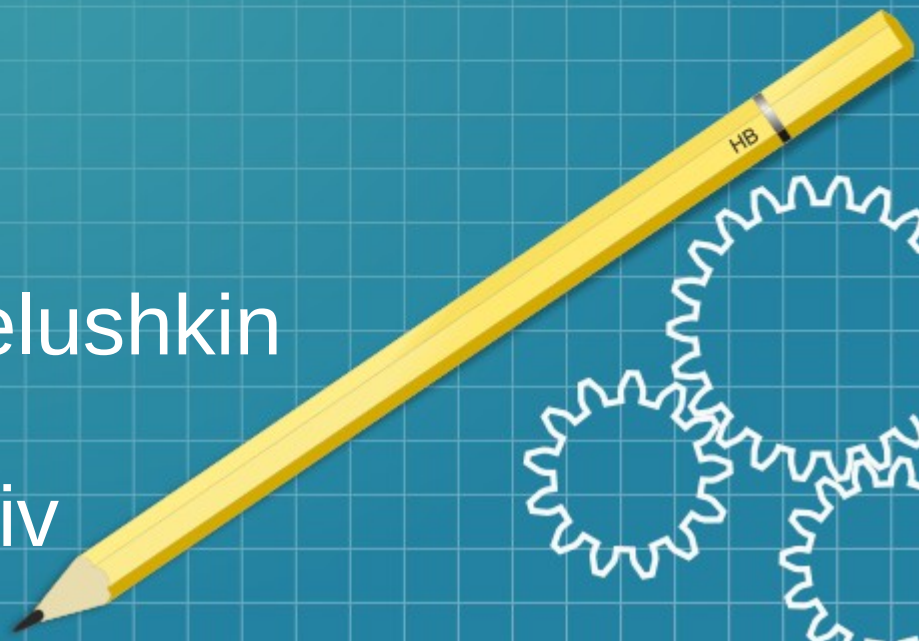
Predicting Wine Quality by using Watson Machine Learning

by Maksym Bielushkin

2019, Kyiv



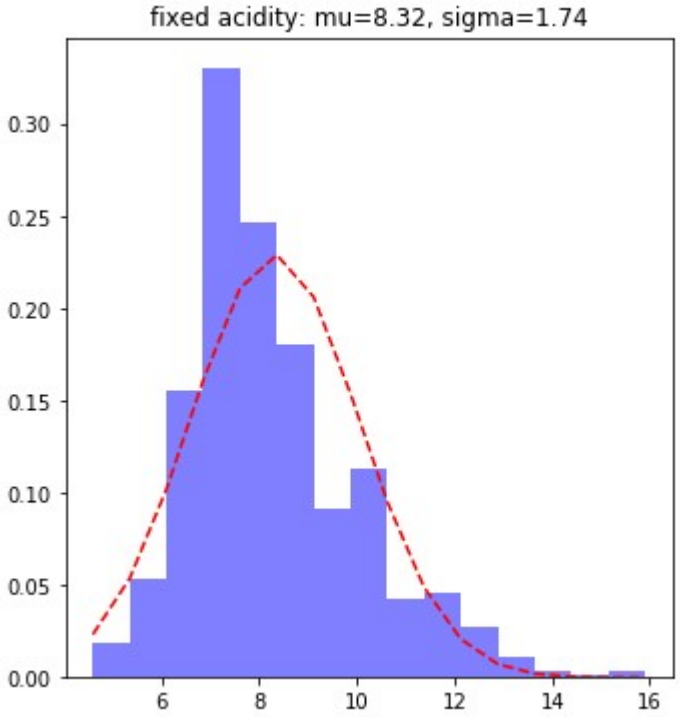
Dataset

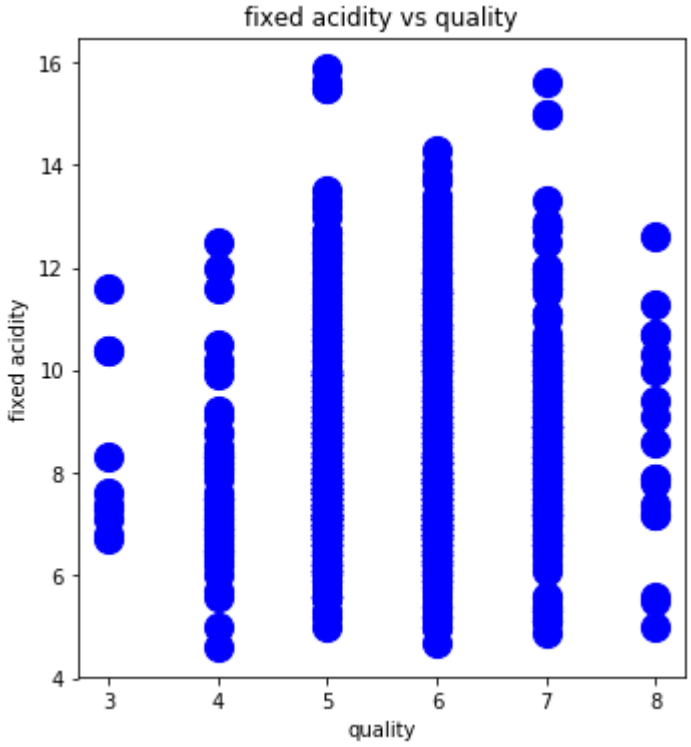
- https://www.kaggle.com/maitree/wine-quality-selection

