
Spark
MLlib

What you will learn ?

- how Spark MLlib works
- what transformers are
- what estimators are
- how to use pipelines


Spark
MLlib


scikit
learn

DataFrame

	Col 1	Col 2	...
Row 1			
Row 2			
...			

DataFrame

transformer.transform()

DataFrame

	Col 1	Col 2	...
Row 1			
Row 2			
...			

	Col 1	Col 2	...
Row 1			
Row 2			
...			

```
from pyspark.ml.feature import VectorAssembler

assembler = VectorAssembler()\
  .setInputCols(["season",
                 "yr",
                 "mnth",
                 "holiday",
                 "weekday",
                 "weathersit",
                 "temp",
                 "atemp",
                 "hum",
                 "windspeed"])\
  .setOutputCol("features")

train01 = assembler.transform(train)
```

```
train01.limit(5).toPandas()
```

	season	yr	mnth	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	label	features
0	1	0	1	0	0	0	1	0.138	0.116	0.434	0.362	822	[1.0, 0.0, 1.0, 0.0, 0.0, 1.0, 0.138333, 0.116...
1	1	0	1	0	0	0	1	0.232	0.234	0.484	0.188	1204	[1.0, 0.0, 1.0, 0.0, 0.0, 1.0, 0.231667, 0.234...
2	1	0	1	0	0	0	2	0.363	0.354	0.696	0.249	801	[1.0, 0.0, 1.0, 0.0, 0.0, 2.0, 0.363478, 0.353...
3	1	0	1	0	1	1	1	0.097	0.118	0.492	0.158	1416	[1.0, 0.0, 1.0, 0.0, 1.0, 1.0, 0.0973913, 0.11...
4	1	0	1	0	1	1	1	0.151	0.151	0.483	0.223	1321	[1.0, 0.0, 1.0, 0.0, 1.0, 1.0, 0.150833, 0.150...

Used for:

- data preprocessing
- text processing
- feature creation
- ....

Estimator

DataFrame

	Col 1	Col 2	...
Row 1			
Row 2			
...			

estimator.fit()

Transformer

```
from pyspark.ml.regression import LinearRegression
lr = LinearRegression()
model = lr.fit(train02)
```

```
from pyspark.ml.regression import LinearRegression
lr = LinearRegression()
model = lr.fit(train02)
```

```
train03 = model.trasform(train02)
```

```
Train03.limit(5).toPandas()
```

	features	label	prediction
0	[1.0, 0.0, 1.0, 0.0, 0.0, 1.0, 0.138333, 0.116...	822	678.100
1	[1.0, 0.0, 1.0, 0.0, 0.0, 1.0, 0.231667, 0.234...	1204	1752.152
2	[1.0, 0.0, 1.0, 0.0, 0.0, 2.0, 0.363478, 0.353...	801	1389.662
3	[1.0, 0.0, 1.0, 0.0, 1.0, 1.0, 0.0973913, 0.11...	1416	1226.827
4	[1.0, 0.0, 1.0, 0.0, 1.0, 1.0, 0.150833, 0.150...	1321	1273.469

Types of learning algorithms:

- regression
- classification
- clusterisation

Types of learning algorithms:

- regression
- classification
- clasterisation
- feature processing
  - MinMaxScaler

```
from pyspark.ml.feature import MinMaxScaler
scaler = MinMaxScaler(inputCol="features", outputCol="scaled_featues")
scaler_model = scaler.fit(train02)
```

