GENERAL ASSEMBLY

DATA ANALYSIS

Brian Chung

LAST TIME

I. Pandas
- Series
- DataFrames
- A bit into Group-by

II. KNN Intro
- Classification Models
- KNN in Python with sklearn

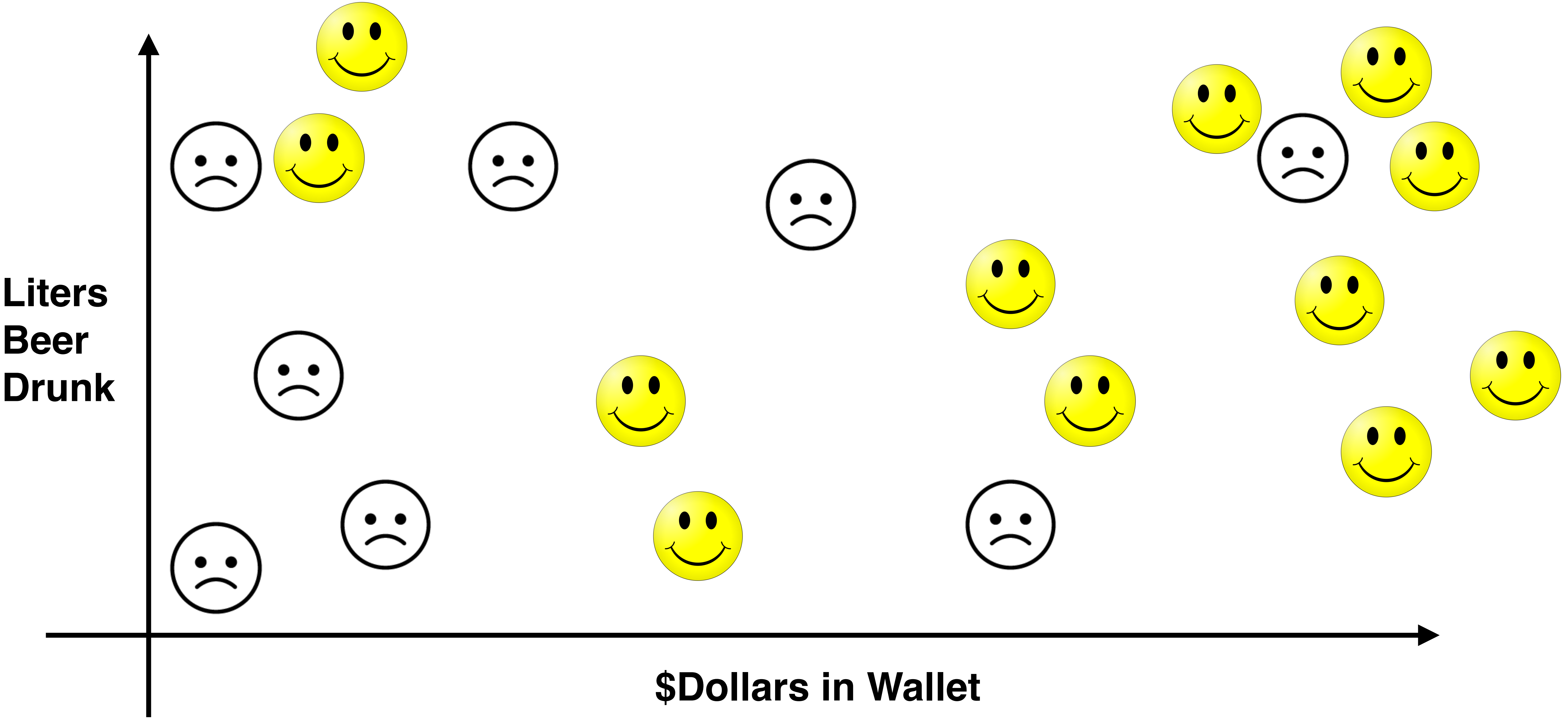
TODAY

I. KNN Review
   - Review
   - Basic Cross Validation

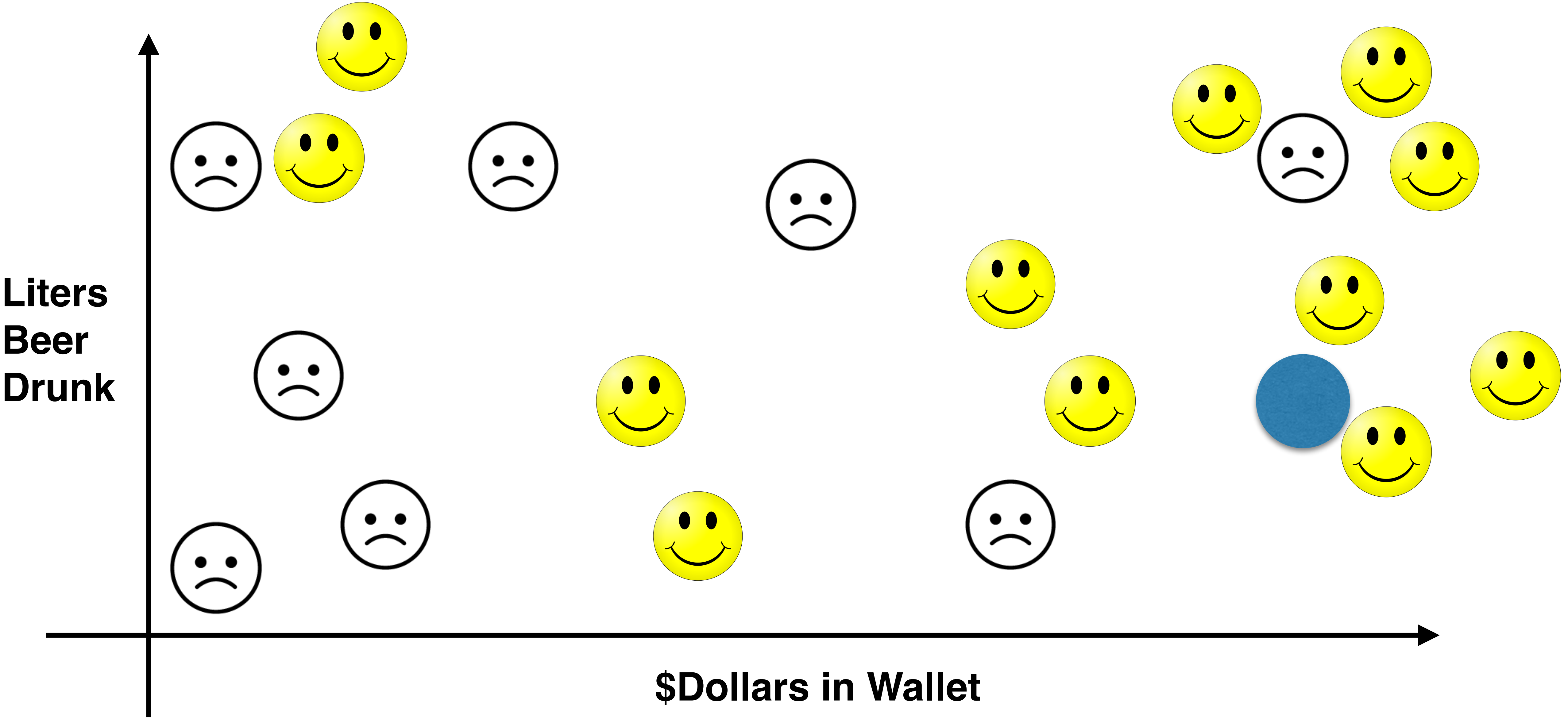
II. Visualization
   - Matplotlib

III. NBA Exercise
   - Entire process!

