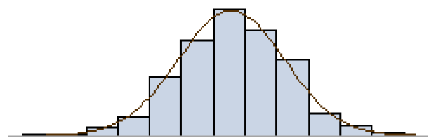
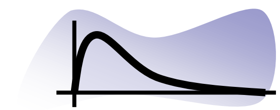
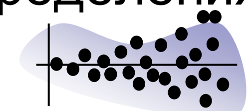
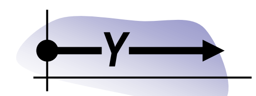
Лекция 9:
Обобщенные линейные модели, логистическая регрессия

Важное предположение линейной регрессии

- Нормальное распределение ошибки с константной дисперсией:



- Часто возникающие «особенности»:
  - Несимметричные распределения отклика 
  - Гетероскедастичность 
  - Ограниченная область определения отклика 

- Что делать?
  - Явно преобразовывать отклик: $E(\,g(y)\,|\,x\,)$

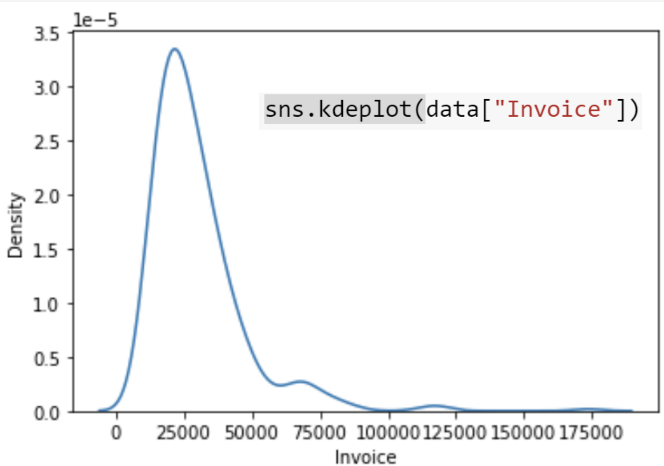
НО, в общем случае: $g^{-1}(E\,(g(y)\,|\,x)) \neq E\,(y\,|\,x\,)$

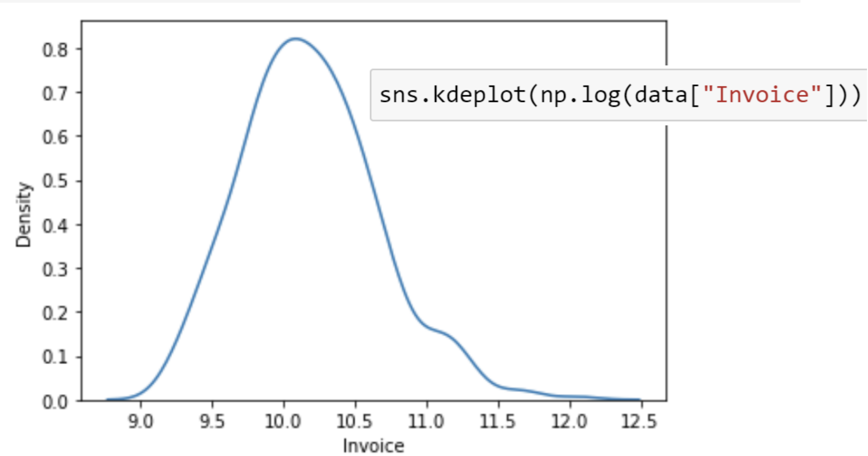
  - Использовать функцию связи: $g(E(y\,|\,x\,))$

Пример

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
data=pd.read_csv("cars0.csv",delimiter=",")
data.head()
```

	Make	Model	Type	Origin	DriveTrain	MSRP	Invoice	EngineSize	Cylinders	Horsepower	MPG_City	MPG_Highway	Weight	Wheelbase	Length
0	Acura	MDX	SUV	Asia	All	36945.0	33337.0	3.5	6.0	265	17	23	4451	106	189
1	Acura	RSX Type S 2dr	Sedan	Asia	Front	23820.0	21761.0	2.0	4.0	200	24	31	2778	101	172
2	Acura	TSX 4dr	Sedan	Asia	Front	26990.0	24647.0	2.4	4.0	200	22	29	3230	105	183
3	Acura	TL 4dr	Sedan	Asia	Front	33195.0	30299.0	3.2	6.0	270	20	28	3575	108	186
4	Acura	3.5 RL 4dr	Sedan	Asia	Front	43755.0	39014.0	3.5	6.0	225	18	24	3880	115	197





Пример (МНК) – все плохо

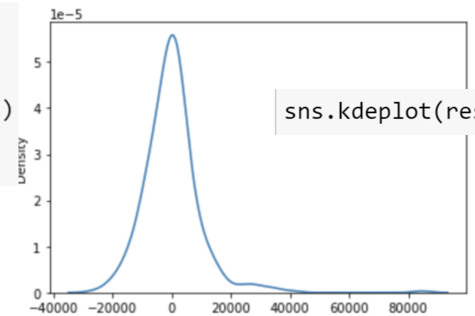
```python
import statsmodels.api as sm

