Feature Engineering with SKLearn-Pandas

By Ramesh Sampath (@sampathweb)

Slides: bit.ly/sk-pandas

Ramesh Sampath

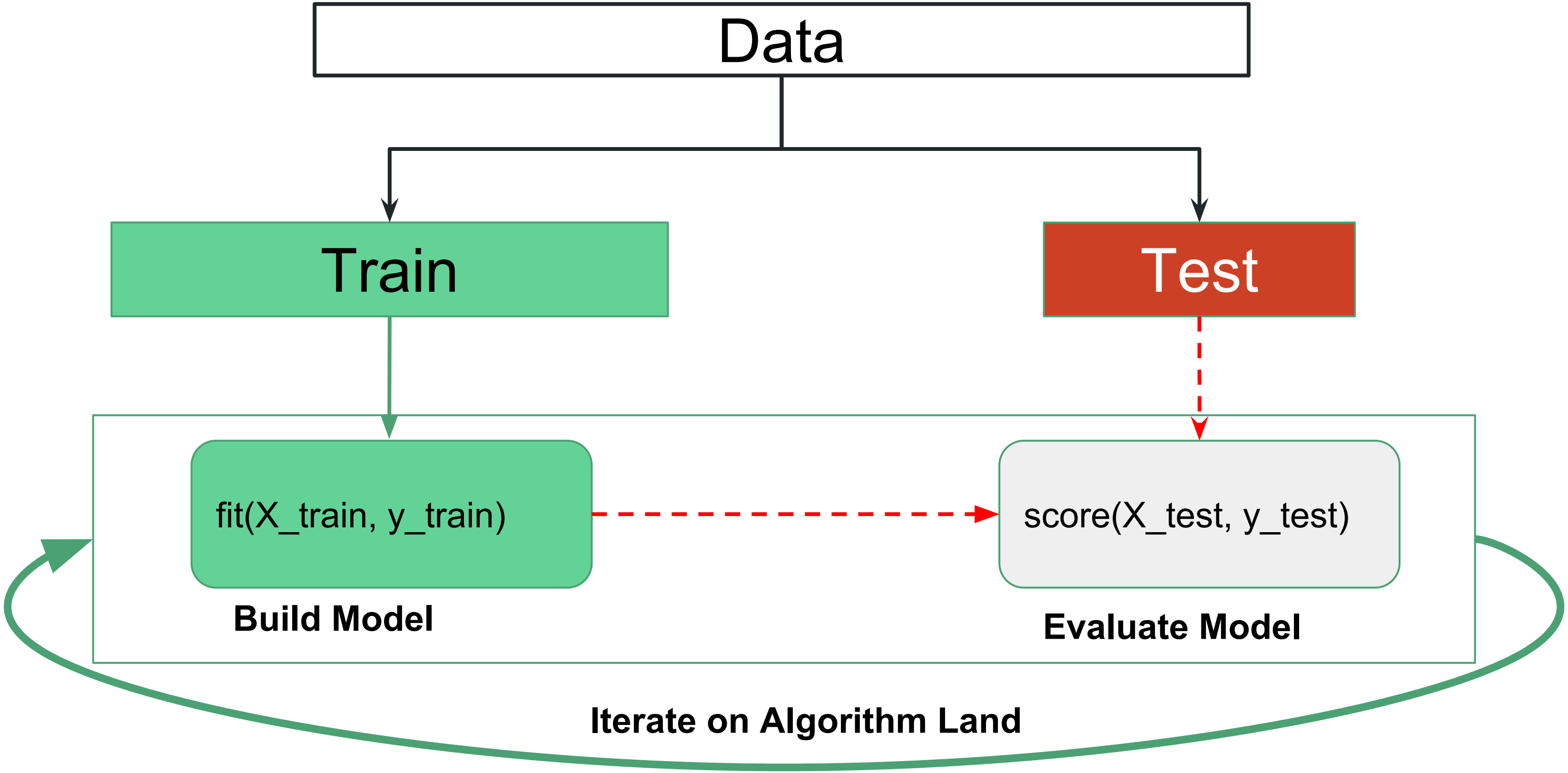
- Data Science Engineer
    - Some Machine Learning Models
    - A lot of Pre-Processing
    - Deploy it as API Services