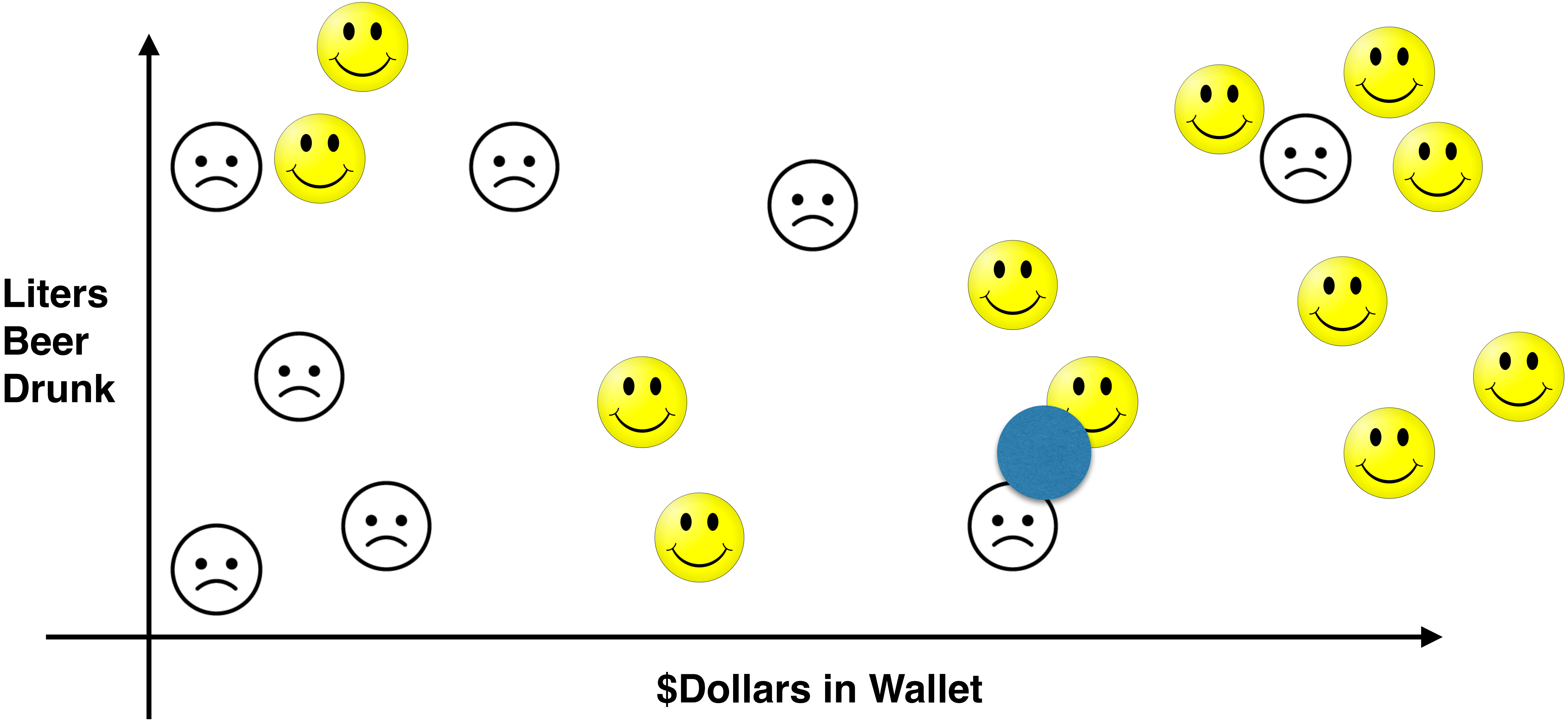
K NEAREST NEIGHBORS - PREDICT HAPPY/SAD



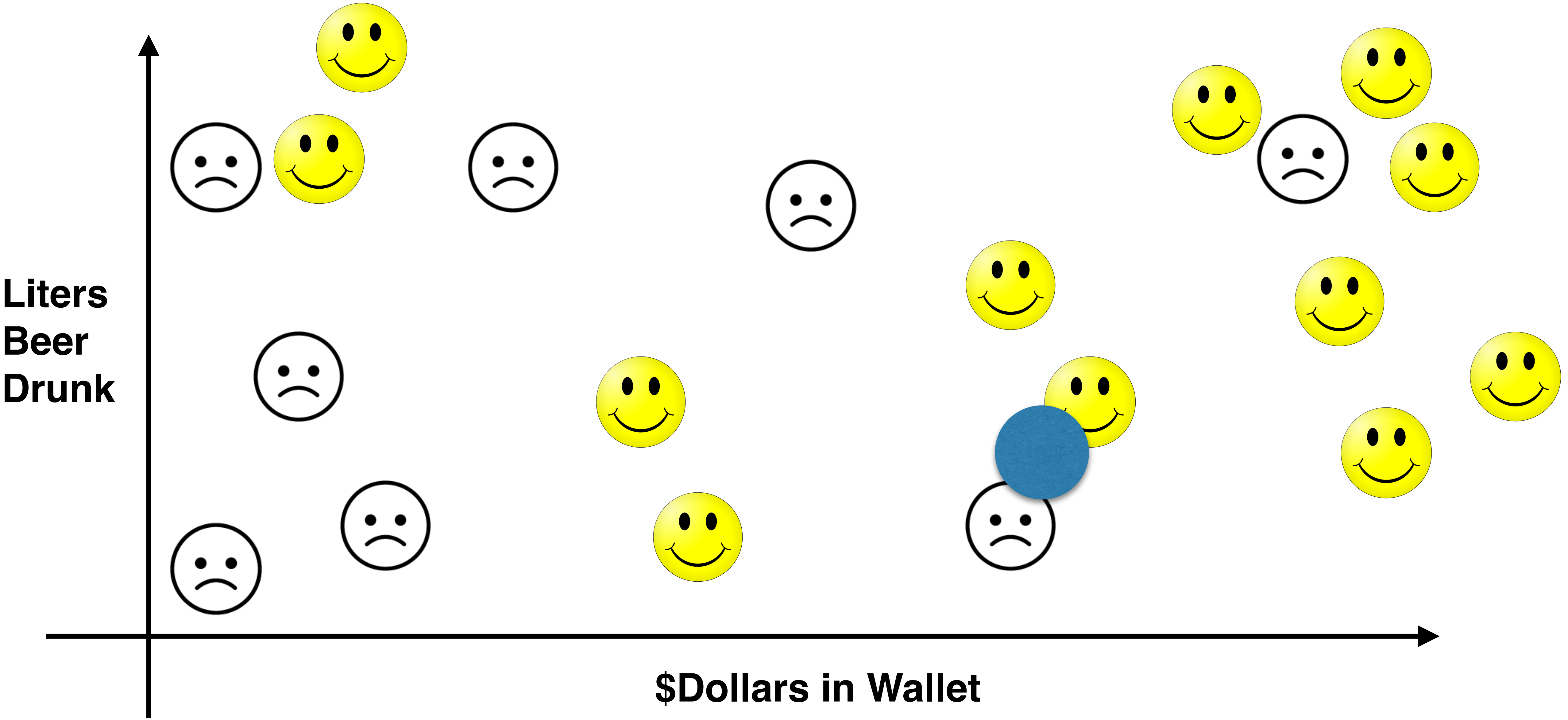
K NEAREST NEIGHBORS - HAPPY OR SAD? K = 2



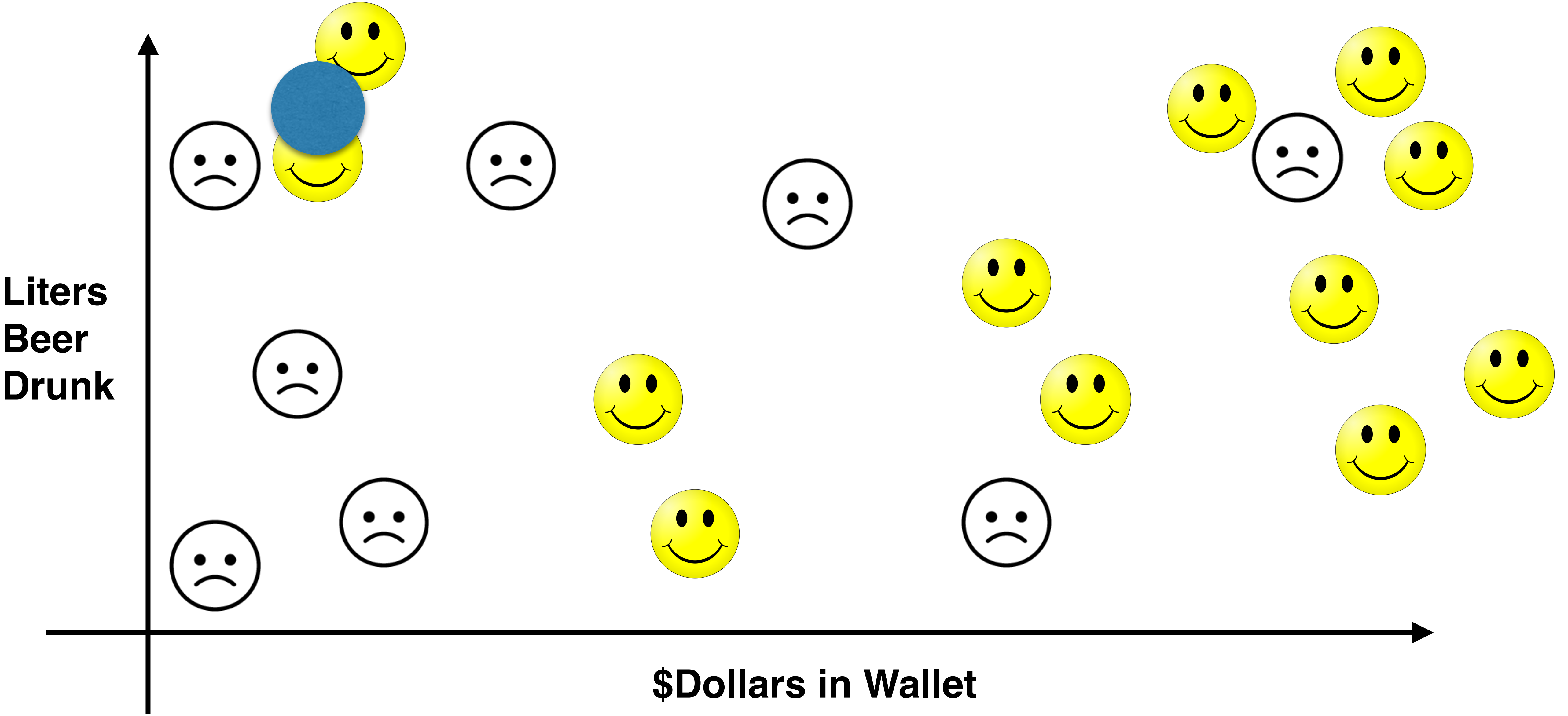
K NEAREST NEIGHBORS - HAPPY OR SAD? K = 2



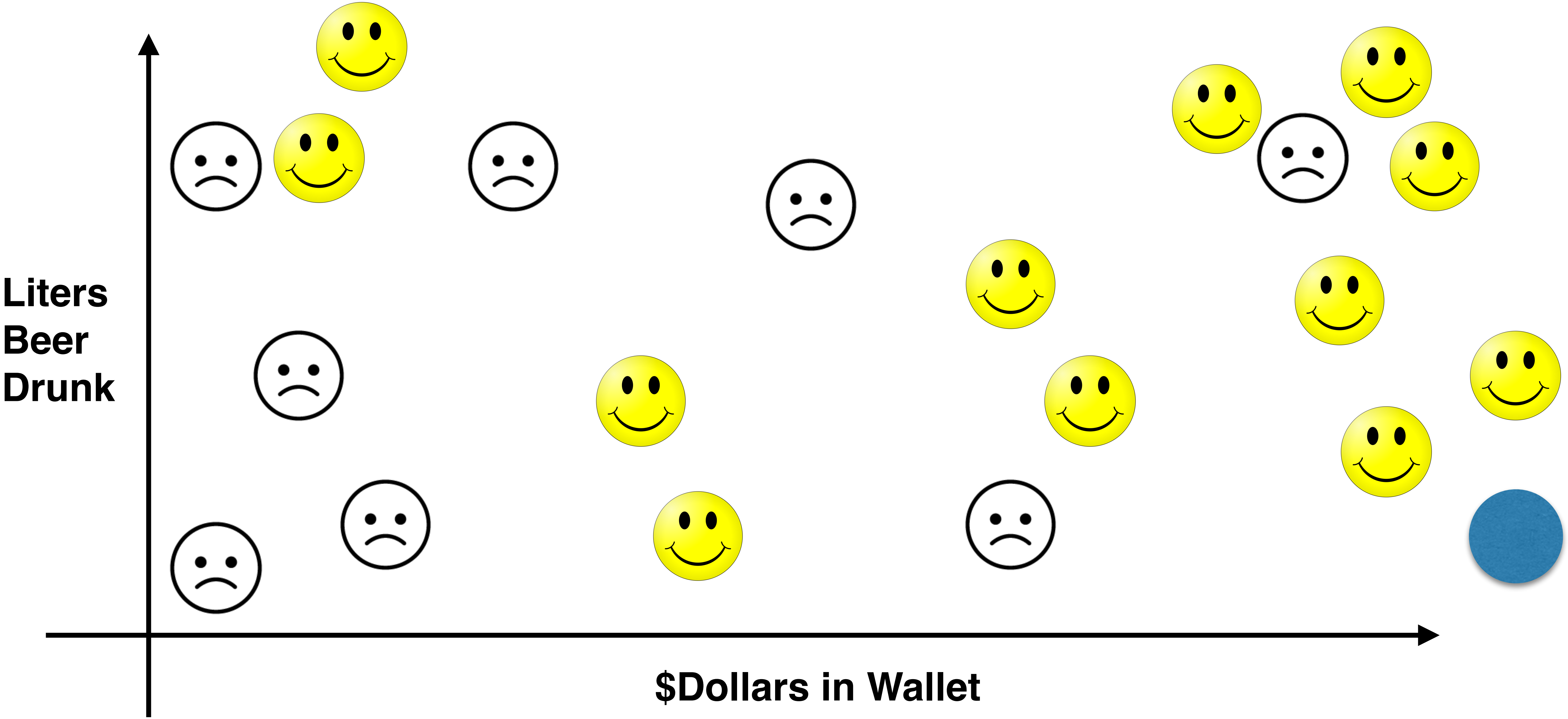
K NEAREST NEIGHBORS - HAPPY OR SAD? K = 20



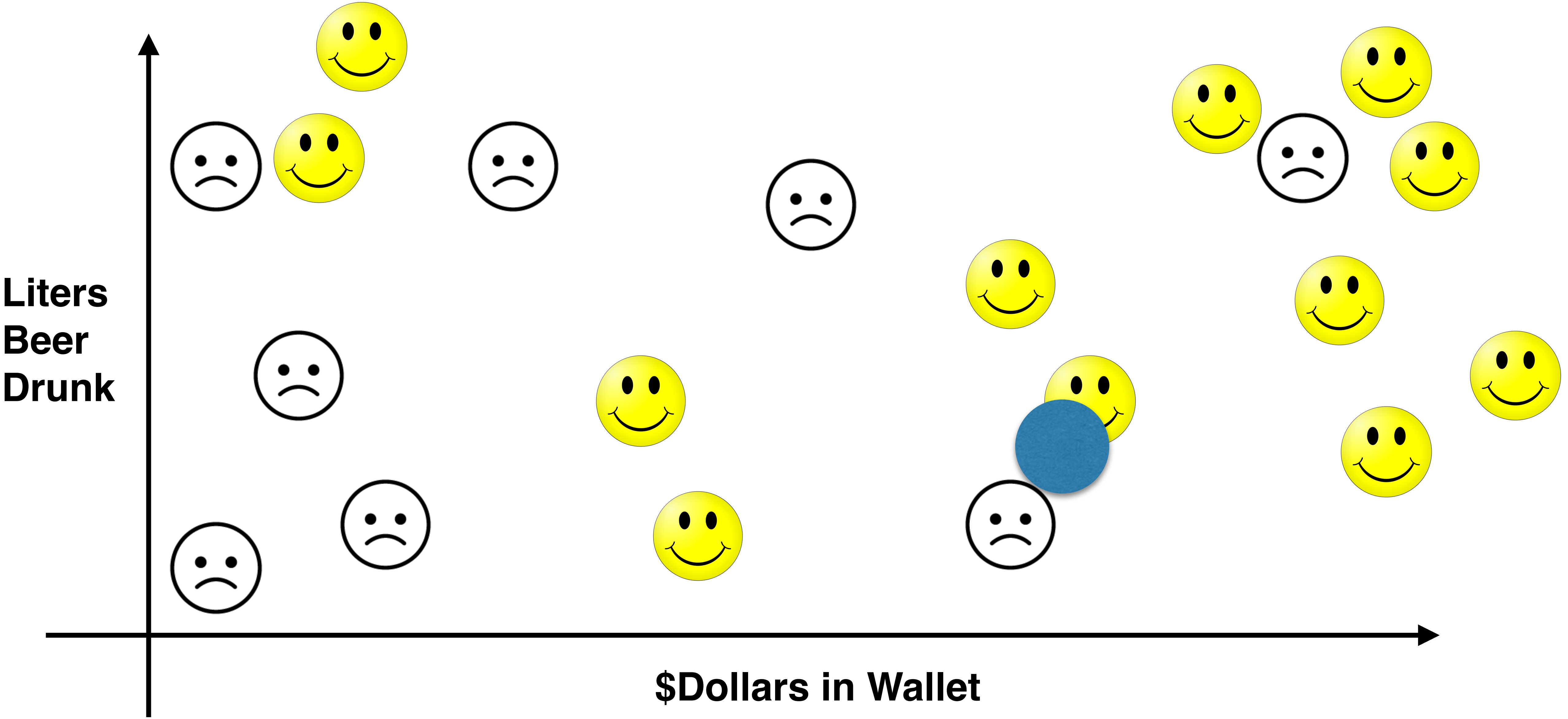
K NEAREST NEIGHBORS - HAPPY OR SAD? K = 20



