10-601 Introduction to Machine Learning

Machine Learning Department
School of Computer Science
Carnegie Mellon University

Model Selection

Matt Gormley
Lecture 4
January 29, 2018

Q&A

Q: How do we deal with ties in k-Nearest Neighbors (e.g. even k or equidistant points)?

A: I would ask you all for a good solution!

Q: How do we define a distance function when the features are categorical (e.g. weather takes values {sunny, rainy, overcast})?

A: Step 1: Convert from categorical attributes to numeric features (e.g. binary)

Step 2: Select an appropriate distance function (e.g. Hamming distance)

Reminders

- **Homework 2: Decision Trees**
  - Out: Wed, Jan 24
  - Due: Mon, Feb 5 at 11:59pm
- **10601 Notation Crib Sheet**

K-NEAREST NEIGHBORS

k-Nearest Neighbors

Chalkboard:

- KNN for binary classification
- Distance functions
- Efficiency of KNN
- Inductive bias of KNN
- KNN Properties

KNN ON FISHER IRIS DATA

Fisher Iris Dataset

Fisher (1936) used 150 measurements of flowers from 3 different species: Iris setosa (0), Iris virginica (1), Iris versicolor (2) collected by Anderson (1936)

Species	Sepal Length	Sepal Width	Petal Length	Petal Width
0	4.3	3.0	1.1	0.1
0	4.9	3.6	1.4	0.1
0	5.3	3.7	1.5	0.2
1	4.9	2.4	3.3	1.0
1	5.7	2.8	4.1	1.3
1	6.3	3.3	4.7	1.6
1	6.7	3.0	5.0	1.7

Full dataset: https://en.wikipedia.org/wiki/Iris_flower_data_set

Fisher Iris Dataset

Fisher (1936) used 150 measurements of flowers from 3 different species: Iris setosa (0), Iris virginica (1), Iris versicolor (2) collected by Anderson (1936)

Species	Sepal Length	Sepal Width
0	4.3	3.0
0	4.9	3.6
0	5.3	3.7
1	4.9	2.4
1	5.7	2.8
1	6.3	3.3
1	6.7	3.0

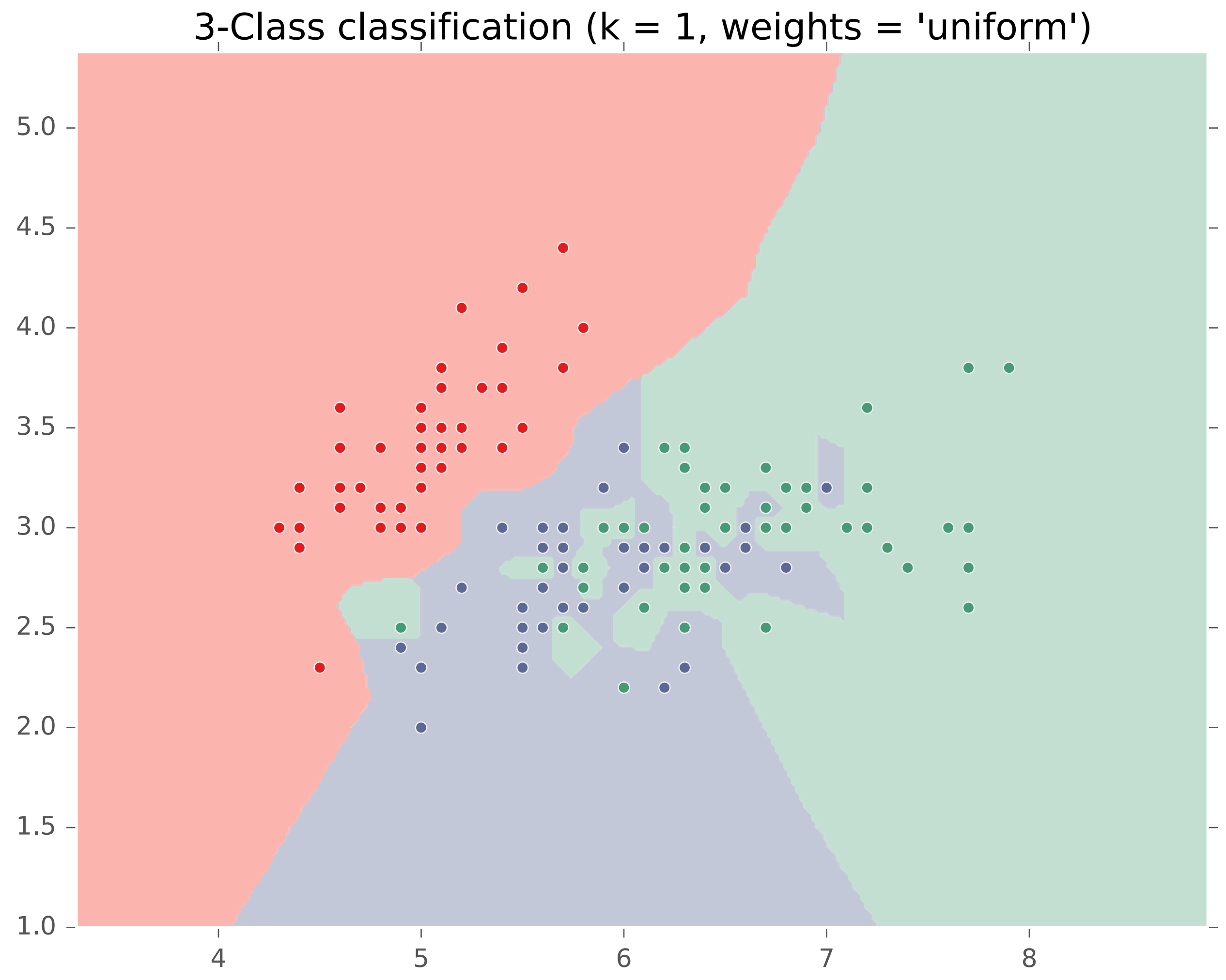
Deleted two of the four features, so that input space is 2D