@sampathweb (github / twitter / linkedin)

What’s the Problem

- Data Scientists Want to -
    - Build Models
    - Tune Models
    - Spend time in Algorithm Land

Ideal World



Data

Train

Test

fit(X_train, y_train)

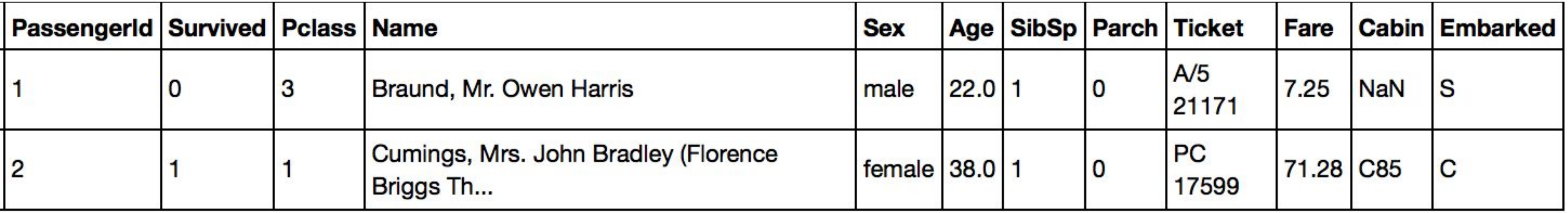
score(X_test, y_test)

Build Model

Evaluate Model

Iterate on Algorithm Land

Data is Messy

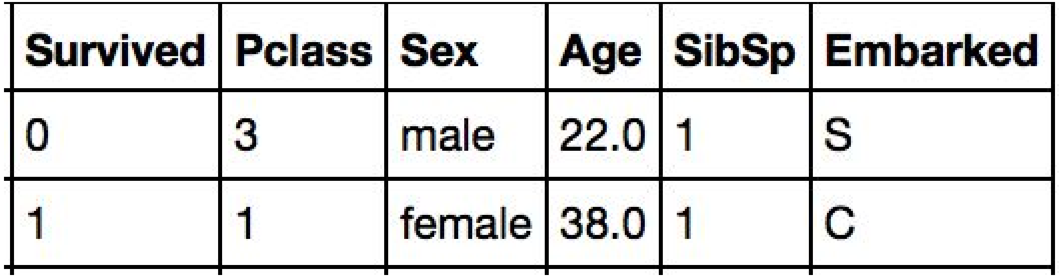


PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.25	NaN	S
2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.28	C85	C

Data needs to be Numerical Vector for Machine Learning.

