П Р А В И Т Е Л Ь С Т В О Р О С С И Й С К О Й Ф Е Д Е Р А Ц И И

Ф Е Д Е Р А Л Ь Н О Е Г О С У Д А Р С Т В Е Н Н О Е
А В Т О Н О М Н О Е О Б Р А З О В А Т Е Л Ь Н О Е
У Ч Р Е Ж Д Е Н И Е

В Ы С Ш Е Г О П Р О Ф Е С С И О Н А Л Ь Н О Г О
О Б Р А З О В А Н И Я

«Н А Ц И О Н А Л Ь Н Ы Й И С С Л Е Д О В А Т Е Л Ь С К И Й
У Н И В Е Р С И Т Е Т

“В Ы С Ш А Я Ш К О Л А Э К О Н О М И К И”»

TELECOM CHURN ANALYSIS

Homework Project “2018/2019

“Churn” team:
Alexandr Andreev
Arthur Akbarov
Ivan Terentev

MSc Program “Data Science”
1st Year

Moscow 2018

Table of contents:

Assignment 1-1. **4**

Dataset description 4

Assignment 1-2: K-means. **6**

Data preparation 6

K-means clustering with chosen features: 6

Clusters interpretation 7

Assignment 2: Bootstrap for cluster interpretation. **9**

Comparing two clusters. 9

Confidence interval for grandmean. 10

Comparing grand mean with cluster mean. 10

Assignment 3: Contingency table **11**

Building nominal features from numerical 11

Contingency table of voicemail message number (Vk) and total international minutes(Il): 12

Relative contingency table for voicemail message number (Vk) and total international minutes(Il): 13

Relative contingency table for voicemail message number (Vk) and Churn(Cl): 13

Quetelet indices q(Vk | Ll) %: 13

Quetelet indices for voicemail category vs churn q(Vk | Cl) %: 14

Chi-square / Summary Quetelet Index 14

Desired Number of Observations 14

Assignment 4: PCA/SVD **16**

PCA. 16

Hidden factor. 18

Assignment 5: Linear Regression **18**

Regression parameters 18

Prediction 18

Analyze 19
Code. **19**
Assignment 1. 19
Assignment 2. 20
Assignment 3. 20
Assignment 4. 21
Assignment 5. 22

Assignment 1-1.

Dataset description

This dataset describes customers of the Telco telecommunication company in USA. A telecommunication company is concerned about the number of customers leaving their landline business for cable competitors. They need to understand who is leaving and why.

This dataset is good for our task because it has lots of numerical features and no NaNs.

Original dataset had 3333 records. For our task we have chosen random sample of 300 records.

Features:

- Personal customer data: *state, area code, phone number*
- Amount of minutes the customer spent on conversations splitted by periods (day, evening, night)
- Total charge amount for conversations
- Tariff plan type(international/country)
- Customers who left within the last month – the column is called Churn

state	account length	area code	phone number	international plan	voice mail plan	number vmail messages	total day minutes	total day calls	total day charge	total eve minutes	total eve calls	total eve charge	total night minutes	total night calls	total night charge	total intl minutes	total intl calls	total intl charge	customer service calls	churn
KS	128	415	382-4657	no	yes	25	265.1	110	45.07	197.4	99	16.78	244.7	91	11.01	10	3	2.7	1	False
OH	107	415	371-7191	no	yes	26	161.6	123	27.47	195.5	103	16.62	254.4	103	11.45	13.7	3	3.7	1	False
NJ	137	415	358-1921	no	no	0	243.4	114	41.38	121.2	110	10.3	162.6	104	7.32	12.2	5	3.29	0	False
OH	84	408	375-9999	yes	no	0	299.4	71	50.9	61.9	88	5.26	196.9	89	8.86	6.6	7	1.78	2	False
OK	75	415	330-6626	yes	no	0	166.7	113	28.34	148.3	122	12.61	186.9	121	8.41	10.1	3	2.73	3	False
AL	118	510	391-8027	yes	no	0	223.4	98	37.98	220.6	101	18.75	203.9	118	9.18	6.3	6	1.7	0	False
MA	121	510	355-9993	no	yes	24	218.2	88	37.09	348.5	108	29.62	212.6	118	9.57	7.5	7	2.03	3	False
MO	147	415	329-9001	yes	no	0	157	79	26.69	103.1	94	8.76	211.8	96	9.53	7.1	6	1.92	0	False
LA	117	408	335-4719	no	no	0	184.5	97	31.37	351.6	80	29.89	215.8	90	9.71	8.7	4	2.35	1	False
WV	141	415	330-8173	yes	yes	37	258.6	84	43.96	222	111	18.87	326.4	97	14.69	11.2	5	3.02	0	False
IN	65	415	329-6603	no	no	0	129.1	137	21.95	228.5	83	19.42	208.8	111	9.4	12.7	6	3.43	4	True
RI	74	415	344-9403	no	no	0	187.7	127	31.91	163.4	148	13.89	196	94	8.82	9.1	5	2.46	0	False
IA	168	408	363-1107	no	no	0	128.8	96	21.9	104.9	71	8.92	141.1	128	6.35	11.2	2	3.02	1	False
MT	95	510	394-8006	no	no	0	156.6	88	26.62	247.6	75	21.05	192.3	115	8.65	12.3	5	3.32	3	False
IA	62	415	366-9238	no	no	0	120.7	70	20.52	307.2	76	26.11	203	99	9.14	13.1	6	3.54	4	False
NY	161	415	351-7269	no	no	0	332.9	67	56.59	317.8	97	27.01	160.6	128	7.23	5.4	9	1.46	4	True
ID	85	408	350-8884	no	yes	27	196.4	139	33.39	280.9	90	23.88	89.3	75	4.02	13.8	4	3.73	1	False
VT	93	510	386-2923	no	no	0	190.7	114	32.42	218.2	111	18.55	129.6	121	5.83	8.1	3	2.19	3	False
VA	76	510	356-2992	no	yes	33	189.7	66	32.25	212.8	65	18.09	165.7	108	7.46	10	5	2.7	1	False

From initial dataset we thrown out these features: state, account length, area code, phone number, international plan, number vmail messages, total day/eve/night calls, total day/eve/night charge and customer service calls.

