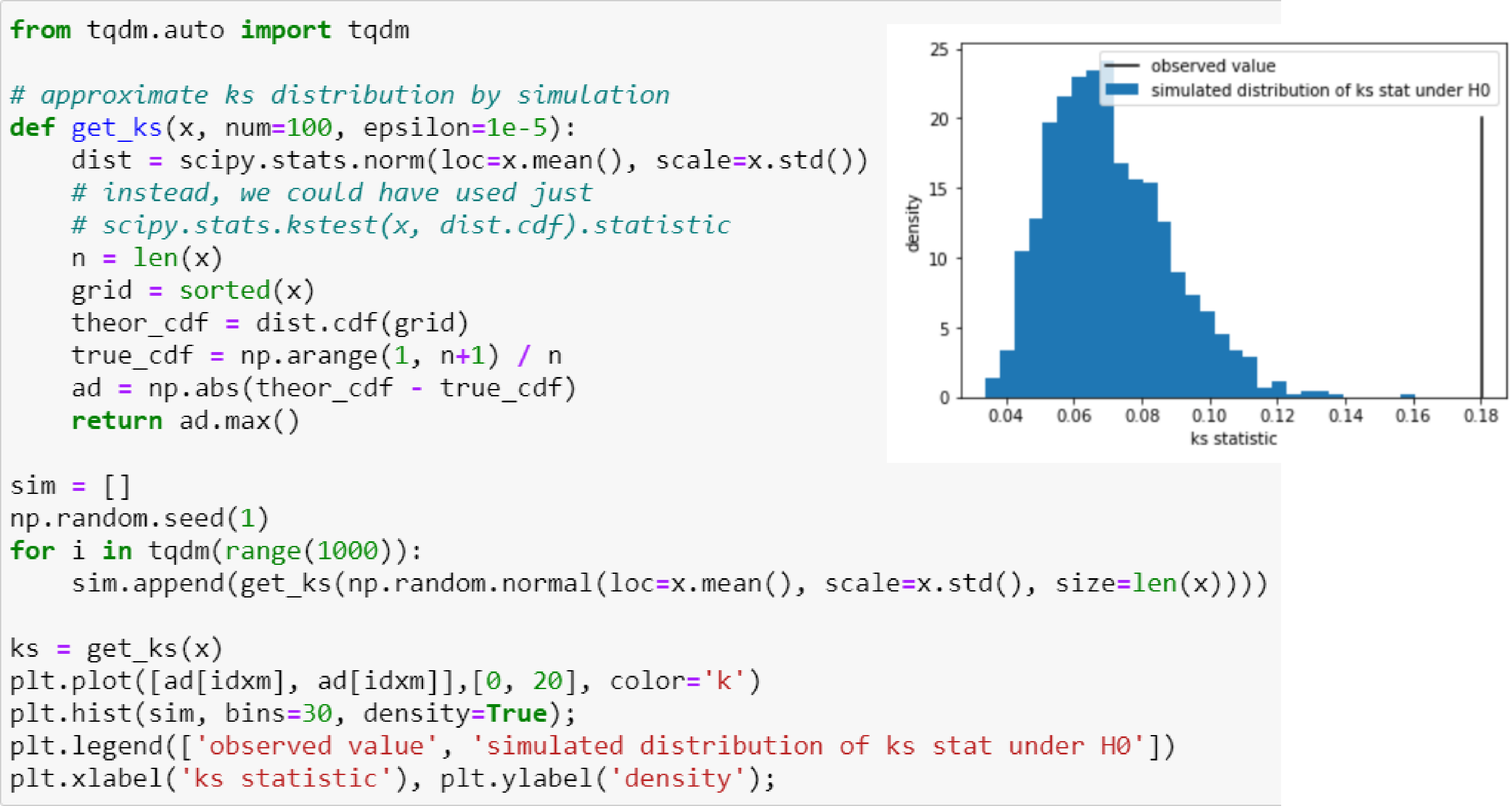
```python
import scipy.optimize, scipy.stats

def mixture(params, out='pdf'):
    # cdf or pdf of a mixture of 2 normal distributions
    mu1, s1, mu2, s2, p1 = params
    p2 = 1 - p1
    if min(s1, s2, p1, p2) <= 0: return lambda x: np.zeros_like(x) + 1e-30
    dists = [scipy.stats.norm(mu1, s1), scipy.stats.norm(mu2, s2)]
    f1, f2 = [getattr(d, out) for d in dists]
    return lambda x: f1(x) * p1 + f2(x) * p2

def nll(params):
    return - sum(np.log(mixture(params)(x)))

result = scipy.optimize.minimize(nll, [3.37, 0.02, 3.41, 0.02, 0.5], tol=1e-4)
params = result.x
print(params)
print(scipy.stats.kstest(x, mixture(params, 'cdf')))
```

```
[3.36584676 0.00598263 3.40915116 0.00782929 0.48503462]
KstestResult(statistic=0.08379016620552004, pvalue=0.7280748690106815)
```

Initial values

P-value by simulation: code

Here we test the hypothesis that X was generated from normal distribution
The resulting p-value is 0 (simulated KS statistic exceeded 0.18 exactly 0 times)



```python
from tqdm.auto import tqdm

# approximate ks distribution by simulation
def get_ks(x, num=100, epsilon=1e-5):
    dist = scipy.stats.norm(loc=x.mean(), scale=x.std())
    # instead, we could have used just
    # scipy.stats.kstest(x, dist.cdf).statistic
    n = len(x)
    grid = sorted(x)
    theor_cdf = dist.cdf(grid)
    true_cdf = np.arange(1, n+1) / n
    ad = np.abs(theor_cdf - true_cdf)
    return ad.max()

sim = []
np.random.seed(1)
for i in tqdm(range(1000)):
    sim.append(get_ks(np.random.normal(loc=x.mean(), scale=x.std(), size=len(x))))

ks = get_ks(x)
plt.plot([ad[idxm], ad[idxm]],[0, 20], color='k')
plt.hist(sim, bins=30, density=True);
plt.legend(['observed value', 'simulated distribution of ks stat under H0'])
plt.xlabel('ks statistic'), plt.ylabel('density');
```

Simulating a distribution

- What you have: a sample $X_1, \ldots X_n$, a hypothesis H_0, and a formula for the test statistic $t(sample)$
- What you want: a p-value for your test
- What to do:

```
real_statistic = test_statistic(sample)
n = size(sample)

simulated_stats = []
for i in range(n_iterations):
    simulated_sample = random.sample(distribution=H0, size=n)
    simulated_stats.append(test_statistic(simulated_sample))

p_value = mean(real_statistic > simulated_stats)
```

For one-sided hypotheses

Joint distributions

- So far, we have mostly studied distributions of a single random variable X
- Instead, we can study the **joint** distribution of a random vector $X = (X_1, X_2, \dots, X_k)$
  - E.g. joint CDF $F_{X_1,\dots,X_k}(x_1, \dots, x_k) = P(X_1 \leq x_1, \dots, X_k \leq x_k)$
  - E.g. join PDF $f_{X_1,\dots,X_k}(x_1, \dots, x_k) = \frac{\partial^k F(x_1,\dots,x_k)}{\partial x_1 \partial x_2 \dots \partial x_k}$
- If the variables are independent, the joint distribution is trivial
  - $F_{X_1,\dots,X_k}(x_1, \dots, x_k) = F_{X_1}(x_1) F_{X_2}(x_2) \dots F_{X_k}(x_k)$ by definition
  - $f_{X_1,\dots,X_k}(x_1, \dots, x_k) = f_{X_1}(x_1) f_{X_2}(x_2) \dots f_{X_k}(x_k)$
  - **Conditional** distribution $p(X_i | X_j)$ equals **marginal** $p(X_i)$
- If the variables are dependent, things get interesting (and practical)

