Risk of developing Type II Diabetes

Using Data Science models, I'm attempting to predict the risk of developing Type 2 diabetes given an Individuals vital health signs.

- Karun Siddana

- August 24th, 2015

Problem Statement

- There is an increasing growing problem of obesity and hypertension in the United States that leads to Type 2 diabetes. A sure shot test for type 2 diabetes has not been developed due to the complexities in the disease.
- Unlike Type 1 diabetes which can be detected at birth, type 2 diabetes is dependent on the individuals lifestyle, exercise component, nutrition and genetic makeup.
- Early detection of the indicators of Type 2 diabetes would allow us to avoid bad food habits and prevent/mitigate the risk of developing type 2 diabetes.

Data Collection

- The Data used was from the Department of Biostatistics at the Vanderbilt University
- http://biostat.mc.vanderbilt.edu/wiki/Main/DataSets
- http://cs-people.bu.edu/dgs/courses/cs105/hall_of_fame/awm.html

Data

- Using the following predictors, I am trying to predict the outcome, whether a person is positive for diabetes.
  - Total Cholesterol Level (chol, hdl, ldl)
  - Blood Glucose Level (stab.glu)
  - HDL
  - Age
  - Gender
  - Body Frame
  - Systolic Blood Pressure levels (bp.1s, bp.1d)
  - Waist Size
  - Hip Size
- Glycosylated Hemoglobin (glyhb) – Response/Output

Data Preparation

- Lots of missing data for the Response Value.
- I searched online, for formulas that could help determine a Glycosylated Hemoglobin (glyhb) – Response/Output based on the average Glucose Level.
- Reference: http://professional.diabetes.org/glucosecalculator.aspx
  relationship between eAG and A1C/glyhb:
  28.7 X A1C – 46.7 = eAG

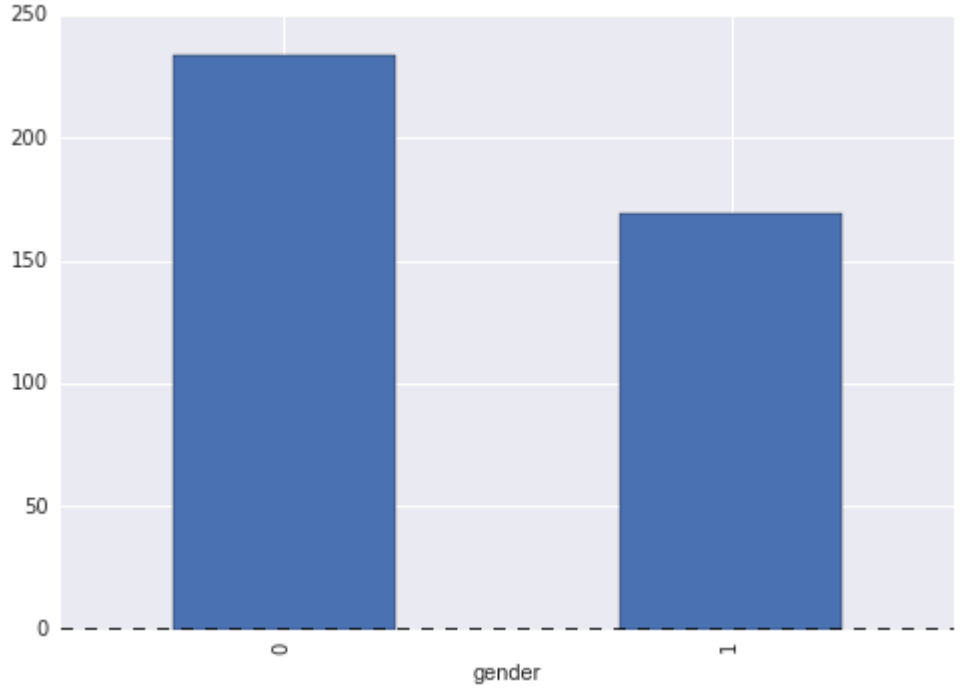
Data Preparation

- Lots of missing data for the Response Value.
- I searched online, for formulas that could help determine a Glycosylated Hemoglobin (glyhb) – Response/Output based on the average Glucose Level.
- Reference:
  http://professional.diabetes.org/glucosecalculator.aspx
  relationship between eAG and A1C:
  28.7 X A1C – 46.7 = eAG
- Generated my own formula, simple formula based on Linear Regression and used y = mx + c.
- y = 0.028*x + 2.453 (where x = stab.glu, y = glyhb)

Data Exploration

- Firstly I plotted several graphs to understand the correlation between the indicators and the response.
- Total Data Points
  - Total Patients = 403
- What kind of data do I have here?
  - Male: 169
  - Female: 234

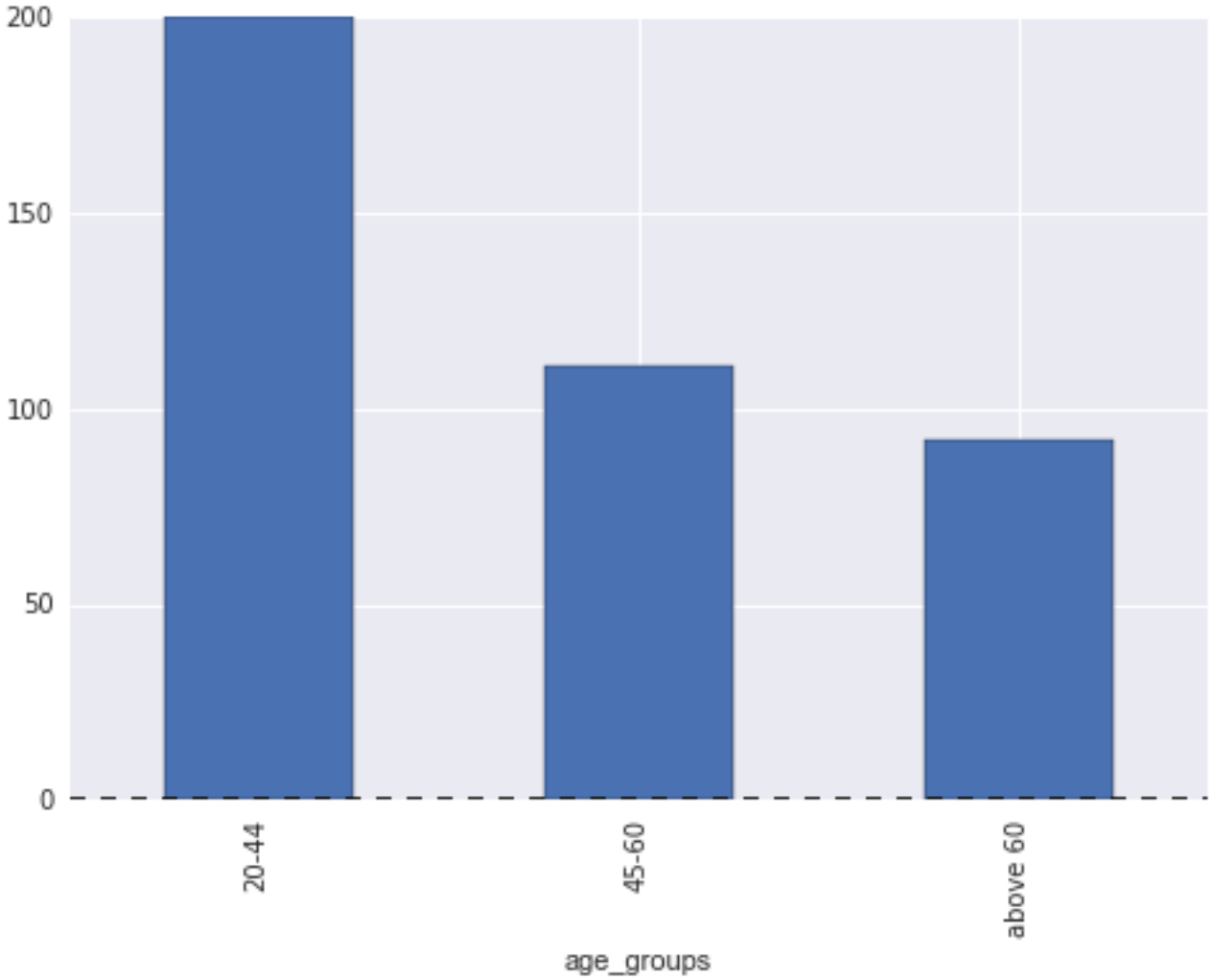
Data Exploration



Gender:

female	234
male	169

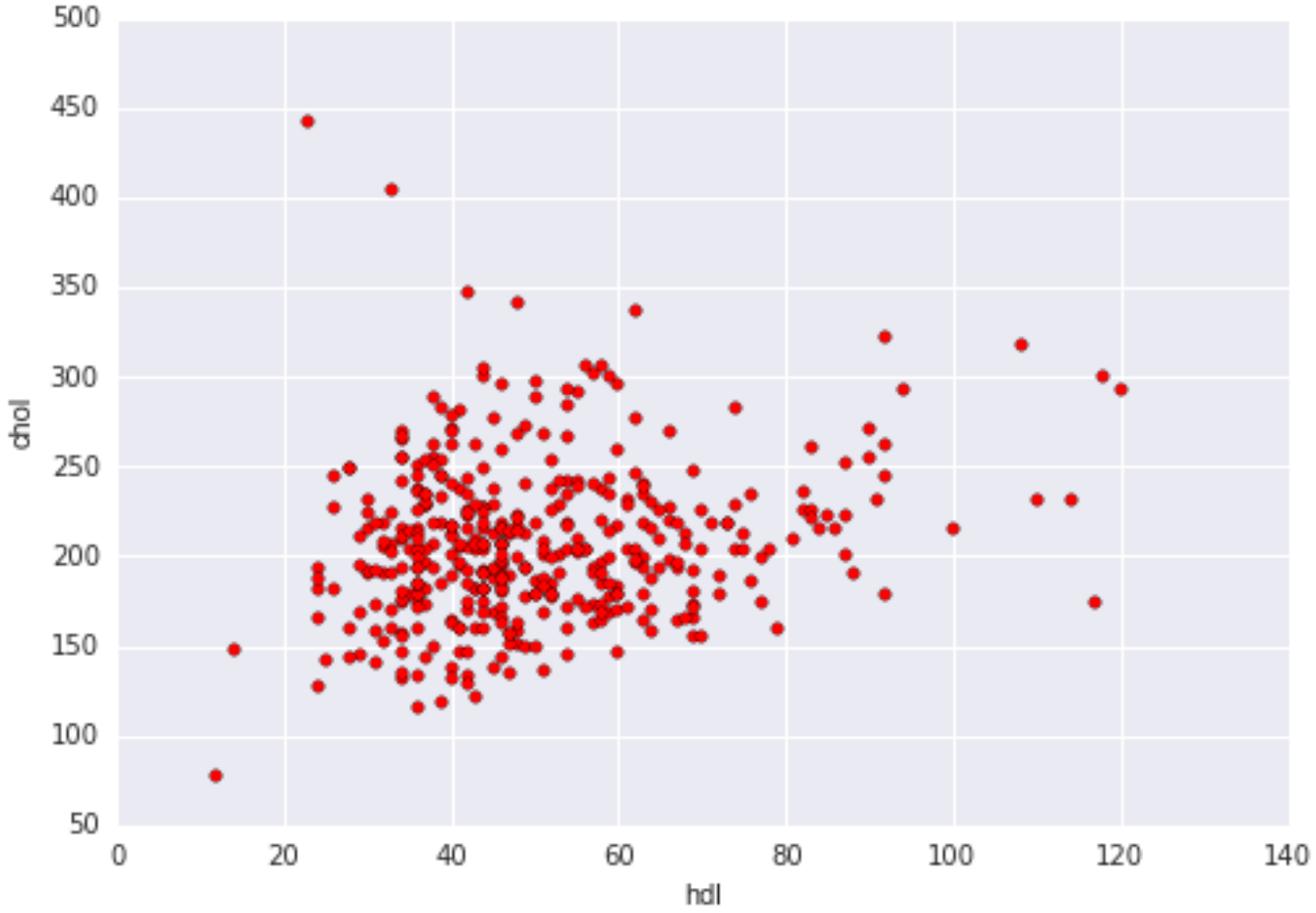
Data Exploration



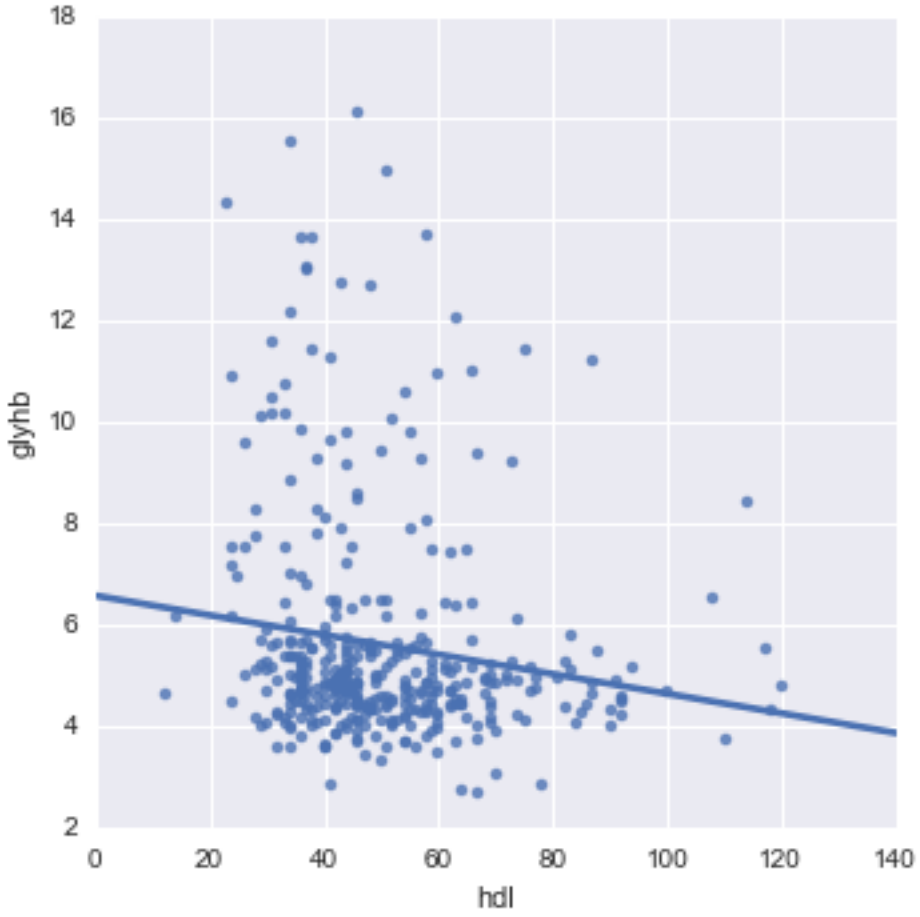
age_groups:

20 – 44	200
45 – 60	111
Above 60	92

Plot – *Scatter Plot (hdl vs chol)*

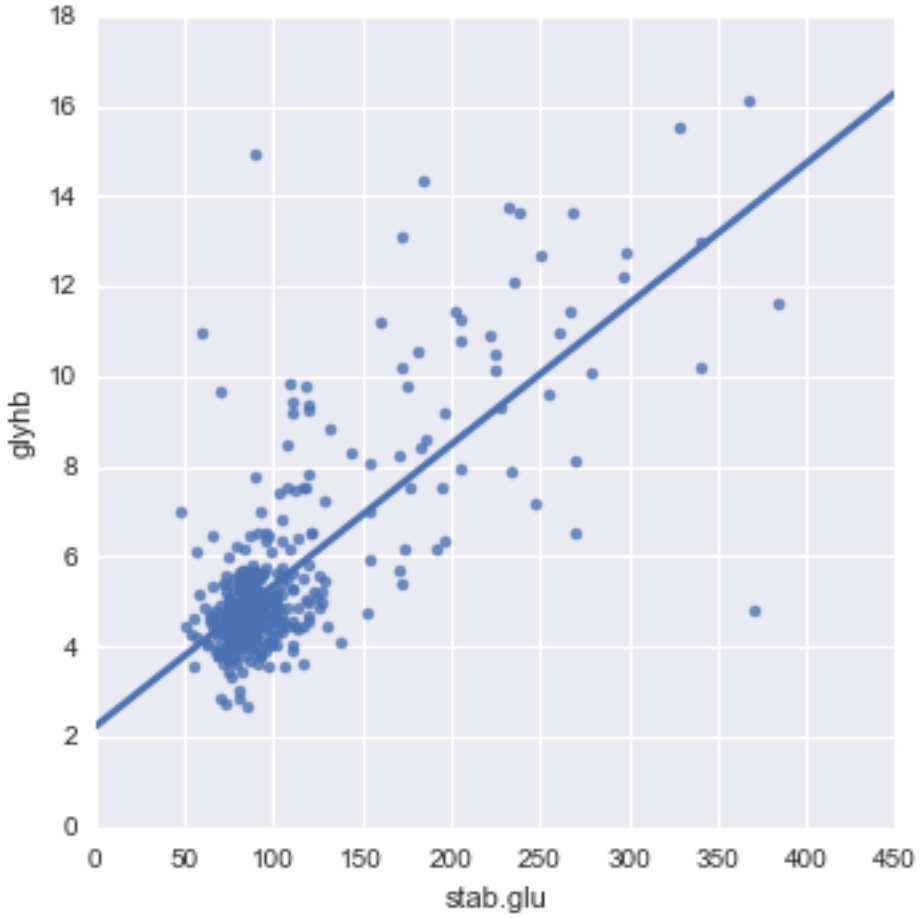


Plot – *hdl* vs *glyhb*



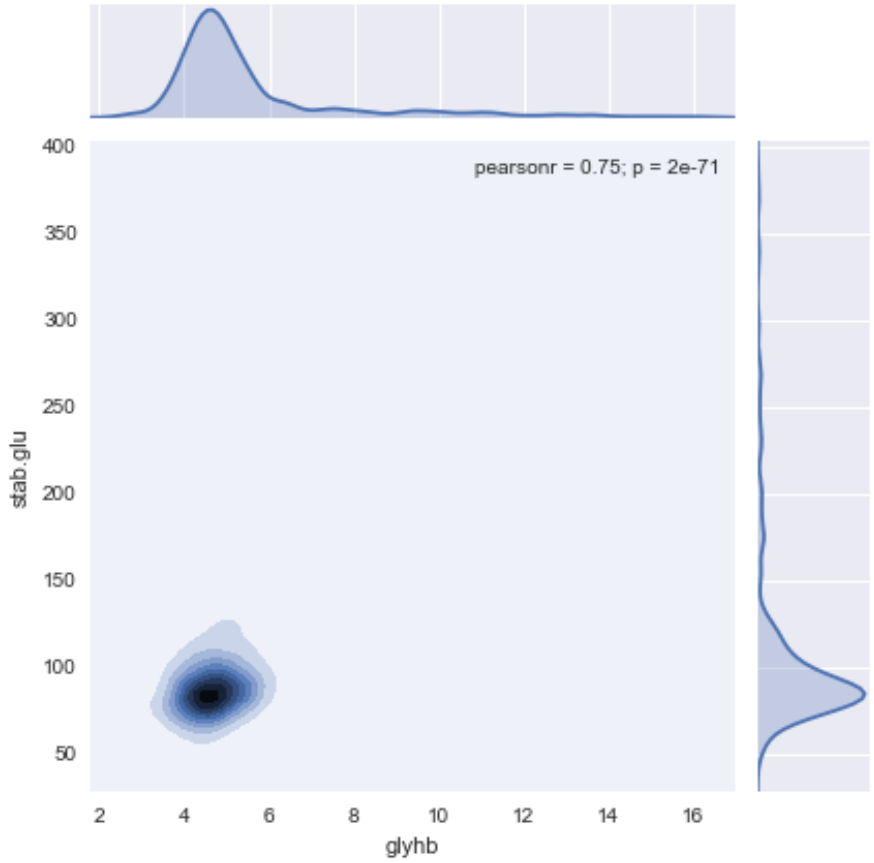
X-axis: High Cholesterol
Y-axis: Glycosylated Hemoglobin

Plot – *stab.glu x glyhb*



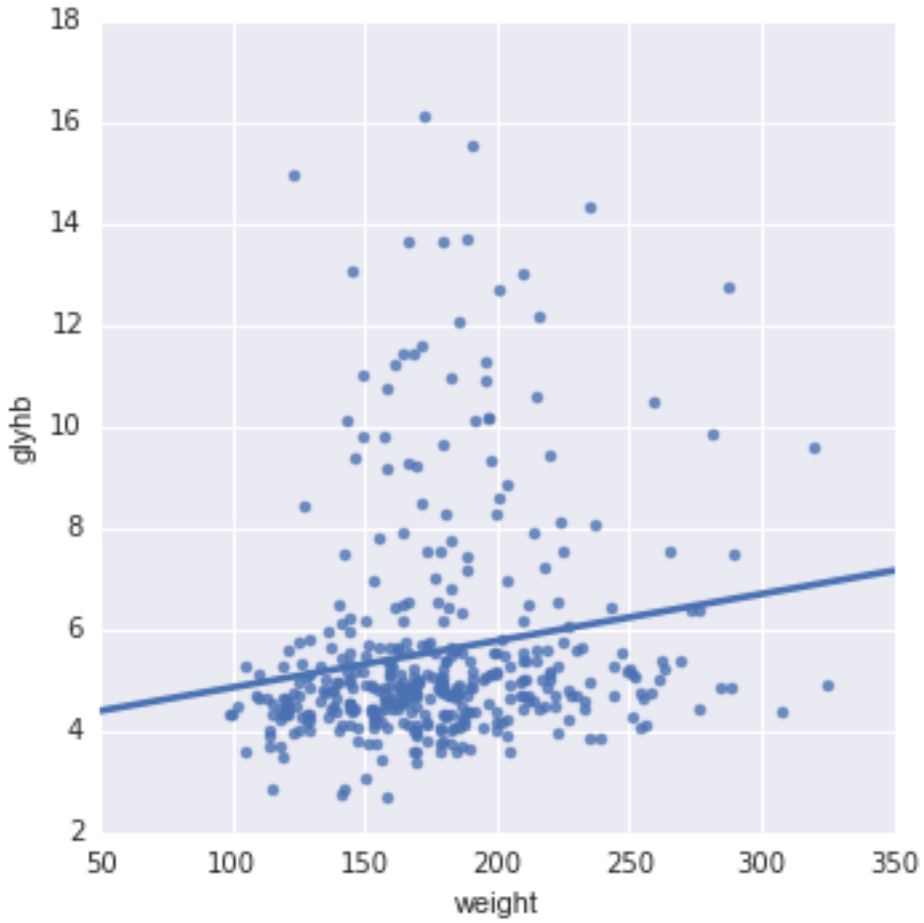
X-axis: Blood Glucose Level
Y-axis: Glycosylated Hemoglobin

Kernel Density Plot – *hdl* x *glyhb*



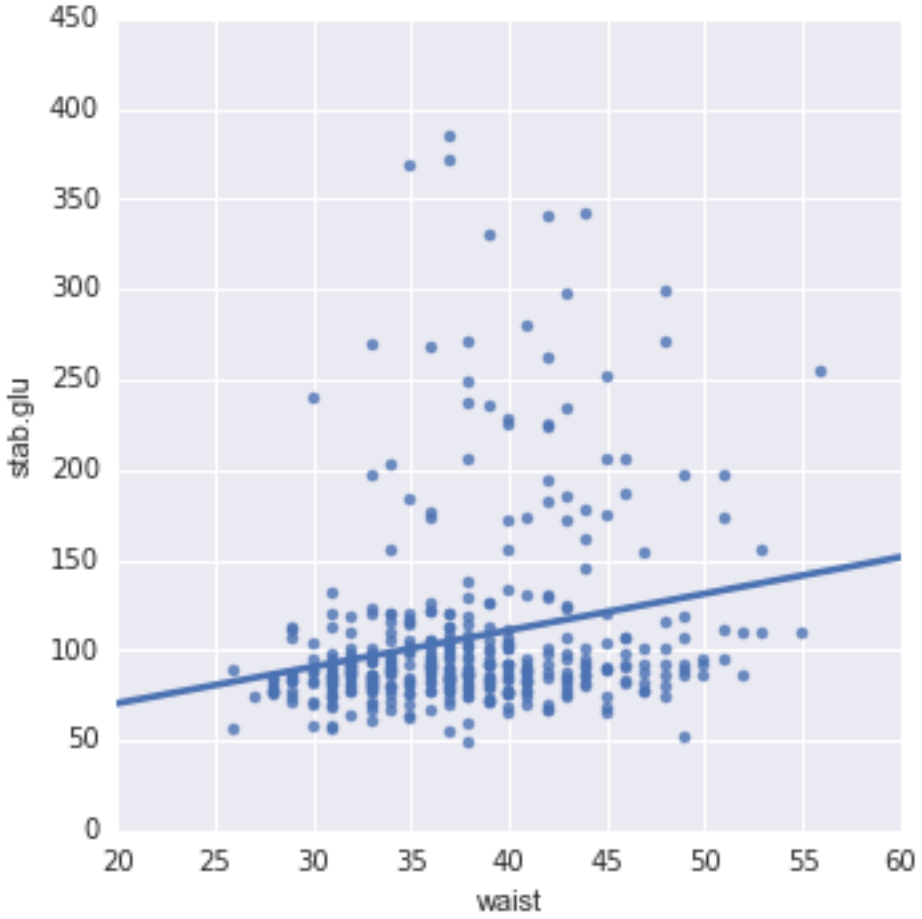
X-axis: Glycosylated Hemoglobin
Y-axis: Blood Glucose Level

