Lecture 13
K-Nearest Neighbors

Dennis Sun
Stanford University
DATASCI / STATS 112

February 10, 2023

1. Review
2. A Controversy in the Wine World
3. Our First Regressor
4. Reminders

1 Review

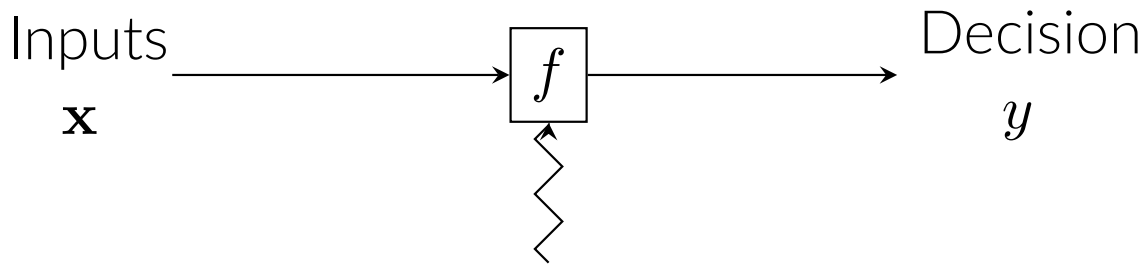
2 A Controversy in the Wine World

3 Our First Regressor

4 Reminders

What is Machine Learning?

Learning is: coming up with a rule for making a decision based on a set of inputs.



Goal of Machine Learning:
Estimate the rule f
from data $(\mathbf{x}_i, y_i)$.

The decision y is typically called the **target** or the **label**.

Two Types of Machine Learning Problems

Machine learning problems are grouped into two types, based on the type of y:

Regression: The label y is quantitative.

Classification: The label y is categorical.

Note that the input features $\mathbf{x}$ may be categorical, quantitative, or a mix of the two.

1 Review

2 A Controversy in the Wine World

3 Our First Regressor

4 Reminders

Orley Ashenfelter
Economics Professor

In 1991, Orley Ashenfelter predicted that the 1986 vintage of Bordeaux wines would be disappointing.

He did this without tasting a drop of the wine.

Robert Parker
The Wine Advocate

Wine critics were outraged.

Robert Parker had predicted that the 1986 vintage would be “very good and sometimes exceptional” based on tasting an early sample.

How did Ashenfelter make this prediction?

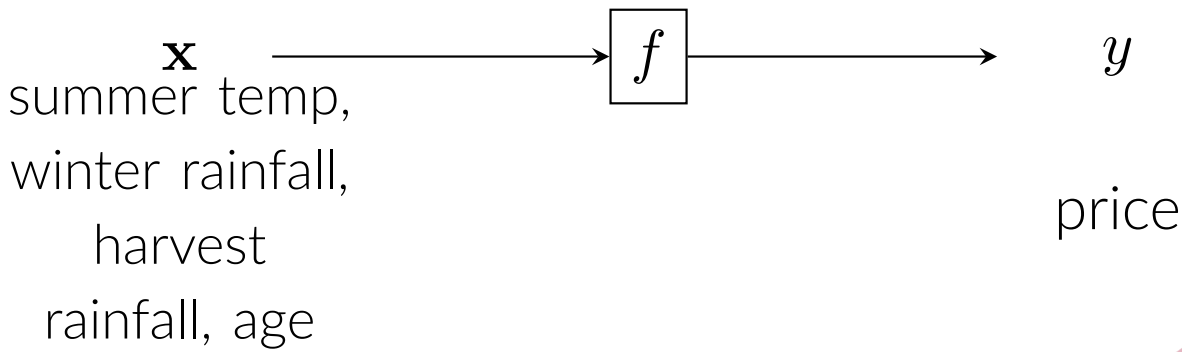
Ashenfelter collected data on summer temperature and winter rainfall in Bordeaux from 1950 to 1991.

The quality of wines becomes apparent after 10 years. So for vintages up to 1980, he also collected their price.

