Lisbon School of Economics & Management
Universidade de Lisboa

Carlos J. Costa

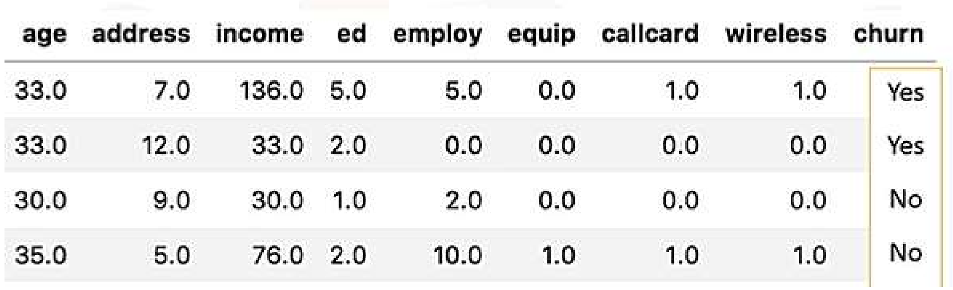
CLASSIFICATION

Classification

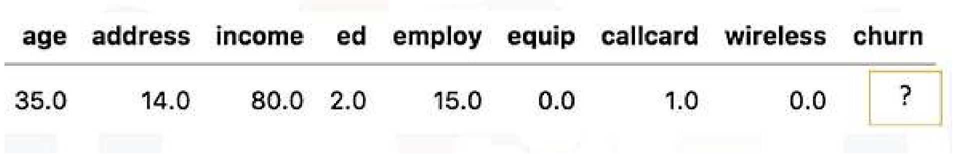
- Supervised learning approach
- Categorizing some unknown items into discrete set of categories or “classes”
- The target attribute is a categorical variable
- To solve a classification problem
  - identify the target or class, which is the variable to predict.
  - the target balancing is mandatory
  - choose the best training strategy to train classification models.

Classification

- Churn (not churn rate) depends from several characteristics of the client, product and communication.



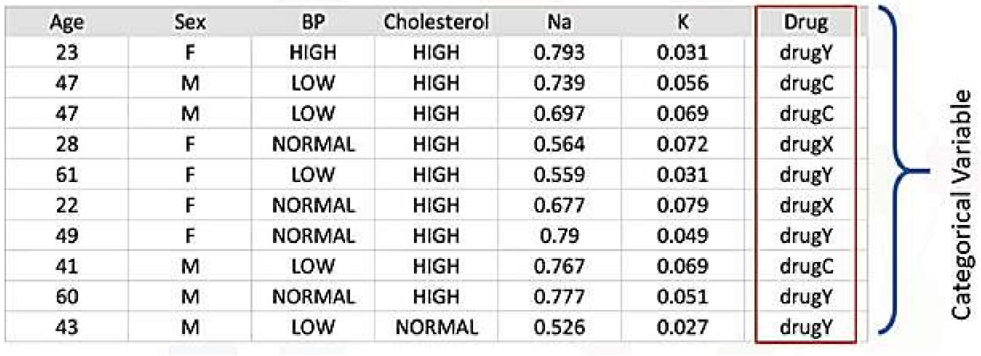
age	address	income	ed	employ	equip	callcard	wireless	churn
33.0	7.0	136.0	5.0	5.0	0.0	1.0	1.0	Yes
33.0	12.0	33.0	2.0	0.0	0.0	0.0	0.0	Yes
30.0	9.0	30.0	1.0	2.0	0.0	0.0	0.0	No
35.0	5.0	76.0	2.0	10.0	1.0	1.0	1.0	No



age	address	income	ed	employ	equip	callcard	wireless	churn
35.0	14.0	80.0	2.0	15.0	0.0	1.0	0.0	?