Assignment 1-2: K-means.

Data preparation

For clustering objects with K-means, we have chosen 4 numeric features and normalized them. Each feature was centered by its mean and normalized by its range. With the formula:

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}}$$

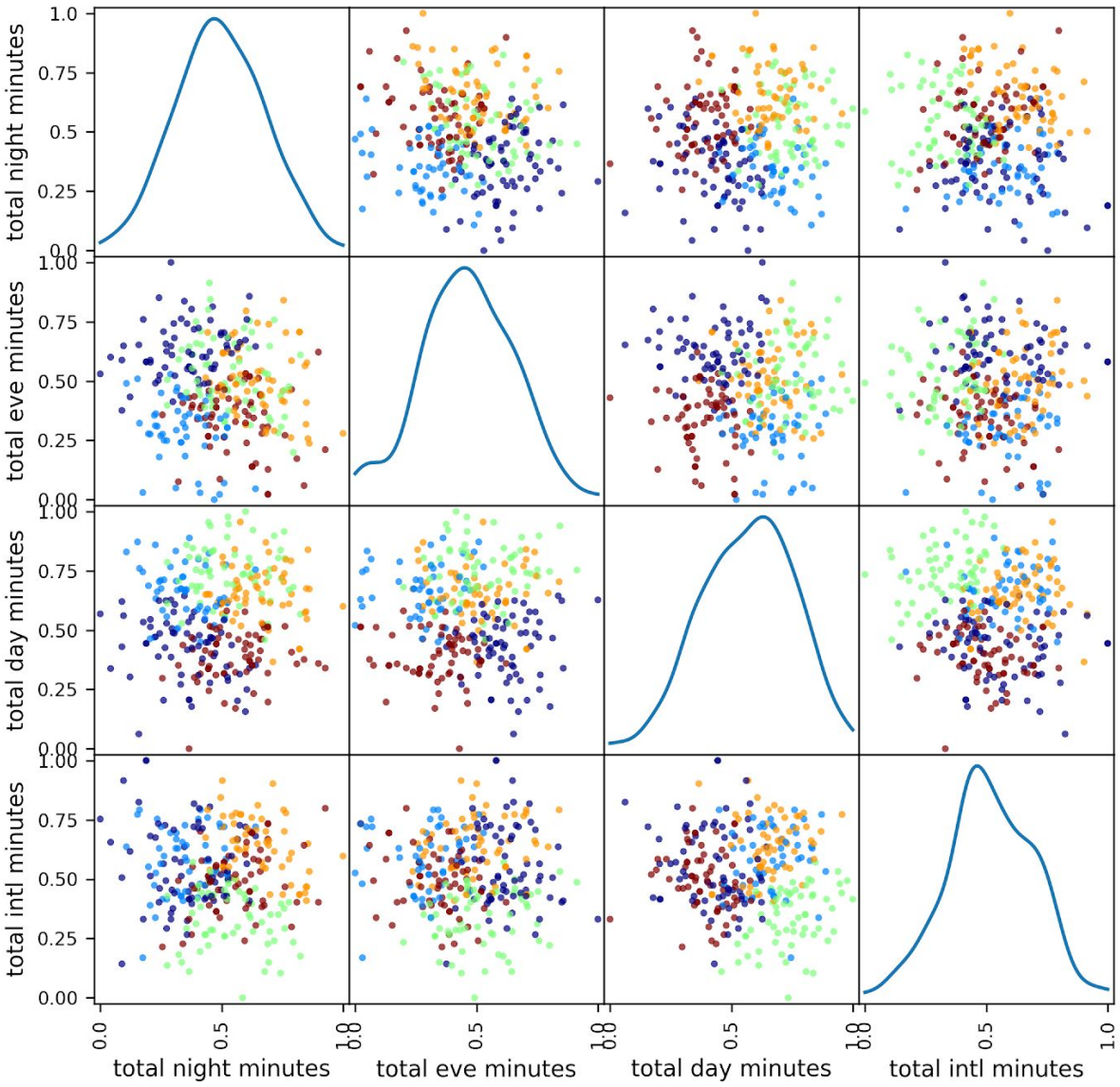
Selected features are:

1. total day minutes (count of minutes in daily calls)
2. total eve minutes (count of minutes in evening calls)
3. total night minutes (count of minutes in night calls)
4. total intl minutes (count of minutes in international calls)

These features are numerical and best reflect the characteristics of the customers for task of churn prediction.

K-means clustering with chosen features:

We clustered the data using K-means method to 5 clusters. The algorithm was ran 10 times with random initial group centers. The best run over the K-means criterion was chosen.