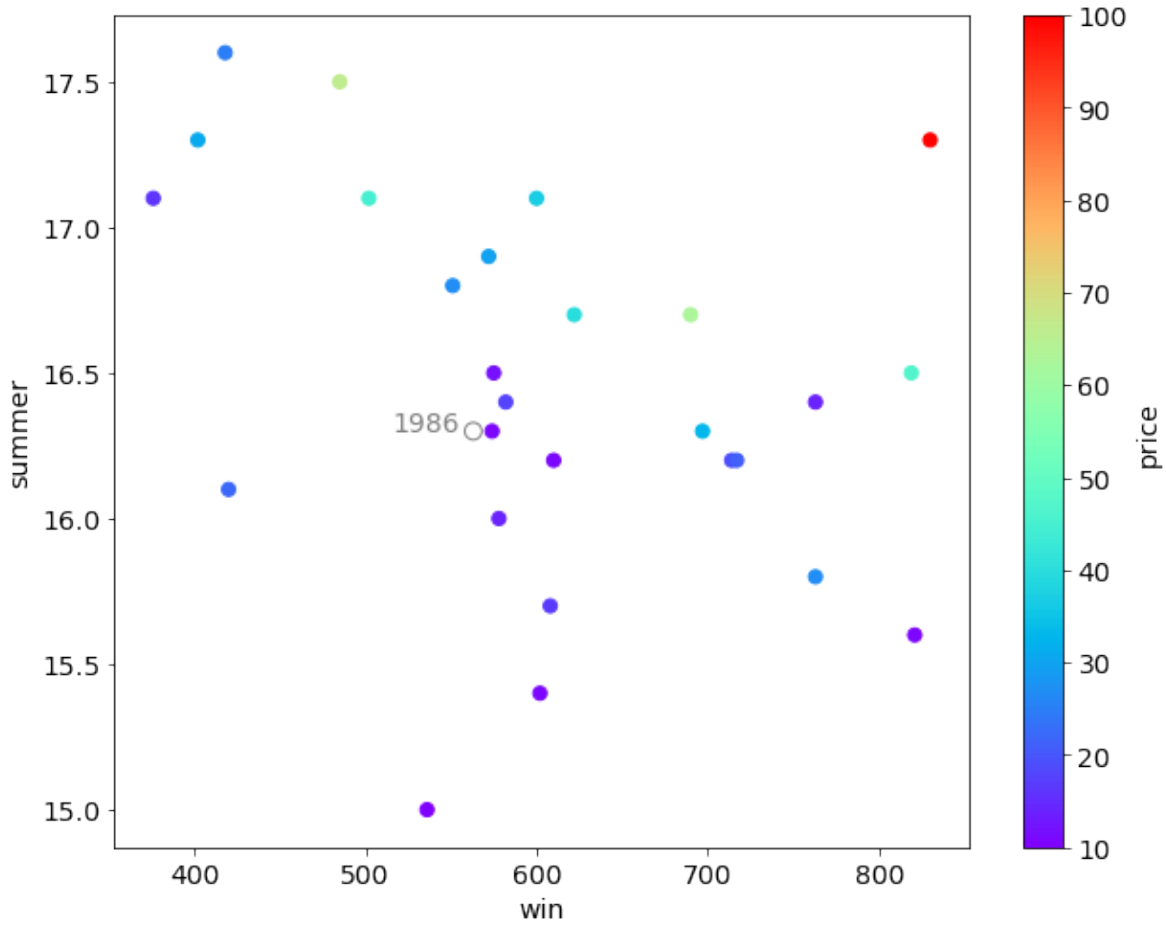
```
df_bordeaux = pd.read_csv(
    "https://dlsun.github.io/pods/data/bordeaux.csv",
    index_col="year")
df_bordeaux
```

year	price	summer	har	sep	win	age
1952	37.0	17.1	160	14.3	600	40
1953	63.0	16.7	80	17.3	690	39
1955	45.0	17.1	130	16.8	502	37
1957	22.0	16.1	110	16.2	420	35
1958	18.0	16.4	187	19.1	582	34
...	...	...	...	...	...	...
1987	NaN	17.0	115	18.9	452	5
1988	NaN	17.1	59	16.8	808	4
1989	NaN	18.6	82	18.4	443	3
1990	NaN	18.7	80	19.3	468	2
1991	NaN	17.7	183	20.4	570	1