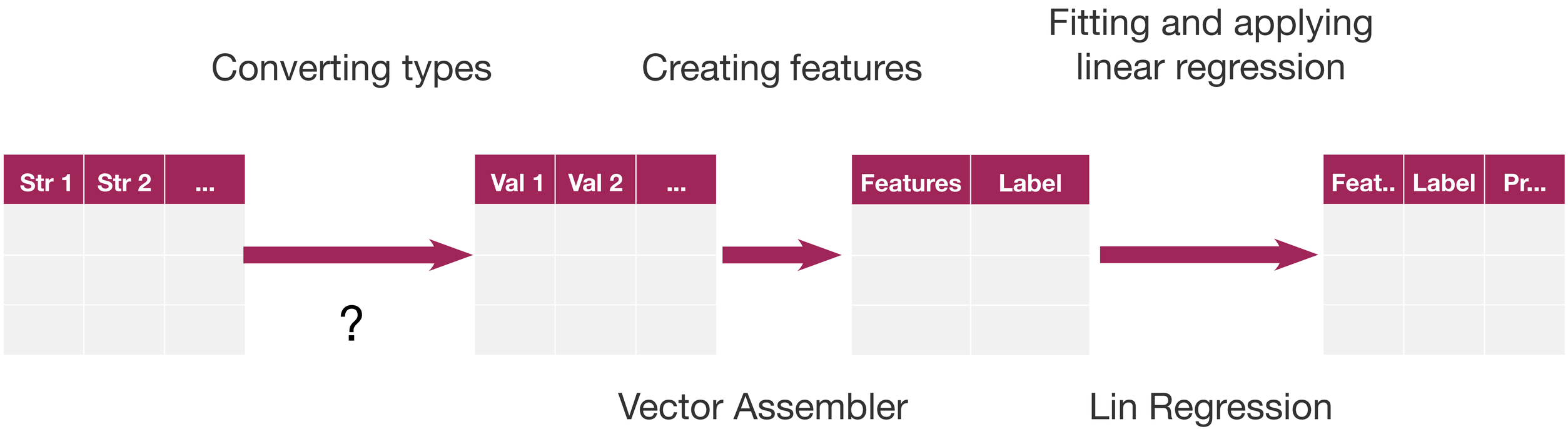
```
scaler_model.transform(train02).limit(5).toPandas()
```

	features	label	scaled_featues
0	[1.0, 0.0, 1.0, 0.0, 0.0, 1.0, 0.138333, 0.116...	822	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0986903276436...
1	[1.0, 0.0, 1.0, 0.0, 0.0, 1.0, 0.216522, 0.250...	1096	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.19611765993, ...
2	[1.0, 0.0, 1.0, 0.0, 0.0, 1.0, 0.231667, 0.234...	1204	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.214989073396,...
3	[1.0, 0.0, 1.0, 0.0, 1.0, 1.0, 0.0973913, 0.11...	1416	[0.0, 0.0, 0.0, 0.0, 0.166666666667, 0.0, 0.04...
4	[1.0, 0.0, 1.0, 0.0, 1.0, 1.0, 0.196364, 0.189...	1349	[0.0, 0.0, 0.0, 0.0, 0.166666666667, 0.0, 0.17...

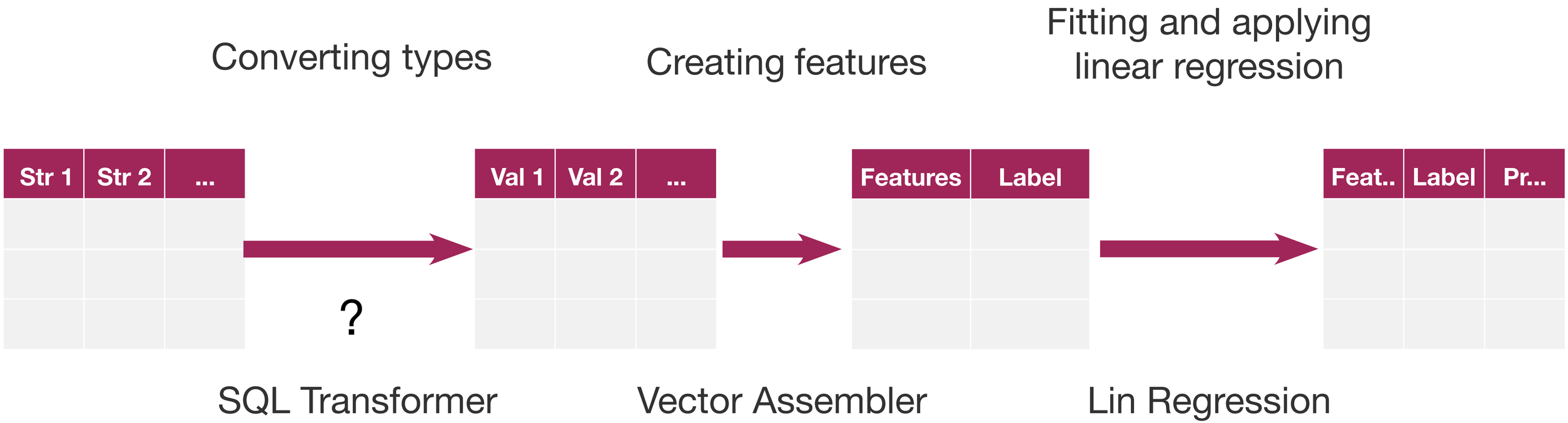
Train estimators on train dataset only!