sns.lmplot– *weight* x *glyhb*



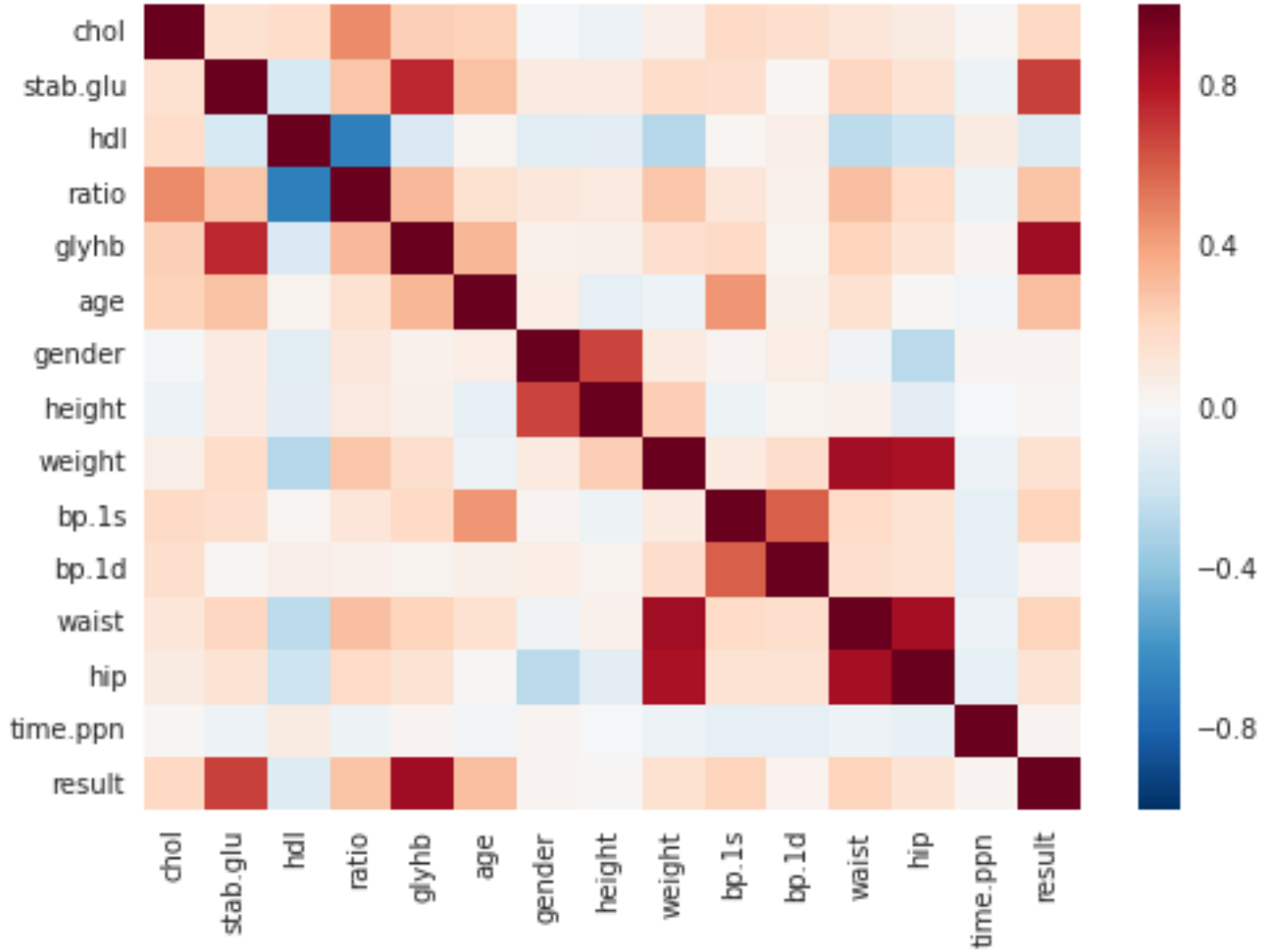
X-axis: Weight
Y-axis: Glycosylated Hemoglobin

sns.lmplot– *waist* x *stab.glu*

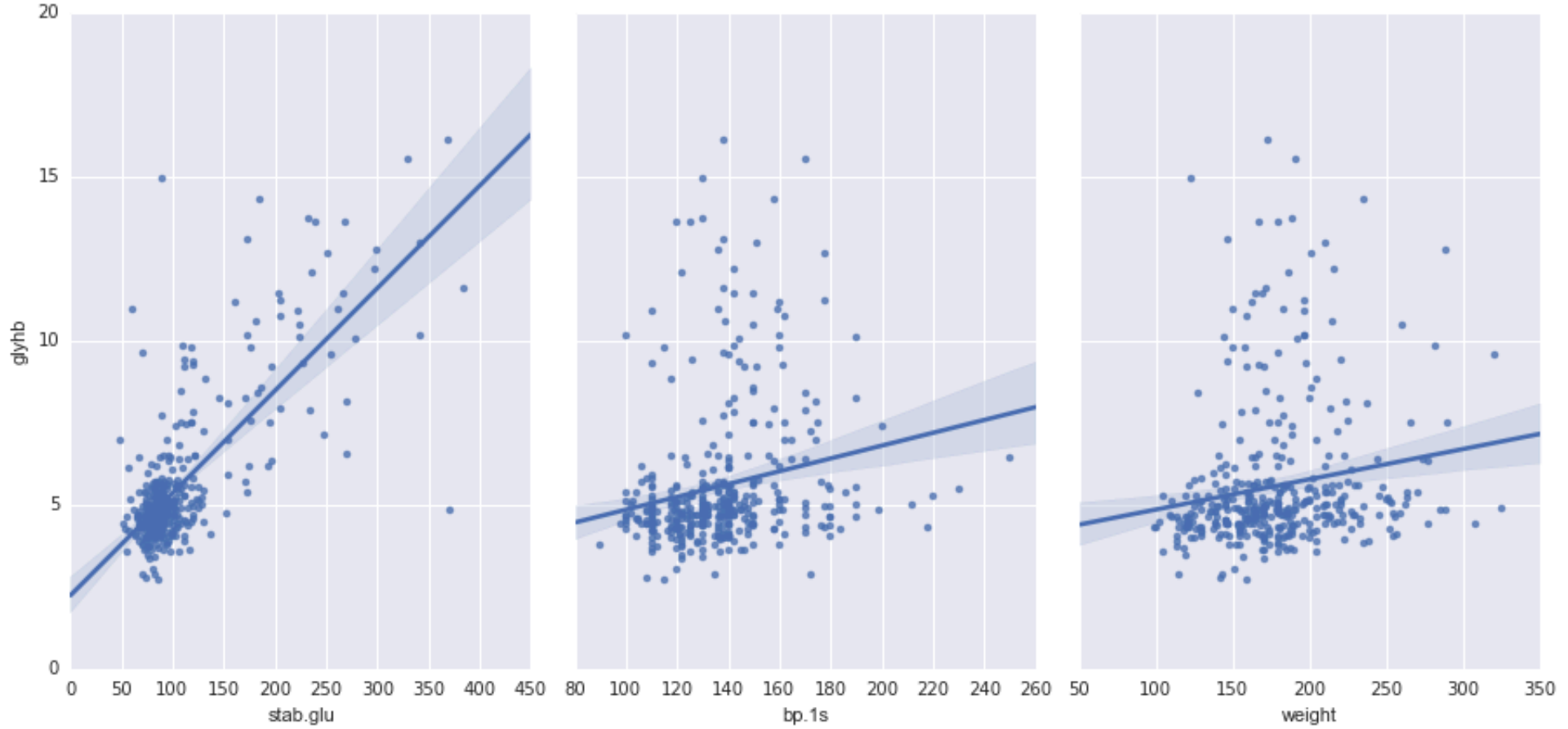


X-axis: waist
Y-axis: Blood Glucose Level

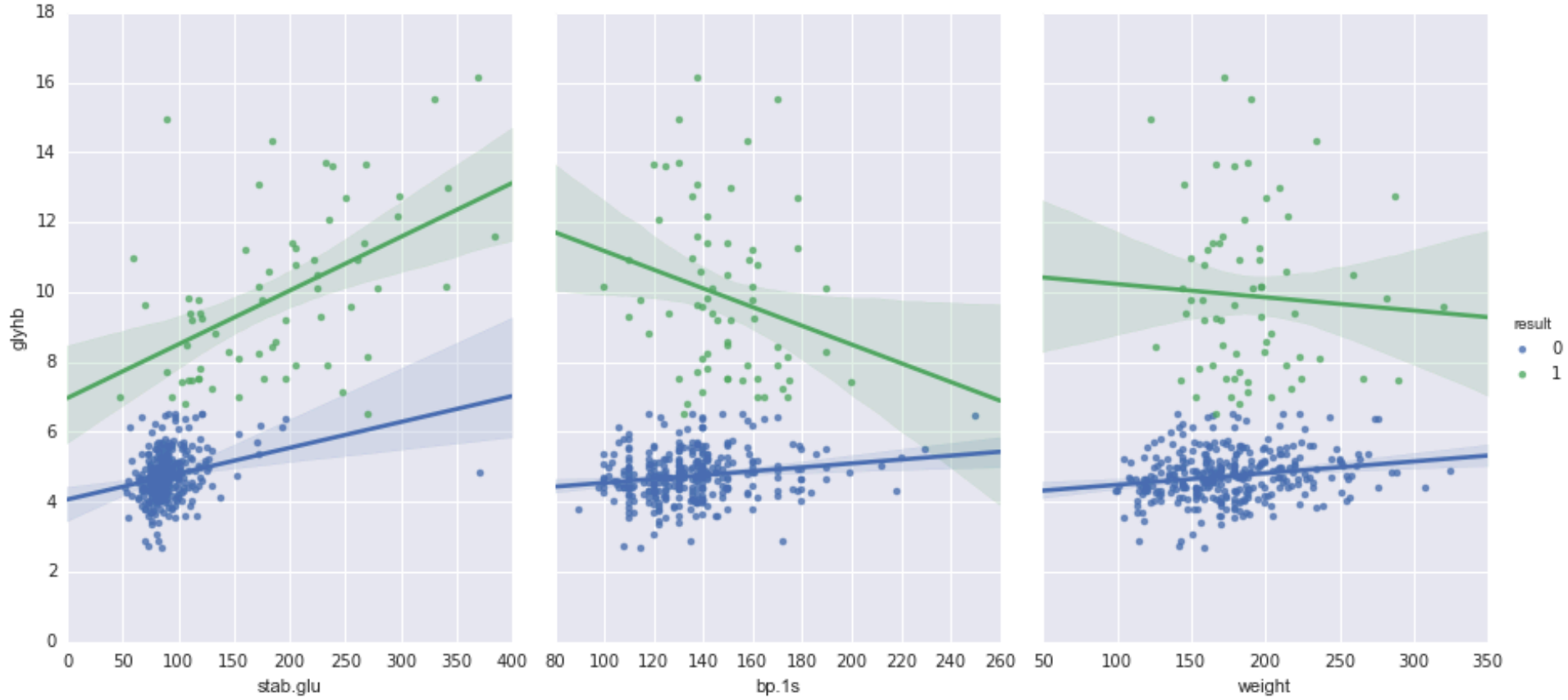
Heatmap


chol
stab.glu
hdl
ratio
glyhb
age
gender
height
weight
bp.1s
bp.1d
waist
hip
time.ppn
result
0.8
0.4
0.0
−0.4
−0.8

sns.pairplot



sns.pairplot



Logistic Regression

- features = ['stab.glu']
  x = diabetes_data[features]
  y = diabetes_data['glyhb']
  y_category = np.where(y > 6.5, 1, 0)

