04-03-22

Friday, March 4, 2022 11:42 AM

```
import numpy as np
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import LinearRegression
x = np.linspace(5, 25, 9).reshape(-1, 1)
y = np.array([5, 6, 7, 10, 12, 15, 18, 19, 20])
poly = PolynomialFeatures(degree = 3, interaction_only = True)
X = poly.fit_transform(x)
model = LinearRegression()
model.fit(X, y)
testx = np.array([8.5, 12.4, 21.7, 23.6]).reshape(-1, 1)
transformed_testx = poly.fit_transform(testx)
testy = model.predict(transformed_testx)
print(testy)  # print other vars as well.
```

```
import matplotlib.pyplot as myplot
myplot.scatter(x, y, color = "blue")
myplot.plot(x, model.predict(X), col = "green")
myplot.scatter(testx, testy, color = "red")
```

"from the plots we can observe that if we increase degree, model will overfit."

"also observe changes when interaction-only is changed"

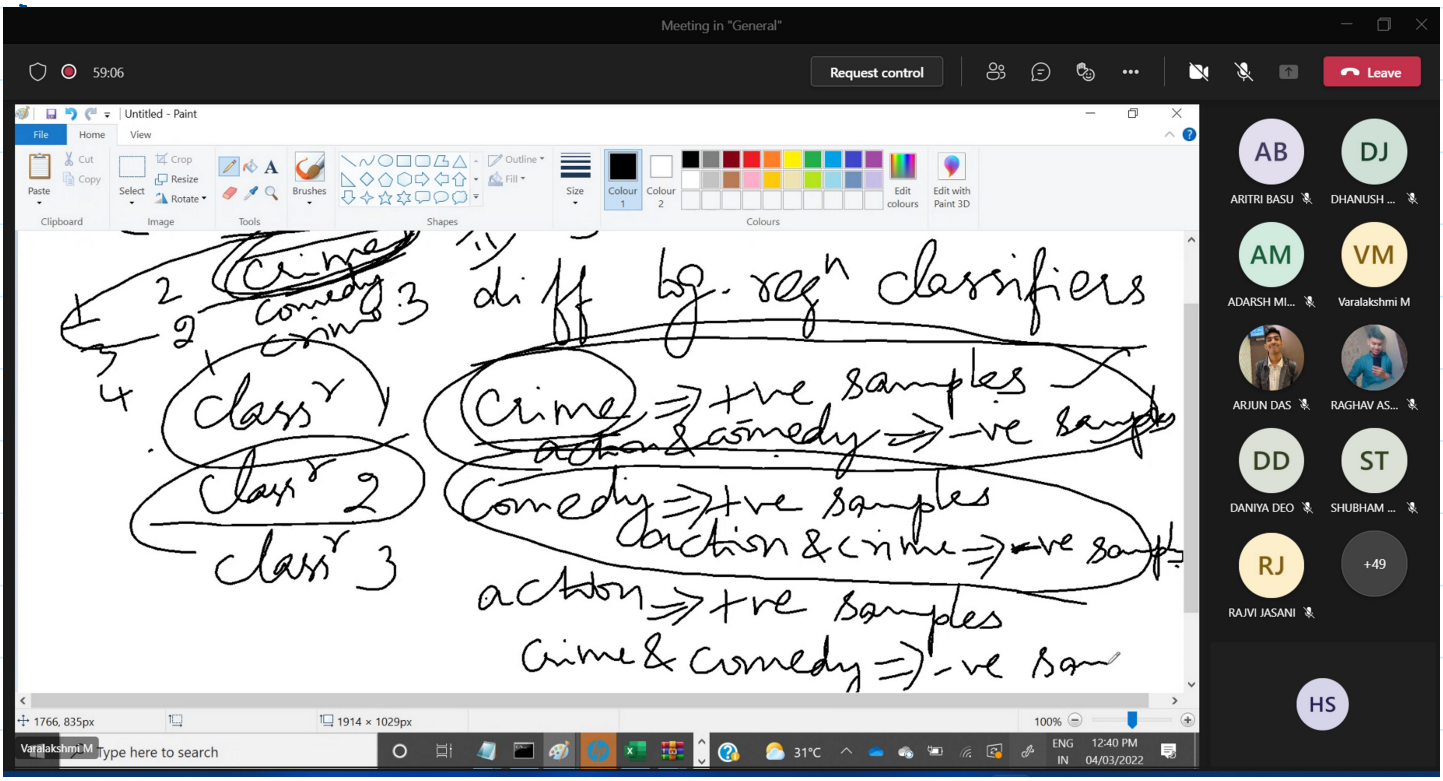
Multinomial classification

transform into binary classification

- one vs rest (OVR/OVA/OAA)
- one vs one (OVO/AVA)

all vs. all

3 diff. logistic regression classifiers



ii) OVO

n classes ⇒ $\frac{n(n-1)}{2}$ classifiers

Logistic regression Reverse of

Logistic Regression Program

↳ inverse of regularization strength
default C = 1

```
x = np.array( [ ] )
y = np.array( [ ] )
from sklearn.linear_model import LogisticRegression
logreg = LogisticRegression( C = 1.5, solver = 'lbfgs',
                             multi_class = 'ovr' )
X = x.reshape (-1, 1)
logreg.fit( X, y)
predy = logreg.predict (X)
print (logreg.predict ([ [190] ]) )
```

predict
vs.
predict_proba

```
myX=np.arange(180,230,0.5)
probabilities=[]
for i in myX:
    ploss,pwin=logreg.predict_proba([[i]])[0]
    probabilities.append(pwin)
myplot.scatter(myX,probabilities)
myplot.scatter(x,y)
```