DATA SCIENCE INTERVIEW PREPARATION NOTES

XUNTAO HU

Contents

1. Statistics 2
1.1. Basics Concepts 2
1.2. Common Families of Distributions 3
1.3. Multi-variable Distributions 4
1.4. Estimators 5
1.5. Sampling 5
1.6. Hypothesis Testing 6
2. Machine Learning 10
2.1. Linear Regression 10
2.2. Logistic Regression 11
2.3. Gaussian Discriminant Analysis 12
2.4. Boostrap 13
2.5. Random Forest 14
2.6. Support Vector Machine 14
2.7. K-Means Clustering 14
2.8. Hierarchical Clustering 14
2.9. Gaussian Mixture Model (EM Algorithm) 15
2.10. Methods of Dimension Reduction 15
3. Pros and Cons List 16
3.1. Regressions 16
3.2. Classifications 17
3.3. Unsupervised Learning 18
4. Recommendation System 19
4.1. Content Based Recommendation 19
4.2. Collaborative Filtering 20
4.3. Technicalities 21
4.4. Pros and Cons 22
5. Deep Learning 22
6. Coding 22

References 22

This note is the official write-up of my personal notes for preparing for interviews for Data Scientist positions.

Remember that this note is not self-contained (yet), therefore it may not fit perfectly into your knowledge system. There are some concepts I missed out, for instance Bayesian Formula, Cross Validation, etc. This note will be beneficial to the reader who already has built the knowledge system of statistics, machine learning, and are looking for a quick summary and review before the interviews.

1. Statistics

To prepare for stats questions I used the book [SI].

1.1. **Basics Concepts.**

1.1.1. *Density functions.*

(1) Cumulative Distribution Function (cdf): $F_X(x) := P(X \leq x)$, the probability of a random variable X taking a value less of equal to x.
(2) Probability Density Function (pdf): $f_X(x) = \frac{d}{dx}F_X(x)$.
(3) Expectation: $Eg(x) = \int_{-\infty}^{\infty} g(t)f_X(t)dt$.
(4) Variance: $Var(X) = E(X - EX)^2 = E(X^2 - 2X * EX + EX^2) = EX^2 - (EX)^2$

We also see

$$F_X(x) = \int_{-\infty}^{x} f_X(t)dt.$$

Or in particular,

$$P(a \leq X \leq b) = F_X(b) - F_X(a) = \int_a^b f_X(t)dt.$$

1.1.2. *Change of variable.* if $Y = g(X)$:

$$F_Y(y) = P(Y \leq y) = P(g(X) \leq y).$$

From here: if g increasing, then $P(g(X) \leq y) = P(X \leq g^{-1}(y)) = F_X(g^{-1}(y))$; if g decreasing, then $P(g(X) \leq y) = 1 - P(X \leq g^{-1}(y)) = 1 - F_X(g^{-1}(y)$.

1.1.3. *Expectation and Variance of Sums.* , if $X_1, \ldots, X_n$ iid,

$$E(\sum g(X_i)) = nEg(X_i)$$

$$Var(\sum gX_i) = nVar(g(X_i))$$

1.2. **Common Families of Distributions.**

1.2.1. *Discrete Distributions.*

(1) Discrete Uniform Distribution:
- pmf: $P(X = x|N) = 1/N$ when $x = 1, \ldots, N$

(2) Bernoulli Distribution:
- pmf: Given $0 \leq p \leq 1$, $P(x = 1) = p$, and $P(x = 0) = 1 - p$
- $EX = p$,
- $Var(X) = E(E - EX)^2 = (1 - p)^2 p + (0 - p)^2(1 - p) = p(1 - p)$

(3) Binomial Distribution:
- pmf: $P(X = x|n, p) = \binom{n}{x} p^x (1 - p)^{n-x}$.
- This gives the probability of exactly x success in n trial when the outcome of each trial is iid Bernoulli(p).
- $EX = np$, $VarX = np(1 - p)$.
- If $X_1, \ldots, X_n \sim bernoulli(p)$ iid, then $\sum X_i \sim binomial(n, p)$.

(4) Poisson Distribution:
- pmf: $P(X = x|\lambda) = \frac{e^{-\lambda}\lambda^x}{x!}$
- $EX = Var(X) = \lambda$
- Given λ, the expected number of occurrence of an event in a given period (e.g. a minute), $P(X = x|\lambda)$ gives the possibility of the event occur x times in the next period.

(5) Negative Binomial Distribution:
- pmf: $P(X = x|r, p) = \binom{x-1}{r-1} p^r (1 - p)^{x-r}$
- Number of trials until the r-th success, while each trial is a binomial variable with prob p.

(6) Geometric Distribution:
- pmf: $P(X = x|p) = p(1 - p)^{x-1}$
- Number of trials until the first success. Special case of Negative Binomial Distribution when $r = 1$.

1.2.2. *Continuous Distributions.*

(1) Uniform Distribution:
- pdf: $f(x|a, b) = 1/(b - a)$ if $a \leq x \leq b$ else 0.
- $EX = (a + b)/2$, $Var(X) = \frac{b^3 - a^3}{3(b-a)} - (\frac{a+b}{2})^2$

(2) Normal DIstribution:
- pdf: $f(x|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} exp(-\frac{(x-\mu)^2}{2\sigma^2})$
- $EX = \mu$, $VarX = \sigma^2$

(3) Gamma Distribution:
- pdf: $f(t|\alpha) = \frac{t^{\alpha-1}e^{-t}}{\Gamma(\alpha)}$, where $\Gamma(\alpha) = \int_0^\infty t^{\alpha-1}e^{-t}dt$.

