HIGHER SCHOOL OF ECONOMICS
NATIONAL RESEARCH UNIVERSITY

Rain Forecast

Homework Project

Anastasiya Dolmatova
MSc Program "Data Science"
1st Year

Moscow 2018

TABLE OF CONTEST

1. Data Description

1.1. Choice of Dataset

People have been interested in weather forecasting since ancient times. Now everyone has an opportunity to look into the future: thousands of websites, newspapers and magazines offer a weather forecast for tomorrow, next week or even next month. However, often these forecasts contradict each other. Moreover, they do not correspond to real weather in the predicted days. All existing rigorous mathematical models do not give a satisfactory results. Nevertheless, meteorologists collect a lot of data from different parts of the planet every day, such as atmospheric pressure, wind, temperature, humidity *etc***.** Many open access databases contain a history of observations of weather phenomena over many years. These data can be processed by methods of data science and machine learning in order to improve the accuracy of the forecast.

One of the open access weather data set contains the data about weather in Australia. This data set is published on the website https://www.kaggle.com/jsphyg/weather-dataset-rattle-package. This data set consists of 145460 entries, but some of them contains empty or "not a number" values. After removing incomplete data, the length of the set is reduced to 56420 records. In order to reduce the number of entries, we limit ourselves to reviewing data from the city of

Canberra. In this case, the data set consists of 1078 records. This dataset can be used to find regularities in changing weather conditions. For example, it can help to forecast if it will be rain tomorrow or not.

1.2. Features description

Each object is described by 24 features:

	Name	Description
1	Date	The date of observation
2	Location	The common name of the location of the weather station (24 possible values)
3	MinTemp	The minimum temperature in degrees celsius
4	MaxTemp	the maximum temperature in degrees celsius
5	Rainfall	the amount of rainfall recorded for the day in mm
6	Evaporation	the so-called Class A pan evaporation (mm) in the 24 hours to 9am
7	Sunshine	the number of hours of bright sunshine in the day.
8	WindGustDir	the direction of the strongest wind gust in the 24 hours to midnight (16 possible values, *e.g.* 'S', 'N', 'SSW')
9	WindGustSpeed	the speed (km/hr) of the strongest wind gust in the 24 hours to midnight
10	WindDir9am	direction of the wind at 9am
11	WindDir3pm	direction of the wind at 3pm

12	WindSpeed9am	wind speed (km/hr) averaged over 10 minutes prior to 9am
13	WindSpeed3pm	wind speed (km/hr) averaged over 10 minutes prior to 3pm
14	Humidity9am	humidity (percent) at 9am
15	Humidity3pm	humidity (percent) at 3pm
16	Pressure9am	atmospheric pressure (hpa) reduced to mean sea level at 9am
17	Pressure3pm	atmospheric pressure (hpa) reduced to mean sea level at 3pm
18	Cloud9am	fraction of sky obscured by cloud at 9am. This is measured in "oktas", which are a unit of eigths. It records how many eigths of the sky are obscured by cloud. A 0 measure indicates completely clear sky whilst an 8 indicates that it is completely overcast.
19	Cloud3pm	fraction of sky obscured by cloud (in "oktas": eighths) at 3pm. See Cload9am for a description of the values
20	Temp9am	temperature (degrees C) at 9am
21	Temp3pm	temperature (degrees C) at 3pm
22	RainToday	boolean: 1 if precipitation (mm) in the 24 hours to 9am exceeds 1mm, otherwise 0
23	RISK_MM	the amount of rain. A kind of measure of the "risk".
24	RainTomorrow	the target variable. Did it rain tomorrow? This value can be used to train any machine learning algorithm to predict if it will be rain tomorrow.

2. K-MEANS CLUSTERING

In this section, one of the most common approaches to cluster analysis will be discussed. K-means clustering aims to partition *n* observations into *k* clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster.

Hereinafter, all the code is written in Python using 'numpy' and 'pandas' libraries. The first library allows to work effectively with matrices, the second one simplifies the referring to specific rows or columns of the data frame. The 'numpy' library contains a standard function for K-means clustering, but for a better understanding, we implement this algorithm ourselves. The code implementing K-means algorithm is shown in Listing **2.1**.

Listing 2.1. K-means.

```
def kmeans(data, k):
    # Number of records and number of features
    n, c = data.shape

    # Choose initial centers randomly
    centers = np.random.randn(k,c)*np.std(data, axis = 0) + np.mean(data,
axis = 0)

      # To store old centers
    centers_old = np.zeros(centers.shape)

      # Clusters label to each row
    clusters = np.zeros(n)

      # Distance between each observation and the cluster centers
    distances = np.zeros((n,k))

    # While new centers are different from the old centerers
    while e > 1e-5:
        # Measure the distance to every center
        for i in range(k):
            distances[:,i] = np.linalg.norm(data - centers[i], axis=1)
        # Assign all training data to closest center
        clusters = np.argmin(distances, axis = 1)
        # Update the center
        centers_old = np.copy(centers)
        for i in range(k):
            centers[i] = np.mean(data[clusters == i], axis=0)
        e = np.linalg.norm(centers - centers_old)

    return ceters, clusters
```

The most natural way is to choose the number of clusters $k=2$ (rainy and clear days) and estimate how these clusters will correspond to real observations. To do this, we choose four quantitative features that can serve as factors reflecting the possibility of rain, such as the amount of sunny hours, humidity and pressure in the afternoon and daily temperature difference ('TempMax'-'TempMin'). Data were standardized before processing; each feature was centered by its mean and normalized by its range (see Listing **2.2**).

