Data 401
Nonlinear Regression and Classification

Dennis Sun

October 31, 2016

1. Review
2. k-Nearest Neighbors
3. Decision Trees

Linear Methods

Linear Methods

- It's not hard to see why linear regression is "linear":

$$Y_i = \beta_0 + \beta_1 x_{i1} + ... + \beta_p x_{ip}.$$

Linear Methods

- It's not hard to see why linear regression is "linear":

$$Y_i = \beta_0 + \beta_1 x_{i1} + ... + \beta_p x_{ip}.$$

- We also have linear classifiers:

Linear Methods

- It's not hard to see why linear regression is "linear":

$$Y_i = \beta_0 + \beta_1 x_{i1} + ... + \beta_p x_{ip}.$$

- We also have linear classifiers:
  - **Logistic regression**: classify based on whether $p_i > .5$.

$$\log\left(\frac{p_i}{1-p_i}\right) = \beta_0 + \beta_1 x_{i1} + ... + \beta_p x_{ip}.$$

Linear Methods

- It's not hard to see why linear regression is "linear":

$$Y_i = \beta_0 + \beta_1 x_{i1} + ... + \beta_p x_{ip}.$$

- We also have linear classifiers:
  - **Logistic regression**: classify based on whether $p_i > .5$.

$$\log\left(\frac{p_i}{1-p_i}\right) = \beta_0 + \beta_1 x_{i1} + ... + \beta_p x_{ip}.$$

  - **Perceptron**: Fit linear regression to binary data, classify by thresholding predicted values.

Linear Methods

- It's not hard to see why linear regression is "linear":

$$Y_i = \beta_0 + \beta_1 x_{i1} + ... + \beta_p x_{ip}.$$

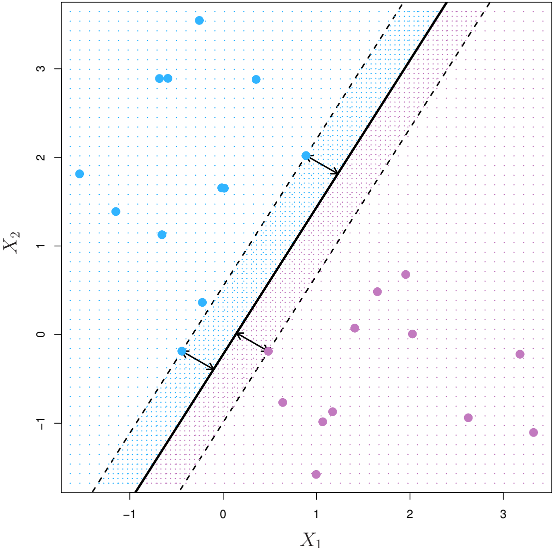
- We also have linear classifiers:
  - **Logistic regression**: classify based on whether $p_i > .5$.

$$\log\left(\frac{p_i}{1-p_i}\right) = \beta_0 + \beta_1 x_{i1} + ... + \beta_p x_{ip}.$$

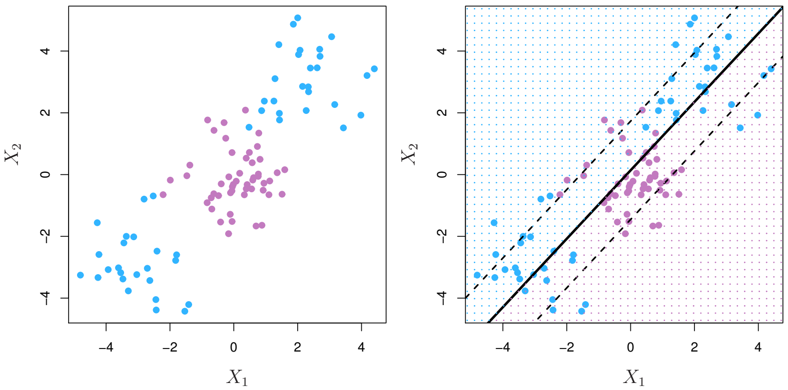
  - **Perceptron**: Fit linear regression to binary data, classify by thresholding predicted values.
  - **Support vector machines** (SVM)

Linear Classifiers

The linear classifiers are linear because their decision boundaries are linear.