ols = sm.OLS(endog=data['Invoice'],
             exog=sm.add_constant(data[['Weight','Length','Horsepower']]), data=data)
results=ols.fit()
results.summary()
```

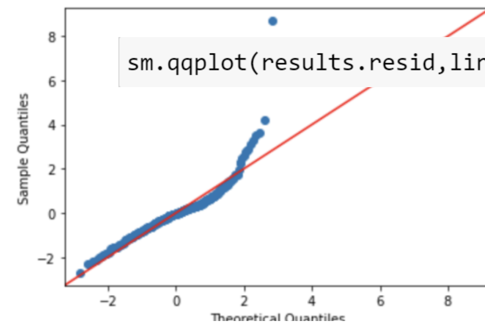
Dep. Variable:	Invoice	**R-squared:**	0.704
Model:	OLS	**Adj. R-squared:**	0.702
Method:	Least Squares	**F-statistic:**	336.3
Date:	Wed, 01 Nov 2023	**Prob (F-statistic):**	1.06e-111
Time:	01:43:01	**Log-Likelihood:**	-4531.2
No. Observations:	428	**AIC:**	9070.
Df Residuals:	424	**BIC:**	9087.
Df Model:	3		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	2.25e+04	6682.648	3.367	0.001	9362.779	3.56e+04
Weight	0.0255	1.015	0.025	0.980	-1.970	2.021
Length	-213.1397	45.054	-4.731	0.000	-301.696	-124.583
Horsepower	218.3874	8.400	26.000	0.000	201.877	234.898

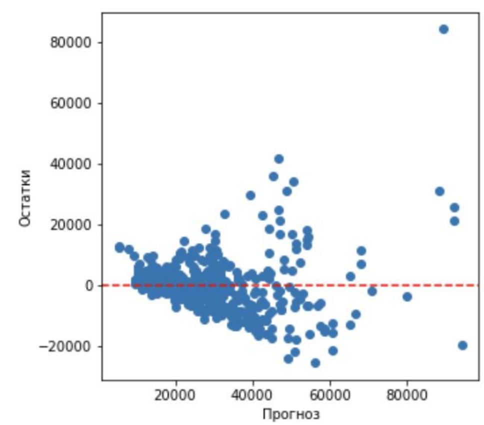
```python
fig, ax = plt.subplots(figsize=(5, 5))
ax.scatter(results.predict(sm.add_constant(data[['Weight','Length','Horsepower']])),
        results.resid)
ax.set_ylabel('Остатки')
ax.set_xlabel('Прогноз')
plt.axhline(y = 0, color = 'r', linestyle = '--')
```




```python
sns.kdeplot(results.resid)
```

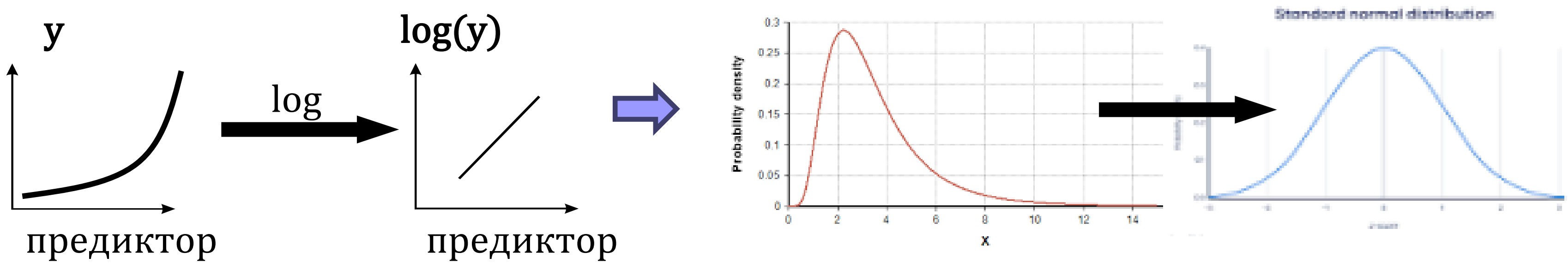



```python
sm.qqplot(results.resid,line="45",fit=True)
```



Преобразование отклика и логнормальная регрессия

- Распределение отклика y логнормальное, тогда распределение с.в. $\log(y)$ – нормальное: $\log(y) \sim N(\mu, \sigma^2)$



- Связь моментов исходной с.в. y и $\log(y)$:

$$E(y) = \exp(\mu + \frac{\sigma^2}{2}),\ D(y) = \left(e^{\sigma^2} - 1\right)(E(y))^2$$

- Это значит, что можно построить МНК регрессию для прогнозирования $\log(y) = w^T x$ и получить исходный отклик:

$$\mu = E(\log(y)|x) = w^T x \Rightarrow E(y|x) = \exp\left(w^T x + \frac{\sigma^2}{2}\right)$$

- Откуда брать σ^2? Можно взять оценку $\sigma^2 \approx MSE_{val}$, желательно на *валидационном* наборе

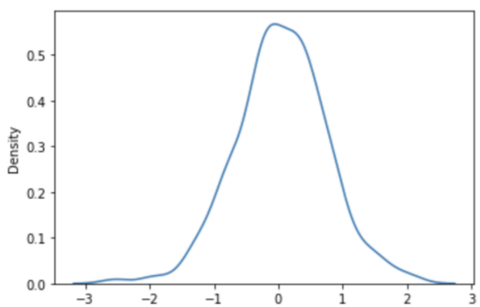
Пример (логнормальная регрессия) – лучше

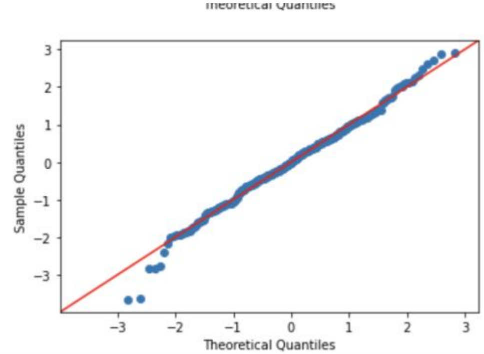
```python
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error

X_train, X_test, y_train, y_test = train_test_split(
    sm.add_constant(data[['Weight','Length','Horsepower']]),
    np.log(data['Invoice']),test_size=0.3)
lnr = sm.OLS(endog=y_train, exog=X_train)
lnr_results=lnr.fit()
mse=mean_squared_error(y_test, lnr_results.predict(X_test))
lnr_results.summary()
```

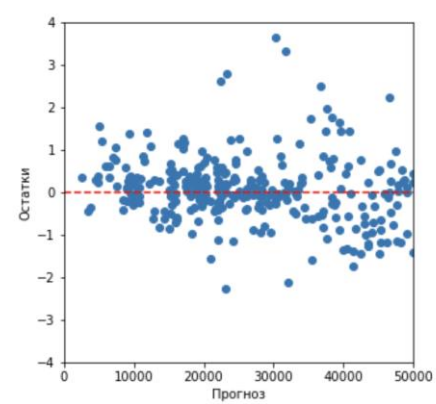
Dep. Variable:	Invoice	**R-squared:**	0.768
Model:	OLS	**Adj. R-squared:**	0.766
Method:	Least Squares	**F-statistic:**	325.9
Date:	Wed, 01 Nov 2023	**Prob (F-statistic):**	2.68e-93
Time:	01:54:31	**Log-Likelihood:**	9.5115
No. Observations:	299	**AIC:**	-11.02
Df Residuals:	295	**BIC:**	3.779
Df Model:	3		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	9.6851	0.209	46.330	0.000	9.274	10.097
Weight	0.0001	2.83e-05	4.264	0.000	6.5e-05	0.000
Length	-0.0060	0.001	-4.324	0.000	-0.009	-0.003
Horsepower	0.0055	0.000	22.407	0.000	0.005	0.006






