**Pipelines**

**Step1: Import**

*>> from sklearn.pipeline import Pipeline, make\_pipeline*

**Step2: Prepare Columns**

*>> from sklearn.compose import ColumnTransformer*

*>> from sklearn.impute import SimpleImputer*

*>> trf1 = ColumnTransformer([*

*("Impute Age", SimpleImputer(),[2]), # Default SimpleImputer Strategyu is mean, [2] is index of columns, index is used as the result is numpy array so it would be easy to handle*

*("impute\_embarked", SimpleImputer(strategy="most\_frequent"),[6])*

*],remainder='passthrough')*

*>> #One Hot Encoding*

*trf2 = ColumnTransformer([*

*('OHE\_SEX\_Embarked', OneHotEncoder(sparse=False, handle\_unknown='ignore'),[1,6])*

*],remainder='passthrough')*

*>> trf3 = ColumnTransformer([*

*('scale',MinMaxScaler(),slice(0,10)) #0-10 columns*

*])*

*>> trf4 = SelectKBest(score\_func=chi2, k=8) #select 8 best*

*>> trf5= DecisionTreeClassifier()*

**Step 3: Make Pipe**

*>> pipe = Pipeline([*

*('trf1',trf1),*

*('trf2',trf2),*

*('trf3',trf3),*

*('trf4',trf4),*

*('trf5',trf5)*

*])*

**Step4: Visualize The Pipe**

*>>#Display Pipeline*

*from sklearn import set\_config*

*set\_config(display='diagram')*

**Step5: Fit The Pipe**

*>>pipe.fit(X\_train,y\_train)*

**Step6: Explore Pipeline**

*>> pipe.classes\_*

*>> pipe.named\_steps*

*>> pipe.named\_steps["trf1"]*

*>> pipe.named\_steps["trf1"].transformers\_*

*>> pipe.named\_steps["trf1"].transformers\_[0][1].statistics\_*

**Cross Validation Using Pipeline**

*# using Cross\_val\_score*

*>> from sklearn.model\_selection import cross\_val\_score*

*>> cross\_val\_score(pipe, X\_train, y\_train, cv=5, scoring="accuracy").mean()*

**Grid Search Using Pipeline**

*>> params ={*

*'trf5\_\_max\_depth':[1,2,3,4,5,10,None]*

*}*

*>> from sklearn.model\_selection import GridSearchCV*

*>> grid = GridSearchCV(pipe, params, cv=5, scoring="accuracy")*

*>> grid.fit(X\_train,y\_train)*

*>> grid.best\_params\_*

{'trf5\_\_max\_depth': 5}

**Export Pipeline**

*>> import pickle as pkl*

*>> pkl.dump(pipe, open("pipe.pkl", "wb"))*

**Import Pipeline**

*>> pipe2 = pkl.load(open("pipe.pkl","rb"))*

*pipe2*

**Predictions**

*>>test\_input2=np.array([2,"female",32.0,0,0,15.2,"S"]) {shape=(7,)*

*>>test\_input2= test\_input2.reshape(1,7) {Shape=(1,7)}*

*>>test\_input2=np.array([2,"female",32.0,0,0,15.2,"S"], dtype=object).reshape(1,7)*

*>> pipe.predict(test\_input2)*

**Scalars**

**Step1: Import**

>>from sklearn.preprocessing import MinMaxScaler,StandardScalar

**Step2: Define**

>> scalar= MinMaxScaler()

**Step3: Fit and Transform**

>> df2["AGE"] = scalar.fit\_transform(df2[["AGE"]])

**Label Encoder**

**Step1: Import**

>> from sklearn.preprocessing import LabelEncoder

**Step2: Define**

>> le = LabelEncoder()

**Step3: Fit**

>> le.fit(df[“Sex”])

**Step4: Transform**

>> le.transform(test[“Sex”]) #male,female

[2,1]

**Step5: Inverse Transform**

>> list(le.inverse\_transform([2,1]))

[male,female]

**One Hot Encoder**

**Step1: Import**

>> from sklearn.preprocessing import OneHotEncoder

**Step2: Define**

>> ohe = OneHotEncoder()

**Step3: Fit**

>> ohe.fit(df[“Sex”])

[1,0] or [0,1]

**Step4: Transform**

>> ohe.transform(test[“Sex”])