Full dataset: https://en.wikipedia.org/wiki/Iris_flower_data_set

KNN on Fisher Iris Data

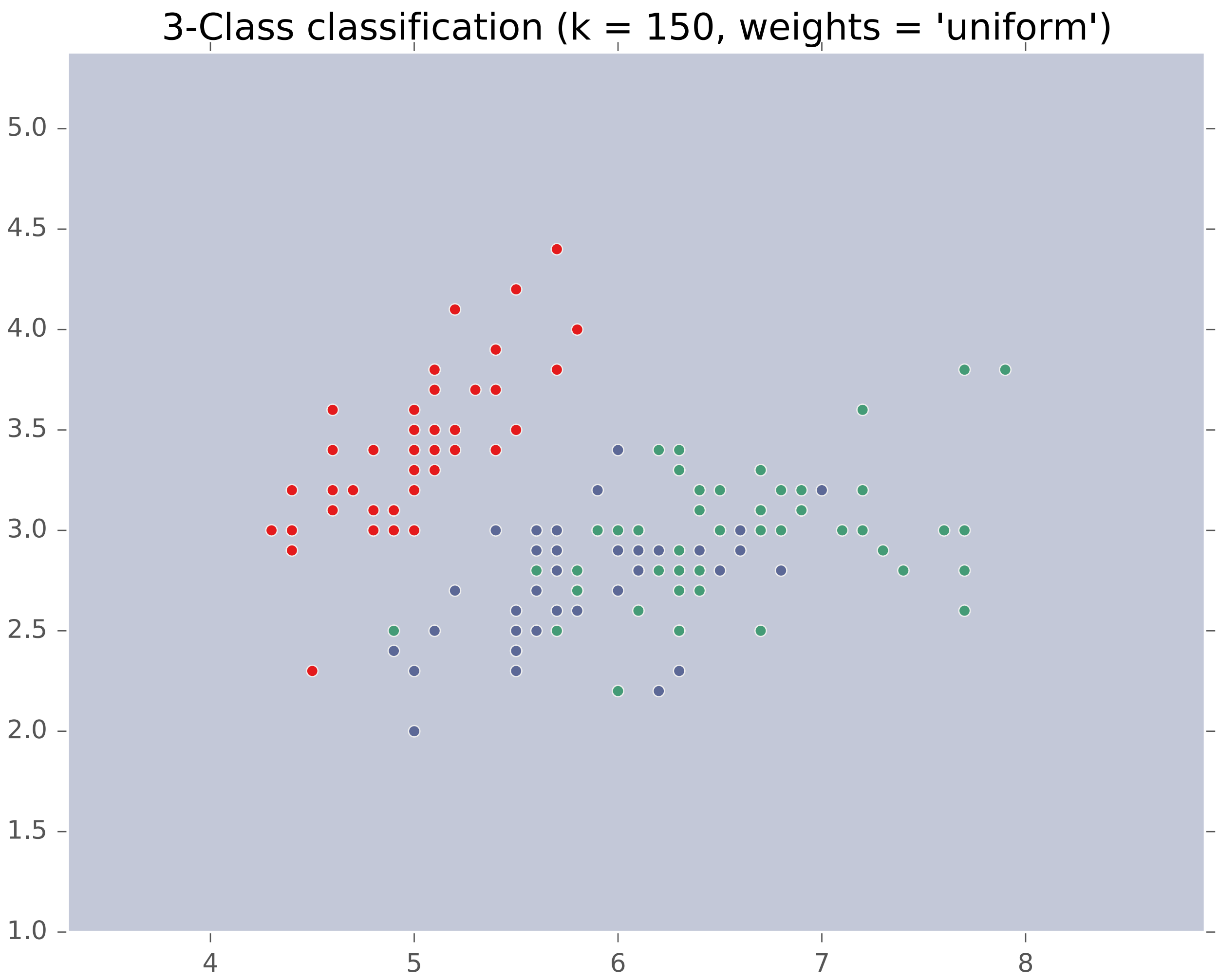
KNN on Fisher Iris Data

Special Case: Nearest Neighbor



KNN on Fisher Iris Data

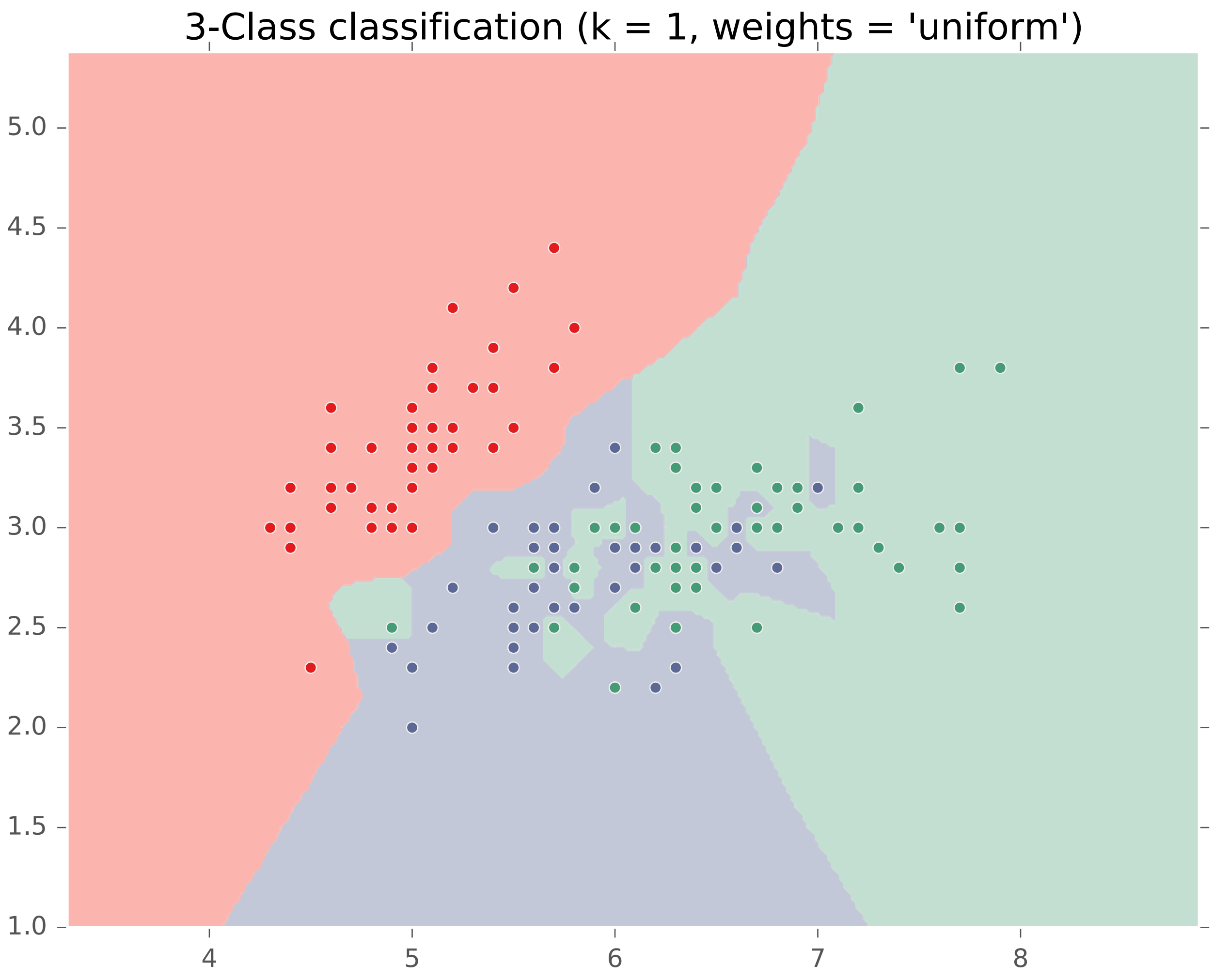
Special Case: Majority Vote



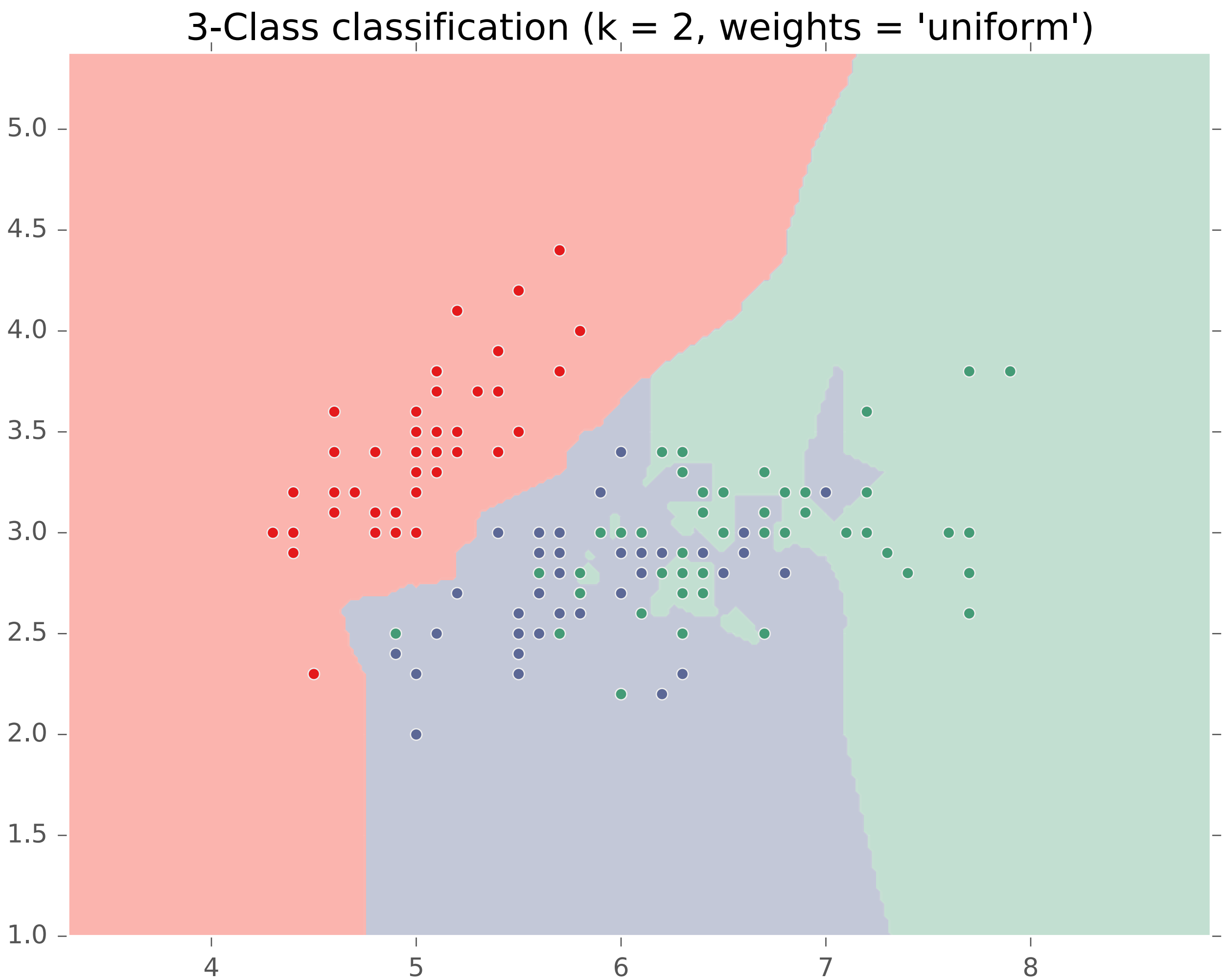
KNN on Fisher Iris Data

KNN on Fisher Iris Data

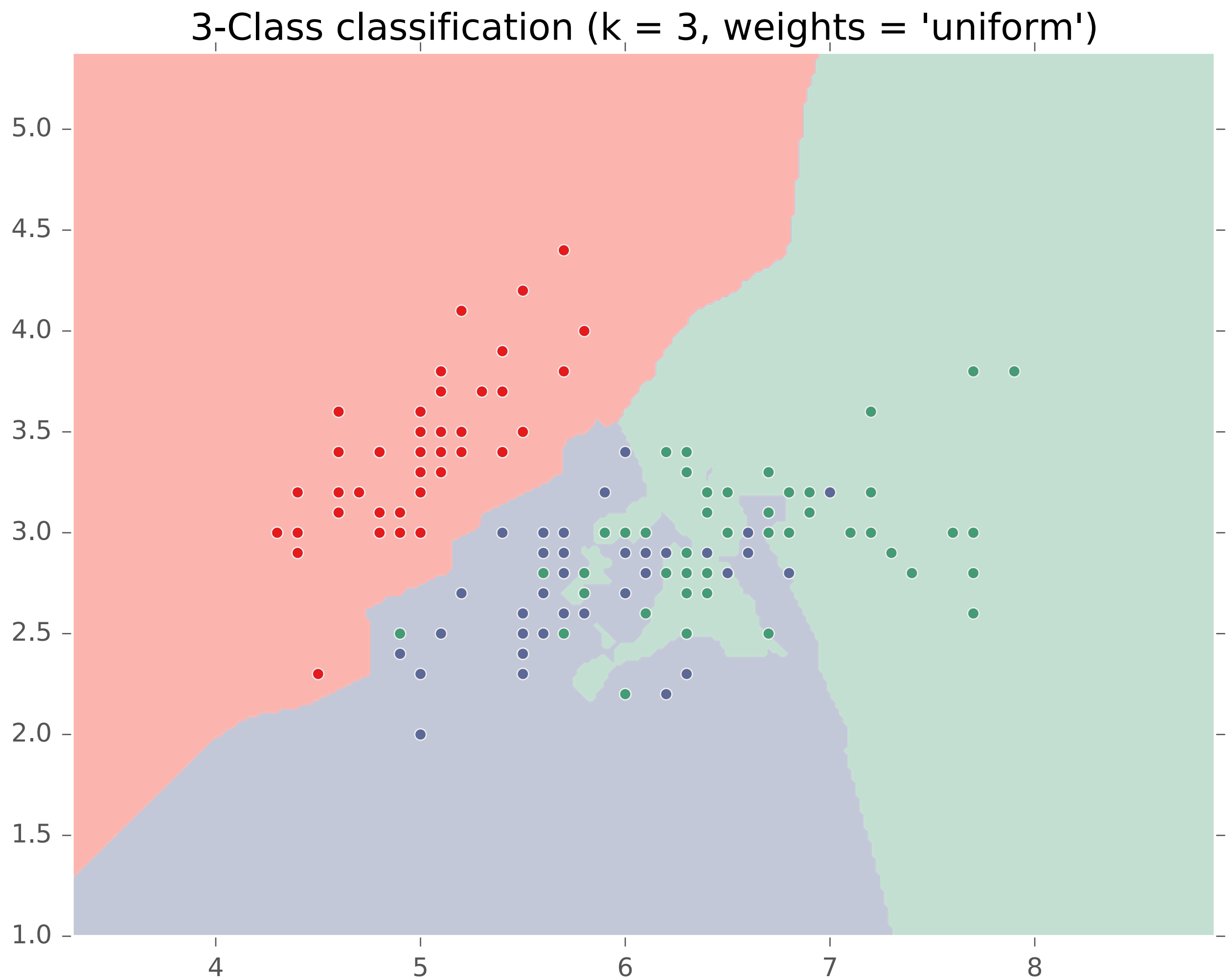
Special Case: Nearest Neighbor



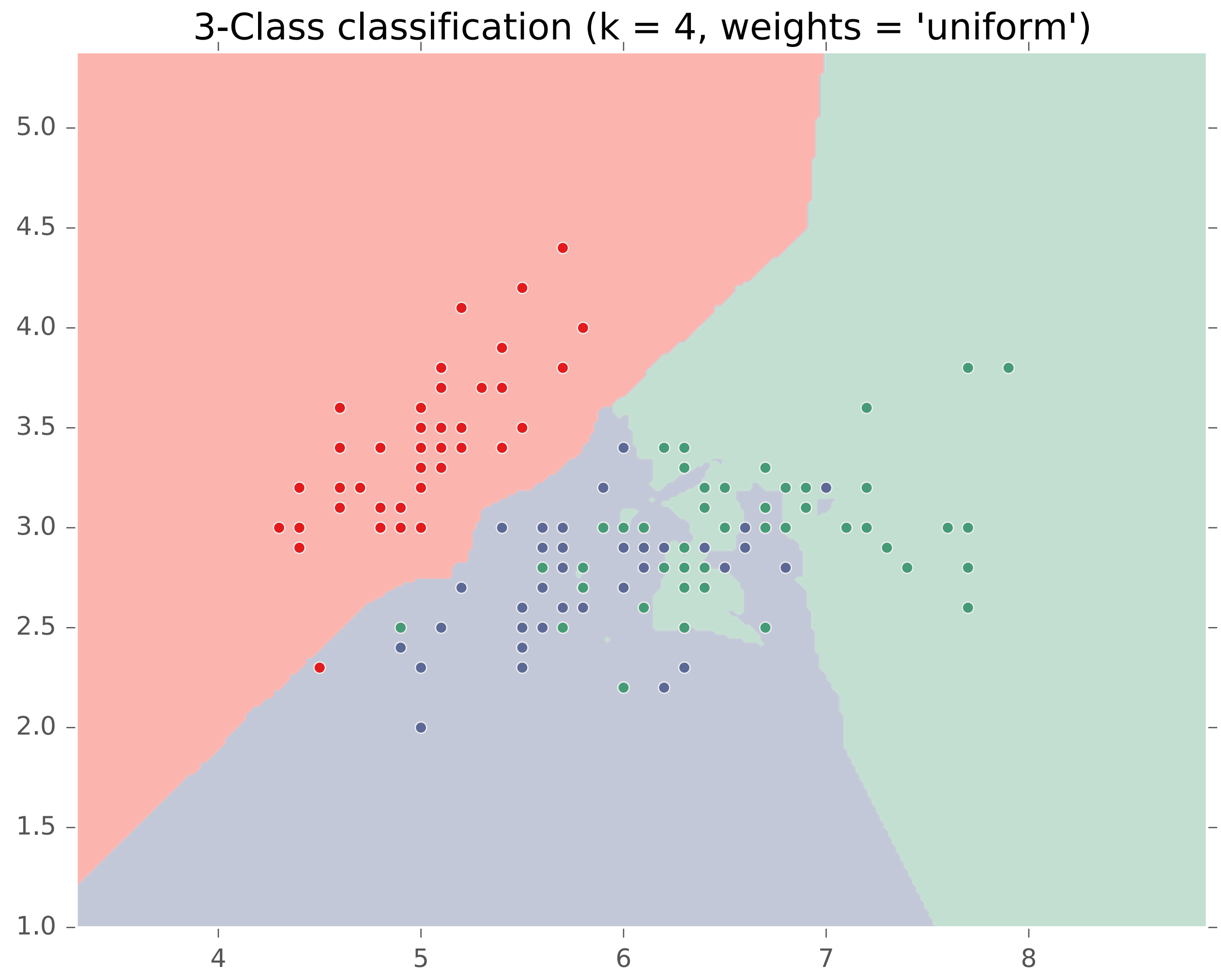
KNN on Fisher Iris Data



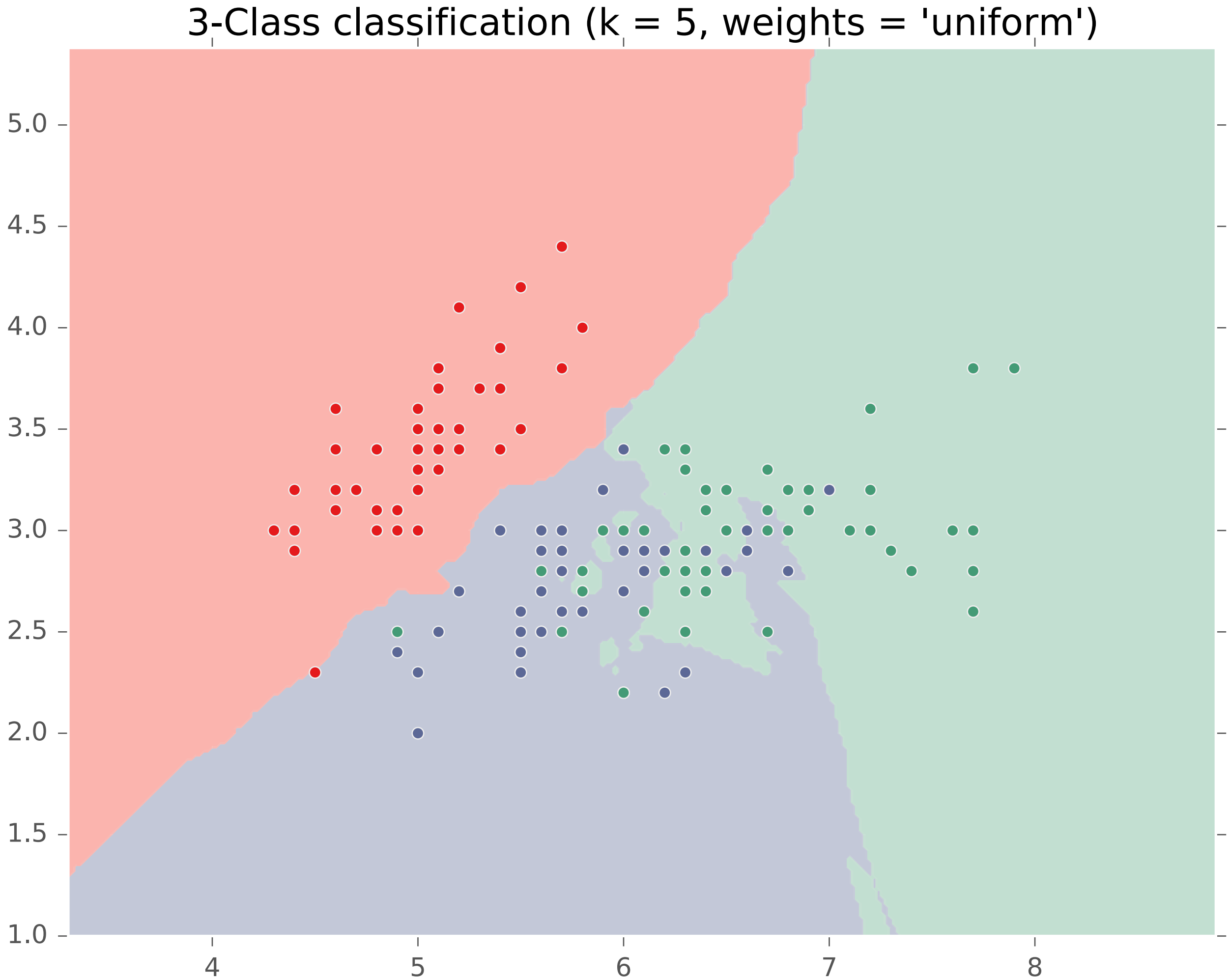
KNN on Fisher Iris Data



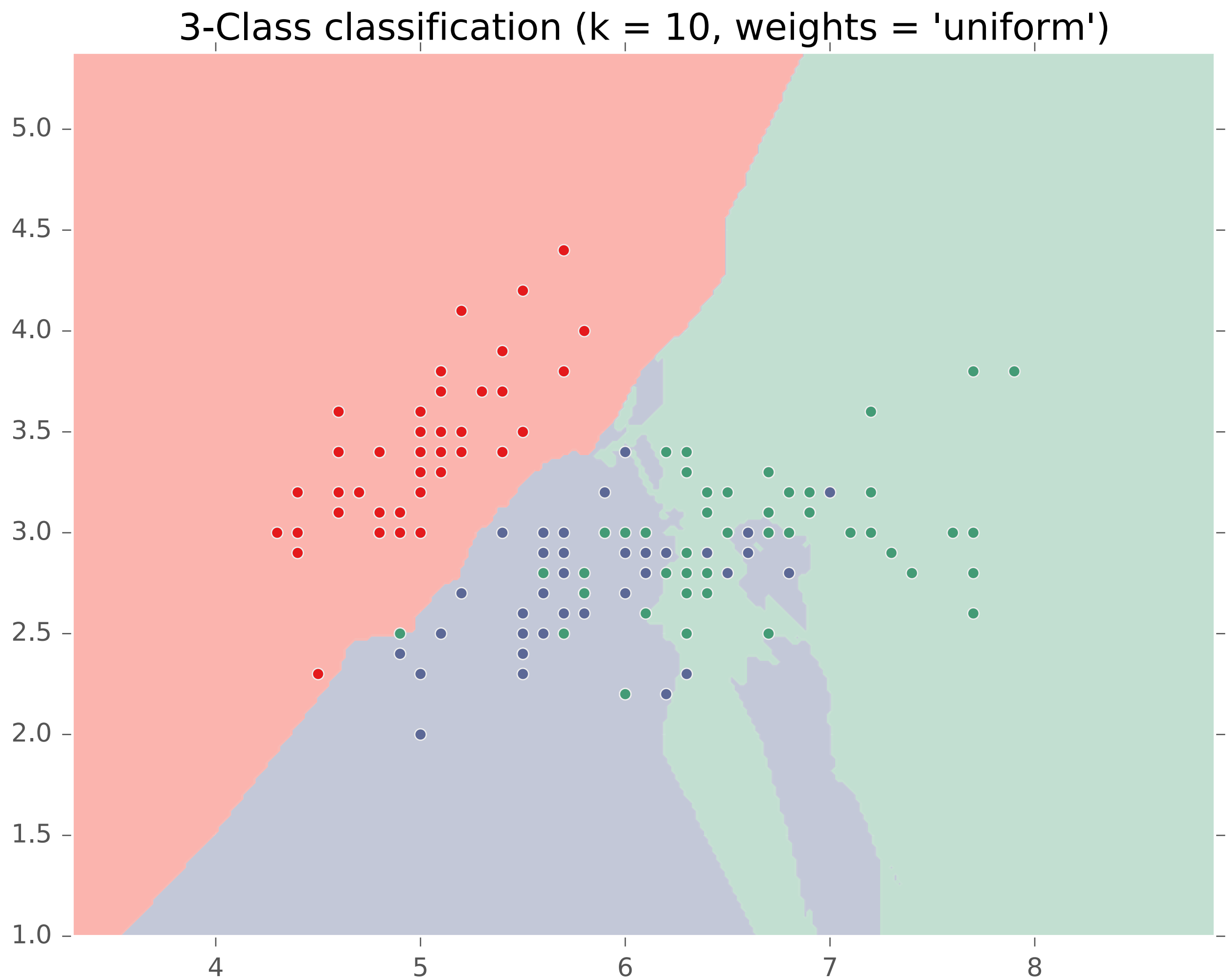
KNN on Fisher Iris Data



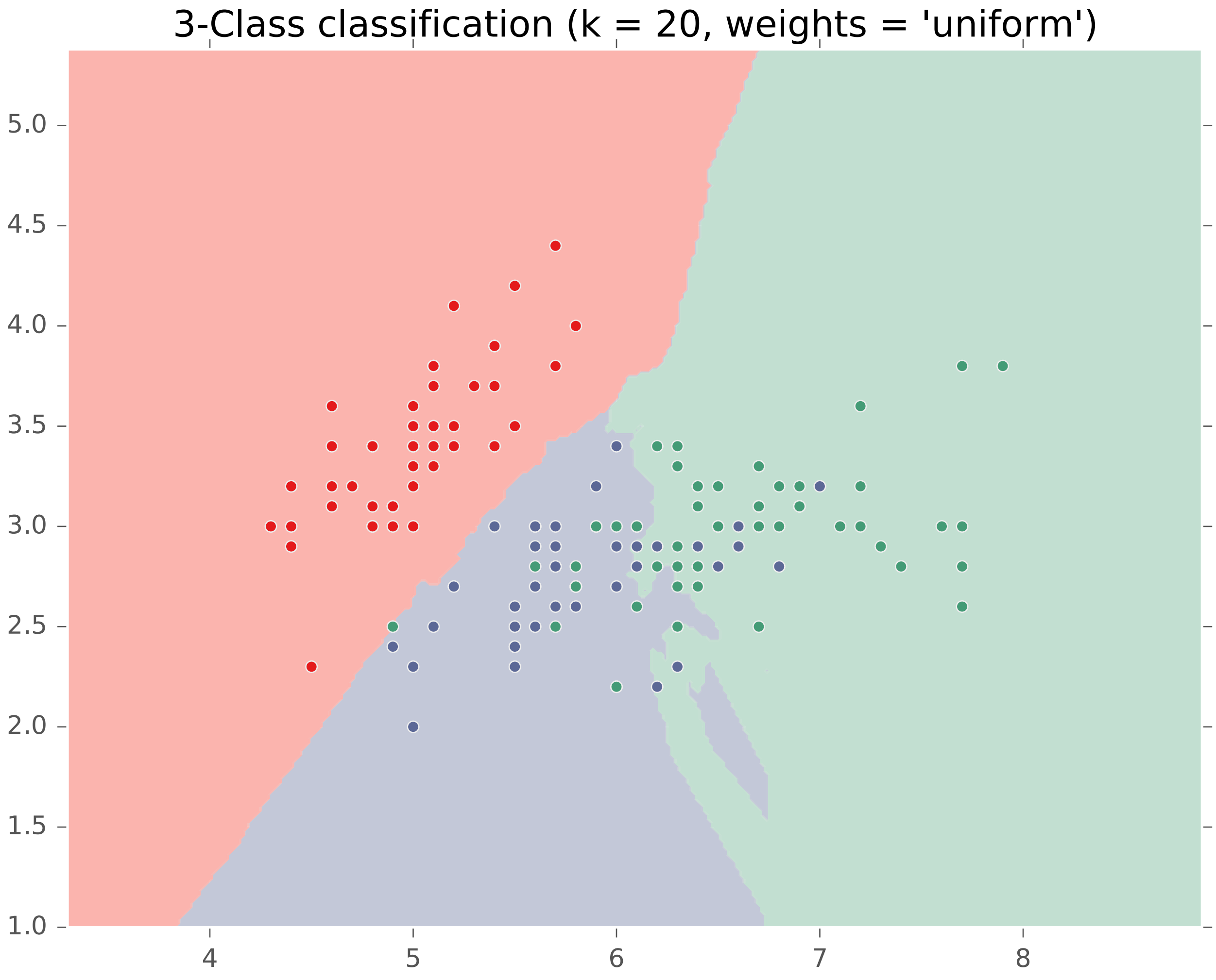
KNN on Fisher Iris Data



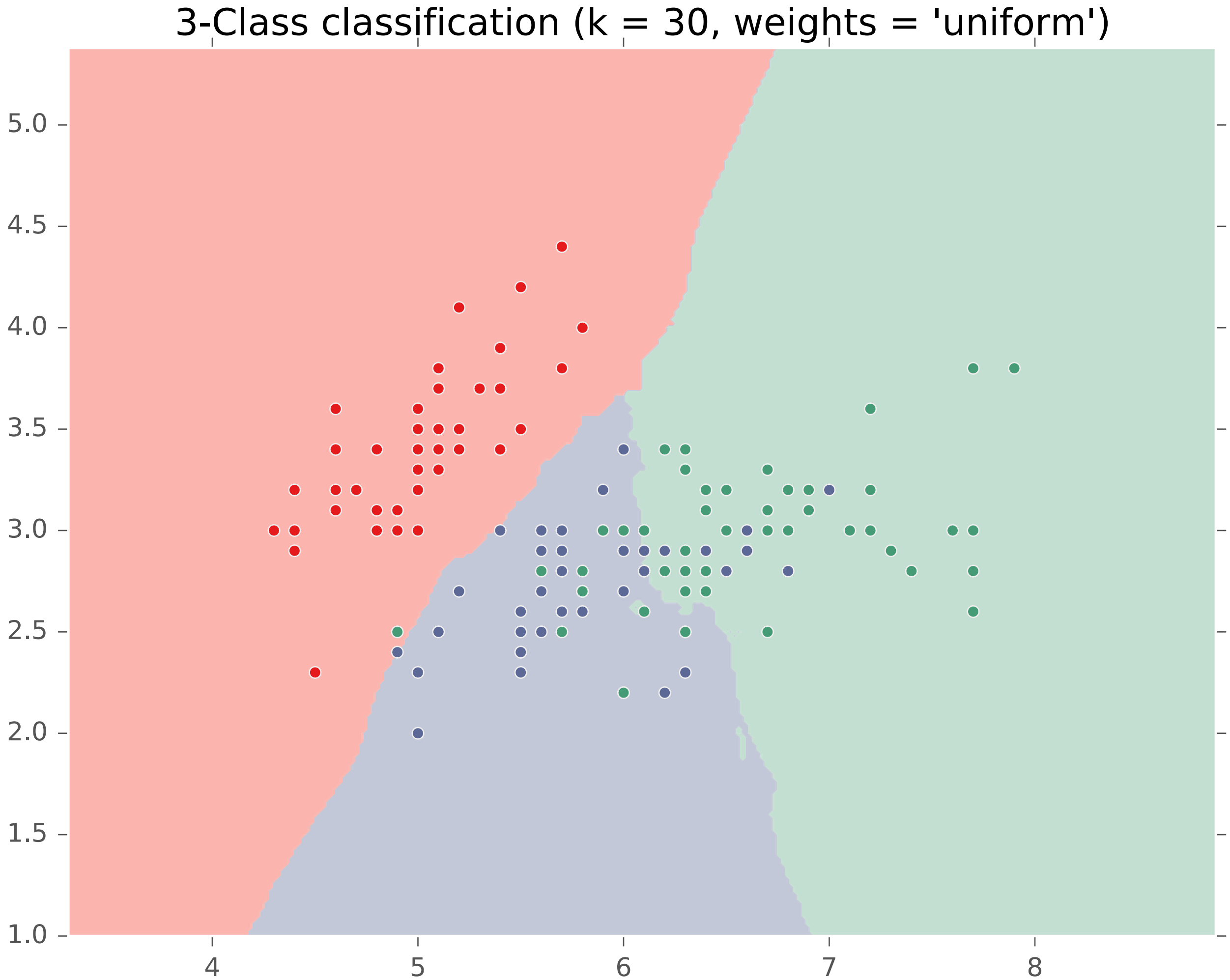
KNN on Fisher Iris Data



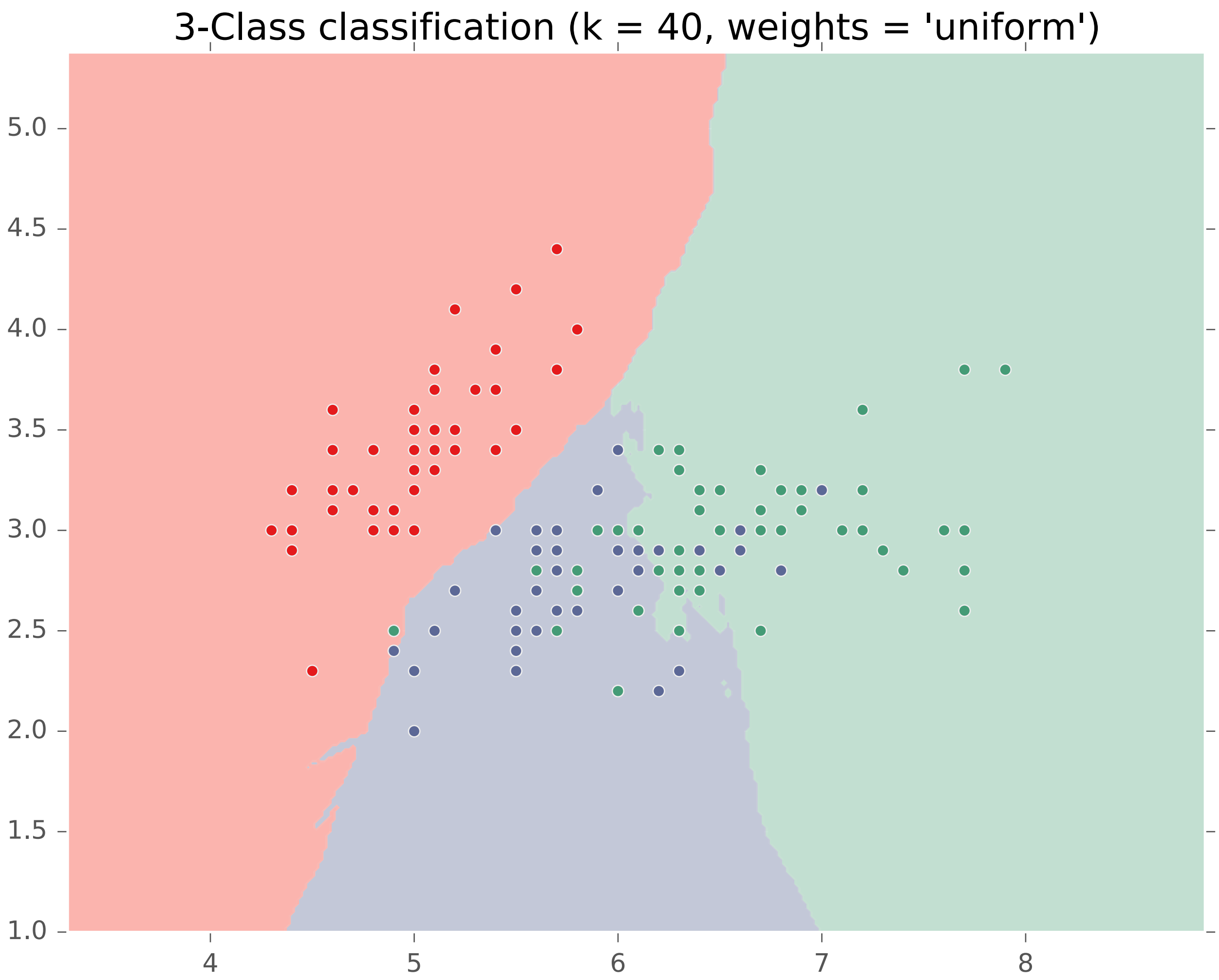
KNN on Fisher Iris Data



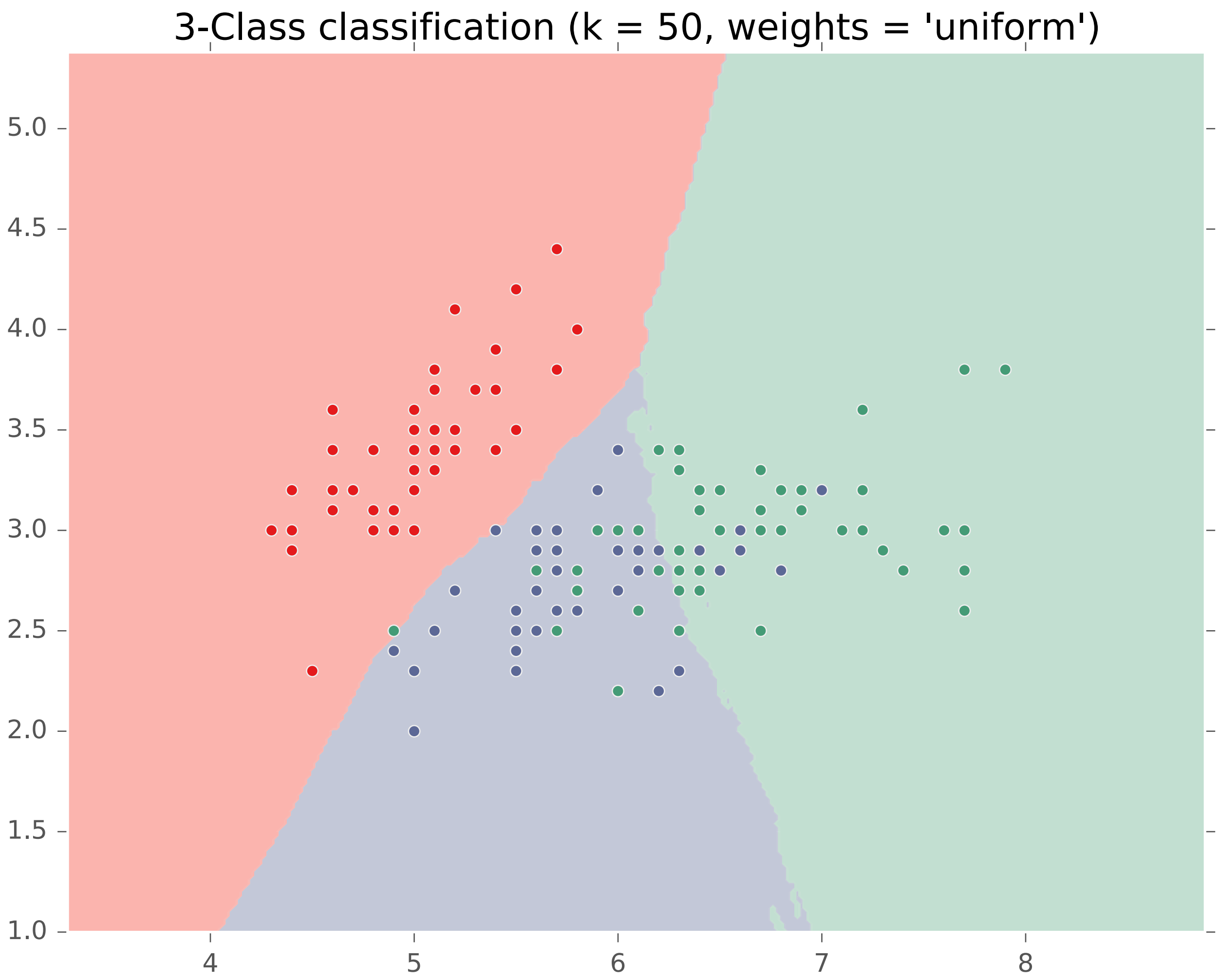
KNN on Fisher Iris Data



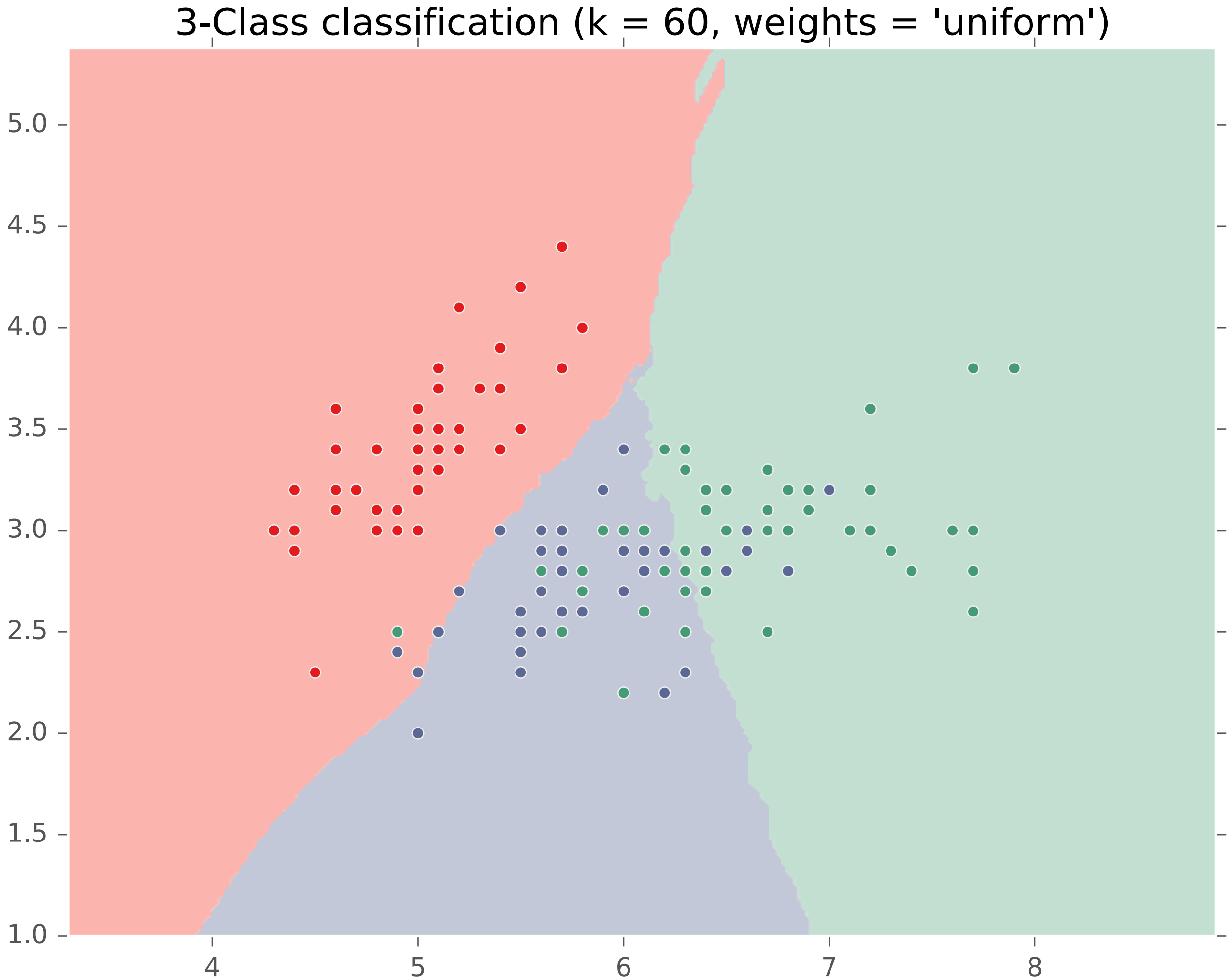
KNN on Fisher Iris Data



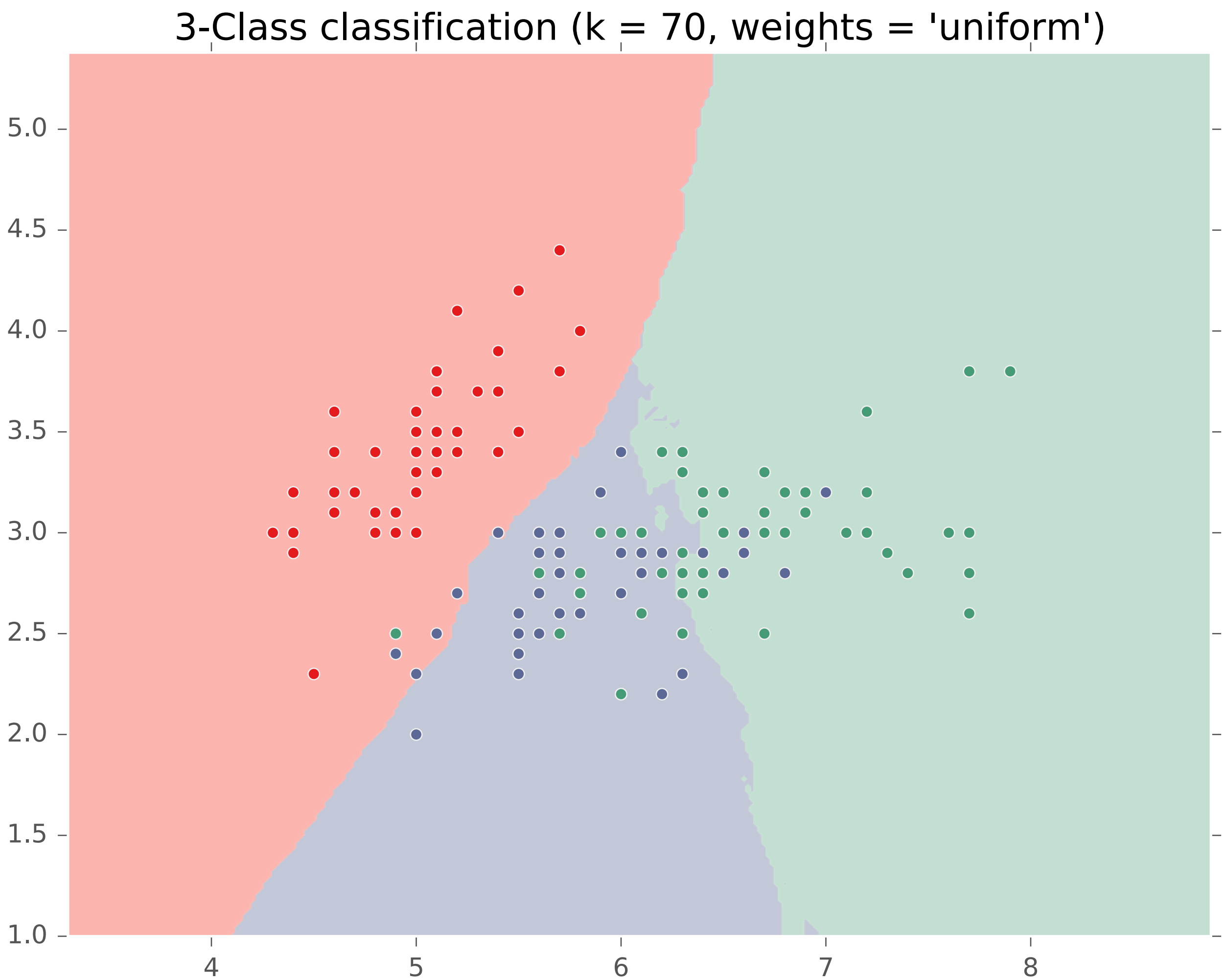
KNN on Fisher Iris Data



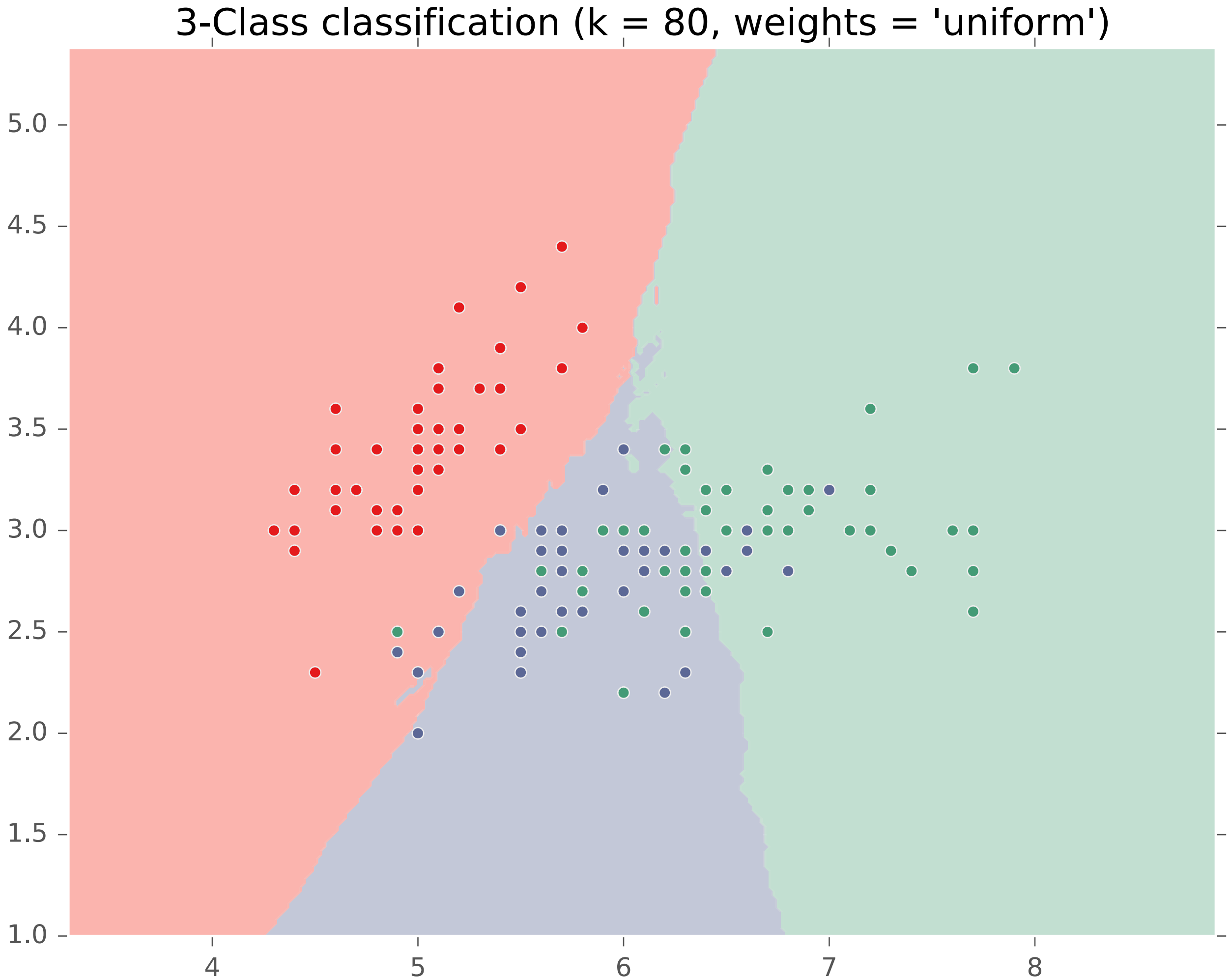
KNN on Fisher Iris Data



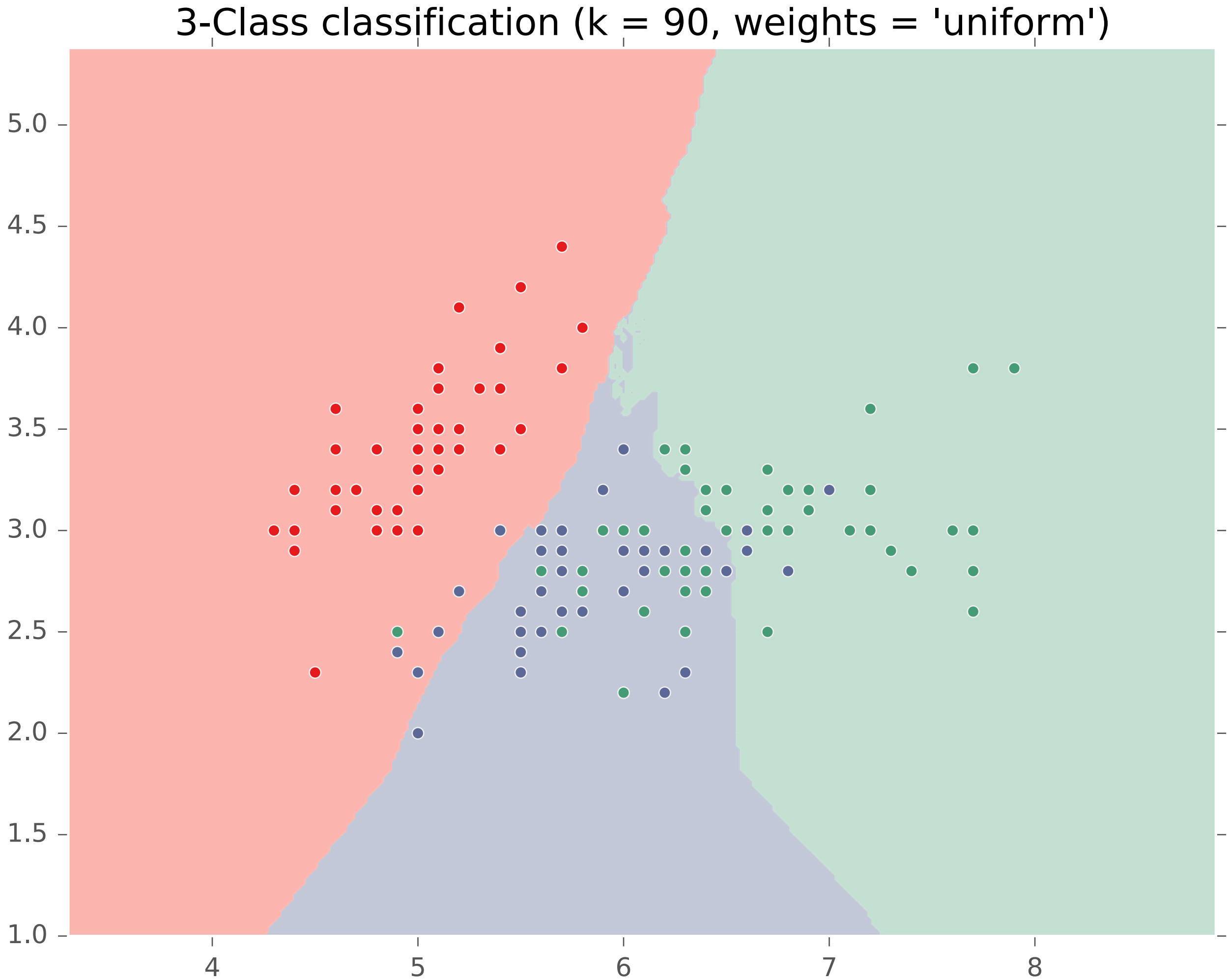
KNN on Fisher Iris Data



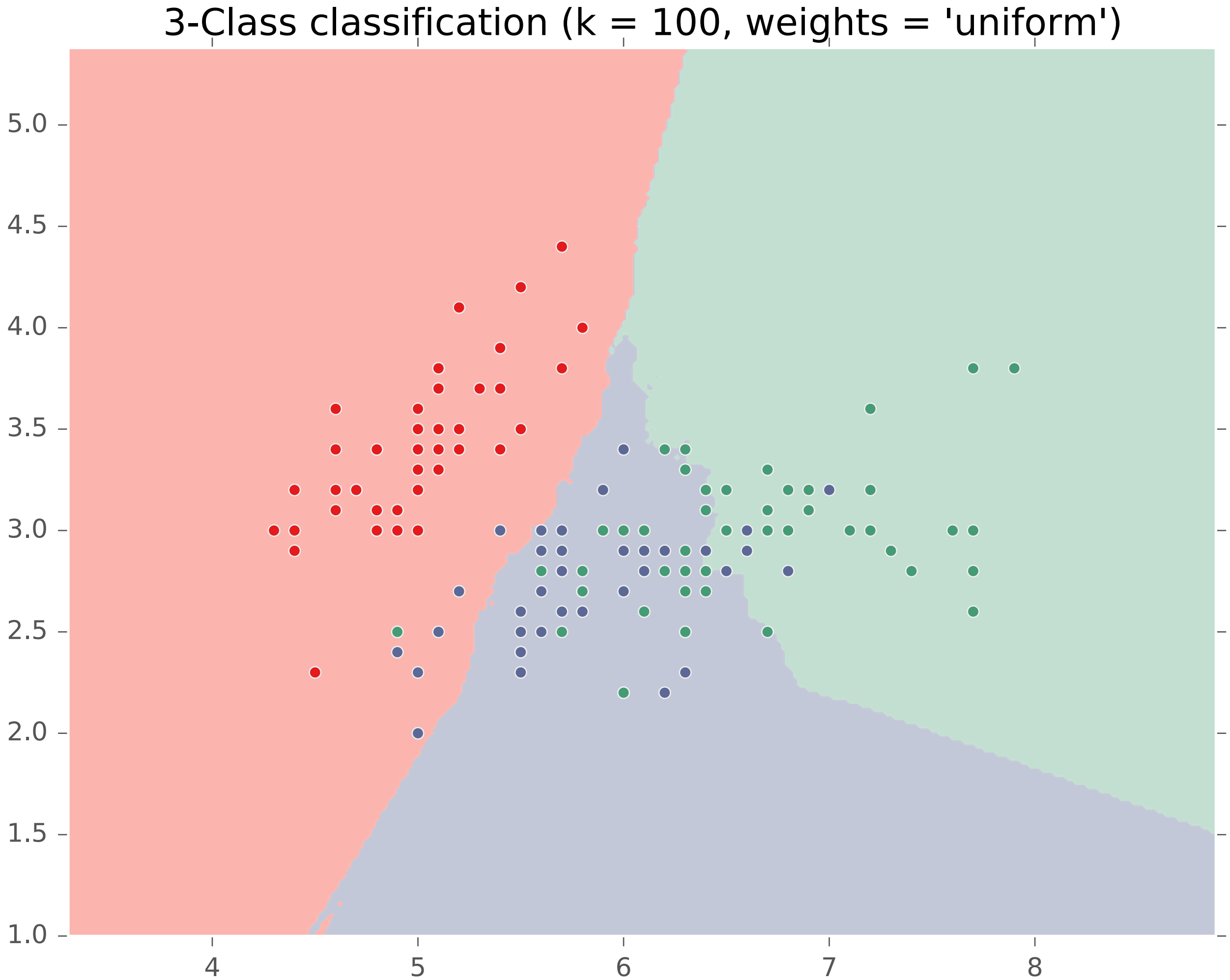
KNN on Fisher Iris Data



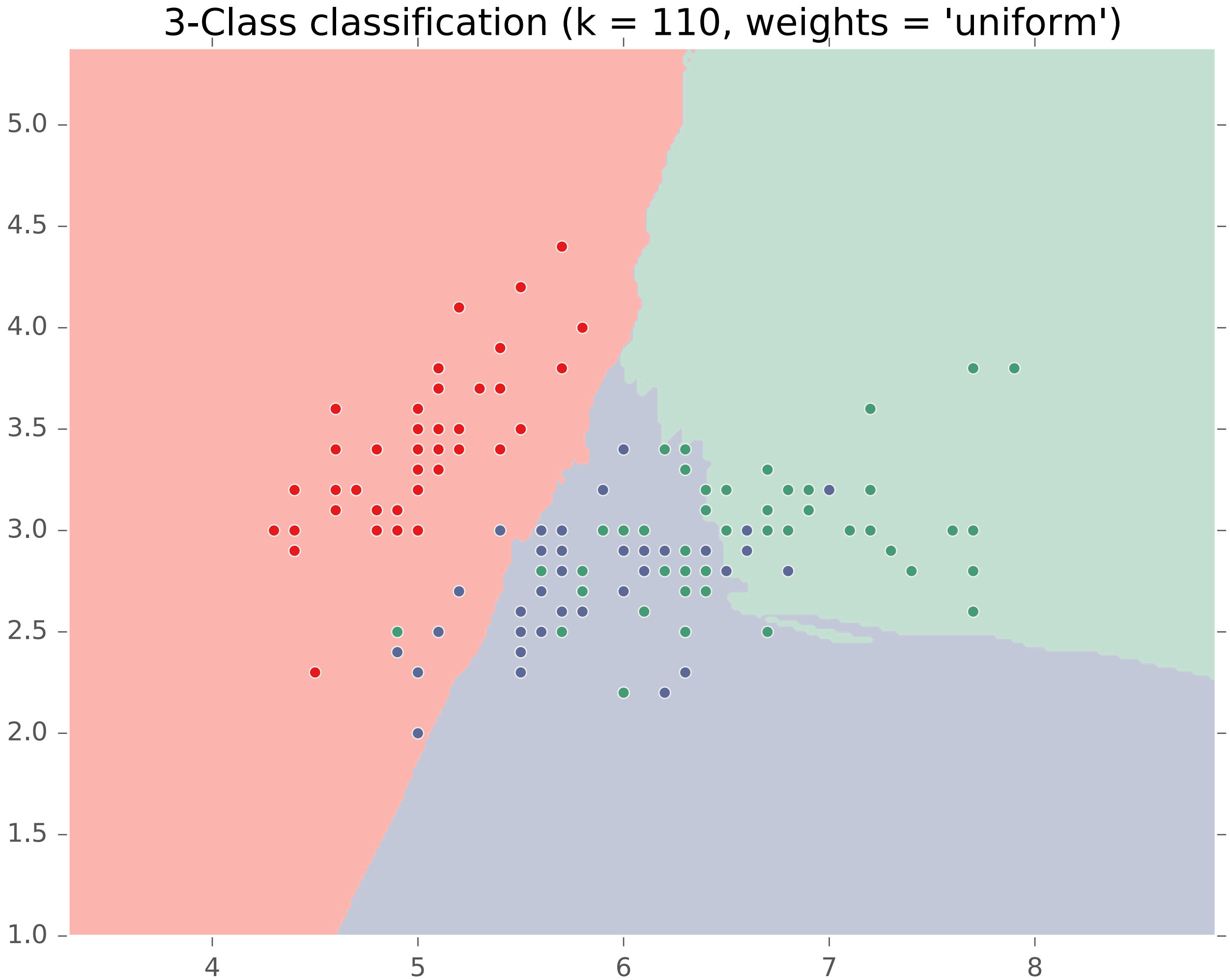
KNN on Fisher Iris Data



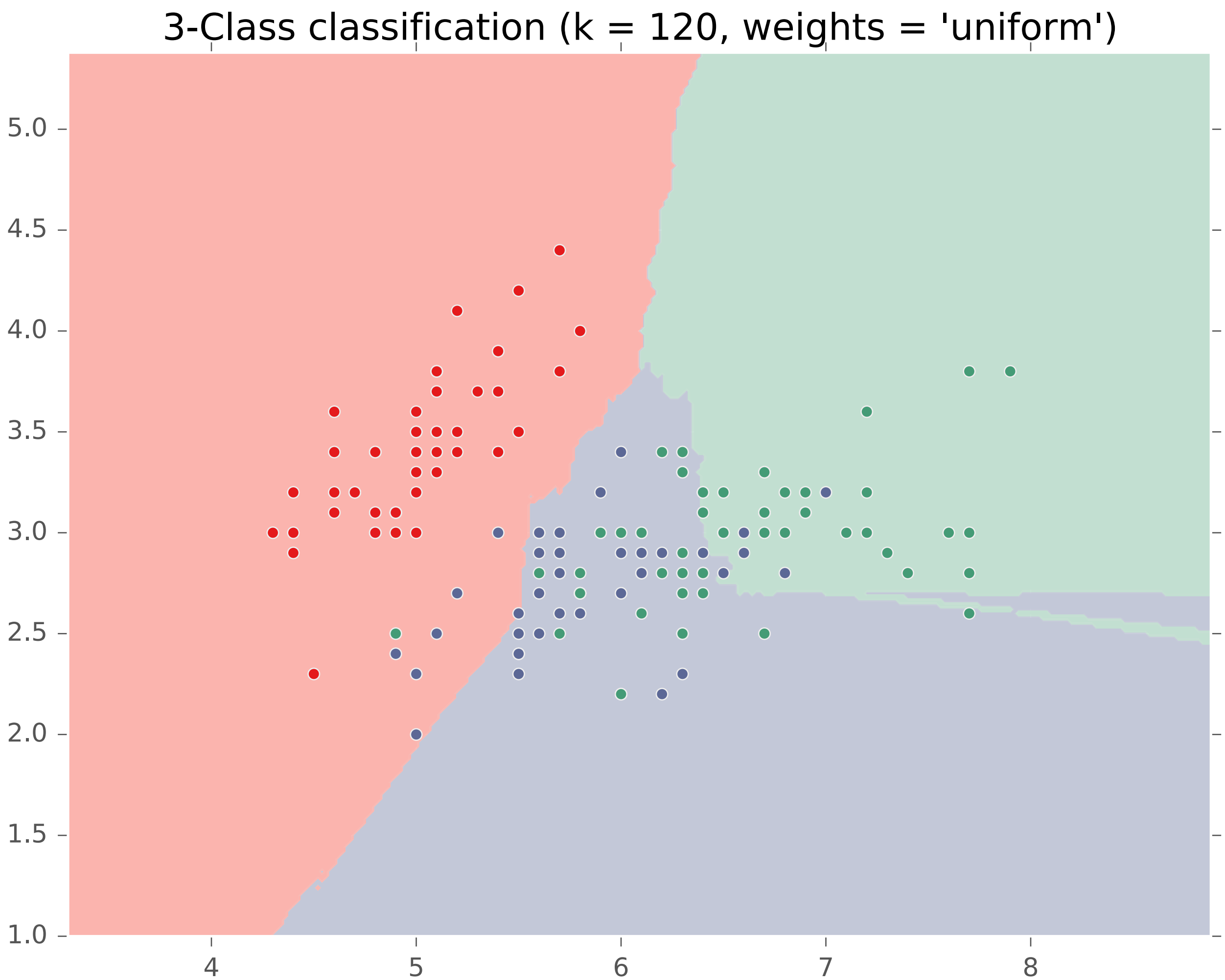
KNN on Fisher Iris Data



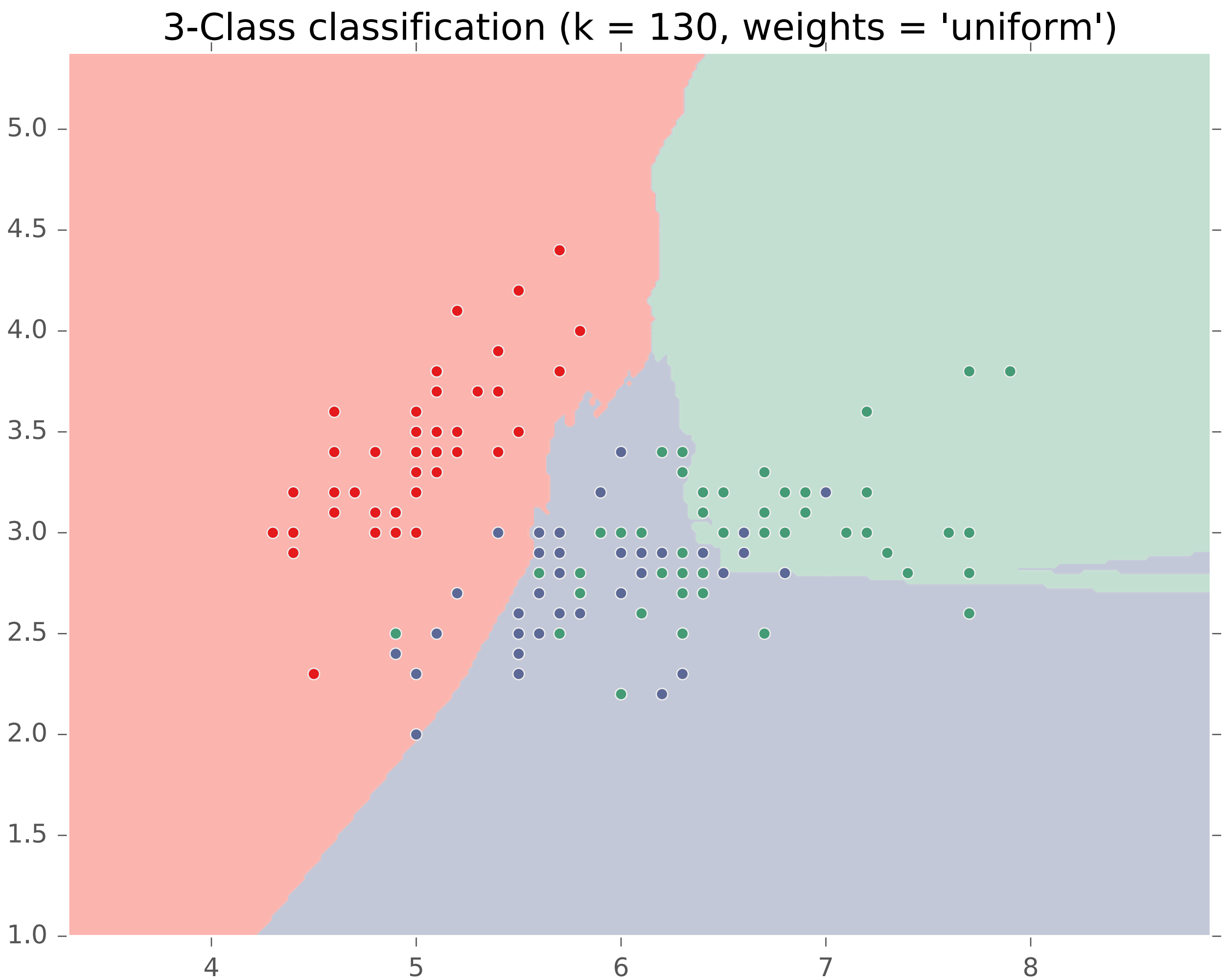
KNN on Fisher Iris Data



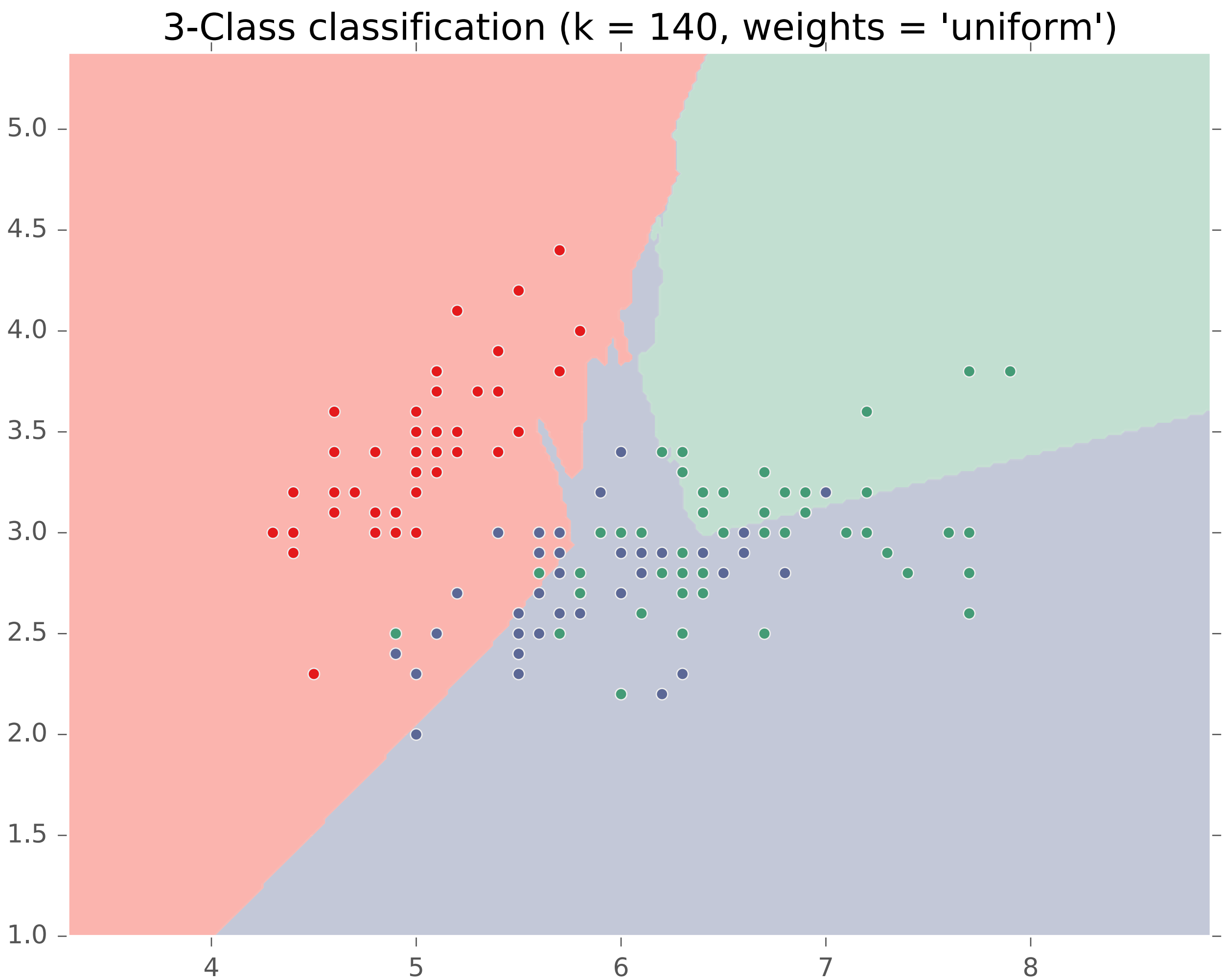
KNN on Fisher Iris Data



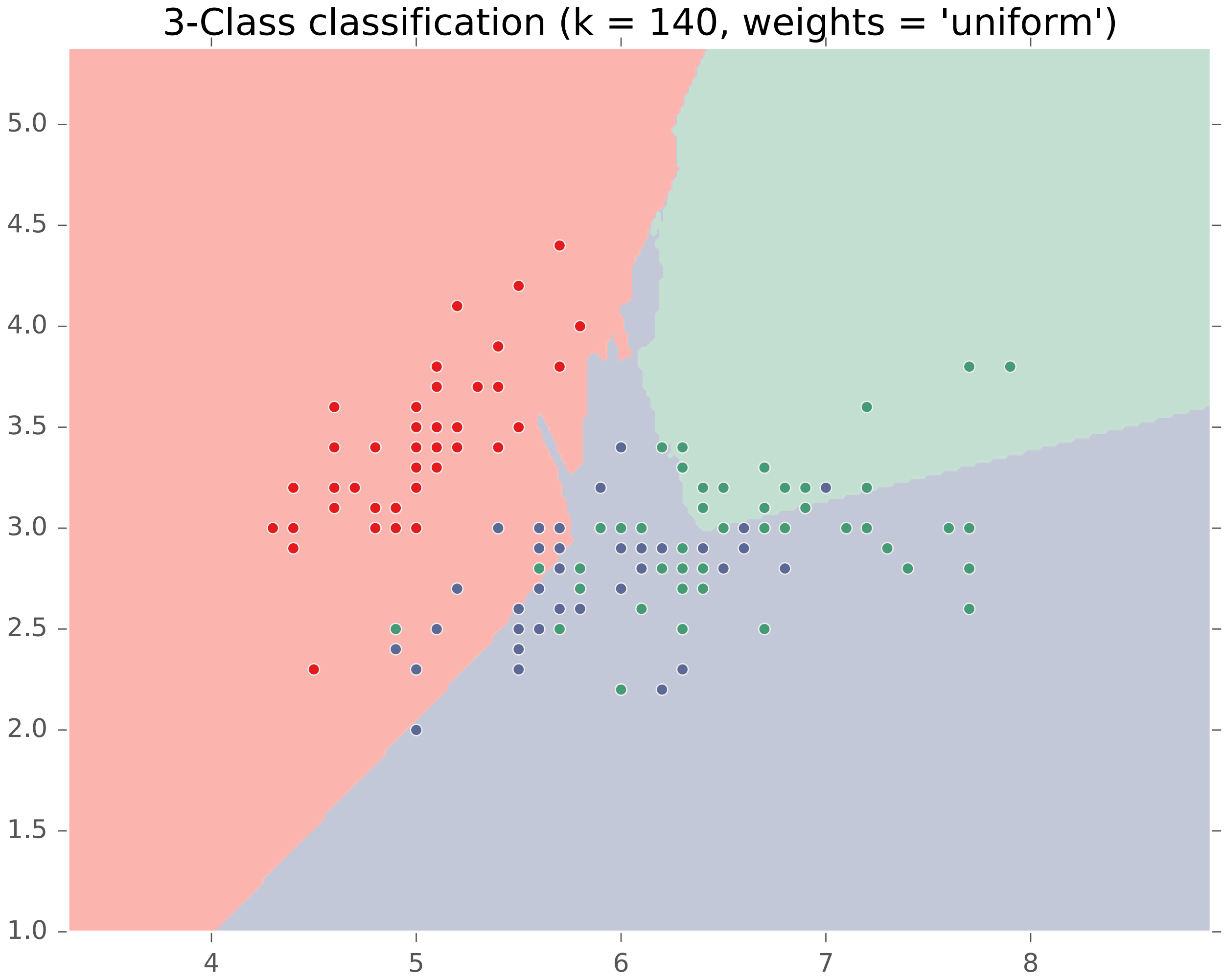