Listing 2.2. K-means preprocessing and criterion.

```
# Standartization
data = np.array((data-data.mean(axis=0)) / (data.max(axis=0) -
data.min(axis=0)))

# Run K-means ten time at different random initializations
for i in range(10):
    centers, clusters = kmeans(data, k)
    k_means_criterion = 0
    # Calculate K-means criterion
    for j in range(k):
        k_means_criterion += np.sum(np.linalg.norm(data[clusters==j] -
centers[j]))
    print(k_means_criterion)
```

By running the algorithm 10 times with different randomly initialized coordinates of the centers and choosing the best one over the K-means criterion, we obtained the following partition:

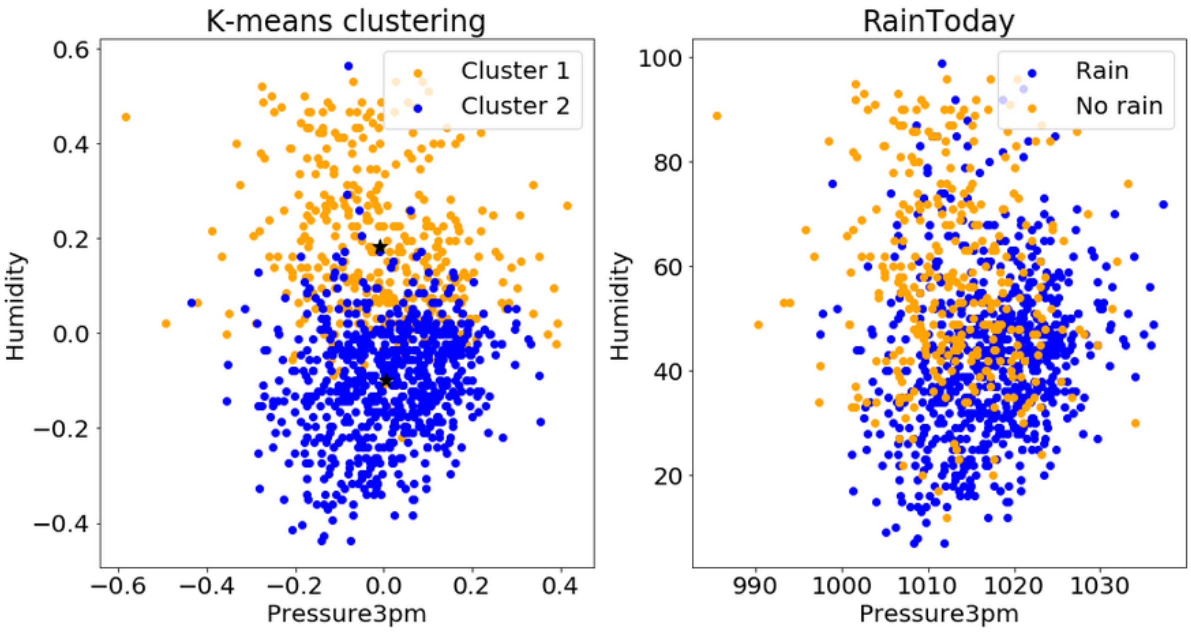

Fig. 2.1. 2D visualization of K-means clustering with k=2. The results of the K-means algorithm is shown in the left figure. Black stars indicate cluster centers. The right figure shows the actual distribution on rainy and dry days depending on pressure and humidity.

One can note, that the data has a poorly clustered structure. Nevertheless, the K-means algorithm allows us to draw some boundary, which can separates rainy days from dry ones more or less accurately. In Table **2.1**, we compare centers of two clusters (on the initial scale) with grand mean.

Table 2.1. Clusters characteristics (k=2).

	Pressure3pm, hPa	**Humidity3pm, %**	**Sunshine, h**	**TempDif, °C**
		Cluster 1		
Center	1016.21	37.36	9.62	14.92
Grand Mean	1015.76	47.14	7.40	12.64
Difference	0.45	-9.78	2.22	2.28
Difference, %	**0**	**-21**	**30**	**18**
		Cluster 2		
Center	1015.02	63.59	3.67	8.79
Grand Mean	1015.76	47.14	7.40	12.64
Difference	-0.74	16.45	-3.73	-3.85
Difference, %	**0**	**35**	**-50**	**-30**

Noticeable, that humidity, temperature difference and number of sunny hours affect the possibility of rain much more significantly than pressure. Most likely, pressure can be eliminated from the clustering process.

We considered different sets of features, but could not single out those that would separate the observations into obvious clusters. However, according to the assignment, we attempted to find k=5 clusters using the same features. Figure **2.2** illustrates the results.

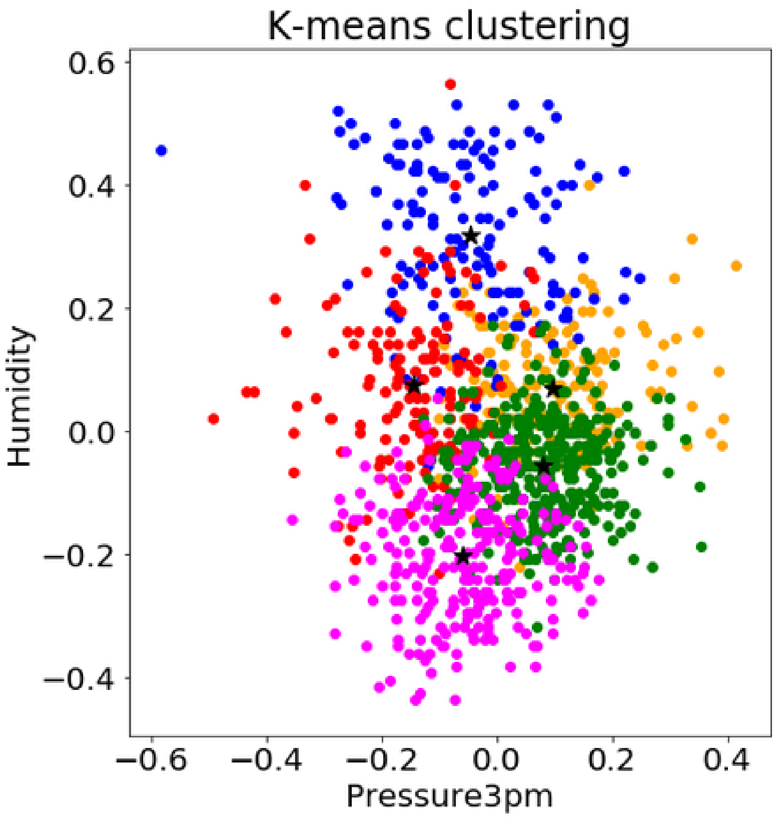

Fig. 2.2. 2D visualization of K-means clustering with k=5. Black stars indicate cluster centers.