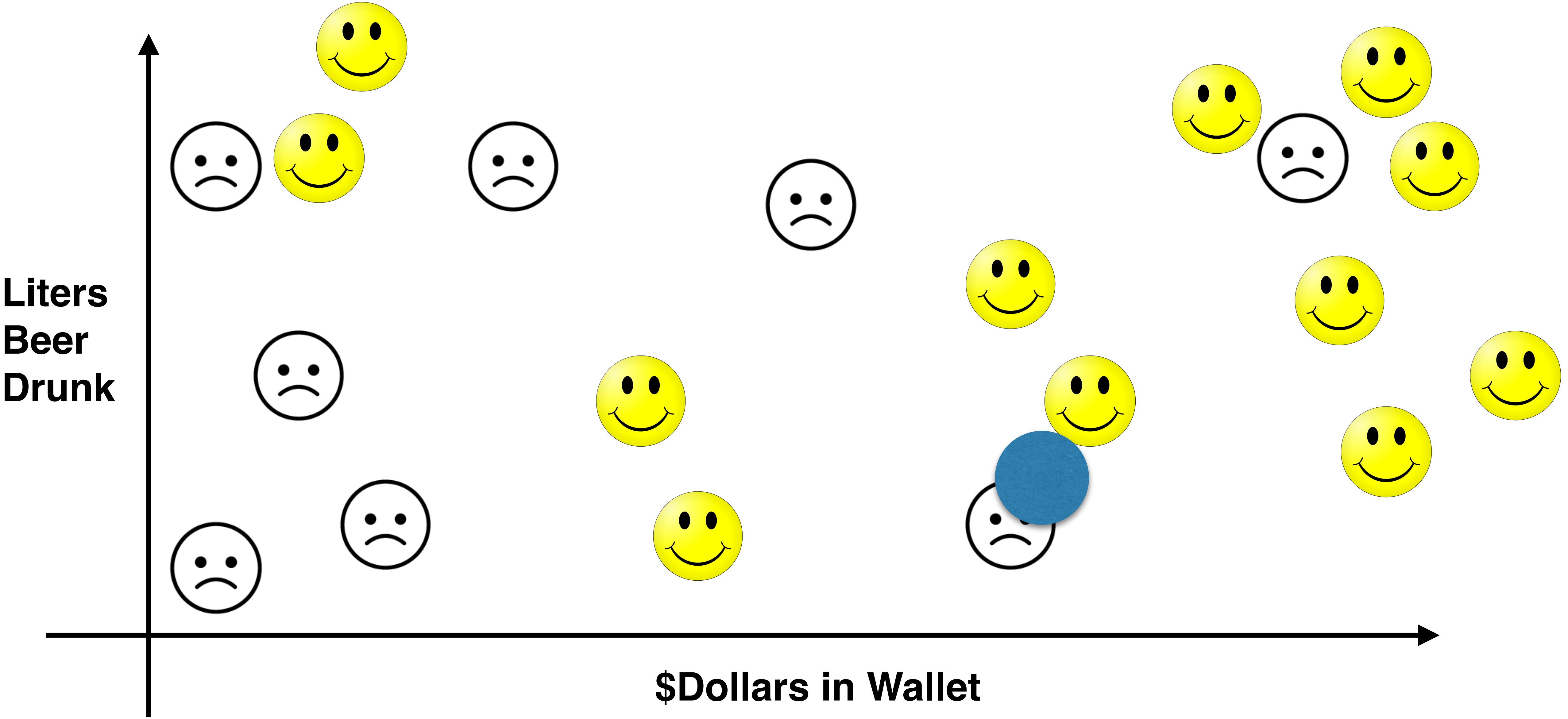
K NEAREST NEIGHBORS - HAPPY OR SAD? K = 20



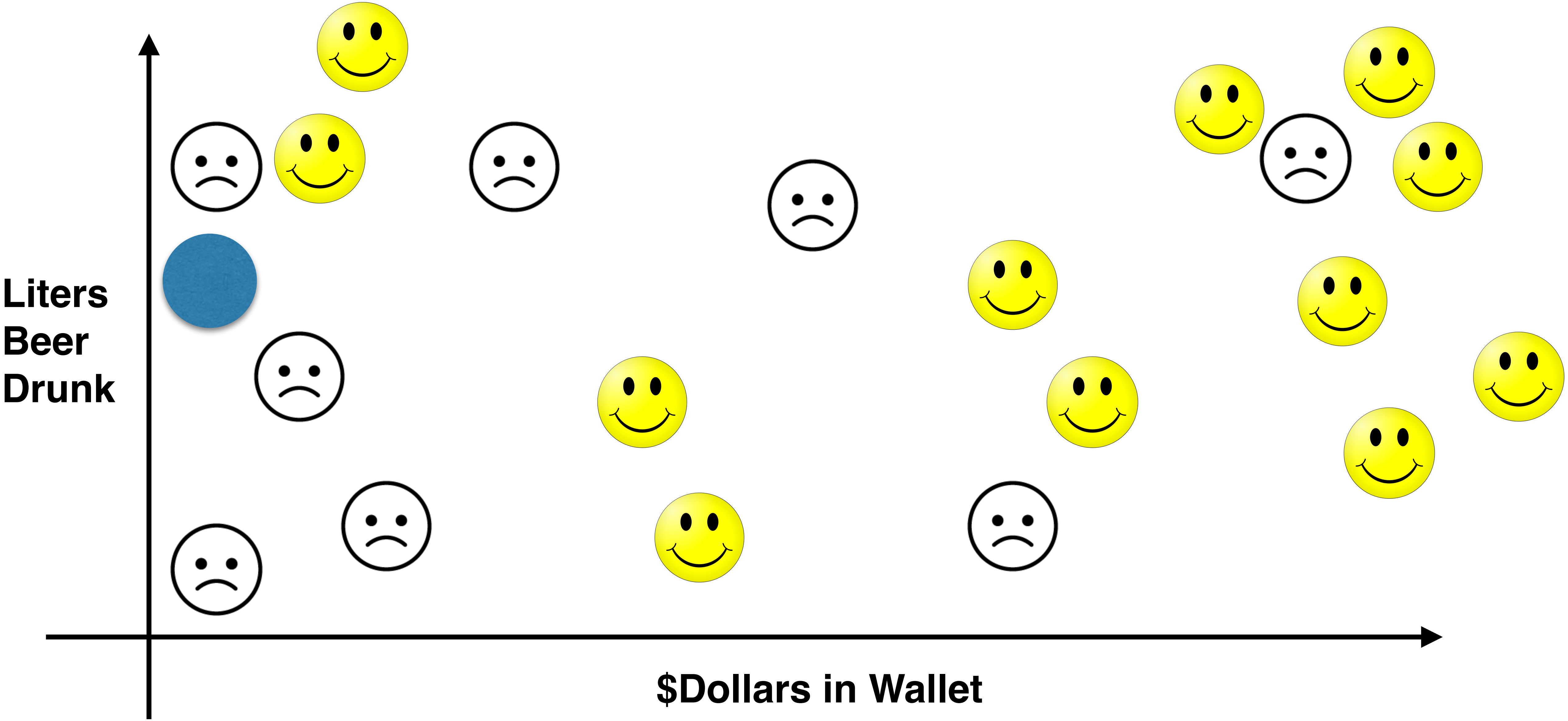
K NEAREST NEIGHBORS - HAPPY OR SAD? K = 1



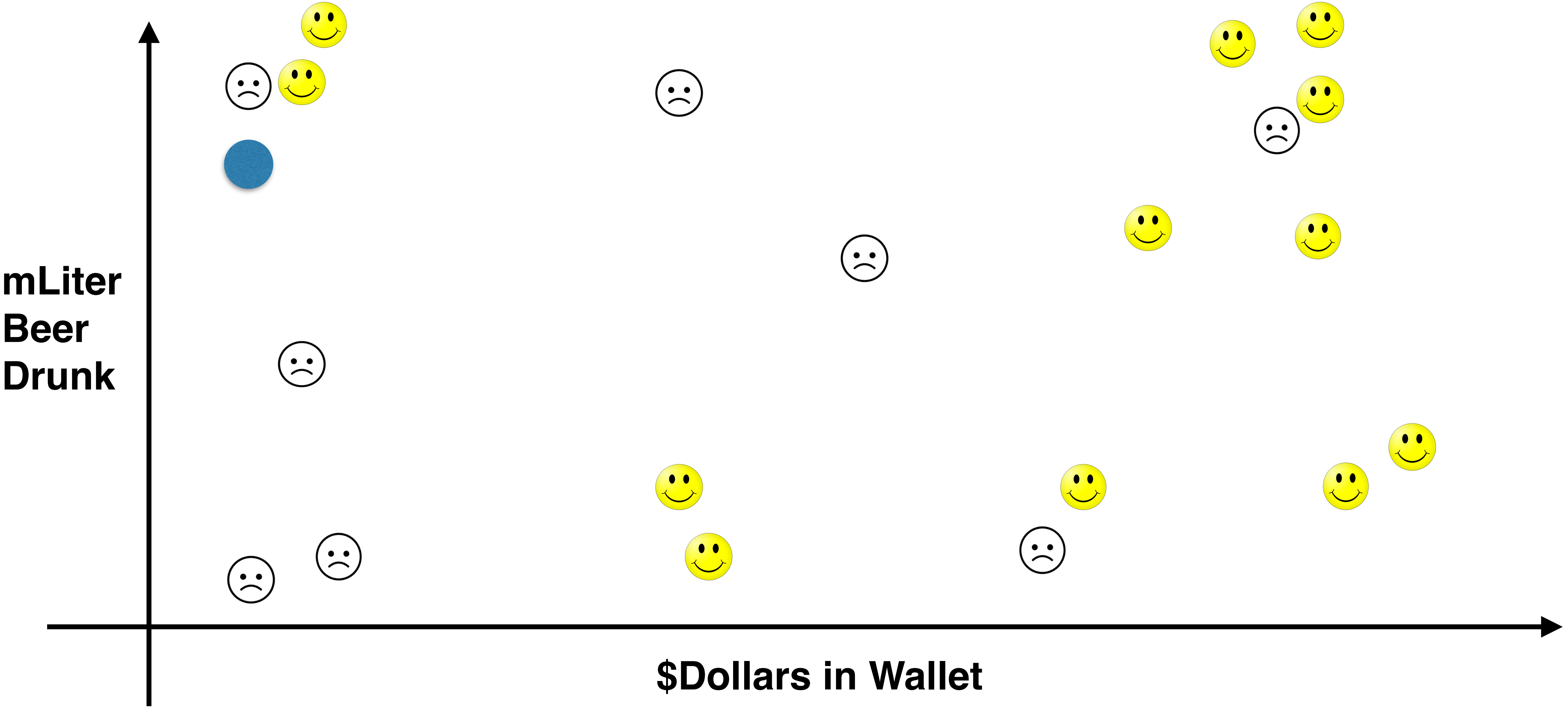
K NEAREST NEIGHBORS - HAPPY OR SAD? K = 1