KNN on Fisher Iris Data



KNN on Fisher Iris Data

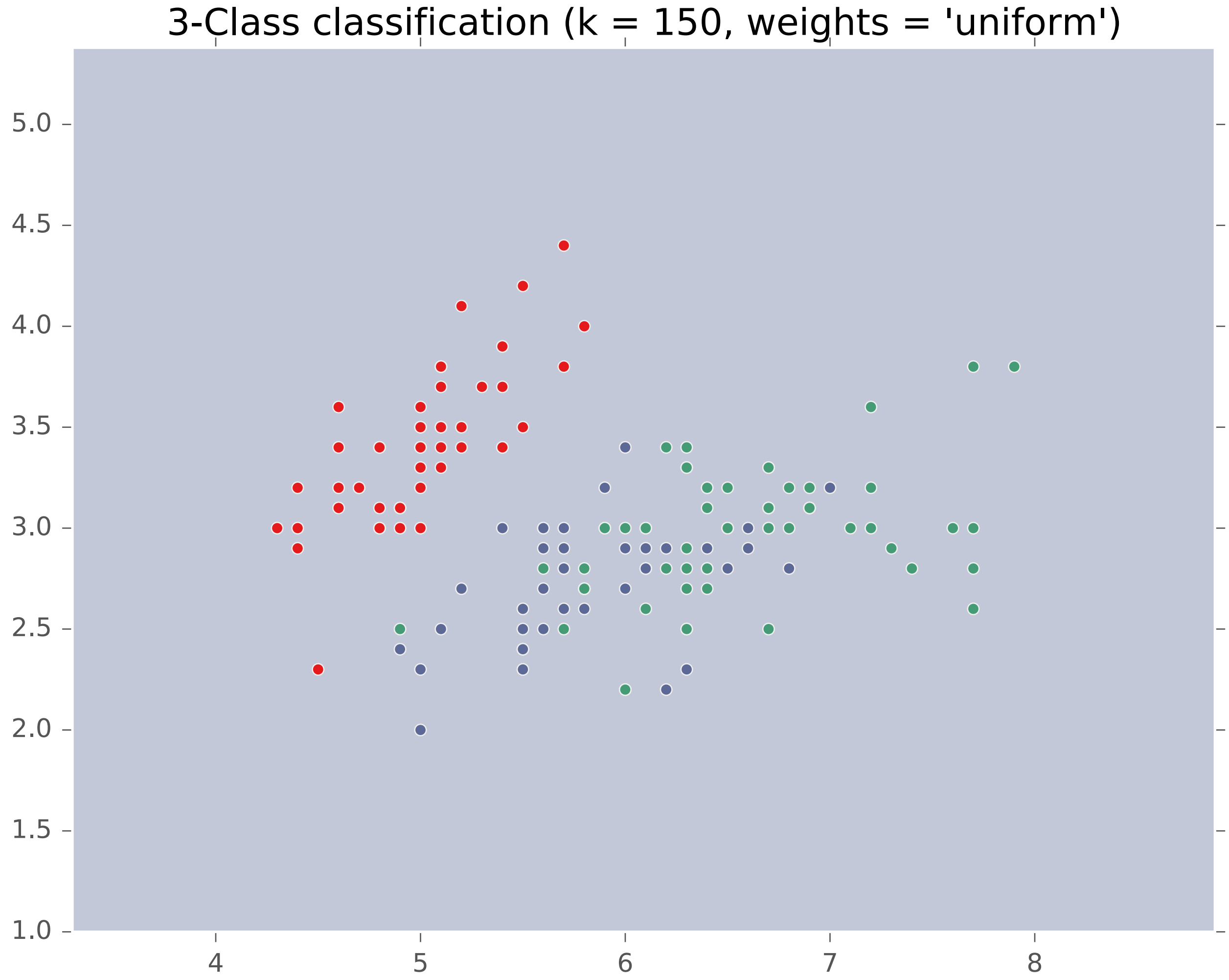


KNN on Fisher Iris Data



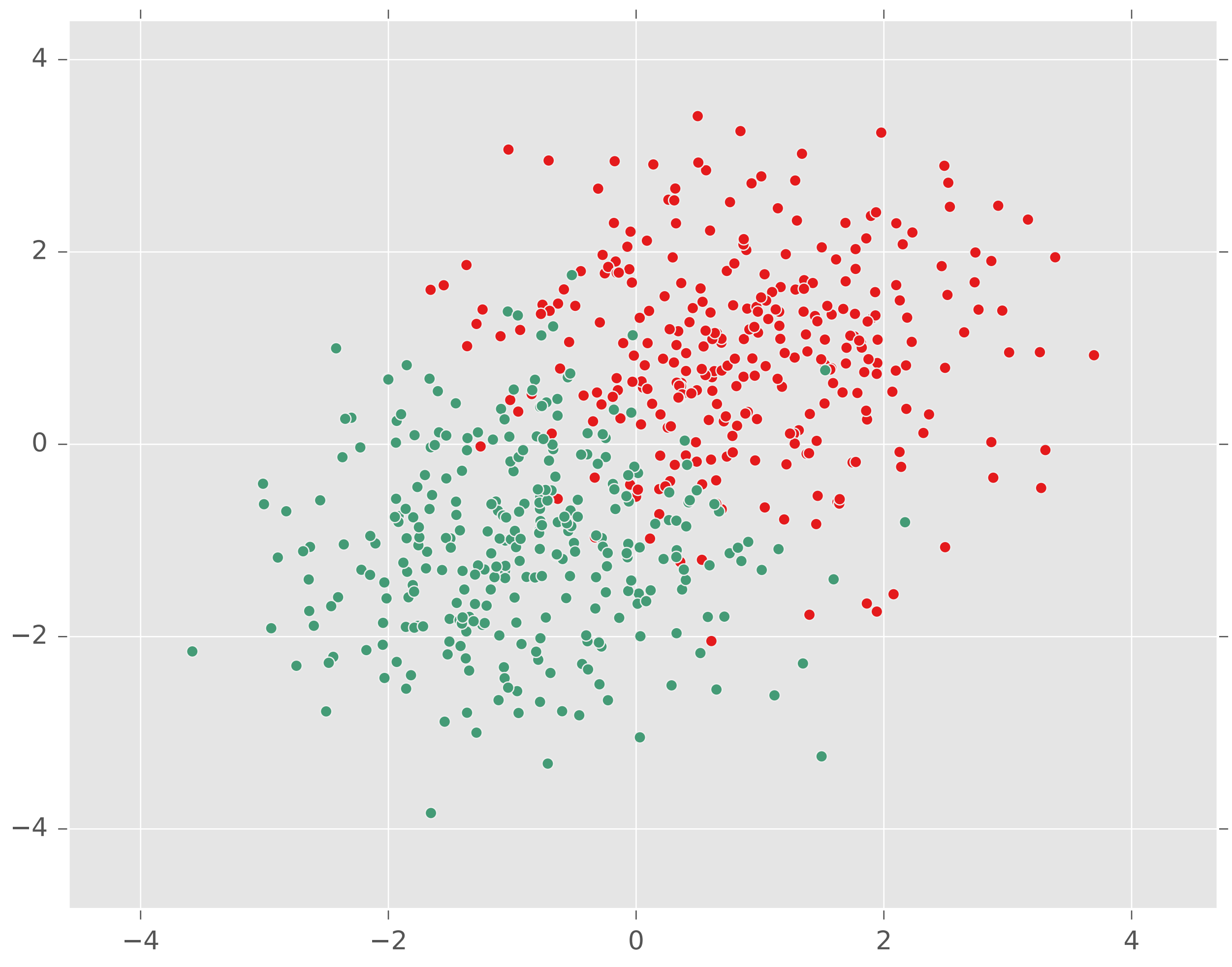
KNN on Fisher Iris Data

Special Case: Majority Vote

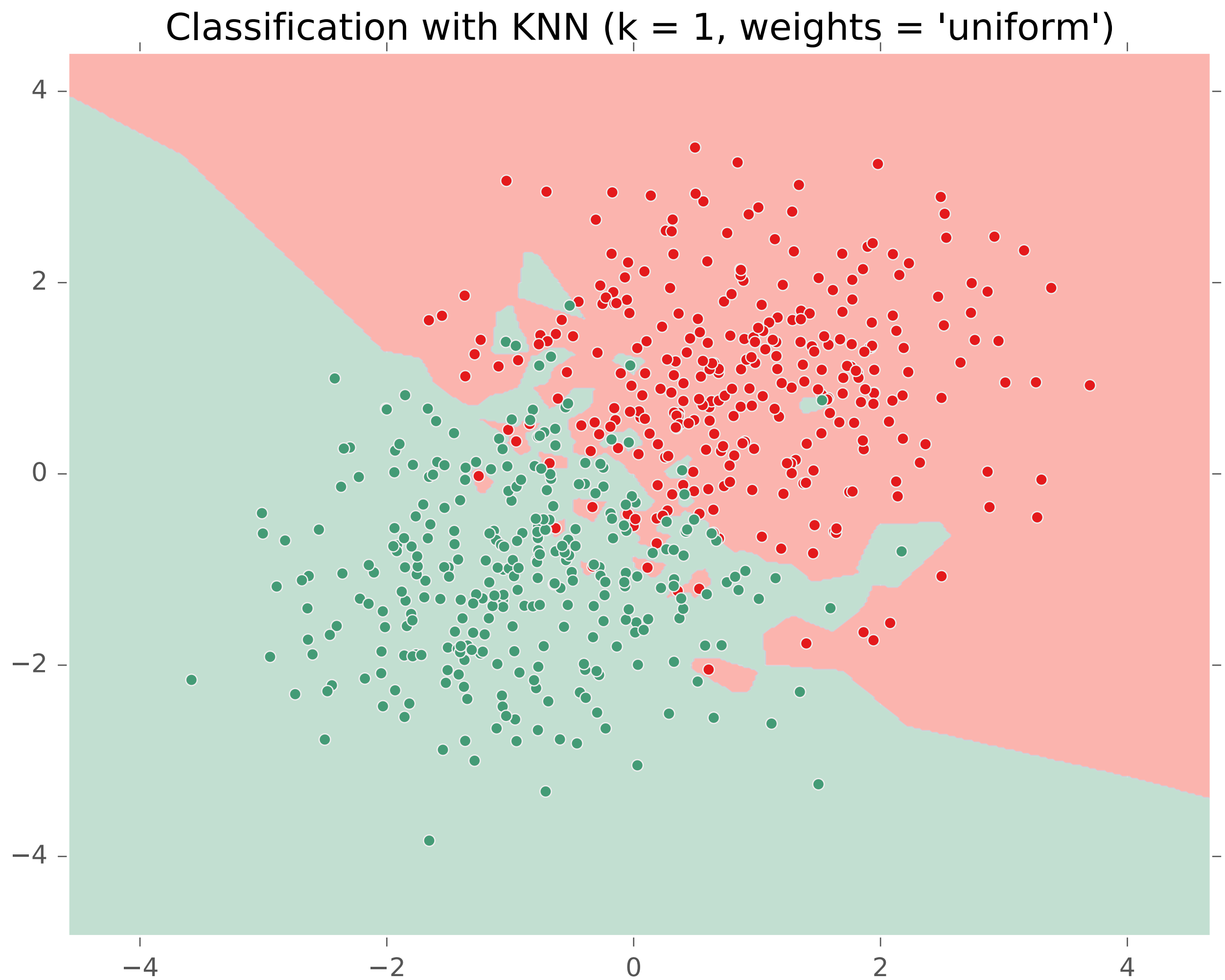


KNN ON GAUSSIAN DATA

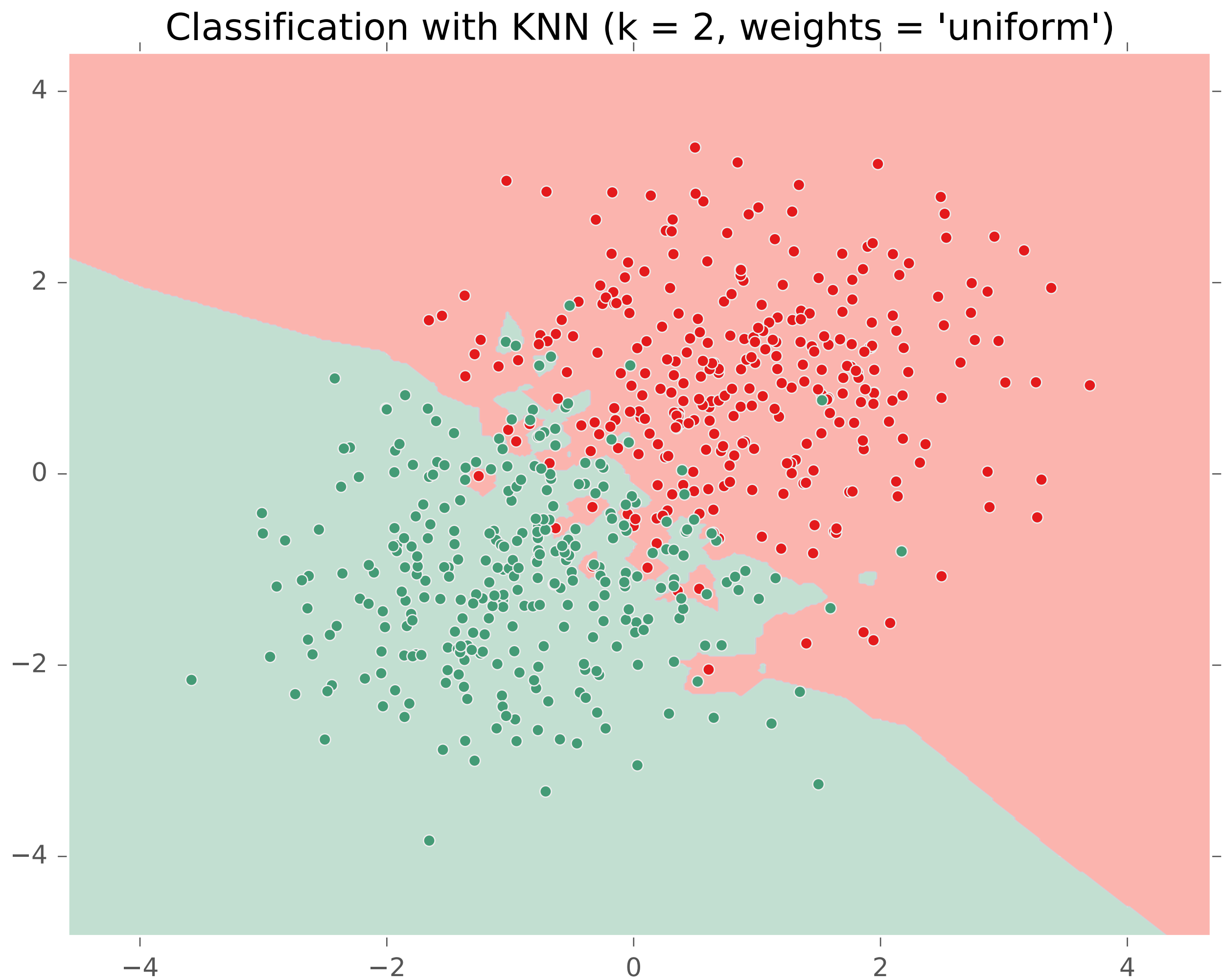
KNN on Gaussian Data



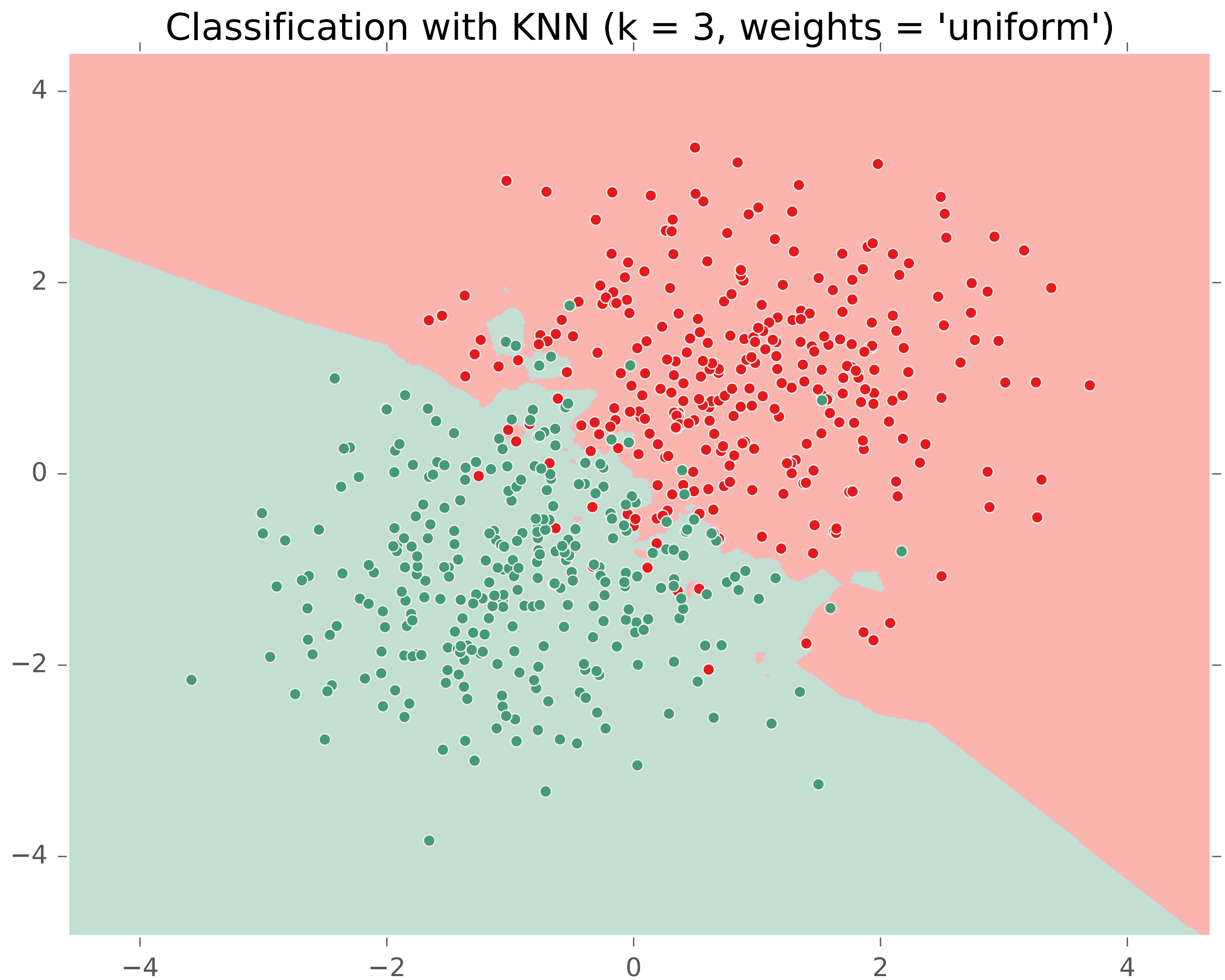
KNN on Gaussian Data



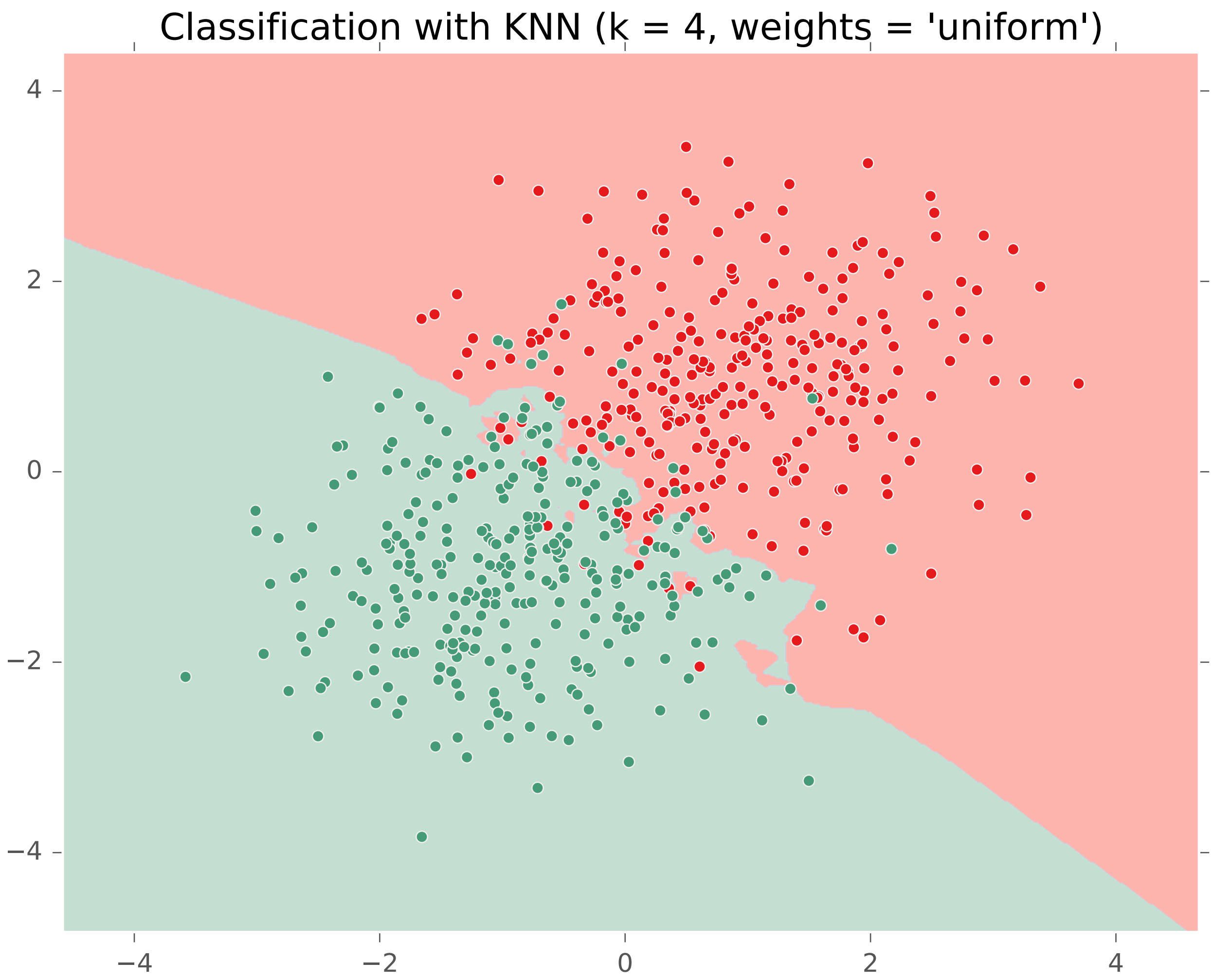
KNN on Gaussian Data



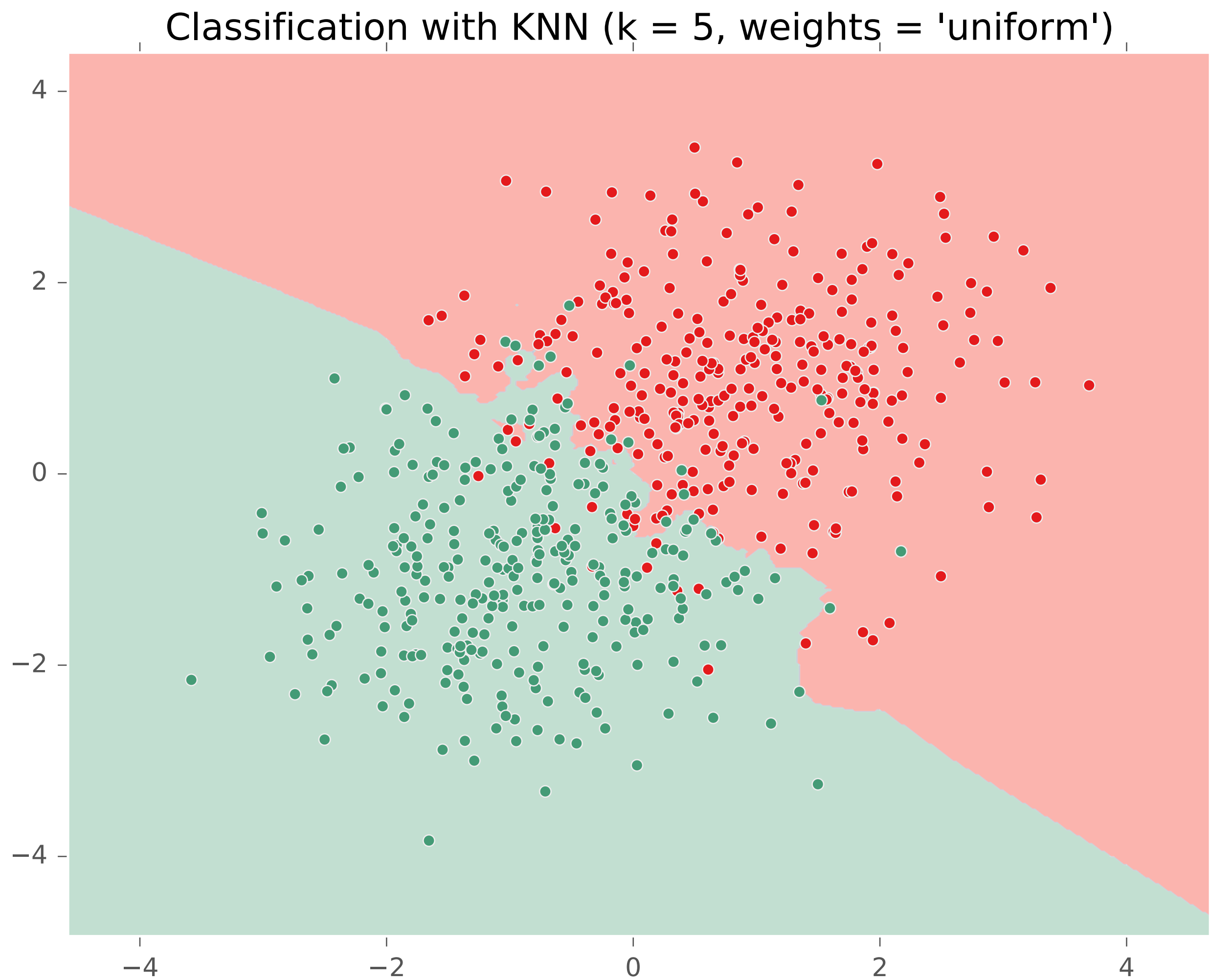
KNN on Gaussian Data



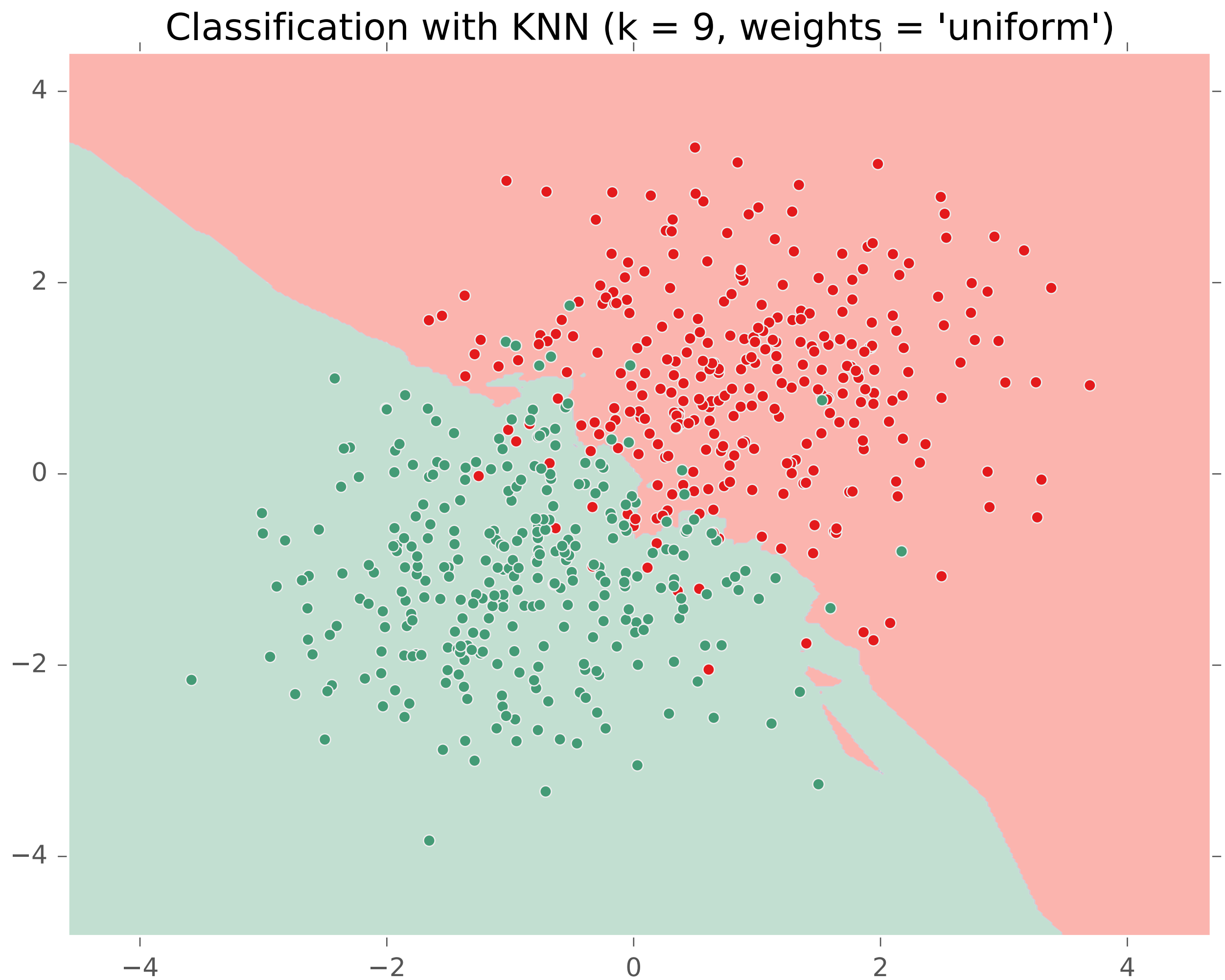
KNN on Gaussian Data



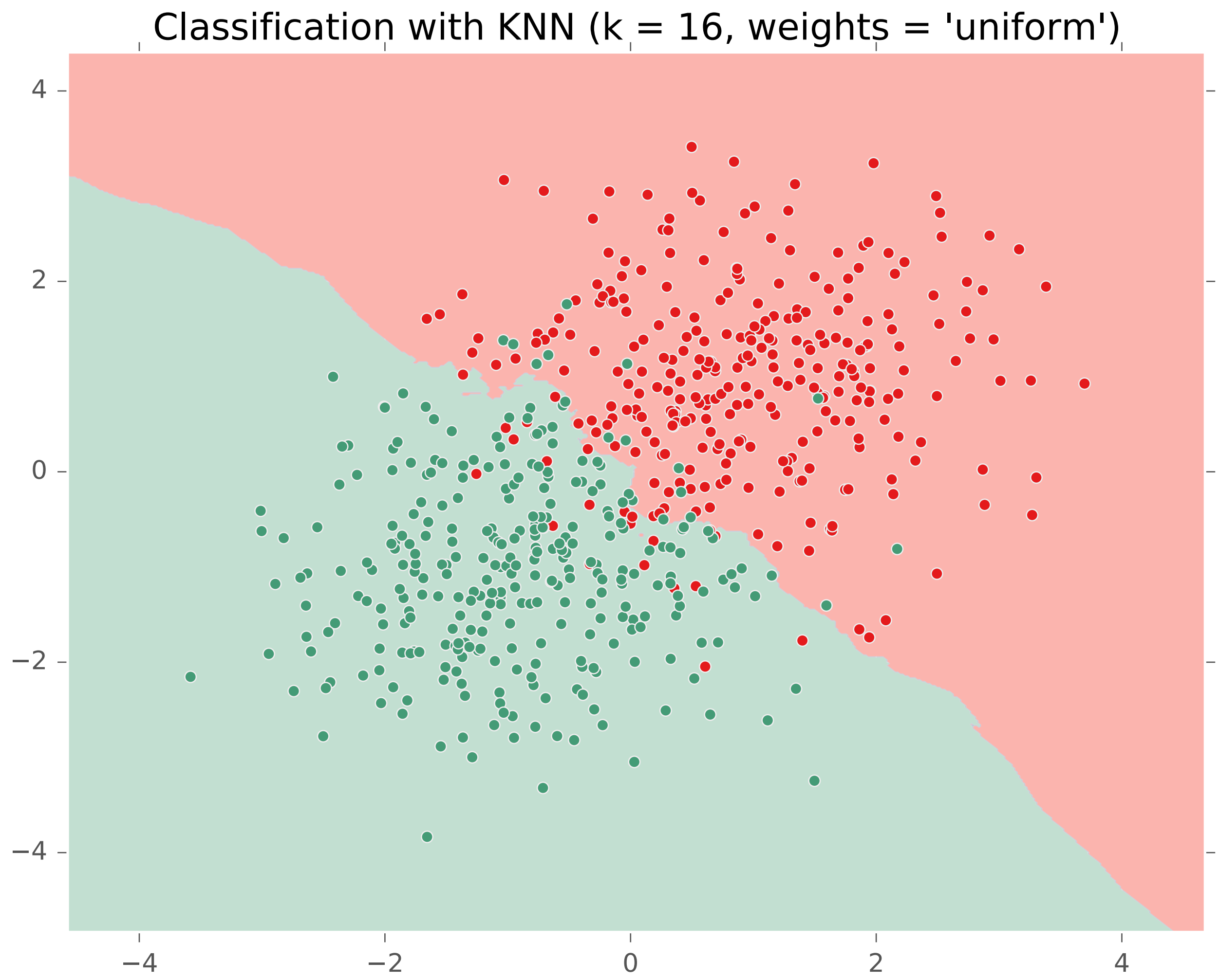
KNN on Gaussian Data



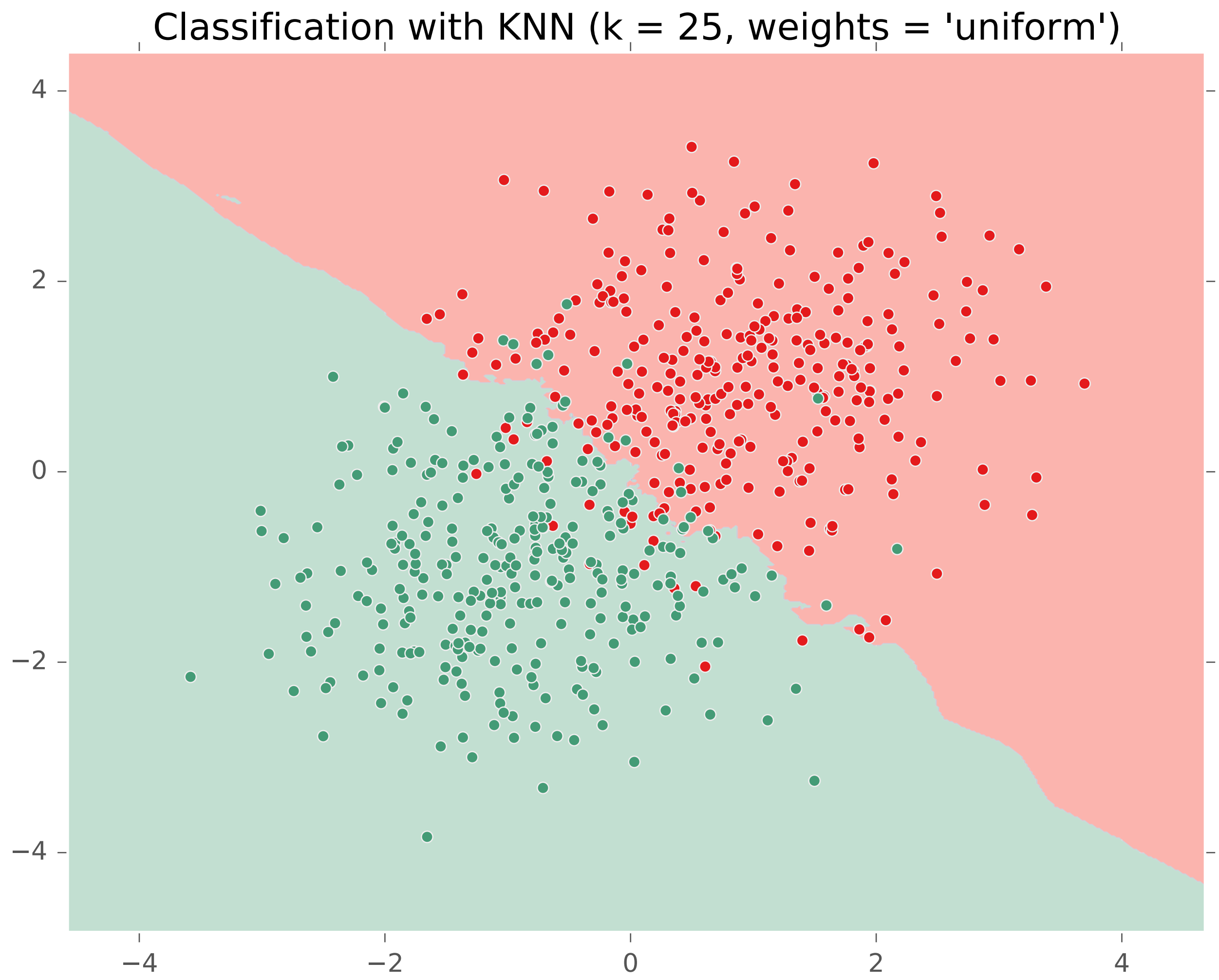
KNN on Gaussian Data



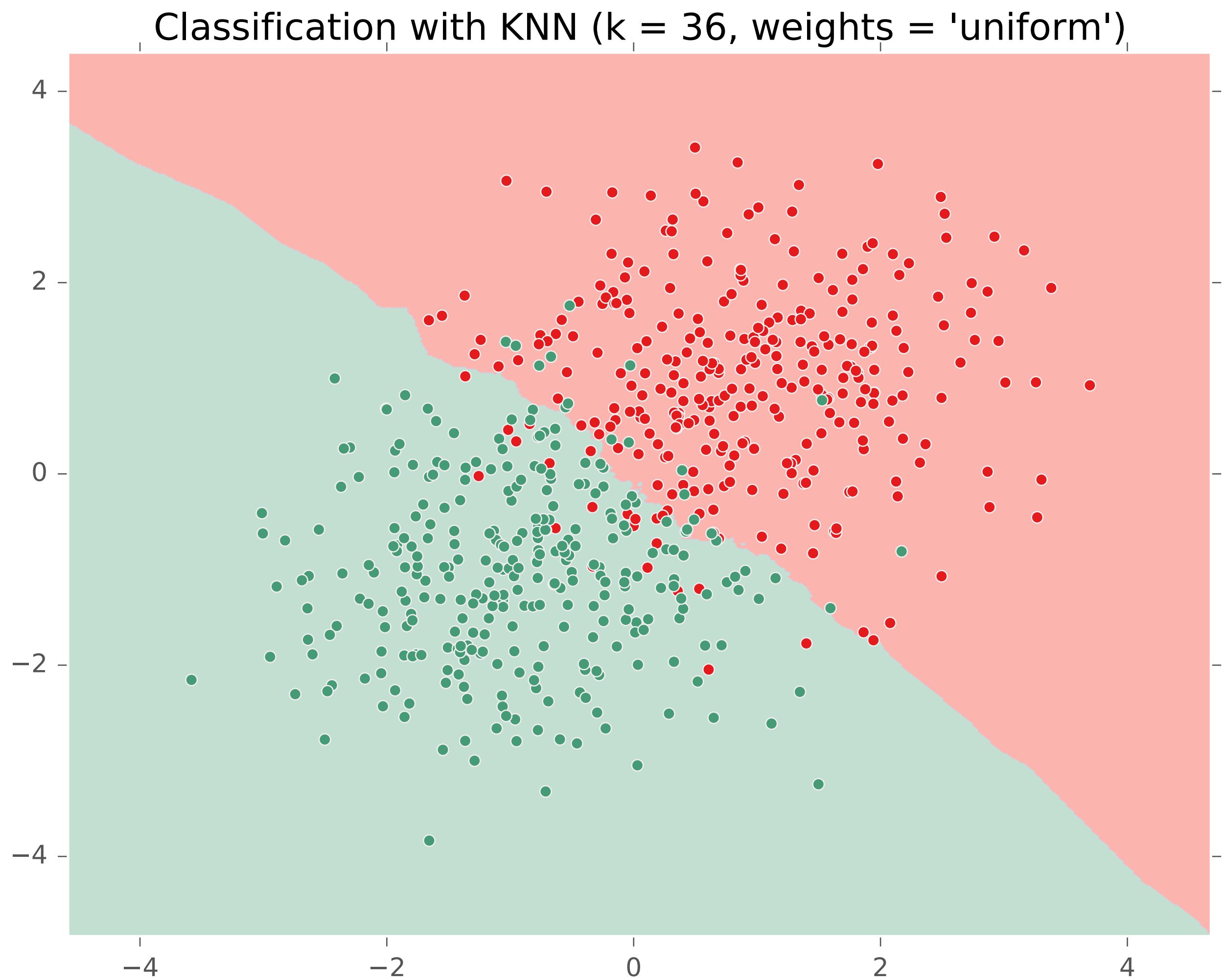
KNN on Gaussian Data



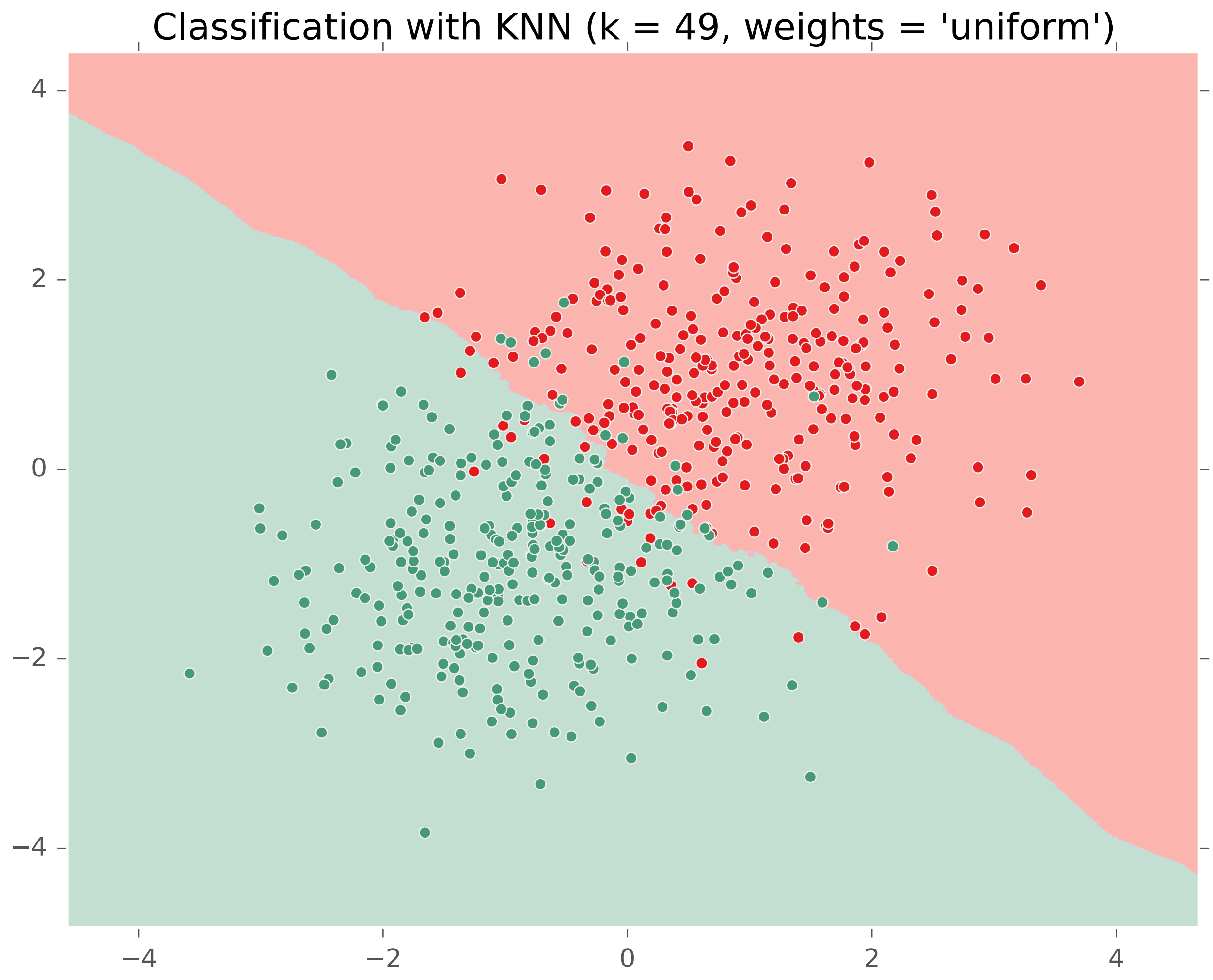
KNN on Gaussian Data



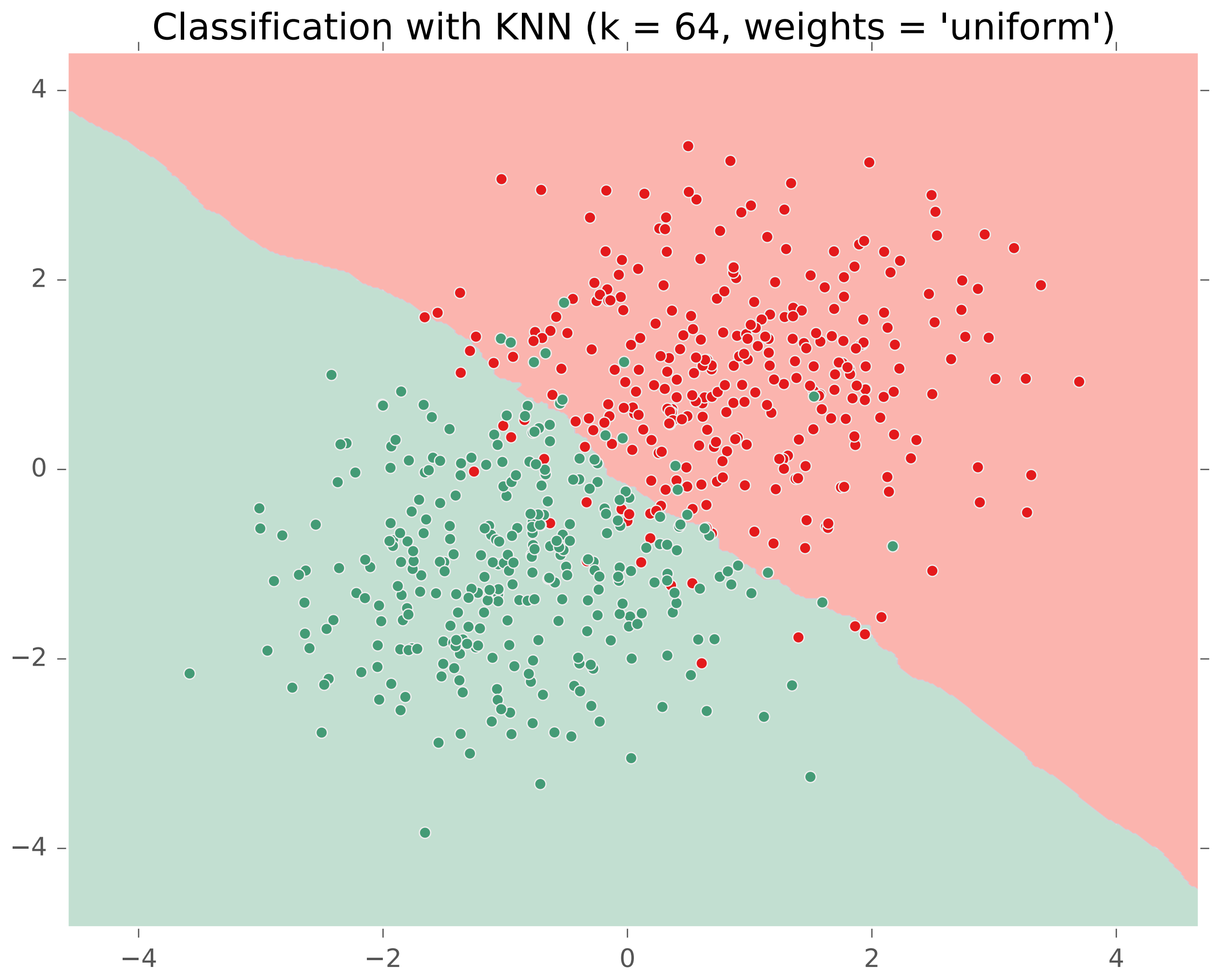
KNN on Gaussian Data



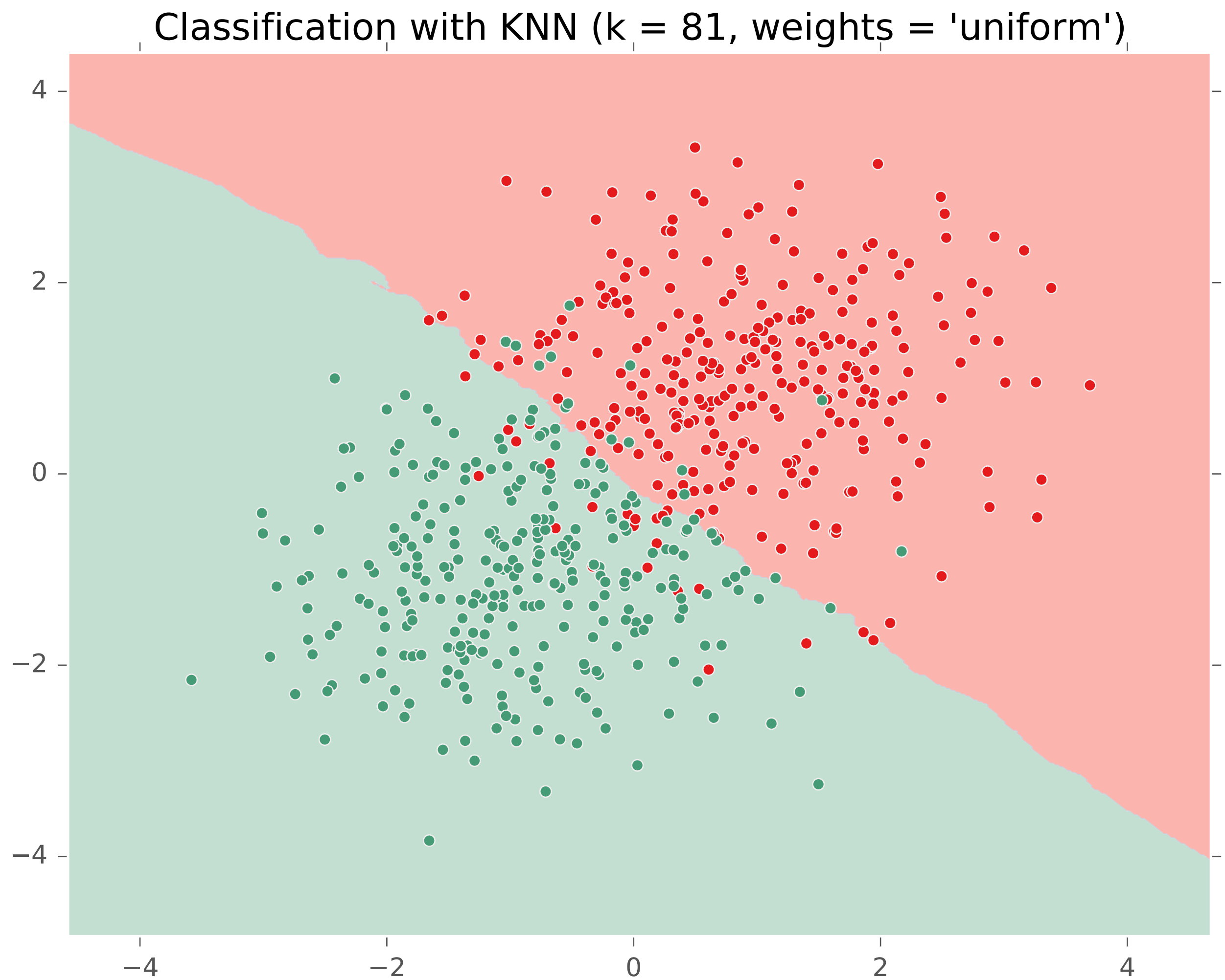
KNN on Gaussian Data



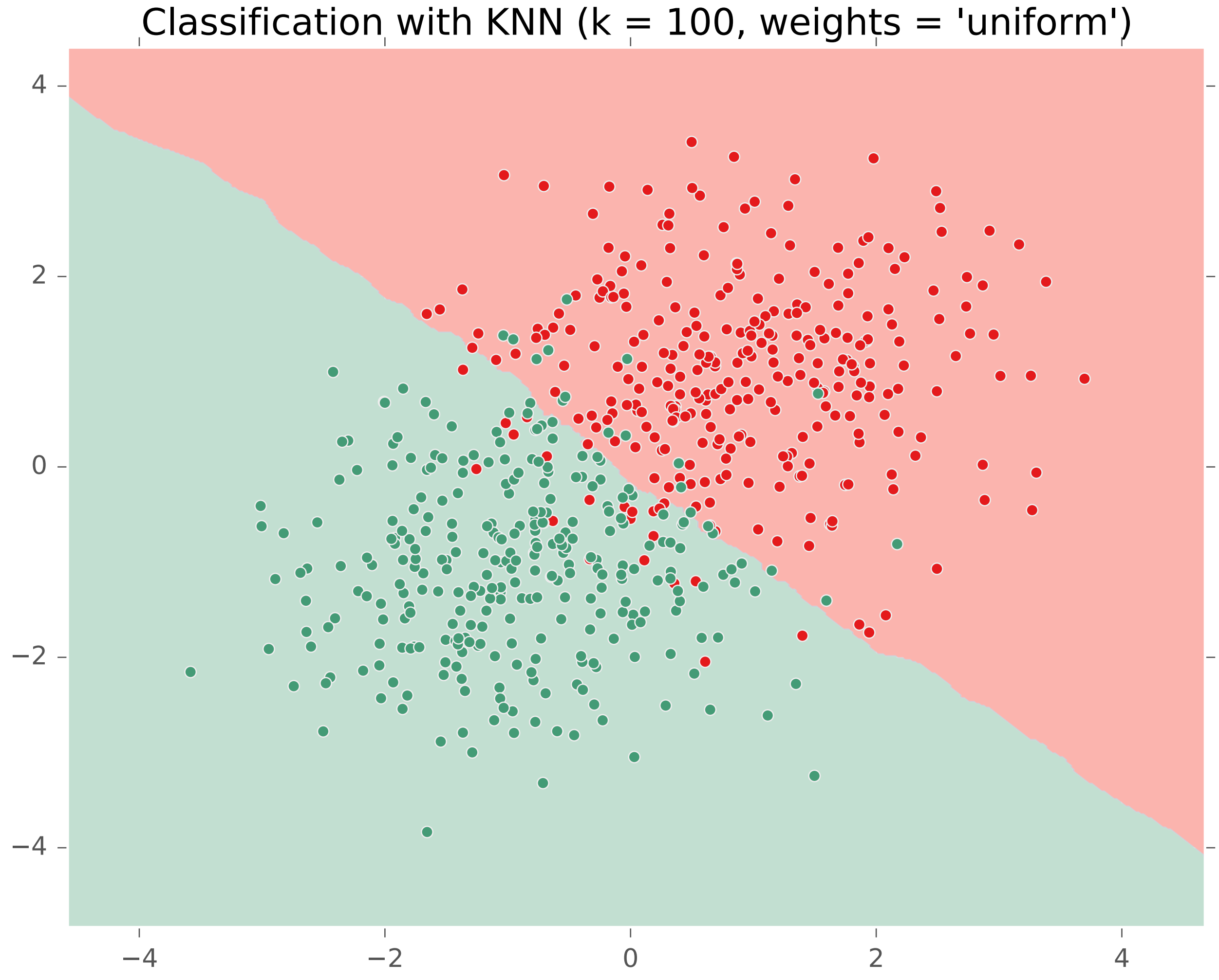
KNN on Gaussian Data



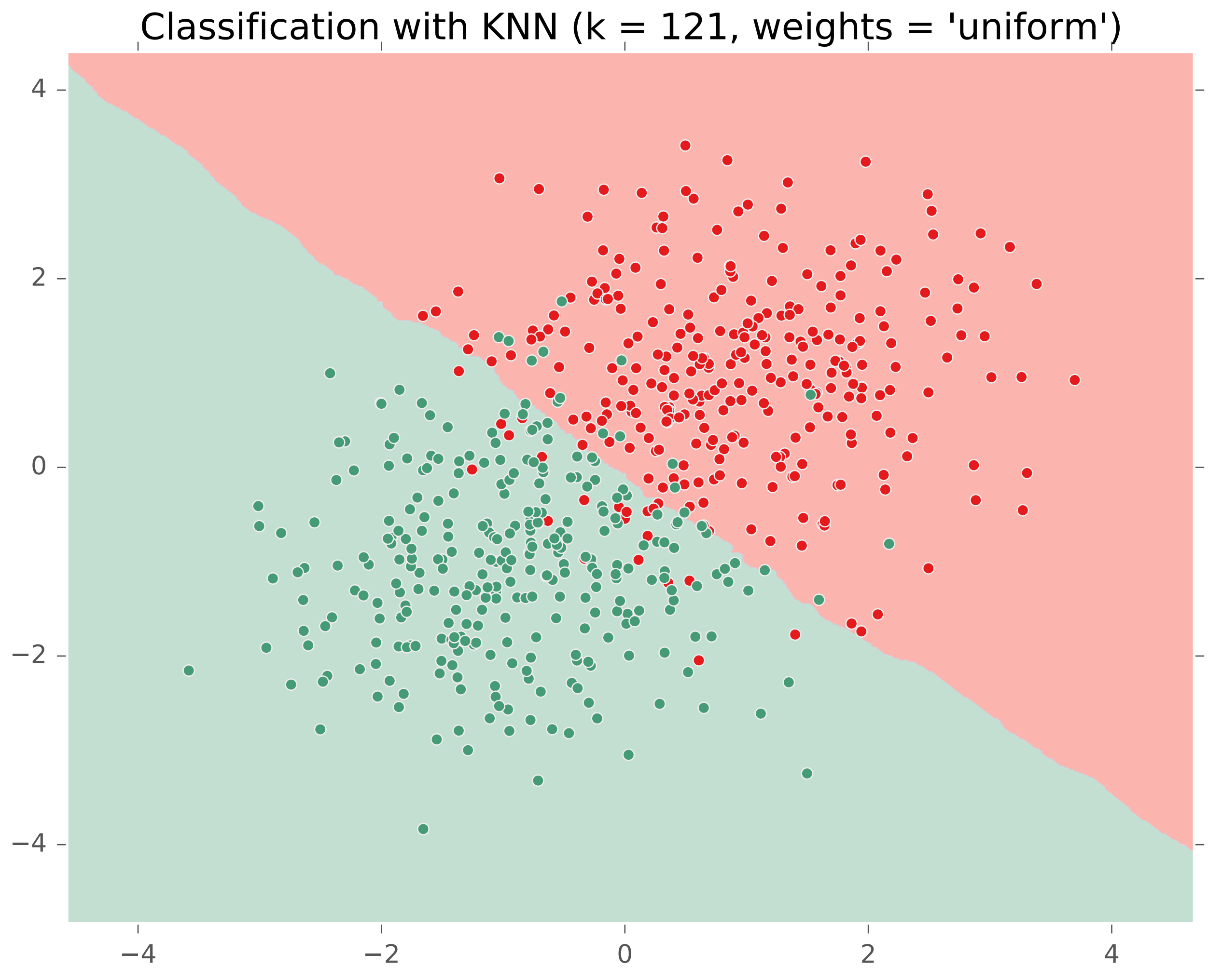
KNN on Gaussian Data



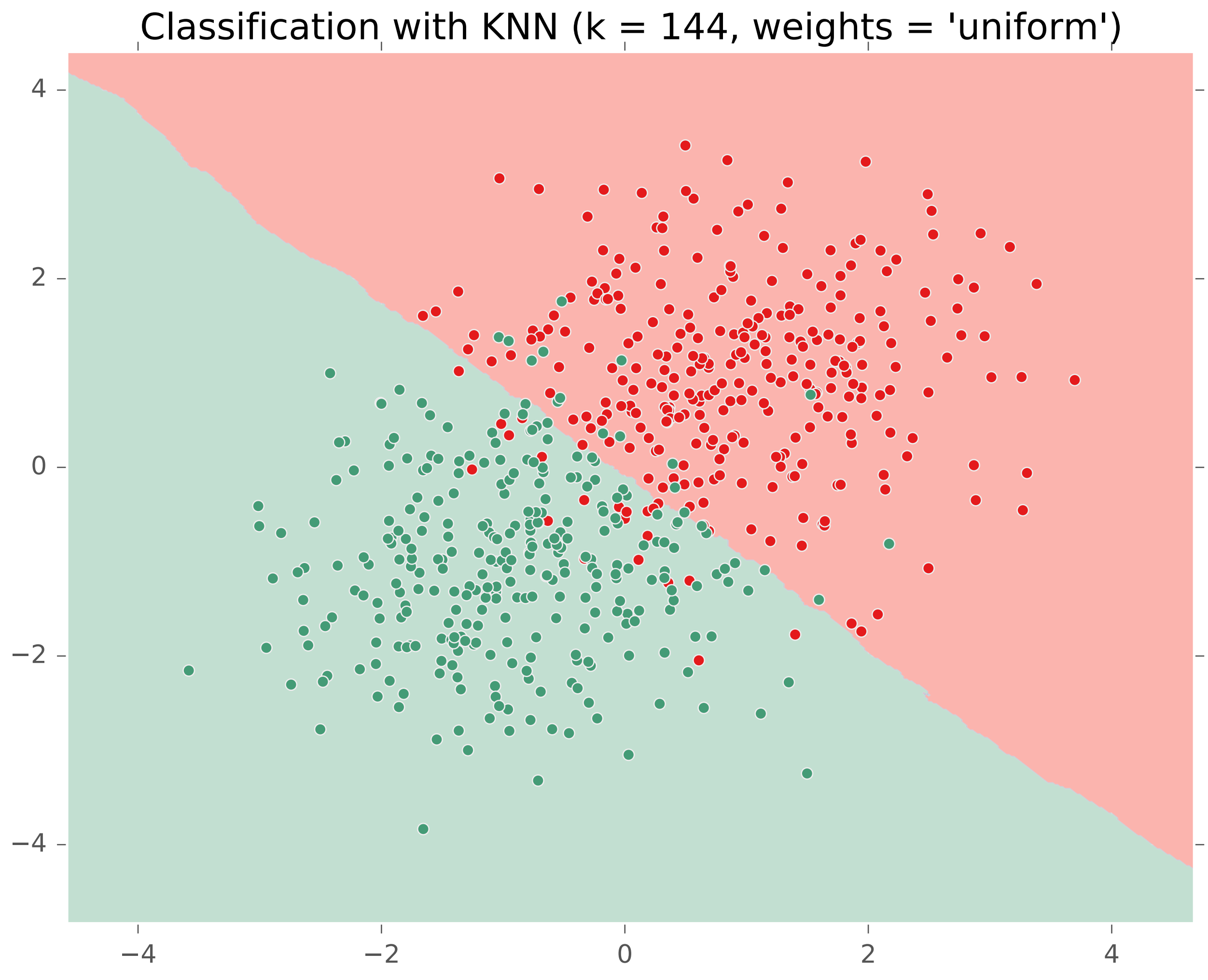
KNN on Gaussian Data



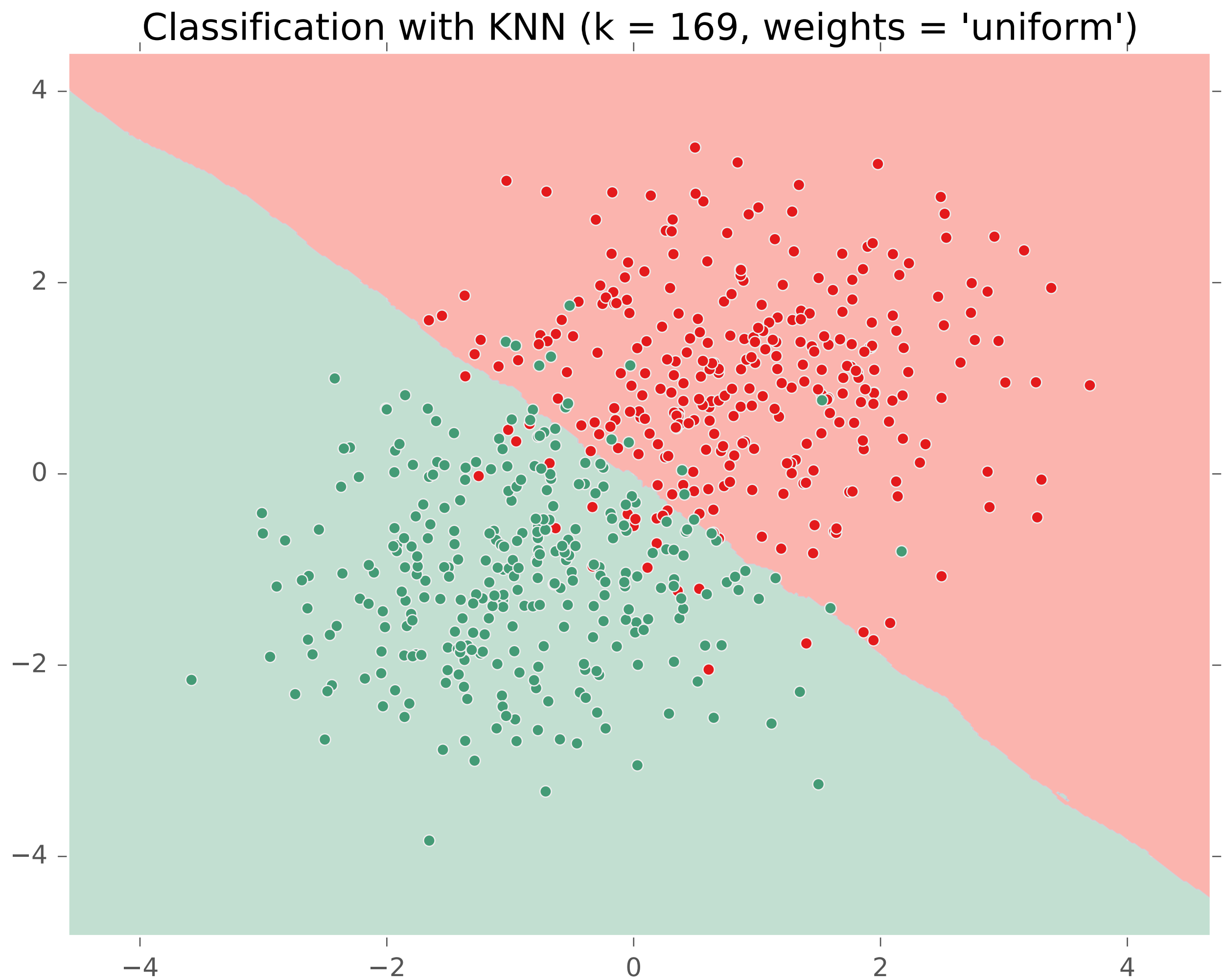
KNN on Gaussian Data



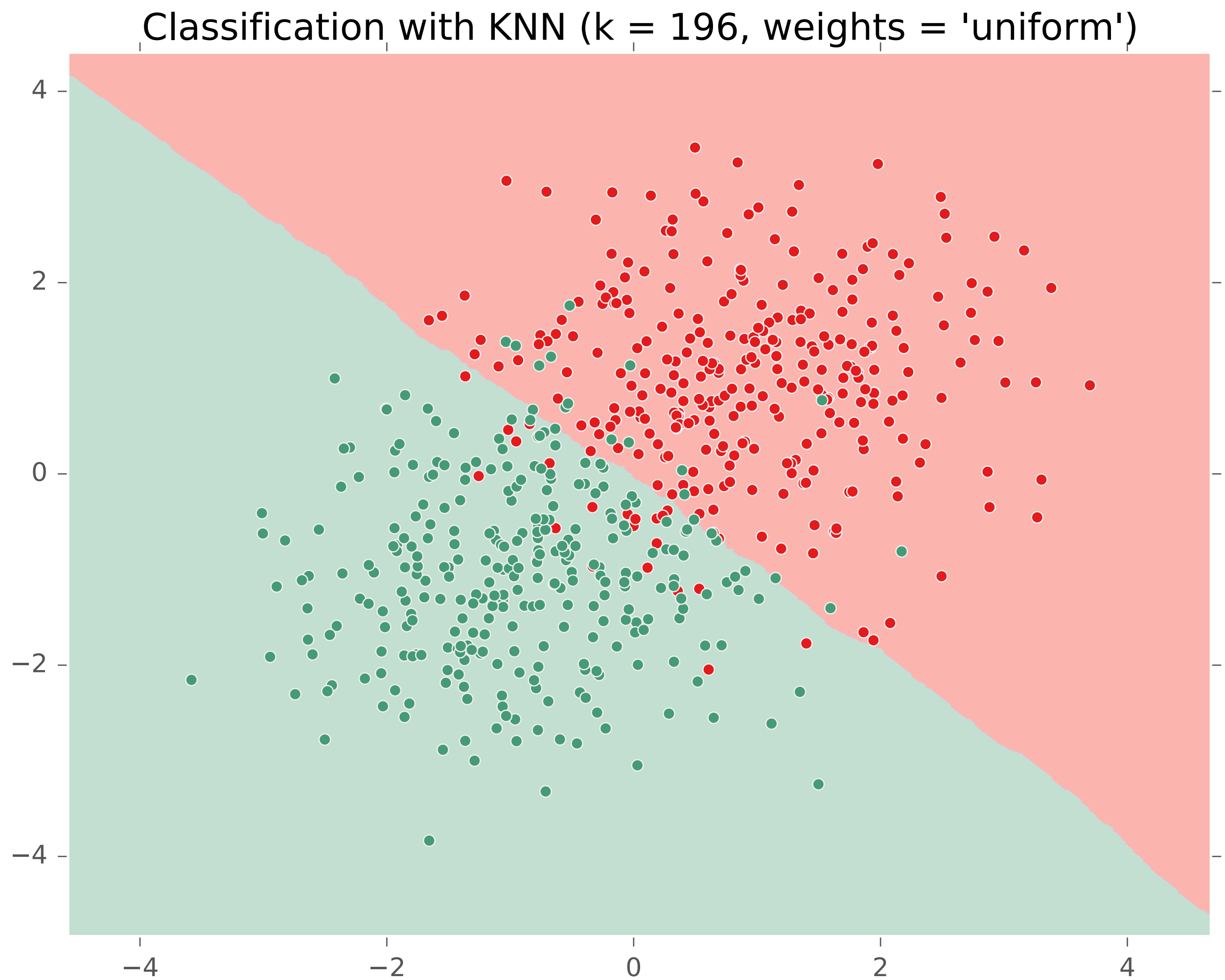
KNN on Gaussian Data



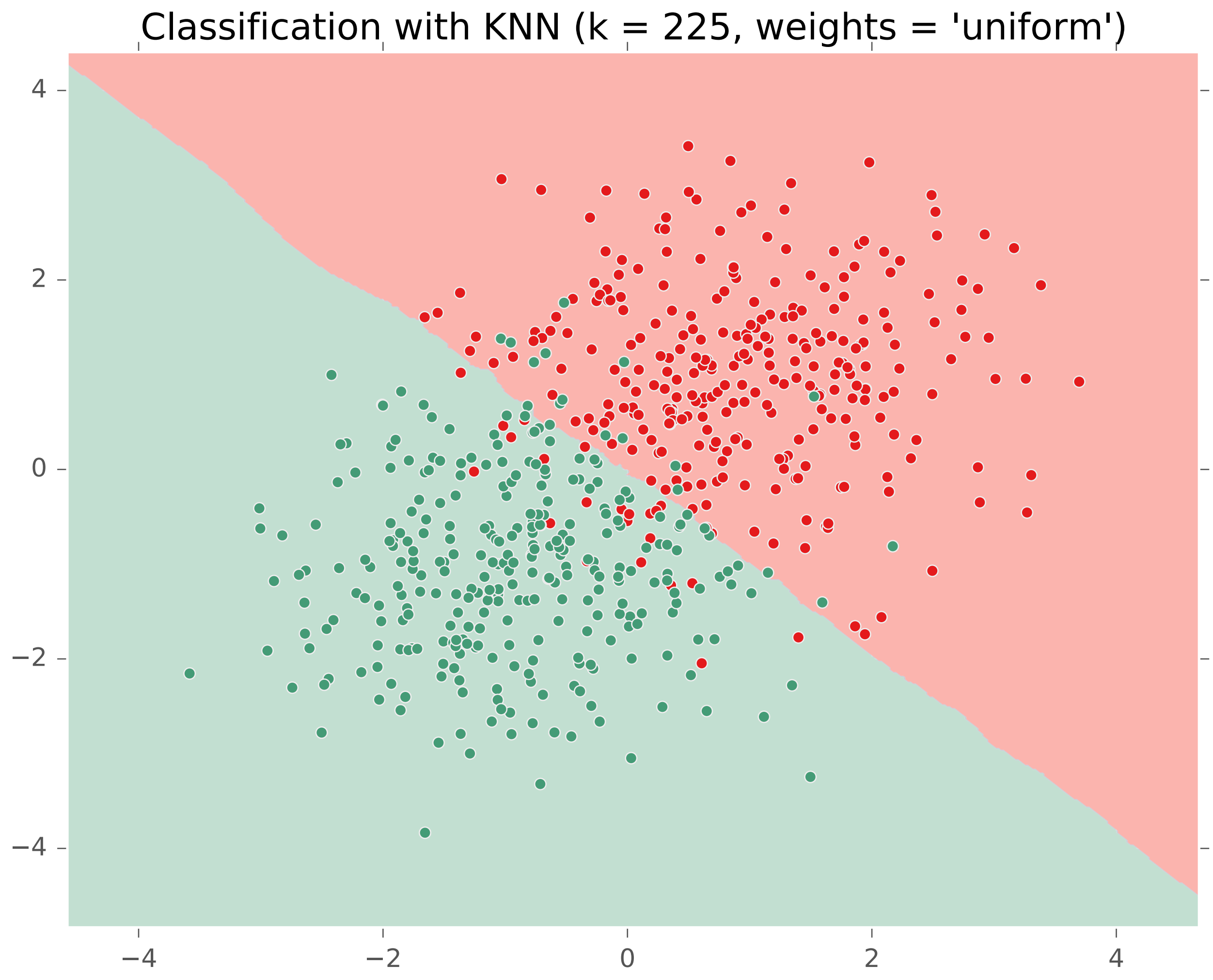
KNN on Gaussian Data



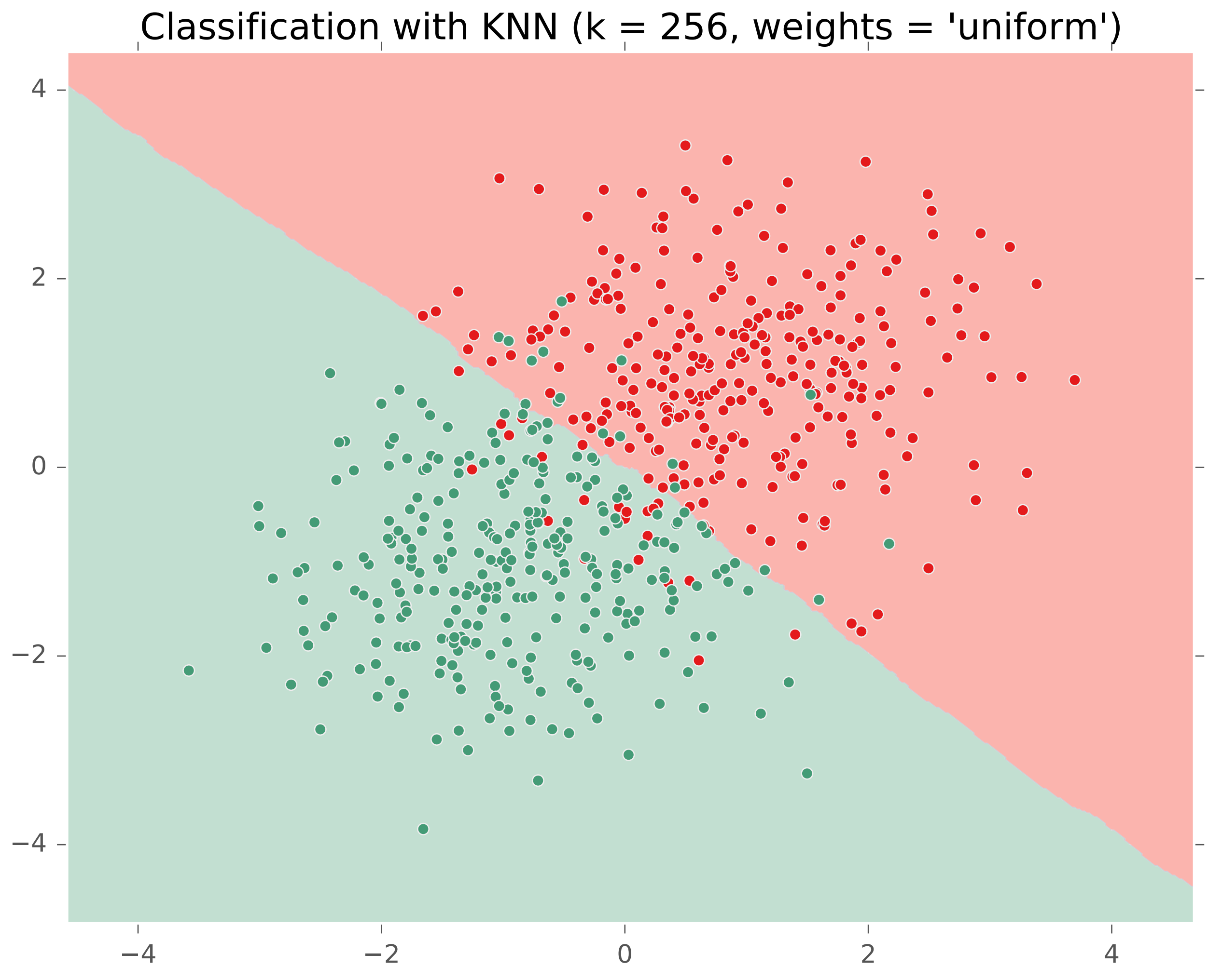
KNN on Gaussian Data



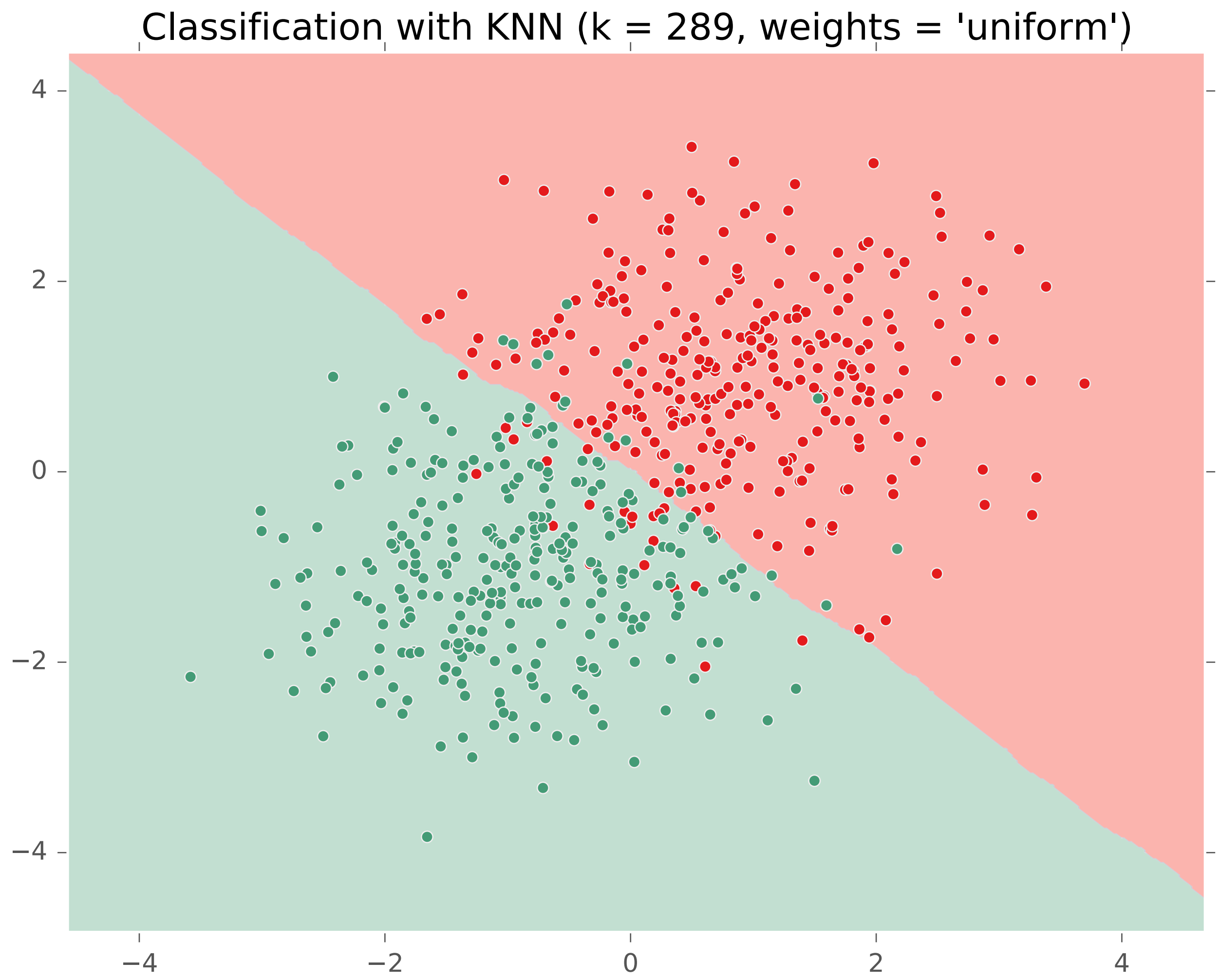
KNN on Gaussian Data



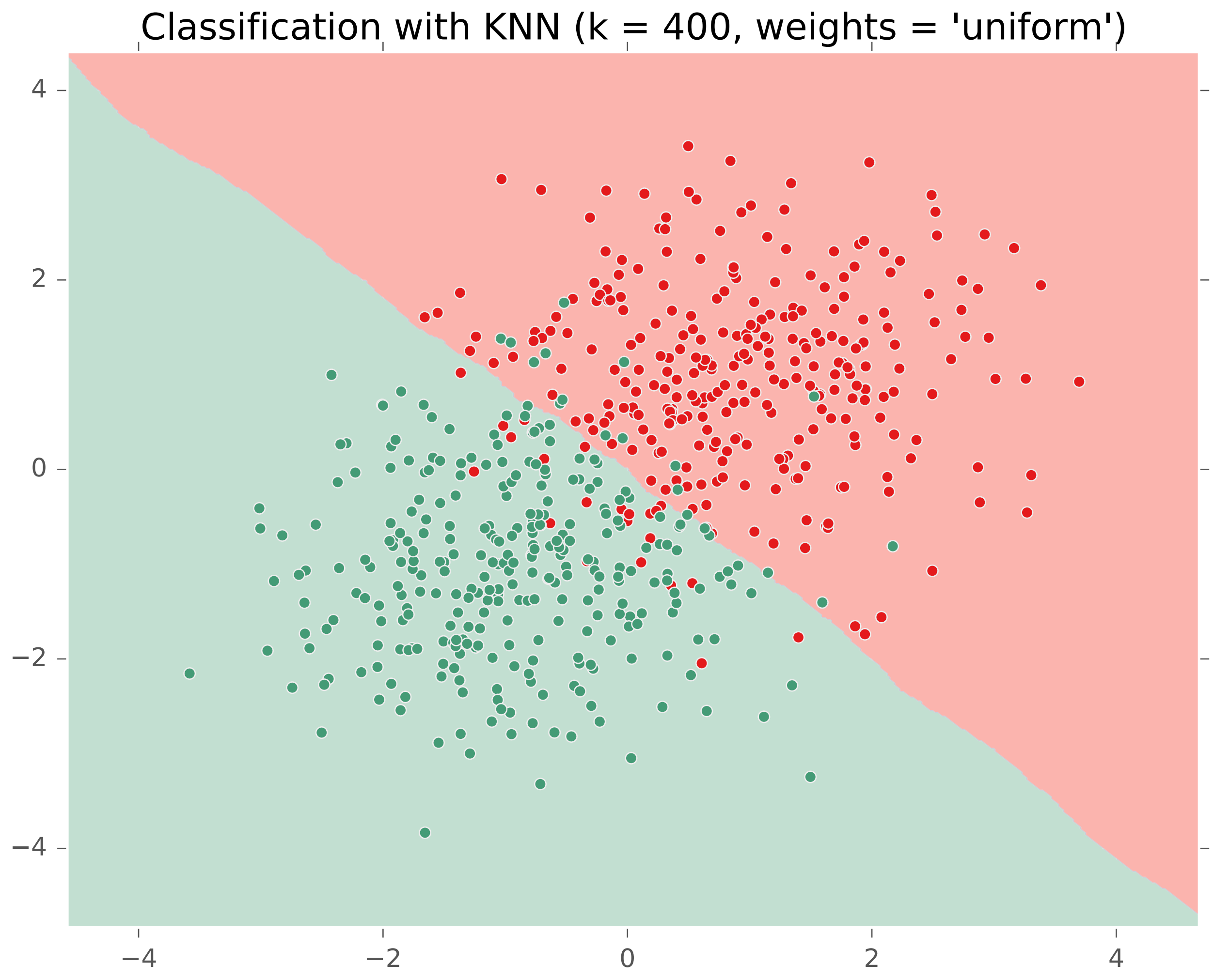
KNN on Gaussian Data