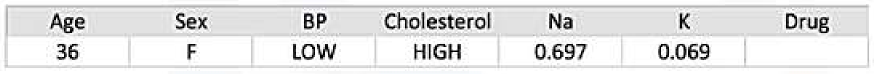
Classification

- What is the best drug according to specific characteristics of the patient



Age	Sex	BP	Cholesterol	Na	K	Drug
23	F	HIGH	HIGH	0.793	0.031	drugY
47	M	LOW	HIGH	0.739	0.056	drugC
47	M	LOW	HIGH	0.697	0.069	drugC
28	F	NORMAL	HIGH	0.564	0.072	drugX
61	F	LOW	HIGH	0.559	0.031	drugY
22	F	NORMAL	HIGH	0.677	0.079	drugX
49	F	NORMAL	HIGH	0.79	0.049	drugY
41	M	LOW	HIGH	0.767	0.069	drugC
60	M	NORMAL	HIGH	0.777	0.051	drugY
43	M	LOW	NORMAL	0.526	0.027	drugY

Categorical Variable



Age	Sex	BP	Cholesterol	Na	K	Drug
36	F	LOW	HIGH	0.697	0.069	

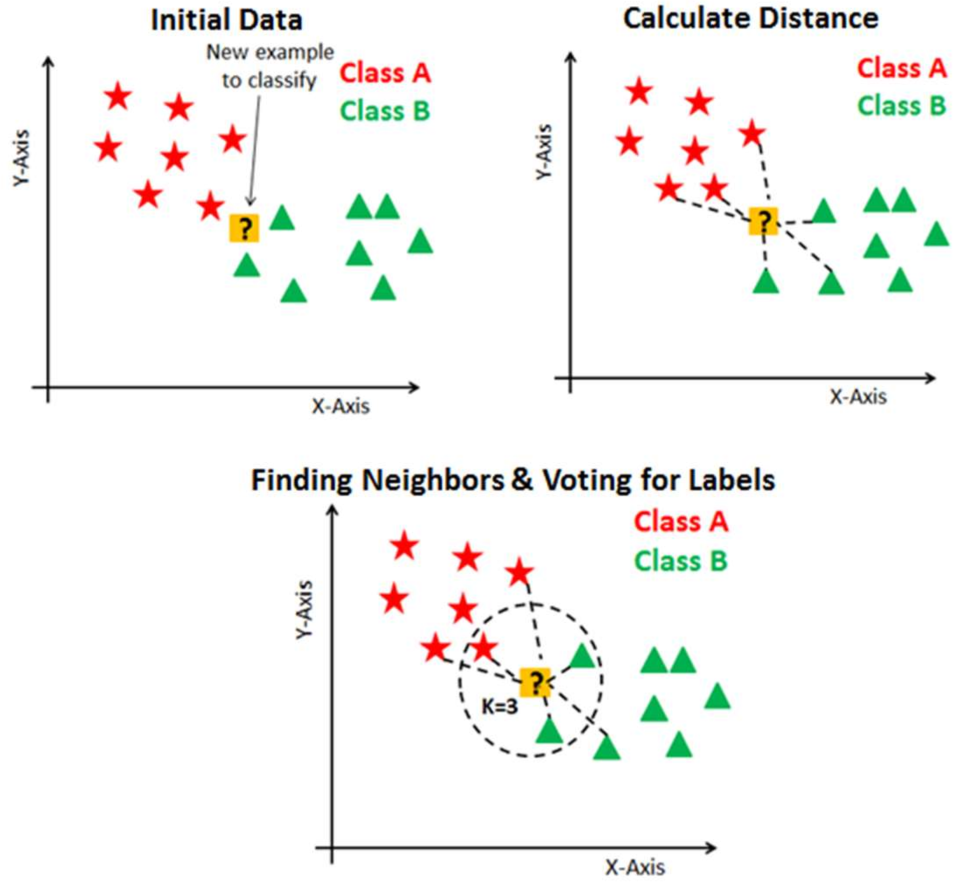
Classification

Classification algorithms in machine learning:

- Decision Trees
- Naive Bayes
- Linear Discriminate Analysis
- K -Near Neighbor (KNN)
- Logistic Regression
- Neural Networks
- Support Vector Machines (SVM)

