C3M1: Peer Reviewed Assignment

Outline:

The objectives for this assignment:

1. Apply Binomial regression methods to real data.
2. Understand how to analyze and interpret binomial regression models.
3. Flex our math skills by determining whether certain distributions are members of the exponential family.

General tips:

1. Read the questions carefully to understand what is being asked.
2. This work will be reviewed by another human, so make sure that you are clear and concise in what your explanations and answers.

In [1]:

```
# Load required libraries
library(tidyverse)
library(dplyr)
```

```
── Attaching packages ─────────────────────────────────────── tidyve
rse 1.3.0 ──

✔ ggplot2 3.3.0     ✔ purrr   0.3.4
✔ tibble  3.2.1     ✔ dplyr   1.1.2
✔ tidyr   1.0.2     ✔ stringr 1.4.0
✔ readr   1.3.1     ✔ forcats 0.5.0

── Conflicts ────────────────────────────────────────── tidyverse_co
nflicts() ──
✖ dplyr::filter() masks stats::filter()
✖ dplyr::lag()    masks stats::lag()
```

Problem 1: Binomial (Logistic) Regression

The National Institute of Diabetes and Digestive and Kidney Diseases conducted a study of 768 adult female Pima Indians living near Phoenix, AZ. The purpose of the study was to investigate the factors related to diabetes.

Before we analyze these data, we should note that some have raised ethical issues with its collection and popularity in the statistics and data science community. We should think seriously about these concerns. For example, Maya Iskandarani wrote a brief [piece (https://researchblog.duke.edu/2016/10/24/diabetes-and-privacy-meet-big-data/)](https://researchblog.duke.edu/2016/10/24/diabetes-and-privacy-meet-big-data/) on consent and privacy concerns raised by this dataset. After you familarize yourself with the data, we'll then turn to these ethical concerns.

First, we'll use these data to get some practice with GLM and Logistic regression.

```
In [2]: # Load the data
        pima = read.csv("pima.txt", sep="\t")
        # Here's a description of the data: https://rdrr.io/cran/faraway/ma
        n/pima.html
        head(pima)
```

A data.frame: 6 × 9

	pregnant	glucose	diastolic	triceps	insulin	bmi	diabetes	age	test
	<int>	<int>	<int>	<int>	<int>	<dbl>	<dbl>	<int>	<int>
1	6	148	72	35	0	33.6	0.627	50	1
2	1	85	66	29	0	26.6	0.351	31	0
3	8	183	64	0	0	23.3	0.672	32	1
4	1	89	66	23	94	28.1	0.167	21	0
5	0	137	40	35	168	43.1	2.288	33	1
6	5	116	74	0	0	25.6	0.201	30	0

1. (a) Data Cleaning? What about Data Scrubbing? Data Sterilizing?

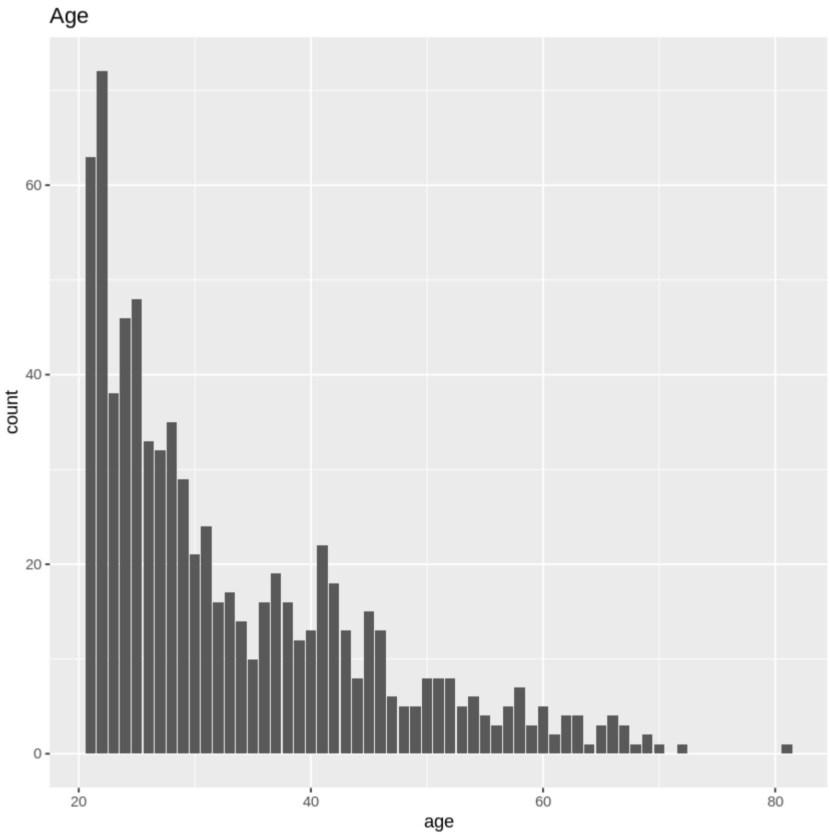
This is a real data set, which means that there's likely going to be gaps and missing values in the data. Before doing any modeling, we should inspect the data and clean it if necesary.

Perform simple graphical and numerical summaries of the data. Pay attention for missing or nonsensical values. Can you find any obvious irregularities? If so, take appropriate steps to correct these problems. In the markdown cell, specify what cleaning you did and why you did it.

Finally, split your data into training and test sets. Let the training set contain 80% of the rows and the test set contain the remaining 20%.