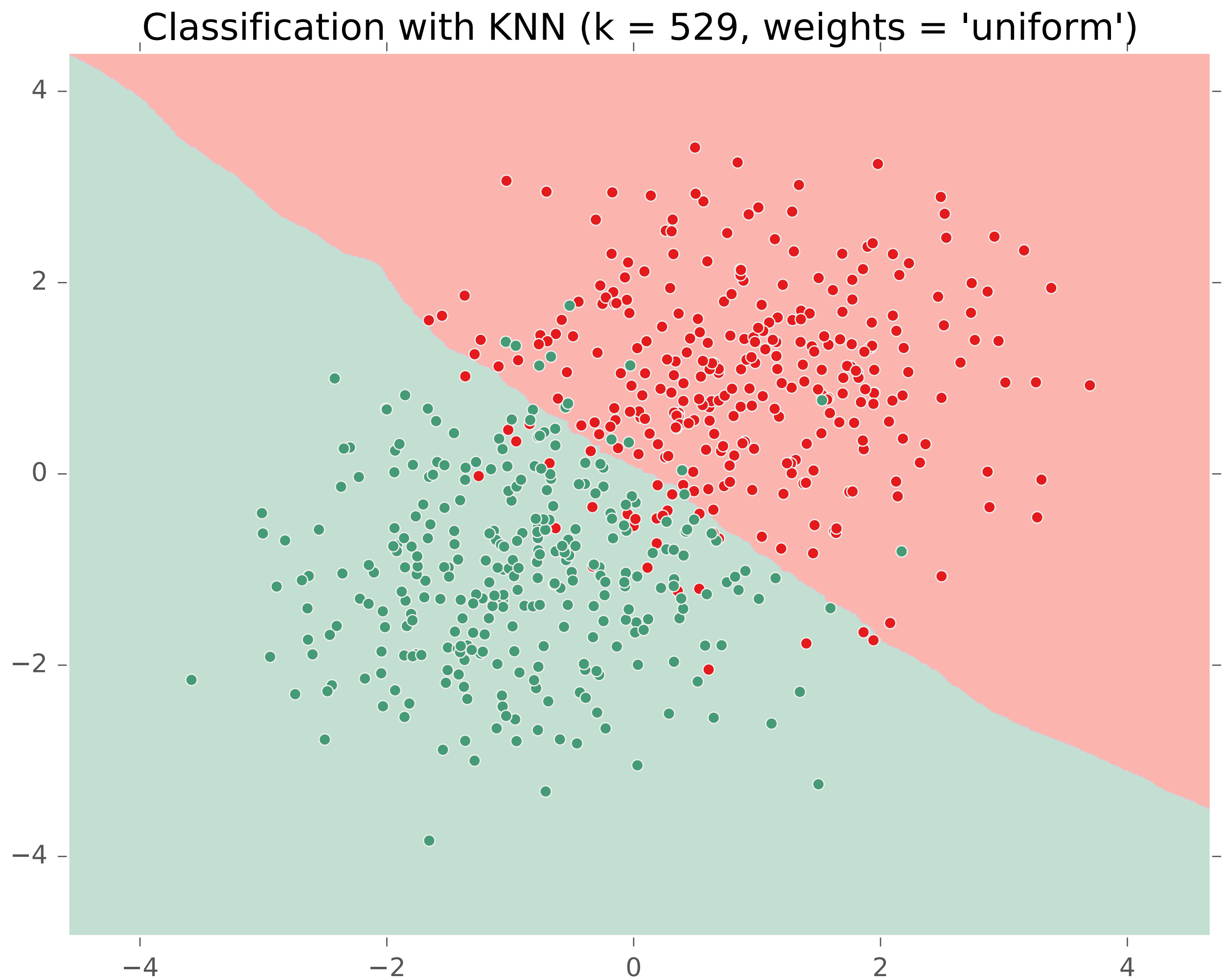
KNN on Gaussian Data



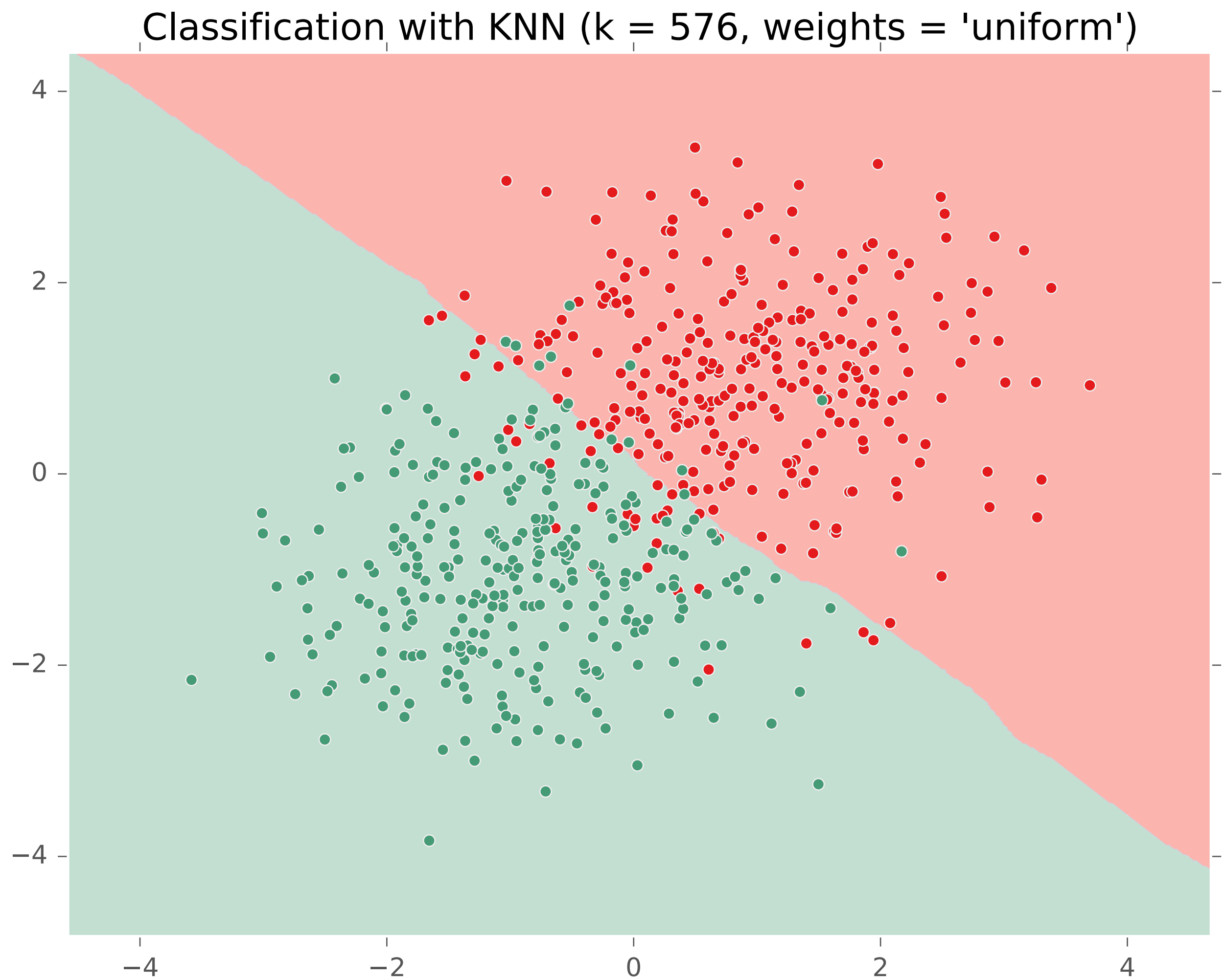
KNN on Gaussian Data



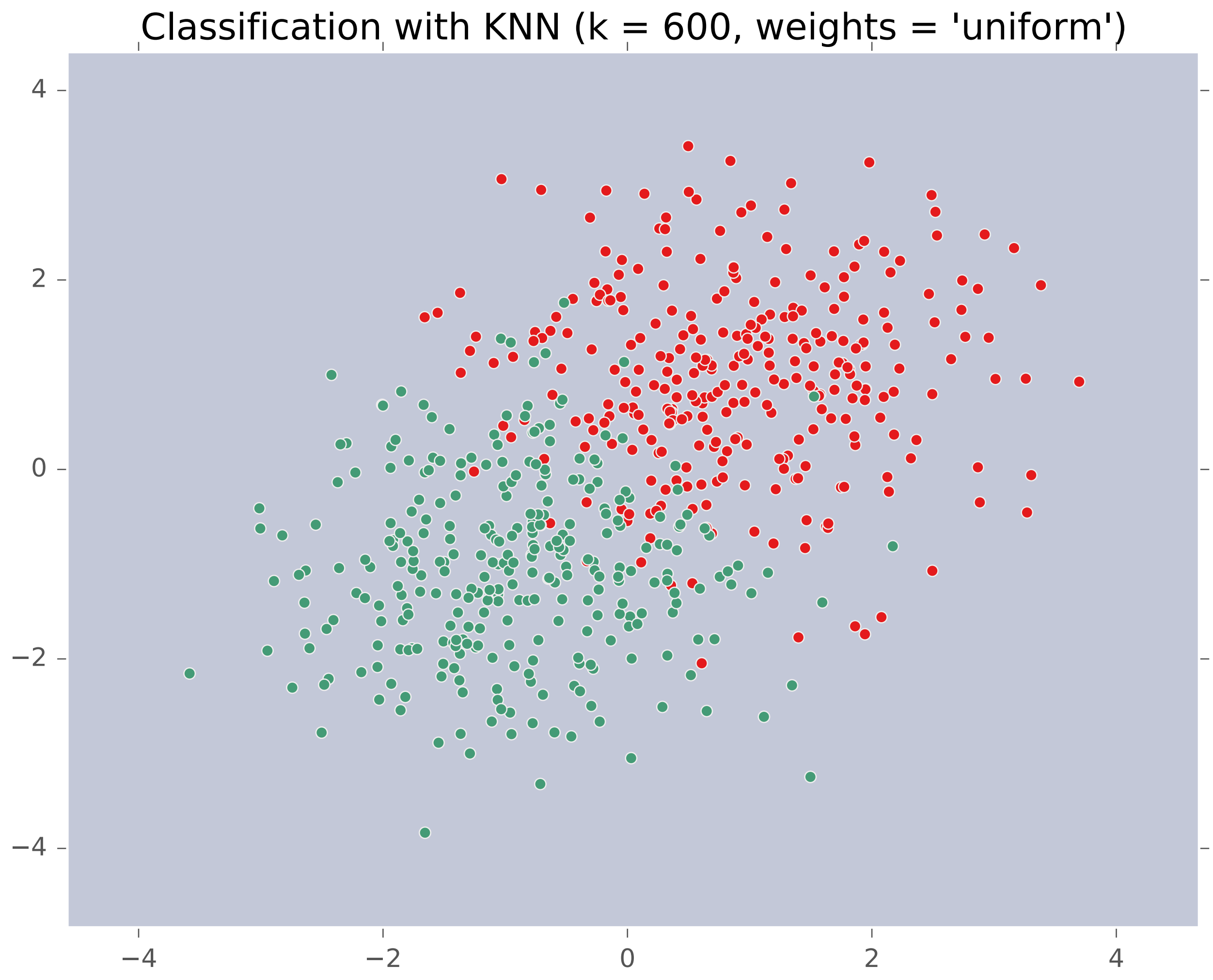
KNN on Gaussian Data



KNN on Gaussian Data



KNN on Gaussian Data



K-NEAREST NEIGHBORS

Questions

- How could k-Nearest Neighbors (KNN) be applied to **regression?**
- Can we do better than majority vote? (e.g. **distance-weighted** KNN)
- Where does the Cover & Hart (1967) **Bayes error rate bound** come from**?**

KNN Learning Objectives

You should be able to…

- Describe a dataset as points in a high dimensional space [CIML]
- Implement k-Nearest Neighbors with O(N) prediction
- Describe the inductive bias of a k-NN classifier and relate it to feature scale [a la. CIML]
- Sketch the decision boundary for a learning algorithm (compare k-NN and DT)
- State Cover & Hart (1967)'s large sample analysis of a nearest neighbor classifier
- Invent "new" k-NN learning algorithms capable of dealing with even k
- Explain computational and geometric examples of the curse of dimensionality

k-Nearest Neighbors

But how do we choose k?

MODEL SELECTION

Model Selection

WARNING:

- In some sense, our discussion of model selection is premature.
- The models we have considered thus far are fairly simple.
- The models and the many decisions available to the data scientist wielding them will grow to be much more complex than what we've seen so far.

Model Selection

Statistics

- *Def*: a **model** defines the data generation process (i.e. a set or family of parametric probability distributions)
