|  |
| --- |
|  |
|  |

**Phase-1 – Mushroom Data**

Basic Exploration – Slide3:

|  |
| --- |
| Number of FEATURES: 22 (all nominally valued) |
| Number of DATA POINTS: 8,124 |

Class #



|  |
| --- |
| Class Distribution: |
| ‐‐ edible: 4208 (51.8%) |
| ‐‐ poisonous: 3916 (48.2%) |
| ‐‐ total: 8124 instances |

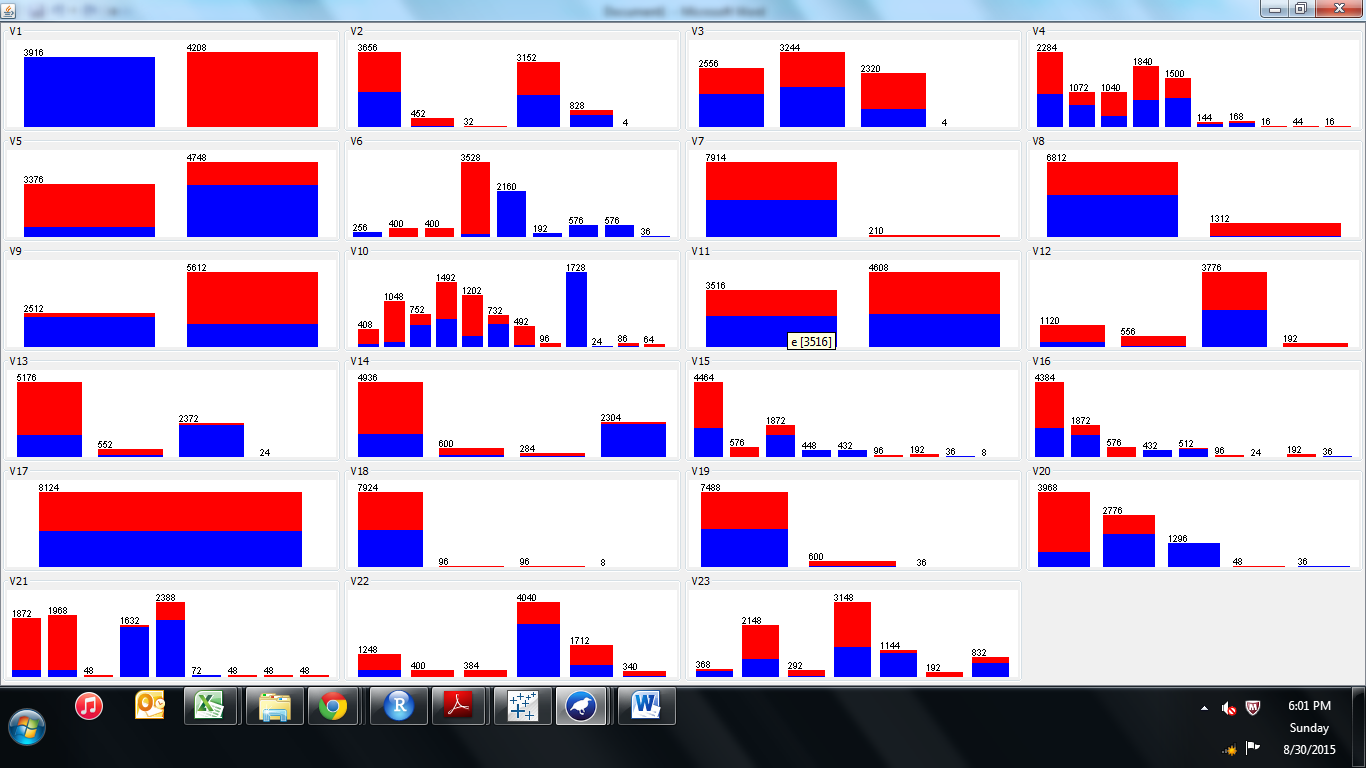
Missing Attribute Values: 2480 of them (denoted by "?"), all for

attribute #11.

