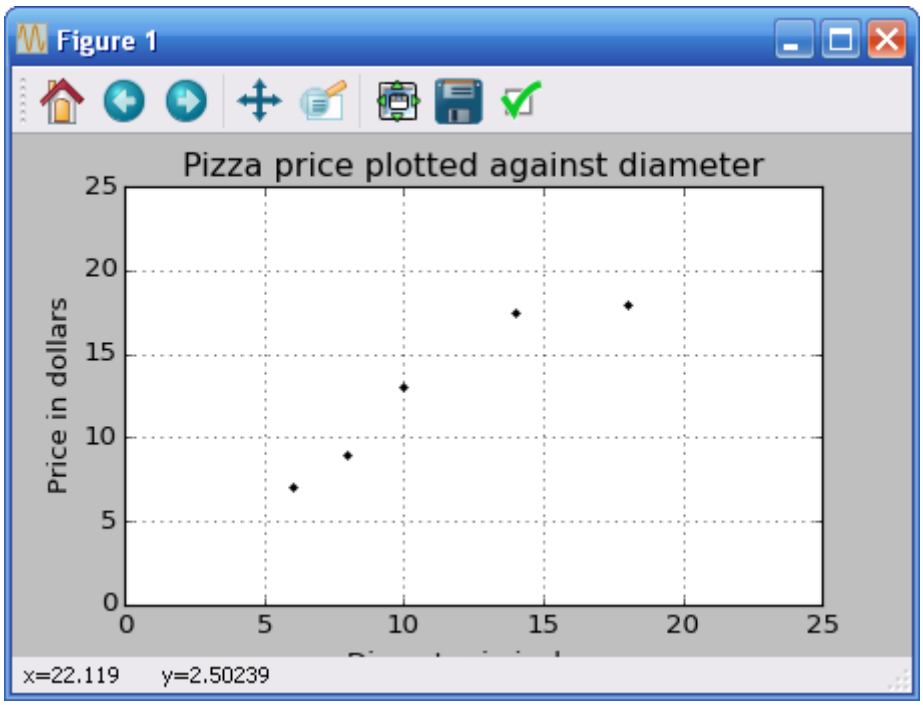
ЛЕКЦИЯ 14. ОСНОВЫ

Вывод графика

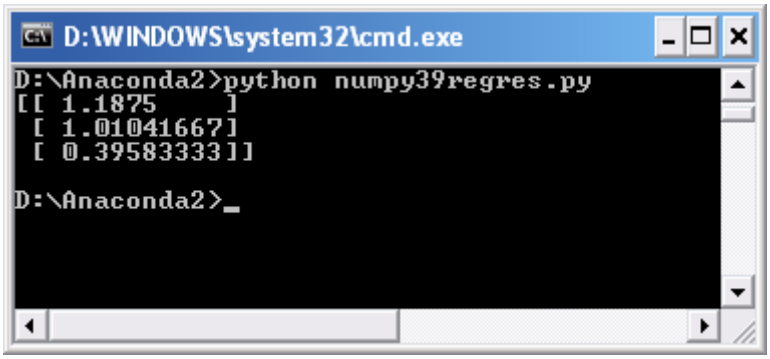
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import scipy.stats as stats
import statsmodels.api as sm
from scipy.optimize import fsolve
from scipy.linalg import *
from numpy import array
from scipy.cluster.vq import vq, kmeans, whiten
from sklearn import metrics
from sklearn.metrics import pairwise_distances
from sklearn import datasets
from sklearn import svm
import matplotlib.pyplot as plt
X = [[6], [8], [10], [14],   [18]]
y = [[7], [9], [13], [17.5], [18]]
plt.figure()
plt.title('Pizza price plotted against diameter')
plt.xlabel('Diameter in inches')
plt.ylabel('Price in dollars')
plt.plot(X, y, 'k.')
plt.axis([0, 25, 0, 25])
plt.grid(True)
plt.show()
```



Предсказание на основе модели

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import scipy.stats as stats
import statsmodels.api as sm
from scipy.optimize import fsolve
from scipy.linalg import *
from numpy import array
from scipy.cluster.vq import vq, kmeans, whiten
from sklearn import metrics
from sklearn.metrics import pairwise_distances
from sklearn import datasets
from sklearn import svm
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
# Training data
X = [[6], [8], [10], [14],   [18]]
y = [[7], [9], [13], [17.5], [18]]
# Create and fit the model
model = LinearRegression()
model.fit(X, y)
print 'A 12" pizza should cost: $%.2f' % model.predict([12])[0]
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+
```