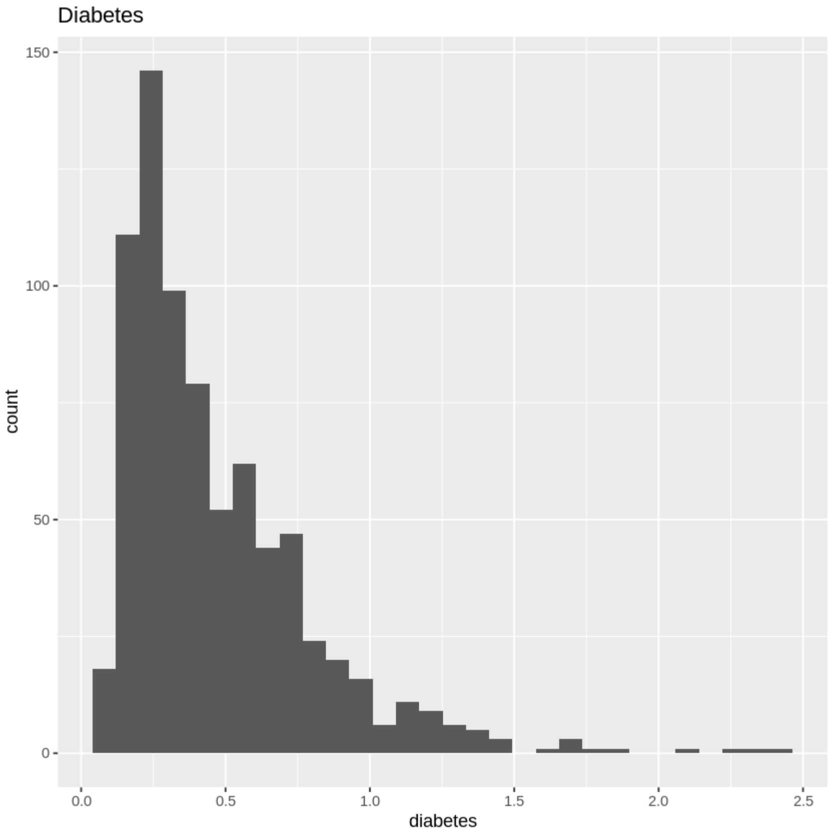
```
In [3]: # Your Code Here
        colSums(is.na(pima))
        ggplot(pima, aes(x= age)) + geom_bar()+ggtitle('Age')
```

pregnant: 0 **glucose:** 0 **diastolic:** 0 **triceps:** 0 **insulin:** 0 **bmi:** 0 **diabetes:** 0 **age:** 0 **test:** 0



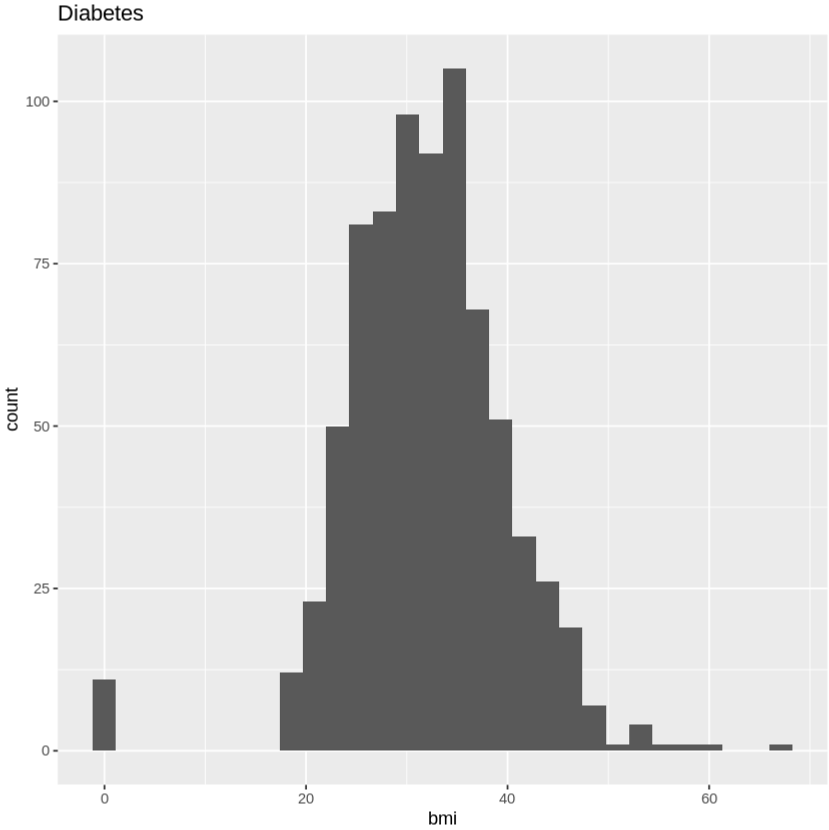
```
In [4]: ggplot(pima, aes(x= diabetes)) + geom_histogram()+ggtitle('Diabetes
        ')
```

```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



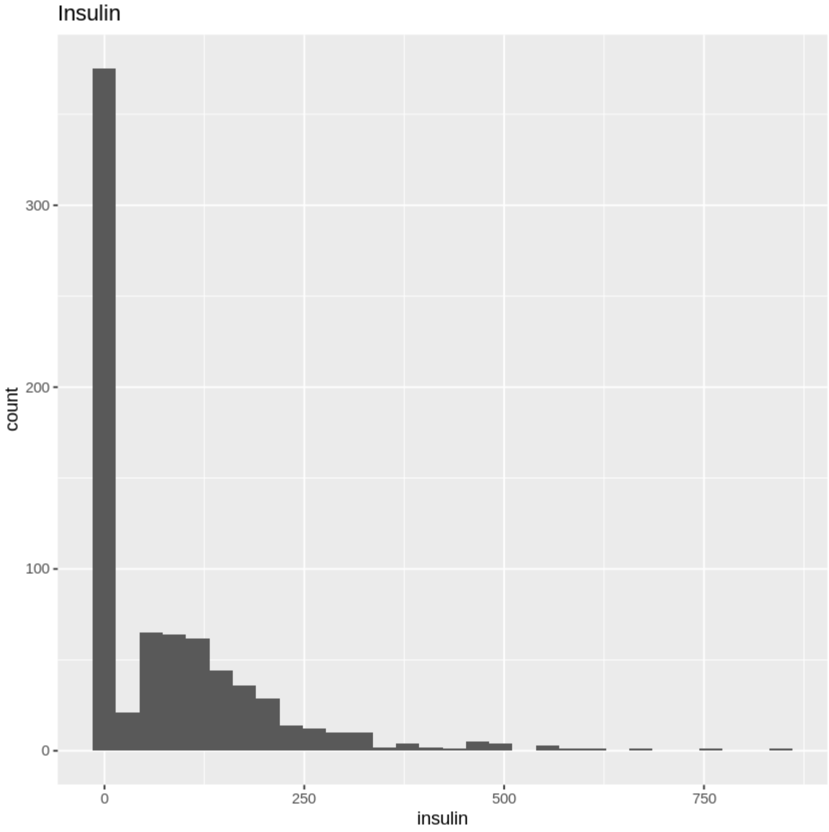
```
In [5]: ggplot(pima, aes(x= bmi)) + geom_histogram()+ggtitle('Diabetes')
```