- Fit the model to a X_train and y_train
  - [('stab.glu', 0.027)]
  - Confusion Matrix

    [86 0]
    [7 8]

  - Accuracy = 0.91 or 91%
  - Specificity = 1.0
  - Sensitivity = 0.60

Linear Regression Model

- features = ['stab.glu', 'weight']

  x = diabetes_data[features]
  y = diabetes_data['glyhb']

- Train Test Split / Fit the Linear Model
  [('stab.glu', 0.028),
  ('weight', 0.0021)]

- Mean Absolute Error = 0.91

- Mean Squared Error = 1.43
  (Remember the scale of the Output it is between 3.5 – 15.0)

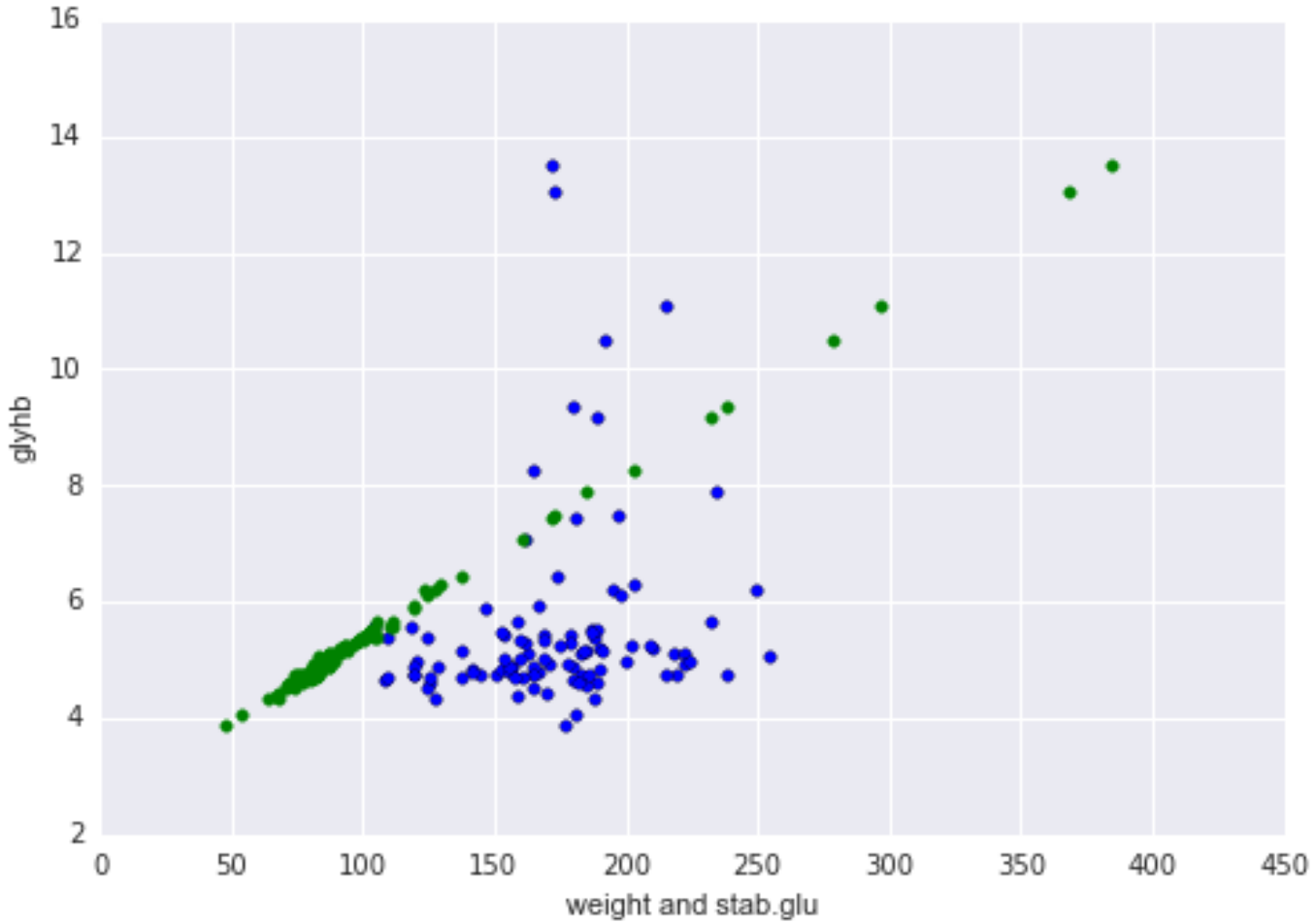
Linear Regression Model

- Cross Validation Score – Mean Squared Error
  -3.67
  -2.02
  -0.76
  -3.60
  -0.55

Linear Regression Model

- Cross Validation Score – Mean Squared Error
  -3.67
  -2.02
  -0.76
  -3.60
  -0.55
- This is Terrible !!!!
- Very large Mean Squared Error for an output that ranges between 3.5 – 15.

Linear Regression Model



Stats Linear Regression Model

- lm = smf.ols(formula='glyhb ~ avg_glu + bp_1s + bp_1d + chol + weight + age + waist + hip + gender + hdl + ratio', data=diabetes_data).fit()

- P- Values:

stab.glu	5.86E-59
bp_1s	5.00E-01
bp_1d	5.68E-01
chol	2.75E-01
weight	7.62E-01
age	1.99E-02
waist	7.80E-01
hip	8.07E-01
gender	4.25E-01
hdl	9.26E-01
ratio	3.12E-01

Stats Linear Regression Model

- Metrics Rsquared = 0.596
- Metrics R2_Score = 0.565

KNN Classification Model

- features = ['weight', 'gender', 'age']

  x = diabetes_data[features]
  y = np.where(diabetes_data.glyhb > 6.5, 1, 0)

  - Knn = KNeighborsClassifier(n_neighbors= 3)
  - Cross Validation = 5
  - [0.80952381 0.8 0.8 0.8 0.8]

KNN Classification Model

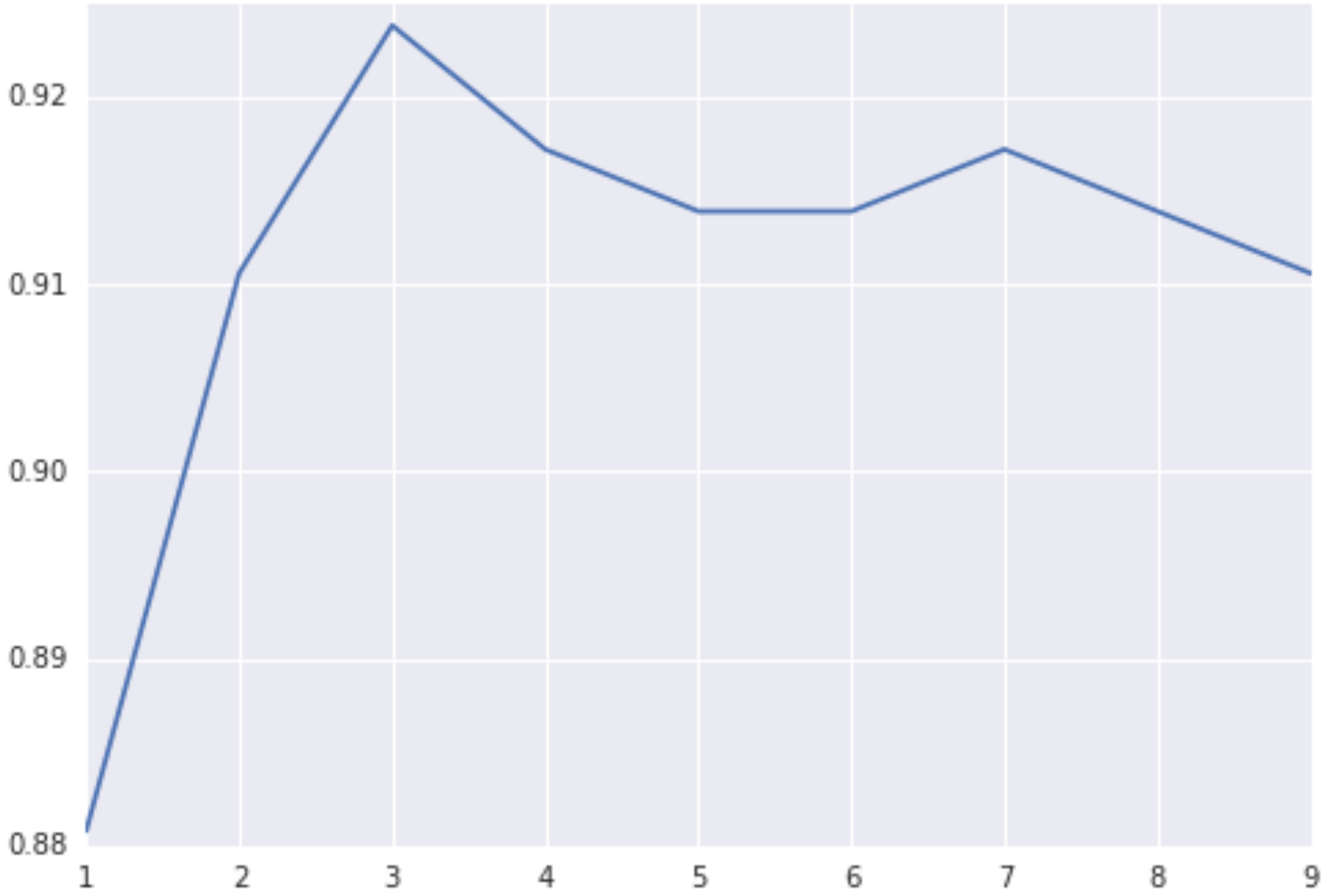
- features = ['weight', 'gender', 'age']

  x = diabetes_data[features]
  y = np.where(diabetes_data.glyhb > 6.5, 1, 0)
  - Knn = KNeighborsClassifier(n_neighbors= 3)
  - Cross Validation = 5
  - [0.80952381 0.8 0.8 0.8 0.8]

