ML Homework 2 Solutions

Ahuva Bechhofer

March 7, 2023

1 Designing socially aware classifiers

1.1 Part 0 - i

Since one attribute can be linked to other attributes, simply removing one does not take into account the other links and does not achieve fairness. For example if a school admission bored decides to take out / disregard income so admission is fair would not achieve fairness since income is related to zip code, the school the student previously attended, extra curricular activities and many more attributes schools do look at.

1.2 Part 1 - ii

Proof: Right $\rightarrow$ Left :
starting with this assumption $P_0[\hat{Y} = 1] = P_1[\hat{Y} = 1]$

$P_0[\hat{Y} = 1] = P[\hat{Y} = 1|a = 0], P_1[\hat{Y} = 1] = P[\hat{Y} = 1|a = 1] \rightarrow$
$P[\hat{Y} = 1] = P[\hat{Y} = 1|a = 0] * P[a = 0] + P[\hat{Y} = 1|a = 1] * P[a = 1]$
From our assumption we can substitute $P[\hat{Y} = 1|a = 0]$ for $P[\hat{Y} = 1|a = 1] \rightarrow$
$P[\hat{Y} = 1] = P[\hat{Y} = 1|a = 0] * [P[a = 0] + P[a = 1]]$
since $P[a = 0] + P[a = 1] = 1 \rightarrow$
$P[\hat{Y} = 1] = P[\hat{Y} = 1|a = 0] = P[\hat{Y} = 1|a = 1] \rightarrow P_a[\hat{Y} = 1]$ where $a \in 0, 1$

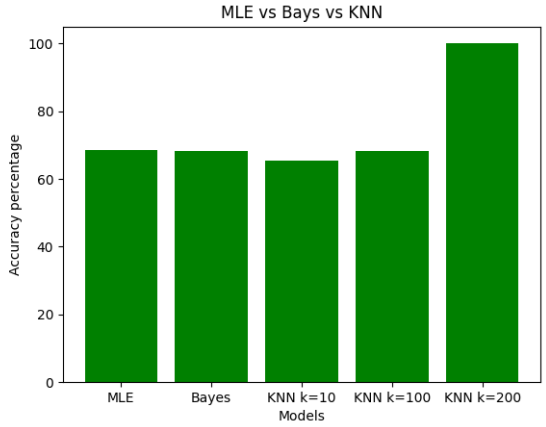
Left $\rightarrow$ Right :
starting with this assumption $P_a[\hat{Y} = 1] = P[\hat{Y} = 1]$
since $a \in 0, 1$ we know $P_1[\hat{Y} = 1] = P[\hat{Y} = 1]$ and $P_0[\hat{Y} = 1] = P[\hat{Y} = 1]$
Therefore $P_0[\hat{Y} = 1] = P_1[\hat{Y} = 1]$

1.3 Part 1 - iii

$P[\hat{Y} = r] = P_a[\hat{Y} = r], \forall a \in \mathbb{N}, \forall r \in \mathbb{R}$