Categorical independence

- Which pairs of variables are independent?
  - W and T
  - X and Y
  - W and T, X and Y
  - None of them

$P(W = w, T = t)$	$T = 1$	$T = 2$	$T = 3$
$W = 10$	0.1	0.2	0.2
$W = 20$	0.1	0.1	0.3

$P(X = x, Y = y)$	$X = 1$	$X = 2$	$X = 3$
$Y = 10$	0.12	0.18	0.3
$Y = 20$	0.08	0.12	0.2

- $P(W = 10, T = 2) \neq P(W = 10) \times P(T = 2)$
- $P(X = x, Y = y) = P(X = x) \times P(Y = y)$ for all x, y

Tests for independence

- How to tell from a sample if X and Y are independent?
- X and Y are categorical:
  - Compare counts: e.g. χ^2 test
  - The value χ^2/n can show the degree of dependence
- X is categorical and Y is numerical
  - Compare group means (t or F tests) or variances
  - Compare conditional CDFs (paired KS test)
- Both X and Y are numerical
  - Estimate correlation coefficient for linear dependence
  - Rank correlation (e.g. Spearman) works with any monotonic dependence
  - Test that correlation is 0 (see the next slide)

Correlation coefficients

- Pearson correlation
  - $r = \frac{\frac{1}{n}\sum(X_i - \bar{X})(Y_i - \bar{Y})}{\hat{\sigma}_X \hat{\sigma}_Y} = \frac{\widehat{Cov}(X,Y)}{\hat{\sigma}_X \hat{\sigma}_Y}$
  - Equals ± 1 iff the relationship is exactly <u>linear</u>
  - In large or normal samples, $t = r\sqrt{\frac{n-2}{1-r^2}} \sim t_{n-2}$, in case if X and Y are independent
  - Independence implies 0 correlation, but not vice versa
- Spearman correlation
  - $\rho = \frac{\widehat{Cov}(rank(X), rank(Y))}{\hat{\sigma}_{rank(X)} \hat{\sigma}_{rank(Y)}}$ (rank is the position after sorting the dataset)
  - Equals ± 1 iff the relationship is exactly <u>monotonic</u>
  - Has the same asymptotic distribution as Pearson correlation
  - Less sensitive to outliers

Example: is correlation significant?

- On a sample of 38 school students, Pearson correlation between height and score on a math test is 0.6
- Test the hypothesis that the true correlation is 0 with 5% significance level
  - Accept H_0: we cannot prove that correlation is not 0
  - Reject H_0: correlation is not 0

- $t = r\sqrt{\frac{n-2}{1-r^2}} = 0.6\sqrt{\frac{38-2}{1-0.6^2}} = 0.6\sqrt{\frac{36}{0.64}} = 4.5$
- If H_0 is true, it has t_{38-2} distribution with 97.5% quantile of 2.02.
- The observed value of test statistic, 4.5, is much higher, so we reject the no-correlation hypothesis

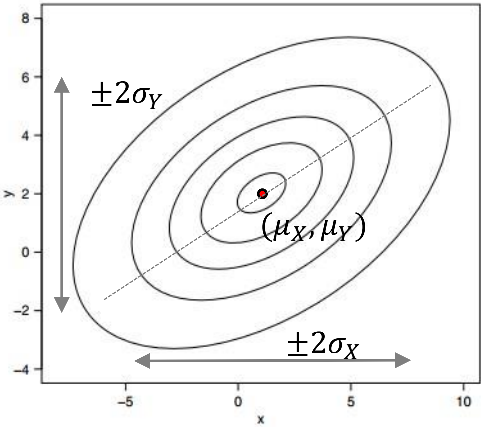
Covariance matrix

- Distribution of a random vector can be summarized by first 2 moments
  - mean vector $\mu_X = \mathbb{E}X = (\mathbb{E}X_1, \dots, \mathbb{E}X_k)$
  - covariance matrix C_X, e.g. for $k = 3$:

$$C_X = \begin{pmatrix} Var(X_1) & Cov(X_1, X_2) & Cov(X_1, X_3) \\ Cov(X_1, X_2) & Var(X_2) & Cov(X_2, X_3) \\ Cov(X_1, X_3) & Cov(X_2, X_3) & Var(X_3) \end{pmatrix}$$

- Now what can we say about a random scalar $\alpha^T X = \alpha_1 X_1 + \alpha_2 X_2 + \cdots + \alpha_k X_k$?
  - Its mean is $\alpha^T \mu_X$
  - Its variance is $\alpha^T C \alpha$

Joint normal

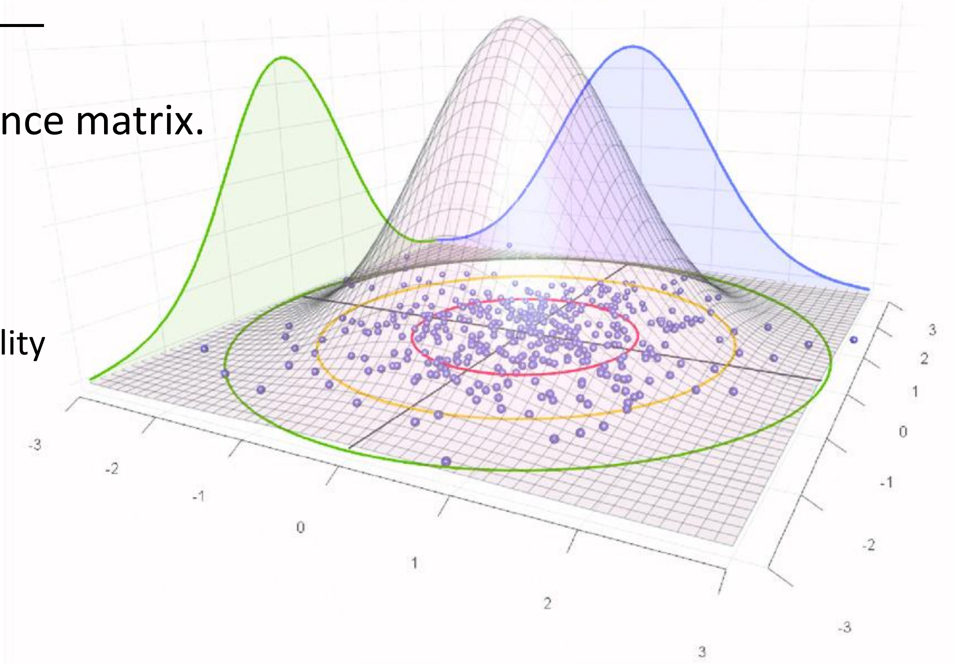


Bivariate case: $f_{X,Y}(x,y) = \frac{1}{Z} e^{-\frac{1}{2(1-\rho^2)}(\tilde{x}^2 - 2\rho\tilde{x}\tilde{y} + \tilde{y}^2)}$

- Here $\tilde{x} = \frac{x-\mu_X}{\sigma_X}$, $\tilde{y} = \frac{y-\mu_Y}{\sigma_Y}$, ρ is a parameter (btw, it is correlation)
- and $Z = 2\pi\sigma_X\sigma_Y\sqrt{1-\rho^2}$ is the normalizing constant.
- Interpretation: most of density is within an ellipse

K-dimensional case: $f(x) = \frac{e^{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)}}{\sqrt{(2\pi)^k|\Sigma|}}$

Here μ and Σ are mean vector and covariance matrix.