- features = ['stab.glu', 'weight', 'gender', 'age']

  x = diabetes_data[features]
  y = np.where(diabetes_data.glyhb > 6.5, 1, 0)
  - knn = KNeighborsClassifier(n_neighbors= 2)
  - Cross Validation = 5
  - [0.95238095 0.95 0.9 1. 1.]

KNN classification Model

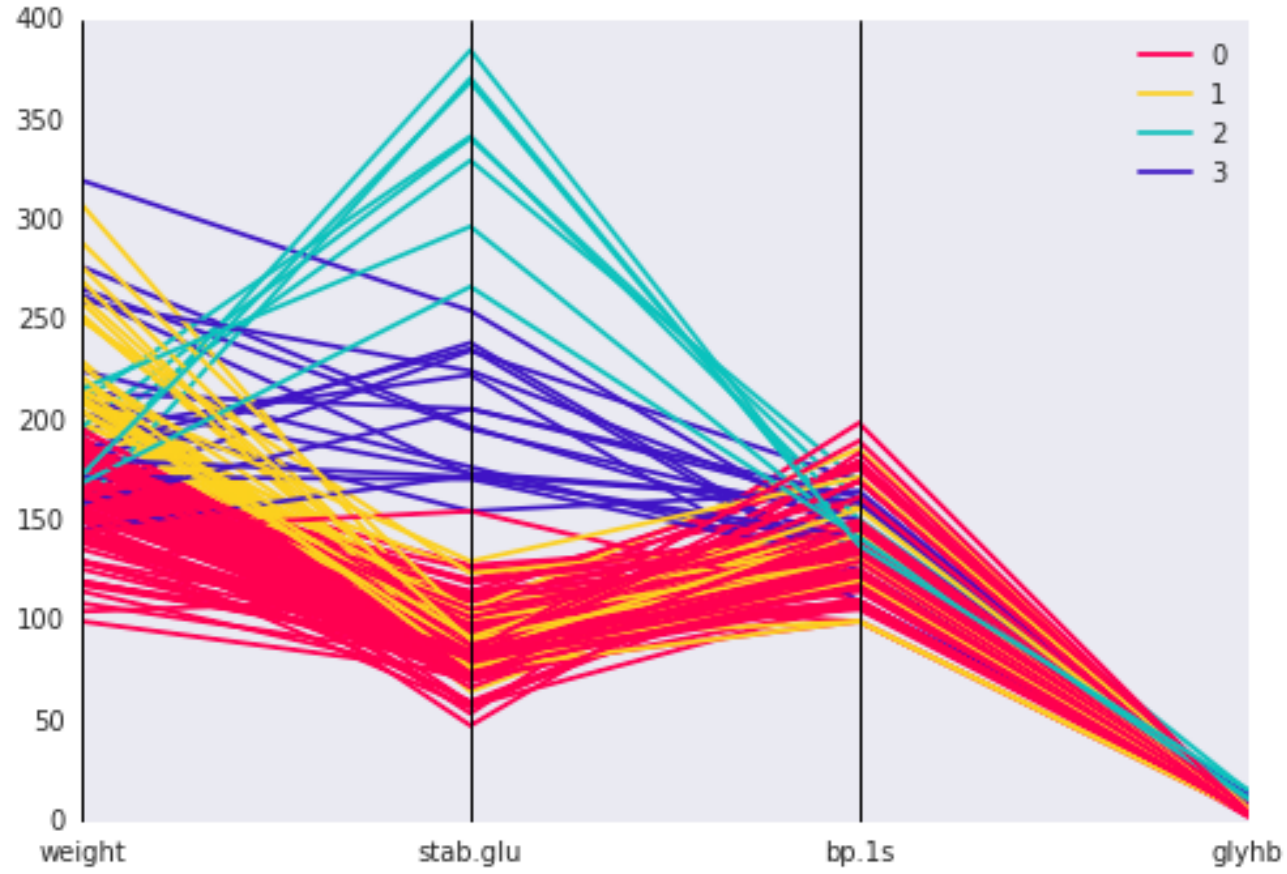


K-Means Clustering Modeling

columns = ['weight', 'stab.glu', 'bp.1s', 'glyhb']
Only on **Males**

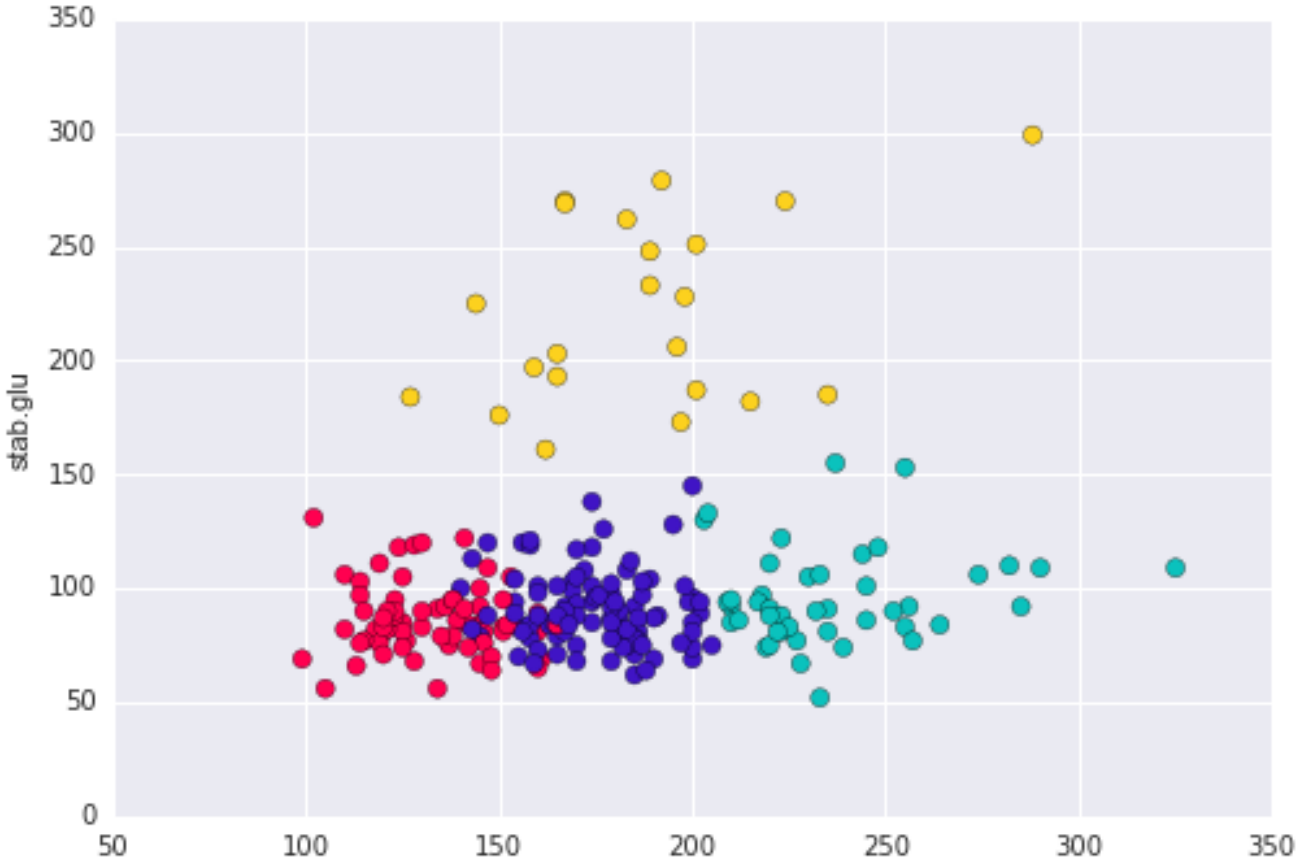
K-Means Clustering Modeling

columns = ['weight', 'stab.glu', 'bp.1s', 'glyhb']
Only on **Males**



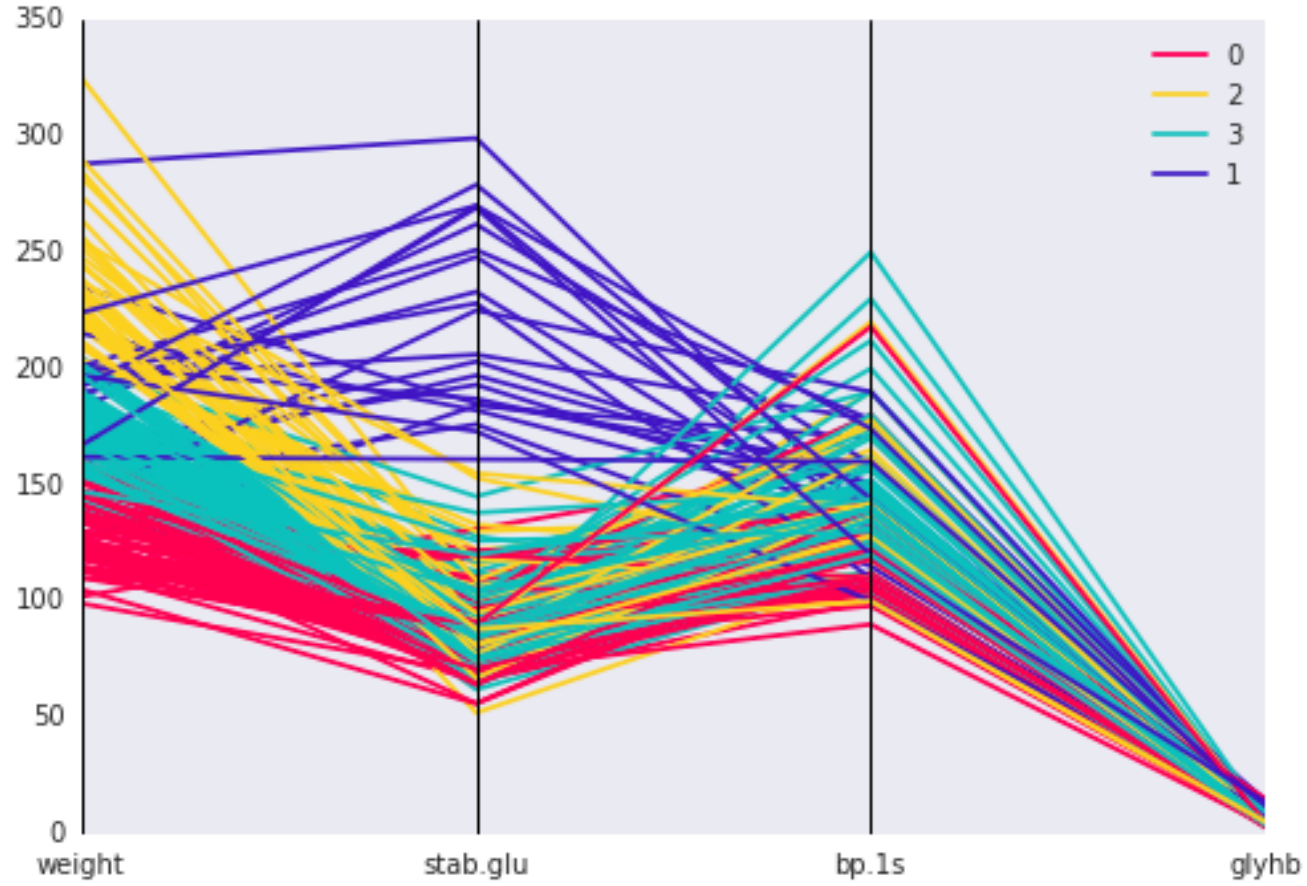
K-Means Clustering Modeling

columns = ['weight', 'stab.glu', 'bp.1s', 'glyhb']
Only on **Females**



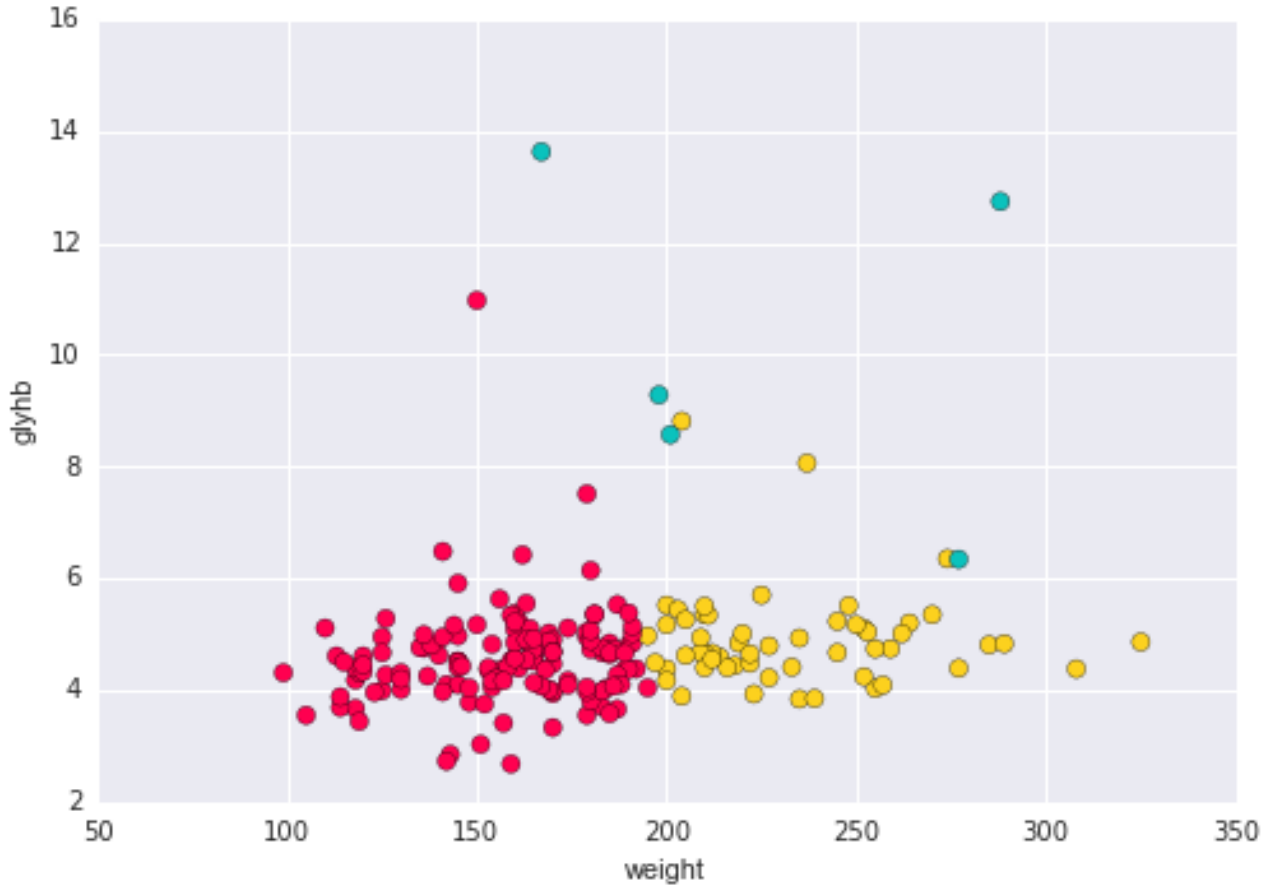
K-Means Clustering Modeling

columns = ['weight', 'stab.glu', 'bp.1s', 'glyhb']
Only on **Females**



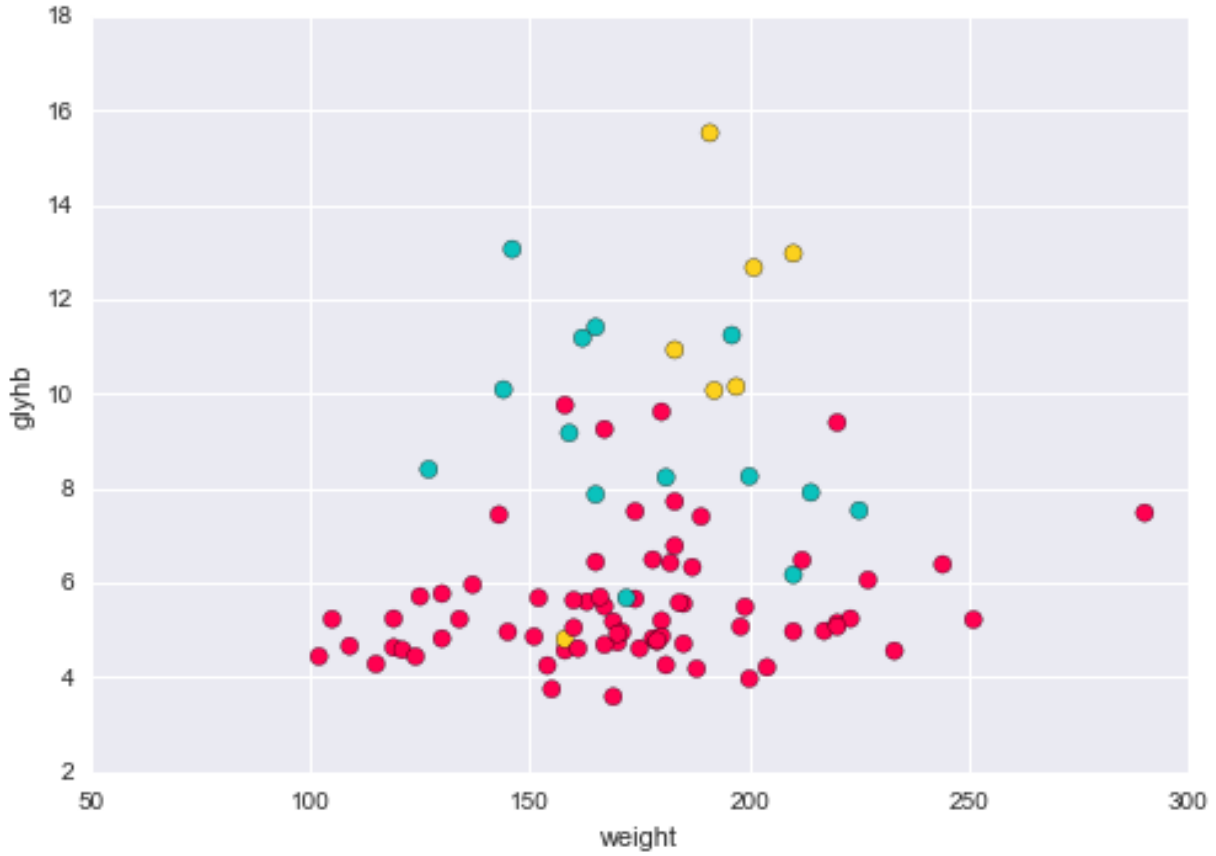
K-Means Clustering Modeling

Columns = ['weight', 'stab.glu', 'bp.1s', 'glyhb', 'waist', 'hip']
Only on **Age Groups = 20 – 44**



K-Means Clustering Modeling

Columns = ['weight', 'stab.glu', 'bp.1s', 'glyhb', 'waist', 'hip']
Only on **Above 60**



Decision Tree Classifier

- response = diabetes_data['result']
  del diabetes_data['result']
  del diabetes_data['glyhb']
- ctree = tree.DecisionTreeClassifier(random_state=1, max_depth=**3**)
- Accuracy = 0.9108