- change $x = \beta t$: $f(x|\alpha, \beta) = \frac{x^{\alpha-1}e^{-x/\beta}}{\Gamma(\alpha)\beta^{\alpha}}$.
- $EX = \alpha\beta$, $VarX = \alpha\beta^2$

(4) χ^2 Distribution with p degree of freedom:
- pdf: $f(x|p) = \frac{x^{p/2-1}e^{x/2}}{\Gamma(p/2)2^{p/2}}$. This is Gamma Distribution with $\alpha = p/2, \beta = 2$.
- $EX = p$, $VarX = 2p$.

(5) Exponential Distribution:
- pdf: $f(x|\beta) = \frac{1}{\beta}e^{-x/\beta}$. This is Gamma Distribution with $\alpha = 1$
- When sampling $X_1, \ldots, X_n \sim exp(\beta)$ iid, we have $\sum X_i \sim Gamma(n, \beta)$.

1.3. **Multi-variable Distributions.** Joint cumulative distribution function $F_{X,Y}(x, y) = P(X \leq x, Y \leq y)$. Joint pdf $f_{X,Y}(x, y) = \frac{\partial^2 F_{X,Y}}{\partial x \partial y}$.

Given a joint distribution $f_{X,Y}(x, y)$, find the pdf for $Z = g(X, Y)$:

$$F_Z(z) = P(Z \leq z) = P(g(X, Y) < z) = \int_{g(X,Y)<z} f_{X,Y}(x, y)dxdy$$

1.3.1. *Conditional Probability.* Marginal density function: $f_X(x) = \int_{-\infty}^{\infty} f_{X,Y}(x, y)dy$. (Similarly for $f_Y(y)$). The conditional pdf is given by

$$f(x|y) = \frac{f_{X,Y}(x, y)}{f_X(x)}$$

If two variables are independent: $f_{X,Y}(x, y) = f_X(x)f_Y(y)$.

1.3.2. *Bivariate Transformations.* . Given variable change: $X, Y \to U = g_1(X, Y), V = g_2(X, Y)$, then

$$f_{U,V}(u, v) = \sum_{X,Y \in A_{U,V}} f_{X,Y}(x, y)$$

If the transformation is invertible, namely we can write $X = h_1(U, V), Y = h_2(U, V)$, then

$$f_{U,V}(u, v) = f_{X,Y}(x, y)|Jac(h_1, h_2)|$$

. Note: sometimes only $U = g(X, Y)$ is given, we can find some $V = g_2(X, Y)$, so that we can compute $f_{U,V}$, and further find the marginal pdf f_U.

1.3.3. *Useful Cases.*

- Sum: $P(X + Y < c)$:

$$F(X + Y) = P(X + Y < c) = \int_{-\infty}^{infty} (\int_{-\infty}^{c-y} f_{X,Y}(x, y)dx)dy$$

- Max: Given $(X_i)_{i=1,\dots,n}$ iid.

$$F_{max}(x) = P(\max X_i < x) = P(X_1 < x, \dots, X_n < x) = \prod P(X_i < x) = F^n(x)$$

- Min: Given $(X_i)_{i=1,\dots,n}$ iid.

$$F_{min}(x) = P(\min X_i < x) = 1 - P(\min X_i > x) = 1 - \prod P(X_i > x) = 1 - (1 - F(x))^n$$

1.3.4. *Expectation, Covariance, Correlation.*

$$Eg(X,Y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x,y) f_{X,Y}(x,y) dxdy$$

$$Cov(X,Y) = E((X - EX)(Y - EY)) = E(X,Y) - EXEY$$

$$\rho_{X,Y} = \frac{Cov(X,Y)}{\sqrt{Var(X) VarY}}$$

If X, Y are independent:

$$E(X,Y) = \int xyf(x,y) = \int xyf_X(x) f_Y(y) = \int xf_X \int yf_Y = EXEY$$

$$Cov(X,Y) = 0$$

Covariance is linear in both variables.

1.4. **Estimators.**

1.5. **Sampling.** Assume $X_1, \dots, X_n$ are iid variables and have $n(\mu, \sigma^2)$ distribution. We then know two quantities:

$$E\bar{X} = \mu, \qquad Var\bar{X} = \frac{\sigma^2}{n}$$

We will have our discussion in two cases: whether the standard variation σ is known or unknown.

1.5.1. *σ known.* In this case $\bar{X} \sim n(\mu, \sigma^2/n)$, so $\frac{\bar{X}-\mu}{\sigma/\sqrt{n}} \sim n(0,1)$.

1.5.2. *σ unknown.* This is a trickier case. We first need to estimate σ using the samples.

An estimate of μ is $\bar{X}$, as seen above. So we can define

$$S^2 = \frac{1}{n-1}\sum (X_i - \bar{X})^2$$

Note that the $n-1$ in the denominator is essential, due to the following fact

$$\begin{aligned}(n-1)ES^2 &= E(\sum (X_i - \bar{X})^2) \\ &= \sum E(X_i - \mu)^2 - nE(\bar{X} - \mu)^2 \\ &= \sum Var(X_i) - nVar\bar{X} \\ &= (n-1)\sigma^2\end{aligned} \tag{1.1}$$

Therefore $ES^2 = \sigma^2$, so S^2 is an unbiased estimator of the variance.