```python
fig, ax = plt.subplots(figsize=(5, 5))
ax.scatter(np.exp(mse/2+lnr_results.predict(
    sm.add_constant(data[['Weight','Length','Horsepower']]))),
    results.resid_pearson)
plt.xlim(0, 50000)
plt.ylim(-4, 4)
ax.set_ylabel('Остатки')
ax.set_xlabel('Прогноз')
plt.axhline(y = 0, color = 'r', linestyle = '--')
```



Обобщенная линейная модель

Функция связи → $\boldsymbol{g}\big(\boldsymbol{E}(\boldsymbol{y}|\boldsymbol{x})\big) = w_0 + w_1 x_1 \dots + w_k x_p = \langle x, w \rangle$

- Распределение отклика принадлежит экспоненциальному семейству $y_i \sim \mathrm{Exp}(\theta, \phi)$, где плотность определена как:
$$p(y|\theta, \phi) = exp\left(\frac{y\theta - c(\theta)}{\phi} + h(y, \phi)\right)$$
- Математическое ожидание с.в. y зависит только от θ через некоторую монотонную *функцию связи* $g(.)$ (link function) как: $\mu = E(y) = \mathrm{c}'(\theta) \Rightarrow \theta = g(\mu) = [c']^{-1}(\mu)$
- Дисперсия с.в. y есть функция от среднего: $D(y) = \phi c''(\theta)$
- Распределение отклика наблюдений может подсказать какую функцию связи и функцию потерь следует выбрать

Важные частные случаи

- Линейная регрессия: $p(y|\mu,\sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{(y-\mu)^2}{2\sigma^2}\right)$
- Логистическая регрессия: $p(y|\mu) = \mu^y (1-\mu)^{1-y}$
- Пуассоновская регрессия: $p(y|\lambda) = \frac{e^{-\lambda}\lambda^y}{y!}$
- Гамма регрессия: $p(y|\nu,\mu) = \frac{1}{\Gamma(\nu)y}\left(\frac{y\nu}{\mu}\right)^{\nu} e^{-\frac{y\nu}{\mu}}$

Регрессия	Отклик	Параметр θ (среднее)	Параметр ϕ	Дисперсия	Каноническая функция связи
Линейная	непрерывный неограниченный	μ	σ	σ^2	тождество g(μ)= μ
Логистическая	бинарный категориальный	μ	1	(1- μ) μ	логит g(μ)= log(μ/(1- μ))
Пуассоновская	«Счетчик» - дискретный положительный	λ	1	λ	логарифм g(μ)= log(μ)
Гамма	непрерывный положительный	μ	ν	μ/ ν^2	обратная g(μ)= 1/μ

Примеры вывода функции связи

- Суть: приведение распределения к каноническому виду $\mathrm{Exp}(\theta, \phi)$
- Линейная регрессия (нормальное распределение):

$$p(y|\mu,\sigma) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(\frac{(y-\mu)^2}{2\sigma^2}\right) = \exp\left(\frac{y\mu - \frac{1}{2}\mu^2}{\sigma^2} - \frac{y^2}{2\sigma^2} - \frac{1}{2}\log(2\pi\sigma^2)\right)$$

$$\theta = g(\mu) = \mu,\ c(\theta) = \tfrac{1}{2}\mu^2 = \tfrac{1}{2}\theta^2$$

- Пуассоновская регрессия (распределение Пуассона):

$$p(y|\lambda) = \frac{e^{-\lambda}\lambda^y}{y!} = \exp\left(\frac{y\log(\lambda) - \lambda}{1} - \log(y)!\right)$$

$$\theta = g(\lambda) = \log(\lambda),\ c(\theta) = \lambda = e^{\theta}$$

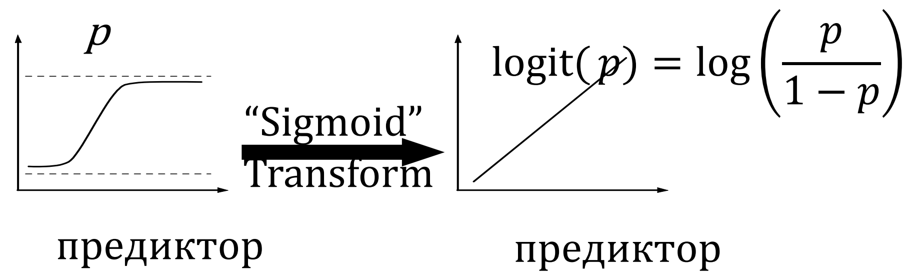
- Логистическая регрессия (распределение Бернулли):

$$p(y|\mu) = \mu^y(1-\mu)^{1-y} = \exp\left(y\log\left(\frac{\mu}{1-\mu}\right) - \log(1-\mu)\right)$$

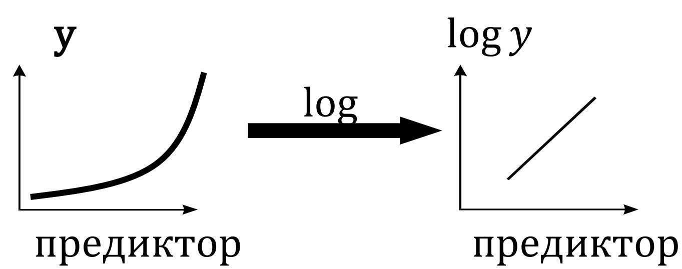
$$\theta = g(\mu) = \log\left(\frac{\mu}{1-\mu}\right),\ c(\theta) = -\log(1-\mu) = \log(1+e^{\theta})$$

Не все так однозначно

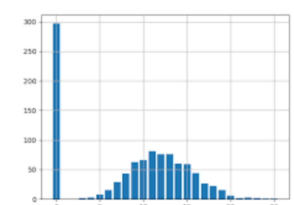
- На практике часто используют неканонические функции связи
- Например, для логистической регрессии:
  - Каноническая logit
  - probit: $\frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\mu} z^2 dz$
  - log-log: $\log(-\log(1-\mu))$



- Для гамма регрессии:
  - Каноническая обратная
  - log, тождественная и др.



- Для «счетчиков»:
  - Чрезмерная дисперсия - может не выполнятся условие $E(y) = D(y) = \lambda$ и тогда используют отрицательно биномиальное распределение, где дисперсия моделируется как функция от среднего и квадрата среднего
  - Может быть «смесь» счетчиков
  - “zero inflated” – смесь 0 и пуассоновского счетчика



Пример гамма регрессии