```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
In [6]: ggplot(pima, aes(x= insulin)) + geom_histogram()+ggtitle('Insulin')
```

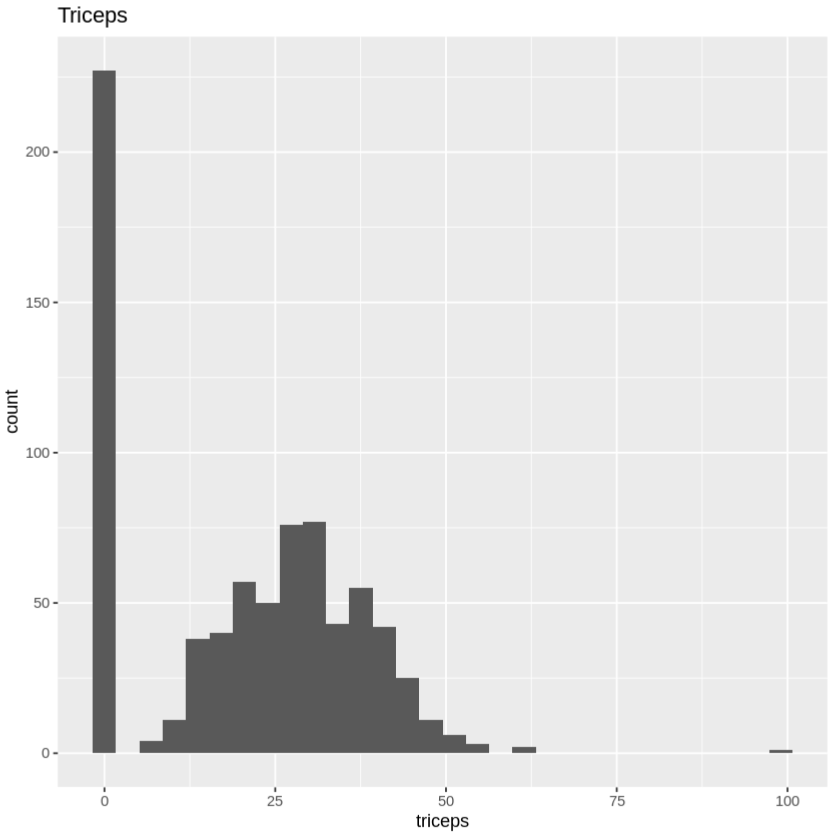
```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



In [7]:

```
ggplot(pima, aes(x= triceps)) + geom_histogram()+ggtitle('Triceps')
#these 0s seem like missing data
```

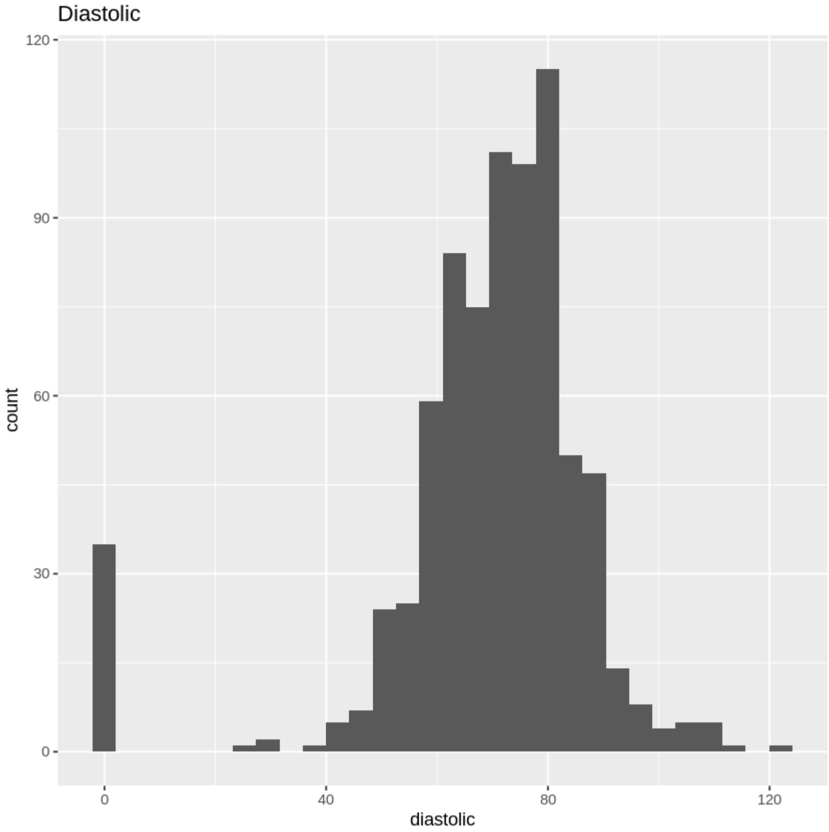
```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



In [8]:

```
ggplot(pima, aes(x= diastolic)) + geom_histogram()+ggtitle('Diastolic')
# i don't think a diastolic blood pressure of 0 is plausible--it is a possibility, but like that person should be in the ER right away, let's turn this into the mean
```

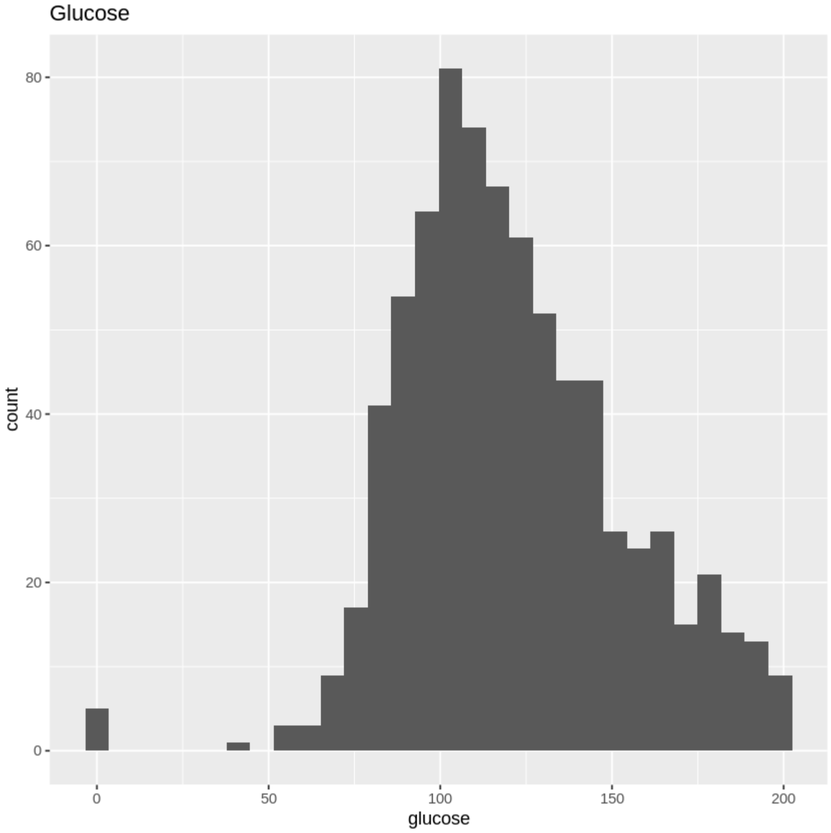
```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



In [9]:

```
ggplot(pima, aes(x= glucose)) + geom_histogram()+ggtitle('Glucose')
#again a glucose of 0 is like crashing in the hospital, need to fix
that
```

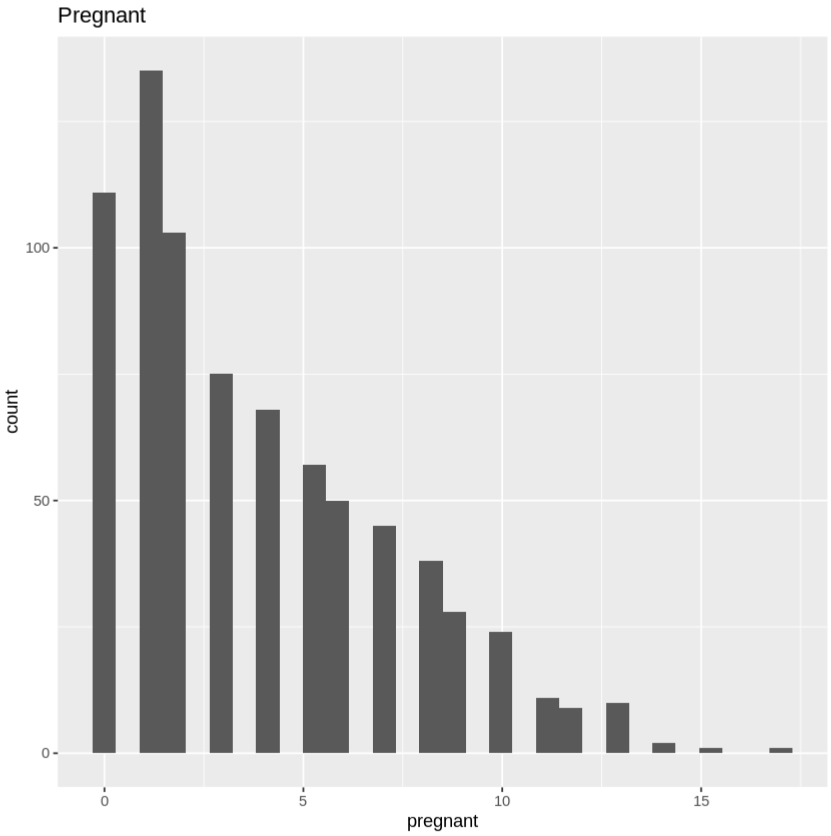
```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



In [10]:

```
ggplot(pima, aes(x = pregnant))+ geom_histogram() + ggtitle('Pregnant')
```

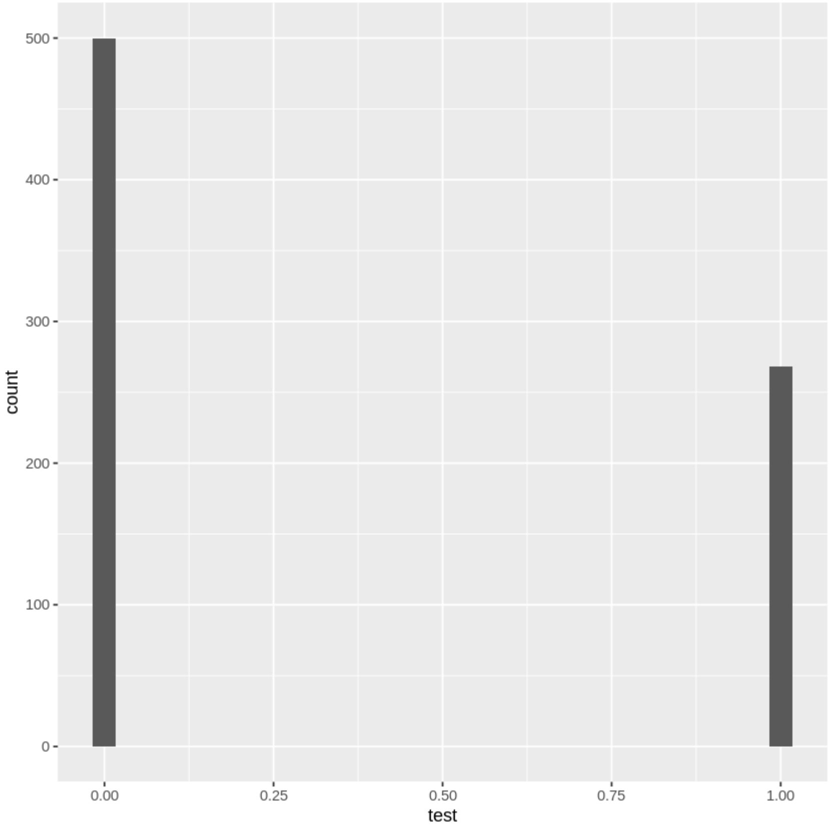
```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



In [11]:

```
ggplot(pima,(aes(x = test))) + geom_histogram()
```

```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



In [12]:

```
bad_cols = c('glucose', 'diastolic', 'triceps', 'bmi', 'insulin')
pima[bad_cols][pima[bad_cols] == 0] = NA
pima = na.omit(pima)
pima = pima %>% mutate(test = as.factor(test))
```

for cleaning, I looked for nas, and then went through each category to find nonsensical values. For glucose, diastolic, and bmi, I replaces 0s with the mean of the dataset, since 0 was not a meaningful value for these categories, and probably represents something more like na.

In [13]:

```
sample <- sample(c(TRUE, FALSE), nrow(pima), replace=TRUE, prob=c
(0.8,0.2))
train_set  <- pima[sample,]
test_set   <- pima[!sample,]
```

1. (b) Initial GLM modelling

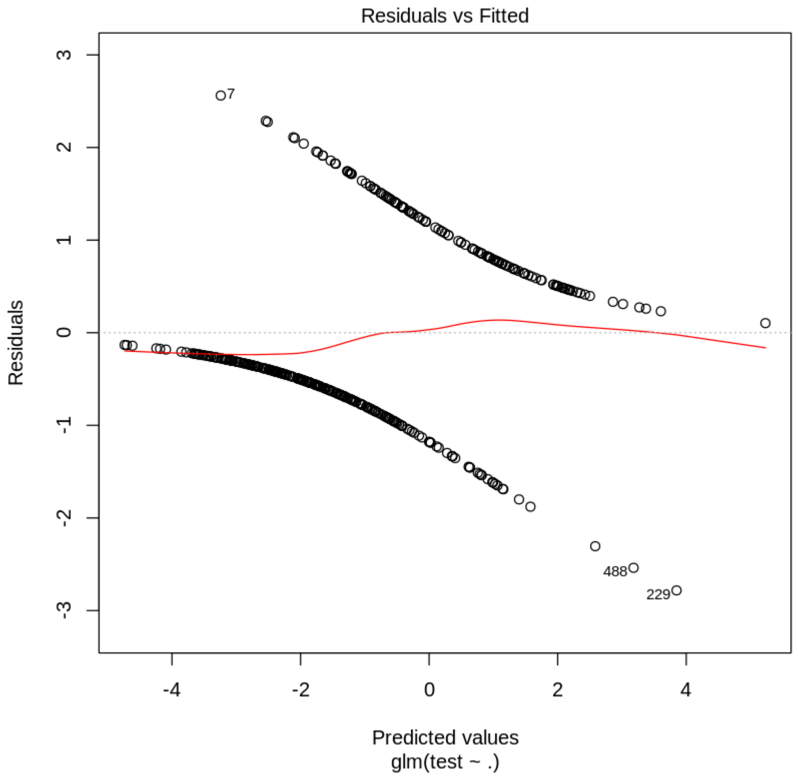
Our data is clean and we're ready to fit! What kind of model should we use to fit these data? Notice that the `test` variable is either 0 or 1, for whether the individual tested positive for diabetes. Because `test` is binary, we should use logistic regression (which is a kind of binomial regression).

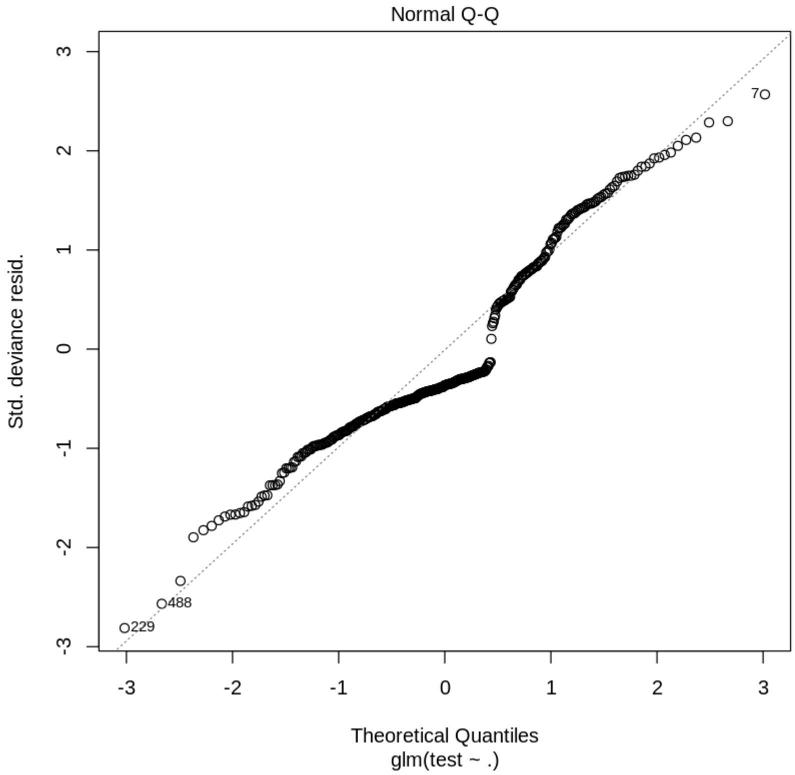
Fit a model with `test` as the response and all the other variables as predictors. Can you tell whether this model fits the data?

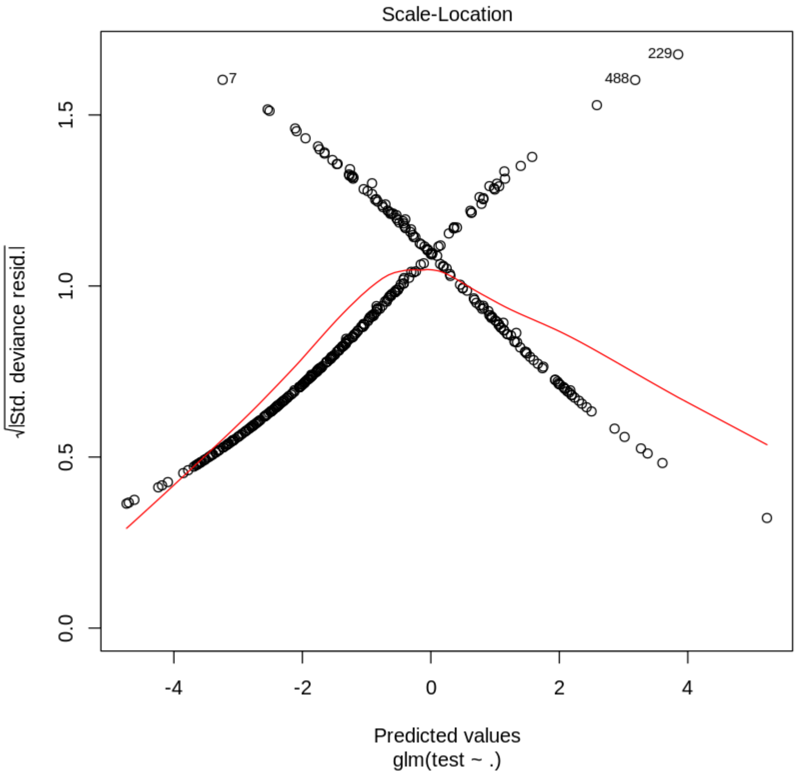
In [14]:

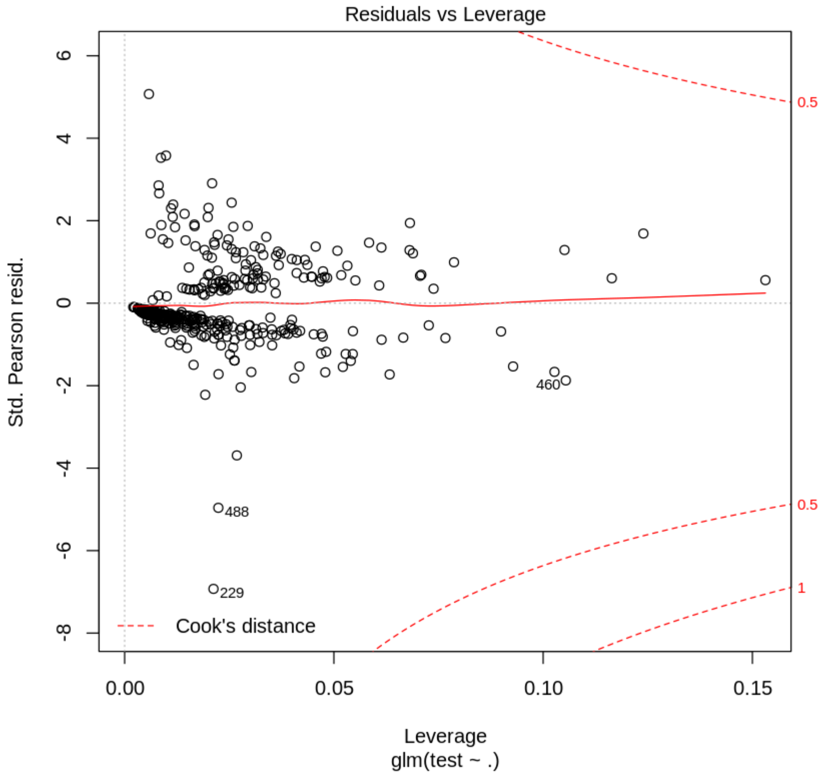
```
glmod = glm(test ~., data = pima, family = 'binomial')
summary(glmod)
# Your Code Here
plot(glmod)
```

```
Call:
glm(formula = test ~ ., family = "binomial", data = pima)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.7823  -0.6603  -0.3642   0.6409   2.5612