Joint vs individual:

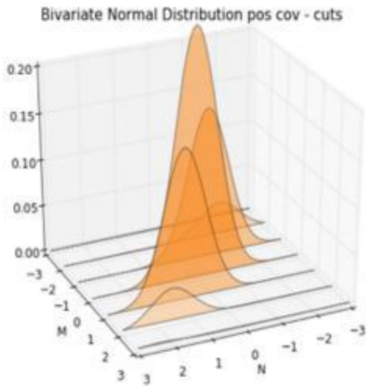
- Individual normality does **not** imply joint normality
- But joint normality means that marginal and conditional distributions are normal

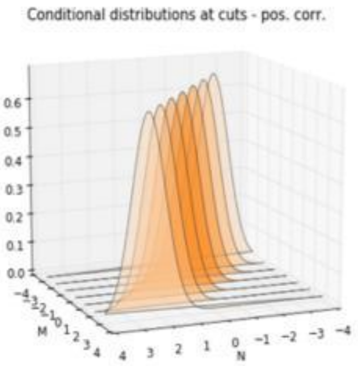
Conditional normal distribution

- Conditional joint normal is also normal!
  - Because it’s the renormalized joint density. E.g. bivariate

$$f\,(y|X=x)=\frac{1}{Z}\frac{1}{f(x)}e^{-\frac{1}{2(1-\rho^2)}\left(\left(\frac{x-\mu_X}{\sigma_X}\right)^2-2\rho\left(\frac{x-\mu_X}{\sigma_X}\right)\left(\frac{y-\mu_Y}{\sigma_Y}\right)+\left(\frac{y-\mu_Y}{\sigma_Y}\right)^2\right)}$$

- From this density, we can calculate conditional moments
  - $\mathbb{E}(Y|X=x)=\mu_y+\rho\frac{\sigma_y}{\sigma_x}(x-\mu_x)$ - linear in x
  - $Var(Y|X=x)=\sigma_y^2(1-\rho^2)$ - does not depend on x
  - In multivariate case, the logic is similar, but involves matrices…
- Conditional joint normal = OLS linear regression





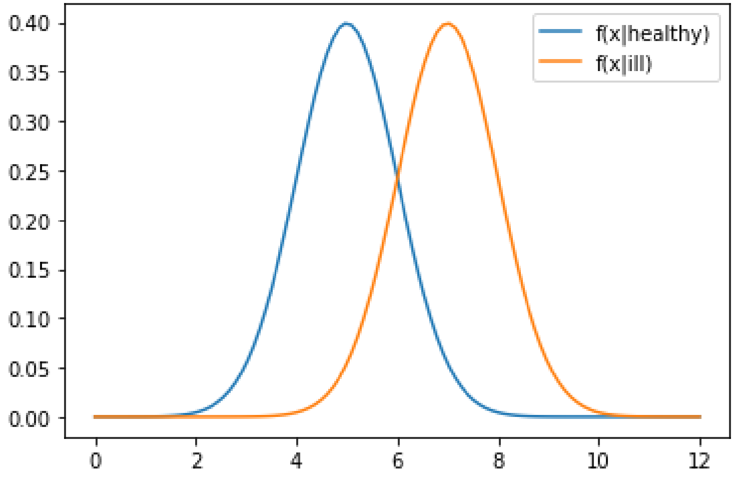
Conditional normal weight

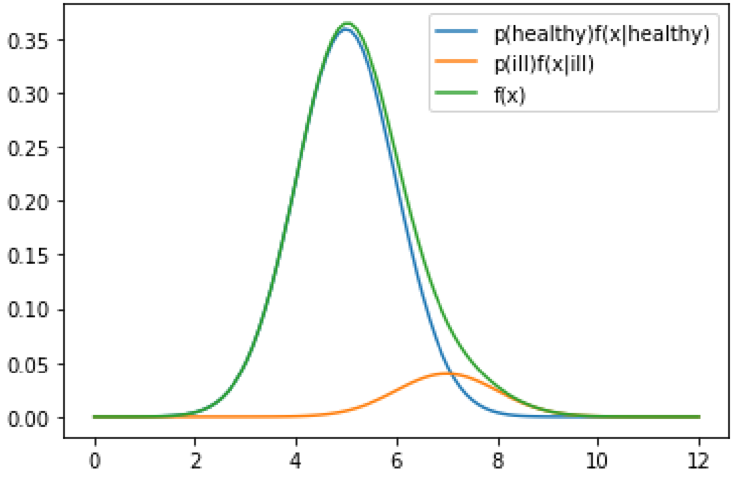
- Suppose that woman height and weight have joint normal distribution with means $(1.6\ m, 60\ kg)$, standard deviations $(0.08m, 10kg)$, and correlation 0.3
- What is the expected weight of a woman $1.76m$ tall?
  - 60 kg
  - 65 kg
  - 66 kg
  - 70 kg
- $\mathbb{E}(Y|X = x) = \mu_y + \rho \frac{\sigma_y}{\sigma_x}(x - \mu_x)$
- $60\ +\ 0.3 \times 10 \times \frac{1}{0.08}(1.76 - 1.6) = 66$

Separating the normal mixture

- Let a continuous X be normal conditionally on a discrete Y
  - E.g. X is a blood test result and Y is health (0 – healthy, 1 - ill).
  - For ill patients, $\mu = \mu_1$, for healthy $\mu = \mu_0$, and they have the same σ.
  - Prior probability of illness is p.
- Then we can calculate conditional probability by Bayes rule

$$P(y|X = x) = \frac{P(y)f(x|y)}{\sum_i P(y_i)f(x|y_i)}$$


conditional densities

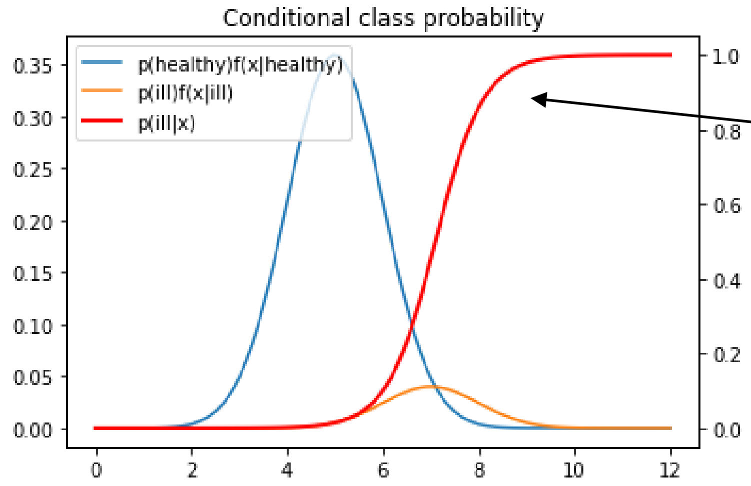

normalized conditional densities

Separating the normal mixture (2)

$$P(Y=1|X=x) = \frac{p\,\frac{1}{z}\exp\left(-\frac{(x-\mu_1)^2}{2\sigma^2}\right)}{p\,\frac{1}{z}\exp\left(-\frac{(x-\mu_1)^2}{2\sigma^2}\right)+(1-p)\,\frac{1}{z}\exp\left(-\frac{(x-\mu_0)^2}{2\sigma^2}\right)} =$$

$$= \frac{1}{1+\frac{1-p}{p}\exp\left(+\frac{(x-\mu_1)^2}{2\sigma^2}-\frac{(x-\mu_0)^2}{2\sigma^2}\right)} =$$

$$= \frac{1}{1+\exp\left(-\left(\log\left(\frac{p}{1-p}\right)+\frac{\mu_0^2-\mu_1^2}{2\sigma^2}+x\,\frac{\mu_1-\mu_0}{\sigma^2}\right)\right)} = \frac{1}{1+e^{-(a+bx)}}$$