```
gamma_model = sm.GLM(data['Invoice'],
    sm.add_constant(data[['Weight','Length','Horsepower']]),
    family=sm.families.Gamma())
gamma_results = gamma_model.fit()
gamma_results.summary()
```

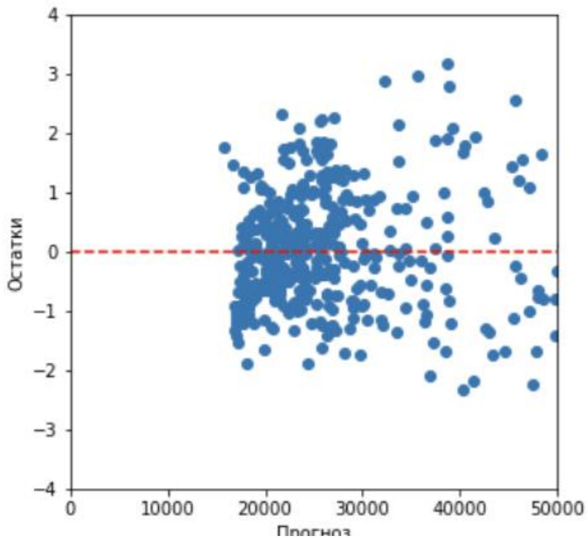
Dep. Variable:	Invoice	**No. Observations:**	428
Model:	GLM	**Df Residuals:**	424
Model Family:	Gamma	**Df Model:**	3
Link Function:	inverse_power	**Scale:**	0.11306
Method:	IRLS	**Log-Likelihood:**	-5686.6
Date:	Wed, 01 Nov 2023	**Deviance:**	310.53
Time:	02:03:07	**Pearson chi2:**	47.9
No. Iterations:	8	**Pseudo R-squ. (CS):**	-74.85
Covariance Type:	nonrobust		

	coef	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
const	4.818e-05	6.76e-06	7.124	0.000	3.49e-05	6.14e-05
Weight	-6.088e-09	5.99e-10	-10.164	0.000	-7.26e-09	-4.91e-09
Length	2.391e-07	4.43e-08	5.402	0.000	1.52e-07	3.26e-07
Horsepower	-1.467e-07	2.88e-09	-50.999	0.000	-1.52e-07	-1.41e-07

Как считать?
Ответ позже

Статистика Уальда (аналогично Стьюденту для МНК)

```
fig, ax = plt.subplots(figsize=(5, 5))
ax.scatter(gamma_results.predict(data[['Weight','Length','Horsepower']]),
           results.resid_pearson)
plt.xlim(0, 50000)
plt.ylim(-4, 4)
ax.set_ylabel('Остатки')
ax.set_xlabel('Прогноз')
plt.axhline(y = 0, color = 'r', linestyle = '--')
```



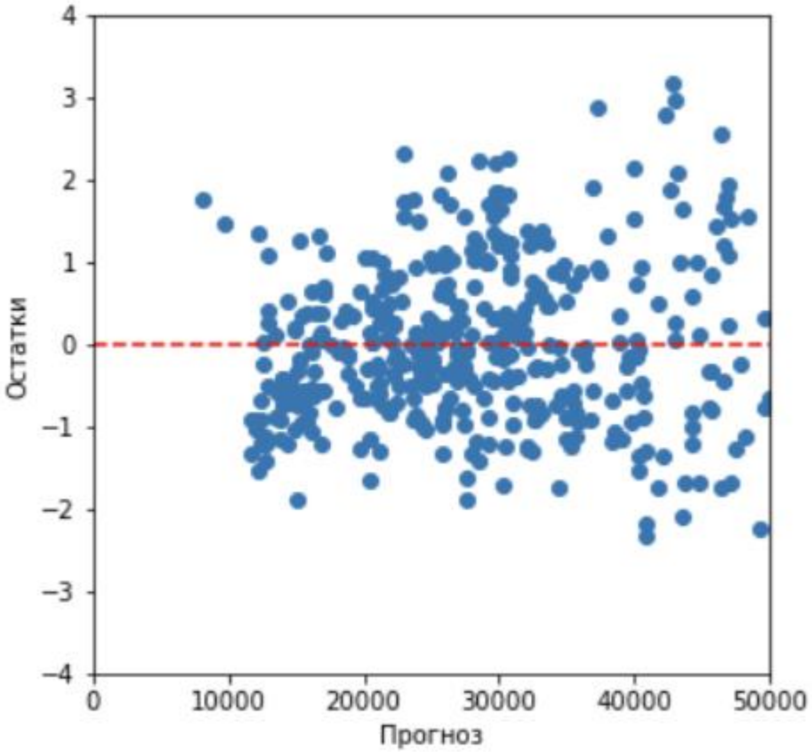
гетероскедастичность?

Пример гамма регрессии с неканонической тождественной функцией связи