We can then consider the distribution of the quantity $\frac{\bar{X}-\mu}{S/\sqrt{n}}$. Note that it is not a normal distribution. Since $\frac{(n-1)S^2}{\sigma^2} \sim \chi^2_{n-1}$, we can compute the pdf of

$$\frac{\bar{X}-\mu}{S/\sqrt{n}} = \frac{\bar{X}-\mu}{\sigma/\sqrt{n}} / \sqrt{\frac{S^2}{\sigma^2}}$$

where we know the numerator has a $n(0,1)$ distribution and the square of the denominator has a $\chi^2_{n-1}/(n-1)$ distribution. The pdf of the above variable is

$$f(t) = \frac{\Gamma((p+1)/2)}{\sqrt{2\pi p}\Gamma(p/2)2^{p/2}}(\frac{2}{1+t/p})^{(p+1)/2}$$

This distribution is called *Student's t-distribution*.

1.6. **Hypothesis Testing.**

1.6.1. *Confidence Interval.* The confidence interval is a range of values that the target value is likely (with certain probability) to fall into. There are two ways to find a (95%) confidence interval;

(1) Use boostraping to get a large number (k) of training sets, do experiments on these sets and get k values of the desired quantity. Sort them from small to large, then take the middle 95% of the values.
(2) The confidence interval can be found using the sample mean, the standard variation and the sample size.

1.6.2. *p-value.* The p-value is used in determining statistical significance in a hypothesis test. The value evaluates the probability that we will make a mistake if we reject the null hypothesis (false positive). We will want small p-value, and the usual threshold is $p = 0.05$.

1.6.3. *z-test.* If we have samples set $\{x_i\}$, and we know that the set follows a normal distribution with known variance σ^2 and unknown mean. We want to test if the mean of the set $\bar{x}$ is significant different to a given value μ_0. We can use the z-test. Compute the following z-value:

$$z = \frac{\bar{x} - \mu_0}{\sigma\sqrt{n}}$$

Our null hypothesis is that H_0: the set does not show a significant difference to a $n(\mu_0, \sigma^2)$ distribution. Under this hypothesis, $z \sim n(0, 1)$. Therefore if our actual data gives us a $z > z_{n,\alpha/2}$ where α is usually taken to be $0.05 = 5\%$. That is to say:

$$z = \frac{\bar{x} - \mu_0}{\sigma\sqrt{n}} > z_{n,\alpha/2}.$$

Then it means that there is less than 5% of chance that this scenario would happen, but it does happen. Therefore we have 95% of confidence to reject the null hypothesis and to state that the mean of the sample set is different from μ_0.

Another way to look at this is via the confidence interval. The 95% confidence interval of the population mean is

$$[\bar{x} - z_{n,\alpha/2} * \sigma/\sqrt{n}, \qquad \bar{x} + z_{n,\alpha/2}\sigma\sqrt{n}].$$

This means we have 95% of confidence that the true value of our mean falls into this interval. And if μ_0 does NOT fall into this interval, that means we have 95% of confidence that μ_0 is not our true value of sample mean.

We also want to mention the relationship to the p-value here: the discussion above compares the calculated z to $z_{n,\alpha/2}$. We found z_{n,α_2} via the normal distribution table. We can also find the corresponding α value of our calculated z. This is the probability of rejecting the null hypothesis and made a mistake, so the α value here is in fact our p-value.

1.6.4. *Example - Coin Flipping.* We want to discuss a most basic example of the z-test, which is also the most commonly seen interview question on the A/B testing.

We have a coin. We want to test if the coin is fair by continuously flipping it and record how many heads and how many tails we will get. Assume we flip the coin n times and get k heads. Let $\hat{p} = k/n$. Naturally, the null hypothesis here is $H_0 : \hat{p} = 0.5$.

IMPORTANT NOTE: it is very easy to confuse about the "experiment" we run here. We don't view the whole event as one experiment. Instead, we view each flip

as an experiment. We can view the result of each flip as an independent Bernoulli variable, which either gets head with probability p, or gets tail with probability $(1-p)$. Here $p = 0.5$, but we can of course use other values, depending on what we want to test.

Now suppose the null hypothesis holds, then the mean of the set of Bernoulli trials is equal to p. Note that this is a binomial variable. We would also like to know the variance of the Bernoulli experiments - as in Section 2 we know that it has variance $p(1-p)/n$.

It is very important to know that when the number of trials is large, we can approximate this quantity using normal distribution (the Central Limit Theorem). Since in this case the variance is known, we can run the z-test with the following z-value

$$z = \frac{\hat{p} - p}{\sqrt{p(1-p)/n}}$$

Therefore if we have $z > z_{n,\alpha/2}$, we can reject the null hypothesis and say that the new version of the ad does make a difference.

Furthermore, we can compute the confidence interval for the result:

$$[\hat{p} - z_{n,\alpha/2}\sqrt{p(1-p)/n}, \qquad \hat{p} + z_{n,\alpha/2}\sqrt{p(1-p)/n}]$$

This means we have $(1-\alpha)$ confidence that the true probability of getting the head lies in this interval. If $p = 0.5$ does not lie in this interval, that means it is different to the true value, and hence the coin is not fair.