Logistic curve

Separating the normal mixture (3)

- The same logic applies if there are m dimensions and k classes:

$$p(y_j|x) = \frac{e^{a_j + b_j x}}{\sum_i e^{a_i + b_i x}}$$

- We can try to fit such a model even if we don’t think that conditional distribution is normal
    - And it’s often a baseline classifier that’s very difficult to beat
    - Find the coefficients a, b and maybe c by gradient descent
    - The result is called *logistic regression*, but it’s not regression, it’s classification

Normal distribution: conclusions

- Joint normal distribution nicely describes relationships between normal variables
- Conditional expectation of join normal is linear
- Relative density of mixture of normal distributions is a logistic function

Linear models

Linear regression: statistical approach

- We want to model $P(Y|X)$, where X and Y are continuous
- One simple case: $Y|X \sim \mathcal{N}(a + Xb, c)$
- Log-likelihood is then $\sum_{i=1}^{n} \log\left(\frac{1}{\sqrt{2\pi c}} e^{-\frac{\left(y_i-(a+bx_i)\right)^2}{2c^2}}\right) =$

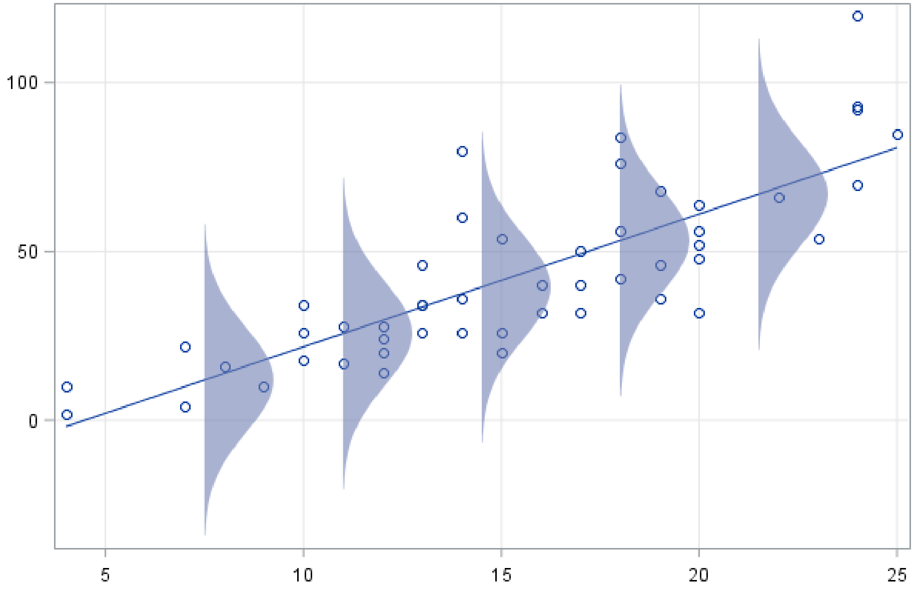
  $= const - \frac{1}{2c^2}\sum_{i=1}^{n}\left(y_i - (a + bx_i)\right)^2$

If $Y|X \sim \mathcal{N}$

then

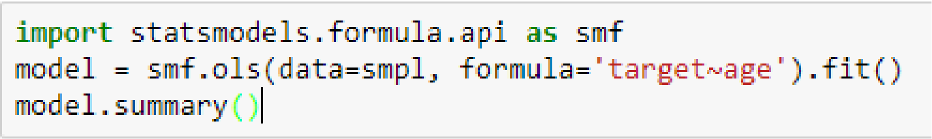
maximize likelihood =

= minimize mean squared error

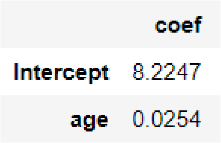


How to fit linear regression

- How to fit the model: maximize likelihood, because $Y|X \sim \mathcal{N}$
- There is a closed-form form solution: $\min_b (Y - Xb)^T (Y - Xb)$ is achieved at $\hat{b} = (X^T X)^{-1} X^T Y$
  - Here X is matrix of features and Y is vector of target values
  - $X^T X/n$ is covariance matrix of X (if it is centered)
  - $X^T Y/n$ is the vector of pairwise covariances of Y and X (if they are centered)
  - In practice, instead of centering we can add column of 1s to X, to create an intercept

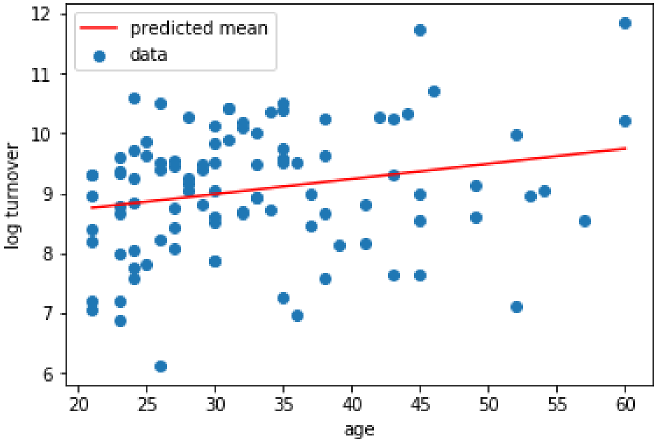


```
import statsmodels.formula.api as smf
model = smf.ols(data=smpl, formula='target~age').fit()
model.summary()
```



	coef
Intercept	8.2247
age	0.0254

$$\mathbb{E}(Y|X) \approx 8.22 + 0.025\,X$$



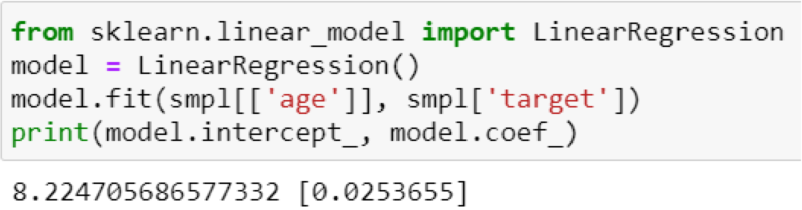
Python modules for regression

scikit-learn

- Lots of various machine learning models
- Few tools for analysis of models
- Focused on prediction