```
gamma_model = sm.GLM(data['Invoice'],
    sm.add_constant(data[['Weight','Length','Horsepower']]),
    family=sm.families.Gamma(sm.families.links.identity()))
gamma_results = gamma_model.fit()
gamma_results.summary()
```

Dep. Variable:	Invoice	**No. Observations:**	428
Model:	GLM	**Df Residuals:**	424
Model Family:	Gamma	**Df Model:**	3
Link Function:	identity	**Scale:**	0.066351
Method:	IRLS	**Log-Likelihood:**	-4359.6
Date:	Wed, 01 Nov 2023	**Deviance:**	24.571
Time:	02:06:57	**Pearson chi2:**	28.1
No. Iterations:	19	**Pseudo R-squ. (CS):**	0.9438
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
const	2.359e+04	4377.250	5.389	0.000	1.5e+04	3.22e+04
Weight	2.8085	0.849	3.307	0.001	1.144	4.473
Length	-209.3271	31.087	-6.734	0.000	-270.256	-148.398
Horsepower	161.8258	8.531	18.969	0.000	145.105	178.547

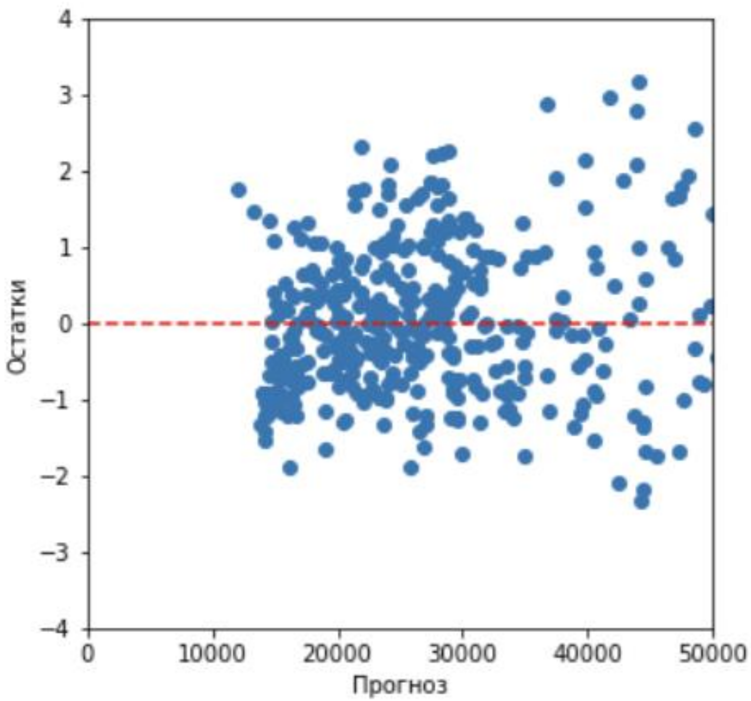


Пример гамма регрессии с неканонической функцией связи log

```
gamma_model = sm.GLM(data['Invoice'],
    sm.add_constant(data[['Weight','Length','Horsepower']]),
    family=sm.families.Gamma(sm.families.links.Log()))
gamma_results = gamma_model.fit()
gamma_results.summary()
```

Dep. Variable:	Invoice	**No. Observations:**	428
Model:	GLM	**Df Residuals:**	424
Model Family:	Gamma	**Df Model:**	3
Link Function:	Log	**Scale:**	0.059580
Method:	IRLS	**Log-Likelihood:**	-4346.8
Date:	Wed, 01 Nov 2023	**Deviance:**	23.319
Time:	02:10:09	**Pearson chi2:**	25.3
No. Iterations:	12	**Pseudo R-squ. (CS):**	0.9614
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
const	9.6571	0.169	57.017	0.000	9.325	9.989
Weight	0.0001	2.57e-05	4.863	0.000	7.47e-05	0.000
Length	-0.0058	0.001	-5.077	0.000	-0.008	-0.004
Horsepower	0.0055	0.000	25.850	0.000	0.005	0.006



Максимизация правдоподобия для GLM методом Ньютона-Рафсона

- Принцип максимума правдоподобия:

$$L(w) = -\log \prod_{i=1}^{l} p(y_i|\theta_i, \phi_i) = -\sum_{i=1}^{l} [y_i\theta_i - c(\theta_i)]/\phi_i \to \min_w \quad ,$$

где $\theta_i = w^T x_i$

- Метод Ньютона-Рафсона (t– номер итерации):

$$w^{t+1} = w^t - \eta_t \left(\nabla^2 L(w^t)\right)^{-1} \nabla L(w^t)$$

- Градиент $\nabla L\left(w^t\right)$:

$$\frac{\partial L(w)}{\partial w_j} = \sum_{i=1}^{l} \frac{y_i - c'(w^T x_i)}{\phi_i} x_i$$

- Матрица Гессе $\nabla^2 L(w^t)$:

$$\frac{\partial^2 L(w)}{\partial w_j \partial w_k} = -\sum_{i=1}^{l} \frac{c''(w^T x_i)}{\phi_i} x_i x_k$$

Метод IRLS
(Iteratively reweighted least squares)

- Обозначения:
  - Взвешенная (по наблюдениям) матрица признаков $\tilde{X} = W_t X$,
  - где X исходная матрица данных,