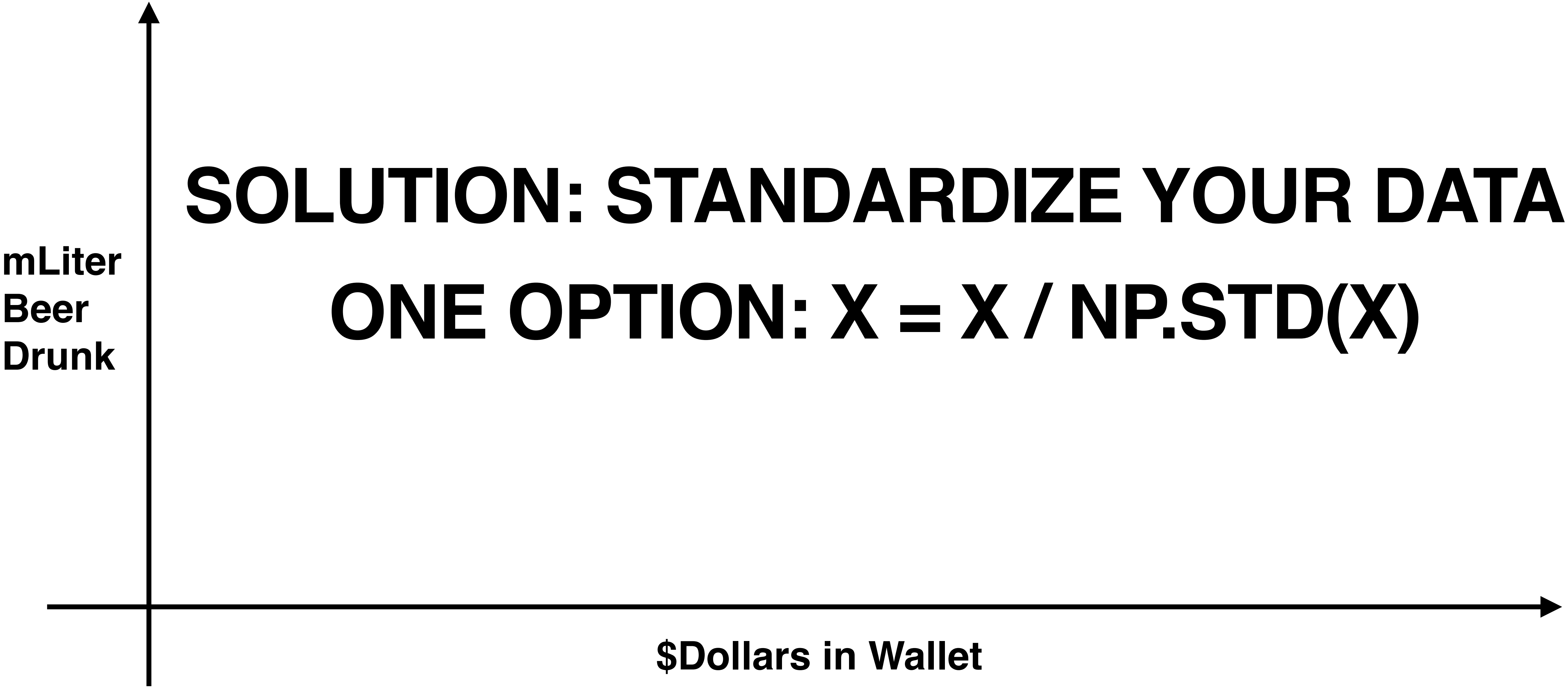
K NEAREST NEIGHBORS - SCALING, K = 2



K NEAREST NEIGHBORS - SCALING



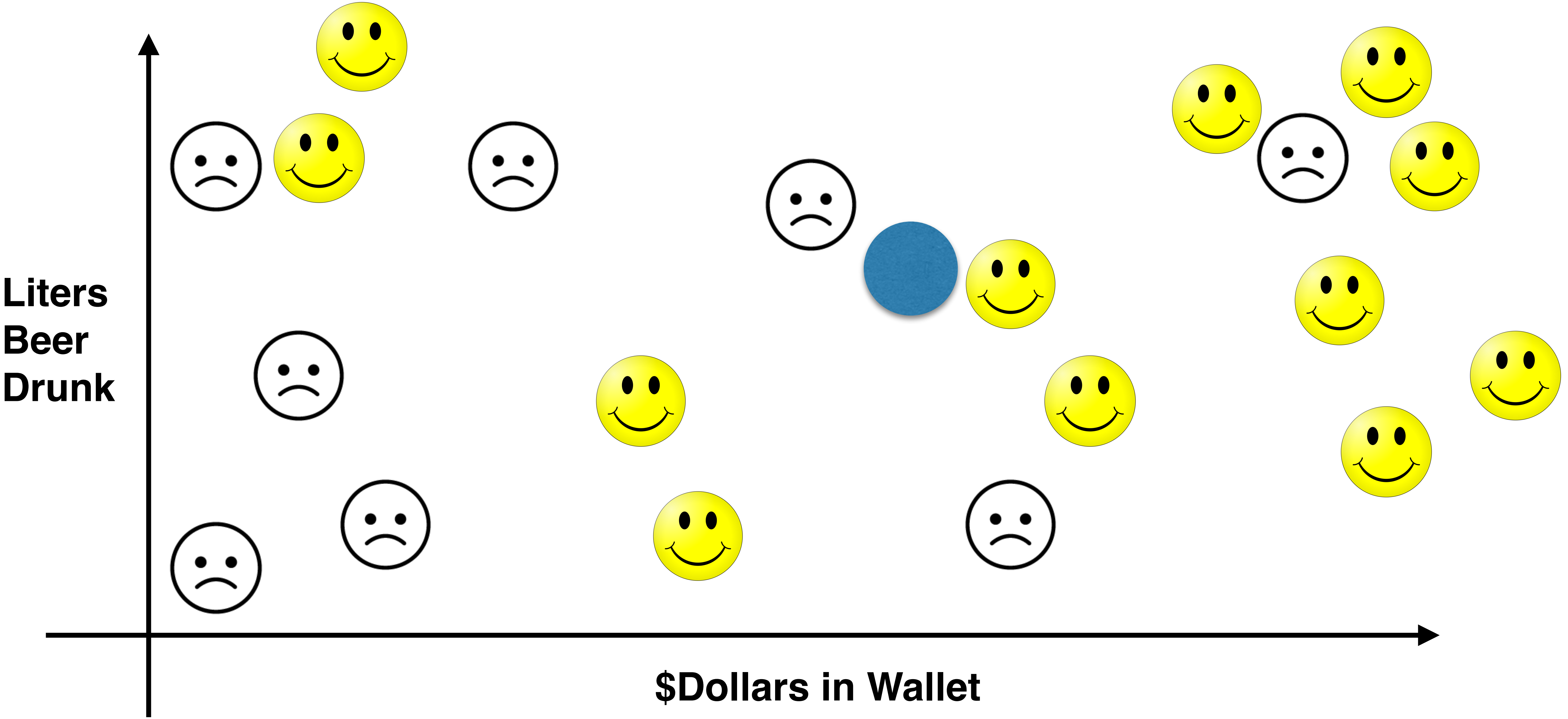
K NEAREST NEIGHBORS - SCALING



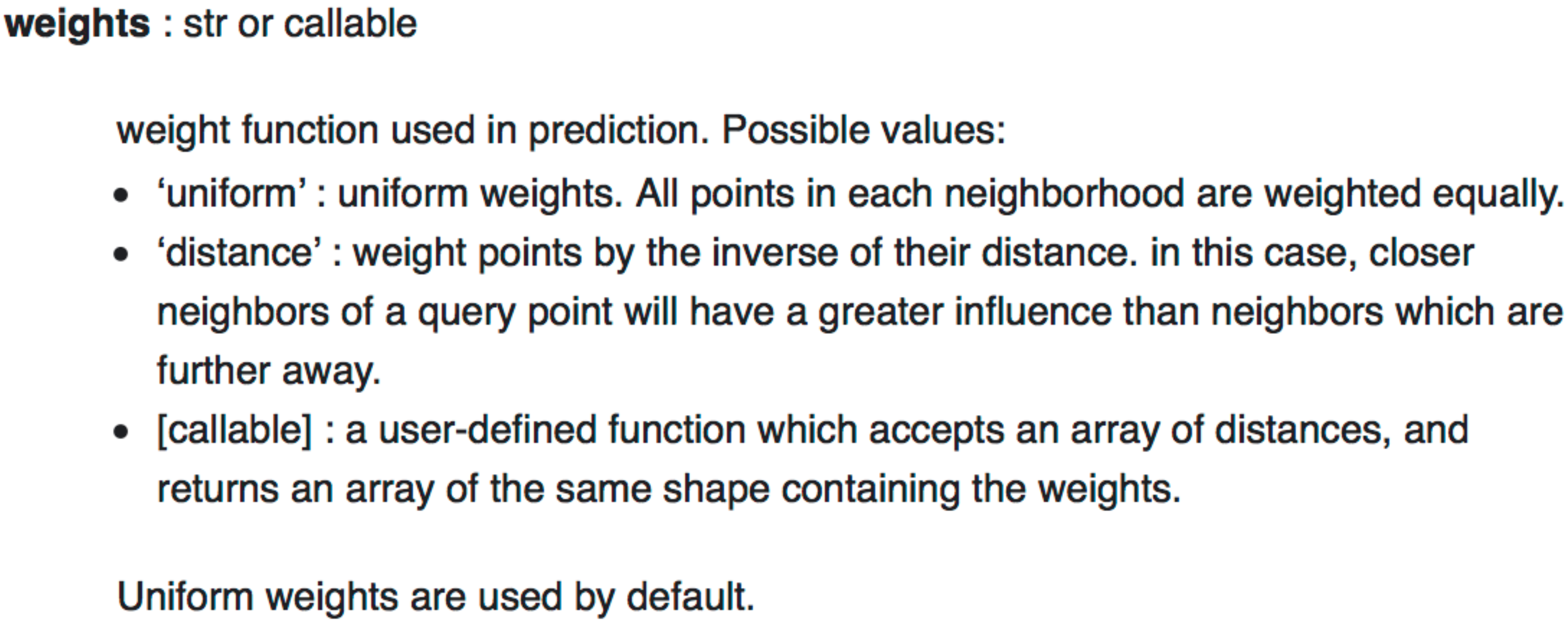
SOLUTION: STANDARDIZE YOUR DATA

ONE OPTION: X = X / NP.STD(X)

K NEAREST NEIGHBORS - WEIGHT FUNCTION, K = 20



K NEAREST NEIGHBORS - WEIGHT FUNCTION, SOLUTION



weights : str or callable

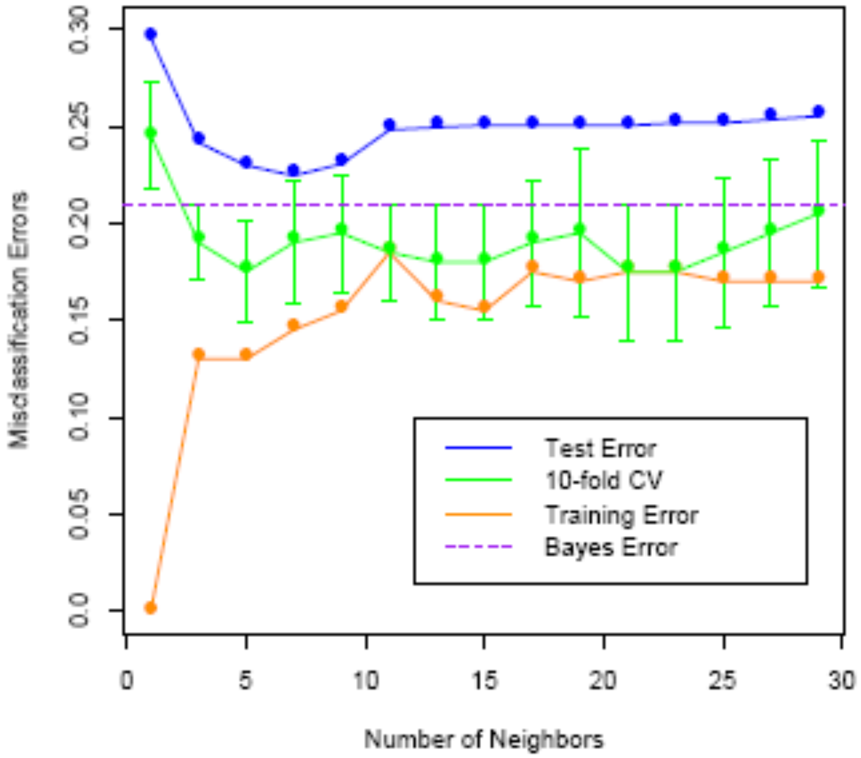
weight function used in prediction. Possible values:

- ‘uniform’ : uniform weights. All points in each neighborhood are weighted equally.
- ‘distance’ : weight points by the inverse of their distance. in this case, closer neighbors of a query point will have a greater influence than neighbors which are further away.
- [callable] : a user-defined function which accepts an array of distances, and returns an array of the same shape containing the weights.

Uniform weights are used by default.

K NEAREST NEIGHBORS - WHAT K????