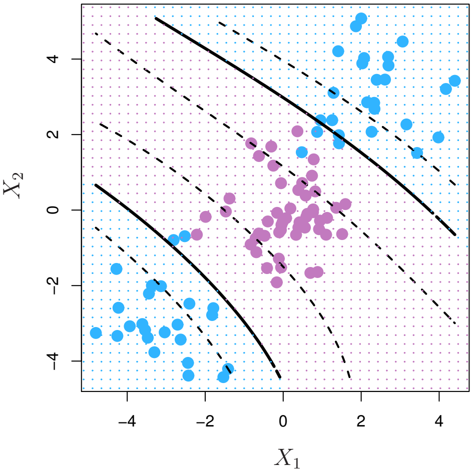
The Need for Nonlinear Classifiers



Nonlinear Methods

One way to add nonlinearity is through **basis expansions**:

$$X_1 \mapsto (X_1, X_1^2)$$
$$X_2 \mapsto (X_2, X_2^2)$$



Nonlinear Methods

In this lecture, we'll learn about two methods that are fundamentally non-linear: **k-nearest neighbors** and **decision trees**.

Nonlinear Methods

In this lecture, we'll learn about two methods that are fundamentally non-linear: k**-nearest neighbors** and **decision trees**.

I want you to think about what the regression functions and the decision boundaries look like for these methods.

1. Review
2. k-Nearest Neighbors
3. Decision Trees

k-Nearest Neighbors

k-Nearest Neighbors

Remember that the goal is to predict the response y^* given predictors $\mathbf{x}^*$.

k-Nearest Neighbors

Remember that the goal is to predict the response y^* given predictors $\mathbf{x}^*$.

Suppose we have a way to measure the **distance** between any two sets of predictors: $d(\mathbf{x}, \mathbf{x}')$.

k-Nearest Neighbors

Remember that the goal is to predict the response y^* given predictors $\mathbf{x}^*$.

Suppose we have a way to measure the **distance** between any two sets of predictors: $d(\mathbf{x}, \mathbf{x'})$.

Then, k-nearest neighbors averages the outputs of the k observations in the *training* data that are "closest" to $\mathbf{x}^*$.

k-Nearest Neighbors

Remember that the goal is to predict the response y^* given predictors $\mathbf{x}^*$.

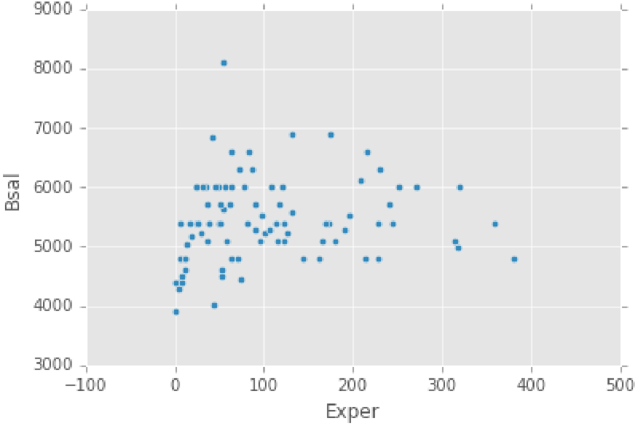
Suppose we have a way to measure the **distance** between any two sets of predictors: $d(\mathbf{x}, \mathbf{x'})$.

Then, k-nearest neighbors averages the outputs of the k observations in the *training* data that are "closest" to $\mathbf{x}^*$.

Intuition: These k neighbors are most similar to $\mathbf{x}^*$, so they should share a similar response.

k-Nearest Neighbors Regression

This data is taken from the Harris bank data set. The data can be found on JupyterHub at `/data/harris.csv`.



k-Nearest Neighbors Regression in Scikit-Learn

```
from sklearn.neighbors import KNeighborsRegressor
model = KNeighborsRegressor(n_neighbors=5, weights="uniform")
model.fit(X, y)
```

k-Nearest Neighbors Regression in Scikit-Learn

```
from sklearn.neighbors import KNeighborsRegressor
model = KNeighborsRegressor(n_neighbors=5, weights="uniform")
model.fit(X, y)
```

Let's investigate what the regression function estimated by k-nearest neighbors looks like.

In-Class Exercise

- *Add the curve that is fitted by k-nearest neighbors to the plot below. What happens as you vary k?*
- *Use cross-validation to select the optimal k.*