  - $W_t = diag\left(\sqrt{\frac{c''(\theta_i)}{\phi_i}}\right)$ – веса наблюдений на t-ой итерации
  - $\widetilde{y_i} = \frac{y_i - c`(\theta_i)}{\sqrt{\phi_i c``(\theta_i)}}$ – модифицированные отклики
- Метод Ньютона-Рафсона принимает вид:

$$w^{t+1} = w^t - \eta_t \underbrace{(\mathrm{X^T W_t W_t X})^{-1} \mathrm{X^T W_t}}_{(\tilde{X}^T\tilde{X})^{-1}\tilde{X}^T} \left(\underbrace{\sqrt{\frac{\phi_i}{c``(\theta_i)}} \frac{y_i - c'(\theta_i)}{\phi_i}}_{\widetilde{y_i}}\right)$$

- На каждом шаге - МНК линейной регрессии с взвешенными наблюдениями и модифицированными откликами:

$$\left\|\tilde{X} - \tilde{y}w\right\|^2 \to \min_w$$

Особенности поиска решения

- При небольшой выборке IRLS – лучший вариант
- Но на больших выборках используют методы:
  - градиентные (в том числе стохастические)
  - квазиньютоновские (в том числе lbfgs)
- Есть варианты борьбы с переобучением:
  - L_1 и L_2 регуляризация
  - пошаговый отбор переменных (вместо тестов Фишера или Стьюдента – тест Уальда, информационные критерии и кросс-валидация работают как и для МНК)
- Для оценки важности переменных используются:
  - стандартные ошибки расчета коэффициентов (за рамками нашего курса)
  - статистика Уальда для оценки важности коэффициентов:

$$\frac{w_i}{SE(w_i)} \sim N(0,1)$$

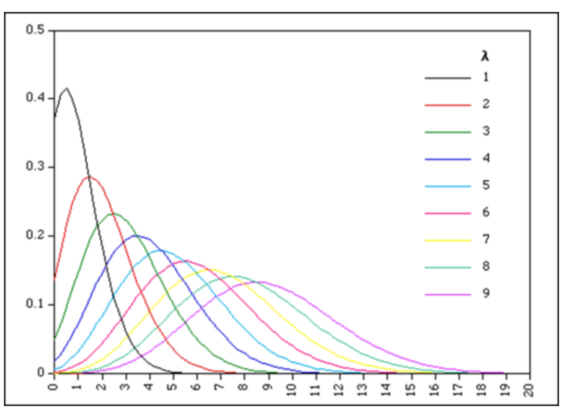
Пуассоновская регрессия

- Для моделирования количества наступлений события или доли (rate) наступлений события как функции от предикторов:
$$\mathbf{log}(E(y|x)) = w_0 + w_1x_1 + w_2x_2 + \ldots + w_px_k \Rightarrow$$
$$\mu(w) = e^{w_0} \cdot e^{w_1x_1} \cdots e^{w_px_p}$$
- Положительный (и как правило дискретный) отклик
- **Функция связи**: log
- **Функция потерь**: $L(x, y, w) = y\log\left(\frac{y}{\mu(w)}\right) - (y - \mu(w))$
- Интерпретация построенной модели:
  - e^{w} – мультипликативный эффект на отклик от изменения предиктора на единицу
  - Например, если $e^{w_1} = 1.2$, тогда увеличение x_1 на одну единицу вызывает 20% увеличение ожидаемого отклика, а если $e^{w_2} = 0.8$, тогда увеличение x_2 на одну единицу вызывает 20% уменьшение ожидаемого отклика

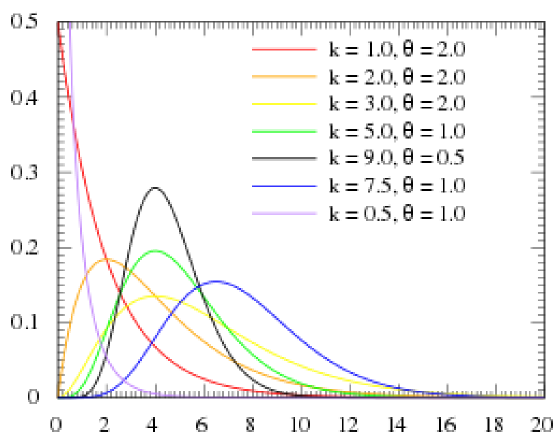
Пуассоновская регрессия

- Пуассоновская регрессия наиболее подходит для *редких* событий
  - распределение отклика должно иметь маленькое среднее (<10 или даже <5, в идеале ~1)
  - иначе гамма и логнормальное распределение может быть лучше чем пуассоновское, если распределение сильно ассиметричное или есть чрезмерная дисперсия
  - или нормальное, если распределение достаточно симметричное

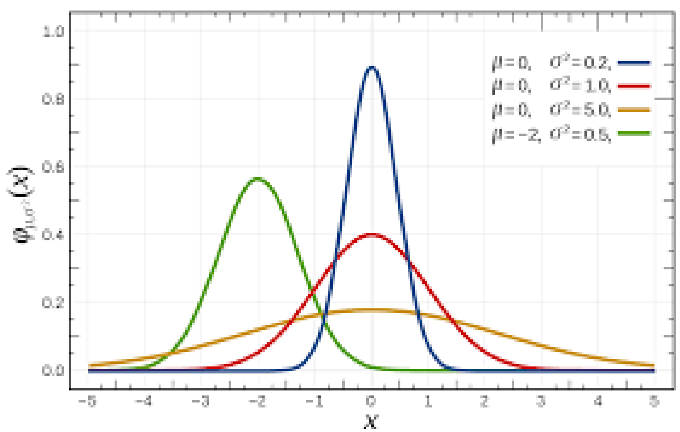
Пуассоновское



Гамма



Нормальное



Пример пуассоновской регрессии