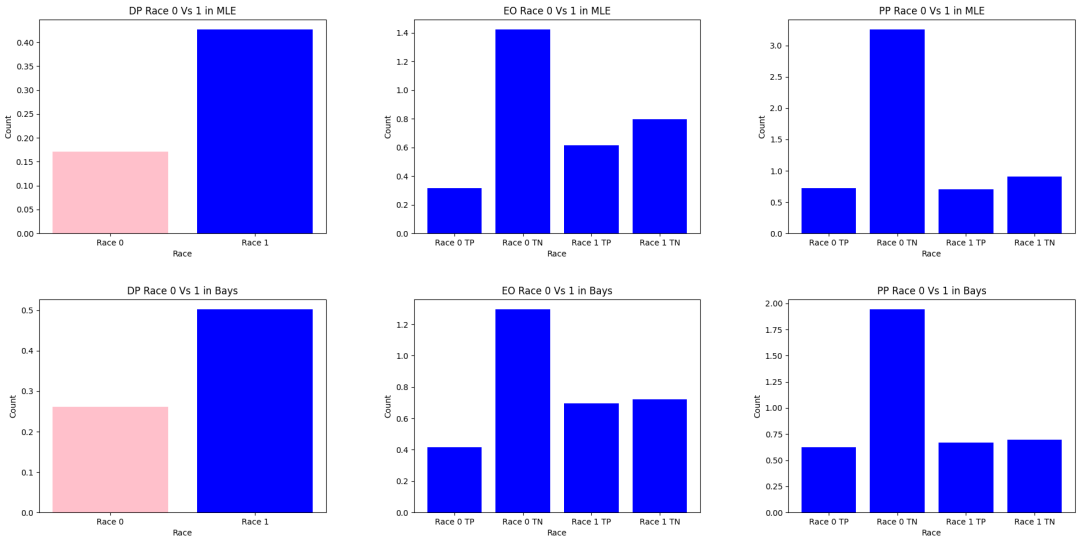
1.4 Part 2 - iv

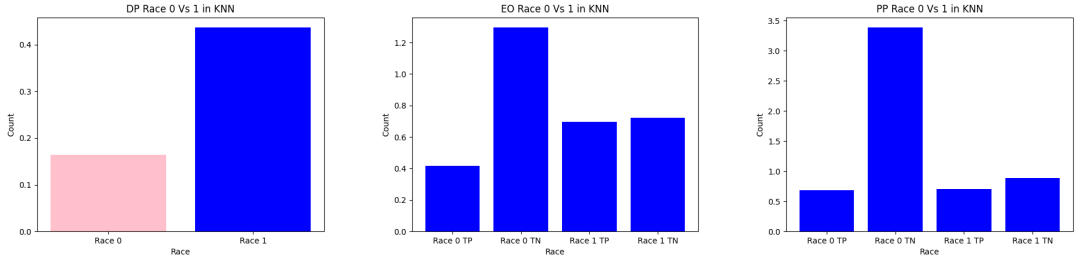
1.5 v



It is common knowledge that the higher the training sample size the better the classifier can predict since it has more previous data to learn from. Our training sample size was not small which is why our accuracies are between 60-70 percent but if we would of been able to provide the models with more training data they should preform better and have higher accuracy.

1.6 vi





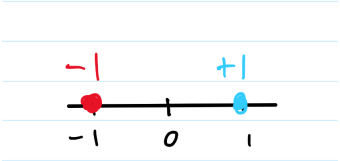
for DP the Bayes classifier was the most fair, for EO Bayes and KNN seem to have the same fairness which is still better than the MLE and lastly for PP Bayes sees to be the most fair.
Since for MLE we just attain the mean and variance from the training data and base the predictions off of that(in an approximate distribution), if the data was slightly skewed in towards one race, the predictions will also be more heavily skewed. For the Bayes since it is a probability model and we are looking across a wide set of features it is less sensitive to slightly biased data and as we see it was the most "fair" model out of them all. For KNN it is also very sensitive to the sample training data since it is looking at the K nearest neighbors if there are more data points from a specific race it is much more likely to predict that race and thus be biased towards that.

1.7 vii

Demographic parity: For example, if two types of races apply to Columbia University, demographic parity is achieved if the percentage of race 1 admitted is the same as the percentage of race admitted, regardless of whether one group is on average more qualified than the other.
credit: https://developers.google.com/machine-learning/glossary/fairness: :text=For

2 Data dependent perceptron mistake bound

2.1 i



The error for this example would be $(\frac{R}{\gamma})^2 = (\frac{1}{1})^2 = 1$
If we run the perceptron algorithm at the first iteration $sign(w \cdot x) \neq y \rightarrow$ means we have a mistake, so we have $sign(0 \cdot 1) = 0 \neq 1 \rightarrow$ mistake, update weights $w = 1 \cdot 1 + 0 = 1$ on the second iteration $sign(1 \cdot -1) = -1$ since there

is no mistake we are done. Since we had only exactly 1 mistake we proved the tightness of the bound.