38 rows × 6 columns

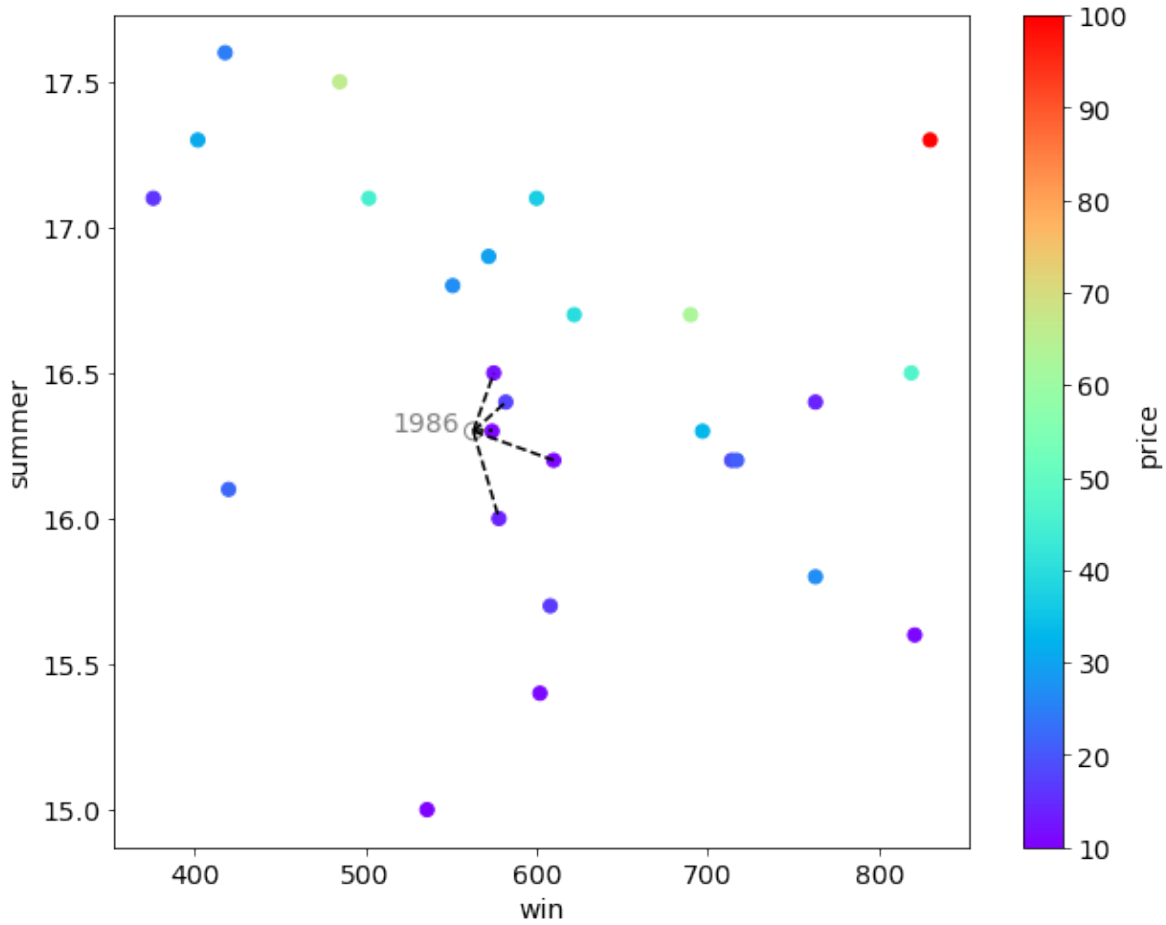


Visualizing the Data



What would you predict is the quality of the 1986 wine?

Visualizing the Data



Insight: The "closest" wines are low quality, so the 1986 vintage is probably low quality as well.

This is the intuition behind k**-nearest neighbors** regression.

1 Review

2 A Controversy in the Wine World

3 Our First Regressor

4 Reminders

K-Nearest Neighbors

The data for which we know the label y is called the **training data**.

The data for which we don't know y (and want to predict it) is called the **test data**.

```
bordeaux_train = df_bordeaux.loc[:1980].copy()
bordeaux_test = df_bordeaux.loc[1981:].copy()
```

K-Nearest Neighbors

1. For each test observation, find the k closest training observations (based on input features $\mathbf{x}$, like summer temperature and winter rainfall).
2. To predict the test label y (e.g., price), average the labels of these k training observations.