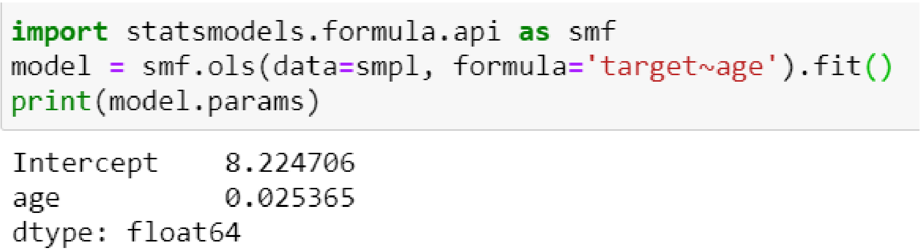
statsmodels

- A few linear and generalized linear models
- Lots of tools for statistical inference
- Focused on interpretation



```
from sklearn.linear_model import LinearRegression
model = LinearRegression()
model.fit(smpl[['age']], smpl['target'])
print(model.intercept_, model.coef_)
```

```
8.224705686577332 [0.0253655]
```

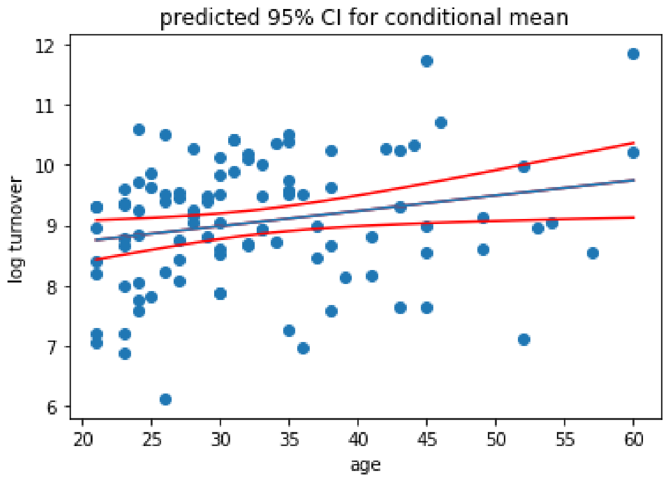


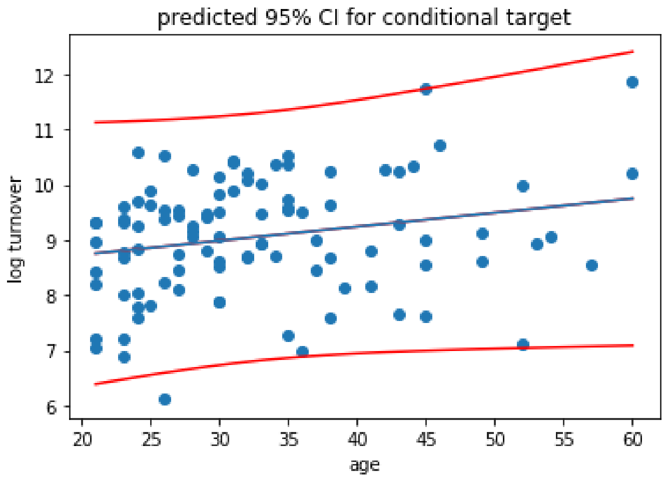
```
import statsmodels.formula.api as smf
model = smf.ols(data=smpl, formula='target~age').fit()
print(model.params)
```

```
Intercept    8.224706
age          0.025365
dtype: float64
```

Prediction interval

- If $Y \sim \mathcal{N}(a + Xb, \sigma^2)$, we cannot predict Y certainly, because $\sigma^2 > 0$
- If we estimate $\hat{a}$ and $\hat{b}$ from sample, they are random variables themselves, so even $\mathbb{E}(Y|X)$ is uncertain
- But with can construct a confidence interval for it, because $\widehat{\mathrm{Y}}(X) = \hat{a} + X\hat{b}$ is asymptotically normal (given X)
  - it will be expanding in X, because $\hat{b}$ is uncertain
  - $Var\left(\hat{a} + X\hat{b}\right) = Var(\hat{a}) + 2XCov\left(\hat{a}, \hat{b}\right) + X^2 Var\left(\hat{b}\right)$
- Confidence interval for Y is by $\hat{\sigma}$ wider than for $\mathbb{E}(Y|X)$


predicted 95% CI for conditional mean

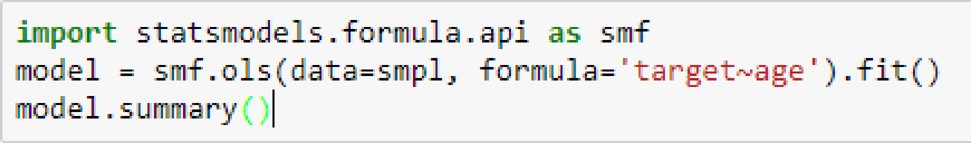

predicted 95% CI for conditional target

Prediction interval

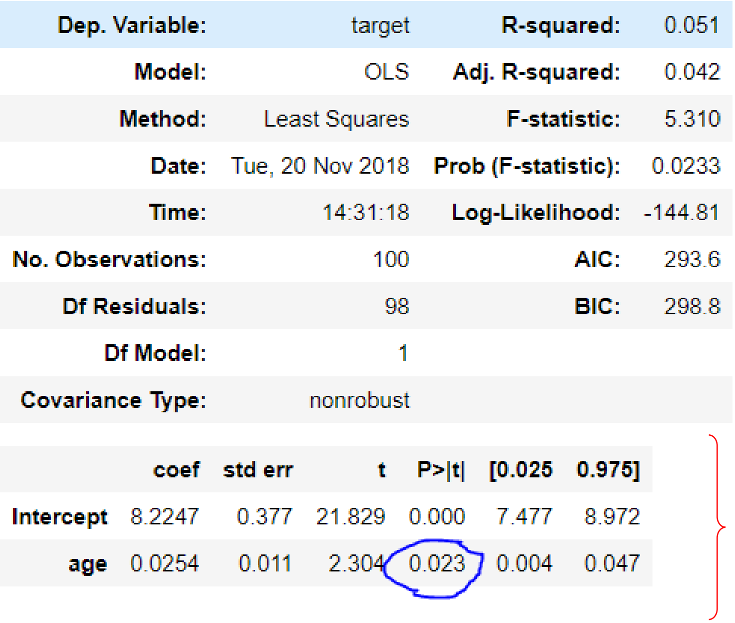
- After regressing weight (kg) on height (cm), you get estimated regression line $\mathbb{E}(w|h) \approx -4 + 0.4h$ with estimated variance $Var(w|h) \approx 5^2$
- In what interval with 95% probability could be the weight of a woman 150 cm tall?
    - $[51, 61]$
    - $[46,66]$
    - $[40,70]$

Testing hypotheses with linear models

- Example: does credit card turnover depend on age?
    - Fit a linear model and test for $b = 0$



```
import statsmodels.formula.api as smf
model = smf.ols(data=smpl, formula='target~age').fit()
model.summary()
```