1.6.5. *One sample t-test.* If we have samples set $\{x_i\}$, and we know that the set follows a normal distribution with unknown variance and unknown mean. We want to test if the mean of the set $\bar{x}$ is significant different to a given value μ_0. Note the difference to the z-test here is that the variance is unknown. Hence we will need to use the estimator of the variance, as discussed above.

Let $S = \frac{\sum(x_i - \bar{x})^2}{n-1}$, the unbiased estimator of σ^2. Then the t-stats to test $H_0 : \bar{x} = \mu_0$ is:

$$t = \frac{|\bar{x} - \mu_0|}{s/\sqrt{n}}.$$

From here on it is the standard procedure: we can either A) read the table and find $t_{n-1,\alpha/2}$ and if $t > t_{n-1,\alpha/2}$ the we reject H_0; or B) read the table and find the corresponding p-value of the t, and compare the p-value with the threshold α (normally 0.05), if $p < \alpha/2$, then we reject H_0.

1.6.6. *Two sample t-test.* This is perhaps the most common A/B testing scenario. The two sample t-test is to test whether two populations are significantly different from one another.

If the two sets of data are **paired**: then we can just use one sample t-test on the differences of the paired samples and $\mu_0 = 0$.

If the two sets of data are **unpaired**: that means assuming the two populations are independent (most common cases). In this case we want to test H_0 : the means of the two populations are equal.

If we assume **equal variance**, then we should look at

$$t = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{s^2(\frac{1}{n_1} + \frac{1}{n_2})}}.$$

If we assume **unequal variance**, then we should look at

$$t = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}}$$

In both cases the degree of freedom is $(n_1 - 1) + (n_2 - 1)$.

1.6.7. *Example: Click Through Rate (CTR).* Consider the scenario that we have two ads (or maybe two version of a website), we want to test whether they will introduce a significant difference. We collected n_1 users who have viewed the new version of the ad. Inside the n_1 users we have k_1 users who click on the ad. We also have a reference group of n_2 users in which k_2 clicked on the on the old version of the ad. Now we have click through rates (CTR) for the two groups $\hat{p}_1 = k_1/n_1$ and $\hat{p}_2 = k_2/n_2$.

IMPORTANT NOTICE: We now want to test whether the *true values* p_1, p_2 of the CTR of the two version is the same. The problem is that we don't know the true values. We can only use the results of the experiment $\hat{p}_1$ and $\hat{p}_2$ to estimate the true values p_1 and p_2.

As before, under the null hypothesis, we can assume that the results both limit to normal distributions. Then we have

$$\hat{p}_1 \sim n(p_1, p_1(1-p_1)/n_1)$$
$$\hat{p}_2 \sim n(p_2, p_2(1-p_2)/n_2)$$

We can deduce that

$$\hat{p}_1 - \hat{p}_2 \sim n(p_1 - p_2, \frac{p_1(1-p_1)}{n_1} + \frac{p_2(1-p_2)}{n_2})$$

where in fact we will plug $\hat{p}_1$ into p_1 and $\hat{p}_2$ into p_2, and compute the following t-stats:

$$t = \frac{|\hat{p}_1 - \hat{p}_2|}{\sqrt{\frac{p_1(1-p_1)}{n_1} + \frac{p_2(1-p_2)}{n_2}}}.$$

The degree of freedom is $n_1 + n_2 - 2$. As long as $t > t_{n_1+n_2-2,\alpha/2}$, we can reject the null hypothesis and declare that the two versions make a difference.

1.6.8. χ^2 *test.*

2. Machine Learning

I used the book [ISLR] to prepare for Machine Learning models.

2.1. **Linear Regression.** Supervised Learning. All discussion here is for simple linear regression, namely the predictor is 1-dim. Given data points (x_i, y_i) where $i = 1, \ldots, n$, where n is the number of the input data.

For a model whose linear function is $\hat{y} = a + bx$. The *cost function* of the linear model is the Residual Sum of Squares:

$$RSS = \sum_1^n (y_i - (a + bx_i))^2.$$

We want the "best" a, b such that RSS is minimized.

Taking $\frac{dRSS}{da} = 0$, we can deduce $a = \bar{y} - b\bar{x}$.

Taking $\frac{dRSS}{db} = 0$, we can get

$$b = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sum (x_i - \bar{x})} = \frac{S_{xy}}{S_{xx}}$$

Assuming all y_i is iid and have variance σ^2, we can compute

$$Var(a) = \sigma^2 \frac{\sum x_i^2}{nS_{xx}}, \qquad Var(b) = \frac{\sigma^2}{S_{xx}}$$

We then know that $b \sim n(\frac{S_{xy}}{S_{xx}}, \frac{\sigma^2}{S_{xx}})$ and $a \sim n(\bar{y} - b\bar{x}, \sigma^2 \frac{\sum x_i^2}{nS_{xx}},)$.

We often don't know the variance of y_i, therefore we need an estimator for that. MLE of σ^2 is called Residual Squared Error:

$$S^2 := RSE^2 = RSS/(n-2)$$

Then the MLE for $Var(b) = \sigma^2/S_{xx}$ is S^2/S_{xx}. So we can use $t - test$ for the null hypothesis $b = 0$:

$$t = \frac{b - 0}{\sqrt{s^2/S_{xx}}} \sim t_{n-2}$$

To reject $b = 0$, we only need $|t| > t_{n-2,\alpha/2}$.