- *Def*: **model parameters** are the values that give rise to a particular probability distribution in the model family
- *Def*: **learning** (aka. estimation) is the process of finding the parameters that best fit the data
- *Def*: **hyperparameters** are the parameters of a prior distribution over parameters

Machine Learning

- *Def*: (loosely) a **model** defines the hypothesis space over which learning performs its search
- *Def*: **model parameters** are the numeric values or structure selected by the learning algorithm that give rise to a hypothesis
- *Def*: the **learning algorithm** defines the data-driven search over the hypothesis space (i.e. search for good parameters)
- *Def*: **hyperparameters** are the tunable aspects of the model, that the learning algorithm does *not* select

Model Selection

Example: Decision Tree

- model = set of all possible trees, possibly restricted by some hyperparameters (e.g. max depth)
- parameters = structure of a specific decision tree
- learning algorithm = ID3, CART, etc.
- hyperparameters = max-depth, threshold for splitting criterion, etc.

Machine Learning

- *Def*: (loosely) a **model** defines the hypothesis space over which learning performs its search
- *Def*: **model parameters** are the numeric values or structure selected by the learning algorithm that give rise to a hypothesis
- *Def*: the **learning algorithm** defines the data-driven search over the hypothesis space (i.e. search for good parameters)
- *Def*: **hyperparameters** are the tunable aspects of the model, that the learning algorithm does *not* select

Model Selection