Pipelines

Pipeline


Converting types
Creating features
Fitting and applying linear regression
Str 1
Str 2
...
?
Val 1
Val 2
...
Features
Label
Feat..
Label
Pr...
Vector Assembler
Lin Regression

Pipeline


Converting types
Creating features
Fitting and applying linear regression
Str 1
Str 2
...
Val 1
Val 2
...
Features
Label
Feat..
Label
Pr...
?
SQL Transformer
Vector Assembler
Lin Regression

```
bike_sharing01 = bike_sharing.select(
    bike_sharing.season.astype("int"),
    bike_sharing.yr.astype("int"),
    bike_sharing.mnth.astype("int"),
    bike_sharing.holiday.astype("int"),
    bike_sharing.weekday.astype("int"),
    bike_sharing.workingday.astype("int"),
    bike_sharing.weathersit.astype("int"),
    bike_sharing.temp.astype("double"),
    bike_sharing.atemp.astype("double"),
    bike_sharing.hum.astype("double"),
    bike_sharing.windspeed.astype("double"),
    bike_sharing.cnt.astype("int").alias("label")
)
```

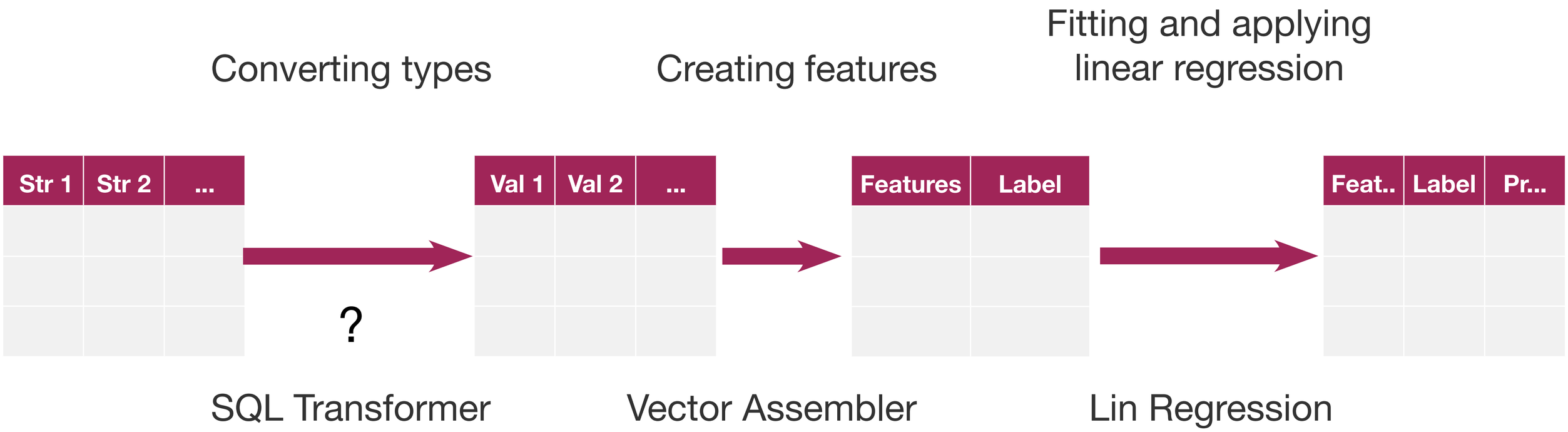
```
from pyspark.ml.feature import SQLTransformer

sql_transformer01 = SQLTransformer(
    statement="""
    SELECT
       cast(season     as int),
       cast(yr         as int),
       cast(mnth       as int),
       cast(holiday    as int),
       cast(weekday    as int),
       cast(workingday as int),
       cast(weathersit as int),
       cast(temp       as double),
       cast(atemp      as double),
       cast(hum        as double),
       cast(windspeed  as double),
       cast(cnt        as int) as label
    FROM __THIS__
    """)
```

```
from pyspark.ml.feature import SQLTransformer

sql_transformer01 = SQLTransformer(
    statement="""
    SELECT
       cast(season     as int),
       cast(yr         as int),
       cast(mnth       as int),
       cast(holiday    as int),
       cast(weekday    as int),
       cast(workingday as int),
       cast(weathersit as int),
       cast(temp       as double),
       cast(atemp      as double),
       cast(hum        as double),
       cast(windspeed  as double),
       cast(cnt        as int) as label
    FROM __THIS__
    """)
```