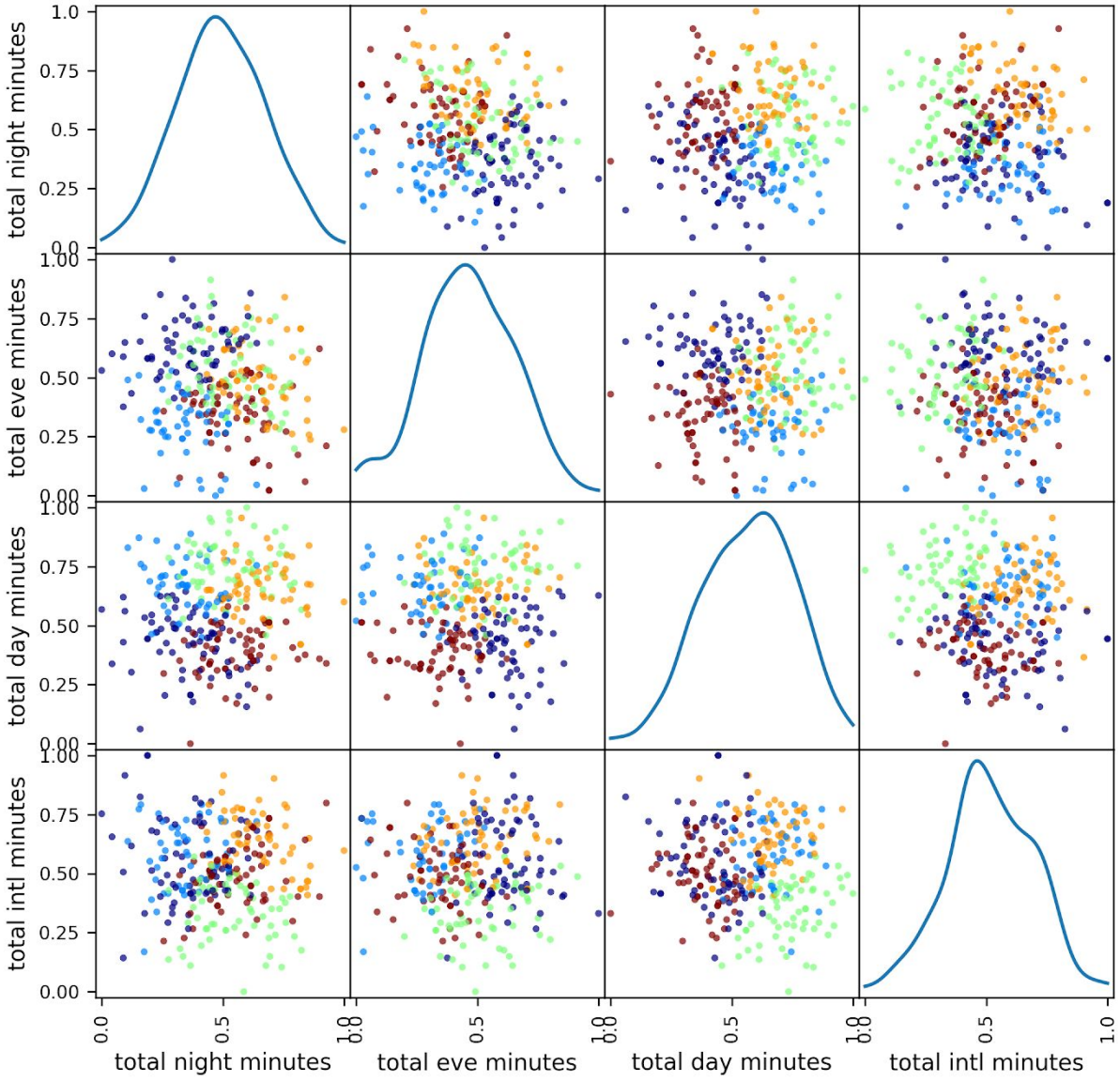


K-means clustering with 5 groups

Clusters interpretation

1. Dark blue group of customers prefer to call in the evening and less at night.
2. Light blue group call less at night, but more international calls at day time.
3. Customers from Light Green cluster prefer not to make international calls and more calls at day time
4. Brown cluster call less at day time and in the evening
5. Tawny are different because they make international calls at night.

The clusters are not perfectly divided. Looks like customers consumer profiles are similar to each other.



K-means clustering with K=9

Clusters here are less separable. We can see green one of customers that make less international call and prefer to do it at the daytime.

The brown one group shows customers who call less at daytime and evening.

Assignment 2: Bootstrap for cluster Interpretation.

Comparing two clusters.

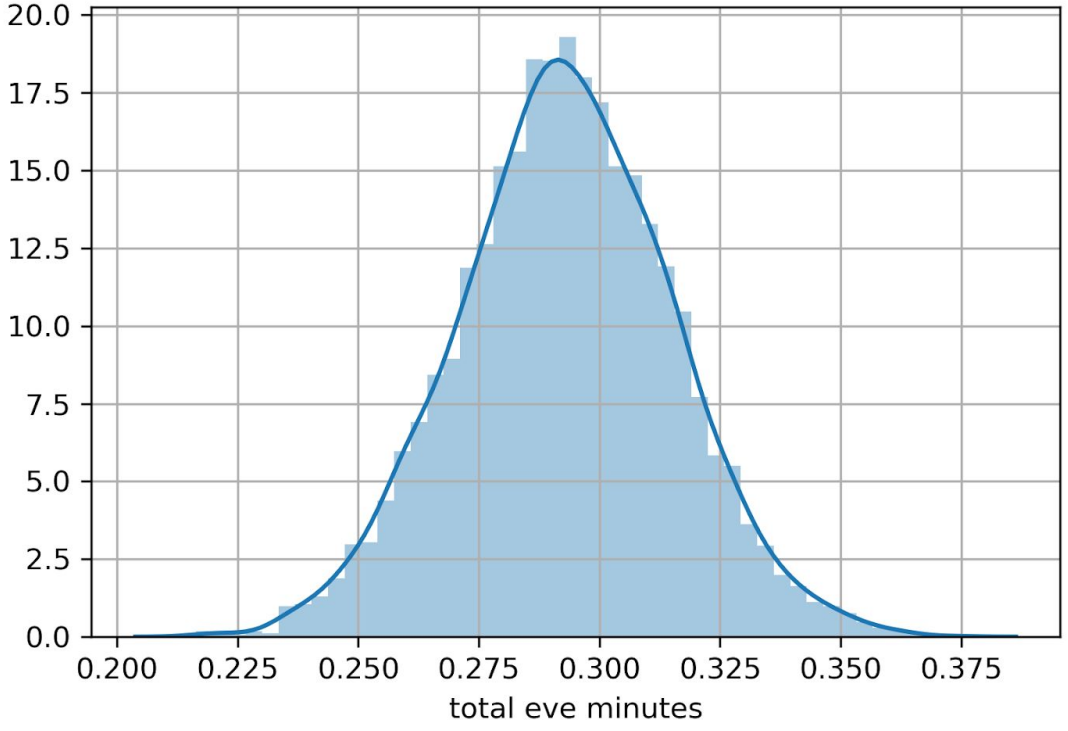
In this part we compare one of the features between two clusters of five with using bootstrap. Chosen feature is " total eve minutes", because it is giving the best divides in clusterization. 5000 samples were made for cluster one and five.

Difference mean: 0.29291160401130106,

Pivotal interval: (0.24957035144177947, 0.33625285658082266),

Non-pivotal interval: (0.24922170807717023, 0.3368381097893703).