$$\boxed{\frac{1}{1+e^{-(\beta_0+\beta_1 x)}}}$$

Vectorizing



Survived	Pclass	Sex	Age	SibSp	Embarked
0	3	male	22.0	1	S
1	1	female	38.0	1	C

Target - Classification

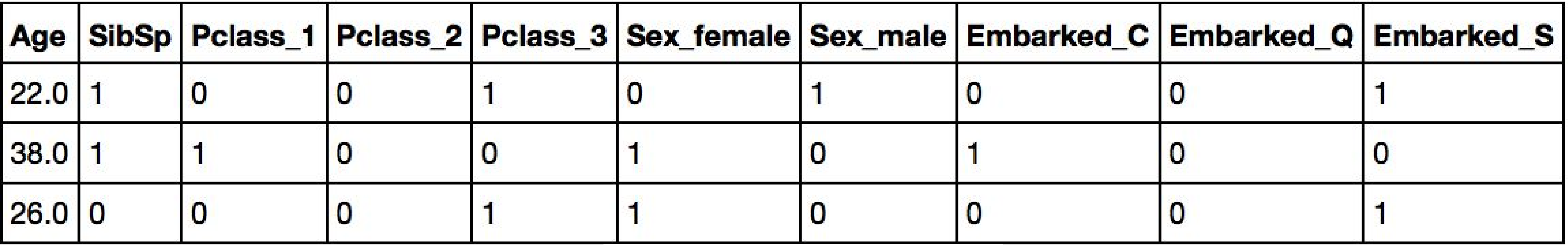
Class - Categorical

Gender - Categorical

Age - Continuous, N/A

Sibling - Count

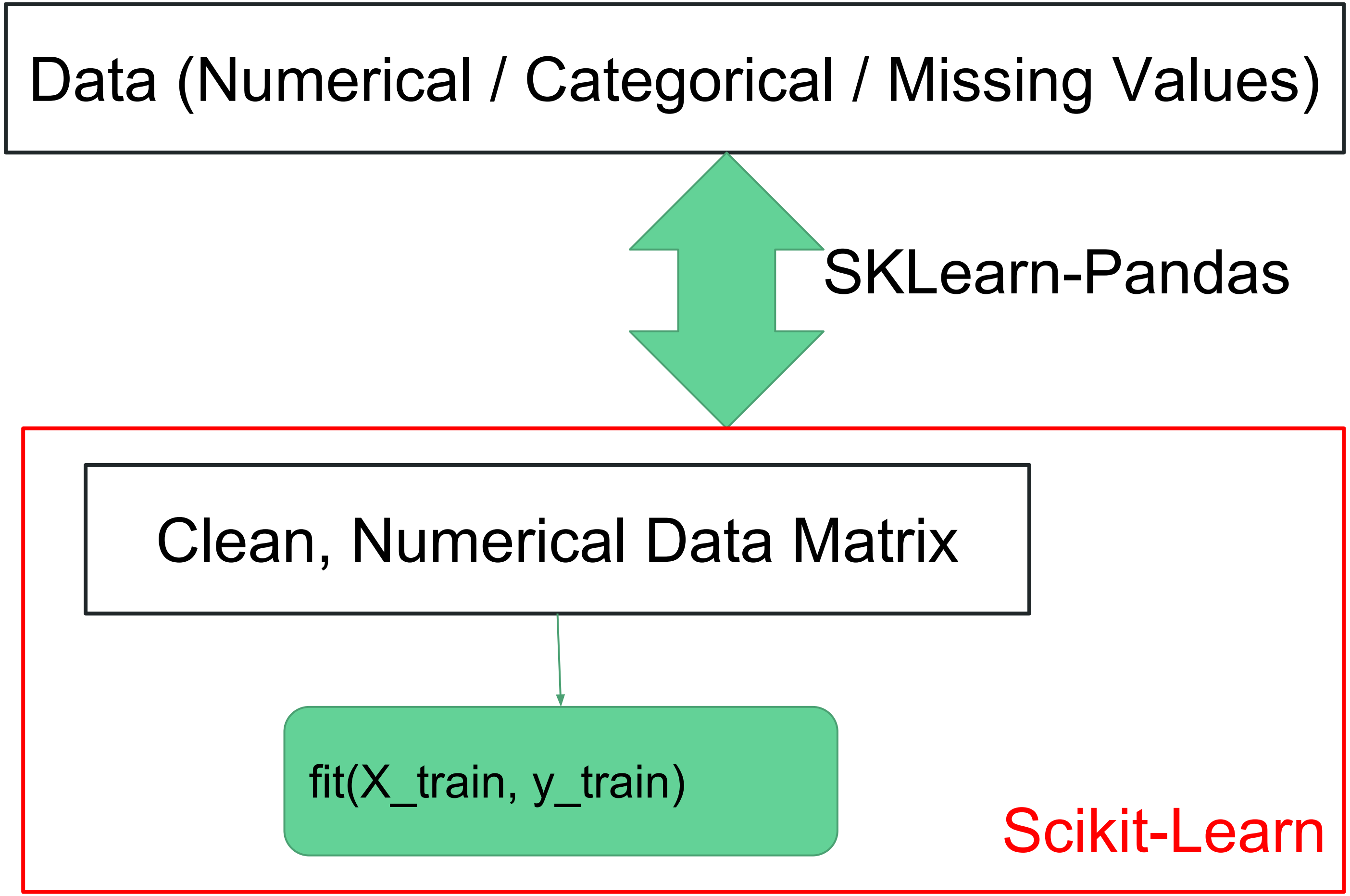
Embarked - Categorical, N/A



Age	SibSp	Pclass_1	Pclass_2	Pclass_3	Sex_female	Sex_male	Embarked_C	Embarked_Q	Embarked_S
22.0	1	0	0	1	0	1	0	0	1
38.0	1	1	0	0	1	0	1	0	0
26.0	0	0	0	1	1	0	0	0	1