2.2 ii

To start let's prove the hint $\|w_T\|^2 \leq \sum_{t=1}^{T} \|x_{it}\|^2$
We can write:
$\|w_T\|^2 = \|w^{(t-1)} + y\vec{x}_t\|^2 = \|w^{(t-1)}\|^2 + 2y(\vec{w}^{(t-1)} * \vec{x}) + \|y\vec{x}\|^2$
Since we know the middle term is negative we get $\|w_T\|^2 \leq \|w^{(t-1)}\|^2 + \|y\vec{x}\|^2$ since we know y is either 1 or -1 we can just take it out as it does not affect the length.
$\|w_T\|^2 \leq \|w^{(t-1)}\|^2 + \|\vec{x}\|^2$
Since $\|w^{(t-1)}\|^2 \leq \|w^{(t-2)}\|^2 + \|\vec{x}_{t-1}\|^2$ we can substitute that into the previous step so we get: $\|w_T\|^2 \leq \|w^{(t-2)}\|^2 + \|\vec{x}_{t-1}\|^2 + \|\vec{x}_t\|^2$ and keep doing this until $\|w_T\|^2 \leq \|w^{(0)}\|^2 + \|\vec{x}_0\|^2 + \|\vec{x}_1\|^2 + \|\vec{x}_2\|^2$...
Since we know $w^{(0)} = 0$ we get :
$\|w_T\|^2 \leq \sum_{t=1}^{T} \|\vec{x}_{it}\|^2$
Now we can break up vector $\vec{x}_{it}$ into it's components so we get:
$\|\vec{x}_{it}\|^2 = \|Px_{it} + (I-P)x_{it}\|^2$ if we multiply this out we get
$\|\vec{x}_{it}\|^2 = \|Px_{it}\|^2 + 2Px_{it} \cdot (I-P)x_{it} + \|(I-P)x_{it}\|^2$ since the two components are orthognal their dot product is 0 which is why the middle term dissappears and we get $\|\vec{x}_{it}\|^2 = \|Px_{it}\|^2 + \|(I-P)x_{it}\|^2$ by the definition in the problem $\|(I-P)x_{it}\|^2 \leq \epsilon^2$ and since this is all in a sum that is done T times we can multiply it by T so we get $\|\vec{x}_{it}\|^2 \leq \|Px_{it}\|^2 + \epsilon^2 \times T$ putting it all together we get: $\|w_T\|^2 \leq \|Px_{it}\|^2 + \epsilon^2 \times T$

2.3 iii

To start let's prove the hint: $w_T \cdot w^* = \sum_{t=1}^{T} y_{it}x_{it} \cdot w^*$
we can rewrite $w_T \cdot w^* = (w^{(t-1)} + yx_{it}) \cdot w^* = w^* \cdot w^{(t-1)} + yx_{it} \cdot w^*$
now for the first term we can do what we did in the last proof where we replace $w^{(t-1)} = w^{(t-2)} + yx_{t-1} \cdot w^*$ and we do this until we get
$w_T \cdot w^* = w^{*0} + y_1x_1 \cdot w^* + y_2x_2 \cdot w^* + ..$
Since $w^{(0)} = 0$ we get : $w_T \cdot w^* = \sum_{t=1}^{T} y_{it}x_{it} \cdot w^*$
Now if we square both sides, for the right side we get:
$\sum_{t=1}^{T}(y_{it}x_{it} \cdot w^*)^2 = \sum_{t=1}^{T}(y_{it}x_{it} \cdot w^*)(y_{it}x_{it} \cdot w^*)^T + \sum_{t=1}^{T}\sum_{i \neq t}^{T}(y_{it}x_{it} \cdot w^*)(y_{it}x_{it} \cdot w^*)$
For the single summation, we can simplify the inside since y_{it}^2 is just 1,
$w^* \cdot w_T = P$ we are left with $\sum_{t=1}^{T} \|Px_{it}\|^2$ for the inside of the double summation, since we know $y_{it}(tw^*x_{it}) \geq \gamma$ and we have two of these multiplied on the inside we have $\geq \gamma^2$ and since one summation runs T times and one runs up to T except for $i \neq t$ so it runs (T-1) times, so we get all together
$(w_T \cdot w^*)^2 \geq \sum_{t=1}^{T} \|Px_{it}\|^2 + T(T-1)\gamma^2$