Interpreting the resulting cluster set is a nontrivial task, It is difficult to say how they correspond to the actual parameters of the system. In order to determine which parameters dominated the construction of clusters, we compare centers of clusters with grand mean (see Table **2.2**).

Table 2.1. Clusters characteristics (k=5).

	Pressure3pm, hPa	**Humidity3pm, %**	**Sunshine, h**	**TempDif, °C**
Cluster 1				
Center	1010.89	54.96	6.96	8.19
Grand Mean	1015.76	47.14	7.40	12.64
Difference	-4.87	7.82	-0.44	-4.45
Difference, %	**0**	**17**	**-6**	**-35**
Cluster 2				
Center	1014.77	72.44	1.74	7.57
Grand Mean	1015.76	47.14	7.40	12.64
Difference	-0.99	25.30	-5.66	-5.07
Difference, %	**0**	**54**	**-76**	**-40**
Cluster 3				
Center	1020.77	48.03	5.93	13.85

Grand Mean	1015.76	47.14	7.40	12.64
Difference	5.01	0.89	-1.47	1.21
Difference, %	**0**	**2**	**-20**	**10**

Cluster 4

Center	1015.36	28.70	10.89	18.37
Grand Mean	1015.76	47.14	7.40	12.64
Difference	-0.40	-18.44	3.49	5.73
Difference, %	**0**	**-39**	**47**	**45**

Cluster 5

Center	1015.64	39.79	9.92	2.81
Grand Mean	1015.76	47.14	7.40	12.64
Difference	-0.12	-7.35	2.52	0.17
Difference, %	**0**	**-16**	**34**	**1**

Again, pressure has the least impact on cluster splitting.

To summarize, the data under consideration does not have clearly defined clusters in its structure. However, the use of clustering techniques helps to highlight some of the properties of the data.

3. BOOTSTRAP FOR CLUSTER INTERPRETATION

We will consider the clusters obtained in a two-cluster partition. This clustering defines the boundaries of rainy and clear days more or less well. One of the features used for clustering is amount of the sunny hours. In Figure **3.1** histograms for the sunshine features within the whole data set and within the clusters are presented.

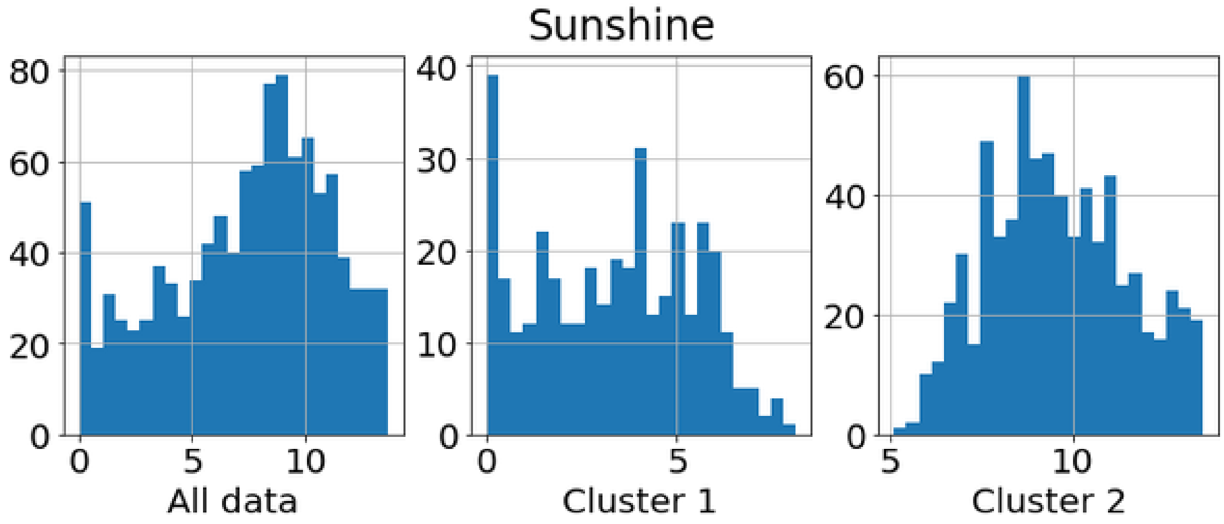

Fig. 3.1. Histograms of 'Sunshine' feature distribution (bins=25).

Obviously, the average value of the 'Sunshine' feature is significantly lower on rainy days. All the distribution is rather far from Gaussian. Means and standard deviations for the distributions are presented in Table **3.1**.

Table 3.1. Mean values and standard deviations for the 'Sunshine' feature.

	All data	**Cluster 1**	**Cluster 2**
Mean	7.40	3.32	9.59
Std	3.59	2.08	1.91

Let us validate the Mean within the clusters using bootstraping. The code for getting *M* Means using bootstrap is presented in Listing **3.1.**

Listing 3.1. Bootstrap.

```
# Generate M Means using bootstrap
def bootstrap(x, M):
    # Number of samples
    N = x.shape[0] - 1
    r = np.ceil(N*np.random.rand(N,M)).astype('int')
    xr = x[r]
    mr = np.mean(xr, axis=0)
    return mr

# Pivotal method
def get_boundaries_pivotal(mr):
    # 95% confidence
    mmr = np.mean(mr)
    smr = np.std(mr)
    lbp = mmr - 1.96 * smr
    rbp = mmr + 1.96 * smr
    return [lbp, rbp]

# Not pivotal method
def get_boundaries_not_pivotal(mr):
    # Take 2.5% and 97.5% percentiles as the boundaries
    lbn = np.percentile(mr, 2.5)
    rbn = np.percentile(mr, 97.5)
    return [lbn, rbn]
```

Figure **3.2** illustrates the histograms for $M = 5000$ Means generates by bootstrap.