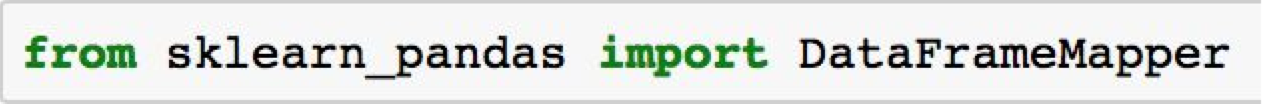
$$\frac{1}{1+e^{-(\beta_0+\beta_1 x)}}$$

Logistic Regression


Data (Numerical / Categorical / Missing Values)
SKLearn-Pandas
Clean, Numerical Data Matrix
fit(X_train, y_train)
Scikit-Learn

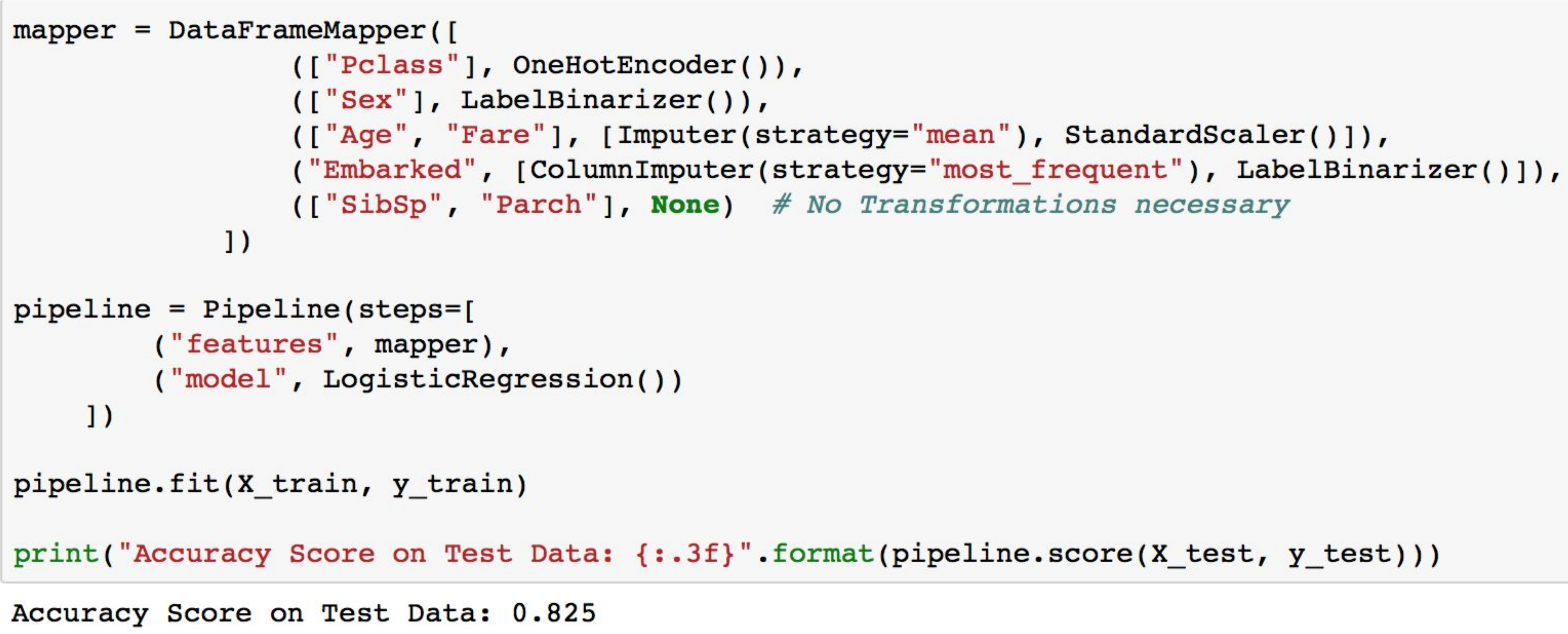
SKLearn-Pandas

- Original code by Ben Hamner (Kaggle CTO) and
- Paul Butler (Google NY) 2013



```
from sklearn_pandas import DataFrameMapper
```