Example: k-Nearest Neighbors

- model = set of all possible nearest neighbors classifiers
- parameters = none (KNN is an instance-based or non-parametric method)
- learning algorithm = for naïve setting, just storing the data
- hyperparameters = k, the number of neighbors to consider

Machine Learning

- *Def*: (loosely) a **model** defines the hypothesis space over which learning performs its search
- *Def*: **model parameters** are the numeric values or structure selected by the learning algorithm that give rise to a hypothesis
- *Def*: the **learning algorithm** defines the data-driven search over the hypothesis space (i.e. search for good parameters)
- *Def*: **hyperparameters** are the tunable aspects of the model, that the learning algorithm does *not* select

Model Selection

Example: Perceptron

- model = set of all linear separators
- parameters = vector of weights (one for each feature)
- learning algorithm = mistake based updates to the parameters
- hyperparameters = none (unless using some variant such as averaged perceptron)

Machine Learning

- *Def*: (loosely) a **model** defines the hypothesis space over which learning performs its search
- *Def*: **model parameters** are the numeric values or structure selected by the learning algorithm that give rise to a hypothesis
- *Def*: the **learning algorithm** defines the data-driven search over the hypothesis space (i.e. search for good parameters)
- *Def*: **hyperparameters** are the tunable aspects of the model, that the learning algorithm does *not* select

Model Selection