K-Nearest Neighbors from Scratch

Recall: Before computng distances, we should scale the variables.

```
X_train = bordeaux_train[["win", "summer"]]
y_train = bordeaux_train["price"]

# Standardize the features.
X_train_mean = X_train.mean()
X_train_sd = X_train.std()
X_train_st = (X_train - X_train_mean) / X_train_sd
```

Now, we get the test data, standardizing it in the same way.

```
x_test = bordeaux_test.loc[1986, ["win", "summer"]]

x_test_st = (x_test - X_train_mean) / X_train_sd
x_test_st
```

```
win      -0.351900
summer   -0.261262
dtype: float64
```

K-Nearest Neighbors from Scratch

Now we calculate the (Euclidean) distances between the test vintage and the vintages in the training data.

```
dists = np.sqrt(((X_train_st - x_test_st) ** 2).sum(axis=1))
dists
```

```
year
1952    1.259860
1953    1.159726
1955    1.314727
1957    1.149883
1958    0.212597
          ...
1976    2.288442
1977    2.269387
1978    1.729248
1979    1.203287
1980    0.474508
Length: 27, dtype: float64
```

Now we just need to sort and take the top $k = 5$.

```
i_nearest = dists.sort_values().index[:5]
i_nearest
```

```
Int64Index([1974, 1958, 1969, 1968, 1980], dtype='int64', name='year')
```

K-Nearest Neighbors from Scratch

Finally, to make a prediction, we average the labels y of these $k = 5$ closest training observations.

```
y_train.loc[i_nearest].mean()
```

```
13.2
```

That's $13.2 for a bottle of wine. This is not a good vintage.

K-Nearest Neighbors in Scikit-Learn

In general, it is easier to fit k-nearest neighbors in Scikit-Learn.

First, using `X_train`, `x_test`, and `y_train` defined above, we scale the data.

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
scaler.fit(X_train)
X_train_st = scaler.transform(X_train)
# Scikit-Learn requires 2D-arrays
X_test_st = scaler.transform(pd.DataFrame([x_test]))
```

Now, we fit k-nearest neighbors using `KNeighborsRegressor`.

```
from sklearn.neighbors import KNeighborsRegressor

model = KNeighborsRegressor(n_neighbors=5)
model.fit(X=X_train_st, y=y_train)
model.predict(X=X_test_st)
```

```
array([13.2])
```

Pipelines in Scikit-Learn

In the code above, we had to be careful to standardize the training and test data in exactly the same way.

Machine learning models involve many more preprocessing steps.

Scikit-Learn's `Pipeline` helps us chain together preprocessing + modeling.

```
from sklearn.pipeline import make_pipeline

pipeline = make_pipeline(
           StandardScaler(),
           KNeighborsRegressor(n_neighbors=5))
```

We can use this `Pipeline` like any other machine learning model.

```
pipeline.fit(X=X_train, y=y_train)
pipeline.predict(X=pd.DataFrame([x_test]))
```

```
array([13.2])
```

The Ending of the Story

Here is how the vintages rank today:

Vintage	Score	Drink Window	Description
1994	85	Drink	Medium-bodied, with good fruit and firm tannins
1993	82	Past peak	Good color, perfumy, fruity and balanced
1992	72	Past peak	Light, simple and often diluted; early maturing
1991	72	Past peak	Lean, tough and light; stick to top names
1990	97	Drink	Opulent, well-structured and harmonious
1989	98	Drink	Bold, dramatic fruit character; tannic and long-aging
1988	93	Drink	Racy, fruity wines, with firm tannins and typical structure; best in Pessac-Léognan, Pomerol and Pauillac
1987	76	Past peak	Delicate, ripe yet diluted
1986	95	Drink	Powerful, intense and tannic; best in Médoc
1985	93	Drink	Balanced, supple and fruity; defines finesse
1984	70	Past peak	Unripe, astringent and dry

Ashenfelter was wrong that 1986 would be a disappointing vintage!

1 Review

2 A Controversy in the Wine World

3 Our First Regressor

4 Reminders

Reminders

- Assignment 3 is due tonight!
- I will hold extra office hours after class until noon! Stick around after class and follow me to my office.
- Start looking for data sets for the final project. Come talk to me about your ideas!