Since the boundaries of the intervals are very close, we are limited to one histogram.



We see that difference of normalized "total eve minutes" between cluster one and five greater than zero. So we can accept the hypothesis that "total eve minutes" in first cluster are greater than in fifth cluster.

Confidence interval for grandmean.

In this part of the task we find the 95% confidence interval for "total eve minutes" grandmean by using bootstrap.

Pivotal interval (0.4428175446248513, 0.4849198129101668),

Non-pivotal interval (0.4430085154061624, 0.4844099047619044).

You can see that the boundaries of the intervals differ in 3 decimal places.

Comparing grand mean with cluster mean.

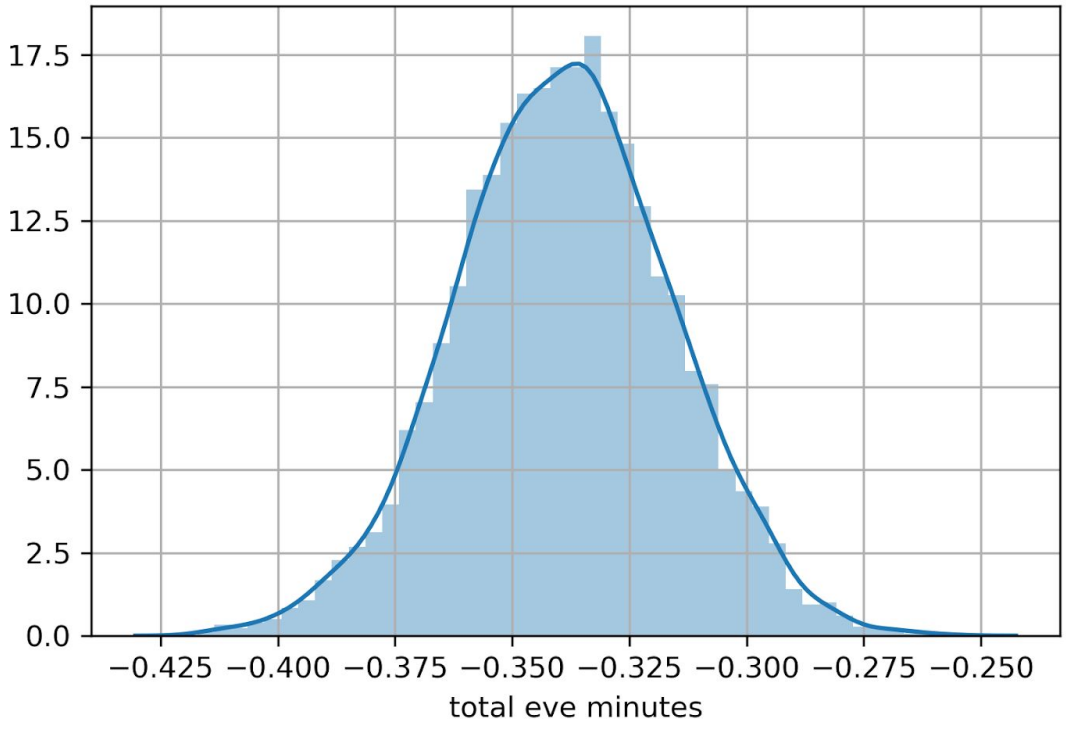
We take first cluster and compare "eve total minutes" mean in him with grandmean by using bootstrap. The following values were obtained:

Difference mean: -0.33901632433080214,

Pivotal interval: (-0.3842331455119954, -0.2937995031496089),

Non-pivotal interval: (-0.3857763133060708, -0.29475263559969433),

and histogram:



We can accept the hypothesis that "total eve minutes" mean in first cluster is less than grand mean.