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.004e+01  1.218e+00  -8.246  < 2e-16 ***
pregnant     8.216e-02  5.543e-02   1.482  0.13825
glucose      3.827e-02  5.768e-03   6.635 3.24e-11 ***
diastolic   -1.420e-03  1.183e-02  -0.120  0.90446
triceps      1.122e-02  1.708e-02   0.657  0.51128
insulin     -8.253e-04  1.306e-03  -0.632  0.52757
bmi          7.054e-02  2.734e-02   2.580  0.00989 **
diabetes     1.141e+00  4.274e-01   2.669  0.00760 **
age          3.395e-02  1.838e-02   1.847  0.06474 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 498.10  on 391  degrees of freedom
Residual deviance: 344.02  on 383  degrees of freedom
AIC: 362.02

Number of Fisher Scoring iterations: 5
```




Normal Q-Q
Std. deviance resid.
3
2
1
0
-1
-2
-3
-3
-2
-1
0
1
2
3
7
488
229
Theoretical Quantiles
glm(test ~ .)


Scale-Location
√|Std. deviance resid.|
1.5
1.0
0.5
0.0
-4
-2
0
2
4
7
229
488
Predicted values
glm(test ~ .)



it is difficult to say if the model fits well given this set of tests; for a binomial, these things are not as predictive as they for linear regression. Probably a good way to go would be to look at accuracy/precision on a test set to see if that's a better fit.

1. (c) Remember Bayes

A quick analytical interlude.

Is diastolic blood pressure significant in the regression model? Do women who test positive have higher diastolic blood pressures? Explain the distinction between the two questions and discuss why the answers are only apparently contradictory.

In [15]:

```
dialm = lm(diastolic ~ test, data = train_set)
summary(dialm)
# Your Code Here
```

```
Call:
lm(formula = diastolic ~ test, data = train_set)

Residuals:
    Min      1Q  Median      3Q     Max 
-45.112  -7.579   0.654   8.888  36.888 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)   69.112      0.877  78.805  < 2e-16 ***
test1          4.467      1.498   2.983  0.00308 ** 
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 12.56 on 310 degrees of freedom
Multiple R-squared:  0.0279,	Adjusted R-squared:  0.02477 
F-statistic: 8.898 on 1 and 310 DF,  p-value: 0.003081
```

diastolic blood pressure is not statistically significant in the model; but it is significantly correlated with a positive test in this model. The distinction is that that case is diastolic pressure not conditioned on a positive test and this case is diastolic pressure conditioned on positive test, different entities in Bayes theorem.

1. (d) GLM Interpretation

We've seen so many regression summaries up to this point, how is this one different from all the others? Well, to really understand any model, it can be helpful to loop back and plug the fitted results back into the model's mathematical form.

Explicity write out the equation for the binomial regression model that you fit in (b). Then, in words, explain how a 1 unit change of `glucose` affects `test` , assuming all other predictors are held constant.

In [16]:

```
# Your Code Here
```

$$\log \frac{p}{1-p} = \beta_0 + \beta_1 pregnant + \beta_2 glucose + \beta_3 diastolic + \beta_4 triceps + \beta_5 insulin + \beta_6 bmi$$
$$+ \beta_7 diabetes + \beta_8 age$$

$$\log \frac{p}{1-p} = -10 + 0.08X_{pregnant} + 0.04X_{glucose} - 0.001X_{diastolic} - 0.01X_{triceps}$$
$$- 0.0008X_{insulin} + 0.07X_{bmi} + 1.145X_{diabetes} + 0.03X_{age}$$

a 1 unit change in gluocose would increase the odds of test being 1 by $e^{0.035}$, which is roughly 1.035

1. (e) GLM Prediction

One of the downsides of Logistic Regression is that there isn't an easy way of evaulating the goodness of fit of the model without predicting on new data. But, if we have more data to test with, then there are many methods of evaluation to use. One of the best tools are confusion matrices, which (despite the name) are actually not that hard to understand.

A confusion matrix compares the predicted outcomes of a Logistic Regression Model (or any classification model) with the actual classifications. For binary classification, it is a 2×2 matrix where the rows are the models' predicted outcome and the columns are the actual classifications. An example is displayed below.

	True	**False**
1	103	37
0	55	64

In the example, we know the following information:

- The [1,1] cell is the number of datapoints that were correctly predicted to be 1. The value (103) is the number of True Positives (TP).
- The [2,2] cell is the number of datapoints that were correctly predicted to be 0. The value is the number of True Negatives (TN).
- The [1, 2] cell is the number of datapoints that were predicted to be 1 but where actually 0. This is the number of False Positives (FP), also called Type I error. In the context of our diabetes dataset, this would mean our model predicted that the person would have diabetes, but they actually did not.
- The [2, 1] cell is the number of datapoints that were predicted to be 0 but where actually 1. This is the number of False Negatives (FN), also called Type 2 error. In the context of our diabetes dataset, this would mean our model predicted that the person would not have diabetes, but they actually did have diabetes.

Use your model to predict the outcomes of the test set. Then construct a confusion matrix for these predictions and display the results.

In [33]:

```
preds = ifelse(predict.glm(glmod, type = "response", test_set, na.r
m = TRUE) > 0.5, 1, 0)
tn = sum(pr == 0 & as.numeric(levels(test_set$test))[test_set$test]
== 0);
tp = sum(pr == 1 & as.numeric(levels(test_set$test))[test_set$test]
== 1);
fp= sum(pr == 1 & as.numeric(levels(test_set$test))[test_set$test]
== 0);
fn= sum(pr == 0 & as.numeric(levels(test_set$test))[test_set$test]
== 1);

confusion_matrix = table(test_set$test, preds)
colnames(confusion_matrix) = c(FALSE, TRUE)
confusion_matrix
```