Classification

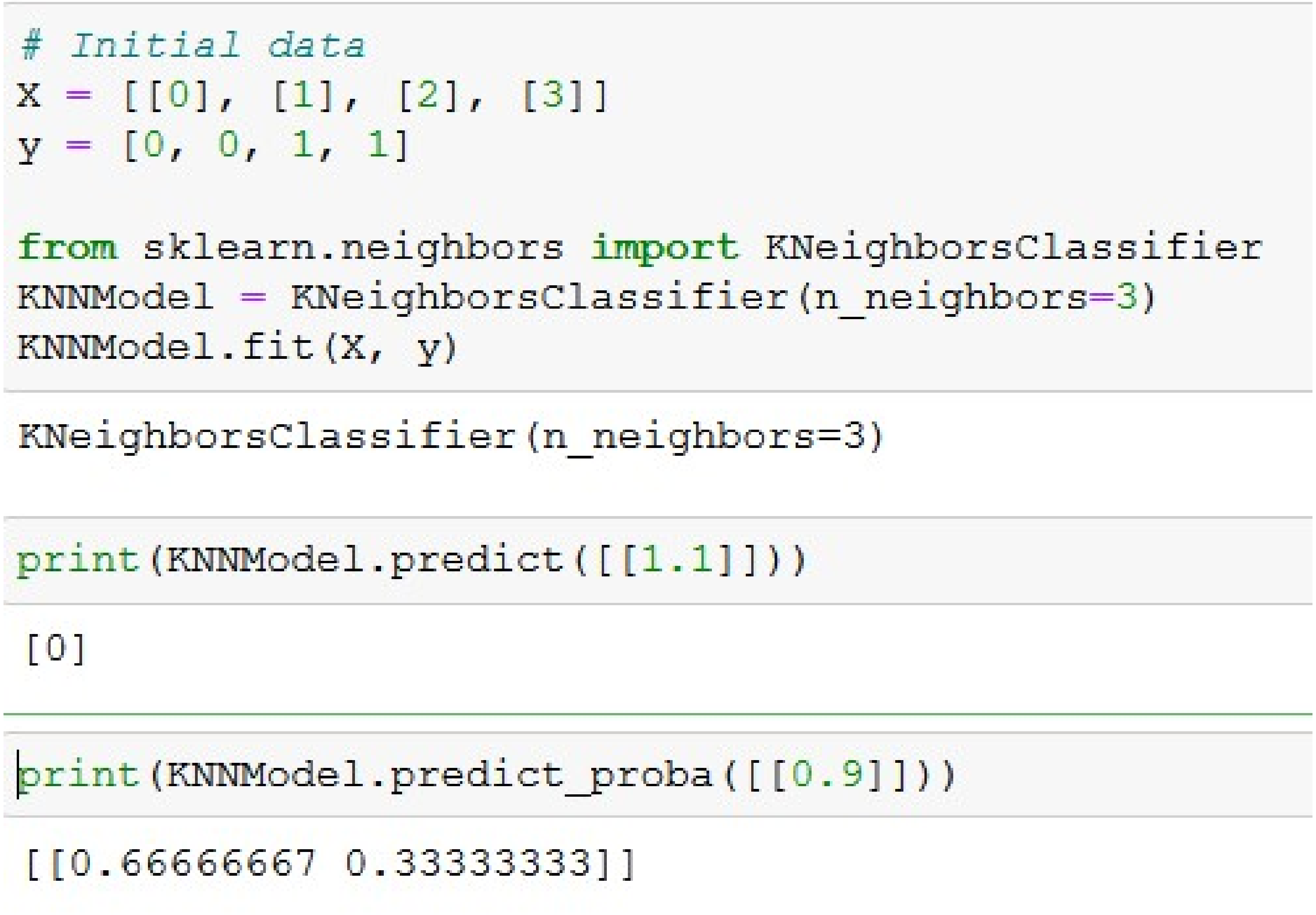
1.KNN (K-Nearest Neighbour):