2.4 iv

If we look at the previous two parts we can establish that $(w_T \cdot w^*)^2 \leq \|w_T\|^2$ since we have a dot product of $(w_T \cdot w^*) = \|w_T\| * cos(\theta)$ where θ is the angle between w_T and w^*, and since cosine is between -1 to 1 multiplying the length by it will be the same length or smaller to the original length.
Now since $(w_T \cdot w^*)^2 \geq \sum_{t=1}^{T} \|Px_{it}\|^2 + T(T-1)\gamma^2$ and $\|w_T\|^2 \leq \|Px_{it}\|^2 + \epsilon^2 \times T$ and that $(w_T \cdot w^*)^2 \leq \|w_T\|^2$ we can set
$\|Px_{it}\|^2 + \epsilon^2 \times T \geq |Px_{it}\|^2 + T(T-1)\gamma^2 \rightarrow \epsilon^2 \times T \geq T(T-1)\gamma^2$ since both sides had the same term we subtracted it.
Now if we solve for T with algebra we get: $T \leq (\frac{\epsilon}{\gamma})^2 + 1$

3 Constrained optimization

3.1

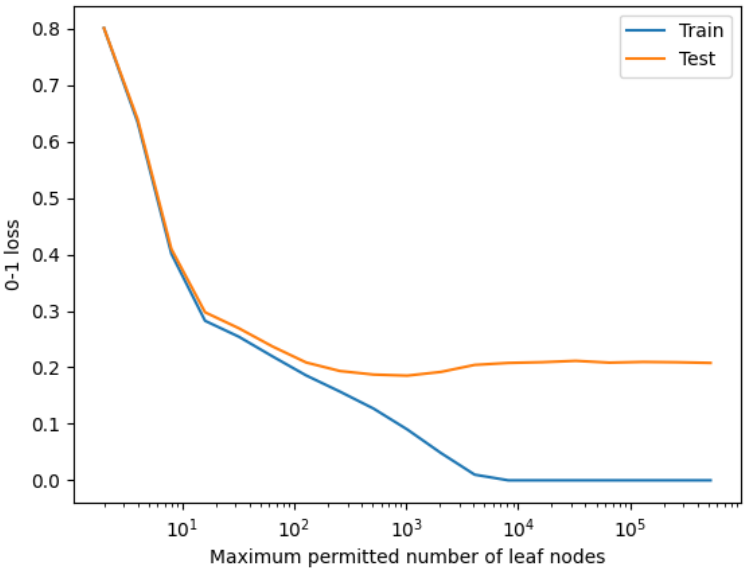
We begin by setting up our lagrange $L(x, \lambda) = \|x - x_a\|^2 + 2\lambda(w \cdot x + x_0)$ we begin by taking the derivative with respect to x, $\frac{dL}{dx} = 2(x - x_a) + 2\lambda w = 0$ we solve for x and get $x = x_a - \lambda w$
we can substitute this into our constraint function $g(x) = w \cdot x + w_0 = 0$ to solve for λ
$w \cdot (x_a - \lambda w) + w_0 = 0 \rightarrow w \cdot x_a - \lambda w \cdot w + w_0 = 0 \rightarrow \lambda \|w\|^2 = w \cdot x_a + w_0 \rightarrow \lambda = \frac{w \cdot x_a + w_0}{\|w\|^2}$ we can now plug this in back into x so we get $x = x_a - (\frac{w \cdot x_a + w_0}{\|w\|^2})w$
since we are minimizing $\|x - x_a\|$ we plug x into this and get
$\|x_a - (\frac{w \cdot x_a + w_0}{\|w\|^2})w - x_a\| = \|(\frac{w \cdot x_a + w_0}{\|w\|^2})w\| \rightarrow \frac{|w \cdot x_a + w_0|}{\|w\|^2} * \|w\| \rightarrow \frac{|w \cdot x_a + w_0|}{\|w\|}$
$\rightarrow \frac{|g(x_a)|}{\|w\|}$

