We have been provided with a subset of Fashion-MNIST dataset containing 28*28 images of "Tshirt" and "Trouser" classes. These images with 784 features were reduced 2 features as follows:

X_1 = The average of all pixel values in the image

X_2 = The standard deviation of all pixel values in the image

Naïve Bayes Model

Following formula can be used for parameter estimation for a 2-D normal distribution:

$$l(\mu, \Sigma|D) = -\frac{N}{2} log|\Sigma| - \frac{1}{2}\sum_{i=1}^{N}(x_i - \mu)^T \Sigma^{-1} (x_i - \mu)$$

Here, μ = mean, Σ = variance, N = input size, D = $\{x_1, x_2, ..., x_n\}$.

Using this formula, μ_{ML}(parameter-wise mean), $\Sigma_{ML}(Covariance\ Matrix)$ parameters for each class can be found as follows:

$$\mu_{ML} = \frac{1}{N}\sum_{i=1}^{N} x_i,$$

$$\Sigma_{ML} = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu_{ML})(x_i - \mu_{ML})^T$$

Here, ML stands for Maximum Likelihood.

For Naïve Bayes, these parameters are assumed to be independent of each other and hence only the diagonal elements from the above d*d covariance matrix are taken into consideration and other elements are assumed to be 0. Hence, just finding the parameter wise variances for each class should be sufficient to predict class using Naïve Bayes model which can be found as follows:

$$\sigma_j^2 = \frac{1}{n}\sum_{j=1}^{n}(x_{ij} - \mu_j)^2$$

Here, σ_j^2 = variance for parameter j, n = number of instances for the given class, μ_j = mean for parameter j.

For the given training data, the parameters were found to be as follows:

μ

	x_1	**x_2**
Y = 0	0.32560777	0.32003609
Y = 1	0.22290531	0.33394171

σ^2

	x_1	x_2
Y = 0	0.01285387	0.00774097
Y = 1	0.00324341	0.00325268

For multi-dimensional input vector, the predicted class for naïve bayes can be given as follows:

$$\hat{y} = argmax_y \left(P(y) \prod_{i=1}^{d} P(x_i|y) \right)$$

We can take log to prevent integer underflow while doing calculations. Hence, the predicted class would be given as:

$$\hat{y} = argmax_y \left[log(P(y)) + \sum_{i=1}^{d} log(P(x_i|y)) \right]$$

Since we have a 2-D input vector, we have d = 2.

For, P(y = 0) and P(y = 1) are the probabilities of these class labels from the training data. Since, both classes have 6000 samples each in the dataset, P(y = 0) = P(y = 1) = 0.5.

For each $P(x_i|y)$, it can be found using the parameters that we calculated earlier and assuming a Gaussian distribution for these parameters. Hence, its log can be found as follows:

$$log\big(P(x_i|y)\big) = log\left[\frac{1}{\sqrt{2\pi\sigma_{ij}{}^2}} e^{-\frac{(x_i-\mu_{ij})^2}{2\sigma_{ij}{}^2}} \right]$$

Using the above methodology, Naïve Bayes provided 83.15% overall accuracy. Its class-wise accuracy was found as follows:

	Accuracy
Y = 0	78.4 %
Y = 1	87.9 %

Logistic Regression Model

The sigmoid function σ(t) is written as:

$$\sigma(t) = \frac{e^t}{1 + e^t}$$

And for logistic regression, the prediction is given as:

$$P(Y = 1|x) = \sigma(w^T x),$$

$$P(Y = 0|x) = 1 - \sigma(w^T x)$$

Here, w is the co-efficient vector with (D+1) dimension and x is the input vector of (D+1) dimension with first value being 1. Log likelihood can be given as:

$$\log(l(w)) = y \log(\sigma(w^T x)) + (1-y) \log(1-\sigma(w^T x))$$

Here, $l(w)$ = likelihood function

And based on this, the loss function can be derived to as follows:

$$\nabla_w log(l(w)) = [y - \sigma(w^T x)]\, x$$

And weight can be updated as:

$$w^{k+1} = w^k + \eta \nabla_w log(l(w)), \text{ for weights other than } w_0$$

$$w_0^{k+1} = w_0^k + \eta[y - \sigma(w^T x)], \text{ for weight } w_0$$

Where η = learning rate.

With $\eta = 1$ and no of iterations = 15000, the weight parameters were found to be as follows:

W_0	-2.54202785586433
W_1	-40.65856276
W_2	41.02319417

And the class-wise accuracy was found as follows:

	Accuracy
Y=0	91.6 %
Y=1	92.6 %

The overall accuracy for Logistic Regression was found to be 92.1%.

As can be seen, Logistic Regression performs considerably better as compared to Naïve Bayes model. This can be attributed to the assumption that Naïve Bayes model makes about parameter independence. While the LR does not make any such assumption and directly learns P(y|x) and hence it performs better.