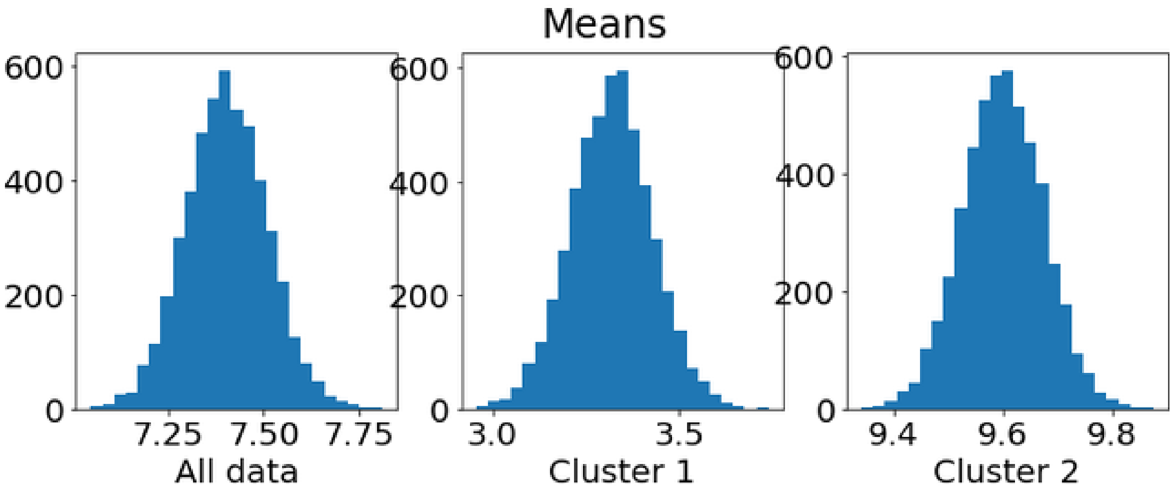

Fig. 3.2. Histograms of 5000 Means generated by Bootstrap (bins=25) for the whole dataset and two clusters.

One can note, that the distributions are quite Gaussian with means around 7.4, 3.3 and 9.6. Cluster 2 corresponds to days without rain, therefore the average number of hours of sunshine in this cluster is above the average and significantly higher than the value in the cluster 1, which correspond to the rainy days.

Let us find the 95% confidence interval for grand mean by using bootstrap. It can be find by using pivotal and not pivotal methods (see Listing **3.1**). The results are presented in Table **3.2**.

Table 3.2. 95% confidence interval for grand mean

	Left	Right
Pivotal	7.18	7.61
Not Pivotal	7.18	7.62

One can see, that the interval boundaries values obtained by two different methods are very close to each other. The mean value 7.4 lies inside the interval.

4. CONTINGENCY TABLE ANALYSIS

We will consider three nominal features. The most obvious way is to choose feature 'RainToday', which is binary and has only two options ('Yes' or 'No'). However, in a real forecast it is important to reflect not only the presence of rain, but also how much precipitation will fall. Therefore, we will introduce a new categorical feature 'RainPower' based on the feature 'Rainfall'. Figure **4.1** shows a histogram of this feature. The greatest number of observations is close to zero, this is understandable, since there are 859 from 1083 days without rain (according to the 'RainToday' feature).

Thus, we define new feature 'RainPower' ' as follows:

- 'No' if 'Rainfall' $\leq$ 1.1 mm. This threshold is chosen so that the number of observations in this category is consistent with the number of observations in which feature 'RainToday' is 'No'.

- 'Rain' if 'Rainfall' > 1.1 mm and ≤ 15 mm. This threshold is chosen, since there is a recession in the histogram on this level. The selected value is consistent with meteorological standards.
- 'Shower' if 'Rainfall' > 15 mm.