4 Decision Trees, Ensembling and Double Descent

4.1 i

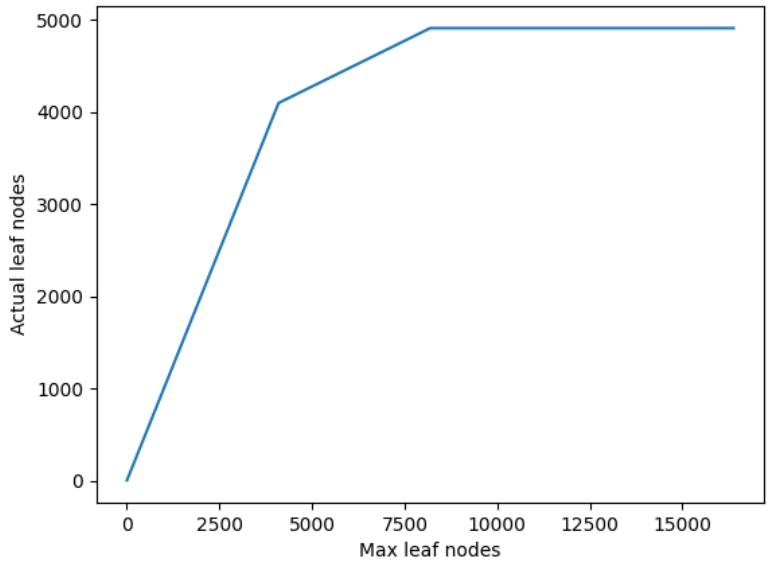
Since the data is blurry images of clothing items that consist of large pixels, it is hard to to make out what image is exactly as opposed to the classical MNIST where it is very easy to classify the number corresponding to each image.
Since a KNN model measures distance between data points, here the data points are pixel values. Since the image is blurry it will make it harder to tell the distance between pixels and in addition since the pixels are large it makes it harder to see the finer details of the image which also affects the classifier as it uses the small details of the images in order to make accurate predictions.

4.2 ii



Min test loss = 0.1875
We can see the model is starting to over fit at a certain point since the test data error is going up and the train data is approaching 0 .

4.3 iii



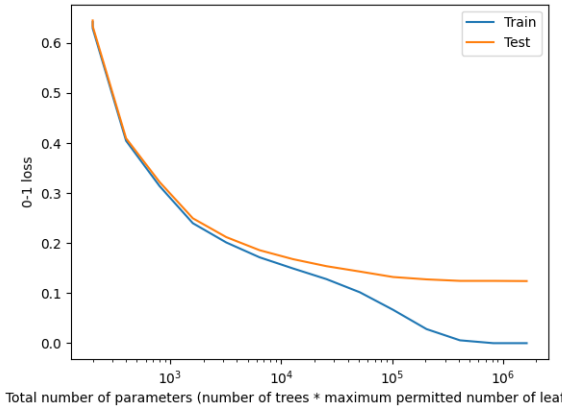
The exact number of leaf nodes used was 4909

4.4 iv

This is a good preventative measure to make sure the model does not over fit. By allowing each tree to focus on a different set of features the model can capture a wider range of patterns and relationships in the data. If individual estimators in the random forest have access to all features, the trees may end up overfitting on specific features which will hurt the trees prediction accuracy.