2.1.1. *Important Stats.* The R-Squared stats is computed using $TSS = \sum(y_i - \bar{y})$ and $RSS = \sum(y_i - \hat{y}_i)$:

$$R^2 = \frac{TSS - RSS}{TSS}$$

It is the portion of the variance explained by the linear model. Interesting fact: $R = Cor(X, Y) = \frac{S_{xy}}{\sqrt{S_{xx}S_{yy}}}$ is the square root of R^2. This makes sense since both quantities measures the the level of linearity among the data points.

More General Case: F-stats will be used when the number of predictor is p

$$F = \frac{(TSS - RSS)/p}{RSS/(n - p - 1)}$$

R-square is the same as the F-stats when $p = 1$.

IMPORTANT NOTE: Both R-Square and F stats can only be used on measuring the level of linearity on the training data. We need to use other measurements such as MSE (mean squared error), or the above mentioned RSS for test error.

2.1.2. *Lasso and Ridge.* The idea on Lasso and Ridge is to add a punishment on the cost function.

Lasso: $\lambda \sum |b_i|$ - L1 norm of the coeffients

Ridge: $\lambda \sum b_i^2$ - L2 norm of the coeffients

One big and essential difference is that Lasso can introduce sparsity into the coefficients. Namely some of the coefficients will become zero. Therefore Lasso is automatically selecting features for us.

However, the purpose of feature selection is to reduce collinearity among the features. Ridge can also have a similar effect. For instance is $x_1 = 2x_2$, then we will get $b_1 = 1/2 * b_2$. In usual linear model both b_1 and b_2 are significant. But when we apply the punish terms, b_1 will drop to zero much faster than b_2. In Lasso it might directly become 0.

2.2. **Logistic Regression.** Sigmoid function:

$$f(x) = \frac{1}{1 + e^x}$$

This function takes value between 0 and 1 (S-shape).

Basic idea of logistic regression is to use the Sigmoid function to estimate the probability of $Y = 1$ or $Y = 0$ given x. The formulation is basically a linear model composite with the sigmoid function:

$$P(Y = 0|X) = \frac{1}{1 + e^{a+bx}}, \qquad P(X) := P(Y = 1|X) = \frac{e^{a+bx}}{1 + e^{a+bx}}$$

Or to write in another way:

$$a + bX = \log(\frac{P(X)}{1 - P(X)})$$

Cost function of the model is often the maximal likehood:

$$l(a, b) = \prod_{y_i=1} P(x_i) \prod_{y_i=0} (1 - P(x_i)).$$

The goal is to find a, b such that $l(a, b)$ is minimized.

2.3. **Gaussian Discriminant Analysis.** We only discuss Linear Discriminant Analysis for now, at the end we will get a touch on the quadratic case.

2.3.1. *Bayesian Distribution.* The basic idea is to use the Bayes Formula to compute the posterior probability using the prior probability. We are given data points $(x_1, \ldots, x_n)$. For each data point x_i we also know its class $y_i = k$, where $k \in \{1, \ldots, m\}$ is the index of a class. We can know the prior probability by counting:

$$\pi_k = P(Y = k).$$

If we also know the pdf of the distribution of the data points within each class k, that is $f(X|Y = k)$, we can compute the posterior probability:

$$P(Y = k|X = x) = \frac{\pi_k f(X = x|Y = k)}{\sum \pi_i f(X = x|Y = i)}$$

This conditional probability can be interpreted as the probability of the data point x belonging to the class k. Then by comparing the probabilities varying k of the same data point, we can choose the k that correspond to the largest probability - that is our prediction of the class x belongs to.

2.3.2. *Gaussian Discriminants.* We now carry out the computation under the assumption that $x \sim n(\mu_k, \sigma_k^2)$, that is

$$f(x|y = k) = \frac{1}{\sigma_k \sqrt{2\pi}} \exp(-\frac{(x - \mu_k)^2}{2\sigma_k^2})$$

We want to make a further assumption:

$$\sigma_1^2 = \ldots = \sigma_m^2 =: \sigma^2$$

Note that this assumption leads to the linearity of the decision boundary. By removing this assumption we may achieve a quadratic boundary at computational costs.

We can now write down the posterior probability:

GDA (2.1) $$P_k(x) := P(Y = k|X = x) = \frac{\pi_k \exp(-\frac{(x-\mu_k)^2}{2\sigma^2})}{m(x)}$$

We use $m(x)$ to denote the denominator, note that it is independent of k.

Taking the logarithm, we get

$$\log P_k(x) = (\frac{\mu_k}{\sigma^2}x - (\frac{\mu_k}{2\sigma^2} - \log \pi_k)) - (\frac{x^2}{2\sigma^2} + \log m(x))$$

Note that the last two terms are independent of k. Therefore if we want to return the k such that the P_k is the largest, we only need to look at

$$\delta_k(x) := \frac{\mu_k}{\sigma^2}x - (\frac{\mu_k}{2\sigma^2} - \log \pi_k)$$

It is a linear function. And if we don't make the equal-variance assumption, then the last but one term must be included, and δ_k will be quadratic.