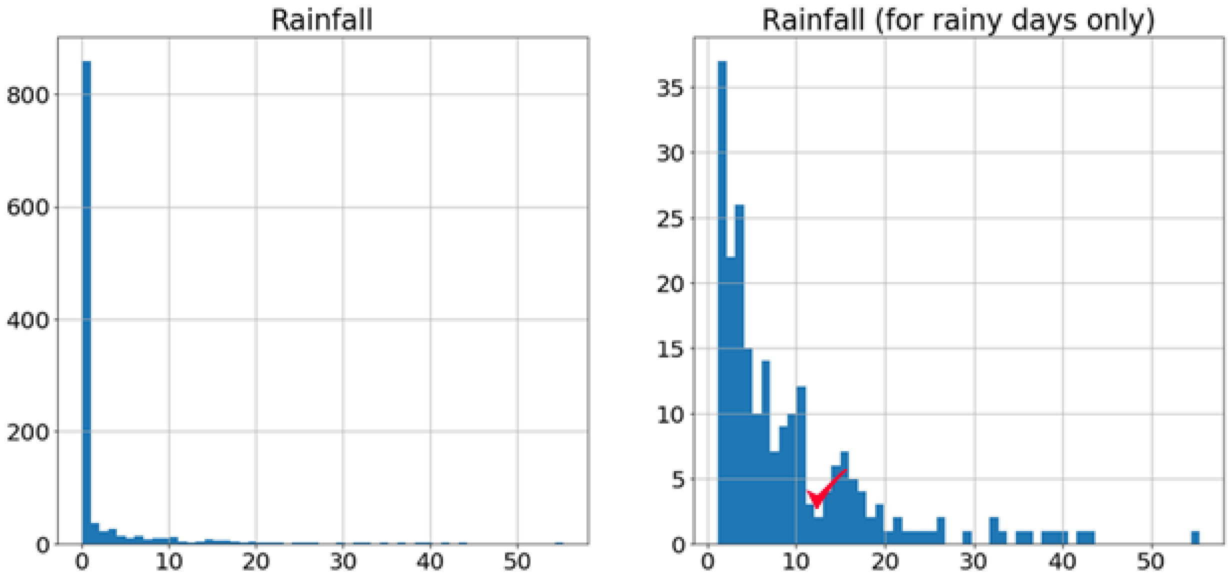

Fig. 4.1. Histogram of the feature 'Rainfall'. The left histogram is plotted for the whole data set. The right histogram is plotted for the rainy days only.

The second nominal feature will be constructed from the feature 'WindGustSpeed' (the speed (km/h) of the strongest wind gust in the 24 hours to midnight). We define its values as follows (see Figure **4.2**):

- 'Light': 'WindGustSpeed' ≤ 30
- 'Medium': 'WindGustSpeed' > 30 and 'WindGustSpeed' ≤ 60
- 'Strong': 'WindGustSpeed' > 60

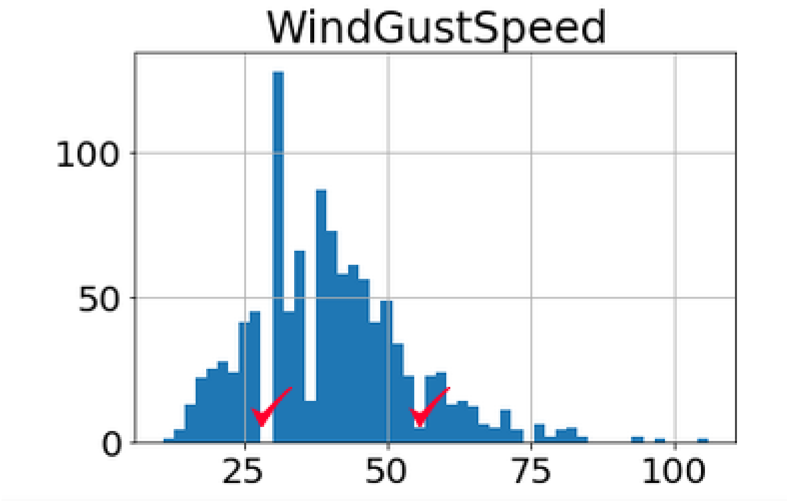

Fig. 4.2. Histogram of the feature 'WindGustSpeed'.

The third feature will be construct from the feature 'Evaporation' (see Fig. **4.3**).

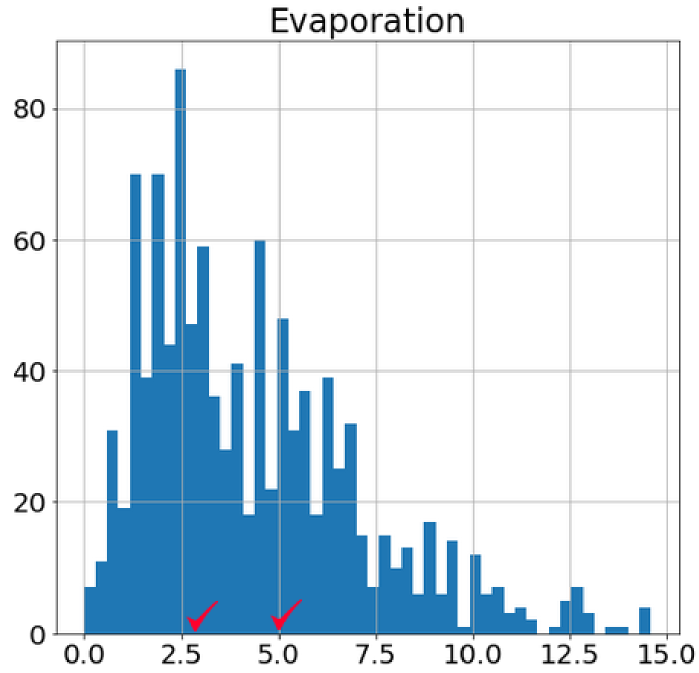

Fig. 4.3. Histogram of the feature 'Evaporation'.

We construct it in such a way that each category has the same number of observations:

- 'Light': 'Evaporation' ≤ 2.6 (percentile 33%)
- 'Medium': 'Evaporation' >2.6 and 'WindGustSpeed' ≤ 6(percentile 66%)
- 'Strong': 'Evaporation' >5

The code that generates new features is presented in Listing **4.1**.

Listing 4.1. New nominal features.

```python
# RainPower
def rain_power(rainfall):
    if rainfall <= 1.1:
        return 'No'
    if rainfall <= 15:
        return 'Rain'
    return 'Shower'

data['RainPower'] = data['Rainfall'].apply(lambda row: rain_power(row))

# WindPower
def wind_power(speed):
    if speed <= 30:
        return 'Light'
    if speed <= 60:
        return 'Meduium'
    return 'Strong'

data['WindPower'] = data['WindGustSpeed'].apply(lambda row: wind_power(row))

# EvaporationPower
threshold1 = data['Evaporation'].quantile(0.33)
threshold2 = data['Evaporation'].quantile(0.66)
def evaporation_level(evap):
    if evap <= threshold1:
        return 'Light'
    if evap <= threshold2:
        return 'Meduium'
    return 'Strong'

data['EvaporationLevel'] = data['Evaporation'].apply(lambda row:
evaporation_level(row))
```

We will build two contingency tables over these features: a conditional frequency table and Quetelet relative index tables. The conditional frequency table for feature 'WindPower' and target feature 'RainPower' is presented in Table **4.1**. It shows the probability that there will be precipitation of a particular power, if the maximum wind speed fell into one or another category.