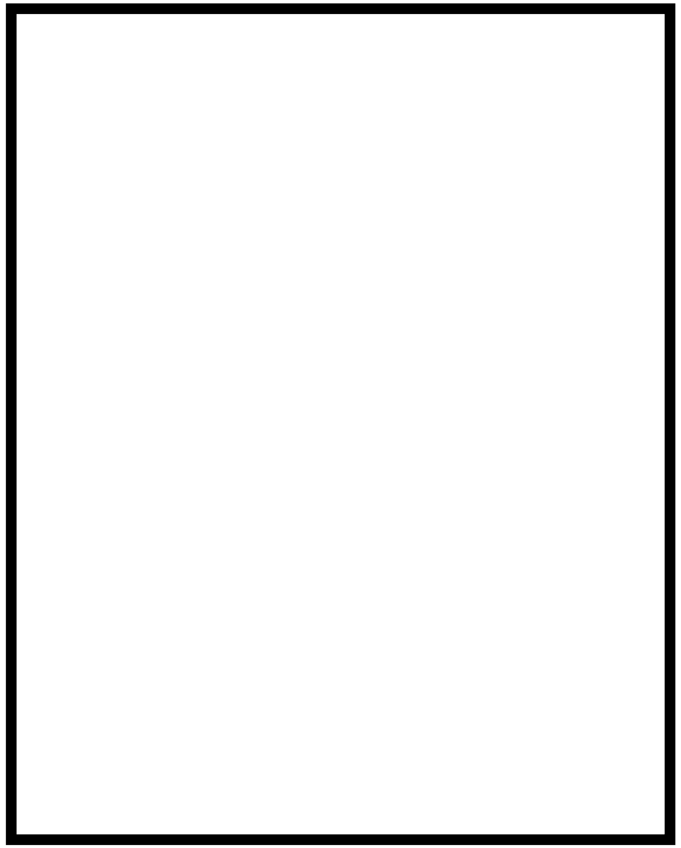
SOLUTION: CROSS VALIDATION (NEXT EXERCISE)



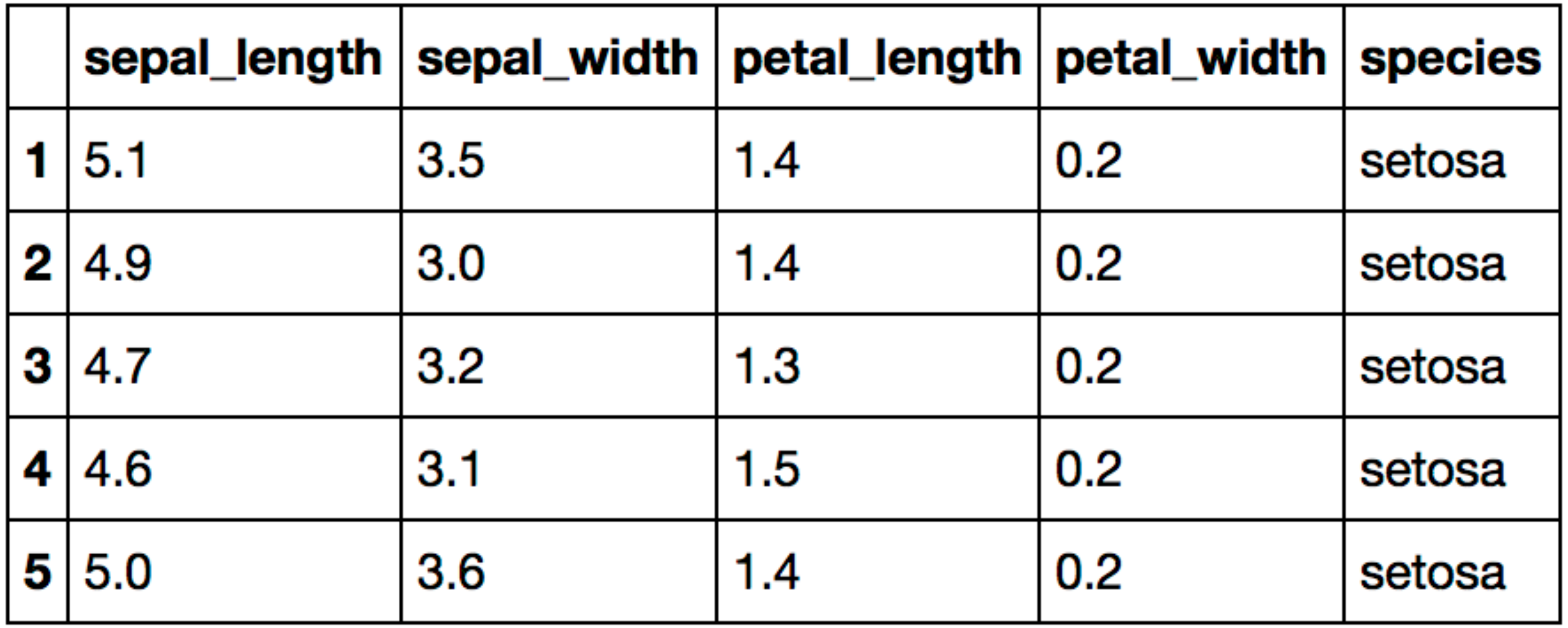
KNN AND PYTHON

WHAT HAPPENS UNDER THE HOOD

KNN CLASSIFICATION - INITIAL DATASET

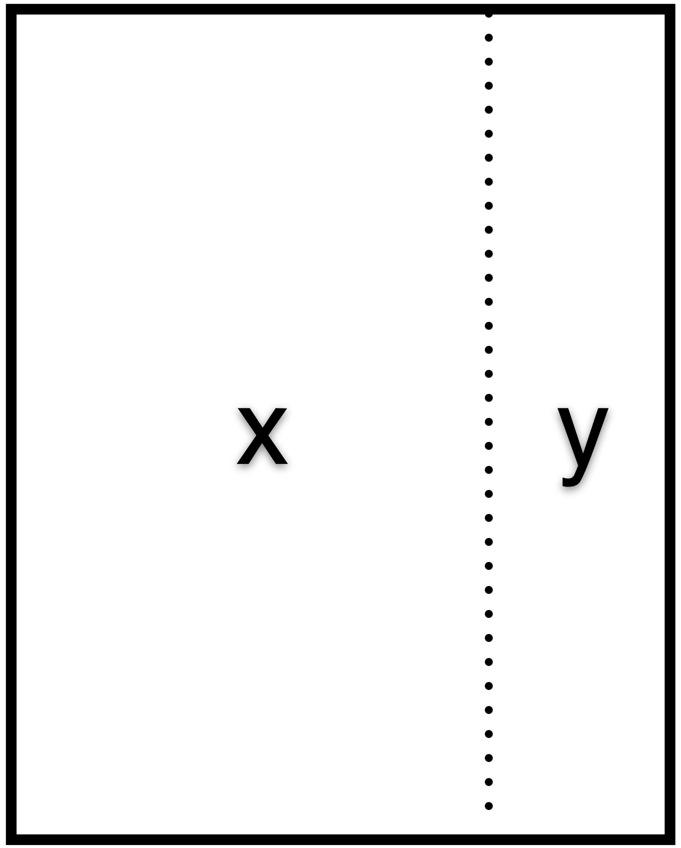


dataset

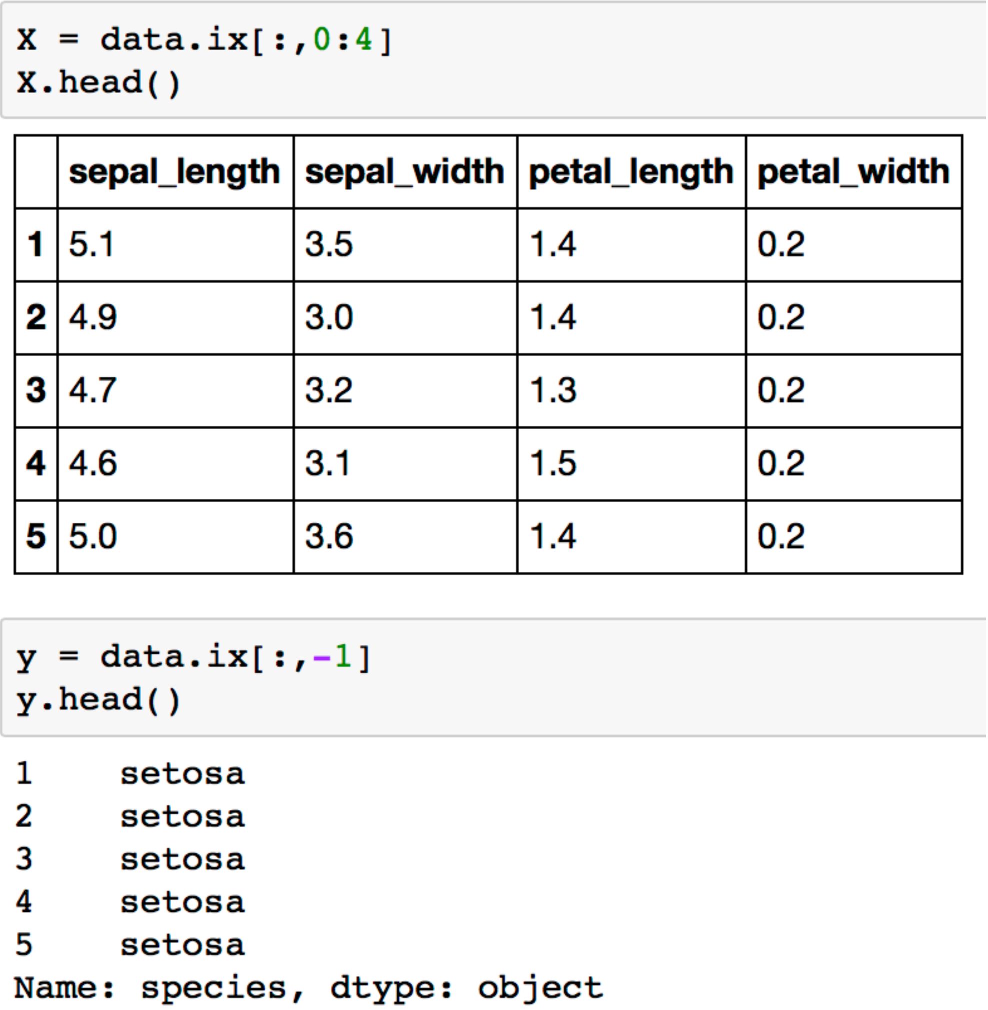


	sepal_length	sepal_width	petal_length	petal_width	species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa

KNN CLASSIFICATION - DETERMINE DATA AND LABELS



dataset



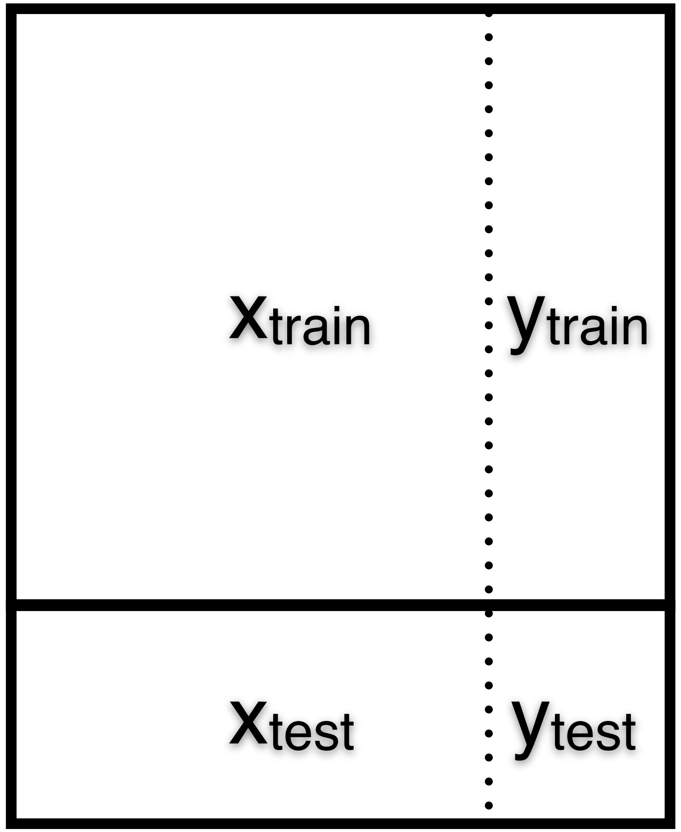
```
X = data.ix[:,0:4]
X.head()
```

	sepal_length	sepal_width	petal_length	petal_width
1	5.1	3.5	1.4	0.2
2	4.9	3.0	1.4	0.2
3	4.7	3.2	1.3	0.2
4	4.6	3.1	1.5	0.2
5	5.0	3.6	1.4	0.2