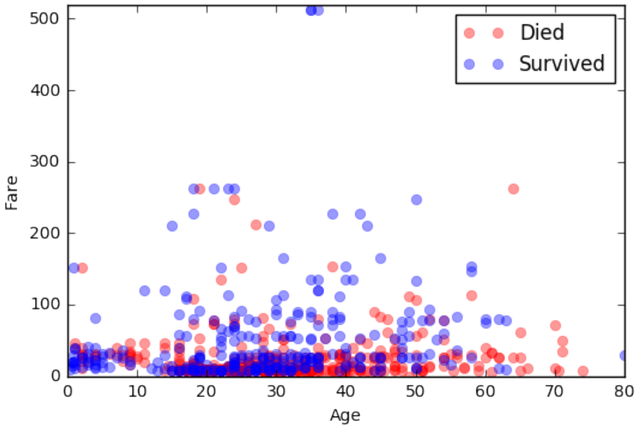
k-Nearest Neighbors Classification

The k-nearest neighbors classifier predicts the output of a test input by *majority vote* based on the outputs of the k “closest” points in the training data.

k-Nearest Neighbors Classification in Pictures

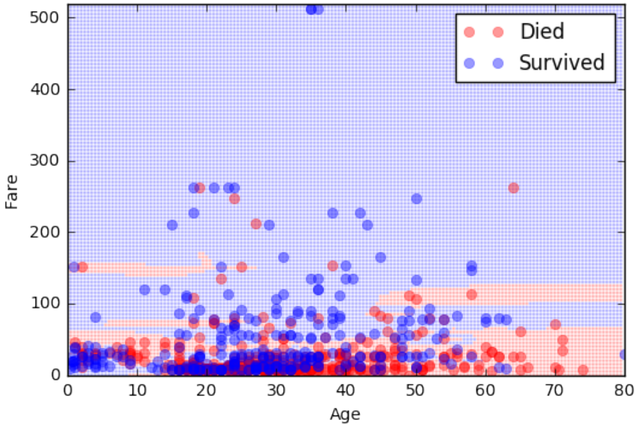
This data is taken from the Titanic data set. The data can be found on JupyterHub at `/data/titanic.csv`.

Let's just consider two variables, `Age` and `Fare`.



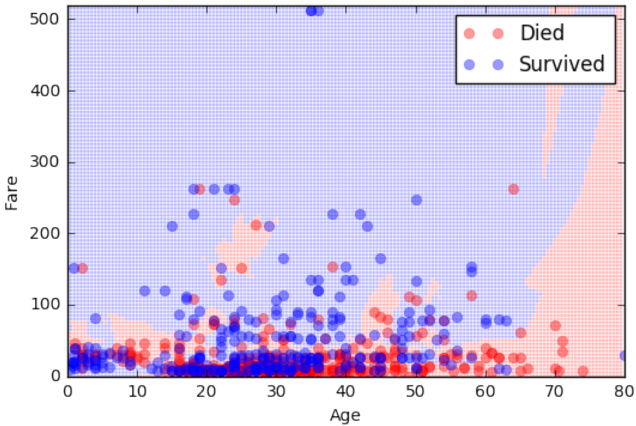
k-Nearest Neighbors Classification in Pictures

Here are the predictions returned by 5-nearest neighbors:



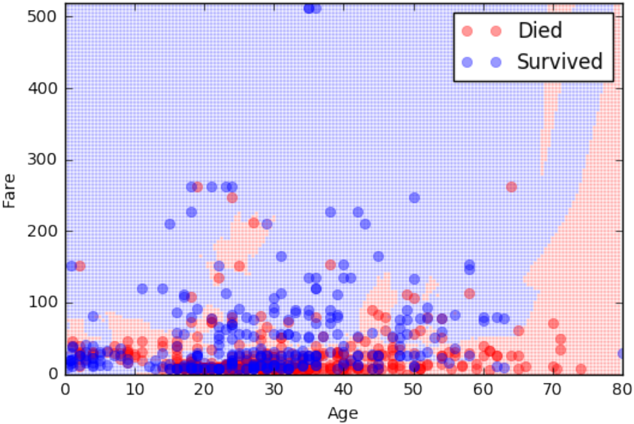
k-Nearest Neighbors Classification in Pictures

Notice that `Fare` and `Age` are on different scales. So I transformed `Fare` to $10\log(1+\texttt{Fare})$ before calculating distances. And the predictions I get are very different!



k-Nearest Neighbors Classification in Pictures

Notice that `Fare` and `Age` are on different scales. So I transformed `Fare` to $10 \log(1 + \texttt{Fare})$ before calculating distances. And the predictions I get are very different!



K-nearest neighbors is sensitive to the choice of distance metric d.