Table 4.1. Conditional frequency table

Rain Power	**Wind Power**		
	Light	**Medium**	**Strong**
No	0.88	0.80	0.54
Rain	0.11	0.16	0.36
Shower	0.01	0.04	0.10
Total	1	1	1

This table shows that the presence of wind does not entail precipitation in most cases, but with strong wind the probability of rain (and even shower) increases.

The Quetelet relative index table for feature 'EvaporationLevel' and target feature 'RainPower' is presented in Table 4.2. Quetelet index allows to measure the extent of association between row and column categories in a contingency table by comparing the local count with an average one.

Table 4.2. Quetelet relative index tables

Rain Power	Evaporation Level (100**q(ELk/RPi)*)		
	Light	**Medium**	**Strong**
No	0.5	4.2	-4.46
Rain	-3.2	-24.1	26.2
Shower	4.3	18.1	-21.4

There are no large numbers in the table that would indicate a significant increase in the probability of any association. The maximum value is 26.2, which means that strong Evaporation Level, given Rain, is 26.2% frequent than on average.

The algorithm for calculating the described tables is shown in Listing **4.2**.

Listing 4.2. Contingency tables.

```
# Contingency table
def cont_table(cat1, cat2):
    # Extract categories
    c1_range = data_p[cat1].unique()
    n = len(c1_range)
    c2_range = data_p[cat2].unique()
    m = len(c2_range)

    # Calculate contingency table
    ct = np.zeros((n,m))
    for i, rp in enumerate(c1_range):
        for j, wp in enumerate(c2_range):
            ct[i, j] = data_p[(data_p[cat1]==rp) &
(data_p[cat2]==wp)][cat1].count()
    return ct

# Contingency relative frequency table
def rct(ct):
    return ct / np.sum(np.sum(ct))


# Conditional frequency table
def cft(ct):
    return ct / np.sum(ct,axis=0)

##Quetelet relative index tables
def qrit(ct):
    # Possibility of Categoty 1
    prp = ct.sum(axis=1) / np.sum(np.sum(ct))
    # Possibility of Category 2
    pel = ct.sum(axis=0) / np.sum(np.sum(ct))
    qrip = ct / np.sum(np.sum(ct))
    qrip = (qrip.T / prp).T
    qrip = (qrip - pel)/pel
    return qrip

# Summary Quetelet index
def sqi(ct):
    return np.sum(np.sum((qrit(ct) * ct)))
```

Let us multiply Quetelet coefficients by the frequency of the corresponding entries in Contingency table. But in our case, corresponding values are too small, so we will multiply results by the number of entities N (in many textbooks, this valued is called 'χ^2'). This leads us to Tables **4.3** and **4.4** whose entries sum up to the value of Pearson’s chi-square coefficient multiply the number of entities.

Table 4.3. Chi-squared*N and its decomposition for 'RainPower' and 'EvaporationLevel'

Rain Power	Evaporation Level			Total (χ^2)
	Light	**Medium**	**Strong**	
No	1.27	12.66	-12.81	1.11
Rain	-1.73	-11.09	20.70	7.87
Shower	0.56	2.90	-2.35	1.11
Total	0.09	4.47	5.53	**10.09** **(0.009)**

Table 4.4. Chi-squared*N and its decomposition for 'RainPower' and 'WindPower'

Rain Power	Wind Power			Total (χ^2)
	Light	**Medium**	**Strong**	
No	22.46	2.74	-15.50	9.70
Rain	-9.65	-2.33	38.08	26.10
Shower	-2.30	-0.34	15.80	13.17
Total	10.52	0.07	38.38	**48.97** **(0.045)**

Finally, we get the following coefficient:

- for 'RainPower' – 'Evaporation Level' $Q = \chi^2 = 0.009$.
- for 'RainPower' – 'WindPower' $Q = \chi^2 = 0.045$;

$Q = \chi^2$ is the average relative increase in the occurrence of 'RainPower' category values when 'WindPower' or 'EvaporationLevel' values become known. According to Chi-square distribution table, the hypothesis of independence is to be rejected with more than 95% confidence in the first case and with more than 99% confident in the second case. To summarize, the statistical relationship between the categories is small, the features can be obtained as statistical independent.

Let us find the numbers of observations that would suffice to see the features as associated at 95% confidence level and 99% confidence level according Pearson's theorem. We have K=3, L=3, therefore number of freedom f=4. At f=4, critical values for χ^2 $\chi^2_{cr}(\alpha = 0.95) = 0.711$, $\chi^2_{cr}(\alpha = 0.99) = 0.297$. In our case, $\chi^2{}_1 = 0.009$, so for 95% confidence level, the numbers of observations are $N_{0.95} = 0.711 / 0.009 = 79$ and $N_{0.95} = 0.711 / 0.045 = 15$; for 99% confidence level, the numbers of observations are $N_{0.98} = 0.297 / 0.009 = 33$ and $N_{0.99} = 0.297 / 0.045 = 6$ respectively.

5. PCA: Hidden Factor & Data Visualization

To decompose into principle components, we will use a subset that includes the following features: 'DifTemp' ('MaxTemp'-'MinTemp'), 'Humidity3pm', 'Pressure3pm', 'WindGustSpeed'. All of these features are quantitative. These characteristics were chosen, since these characteristics are traditionally used the key factors in the models of weather prediction.

First of all, we apply standardization function to the matrix. Standardization can be performed in two ways: a) over ranges and (b) over standard deviations. We will test both options. Listing **5.1** presents the algorithm for PCA decomposition.