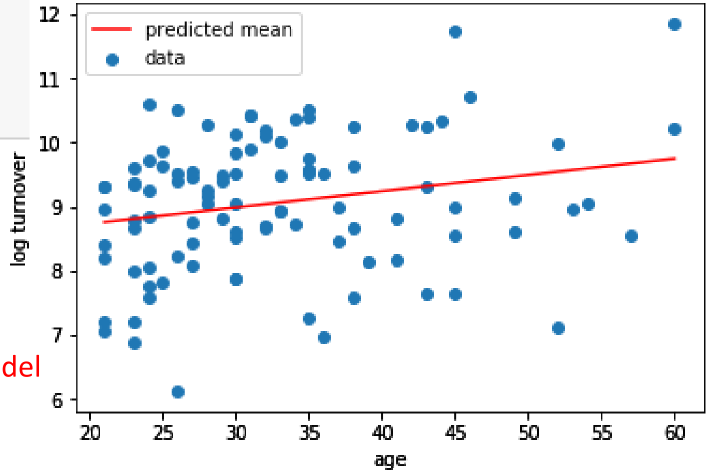


Dep. Variable:	target	R-squared:	0.051
Model:	OLS	Adj. R-squared:	0.042
Method:	Least Squares	F-statistic:	5.310
Date:	Tue, 20 Nov 2018	Prob (F-statistic):	0.0233
Time:	14:31:18	Log-Likelihood:	-144.81
No. Observations:	100	AIC:	293.6
Df Residuals:	98	BIC:	298.8
Df Model:	1		
Covariance Type:	nonrobust		

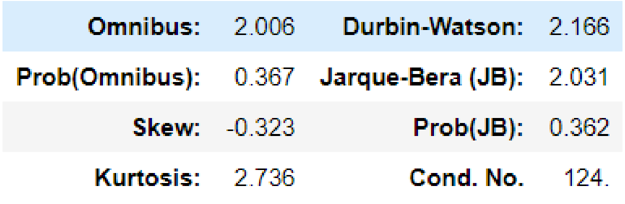
Whole model

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	8.2247	0.377	21.829	0.000	7.477	8.972
age	0.0254	0.011	2.304	0.023	0.004	0.047

Coefficients



Residuals



Omnibus:	2.006	Durbin-Watson:	2.166
Prob(Omnibus):	0.367	Jarque-Bera (JB):	2.031
Skew:	-0.323	Prob(JB):	0.362
Kurtosis:	2.736	Cond. No.	124.

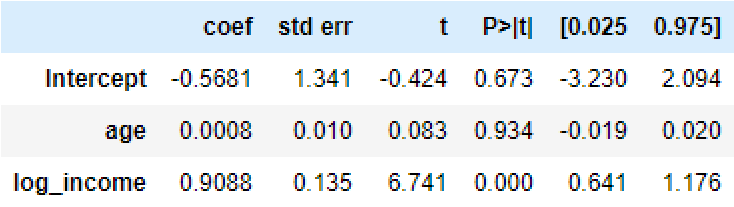
Testing hypotheses with linear models

- Coefficients in OLS are sample estimates: $\hat{b} = (X^T X)^{-1} X^T Y$
  - If we treat only Y as random (=condition everything on X), they are just a linear function of Y, and therefore have normal distribution with known mean and variance
  - $\mathbb{E}(\hat{b}) = (X^T X)^{-1} X^T \mathbb{E} Y = (X^T X)^{-1} X^T X b = b$ -> they are unbiased!
  - $COV(\hat{b}) = COV(M \times Y) = M \times COV(Y) \times M^T = \sigma^2 M M^T = \sigma^2 (X^T X)^{-1}$, where $M = (X^T X)^{-1} X^T$, and COV means covariance matrix
- Therefore, we can construct a t-statistic $t_j = \frac{\hat{b}_j - b_j}{\hat{\sigma}_{\widehat{b_j}}}$ to test the hypothesis that the population value of jth coefficient is really b_j
  - It will be Student with $n - k$ degrees of freedom, where n is sample size and k is number of estimated coefficients (size of $\hat{b}$).
  - Testing that $b_j = 0$ means testing that Y depends on X_j

Tests of “causal” dependence

- Our intuition tells that age affects turnover indirectly
  - We see that older people on average spend more, but it’s strange. Why so?
  - Old people earn more than young people
  - Richer people can spend more with their credit card
  - But does age affect turnover *if income is kept constant*?
- Solution: use a multivariate linear regression
  - $\log(turnover) \sim \mathcal{N}(a + age \times b_1 + \log(income) \times b_2\,, \sigma^2)$
  - Test for $b_1 = 0$

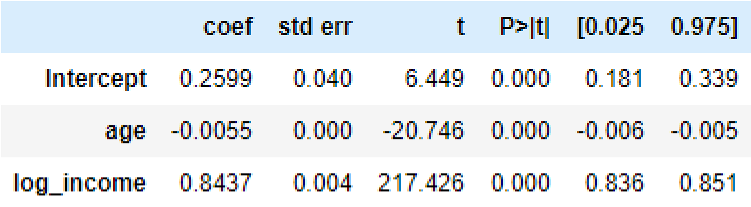
Small sample (100 obs.)



	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	-0.5681	1.341	-0.424	0.673	-3.230	2.094
age	0.0008	0.010	0.083	0.934	-0.019	0.020
log_income	0.9088	0.135	6.741	0.000	0.641	1.176

Impact of age is not significantly different from 0

Large sample (100K obs.)

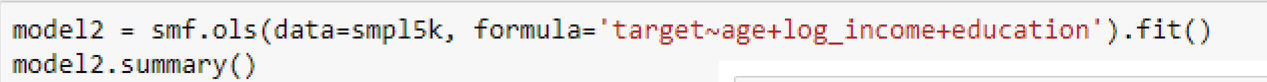


	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	0.2599	0.040	6.449	0.000	0.181	0.339
age	-0.0055	0.000	-20.746	0.000	-0.006	-0.005
log_income	0.8437	0.004	217.426	0.000	0.836	0.851

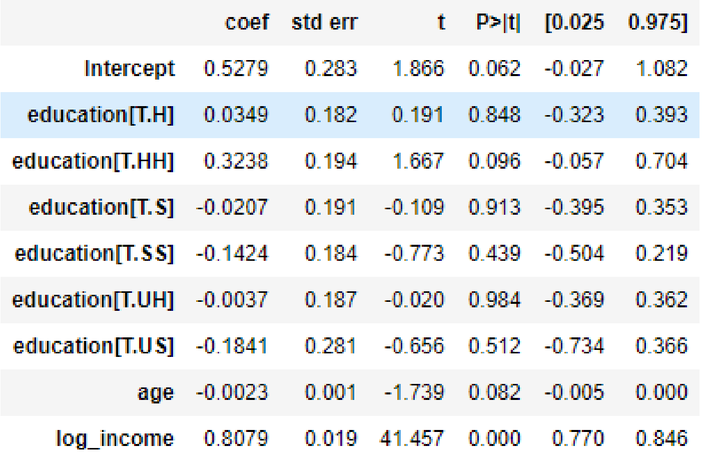
Impact of age is significantly different from 0 and negative!
Older people spend less than young people with same income (on average)