Initial Data

Calculate Distance

Finding Neighbors & Voting for Labels



```
# Initial data
X = [[0], [1], [2], [3]]
y = [0, 0, 1, 1]

from sklearn.neighbors import KNeighborsClassifier
KNNModel = KNeighborsClassifier(n_neighbors=3)
KNNModel.fit(X, y)
```

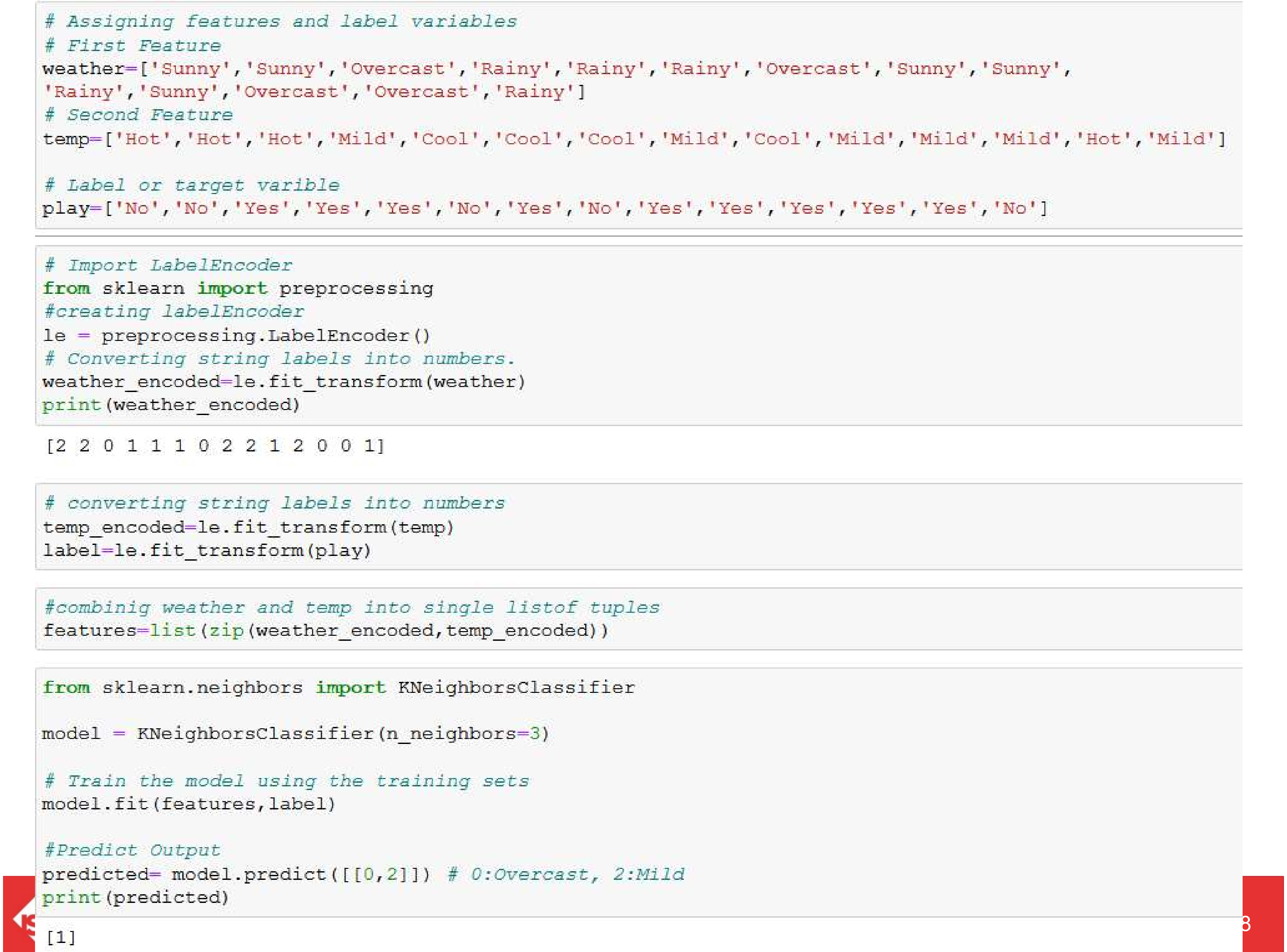
```
KNeighborsClassifier(n_neighbors=3)
```

```
print(KNNModel.predict([[1.1]]))
```

```
[0]
```

```
print(KNNModel.predict_proba([[0.9]]))
```

```
[[0.66666667 0.33333333]]
```



```
# Assigning features and label variables
# First Feature
weather=['Sunny','Sunny','Overcast','Rainy','Rainy','Rainy','Overcast','Sunny','Sunny',
'Rainy','Sunny','Overcast','Overcast','Rainy']
# Second Feature
temp=['Hot','Hot','Hot','Mild','Cool','Cool','Cool','Mild','Cool','Mild','Mild','Mild','Hot','Mild']

# Label or target varible
play=['No','No','Yes','Yes','Yes','No','Yes','No','Yes','Yes','Yes','Yes','Yes','No']
```

```
# Import LabelEncoder
from sklearn import preprocessing
#creating labelEncoder
le = preprocessing.LabelEncoder()
# Converting string labels into numbers.
weather_encoded=le.fit_transform(weather)
print(weather_encoded)
```

```
[2 2 0 1 1 1 0 2 2 1 2 0 0 1]
```

```
# converting string labels into numbers
temp_encoded=le.fit_transform(temp)
label=le.fit_transform(play)
```