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.0	0.27	0.36	20.7	0.045	45.0	170.0	1.0010	3.00	0.45	8.8	6
1	6.3	0.30	0.34	1.6	0.049	14.0	132.0	0.9940	3.30	0.49	9.5	6
2	8.1	0.28	0.40	6.9	0.050	30.0	97.0	0.9951	3.26	0.44	10.1	6
3	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.9956	3.19	0.40	9.9	6
4	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.9956	3.19	0.40	9.9	6

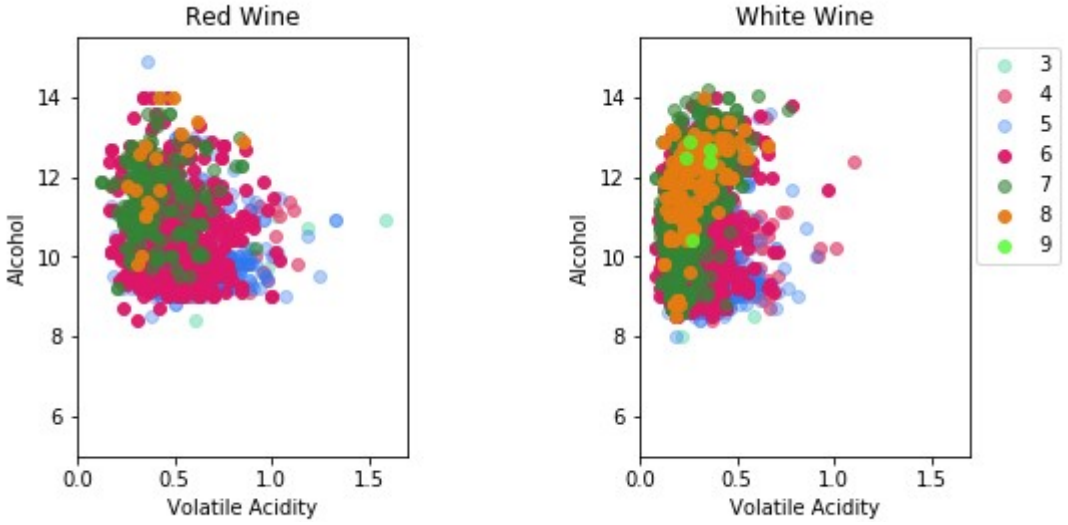
ETL step

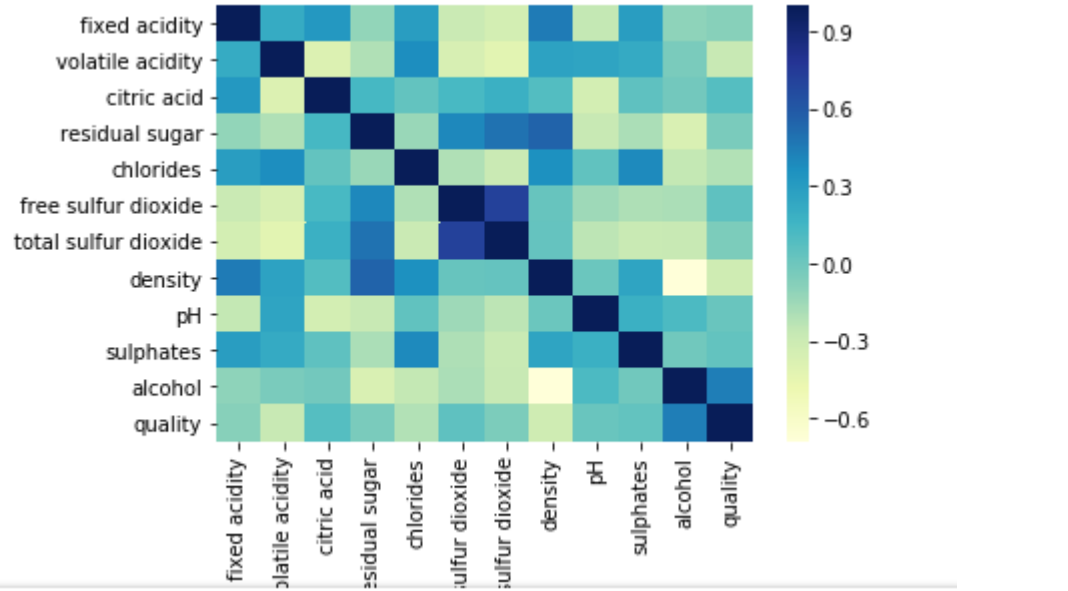
- Apache Spark, Pandas
- Matplotlib, Seaborn
- Python