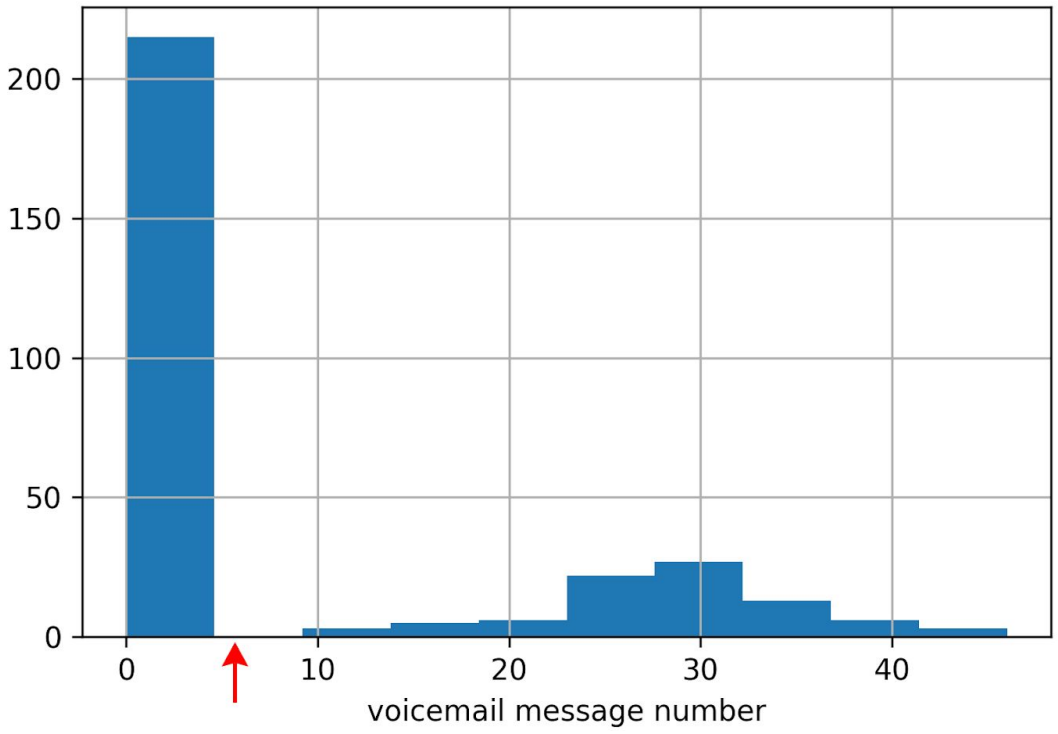
Assignment 3: Contingency table

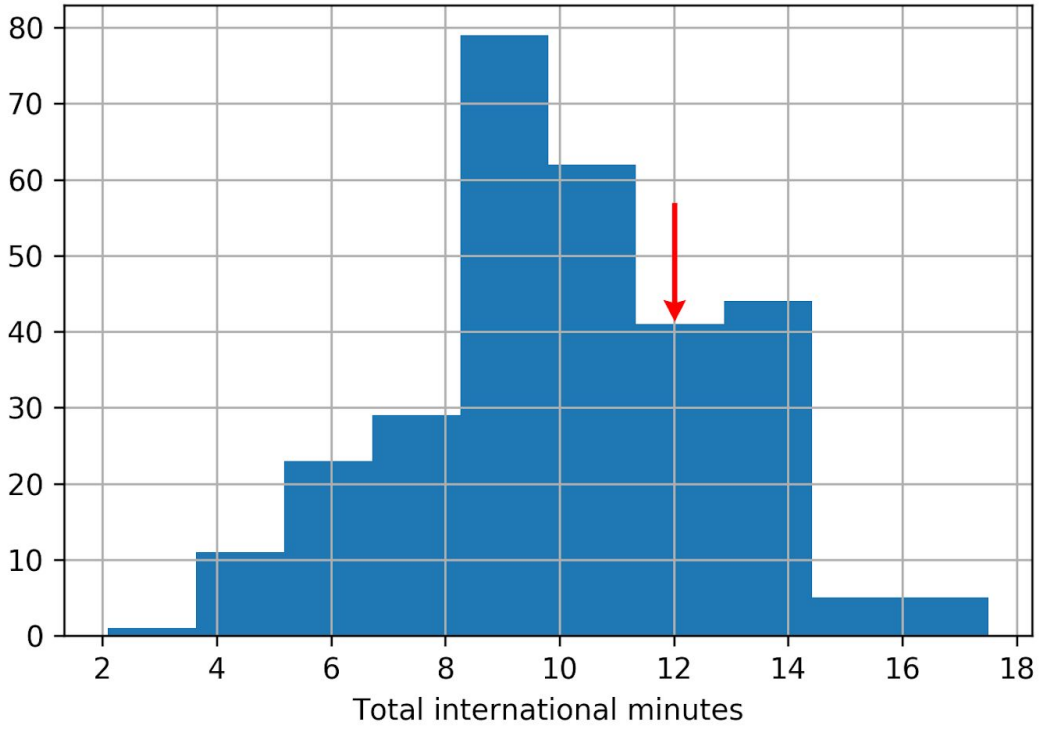
Building nominal features from numerical

For *voicemail message number* we use set

>> a = [0, 5, 50]

of 3 border points in which 0 and 50 mark the end points





Categories of *Total international minutes* can be defined by breal and end points

>> b = [2, 12, 18]

The dataset already has nominal feature *churn* with values True or False.

Contingency table of *voicemail message number (Vk)* and *total international minutes(Il):*

number vmail messages category	total international minutes category		Total
	L1	L2	
V1	161	54	215
V2	59	26	85
Total	220	80	300

It is more comfortable to make decisions on relative occurrence and feature dependencies on relative contingency table. Lets build it.

Relative contingency table for voic*email message number* (Vk) and *total international minutes*(Il):

number vmail messages category	total international minutes category		Total
	L1	L2	
V1	0.536667	0.180000	0.716667
V2	0.196667	0.086667	0.283334
Total	0.733334	0.266667	1.0

Pair of features L1 and V2 is the most probable. But it doesn't give us much information on dependencies between features because these features are common even without knowing about the other feature.

Relative contingency table for voic*email message number* (Vk) and *Churn*(Cl):

number vmail messages category	Churn category		Total
	C1(False)	C2(True)	
V1	0.573333	0.143333	0.716667
V2	0.263333	0.020000	0.283333
Total	0.836667	0.163333	1.0

Quetelet indices q(Vk | Ll) %:

number vmail messages category	Quetelet 100*q(Vk | Ll)		Prob p(Vk)
	L1	L2	
V1	2.11	-5.81	0.71667
V2	-5.35	14.71	0.28333
Prob p(Li)	0.73333	0.26667	1.0

Quetelet index table gives us the information about feature dependencies in more obvious manner. As we can see V2 given L2 is 14.71% more frequent than on average. Looks like these features are not highly dependant.

Quetelet indices for voicemail category vs churn q(Vk | Cl) %:

number vmail messages category	Quetelet 100*q(Vk | Cl)		Prob p(Ll)
	C1(False)	C2(True)	
V1	-4.38	22.45	0.71667
V2	11.09	-56.78	0.28333
Prob p(Vk)	0.836667	0.163333	1.0

As we can see if customer has churned, it is 56.78% less frequent called enough than on average. Looks like these features are slightly dependant.

Chi-square / Summary Quetelet Index

Let's calculate the value of X^2 (or summary Quetelet index Q)

Chi-square for voicemail message number (Vk) and total international minutes

```
print('X^2 for number vmail messages and total international minutes:
{:.4f}'.format(chi_square(telcom_nume2, 'number vmail messages',
'total intl minutes')))
```

X^2 for *number vmail messages* and *total intl minutes*: 0.0031

We can see that the possible dependence between the values is very small.

```
print('X^2 for number vmail messages and churn:
{:.4f}'.format(chi_square(telcom_nume2, 'number vmail messages',
'churn')))
```

X^2 for number vmail messages and churn: 0.0249

On the other hand possible dependance between voicemail messages and churn is much bigger.

Desired Number of Observations

Pearson: Under the hypothesis that the features are independent in the population, and entity sampling has been done randomly and independently, the density function of random variable NX^2 tends to distribution X^2 with (K-1)(L-1) degrees of freedom.