2.3.3. *Unbiased Estimators.* Finally, in order to compare the δ_ks, we need to have unbiased estimators for π_k, μ_k and σ^2. We already know that $\pi_k = n_k/(\sum n_k)$ can be computed by getting the count (n_k) of the points that belonged to the k-th class . We can estimate the rest by

GDA_MLE (2.2)
$$\mu_k = \frac{\sum_{y_i=k} x_i}{n_k} \qquad \sigma^2 = \frac{1}{n-m}\sum_{k=1}^{m}\sum_{y_i=k}(x_i - \mu_k)^2.$$

2.4. **Boostrap.** Boostrap is the name for sampling with replacement. It can create training set of desirable size. We can use Boostrap to estimate standard deviation and to reduce variance.

2.4.1. *Estimating the standard deviation.*

(1) Sample with replacement n samples from the original data set.
(2) Compute the std / median / mean (the quantity that you want to estimate) from the sample pile
(3) Average among all sample piles.

2.4.2. *Reducing Variance.* This is also called Boostrap Aggregation (Bagging) method. The basic reason we use here is if we take $z_1, \ldots, z_k \sim n(0, \sigma^2)$, then $\bar{z} \sim n(0, \sigma^2/n)$.

(1) Boostrap and get k training sets
(2) train the model (often an inflexible one, such as a decision tree model) on each training sets and get k models with function $f_1, \ldots, f_k$.
(3) Average and get $\bar{f}$.

2.5. **Random Forest.** Random Forest is basically a Bagging model, plus an important feature: Every time when we train the models, instead of using the full set of features, we randomly pick $m < p$ features from the set of features (typically $m = \sqrt{p}$).

If there is one significant feature in the set of features, then very likely that all bagged trees looking similar. Using Random Forest prevents that, and hence further reduce variance.

2.5.1. *Out of bag Error (OOB).*

(1) Estimate test error on each model
(2) Compute the mean test error on the data point x_i for those models that does NOT contain x_i in their training set.

2.6. **Support Vector Machine.**

2.7. **K-Means Clustering.** Unsupervised Learning.

2.7.1. *Algorithm.*

(1) Randomly Assign each observation to a group (indexed by $1, \ldots, k$ for instance).
(2) Compute the cluster centroid (mean of each feature within the cluster).
(3) Reassign each observation to the cluster that has the closest centroid.
(4) repeat the two steps above until convergence.

Since the K-means can only converge to a local minimum, it is suggested that we repeat the steps k times and choose the "best" one.

Although we don't have y_i so we can't compute the accuracy, but we can still measure which model is better by using the sum-of-square deviation:

$$W(C_k) := \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_j (x_{ij} - x_{i'j})^2 = 2 \sum_{i \in C_k} \sum_j (x_{ij} - \bar{x}_{kj})^2$$

where i is the index for data points, and j is the index for features. So $W(C_k)$ is the sum of Euclidean distance of the data points to the cluster centroid.

2.7.2. *How to Choose k.* We usually choose different k to train and get different models. When k increase, $W(C_k)$ is almost guaranteed to reduce. But the reducing rate will slow down when k become larger (L-shape curve). We usually choose the turning point of the L shape curve to be the ideal k that balance the cost function and computer power.

2.8. **Hierarchical Clustering.** Unsupervised Learning. One big advantage is that we don't have to prefix k. One big disadvantage is that when we choose different linkage and distance, the result might vary greatly.

2.8.1. *Algorithm.*

(1) Select the type of Distance and the type of Linkage.
(2) Treat each object as its own cluster. For $i = 0, \ldots, n-1$ do:
Compute $\binom{n-i}{2}$ pair-wise distance, fuse the closest two observations.
(3) The process ends when we have only one cluster. We cut the tree off at the level of a desirable k branches.

2.8.2. *Linkages.*

- Complete: Maximal distance between two points from the two clusters
- Single: Minimal distance ...
- Average: Mean distance ...
- Centroid: Distance between two centroids ...

2.9. **Gaussian Mixture Model (EM Algorithm).** This section is closely related to the Gaussian Discriminant Analysis section. Given a set of data points $(x_1, \ldots, x_n)$, and number of classes k. Note that we don't know the classes y_i of each data point, therefore this is an unsupervised learning algorithm.

We want to classify the data points into k classes, so that the distribution of data points within each class has a normal distribution. Namely, we assume

$$P(x_i|z_i = j) \sim n(\mu_j, \sigma_j)$$

Note that the z_i is the class of x_i, and we don't know z_i.

2.9.1. *EM algorithm.* EM stands for Expectation-Maximization. Initialize by randomly assign classes, then repeat until convergence:

Step E : Given the current distributions, compute the posterior probability that the data point x_i belongs to the j-th class:

$$w_j^{(i)} = P(z_i = j|X = x_i, \pi_j, \mu_j, \sigma_j)$$

as in (2.1).

Step M : We have the most updated z_i, we can then update the parameters using the Maximal Likelihood Estimators of the current data as in (2.2). The likelihood function is:

$$l(\pi_i, \mu_i, \sigma_i) = \sum \log P(x_i, \pi_i, \mu_i, \sigma_i).$$

2.9.2. *Relation to GDA.* We can see that Gaussian Discriminant Analysis is the process from Step M to Step E: we first use the current labeled data point to compute the MLE for π, μ, σ, then when a new data point comes in, we can use Step E to compute for the probability of each class.

