|  |  |  |
| --- | --- | --- |
| **Supervised Learning** | | |
| **Linear**  **regression** | Step 1. Hypothesis:  Step 2. Cost  Step 3: Gradients | A close up of a map  Description automatically generated |
| A close up of a map  Description automatically generated |
| **Logistic**  **regression** | Step 1. Hypothesis:  Step 2. Cost  Step 3. Gradients: | A picture containing food  Description automatically generated |
| Support  vector  machines  (SVM) | Cost | Related image |
| SVM with  Gaussian  Kernel | Step 1. Hypothesis *Given x, compute features , parameters*  *Predict “y=1” if*  Step 2. Training  *min* | A picture containing text, map  Description automatically generated |
| **Neural network**  *Classification* | Step 1. Randomly initialize weights  Initialize parameters (i.e. )  Step 2. Forward propagation      Step 3. Cost function  Step 4. Backpropagation to compute partial derivatives  “error” of node in layer .   |  |  | | --- | --- | |  |  | |  |   Step 5. Use gradient checking to compare computed using backpropagation vs. using numerical estimate of gradient of .  Step 6. Use gradient descent or advanced optimization method with backpropagation to try to minimize as a function of parameters .  result = minimize(cost\_func, initial\_nn\_params, method='CG', jac=grad\_func,  options={'disp': True, 'maxiter': 50.0})  nn\_params = result.x  Jcost = result.fun | A picture containing building, window, drawing  Description automatically generated |

|  |  |  |
| --- | --- | --- |
| **Unsupervised Learning** | | |
| K-means | Step 1. Centroids  denotes the index of cluster centroids closet to x(i)  Step 2. Means  denotes the average(mean) of points assigned to cluster k  Step 3. Cost function | K=3  A close up of a map  Description automatically generated |
| **Principal**  **Component**  **Analysis**  (PCA) | Step1. Feature scaling (Mean normalization)  Mean:  Standard deviation:  Mean normalize:  Step 2. Calculate U, S, V.    U, S, V = numpy.linalg.svd(sigma)  Ureduce = U[:, 0:K].T  Z = Ureduce\*X = X\_norm \* U[:, 0:K]  X\_approximate = X\_recovered = Z \* U[:, 0:K].T  Step 3. Pick the smallest value of k,  99% of variance retained | PC Scatterplot |
| **Dimensionality**  **Reduction**  *Data compression* |
| **Anomaly**  **Detection**  *Fraud detection,*  *Manufacturing,*  *Monitoring computers in a data center* | **Gaussian (Normal) distribution**  **Mean:**  **Variance:**  **Probability:** | A close up of a map  Description automatically generated |
| 1. Original model | Step 1. Choose feature  Training set:  Density estimation: **1, 2, …, n**  Choose features that might be indicative of anomalous examples.  Step 2. Fit parameters  Step 3. Given new example , compute  Probability | A close up of a piece of paper  Description automatically generated |
| Multivariate Gaussian | Step 1. Choose feature  Training set:  Density estimation: 1, 2, …, n  Step 2. Fit parameters  Parameters: (covariance matrix)  Mean:  Variance:  Step 3. Given new example , compute  Diagonal Sigma:  Probability: | A close up of text on a white background  Description automatically generated |