Listing 5.1. Principal Components Analysis.

```
features = ['DifTemp', 'Humidity3pm', 'Pressure3pm', 'WindGustSpeed']
X = data_p[features]

# Standartization
X_std = np.array((X - X.mean())/X.std())
X_range = np.array((X - X.mean())/(X.max()-X.min()))

# For X_std
# SVD
U, s, Vh = np.linalg.svd(X_std, full_matrices=False)
# DataScatter
ds = np.sum(np.sum(X_std ** 2))
# Explained variance
print(s**2 / ds)

# For X_range
# SVD
U, s, Vh = np.linalg.svd(X_range, full_matrices=False)
# DataScatter
ds = np.sum(np.sum(X_range ** 2))
# Explained variance
print(s**2 / ds)
```

Table **5.1** shows contributions of all the principal components to the data scatter, naturally and per cent.

Table 5.1. Contributions of principal components to the data scatter

Component	**Standardization over range**		**Standardization over std**	
	Naturally	**%**	**Naturally**	**%**
1	72	57.0	1957	45.4
2	33	26.4	1563	36.3
3	12	9.6	499	11.6
4	9	7.0	288	6.7

One can see that in the both cases, the first two components explain more than 80% of the variance.

Figure **5.2** illustrates the data projected on the first two components.

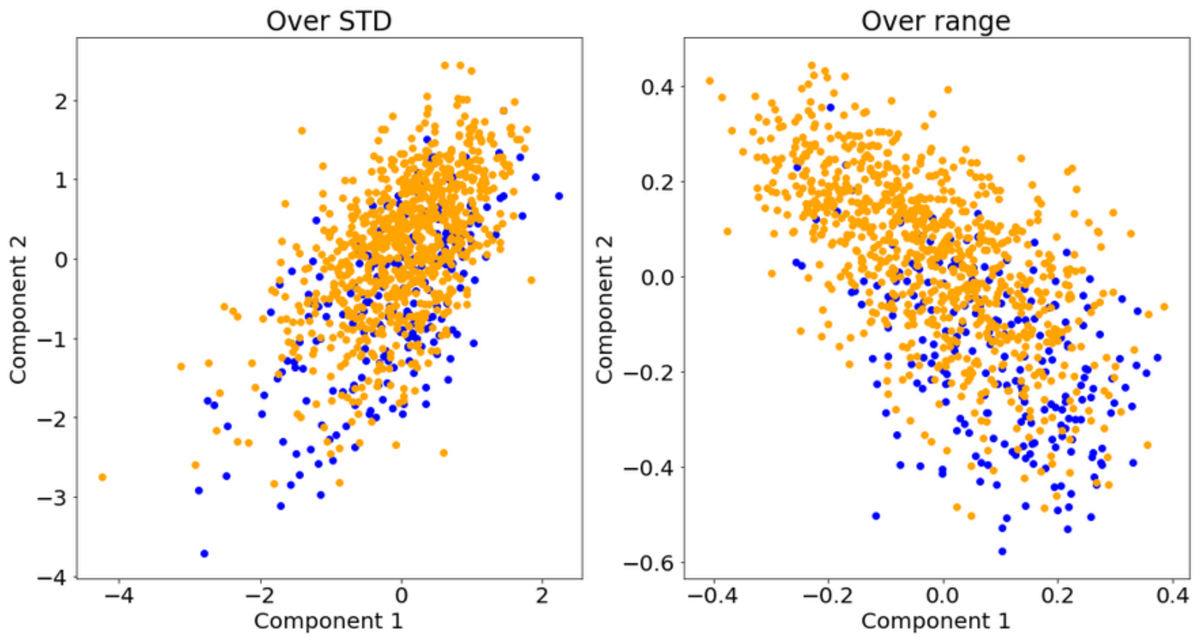

Fig. 5.1. Principal components visualization. Left figure corresponds to standardization over standard deviation, right figure corresponds to standardization over range. Rainy days are plotted by blue circles, wet days are plotted by orange circles.

Noticeable, that in this case, two types of data points cannot be easily separated from each other. Visually, it seems that standardization over range allows to separate the classes more accurately.

In order to find a hidden factor, we will find the first singular triplet. The code is presented in Listing **5.2**.

Listing 5.2. Hidden factor.

```
# First singular triplet
# Scale X
X_scaled = X * 100 / X.max()
# SVD
U, s, Vh = np.linalg.svd(X_scaled, full_matrices=False)
# First singular triplet
z = U[:, 0]
mu = s[0]
c = s.T
#rescaling
alpha = 1 / np.sum(s)
# PCA hidden factor
z_scale = X_scaled.dot(s)*alpha
# Determine its contribution to the data scatter
mu**2 / np.sum(np.sum(X_scaled **2))
```

The found hidden factor is $hf = 0.74 * DifTemp + 0.14 * Humidity + 0.08 * \mathrm{Pr}essure + 0.04 * WindGusiDir$. The main contribution to the factor is made by the temperature change during the day. Humidity is in the second place. Wind speed has almost no effect on the hidden factor. The contribution of the hidden factor is quite good: more than 95%.

The principle components can be found in different ways. The implementation of conventional approach is presented in Listing **5.3**.

Listing 5.3. Conventional PCA.

```
# Conventional PCA
n,m = X.shape
# Centering
Y = X - np.mean(X, axis=0)
# Tbale B
B = Y.T @ Y / n
# Eigenvalues and eigenvectors
```

```
la, c = np.linalg.eig(B)
idx = la.argsort()
c = c[:, idx]
la = la[idx]
z = np.zeros((m,n))
# Principal components
for i in range(m):
    z[i, :] = Y @ c[:,-i-1] / np.sqrt(n*la[-i-1])
print(np.round(z,2))
```

All principle components found by this approach completely coincide with the principle components found by SVD approach.

6. 2D Regression

Consider the dependence of humidity in the afternoon on the number of sunny hours in the day. It is intuitively clear that these two characteristics should be proportional to each other. This dependence is shown in Figure **6.1**.