In our cases K = 2, L=2, therefor f = 1. At f=1 there is a 5% chance that the NX^2 value will be greater than 3.84 if the hypothesis if independence is true and 1% chance that NX^2 value will be greater than 6.63

```
for confidence in [0.95, 0.99]:
     print(scs.chi2(df=1).ppf(confidence))
3.841458820694124
6.6348966010212145
```

We need to find such N that NX^2 would be greater than specified values, so we will calculate $\frac{3.84}{X^2}$ for 5% and $\frac{6.63}{X^2}$ for 1 %.

number vmail messages and *total intl minutes* are associated with confidence 0.95

when N >= 1927.1

number vmail messages and *total intl minutes* are associated with confidence 0.99

when N >= 2962.4

number vmail messages and *churn* are associated with confidence 0.95

when N >= 240.8

number vmail messages and *churn* are associated with confidence 0.99

when N >= 370.1

In out case N=300. So we can state that we have not enough data to say that *number vmail messages* and *total intl minutes* are dependent even with 0.95 confidence not to mention about confidence 0.99.

For *number vmail messages* and *churn* we can say with 0.95 confidence that that they are dependant. But it is not enough data to state it with 0.95 confidence.

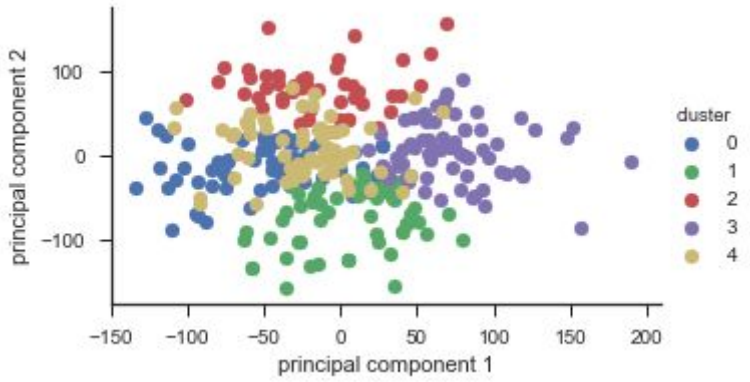
Assignment 4: PCA/SVD

PCA.

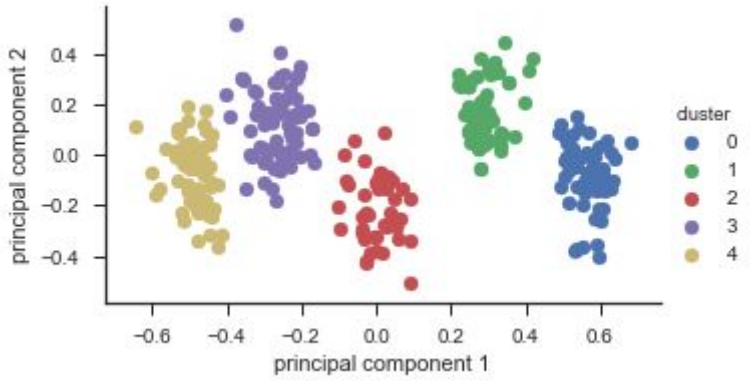
We have selected four features for this task: 'total night minutes', 'total eve minutes', 'total day minutes', 'total intl minutes'. We chose them because we believe that these features can completely describe an application from the business point of view.

normalization	over standard deviations		over ranges	
datascatter	1413.7		469.8	
1 principal component	531.3	37.5 %	408.3	86.9 %
2 principal component	327.4	23.1 %	41.1	8.7 %
3 principal component	286.2	20.2 %	10.4	2.2 %
4 principal component	268.7	19.0 %	9.9	2.1 %

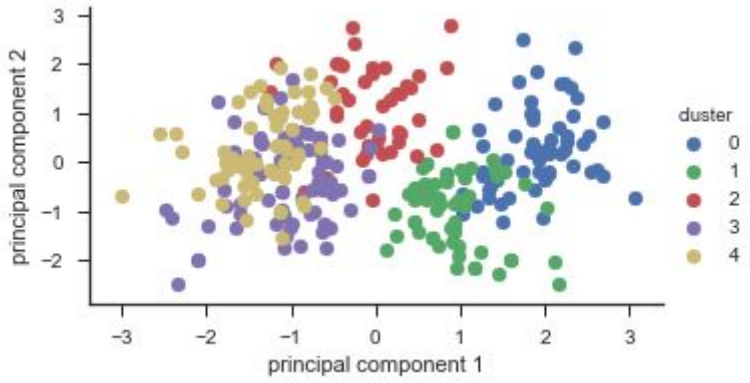
Scatterplot on X



Scatterplot on X normalized over standard deviation



Scatterplot on X normalized over range normalization



Hidden factor.

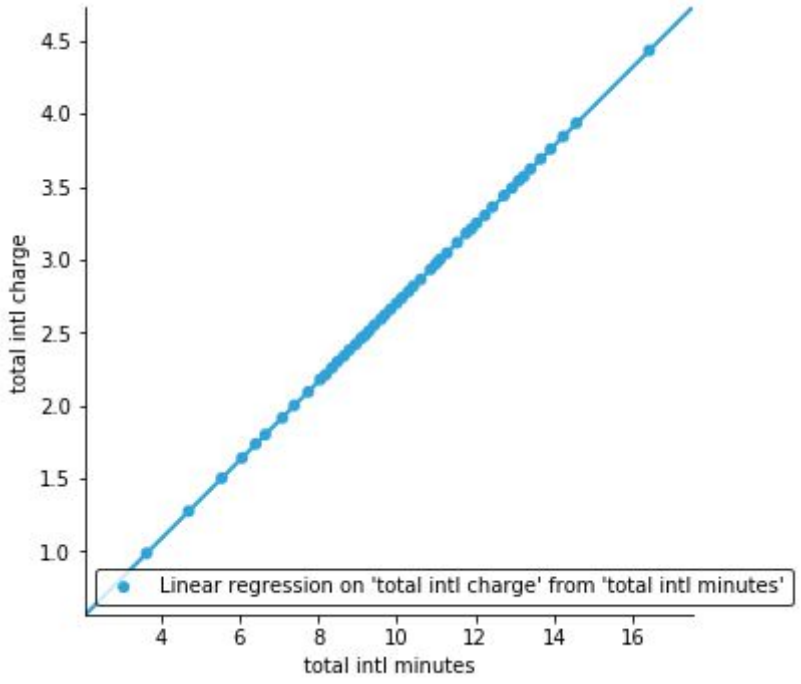
We rescale all the features to [0, 100] and then decompose the result using SVD.