Prior Probabilities of each class:

|  |  |
| --- | --- |
| **Class** | **Prob.** |
| P | 0.480919163 |
| E | 0.519080837 |

Histogram of for each feature:



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Attribute Information: (classes: edible=e | poisonous=p) |  |  |  |  |
| 1. cap‐shape: bell=b | conical=c | convex=x | flat=f |  |  |
| knobbed=k | sunken=s |  |  |  |  |
| 2. cap‐surface: fibrous=f | grooves=g | scaly=y | smooth=s |  |  |
| 3. cap‐color: brown=n | buff=b | cinnamon=c | gray=g | green=r |  |
| pink=p | purple=u | red=e | white=w | yellow=y |  |
| 4. bruises?: bruises=t | no=f |  |  |  |  |
| 5. odor: almond=a | anise=l | creosote=c | fishy=y | foul=f |  |
| musty=m | none=n | pungent=p | spicy=s |  |  |
| 6. gill‐attachment: attached=a | descending=d | free=f | notched=n |  |  |
| 7. gill‐spacing: close=c | crowded=w | distant=d |  |  |  |
| 8. gill‐size: broad=b | narrow=n |  |  |  |  |
| 9. gill‐color: black=k | brown=n | buff=b | chocolate=h | gray=g |  |
| green=r | orange=o | pink=p | purple=u | red=e |  |
| white=w | yellow=y |  |  |  |  |
| 10. stalk‐shape: enlarging=e | tapering=t |  |  |  |  |
| 11. stalk‐root: bulbous=b | club=c | cup=u | equal=e |  |  |
| rhizomorphs=z | rooted=r | missing=? |  |  |  |
| 12. stalk‐surface‐above‐ring: fibrous=f | scaly=y | silky=k | smooth=s |  |  |
| 13. stalk‐surface‐below‐ring: fibrous=f | scaly=y | silky=k | smooth=s |  |  |
| 14. stalk‐color‐above‐ring: brown=n | buff=b | cinnamon=c | gray=g | orange=o |  |
| pink=p | red=e | white=w | yellow=y |  |  |
| 15. stalk‐color‐below‐ring: brown=n | buff=b | cinnamon=c | gray=g | orange=o |  |
| pink=p | red=e | white=w | yellow=y |  |  |
| 16. veil‐type: partial=p | universal=u |  |  |  |  |
| 17. veil‐color: brown=n | orange=o | white=w | yellow=y |  |  |
| 18. ring‐number: none=n | one=o | two=t |  |  |  |
| 19. ring‐type: cobwebby=c | evanescent=e | flaring=f | large=l |  |  |
| none=n | pendant=p | sheathing=s | zone=z |  |  |
| 20. spore‐print‐color: black=k | brown=n | buff=b | chocolate=h | green=r |  |
| orange=o | purple=u | white=w | yellow=y |  |  |
| 21. population: abundant=a | clustered=c | numerous=n |  |  |  |
| scattered=s | several=v | solitary=y |  |  |  |
| 22. habitat: grasses=g | leaves=l | meadows=m | paths=p |  |  |
| urban=u | waste=w | woods=d |  |  |  |

**Information Gain of each FEATURE**

Attribute Evaluator (supervised, Class (nominal): 1 V1):

Information Gain Ranking Filter



**Mushroom – Naive Bayes classifier Slide -4**

1. Random sampling in XL miner for all vars. 60 % (4874) train and 40 % (3250) Validation (test) data



Train Class distribution: Test Class distribution:

|  |  |  |
| --- | --- | --- |
| Class | # Cases | % |
| P | 2375 | 48.73% |
| E | 2499 | 51.27% |
| Overall | 4874 | 100.00% |

|  |  |  |
| --- | --- | --- |
| Class | # Cases | % |
| p | 1541 | 47.42% |
| e | 1709 | 52.58% |
| Overall | 3250 | 100.00% |

ii)

Build a Naïve Bayes classifier using the training data


iii)

Evaluate the Naïve Bayes classifier on test data.

K=All (all features)





Now build the NB classifier with top k features based on the Information gain (k = 5, k = 10, k = all)

K=5





K=10





**Ans:**

**So K = all has best accuracy of 0.34 % error on validation data**

**K=10 has 0.62 % and k=5 has 0.68 % error on validation data**

**Mushroom – Decision Tree classifier – Slide 5**

**Decision Tree run in R**

1. **Minimum complexity – only 3 top features as factors**

**R output :**

> View(mushroom\_train)

> myFormula <- V1 ~ factor(V6) + factor(V21) + factor(V10)

> View(mushroom\_train)

> mushroom\_ctree <- ctree(myFormula, data = mushroom\_train)

> table(predict(mushroom\_ctree), mushroom\_train$V1)

e p

e 2481 28

p 0 2365

> print(mushroom\_ctree)