If "learning" is all about picking the best parameters how do we pick the best hyperparameters?

Statistics

- *Def*: a **model** defines the data generation pro family of para distributions)
- *Def*: **model pa** values that giv particular prob distribution in
- *Def*: **learnin** aka. estimation) is the proce f finding the parame s that best fit the data
- *Def*: **hyperparameters** are the parameters of a prior distribution over parameters

Machine Learning

- *Def*: (loosely) a **model** defines the pace over which forms its search
- **arameters** are the ues or structure the learning algorithm e to a hypothesis
- **ning algorithm** defines the d a-driven search over the hyp esis space (i.e. search for goo arameters)
- *Def*: **hyperparameters** are the tunable aspects of the model, that the learning algorithm does *not* select

Model Selection

- Two *very* similar definitions:
  - *Def*: **model selection** is the process by which we choose the "best" model from among a set of candidates
  - *Def*: **hyperparameter optimization** is the process by which we choose the "best" hyperparameters from among a set of candidates **(could be called a special case of model selection)**

- **Both** assume access to a function capable of measuring the quality of a model

- **Both** are typically done "outside" the main training algorithm --- typically training is treated as a black box

Example of Hyperparameter Opt.

Chalkboard:

- Special cases of k-Nearest Neighbors
- Choosing k with validation data
- Choosing k with cross-validation

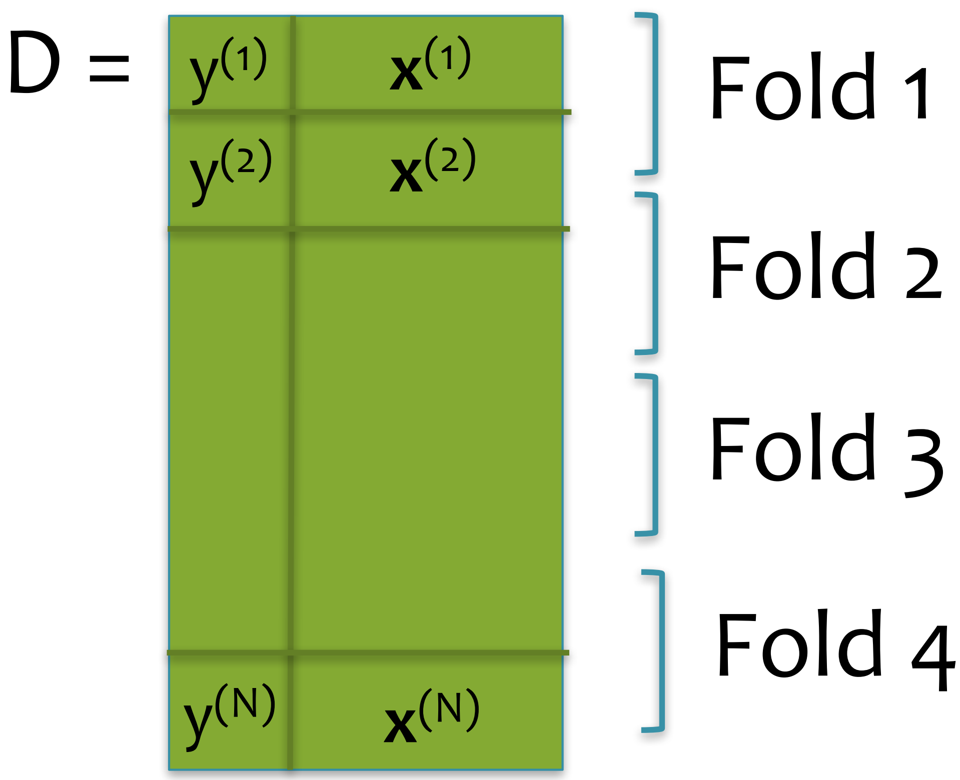
Cross-Validation