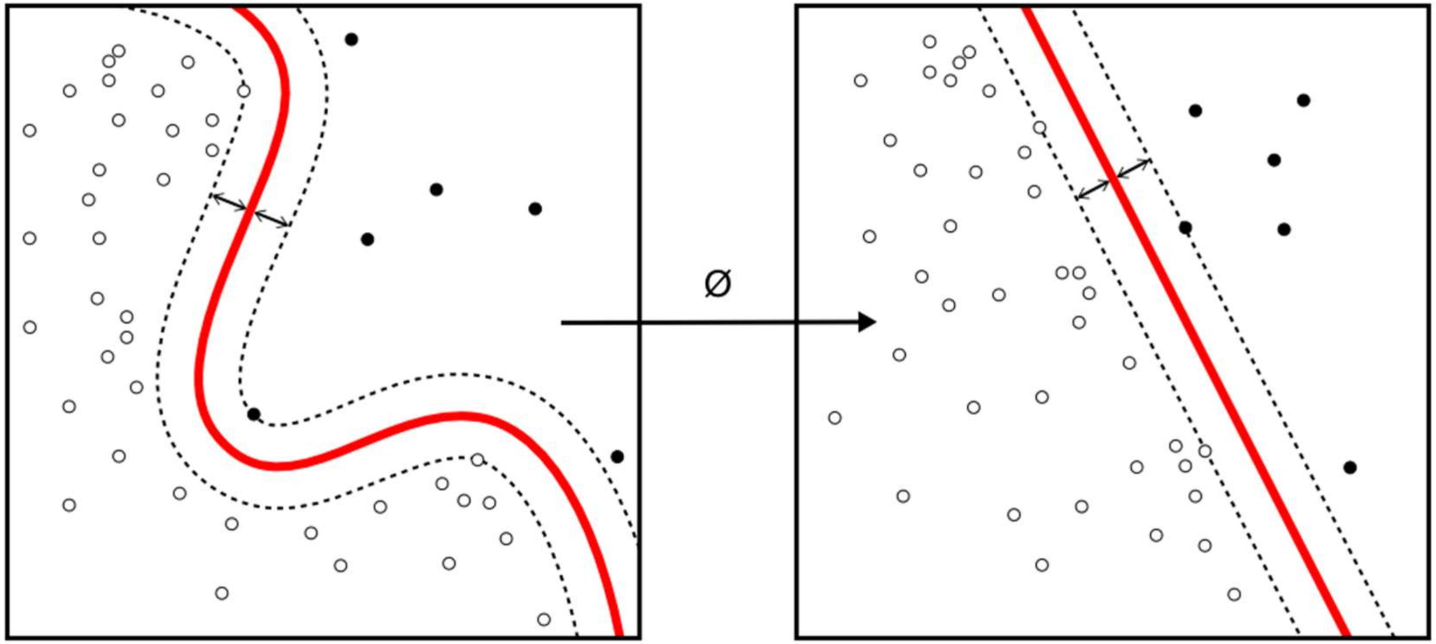
```
#combinig weather and temp into single listof tuples
features=list(zip(weather_encoded,temp_encoded))
```

```
from sklearn.neighbors import KNeighborsClassifier

model = KNeighborsClassifier(n_neighbors=3)

# Train the model using the training sets
model.fit(features,label)

#Predict Output
predicted= model.predict([[0,2]]) # 0:Overcast, 2:Mild
print(predicted)
```

```
[1]
```