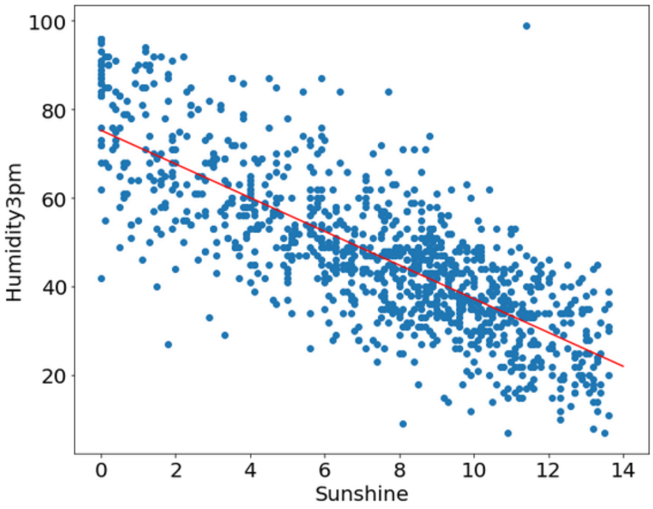

Fig. 6.1. The dependence of humidity in the afternoon on the number of sunny hours in the day. Linear approximation is indicated by a red line.

Construct a linear regression for these two features. One can find regression coefficients using the least squares method, by solving a matrix equation or by calculating correlation and determinacy coefficients. The code for the matrix method is shown in Listing 1.

Listing 6.1. Linear regression.

```
# Linear regression
x = data['Sunshine']
y = data['Humidity3pm']

# Matrix method
X = np.vstack((x, np.ones(n))).T
u = y
w = np.linalg.inv(X.T @ X) @ X.T @ u

# Correlation coefficient
rho = ((x - x.mean()) * (y - y.mean())).sum() / n / (x.std() * y.std())
# Determinacy coefficient
rho2 = rho ** 2

# Check coefficient
a = rho * y.std() / x.std()
b = y.mean() - a*x.mean()

# Make prediction for the first five values
y_predicted = a*x[0:5] + b
```

As a result, we have the following parameters of two-dimensional linear regression: the slope of the line is $a = -3.80$ (every sunny hour reduces humidity by 3.8%), and the intercept is $b = 75.29$ (if it is cloudy all day, the average humidity is 75.29%). A negative slope means that the relationship between humidity and the number of sunny hours is inverse. Indeed, the longer the sun shines, the more moisture evaporates, and the humidity decreases.

Let us find the correlation and determinacy coefficients. The correlation coefficient can be found by the formula

$$\rho = \frac{\sum_{i=1}^{N}(y_i - \overline{y})(x_i - \overline{x}) / N}{\sigma(x)\sigma(y)} \quad ,$$

where $\overline{x}$ and $\overline{y}$ are the mean values of x and y respectively, $\sigma(x)$ is the standard deviation. The determinacy coefficient is determined as ρ^2 . Using these formulas, one obtains

$$\rho = \ -0.78,$$
$$\rho^2 = 0.60.$$

The correlation coefficient defines a measure of degree of a linear relation between x and y. If $|\rho| = 1$, the relation is perfectly linear. The value 0.78 is relatively high. It is negative, because humidity and sunny hours are related negatively, the slope a is negative. The determinacy coefficient shows that about 60% of the variance $\sigma^2(y)$ is taken into account by the linear regression.

These coefficients also allow us to find linear regression parameters (see Listing **6.1**):

$$a = \rho \frac{\sigma(y)}{\sigma(x)} = -3.8$$
$$b = \overline{y} - a\overline{x} = 75.26.$$

One can see that the regression coefficients found by two different methods are very close to each other.

Make a prediction of the target values for the first five predictors values in the dataset:

Table 6.1. Prediction of humidity using linear regression.

	Sunshine, h	Humidity3pm, %	Humidity3pm predicted, %	Relative error, %
1	6.3	29	***51.32***	77.0
2	9.7	36	***38.40***	6.6
3	3.3	69	***62.72***	9.0
4	9.1	56	***40.68***	27.4
5	10.6	49	***34.98***	28.6

One can see that in this case, linear regression captures a general pattern of relation between the number of sunny hours and humidity, but does not give an exact dependence.

In addition to the determinacy coefficient, various metrics are used to rate the quality of the approximation, such as

- MRE (Mean Relative Error)

$$MRE = \frac{\sum_{i=1}^{N} \left| y_i - \tilde{y}_i \right| / \left| y_i \right|}{N} = 22.6\%.$$

- RMSE (Root Mean Squared Error)

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} \left(y_i - \tilde{y}_i \right)}{N}} = 11.07.$$

- RAE (Relative Absolute Error)

$$RAE = \frac{\sum_{i=1}^{N} \left| y_i - \tilde{y}_i \right|}{\sum_{i=1}^{N} \left| y_i - \overline{y} \right|} = 63\% \ .$$

- MRAE (Mean Relative Absolute Error)

$$RAE = \frac{1}{N} \frac{\sum_{i=1}^{N} \left| y_i - \tilde{y}_i \right|}{\sum_{i=1}^{N} \left| y_i - \overline{y} \right|} = 0.6\%$$

- *etc.*

In general, these metrics give different estimates of the quality of the regression. In our case, linear regression has a rather high error. This can be seen by the determinacy coefficient and by other metrics.

CONCLUSION

In this paper, some data analysis techniques were applied to the task of weather forecasting. The complex nature of the dependency of the probability of rain on such parameters as humidity, temperature, pressure, *etc.* was demonstrated. We attempted to apply cluster analysis using K-means method. It was shown that the data has a low cluster structure. The factors that have the greatest influence on the rain probability were discovered by applying principal component analysis. The dependences of some nominal features on others nominal features were also investigated using the contingency tables and their analysis.