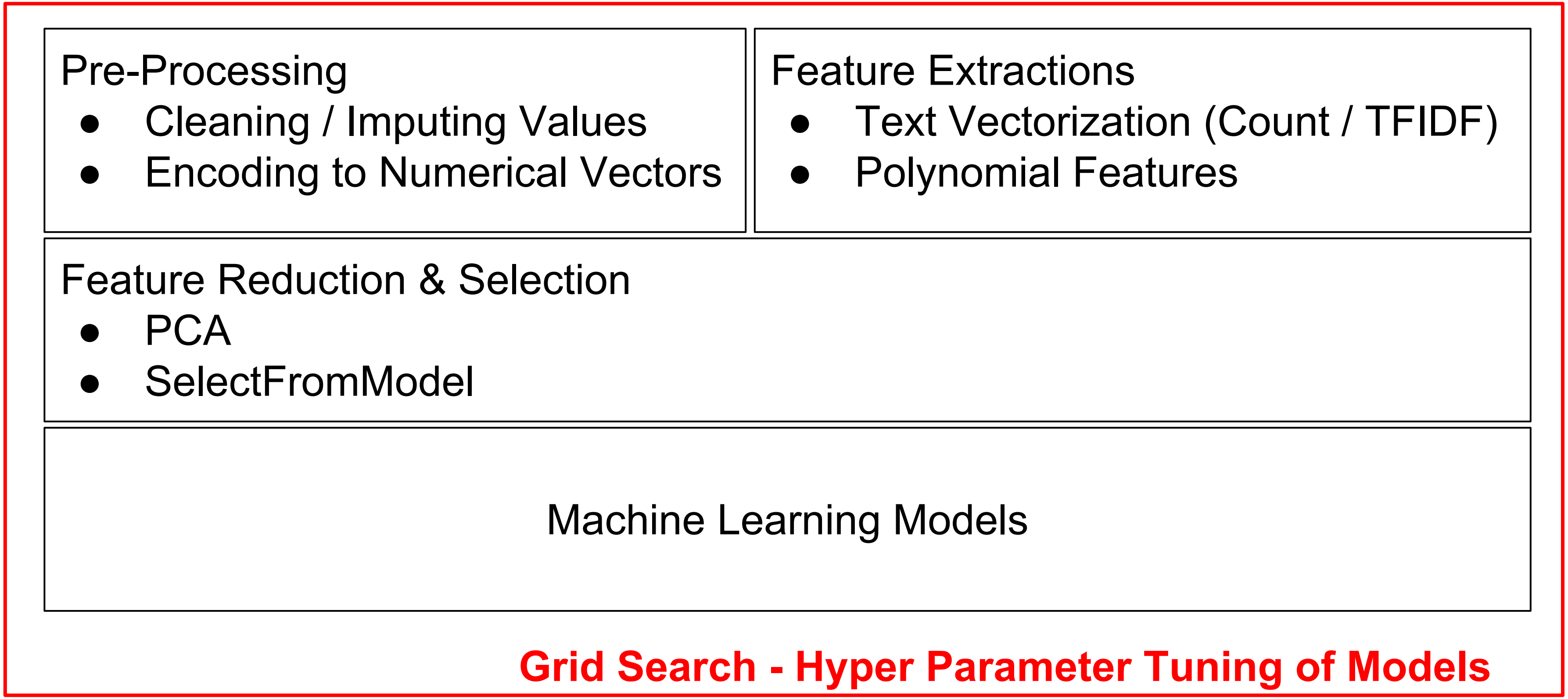
SKLearn-Pandas



```
mapper = DataFrameMapper([
                ([“Pclass”], OneHotEncoder()),
                ([“Sex”], LabelBinarizer()),
                ([“Age”, “Fare”], [Imputer(strategy=“mean”), StandardScaler()]),
                (“Embarked”, [ColumnImputer(strategy=“most_frequent”), LabelBinarizer()]),
                ([“SibSp”, “Parch”], None)  # No Transformations necessary
            ])

pipeline = Pipeline(steps=[
        (“features”, mapper),
        (“model”, LogisticRegression())
    ])

pipeline.fit(X_train, y_train)

print(“Accuracy Score on Test Data: {:.3f}”.format(pipeline.score(X_test, y_test)))
```

```
Accuracy Score on Test Data: 0.825
```