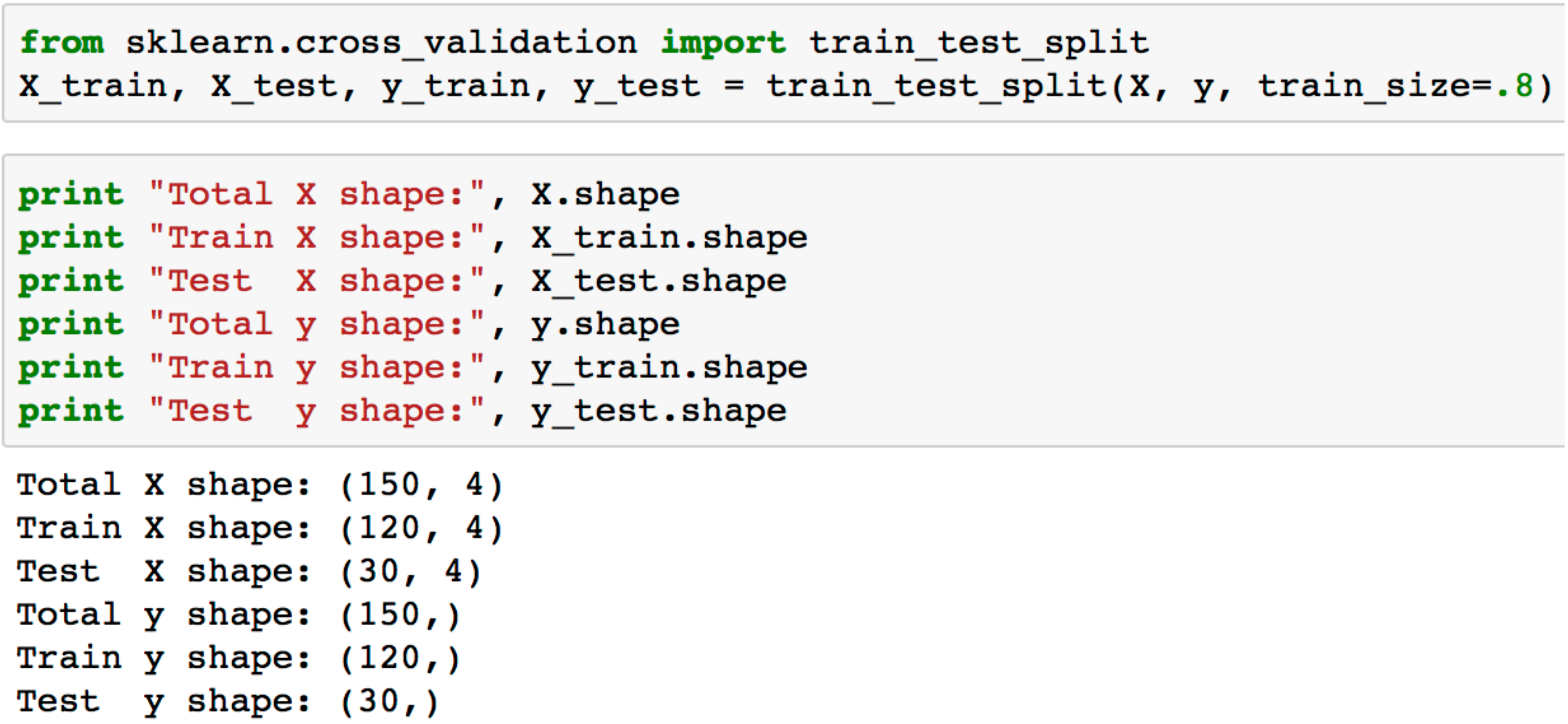
```
y = data.ix[:,-1]
y.head()
```

```
1     setosa
2     setosa
3     setosa
4     setosa
5     setosa
Name: species, dtype: object
```

KNN CLASSIFICATION - SPLIT INTO TRAIN AND TEST



split dataset

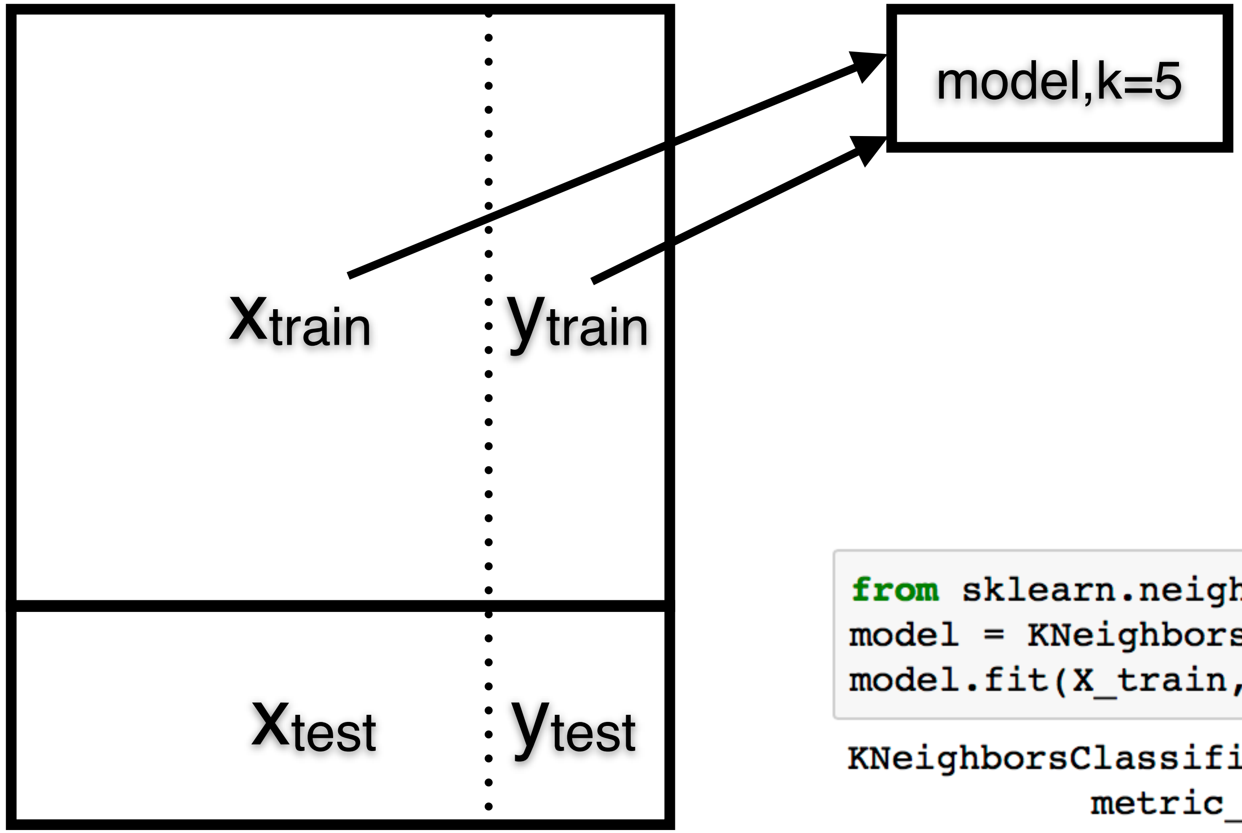


```
from sklearn.cross_validation import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, train_size=.8)
```

```
print "Total X shape:", X.shape
print "Train X shape:", X_train.shape
print "Test  X shape:", X_test.shape
print "Total y shape:", y.shape
print "Train y shape:", y_train.shape
print "Test  y shape:", y_test.shape
```

```
Total X shape: (150, 4)
Train X shape: (120, 4)
Test  X shape: (30, 4)
Total y shape: (150,)
Train y shape: (120,)
Test  y shape: (30,)
```

KNN CLASSIFICATION - BUILD MODEL ON TRAIN DATA

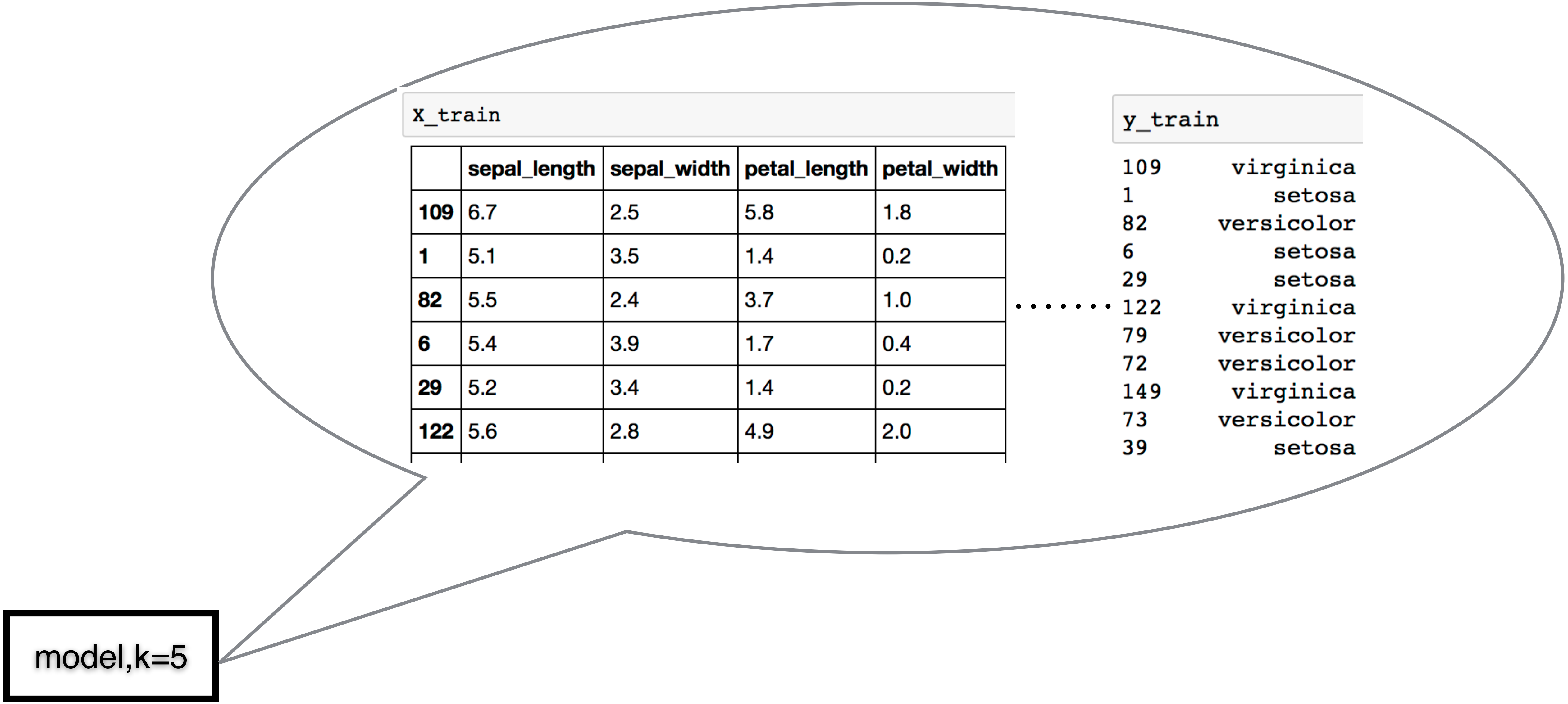


fit model on train

```
from sklearn.neighbors import KNeighborsClassifier
model = KNeighborsClassifier(n_neighbors=5)
model.fit(X_train, y_train)
```

```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
           metric_params=None, n_neighbors=5, p=2, weights='uniform')
```