Contribution of first component is 91.434%.

Hidden factor was found [-0.48018844, -0.45225169, -0.55853132, -0.50292169] .

Assignment 5: Linear Regression

We now consider how we could predict “total intl charge” if we knew “total intl minutes”.



A scatter plot of the example data. The line consists of the predictions, the points are the actual data, and the vertical lines between the points and the black line represent errors of prediction.

Regression parameters

slope = 0.26996178210203986
intercept = 0.0008555803725198707
r2 = 0.9999844336384671
correlation coefficient = 0.9999922167889443
mean relative absolute error = 0.0009980451256663588

Prediction

Let predict sample on three values: 10, 20, 40.

10 -> 2.7004734013929186
20 -> 5.400091222413318
40 -> 10.799326864454114

It’s clear that “total intl charge” linear depends on “total intl minutes”.

Analyze

We have very linear data dependence, so the determinacy coefficient is almost equals to 1 and mean relative absolute error is almost equals to zero.

Code.

In our work we used these PYTHON libraries:

- ★ numpy
- ★ pandas
- ★ matplotlib
- ★ seaborn
- ★ sklearn
- ★ scipy

Assignment 1.

Read and transform dataset

```
telcom = pd.read_csv(r"bigml_59c28831336c6604c800002a.csv")
telcom = telcom.sample(n=300, random_state=17, replace=True)
telcom_numeric = telcom[['total night minutes', 'total eve minutes',
'total day minutes', 'total intl minutes']]
```

Scale it for K-Means

```
telcom_numeric_scaled =
((telcom_numeric-telcom_numeric.min())/(telcom_numeric.max()-telcom_n
umeric.min()))
```

Apply K-Means algorithm from sklearn library.

```
kmeans = KMeans(n_clusters=5, init="random", n_init=10,
random_state=17)
kmeans.fit(telcom_numeric_scaled)
```

```
>>
```

```
KMeans(algorithm='auto', copy_x=True, init='random', max_iter=300,
    n_clusters=5, n_init=10, n_jobs=None,
```

```
precompute_distances='auto',
    random_state=17, tol=0.0001, verbose=0)
```

Code to visualize K-Means result with Scatter Plot with numerical features.

```
y_kmeans = kmeans.labels_
scatter_matrix(telcom_numeric_scaled,
                alpha = 0.7, figsize = (8, 8), diagonal = 'kde',
c=y_kmeans, cmap="jet")
plt.savefig("k-means5.png", dpi=300)
plt.show()
```

Assignment 2.

```
def mean_std(data):
    return data.mean(), data.std()

def comp(data, labels, cluster, feature, n_samples=5000, alpha=0.05):
    m1 = bootstrap(data, labels, cluster[0], feature, n_samples)
    if len(clusters) == 1:
        m2 = bootstrap(data, np.ones(data.shape[0]), 1, feature, n_samples)
    else:
        m2 = bootstrap(data, labels, clusters[1], feature, n_samples)

    d = m1 - m2
    q = m1 / m2
    sns.distplot(m1 - m2)

    d_mean, d_std = mean_std(d)
    q_mean, q_std = mean_std(q)

    quantile = sc.stats.norm.ppf(1 - alpha / 2)
    d_piv = d_mean - quantile * d_std, d_mean + quantile * d_std
    q_piv = q_mean - quantile * q_std, q_mean + quantile * q_std

    d_non = mquantiles(d, alpha / 2)[0], mquantiles(d, 1 - alpha / 2)[0]
    q_non = mquantiles(q, alpha / 2)[0], mquantiles(q, 1 - alpha / 2)[0]
    return {'Dif': {'mean': d_mean, 'piv': d_piv, 'non': d_non},
        'Quo': {'mean': q_mean, 'piv': q_piv, 'non': q_non}}

def non(data, labels, cluster, feature, n_samples=5000, alpha=0.05):
    means = bootstrap(data, labels, cluster, feature, n_samples)
    mean = means.mean()

    lb = mquantiles(means, alpha / 2)[0]
```

```
    rb = mquantiles(means, 1 - alpha / 2)[0]
    return ({'mean': mean, 'boundaries': (lb, rb)})


def bootstrap(data, labels, cluster, feature, n_samples=5000):
    X = pd.DataFrame(data[labels == cluster][feature]\
                     .sample(frac=n_samples, replace=True))
    X['sample'] = np.repeat(range(n_samples), (labels == cluster).sum())

    means = X.groupby('sample')[feature].mean()
    return means

def piv(data, labels, cluster, feature, n_samples=5000, alpha=0.05):
    means = bootstrap(data, labels, cluster, feature, n_samples)
    mean, std = mean_std(means)

    quantile = sc.stats.norm.ppf(1 - alpha / 2)
    lb = mean - quantile * std
    rb = mean + quantile * std
    return ({'mean': mean, 'boundaries': (lb, rb)})
```

Assignment 3.

Code to convert numerical features to nominal ones.

```
def vmail_quantize(value):
     return 0 if value < 5 else 1
def intl_minutes_quantize(value):
     return 0 if value < 12 else 1

vmail = telcom_nume['number vmail messages'].apply(vmail_quantize)
intl_minutes = telcom_nume['total intl
minutes'].apply(intl_minutes_quantize)
churn = telcom_nume['churn']
```

Code to build contingency table.

```
pd.crosstab(vmail, intl_minutes)/300
pd.crosstab(vmail, churn)/300
```

