10-701
Machine Learning

Classification
Where we are

- **Density Estimator**: Inputs → Probability
  - Today

- **Classifier**: Inputs → Predict category
  - Today

- **Regressor**: Inputs → Predict real no.
  - Later
Classification

• Assume we want to teach a computer to distinguish between cats and dogs …
Bayes decision rule

- If we know the conditional probability \( p(x \mid y) \) and class priors \( p(y) \) we can determine the appropriate class by using Bayes rule:

\[
P(y = i \mid x) = \frac{P(x \mid y = i) P(y = i)}{P(x)} = q_i(x)
\]

- We can use \( q_i(x) \) to select the appropriate class.
- We choose class 0 if \( q_0(x) \geq q_1(x) \) and class 1 otherwise.
- This is termed the ‘Bayes decision rule’ and leads to optimal classification.
- However, it is often very hard to compute …
Bayes decision rule

\[ P(y = i \mid x) = \frac{P(x \mid y = i)P(y = i)}{P(x)} = q_i(x) \]

- We can also use the resulting probabilities to determine our confidence in the class assignment by looking at the likelihood ratio:

\[ L(x) = \frac{q_0(x)}{q_1(x)} \]

Also known as likelihood ratio, we will talk more about this later.
Bayes decision rule: Example

Normal Gaussians

\[ \mu_1, \Sigma_1 \quad \mu_2, \Sigma_2 \]

Normal Gaussians

\[ \mu_1, \Sigma_1 \quad \mu_2, \Sigma_2 \]
Bayes error

• For the Bayes decision rule we can calculate the probability of an error.

• This is the probability that we assign a sample to the wrong class, also known as the risk.

• The risk for sample \( x \) is:

\[
R(x) = \min\{P_1(x)P(y=1), P_0(x)P(y=0)\} / P(x)
\]

Risk can be used to determine a ‘reject’ region.
Bayes error

- The probability that we assign a sample to the wrong class, is known as the **risk**

- The risk for sample $x$ is:
  \[ R(x) = \min\{P_1(x)P(Y=1), P_0(x)P(Y=0)\} / P(x) \]

- We can also compute the expected risk (the risk for the entire range of values of $x$):
  \[
  E[r(x)] = \int r(x)p(x)dx \\
  \quad = \int_{L_0}^{L_1} \min\{p_1(x)p(y=1), p_0(x)p(y=0)\} dx \\
  \quad = p(y=0) \int_{L_0}^{L_1} p_0(x) dx + p(y=1) \int_{L_0}^{L_1} p_1(x) dx
  \]

$L_1$ is the region where we assign instances to class 1
The risk value we computed assumes that both errors (assigning instances of class 1 to class 0 and vice versa) are equally harmful.

However, this is not always the case.

Why?

In general our goal is to minimize loss, often defined by a loss function: $L_{0,1}(x)$ which is the penalty we pay when assigning instances of class 0 to class 1

$$E[L] = L_{0,1} p(y = 0) \int_{L_1} p_0(x) dx + L_{1,0} p(y = 1) \int_{L_0} p_1(x) dx$$
Types of classifiers

• We can divide the large variety of classification approaches into roughly two main types

1. Instance based classifiers
   - Use observation directly (no models)
   - e.g. K nearest neighbors

2. Generative:
   - build a generative statistical model
   - e.g., Naïve Bayes

3. Discriminative
   - directly estimate a decision rule/boundary
   - e.g., decision tree
Assume we want to teach a computer to distinguish between cats and dogs …

Several steps:
1. feature transformation
2. Model / classifier specification
3. Model / classifier estimation (with regularization)
4. feature selection
Classification

- Assume we want to teach a computer to distinguish between cats and dogs …

Several steps:
1. feature transformation
2. Model / classifier specification
3. Model / classifier estimation (with regularization)
4. feature selection

How do we encode the picture? A collection of pixels? Do we use the entire image or a subset? …
Classification

- Assume we want to teach a computer to distinguish between cats and dogs …

Several steps:
1. feature transformation
2. Model / classifier specification
3. Model / classifier estimation (with regularization)
4. feature selection