Feature Engineering Pipeline



Pre-Processing

- Cleaning / Imputing Values
- Encoding to Numerical Vectors

Feature Extractions

- Text Vectorization (Count / TFIDF)
- Polynomial Features

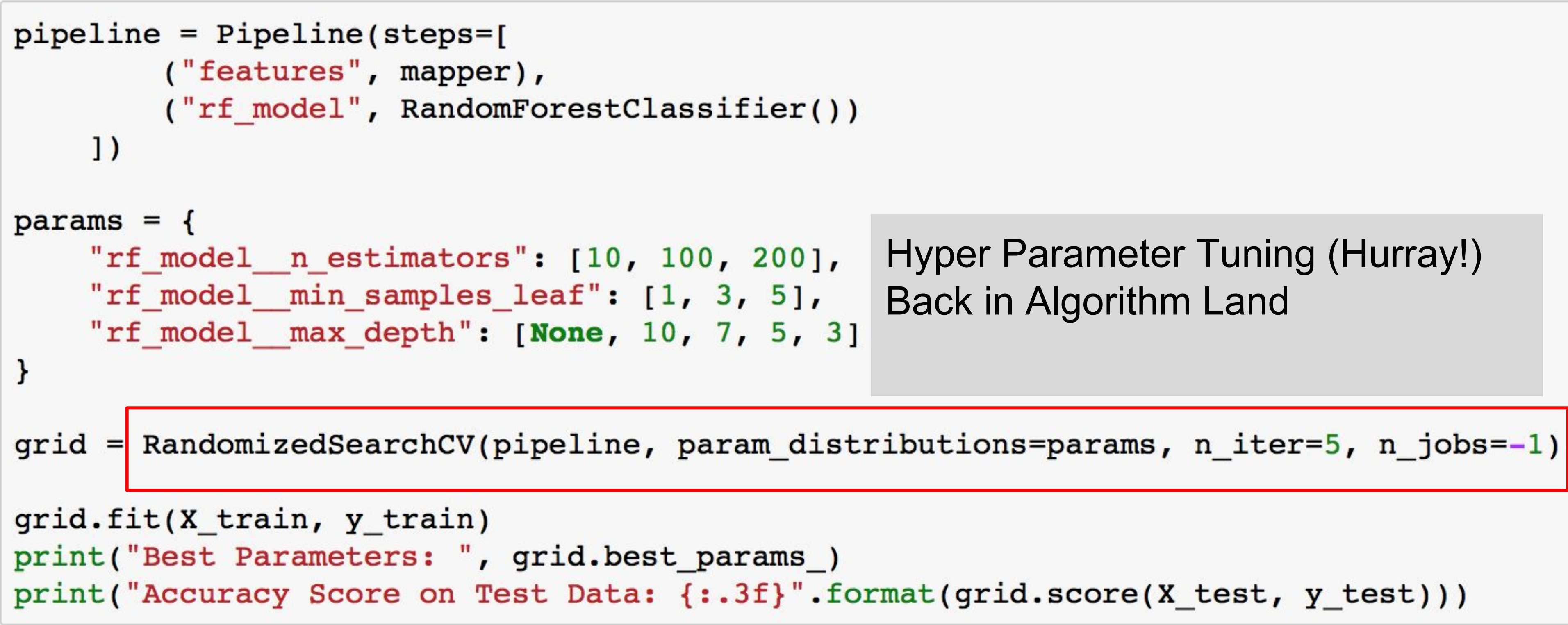
Feature Reduction & Selection

- PCA
- SelectFromModel

Machine Learning Models

Grid Search - Hyper Parameter Tuning of Models

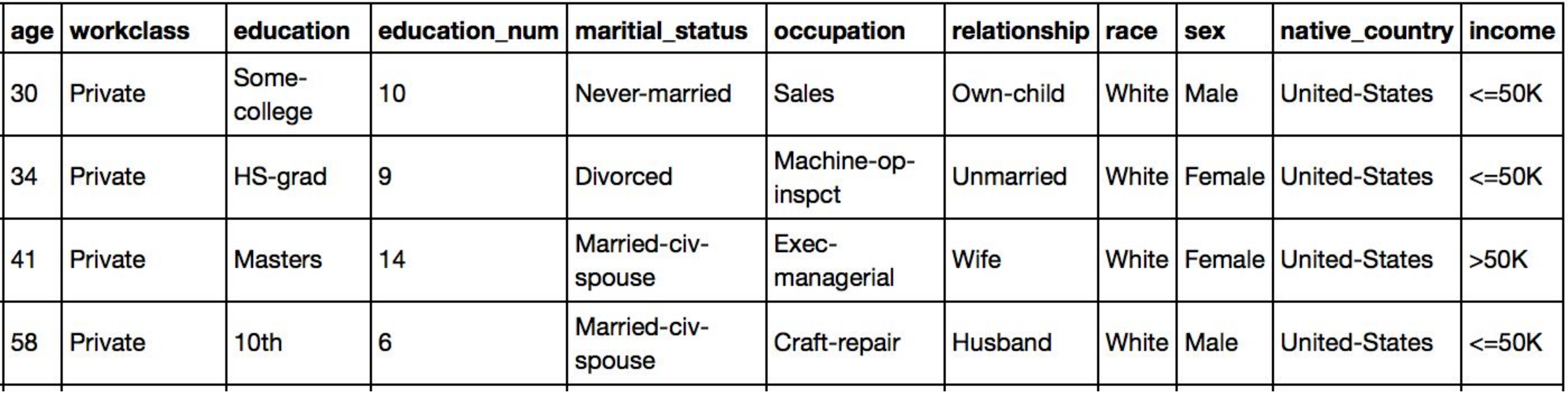
Grid Search



```
pipeline = Pipeline(steps=[
        ("features", mapper),
        ("rf_model", RandomForestClassifier())
    ])

params = {
    "rf_model__n_estimators": [10, 100, 200],
    "rf_model__min_samples_leaf": [1, 3, 5],
    "rf_model__max_depth": [None, 10, 7, 5, 3]
}

grid = RandomizedSearchCV(pipeline, param_distributions=params, n_iter=5, n_jobs=-1)

grid.fit(X_train, y_train)
print("Best Parameters: ", grid.best_params_)
print("Accuracy Score on Test Data: {:.3f}".format(grid.score(X_test, y_test)))
```