- Feature Importance: Next Page

Decision Tree Classifier

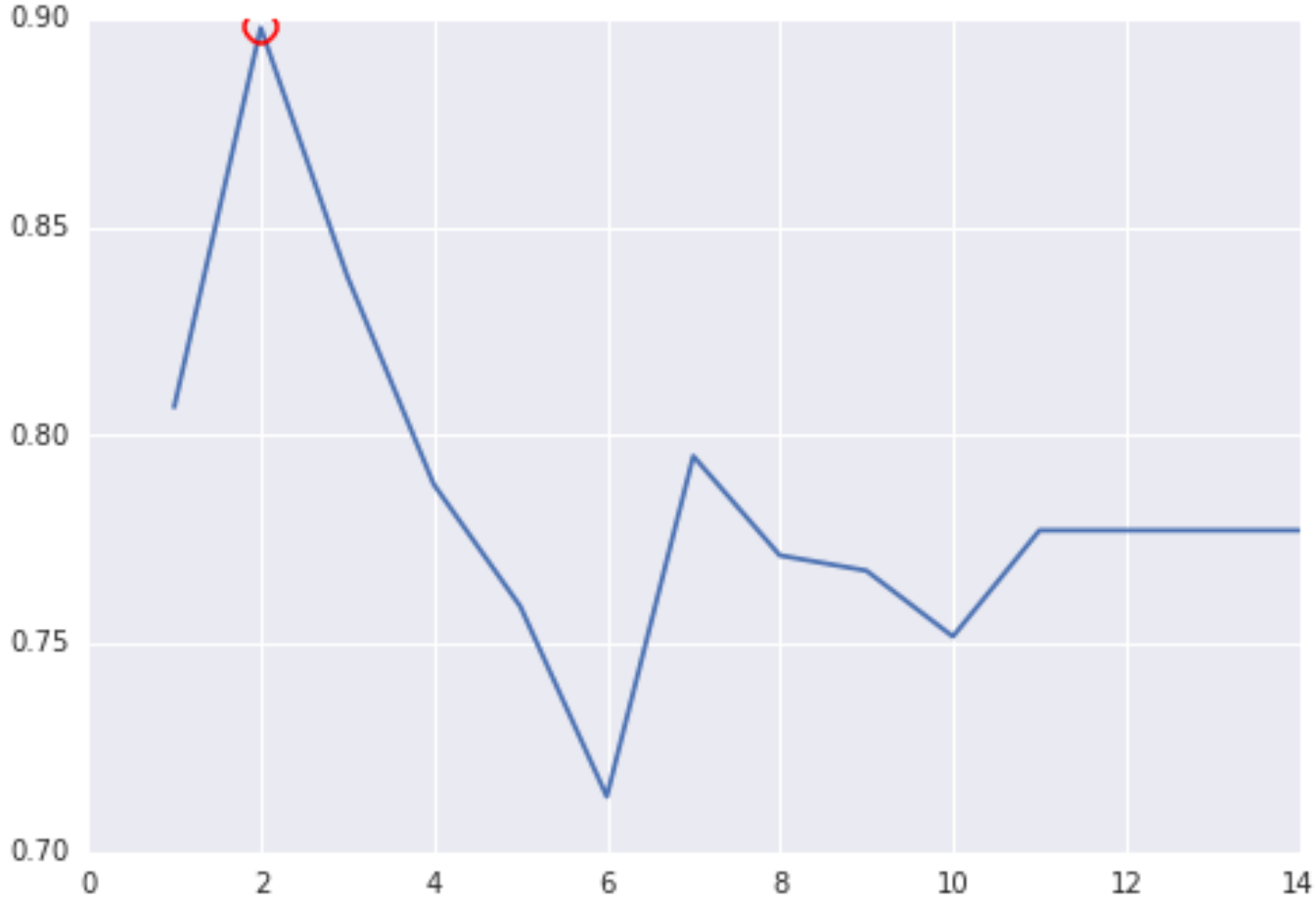
- Feature Importance:

1	stab.glu	0.84629
8	bp.1s	0.071607
9	bp.1d	0.053125
11	hip	0.024281
4	age	0.004697
0	chol	0
2	hdl	0
3	ratio	0
5	gender	0
6	height	0
7	weight	0
10	waist	0
12	time.ppn	0
13	age_groups	0

Accuracy = 0.91

This shows that **Weight** is not as important with a depth of 3. So I did a grid Search to find the most optimum depth.

Decision Tree Classifier



Depth = 2

Decision Tree Classifier

- Even though the roc_auc curve shows that 2 is the optimum maximum depth, it doesn't match up with my hypothesis or any of the results from Linear Regression.
- So I changed the depth = 4

	0	1
1	stab.glu	0.719165
8	bp.1s	0.102269
4	age	0.088261
9	bp.1d	0.044834
11	hip	0.02498
7	weight	0.020492
0	chol	0
2	hdl	0
3	ratio	0
5	gender	0
6	height	0
10	waist	0

Accuracy = 0.88

Support Vector Matrices

- features = ['stab.glu', 'weight', 'gender', 'age']
- Used Train Test Split
- Fit it it to my Training Data
- Scored it using the Test Data
  - cross_val_score(clf, X_test, y_test, cv=5, scoring='accuracy').mean()
  - Accuracy = 0.85

Logistic Regression

- features = ['stab.glu', 'weight', 'bp.1s']
  x = diabetes_data[features]
  y = diabetes_data['glyhb']
  y_category = np.where(y > 6.5, 1, 0)

- Fit the model to a X_train and y_train
