linear functions of the form $h(x) = a * x + b$, then this step is equivalent to finding the parameters a and b that best match the training data, according to some linear regression estimator, e.g. the least squares estimator.

3. Now we have a set of fitted functions $h_i \in \mathcal{H}$, and the next step is to choose the best one. For this, we use the validation set to measure the error rate and choose the function with the lowest error, which we denote h_{best}, and its hypothesis class is denoted by $\mathcal{H}_{best}$.

4. Finally, fit a new function $h_i \in \mathcal{H}_{best}$ using the training set plus the validation set, and measure its error using the test set. Since the test set was not used in the learning process, the resulting error rate can be considered an estimate of the generalization error.

Depending on the amount of labeled data available, and the complexity of the underlying structure in the data, it may be necessary to repeat this process with different divisions of the labeled data into the respective training set, validation set, and set. The standard technique for this repetition process is known as *cross-validation*. Due to space limitations, we will not cover cross-validation in detail, but it was employed in [P3] and [P4].

3.4 Unsupervised Learning

Unsupervised learning is, in many ways, quite similar to supervised learning, except that there are no labeled data. In other words, there are only input data, and the goal is to learn something about the structure or patterns in the input data. In this way, unsupervised learning is very similar to traditional statistical methods, where the goal is to infer a statistical model from a set of data. Many unsupervised learning methods, such as density estimation, come straight from statistics. Others differ only in the name or some other superficial characteristics. Especially in recent years, there are large overlaps between statistics research and unsupervised learning research[9].

Consider again the data presented in Table 3.1 and Figure 3.1. Suppose Mary had not gone to the trouble of labeling the data with the actual mobility context associated with each data sample. We would have then only a two-dimensional dataset of input data, and we could make a similar plot as Figure 3.1, except the legend would be missing and we would also not have the information necessary to

[9]This is also true to a certain extent in supervised learning, but the similarity is more striking in unsupervised learning.