One-hot encoding and F-test

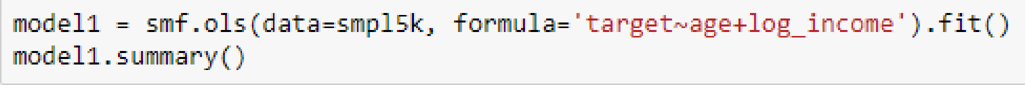
- Okay, we have dependency on numeric *age* and *income*, what about categorical *education*?
    - Just convert it to numeric!
- The simplest way: create indicator (aka one-hot, aka dummy) variables
    - Such as *education.H = 1 iff education = 'H' else 0*
    - Create them for all possible values except one, to avoid linear dependency (otherwise, $X^T X$ will not be invertible). This one must be 1 when all others are 0.



```
model2 = smf.ols(data=smpl5k, formula='target~age+log_income+education').fit()
model2.summary()
```

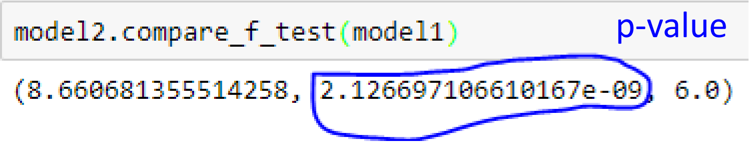


	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	0.5279	0.283	1.866	0.062	-0.027	1.082
education[T.H]	0.0349	0.182	0.191	0.848	-0.323	0.393
education[T.HH]	0.3238	0.194	1.667	0.096	-0.057	0.704
education[T.S]	-0.0207	0.191	-0.109	0.913	-0.395	0.353
education[T.SS]	-0.1424	0.184	-0.773	0.439	-0.504	0.219
education[T.UH]	-0.0037	0.187	-0.020	0.984	-0.369	0.362
education[T.US]	-0.1841	0.281	-0.656	0.512	-0.734	0.366
age	-0.0023	0.001	-1.739	0.082	-0.005	0.000
log_income	0.8079	0.019	41.457	0.000	0.770	0.846



```
model1 = smf.ols(data=smpl5k, formula='target~age+log_income').fit()
model1.summary()
```

- Is *model2* no better than *model1*?
- If the true coefs for education are all 0, then $\frac{(\widehat{\sigma_1^2} d1 - \widehat{\sigma_2^2} d2)/(d1-d2)}{\widehat{\sigma_2^2}} \sim F_{d1-d2,d1}$
- This hypothesis can be rejected



```
model2.compare_f_test(model1)
```

p-value

```
(8.660681355514258, 2.126697106610167e-09, 6.0)
```

Nonlinear linear regression

- Which formulas of dependence between Y and X cannot be converted to linear by change of variables?
  1. $Y = aX^2 + bX + c$
  2. $Y = \exp(a + bX)$
  3. $Y = \begin{cases} a, if\ X \leq 0 \\ a + bX, if\ x > 0 \end{cases}$
  4. $Y = \exp(aX) + \exp(bX^2)$
- Only the last one
  1. $Y = a[X^2] + bX + c$
  2. $\log Y = a + bX$
  3. $Y = a + b[X, if\ X > 0\ else\ 0]$
  4. ???

Standard linear regression: summary

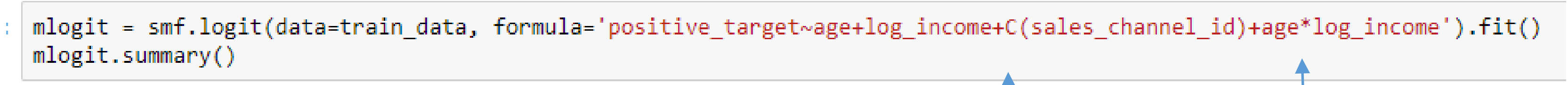
- Rely on the assumption $P(Y|X) \sim \mathcal{N}$
  - If many different Xs affect Y, it's often approximately true
- Rely on the assumption that $\mathbb{E}(Y|X)$ is linear in X
  - It makes model interpretation very straightforward
  - It can be extended to non-linearity (e.g. polynomial features)
- Some more assumptions:
  - Different observations are independent
  - $Var(Y|X) = const$
  - There are extensions for some cases when it is not true
- But what to do if $p(Y|X)$ is far from normal?
  - E.g. if Y is strictly positive?
  - E.g. if Y is discrete?
  - We can still use OLS, but prediction/confidence intervals and standard tests will be incorrect, and prediction quality might be worse than possible

Generalized linear models

Generalized linear models

- If we know the family (shape) of $P(Y|X)$, we can estimate its parameters (with maximum likelihood) just as well
- Sometimes, it requires only a wrapper around linear function
  - The model: $\hat{Y} = f(a + Xb)$
  - f is called activation function
- This assumption can work for many different distributions:
  - E.g. for binary Y we can model $P(Y = 1|X) = \mathbb{E}(Y|X) = \frac{1}{1+e^{-(a+Xb)}}$ (logistic model)
  - E.g. for Poisson ($P(Y = k|X) = \frac{\lambda^k}{k!}e^{-\lambda}$) we can model $\lambda = \exp(a + Xb)$
- With generalized linear models, we can do the same kind of coefficient significance analysis as with linear ones
- Distributions can be derived from (asymptotic) properties of maximum likelihood

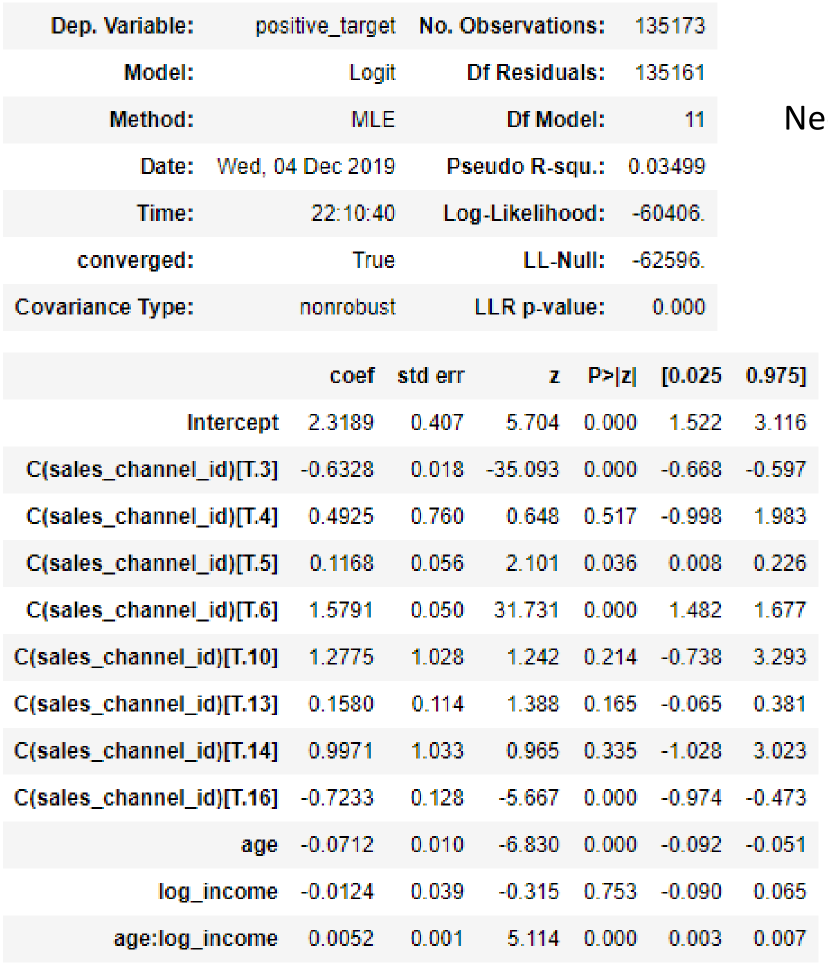
Example of logistic “regression”



```
mlogit = smf.logit(data=train_data, formula='positive_target~age+log_income+C(sales_channel_id)+age*log_income').fit()
mlogit.summary()
```