Cross validation is a method of estimating loss on held out data

Input: training data, learning algorithm, loss function (e.g. 0/1 error)

Output: an estimate of loss function on held-out data

Key idea: rather than just a single "validation" set, use many!
(Error is more stable. Slower computation.)



Algorithm:

Divide data into folds (e.g. 4)

1. Train on folds {1,2,3} and predict on {4}
2. Train on folds {1,2,4} and predict on {3}
3. Train on folds {1,3,4} and predict on {2}
4. Train on folds {2,3,4} and predict on {1}

Concatenate all the predictions and evaluate loss (*almost* equivalent to averaging loss over the folds)

Model Selection

WARNING (again):

- This section is only scratching the surface!
- Lots of methods for hyperparameter optimization: (to talk about later)
  - Grid search
  - Random search
  - Bayesian optimization
  - Graduate-student descent
  - …

Main Takeaway:

- Model selection / hyperparameter optimization is just another form of learning

Model Selection Learning Objectives

You should be able to…

- Plan an experiment that uses training, validation, and test datasets to predict the performance of a classifier on unseen data (without cheating)
- Explain the difference between (1) training error, (2) validation error, (3) cross-validation error, (4) test error, and (5) true error
- For a given learning technique, identify the model, learning algorithm, parameters, and hyperparamters
- Define "instance-based learning" or "nonparametric methods"
- Select an appropriate algorithm for optimizing (aka. learning) hyperparameters