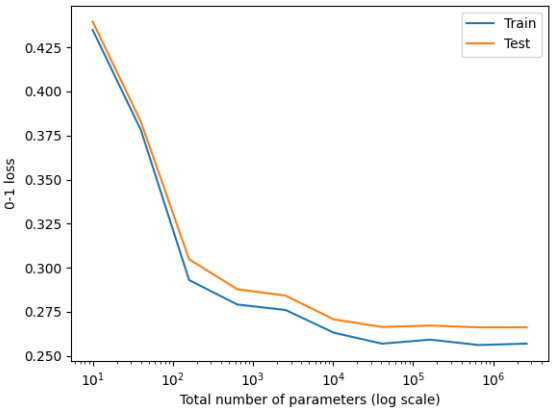
4.5 v

a:



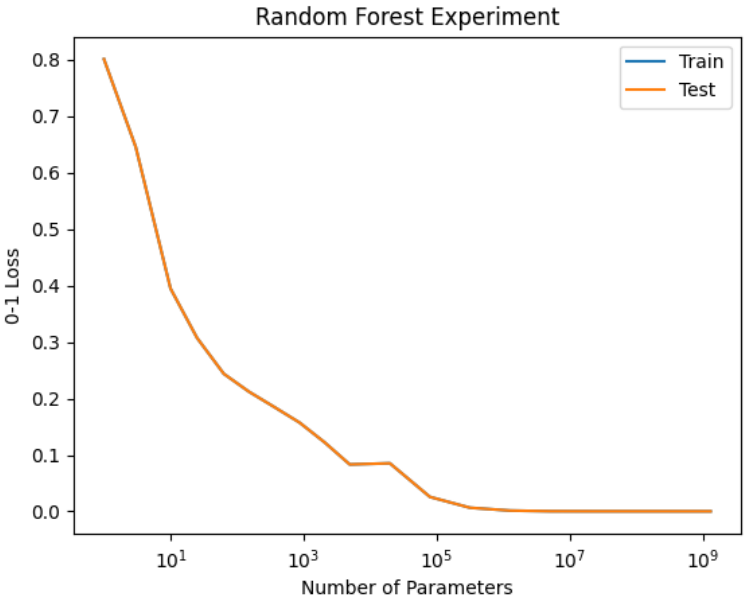
Yes an overfit is still possible since the number or leaves is unbounded, the individual estimators number of leaves can grow past what we calculated beforehand and thus the mdoels overfits. This can be seen in the graph when the training error is approaching 0.

b:



My loss actually increased to 0.2662. I did not get a chance to run this for a similar number of total parameters as the individual tee but I assume the error will decrease past that of an individual tree as now we have a forest that will not overfit so predictions should be more accurate on an ensemble method.

4.6 vi



I had some issues plotting the train curve(I couldn't figure out why)
It seems we get an increase(which should be more U shaped) at the end of phase 1 then a steep decrease at the beginning of phase 2.
I remember learning in the past that not all features should be used for prediction since that leads to overfitting, we want a model to be general enough so it can do well on unseen data, but here the test error initially decreases, then increases, and then decreases again as the number of parameters increases instead of just increasing after a certain point.

Note: a lot of my code for Q4 is commented out since I copy pasted parts reusing the same variable names, so I would comment out the previous parts as to not get conflicts.

Citations:
https://scikitlearn.org/stable/tutorial/basic/tutorial.html
https://numpy.org/doc/stable/reference/generated/numpy.transpose.htm
cikitlearn.org/stable/tutorial/text$_a nalytics/working_w ith_t ext_d atahtmltraining-a-classifier$
$https://scikitlearn.org/stable/modules/generated/sklearn.metrics.zer_o ne_l oss.html$