```
   preds
    FALSE TRUE
  0    51    6
  1     9   14
```

1. (f) Evaluation Statistics

Using the four values from the confusion matrix, we can construct evaulation statistics to get a numerical approximation for our model's performance. Spend some time researching accuracy, precision, recall and F score.

Calculate these values for your model's predictions on the test set. Clearly display your results. How well do you think your model fits the data?

In [50]:

```
# Accuracy
cat("The Accuracy is :", format((51+14)/(51+14+6+9)))
# Your Code Here
```

```
The Accuracy is : 0.7565789
```

In [37]:

```
cat("The Precision is :", format((tp)/(tp+fp)))
```

```
The Precision is : 0.7
```

In [38]:

```
cat("The Recall is : ", format(tn/(tn+fp)))
```

```
The Recall is :  0.8947368
```

In [39]:

```
cat("The F Score is : ", format(2 * ((0.7 *0.8947368)/(0.7 +0.89473
68))))
```

```
The F Score is :  0.7854785
```

I think the model does a decent job at accuracy.It does a bit better on recall than precision. One of the issues is that the model is not totally balanced in its outcomes, so you end up with some weirdness in metrics.

1. (g) Understanding Evaluation Statistics

Answer the following questions in the markdown cell below.

1. Give an example scenario for when accuracy would be a misleading evaulation statistic.
2. Confusion matrices can also be used for non-binary classification problems. Describe what a confusion matrix would look like for a response with 3 levels.
3. You'll have to take our word on the fact (or spend some time researching) that Type I error and Type II error are inversely related. That is, if a model is very good at detecting false positives, then it will be bad at detecting false negatives. In the case of our diabetes dataset, would you prefer a model that overestimates the Type 1 error or overestimates the Type II error. Justify your answer.

1. Say you had a binary dataset that looked at risks for a very rare cancer in a population of lab mice, of which say 10/1000 had this cancer, and 990/1000 did not. The model trained on the dataset is very accurate, but it has a really high true negative rate, and misses almost all of the cases it is trying to detect, as it has a very low true positive rate. This model would be quite accurate, but would not solve the problem that the model is attempting to solve.
2. A 3 level confusion matrice would be fairly similar, but it would just extend the logic, so it would have the accurate matches between the different categories at different levels, and each row would have the true matches between a factor, and then the counts of the two mismatched factors.
3. You can sort of see the Type 1/Type 2 issues with the precision/recall scores in this model. The precision is high and the recall is fairly low, so there's a lot more type 2 error in this model, inasmuch as it does a poor job on weeding out false negatives. I think for the diabetes treatment question, because of the potential fatal harms of treating non-diabetics with insulin, we should definitely prefer a much lower type 2 rate than a type 1 rate.

1. (h) Ethical Issues in Data Collection

Read Maya Iskandarani's [piece (https://researchblog.duke.edu/2016/10/24/diabetes-and-privacy-meet-big-data/)](https://researchblog.duke.edu/2016/10/24/diabetes-and-privacy-meet-big-data/) on consent and privacy concerns raised by this dataset. Summarize those concerns here.

First, there's the broader ethical problem of observing data, and specifically not letting indigenous tribes control their own data and tell their own story. The kind of usage of this data is very invasive, and speaks to a sort of broader set of distrust/issues around the relationships between government agencies and the tribes they observe. The point Iskandarani brings up is can the tribes have truly consented to this usage; over 40 years of data were collected when only ten was actually signed off on, and the uploading of this data into one of the main ML repositories means that health data of the Pima Tribe is now used by data analysts and scientists all over the world, and thus, could they have really accepted such terms, when they could not possibly have known that this would happen. No medical researcher could have given the tribe this info, and the tribe could not possibly have consented to this.

Problem 2: Practicing those Math skills

One of the conditions of GLMs is that the "random component" of the data needs to come from the Exponential Family of Distributions. But how do we know if a distribution is in the Exponential Family? Well, we could look it up. Or we could be proper mathematicians and check the answer ourselves! Let's flex those math muscles.

2. (a) But it's in the name...

Show that $Y \sim exponential(\lambda)$, where λ is known, is a member of the exponential family.

$Y \sim exponential(\lambda)$ has a probability distribution of $\lambda e^{-\lambda x}$

So to prove that it is a member of the exponential family, we need to prove that it can be written in the form:

$f(y; \theta, \phi) = exp\{\frac{y(\theta)-b(\theta)}{a(\theta)} + c(y, \theta)\}$

$exp\{log\lambda * loge^{-\lambda x}\}$

These simplify to give us: $exp\{log\lambda - \lambda * x\}$

Rearranging these, we get $exp\{-\lambda x + log(\lambda)\}$, which is one of the forms of the exponential family

In this case, $-\lambda x$ is θ, and $log(\lambda)$ is $c(y, \phi)$

2. (b) Why can't plants do math? Because it gives them square roots!

Let $Y_i \sim exponential(\lambda)$ where $i \in \{1, \dots, n\}$. Then $Z = \sum_{i=1}^{n} Y_i \sim Gamma(n, \lambda)$. Show that Z is also a member of the exponential family.

since we've already proved that the exponential distribution is part of the exponential family, the goal here is to show that the gamma distribution is in the exponential family, thus proving that Z is.

the PMF of the Gamma Distribution is $f(x|n, \lambda) = \frac{1}{\gamma(n)\lambda^n} x^{n-1} exp^{-\frac{x}{\lambda}}$

putting this to the exponential results in $exp\{\frac{1}{\gamma(n)\lambda^n} x^{n-1} exp^{-\frac{x}{\lambda}}\}$

To simplify that a little, we get $exp(nlog(x) - nlog(\lambda) - log(\gamma(n)) * exp^{-\frac{x}{\lambda}})$

We can align this with the exponential distribution, where θ is equal to $log(x)$ and $c(y, \phi)$ is equal to $-nlog(\lambda) - log(\gamma(n))$

```
In [ ]:
```