ETL step


Red Wine
White Wine
Alcohol
Volatile Acidity
14
12
10
8
6
0.0
0.5
1.0
1.5
3
4
5
6
7
8
9


fixed acidity
volatile acidity
citric acid
residual sugar
chlorides
free sulfur dioxide
total sulfur dioxide
density
pH
sulphates
alcohol
quality
0.9
0.6
0.3
0.0
−0.3
−0.6

ETL step

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
count	4898.000000	4898.000000	4898.000000	4898.000000	4898.000000	4898.000000	4898.000000	4898.000000	4898.000000	4898.000000	4898.000000	4898.000000
mean	6.854788	0.278241	0.334192	6.391415	0.045772	35.308085	138.360657	0.994027	3.188267	0.489847	10.514267	5.877909
std	0.843868	0.100795	0.121020	5.072058	0.021848	17.007137	42.498065	0.002991	0.151001	0.114126	1.230621	0.885639
min	3.800000	0.080000	0.000000	0.600000	0.009000	2.000000	9.000000	0.987110	2.720000	0.220000	8.000000	3.000000
25%	6.300000	0.210000	0.270000	1.700000	0.036000	23.000000	108.000000	0.991723	3.090000	0.410000	9.500000	5.000000
50%	6.800000	0.260000	0.320000	5.200000	0.043000	34.000000	134.000000	0.993740	3.180000	0.470000	10.400000	6.000000
75%	7.300000	0.320000	0.390000	9.900000	0.050000	46.000000	167.000000	0.996100	3.280000	0.550000	11.400000	6.000000
max	14.200000	1.100000	1.660000	65.800000	0.346000	289.000000	440.000000	1.038980	3.820000	1.080000	14.200000	9.000000

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4898 entries, 0 to 4897
Data columns (total 12 columns):
fixed acidity           4898 non-null float64
volatile acidity        4898 non-null float64
citric acid             4898 non-null float64
residual sugar          4898 non-null float64
chlorides               4898 non-null float64
free sulfur dioxide     4898 non-null float64
total sulfur dioxide    4898 non-null float64
density                 4898 non-null float64
pH                      4898 non-null float64
sulphates               4898 non-null float64
alcohol                 4898 non-null float64
quality                 4898 non-null int64
dtypes: float64(11), int64(1)
memory usage: 459.3 KB
None
```

```
# Double check for null values in `red`
pd.isnull(red)
```

[7]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	fre
0	False	False	False	False	False	
1	False	False	False	False	False	
2	False	False	False	False	False	
3	False	False	False	False	False	
4	False	False	False	False	False	
5	False	False	False	False	False	
6	False	False	False	False	False	

What predicted?

- 2 types of prediction
- Quality of wine bad or good
- Wine quality itself

System ML

- Logistic Regression R2 score: 0.861742
- Quality of wine: Good or Bad is predicted (binary classification)
- Winequality-red dataset is used

https://github.com/belushkin/IBM/blob/master/SystemML.ipynb

PySpark

- Linear Regression (wine quality prediction)

  RMSE: 0.736556

  r2: 0.176016

- Winequality-red dataset is used

https://github.com/belushkin/IBM/blob/master/LinearRegression.ipynb

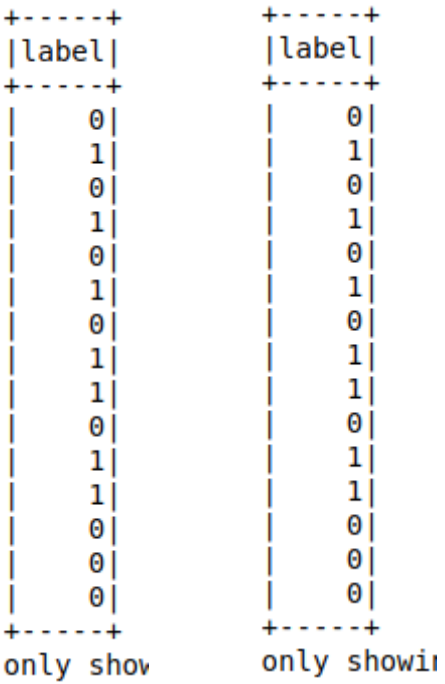
```
+-----------------+-------+--------------------+
|       prediction|quality|            features|
+-----------------+-------+--------------------+
|5.869559864798003|    6.0|[5.0,0.7400000095...|
|5.665521454793363|    5.0|[5.0,1.0399999618...|
| 6.09153656061749|    7.0|[5.09999990463256...|
|5.959804873149034|    7.0|[5.09999990463256...|
|5.775795584293244|    5.0|[5.59999990463256...|
|5.954873616514447|    5.0|[5.59999990463256...|
|5.586980062290905|    4.0|[5.59999990463256...|
|5.928305788699809|    7.0|[5.59999990463256...|
|5.683334709772041|    5.0|[5.59999990463256...|
| 5.86505764630658|    6.0|[5.69999980926513...|
|5.422212333841235|    4.0|[5.69999980926513...|
|5.672918317700854|    6.0|[5.80000019073486...|
|5.719515405404755|    6.0|[5.90000009536743...|
|5.931200447812138|    6.0|[6.0,0.4900000095...|
|5.603560521481262|    5.0|[6.0,0.5,0.039999...|
|5.909009844393404|    6.0|[6.0,0.5799999833...|
| 5.66094876484712|    5.0|[6.09999990463256...|
|5.725983625919463|    5.0|[6.09999990463256...|
|5.635184600026006|    6.0|[6.09999990463256...|
|5.763467479447425|    5.0|[6.09999990463256...|
+-----------------+-------+--------------------+
only showing top 20 rows
```

PySpark

- Gradient-Boosted Trees (GBTs)

  f1 score: 0.8901
- Quality of wine: Good or Bad is predicted (binary classification)
- Winequality-red dataset is used

https://github.com/belushkin/IBM/blob/master/GBT.ipynb