2.10. **Methods of Dimension Reduction.**

3. Pros and Cons List

The general guideline for Machine Learning models is their flexibility.

- Flexible Models: Such as Boosting models. They tend to have *less bias, and more variance* on test sets. They fit better on training sets and may sometimes overfit. Not easy to interpret.
- Inflexible Models: Such as Linear models. They tend to have *more bias, and less variance* on test sets. They usually have more rigid shape and easier to interpret.

3.1. **Regressions.**

3.1.1. *Linear Regression.*

- Pros:
  (1) Easy to understand;
  (2) Can do feature selection by looking at p-values of each coeffiecient
  (3) Fast / Computationally cheap
  (4) Low test variance and avoid overfitting (Inflexible)
- Cons:
  (1) Suffer from high leverage
  (2) For the best result, the variables are preferred to be numerical
  (3) Can't be used on Classification Problems
  (4) Suffer from multi-linearity among variables
  (5) Has an assumption on the true relationships (Linear, Quadratic...)
  (6) Does not accommodate for missing values.

3.1.2. *Lasso VS Ridge.* Both requires scaling the parameters.

- Pros:
  (1) (Lasso) Can introduce sparsity into the coefficients. So if there is a subset of significant features, then it is a pro.
  (2) (Both) Reduce collinearity among the feature.
- Cons:
  (1) If all features are important, then maybe not as good as usual LR or Ridge (too automatics.)

3.1.3. *Decision Trees.*

- Pros:
  (1) Easy to explain (can plot easily)
  (2) Handles categorical features.
  (3) Works well if the true decision boundary is parallel to feature axis.
- Cons:

(1) (Due to flexibility) may easily overfit.
(2) Do not work well is the true decision boundary is not parallel.
(3) (Non-robust) small change can lead to a big chance in the result.

3.1.4. *Random Forest.*

- Pros:
  (1) Reduce Variance by averaging over multiple models.
  (2) Can use OOB estimates for the test errors
  (3) Avoid highly correlated features
- Cons:
  (1) Not easy to explain / interpreted visually
  (2) May overfit
  (3) Learn slowly if not pruned suitably.

3.1.5. *Boosted Trees.*

- Pros:
  (1) Using smaller trees enhance model interpretability (stumps (1-level trees) gives an additive model)
- Cons:
  (1)

3.2. **Classifications.**

3.2.1. *Naive Bayes.*

- Pros:
  (1) Simple to implement
  (2) Computationally fast
  (3) Works well in higher dimensions
- Cons:
  (1) Highly replying on the variables being independent
  (2) Zero Frequency (for missing data in certain category)
  (3) not so accurate

3.2.2. *Logistic Regression.*

- Pros:
  (1) (Due to Inflexibility) Low variance
  (2) (Soft Assign) Gives probability for outcomes
  (3) Work well with different boundary shapes
  (4) Can use L1/L2 regularization to avoid overfitting
- Cons:
  (1) High bias

3.2.3. *Support Vector Machines.*

- Pros:
  (1) Perform well to non-linear boundary using Kernal Trick
  (2) Handles High dimensional data
- Cons:
  (1) Need to use a correct kernel

3.2.4. *Gaussian Discriminant Analysis.*

- Pros:
  (1) Multi-class prediction
  (2) When data size is large, well performed in accuracy and fast
  (3) require less data to train well. (similar to Naive Bayes).
- Cons:
  (1) Assume normal distribution
  (2) Not so robust comparing to Logistic regression
  (3) No good for few-categorical variables
  (4) Suffer collinearity.

3.3. **Unsupervised Learning.**

3.3.1. *K-means Clustering.*

- Pros:
  (1) Simple / Fast for large data sets
  (2) Easy to interpret
  (3) Accuracy
  (4) Spherical Clusters (can be a con too)
- Cons:
  (1) Need to choose K
  (2) Local method - may converge to different clusters
  (3) Clusters are of equal sizes
  (4) Sensitive to scales
  (5) Only for Numerical data

3.3.2. *Gaussian Mixture Model.*

- Pros: (Comparing to K-means)
  (1) Doesn't depend on L-2 norm
  (2) Soft Assign, give estimate to probabilities.
- Cons:
  (1) Need to assume the data are normally distributed in each class.

4. Recommendation System

I learn about the recommendation system from [MMD, Chapter 9]. We will review two types of recommendation system here: Content-Based Recommendation and Collaborative Filtering.

Both types of our recommendation systems rely on the *Utility Matrix*:

User \ Product	X	Y	Z
A	4	5	6
B	7	8	9
C	10	11	12

Every entry of the table reflects relationships between a user and a product. For instance it can be ratings if we are talking about movies and their viewers, or a boolean variable if we are recording whether or not the user has used the product.

4.1. **Content Based Recommendation.**

4.1.1. *Step 1. Establish Features.*

In this step we need to choose features of our products. Examples of the features are: A) For Movies: whether an actor is in the movie; or B) For Computers: the parameters such as disk size, memory, processor type, etc; or C) For News Articles: TF-IDF of words. note that these features must be shared among all our products, for instants if our products are movies and we want to use actors as features, then we will need to include all (famous?) actors.