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import scipy.stats as stats
import statsmodels.api as sm
from scipy.optimize import fsolve
from scipy.linalg import *
from numpy import array
from scipy.cluster.vq import vq, kmeans, whiten
from sklearn import metrics
from sklearn.metrics import pairwise_distances
from sklearn import datasets
from sklearn import svm
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from numpy.linalg import lstsq
X = [[1, 6, 2], [1, 8, 1], [1, 10, 0], [1, 14, 2], [1, 18, 0]]
y = [[7],    [9],    [13],    [17.5],  [18]]
print lstsq(X, y)[0]
```



++

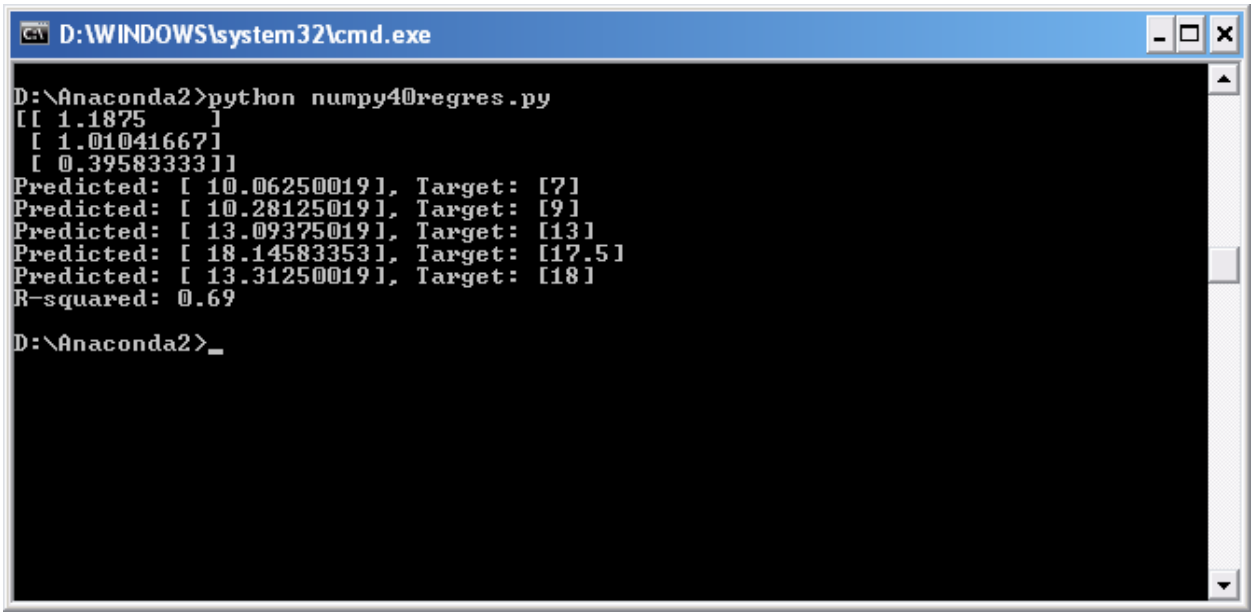
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import scipy.stats as stats
import statsmodels.api as sm
from scipy.optimize import fsolve
from scipy.linalg import *
from numpy import array
from scipy.cluster.vq import vq, kmeans, whiten
from sklearn import metrics
```

```
from sklearn.metrics import pairwise_distances
from sklearn import datasets
from sklearn import svm
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from numpy.linalg import lstsq

X = [[1, 6, 2], [1, 8, 1], [1, 10, 0], [1, 14, 2], [1, 18, 0]]
y = [[7],    [9],    [13],    [17.5],  [18]]
X_test = [[1,8, 2], [1,9, 0], [1,11, 2], [1,16, 2], [1,12, 0]]
y_test = [[11],   [8.5],  [15],    [17],    [11]]

print lstsq(X, y)[0]
model = LinearRegression()
model.fit(X, y)

predictions = model.predict(X_test)
for i, prediction in enumerate(predictions):
    print 'Predicted: %s, Target: %s' % (prediction, y[i])
print 'R-squared: %.2f' % model.score(X_test, y_test)
```