  - ('stab.glu', 0.0328)
  - ('weight', -0.0060)
  - ('bp.1s', 0.0040)
  - ('bp.1d', -0.0281)
  - Confusion Matrix

    [86 0]
    [5 10]

  - Accuracy = 0.95 or 95%
  - Specificity = 1.0
  - Sensitivity = 0.66

Linear Regression Model

- features = ['stab.glu','weight', 'bp.1s']
  x = diabetes_data[features]
  y = diabetes_data['glyhb']
- Train Test Split / Fit the Linear Model
- Mean Absolute Error = 0.91
- Mean Squared Error = 1.42
  (Remember the scale of the Output it is between 3.5 – 15.0)

Linear Regression Model

Green – stab.glu
Blue – Weight
Red – bp.1s

Questions?

Thank You !!! Sinan

- I'd like to thank Sinan, for being awesome, interested in our problems, working with us over weekends.
- I'd like to thank all the TA's Ramesh, Liam and Patrick
- Also wanna thank Vanessa, for constantly checking on how were doing, the feedback forms and the Slack and just being really collaborative.

References

- Department of Biostatistics at the Vanderbilt University
- http://cs-people.bu.edu/dgs/courses/cs105/hall_of_fame/awm.html
- http://www.cdc.gov/diabetes/pubs/pdf/diabetesreportcard.pdf
- http://www4.stat.ncsu.edu/~boos/var.select/diabetes.html
- http://cs-people.bu.edu/dgs/courses/cs105/hall_of_fame/awm.html
- http://www.cdc.gov/diabetes/pubs/pdf/diabetesreportcard.pdf

GitHub – Karun Siddana

- https://github.com/ksiddana/SF_DAT_15_WORK