4.1.2. *Step 2. Build Feature Vectors.*

We will then need to build feature vectors for the products. For instance we can have the following table for computers:

Features \ Computer	X	Y	Z
Processor	3.06	2.68	2.92
Disk Size	500	320	640
Memory	6	4	6

Each column in the table is a feature vector for a computer. It is very important to note that these features can be categorical, and also they are almost certain to have different scales. For example the unit in 'Memory' is Gigabytes, while the 'Processor'

can be their speed. For later good, we will need a way to normalize them into similar scales. There are multiple ways for normalization, we will talk about them later.

4.1.3. *Find Feature Vector for Each User.* In this step we need to combine the utility matrix and the feature vectors to build vectors for each user. This resulting matrix represents the preference of the each user to each feature. For instance let us assume the utility matrix above is the rating for each user to computers. Take the first row of the utility matrix, denote it by $\overrightarrow{UA}$. Then take the feature matrix X in Step 2. Multiple them and get

$$\vec{A} := \overrightarrow{UA} \cdot X$$

$\vec{A}$ will look like the following

Features \ User	A
Processor	5.36
Disk Size	7.9
Memory	1.0

Again, please note that we will have to normalize the feature matrix before we do this multiplication. And this step can also be done between boolean variables and ratings (For instance, movies with actors as features and user ratings) with more caution.

4.1.4. *Step 4. Recommend New Products.*

Now given a new product W, we will compute its feature vector, which is intrinsic to the product. We can then compute the *similarity* between the product and the users. One commonly used similarity metric is the cosine similarity:

$$\frac{\vec{W} \cdot \vec{A}}{||\vec{W}|| \cdot ||\vec{A}||}$$

And we will recommend W to those users with larger similarity.

4.2. **Collaborative Filtering.** This method is a quite different approach to the content based recommendation. Sometimes it is hard to determine the features across a huge variety of products, for example it is impossible to establish features for millions of merchandise on Amazon. By using Collaborative Filtering, we don't need to find features, but simply using the similarity among the group of users (or the group of products).

The basic idea is: if user A and user B have similar tastes, then it is safe to recommend to user A what user B likes.

4.2.1. *Step 1.* We need to measure the similarity among users. Several metrics is commonly applicable:

- Jaccard Distance: We convert every entry to boolean. Then use
$$dist(A, B) = \frac{A \cap B}{A \cup B}$$
- Cosine Similarity: We need to normalize each entry first, then use
$$\frac{\vec{W} \cdot \vec{A}}{||\vec{W}|| \cdot ||\vec{A}||}$$
- Pearson Correlation coefficient.

4.2.2. *Step 2.* Look at the k users that has the highest similarity to a new user U, recommend to U what the k users likes.

Remark 4.1. We want to make further remarks here. Due to symmetry, we can either use users or products to do collaborative filtering. Namely besides the approach above, we can also recommend a new product to the users who like the k most similar products to the new product. Often clustering methods are useful after comparing products. Hierarchical clustering is more often used.

Usually Item-Item will be preferred over User-User, because items are simpler to quantified.

4.3. **Technicalities.**

4.3.1. *UV-Decomposition.* One of the largest difficulty of the recommendation system is that there is usually a lot of sparsity in the utility matrix. In order to fill the null values, the UV decomposition is often used.

Goal: given $M_{n,m}$, want to find $U_{n,d}, V_{d,m}$ such that $M = UV$

(1) Preprocessing M (Normalization)
(2) Initializing U, V (set d, set U, V equal to mean of M)
(3) Compute the RMSE between M and UV as the cost function, use gradient descent to optimize the cost function. Repeat until convergence.
(4) Avoid overfitting by taking averages.

4.3.2. *Normalization Methods.* In recommendation systems we often need to normalize our vectors. There are multiple ways:

(1) $x_i - \bar{x}$
(2) $\frac{x_i - x_{min}}{x_{max} - x_{min}}$
(3) $x_i/\bar{x}$, if $\bar{x} \neq 0$
(4) $x_i/(\sum x_i)$, if $\bar{x} \neq 0$
(5) $\frac{x_i - \bar{x}}{\sqrt{\sum (x_i - \bar{x})^2}}$

4.4. **Pros and Cons.**

4.4.1. *Content based Recommendation.*

- Pros:
  (1) No need for data from other users (no cold start)
  (2) Does not suffer that much from sparsity problem
  (3) Able to recommend to users with unique taste
  (4) Able to recommend new and unpopular items
  (5) Able to provide explanations
- Cons:
  (1) Finding appropriate features is hard
  (2) Need data for new users
  (3) Over specialization (hard to introduce diversity)

4.4.2. *Collaborative Filtering.*

- Pros:
  (1) No feature selection needed.
- Cons:
  (1) Cold start problem (need enough users)
  (2) Sparsity (hard to find users that rate the same items)
  (3) First Rater (cannot recommend an item that has not been previously rated)
  (4) Popularity biased (tend to recommend popular items / cannot recommend items to users with unique taste)

5. Deep Learning

6. Coding

References

SI [SI] G. Casella, R. L. Berger, Statistics Inferences (2001) *Duxbury Press.* ISBN 0-534-24312-6.

ISLR [ISLR] G. James, D. Witten, T. Hastie, R. Tibshirani, An Introduction to Statistics Learning with Applications in R (2013) *Springer* ISBN 1461471370.

MMD [MMD] J. Leskovec, J. Ullman Mining of Massive Datasets (2014) *Cambridge University Press* ISBN 9781139924801.

Mathematics Department, Stony Brook University, Stony Brook, NY 11794-3651, USA