Conditional inference tree with 5 terminal nodes

Response: V1

Inputs: factor(V6), factor(V21), factor(V10)

Number of observations: 4874

1) factor(V6) == {a, l, n}; criterion = 1, statistic = 4591.204

2) factor(V21) == {r}; criterion = 1, statistic = 1622.073

3)\* weights = 45

2) factor(V21) == {b, h, k, n, o, u, w, y}

4) factor(V21) == {w}; criterion = 1, statistic = 165.73

5) factor(V10) == {e, g, p}; criterion = 1, statistic = 46.679

6)\* weights = 168

5) factor(V10) == {w, y}

7)\* weights = 198

4) factor(V21) == {b, h, k, n, o, u, y}

8)\* weights = 2143

1) factor(V6) == {c, f, m, p, s, y}

9)\* weights = 2320

> plot(mushroom\_ctree)

> plot(mushroom\_ctree, type="simple")

> testPred <- predict(mushroom\_ctree, newdata = mushroom\_test)

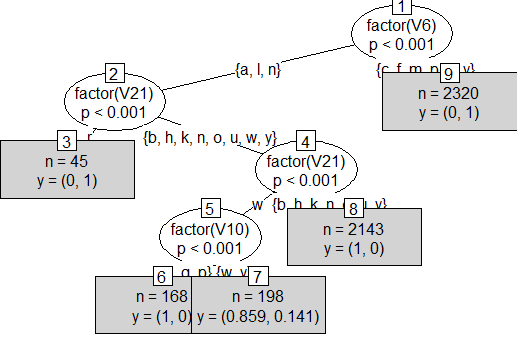
> table(testPred, mushroom\_test$V1)

testPred e p

e 1727 20

p 0 1503

Error = 1.14%



1. **Maximum complexity – top 10 features as factors**

> myFormula <- V1 ~ factor(V6) + factor(V21) + factor(V10) +factor(V20) +factor(V13) +factor(V14) +factor(V15) +factor(V16) +factor(V9) +factor(V22)

> mushroom\_ctree <- ctree(myFormula, data = mushroom\_train)

>

> table(predict(mushroom\_ctree), mushroom\_train$V1)

e p

e 2481 9

p 0 2384

> print(mushroom\_ctree)

Conditional inference tree with 9 terminal nodes

Response: V1

Inputs: factor(V6), factor(V21), factor(V10), factor(V20), factor(V13), factor(V14), factor(V15), factor(V16), factor(V9), factor(V22)

Number of observations: 4874

1) factor(V6) == {a, l, n}; criterion = 1, statistic = 4591.204

2) factor(V21) == {r}; criterion = 1, statistic = 1622.073

3)\* weights = 45

2) factor(V21) == {b, h, k, n, o, u, w, y}

4) factor(V16) == {n, y}; criterion = 1, statistic = 1414.958

5) factor(V13) == {k}; criterion = 1, statistic = 49.631

6)\* weights = 19

5) factor(V13) == {f, s, y}

7) factor(V22) == {c, y}; criterion = 1, statistic = 46

8)\* weights = 18

7) factor(V22) == {v}

9)\* weights = 29

4) factor(V16) == {e, g, o, p, w}

10) factor(V9) == {n}; criterion = 1, statistic = 61.66

11) factor(V21) == {h, n, u}; criterion = 1, statistic = 148

12)\* weights = 121

11) factor(V21) == {k, w}

13) factor(V22) == {v}; criterion = 1, statistic = 27

14)\* weights = 14

13) factor(V22) == {c, y}

15)\* weights = 14

10) factor(V9) == {b}

16)\* weights = 2294

1) factor(V6) == {c, f, m, p, s, y}

17)\* weights = 2320

> plot(mushroom\_ctree)

> plot(mushroom\_ctree, type="simple")

> testPred <- predict(mushroom\_ctree, newdata = mushroom\_test)

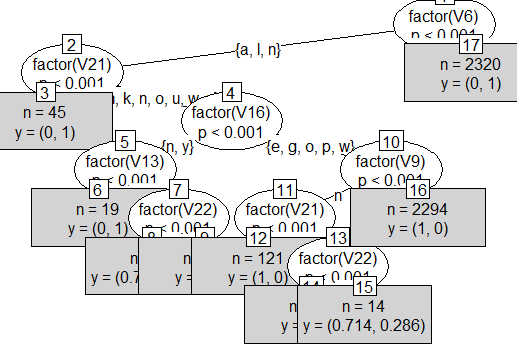
> table(testPred, mushroom\_test$V1)

testPred e p

e 1727 7

p 0 1516

Error = 0.4 %



|  |
| --- |
| Result: |
| **Minimum complexity – only 3 top features as factors – Accuracy is higher than Maximum complexity – top 10 features as factors** |
| Error is reduce to 0.4 % only with maximum complexity. |

Mushroom – Nearest Neighbor – Slide 6

Created dummy for each of top 10 feature variables for their factor using XL Miner.