```
dt=pd.read_csv("ships.csv",delimiter=",")
dt.head()
```

	type	age_period	operation_period	months	damages
0	1	1	1	127	0
1	1	1	2	63	0
2	1	2	1	1095	3
3	1	2	2	1095	4
4	1	3	1	1512	6

```
X.head()
```

	Intercept	C(type)[T.2]	C(type)[T.3]	C(type)[T.4]	C(type)[T.5]	months
0	1.0	0.0	0.0	0.0	0.0	127.0
1	1.0	0.0	0.0	0.0	0.0	63.0
2	1.0	0.0	0.0	0.0	0.0	1095.0
3	1.0	0.0	0.0	0.0	0.0	1095.0
4	1.0	0.0	0.0	0.0	0.0	1512.0

```
from patsy import dmatrices
import statsmodels.api as sm
y, X = dmatrices("damages~C(type)+months", dt, return_type="dataframe")
pois_model = sm.GLM(y,X, family=sm.families.Poisson())
pois_results = pois_model.fit()
pois_results.summary()
```

Generalized Linear Model Regression Results

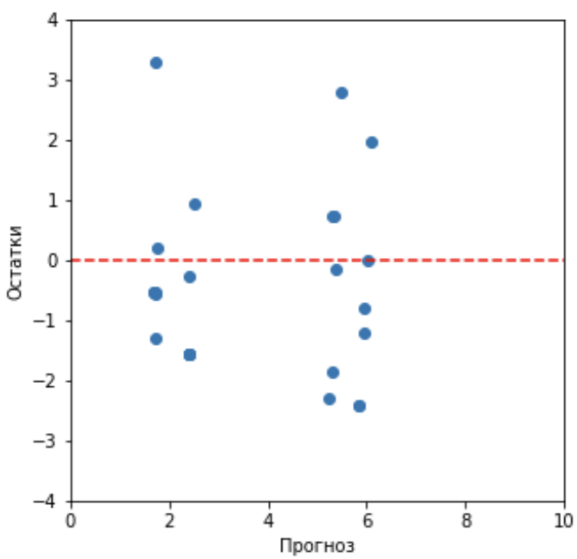
Dep. Variable:	damages	**No. Observations:**	34
Model:	GLM	**Df Residuals:**	28
Model Family:	Poisson	**Df Model:**	5
Link Function:	Log	**Scale:**	1.0000
Method:	IRLS	**Log-Likelihood:**	-125.73
Date:	Tue, 31 Oct 2023	**Deviance:**	153.59
Time:	02:55:48	**Pearson chi2:**	151.
No. Iterations:	6	**Pseudo R-squ. (CS):**	1.000
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	1.7650	0.154	11.429	0.000	1.462	2.068
C(type)[T.2]	1.4035	0.194	7.219	0.000	1.022	1.785
C(type)[T.3]	-1.2434	0.327	-3.798	0.000	-1.885	-0.602
C(type)[T.4]	-0.8902	0.287	-3.097	0.002	-1.454	-0.327
C(type)[T.5]	-0.1078	0.235	-0.460	0.646	-0.568	0.352
months	1.96e-05	4.61e-06	4.249	0.000	1.06e-05	2.86e-05

Пример пуассоновской регрессии

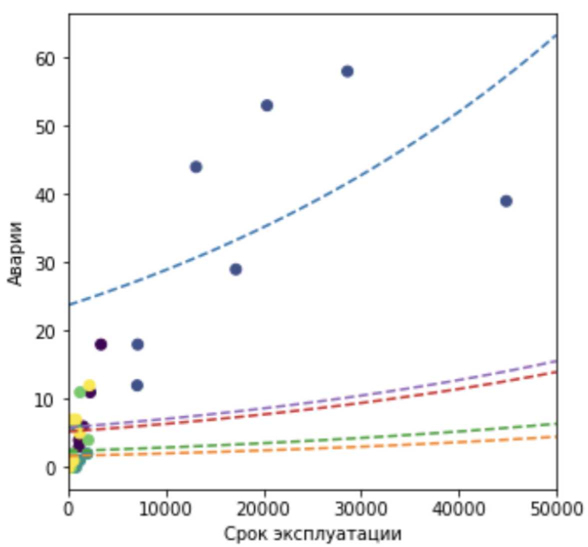
```
fig, ax = plt.subplots(figsize=(5, 5))
ax.scatter(pois_results.predict(X),
           pois_results.resid_pearson)
plt.xlim(0, 10)
plt.ylim(-4, 4)
ax.set_ylabel('Остатки')
ax.set_xlabel('Прогноз')
plt.axhline(y = 0, color = 'r', linestyle = '--')
```

```
<matplotlib.lines.Line2D at 0x2e7d9070c70>
```



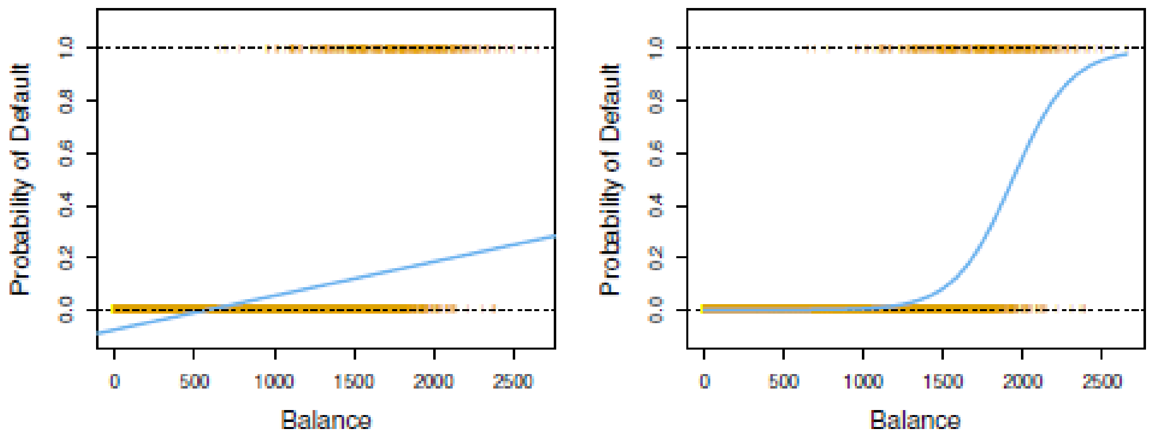

```
fig, ax = plt.subplots(figsize=(5, 5))
ax.scatter(dt['months'], dt["damages"],c=dt["type"])
m = 50000
p = 100
plt.xlim(0, m)
ax.set_ylabel('Аварии')
ax.set_xlabel('Срок эксплуатации')