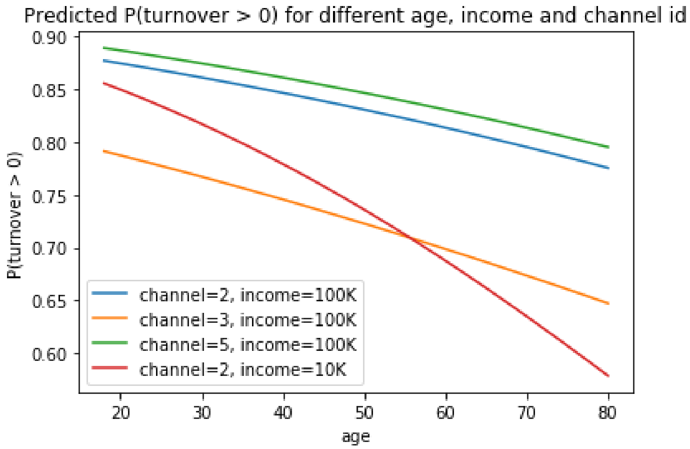
Need categorical encoding

“interaction” between variables



Dep. Variable:	positive_target	**No. Observations:**	135173
Model:	Logit	**Df Residuals:**	135161
Method:	MLE	**Df Model:**	11
Date:	Wed, 04 Dec 2019	**Pseudo R-squ.:**	0.03499
Time:	22:10:40	**Log-Likelihood:**	-60406.
converged:	True	**LL-Null:**	-62596.
Covariance Type:	nonrobust	**LLR p-value:**	0.000

	coef	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
Intercept	2.3189	0.407	5.704	0.000	1.522	3.116
C(sales_channel_id)[T.3]	-0.6328	0.018	-35.093	0.000	-0.668	-0.597
C(sales_channel_id)[T.4]	0.4925	0.760	0.648	0.517	-0.998	1.983
C(sales_channel_id)[T.5]	0.1168	0.056	2.101	0.036	0.008	0.226
C(sales_channel_id)[T.6]	1.5791	0.050	31.731	0.000	1.482	1.677
C(sales_channel_id)[T.10]	1.2775	1.028	1.242	0.214	-0.738	3.293
C(sales_channel_id)[T.13]	0.1580	0.114	1.388	0.165	-0.065	0.381
C(sales_channel_id)[T.14]	0.9971	1.033	0.965	0.335	-1.028	3.023
C(sales_channel_id)[T.16]	-0.7233	0.128	-5.667	0.000	-0.974	-0.473
age	-0.0712	0.010	-6.830	0.000	-0.092	-0.051
log_income	-0.0124	0.039	-0.315	0.753	-0.090	0.065
age:log_income	0.0052	0.001	5.114	0.000	0.003	0.007



- For different channels, predictions are “parallel”
- Because of interaction, changing income changes the slope for age (and vice versa)

Prediction with logistic regression

- You have trained a logistic regression to predict probability that a bank client does not return a loan by two features:
  - X_1 is number of open loans
  - X_2 is 1 if the client has a current account in our bank, otherwise 0
- The learned parameters are these: intercept is -1, coefficients are 0.75 and -0.25
- What is the probability of not returning a loan for a client with 3 open loans and a current account in our bank?
  - $\sim 25\%$
  - $\sim 50\%$
  - $\sim 75\%$
- $P(Y=1) = \frac{1}{1+e^{-(-1+0.75X_1-0.25X_2)}} = \frac{1}{1+e^{-1}} \approx 73\%$

How to make a not-so-bad machine learning model

- Construct a good generalized linear model
  - Find good “regressors” (features) X
  - Choose a correct shape for $P(Y|X)$ (⇒ activation function f)
  - Maybe, this will already be enough
- Why one needs deeper models (e.g. gradient boosting or neural networks)?
  - To extract highly nonlinear features from X
  - To model non-obvious interactions between features
  - To transform the final prediction in a flexible way
- Linear models are usually good baselines

Loss vs likelihood for regression

Name	Formula	Minimized by	Likelihood for
Mean squared error (L2)	$\frac{1}{n}\sum_{i=1}^{n}\left(Y-\hat{Y}\right)^{2}$	Arithmetic mean	Normal distribution
Mean absolute error (L1)	$\frac{1}{n}\sum_{i=1}^{n}\left\|Y-\hat{Y}\right\|$	Median	Laplace distribution (symmetric exponential)
Mean squared logarithmic error	$\frac{1}{n}\sum_{i=1}^{n}\left(\log\frac{Y+\delta}{\hat{Y}+\delta}\right)^{2}$	Geometric mean	Lognormal distribution
Mean average percentage error	$\frac{1}{n}\sum_{i=1}^{n}\left\|\frac{Y-\hat{Y}}{Y}\right\|$	No closed form minimizer	It cannot exist

Loss and metrics for classification

- For any discrete distribution, log likelihood is the same (a.k.a. cross-entropy)

$$LL = \sum_{i=1} \sum_{c \in C} [y_i = c] \log \hat{p}(y_i = c)$$

- Scores for categories $(z_1, \dots, z_k)$ from any model can be converted to probabilities by softmax transformation $P(Y = y) = \frac{e^{Z_y}}{\sum_{i=1}^{k} e^{Z_i}}$
- More interpretable metrics are calculated not from probabilities, but usually from the confusion matrix
  - But they are generally not used for estimating model parameters, only for evaluation

Accuracy	$\frac{TP + TN}{TP + TN + FP + FN}$
Precision	$p = \frac{TP}{TP + FP}$
Recall	$r = \frac{TP}{TP + FN}$

F-score	$f = \frac{2pr}{p + r}$

Relevance of a loss function

- You want to predict the probability P_i that form is a "good" one. "Goods" are denoted by 1, "bads" by 0.
- Your manager proposes to minimize mean absolute error: $\frac{1}{n}\sum_{i=1}^{n}|Y_i - P_i|$
- Will such a model predict meaningful probabilities?
    - Yes, why not
    - No, it is broken
- MAE loss will be minimized if all predictions are 0 or 1, so the predicted numbers will not reflect probability

Summary: probability in ML

- We build models mostly because we want to predict $Y|X$
- Y is random ⇒ our prediction is a parameter of its conditional distribution (e.g. mean or some quantile)
- (Generalized) linear models are easy to train and allow interpreting and testing coefficients
- To train a model = to estimate its parameters
  - Maximum likelihood can often do it for us
  - It is usually equivalent to minimizing a loss function
- We might need to choose a distribution family for $Y|X$
  - Sometimes, statistical tests can tell if it is chosen wrongly
- We can evaluate how good a prediction is by looking at different metrics (which are in fact sample statistics)

What to do next

- One more home assignment in pairs
    - Due in 9 days
- The summary assignment
    - Individually
    - Will be in a week
- Recommended reading:
    - *https://seeing-theory.brown.edu/regression-analysis/*
    - *A Modern Introduction to Probability and Statistics* by F.M. Dekking - chapters 17.4, 22, 27.3