Here is the KNN result using all 69 variables created as dummy categories (binary)





% Error is almost Zero for K=1,3,5 while for K=7 Error is 0.09 %

Phase:2

**Mnist Data –**

**MNIST – Explore – Slide-7**

1. **Fisher Projection of the entire train data:**

Created a R code for LDA and PCA on Mnist Data using train and test Mnist data provided for the analysis. Kindly note before running LDA and PCA, removed collinear and variables with mean and median 0 as they do not have any value in the column. Hence neglected 83 variables from the datasets.

R code for LDA:

library("MASS", lib.loc="C:/Program Files/R/R-3.2.0/library")

summary(mnist\_train\_new)

mnist\_train\_1 <- mnist\_train\_new[,-1]

train.lda <- lda(label ~ .,data=mnist\_train\_1) # training model

train.lda$LD1

train.lda$counts

plda <- predict(object = train.lda,

newdata = mnist\_train\_1[-1])

plda$x

head(plda$x) # LD projections

> head(plda$x) # LD projections

LD1 LD2 LD3 LD4 LD5 LD6 LD7 LD8

1 0.7069822 3.7021910 -0.5461599 1.08359013 -1.2820574 -0.6402381 -0.1616459 0.7117458

2 -4.7533733 -3.2570926 -2.9836820 -1.24400102 -1.8809339 -0.8985641 0.1144136 -1.0974090

3 0.4264746 5.1687065 -0.2150281 0.24889477 -3.7378078 0.1689028 0.5468668 0.1640582

4 -0.9784101 -0.5555026 1.1479453 -0.32452764 -0.9975683 -0.8583900 0.9794973 1.8460439

5 -4.8781840 -3.2443670 -4.7238761 -0.85004569 -1.9231773 -2.0935868 0.1667240 -2.2285545

6 -0.9629369 -1.2464329 -1.4953019 0.02343692 0.0826001 -1.3643632 0.4588783 0.3685504

LD9

1 0.09805234

2 1.23579005

3 -0.31479823

4 -0.20796284

5 0.99932801

6 0.06714345

#PCA

mnist\_train\_scale <- as.data.frame(scale(mnist\_train\_new)) # standardise the variables

mnist\_train.pca <- prcomp(mnist\_train\_1)

summary(mnist\_train.pca)

> summary(mnist\_train.pca)

Importance of components:

PC1 PC2 PC3 PC4 PC5 PC6 PC7

Standard deviation 578.60246 495.8700 459.40361 429.79831 409.96452 384.41296 335.46180

Proportion of Variance 0.09749 0.0716 0.06146 0.05379 0.04894 0.04303 0.03277

Cumulative Proportion 0.09749 0.1691 0.23055 0.28434 0.33329 0.37632 0.40909

PC8 PC9 PC10 PC11 PC12 PC13 PC14

Standard deviation 315.14357 308.24668 284.00843 268.49820 265.90717 241.79799 241.10303

Proportion of Variance 0.02892 0.02767 0.02349 0.02099 0.02059 0.01703 0.01693

Cumulative Proportion 0.43801 0.46568 0.48917 0.51016 0.53075 0.54778 0.56470



PCA Plots:





**MNIST – Logistic Regression -- Slide 8**

Since the data is huge and building 45 classifications model is practically impossible in the give time frame of assignment.

So, I have run the models only for two classification Label = (0,1) and Label = (5,6).

1. **Logistic Regression model with Top 9 Fisher features.**

**Model for Label =0 and Label =1**





**Model for Label =5 and Label =6**







1. **Logistic Regression model with Top 9 PCA features.**

**Model for Label =0 and Label =1**







**Model for Label =5 and Label =6**







Hence, for Logistic Models:

Average Error for LDA models are based on 2 random runs out of 45 are (0.28+2.36)/2 = 1.32

Average Error for PCA models are based on 2 random runs out of 45 are (0.26+4.9)/2 = 2.58

**ANS:**

LDA models will have better average accuracy than PCA models for all 45 classifiers.