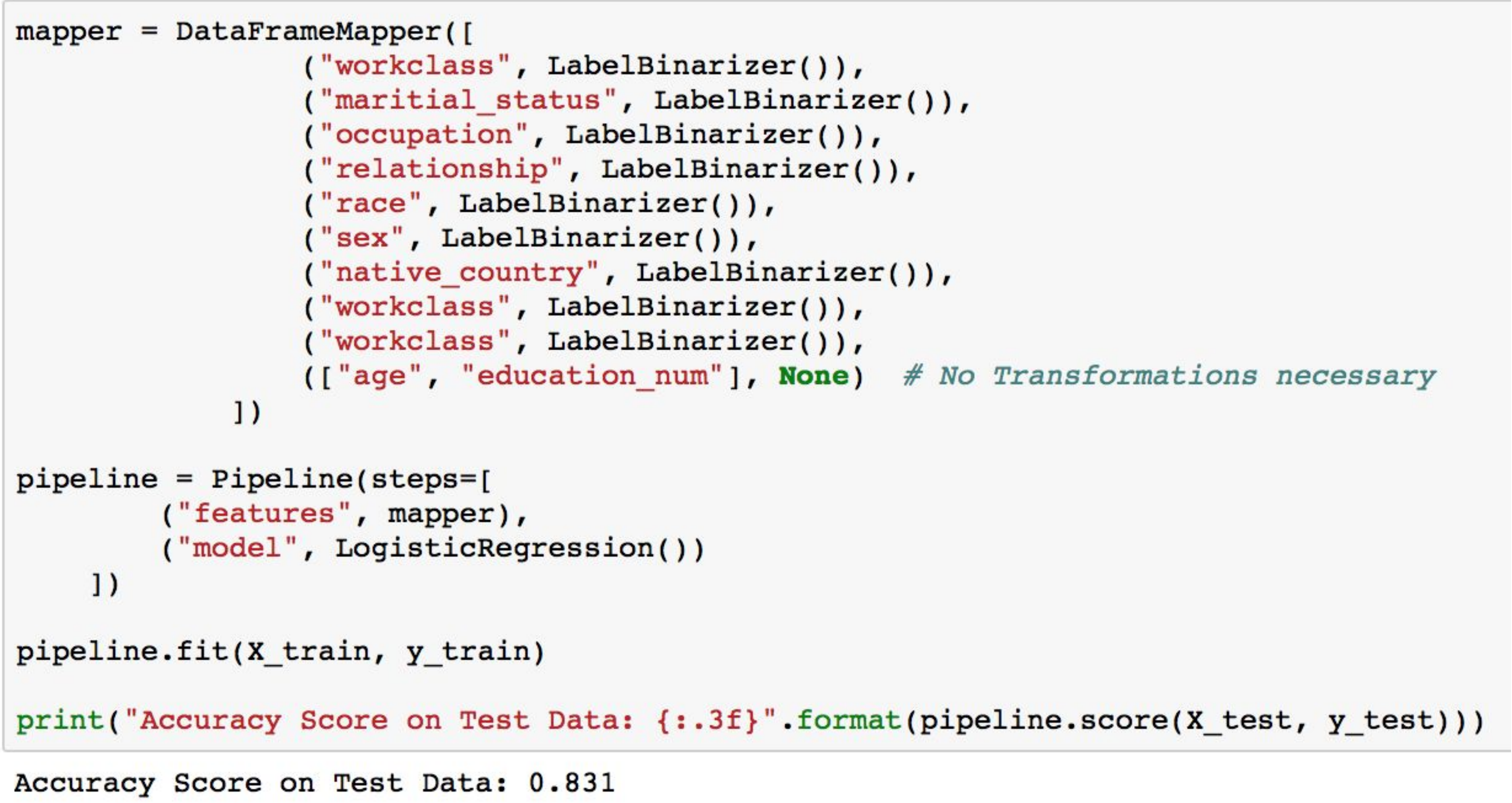
Hyper Parameter Tuning (Hurray!)
Back in Algorithm Land

Data is Messy



age	workclass	education	education_num	maritial_status	occupation	relationship	race	sex	native_country	income
30	Private	Some-college	10	Never-married	Sales	Own-child	White	Male	United-States	<=50K
34	Private	HS-grad	9	Divorced	Machine-op-inspct	Unmarried	White	Female	United-States	<=50K
41	Private	Masters	14	Married-civ-spouse	Exec-managerial	Wife	White	Female	United-States	>50K
58	Private	10th	6	Married-civ-spouse	Craft-repair	Husband	White	Male	United-States	<=50K

Pipeline



```
mapper = DataFrameMapper([
                ("workclass", LabelBinarizer()),
                ("maritial_status", LabelBinarizer()),
                ("occupation", LabelBinarizer()),
                ("relationship", LabelBinarizer()),
                ("race", LabelBinarizer()),
                ("sex", LabelBinarizer()),
                ("native_country", LabelBinarizer()),
                ("workclass", LabelBinarizer()),
                ("workclass", LabelBinarizer()),
                (["age", "education_num"], None)  # No Transformations necessary
            ])

pipeline = Pipeline(steps=[
        ("features", mapper),
        ("model", LogisticRegression())
    ])

pipeline.fit(X_train, y_train)

print("Accuracy Score on Test Data: {:.3f}".format(pipeline.score(X_test, y_test)))
```

```
Accuracy Score on Test Data: 0.831
```

Jupyter Notebook

https://github.com/sampathweb/odsc-feature-engineering-talk

Credits

- Scikit-Learn (https://github.com/scikit-learn/scikit-learn)
- Sklearn-Pandas (https://github.com/paulgb/sklearn-pandas)

StackOverflow Posts:

- http://stackoverflow.com/questions/24458645/label-encoding-across-multiple-columns-in-scikit-learn
- http://stackoverflow.com/questions/34710281/use-featureunion-in-scikit-learn-to-combine-two-pandas-columns-for-tfidf

Thank You!

Slides: bit.ly/sk-pandas

@sampathweb (Github / Twitter / Linkedin)