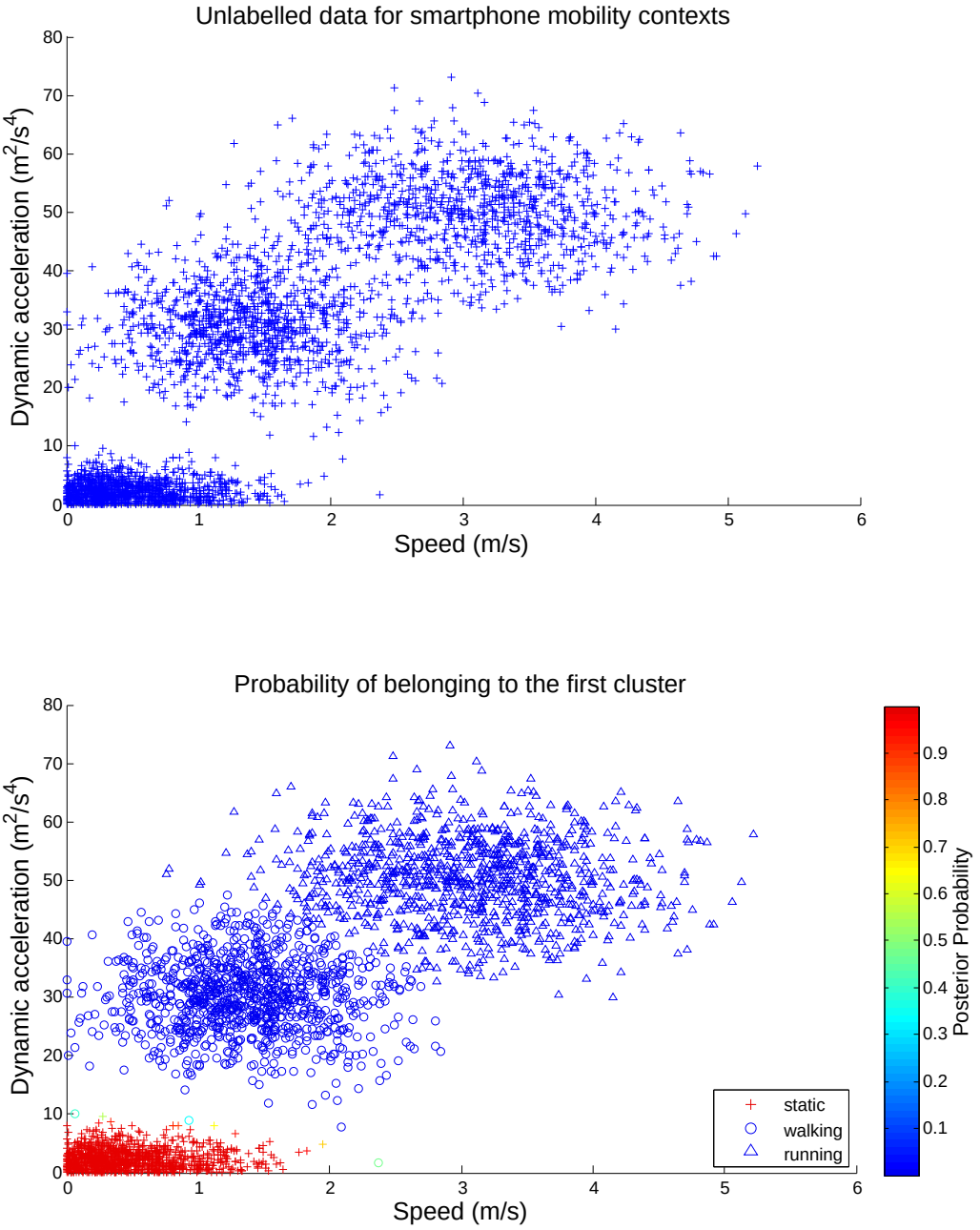

Figure 3.2 *This is another caption.*

label the samples with different colors as in Figure 3.1. The top part of Figure 3.2 shows such a plot.

One unsupervised learning task would be identify different clusters or groups present in the data. Depending on the data and the application, it may or may not be apparent how many clusters are inherently present in the data, so the number of clusters may also be a parameter to determine as part of the unsupervised learning task. As is the case in supervised learning, there are a plethora of different unsupervised learning algorithms available in the literature that perform clustering. Possibilities include k-means clustering (Hartigan and Wong, 1979), OPTICS (Ankherst et al., 1999), and the expectation-maximization (EM) algorithm (Dempster et al., 1977). In particular, the EM algorithm has its roots in statistitcs and can fit observed data to an arbitrary statistical model.

To provide an example of clustering, we used the EM algorithm to fit a Gaussian mixture model (GMM) to the data that we have previously seen in the top half of Figure 3.2. A GMM is of the form:

$$p(\mathbf{x}|\Theta) = \sum_{k=1}^{K} \pi_k \phi_k(\mathbf{x}; \boldsymbol{\theta}_k) \tag{3.4}$$

where $\mathbf{x}$ is a random vector, K is the number of components in the mixture model, $\phi_k(\mathbf{x}; \boldsymbol{\theta}_k)$ are normal distributions with parameters $\boldsymbol{\theta}_k = (\boldsymbol{\mu_k}, \boldsymbol{\Sigma_k})$, π_k are mixing weights satisfying $\pi_1 + ... + \pi_K = 1, \pi_k \geq 0$, and $\Theta = \{\pi_1, ..., \pi_K, \theta_1, ..., \theta_K\}$ is the complete set of model parameters[10].