KNN CLASSIFICATION - WHAT'S INSIDE THE MODEL?



```
X_train
```

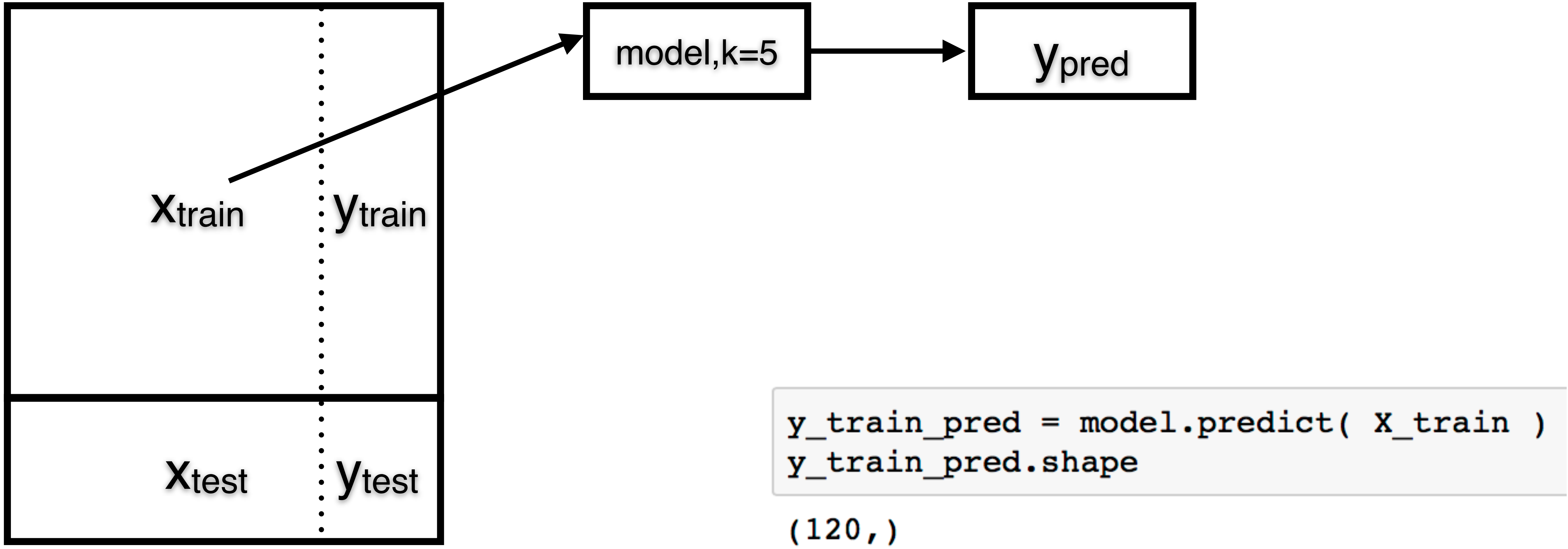
	sepal_length	sepal_width	petal_length	petal_width
109	6.7	2.5	5.8	1.8
1	5.1	3.5	1.4	0.2
82	5.5	2.4	3.7	1.0
6	5.4	3.9	1.7	0.4
29	5.2	3.4	1.4	0.2
122	5.6	2.8	4.9	2.0

```
y_train
```

```
109     virginica
1          setosa
82     versicolor
6          setosa
29         setosa
122     virginica
79     versicolor
72     versicolor
149     virginica
73     versicolor
39         setosa
```

model,k=5

KNN CLASSIFICATION - PREDICT MODEL ON TRAIN DATA



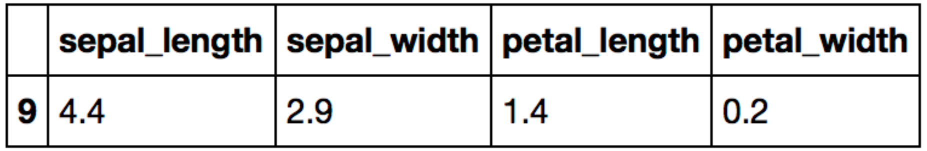
```
y_train_pred = model.predict( X_train )
y_train_pred.shape
```

```
(120,)
```

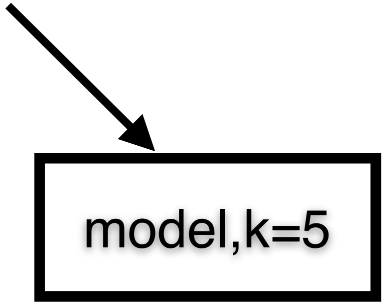
predict y for x_{train}

KNN CLASSIFICATION - WHAT HAPPENS WHEN YOU PREDICT?

Predict the class of this row



	sepal_length	sepal_width	petal_length	petal_width
9	4.4	2.9	1.4	0.2

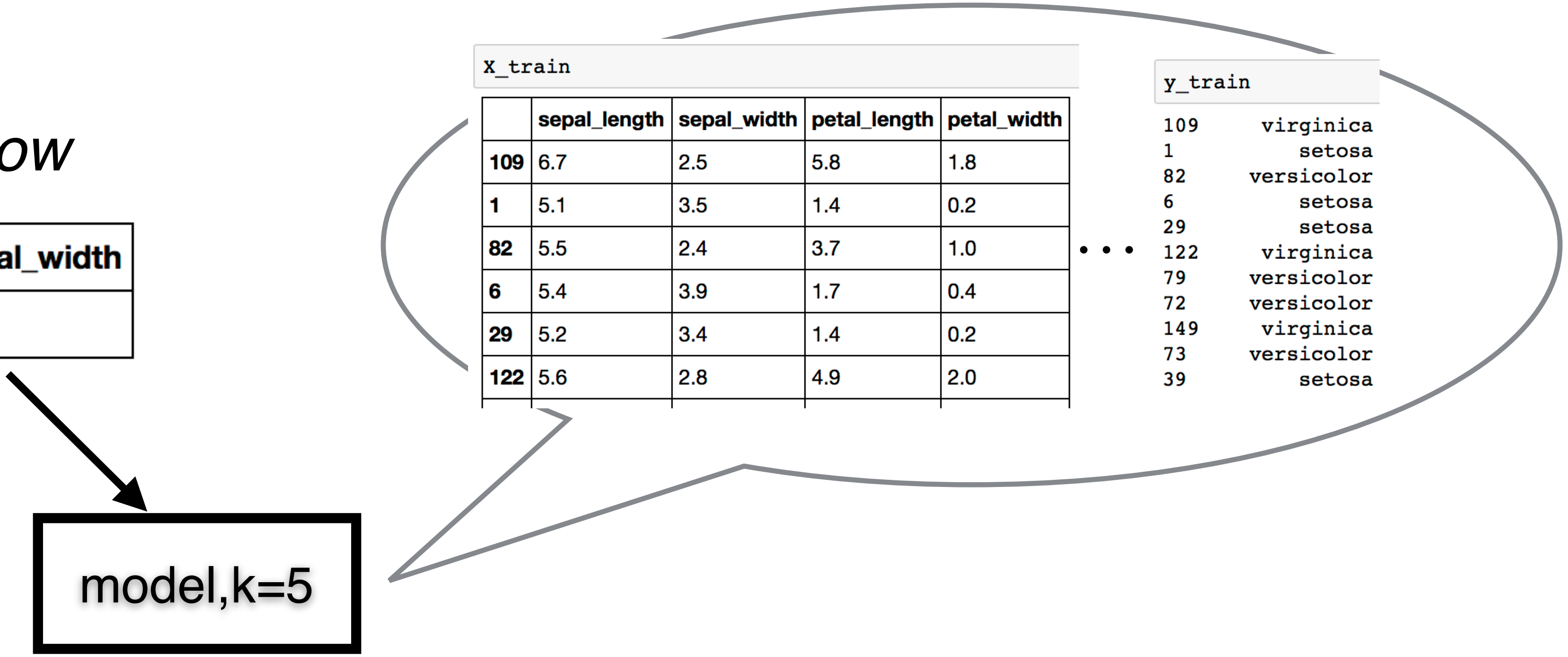


model,k=5

KNN CLASSIFICATION - WHAT HAPPENS WHEN YOU PREDICT?

Predict the class of this row

	sepal_length	sepal_width	petal_length	petal_width
9	4.4	2.9	1.4	0.2



```
X_train
```

	sepal_length	sepal_width	petal_length	petal_width
109	6.7	2.5	5.8	1.8
1	5.1	3.5	1.4	0.2
82	5.5	2.4	3.7	1.0
6	5.4	3.9	1.7	0.4
29	5.2	3.4	1.4	0.2
122	5.6	2.8	4.9	2.0

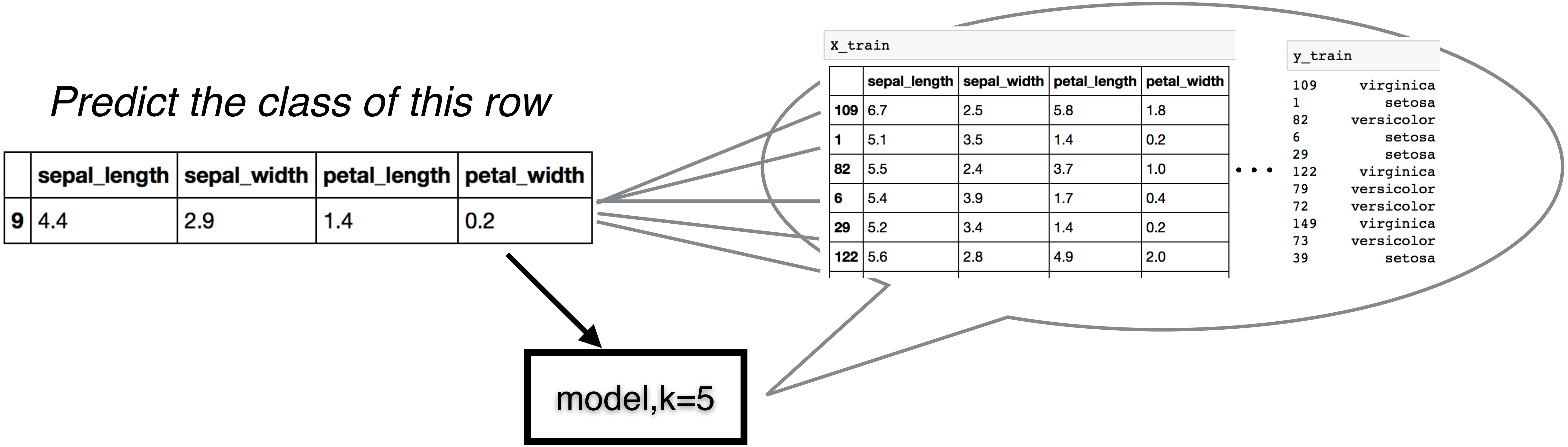
. . .

```
y_train
```

```
109     virginica
1          setosa
82     versicolor
6          setosa
29         setosa
122     virginica
79     versicolor
72     versicolor
149     virginica
73     versicolor
39         setosa
```

KNN CLASSIFICATION - WHAT HAPPENS WHEN YOU PREDICT?

Predict the class of this row



	sepal_length	sepal_width	petal_length	petal_width
9	4.4	2.9	1.4	0.2

X_train

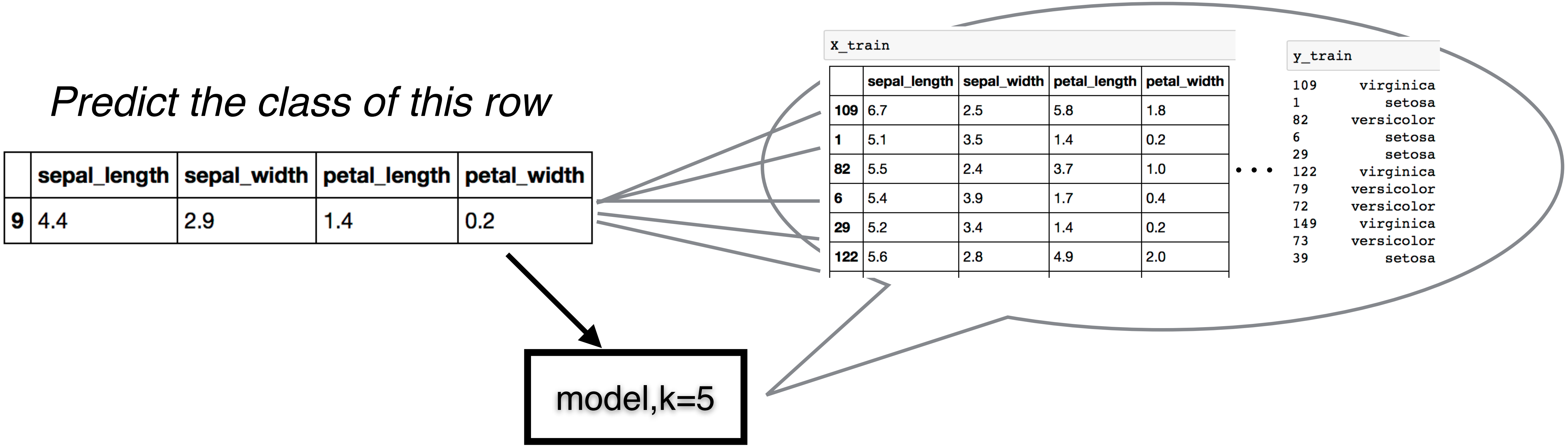
	sepal_length	sepal_width	petal_length	petal_width
109	6.7	2.5	5.8	1.8
1	5.1	3.5	1.4	0.2
82	5.5	2.4	3.7	1.0
6	5.4	3.9	1.7	0.4
29	5.2	3.4	1.4	0.2
122	5.6	2.8	4.9	2.0

. . .

y_train

```
109     virginica
1          setosa
82     versicolor
6          setosa
29         setosa
122     virginica
79     versicolor
72     versicolor
149     virginica
73     versicolor
39         setosa
```

KNN CLASSIFICATION - WHAT HAPPENS WHEN YOU PREDICT?