for i in range(1,6):
    X=np.array([np.full(p,1), np.full(p,i==1),np.full(p,i==2),
                np.full(p,i==3),np.full(p,i==4),
                np.linspace(0,m,p)]).transpose()
    plt.plot(np.linspace(0,m,p),
         (pois_results.predict(X)), linestyle="--")
```



Логистическая регрессия

- Почему нельзя моделировать вероятность как непрерывный отклик с помощью линейной регрессии?



- Как представить категориальный отклик в виде числовой переменной?
- Если отклик закодирован (1=Yes, 0=No), а прогноз 1.1 или -0.4, что это означает?
- Если переменная имеет только два значения (или несколько), имеет ли смысл требовать постоянство дисперсии или нормальность ошибок?
- Вероятность ограничена, а линейная функция нет. Принимая во внимание ограниченность вероятности, можно ли предполагать линейную связь между предиктором и откликом?

Логистическая регрессия

Уравнение регрессии:

$$\mathrm{logit}(p_i) = \mu = w_0 + w_1 x_{1i} + \cdots + w_p x_{pi}$$

Вероятность — параметр — предиктор

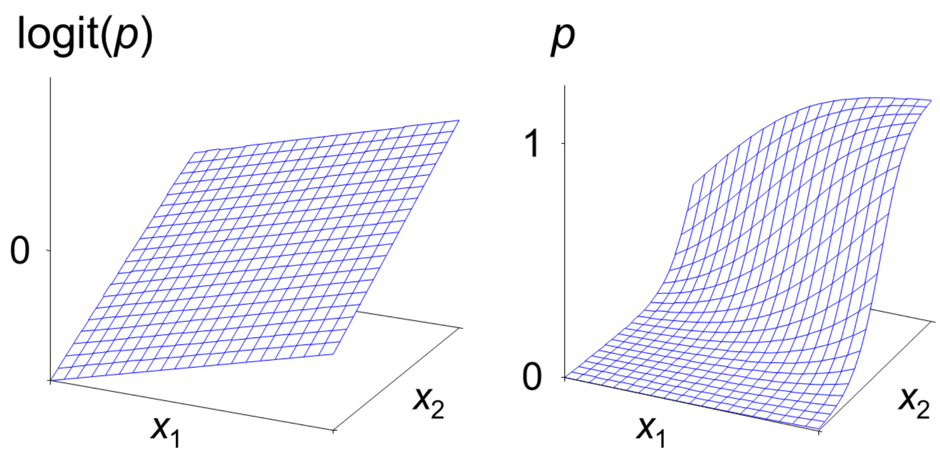
$$p_i = p(y = 1|x) = 1 - p(y = -1|x)$$

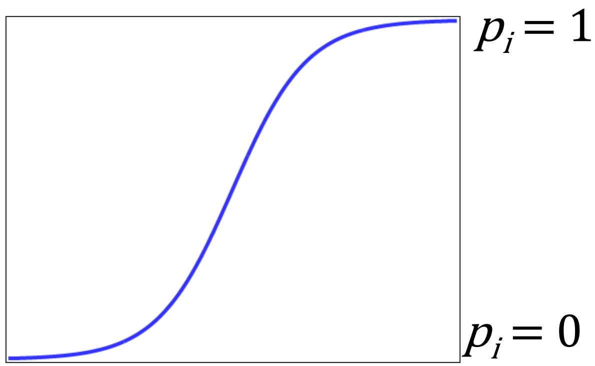
Функция связи (логит) и обратная ей (логистическая):

$$\mathrm{logit}(p_i) = \log\left(\frac{p_i}{1-p_i}\right) = \mu \Rightarrow$$

$$\Rightarrow \quad p_i = \sigma(\mu) = \frac{1}{1+e^{-\mu}} = \frac{1}{1+e^{-x^T w}}$$

Основное предположение линейной логистической регрессии (линейная зависимость логита вероятности от предикторов):



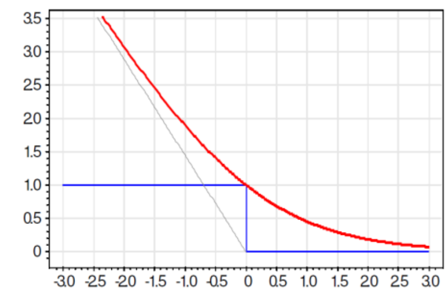


меньше $\leftarrow \mu \rightarrow$ больше

Ограничивает значение отклика

Функция потерь логистической регрессии

- **Функция потерь** (логарифмическая) является аппроксимацией негладкой функции потерь sign(.):



$$L(y,x,w) = \log\left[1 + \exp\left(-yw^T x\right)\right] \geq$$
$$\geq \ \mathrm{sign}(yw^T x)$$

- Градиент $\nabla Q(w)$ и матрица Гессе $\nabla^2 Q(w)$ для метода Ньютона-Рафсона:

$$w^{t+1} = w^t - \eta_t \left(\nabla^2 Q(w^t)\right)^{-1} \nabla Q(w^t)$$

$$\frac{\partial Q(w)}{\partial w_j} = \sum_{i=1}^{l} (1-\sigma_i) y_i x_i, \quad \frac{\partial^2 Q(w)}{\partial w_j \partial w_k} = -\sum_{i=1}^{l} (1-\sigma_i)\sigma_i y_i x_i x_k$$

где $\sigma_i = \sigma\left(y_i w^T x_i\right)$, $\sigma(z) = \frac{1}{1+e^{-z}}$ - сигмоидальная функция

IRLS для логистической регрессии

- На каждом шаге:
  - МНК линейной регрессии с взвешенными наблюдениями и модифицированными остатками, старающийся улучшить эмпирический риск на самых «сложных» примерах:

$$Q(w) = \sum_{i=1}^{l}(1-\sigma_i)\sigma_i\left(w^T x_i - \frac{y_i}{\sigma_i}\right)^2 \to \min_w \quad \leftrightarrow \left\|\tilde{X} - \tilde{y}w\right\|^2 \to \min_w$$

- где:
  - Взвешенная (по наблюдениям) матрица признаков $\tilde{X} = W_t X$
  - X исходная матрица данных,
  - $W_t = diag\big((1-\sigma_i)\sigma_i\big)$ – веса наблюдений на t-ой итерации,
  - поскольку $\sigma_i = P(y_i|x_i)$- вероятность правильной классификации x_i, то чем ближе x_i к границе, тем больше вес $(1-\sigma_i)\sigma_i$ и «сложнее» пример
  - $\tilde{y_i} = \frac{y_i}{\sigma_i}$ – модифицированные отклики, чем выше вероятность ошибки тем больше $\frac{1}{\sigma_i}$

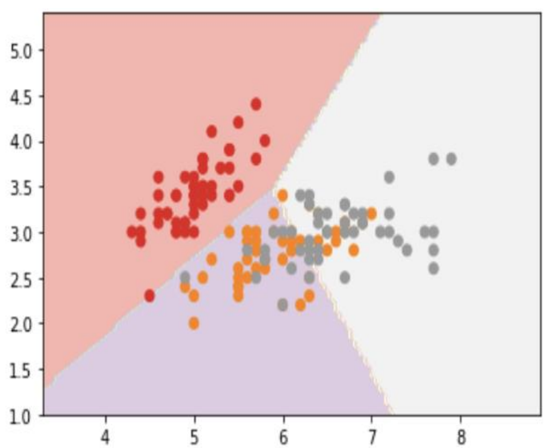
Многоклассовая логистическая регрессия и функция softmax

```
import matplotlib.pyplot as plt
from sklearn.linear_model import LogisticRegression
from sklearn import datasets
from sklearn.inspection import DecisionBoundaryDisplay

iris = datasets.load_iris()
X = iris.data[:, :2]
Y = iris.target

logreg = LogisticRegression()
logreg.fit(X, Y)

DecisionBoundaryDisplay.from_estimator(
    logreg, X, cmap="Pastel1")
plt.scatter(X[:, 0], X[:, 1], c=Y, cmap="Set1")
plt.show()
```



- Логистическая регрессия с двумя классами обобщается на случай K классов (многомерная логистическая функция):

$$p(y = k|x) = \frac{e^{x^T w_k}}{\sum_{j=1}^{K} e^{x^T w_j}}$$

- Для *каждой* пары классов существует своя граница - линейная разделяющая функция, где вероятности классов совпадают
- Многоклассовая логистическая регрессия также называется *мультиномиальной регрессией, а* многомерная логистическая функция -softmax, *которая* «нормализует» K-мерный вектор так, чтобы сумма координат = 1