Pipeline


Converting types
Creating features
Fitting and applying linear regression
Str 1
Str 2
...
?
Val 1
Val 2
...
Features
Label
Feat..
Label
Pr...
SQL Transformer
Vector Assembler
Lin Regression

Pipeline

Converting types

Creating features

Fitting and applying linear regression

Str 1	Str 2	...

?

Val 1	Val 2	...

Features	Label

Feat..	Label	Pr...

SQL Transformer

Vector Assembler

Lin Regression

SQL Transformer

```
sql_transformer02 = SQLTransformer(
    statement="""
    SELECT
       features,
       label
    FROM __THIS__
    """)
```

```
from pyspark.ml import Pipeline

pipeline = Pipeline(stages=[
                    sql_transformer01,
                    assembler,
                    sql_transformer02,
                    lr
                ])
```

```
pipeline_model = pipeline.fit(train)
```

```
train01 = pipeline_model.transform(train)

test01 = pipeline_model.transform(test)

test01.limit(3).toPandas()
```

	features	label	prediction
0	[2.0, 0.0, 4.0, 0.0, 2.0, 2.0, 0.5025, 0.49305...	2034	2656.789689
1	[2.0, 0.0, 4.0, 0.0, 1.0, 1.0, 0.5125, 0.50314...	3429	3847.812611
2	[2.0, 0.0, 4.0, 0.0, 6.0, 2.0, 0.46, 0.450121,...	4036	2608.635643

```
print "r2   =", evaluator.evaluate(test01,
                    {evaluator.metricName: "r2"})

print "rmse =", evaluator.evaluate(test01,
                     {evaluator.metricName: "rmse"})
```

```
r2   = 0.784378126749
rmse = 912.341004982
```

```
pipeline_model.save("pipeline_model")
```

```
from pyspark.ml import PipelineModel

new_pipeline_model = PipelineModel.load("pipeline_model")
```

```
test02 = new_pipeline_model.transform(test)


print "r2   =", evaluator.evaluate(test02,
                    {evaluator.metricName: "r2"})

print "rmse =", evaluator.evaluate(test02,
                     {evaluator.metricName: "rmse"})
```

```
r2   = 0.784378126749
rmse = 912.341004982
```

What you will learn ?

- how Spark MLlib works
- what transformers are
- what estimators are
- how to use pipelines