The EM algorithm itself is a widely-used iterative algorithm used to find the maximum likelihood estimate (MLE) of the model parameters (which we denote with Θ as above) for an underlying distribution $p(\mathbf{x}|\Theta)$ used to model a given dataset, which we denote as $\mathcal{D} = (\mathbf{x}_1, ..., \mathbf{x}_N)$ (Bilmes, 1998). The MLE is obtained by maximizing a function Q equal to the expected value of the loglikelihood $\mathcal{L}(\Theta|\mathcal{D}, \mathcal{Y})$, given the observed data $\mathcal{D}$ and the current parameter estimates $\Theta^{(i-1)}$:

$$Q(\Theta, \Theta^{(i-1)}) = E[\log \mathcal{L}(\Theta|\mathcal{D}, \mathcal{Y})|\mathcal{D}, \Theta^{(i-1)})] = E[\log p(\mathcal{D}, \mathcal{Y}|\Theta)|\mathcal{D}, \Theta^{(i-1)})] \tag{3.5}$$

where $\mathcal{Y} = (y_1, ..., y_N)$ is a vector of latent variables that indicate to which component of the GMM a given data sample $\mathbf{x}_j$ belongs. The latent variables can be expressed in various ways, but perhaps the simplest expression is that $y_j = k$ when $\mathbf{x}_j$ belongs to component k. In the above equation i indexes the current iteration interval of the algorithm, so $\Theta^{(i-1)}$ represents the parameter estimate from the pre-

[10]The notation used for the GMM is similar but not identical to that given in (Bilmes, 1998).

vious iteration (or the initial estimate, if $i = 1$).

Before applying the EM algorithm to find the parameters Θ of a GMM, one must decide on the number of components K to incorporate into the GMM. As we shall see, each component k in the model will correspond to a cluster in the final clustering result; thus, this step is, in practice, the same as determining the number of clusters, and we can consider K to be a hyperparameter in the estimation problem.

Various methods can be used to determine the best value for K. For low-dimensional data, a practical method is to simply plot the data (as we did in the top half of Figure 3.2) and try to visualize the inherent number of clusters. For high-dimensional data ($D > 3$), this simple approach is not necessarily adequate, nor does it support the goal of automation described earlier. Therefore, a more sophisticated, systematic approach may be preferred, such as the one described in (Vlassis and Likas, 2002). For this example, we assume in the interest of space that the choice of K is already clear, and for these data $K = 3$ seems to be a reasonable choice.

The next step is simply to apply the EM algorithm to determine the parameters Θ of our three-component GMM. A detailed description of the EM algorithm is beyond the scope of this thesis, but here we provide a brief overview.

First, EM requires an initial estimate of Θ, and various initialization techniques to provide sensible initial estimates can be found in the literature . A simple approach is to use the given dataset $\mathcal{D}$: e.g. select K random samples to initialize $\boldsymbol{\mu_k}$ and use the covariance matrix of $\mathcal{D}$ for each of the initial K covariance matrices Σ_k (Smyth, 2015).

After initialization, the algorithm then alternates between computing an expectation function (known as the E-step) and finding the parameters Θ that maximize this function (known as the M-step). At each E-step, the algorithm calculates a new $Q(\Theta, \Theta^{(i-1)})$. In the M-step, an updated estimate $\Theta^{(i)}$ of the parameter set is obtained by maximizing $Q(\Theta, \Theta^{(i-1)})$, according to:

$$\Theta^{(i)} = \arg\max_{\Theta} Q(\Theta, \Theta^{(i-1)}) \tag{3.6}$$

The algorithm terminates when $Q(\Theta, \Theta^{(i-1)})$, evaluated at $\Theta = \Theta^{(i)}$, converges towards a maximum value (i.e. improvement is below some threshold value ϵ).

Finally, once the parameters Θ are estimated, we can determine the posterior probability that a data sample $\mathbf{x}_j$ belongs to a particular component k of the GMM,

according to its so-called "membership weight" (Smyth, 2015):

$$w_j^k = p(y_j = k|\mathbf{x}_j, \Theta) = \frac{p_k(\mathbf{x}_j|\theta_k)\pi_k}{\sum_{m=1}^{K} p_m(\mathbf{x}_j|\theta_m)\pi_m} \tag{3.7}$$

Recall that each component of the GMM corresponds to a cluster, and therefore the membership weight for a given k is the posterior probability that the data sample belongs to cluster k. The bottom half of Figure 3.2 and Figure 3.3 show the posterior probabilities for our example data, corresponding to membership in each of the three clusters. Note that a dividing line between membership in each cluster can be drawn where the posterior probability reaches 0.5.