Predict the class of this row

	sepal_length	sepal_width	petal_length	petal_width
9	4.4	2.9	1.4	0.2

model,k=5

X_train

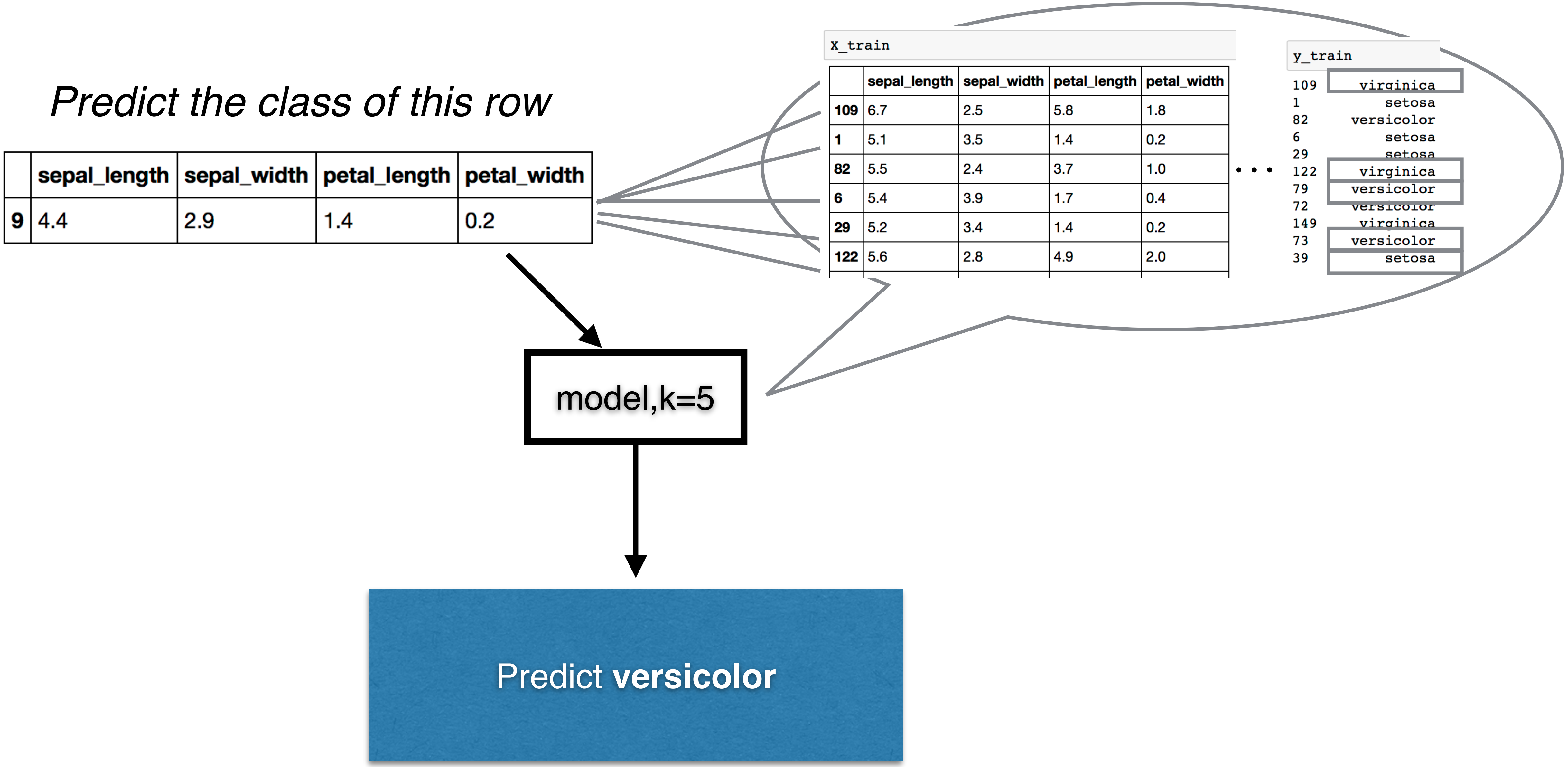
	sepal_length	sepal_width	petal_length	petal_width
109	6.7	2.5	5.8	1.8
1	5.1	3.5	1.4	0.2
82	5.5	2.4	3.7	1.0
6	5.4	3.9	1.7	0.4
29	5.2	3.4	1.4	0.2
122	5.6	2.8	4.9	2.0

y_train

```
109     virginica
1          setosa
82     versicolor
6          setosa
29         setosa
122     virginica
79     versicolor
72     versicolor
149     virginica
73     versicolor
39         setosa
```

$$Distance = \sqrt{\begin{array}{l}(sepal_{length}(x_j) - sepal_{length}(x_k))^2 + (sepal_{width}(x_j) - sepal_{width}(x_k))^2 \\ + (petal_{length}(x_j) - petal_{length}(x_k))^2 + (petal_{width}(x_j) - petal_{width}(x_k))^2\end{array}}$$

KNN CLASSIFICATION - WHAT HAPPENS WHEN YOU PREDICT?



Predict the class of this row

	sepal_length	sepal_width	petal_length	petal_width
9	4.4	2.9	1.4	0.2

```
X_train
```

	sepal_length	sepal_width	petal_length	petal_width
109	6.7	2.5	5.8	1.8
1	5.1	3.5	1.4	0.2
82	5.5	2.4	3.7	1.0
6	5.4	3.9	1.7	0.4
29	5.2	3.4	1.4	0.2
122	5.6	2.8	4.9	2.0

. . .

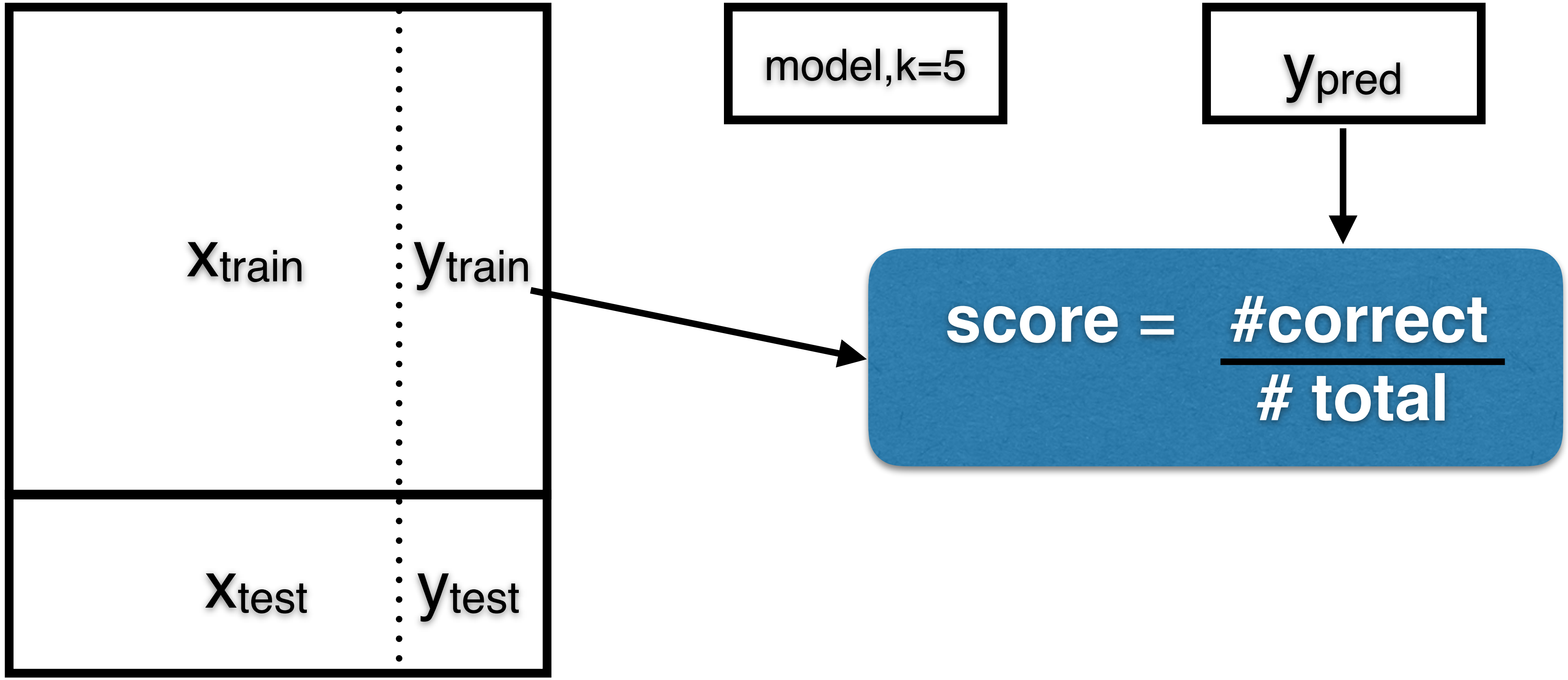
```
y_train
```

```
109     virginica
1          setosa
82     versicolor
6          setosa
29         setosa
122     virginica
79     versicolor
72     versicolor
149     virginica
73     versicolor
39         setosa
```

model,k=5

Predict **versicolor**

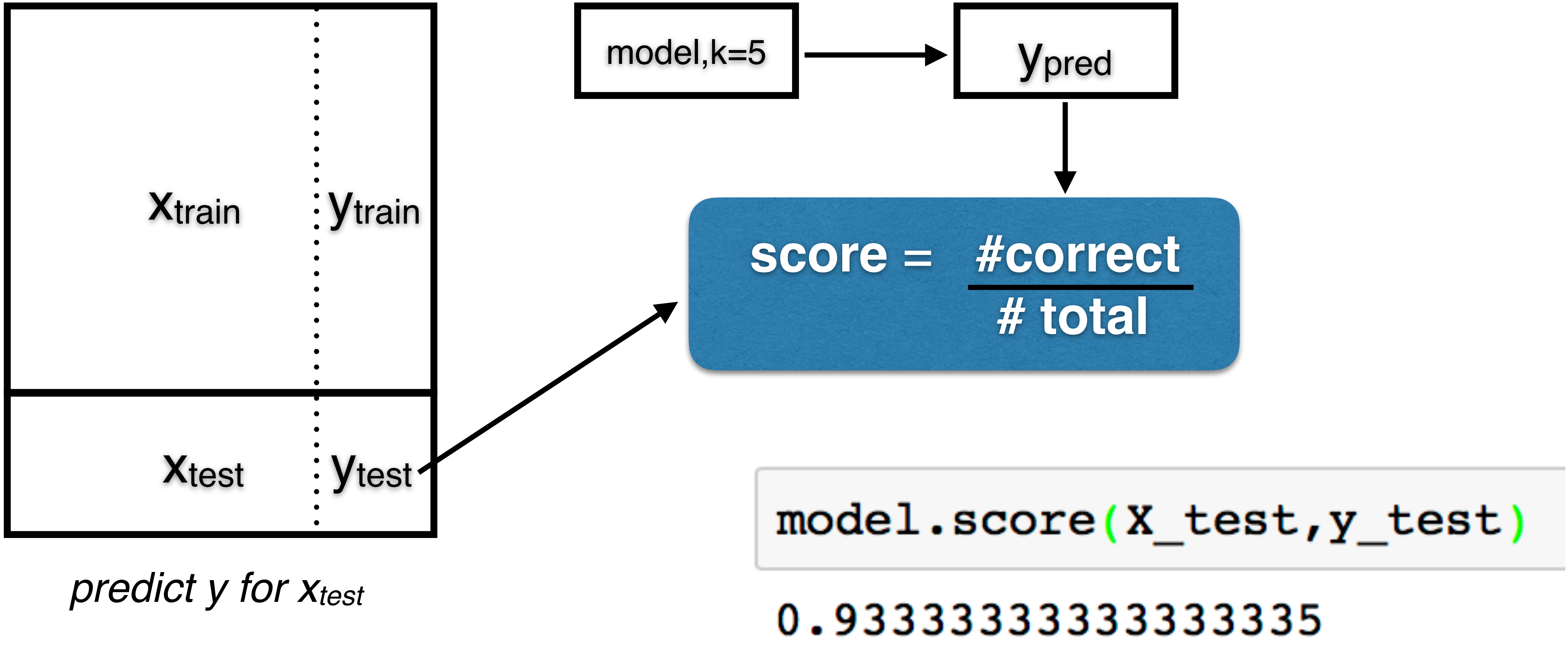
KNN CLASSIFICATION - LASTLY, GENERATE THE SCORE



$$\text{score} = \frac{\#\text{correct}}{\#\ \text{total}}$$

predict y for x_{train}

KNN CLASSIFICATION - REPEAT ON THE TEST SET



$$\text{score} = \frac{\#\text{correct}}{\#\ \text{total}}$$

predict y for x_{test}

```
model.score(X_test,y_test)
```

```
0.93333333333333335
```

LAB

KNN DIGITS LAB

LAB

VISUALIZATION LAB

LAB

NBA EXERCISE

EXIT

- Exit tickets
    - DAT 1, Lesson 4, EDA
- Project Milestone 1, Due Dec 21
- Office Hours, Thursday 5pm to 8pm