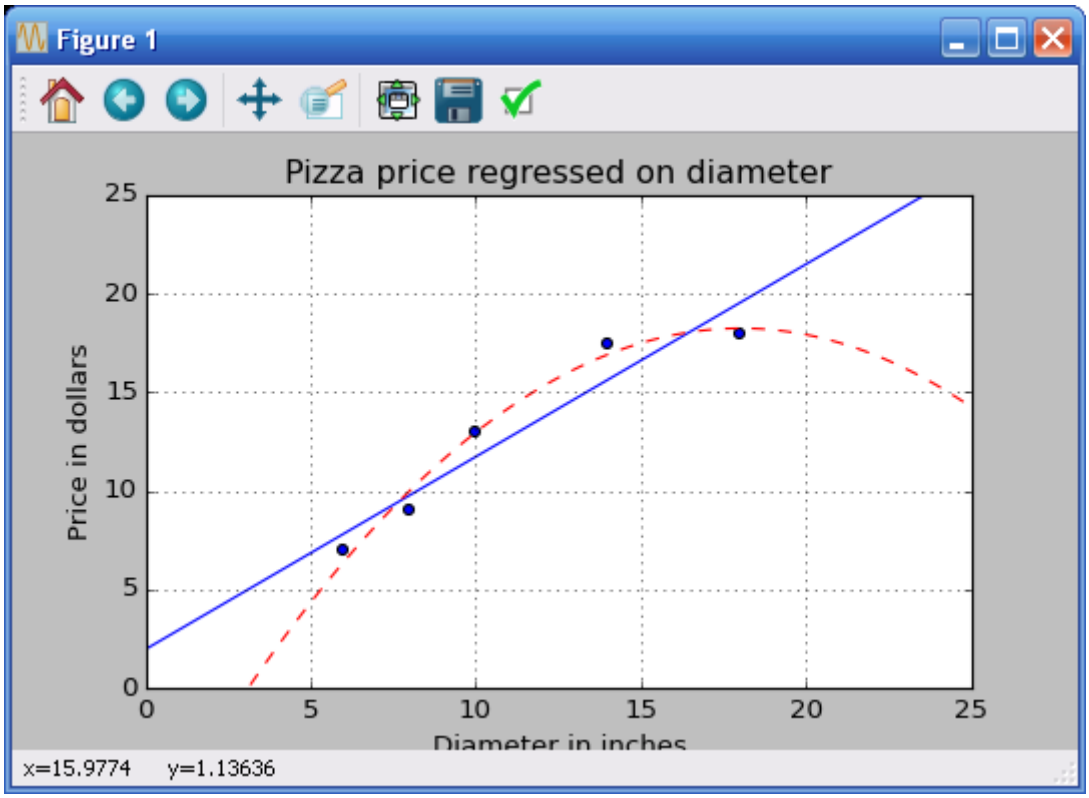
++

Полиномиальная регрессия

```
import numpy as np
import matplotlib.pyplot as plt
```

```
import pandas as pd
import scipy.stats as stats
import statsmodels.api as sm
from scipy.optimize import fsolve
from scipy.linalg import *
from numpy import array
from scipy.cluster.vq import vq, kmeans, whiten
from sklearn import metrics
from sklearn.metrics import pairwise_distances
from sklearn import datasets
from sklearn import svm
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from numpy.linalg import lstsq
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
X_train = [[6], [8], [10], [14],   [18]]
y_train = [[7], [9], [13], [17.5], [18]]
X_test = [[6],  [8],   [11], [16]]
y_test = [[8], [12], [15], [18]]
regressor = LinearRegression()
regressor.fit(X_train, y_train)
xx = np.linspace(0, 26, 100)
yy = regressor.predict(xx.reshape(xx.shape[0], 1))
plt.plot(xx, yy)
quadratic_featurizer = PolynomialFeatures(degree=2)
X_train_quadratic = quadratic_featurizer.fit_transform(X_train)
X_test_quadratic = quadratic_featurizer.transform(X_test)
regressor_quadratic = LinearRegression()
regressor_quadratic.fit(X_train_quadratic, y_train)
xx_quadratic = quadratic_featurizer.transform(xx.reshape(xx.shape[0], 1))
plt.plot(xx, regressor_quadratic.predict(xx_quadratic), c='r',linestyle='--')
plt.title('Pizza price regressed on diameter')
plt.xlabel('Diameter in inches')
plt.ylabel('Price in dollars')
plt.axis([0, 25, 0, 25])
plt.grid(True)
plt.scatter(X_train, y_train)
plt.show()
print X_train
print X_train_quadratic
print X_test
print X_test_quadratic
print 'Simple linear regression r-squared', regressor.score(X_test, y_test)
```

print 'Quadratic regression r-squared', regressor_quadratic.score(X_test_quadratic, y_test)



++

ЗАДАНИЕ.

1. Создать два массива: X и Y. Массив Y построить как известный полином второй степени. Задать этот полином произвольно. Наложить на массив Y случайные отклонения по нормальному закону.
2. Построить две регрессионные модели – первой и второй степени полиномы. Оценить ошибку регрессионной модели.