Code to calculate X^2

```
def chi_square(df, vert, horz):
      ver_column, hor_column = df[vert], df[horz]
      result = 0
      for ver_value in sorted(ver_column.unique()):
      ver_equal = ver_column == ver_value
      for hor_value in sorted(hor_column.unique()):
            horz_equal = hor_column == hor_value
            p_vk = ((ver_equal) &
                  (horz_equal)).mean()
            p_k = (ver_equal).mean()
            p_v = (horz_equal).mean()
            result += (p_vk - p_k * p_v)**2 / (p_k * p_v)
      return result
```

```
Code to calculate Desired Number of Observations for Pearson criterion
feat1 = 'number vmail messages'
for feat2in ['total intl minutes', 'churn']:
      for confidence in [0.95, 0.99]:
      min_N = (scs.chi2(df=2).ppf(confidence) /
                  chi_square(telcom_nume2, feat1, feat2)[0])
      print('{} and {} are associated with confidence {:.2f}\n'
'\twhen N >= {:.1f}'.format(
      feat1, feat2, confidence, min_N))
```

Assignment 4.

```
#contributions of components
P, s, Q = np.linalg.svd(X_scaled , full_matrices=True)
con=np.zeros (shape =[2 ,5])
con[0 ,1]= s[0]* s[0]
con[0 ,2]= s[1]* s[1]
con[0 ,3]= s[2]* s[2]
con[0 ,4]= s[3]* s[3]
datascat_scaled = con[0 ,1] + con[0 ,2] + con[0 ,3] + con[0 ,4]
print('over range', datascat_scaled)
con[1 ,1]= s[0]* s[0]*100/ datascat_scaled
con[1 ,2]= s[1]* s[1]*100/ datascat_scaled
con[1 ,3]= s[2]* s[2]*100/ datascat_scaled
con[1 ,4]= s[3]* s[3]*100/ datascat_scaled
meaning=['contribution of component,  naturally', 'contribution of component, per cent']
contribution=DataFrame(data=con, columns=['','pc1','pc2','pc3','pc4'])
contribution['']=meaning
contribution
```

```
#contributions of components
P, s, Q = np.linalg.svd(X_std , full_matrices=True)
con=np.zeros (shape =[2 ,5])
con[0 ,1]= s[0]* s[0]
con[0 ,2]= s[1]* s[1]
con[0 ,3]= s[2]* s[2]
con[0 ,4]= s[3]* s[3]
datascat_std = con[0 ,1] + con[0 ,2] + con[0 ,3] + con[0 ,4]
print('over deviation', datascat_std)
con[1 ,1]= s[0]* s[0]*100/ datascat_std
con[1 ,2]= s[1]* s[1]*100/ datascat_std
con[1 ,3]= s[2]* s[2]*100/ datascat_std
con[1 ,4]= s[3]* s[3]*100/ datascat_std
meaning=['contribution of component,  naturally', 'contribution of component, per cent']
contribution=DataFrame(data=con, columns=['','pc1','pc2','pc3','pc4'])
contribution['']=meaning
contribution
```

```
from sklearn.decomposition import PCA
from  matplotlib import pyplot
import seaborn
seaborn.set(style='ticks')

pca = PCA(n_components=2)
principalComponentsX = pca.fit_transform(X)
principalComponentsX_scaled = pca.fit_transform(X_scaled)
principalComponentsX_std = pca.fit_transform(X_std)
principalX = DataFrame(data = principalComponentsX, columns=['principal component 1', 'principal component 2'])
principalX_scaled = DataFrame(data = principalComponentsX_scaled, columns=['principal component 1', 'principal component 2'])
principalX_std = DataFrame(data = principalComponentsX_std, columns=['principal component 1', 'principal component 2'])
principalX['cluster'] = X['cluster']
principalX_scaled['cluster'] = X['cluster']
principalX_std['cluster'] = X['cluster']

fg = seaborn.FacetGrid(data=principalX, hue='cluster', aspect=1.61)
fg.map(pyplot.scatter, 'principal component 1', 'principal component 2').add_legend()
fg = seaborn.FacetGrid(data=principalX_scaled, hue='cluster', aspect=1.61)
fg.map(pyplot.scatter, 'principal component 1', 'principal component 2').add_legend()
fg = seaborn.FacetGrid(data=principalX_std, hue='cluster', aspect=1.61)
fg.map(pyplot.scatter, 'principal component 1', 'principal component 2').add_legend()
```

```
U, s, V = sla.svd(rescaled_pca_data)
z=V[0, :]
alpha=1/sum(z)
z=z*alpha
hidden factor=rescaled_pca_data.dot(z)
```

Assignment 5.

```
%%output size=150
from sklearn.linear_model import LinearRegression

def simple_reg(predictor, target):
    '''вспомогательная функция, отрисовка прямой для простой регресии'''
    check = LinearRegression()
    check.fit(predictor.reshape(-1,1),target)
    slope = check.coef_
    intercept = check.intercept_
    print("slope =", check.coef_[0])
    print("intercept =", check.intercept_)
    r2 = check.score(predictor.reshape(-1,1),target)
    print('R2 =', r2)
    print('correlation coefficient =', np.sqrt(r2))
    array = (abs(1 - target / (slope*predictor + intercept))).mean()
    print('mean relative absolute error =', array)
    print('10 ->', check.intercept_ + check.coef_[0]*10)
    print('20 ->', check.intercept_ + check.coef_[0]*20)
    print('40 ->', check.intercept_ + check.coef_[0]*40)
    return hv.Curve((np.array([i for i in np.linspace(min(predictor),max(predictor),100)]),
                     check.coef_*np.array([i for i in np.linspace(min(predictor),
                                                                  max(predictor),100)]) + check.intercept_))

def make_bucket(df,feature, n =  50):
    '''функция, бьющая на бакеты (по умолчанию 100 точек)'''
    return df.assign(bucket = np.ceil(df[feature].rank(pct = True) * n))

df = data[list(data.select_dtypes(['number']))]

plt.rcParams["figure.figsize"] = (200,200)

featureX = 'total intl minutes'
featureY = 'total intl charge'
df.pipe(make_bucket, featureX)\
  .groupby(by = ['bucket']).mean()\
  .pipe(lambda x:hv.Scatter(zip(np.array(x[featureX]), np.array(x[featureY])),
                            kdims = [featureX],vdims=[featureY],
                            label = f"Linear regression on '{featureY}' from '{featureX}'", )
        *simple_reg(np.array(df[featureX]),np.array(df[featureY])))
```