```
+-----+
|label|
+-----+
|    0|
|    1|
|    0|
|    1|
|    0|
|    1|
|    0|
|    1|
|    1|
|    0|
|    1|
|    1|
|    0|
|    0|
|    0|
+-----+
only shov
```

```
+-----+
|label|
+-----+
|    0|
|    1|
|    0|
|    1|
|    0|
|    1|
|    0|
|    1|
|    1|
|    0|
|    1|
|    1|
|    0|
|    0|
|    0|
+-----+
only showi
```

Keras

- Feed-forward neural network or Multi-Layer Perceptron is used in this lab
- Wine quality prediction:

  MSE: 0.4849

  MAE: 0.5514

  R2 score: 0.3619

https://github.com/belushkin/IBM/blob/master/Keras%204.ipynb

4.6: Compare the results

Check the predictions

```
y_pred = y_pred.astype(int)
predictions = np.column_stack((y[test], y_pred));
print(predictions[:15])
```

```
[[5 5]
 [5 4]
 [7 5]
 [5 5]
 [4 4]
 [5 5]
 [6 5]
 [5 5]
 [6 5]
 [5 5]
 [5 4]
 [5 5]
 [5 6]
 [6 6]
 [5 4]]
```

Data product example

- I decided not to set up web form since I totally reproduced this example: https://github.com/IBM/predictive-model-on-watson-ml/

  https://predictive-model-on-watson-ml-20190220141448821.mybluemix.net/

Data product example

I represent deployed model with REST API available

6.3 Invoke prediction model deployment

Define a method to call scoring url. Replace the **scoring_url** in the method below with the scoring_url returned from above.

[98]:
```
get_prediction_ml(fa, va, ca, rs, ch, fsd, tsd, d, p, s, a):
scoring_url = 'https://ibm-watson-ml.mybluemix.net/v3/wml_instances/2074d379-2141-42dc-9340-a7
scoring_payload = { "fields":["fixed acidity", "volatile acidity", "citric acid", "residual su
header = {'authorization': 'Bearer ' + watson_ml_token, 'content-type': "application/json" }
scoring_response = requests.post(scoring_url, json=scoring_payload, headers=header)
print(scoring_response.text)
return (json.loads(scoring_response.text).get("values")[0][0])
```

181]:
```
data = np.array([7, 0.27, 0.36 ,20.7 ,0.045 ,45 ,170 ,1.001 ,3 ,0.45 ,8.8]).reshape(-1, 1)
data = StandardScaler().fit_transform(data)
data.T

# print('What is quality of wine with such characteristics?: {}'.format(get_prediction_ml(0.45
```

ut[181]:
```
array([[-0.33913629, -0.47890591, -0.47703678, -0.05461267, -0.48357875,
         0.45005329,  3.04607165, -0.4637244 , -0.42220887, -0.47516765,
        -0.30175362]])
```

[ ]:

Materials

Here is all notebook are stored

- https://github.com/belushkin/IBM