**Machine Learning by Andrew Ng**

**Week 3 short notes**

Classification problems are also called as logistic regression problems, the existing linear regression model cannot even closely predict the correct outcomes for classification problems hence a more suitable mode is introduced. In this the hypothesis function is a **sigmoid** of the hypothesis function in case of linear regression.

Hypothesis of a Logistic Regression: ***hθ*(*x*)=*****g(θTx)***

***g*(*z*)=1/(1+*e*−*z)***

This restricts the value of outcomes between 0 and 1. The closer the outcome is to 0 or 1 so is it’s probability of it being 0 or 1.

**Decision Boundary** is a line created by the hypothesis function which separates the area where y=0 from the area where y=1.

This cost function should now be devised to obtain the most accurate hypothesis possible, to ensure this out cost function must be strictly concave to get reliable optimal value (global minima).

Cost Function for Logistic Regression**:**

***J(θ)=(-1/m) [ ∑ y(i) .log hθ(x(i)) – (1 -y(i)) log(1- hθ(x(i)))]***

Gradient Descent for logistic regression is obtained in the same way as it is obtained in case of linear regression i.e.

***θj*:= *θj* – *α (* ∂(*J*(*θ*)) / ∂*θj ) = θj* – *α (*1/*m)* ∑ (*hθ* ( *x*(*i*)) − *y*(*i*) ) ⋅ *x*(*i*)*j)***

**Multiclass- Classification** is the case in which we have more than 2 types in which the outcomes can be classified under. In this case, we treat each category as a special case of binary classification where we take the two cases to be belong to the particular category and not belonging to that particular category. This is usually called one-VS- All classification, named after the method followed.

**Overfitting** is an issue in which the hypothesis obtained by adding extra features, so perfectly fits the given dataset that it turns out to be not a very good predictor for any our practical usage.

**Underfitting** is again an issue faced when the features in the hypothesis are too less hence making it a bad predictor.

These problems can be overcome by implementing certain changes.

One way is to reduce or increase the number of features in cases wherever required.

The other is the method of regularization,

In this we keep all the features but reduce the magnitude of these features and works well when we have a lot of slightly useful features.

Regularized Cost Function:

*minθ* (1/2*m)* [ ∑ (*hθ*(*x*(*i*))−*y*(*i*)) 2+ *λ* ∑*θ*2*j* ]

The λ, or lambda, is the **regularization parameter**. It determines how much the costs of our theta parameters are inflated.

Regularized Gradient Descent:

***θ*0:=*θ*0 − *α*** (**1/*m)*** **∑ (*hθ* (*x*(*i*)) − *y*(*i*)) *x*(*i*)0**

***θj*:=*θj* − *α*** **[(** (**1/*m)*** **∑ ( *hθ* (*x*(*i*))−*y*(*i*)) *x*(*i*)*j* )+(*λ/m)θj*]**

**note: Regularization is not applied to the** ***θ*0** and ***hθ*(*x*)=*****θTx***

Regularization of Logistic Regression.

Regularized Cost Function:

**J(θ) = (-1/m) [∑ y(i) .log hθ(x(i)) – (1 - y(i)) log(1- hθ(x(i)))]+(λ/2m)** **∑θ2j**

Regularized Gradient Descent:

***θ*0:=*θ*0 − *α*** (**1/*m)*** **∑ (*hθ* (*x*(*i*)) − *y*(*i*)) *x*(*i*)0**

***θj*:=*θj* − *α*** **[(** (**1/*m)*** **∑ ( *hθ* (*x*(*i*))−*y*(*i*)) *x*(*i*)*j* )+(*λ/m)θj*]**

note: **Regularization is not applied to the** ***θ*0** and ***hθ*(*x*)=*****g(θTx)***