**MNIST – K-Nearest Neighbor – Slide 9**

Due to data size limitation and computation on all of 784 non transformed variables, I am comparing the KNN result for PCA and LDA only.

As I have transformed the variable using Fisher projection and PCA, the accuracy of model will be higher compared to non-transformed. Sample of each of model is being run with accuracy for all K =1 to K=7.

**Sample for 50 data for each classes as test and train using XL Miner**



**KNN for PCA**





**KNN for LDA**





**KNN without Transformation:**





**ANS:**

**Accuracy for KNN without transform variables are very low and time consuming for resources. Hence LDA and PCA have better Accuracy.**

Summary Results KNN (K=1 to K=7)

With No transformation: With Fisher Projection (9) With PCA (9)

**Solution:  
LDA fisher with K =7 has best accuracy in classification.**

**MNIST – Bayesian Classifier – Slide10**

Naïve Bayes using Fisher Projection:

R code and output:

> library("e1071", lib.loc="~/R/win-library/3.2")

> train<- data\_final\_train[,710:718]

> test<- data\_final\_test[,710:718]

>

> train$label<- data\_final\_train$label

> test$label<- data\_final\_test$label

> View(test)

> model <- naiveBayes(label ~ ., data = train, laplace = 3)

> pred <- predict(model, test[,-10])

> table(pred, test$label)

pred 0 1 2 3 4 5 6 7 8 9

0 1530 0 8 3 1 17 26 10 4 13

1 0 1744 5 9 3 1 0 22 40 3

2 12 21 1476 59 18 22 22 25 32 5

3 8 8 50 1446 1 54 0 17 49 21

4 5 1 26 3 1482 6 10 25 12 82

5 25 6 8 82 13 1244 37 4 75 9

6 25 1 39 6 12 34 1512 1 12 0

7 1 2 4 30 3 12 0 1534 2 69

8 17 74 59 54 23 74 25 7 1405 22

9 1 2 10 26 98 23 0 132 34 1475

Error:



Average Error using LDA (9) is 11.61 %

Naïve Bayes using PCA (9):

|  |
| --- |
| train<- data\_final\_train[,719:727]  > test<- data\_final\_test[,719:727]  >  > train$label<- data\_final\_train$label  > test$label<- data\_final\_test$label  >  > model <- naiveBayes(label ~ ., data = train, laplace = 3)  > pred <- predict(model, test[,-10])  >  > table(pred, test$label)    pred 0 1 2 3 4 5 6 7 8 9  0 1339 0 28 14 5 49 37 10 16 13  1 0 1744 23 13 42 43 35 75 50 34  2 6 47 1289 47 8 12 52 28 54 14  3 36 8 49 1217 1 183 16 0 140 18  4 3 0 69 11 1198 43 25 50 15 436  5 177 29 24 193 10 984 103 26 91 40  6 57 12 93 16 31 25 1360 2 13 5  7 3 1 29 25 9 11 0 1455 3 102  8 3 16 76 148 28 75 3 45 1204 66  9 0 2 5 34 322 62 1 86 79 971 |
|  |
| |  | | --- | | > | |



Average Error using PCA (9) is 24 %

Naïve Bayes using Non transformed all Pixel variables:

> train<- data\_final\_train[,1:709]

> test<- data\_final\_test[,1:709]

> model <- naiveBayes(label ~ ., data = train, laplace = 3)

> pred <- predict(model, test[,-1])

>

> table(pred, test$label)

pred 0 1 2 3 4 5 6 7 8 9

0 1351 0 150 64 14 115 12 6 24 6

1 1 1796 58 101 26 60 34 28 269 34

2 6 0 259 2 5 2 3 0 2 0

3 2 4 238 558 4 27 1 11 17 1

4 2 0 12 4 318 11 1 20 12 14

5 0 0 21 9 5 53 5 1 3 0

6 110 13 521 138 172 106 1541 16 40 2

7 0 0 2 5 3 1 1 420 1 6

8 117 35 394 649 255 949 31 76 1019 38

9 35 11 30 188 852 163 3 1199 278 1598



Average Error using non transformed variables is 33.61 %

**ANS:**

Bayesian LDA models have better average accuracy than PCA and non- transformed models for all 45 classifiers.