Classification

2. SVM (support vector machine)

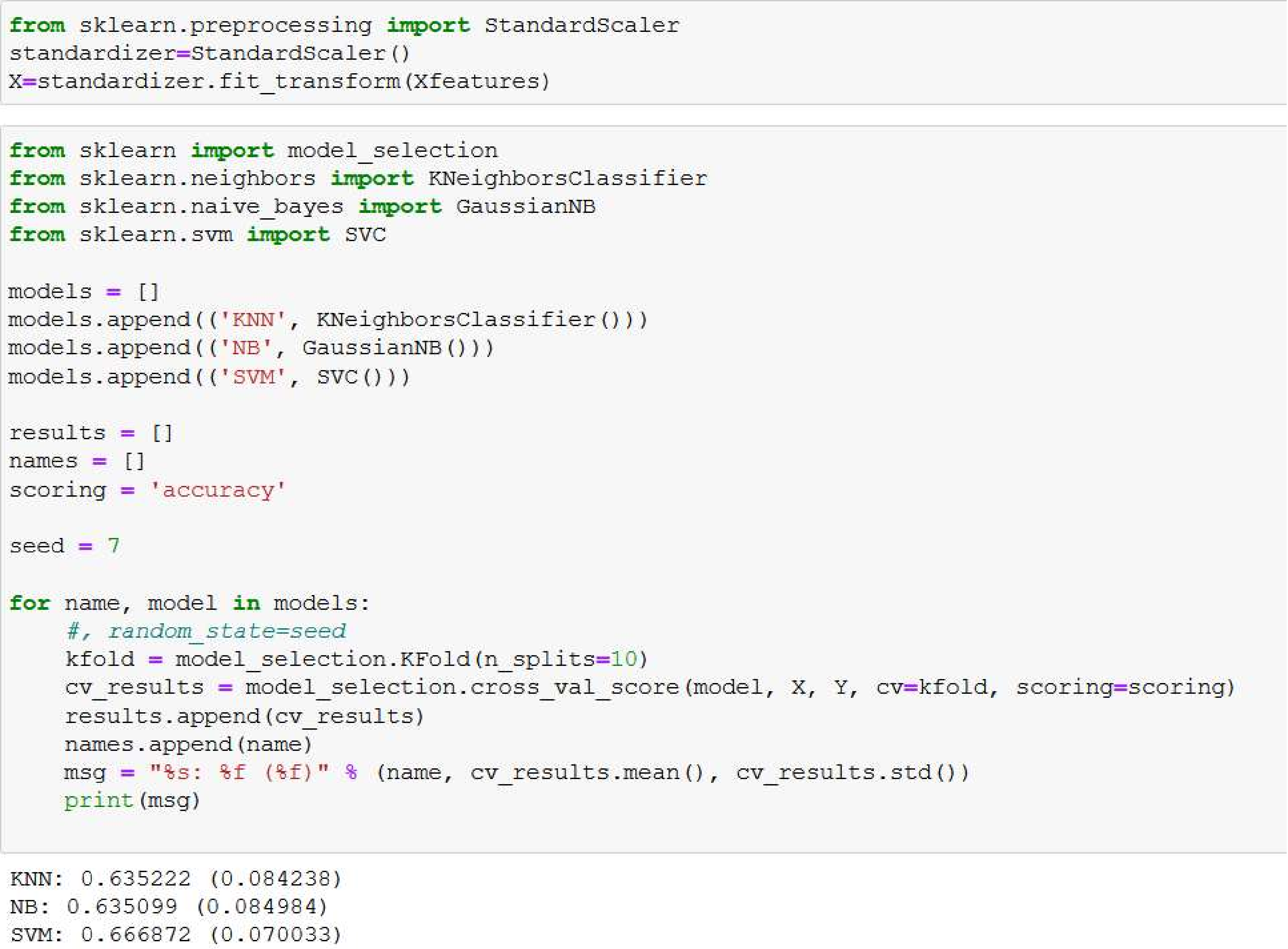


Classification

$$P(y|X) = \frac{P(X|y)P(y)}{P(X)}$$

$$X = (x_1, x_2, x_3, \ldots\ldots, x_n)$$

3. Navie Bayes



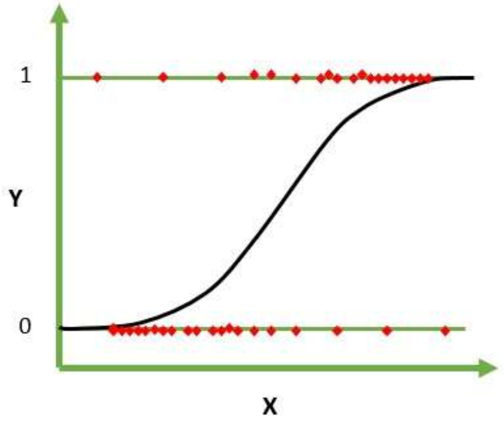
```python
from sklearn.preprocessing import StandardScaler
standardizer=StandardScaler()
X=standardizer.fit_transform(Xfeatures)
```

```python
from sklearn import model_selection
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC

models = []
models.append(('KNN', KNeighborsClassifier()))
models.append(('NB', GaussianNB()))
models.append(('SVM', SVC()))

results = []
names = []
scoring = 'accuracy'

seed = 7

for name, model in models:
    #, random_state=seed
    kfold = model_selection.KFold(n_splits=10)
    cv_results = model_selection.cross_val_score(model, X, Y, cv=kfold, scoring=scoring)
    results.append(cv_results)
    names.append(name)
    msg = "%s: %f (%f)" % (name, cv_results.mean(), cv_results.std())
    print(msg)
```

```
KNN: 0.635222 (0.084238)
NB: 0.635099 (0.084984)
SVM: 0.666872 (0.070033)
```

Logistics Regression

- A regression that having binary dependent variable
- in its basic form, uses a logistic function to model a binary dependent variable



Random Forest

- are an ensemble learning method for classification, regression and other tasks
- operates by constructing a multitude of decision trees at training time
- outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees.

References

- Albon, Ch. (2018) *Machine Learning with Python Cookbook*. O'Reilly
- Domingos, P. (2015) *The Master Algorithm*, Penguin Books
- Hinton, J.; Sejnowski, T.(1999). *Unsupervised Learning: Foundations of Neural Computation*. MIT Press
- Morgan; P. (2019) *Data Science from Scratch with Python*, AI Science
- Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective* (1 edition). Cambridge, MA: The MIT Press.
- Otte, E.; Rousseau, R. (2002). "Social network analysis: a powerful strategy, also for the information sciences". *Journal of Information Science*. 28 (6): 441–453. doi:10.1177/016555150202800601.
- Stuart J. R., Norvig, P. (2010) *Artificial Intelligence: A Modern Approach*, Third Edition, Prentice Hall ISBN 9780136042594.