COLUMN TRANSFORMER

THOMAS J FAN
@THOMASJPFAN

SCIKIT-LEARN CORE DEVELOPER

TITANTIC DATASET

```
data = pd.read_csv(titanic_url)
data.sample(5)
```

	pclass	survived	name	sex	age	sibsp	parch	ticket	fare	cabin	embarked	boat	body	home.dest
1242	3	0	Thomas, Mr. John	male	NaN	0	0	2681	6.4375	NaN	C	NaN	NaN	NaN
278	1	1	Stahelin-Maeglin, Dr. Max	male	32.0	0	0	13214	30.5000	B50	C	3	NaN	Basel, Switzerland
1164	3	0	Saad, Mr. Amin	male	NaN	0	0	2671	7.2292	NaN	C	NaN	NaN	NaN
684	3	0	Bourke, Mrs. John (Catherine)	female	32.0	1	1	364849	15.5000	NaN	Q	NaN	NaN	Ireland Chicago, IL
686	3	1	Bradley, Miss. Bridget Delia	female	22.0	0	0	334914	7.7250	NaN	Q	13	NaN	Kingwilliamstown, Co Cork, Ireland Glens Falls...

NUMERIC FEATURES - 1

```
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.impute import SimpleImputer

numeric_transformers = Pipeline([
    ('imputer', SimpleImputer(strategy='median')),
    ('scaler', StandardScaler())
])
```

NUMERIC FEATURES - 2

```
from sklearn.compose import ColumnTransformer

preprocessor = ColumnTransformer([
    ('numeric', numeric_transformers, ['age', 'fare']),
    ...
])
```

CATEGORICAL FEATURES - 1

```
from sklearn.preprocessing import OneHotEncoder

categorical_transformer = Pipeline([
    ('imputer', SimpleImputer(strategy='constant',
                              fill_value='missing')),
    ('onehot', OneHotEncoder(handle_unknown='ignore'))
])
```

CATEGORICAL FEATURES - 2

```
preprocessor = ColumnTransformer([
    ('numeric', numeric_transformers, ['age', 'fare']),
    ('categorical', categorical_transformer,
        ['embarked', 'sex', 'pclass', 'parch', 'sibsp'])
])
```

CLASSIFIER

```
from sklearn.ensemble import VotingClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.neural_network import MLPClassifier

classifier = VotingClassifier([
    ('lr', LogisticRegression(solver='lbfgs', max_iter=100)),
    ('gb', GradientBoostingClassifier()),
    ('mlp', MLPClassifier(max_iter=1000))
], voting='soft')
```

PIPELINE

```
preprocessor = ColumnTransformer([
    ('numeric', numeric_transformers, ['age', 'fare']),
    ('categorical', categorical_transformer,
        ['embarked', 'sex', 'pclass', 'parch', 'sibsp'])
])

pipe = Pipeline([
    ('preprocessor', preprocessor),
    ('classifier', classifier)
])
```

TRAIN/TEST SET

```
from sklearn.model_selection import train_test_split
X = data.drop('survived', axis=1)
y = data['survived']

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.25)

pipe.score(X_test, y_test)
# 0.811
```

THANK YOU!