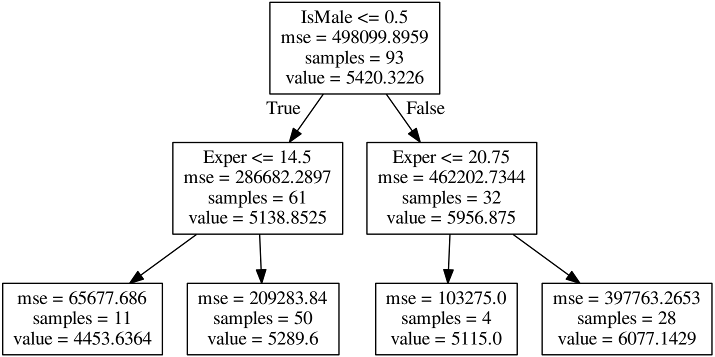
k-Nearest Neighbors Classification in Scikit-Learn

```
from sklearn.neighbors import KNeighborsClassifier
model = KNeighborsClassifier(n_neighbors=5, weights="uniform")
model.fit(X, y)
```

1. Review
2. k-Nearest Neighbors
3. Decision Trees

Decision Trees

Decision tree for predicting starting salary, trained on the Harris bank data.



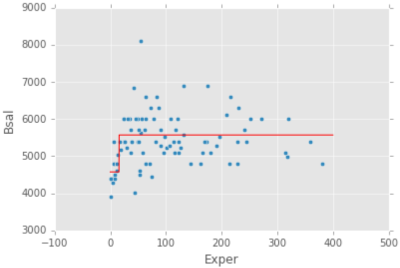
Training a Decision Tree

Training a Decision Tree

Search through all possible variables and all possible splits. Find the split that minimizes the mean-squared error.

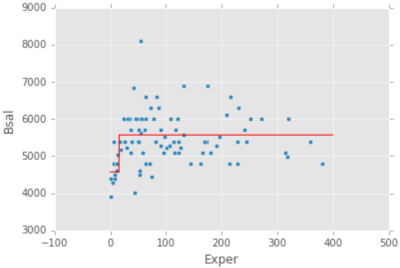
Training a Decision Tree

Search through all possible variables and all possible splits. Find the split that minimizes the mean-squared error.



Training a Decision Tree

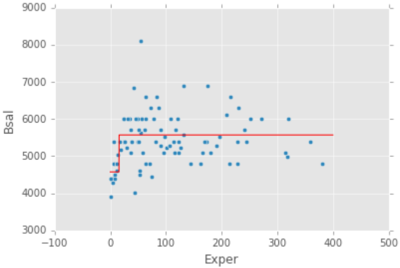
Search through all possible variables and all possible splits. Find the split that minimizes the mean-squared error.



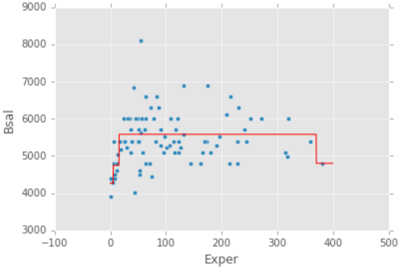
Repeat for each child node.

Training a Decision Tree

Search through all possible variables and all possible splits. Find the split that minimizes the mean-squared error.



Repeat for each child node.



Decision Tree Regression in Scikit-Learn

```
from sklearn.tree import DecisionTreeRegressor
model = DecisionTreeRegressor(max_depth=2)
model.fit(X, y)
```

Decision Tree Regression in Scikit-Learn

```
from sklearn.tree import DecisionTreeRegressor
model = DecisionTreeRegressor(max_depth=2)
model.fit(X, y)
```

Printing the Decision Tree

```
from sklearn.tree import export_graphviz
with open("tree.dot", "w") as f:
  export_graphviz(model, out_file=f, feature_names=...)
```

Decision Tree Regression in Scikit-Learn

```
from sklearn.tree import DecisionTreeRegressor
model = DecisionTreeRegressor(max_depth=2)
model.fit(X, y)
```

Printing the Decision Tree

```
from sklearn.tree import export_graphviz
with open("tree.dot", "w") as f:
  export_graphviz(model, out_file=f, feature_names=...)
```

This writes the tree to a `.dot` file. You can convert it to a more usable format (e.g., `.pdf`) using the command-line `dot` tool:

`dot -Tpdf tree.dot -o tree.pdf`.