What type of classifier should we use?
Classification

- Assume we want to teach a computer to distinguish between cats and dogs …

Several steps:
1. feature transformation
2. Model / classifier specification
3. Model / classifier estimation (with regularization)
4. feature selection

How do we learn the parameters of our classifier? Do we have enough examples to learn a good model?
Classification

- Assume we want to teach a computer to distinguish between cats and dogs …

Several steps:
1. feature transformation
2. Model / classifier specification
3. Model / classifier estimation (with regularization)
4. feature selection

Do we really need all the features? Can we use a smaller number and still achieve the same (or better) results?
Supervised learning

- Classification is one of the key components of ‘supervised learning’
- Unlike other learning paradigms, in supervised learning the teacher (us) provides the algorithm with the solutions to some of the instances and the goal is to generalize so that a model / method can be used to determine the labels of the unobserved samples.
Types of classifiers

- We can divide the large variety of classification approaches into roughly two main types

1. Instance based classifiers
   - Use observation directly (no models)
   - e.g. K nearest neighbors

2. Generative:
   - build a generative statistical model
   - e.g., Bayesian networks

3. Discriminative
   - directly estimate a decision rule/boundary
   - e.g., decision tree
K nearest neighbors
K nearest neighbors (KNN)

- A simple, yet surprisingly efficient algorithm
- Requires the definition of a distance function or similarity measures between samples
- Select the class based on the majority vote in the k closest points
K nearest neighbors (KNN)

- Need to determine an appropriate value for $k$
- What happens if we choose $k=1$?
- What if $k=3$?
K nearest neighbors (KNN)

- Choice of k influences the ‘smoothness’ of the resulting classifier.
- In that sense it is similar to a kernel methods (discussed later in the course).
- However, the smoothness of the function is determined by the actual distribution of the data \( p(x) \) and not by a predefined parameter.
The effect of increasing $k$
The effect of increasing k

We will be using Euclidian distance to determine what are the k nearest neighbors:

\[ d(x, x') = \sqrt{\sum_{i} (x_i - x'_i)^2} \]
KNN with $k=1$
KNN with k=3

Ties are broken using the order:
Red, Green, Blue
KNN with $k=5$

Ties are broken using the order:

Red, Green, Blue
Comparisons of different k’s

K = 1

K = 3

K = 5
A probabilistic interpretation of KNN

• The decision rule of KNN can be viewed using a probabilistic interpretation
• What KNN is trying to do is approximate the Bayes decision rule on a subset of the data
• To do that we need to compute certain properties including the conditional probability of the data given the class \( p(x|y) \), the prior probability of each class \( p(y) \) and the marginal probability of the data \( p(x) \)
• These properties would be computed for some small region around our sample and the size of that region will be dependent on the distribution of the test samples*

* Remember this idea. We will return to it when discussing kernel functions
Computing probabilities for KNN

• Let $V$ be the volume of the $m$ dimensional ball around $z$ containing the $k$ nearest neighbors for $z$ (where $m$ is the number of features).
• Then we can write

$$p(x)V = P = \frac{K}{N} \quad p(x) = \frac{K}{NV} \quad p(x \mid y = 1) = \frac{K_1}{N_1V} \quad p(y = 1) = \frac{N_1}{N}$$

• Using Bayes rule we get:

$$p(y = 1 \mid z) = \frac{p(z \mid y = 1)p(y = 1)}{p(z)} = \frac{K_1}{K}$$

$z$ – new data point to classify
$V$ – selected ball
$P$ – probability that a random point is in $V$
$N$ – total number of samples
$K$ – number of nearest neighbors
$N_1$ – total number of samples from class 1
$K_1$ – number of samples from class 1 in $K$
Computing probabilities for KNN

Using Bayes rule we get:

\[ p(y = 1 \mid z) = \frac{p(z \mid y = 1) p(y = 1)}{p(z)} = \frac{K_1}{K} \]

Using Bayes decision rule we will chose the class with the highest probability, which in this case is the class with the highest number of samples in K.
Important points

• Optimal decision using Bayes rule
• Types of classifiers
• Effect of values of k on knn classifiers
• Probabilistic interpretation of knn