Decision Tree Regression in Scikit-Learn

```
from sklearn.tree import DecisionTreeRegressor
model = DecisionTreeRegressor(max_depth=2)
model.fit(X, y)
```

Printing the Decision Tree

```
from sklearn.tree import export_graphviz
with open("tree.dot", "w") as f:
  export_graphviz(model, out_file=f, feature_names=...)
```

This writes the tree to a `.dot` file. You can convert it to a more usable format (e.g., `.pdf`) using the command-line `dot` tool:

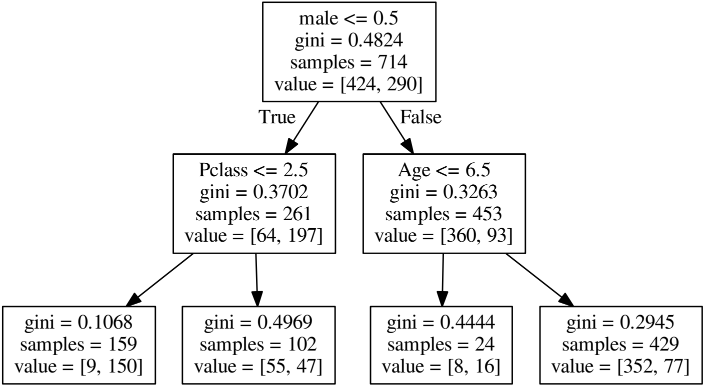
`dot -Tpdf tree.dot -o tree.pdf`.

In-Class Exercise

- *Fit a decision tree to the Harris bank data to predict beginning salary (`Bsal`) from experience (`Exper`). Plot the estimated regression function. What happens as you vary the tree depth?*
- *Use cross-validation to select the optimal tree depth.*

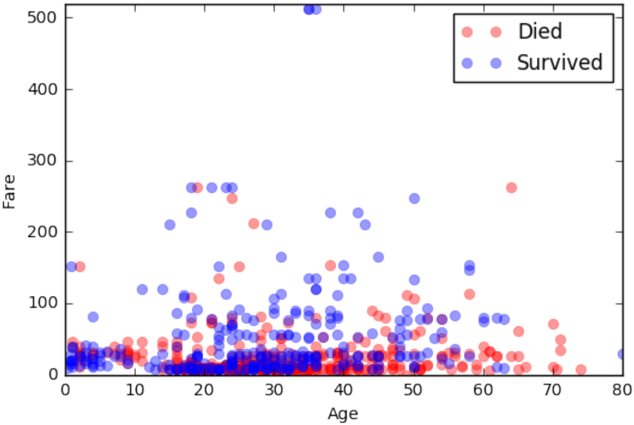
Decision Tree Classification

Decision tree for predicting survival on the Titanic.

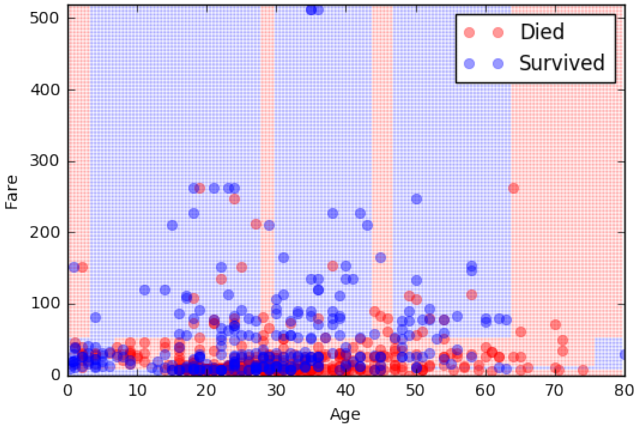


Decision Tree Classification in Pictures

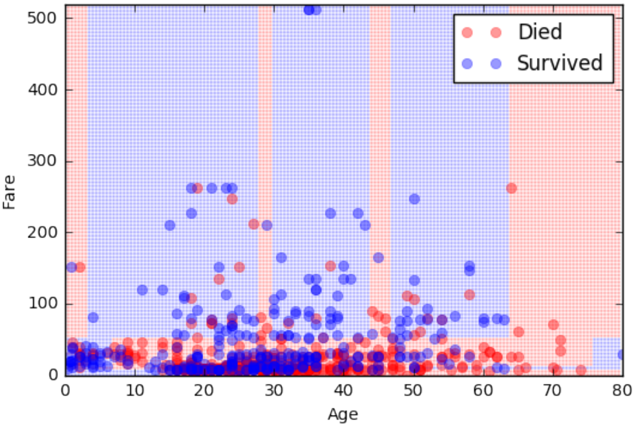
Let's again just consider two variables, `Age` and `Fare`. What will the decision boundary of a decision tree classifier look like?



Decision Tree Classification in Pictures



Decision Tree Classification in Pictures



When you fit a decision tree, the predicted value tends to be constant over rectangular regions. This is because decision trees only split on one variable at a time.

Decision Tree Classification in Scikit-Learn

```
from sklearn.tree import DecisionTreeClassifier
model = DecisionTreeClassifier(max_depth=5)
model.fit(X, y)
```